Université de Montréal

Méthodes et algorithmes pour l'amélioration de l'inférence de l'histoire évolutive des génomes

par

Finagnon Marc-Rolland Emmanuel Noutahi

Département de Biochimie et Médecine moléculaire
Faculté de Médecine

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Bio-informatique

26 juillet 2018

RÉSUMÉ

Les phylogénies de gènes offrent un cadre idéal pour l'étude comparative des génomes. Non seulement elles incorporent l'évolution des espèces par spéciation, mais permettent aussi de capturer l'expansion et la contraction des familles de gènes par gains et pertes de gènes. La détermination de l'ordre et de la nature de ces événements équivaut à inférer l'histoire évolutive des familles de gènes, et constitue un prérequis à plusieurs analyses en génomique comparative. En effet, elle est requise pour déterminer efficacement les relations d'orthologies entre gènes, importantes pour la prédiction des structures et fonctions de protéines et les analyses phylogénétiques, pour ne citer que ces applications.

Les méthodes d'inférence d'histoires évolutives de familles de gènes supposent que les phylogénies considérées sont dénuées d'erreurs. Ces phylogénies de gènes, souvent reconstruites à partir des séquences d'acides aminés ou de nucléotides, ne représentent cependant qu'une estimation du vrai arbre de gènes et sont sujettes à des erreurs provenant de sources variées, mais bien documentées. Pour garantir l'exactitude des histoires inférées, il faut donc s'assurer de l'absence d'erreurs au sein des arbres de gènes. Dans cette thèse, nous étudions cette problématique sous deux aspects.

Le premier volet de cette thèse concerne l'identification des déviations du code génétique, l'une des causes d'erreurs d'annotations se propageant ensuite dans les phylogénies. Nous développons à cet effet, une méthodologie pour l'inférence de déviations du code génétique standard par l'analyse des séquences codantes et des ARNt. Cette méthodologie est centrée autour d'un algorithme de prédiction de réaffectations de codons, appelé CoreTracker. Nous montrons tout d'abord l'efficacité de notre méthode, puis l'utilisons pour démontrer l'évolution du code génétique dans les génomes mitochondriaux des algues vertes.

Le second volet de la thèse concerne le développement de méthodes efficaces pour la correction et la construction d'arbres phylogénétiques de gènes. Nous présentons deux méthodes exploitant l'information sur l'évolution des espèces. La première, ProfileNJ , est déterministe et très rapide. Elle corrige les arbres de gènes en ciblant exclusivement les sous-arbres présentant un support statistique faible. Son application sur les familles

de gènes d'Ensembl Compara montre une amélioration nette de la qualité des arbres, par comparaison à ceux proposés par la base de données. La seconde, GATC, utilise un algorithme génétique et traite le problème comme celui de l'optimisation multi-objectif de la topologie des arbres de gènes, étant données des contraintes relatives à l'évolution des familles de gènes par mutation de séquences et par gain/perte de gènes. Nous montrons qu'une telle approche est non seulement efficace, mais appropriée pour la construction d'ensemble d'arbres de référence.

Mots-clés : arbres de gènes ; histoire évolutive ; phylogénie ; réconciliation ; orthologie ; code génétique ; réassignation de codon ; évolution

ABSTRACT

Gene trees offer a proper framework for comparative genomics. Not only do they provide information about species evolution through speciation events, but they also capture gene family expansion and contraction by gene gains and losses. They are thus used to infer the evolutionary history of gene families and accurately predict the orthologous relationship between genes, on which several biological analyses rely.

Methods for inferring gene family evolution explicitly assume that gene trees are known without errors. However, standard phylogenetic methods for tree construction based on sequence data are well documented as error-prone. Gene trees constructed using these methods will usually introduce biases during the inference of gene family histories. In this thesis, we present new methods aiming to improve the quality of phylogenetic gene trees and thereby the accuracy of underlying evolutionary histories of their corresponding gene families.

We start by providing a framework to study genetic code deviations, one possible reason of annotation errors that could then spread to the phylogeny reconstruction. Our framework is based on analysing coding sequences and tRNAs to predict codon reassignments. We first show its efficiency, then apply it to green plant mitochondrial genomes.

The second part of this thesis focuses on the development of efficient species tree aware methods for gene tree construction. We present ProfileNJ , a fast and deterministic correction method that targets weakly supported branches of a gene tree. When applied to the gene families of the Ensembl Compara database, ProfileNJ produces an arguably better set of gene trees compared to the ones available in Ensembl Compara. We later use a different strategy, based on a genetic algorithm, allowing both construction and correction of gene trees. This second method called GATC, treats the problem as a multi-objective optimisation problem in which we are looking for the set of gene trees optimal for both sequence data and information of gene family evolution through gene gain and loss. We show that this approach yields accurate trees and is suitable for the construction of reference datasets to benchmark other methods.

Keywords: gene tree; evolutionary history; phylogeny; reconciliation; orthology; genetic code; codon reassignment; evolution

Table des matières

Liste des tableaux

Table des figures

LISTE DES SIGLES ET DES ABRÉVIATIONS

AA	Acide Aminé
aaRS	Aminoacyl-ARNt synthétase
ADN	Acide Désoxyribonucléique
ADNmt	ADN mitochondrial
ARN	Acide Ribonucléique
ARNm	ARN messager
ARNt	ARN de transfert
AU	Approximately Unbiased
BLAST	Basic Local Alignment Search Tool
DL	Duplication Loss
DTL	Duplication Transfer Loss
GATC	Genetic Algorithm for gene Tree Construction
GTR	General Time Reversible
ILS	Incomplete Lineage Sorting
LCA	Last Common Ancestor
MCMC	Markov Chain Monte Carlo
ML	Maximum Likelihood
MPR	Most Parsimonious Reconciliation
NAD	Non Apparent Duplication
NJ	Neighbor Joining
NNI	Nearest Neighbor Interchange
RF	Robinson-Foulds
SPR	Subtree Prune and Regraft
TBR	Tree bisection and reconnection
UPGMA	Unweighted Pair Group Method with Arithmetic Mean

DÉDICACES

À la science, en espérant y avoir contribué !

REMERCIEMENTS

Cette thèse n'aurait sans doute pas été menée à terme sans l'encouragement et l'implication directe ou indirecte de nombreuses personnes, que je tiens à remercier.

J'aimerais tout d'abord remercier grandement ma directrice de recherche, Nadia El-Mabrouk, pour sa disponibilité permanente, ses conseils, ses brillantes intuitions, son soutien moral et financier ainsi que son aide précieuse dans la rédaction des articles et la relecture de ma thèse. Je me rappelle encore être arrivé au LBIT, après une offre de stage à la fin de mon baccalauréat. Quatre ans plus tard je dépose une thèse. Ce fut un réel plaisir pour moi d'avoir travaillé sous ta supervision, Nadia.

Je voudrais aussi remercier les professeurs et les superviseurs qui m'ont encadré pendant tout mon cursus universitaire. Merci à Gertraud Burger et Sandrine Moreira avec qui j'ai effectué mon premier stage, qui a d'ailleurs abouti à mon premier article scientifique. Merci à Daniel Zenklusen et à son équipe, qui m'ont initié au travail en équipe. Mon stage avec eux m'a permis d'expérimenter directement l'application pratique des méthodes algorithmiques en bioinformatique à la recherche en laboratoire et de réaliser comment les deux domaines s'adaptent l'un à l'autre. Enfin, merci à Franz Lang qui a été le collaborateur et superviseur principal pour le travail réalisé sur les altérations du code génétique dans cette thèse. Un grand merci à lui pour ces conseils précieux, son encadrement et pour m'avoir initié à la rigueur et au scepticisme scientifique.

Merci également au personnel de soutien des départements de biochimie et d'informatique que j'ai côtoyé pendant ces dernières années. Merci spécialement à Elaine Meunier pour la patience dont elle fait preuve.

Mes vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à cette thèse en acceptant de l'examiner.

Un grand merci aussi à tous les membres du LBIT avec qui j'ai partagé ces dernières années : Manuel, Robin, Miguel, Billel, Marie et Anna. Je ne pardonne quand même pas Robin et Marie pour la session de solitude en automne 2017.

Merci au serveurs de calcul canada, qui ne m'ont jamais laissé tomber, même quand les dates limites s'approchaient dangereusement.

Je remercie enfin celles et ceux qui me sont chers et dont les encouragements m'ont accompagnés tout au long de ces années. Je tiens à exprimer ma reconnaissance envers ma famille qui m'a toujours supporté et soutenu pendant tout ce parcours. Merci aussi à Steven pour le travail de relecture, à Pascal, José-Marie, Jawad et toute la clique "El Papineau" et affiliée (oui, ce nom restera, n'en déplaise à certains). Merci à Prudencio et à Mazid, entre doctorants on se comprend. Ne comptez pas sur moi toutefois pour relire votre thèse, mon horoscope prédit que je serai occupé.

Et pour terminer, je remercie les différents organismes qui m'ont supporté financièrement durant mes études doctorales : la Faculté des Études Supérieures et Postdoctorales de l'Université de Montréal (FESP) et le Fonds de recherche du Québec - Nature et technologies (FRQNT).

Chapitre 1

Introduction

La phylogénie est l'étude des relations évolutives qui lient les organismes vivants, afin de mieux comprendre les mécanismes de l'évolution. Jusqu'au milieu de années 1950, les études phylogénétiques se basaient principalement sur la comparaison de caractéristiques morphologiques et physiologiques [1]. Toutefois, la découverte de l'ADN comme support de l'hérédité et les nombreuses avancées en biologie moléculaire qui s'en sont suivies ont bouleversé le domaine. En particulier, la découverte de la conservation de séquences protéiques des *familles de gènes* à travers différents domaines du vivant, grâce aux travaux pionniers de Zuckerkandl, Pauling et Dayhoff [2–5], a permis l'essor d'un nouveau domaine : la phylogénie moléculaire.

L'hypothèse fondamentale sous-jacente est que l'histoire évolutive des espèces peut être inférée à partir de l'information génétique contenue dans leurs génomes. Pour la phylogénie moléculaire, le focus porte donc désormais sur la séquence des macromolécules biologiques (ADN, ARN, protéines), et moins sur la comparaison des traits morphologiques résultant de leurs expressions, permettant de ce fait une amélioration nette de la précision des inférences. En effet, il est parfois difficile de convertir, de façon objective et sans ambiguïtés, les caractères morphologiques en format numérique utilisable au sein des modèles mathématiques d'évolution. Par ailleurs, certaines similitudes physiologiques ne sont pas héritées d'un ancêtre commun, mais plutôt issues d'une convergence évolutive. Les classifications principalement basées sur ces caractères morphologiques peuvent donc produire des phylogénies erronées. Un cas bien connu de convergence évolutive est la présence d'ailes chez la chauve-souris, pourtant un mammifère et non un oiseau. Un autre exemple tout aussi remarquable est la forte similitude physiologique entre le thylacine, un marsupial, et les canidés (d'où son autre nom : loup de Tasmanie).

L'avantage principal de la phylogénie moléculaire reste néanmoins sa capacité à élucider les relations entre organismes, à une résolution beaucoup plus fine que la phylogénie traditionnelle. À titre d'exemple, certaines algues vertes constituent des complexes d'espèces cryptiques, morphologiquement indiscernables, mais cachant toutefois une grande diversité génétique [6].

Bien que la phylogénie moléculaire tire principalement avantage de l'évolution des séquences de gènes au sein des génomes par mutations ponctuelles (substitutions, insertions et délétions de bases), les génomes évoluent également par le biais d'autres processus complexes (duplications, pertes, gains, transferts horizontaux, réarrangements, tri incomplet des lignées, etc.) qui affectent non seulement leurs contenus en gènes, mais également la distribution, la régulation et l'arrangement de leurs gènes. Ces divers phénomènes jouent un rôle essentiel dans la plasticité de l'évolution et l'adaptation des organismes à leurs environnements. En particulier, la duplication de gènes permet la création de nouvelles copies de gènes et est reconnue depuis la fin des années 1960 comme l'un des moteurs principaux d'acquisition de nouvelles fonctions [7]. De façon analogue, les transferts horizontaux de gènes (HGT pour *Horizontal Gene Transfer*), très répandus chez les procaryotes, constituent l'un des mécanismes centraux d'évolution chez ces derniers, notamment en ce qui concerne leur rôle dans l'augmentation de la résistance des bactéries aux antibiotiques. En raison de ces processus, même si les séquences de gènes sont utilisées pour inférer l'évolution des espèces, les arbres de gènes sont fondamentalement distincts de ceux des espèces.

Alors que les arbres d'espèces modélisent comment se forment et divergent de nouvelles espèces, les arbres de gènes capturent plutôt l'évolution des gènes à l'intérieur des branches de l'arbre d'espèces. Ainsi, les familles de gènes peuvent présenter des histoires évolutives différentes d'une famille à une autre, et potentiellement incompatibles avec l'histoire des espèces (voir Figure 1.1).

Les différences observées entre arbre de gènes et arbre d'espèces renferment l'information nécessaire pour inférer l'histoire évolutive de la famille de gènes correspondante. Dans le plus simple des cas où les gènes évoluent uniquement par spéciation et qu'il n'y a pas de pertes, il existe une bijection entre les nœuds de l'arbre des gènes et ceux de l'arbre des espèces ; la topologie des deux arbres devrait donc être identique. En présence d'autres types d'évènements évolutifs (duplication, perte, HGT), l'histoire évolutive est inférée en utilisant un procédé appelé *réconciliation*, qui explique les divergences entre les deux arbres par les différents évènements évolutifs affectant les gènes [8] (voir section 3.3 au chapitre 3).

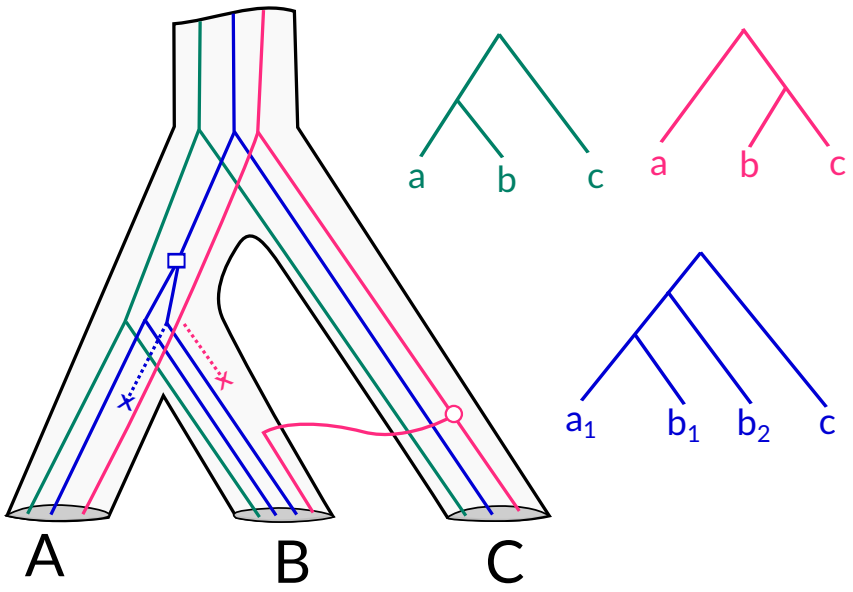


Figure 1.1. Évolution de trois familles différentes de gènes représentées par leurs arbres de gènes (en trait bleu, vert et rose) à l'intérieur d'un arbre d'espèces (tube). L'évolution de chacune de ces familles est marquée par plusieurs types d'évènements. La famille verte ne subit aucun évènement macro-évolutif à part les spéciations, aboutissant en un arbre de gène avec une topologie identique à celle de l'arbre des espèces. La famille bleue subit une duplication (rectangle) après la spéciation séparant l'espèce C de l'ancêtre commun des espèces A et B, puis une perte (trait pointillé finissant par une croix) de l'une des copies dans le génome A . Enfin, l'évolution de la famille rose est marquée par la perte d'un gène dans B, puis un transfert horizontal d'un gène (cercle) de C à B. Ces différents évènements ont donné lieu à des topologies d'arbres de gènes bien distinctes entre elles, et parfois différentes de celle de l'arbre des espèces.

L'inférence de l'histoire évolutive des familles de gènes, ainsi que des processus biologiques ayant marqué cette évolution, est une étape préliminaire et cruciale à plusieurs analyses biologiques. Elle permet en effet d'établir les relations liant les gènes d'une même famille, en distinguant les gènes orthologues qui ont divergé par un évènement de spéciation, des paralogues et des xénologues qui divergent respectivement par duplication et transfert horizontal. Cette distinction est importante en génomique comparative, surtout pour l'annotation fonctionnelle des gènes, en raison de la *conjecture des orthologues* qui stipule que les gènes orthologues ont tendance à conserver la même fonction en comparaison aux paralogues [9, 10]. Les autres applications de l'inférence de l'histoire évolutive des familles de gènes incluent la reconstruction de gènes et génomes ancestraux, l'inférence exacte de phylogénies, les études de coévolution d'espèces, etc.

Afin de garantir la précision des résultats obtenus pour ces différentes analyses, il faudrait donc s'assurer que les histoires évolutives inférées par la réconciliation sont correctes. Puisque le principe de la réconciliation repose entièrement sur les différences topologiques entre arbres d'espèces et arbre de gènes, son efficacité à inférer les vraies histoires évolutives dépend donc fortement de la précision de ces deux phylogénies [11]. Cette dernière, quant à elle, est influencée par plusieurs facteurs incluant la qualité des données (erreurs de séquençage et d'annotation, information manquante, etc.), et les performances des algorithmes d'alignement et d'inférence phylogénétique.

Dans le premier cas, les innovations technologiques de ses dernières années dans le domaine du séquençage de génome, ainsi que les méthodes modernes de comparaison de séquences, permettent de réduire considérablement les erreurs de séquençage et de remédier

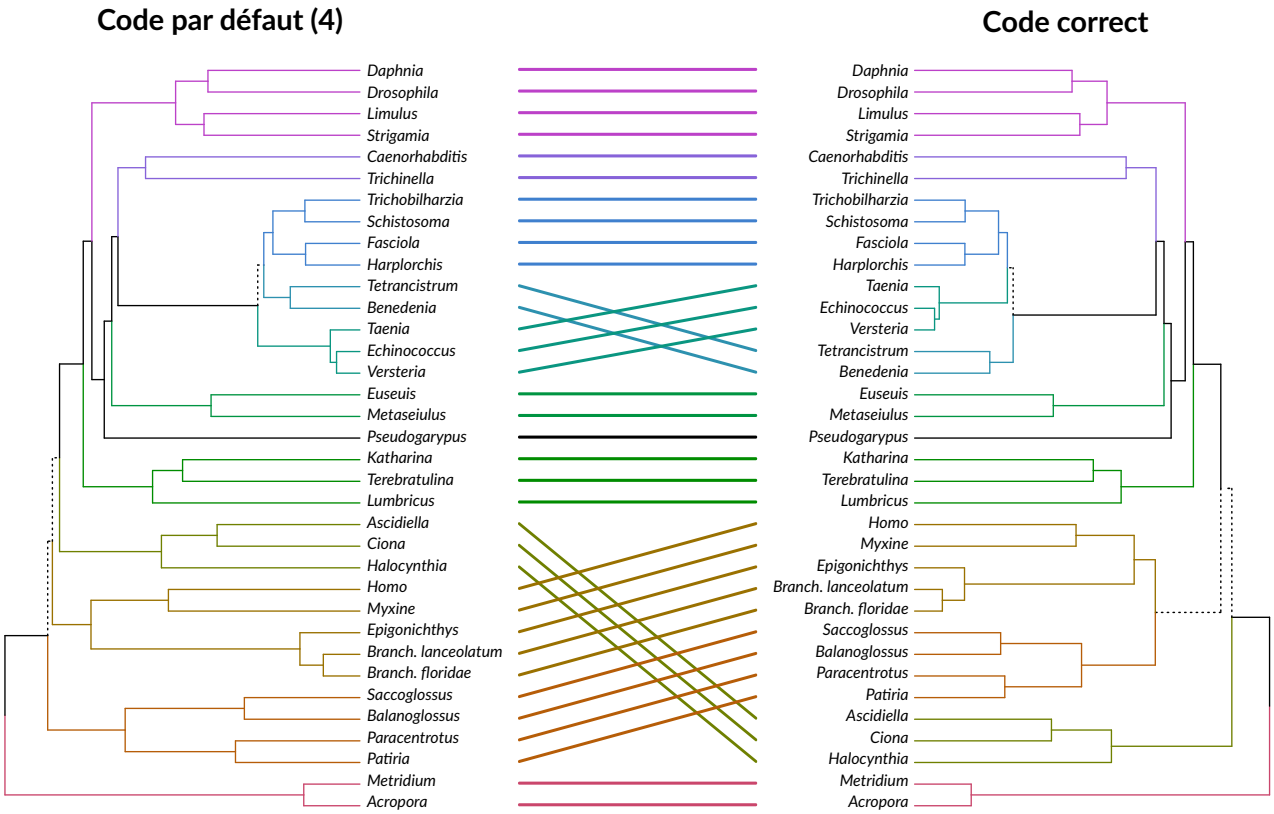


Figure 1.2. Différence topologique entre les phylogénies de métazoaires inférés à partir des 13 protéines mitochondriales (Cob, Cox1-2-3, Atp6-9 et Nad1-2-3-4-4L-5-6), lorsque le code mitochondrial par défaut (table 4) est utilisé vs lorsque les réassignations de codons spécifiques à chaque génome sont considérées. On note une différence significative entre les deux topologies (les sous-arbres divergents sont indiqués en pointillés).

aussi au manque de données. Toutefois, certaines erreurs peuvent persister à cause de la grande quantité de données obligeant à faire confiance aux annotations automatiques. Plusieurs études estiment ainsi que les bases de données biologiques regorgeraient d'erreurs d'annotations [12–15]. Les sources d'erreurs répertoriées par ces études ne sont toutefois pas exhaustives. En effet, ce n'est pas uniquement les gènes qui évoluent au sein des génomes ; le *code génétique* qui détermine la correspondance entre codons (unité structurelle des gènes) et acides aminés (composante élémentaire des protéines) est lui aussi capable d'évoluer. Cette évolution se traduit par la modification de l'identité de certains codons, changeant leur décodage vers d'autres acides aminés. On parle alors de *réassignation* de codons.

L'interprétation que l'on fait de la séquence d'un gène peut ainsi varier considérablement en fonction du code génétique utilisé par l'organisme dont il provient. Malheureusement, les méthodes automatiques d'annotation supposent bien souvent que le décodage des codons est standard et universel. Non détectés, les changements au code génétique peuvent avoir plusieurs conséquences désastreuses sur la précision des analyses biologiques subséquentes, puisqu'elles affectent l'entièreté des séquences protéiques des génomes et peuvent aussi introduire des erreurs dans la prédiction des cadres de lecture ouverts en modifiant, par exemple, la position du codon-stop. Dans le cas des analyses phylogénétiques, les altérations non détectées du code génétique auront tendance à gonfler artificiellement les taux de substitution sur les branches des génomes concernés. Comme conséquence potentielle, ces phylogénies subiront des artefacts de reconstruction comme l'attraction des longues branches. À la Figure 1.2, nous illustrons l'impact des réassignations de codons sur la topologie de la phylogénie des métazoaires, dont les déviations du code génétique mitochondrial sont bien étudiées. Cette figure présente la topologie de l'arbre des métazoaires, inférée à partir des séquences de protéines mitochondriales, lorsque le code par défaut et le vrai code pour chaque génome sont utilisés. On peut y voir une différence topologique importante entre les deux arbres.

Même en supposant l'absence complète d'erreurs de séquençage et d'annotation au sein des séquences génomiques, il est pratiquement impossible de garantir, avec certitude, que les arbres phylogénétiques reconstruits avec les méthodes d'inférence actuelles ne comportent pas d'erreurs. En effet, les erreurs affectant la topologie des arbres de gènes concernent non seulement les artefacts classiques de reconstruction phylogénétique inhérents aux méthodes d'inférence utilisées [16, 17], mais aussi l'insuffisance de l'information des séquences pour la résolution correcte et sans ambiguïtés des relations entre gènes. Comme mentionné précédemment, l'évolution des familles de gènes est souvent marquée par la présence de processus complexes (duplications, pertes, HGT), non nécessairement observables au niveau des séquences génomiques. Malheureusement, la plupart des méthodes d'inférence phylogénétique

reposent presque exclusivement sur les alignements de séquences et infèreront des topologies erronées lorsque ces derniers sont non-informatifs. Par exemple, une mauvaise différentiation des séquences résultera en une topologie douteuse en raison de la présence de nœuds incorrectement ou partiellement résolus. C'est ce qui arrive parfois avec les méthodes bayésiennes dont le fonctionnement consiste à échantillonner l'espace de solutions, puis reconstruire un arbre consensus à partir de ces solutions. Des contradictions topologiques entre solutions peuvent ainsi apparaître, amenant à la présence de nœuds non binaires appelés *polytomies* dans l'arbre consensus. Alternativement, les branches ayant un faible support statistique, indiquant donc la présence d'incertitudes, peuvent également être contractées pour produire ces polytomies. Bien que d'un point de vue évolutif, on pourrait admettre l'existence de *vraies polytomies* causées par des duplications ou plusieurs spéciations synchrones donnant lieu à plus de deux descendants [18, 19], de tels évènements sont naturellement rares [20–22]. Il est donc traditionnellement admis que le manque de résolution au sein des arbres de gènes résulte d'une information phylogénétique insuffisante.

Le développement de méthodes minimisant ces erreurs au sein des arbres de gènes dans le but d'améliorer l'exactitude des inférences d'histoires évolutives de familles de gènes apparaît donc indispensable. Dans cette thèse, nous nous intéressons à ce problème.

Le travail que nous présentons s'inscrit donc dans le cadre global de l'inférence de l'histoire évolutive de familles de gènes et de génomes ; et nous étudions la question sous deux aspects principaux : la prédiction des altérations du code génétique obstruant le signal phylogénétique, et l'inférence d'histoires évolutives tenant compte des incertitudes au sein des arbres de gènes. La structure globale de la thèse est décrite ci-après. Les chapitres 2 et 3 font une mise en contexte au sujet, alors que les chapitres 4 à 7 présentent nos contributions originales sous forme de quatre articles, regroupés selon les deux thèmes principaux étudiés.

Chapitres 2 et 3 : Mise en contexte

Le chapitre 2 introduit succinctement quelques notions et concepts biologiques nécessaires à la compréhension de cette thèse, notamment en ce qui concerne l'évolution des génomes. Dans ce chapitre, nous présentons aussi le code génétique et les hypothèses concernant son origine et son évolution. Nous répertorions également les déviations connues du code et les mécanismes proposés pour les expliquer. Enfin, nous terminons le chapitre par une discussion brève sur les concepts algorithmiques pour la prédiction d'altérations du code génétique.

Dans le chapitre 3, nous passons en revue les différentes notions en rapport avec l'évolution des familles de gènes, sur lesquelles se basent les résultats présentés dans les chapitres 6

et 7. Pour ce faire, nous présentons tout d'abord les notations et définitions formelles associées au concept d'arbre phylogénétique. Nous décrivons par la suite les modèles d'évolution des familles de gènes par duplications, pertes, spéciations et transferts horizontaux, puis nous survolons les méthodes existantes d'inférence et de correction d'arbres de gènes.

Chapitres 4 et 5 : Évolution du code génétique

L'évolution du code génétique est un fait bien établi, comme en témoigne le nombre grandissant de tables génétiques dans la base de données NCBI [23]. Avec l'augmentation exponentielle du nombre de séquences génomiques dans les bases de données publiques, on devrait s'attendre à ce que de nouvelles déviations au code standard soient découvertes. Par ailleurs, il est fortement probable que parmi les génomes déjà annotés, en particulier ceux d'organites, des réaffectations de codons soient restées non-détectées [24].

En théorie, on peut identifier les changements au sein du code génétique en comparant les séquences nucléotidiques des gènes aux vraies séquences des protéines qu'ils encodent. Cependant, l'existence de plusieurs modifications post-transcriptionnelles changeant la séquence des protéines (par exemple l'édition d'ARN) rend le problème plus complexe. Il faudrait alors comparer bien plus que quelques gènes pour s'assurer que le changement d'identité d'un codon affecte le génome en entier, afin de faire la différence entre les déviations du code génétique et les modifications post-transcriptionnelles. Le séquençage de protéomes peut toutefois se révéler pénible en raison du coût et de la difficulté à purifier correctement les protéines dans certains taxa. Des méthodes *in silico* pour la prédiction automatique de réaffectations de codons s'avèrent donc indispensables.

Dans le chapitre 4 présentant l'article intitulé *"CoreTracker : accurate codon reassignment prediction, applied to mitochondrial genomes"* publié dans *Bioinformatics*, nous présentons CoreTracker, une nouvelle méthode précise et robuste pour la prédiction de réaffectation de codons à partir d'alignements de séquences protéiques d'un groupe d'espèces d'intérêt. En appliquant cette méthode aux génomes mitochondriaux de levures et de métazoaires, nous montrons qu'elle surpasse les quelques alternatives existantes.

Dans le chapitre 5 ayant pour titre : *"Rapid genetic code evolution in green algal mitochondrial genomes"*, nous effectuons une analyse en profondeur de l'évolution du code génétique dans les génomes mitochondriaux d'algues vertes. Cet article expose, pour la première fois, des preuves concrètes de réaffectations de codons, entre acides aminés, dans ces génomes. Nous montrons que le code génétique a évolué de façon indépendante chez plusieurs chlorophytes mais pas chez les plantes terrestres, et présentons les scénarios évolutifs les plus

probables à l'origine de ces changements. Il ressort également de ce travail que les mécanismes qui sous-tendent l'évolution du code génétique ne sont pas mutuellement exclusifs. Nos résultats suggèrent, en effet, que les réaffectations de codons sont souvent causées par une combinaison de facteurs et sont facilitées par la présence de certaines conditions, telles qu'une réduction importante de la taille du protéome.

Chapitres 6 et 7 : Correction et construction d'arbres de gènes

Les incertitudes au sein des arbres phylogénétiques de familles de gènes causées par l'insuffisance de l'information des séquences, peuvent introduire des biais importants dans l'inférence et l'interprétation de l'évolution de ces familles. Ainsi, le développement de nouvelles méthodes de construction et de correction d'arbres de gènes incorporant des informations additionnelles (organisation génomique, ordre des gènes, phylogénie des espèces, etc.) s'avère donc nécessaire. Plusieurs études récentes ont montré qu'il est possible d'obtenir de meilleurs arbres de gènes donnant lieu à des histoires évolutives beaucoup plus crédibles, en incorporant l'information sur l'évolution des espèces [25–28]. Les chapitres 6 et 7 corroborent ces résultats tout en explorant des façons différentes d'intégrer cette information à travers la réconciliation entre arbres de gènes et arbres d'espèces.

Le chapitre 6 présente l'article "*Efficient gene tree correction guided by genome evolution*" qui décrit ProfileNJ , une méthode déterministe, rapide et efficace pour corriger *a posteriori* les erreurs au sein des arbres de gènes. La méthode cible les nœuds non résolus ou faiblement supportés et les corrige par la résolution des polytomies induites de façon à minimiser les incongruences entre les histoires évolutives des gènes et celles des espèces. Cette résolution se fait sous un modèle qui considère les spéciations et la duplication/perte de gènes en utilisant PolytomySolver, un algorithme linéaire développé par notre groupe [29]. ProfileNJ est intégré dans un pipeline de correction d'arbres de gènes appelé RefineTree, qui comprend d'autres outils de correction utilisant d'autres sources d'informations telles que la synténie. Nous montrons que l'application de ProfileNJ à toutes les 20519 familles de gènes de la base de données Ensembl Compara permet de générer un nouvel ensemble d'arbres objectivement meilleur que celui d'Ensembl.

Dans le chapitre 7 présentant l'article *"GATC : A Genetic Algorithm for gene Tree Construction under the Duplication Transfer Loss model of evolution"*, nous investiguons une approche alternative basée sur un algorithme génétique pour la construction ou la correction d'arbres de gènes. Cette nouvelle méthode étudie le problème d'inférence d'arbres de gènes en tant qu'un problème d'optimisation de la topologie des arbres de gènes sous deux critères simultanés : l'information des séquences et celle sur la topologie de l'arbre des espèces.

GATC permet également de remédier à certaines limitations de ProfileNJ , notamment en ce qui concerne le modèle d'évolution qui ne tenait précédemment pas compte des transferts horizontaux, et le fait que l'optimisation ne ciblait que certains nœuds de l'arbre de gènes, ne garantissant donc pas une amélioration globale.

Finalement, dans la conclusion de cette thèse (chapitre 8), nous discutons des forces et des limites des méthodes décrites et présentons quelques pistes pour des travaux futurs.

Chapitre 2

Information génétique

La découverte des mécanismes d'évolution des organismes passe forcément par le décodage du contenu de leur matériel génétique ainsi que la compréhension des mécanismes régissant la transmission et l'expression des gènes.

Dans ce chapitre, nous rappelons brièvement les notions de bases de l'organisation du matériel génétique, son expression, les modifications qu'il peut subir et leurs conséquences.

Dans la section 2.1, nous discutons des concepts biologiques nécessaires à la compréhension de cette thèse. Il s'agira, tout d'abord, de définir les notions de gènes, génomes et espèces, puis de présenter brièvement les modes d'évolution des gènes et des génomes.

Enfin, la section 2.2 concerne l'expression de l'information génétique par traduction des gènes. Nous insisterons sur l'existence de variabilité inter génomique du code génétique pour souligner le fait qu'il n'est pas immuable, et tout comme les génomes, est aussi capable d'évoluer.

2.1. Espèce, génome et expression de l'information génétique

Dans cette thèse, nous suivons la définition d'espèces proposée par Ernst Mayr et communément admise par la communauté scientifique [30–32]. Une espèce représente une population d'organismes dont les individus sont capables de se reproduire entre eux, produisant des descendants du même groupe qui sont viables et féconds. Le génome d'un organisme représente l'ensemble de son matériel génétique et est transmis à sa descendance après reproduction. Il est généralement encodé dans la molécule d'ADN, une longue chaîne souvent formée de deux brins complémentaires contenant une suite de quatre désoxyribonucléotides (A : adénine, T :thymine, C :cytosine et G : guanine) où A et T sont complémentaires et peuvent s'apparier, de même que G et C. Notons que d'autres types d'appariements non standard de bases existent.

Les génomes sont organisés en entités physiques appelées chromosomes correspondant chacune à une seule molécule d'ADN. Nous utiliserons, par convenance, le terme "génome" dans cette thèse pour désigner exclusivement l'ensemble des suites de séquences nucléotidiques formant l'ADN, ignorant de ce fait les molécules biochimiques qui lient naturellement l'ADN dans le vivant. Chez les eucaryotes, nous distinguerons aussi le génome mitochondrial, confiné à l'intérieur des mitochondries, du génome nucléaire, localisé dans le noyau des cellules.

À l'intérieur des génomes, on retrouve des régions continues de nucléotides, sur un seul brin, appelées gènes. Les gènes constituent les unités physiques, moléculaires et fonctionnelles de l'hérédité, et sont donc responsables des traits phénotypiques observés dans un groupe d'espèces. Ils peuvent être codants, c.-à-d. que l'information génétique représentée par l'ordre des nucléotides qu'ils contiennent peut être décodée en macromolécules biologiques appelées protéines, directement à l'origine des fonctions cellulaires.

L'information génétique peut être copiée, transmise et exprimée. Le dogme central de la biologie moléculaire, énoncé à la fin des années 1950 par Francis Crick, établit les bases nécessaires pour comprendre les relations entre ces processus (voir Figure 2.1).

Au cours de la réplication de l'ADN, une nouvelle molécule d'ADN identique à l'originale est synthétisée. Ce processus permet de dupliquer le matériel génétique pendant la division cellulaire, assurant ainsi que les cellules filles obtenues ont la même information génétique que la cellule mère.

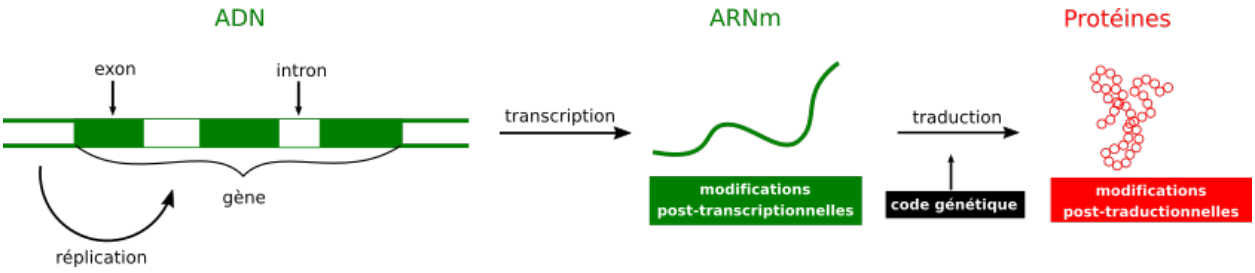


Figure 2.1. Un des procédés d'expression des gènes spécifiés par le dogme central de la biologie moléculaire. Les composantes formées de nucléotides sont indiquées en vert (ADN et ARN) et celles constituées d'acides aminés sont en rouge (protéines). Les deux grandes étapes à savoir transcription et traductions sont indiquées. Le code génétique intervient au niveau de la traduction des ARNm en protéines.

L'information génétique héréditaire peut également être transmise à des molécules autres que l'ADN, à savoir les ARNs et les protéines, afin d'assurer des fonctions biologiques au sein des cellules. On parle alors d'expression génique.

L'expression des gènes codants se fait généralement en deux étapes (voir Figure 2.1). Tout d'abord, la séquence du gène est copiée pour former une nouvelle molécule, simple brin, où les thymines sont remplacées par de nouveaux nucléotides appelés uracile (U). La séquence obtenue est appelée ARN messager (*ARNm*). L'étape de conversion du gène en ARNm est connue sous le nom de transcription. Après transcription, l'ARN peut être soumis à un ensemble de modifications dites post-transcriptionnelles. En particulier, chez les eucaryotes, la transcription est souvent suivie de l'excision des régions non codantes des gènes, appelées introns (par opposition aux régions codantes nommées exons).

Au cours de la seconde phase appelée traduction, l'ARNm est traduit en chaînes polypeptidiques constituant les protéines, grâce à une machinerie au centre de laquelle se trouve un complexe ribonucléoprotéique appelé ribosome. La synthèse des polypeptides s'effectue à l'intérieur du ribosome, en lisant la molécule d'ARNm par triplets successifs de nucléotides appelés codons, et en faisant correspondre à chaque codon un des 20 acides aminés standard représentant les unités élémentaires des protéines. La traduction se résume donc essentiellement à un réencodage de l'information d'un alphabet de 4 nucléotides, vers un alphabet de 20 acides aminés (AA).

L'intermédiaire clé dans ce transfert d'information est un ARN non codant, appelé ARN de transfert (ARNt), qui adopte une structure secondaire en forme de feuille de trèfle (Figure 2.2.b). L'ARNt est spécifiquement chargé avec un acide aminé au cours d'une réaction catalysée par des protéines nommées aminoacyl-ARNt synthétases (aaRS). Il dispose aussi d'un anticodon, généralement positionné entre les bases 34-36, qui s'apparie aux codons de l'ARNm en utilisant des règles étendues de complémentarité entre nucléotides. Chaque codon est donc lu par un ARNt spécifiquement chargé avec l'AA adéquat. Étant donné qu'il y a 64 codons pour une vingtaine d'AA, plusieurs codons sont décodés par le même acide aminé. Il est à noter que l'appariement entre anticodon et codon n'a pas à être parfait, ce qui fait que plusieurs codons différents peuvent être reconnus par le même ARNt, réduisant considérablement le répertoire des ARNt à un nombre plus petit que celui des codons. Les ARNt qui partagent la même identité, et donc chargés avec le même AA sont désignés par le terme *ARNt isoaccepteurs*, alors que ceux ayant une identité différente sont des *ARNt alloaccepteurs*.

En général, la traduction commence à partir d'un codon appelé *codon d'initiation* qui est souvent le codon AUG codant pour l'acide aminé méthionine. Il n'est pas rare que d'autres codons comme celui de la valine (GUG) soient utilisés à la place. D'autres codons spéciaux appelés *codons-stop* ou *codons non-sens* (UGA, UAG, UAA) servent à signaler la fin de la traduction et permettent la libération de la chaîne polypeptidique. Contrairement aux

a.

	U	C	A	G
U	UUU Phe UUC UUA Leu UUG	UCU UCC Ser UCA UCG	UAU Tyr UAC UAA stop UAG	UGU Cys UGC UGA stop UGG Trp
C	CUU CUC Leu CUA CUG	CCU CCC Pro CCA CCG	CAU His CAC CAA Gln CAG	CGU CGC Arg CGA CGG
A	AUU AUC Ile AUA AUG Met	ACU ACC Thr ACA ACG	AAU Asn AAC AAA Lys AAG	AGU Ser AGC AGA Arg AGG
G	GUU GUC Val GUA GUG	GCU GCC Ala GCA GCG	GAU Asp GAC GAA Glu GAG	GGU GGC Gly GGA GGG

b.

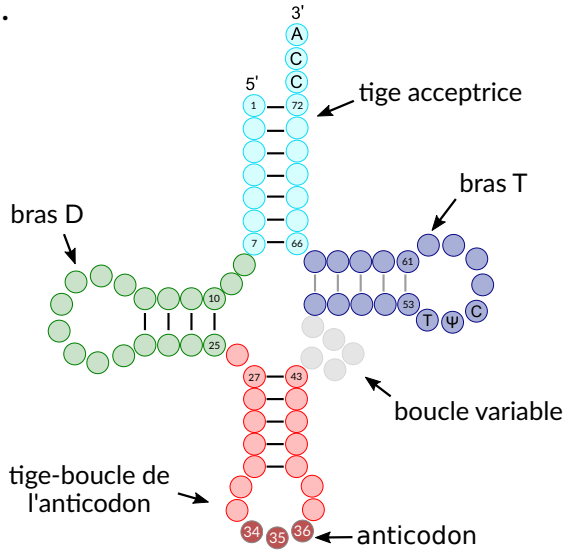


Figure 2.2. **a.** Le code génétique standard représenté sous forme d'une table. La correspondance entre chaque codon et chaque acide aminé est indiquée. Nous utilisons le rouge pour mettre en évidence les codons-stop et le vert pour le codon initiateur. **b.** Structure secondaire en feuille de trèfle des ARNt. Chaque tige-boucle est indiquée par une couleur différente et chaque nucléotide par un cercle. Les lignes courtes entre deux nucléotides représentent l'appariement entre la paire considérée de nucléotides. Ces appariements peuvent ne pas être canoniques. La position de certains nucléotides est indiquée. En particulier, les nucléotides 34,35 et 36 forment l'anticodon. Notez que la nomenclature utilisée est celle standard basée sur la Phe-ARNt de la levure [33] et qu'il existe de nombreux ARNt dont les structures n'obéissent pas à cette forme standard.

autres codons, les codons-stop ne sont normalement pas reconnus par un ARNt, mais par une protéine appelée facteur de terminaison.

L'ensemble des règles définissant la correspondance entre codons et acides aminés est appelé le *code génétique* (voir Figure 2.2.a) et est souvent représenté par une table montrant le décodage de chaque codon en l'AA correspondant. Dans la section 2.2, nous discuterons en profondeur des particularités de ce code.

2.1.1. Évolution des génomes

La structure, la régulation et la variation des processus biologiques au sein des organismes, sont fondamentalement déterminées par l'évolution de leurs génomes. Ces derniers évoluent essentiellement par mutations. Les mutations peuvent être causées par des facteurs environnementaux, des erreurs dans le processus de réplication de l'ADN ou encore par la

recombinaison génétique. Elles peuvent être transmises à la descendance de l'organisme lorsqu'elles sont localisées dans le génome des cellules reproductrices. On décline souvent les mutations en deux grandes catégories : les *mutations ponctuelles* et les *mutations globales*.

Les mutations ponctuelles affectent un ou plusieurs nucléotides de la séquence d'un même gène. Elles incluent la substitution qui consiste en un remplacement d'un nucléotide par un autre ; l'insertion où de nouveaux nucléotides sont ajoutés, et enfin la délétion (suppression) d'un ou plusieurs nucléotides. Ces mutations affectent la séquence des gènes et donc l'information génétique qu'ils contiennent. Elles peuvent être *non-synonymes* ou *synonymes*, et dans ce dernier cas, ne changent pas la séquence de la protéine exprimée, parce que plusieurs codons différents peuvent coder pour le même AA. Par exemple, la substitution de la dernière base des codons CUN ne changera par le décodage du codon en tant que leucine, et n'aura de ce fait pas d'incidence sur la séquence protéique. Notons toutefois que certaines substitutions synonymes peuvent avoir des impacts sur l'expression des gènes, en affectant la vitesse de la traduction (appariement inefficace avec l'anticodon, ou rareté des ARNt correspondants) [34–37].

Les mutations globales, par contre, agissent à une échelle plus large et affectent l'architecture des chromosomes. On parle alors de *réarrangements chromosomiques*. Ces réarrangements peuvent modifier le contenu, l'organisation et l'ordre des gènes au sein des génomes. Sous l'appellation mutations globales, nous regroupons les évènements de duplication, perte, transfert horizontal, translocation, inversion, transposition et de fusion/fission de chromosomes. Dans cette thèse, nous nous intéresserons plus particulièrement aux phénomènes de duplications, de pertes, et à un certain degré, aux transferts horizontaux.

La duplication présente un intérêt particulier, puisqu'il s'agit d'un des processus fondamentaux dans l'évolution des espèces à cause de son rôle dans la création de nouveaux gènes et par extension, d'innovations biologiques [7, 38–40].

Les mécanismes biologiques à l'origine des duplications sont variés et incluent la duplication entière de génome, la duplication en tandem, transposée, et la rétro-transposition. À travers ces différents mécanismes, de nouvelles copies de gènes peuvent être créées. En raison de la redondance fonctionnelle des gènes dupliqués, l'une des copies bénéficie souvent d'une pression sélective relaxée et devient alors libre d'évoluer, donnant lieu à quatre scénarios connus :

1. La copie dupliquée conserve la fonction originale du gène et permet ainsi d'augmenter son niveau d'expression.
2. Plus rare, la copie acquiert une nouvelle fonction en accumulant des mutations ponctuelles.

3. Les deux copies se sous-fonctionnalisent pour devenir complémentaires, nécessitant leur présence simultanée pour la préservation de la fonction ancestrale.
4. Enfin, le sort le plus fréquent de la copie de gène dupliqué est la perte totale de sa fonction par accumulation excessive de mutations. On parle dans ce cas de pseudogénisation.

En plus de la duplication, les gènes peuvent aussi être acquis par transfert horizontal. Le *transfert horizontal de gènes* (HGT) constitue une intégration de matériel génétique provenant d'un autre génome. Ce mode d'acquisition de matériel génétique s'oppose à la transmission verticale où l'information génétique est héritée d'un organisme parent par sa descendance. Les HGT sont particulièrement répandus chez les procaryotes. L'existence de transfert horizontal de gènes chez les eucaryotes est également bien documentée (voir la revue par Keeling et Palmer dans [41]). Cependant, le degré de leur importance dans l'évolution des eucaryotes reste controversé [42, 43].

Enfin, un dernier processus courant d'évolution des génomes qui affecte leur contenu est la perte de gènes. Cette perte peut être physique, c.-à-d., causée par la délétion d'un segment chromosomique, ou peut être due à une pseudogénisation. Dans cette thèse, nous ne distinguerons toutefois pas ces causes.

En raison de ces modes d'évolution des génomes par mutations héritables, il existe un polymorphisme génétique entre individus d'une même espèce. Les mutations ponctuelles agissant sur les séquences nucléotidiques de gènes peuvent en effet créer plusieurs variants d'un même gène au sein d'une population. Ces variants, appelés *allèles*, contribuent à la diversité génétique au sein des espèces et sont l'objet d'étude en génétique des populations.

La variabilité génétique est encore plus importante entre espèces, ce qui fait de l'évolution des génomes le processus clé pour comprendre la diversité biologique et retracer l'histoire évolutive des espèces.

Dans cette thèse, l'accent est mis sur la variabilité génétique inter espèces, ce qui nous permet de représenter toute une espèce par un génome de référence provenant d'un unique individu. Nous continuerons à faire la distinction entre "génomes" et "espèces" dans cette section, mais soulignons le fait que les deux concepts pourront être utilisés de façon interchangeable dans le reste de cette thèse.

2.1.2. Spéciation : formation de nouvelles espèces

Lorsqu'au sein d'une population d'une espèce donnée, un groupe accumule suffisamment de mutations génomiques et diverge assez pour ne plus pouvoir se reproduire avec le groupe

restant, on assiste à une scission de la population donnant naissance à de nouvelles espèces. Ce phénomène que l'on désigne par *spéciation* marque la formation de deux nouvelles espèces à partir d'une espèce ancestrale. Il est très souvent facilité par la présence de barrières géographiques séparant les deux groupes, les forçant ainsi à évoluer indépendamment.

Pour que les mutations accumulées engendrent une spéciation, il faut essentiellement qu'elles soient communes à tout le groupe divergent et donc *fixées* dans ce groupe. Les mécanismes sous-jacents de la spéciation ne sont toujours pas entièrement élucidés, mais on admet la contribution importante de plusieurs forces évolutives telles que la dérive génétique et la sélection naturelle, sous l'influence de facteurs liés à l'environnement ou à la taille de la population.

Après une spéciation, les deux espèces descendantes héritent du matériel génétique de l'espèce ancestrale. Ceci nous permet de définir le concept de famille de gènes.

2.1.3. Famille de gènes

Une famille de gènes est un ensemble de gènes qui descendent tous du même ancêtre commun. Cette définition de famille de gènes soulève des questionnements, parce qu'en fin de compte, si on remonte assez loin dans l'évolution, tous les gènes descendent d'une même séquence ancestrale et forment donc une même et unique famille [44].

En pratique, on se limite aux gènes descendants du même ancêtre commun qui partagent une similarité de séquences et retiennent souvent des fonctions similaires en raison du lien bien documenté entre séquence, structure et fonction [45–48]. Cette seconde définition permet de regrouper les gènes en familles, par comparaison de leurs séquences.

Un exemple de famille de gènes avec une histoire complexe est celui des globines. Les globines se déclinent en plusieurs types (les chaînes des différentes hémoglobines, la myoglobine, la cytoglobine, etc.) qui, bien que possédant toutes plus ou moins la même fonction fondamentale de stockage/transport de l'oxygène, exhibent une diversité significative au niveau structural et dans les nuances entre leurs fonctions [49–51].

La notion de famille de gènes est intimement liée à l'ensemble d'espèces considéré. Une implication directe de cette notion est que chacun des gènes d'un groupe d'espèces doit appartenir à une famille, définissant donc une classe d'équivalence sur l'ensemble des gènes. Notons toutefois l'existence de gènes hybrides (obtenus par exemple, par réarrangements génomiques), qui rendent difficile cette classification, mais que nous ignorons dans cette thèse.

Deux gènes appartenant à la même famille de gènes sont *homologues*. Parmi les gènes homologues, nous distinguons :

— les gènes *orthologues*, qui descendent par spéciation d'un même ancêtre commun. Par exemple, les gènes de l'hémoglobine β chez l'humain et la grenouille sont orthologues. Il est important de noter que la relation d'orthologie entre gènes n'est pas une relation d'équivalence. Le gène de l'hémoglobine ϵ chez l'humain est aussi orthologue au gène de l'hémoglobine β chez la grenouille, mais l'hémoglobine β et ϵ de l'humain n'ont pas divergé par spéciation et ne sont donc pas orthologues.

— les gènes *paralogues*, qui divergent par un évènement de duplication de leur ancêtre commun. Par exemple, les gènes des chaînes α et β de l'hémoglobine chez l'humain sont paralogues, car ils sont issus d'une duplication. De la même façon les gènes de l'hémoglobine α chez l'humain et de la myoglobine chez la souris sont aussi paralogues, car leur divergence remonte à la duplication très ancestrale d'une globine.

— les gènes *xénologues*, qui divergent par un transfert horizontal.

Nous ne considérons ici que la définition la plus simple de la xénologie. Il existe d'autres définitions, comme celle de Fitch [52] qui stipule que les gènes sont xénologues si leur histoire évolutive, à partir de leur divergence, est marquée par la présence d'au moins un évènement de transfert. Nous référerons le lecteur à l'article pertinent de Darby *et al.* [53] qui propose une classification des xénologues en plusieurs sous-types.

Les gènes orthologues présentent un intérêt particulier en génomique comparative. On conjecture qu'à des taux similaires de divergence de séquence, ils ont beaucoup plus tendance à conserver la même fonction, comparés aux autres homologues [9, 54]. Cette hypothèse reste débattue (voir [10, 55–58] pour une variété d'opinions), et il n'est pas rare que les orthologues divergent au niveau fonctionnel [59] ou que des gènes partagent la même fonction sans pour autant être orthologues [60, 61]. Malgré tout, cette conjecture sur les orthologues est l'hypothèse prévalente, et plusieurs analyses génomiques à savoir l'annotation de nouvelles séquences, la prédiction de la structure et de la fonction des protéines ou encore les analyses phylogénétiques, reposent presque entièrement sur l'identification des gènes orthologues [62, 63]. Pour inférer efficacement les relations d'orthologie entre gènes, il faut nécessairement déterminer d'abord l'histoire évolutive des familles de gènes. Dans le chapitre 3, nous détaillons des méthodes pour réaliser cette tâche.

2.2. Le code génétique et son évolution

Le code génétique représente l'ensemble des règles permettant le déchiffrage de l'information génétique contenue dans les séquences d'acides nucléiques, en séquences protéiques.

Ce code assigne à chacun des $4^3 = 64$ codons, un acide aminé spécifique ou un signal de terminaison de la traduction.

L'histoire de la détermination du code génétique s'étale sur plusieurs années avec la contribution d'un grand nombre de chercheurs. Il s'agit de l'une des plus grandes avancées scientifiques du 20e siècle en biologie, en dépit des limites technologiques de l'époque qui rendaient difficile l'analyse des séquences moléculaires.

Dès 1955, Crick proposait sa théorie sur l'existence d'une molécule adaptatrice qui assurait le transfert de l'information génétique des nucléotides vers les acides aminés et representerait ainsi le principal acteur exécutant le code. Mais on attendra encore plusieurs années avant que la structure des ARNt et leur rôle dans la traduction ne soient établis [64].

Tout d'abord, il a fallu déterminer comment la correspondance entre nucléotides et acides aminés était effectuée. En se basant sur la taille de l'alphabet des protéines et sur celui des séquences de nucléotides, Gamow a postulé qu'il fallait que chaque acide aminé soit codé par un triplet de nucléotides, sous l'hypothèse d'une colinéarité entre les séquences d'AA et celles d'acides nucléiques [65, 66]. Ce résultat a ensuite été confirmé par l'expérience de Crick *et al.* [67], qui ont montré que le code opérait effectivement à partir de triplets de nucléotides, grâce aux études de mutations sur le cistron rIIB du phage T4. Cette découverte a aussi révélé que les triplets ne sont pas chevauchants, et qu'il existe un sens unique de traduction, avec un point de démarrage précis. Le travail de Crick et de ses collègues démontre également que la relation définie par le code génétique n'est pas injective, c.-a.-d, que plusieurs codons codent pour le même acide aminé. On dit que le code est dégénéré.

La correspondance précise entre chaque arrangement des triplets de nucléotides et chaque acide aminé a ensuite été progressivement déterminée par synthèse de polypeptides à partir de polymères de nucléotides spécifiquement élaborés. La cumulation de ses travaux a donné lieu au décryptage du code génétique de la bactérie *Escherichia coli* entre 1961 et 1966 [68].

2.2.1. Propriétés du code génétique

L'examen minutieux du code génétique après son décryptage a révélé qu'il possède certaines propriétés qui semblent conservées à travers tout le vivant.

Le code est dégénéré. Comme nous l'avons vu plus tôt, plusieurs codons codent pour le même acide aminé. À l'exception de la méthionine et du tryptophane, tous les autres acides aminés standard sont codés par plusieurs codons. La redondance du code génétique est beaucoup plus prononcée au niveau de la troisième base des codons en raison de l'existence

d'appariements non canoniques entre codon et anticodon à cette position. L'existence occasionnelle de bases modifiées dans la séquence des ARNt accentue aussi la présence de ces liaisons. La dégénérescence du code est donc causée, non seulement par la présence d'ARNt isoaccepteurs, mais aussi par le fait que certains ARNt peuvent reconnaître plusieurs codons. De tels codons partageant la même identité sont appelés *codons synonymes*. Par exemple, la leucine, l'arginine et l'alanine ont chacun six codons synonymes (voir Figure 2.2). Un avantage de la redondance du code génétique est qu'elle permet aux organismes de mieux tolérer les mutations aléatoires dans leurs génomes, en réduisant les risques que celles-ci affectent les séquences protéiques (cf. mutations synonymes à la section précédente). On peut toutefois noter que même si le code est redondant, la plupart des organismes exhibent un biais, variable selon les espèces, dans l'usage des codons, avec une préférence nette pour certains codons synonymes. Ce biais est souvent positivement corrélé avec un biais mutationnel ou une sélection pour une synthèse protéique plus efficace.

Le code est univoque. Bien qu'il soit dégénéré, le code n'est pas ambigu, dans le sens où chaque codon ne spécifie naturellement qu'un seul acide aminé. Il est important de comprendre certaines nuances de cette propriété. Les décalages du cadre de lecture, les sauts de ribosomes, les erreurs de lecture et les modifications post-transcriptionnelles, qui créent, pour certains ARNm, une disparité entre leurs séquences et celle des protéines qu'ils encodent, n'affectent pas la légitimité de cette propriété. Ces phénomènes que nous désignerons par *recodage* n'altèrent donc pas l'identité des codons, qui reste unique. Nous verrons quand même, dans la suite, qu'il existe des cas rares où des codons possèdent une double identité.

Le code est structuré. La distribution des codons en classes d'équivalence partageant la même identité n'est pas aléatoire. Très souvent, les acides aminés les plus fréquemment utilisés au sein des protéines sont associés à un nombre plus important de codons synonymes et inversement. Par ailleurs, les AA avec des propriétés physico-chimiques similaires sont généralement encodés par des codons proches, séparés par une seule substitution. Par exemple, les deux seuls AA avec un groupement carboxyle, et donc acides, soit l'acide aspartique (Asp) et l'acide glutamique (Glu), sont respectivement codés par les codons GAA, GAG et GAU, GAC qui ne diffèrent qu'au niveau de la troisième base. Ces observations ont suscité beaucoup d'intérêts dans la communauté scientifique et ont poussé plusieurs chercheurs à formuler l'hypothèse selon laquelle le code serait optimisé pour réduire les

effets des mutations et/ou des erreurs pendant la traduction.

Le code est (presque) universel. De prime abord, le code génétique semble partagé par presque tous les êtres vivants. Ainsi, le code génétique déterminé pour le génome bactérien de *E. coli* reste valide pour les humains et les autres organismes d'autres domaines du vivant. Le fait que le code soit universel supporte l'idée d'un ancêtre commun unique pour toutes les espèces, chez qui le code était déjà présent. L'universalité apparente du code est d'ailleurs exploitée pour la synthèse en grande quantité de protéines humaines à des fins thérapeutiques à partir d'autres systèmes biologiques facilement contrôlables. Nous verrons toutefois, dans le reste de cette section, qu'il existe un nombre considérable de génomes, principalement non-nucléaires, utilisant des codes génétiques pour lesquels les identités de certains codons sont différentes. Nous parlerons plus explicitement d'évolution du code. L'existence de ces exceptions introduit donc les notions de *code génétique standard* (celle vue jusqu'à maintenant) et d'altérations ou déviations du code.

2.2.2. Évolution du code génétique

L'assignation des codons aux AA est déterminée par la liaison précise et spécifique de ces derniers à l'ARNt, puis par l'appariement entre anticodons des ARNt et codons des séquences d'ARNm. Ce procédé sophistiqué et l'apparente universalité du code a conduit plusieurs à suggérer que le code était fixe. Crick a ainsi postulé sa théorie du "Frozen accident" qui affirme que l'assignation des codons aux AA est aléatoire et que le code est universel et immuable parce que toute variation dans le décodage des codons affecterait simultanément les séquences de la majorité des protéines, et aurait ainsi des conséquences létales pour les organismes [69].

Cependant, la découverte d'un code génétique alternatif utilisé par le génome mitochondrial (ADNmt) humain a très vite démontré que les codons pouvaient être réassignés à d'autres AA, et a suggéré que le code était capable d'évoluer [70, 71]. Depuis lors, plusieurs études ont révélé l'existence de nombreuses déviations du code génétique, dans une multitude d'organismes provenant de tous les règnes du vivant [72–74]. À l'écriture de cette thèse, 24 tables génétiques distinctes, que l'on suppose toutes dérivées du code standard [74], sont répertoriées sur la base de données NCBI [23], et on s'attend à ce qu'il y en ait beaucoup plus. Bien que les causes de l'évolution du code génétique ne soient toujours pas élucidées, certains auteurs proposent que la réassignation de codon pourrait conférer un avantage sélectif aux organismes. Par exemple, dans [75], les auteurs proposent que les modifications du code génétique offrent une résistance accrue aux infections virales.

2.2.2.1. *Origine du code génétique*

La possibilité que le code puisse évoluer ne fait que soulever plus de questions par rapport à son origine, donnant lieu à une série de spéculations et de controverses. Il est généralement admis que le code actuel a évolué à partir d'un état plus simple [76, 77], mais comment s'est mis en place le système complexe observé aujourd'hui ? Et quelles sont les parts des contraintes chimiques, des forces évolutives et du hasard dans son établissement ? Les réponses à ces questions demeurent inconnues [69].

Certaines caractéristiques intrigantes du code génétique suggèrent que son évolution vers sa forme actuelle, parmi les 1.5×10^{84} autres possibilités, n'est pas simplement le fruit du hasard [77]. On peut noter : (1) son organisation structurelle en blocs, qui contribuerait à sa robustesse contre les mutations ponctuelles, les erreurs de décodage et les décalages de lecture [78–80] ; (2) la relation entre la seconde base des codons et les propriétés physico-chimiques des acides aminés [81] ; (3) la corrélation négative entre le nombre de codons synonymes et le poids des AA qu'ils encodent [82], et celle positive avec la fréquence de ces AA [83, 84].

L'élucidation de l'origine du code génétique est étroitement liée au problème de l'identification des molécules précurseures des fonctions biologiques dans le monde prébiotique. Une hypothèse populaire est celle de l'ARN comme molécule primordiale assurant à la fois les fonctions biologiques et le stockage de l'information génétique [69, 85, 86]. Cette hypothèse est supportée par la découverte des ribozymes, ARN possédant une fonction catalytique [87, 88], et laisse supposer que la synthèse originale de protéines était entièrement catalysée par des molécules d'ARN.

Puisque les ARN ribosomiques sont extrêmement conservés [89] et possèdent des domaines pouvant former des liaisons peptidiques [90], ils auraient pu être utilisés pour synthétiser les premiers peptides [76, 91–93]. Il n'est pas certain que ces peptides aient eu des fonctions biologiques spécifiques, mais on spécule qu'ils auraient contribué à stabiliser la machinerie naissante de traduction. La prochaine étape présumée pour l'établissement d'un code ancestral est l'acquisition de molécules d'ARNt pour le transport des acides aminés et le recrutement d'un ARN d'ancrage afin d'assurer que les ARNt soient maintenus sur place pendant toute la durée de la synthèse [91, 92]. Cet ARN d'ancrage constituerait l'ancêtre de l'ARNm que nous connaissons aujourd'hui [94].

Les protéines produites par ce code primitif ne contenaient pas tous les 20 acides aminés standard, puisqu'en effet, seulement 10 acides aminés (Gly, Ala, Asp, Glu, Val, Ser, Ile, Leu,

Pro, Thr), dits précurseurs, pouvaient être synthétisés à partir des matériaux inorganiques du monde prébiotique [95, 96].

Il est évident que le système primordial ainsi décrit, ne peut-être spécifique et ne garantit pas non plus une traduction efficace et fidèle. Le code génétique a donc dû évoluer et se développer à partir de ce système, vers sa forme moderne dans laquelle l'assignation des codons aux acides aminés est faite d'une façon extrêmement structurée.

Trois théories non mutuellement exclusives ont été proposées pour expliquer cette évolution à partir de facteurs physico-chimiques et biologiques (voir les revues par Koonin dans [77] et [97]) :

1. **La théorie stéréochimique**. Elle postule que l'assignation entre codons et AA a été déterminée par l'existence d'affinités physico-chimiques entre acides nucléiques et acides aminés [65, 98]. Sous cette hypothèse, l'assignation des codons aux acides aminés n'est pas aléatoire, mais potentiellement unique en raison de contraintes chimiques. La théorie stéréochimique est partiellement supportée par les analyses de liaison entre oligonucléotides synthétiques et acides aminés.
2. **La théorie adaptative**. Elle affirme que l'évolution du code génétique est principalement causée par des forces de sélection visant à minimiser les effets des erreurs durant la synthèse protéique [99, 100]. Cette théorie est supportée par le fait que les acides aminés ayant des propriétés physico-chimiques similaires sont encodés par des codons similaires, et par le fait que plusieurs études de simulation montrent qu'il existe un biais important vers la minimisation d'erreurs dans le code standard, en comparaison aux codes alternatifs [101, 102].
3. **La théorie de la coévolution**. Elle stipule que la structure du code reflète directement celle des voies de biosynthèse des acides aminés, suggérant ainsi une co-évolution entre les deux [103]. Selon cette théorie, les codons des acides aminés précurseurs ont été réassignés plus tard aux acides aminés secondaires synthétisés à partir de ces derniers.

2.2.2.2. *Déviations connues du code génétique*

À l'heure actuelle, un grand nombre de déviations au code génétique standard ont été découvertes dans plusieurs organismes recouvrant l'entièreté des domaines du vivant. En particulier, les réassignations de codons semblent surreprésentées au sein des génomes mitochondriaux, par rapport aux génomes nucléaires. Cette disparité est expliquée par la taille réduite des génomes mitochondriaux et la pression mutationnelle plus élevée à laquelle ils sont soumis [104].

Nous adopterons la nomenclature $C(X \rightarrow Y)$ pour représenter la réassignation d'un codon C de l'acide aminé X à l'acide aminé Y.

Les réassignations *stop-à-sens* de codons-stop à l'un des AA standard Trp, Glu, Gln, Cys, Tyr, ainsi qu'aux deux AA rares que sont la sélénocystéine et la pyrrolysine, ont été plusieurs fois rapportées dans la littérature (voir revue dans [74, 105]). La réassignation UGA(Stop → Trp) reste cependant celle *stop-à-sens* la plus fréquente et est retrouvée dans plusieurs génomes mitochondriaux, ainsi que dans certains systèmes non mitochondriaux [106].

Une autre réassignation fréquente, cette fois-ci *sens-à-sens*, est AUA(Ile → Met) très répandue dans les génomes mitochondriaux de levures et de métazoaires [72, 107]. AUA(Ile → Met) et UGA(Stop → Trp) présentent un intérêt particulier, parce qu'elles augmentent le nombre de codons synonymes de la méthionine et du tryptophane, les seuls acides aminés encodés par des codons uniques dans le code standard. Cette extension du nombre de codons synonymes de Met et Trp à d'autres codons appartenant au même bloc (resp. AUR et UGR) a pour effet attendu une réduction du taux d'erreurs de la synthèse protéique. Bien qu'on admette en général que l'évolution du code est neutre par nature et qu'elle n'est pas principalement déterminée par la sélection positive, la fréquence de ces deux réassignations dans le vivant suggère qu'elles confèreraient un certain avantage sélectif.

Dans les génomes mitochondriaux de levure, on retrouve aussi les réassignations CUN(Leu → Thr) et CUN(Leu → Ala) [108–110]. Les levures *Candida* réassignent également le codon CUG de Leu à Ser dans leurs génomes nucléaires [111, 112]. Récemment, on a aussi découvert que ce codon est lu comme Ala dans une poignée d'autres hémiascomycètes [113–115].

Chez les métazoaires, les variations du code génétique mitochondrial sont nombreuses et concernent très souvent les codons AGR de l'arginine [72, 74]. Dans de nombreux génomes, ces codons sont réassignés à Gly, Ser, ou sont parfois utilisés comme stop.

La table suivante présente une liste des réassignations de codons documentées dans le vivant. Elle montre que les altérations du code génétique standard sont présentes dans presque tous les domaines du vivant, très variées et peuvent affecter aussi bien les génomes d'organelles que ceux nucléaires.

Tableau 2. I. Liste représentative, mais non exhaustive de réassignations naturelles de codons dans le vivant.

Réassignation	**Génomes**	**Références**
AUA(Ile→Met)	ADNmt des Xantophytes (algues) et de la plupart des métazoaires et des levures	voir revue dans [72, 74, 116]
CUN(Leu→Thr)	ADNmt de certains *Saccharomyces* (levures)	[108, 109]
CUN(Leu→Ala)	ADNmt de *Ashbya* (levures)	[110, 116]
UAG(Stop→Ala)	ADNmt des Chlorophyceae (algues vertes)	[117]
UAG(Stop→Leu)	ADNmt des Chlorophyceae, *Chytrids* et des *Spizellomyces*	[117–119]
UGA(Stop→Trp)	Firmicutes, ADNmt des métazoaires et de la plupart des micro-organismes eucaryotes (*Monosiga*, *Amoebidium*, *Acanthamoeba*, Ascomycètes, Rhodophytes, Kinetoplastides, *Pedinomonas minor*, *Pycnococcus provasolii*, *Chondrus crispus*, etc.)	voir revue dans [72, 74] et [120–122]
UAG(Stop→Pyl)	Certaines archées et bactéries (*Clostridia*, *Negativicutes*, γ-*Proteobacteria*)	[123–125]
UAR(Stop→Gln)	Phages, ADN nucléaire de certaines algues vertes et de certains protistes	[105, 126–129]

UCR(Ser→Gly)	ADNmt des Sphaeropleales (algues vertes)	[118, 119, 130]
AGR(Arg→Ser)	ADNmt de la plupart des invertébrés	[72, 74, 116, 131]
AGR(Arg→Gly)	ADNmt des Tunicates	[72, 74, 116, 131]
AGR(Arg→Stop)	ADNmt des vertébrés	[74]
AAA(Lys→Asn)	ADNmt des Échinodermes	[72, 118, 132]
AGG(Arg→Lys)	ADNmt de plusieurs arthropodes	[116, 133]
CUG(Leu→Ser)	ADN nucléaire de plusieurs espèces de *Candida* et certains Ascomycètes	[111, 134–136]
CUG(Leu→Ala)	ADN nucléaire de certains Ascomycètes (*Pachysolen*, *Nakazawaea*, etc.)	[113, 115, 137]

2.2.2.3. *Mécanismes d'évolution du code*

Puisque la modification du code génétique affecte tous les gènes d'un organisme, sa cause la plus rationnelle est une altération des biomolécules impliquées dans la synthèse des protéines (voir Figure 2.3). Cette altération peut être une mutation à des sites critiques qui changent l'identité des ARNt. Notez que l'identité des ARNt est souvent déterminée par la présence d'éléments ou d'appariements précis à des positions particulières de sa structure (comme le bras accepteur), par des changements conformationnels et parfois par des modifications post-transcriptionnelles (voir les revues dans [73, 138, 139]). Une modification du site permettant la reconnaissance de l'AA par l'aaRS aura un effet similaire sur l'identité des ARNt. Par ailleurs, des mutations localisées au niveau de l'anticodon amèneront l'ARNt à reconnaître des codons complètement différents, changeant ainsi leurs décodages. Pour les codons stop, la mutation peut être localisée au niveau du facteur de terminaison de la traduction, empêchant la reconnaissance du codon stop.

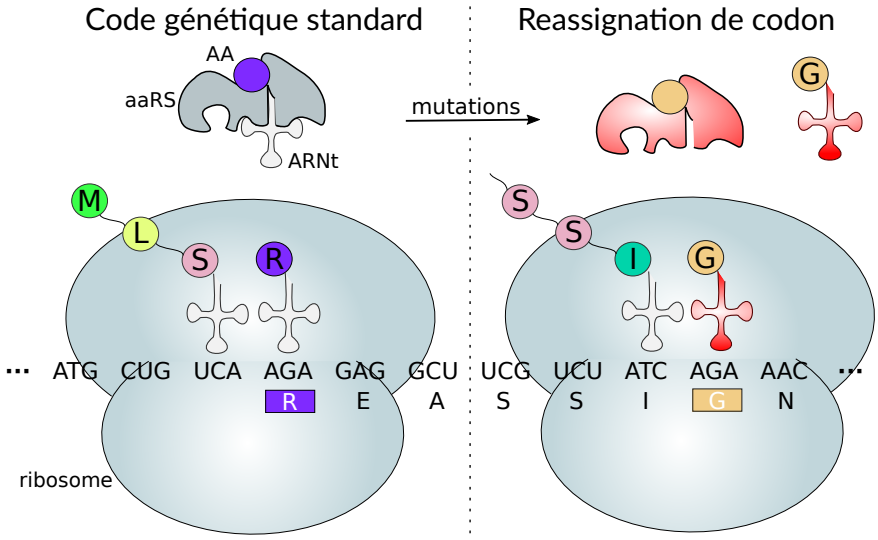


Figure 2.3. Réassignation de codons. Le code génétique standard et la traduction normale sont montrés à gauche. À droite nous avons le code résultant après la réassignation AGA(R → G). Nous mettons en évidence comment certaines mutations au sein de la machinerie de traduction (au niveau de l'aaRS ou de l'ARNt) peuvent conduire à la réassignation d'un codon.

Plusieurs mécanismes ont été proposés pour expliquer les réassignations de codons (voir Figure 2.4).

Capture de codons. Ce mécanisme est aussi connu sous le nom de "disparition de codons". Selon cette hypothèse formulée par Osawa *et al.* [140–142], les réassignations de codons sont initiées par la disparition complète du codon en raison des processus stochastiques de mutation affectant le contenu en G+C des génomes. Cette disparition est suivie de la perte de l'ARNt correspondant, devenu inutile. Après réapparition des codons, ainsi libres, par relaxation de la pression sur la composition en bases G+C ou par dérive génétique, ils sont capturés par d'autres ARNt chargés avec un acide aminé différent. La présence d'un ARNt ayant évolué pour reconnaître le codon à sa réapparition est indispensable, car autrement la synthèse protéique serait bloquée au niveau de ce codon. Ce mécanisme a été proposé pour expliquer, par exemple, la réassignation des codons stop UGA en tryptophane chez *Mycoplasma capricolum* [143]. En général, puisque les codons stop sont rares par nature, il n'est pas impossible que certains d'entre eux disparaissent des séquences codantes des génomes. De plus, il est aussi envisageable qu'une mutation subséquente des facteurs de terminaison empêche aussi la reconnaissance de ces codons. La théorie est toutefois critiquée, car la disparition totale d'un codon, strictement par des processus d'évolution

neutre, est peu probable lorsque les génomes sont larges.

Intermédiaire ambigu. Contrairement à la théorie précédente, Schultz et Yarus affirment que la disparition totale du codon avant la réassignation n'est pas nécessaire [144, 145]. Ils proposent, à la place, l'existence d'un état intermédiaire dans lequel le codon est traduit en deux acides aminés distincts. La réassignation a lieu lorsqu'un avantage sélectif survient pour le décodage non standard et est nécessairement marquée par la perte du décodage canonique [146]. Dans les articles [112], [135] et [147], des auteurs suggèrent qu'un tel décodage ambigu est supporté par la réassignation CUG(Leu → Ser) dans le génome nucléaire de certaines levures [148, 149]. En effet, un ARNt(CAG), pour ces codons, qui possède une double identité (leucine, sérine), a été identifié dans le génome nucléaire de plusieurs *Candida* [136, 149]. Le décodage ambigu de codons a aussi été expérimentalement observé par introduction de nouveau ARNt compétiteurs dans les génomes de *C. albicans* et *S. cerevisiae* [150, 151], montrant que le scénario est biologiquement plausible. Notons toutefois que ce type de décodage a sûrement un impact important sur les fonctions biologiques au sein des organismes concernés, puisqu'il introduit une perturbation non-négligeable dans leurs protéomes. Certains auteurs suggèrent que ce mécanisme serait utilisé pour l'adaptation des micro-organismes à un environnement hostile/extrême. [135, 151]

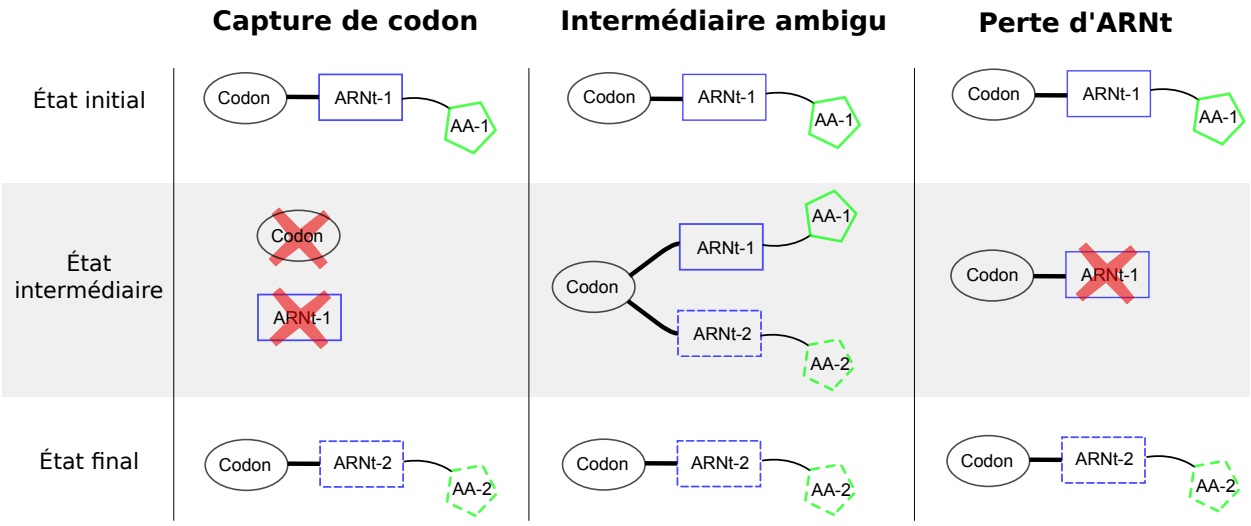


Figure 2.4. Trois mécanismes de réassignation de codons et leurs étapes de transition.

Réduction de la taille du génome. Cette hypothèse stipule que des pressions évolutives, comme la vitesse de réplication, visant à maintenir une taille réduite des génomes, conduisent à la minimisation de la machinerie de traduction et éventuellement aux réassignations de codons [152, 153]. Une forte pression négative sur la taille des génomes mitochondriaux pourrait en effet causer une modification de la fréquence des codons, et potentiellement la perte de certains ARNt, soit directement ou par relaxation de la pression pour les maintenir dans le génome. Ce mécanisme explique bien les nombreuses pertes, non-assignations et réassignations de codons en particulier des codons stop, dans les génomes mitochondriaux de plusieurs eucaryotes [154].

Perte d'ARNt canonique. Beaucoup plus récemment, Mühlhausen et Kolmar ont proposé un nouveau mécanisme qui stipule que les réassignations de codons serait principalement causées par la perte (de fonction ou de gènes) des ARNt ou des facteurs de terminaison [113, 155]. Cette hypothèse se distingue de celle précédente par l'absence d'une pression sélective pour réduire la taille des génomes. Pour compenser la perte des ARNt canoniques, les codons pourraient être décodés par d'autres ARNt, isoaccepteurs ou possédant un anticodon qui permettrait un *appariement bancal*. La reconnaissance du codon par un ARNt avec une identité différente pourrait ultimement conduire à sa réassignation. Notons toutefois que ce modèle ne peut être fonctionnel que si l'anticodon de l'ARNt captureur ne fait pas partie des éléments déterminant son identité. Les auteurs soutiennent que ce mécanisme expliquerait la polyphylie du décodage des codons CUG en tant que leucine, sérine ou alanine dans les génomes nucléaires des hémiascomycètes, réfutant ainsi la théorie prédominante du décodage ambigu. Ils vont plus loin en avançant que le mécanisme pourrait expliquer toutes les réassignations rapportées à date. Cependant, l'absence complète de codons, autrement réassignés dans des génomes voisins, alors qu'un ARNt correspondant est disponible, prouve bien que ce mécanisme n'est pas le bon dans certains cas.

Modélisation des réassignations par gains et pertes. Enfin, Sengupta et Higgs ont proposé un modèle unificateur qui considère les réassignations de codons comme un processus de gains et pertes, où l'ordre et le type de ces évènements déterminent le mécanisme d'évolution du code [72]. Ce modèle découle de l'observation selon laquelle une réassignation inclue toujours un évènement de perte (codon, ARNt ou changement de l'identité d'un ARNt) et/ou un gain (apparition d'un nouvel ARNt ou gain de fonction pour un ARNt existant), qu'on peut modéliser mathématiquement pour étudier l'évolution du code par simulation. Leur modèle intègre les mécanismes de capture de codons et

d'intermédiaire ambigu, en plus de deux nouveaux mécanismes qu'ils proposent : *les codons non-assignés* et les *changements compensatoires*. Le mécanisme de réassignation par codons non-assignés intervient lorsque l'ARNt ou le facteur de terminaison est perdu, laissant le codon non-assigné, avant qu'il n'y ait un évènement de gain pour décoder le codon sous une nouvelle identité et pallier le désavantage sélectif résultant. Ce mécanisme est donc analogue à la perte d'ARNt canonique proposé par Mühlhausen et Kolmar. Les réassignations initiées par la perte d'un ARNt ne sont toutefois pertinentes que s'il existe d'autres ARNt capables de décoder le codon, même de façon inefficace. Selon la réassignation par changements compensatoires, les évènements de gain/perte apparaissent de façon continuelle dans le génome et le désavantage sélectif qu'ils peuvent causer est neutralisé par des mutations compensatoires (voir [156]) avant qu'ils ne soient fixés dans la population. Ces mutations compensatoires peuvent conduire finalement à l'altération du code.

Les mécanismes d'évolution du code génétique, énumérés ci-dessus, soulèvent plusieurs débats et controverses en raison des scénarios contradictoires proposés par différents auteurs. Par exemple, rien que pour la réassignation des codons CUN de la leucine vers la thréonine dans le génome mitochondrial des *Saccharomyces* et vers l'alanine chez *Ashbya* (voir Figure 2.5), trois mécanismes différents ont été proposés : intermédiaire ambigu [157], réduction de la taille génomique [113], capture de codons [72, 110].

Il est donc difficile de déterminer de façon exacte le mécanisme ayant causé une réassignation, d'autant plus que la plupart des mécanismes partagent en commun plusieurs attributs. Par exemple, la capture de codon, l'intermédiaire ambigu et la perte d'ARNt canonique nécessitent tous une baisse importante (voire totale) de la fréquence d'utilisation d'un codon, que cela soit par l'effet aléatoire de mutations neutres, ou par sélection contre ce codon en raison du désavantage sélectif qui survient pendant la phase transitoire de la réassignation. À cela s'ajoute l'ordre, souvent inconnu, des évènements de perte de l'ARNt canonique et d'acquisition d'un ARNt mutant qui déterminent essentiellement le type de mécanisme. Une vue personnelle que nous avons et qui est supportée par nos résultats présentés dans le chapitre 5, est qu'il faut considérer et étudier indépendamment chaque cas de réassignations de codons, sans chercher nécessairement à les forcer à s'accorder avec un des mécanismes existants.

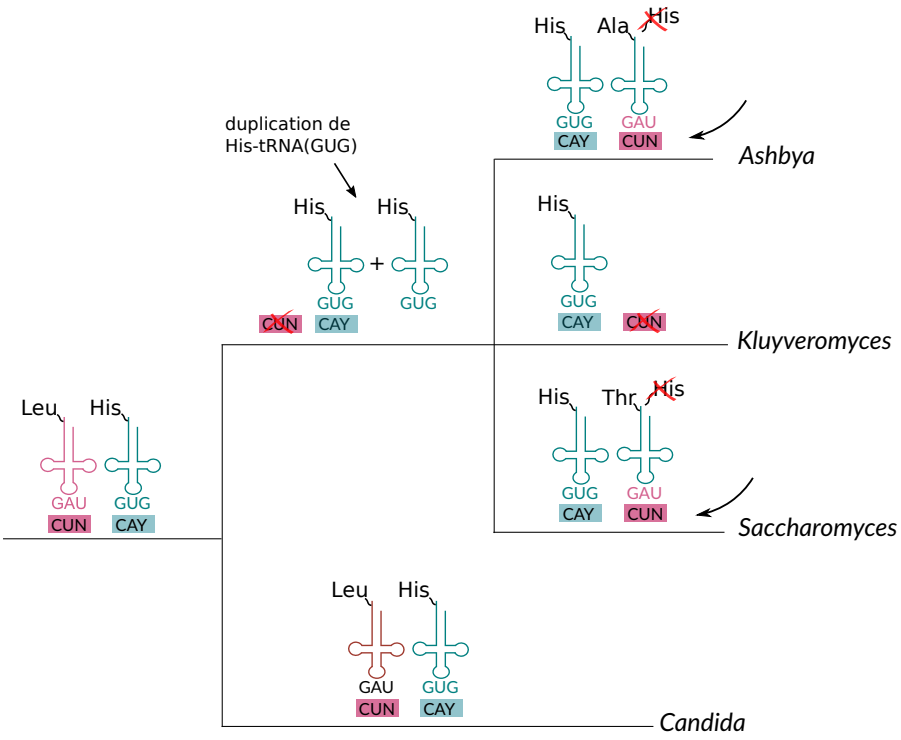


Figure 2.5. Réassignations des codons CUN dans les génomes mitochondriaux de certaines levures. Les codons CUN sont normalement décodés par un Leu-ARNt(UAG), mais chez les *Saccharomyces*, ils sont décodés par un Thr-ARNt(UAG), obtenu par duplication et mutation du His-ARNt(GUG) ancestral. Chez *Ashbya*, ils sont décodés par un Ala-ARNt(UAG) partageant la même origine que le Thr-ARNt(UAG) des *Saccharomyces*. On note bien la disparition des codons CUN (encore observable chez *Kluyveromyces*) indiquée par une croix rouge, ainsi que la perte du Leu-ARNt(UAG) ancestral, mais le scénario évolutif donnant lieu à la réassignation reste débattu.

2.2.2.4. *Prédiction des altérations du code génétique*

L'étude des réassignations de codons permet non seulement de mieux comprendre les mécanismes moléculaires de l'évolution, mais a également plusieurs utilités en biologie moléculaire [158]. La connaissance des règles régissant l'évolution du code peut en effet être exploitée en génie génétique pour étendre le code génétique d'un génome hôte [159–162] et contrôler la précision de la synthèse protéique afin de produire, efficacement, des protéines recombinantes actives [163, 164]. Il devient aussi naturellement possible d'introduire des variations structurelles au sein des protéines pour étudier beaucoup plus rigoureusement leurs

fonctions [164] et faciliter le développement de médicaments ciblant spécifiquement l'activité de ces protéines [165].

Plus important encore, le code génétique a une conséquence considérable sur l'exactitude des annotations de génomes, et par extension l'ensemble des analyses effectuées à partir de ces génomes. Par conséquent, la détermination du vrai code génétique représente une étape indispensable à effectuer avant toute analyse génomique.

Ces raisons expliquent l'importance de développer des méthodes efficaces pour prédire les réassignations de codons. En principe, la réassignation de codon peut être inférée en comparant le vrai protéome (obtenu par spectrométrie de masse à très haute résolution) au protéome prédit à partir du transcriptome, étant donné un code génétique provisoire. En cas d'altération du code génétique, on s'attend à observer un patron conntsistent de différences qu'on peut ensuite mettre en contraste avec les séquences génomiques pour identifier les codons réassignés. Les réassignations peuvent ensuite être validées par analyse biochimique de l'activité des ARNt et des aaRS. Cette procédure présente l'avantage de pouvoir différencier convenablement les modifications du code génétique, et les évènements de recodage qui n'affectent que quelques ARNm spécifiques. Cependant, elle n'est pas pratique, parce qu'elle nécessite la disponibilité du génome, du transcriptome et du protéome, et à un coût (temps et argent) important. À la place, il est possible de développer des approches *in silico* d'une efficacité comparable, tout en étant plus rapides.

Les méthodes computationnelles pour la prédiction de réassignations de codons exploitent l'information sur l'évolution des gènes homologues au sein des génomes. En raison de la pression pour maintenir la fonction et donc la séquence des gènes orthologues, plusieurs sites de ces gènes restent conservés au cours de l'évolution des génomes. En présence d'une modification non-détectée du code génétique dans un génome G_x, on notera des différences systématiques entre les séquences protéiques des gènes de G_x et leurs orthologues au sein des génomes proches de G_x. Ces différences s'observeront également au niveau des sites hautement conservés. Ainsi, par comparaison de la séquence nucléotide des gènes à leurs séquences protéiques attendues (à partir des orthologues), on peut identifier les réassignations de codons.

```
G1: MEKT···TINT···T···*
G2: MEKT···TMNT···T···*
G3: MDKT···TINT···T···*
G4: MDKL···LL-L···L···*
G5: MD-T···TINT···T···*
```

Par exemple, pour les gènes orthologues de la figure ci-dessus, la thréonine est systématiquement substituée par la leucine dans le génome G_4. Si cette substitution est récurrente

dans tout le génome, même en considérant une correction pour l'évolution par mutations neutres et aléatoires, alors on a de bonnes raisons de penser que les codons de la leucine trouvés aux positions correspondantes sont réassignés vers la thréonine dans G_4. Bien évidemment, une telle approche requiert de disposer de séquences de référence pour lesquelles la vraie traduction des gènes est connue, ou de façon plus permissive, s'assurer que la réassignation étudiée n'est pas commune à tous les génomes considérés, afin qu'un signal soit observable. De plus, il n'est pas toujours facile d'évaluer la précision de la méthode. Comment, par exemple, distinguer entre réassignations de codons et mutations authentiques et fréquentes ?

Dans le chapitre 4, nous présentons une méthode, pour la prédiction de déviations du code génétique, qui s'appuie principalement sur l'hypothèse formulée ci-dessus, mais utilise une démarche garantissant une précision et une sensibilité élevée. Nous comparons cette méthode aux rares alternatives qui existent et qui sont aussi basées sur un principe similaire.

Chapitre 3

Inférence de l'histoire évolutive des familles de gènes

Ce chapitre introduit les notions et terminologies relatives à l'inférence de l'histoire évolutive, que nous utilisons dans les chapitres 6 et 7 de cette thèse.

Tout d'abord, dans la section 3.1, nous décrivons l' "arbre" comme structure de données pour représenter les relations de parentés entre entités. Nous présentons ensuite les méthodes de comparaison de topologies d'arbres (Robinson-Foulds, distance NNI/SPR, etc.) et finissons la section en établissant une distinction entre arbre de gènes et d'espèces puis en présentant les étapes nécessaires à la construction des arbres phylogénétiques.

La section 3.2 présente les méthodes génériques de construction d'arbres phylogénétiques (de gènes ou d'espèces) à partir d'alignements de séquences. La compréhension du fonctionnement de ces méthodes est requise pour identifier les sources d'erreurs potentielles et justifier la nécessité de développer des algorithmes, beaucoup plus spécifiques, qui tirent avantage de l'existence d'autres sources d'information sur l'évolution des familles de gènes.

La section 3.3 est consacrée à l'inférence des évènements marquant l'histoire des familles de gènes à partir des différences topologiques entre arbres de gènes et arbres d'espèces. Cette section clarifiera le concept de la "réconciliation".

Finalement, dans la section 3.4, nous présentons les méthodes récentes, que nous appelons *intégratives*, qui sont spécifiques à l'inférence d'arbres de gènes. Ces méthodes exploitent, outre l'information des séquences, de nouvelles sources de données pour construire ou corriger les arbres de gènes. Les algorithmes que nous décrivons dans les chapitres 6 et 7 s'inscrivent dans ce dernier cadre.

3.1. Arbres

Un *arbre* est une structure de données permettant de représenter et d'organiser de l'information sous forme d'arborescence. Les premières utilisations de l'arbre pour représenter

des relations entre entités biologiques remontent au 19e siècle avec Lamarck qui l'avait utilisé dans *"Philosophie Zoologique"* pour établir un "ordre de formation des animaux" [166]. Ce n'est toutefois qu'avec Darwin, dans son ouvrage fondateur *"De l'origine des espèces"* que les arbres seront utilisés pour représenter un *processus dynamique d'évolution* des espèces à partir d'ancêtres communs, notamment avec l'introduction de la notion de divergence entre espèces [167].

De Darwin à aujourd'hui, la théorie de l'évolution a été raffinée et les outils utilisés pour la décrire ont aussi été formalisés. Dans cette section, nous introduisons les terminologies et notations formelles sur les arbres, d'abord comme outils mathématiques, puis comme processus décrivant l'évolution de gènes et d'espèces.

Notons que les arbres ne permettent pas forcément de représenter efficacement l'évolution en présence d'évènements de réticulation tels que la fusion de gènes, les enjambements chromosomiques, les transferts horizontaux et l'hybridation d'espèces, et peuvent même parfois induire des erreurs d'interprétation. À la place, il est préférable d'utiliser les *réseaux phylogénétiques* [168, 169]. Dans cette thèse toutefois, outre les transferts de gènes, les évènements de recombinaison ne sont pas pris en compte, ce qui nous permet de nous restreindre à la représentation de l'évolution à partir d'arbres.

3.1.1. Notation et définitions

Soit T un arbre, c.-à-d. un graphe connexe et acyclique. Nous désignons respectivement par $V(T)$, $E(T)$ et $L(T)$, l'ensemble des *sommets* (ou *noeuds*), des *arêtes* (ou *branches*) et des *feuilles* de T. Par cette définition $L(T) \subset V(T)$ et on dit que T est un arbre sur $L(T)$. La taille d'un arbre correspond au nombre de sommets $|V(T)|$ qu'il possède. Nous utiliserons également $|T|$ pour nous y référer.

Le degré d'un sommet x désigne le nombre de sommets voisins de x (reliés par une arête à x) dans T. Chaque feuille de T a un degré de 1 et on appelle *noeuds internes*, les autres noeuds dans $V(T) \setminus L(T)$. T est dit *enraciné* s'il possède un sommet unique $r(T)$ appelé racine dont on se sert pour orienter les arêtes. Sauf indications contraires, tous les arbres considérés dans cette thèse sont enracinés et chaque noeud interne, excluant la racine, a un degré plus grand que 2.

Nous désignons par $e = (x,y) \in E(T)$, l'arête e reliant les sommets x et y, si elle existe, x étant le sommet le plus proche de la racine. Soit x et y deux noeuds de $V(T)$, y est un *descendant* (resp. descendant propre) de x si et seulement si x se retrouve sur le chemin allant de la racine $r(T)$ à y, incluant y (resp. excluant y). On peut alors écrire $y \leq x$ (resp. $y < x$). x est alors appelé un *ancêtre* (resp. ancêtre propre) de y. On dit que x et y sont

incomparables, si y n'est ni le descendant, ni l'ancêtre de x. Si (x, y) est une arête de T, il s'en suit que x est un ancêtre de y, et on dit plus précisément que x est le *parent* $p(y)$ de y. Dans ce cas, y est un *enfant* de y et on écrit $y \in Ch(x)$ (Ch référant à "Children") . La racine $r(T)$ est le seul noeud de T sans parent. Si T est un arbre enraciné, la profondeur d'un sommet x notée $\delta(x)$ désigne le nombre de noeuds, x inclu, le séparant de $r(T)$, avec $\delta(r(T)) = 0$.

Pour un arbre T, nous désignons par T_x le sous-arbre de T enraciné en $x \in V(T)$ et contenant x ainsi que tous ses descendants. Deux sous-arbres T_x et T_y sont dit *séparés*, si et seulement si x et y sont incomparables dans T.

L'*ancêtre commun le plus récent* ou LCA (pour l'anglais *Lowest Common Ancestor*) d'un ensemble de noeuds $X \subseteq V(T)$ est noté $lca_T(X)$ et désigne l'ancêtre, le plus éloigné possible de $r(T)$, de tous les noeuds de X. Par exemple, sur la Figure 3.1, $lca_T(\{a,b,c\})$ dans l'arbre T_2 est le noeud v. Nous désignons également par T_L l'arbre ayant pour ensemble de feuilles $L \cap L(T)$ qui est obtenu à partir du sous-arbre de T enraciné en $lca_T(L \cap L(T))$ en supprimant toutes les feuilles qui ne sont pas présentes à la fois dans L et $L(T)$, puis en supprimant les noeuds internes résultants qui ont moins de deux enfants. Pour deux arbres T' et T tel que $L(T') \subset L(T) \wedge T_{L(T')} = T'$, nous disons que T *affiche* T'.

Si x est un noeud de $V(T)$, le clade de x dénoté par $c(x)$ est le sous-ensemble de feuilles $L(T_x)$. Nous désignons par $\mathcal{C}(T)$ l'ensemble des clades de T.

Un parcours d'un arbre est une procédure qui ordonne les noeuds d'un arbre pour le traverser. Lors d'un parcours préfixe, chaque noeud est visité avant ses enfants. Lors d'un parcours postfixe par contre, les noeuds sont visités uniquement après avoir traversé leurs enfants.

Ci-dessous, nous définissons certaines notions sur les arbres. Ces notions sont illustrées sur la Figure 3.1.

Enracinement d'arbre

L'enracinement d'un arbre T revient à définir une racine $r(T)$ pour cet arbre. Il s'agit d'un procédé qui consiste à ajouter un nouveau noeud z entre une arête (x,y) et à orienter l'arbre de façon à ce que $z = r(T)$, créant ainsi deux nouvelles arêtes (z, x) et (z,y). Notons que z n'a pas de parent dans T. Le nombre total d'enracinements possibles d'un arbre non enraciné équivaut au nombre total d'arêtes de cet arbre, soit $|E(T)|$. Un arbre peut être ré-enraciner par suppression de sa racine et création d'une nouvelle racine sur une arête différente.

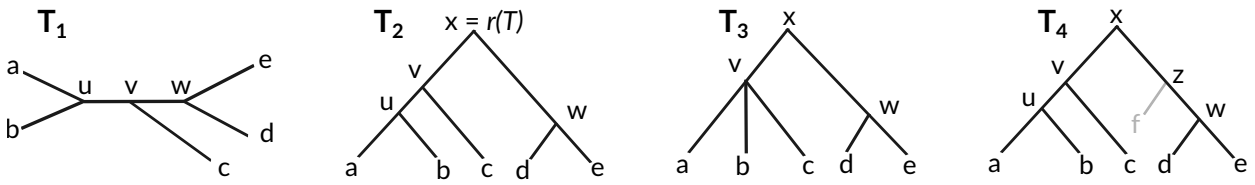


Figure 3.1. Illustrations des notions d'enracinement, binarisation, et extension d'arbres. T_1 est un arbre non enraciné sur $L(T_1) = \{a, b, c, d, e\}$. T_2 est un enracinement de T_1 sur l'arête (v,w) par ajout d'un nouveau sommet x qui représente la racine de T_2. T_3 est un arbre non binaire obtenu par la contraction de l'arête (v,u) dans T_2. Une conséquence est que T_2 est une binarisation de T_3. Enfin T_4 est une extension de T_2 par la greffe d'une nouvelle feuille f (en gris) sur la branche (x,w). Cette greffe nécessite l'ajout d'un nouveau noeud interne z entre x et w.

Extension d'un arbre

Un arbre T' est une extension d'un arbre T si T' peut s'obtenir par une séquence de *greffe*, où chaque greffe consiste à subdiviser une arête (x,y) de $E(T)$ en créant un nouveau noeud intermédiaire z entre x et y, puis en ajoutant une nouvelle feuille z' avec pour parent z. À la place d'une feuille z', on peut aussi greffer un arbre enraciné en z'.

Arbre binaire et non binaire

On dit qu'un arbre enraciné T est *binaire* si tous ses noeuds internes ont exactement 2 enfants. Pour un noeud x d'un arbre binaire, nous désignons respectivement par x_l et x_r ses enfants gauche et droit. Si T est non enraciné, alors T est binaire si et seulement si tous ses sommets sont de degrés 1 ou 3.

Un *arbre étoile* est un arbre T qui a un seul noeud interne qui représente sa racine. Chaque enfant de $r(T)$ est une feuille de $L(T)$. Nous appelons *polytomie*, un arbre étoile dont la racine a un degré plus grand que 2. Dans le cas d'un arbre enraciné, les polytomies correspondent donc aux noeuds internes avec plus de deux enfants. Un arbre *non binaire* est un arbre qui contient au moins une polytomie.

Contraction de branches

La *contraction* d'une arête (x,y) d'un arbre T consiste à supprimer le noeud y, puis à rattacher toutes les arêtes incidentes à y, autres que (x,y) au noeud x. Tous les enfants de y deviennent donc des enfants de x. Si T est un arbre binaire, la contraction d'une arête (x,y) crée donc une polytomie en x.

Binarisation d'un arbre

La *binarisation* d'un arbre T, que nous appellerons aussi *résolution* ou *affinage binaire* de T, est un arbre binaire T' sur $L(T)$ avec $V(T) \subset V(T')$ tel que pour toute paire $\{x, y\}$ de noeuds dans T, si x est un ancêtre de y dans T alors x est également un ancêtre de y dans T'. Plus formellement :

Definition 3.1.1 (Affinage binaire)**.** *Soit T un arbre non binaire. Un affinage binaire $T' = B(T)$ de T est un arbre binaire tel que $V(T) \subseteq V(T')$ et pour tout noeud $x \in V(T)$, $L(T_x) = L(B_x)$. Nous désignerons par $\mathcal{R}(T)$, l'ensemble de toutes les binarisations possibles de T.*

Le nombre total de résolutions possibles d'une polytomie avec $n+1$ feuilles correspond au nombre total d'expressions formées à partir de n paires de parenthèses correctement assemblées (aussi connu en tant que mots de Dyck de longueur $2n$). Un dénombrement du nombre de solutions à ce problème donne $\binom{2n}{n} - \binom{2n}{n-1}$, qui équivaut au nombre de Catalan $C_n = \frac{(2n)!}{(n+1)!n!}$. Il y a donc un nombre exponentiel d'affinages binaires pour un arbre non binaire. Comme nous le verrons dans la suite de cette thèse, l'affinage binaire de polytomies est un problème d'intérêt en phylogénie. On définit souvent des critères précis pour lesquels il faut produire les résolutions optimales.

3.1.2. Comparaison de topologies d'arbres

Les arbres, définis sur un même ensemble de feuilles, peuvent être comparés pour évaluer à quels points ils sont similaires. Cette comparaison s'effectue généralement sur la base de leur topologie, en respectant les étiquettes à leurs feuilles. On utilise très souvent des métriques qui permettent de mesurer à quel point deux arbres sont similaires. La plus populaire de ces métriques est la distance Robinson-Foulds (RF) [170] qui permet de comparer deux arbres non enracinés en comptant le nombre de différences symétriques entre les ensembles de leurs *splits*, c.-à-d. des bipartitions définies sur les arêtes de chaque arbre.

Une bipartition définie sur une arête $e = (x,y)$ d'un arbre T correspond à la partition $B_e = \{c(x), c(y)\}$ où $c(x)$ et $c(y)$ représentent respectivement les clades des sous-arbres enracinés en x et y, après suppression de l'arête e. Ainsi, $c(x) \cup c(y) = L(T)$ et $c(x) \cap c(y) = \emptyset$. B_e est dite non-triviale si au moins une de ses composantes n'est pas un singleton (c.-à-d. e n'est pas une arête incidente à une feuille). Par exemple, pour l'arbre T_1 de la Figure 3.1, $B_{u,v} = \{\{a,b\}, \{c,e,d\}\}$ est une bipartition non triviale. Soit $\mathcal{B}_1$ et $\mathcal{B}_2$ les ensembles respectifs des bipartitions de deux arbres T_1 et T_2, la distance RF entre T_1 et T_2 que nous noterons

$RF(T_1, T_2)$ vaut $|\mathcal{B}_1 \setminus \mathcal{B}_2| + |\mathcal{B}_2 \setminus \mathcal{B}_1|$. Cette distance peut se calculer en temps linéaire et vaut 0 lorsque les deux arbres sont identiques et a une valeur maximale de $2n - 6$ lorsque toutes les bipartitions non-triviales des deux arbres sont différentes. En pratique, on utilise souvent une distance RF normalisée sur sa valeur maximale. Des variations de la métrique existent, en particulier, Górecki et Eulenstein [171] l'ont adaptée pour comparer des arbres enracinés à des arbres non enracinés.

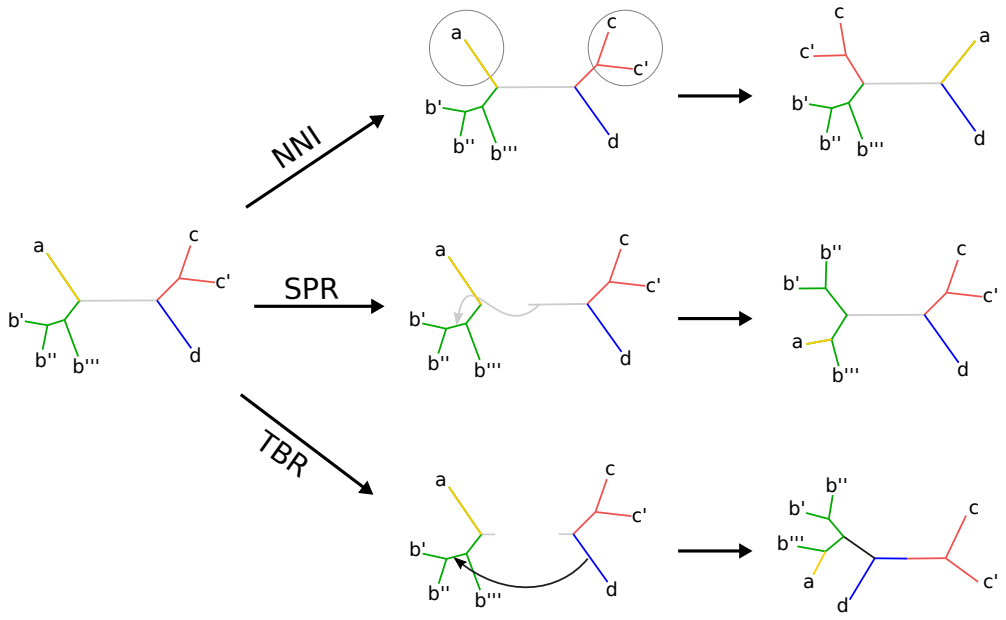


Figure 3.2. Mouvement de réarrangement : NNI, SPR et TBR illustrés sur un arbre binaire non enraciné. L'arête interne grise est utilisée pour montrer les différences entre chaque opération. L'opération NNI interchange les sous arbres a et (c,c'). Le SPR détache l'arête grise avec le sous-arbre $(d, (c, c'))$ et rattache l'ensemble sur une nouvelle branche de l'arbre restant. Le TBR supprime l'arête grise puis créer une nouvelle branche reliant des nouveaux noeuds internes dans chacun des sous-arbres résultants.

D'autres mesures basées sur les opérations modifiant la topologie des arbres existent. Il s'agit de mouvements de réarrangements topologiques habituellement utilisés comme heuristique de recherche pour explorer l'espace d'arbres que l'on peut définir sur un ensemble d'entités [172, 173]. Dans la section 3.2, nous reviendrons sur leur usage pour l'inférence d'arbres phylogénétiques.

Il est possible de définir une distance sur ces opérations. Dans ce cas, on cherche à trouver le nombre minimum de réarrangements pour transformer un arbre T_1 en un autre arbre T_2. Les opérations les plus couramment utilisées sont le *Nearest Neighbour Interchange* (NNI),

le *Subtree Pruning and Regrafting* (SPR) et le *Tree Bisection and Reconnection* (TBR) (voir Figure 3.2). Le NNI échange deux sous-arbres séparés par une arête interne [174]. Le SPR enlève une arête e, et le sous-arbre qui y est attaché, puis greffe e sur un arête de l'arbre restant. Enfin le TBR consiste à enlever un sous-arbre, créer un nouveau sommet sur une arête choisie pour chacun des deux arbres ainsi obtenus, puis à joindre ces deux sommets par une nouvelle arête. Notons qu'un mouvement NNI est un mouvement SPR qui à son tour est un mouvement TBR. La complexité croissante de ces opérations est corrélée avec la taille de l'espace de topologie alternative qu'elles peuvent explorer. Le calcul de la distance NNI/SPR/TBR optimale entre deux arbres est un problème NP-difficile [175].

3.1.3. Arbres phylogénétiques d'espèces et de gènes

Les arbres phylogénétiques (nous utiliserons aussi le terme *phylogénie*) sont des arbres utilisés pour représenter l'évolution d'un ensemble d'entités. Dans le cadre de cette thèse, ces entités seront, soit des gènes, soit des espèces.

Au sein des arbres phylogénétiques, les longueurs de chaque branche ont de l'importance. Elles représentent en effet une mesure de la distance évolutive séparant deux sommets (espèces ou gènes). La longueur des branches est souvent indiquée en nombre de substitutions par site et peut être convertie en temps, lorsque le taux de substitution sur la branche d'intérêt est connu.

Un arbre phylogénétique d'espèces S est un arbre décrivant l'évolution d'un ensemble $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ d'espèces existantes (ou de fossiles). Il s'agit donc d'un arbre sur $L(S) = \Sigma$, dont les noeuds internes représentent des évènements ordonnés de spéciation marquant la divergence entre espèces (cf. section 2.1.2). Un exemple d'arbre d'espèces pour les primates est illustré sur la Figure 3.3.

Un arbre de gènes G, par contre, représente l'histoire de la transmission d'un gène d'une espèce ancestrale à des espèces existantes donnant lieu à une famille de gènes homologues Γ. Comme illustré sur la Figure 3.4, les arbres de gènes peuvent être imbriqués au sein de celui des espèces pour mettre en évidence cette histoire.

Nous dénoterons par $s(x)$ l'espèce de Σ d'où provient chaque gène x de G. Lorsqu'il n'y a pas besoin de faire une distinction entre les copies d'un gène, les gènes peuvent juste être représentés par leur génome d'origine dans G, ce qui peut parfois donner lieu à des étiquettes répétées dans $L(G)$.

Les noeuds internes des arbres de gènes peuvent correspondre à des évènements de spéciation, marquant la transmission du gène de l'espèce ancestrale σ à chacune de ses espèces descendantes σ_l et σ_r. Les spéciations dans les arbres de gènes sont donc directement reliées

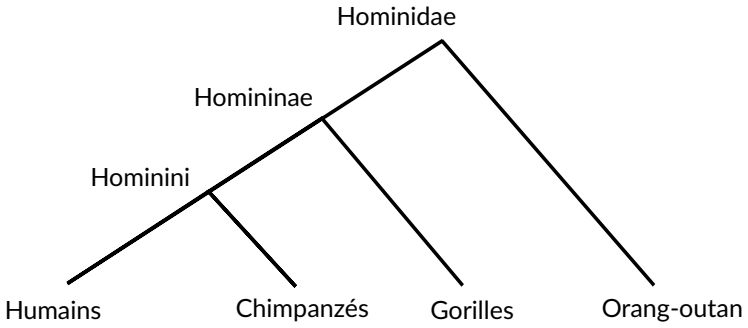


Figure 3.3. Un arbre d'espèces qui décrit les relations entre primates. Les feuilles de l'arbre représentent des espèces existantes actuellement et les noeuds internes les espèces ancestrales. Les noeuds internes correspondent à des spéciations donnant lieu à des groupes différents d'espèces. La représentation de cet arbre ne tient pas compte de la distance évolutive séparant les différentes espèces.

aux spéciations dans l'arbre d'espèces. En plus des spéciations, les noeuds internes des arbres de gènes peuvent également correspondre à d'autres types d'évènements comme les duplications, et les transferts horizontaux, qui affectent l'évolution des familles de gènes au cours de l'évolution des espèces. Après une spéciation ou une duplication, on s'attend à ce que toutes les copies du gène soient transmises aux espèces descendantes. Leur absence indique donc que le gène a été perdu à un moment donné au cours de l'évolution.

En raison de ces modes d'évolution des familles de gènes, il existe parfois des incongruences entre les topologies d'arbres de gènes et celle de l'arbre d'espèces. Par ailleurs, même lorsque les topologies des deux arbres sont apparemment identiques, elles peuvent ne pas refléter la même histoire. Certains noeuds internes de l'arbre des gènes peuvent en effet être des noeuds cachant une duplication. Par exemple, sur la Figure 3.4 des copies différentes d'un gène ancestral dupliqué sont perdues dans les espèces A et B après la spéciation en α. Le résultat est une topologie d'arbre de gènes identique à celle de l'arbre des espèces, lorsqu'on remplace chaque gène par l'espèce d'où il provient. Dans ce cas particulier, même s'il est beaucoup plus parcimonieux d'expliquer la divergence entre les gènes a_1 et b_2 par un évènement de spéciation, ce qui les rendrait orthologues, les deux gènes sont en fait paralogues.

Dans la section 3.3, nous montrerons comment les arbres de gènes et d'espèces peuvent être utilisés pour inférer les duplications, pertes et transferts de gènes, et par conséquent, l'histoire évolutive des familles de gènes. Étant donné que plusieurs histoires évolutives distinctes peuvent être inférées pour chaque famille de gènes, nous décrirons plutôt les méthodes

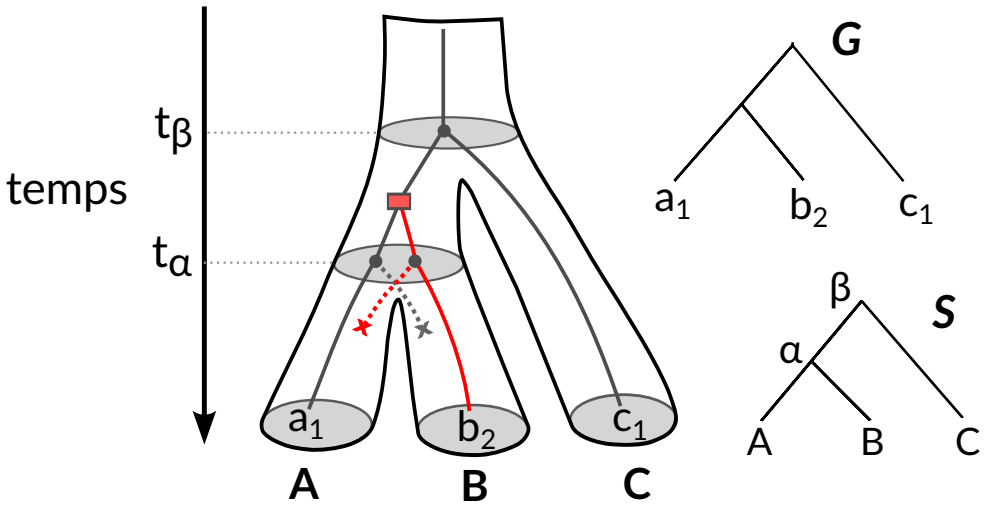


FIGURE 3.4. Arbres d'espèces vs arbres de gènes. La figure décrit l'évolution d'une famille de gènes : $\Gamma = \{a_1, b_2, c_1\}$ au sein des espèces $\Sigma = \{A, B, C\}$. Chaque gène en caractère minuscule appartient à l'espèce en majuscule. La figure montre une duplication entre les spéciations β et α, générant une deuxième lignée (en rouge) pour le gène. Au temps de spéciation t_α, la copie rouge est perdue en A et celle noire est perdue en B, donnant l'impression qu'il n'y avait qu'une seule copie du gène ancestral en α.

optimisant un critère de parcimonie. Ces méthodes obéissent ainsi au principe du *rasoir d'Occam* selon laquelle l'hypothèse la plus simple est la plus probable. Nous supposons ainsi que le cas décrit à la Figure 3.4 est un cas extrême qui survient rarement au cours de l'évolution des familles de gènes.

Dans la section 3.2, nous présenterons les méthodes de reconstruction d'arbres, et plus spécifiquement d'arbres de gènes, à partir de plusieurs sources d'informations, dont les alignements de séquences. Pour que cette section soit compréhensible, il nous faut d'abord établir les fondements de l'inférence d'arbres phylogénétiques de gènes et d'espèces. Les grandes étapes d'une telle reconstruction sont décrites ci-après.

3.1.4. Des séquences aux arbres d'espèces

La variabilité génétique peut être exploitée pour inférer les arbres phylogénétiques car ces derniers ne sont pas directement observables.

Comme l'illustre la Figure 3.5, la première étape de la reconstruction consiste à déterminer les séquences génomiques des espèces et à les annoter. Cette étape regroupe donc le séquençage, l'assemblage et l'annotation de génomes. Nous n'aborderons pas ces notions,

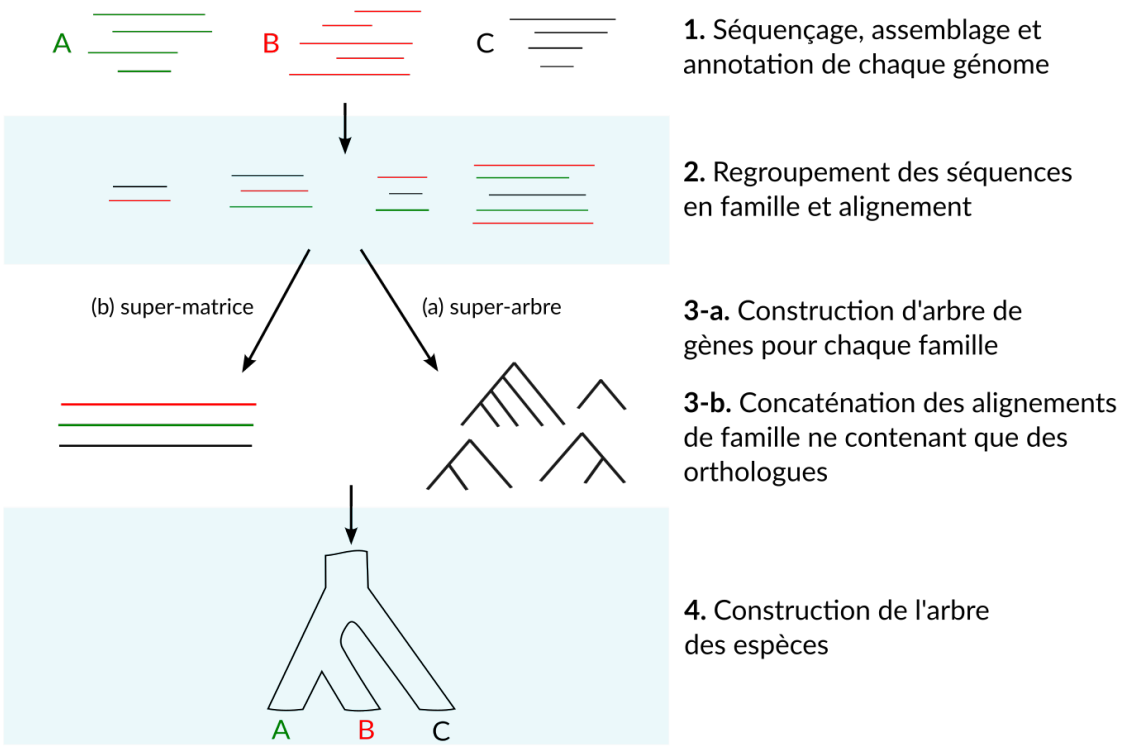


Figure 3.5. Étapes de reconstruction des arbres phylogénétiques à partir de séquences de gènes. Cette figure est adaptée de la Figure 1 de [176].

mais tenons à rappeler que l'annotation des génomes repose sur la similarité des séquences de gènes homologues (voir chapitre précédent).

Ensuite, les séquences homologues sont regroupées en familles de gènes puis alignées (étape 2 de la Figure 3.5). Très souvent, il s'avère nécessaire de peaufiner l'alignement en identifiant et en enlevant les régions ambiguës ou mal-alignées [177], souvent causées par une variation de la vitesse évolutive au sein des séquences. Différentes méthodes d'inférence phylogénétique, décrites dans la section 3.2 permettent ensuite de reconstruire l'arbre phylogénétique à partir de l'alignement.

Bien que fonctionnant relativement bien, ce système de reconstruction en deux étapes présente des limites, non seulement à cause de l'effet induit par de potentielles erreurs d'alignement, mais surtout de l'interdépendance entre alignement et phylogénie [178]. En effet, un alignement dépeignant correctement l'homologie entre sites nécessiterait un arbre décrivant les différentes relations entre sites et leur transmission au sein des séquences, cette dernière information ne pouvant être obtenue qu'à partir d'un alignement multiple de séquences. La non-prise en compte de cette dépendance pourrait résulter en une accumulation

d'erreurs se propageant dans les analyses subséquentes. Comme suggérés par Sankoff, Morel et Cedergren, la phylogénie et l'alignement de séquence devraient idéalement être estimés simultanément [178]. Quelques efforts dans ce sens existent [179, 180], mais la conception d'algorithmes efficaces et surtout utilisables en pratique, demeure d'actualité.

Les différentes étapes de construction des phylogénies sont brièvement détaillées ci-après (voir aussi [181]).

3.1.4.1. *Construction des familles de gènes*

Regrouper dans une même famille les gènes partageant la même origine est une étape préliminaire pour les reconstructions phylogénétiques. En théorie, la construction des familles de gènes nécessite la connaissance de leurs histoires évolutives pour déterminer leur homologie. Pour éviter une dépendance circulaire, nous utilisons l'une des propriétés des gènes homologues, à savoir la conservation de la similarité de leurs séquences. L'hypothèse sous-jacente est qu'en raison de certaines pressions évolutives comme la conservation de fonction, les gènes descendants d'un même ancêtre auront tendance à avoir des séquences plus similaires que deux gènes non homologues. On s'attend aussi par conséquent à ce que les gènes partageant une forte similarité de séquences aient divergé plus récemment, n'ayant pas encore accumulé assez de mutations.

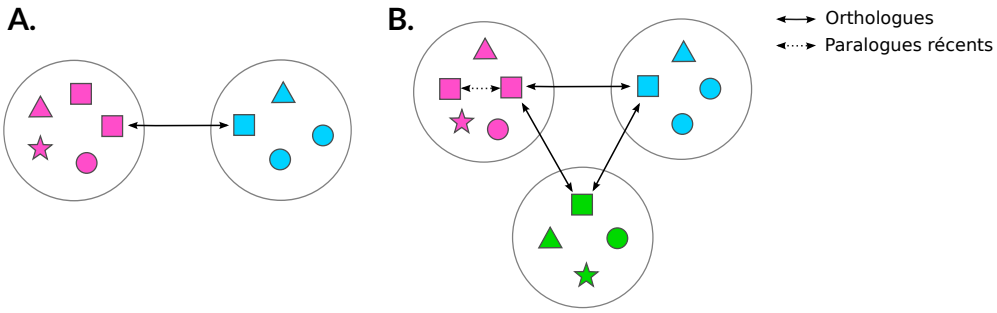


Figure 3.6. Construction de familles de gènes par similarité de séquences. Les gènes sont représentés par des formes géométriques variées, et les génomes par les cercles entourant ces formes. Les gènes de la même famille ont la même forme et les gènes provenant du même génome ont la même couleur. **A**. Meilleure similarité réciproque entre deux gènes. Cette méthode ne détecte par le paralogue dans le génome en magenta **B.** Extension de la meilleure similarité réciproque qui utilise une recherche *all-vs-all* pour construire un graphe de relation dont on se sert ensuite pour créer les groupes de gènes homologues. Les relations entre la famille de gènes carrée sont indiquées.

En se basant sur ce principe, les familles de gènes peuvent être construites en recherchant des séquences de gènes dont la similarité dépasse un certain seuil [182]. Pour ce faire, il est possible d'utiliser des algorithmes d'alignement et/ou de comparaison tels que Smith-Waterman et BLAST [183]. BLAST est une heuristique de recherche qui permet d'identifier très rapidement dans une base de données de séquences génomiques, celles similaires à la séquence en entrée. L'algorithme procède en identifiant au préalable des régions de similarité maximale (HSP pour *Highest Scoring Pairs*) entre deux séquences, puis étend le plus possible ces régions. BLAST se décline sous plusieurs formes prenant en compte les séquences nucléotidiques et protéiques et retourne un score appelé *e-value* qui indique à quel point le *hit* obtenu est significatif.

La procédure la plus élémentaire pour construire les familles de gènes consiste à identifier les gènes orthologues en utilisant le principe de la meilleure similarité réciproque [184]. Selon cette procédure, deux gènes provenant de deux espèces différentes sont homologues, et plus précisément orthologues, s'ils représentent l'un pour l'autre le meilleur *hit* après une recherche de similarité, à partir d'outil comme BLAST, dans l'entièreté des gènes de chaque génome. Cette méthode a plusieurs limites. Tout d'abord, elle ne permet que de reconstruire des *groupes d'orthologues apparents*, ignore les autres types de relations et bien souvent construit des familles de gènes incomplètes. Par ailleurs, les scores de similarité BLAST, représentent généralement une mauvaise estimation de la vraie distance évolutive entre séquences [185]. La plupart des algorithmes de construction de familles de gènes (COG [9], InParanoid [186], OrthoMCL [187]) étendent donc cette approche à plusieurs génomes et aux gènes similaires provenant du même génome en construisant des réseaux de relations comprenant orthologues et paralogues récents, où chaque arête du graphe obtenu est pondéré par le score de similarité séparant la paire de séquence jointe (Figure 3.6). Ces techniques diffèrent essentiellement par l'algorithme utilisé pour former des groupes de gènes homologues à partir du graphe de similarité. Elles souffrent souvent d'un manque de sensibilité et bien qu'assez précises pour identifier les homologues, n'offrent aucune garanti en ce qui concerne l'exactitude des relations entre gènes inférées.

Une alternative à ces méthodes, légèrement plus sensible, et fréquemment utilisée lorsque la famille d'intérêt est déjà bien caractérisée dans quelques organismes modèles, consiste à construire tout d'abord un modèle de Markov caché sur les séquences, puis à utiliser ce modèle pour rechercher de nouveaux gènes homologues.

Soulignons aussi l'existence de méthodes basées sur les arbres tels que PANTHER [188], Ensembl Compara [189], et metaPhOrs [190] qui construisent un arbre phylogénétique sur un groupe de gènes homologues puis utilisent la réconciliation (voir section 3.3), ou des

algorithmes similaires, afin de distinguer entre gènes orthologues, paralogues et xénologues. Le fonctionnement de ces méthodes nécessite toutefois la construction préalable d'une famille de gènes homologues à partir des méthodes évoquées plus haut.

3.1.4.2. *Alignements multiples de séquences*

Les gènes homologues ont souvent des longueurs différentes en raison des insertions et délétions de nucléotides pouvant aussi affecter la longueur des séquences protéiques. L'alignement de séquences a deux objectifs principaux. En premier lieu, il sert à identifier les régions hautement similaires en maximisant l'identité ou la similarité biochimique des résidus à chaque position le long des séquences. Il permet également d'organiser ces régions en sites homologues comparables. Par soucis de concision, nous supposerons que chaque résidu de la séquence ancestrale évolue indépendamment des autres, et a donc une descendance dans les séquences observées (sauf si perdu).

Aligner des séquences revient donc à aligner les sites homologues au sein de ces séquences. Pour ce faire, on insère des trous (ou *gaps*) à la place des résidus lorsqu'un site homologue est manquant dans chaque séquence considérée. Plus formellement, étant donné un ensemble de séquence $\mathcal{S} = \{S_1, S_2, ...S_k\}$, un alignement multiple A de S est un ensemble de séquences $A = \{A_1, A_2, ...A_k\}$ ayant la même longueur et obtenues en insérant des gaps dans chacune des séquences S_i, de façon à optimiser un certain coût d'alignement. En principe, le *sum-of-pairs*, c.-à-d. la somme du score induit par l'alignement entre chaque paire de séquences est le critère d'optimisation utilisé. Il est toutefois très difficile d'obtenir l'alignement multiple optimal, étant donné que le problème est NP-complet [191]. Les programmes d'alignements les plus courants (Clustal [192], Muscle [193], MAFFT [194], etc.) utilisent généralement des heuristiques qui garantissent une solution souvent satisfaisante, mais pas forcément optimale.

3.1.4.3. *Inférence d'arbres d'espèces*

Bien que cette thèse porte sur la reconstruction de l'histoire évolutive des arbres de gènes, comme nous le verrons dans la section 3, cette reconstruction nécessite un arbre d'espèces. Ci-dessous, nous présentons sommairement les deux stratégies communément utilisées pour reconstruire les arbres d'espèces (voir aussi la Figure 3.5). Nous référons le lecteur aux revues de Warnow [181, 195] et de De Queiroz et Gatesy [196] pour plus de détails sur ces stratégies.

Les arbres d'espèces peuvent être inférés à partir de séquences de marqueurs génétiques uniques tels que l'ARN ribosomique. Cependant, les méthodes les plus récentes privilégient l'utilisation de l'information provenant de plusieurs locus (familles de gènes). On peut ainsi obtenir une meilleure estimation de la phylogénie des espèces.

La méthode super-matrice construit les arbres d'espèces à partir d'une concaténation des séquences alignées de familles de gènes [196]. Ces familles sont préalablement filtrées pour ne conserver que les gènes qui sont orthologues sans ambiguïtés. Pour ce faire, on se restreint souvent aux familles ne contenant que des gènes n'ayant qu'une seule copie. L'approche super-matrice ignore toutefois les différences évolutives entre les familles de gènes. Cependant, les progrès récents permettent aujourd'hui de considérer des modèles de substitution différents pour chaque famille (voir Figure 3.5).

En revanche, l'approche super-arbre consiste à construire des arbres de gènes individuels pour chaque famille puis à les combiner en un seul arbre d'espèces [197]. Elle présente l'avantage de permettre la visualisation, lorsqu'ils existent, des conflits au sein de l'ensemble d'arbres. Les critiques de cette méthode incluent le manque de justification statistique au cours de la construction du super-arbre et aussi le manque de considération des incertitudes au niveau des sous-arbres de gènes.

Quelle que soit l'approche utilisée, les arbres d'espèces sont inférés en utilisant des méthodes de construction d'arbres, à partir d'alignements de séquences, que nous décrivons dans la section suivante.

3.2. Méthodes phylogénétiques classiques de construction d'arbres

Dans cette section, nous survolons les méthodes génériques d'inférence d'arbres phylogénétiques. Ces méthodes permettent d'inférer des arbres à partir de n'importe quelle source de données que l'on peut organiser en une matrice A de taille $m \times n$ où m désigne le nombre d'entités et n le nombre de sites, pourvu qu'il existe un modèle évolutif correspondant. Elles peuvent donc servir à la construction d'arbres phylogénétiques d'espèces ou de gènes. Dans le cadre de cette thèse, A fait référence à l'alignement d'un ensemble de séquences $\mathcal{S} = \{S_1, S_2, ...S_k\}$. Nous ne décrirons que brièvement ces méthodes et référons le lecteur à l'excellent livre par Felseinstein [173] sur le sujet, et aux revues par Yang et Rannala [198] et Warnow [181].

Les méthodes phylogénétiques classiques d'inférence sont basées soit sur la distance entre séquences, soit sur la similarité de caractères. Les méthodes basées sur la distance calculent une distance évolutive entre chaque paire de séquences homologues, et la matrice de distance ainsi obtenue est utilisée pour construire les arbres grâce à des algorithmes de *clustering*. Les méthodes basées sur la similarité de caractères incluent le maximum de parcimonie (MP), le maximum de vraisemblance (ML pour *Maximum Likelihood*) et les méthodes d'inférence

bayésienne. Ces méthodes comparent simultanément les séquences dans l'alignement multiple, un site à la fois, afin d'inférer l'arbre T dans l'espace possible $\mathcal{T}$ de topologies, tel que T maximise le score cumulé ρ sur l'ensemble des sites. Trouver un tel arbre est un problème NP-difficile, et en raison de la grande taille de $\mathcal{T}$ qui croît de façon exponentielle avec le nombre de séquences, des heuristiques de recherche comme le *branch-and-bound* et le *hill-climbing* doivent être utilisées. Ainsi, un arbre initial T_0 est d'abord construit ; puis l'espace $\mathcal{T}$ est réduit au voisinage de T_0 par des réarrangements locaux de branches (NNI, SPR, TBR, cf. section 3.1.2). Chacune de ces méthodes, qu'elle soit basée sur la distance ou la similarité des caractères, possède ses propres caractéristiques, avantages et inconvénients.

3.2.1. Méthodes basées sur la distance

Toutes les méthodes de construction basées sur la distance génétique débutent par le calcul d'une matrice de distance D dont chaque entrée D_{ij} représente la distance génétique entre les séquences S_i et S_j, calculée sous un modèle évolutif précis. Après le calcul de cette matrice, l'arbre phylogénétique peut être construit selon plusieurs critères. On peut chercher un arbre T qui minimise l'erreur, pour toute les paires de séquences, entre la distance d_{ij} induite par ses branches, et la distance D_{ij} présente au sein de la matrice. Alternativement, il peut s'agir de trouver l'arbre avec le *minimum d'évolution*, c.-à-d., l'arbre dont la somme des longueurs de toutes les branches est la plus faible.

Des algorithmes de regroupement hiérarchique tel que UPGMA (Unweighted Pair Group Method with Arithmetic Mean) et NJ (Neighbor Joining) sont utilisés. Ces algorithmes réduisent considérablement l'espace d'exploration, ce qui rend les méthodes de distances très rapides en comparaison aux autres méthodes d'inférence d'arbres.

Dans le cas du Neighbor-Joining, un arbre non enraciné est construit en joignant les sous-arbres les plus proches entre eux, tout en étant les plus éloignés possible des autres. Il s'agit de la méthode de distance la plus utilisée. Elle est idéale pour analyser de grands ensembles de données, mais devient peu fiable lorsque les séquences utilisées sont très divergentes [198]. Comme les autres méthodes de distance, son inconvénient majeur provient du fait que la conversion de l'alignement multiple en matrice de distance entraîne une perte d'information sur l'évolution des séquences [199].

NJ et plusieurs de ses variantes améliorant la vitesse d'exécution sont implémentées dans des outils comme MEGA [200] et BIONJ [201].

3.2.2. Méthodes basées sur la parcimonie

Les méthodes basées sur le maximum de parcimonie ont pour but de trouver l'arbre qui minimise le nombre d'évènements évolutifs (substitution de résidus dans notre cas) nécessaires pour expliquer la matrice A. Parmi les méthodes énumérées, il s'agit de la seule à ne pas utiliser un modèle explicite de substitutions. Pour chacune des topologies $T_k \in \mathcal{T}$, un score $\rho_i^{T_k}$ est calculé à chaque site i de l'alignement. L'arbre maximisant la parcimonie est celui avec le meilleur score $\rho = \min_{T_k \in \mathcal{T}} \sum_i \rho_i^{T_k}$. Fitch a proposé en 1971 un algorithme pour reconstruire les états ancestraux à chaque noeud interne, permettant ainsi de calculer le nombre minimal de substitutions pour chaque site sur un arbre binaire en $O(m|\Xi|)$, où Ξ désigne l'alphabet [202]. À la place de compter le nombre de substitutions, on pourrait attribuer un coût aux substitutions et chercher à minimiser le coût total pour un site. Dans ce cas, l'algorithme de Sankoff-Rousseau permet de trouver le coût optimal de l'arbre en temps $O(m|\Xi|^2)$ [203].

L'inconvénient majeur du maximum de parcimonie est l'absence complète d'un modèle évolutif des données [198]. La parcimonie n'est pas non plus statistiquement consistante, c'est-à-dire que l'ajout de données ne garantit pas de converger vers le vrai arbre. Felsenstein montrait également en 1978 que la parcimonie souffre énormément du problème d'*attraction des longues branches* [16]. En effet, en raison d'une convergence évolutive non liée à une relation de parenté (homoplasie), plusieurs séquences peuvent paraître très similaires. Les arbres phylogénétiques inférés par MP auront tendance à systématiquement regrouper ensemble ces séquences.

Le maximum de parcimonie est implémenté dans plusieurs programmes phylogénétiques (MEGA, PHYLIP [204], PAUP* [205], TNT [206] ...), mais est rarement utilisé de nos jours pour les inférences d'arbres à partir de séquences biologiques.

3.2.3. Modèle d'évolution de séquences

Les méthodes de distances et les méthodes probabilistes (ML et inférence bayésienne) utilisent un modèle explicite d'évolution qui spécifie comment les résidus (nucléotides ou acides aminés) mutent. Les modèles évolutifs se limitent très souvent aux substitutions, et ignorent les indels (insertion-délétion) dans les alignements de séquences. Il existe quand même des modèles propres aux indels, comme le modèle TKF91 proposé par Thorne, Kishino et Felsenstein [207].

Les modèles d'évolution d'ADN modélisent les substitutions nucléotidiques sous forme de chaîne de Markov, où chaque état est représenté par un nucléotide et il existe des probabilités

de transition d'un nucléotide à un autre, déterminées par leur fréquence stationnaire et par des taux de substitution. Les différences entre modèles évolutifs de nucléotides surviennent au niveau des hypothèses faites par rapport à la fréquence des résidus et au taux de substitution. Le modèle de Jukes-Cantor ou JC69 [208] présume des changements équiprobables et la même fréquence pour tous les nucléotides. Le modèle K80 [209], pénalise différemment les transversions (substitutions entre une purine et une pyrimidine) et les transitions (substitutions entre deux purines ou entre deux pyrimidines). Enfin, dans le modèle GTR [210], les fréquences et les taux de substitutions sont libres de varier. Il s'agit du modèle le plus général parmi les modèles considérés comme étant réversibles dans le temps, *c.-à.-d.* ceux supposant que les taux dans deux sens de substitutions, pour une paire de nucléotides, sont équivalents. Ces modèles sont parfois étendus en supposant une variation, modélisée par une distribution *gamma*, des taux de substitutions d'un site à un autre. On parle alors de JC+Gamma, K80+Gamma ou GTR+Gamma.

Les modèles protéiques de substitutions sont souvent représentés par des matrices de substitution déduites d'alignements de plusieurs familles de protéines. Le modèle de substitution le plus connu est celui de Dayhoff [4]. Jones, Taylor, et Thornton ont développé en 1992 une procédure automatisée basée sur le modèle de Dayhoff pour déterminer, sur de grandes bases de données d'alignements, des matrices de substitutions nommées JTT [211]. Bien qu'améliorant grandement le modèle de Dayhoff, le modèle JTT n'empêche pas de sous-estimer le nombre de mutations et suppose aussi une homogénéité des taux de substitution le long des séquences. Lartillot et Philippe proposèrent ainsi en 2004 le modèle CAT, un modèle hétérogène qui admet différentes matrices de substitution d'un site à l'autre [212]. Le nombre de profils de substitutions devient donc un paramètre du modèle et est estimé en même temps que tous les autres paramètres avec un *prior* tiré d'une loi de Dirichlet.

3.2.4. Méthodes probabilistes

Maximum de vraisemblance

Le maximum de vraisemblance est une technique utilisée en statistique pour déterminer les paramètres inconnus d'un modèle, à partir d'un ensemble de données. La fonction de vraisemblance est une probabilité d'observer les données, étant donné un ensemble de paramètres du modèle. La technique est née des travaux de *Fisher* dans les années 1920 [213], mais son application aux reconstructions phylogénétiques est plus récente.

En phylogénie, les données sont représentées par l'alignement multiple A de séquences et le but est donc de déterminer les paramètres d'un modèle spécifique (topologie T de l'arbre, longueur l des branches et paramètres θ du modèle de substitution des nucléotides/acides

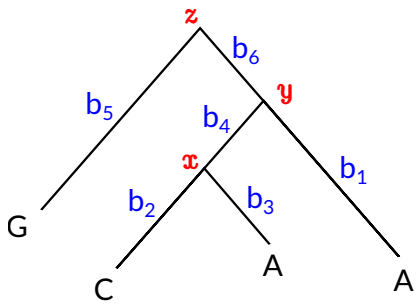


FIGURE 3.7. Calcul de la vraisemblance d'un arbre pour un seul site. Les étiquettes aux feuilles correspondent aux nucléotides du site dans chacune des séquences. Les branches sont numérotées et indiquées en bleue, tandis que les états potentiels des trois noeuds internes sont en rouges.

aminés) qui maximisent la probabilité $P(A|T,\theta,l)$ d'observer A. Cette probabilité ou vraisemblance est estimée à chaque site de l'alignement. Elle se calcule sous forme du produit de la vraisemblance à tous les sites A_i de l'alignement, ce qui est rendu possible par le fait que nous supposons une évolution indépendante des sites :

$$L(T,\theta,l) = \prod_{i}^{n} P(A_i|T,\theta,l)$$

Pour calculer cette vraisemblance, il faut donc savoir calculer $P(A_i|T,\theta,l)$ à chaque site. Ce terme nécessite toutefois une intégration sur tous les noeuds internes de l'arbre, en considérant toutes les combinaisons d'états des sites ancestraux.

Par exemple, pour l'arbre de la Figure 3.7 montrant l'assignation d'une position de l'alignement aux feuilles, lorsqu'on suppose une indépendance des branches et un alphabet de nucléotides Ξ, on a :

$$\begin{aligned} L_0 &= \sum_{z\in\Xi}\sum_{y\in\Xi}\sum_{x\in\Xi} P(A,A,C,G,x,y,z|T) \\ &= \sum_{z\in\Xi}\sum_{y\in\Xi}\sum_{x\in\Xi} P(z)P(y|z,b_6)P(A|y,b_1)P(x|y,b_4)P(A|x,b_3)P(C|x,b_2)P(G|z,b_5) \end{aligned} \tag{3.2.1}$$

Le calcul de cette équation bien que facile peut être coûteux en temps, d'autant plus que le nombre de termes augmente exponentiellement avec le nombre de séquences. De plus, cela doit se faire pour tous les sites de l'alignement, et ce, pour toutes les topologies d'arbres dans $\mathcal{T}$.

L'algorithme de "pruning", par programmation dynamique, décrit par Felsenstein [214] en 1981, permet néanmoins de calculer Eq. 3.2.1 en temps cubique, assurant ainsi un calcul

de la vraisemblance en temps raisonnable pour les ordinateurs modernes. Le maximum de vraisemblance bénéficie du fait que l'enracinement de l'arbre n'a pas d'impact sur la valeur de l'Eq. 3.2.1. On peut donc enraciner arbitrairement l'arbre sur une branche avant le calcul de $L(T, \theta, l)$.

La reconstruction par maximum de vraisemblance procède en estimant les paramètres du modèle et en explorant des topologies de $\mathcal{T}$ jusqu'à ce qu'une convergence soit atteinte. La méthode a très tôt été implémentée dans les programmes phylogénétiques comme MOLPHY [215] et PHYLIP [204]. Une nouvelle génération d'implémentations modernes, plus précises et surtout beaucoup plus rapides a vu le jour avec le développement de programmes comme PhyML [216], RAxML [217], FastTree [218] et IQ-TREE [219].

D'un point de vue théorique, les méthodes ML sont mieux justifiées que celles basées sur la distance et la parcimonie, comme le montrent d'ailleurs les résultats de plusieurs simulations [220–222]. Aussi, contrairement à la parcimonie, le maximum de vraisemblance est en général statistiquement consistant, et converge vers le vrai arbre, sous le bon modèle évolutif, si l'on dispose de suffisamment de données [223, 224]. Soulignons qu'en présence de certains phénomènes bologiques comme le tri incomplet des lignées ancestrales, le maximum de vraisemblance peut ne pas être statistiquement consistant [225].

Inférence bayésienne

Les inférences bayésiennes sont basées sur le principe de Bayes :

$$P(T, \theta, l|A) = \frac{P(T, \theta, l)P(A|T,\theta, l)}{P(A)} \tag{3.2.2}$$

Elles nécessitent la connaissance d'une probabilité *a priori* $P(T, \theta, l)$, et la vraisemblance $P(A|T,\theta, l)$ afin de produire une probabilité *a posteriori* $P(T,\theta, l|A)$ désignant la probabilité que le modèle choisi, avec les paramètres adéquats (topologie de l'arbre, longueur des branches, etc.) soit celui ayant généré les données. Les paramètres du modèle ne sont plus considérés comme des variables fixes, mais plutôt comme des variables aléatoires associées à une distribution statistique. Alors que le maximum de vraisemblance cherche l'arbre phylogénétique expliquant le mieux les séquences observées, l'inférence bayésienne examine la densité de probabilité des phylogénies pour trouver un arbre maximisant la probabilité de générer l'alignement. Elle répond ainsi directement à la question biologique de savoir quel est l'arbre le plus évident.

Le terme de normalisation $P(A)$ dans Eq. 3.2.2 nécessite une intégration sur toutes les phylogénie et est donc très difficile à calculer. Grâce à l'introduction de méthodes numériques

d'échantillonnage de la densité *a posteriori* comme la méthode de Monte-Carlo par Chaîne de Markov (MCMC), ce calcul est possible, même s'il reste coûteux en temps [226].

Une conséquence du principe de fonctionnement des méthodes d'inférences bayésiennes est qu'elles n'infèrent pas un seul arbre, mais plutôt une distribution d'arbres probables qu'il faut ensuite *amalgamer* en un seul arbre. Très souvent, on construit simplement un arbre consensus basé sur l'une des variations de la règle de la majorité. En d'autres termes, on retient dans l'arbre final les clades majoritairement observés dans l'échantillon d'arbres inférés.

Il existe plusieurs programmes d'inférence bayésienne, les plus connus étant MrBayes [227], Beast [228] et PhyloBayes [229].

Tout comme la méthode de vraisemblance, les inférences bayésiennes sont statistiquement consistantes et très efficaces. Leur efficacité dépend cependant du modèle *a priori* utilisé, qui est souvent difficile à correctement déterminer pour la plupart des données. Par ailleurs, il arrive parfois que les probabilités *a posteriori* retournées par ces méthodes soient très élevées pour de mauvaises phylogénies [230, 231].

3.2.5. Erreurs au sein des arbres de gènes

Les algorithmes d'inférence classiques décrits plus haut ne sont pas du tout exempts d'erreurs de reconstruction. Ces erreurs peuvent se manifester sous forme d'artefacts tels que l'attraction des longues branches (LBA pour *long branch attraction*). Le LBA consiste en un groupement des lignées ayant des branches plus longues que les autres, peu importe la vraie relation phylogénétique les reliant [16].

Ci-dessous, nous énumérons quelques sources d'erreurs au sein des inférences phylogénétiques :

Erreurs intrinsèques aux séquences : En présence d'erreurs de séquençage et/ou d'annotations (incluant une mauvaise traduction de séquences nucléotidiques en séquences protéiques), les arbres inférés sur ces données ne refléteront pas la vraie histoire évolutive. Le même problème survient en raison de signaux non phylogénétiques au sein des séquences. Par exemple, l'existence d'un *biais de composition* en nucléotides (ou acides aminés) aura pour conséquence un regroupement des séquences partageant des compositions similaires, puisque la plupart des modèles supposent que la composition des séquences reste stationnaire au cours du temps. De la même façon, les sites à l'intérieur des séquences sont souvent soumis à des vitesses d'évolution différentes en raison d'une différence de pression évolutive et/ou

fonctionnelle, et ces vitesses peuvent varier à travers le temps. La précision des méthodes d'inférences phylogénétiques est donc affectée lorsqu'on ne tient pas compte de ces aspects.

De plus, l'homoplasie qui correspond à une similarité de caractères causée non pas par une évolution commune, mais plutôt par la convergence ou par la réversion (retour d'un caractère à son état primitif) peut aussi introduire ces signaux non phylogénétiques. En effet, derrière la similarité entre deux séquences, peut se cacher une distance évolutive beaucoup plus grande qu'en apparence.

Une autre source considérable d'erreurs est celle provenant des alignements de séquences. Ces erreurs sont souvent causées par un mauvais placement des gaps et ont pour conséquence d'aligner des sites non homologues, introduisant ainsi des erreurs dans les inférences.

Enfin, il se peut que les séquences ne permettent tout simplement pas une bonne différentiation des lignées, parce que les signaux phylogénétiques s'y trouvant sont trop faibles. Ce problème peut survenir lorsque les séquences sont trop similaires ou lorsqu'il existe une *saturation phylogénétique* c.-à-d. que les séquences homologues ont subi trop de mutations depuis leur divergence, de façon à ce qu'il soit impossible de reconstruire avec précision leur histoire phylogénétique, quelle que soit la méthode employée. Dans le cas précis des arbres de gènes, des erreurs stochastiques [232] supplémentaires interviennent, car le nombre limité de sites dans un seul gène ne fournit pas assez d'informations pour une bonne résolution des phylogénies.

Mauvais regroupement des gènes en familles : Un mauvais regroupement des familles de gènes peut soit résulter en un problème de données manquantes, soit grouper des gènes ne partageant aucune origine commune. Dans le premier cas, plusieurs études ont montré que les données manquantes diminuent la puissance statistique des méthodes d'inférence, et augmentent le risque d'artefacts de reconstruction [233–236]. Inversement, la présence de gènes non homologues aura un impact similaire, en induisant des erreurs topologiques.

Modèle d'évolution inadéquat : Bien que les méthodes d'inférence couramment utilisées soient statistiquement consistantes, elles restent extrêmement sensibles aux déviations de modèles, et convergent dans ce cas vers des phylogénies erronées [221, 237, 238]. L'exactitude des inférences phylogénétiques dépend donc du choix du modèle d'évolution pour les données dont on dispose. Lorsque le modèle choisi est inadéquat ou trop simple pour capturer la complexité biologique, la résolution des arbres est moins bonne, et des erreurs de topologie apparaissent [239]. En pratique, les hypothèses considérées sont souvent enfreintes, parce que les modèles actuels ne permettent pas encore de modéliser l'étendue de la complexité

de l'évolution. Il est toutefois possible de choisir le modèle le plus approprié en comparant la vraisemblance sous une variété de modèles avec un test du rapport de vraisemblance. Des méthodes telles que ModelTest [240] existent à cet effet.

Limites de l'exploration de l'espace de solutions : Puisque les méthodes de reconstructions utilisent souvent des heuristiques pour explorer l'espace des phylogénies, il n'y a aucune garantie que l'entièreté de cet espace soit explorée, et par conséquent, l'arbre retourné pourrait ne correspondre qu'à un optimum local.

3.2.6. Support statistique sur les arbres

Les supports statistiques constituent un ensemble de métriques pour évaluer les incertitudes au sein d'un arbre. Elles peuvent aussi être utilisées pour choisir une topologie d'arbre plutôt qu'une autre. Ces supports ne permettent toutefois que de juger de la fiabilité des arbres en fonction du modèle choisi et des données disponibles. Ainsi, un support élevé ne veut pas forcément dire que l'arbre ou le clade correspondant est correct. La plupart des tests statistiques sont basés sur le score de vraisemblance des arbres.

3.2.6.1. *Support sur les branches*

Lorsqu'il s'agit d'obtenir un support statistique pour la topologie d'un arbre, le test non paramétrique de bootstrap peut être utilisé [172, 241]. Le bootstrap est un ré-échantillonnage des données par tirage avec remise. On choisi donc aléatoirement des sites (colonnes) de l'alignement initial pour construire un nouvel alignement. Une réplique bootstrap est un arbre reconstruit à partir de ce nouvel alignement. Ce processus est répété plusieurs fois (en général 100 ou 1000 fois). L'arbre inféré est ensuite comparé aux répliques bootstrap pour déterminer le support bootstrap de chaque branche, c.-à-d. leur niveau de conservation dans les répliques. Il s'agit donc d'un test pour déterminer la robustesse des clades d'un arbre et non pour vérifier si ce dernier est correct. Une valeur de bootstrap $B > 95\%$ sur une branche indique souvent un niveau de confiance élevé pour le clade sous cette branche. Une alternative plus rapide au bootstrap est le aLRT (Approximate Likelihood-Ratio Test for Branches) [242] récemment implémenté dans PhyML. Pour les arbres obtenus par inférence bayésienne, la probabilité a posteriori de chaque sous-arbre (proportionnelle au nombre de topologies acceptées dans $\mathcal{T}$ qui présentent ce sous-arbre) sert de support.

3.2.6.2. *Tests statistiques pour la comparaison d'arbres phylogénétiques*

Pour comparer des arbres phylogénétiques, les tests non paramétriques les plus utilisés sont les tests KH [243], SH [244] et AU [245]. Ces tests utilisent une procédure de ré-échantillonnage basée sur le bootstrap pour évaluer le support statistique des arbres. Ils permettent de déterminer, sachant l'information des séquences, si des topologies différentes d'arbres sont statistiquement équivalentes ou si l'une des topologies est significativement meilleure. Nous mentionnons ces tests parce qu'ils servent à s'assurer que les arbres construits avec les méthodes utilisant des sources additionnelles d'informations, que nous décrivons dans cette thèse, restent optimaux pour le critère des séquences.

Le test KH est approprié pour comparer deux arbres phylogénétiques, mais est biaisé si l'un des arbres est *a priori* l'arbre maximisant la vraisemblance [245, 246]. Le test KH aura alors tendance à rejeter les topologies alternatives, pourtant valables, en faveur de l'arbre ML. Une alternative au KH est le test SH, un test conservateur qui permet la comparaison de plusieurs arbres. Toutefois ce test présente aussi un inconvénient, puisque là où il fonctionne le mieux c'est lorsque toutes les topologies alternatives sont incluses dans l'analyse, ce qui n'est pas possible en pratique. De plus, avec ce test, il est parfois difficile de rejeter l'hypothèse nulle selon laquelle toutes les topologies d'arbres testées expliquent de façons équivalentes les données [247]. Le test non paramétrique le plus précis et le moins biaisé connu actuellement est le test AU. Tous ces tests sont implémentés et disponibles dans le programme consel [248].

3.3. Inférence de l'histoire évolutive des familles de gènes par réconciliation

Dans cette section, nous présenterons les algorithmes d'inférence de l'histoire évolutive des gènes c.-à-d., d'identification des différents évènements macro-évolutifs qui surviennent lors de l'évolution des familles de gènes. La détermination de l'histoire évolutive des familles de gènes est indispensable pour l'inférence des relations d'orthologie, paralogie et xénologie entre gènes. Les algorithmes que nous décrivons sont basés sur la réconciliation de l'arbre des gènes avec celui des espèces.

La réconciliation peut se faire sous une multitude de modèles biologiques différents, les plus communs étant les modèles de Duplication (D), Duplication-Perte (DL pour *Duplication Loss*), Duplication-Transfert-Perte (DTL). D'autres évènements donnant lieu à une phylogénie discordante entre gènes et espèces, tels que le *tri incomplet des lignages* (ILS pour *Incomplete Lineage Sorting*) qui survient lorsque les allèles sont perdus et hérités différemment entre les espèces, ont aussi été considérés, mais ne seront que brièvement mentionnés.

Notre objectif dans cette section n'est pas de couvrir la littérature exhaustive sur la réconciliation (nous référons le lecteur à la revue par Rusin *et al.* [249] sur le sujet), mais uniquement de présenter la théorie pertinente à la compréhension des méthodes de correction/construction d'arbres de gènes, des chapitres 6-7. Puisque nous nous intéressons uniquement aux évènements de duplication, perte, spéciation et HGT (voir Figure 3.8), nous décrirons principalement les modèles DL et DTL.

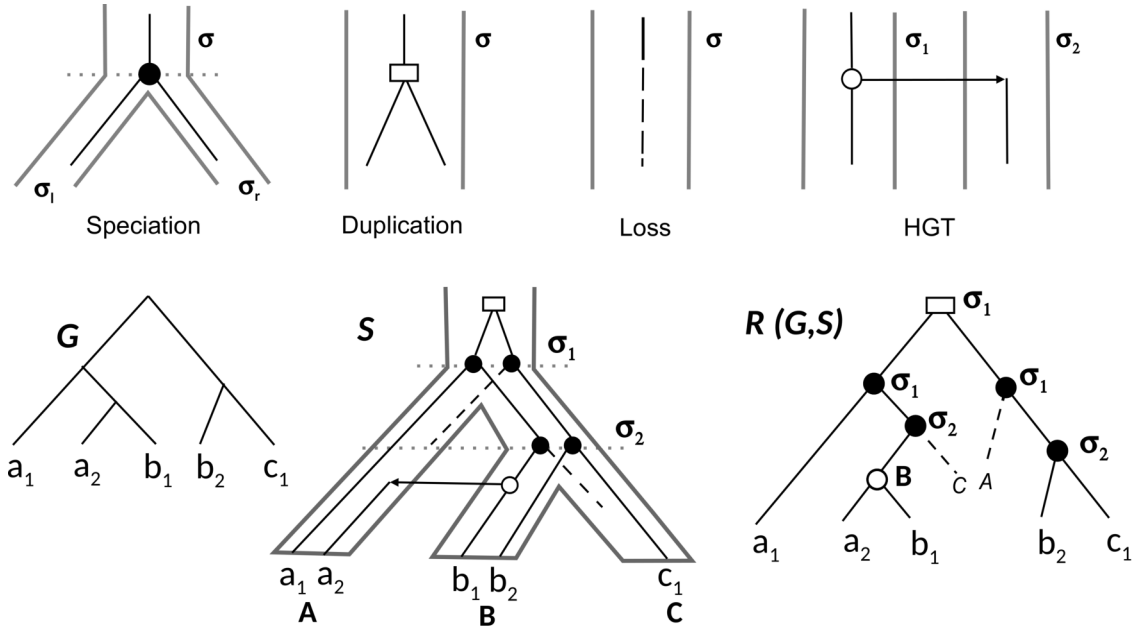


Figure 3.8. Haut : Représentation des évènements de spéciation (cercle noir), duplication (rectangle blanc), perte (ligne en pointillés) et HGT (cercle blanc). Après une spéciation, σ_l et σ_r indiquent les deux espèces descendantes de σ ; pour un HGT, σ_1 désigne le génome source et σ_2 la cible du transfert. σ_1 et σ_2 doivent être incomparables. **Bas** : (gauche) Un arbre de gènes G pour la famille $\Gamma = \{a_1, a_2, b_1, b_2, c_1\}$, où chaque gène en minuscule provient du génome en caractère majuscule. (centre) une histoire évolutive de Γ représentée à l'intérieur de l'arbre des espèces $S = (A, (B,C))$; (droite) la réconciliation $R(G,S)$ correspondant à l'histoire évolutive décrite sur l'arbre au centre. Chaque noeud x de $R(G,S)$ est étiqueté avec le génome $s(x)$ d'où il provient. La branche (B, a_2) est une arête de transfert.

3.3.1. Réconciliation la plus parcimonieuse entre arbres binaires de gènes et d'espèces

Nous appelons $R(G,S)$, ou simplement R, une *réconciliation* d'un arbre de gènes G avec un arbre d'espèce S, une extension de G, dont chaque noeud x est étiqueté par son génome

d'origine $s(x) \in V(S)$, qui reflète une histoire de spéciation et de gain/perte de gènes cohérent avec S (voir Figure 3.8). De plus, des branches de R peuvent aussi être désignées comme *arêtes de transfert* ou *arêtes HGT* lorsqu'elles représentent un transfert horizontal de gène. Plus formellement :

Definition 3.3.1 (Arbre de gènes réconcilié)**.** *Soit G un arbre de gènes binaire et S l'arbre binaire des espèces. Une réconciliation $R(G,S)$ de G avec S est une extension de G telle que pour chaque noeud interne x de $R(G,S)$, l'une des situations suivantes est vraie :*

1. *$s(x_l)$ et $s(x_r)$ sont les deux enfants de $s(x)$; dans ce cas, x est un noeud de spéciation.*
2. *$s(x_l) = s(x_r) = s(x)$; dans ce cas, x est un noeud représentant une duplication dans $s(x)$.*
3. *$s(x_l)$ (resp. $s(x_r)$) est égal à $s(x)$ et $s(x_r)$ (resp. $s(x_l)$) est incomparable à $s(x)$. Soit $y = x_r$ (resp. $y = s_l$) tel que $s(y)$ est incomparable à $s(x)$. Alors x est un noeud représentant un évènement de transfert horizontal d'un gène du génome source $s(x)$ au génome cible $s(y)$, et l'arête (x,y) est une arête HGT.*

Enfin chaque feuille x greffée sur R (donc absente de G) représente une perte dans $s(x)$.

$R(G,S)$ permet de définir explicitement les relations d'orthologie, paralogie et xénologie entre gènes. Deux gènes sont orthologues si leur LCA dans $R(G,S)$ est un noeud de spéciation, paralogues si c'est un noeud de duplication, et enfin, on les qualifiera de xénologues si le LCA est un noeud HGT. Par exemple, sur la Figure 3.8, les gènes b_2, c_1 sont orthologues, b_2, a_1 sont paralogues et a_2, b_1 sont xénologues.

Bien qu'il existe des algorithmes probabilistes de réconciliation, modélisant l'évolution des gènes sous forme d'un processus markovien de naissances et morts qui décrit les gains et pertes de gènes [26, 250, 251] (voir Figure 3.9 et [252] pour une revue), notre attention portera sur les modèles de parcimonie dont le principe général consiste à minimiser un coût d'évènements.

En effet, la plupart des modèles probabilistes requièrent la connaissance des taux des évènements de duplications, pertes et transferts, et doivent souvent explorer l'espace des réconciliations possibles entre arbres de gènes et arbre des espèces. Trouver la réconciliation entre arbre de gènes et arbre d'espèces consiste en effet à chercher, parmi toutes les réconciliations possibles, celle maximisant soit la vraisemblance, soit une probabilité a posteriori.

Les méthodes de parcimonie, aussi appelées *MPR* (pour *Most Parcimonious Reconciliation*), consistent, quant à elles, à prédire l'histoire donnant lieu au moins d'évènements

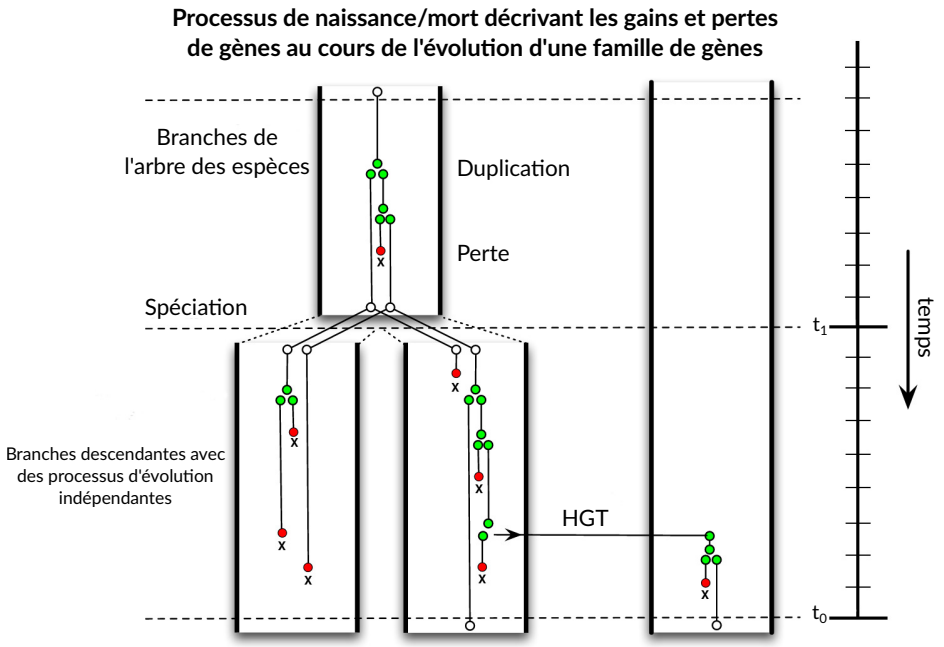


FIGURE 3.9. Processus de mort-naissance modélisant l'évolution des familles de gènes à l'intérieur d'un arbre d'espèce (tube). Les duplications et transferts de gènes sont considérés comme des gains (naissance de nouveaux gènes), alors que la perte est considérée comme la "mort" d'un gène. Ces évènements apparaissent uniquement aux points de discrétisation à des temps précis au cours de l'évolution. La discrétisation de l'arbre des espèces se fait sur les intervalles de temps entre spéciations (t_1 et t_0), subdivisés en des intervalles plus courts. Pour rendre le modèle simple, chaque branche de l'arbre des espèces est considérée comme évoluant différemment. Cette figure est adaptée de la Figure 2 de [252]

possible. Le critère à optimiser est généralement le nombre d'évènements (ignorant la spéciation) que le modèle admet ($\sum_{i\in\{D,L,T\}} n_i$). Par exemple sous le modèle DL, on cherche à minimiser le nombre $n_D + n_L$ de duplications et pertes. Il est également possible de pondérer les opérations en donnant des coûts c_i à chaque évènement. Dans ce cas, on cherche à minimiser la somme pondérée du nombre total d'évènements $\sum_{i\in\{D,L,T\}} c_i \times n_i$.

Bien entendu, la MPR peut-être contestée puisqu'elle ne tient pas compte des réconciliations alternatives et dépend des coûts choisis pour chaque opération. Plus important encore, rien ne garantit que l'évolution de la famille a suivi le chemin évolutif le plus évident (revoir Figure 3.4). Néanmoins, dans [253] et [254], Doyon *et al.* se sont intéressés à l'exploration exhaustive et efficace de l'espace de réconciliation DL. Sur des familles simulées et réelles, ils ont montré que les réconciliations les plus probables se limitaient souvent à un petit sous-ensemble d'histoires incluant presque toujours la MPR.

3.3.1.1. *Réconciliation DL*

Dans le cas de la réconciliation en absence de HGT, il a été montré que le *LCA-mapping* peut-être utilisé pour déterminer, en temps linéaire, une réconciliation minimisant le coût de duplications, ou la réconciliation **unique** minimisant le coût de duplications et pertes [8, 255–258].

Le LCA-mapping entre G et S associe à chaque noeud $x \in V(G)$ un génome $s(x) \in V(S)$, tel que $L(S_{s(x)})$ est le plus petit ensemble de génomes comprenant tous les gènes de $L(G_x)$. Plus formellement :

Definition 3.3.2 (LCA-mapping)**.** *Le LCA-mapping est une fonction $s : V(G) \to V(S)$ qui associe à chaque gène x de G un génome $s(x) \in V(S)$ tel que :*

1. *si $x \in L(G)$ alors, $s(x)$ correspond au génome $\sigma(x)$ d'où le gène x provient,*
2. *sinon, $s(x) = lca_S(\{s(y) : y \in L(G_x)\})$*

Pour une paire (G, S), le LCA-mapping est unique et peut servir à déterminer une LCA-réconciliation $R(G,S)$ (voir Figure 3.10) selon les principes suivants :

1. un noeud interne x de G est un noeud de duplication si $s(x_l) = s(x)$ et/ou $s(x_r) = s(x)$, c.-à-d. que x a la même étiquette qu'au moins un de ses enfants (par exemple la racine de G sur la Figure 3.10),
2. sinon x est un noeud de spéciation.

On peut déduire l'ensemble des pertes sur une branche comme étant le nombre minimum de greffes sur G pour avoir une réconciliation $R(G,S)$ vérifiant la Définition 3.3.1. Plus exactement, le nombre total de pertes associées à une branche (x,y) s'obtient exactement par $\delta(s(y)) - \delta(s(x)) - 1 + estDUP(x)$, où la fonction $estDUP(x)$ retourne 1 si x est un noeud de duplication et 0 sinon.

Soit x un noeud de duplication inféré par la LCA-réconciliation. Nous tenons à distinguer deux types de noeuds associés à une duplication. Si $L(G_{x_l}) \cap L(G_{x_r}) = \emptyset$ alors on dit que x est une *duplication non apparente* ou NAD, puisque la duplication ne se voit pas immédiatement car aucune répétition de copies de gènes n'est observée dans les génomes. L'évènement de duplication est donc uniquement inféré à partir de la différence topologique entre G et S. Dans le cas contraire, on dit que x est une duplication apparente. Par exemple, sur la Figure 3.10, la duplication la plus basse est un NAD alors que celle à la racine est une duplication apparente.

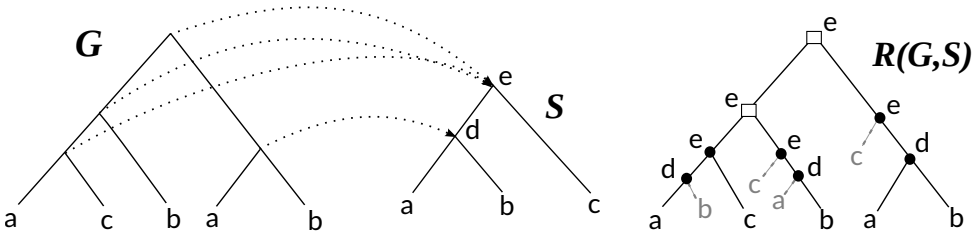


FIGURE 3.10. LCA-réconciliation entre G et S. Les feuilles de l'arbre de gènes représentent directement les génomes d'où ils proviennent. Les noeuds de l'arbre S des espèces sont étiquetés par les espèces ancestrales et le *LCA-mapping* entre chaque noeud interne de G et un noeud de S est indiqué par une flèche en pointillée. Nous montrons ensuite l'arbre réconcilié $R(G, S)$ avec chaque noeud étiqueté par le lca-mapping et les noeuds de duplications indiqués par des rectangles, tandis que les noeuds de spéciation sont montrés avec des cercles noirs. La perte induite sur chaque branche par la réconciliation est indiquée par une feuille en gris. Cette réconciliation optimale coûte 2 duplications et 4 pertes.

3.3.1.2. *Réconciliation DTL*

Lorsque les transferts horizontaux sont admis dans le modèle d'évolution, la réconciliation la plus parcimonieuse n'est pas unique et la LCA-réconciliation ne permet pas d'inférer l'histoire optimale (voir Figure 3.11). En effet, sous le modèle DTL, l'histoire évolutive d'un gène n'est plus restreinte aux branches de ces ancêtres dans G, puisqu'il existe maintenant une transmission horizontale d'information, en opposition à l'héritage strictement vertical que nous avions considéré précédemment. Comme l'énonce la définition 3.3.1, il faut en plus considérer les arêtes de transfert entre paires de génomes incomparables dans $V(S)$.

Nous avons toutefois des restrictions biologiques supplémentaires parce qu'un transfert ne peut se faire que si les génomes source et cible impliqués ont coexisté au moment du transfert. On parle alors de *consistance du scénario HGT*.

Dans [259], Tofigh *et al.* ont montré qu'un scénario sera consistant seulement si la réconciliation DTL correspondante est *acyclique*. Ils définissent l'acyclicité comme suit :

Definition 3.3.3. *Une DTL-réconciliation $R(G,S)$ est acyclique si et seulement s'il existe un ordre total $<_{V(S)}$ sur $V(S)$ tel que :*

1. *pour toute arête $(\sigma, \sigma') \in E(S)$, $\sigma < \sigma'$ et*
2. *si (x,y) et (x', y') sont des arêtes de transfert dans G, et y' est un descendant de y dans $R(G,S)$ alors $p(s(x)) <_{V(S)} s(y')$.*

Nous ne donnerons pas la preuve formelle de la Définition 3.3.3, mais elle est assez intuitive et s'illustre facilement en définissant un ordre temporel sur les génomes.

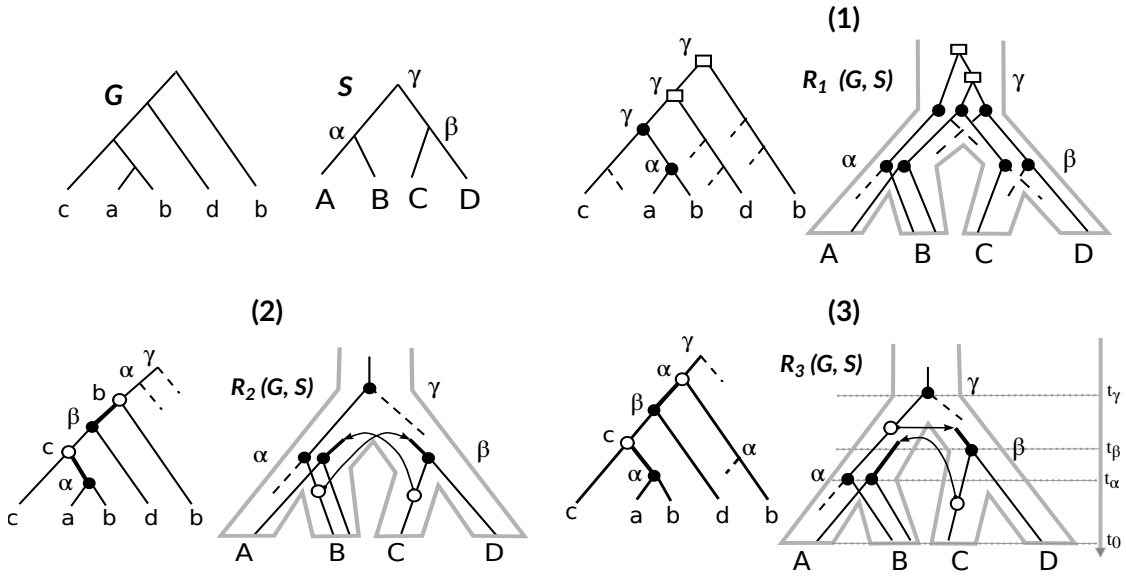
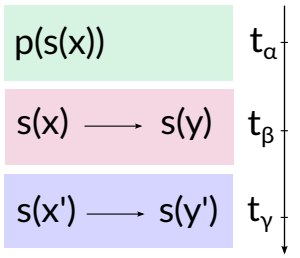


Figure 3.11. Trois réconciliations différentes entre l'arbre de gènes G sur la famille $\Gamma = \{a,b,b,c,d\}$ et l'arbre des espèces S. Chaque gène en minuscule de Γ provient du génome dans $L(S)$ en majuscule. (1) Une LCA-réconciliation optimale pour les coûts D et DL (2 duplications et 5 pertes). (2) Un scénario DTL avec deux HGT et deux pertes. Ce scénario est cyclique est n'est donc pas impossible à réaliser. (3) Un autre scénario DTL avec 2 HGT et une perte, cette fois-ci acyclique, mais tout de même impossible à réaliser, car ne respecte par les temps de spéciations définis.

Si (x,y) est une arête de transfert alors $s(x)$ et $s(y)$ doivent être contemporaines et donc appartiennent à la même période de temps t_β. De la même façon, pour l'arête (x', y'), $s(x')$ et $s(y')$ appartiennent à la même ère t_γ. Puisque y' est un descendant de y dans R alors t_β a lieu avant ou en même temps que t_γ, soit $t_\beta \geq t_\gamma$. Soit t_α l'ère du génome parent $p(s(x))$ de $s(x)$, nous avons $t_\alpha < t_\beta$ et $t_\beta \geq t_\gamma$ alors on doit avoir $t_\alpha < t_\gamma$ pour ne pas avoir une histoire cyclique.



À titre d'exemple, le scénario R_2 de la Figure 3.11 est cyclique, car en appliquant la Définition 3.3.3 sur les deux arêtes de transferts, on aboutit à une contradiction ($\alpha < \alpha$). Le scénario R_3 quant à lui est acyclique.

Trouver une réconciliation DTL acyclique est un problème NP-difficile lorsque l'arbre des espèces dont on dispose n'est pas daté [260–263]. Toutefois, lorsque la contrainte d'acyclicité est relaxée, le problème peut-être résolu en temps polynomial [259, 264]. Dans ce dernier cas, le calcul de la MPR-DTL, se fait en considérant, lors d'un parcours post-fixe de G, toutes les associations possibles entre les noeuds internes de G et chaque noeud de S, afin de trouver la combinaison garantissant le coût minimum à la racine $r(G)$. L'algorithme est basé sur la programmation dynamique.

Plus précisément soit $c(x,\sigma)$ le coût minimum de la réconciliation entre G_x et S, tel que $s(x) = \sigma \in V(S)$. Nous parcourons G en ordre postfixe pour remplir la matrice de coût c, avec pour cas de base (correspondant aux feuilles $x \in L(G)$) :

$$\textit{pour}\ \ x \in L(G),\ c(x,\sigma) = \begin{cases} 0, & \text{si } \sigma = s(x), \\ +\infty, & \text{sinon.} \end{cases}$$

Si x est un noeud interne, nous devons considérer les trois cas où x est un noeud de spéciation, un noeud de duplication, ou un noeud HGT, avec pour coût respectif $c_S(x,\sigma)$, $c_D(x,\sigma)$ et $c_T(x,\sigma)$. Le coût de réconciliation correspond au minimum entre ces trois coûts, soit $c(x,\sigma) = \min\{c_S(x,\sigma),c_D(x,\sigma),c_T(x,\sigma)\}$. L'algorithme s'arrête à la racine de G et retourne le coût optimal $c(G,S) = \min_{\sigma \in V(S)} c(r(G),\sigma)$ de la réconciliation de G avec S.

Tofigh *et al.* [259] ont décrit les récurrences ci-après pour calculer c_S, c_D et c_T, pour des pondérations unitaires et lorsque les pertes sont ignorées, rendant leur algorithme optimal pour le coût Duplication-Transfert.

$$c_S(x,\sigma) = \begin{cases} \min_{u,v}\{c(x_l,t) + c(x_r,u)\ \} \text{ pour tout } t,u \text{ incomparable entre eux} \\ \qquad lca(t,u) = \sigma, \ \text{ Si } \sigma \text{ est un noeud interne de } S, \\ +\infty, \qquad\qquad\qquad\qquad\qquad\qquad \text{sinon.} \end{cases}$$

$$c_D(x,\sigma) = \min_{u,v}\{1 + c(x_l,t) + c(x_r,u)\ \} \text{ pour tout descendant } t,u \text{ de } \sigma \text{ in } S$$

$$c_T(x,\sigma) = \min_{u,v}\{1 + c(x_l,t) + c(x_r,u)\ \} \text{ pour tout } \ t \text{ descendant de } \ \sigma \text{ in } S \\ \text{et } \ u \text{ incomparables à } \sigma$$

L'implémentation de ces récurrences résulte en un algorithme ayant une complexité en temps de $O(|G||S|^2)$.

Il est possible d'adapter la récurrence précédente pour tenir compte des pertes. David et Alm ont ainsi décrit un algorithme pour le coût DTL avec la même complexité [265] ; puis dans un papier plus récent, Bansal *et al.* [264] ont décrit un algorithme implémenté dans RANGER-DTL avec une complexité améliorée $O(|G||S|)$.

Dans le cas où l'on dispose d'informations sur le temps de divergence des espèces, c.-à-d., que S est explicitement daté, les HGT inférés par la réconciliation doivent être compatibles avec les temps de spéciation. Par exemple, le scénario R_3 de la Figure 3.11 ne respecte pas la datation de S. Les auteurs de [264] ont aussi montré qu'une réconciliation DTL respectant la datation de S peut s'obtenir en temps $O(|G||S|\log(|S|))$. Notons toutefois qu'un scénario respectant la datation de S dans lequel tous les HGT sont réalisés entre espèces coexistantes ne garantit pas une cohérence temporelle globale. Ce dernier problème d'une MPR consistante pour le temps, a été considéré par Doyon *et al.* [260] qui ont décrit un algorithme en $O(|G||S|^2)$ qui subdivise l'arbre daté des espèces en tranches de temps (similaire à la subdivision montrée à la Figure 3.9), et explore successivement les HGT selon ces tranches. La méthode ne requiert donc pas une datation précise de l'arbre des espèces, tâche souvent difficile à accomplir [266], mais juste un ordre sur les spéciations. Les autres méthodes décrites restent toutefois pertinentes lorsque cette information n'est pas disponible.

3.3.1.3. *Au-delà des modèles DL et DTL*

Plusieurs variations du problème de réconciliation utilisant d'autres critères d'optimisation existent. Ces variations considèrent soit des types additionnels ou différents d'évènements ou essayent de répondre à une différente formulation du problème.

Dans son article précurseur de 1997, Maddison considérait la réconciliation sous HGT ou en présence d'hybridation et proposait de compter le nombre minimum de mouvements de branches pour convertir S en G tout en préservant les contraintes temporelles de chacun des arbres [267]. Dans le même papier, l'auteur a aussi montré que l'association entre les noeuds de G et de S pourrait être utilisée pour trouver la MPR lorsque les incongruences entre les deux arbres ne sont expliquées que par les ILS. Wu et Zhang [268] ont ensuite montré que la LCA-réconciliation permettait de trouver l'unique MPR dans ce cas.

Lorsque les duplications et pertes de gènes sont considérées en plus des ILS, le problème devient NP-difficile [269]. Dans [270], Rasmussen *et al.* ont proposé un algorithme probabiliste qui utilise un modèle introduisant un arbre intermédiaire appelé *arbre des locus* entre G et S. Dans [271], le même groupe d'auteurs a reformulé le problème en utilisant la parcimonie, et a conçu une heuristique dont la complexité théorique et la précision restent ambiguës.

Dans une série de papiers [272, 273] du groupe de Durand, le problème est reformulé comme celui de la réconciliation minimisant le coût DL/DTL entre un arbre de gènes binaire et un arbre d'espèces non binaire obtenu par contraction des branches courtes, sources reconnues d'ILS. Les auteurs introduisent à cet effet la notion de *duplications conditionnelles*, c.-à-d. des noeuds de duplication dans G qui dépendent de la binarisation de S. Ils proposent

des algorithmes exponentiels pour les coûts DL et DTL. Une limitation de leurs algorithmes est l'absence d'une pénalisation explicite des ILS.

Enfin, Chan *et al.* [274] ont proposé un algorithme soluble à paramètre fixé (FPT pour *Fixed-Parameter Tractable*) qui calcule la MPR ayant un scénario consistant en temps, sous un modèle tenant compte des duplications, pertes, HGT et ILS. À notre connaissance, il s'agit du premier modèle de réconciliation qui considère simultanément ces quatre évènements dans un cadre parcimonieux.

Le développement de méthodes efficaces étendant la MPR pour considérer d'autres types d'évènements ou pour de nouvelles problématiques, reste un domaine activement étudié. Par exemple, Hasić et Tannier ont proposé récemment une méthode de réconciliation tenant compte des conversions de gènes [275]. Dans un travail en cours dans notre laboratoire, El-Mabrouk *et al.* considèrent la problématique de la super-réconciliation, dans laquelle on désire inférer conjointement l'histoire évolutive de plusieurs familles de gènes présentes sur un même block synténique, par duplications et pertes segmentales.

3.3.2. Incertitudes au sein des arbres phylogénétiques et erreurs de réconciliation

Les progrès importants de ces dernières années, surtout en ce qui concerne l'amélioration de la complexité en temps des méthodes, ont permis l'utilisation à grande échelle de la réconciliation pour élucider de façon précise les relations entre gènes d'une même famille. Toutefois, la réconciliation présente certaines limites.

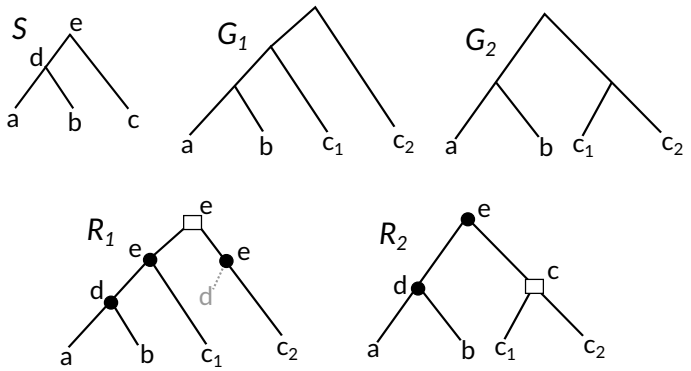


Figure 3.12. Erreur de réconciliation causée par les erreurs dans G. Nous montrons pour deux arbres G_1 et G_2 ayant des topologies presque identiques (seuls leurs enracinements sont différents) que les histoires R_1 et R_2 inférées par réconciliation DL sont totalement divergentes. Par exemple, les gènes a et c_2 sont paralogues dans R_1, alors qu'ils sont orthologues dans R_2.

Même si l'on ignore les problèmes d'inférence liés à l'utilisation de modèles inappropriés, par exemple un modèle DL pour des gènes de bactéries où l'on sait que les HGT sont fréquents, il existe une autre source importante à l'origine des erreurs de réconciliation. En effet, le principe même de la réconciliation la rend extrêmement sensible à la topologie considérée de l'arbre des gènes. Tel que nous l'illustrons sur la Figure 3.12, une différence mineure de la position d'une seule branche dans G peut complètement changer l'histoire inférée par la réconciliation, ainsi que les relations d'orthologie et de paralogie prédites entre gènes.

Par conséquent, la présence d'erreurs d'inférence au sein des arbres de gènes ou d'espèces introduira un biais important dans les inférences d'histoires évolutives par la réconciliation [11]. Comme nous le montrons dans la section 3.2.5, les erreurs au sein des arbres phylogénétiques inférés à partir de l'information des séquences sont variées et fréquentes. Dans le cas des arbres d'espèces, nous pouvons essayer de réduire considérablement ces erreurs en utilisant les approches phylogénomiques (super-matrice et super-arbre) avec des familles de gènes dont l'évolution est stable. Par contre pour les arbres de gènes, le problème est accentué par le fait que les séquences des familles de gènes n'offrent pas suffisamment d'information pour discerner entre topologies alternatives et supporter avec confiance les branches des arbres reconstruits.

Pour ces raisons, le développement de méthodes de réconciliation tenant compte des incertitudes au sein des arbres de gènes est important. L'autre possibilité s'attaquant au même problème, consiste à s'assurer que les arbres de gènes utilisés contiennent le moins d'erreurs possible. Pour ce faire, il faudrait soit développer de nouvelles méthodes de construction plus efficaces et spécifiques aux arbres de gènes, ou développer des méthodes corrigeant de façon minimale la topologie des arbres de gènes afin d'inférer de meilleures réconciliations. Dans la section 3.4, nous discuterons de ces méthodes.

3.4. Nouvelles méthodes intégratives d'inférence d'arbres de gènes

Afin d'améliorer l'inférence des arbres de gènes, des méthodes de reconstruction plus spécifiques aux arbres de gènes ont été développées. Parmi ces méthodes, on peut retrouver celles estimant conjointement alignements et phylogénies traitant ainsi du problème d'interdépendance entre les deux pour limiter les erreurs qui se propagent de l'alignement aux phylogénies [179, 180, 276]. Un inconvénient majeur de ces méthodes est qu'elles nécessitent d'échantillonner à la fois l'espace des alignements et des arbres possibles, ce qui résulte en

une complexité en temps qui les rend souvent inapplicables aux larges données. D'autres méthodes co-estimant arbres de gènes et arbre d'espèces existent [25, 277]. Ces méthodes procèdent souvent de façon itérative en utilisant les arbres de gènes pour raffiner celui des espèces et vice versa.

Les méthodes les plus communes restent toutefois celles qui intègrent des informations additionnelles autres que celle des séquences pour reconstruire les arbres de gènes. Certaines de ces méthodes incorporent l'information sur la structure des macromolécules biologiques [278–280]. D'autres, considèrent plutôt la conservation de l'ordre des gènes [281]. Une dernière catégorie utilise la topologie de l'arbre des espèces pour tenter d'inclure l'information sur l'évolution, à l'échelle macro-évolutive, des familles de gènes.

Nous nous intéressons à cette dernière catégorie de méthodes dont l'efficacité a été maintes fois prouvée [11, 25–27, 282–285]. Plus précisément, nous survolerons dans cette section, *les méthodes intégratives* qui tiennent compte de l'information sur la topologie des arbres d'espèces au moyen de la réconciliation. Il s'agira donc de décrire sommairement, les travaux pré-datant les nôtres (chapitre 6 et 7), ainsi que les alternatives concurrentes qui existent.

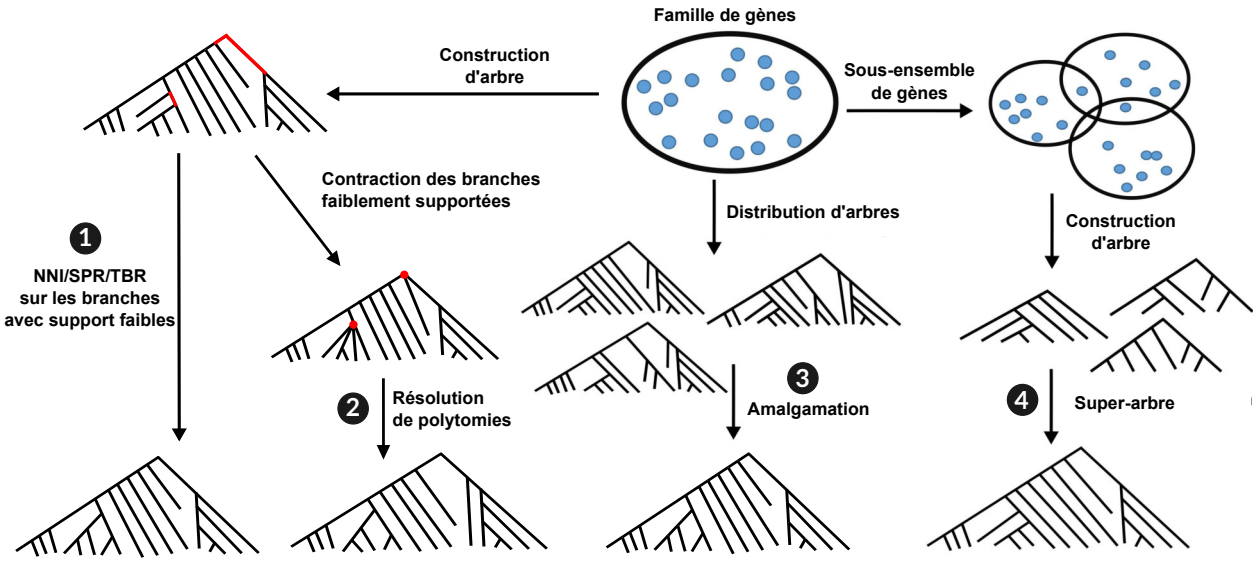


Figure 3.13. Quelques exemples de stratégies utilisées par les méthodes intégratives pour construire/corriger les arbres de gènes.

Les méthodes intégratives utilisent plusieurs stratégies différentes pour construire les arbres de gènes. L'une de ces stratégies consiste à optimiser directement une mesure jointe pour l'information des séquences et celle de la réconciliation. Alternativement, on pourrait procéder en deux étapes (Figure 3.13) :

1. Examiner le voisinage de l'arbre initial, en particulier aux alentours des branches faiblement supportées, afin d'identifier un arbre de moindre coût de réconciliation.
2. Contracter les branches non supportées, puis résoudre les polytomies ainsi obtenues de façon à minimiser le score de réconciliation avec l'arbre des espèces.
3. Construire une amalgamation minimisant le score de réconciliation à partir d'une distribution d'arbres probables pour l'alignement.
4. Construire un super-arbre à partir d'un ensemble d'arbres inférés sur des sous-familles (possiblement chevauchantes) de la famille de gène.

3.4.1. Exploration du voisinage des arbres

Cette stratégie corrige un arbre de gènes G en testant des topologies alternatives obtenues en appliquant sur G des opérations de réarrangement d'arbres, avec pour but d'identifier un arbre G' dans le voisinage de G ayant un meilleur score de réconciliation. Elle fait usage des heuristiques habituelles (*branch-and-bound*, *hill-climbing*, etc.) pour énumérer les arbres candidats G' potentiels, et utilise les algorithmes vus dans la section 3.3 pour calculer le score de réconciliation entre chaque G' et S. La méthode ne garantit toutefois pas qu'une solution optimale sera retournée. En effet, même dans le pire des cas où on explore toutes les topologies alternatives dans le voisinage de G, il reste possible que l'arbre optimal ne se trouve pas dans cet sous-espace restreint.

Cette stratégie a été implémentée dès les premières versions de NOTUNG [286] (parue en 2000) où des opérations NNI sont appliquées sur les branches à supports faibles d'un arbre enraciné G donné en entrée. Ainsi, le programme requiert la présence de supports sur les branches de G. Jusqu'à présent, la correction d'arbres par réarrangement de NOTUNG ne peut se faire que sous le modèle DL de réconciliation.

Dans [287, 288], Górecki et Eulenstein ont considéré le problème pour des arbres de gènes non enracinés. Ils ont décrit une procédure implémentée dans un programme appelé TT, qui permet de trouver un arbre enraciné G' minimisant le score de réconciliation, à partir de l'arbre initial non enraciné G tel que G' se trouve à une distance NNI maximale k de G. Encore une fois, le modèle de réconciliation est limité aux duplications et pertes.

La même stratégie est également utilisée en 2013 dans MowgliNNI [289], mais cette fois-ci, sous un modèle DTL de réconciliation, avec des scénarios temporellement consistants. Les arbres G et S pris en entrées par MowgliNNI doivent être enracinés et S doit obligatoirement être daté.

Enfin, Treefix [284], publié en 2012, utilise la même stratégie, mais en ajoutant dans ses critères la conservation de la vraisemblance de l'arbre. En partant de l'arbre ML G optimal,

l'algorithme optimise le coût de la réconciliation (DL) en explorant, par *hill climbing*, l'espace des arbres statistiquement équivalents à G, selon un test SH. Ainsi on cherche à s'assurer que la vraisemblance des arbres corrigés ne se dégrade pas. Cependant, la convergence de l'algorithme n'est pas garantie, car il peut tomber dans un optimum local. TreeFix est également très coûteux en temps (voir la comparaison, présentée dans le chapitre 6, avec notre méthode ProfileNJ). Dans un autre papier, les auteurs ont également décrit Treefix-DTL [290], une extension de TreeFix qui considère les HGT à travers l'algorithme de réconciliation DTL disponible dans Ranger-DTL [264].

3.4.2. Résolution de polytomies

Avec la stratégie de résolution des polytomies, les branches incertaines d'un arbre de gènes G (déterminées par leurs supports) sont contractées pour produire un arbre non binaire G_{nb}. L'objectif est alors de construire une résolution $G' \in \mathcal{R}(G_{nb})$, qui soit optimale pour le score de réconciliation avec l'arbre d'espèces S.

Un premier algorithme minimisant le coût DL est présenté en 2006 par Durand *et al.* [291] et est implémenté dans NOTUNG. Cet algorithme a une complexité en temps de $O\Big(|S||G|\Delta^3\Big)$ où Δ représente la taille de la plus grande polytomie de G_{nb}. Les auteurs donnent également un algorithme permettant d'énumérer toutes les solutions au problème.

La même année, Chang et Eulenstein [292] ont décrit un algorithme, avec une meilleure complexité, qui fonctionne en $O\Big(|S||G|^2\Big)$. L'un de leurs résultats les plus pertinents était que chaque polytomie de G_{nb} pouvait être considérée indépendamment, permettant ainsi la résolution de G_{nb} par un parcours en profondeur dans lequel chaque polytomie G_x est itérativement résolue.

En 2012, à travers les travaux de Manuel Lafond [293], un premier algorithme linéaire a été développé pour la résolution d'une polytomie P en $O\Big(|P| + |S|\Big)$, aboutissant en une complexité $O\Big(|G||S|\Big)$ pour l'arbre G_{nb}. Cet algorithme permet aussi d'énumérer toutes les solutions.

Un travail effectué par Zheng et Zhang [294] et publié en 2014 a généralisé la complexité linéaire à la résolution entière de l'arbre non binaire, en temps $O(|G|)$. La méthode est toutefois limitée au coût unitaire de duplication et perte, et est seulement capable de retourner une solution optimale. Un résultat important que les auteurs montrent est que pour chaque polytomie P, nous n'avons besoin de considérer qu'un arbre d'espèces réduit S^*_P qui contient le plus petit ensemble de génomes requis pour la résolution de P.

En utilisant la même idée, nous avons développé PolytomySolver [29], une optimisation de la programmation dynamique décrite dans [293] qui permet, non seulement de résoudre

entièrement un arbre non binaire en temps linéaire pour les coûts unitaires, mais peut également retourner toutes les solutions. Lorsque des poids variables et non unitaires sont utilisés, l'algorithme a une complexité de $O\big(|S||G|\big)$. Une particularité de PolytomySolver est qu'il permet également d'affecter individuellement des coûts d'évènements à chaque espèce, ce qui peut s'avérer utile puisque les taux de pertes et duplications de gènes varient énormément entre génomes.

La méthode de PolytomySolver est décrite en détail dans l'article figurant à l'Annexe E avec toutes les optimisations et preuves permettant d'obtenir les complexités indiquées. Ci-dessous, nous résumons son fonctionnement pour déterminer le score de réconciliation entre la meilleure résolution P' d'une polytomie unique P et un arbre S.

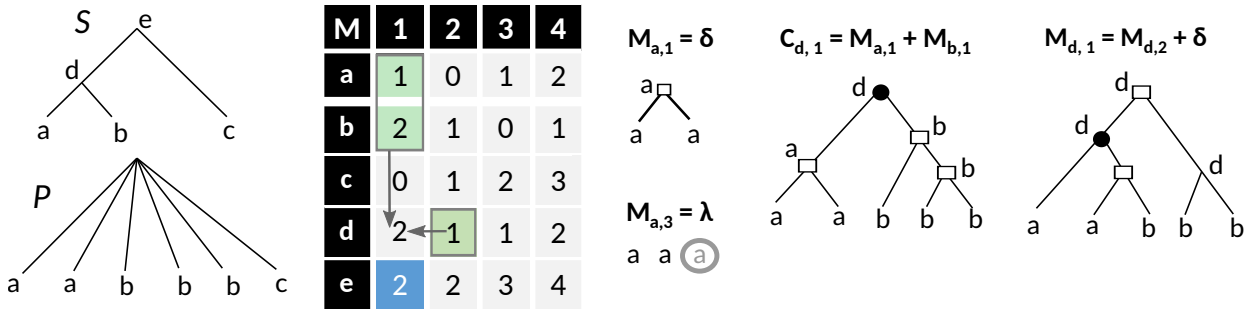


Figure 3.14. Une polytomie P, un arbre d'espèces S et la table M. Les rectangles sur les arbres représentent des duplications et les cercles noirs des spéciations. À droite de M, l'ensemble d'arbres correspondants aux $(a,1)$ et $(a,3)$-résolutions est montré, avec le cercle gris utilisé pour souligner une perte. Nous illustrons la $(d,1)$-résolution enracinée à un noeud de spéciation et correspondant à $C_{d,1} = 3$ (obtenu par la flèche verticale dans M), ainsi qu'une $(d,1)$-résolution plus optimale qui est obtenue à partir de la $(d,2)$-résolution (flèche horizontale dans M). Le coût optimal de la réconciliation entre P et S est indiqué en bleu à la case $M_{e,1} = 2$

L'algorithme procède par récursion sur les sous-arbres de S. Pour une espèce $\sigma \in V(S)$, désignons par $m(\sigma)$ le nombre de gènes dans $L(P)$ qui proviennent de σ. Nous appellerons une (σ, k)-*résolution* de P, un ensemble de k arbres de gènes réconciliés $\mathcal{T} = \{T_1, \ldots, T_k\}$ tel que pour tout $1 \leq i \leq k$, $s(r(T_i)) = \sigma$, et chaque feuille x de P avec $s(x)$ descendant de σ se retrouve dans un des arbres de $\mathcal{T}$ (voir Figure 3.14 pour un exemple). Toute feuille de $\mathcal{T}$ qui n'est pas présente dans $L(P)$ désigne une perte. Le coût de $\mathcal{T}$, que nous dénotons $c(\mathcal{T})$, est la somme du coût de réconciliation de tous les arbres $T_i \in \mathcal{T}$.

Notre objectif est de remplir une table M de taille $|S| \times k$ où $M_{\sigma,k}$ représente le coût minimum d'une (σ, k)-résolution avec $1 \leq k \leq |P| - 1$ (pour tout autre k, $M_{\sigma,k} = +\infty$). Le

score de la réconciliation optimale de la meilleure résolution P' avec S se trouve à la case $M_{r(S),1}$. Il s'agit donc du coût d'avoir un seul arbre enraciné en $r(S)$. La table M est remplie ligne par ligne, dans un parcours en profondeur de S. L'idée principale pour son remplissage suit.

Nous pouvons définir la valeur de la $M_{\sigma,m(\sigma)}$ à 0, car nous n'avons aucune action à effectuer, étant donné que nous disposons déjà de $m(\sigma)$ gènes dans P pour former la $(\sigma, m(\sigma))$-résolution. Par exemple, sur la Figure 3.14, $M_{a,2}$ vaut 0 car deux gènes de a sont trouvés dans P.

Si, par contre, $k \neq m(\sigma)$ alors $M_{\sigma,k,k!=m(\sigma)}$ peut s'obtenir en considérant le score déjà optimal de la case à gauche $M_{\sigma,k-1}$ puis en ajoutant un nouveau gène de σ dans la $(\sigma, k-1)$-résolution, pour signifier une perte de gènes dans σ (voir $M_{a,3}$ sur la Figure 3.14). Alternativement, nous pouvons partir de $M_{\sigma,k+1}$ et joindre ensemble deux arbres de la $(\sigma, k+1)$-résolution pour marquer une duplication de gènes dans σ (voir $M_{a,1}$ sur la Figure 3.14).

Lorsque σ est un noeud interne de S, il peut aussi s'agir d'un noeud de spéciation ayant donné naissance aux espèces σ_l et σ_r. Il nous faut donc considérer toutes les (σ, k)-résolutions ne contenant que des arbres de spéciations. Le score $C_{\sigma,k}$ de chacune de ces spéciations correspond à la somme des coûts pour avoir $k - m(\sigma)$ gènes en σ_l et en σ_r parce qu'on sait qu'on en a déjà $m(\sigma)$ (voir illustration pour $C_{d,1}$ sur la Figure 3.14). Plus formellement,

$$C_{\sigma,k} = \begin{cases} M_{s_l,k-m(\sigma)} + M_{s_r,k-m(\sigma)} & \text{si } \ k > m(\sigma) \\ +\infty & \text{sinon} \end{cases}$$

Nous montrons dans [29] que ces idées peuvent se traduire par les équations de récurrences définies dans le théorème 3.4.1 :

Theorem 3.4.1 (Recurrence 1)**.** *$\forall\ \sigma \in V(S)$ et $\forall\ k$ tel que $1 \geq k \geq |P|$,*

Si $\sigma \in L(S)$ (cas de base), $M_{s,k} = \left|k - m(\sigma)\right|$

Sinon, si σ est un noeud interne de $V(S)$, désignons respectivement par k_1 et k_2 la plus petite et la plus grande valeurs de k tel que $C_{\sigma,k_1} = C_{\sigma,k_2} = \min_k C_{\sigma,k}$.

$$M_{s,k} = \begin{cases} C_{\sigma,k} & \textit{si } k_1 \leq k \leq k_2 \\ \min(C_{\sigma,k}, M_{\sigma,k+1} + 1) & \textit{si } k < k_1 \\ \min(C_{\sigma,k}, M_{\sigma,k-1} + 1) & \textit{si } k > k_2 \end{cases}$$

Un simple algorithme *retour sur trace* permet de trouver la résolution optimale P' à partir de la matrice M.

La résolution de polytomies est utilisée comme stratégie de correction par plusieurs programmes tels que NOTUNG [286], ecceTERA [295] et TxT [296] (TxT a la particularité d'accepter des arbres d'espèces non binaires). Dans le chapitre 6, nous présentons aussi une méthode de correction basée sur PolytomySolver qui présente la particularité de réintroduire l'information non-considérée des séquences pendant la résolution. Tous ces algorithmes acceptent des arbres de gènes non enracinés et trouvent l'enracinement minimisant le score de réconciliation en testant toutes les branches. Dans une série d'articles, Górecki et Eulenstein montrent une façon plus efficace pour trouver le meilleur enracinement, sans avoir à considérer toutes les branches [171, 297, 298].

Jusqu'à présent, nous n'avons considéré que la résolution de polytomies sous un modèle de réconciliation DL. Lorsque les HGT sont permis, les polytomies ne peuvent plus être considérées indépendamment les unes des autres. Kordi et Bansal [299] montrent que le problème devient NP-difficile et proposent, dans [300], un algorithme exact testant toutes les binarisations $G' \in \mathcal{R}(G)$. Un algorithme similaire, implémenté dans NOTUNG, est mentionné dans [301], et les auteurs proposent également d'utiliser des heuristiques dont l'une consiste à résoudre l'arbre non binaire des gènes sous le modèle DL, puis à chercher une topologie qui minimise le coût de la réconciliation DTL, dans le voisinage de cette résolution. Un autre algorithme, toujours exponentiel, mais avec une complexité légèrement meilleure est décrit dans [302] et implémenté dans ecceTERA. Ici, les auteurs utilisent les principes d'amalgamation que nous verrons dans la section suivante pour résoudre l'arbre non binaire de gènes. Un inconvénient de la méthode est qu'il faut considérer toutes les partitions pouvant être générées à partir des clades correspondant aux polytomies, augmentant ainsi la complexité en espace.

En raison de la complexité en temps, la correction d'arbres de gènes par résolution de polytomies sous un modèle DTL, n'est pas forcément une stratégie efficace, surtout pour les grands arbres de gènes.

Notons que le problème de la résolution d'un arbre de gènes non binaire a aussi récemment été considéré dans un modèle d'évolution qui considère les évènements de duplication, perte et ILS [303].

3.4.3. Amalgamation et Super-arbres de gènes

Contrairement aux stratégies précédentes, les méthodes intégratives qui sont basées sur l'amalgamation ou l'inférence d'un super-arbre de gènes, nécessitent la construction préalable d'un ensemble $\mathcal{G} = \{G_1, G_2, ...G_n\}$ d'arbres sur la famille de gènes Γ. Lorsque tous les arbres G_i sont sur des sous-ensembles parfois chevauchants de Γ ($L(G_i) \subset \Gamma$) on parle de

construction d'un super-arbre. Lorsqu'ils sont tous sur Γ ($\forall\ i,\ L(G_i) = \Gamma$), on utilise le terme plus spécifique d'amalgamation.

3.4.3.1. *Amalgamation*

L'amalgamation consiste à construire, à partir d'une distribution d'arbres $\mathcal{G}$ (des répliques bootstrap ou des arbres inférés avec une méthode bayésienne), un arbre G qui minimise le score de la réconciliation avec comme contrainte : tout clade de $\mathcal{C}(G)$ doit se retrouver dans l'ensemble $\mathcal{C}(\mathcal{G})$ qui contient tous les clades des arbres $\mathcal{G}$ en entrée.

Ainsi, seuls les clades retrouvés dans l'échantillon d'arbres en entrée et donc probablement supportés par l'information des séquences sont utilisés. L'algorithme calcule, au préalable, le coût de la réconciliation optimale pour chaque clade $c[x] = L(G_{i,x})$ avec les noeuds de S. Ce coût peut s'obtenir à partir des coûts des deux clades $c[x_l]$ et $c[x_r]$ pouvant être induits par les sous-arbres enracinés en x. Il s'agit ainsi d'une procédure post-fixe qui retourne le score optimal au niveau du clade de la racine (correspond à Γ). Un avantage de l'amalgamation est qu'on peut facilement définir un critère de crédibilité sur les clades de $\mathcal{C}(\mathcal{G})$ et ne considérer que ceux satisfaisant ce critère.

À notre connaissance, "l'amalgamation" a été introduite pour la première fois par David et Alm dans leur papier décrivant AnGST [265]. Les auteurs ont utilisé le principe pour inférer des histoires évolutives parcimonieuses (sous DTL) à partir d'une population d'arbres. AnGST requiert un arbre d'espèces daté et les arbres de gènes doivent être non enracinés.

Notons toutefois qu'une méthode basée sur le même principe et qui consiste à fusionner cinq arbres de gènes sur le même ensemble de feuilles (mais construits par des méthodes différentes) afin d'obtenir un arbre hybride minimisant le nombre de pertes et duplications avait déjà été décrite en 2006 par Heng Li, dans sa thèse [304].

Dans [26], l'amalgamation est utilisée, mais cette fois-ci dans un "framework" probabiliste appelé ALE (pour *Amalgamated Likelihood Estimation*), qui combine un modèle de réconciliation probabiliste appelé ODT [305] ainsi que les probabilités conditionnelles des clades (introduites par Höhna et Drummond dans [306] et perfectionnées par Largret dans [307]), et explore de façon exhaustive l'ensemble des arbres de gènes réconciliés pouvant être amalgamés. ALE reconstruit l'arbre des gènes G maximisant la vraisemblance jointe pour l'alignement de séquence et la réconciliation avec un arbre d'espèce S. Une formulation similaire, qui utilise la parcimonie pour la réconciliation est donnée dans [285] et implémentée dans ecceTERA. Les auteurs montrent que leur méthode atteint une efficacité comparable à celle d'ALE, tout en étant beaucoup plus rapide.

3.4.3.2. *Super-arbre*

La construction de super-arbre est un problème très bien étudié [308–314]. Dans sa formulation la plus simple, l'objectif est de vérifier si un ensemble $\mathcal{G}$ d'arbres est consistant c.-à-d. qu'il existe un super-arbre G qui les affiche tous, et si oui, construire G. L'algorithme BUILD [310] trouve une solution à cette question en temps polynomial pour un ensemble d'arbres enracinés. Notez que le super-arbre construit par BUILD n'est pas forcément binaire.

Dans [315, 316], les auteurs se sont intéressés au problème de reconstruction d'un super-arbre de gènes qui minimise le score de réconciliation avec un arbre des espèces. Ils ont montré que le problème était NP-difficile même lorsque l'objectif est de minimiser uniquement le nombre de duplications. Un algorithme exponentiel, $MinSGT$, est ensuite donné pour le problème.

En contraste avec la stratégie de résolution de polytomies, la construction de super-arbres permet de *briser* les monophylies tout en conservant la relation topologique liant les gènes. Cette stratégie est donc particulièrement pertinente lorsque certains gènes sont soumis à une sélection négative et par conséquent regroupés dans la phylogénie des gènes, en raison d'un biais [317].

À notre connaissance, il n'existe pas de méthode de reconstruction de super-arbres de gènes qui considère à la fois duplications, pertes et transferts. Cependant, une méthode publiée en 2014 permet la construction de super-arbres en minimisant la distance SPR, et modélise ainsi implicitement les HGT, mais pas les duplications et les pertes [318].

3.4.4. Autres méthodes

Les méthodes intégratives utilisant l'information de l'arbre des espèces ou la réconciliation ne se limitent pas à celles mentionnées plus haut. Ci-dessous, nous énumérons les autres méthodes connues pour le problème :

— **TreeBeST (gene Tree Building guided by Species Tree)** [304] : Cette méthode est utilisée pour construire les bases de données TreeFam [28] et Ensembl Compara [319]. Plusieurs stratégies de construction sont proposées. L'une d'elles simule le fonctionnement des méthodes de vraisemblance comme PhyML, sauf que la vraisemblance $L(G|A)$ de l'arbre est multipliée par la vraisemblance de sa réconciliation la plus parcimonieuse $L(G, R(G,S)|S)$. Ainsi une mesure jointe séquence-réconciliation est maximisée. Une autre stratégie amalgame des arbres construits par des méthodes différentes pour minimiser le score de la réconciliation DL. Enfin, la dernière construit un arbre NJ de gènes en utilisant l'arbre des espèces comme gabarit. La topologie de

l'arbre de gènes construit présentera le moins d'inconsistance possible avec celui de l'arbre des espèces.

— **Méthode d'inférence bayésienne incorporant la réconciliation :** Ces méthodes définissent une probabilité jointe pour l'alignement de séquences et la réconciliation avec l'arbre des espèces. La procédure estimant la distribution des arbres de gènes les plus probables par MCMC est la même que celle décrite dans la section 3.2.4, la seule différence étant le modèle d'évolution qui tient compte des taux de duplications, pertes, et transferts en plus des mutations de séquences. Ces méthodes incluent JPrIME-DLTRS (anciennement PrIME-GSR) [250, 251, 320] et SPIMAP [282].

— **GIGA (Gene tree Inference in the Genomic Age)** [27] : Il s'agit d'une méthode basée sur les distances entre séquences, qui incorpore additionnellement l'information de l'arbre d'espèces S. La méthode utilise un ensemble de règles définies par rapport à la topologie de S pour joindre les sous-arbres lors de la reconstruction.

— **Minimisation du nombre de NAD :** Cette approche est proposée par Doroftei et El-Mabrouk dans [321] et consiste à corriger un arbre de gènes en enlevant le nombre minimum de feuilles pour que l'arbre ne présente pas de NAD. Le problème est NP-difficile [322] et les auteurs donnent une heuristique.

— **Correction d'arbre de gènes à partir d'un ensemble de relations d'orthologie :** cette approche est proposée par Lafond *et al.* dans [323]. Étant donné un ensemble de relations $\mathbb{R}$ d'homologies (orthologie et paralogie) entre paires de gènes, et un arbre de gènes G, l'objectif consiste à corriger G de façon minimale afin qu'il puisse satisfaire toutes les relations de $\mathbb{R}$. Les auteurs montrent que le problème est NP-difficile et explorent quelques avenues algorithmiques.

— **LabelGTC :** Il s'agit d'une approche récente proposée par El-Mabrouk et Ouangraoua [324] qui vise à unifier les stratégies de correction d'arbres de gènes sous le même "framework", dans le but de tirer pleinement avantage de leurs forces. Les auteurs montrent que les deux problèmes de résolution de polytomies et de construction de super-arbres de gènes ($MinSGT$) sont apparentés et constituent des cas spéciaux d'un problème plus général formulé comme suit :

> `Problème LabelGTC` : Étant donné un arbre de gènes G dont les arêtes sont étiquetées par 0 ou 1 pour indiquer leurs supports, un arbre d'espèces S et un ensemble $\mathcal{C}S$ de sous-arbres terminaux et non chevauchants de G, LabelGTC retourne un arbre corrigé G' de moindre coût de réconciliation que G, tel que G' affiche tous les sous-arbres de $\mathcal{C}S$.

Les auteurs proposent un algorithme pour le problème qui est toutefois exponentiel en raison de la complexité de $MinSGT$. Une autre difficulté rencontrée par la méthode concerne la façon de choisir l'ensemble des sous-arbres $\mathcal{CS}$ à préserver. Dans un travail en cours, une façon de définir ces sous-arbres à partir de scores de similarité fonctionnelle entre groupes de gènes est explorée.

L'avantage des nouvelles méthodes d'inférence d'arbres de gènes dites intégratives par rapport à celles classiques n'est plus vraiment à démontrer. Cependant, elles ne sont pas sans faiblesses. Leur efficacité dépend en effet de la pertinence et de l'exactitude des sources additionnelles d'informations qu'elles utilisent. Par exemple, les erreurs au sein des arbres d'espèces, ainsi que l'utilisation de modèles macro-évolutifs inadéquats sont des facteurs parmi tant d'autres, qui peuvent influencer leur précision.

Par ailleurs, les corrections appliquées par certaines de ces méthodes sont incrémentales et ne garantissent donc pas que l'arbre final soit optimal pour l'ensemble des critères considérés. En améliorant le score de la réconciliation, il est en effet possible que l'arbre obtenu soit beaucoup moins bon pour l'alignement des séquences. Le développement de méthodes combinant efficacement les informations fournies par ces deux sources (et si possible, par d'autres) demeure donc un problème d'intérêt.

Dans le chapitre 7, nous décrirons un algorithme génétique que nous appelons GATC (Genetic Algorithm for gene Tree Construction) qui interprète le problème comme celui d'une optimisation multi-objectif où l'on désire construire un arbre optimal à la fois pour l'information des séquences et celles de l'arbre des espèces.

Chapter 4

CoreTracker: accurate codon reassignment prediction, applied to mitochondrial genomes

L'article présenté dans ce chapitre a été publié dans la revue *Bioinformatics*. Il décrit "CoreTracker", une méthode efficace pour la prédiction d'altérations du code génétique. Le matériel supplémentaire de cet article est disponible à l'annexe A.

Puisque le code génétique définit les correspondances entre codons et acides aminés nécessaires pour la traduction des ARN messagers d'un génome, sa détermination est d'une importance capitale avant toute analyse se basant sur les séquences de ce génome. La moindre altération non détectée du code pourrait donner lieu à des séquences protéiques erronées, propageant des erreurs sur les analyses subséquentes, y compris celles phylogénétiques. Les bases de données actuelles regorgent malheureusement de séquences génomiques très souvent annotées avec le mauvais code, car la présence potentielle de réassignations de codons dans ces génomes n'est pas considérée.

De façon générale, lorsque la réassignation implique un codon-stop, il est facile de l'identifier à partir des outils usuels d'annotation. En effet, s'il s'agit d'un codon-stop qui se transforme en codon sens, des domaines C-terminaux supplémentaires non partagés par aucun des autres gènes homologues auront tendance à être prédits, indiquant ainsi que le vrai codon-stop a été manqué. Dans le cas inverse, des protéines trop courtes seront prédites. Le problème de la prédiction de réaffectation de codons devient plus difficile lorsque les codons changent de décodage d'un acide aminé à un autre. Dans le chapitre 5, le défi que représente ce problème est parfaitement illustré chez les algues vertes où nous montrons la présence de "réassignations sens-à-sens", jusque là jamais détectées, au sein des génomes mitochondriaux publiés de certaines de ces algues. Bien que les études précédentes sur ces génomes aient révélé la présence de réassignations stop impliquant des terminateurs de traduction, aucune n'avait considéré la possibilité de réassignations entre acides aminés.

Ces différentes raisons justifient amplement le besoin de développer des méthodes efficaces pour la prédiction des modifications du code génétique, et particulièrement celles concernant le changement du décodage des codons d'un acide aminé à un autre. La méthode décrite dans ce chapitre apporte une solution à ce dernier problème.

CoreTracker est basée sur l'évaluation statistique des différences entre séquences d'acides nucléiques et celles attendues des protéines, en exploitant l'information sur la conservation d'acides aminés au sein de protéines orthologues.

CoreTracker diffère des méthodes existantes (GenDecoder [325], FACIL [326] et Bagheera [327]) sur trois points principaux qui permettent d'obtenir des prédictions plus fiables:

1. La méthode est capable de tenir compte du contexte phylogénétique des génomes et peut prédire simultanément les réassignations de codons dans un groupe de génomes apparentés. Grâce au contexte phylogénétique, CoreTracker ne se limite pas à prédire les réassignations de codons, mais fournit un cadre d'étude pour l'évolution du code génétique dans un groupe d'espèces. Le contexte phylogénétique est aussi indispensable pour la validation des prédictions, en exploitant l'information disponible sur la proximité évolutive entre les génomes, et permet de réduire considérablement les fausses prédictions causées par la variation importante du taux de substitution entre génomes.
2. Elle intègre des étapes de validation qui permettent de s'assurer que les réassignations prédites ne sont pas erronées. La validation mesure essentiellement l'effet de l'utilisation des déviations du code génétique sur la similarité entre les séquences orthologues. Nous supposons en effet qu'une conséquence directe des réassignations non détectées est la surestimation de la vraie distance évolutive entre séquences.
3. Elle peut prédire la réassignation d'un codon à plusieurs acides aminés différents (si justifiée), ce qui est particulièrement utile puisqu'on sait que le *décodage ambigu* fait partie des mécanismes par lesquels les codons sont réassignés. Un avantage de cet aspect, accentué par la considération du contexte phylogénétique, est la possibilité d'identifier les cas de réassignation de codons en cours.

CoreTracker utilise une forêt d'arbres décisionnels ou RF (pour *Random Forests*) [328] pour prédire les réassignations de codons. Les forêts d'arbres décisionnels désignent une famille de techniques d'apprentissage automatiques qui sont composées d'un ensemble d'arbres de décision, chacun construit et entraîné sur un sous-échantillon des données d'entraînements. La construction des arbres de décision se fait par partitionnements binaires et récursifs de l'espace des données en fonction de leur attributs, l'objectif étant d'optimiser

un critère prédictif. Pour un problème de classification, les noeuds internes de ces arbres correspondent donc à une décision de partitionnement visant à maximiser la séparation entre les classes du problème d'intérêt. Les résultats d'une forêt d'arbres décisionnels s'obtiennent par aggrégation des prédictions d'une collection aléatoire d'arbres décisionnels, offrant ainsi une meilleure puissance prédictive. Pour chaque réassignation, d'un codon $N_1N_2N_3$ d'un acide aminé X_i vers un acide aminé X_j, que CoreTracker considère probable au sein d'un génome, un ensemble d'attributs (incluant la différence d'usage de $N_1N_2N_3$ entre les positions conservées en X_i et X_j) est collecté. Il s'agit des valeurs prédictives qui seront utilisées au sein du RF (voir 4. I). Nous avons entrainé le prédicteur RF sur un ensemble de données construit à partir de réassignations bien caractérisées dans un groupe de 25 génomes mitochondriaux de métazoaires. Les prédictions finales faites par CoreTracker sont donc basées sur la classification retournée par le prédicteur.

Nous montrons sur des génomes mitochondriaux et nucléaires que CoreTracker est beaucoup plus efficace, plus robuste et plus flexible que les alternatives. En particulier, contrairement à GenDecoder [325] et Bagheera [327], CoreTracker n'a pas de restrictions, ni en ce qui concerne le type et l'origine (embranchement taxonomique) des génomes en entrée, ni en ce qui concerne les codons réassignés. La méthode reste toutefois légèrement sensible aux substitutions fréquentes entre acides aminés similaires. Cependant, comme le démontrent nos résultats dans le chapitre 5, étant donné que seules les réassignations les plus plausibles sont prédites, la phase de vérification par analyse de la structure et de l'évolution des ARNt se trouve simplifiée.

Bien entendu, les vérifications par analyses biochimiques *in vitro* ou *in vivo* demeurent requises, en particulier pour confirmer les prédictions lorsqu'un doute persiste, mais l'application de CoreTracker permet de filtrer efficacement le vaste espace de réassignations potentielles de codons à une liste de prédictions facilement testables. Elle peut ainsi servir à guider les expérimentations biochimiques pour ne cibler que les composantes d'intérêt de la machinerie de traduction. Par exemple, dans [113] et [137], les auteurs ont utilisé la spectrométrie de masse pour montrer l'utilisation des codons CUG de la leucine comme alanine ou sérine dans certains génomes nucléaires d'ascomycètes. En utilisant CoreTracker, nous arrivons exactement à la même conclusion (voir annexe A pour les résultats partiels de ce travail en cours). Puisque le séquençage du protéome ne peut objectivement pas être répété pour chaque nouveau génome d'ascomycètes séquencé afin de déterminer le décodage des codons CUG, l'application systématique de CoreTracker durant l'annotation de nouveaux génomes constitue une alternative intéressante.

CoreTracker requiert en entrée un arbre phylogénétique des espèces et les séquences nucléotidiques codantes de plusieurs familles de gènes, préférentiellement celles ne contenant que des orthologues. Lorsque les séquences protéiques de ces gènes sont additionnellement fournies, l'utilisateur a le choix entre proposer son propre alignement, ou utiliser le pipeline d'alignement propre à CoreTracker qui utilise un processus d'affinage à partir de modèles de Markov cachés. CoreTracker est offert avec plusieurs outils d'accompagnement permettant d'effectuer une variété de tâches incluant la gestion, la traduction et l'alignement de séquences ainsi que l'évaluation du biais d'usage de codons/acides aminés et la prédiction de réassignations ancestrales.

Contributions: le projet a été réalisé conjointement avec Virginie Calderon, ancienne étudiante au post-doc du laboratoire, sous la supervision de Nadia El-Mabrouk. Tous les auteurs ont participé au développement de la méthodologie et à la rédaction et révision du manuscrit. Emmanuel Noutahi a implémenté la méthode, développé les outils l'accompagnant, choisi et élaboré les tests statistiques et conçu les tests de validation. Emmanuel Noutahi et Virginie Calderon ont produit et analysé les résultats présentés dans l'article, sur les métazoaires et les levures. Tous les auteurs ont lu et approuvé le manuscrit final.

CoreTracker: accurate codon reassignment prediction, applied to mitochondrial genomes.

Bioinformatics 2017 **33**(21):3331-3339 doi:10.1093/bioinformatics/btx4

Emmanuel Noutahi [1], **Virginie Calderon**[1], **Mathieu Blanchette** [2], **B Franz Lang** [3], **Nadia El-Mabrouk**[1]

Abstract

Motivations. Codon reassignments have been reported across all domains of life. With the increasing number of sequenced genomes, the development of systematic approaches for genetic code detection is essential for accurate downstream analyses. Three automated prediction tools exist so far: FACIL, GenDecoder, and Bagheera; the last two respectively restricted to metazoan mitochondrial genomes and CUG reassignments in yeast nuclear genomes. These tools can only analyze a single genome at a time and are often not followed by a validation procedure, resulting in a high rate of false positives.

Results. We present CoreTracker, a new algorithm for the inference of sense-to-sense codon reassignments. CoreTracker identifies potential codon reassignments in a set of related genomes, then uses statistical evaluations and a random forest classifier to predict those that are the most likely to be correct. Predicted reassignments are then validated through a phylogeny-aware step that evaluates the impact of the new genetic code on the protein alignment. Handling simultaneously a set of genomes in a phylogenetic framework, allows tracing back the evolution of each reassignment, which provides information on its underlying mechanism. Applied to metazoan and yeast genomes, CoreTracker significantly outperforms existing methods on both precision and sensitivity.

Availability and implementation. CoreTracker is written in Python and available at https://github.com/UdeM-LBIT/CoreTracker.

1. DIRO, Université de Montréal
2. School of Computer Science, McGill University
3. Département de Biochimie, Centre Robert Cedergren, Université de Montréal

4.1. Introduction

The genetic code sets the rules for translating the genetic information from coding sequences (genes or mRNAs) into proteins. It was initially deciphered in the 1960s [329, 330]. Using a combination of *in vitro* and *in vivo* experiments conducted on *E. coli*, the complete set of 64 codons has been mapped to either amino acids or a translation termination signal. Long seen as universally conserved among all domains of life (Crick, 1968), the discovery that human and yeast mitochondria interpret the UGA stop codon as tryptophan [70, 71] has challenged the hypothesis of a "frozen" universal code, revealing its evolvability. Since then, many instances of other codon reassignments have been reported across all three domains of life, in both organelle and nuclear genomes [72, 74, 147, 155, 158, 331, 332].

The evolution of codon reassignment requires coordinated changes at the gene/RNA sequence level and the translation machinery in charge of recognizing and assigning a given codon to an amino acid. Two main, mutually non-exclusive mechanisms have been initially proposed for natural codon reassignment. According to the *codon capture mechanism*, the codon first disappears completely from the genome due to mutational pressure, followed by the complete disappearance of the corresponding tRNA or release factor [140, 141]. Then, re-use of this codon by a different amino acid (AA) would emerge, in conjunction with the appearance of a corresponding tRNA and/or tRNA synthetase, with different specificities. This mechanism is present mainly in small mitochondrial genomes encoding small gene sets, for which the disappearance of a codon (leading to an unassigned codon), is feasible. The second mechanism, called *ambiguous intermediate*, does not require the disappearance of the codon during reassignment [144]. Instead, it is ambiguously decoded either by a single tRNA recognized by different aminoacyl tRNA synthetases, or by two different tRNAs (alternatively, a tRNA and a release factor). In the case of the CUG reassignment to serine in some yeast nuclear genomes, it has been hypothesized that a selective advantage could have arisen from a decoding ambiguity, gradually allowing for the reassignment of the codon [146]. More recently, this hypothesis has been challenged by the discovery of CUG reassignment to alanine in *Pachysolen tannophilus* [113–115], suggesting a different mechanism (*tRNA loss driven codon reassignment*) that could explain the polyphyly of the CUG codon usage in yeasts. Sengupta and Higgs have also proposed a classification through gain and loss scenarios [333] which integrates the *codon capture* and *ambiguous intermediate* mechanisms, in addition to their *unassigned codon* and *compensatory change* scenarios.

Computational prediction of codon reassignments is straightforward when stop codons are involved, as proteins with either premature termination or multiple additional C-terminal

domains will be predicted. Sense-to-sense codon identity changes are more difficult to infer, in distinction to ongoing mutations, and when an identity switch occurs among biochemically similar AAs. This motivates the development of appropriate bioinformatics methods. To our best knowledge, three tools, based on comparative sequence analyses, have been developed to predict genetic codes. The GenDecoder web server [325], exclusively designed for metazoan mitochondrial genomes, infers codon reassignments by comparing translations of the standard protein-coding genes of a genome of interest to a set of pre-aligned reference profiles including 54 metazoans. Each codon of the input genome is assigned to the AA to which it is most frequently aligned. The second tool, FACIL [326], which is not specific to mitochondrial genomes, aligns the sequences of interest to PFAM protein domains, then uses Random Forests (RF) to infer the most probable AA-codon match. Finally, Bagheera [327] is a web server for predicting CUG codons reassignment to serine in yeast nuclear genomes, based on the comparison of 38 cytoskeletal and motor proteins to a reference protein dataset. CUG reassignments are predicted by comparing CUG positions within the predicted genes to the reference dataset. The first two methods are restricted to predictions that apply to the codon capture mechanism, as they do not consider the possibility of ambiguous decoding of a codon to more than one AA [75, 334, 335]. Moreover, their predictions are not validated *a posteriori* by measuring the effect of predicted reassignments on the protein alignment quality. This lack of validation, plus a high sensitivity to substitutions between close AAs, usually leads to a high rate of false positives as we show in the result section. Bagheera, on the other hand, has a validation step based on $tRNA_{CAG}$ identity prediction by comparing its sequence to a set of reference tRNAs. The main drawback of this method, however, is its limited scope as it concerns exclusively CUG codon reassignments to serine in yeast nuclear genomes.

Further, these methods are limited to the study of one genome at a time, completely ignoring its phylogenetic context (although Bagheera can perform an *a posteriori* phylogenetic grouping). In contrast, we argue that inferences based on the simultaneous study of multiple related genomes and their phylogenetic relatedness will lead to more sensitive predictions. Such an approach eliminates the dependency on an a priori reference set, thus allowing predictions in newly sequenced phylogenetic groups, and enabling the inclusion of non-standard proteins only shared by certain genomes. Furthermore, a phylogenetic framework can provide data for a better distinction between codon reassignments and ongoing mutations, and since codon reassignment is a progressive change, studying multiple genomes simultaneously will help identify footprints of ongoing reassignments. This innovative way of detecting codon reassignments can offer better insights toward the understanding of the

underlying mechanisms of codon reassignments while systematically tracing back their evolutionary path.

Based on these ideas, we developed CoreTracker, a new algorithm exhaustively exploring sense-to-sense codon reassignments across any given group of genomes. It is the first automated approach for the prediction of codon reassignments that includes a phylogenetic framework and also extends to the context of ambiguous decoding of a codon to various amino acids.

Starting from a set of conserved positions in protein multiple alignments (derived by translating gene sequences with a given initial translation code) and a phylogenetic tree of the considered species, CoreTracker identifies candidate codons for reassignment, based on the recurring incidence of unexpected amino acids in conserved positions. Using a random forest classification approach, candidate codons are then evaluated according to a set of features related to various characteristics of the potential reassignment.

Although both use a random forest approach, CoreTracker and FACIL are significantly different. In contrast to FACIL, which is a complete parameter-free approach, CoreTracker has control over its level of precision and recall. In addition, CoreTracker integrates a correction using a similarity matrix accounting for frequent substitutions between close AAs, and a validation step, which evaluates the impact of a predicted reassignment on the alignment quality given a phylogenetic tree.

We applied CoreTracker to yeast and metazoan mitochondrial genomes and respectively predicted 54 and 85 codon reassignments. We were able to retrieve all known reassignment types and to extend them to newly analyzed genomes. On both datasets, CoreTracker achieved high precision and recall, outperforming FACIL and GenDecoder. We also compared CoreTracker to Bagheera and FACIL on a yeast nuclear dataset, on which CoreTracker also made accurate predictions for CUG codon reassignments and was even able to predict the CUG(Leu, Ala) in *Pachysolen tannophilus* missed by the other methods.

4.2. Methods

4.2.1. Overview of CoreTracker

The algorithm steps are given below and illustrated on Figure 4.1 (see supplementary methods for more detailed information on each step). CoreTracker requires as input, groups of orthologous genes (nucleotide sequences) from the species of interest, and a corresponding phylogenetic tree. To identify orthologous positions, genes are first translated into proteins

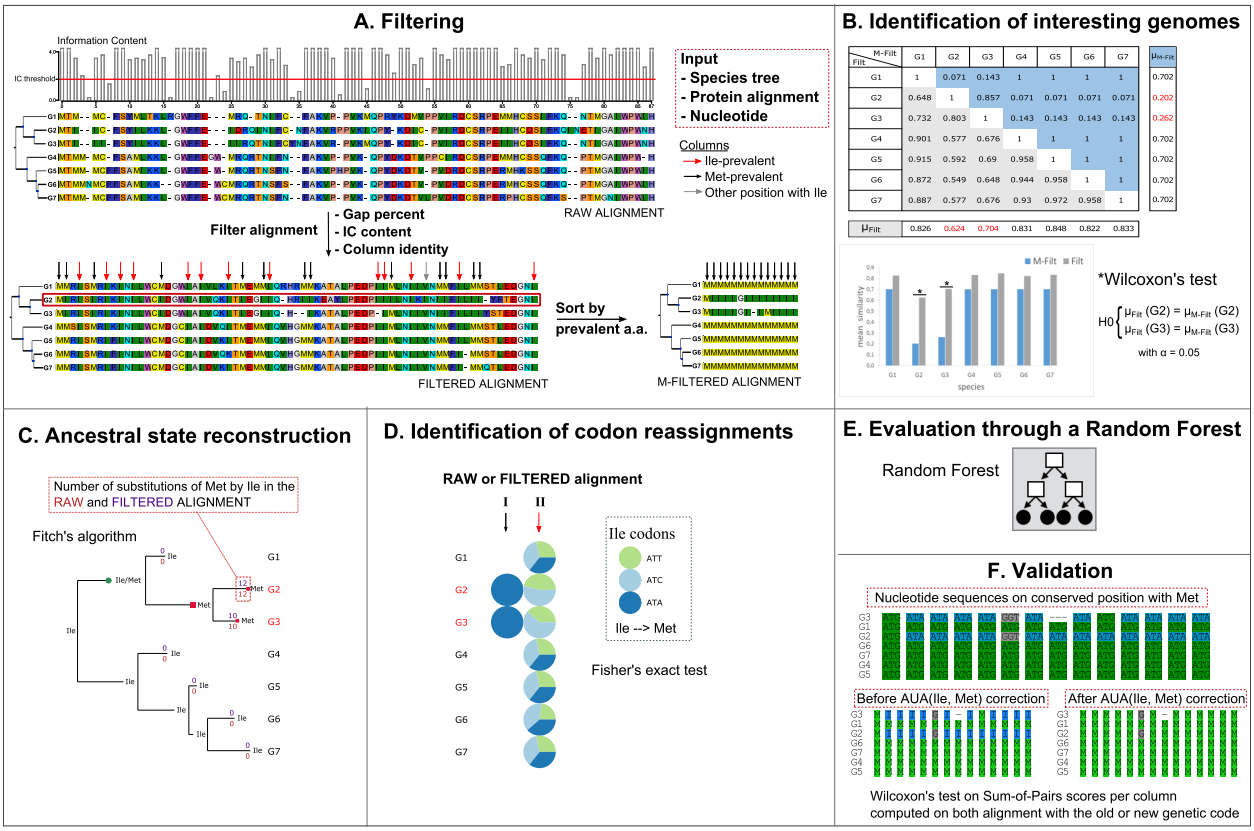


Figure 4.1. Overview of CoreTracker algorithm, illustrated on a simulated example of the reassignment of AUA from isoleucine to methionine. **(A)** The raw, filtered and Met-filtered protein alignments with the relationships between the input genomes. The histogram on top of the raw alignment represents information content at each position. Red, black and gray arrows on top of the filtered alignment's columns indicate respectively the Ile prevalent, Met prevalent and other columns containing Ile. The Met-filtered alignment on right is the concatenation of Met prevalent columns. **(B)** Similarity score matrix between all genome pairs, according to the filtered alignment (bottom gray part of the table), and Met-filtered alignment (top blue part), with mean values represented nearby. A Wilcoxon's signed-rank test is used to evaluate the difference between μ(Filt) and μ(Met-Filt), with stars indicating interesting genomes. **(C)** Leaves corresponding to genomes marked as interesting for a reassignment to Met($G2$,$G3$) are labeled Met. The monophyletic group affected by such reassignment, identified by the red node, is inferred using Fitch algorithm. **(D)** Codon usage of Ile is reported in columns of the alignment (raw or filtered), prevalent in Met (I) or Ile (II). **(E)** To quantify the reliability of the predictions, a Random Forest approach is implemented using ten variables. **(F)** Sum-of-Pairs score for each column affected by the reassignment with the initial and new genetic code are computed then evaluated using Wilcoxon's test.

according to a preliminary, user-defined genetic code. They are then aligned and concatenated into a *raw alignment*, which is filtered by removing highly variable positions that may introduce noise. The obtained *filtered alignment* is used to identify potential codon reassignments. For each amino acid X_i an X_i-filtered alignment is obtained by keeping only columns in which X_i is the prevalent amino acid (Figure 4.1A).

Denote by $C\,(X_i,\, X_j)$ the reassignment of a codon C from X_i to a different amino acid X_j . A candidate genome G for such reassignment is identified by comparing the *average* and *observed* conservation of X_j , respectively computed from the filtered and X_j-filtered alignments. An amino acid X_j that is less conserved than average in G might reflect a different genetic code. In our example (Figure 4.1B), average (μ) and observed (μ_{Met}) conservation of methionine are significantly different in genomes $G2$ and $G3$. These genomes are candidates for the reassignment of isoleucine (which appear in the Met-filtered alignment) to methionine and are labeled as "Met" on the tree (Figure 4.1C). Fitch parsimony [202] is then used on the species tree to trace back the history of each reassignment on the phylogeny. The analysis of the obtained history also helps recover the candidate genomes that would have been missed by steps **A** and **B**, due to an excess of filtering or a low amino acid usage.

On the other hand, an interesting reassignment from X_i to X_j in a genome G is selected according to codon usage. Codon usage is reported for all codons C encoding X_i in G (isoleucine in Figure 4.1D) in two types of columns from the filtered alignment where X_i appears in G : I) columns prevalent in X_j (methionine in Figure 4.1D), indicating a potential reassignment $C\,(X_i,\, X_j)$ in G, and II) columns prevalent in X_i (isoleucine in Figure 4.1D). In a "*perfect*" dataset where C is fully reassigned from X_i to X_j, C would not be present anymore in X_i prevalent columns (type II columns). In practice, due to methodological issues (sequencing errors and alignment quality), ambiguous translation or mutations caused by true genetic divergence, the intersection between X_i and X_j prevalent columns is not expected to be empty. We use a generalized Fisher's exact test instead, to evaluate the difference in synonymous codon usage between the two columns. The p-values returned by this test are strong indicators for the identification of reassignments.

To quantify the reliability of each prediction, we implement a Random Forest (RF) approach (see supplementary methods). RF is a non-parametric classification algorithm [328] which uses many classification trees in parallel. The features used here are described in Table 4. I.

The RF was only trained on a set of 25 metazoan mitochondrial genomes (see supplementary Table S1) extensively studied in the literature [72, 74, 336], and for which we can assume that almost all reassignments have been correctly identified. In order to determine

the most relevant features, we measure their importance according to the Gini impurity index [337] (see supplementary Figure S1). From the ten selected features, the Fisher's exact test contributed the most to the predictive performance of the model. Surprisingly, the fraction of genes affected by the reassignment (`Gene.fraction`) and the distance to the closest reassigned node (`Fitch`) contributed the least.

Predictions made by the RF are then run through two validation steps that measure their impact on the proteome alignment (see section 1.6 in supplementary information for more details. The first step validates the predictions per clade, while the second considers all genomes simultaneously. Both require a re-translation of gene sequences using the newly inferred genetic code. Since a reassignment shared by a whole clade is more likely to affect the same positions across all genomes in the clade, validating by clade ensures that randomly distributed sequence mutations are not inferred as codon reassignments. On the other hand, considering all genomes simultaneously reduces false predictions caused by clade-specific sequence mutations. For both validation steps, we expect an improvement of the similarity between sequences in the protein alignment for a genuine reassignment, if the number of affected codons is significant. (Figure 4.1F). These validation tests are not relevant, however, for reassignments affecting too few positions in the genome.

Table 4. I. All variables evaluated with the random forest for a reassignment $C\,(X_i,\, X_j)$ in a genome G. An asterisk is used to highlight the features that were selected in the current version of CoreTracker.

Features	**Description**
Fitch*	Distance to the closest reassigned node
Freq.Rea*	Frequency of codon in X_j prevalent columns
Freq.Used*	Frequency of codon in X_i prevalent columns
Freq.Mixt	Frequency of codon in other columns containing X_i
Codon.usage*	Usage of the codon in genome G
Fisher.p-value*	P-value of the Fisher's test
Subst.count*	Number of (X_i, X_j) substitutions in the alignment
Codon.likelihood*	Telford score for the codon coding for X_j
Gene.fraction*	Fraction of genes containing the reassigned codon
G.length*	Length of the concatenated genomes
Codon.ID*	One-shot encoding representing the ID of the codon (1-64)
Suspected	Genome G is in the set of interesting genomes (0/1)

4.2.2. Dataset of mitochondrial genomes

We annotated 104 genomes (see supplementary Table S2) from a wide range of yeast mitochondrial genomes, using MFannot (http://megasun.bch.umontreal.ca/cgi-bin/mfannot/mfannotInterface.pl). The only sense-to-sense codon reassignments reported in the literature for these genomes involve CUN and AUA codons. The corresponding genetic code is number 3 in NCBI. As for the 40 metazoan mitochondrial genomes considered in our study (supplementary Table S2), several genetic codes (2, 4, 5, 9 and 13 as referred in NCBI) are required for translation. As all these genetic codes are derived from genetic table 4, we used it to translate the fourteen mtDNA-encoded protein (Cob, Cox1-2-3, Atp6-8-9 and Nad1-2-3-4-4L-5-6) in both the yeast and metazoan mitochondrial genomes and, when necessary, we correct frame-shifts.

4.2.3. Phylogenies of metazoan and yeast species

The yeast phylogeny was constructed using thirteen standard mtDNA-encoded derived protein sequences (Cob, Cox1-2-3, Atp6-9 and Nad1-2-3-4-4L-5-6). The alignment was done in two steps. Sequences were first pre-aligned with Muscle [193] and then refined with *hmmalign* from the HMMER package [338]. Alignments were then concatenated, resulting in a dataset that includes 104 species and has 5812 amino acid positions. The phylogenetic analysis was performed with PhyloBayes [229], using the CAT/GTR model, six discrete categories, four independent chains, 6000 cycles and the -dc parameter to remove constant sites. The first 1000 cycles were discarded as burn-in.

The metazoan phylogeny, on the other hand, is based on the phylogeny of Figure 4 from [72]. New genomes were added, while maintaining the relationships between groups [339, 340].

4.2.4. Comparison of CoreTracker, FACIL and GenDecoder predictions on mitochondrial dataset

We compared CoreTracker, FACIL and GenDecoder predictions on the metazoan dataset. As GenDecoder is restricted to metazoan, only CoreTracker and FACIL were compared on the yeast dataset. We ran CoreTracker with default parameters and without the HMM alignment refinement step. FACIL was iteratively run on each genome in the dataset. A python script was written to query and retrieve the genetic code of each genome from the GenDecoder webserver. Since GenDecoder uses only sequence comparison to predict the genetic code, we used parameters slightly more stringent than the default (metazoan reference dataset,

filtering out columns with more than 20% gap and keeping only "highly conserved S<1.0" sites according to Shannon entropy) in order to increase precision. For all three programs, genetic code 4 was set as reference code. Non-determined predictions (marked by a "?" or "-" for GenDecoder or an "X" for FACIL) were discarded. We kept GenDecoder's unreliable predictions (reported in lower cases) when comparing with CoreTracker and FACIL, as a sizeable proportion of its non-reassigned codons (305/2072) were reported as unreliable. This information on non-reassigned codons is hidden if we remove lower-case predictions, due to precision and recall being computed according to predicted codon reassignments only.

Due to the lack of a gold standard dataset for codon reassignments (even the NCBI annotations cannot be trusted), an initial step was to build a composite reference standard for comparison. For this purpose, literature reviews on codon reassignments in yeast and metazoan, information on species phylogenetic positions (for genomes with no reported predictions but located in a clade affected by a particular codon reassignment) and predictions shared by the three methods were considered to establish a list of true positives consisting of 90 codon reassignments of seven types in metazoans and 72 of six types in yeasts (see supplementary Table S3). Contradictories and ambiguous cases were discarded as well as expected reassignments in genomes avoiding the codon. Since some of the genomes in the metazoan dataset were used in the training set for CoreTracker, we also assessed performance when those genomes were excluded.

4.3. Results

We applied CoreTracker, using default parameters, to 40 metazoan (including the 25 used as the RF training set) and 104 yeast mitochondrial genomes (see supplementary Table S2 for a list of the genomes used, and Table S4 for predicted reassignments and each feature value). CoreTracker was also applied to 23 nuclear yeast genomes. Results on the nuclear genomes are reported and discussed in section 2.1 of the supplementary material (see Table S5 and Figure S3).

4.3.1. Predicted codon reassignments in metazoan mitochondrial genomes

In metazoan mitochondrial genomes, CoreTracker predicted and validated 85 potential sense-to-sense codon reassignments of eight different types (see Figure 4.2), among them the seven described in the literature: AUA(Ile, Met), AGG-AGA(Arg, Ser), AGG-AGA(Arg, Gly), AAA(Lys, Asn) and AGG(Arg, Lys). The new prediction is AUA(Ile, Leu), which was predicted in five genomes (*Euseius*, *Strigamia*, *Lumbricus*, *Epigonichthys* and *Ciona*).

We were able to extend the known codon reassignments to 13 new genomes. For example, the AUA(Ile, Met) reassignment has been reported in the literature for 14 genomes (see Fig. 4 of [72]). Here, the prediction was extended to five more genomes located in monophyletic groups where the reassignment has already been detected.

Regarding AGR(Arg, Ser) and AGR(Arg, Gly) reassignments, most of them were correctly inferred and extended to the remaining genomes of the considered taxonomic group. Two additional genomes, *Aphrocallistes vastus* and *Metaseiulus occidentalis*, belonging to different monophyletic groups, were predicted to have both reassigned AGR codons to serine and glycine. CoreTracker also predicted AGA(Arg, Ser) in *Strigamia maritima*, while the synonymous arginine codon AGG was predicted to be reassigned to lysine. This pattern of reassignment has previously been observed in *Limulus polyphemus* and some other arthropods [133] highlighting the fact that synonymous codons can be independently reassigned to different amino acids.

As for AAA(Lys, Asn), the four previously known reassignments were retrieved and extended to eight of the nine analyzed *Platyhelminthes*. This reassignment was not predicted in *Echinococcus equinus* since its AAA codon usage is very low, and the codon is absent in asparagine-predominant columns (column I). In this genome and also in *Haplorchis taichui*, AAA(Lys, Ser) was predicted but not validated (not shown) because too few codons were involved. A closer look at the alignment showed that, in the positions affected by AAA(Lys, Ser), asparagine codons were found in other closely related *Platyhelminthes* genomes. This further supports AAA(Lys, Asn) reassignment in *Platyhelminthes*, under the hypothesis of an ancestral substitution from serine to asparagine in some positions.

The new identified reassignment type AUA(Ile, Leu), although supported by the validation steps, remains questionable for various reasons. Aside from the fact that the two amino acids are highly similar and frequently substituted each other, the reassignment concerns five genomes that are spread apart in the phylogeny and already predicted to have reassigned AUA to methionine. Furthermore, according to the validation steps, AUA(Ile, Met) seems to improve the alignment more than AUA(Ile, Leu) (p-value of 1.54e-10 vs 4.06e-04). A closer look at positions affected by AUA(Ile, Leu) in the alignment showed that leucine's CUA and UUA codons are used in related genomes, suggesting sequence substitutions at the nucleotide level rather than codon reassignment. In some of these positions, we also found valine and methionine in a few genes (Cox1, Cob, and Nad3). These mutations mostly occur in transmembrane domains where substitutions between hydrophobic residues is tolerated [341, 342], further supporting nucleotide substitutions over reassignment. However, potential decoding of AUA codons as leucine cannot be ruled out entirely without biochemical evidence.

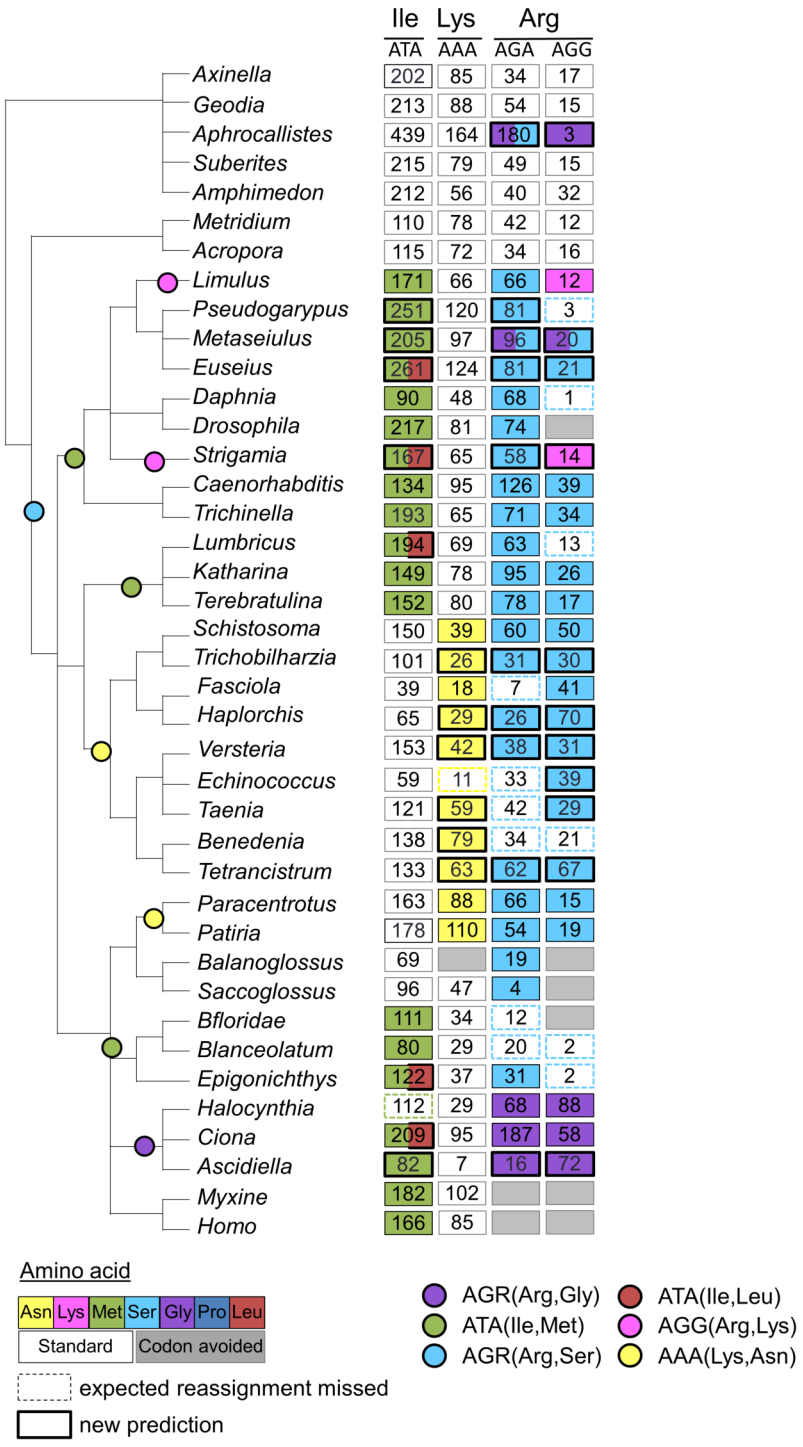


Figure 4.2. Known and inferred reassignments in metazoan mitochondrial genomes. The species tree on the left is based on the literature [72]. Numbers in rectangle indicate, for each genome, the reassigned codon (columns) count in the 13 standard mitochondrial genes

Notice that few known AGR(Arg, Ser) and one AUA(Ile, Met) reassignments were missed. As shown in Figure 4.2, these missed predictions coincide with low codon usage in corresponding genomes. It is possible that an excess of filtering plus a very low usage of these codons in conserved positions, concealed the reassignment signal.

4.3.2. Predicted codon reassignments in yeast mitochondrial genomes

In yeast mitochondrial genomes, CoreTracker predicted and validated 54 codons reassignments of seven types (see Figure 3). These types include known CUA-CUU-CUG(Leu, Thr), AUA(Ile, Met) and CUA-CUU(Leu, Ala) reassignments and a new AUA(Ile, Leu) reassignment.

Among the 54 inferred reassignments, ambiguity only lies for the translation of the codon AUA to either methionine or leucine in the two *Ashbya gossypii* species. As in the metazoan dataset, AUA(Ile, Leu) is most likely a false positive.

In yeast mitochondrial genomes, CUN codon reassignments from leucine to threonine were previously detected in *Saccharomycetaceae* mitochondrial genomes [109, 141]. CoreTracker predicted this reassignment for 23 of the 104 analyzed genomes. The predicted least common ancestor of all genomes affected by this reassignment is shown on the phylogeny (Figure 4.3). The corresponding monophyletic group involves 26 genomes. This subgroup also contains two genomes, *Kluyveromyces lactis* and *Kluyveromyces marxianus*, where CUN(Leu, Thr) was not predicted, due to the complete absence of CUN codons in the 13 mitochondrial genes we considered.

CUU and CUA codons have been reported to be reassigned to alanine in *Ashbya gossypii ATCC10895* [110]. Our dataset includes two additional *Ashbya* genomes: *Ashbya gossypii FDAG1* and *Ashbya aceri*. Both CUU and CUA codons are predicted to be reassigned to alanine with a high probability (0.99), in all *Ashbya* species. Since CUC and CUG codons are avoided in most yeast mitochondrial genomes, they were often not found after filtering the alignment and were therefore rarely predicted to be reassigned.

In the literature, the AUA codon was reported reassigned from isoleucine to methionine in *Saccharomyces* and *A. gossypii* mitochondrial genomes [157]. Our predictions are in agreement with these results. According to the ancestral state reconstruction, such a reassignment appears to have occurred twice during the evolution of two *Saccharomycetaceae* monophyletic lineages (indicated by a green circle on Figure 3). However, the low codon usage in *Saccharomycetaceae* outside *Saccharomyces* and *Ashbya* clades (for example *Lachancea*), suggest that a single event may have affected the whole group leading eventually to AUA reassignment in certain subgroups.

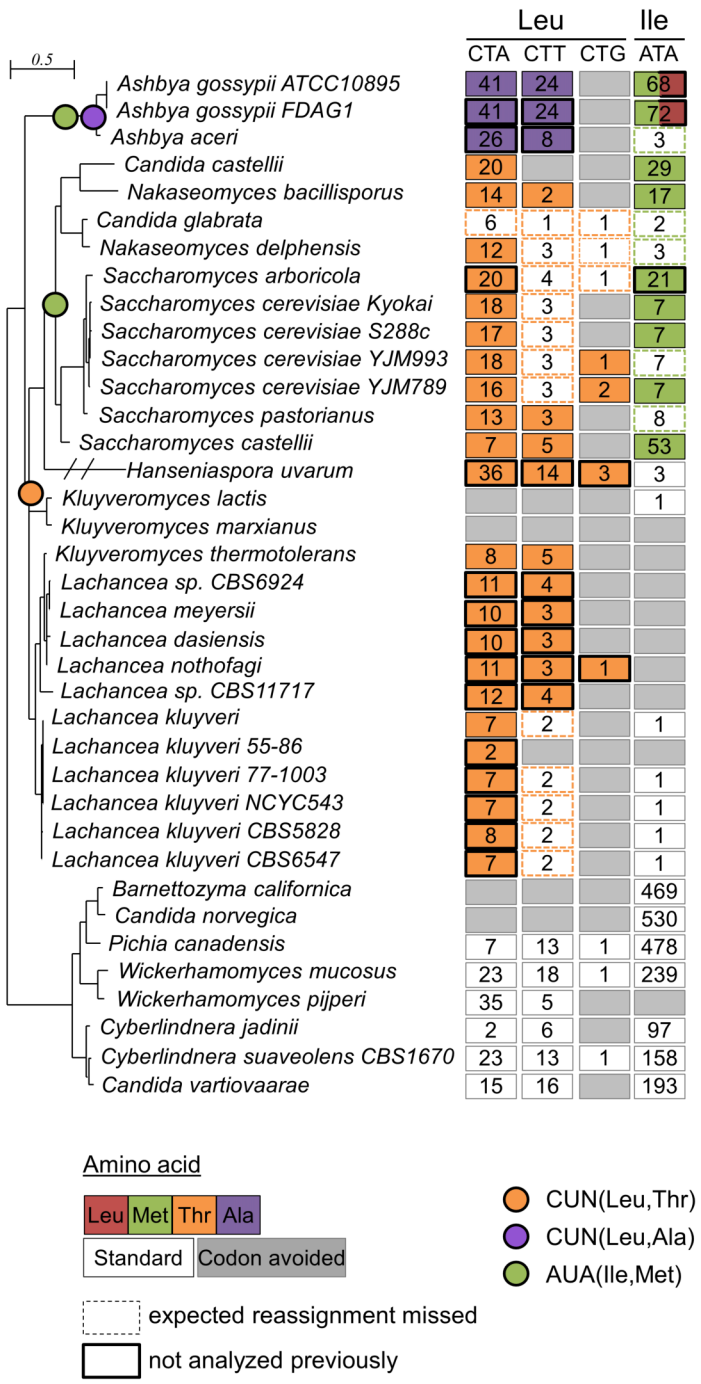


Figure 4.3. Known and inferred reassignments in yeast mitochondrial genomes. The species tree on the left was computed using the 13 standard mitochondrial proteins (See Methods) and has been pruned to keep only genomes relevant to the discussion section. Numbers in rectangle indicate for each reassigned codon (columns) in each genome, the codon usage in the 13 standard mitochondrial genes. Empty grey rectangles indicate that the codon is completely avoided in the corresponding genome

Finally, as in metazoans, some expected reassignments are missed by CoreTracker due to low codon usage in conserved coding regions. In particular, the avoidance of CUC, CUG and even CUU codons in most *Saccharomycetaceae* make the prediction and validation of their reassignments difficult.

4.3.3. Efficiency of CoreTracker compared with FACIL and GenDecoder

We compared CoreTracker to GenDecoder and FACIL on metazoan and yeast mitochondrial datasets (Table 4. II). The latter algorithm is more comparable to CoreTracker, as it is not restricted to any particular phylum or type of genome. The three methods were compared on a manually curated composite reference dataset (see section 4.2.4 in methods), in terms of precision, recall and F-score (harmonic mean of precision and recall). This dataset contains 90 codon reassignments of seven types in metazoans and 72 of six types in yeasts (see supplementary Table S3).

On the metazoan mitochondrial dataset, CoreTracker achieved the highest precision (89.4%) and F1-score (86.9%), while FACIL, the closest in terms of precision has a F1-score of only 69.9%. This high precision was also achieved when genomes from CoreTracker's training set were removed. Only GenDecoder was slightly more sensitive than CoreTracker (87.8% vs 84.4%), but it had a lower precision. By removing GenDecoder's unreliable predictions (reported in lower-case), it's precision jumps from 54.9% to 89.3%, at a cost of a decrease in sensitivity (74.4%), making it the second best tool according to the F1-Score (81.2%) (see section 4.2.4 for a discussion on keeping or removing GenDecoder's lower cases). It is noteworthy that CoreTracker can achieve better recall by decreasing the stringency of the alignment filtering parameters. As for the yeast mitochondrial dataset, CoreTracker significantly outperformed FACIL by displaying precision as high as 96.3% and a recall of 72.2%, compared to FACIL with a precision of only 25.6% and a recall of 47.2%. A closer look at the two programs' predictions on the yeast dataset suggest that they both have difficulties predicting reassignments involving codons with extremely low usage, such as CUU(Leu, Thr) and CUG(Leu, Thr) in *Saccharomycetaceae*, which explains the low recall. However, CoreTracker was able to detect the CUG(Leu, Thr) reassignments with only one codon count in *L. nothofagi* that FACIL missed (see discussion). FACIL also failed to predict most AUA(Ile, Met) reassignments, whereas CoreTracker was able to predict them in 9 genomes.

In addition, the alternative comparison of supplementary Figure S2, show that GenDecoder and FACIL predicted several unlikely codon reassignments types. In particular, GenDecoder made 48 unheard-of codon reassignment prediction types, not shared with any of the two other methods, and never reported in the literature. Although the three methods

Table 4. II. CoreTracker, FACIL, and GenDecoder are compared in terms of precision (P), recall (R) and F1-score (F1) on metazoan and yeast mitochondrial genomes dataset. CoreTracker performs better than the other methods. For direct comparison, the numbers of true positives (TP), false positives (FP) and false negatives (FN) are also provided.

	CoreTracker						**FACIL**						**GenDecoder**					
	TP	FP	FN	P	R	F1	TP	FP	FN	P	R	F1	TP	FP	FN	P	R	F1
Metazoan (all)	76	9	14	89,4%	84,4%	86,9%	64	29	26	68,8%	71,1%	69,9%	79	65	11	54,9%	87,8%	67,5%
Metazoan (new genomes)	31	6	10	83,8%	75,6%	79,5%	30	13	11	69,8%	73,2%	71,4%	33	40	8	45,2%	80,5%	57,9%
Yeast	52	2	20	96,3%	72,2%	82,5%	34	99	38	25,6%	47,2%	33,2%	-	-	-	-	-	-

have difficulty to distinguish codon reassignments from substitutions between amino acids with similar properties due to neutral evolution, this effect is more noticeable in GenDecoder's and FACIL's predictions.

4.4. Discussion

CoreTracker is the first automated tool developed for codon reassignment prediction that uses a phylogenetic context. It is a generic method that allows exploring a large number of genomes from any taxonomic group, with distinct reassignment scenarios. Application to mitochondrial and nuclear genomes highlights its ability to efficiently predict known sense-to-sense reassignments.

4.4.1. Reassignments in metazoan and yeast mitochondrial genomes

In metazoan genomes, CoreTracker correctly predicted the five previously known and well characterized sense-to-sense reassignment types and was able to extend them to newly analyzed genomes. Here, we discuss the unexpected AGR(Arg, Ser) and AGR(Arg, Gly) simultaneous predictions in the hexactinellid sponge *Aphrocallistes vastus* and in the arachnid *Metaseiulus occidentalis*.

It has been previously suggested [72] that AGR(Arg, Gly) is caused by the gain of a new Gly-tRNA$_{\mathrm{UCU}}$ after the reassignment of AGR(Arg, Ser). This tRNA then outcompetes Ser-tRNA$_{\mathrm{GCU}}$ to decode AGR codons, leading eventually to AGR(Arg, Gly). However, it is also known that in several invertebrate groups, the G base of Ser-tRNA$_{\mathrm{GCU}}$ is mutated into a U, resulting in Ser-tRNA$_{\mathrm{UCU}}$ which is also able to decode AGR codons, and sometimes with better affinity [133].

Previous studies have revealed the presence of Ser-tRNA$_{\mathrm{UCU}}$ in the mitochondrial genome of *Aphrocallistes* [343, 344]. This tRNA has also been previously predicted in *Metaseiulus occidentalis* [345]. The two genomes also have only one Arg-tRNA$_{\mathrm{UCG}}$ and a single Gly-tRNA$_{\mathrm{UCC}}$, supporting further the decoding of AGR codons as serine by Ser-tRNA$_{\mathrm{UCU}}$. Considering these results and the fact that only a single point mutation from G to A is required to transform glycine's GGR codons into AGR codons (which can occur due to sequencing errors or sequence mutation), it can be argued that the AGR(Arg, Gly) prediction in *A. vastus* and *M. occidentalis* are most likely false positives. However, in *A. vastus*, according to CoreTracker's output, while AGR codons are completely avoided in arginine-conserved positions, AGA was found in 14 conserved serine positions compared to 16 conserved glycine positions and AGG in three glycine conserved positions. In most of those glycine positions (particularly abundant in Cox1 and Cob gene), GGN codons were found in other sponges,

while the three tunicates known for AGR(Arg, Gly) used AGR codons. Moreover, AGG(Arg, Gly) was also predicted in *A. vastus* and *M. occidentalis* by GenDecoder, suggesting that AGR codons could be ambiguously decoded as serine and glycine in these genomes. Further investigation through tRNA phylogenetic analysis and enzymatic study, which is beyond the scope of this paper, is therefore needed.

In yeast mitochondrial genomes, CoreTracker confirmed and extended known CUA-CUU(Leu, Ala) in the *Ashbya* and CUN(Leu, Thr) in *Saccharomycetaceae* except for *K. lactis* and *K. marxianus* in which the CUN codons are absent. CUN reassignments have been extensively studied, and it has been shown that reassigned CUN codons are decoded by a tRNA_{UAG} that emerges from the duplication of His-tRNA_{GUG} then further diverges into two distinct identities to decode CUN as either threonine or alanine. It is believed that this reassignment is preceded by the disappearance of CUN codons, as illustrated on Figure 4.3 by their absence in *Kluyveromyces* and their low usage in other *Saccharomycetaceae* genomes. Although CoreTracker was able to predict most CUU and CUA reassignments, reassignments involving CUG and CUC were harder to predict, due to their extremely low usage. This low usage was expected, however, since yeast mitochondrial genomes are AT-rich with strong mutation pressure toward A and U. In *S. cerevisiae YJM993* and *L. nothofagi* where CUG codons appear only once, CoreTracker predicted CUG (Leu, Thr) with strong support. This reassignment was also predicted by FACIL in *S. cerevisiae YJM993* but missed in *L. nothofagi*. From CoreTracker's output, it can be observed that the leucine codon usage difference in leucine and threonine conserved columns is extremely high (p-value of 3.41e-10 in *S. cerevisiae* and 1.57e-07 in *L. nothofagi*). In fact, UUA was almost the only codon used in leucine conserved positions, while CUA and the only CUG were found in highly conserved threonine positions. As synonymous codon usage is one of the most important features of the RF model, this weighs a lot. Furthermore, CUG appeared in an extremely conserved threonine column with only a few *S. cerevisiae* using CUA while all the remaining genomes used threonine's ACA or ACU codons. Therefore, the clade validation test for CUG was successful. The second validation test was also successful since it simultaneously considers all synonymous codons predicted reassigned to the same amino acid, thereby CUG was validated alongside the more frequent CUA and CUU codons. If this second validation test were to be performed separately for each codon, there would not be enough positions for the improvement in the alignment quality to be significant (p-value of 2.57e-01) for CUG (Leu, Thr), and the reassignment would have failed the validation test. Note that CoreTracker has a parameter to set the minimum occurrence of a codon required before prediction, which by default is one.

As for AUA(Ile, Met), it was inferred in both *Ashbya* and *Saccharomyces*. This reassignment was previously reported to be linked to the loss of the gene encoding Ile-tRNA$_{CAU}$ in the ancestral *Saccharomycetaceae* genome, followed by a gain of function in the Met-tRNA$_{CAU}$ which was then able to decode both AUG and AUA codons [72]. Such hypothesis requires the avoidance of AUA codons at one point of the reassignment process. As shown on Figure 4.3, AUA codons are effectively either completely absent or avoided in other *Saccharomycetaceae*, supporting that theory.

4.4.2. Limitations, flexibility and possible extensions

Although CoreTracker was able to detect some reassignments of weakly used codons and reassignments occurring in single genomes (such as AGG(Arg, Lys) in *Strigamia* on Figure 2), codon usage remains a limitation of the method. In fact, in the validation step, measuring the impact of the new genetic code on the proteome is informative only if the new genetic code affects enough positions to significantly alter the alignment quality. To overcome this limitation, prior knowledge on functional domains may be used to attribute different weights to reassigned positions according to their location in the gene. In our study, we checked the location of the predicted reassignments according to PFAM domains. However, as almost all predicted reassignments were in such domains, this step did not offer any significant filtering advantage. Alternatively, using more proteins when available, as it is the case for nuclear genomes, can help reduce the effect of codon usage limitation. However, the trade-off will be the increased running time needed to analyze this larger dataset.

As input protein sequences are obtained from the translation of annotated genes, CoreTracker solely predicts sense-to-sense reassignments. Although this can be seen as a limitation, reassignments involving a stop codon are easily predicted by most existing annotation tools. In fact, missing C-terminal domains or proteins with abnormally long or short length, compared to others from the same family, are reliable indicators of stop-to-sense and sense-to-stop codon reassignments. By default, CoreTracker also removes from the input dataset, genes with frame-shifts. Since this might not be the desired action, we provide tools to help identify and correct frame-shifts.

In order to measure the performance of CoreTracker, we evaluated its precision and sensitivity compared to GenDecoder and FACIL on mitochondrial genomes. In contrast with CoreTracker, neither GenDecoder nor FACIL uses a phylogenetic framework. Instead, each genome is analyzed independently. Applying the three algorithms on metazoan and yeast mitochondrial genomes, CoreTracker performed better than both GenDecoder and FACIL. Indeed, although GenDecoder displayed a slightly higher sensitivity than CoreTracker, both

GenDecoder and FACIL demonstrated lower precision. Because GenDecoder and FACIL are based on pre-established reference datasets that do not necessarily allow an appropriate taxon sampling, both algorithms are highly sensitive to amino acid substitutions, which largely explains their lack of precision. CoreTracker addresses this issue by offering a wide range of control over its precision, and built-in steps that ensure accuracy. Aside from the possibility to perform appropriate taxa sampling, facilitated by the provision of dataset merging tools, the added validation steps help filter out unreliable predictions. For instance, on the metazoan dataset, 97 predictions were initially made, but only 85 were validated, increasing precision by 11.06%. As CoreTracker is user-oriented, it outputs several additional information (both visual and statistical) on each prediction, and even on some non-predicted reassignments to help users decide if a predicted (or non-predicted) reassignment should be further inquired into.

CoreTracker's efficiency on nuclear genomes was also assessed by comparing its predictions on a yeast nuclear dataset to Bagheera's and FACIL's. On this dataset, CoreTracker accurately predicted CUG codon reassignments, missing only CUG(Leu, Ser) in *L. elongisporus* (see Figure S3). More importantly, it was the only method able to detect the CUG(Leu, Ala) in *Pachysolen tannophilus*. This application to nuclear genomes demonstrates its consistent high accuracy and its wide range of applicability.

It is worth noticing that CoreTracker's predictions are strongly dependent upon the sequence alignment. However, as shown in Figure S4, the method is robust even towards unrealistic high rates of errors in the alignment. Nevertheless, we provide a default alignment pipeline (see methods section) that use HMMs to refine alignments. Although CoreTracker does not require the full resolution of the input phylogenetic tree (as shown by the metazoan phylogenetic tree used), predictions depend on the considered phylogenetic tree. In particular, a reassignment affecting a whole clade will only be predicted if a genome outside this clade and missing the reassignment is included in the analysis.

One useful information that was not included in CoreTracker is the analysis of tRNAs, given that codon reassignments are often linked to changes in tRNAs. Analyzing tRNA sequences is particularly useful when a set of reference tRNAs that have changed identity is available. In this case, it will be possible to predict reassignments by comparing predicted tRNAs of a query genome to the reference tRNA set, as done by Bagheera. This approach is suitable when predictions are restricted to a specific codon reassignment but less applicable when all potential codon reassignments are being considered as it is the case for CoreTracker. In this latter scenario, analysis of the full tRNA repertoire is necessary and will require complete characterization of the tRNA identity determinants, and a reliable

tRNA phylogenetic tree. Unfortunately, these information are often not available [139, 346] or difficult to reliably obtain without human manual intervention. Furthermore, tRNAs are not the only components involved in codon reassignments, since mutations in aminoacyl tRNA synthetases can also potentially lead to codon reassignments. Therefore, despite being linked, there is not necessarily a one-to-one correspondence between tRNA evolution and codon reassignments. Although tRNA analysis could not be automatically included in CoreTracker, it still represents an important validation test that should be done when possible. Future extensions of CoreTracker will involve accounting for branch length variation in the ancestral reconstruction step and the random forest as well. Other changes at the mRNA level that could be confounded with codon reassignments, such as RNA editing, will also be taken into account, in order to make CoreTracker a truly universal tool for codon reassignment predictions. Possible ways to achieve distinction between RNA editing, codon reassignment, and artifacts due to amino acid substitution, include consideration of relative codon usage between genomes, appropriate codon substitution models and above all, tRNA phylogenetic analysis as an additional validation step.

Acknowledgements

We thank G. Thauvette for helpful discussions. We are grateful to J. Nosek for providing his comprehensive collection of yeast mitochondrial genomes and to three anonymous reviewers for their comments and suggestions that significantly improved the article.

Chapter 5

Rapid genetic code evolution in green algal mitochondrial genomes

Ce chapitre est un article en préparation (voir l'annexe B pour le matériel supplémentaire). Il s'agit d'une continuation directe du travail présenté au chapitre précédent. Nous étudions ici l'évolution du code génétique dans les génomes mitochondriaux des algues vertes et en particulier des chlorophytes (voir Figures 5.1 pour un aperçu phylogénétique).

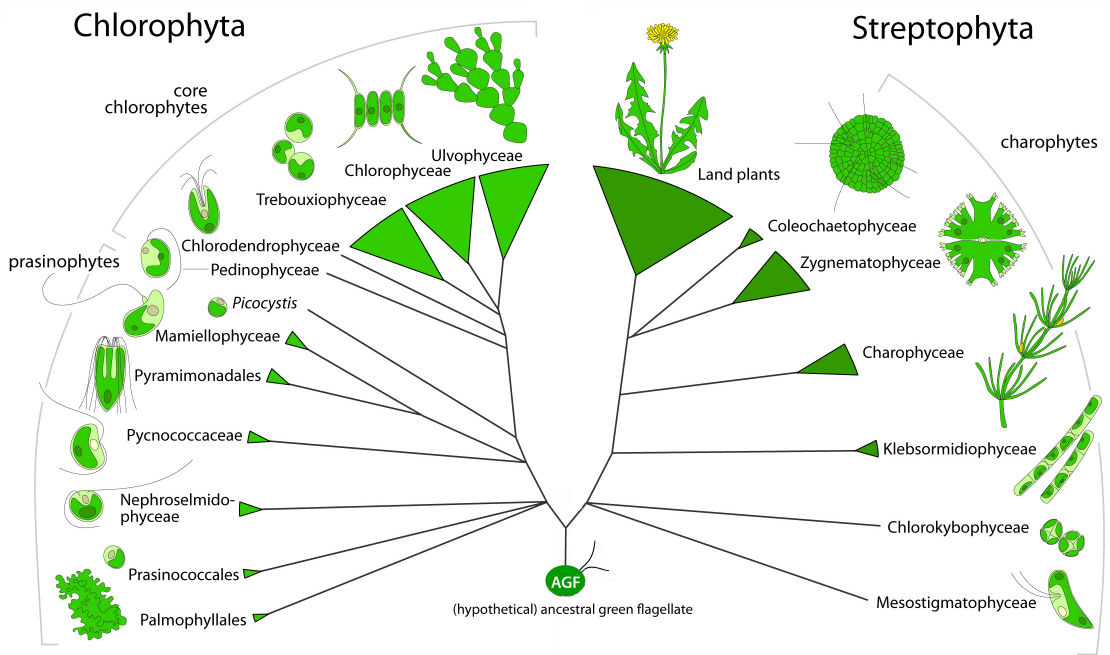


Figure 5.1. Aperçu de la phylogénie des plantes (Viridiplantae) essentiellement formée des algues vertes et des plantes terrestres. Cette Figure est une adaptation, avec la permission de l'éditeur, de la Figure 2 de la revue par Leliart *et al.* sur l'évolution des algues vertes [347].

Le choix des génomes mitochondriaux d'algues vertes est justifié par la découverte de réassignations impliquant les codons stop chez certains chlorophytes [118, 120, 121, 130], et par le taux de substitutions anormalement élevé, à l'origine des attractions de longues branches documentées lors de la reconstruction de leurs phylogénies [120, 121]. Notre hypothèse principale est que ce taux cacherait plutôt des réassignations de codons. Pour tester cette hypothèse, nous avons développé un cadre d'étude de réassignations de codons centré autour de CoreTracker. Ce nouveau cadre intègre additionnellement à l'analyse des séquences codantes que fait CoreTracker, l'étude de l'évolution des ARNt. Nous étudions cette évolution en inférant la vraie identité des ARNt, puis leur histoire évolutive à partir de leur phylogénie et de leur ordre au sein des génomes.

Comme nous le montrons dans le chapitre, un avantage de cette approche est qu'elle permet d'élucider avec précision les scénarios évolutifs conduisant aux altérations du code génétique. Notons que l'approche aura du mal à identifier correctement les scénarios évolutifs lorsque la mutation responsable du changement dans le code est localisée au niveau de l'aaRS ou est causée par un ARNt importé d'un autre compartiment cellulaire. Elle prédira toutefois les réassignations résultantes, ce qui permet de tester ensuite les diverses hypothèses par analyse biochimique.

Les résultats obtenus à partir de notre cadre d'étude montrent, pour la première fois, la présence de réassignations sens-à-sens dans les génomes mitochondriaux de chlorophytes. Nos résultats montrent clairement que la réassignation des codons AGG chez les Sphaeropleales est primordialement causée par la perte de l'Arg-ARNt(UCU) ancestral, canonique au codon. Nous démontrons aussi que les ARNt mutants décodant ces codons sous une nouvelle identité ont des origines bien distinctes, suggérant donc que les voies évolutives conduisant aux réassignations sont diverses et uniques à chaque génome, même si l'évènement initial déclencheur est partagé.

Enfin, nos résultats indiquent également que plusieurs forces évolutives rentrent en jeu au cours de l'évolution du code génétique. Ces forces incluent l'usage des codons, la pression mutationnelle G+C, la minimisation des génomes, la réduction de la taille des protéomes, et l'évolution des ARNt par mutations, pertes et gains. Étrangement, plusieurs génomes présentant certaines de ces caractéristiques conservent le décodage standard, ce qui soulève plusieurs questions sur la façon dont ces forces évolutives agissent conjointement pour causer les réaffectations de codons.

Bien que nos résultats soient convaincaints, certains d'entre eux nécessiterait une validation expérimentale. En particulier, les réassignations AUA(Ile→Met) et UGA(Stop→Trp) prédites chez le prasinophyte *Pycnococcus provasolii* devraient idéalement être validées par

la charactérisation de l'activité biochimique des deux ARNt mutants Met-ARNt(CAU) et Trp-ARNt(CCA) ainsi que celle des synthétases correspondantes.

Contributions: Emmanuel Noutahi a mis en place la méthodologie, effectué les analyses et produit les résultats, sous la supervision de Franz Lang. Le travail présenté est basé sur les résultats préliminaires obtenus au cours de l'évaluation de CoreTracker par Virginie Calderon et Emmanuel Noutahi. Le manuscrit a principalement été rédigé par Emmanuel Noutahi, avec les contributions importantes de Franz Lang et de Nadia El-Mabrouk et les commentaires et suggestions des autres auteurs.

Rapid genetic code evolution in green algal mitochondrial genomes

In preparation

Emmanuel Noutahi [1], Virginie Calderon [2], Mathieu Blanchette [3], Nadia El-Mabrouk[1], B Franz Lang [4],

5.1. Abstract

Genetic code deviations involving stop codons have been described in a handful of green plants mitochondrial genomes. The remarkably high sequence evolution rate in several green plants lineages suggests, however, that alterations involving a transition from one sense codon to another might have gone unnoticed. To gain insight into the mitochondrial genetic code evolution in green plants, we have performed an in-depth study of codon reassignments across 36 green plants mtDNAs. Besides confirmation of known stop-to-sense reassignments, our study documents the first cases of sense-to-sense codon reassignments in Chlorophyta mtDNA. In most Sphaeropleales, we report the decoding of AGG codons (normally arginine) as alanine by remolded tRNAs(CCU) of various origin. In *Chromochloris*, we report the presence of mutant tRNAs decoding AGG as methionine and the synonymous codons CGG as leucine. Although the underlying mechanism remains unknown, we have also found strong evidence supporting the decoding of AUA codons (normally isoleucine) as methionine in *Pycnococcus*. Finally, we show that while the mitochondrial genetic code has evolved independently and through distinct mechanisms in these lineages, changes were mostly facilitated by the substantial genome reduction they underwent and their high A/T mutation pressure. Our results rely on a new conceptual framework for studying genetic code evolution that explicitly accounts for codon reassignments effect on the disparity between DNA and protein sequences, as well as for tRNAs evolution through duplications, losses, remolding and structural modifications.

1. DIRO, Université de Montréal
2. Institut de Recherches Cliniques de Montréal
3. School of Computer Science, McGill University
4. Département de Biochimie, Centre Robert Cedergren, Université de Montréal

5.2. Introduction

Green plants (Viridiplantae) constitute a monophyletic group divided into two major lineages: the Chlorophyta (Chlorophyceae, Ulvophyceae, Trebouxiophyceae and Prasinophyceae classes) and the Streptophyta (Charophyta plus land plants) [348]. In the following, we will use this taxonomic definition, rather than the term 'green algae' (Charophyta plus Chlorophyta, a paraphyletic grouping) that was previously common in the literature.

In recent years, the mitochondrial genomes (mtDNA) of green plants, in particular, Chlorophyta, have gained interest due to their complex genome organization and pronounced structural diversity across clades, correlating with major differences in evolutionary rates [349, 350]. Compared with the hyper-inflated, sometimes multi-partite mitochondrial genomes of land plants, those of Chlorophyta tend to have a more compact genome organization, with a reduced gene and intron count. To date, several distinct patterns of evolution have been described among the Chlorophyta, ranging from highly reduced/derived to ancestral types. In several Chlamydomonadales (Chlorophyceae class) and in *Pedinomonas minor*, a member of the deeply diverging Prasinophyceae, the mtDNAs feature both a greatly reduced gene content and a highly accelerated rate of sequence evolution. They are characterized by the lack of ribosomal protein-coding genes, a severely reduced tRNA repertoire, and the presence of fragmented rRNA genes [120, 351–354]. By contrast, the much larger mtDNAs of other Prasinophyceae (*Nephroselmis*, *Ostreococcus*, *Micromonas*) and *Prototheca* (class Trebouxiophyceae) have retained more ancestral, prokaryotic features, characterized by the presence of genes encoding ribosomal proteins, a nearly complete set of tRNAs, and the occasional presence of genes for 5S rRNA and RNase P RNA [120, 349, 350, 355]. Mitochondrial genomes with an intermediate type between derived and ancestral have also been described. In most Sphaeropleales (sister group of the Chlamydomonadales within the Chlorophyceae phylum) and in *Pycnococcus provasolii* (Prasinophyceae), mtDNAs display features from both the reduced and ancestral types [118, 119, 121, 130]. Their size and gene content are not as reduced as that of the Chlamydomonadales, but they lack genes for RNase P RNA, 5S rRNA, and a few tRNAs. In addition, ribosomal genes are fragmented, and gene pieces are scrambled across the genome, as in Chlamydomonadales. The intermediate type has been suggested to represent a transitional stage in the mitochondrial genome streamlining in the chlorophyte lineage [119].

The extensive changes in the reduced and intermediate type of mitochondrial genomes in Chlorophyta have resulted in considerable changes in their translation machinery. Deviations of the genetic code have been noted in some of them (all involving stop codon reassignments),

contrasting with their ancestral counterparts and with land plants where the standard translation code is used. For example, in several Sphaeropleales, the serine UCA and UCG codons have been reported as stop codons [118, 119, 130]. There is further evidence to suggest that UAG (stop) codons are decoded as either alanine or leucine in some chlorophycean mitochondrial genomes [117, 118, 130]. Finally, in *Pedinomonas minor* and *Pycnococcus provasolii* mtDNA, UGA codons are reassigned to tryptophan [121, 356], with the latter genome also using UUG and UUA leucine codons for translation termination. We hypothesize that the list of reported deviations from the standard translation code may be incomplete because all described cases only involve easily identifiable stop codons reassignments. In other words, transitions from one sense codon to another might have gone unnoted, in particular in the context of the remarkably high sequence evolution rates in some lineages. Therefore, we have decided to apply our recently developed tool called CoreTracker for the identification of codon reassignments [116], which has been accurately predicting codon reassignments, including known sense codon changes CUN(Leu → Thr or Ala) in yeast mitochondria [108–110, 357] and AGG(Arg → Ser or Gly) in metazoan mitochondria [72, 74]. Expanding the repertoire of known assignments will help with understanding the underlying complex evolutionary scenarios and corresponding biochemical constraints, including foremost tRNA structures, tRNA synthetase activities, duplication and neo-functionalization of tRNAs and tRNA synthetases, as well as biases in mitochondrial codon usage patterns, allowing reassignments to take place.

The discussion about the genetic and evolutionary mechanisms that explain codon evolution started with the discovery of an alternative code (UGA → Trp) in human and yeast mitochondria [70, 71], followed by a variety of additional codon reassignments in almost all domains of life, especially in animal and fungal mitochondrial genomes (e.g., AGR(Arg → Ser) in most invertebrate, AAA(Lys → Asn) in Platyhelminthes, CUN(Leu → Thr or Ala) in yeast, UAG(Stop → Leu) in several chytrids (see our review in [73]). Current evolutionary scenarios have been reconciled under three main theories. The *codon capture* hypothesis states that codon reassignment arises from the disappearance of a set of codons, subsequently followed by the loss of their cognate tRNA, before a new mutant tRNA evolves to read the codon differently [140, 142]. In contrast, the *ambiguous intermediate* hypothesis postulates that codon reassignments are driven by the existence of an intermediate state in which codons are decoded into different amino acids [144, 145]. On the other hand, the *genome streamlining* hypothesis states that evolutionary pressure on genome size reduction leads to minimization of the translational machinery potentially resulting in codon reassignments [152, 153]. Finally a *tRNA loss driven codon reassignment* hypothesis was recently

proposed [114, 155]. As its name suggests, this hypothesis posits that codon reassignment is primarily driven by mutation or loss of a tRNA or a release factor, such that decoding of the cognate codon is halted before a new tRNA intervenes to decode the codon under a new identity.

In this work, we have undertaken an in-depth study of codon reassignment across 36 Viridiplantae mtDNAs, to gain insight into the evolution of their mitochondrial genetic code. We report, for the first time, sense-to-sense codon reassignment in several chlorophytes mtDNAs. Our results rely on a new conceptual framework for studying genetic code evolution, that explicitly accounts for codon reassignments causing a disparity between DNA and protein sequences, as well as for tRNA evolution through duplications, losses, remolding and structural change. Application of this framework has led us to uncover a complex evolution of the genetic code in the reduced and intermediate mtDNA types within Chlorophyta, with several independent and distinct scenarios, even among closely related clades, leading to sense and stop codon reassignments.

5.3. Methods

5.3.1. Outline of the framework

We have developed a new framework for studying genetic code evolution, extending CoreTracker [116], a method for inferring codon reassignments. CoreTracker evaluates statistically significant differences between nucleotide sequences and expected amino acids in the derived protein sequences, taking conservation at each position into consideration. It differs from similar approaches [325–327] by the fact that it simultaneously handles a set of related genomes in a phylogenetic context, and integrates a validation step ensuring high precision. This new approach was proven more accurate and more flexible than known alternatives, as it allows prediction of codon reassignments, without restriction to any specific phyla or genome type.

The extended framework consists of four complementary modules allowing for both predictions of genetic code alterations and inference of underlying evolutionary scenarios. We present a summary of the framework below. A more detailed discussion of each module, with an illustration on the UGA(Stop $\rightarrow$ Trp) codon reassignment in *Pedinomonas minor,* is provided in section 1 of the supplementary material.

In the first module, candidate codon reassignments are predicted for a set of phylogenetically related genomes, based on comparative sequence analysis of the aligned proteins versus corresponding nucleotide sequences. Since genetic code alteration is tightly linked

to changes in tRNAs, we investigate the evolutionary history of tRNAs in a second step. The main objective is to identify tRNAs with anticodons able of decoding candidate codons for reassignment and to infer their evolutionary history. Evidence of codon reassignment through tRNA identity switch includes (i) histories of multiple tRNA duplications and losses, (ii) structural tRNA remolding, and (iii) mutations in the anticodon loop or acceptor stem which are the main aminoacyl-tRNA synthetase recognition elements.

Although the complete set of mitochondrial-specific tRNA identity rules is largely unknown, major identity determinants and anti-determinants of several bacterial tRNAs are well characterized [138, 139, 358] and seemingly valid for mitochondria [73, 359]. The presence (or lack thereof) of these (anti)-determinants in the secondary structure of considered tRNAs is evaluated in a third step to infer their identity. Further verification by using computational methods specific to the problem such as TFAM [360] is also performed. Finally, results of the three steps, together with information on gene order and ancestral tRNA gene content, are jointly used to infer the evolutionary history of predicted mitochondrial genetic code changes.

5.3.2. Mitochondrial genome dataset

The dataset considered in this study involves 36 complete green plant mitochondrial genomes taken from NCBI (see Supplemental Table S1), including 19 Chlorophyta, and among them, the ten recently sequenced Sphaeropleales mtDNAs [130]. The mitochondrial protein-coding gene dataset was constructed directly from the GenBank genome annotations, except for *Kirchneriella aperta* which required re-annotation using MFannot (http://megasun.bch.umontreal.ca/cgi-bin/mfannot/mfannotInterface.pl), and subsequent manual checking of completeness/correctness of the automated annotations. Translated coding sequences were obtained by considering already proposed stop codon reassignments. We further eliminated the disparity between genomic coding sequences and corresponding protein sequences in land plants, caused by C→ U RNA editing. For this purpose, edited sites were obtained from the REDIdb RNA editing database [361] and NCBI annotations.

5.3.3. Phylogenetic species tree

As accurate clade structure is important for correctly inferring the evolutionary history of code alterations, the phylogenetic species tree topology is based on the current view of green plant evolution [130, 347, 362–364]. For comparison and discussion purposes, a Bayesian tree for Chlorophyta was also constructed from the concatenation of the 13 standard mtDNA-encoded proteins (Cob, Cox1-2-3, Atp6-9, and Nad1-2-3-4-4L-5-6) with PhyloBayes [229],

while taking into account genetic code alterations. For PhyloBayes analyses, we removed constant sites with the -dc parameter and used the CATGTR model with six discrete categories and four independent chains. The chains were run for 10000 cycles, with the first 7000 used as burn-in, ensuring a maximum discrepancy of 0.1. We also built a maximum likelihood tree with RAxML v8.2.11 (LG + Gamma, 100 bootstraps) [217]. The ML tree returned by RAxML and the Bayesian tree displayed the same topology.

5.3.4. Codon reassignment prediction

Prediction of sense-to-sense codon reassignment was performed with CoreTracker [116] using an HMM alignment refinement and the following parameters: –id 0.5 –ic 0.3 –gap 0.4, which correspond respectively to the minimum amino acid identity, the minimum information content and the maximum gap proportion accepted in each column of the alignment. The input submitted to CoreTracker is the phylogenetic species tree, the nucleotide sequences of conserved mitochondrial protein-coding genes, namely those of the respiratory chain complex (Cob, Cox1-2-3, Atp1-4-6-8-9, Nad1-2-3-4-4L-5-6-7-9, and Sdh3), as well as their corresponding translated amino acid sequences. Note that, by using the option "–gap 0.4", any gene missing in more than 40% of the genomes will be automatically removed during analysis.

Stop-to-sense and sense-to-stop codon reassignments were obtained from genome annotations and confirmed by comparing the length of annotated proteins to their homologs, to detect missing C-terminal domains and stop codon read-through.

5.3.5. Transfer RNA analysis

The 859 tRNAs from the 36 green plants mitochondrial genomes were annotated with RNAfinder (http://megasun.bch.umontreal.ca/cgi-bin/RNAweasel/RNAweaselInterface.pl) and confirmed with tRNAscan-SE [365] using parameter -O for organellar tRNAs. Among the predicted tRNA genes, five contained intronic regions which we removed before aligning the sequences. Multiple sequence alignment of tRNAs was done with LocARNA [366] and an in-house script that uses the consensus secondary structure returned by RNAfinder as structural constraints. After manually editing the alignment to remove the hypervariable region, a maximum likelihood (ML) tree was inferred with FastTree v2.1.7 [218] under the GTR+GAMMA model. Bootstrapping was performed for the ML tree with 1000 replicates. Note that the phylogenetic trees constructed from tRNA sequences rely only on few informative positions and therefore do not necessarily exhibit sufficient and accurate resolution of branching order. Nevertheless, they provide correct tRNA grouping in most instances, which is the information required for our purpose of inferring the evolution of the genetic

code. Inside the phylogenetic tree, tRNAs were consistently grouped into large and distinct clades of isoacceptors (tRNAs charging the same amino acid), with only a few exceptions. These clades often received high bootstrap, but the relationship between some isoacceptors inside them could not be confidently determined with the corresponding branches having low bootstrap support (<0.2).

To predict the identity class of some tRNAs of interest, namely those with either a questionable identity or an evolutionary history compatible with codon reassignments, we used TFAM [360]. For this, we first compiled sets of green plants mitochondrial tRNA isoacceptors with unambiguous identity for the following seven amino acids: methionine, leucine, tryptophan, alanine, cysteine, arginine, and tyrosine. These groups correspond to tRNA isoacceptors either sharing the same decoding with a tRNA that has a questionable identity or located in its close vicinity inside the phylogenetic tree. Using TFAM, we first constructed a profile for each of the seven group, then compute a score for every tRNA with a dubious identity against these profiles (see supplemental Table S2)

To confidently assess the phylogenetic placement of the tRNAs of interest, relative to the seven groups, while considering the eventual presence of ambiguous phylogenetic signals, we built a split network. The network was generated with SplitsTree v.4.14.6 [367] using the neighbor-net method. To reduce its complexity, sequence redundancy between tRNAs with trustworthy identity was removed using the CD-HIT suite [368], generating a dataset of 199 representative sequences with <97% identity.

Finally, we built an alignment for each tRNA group and created a covariance model using cmbuild and cmcalibrate from the Infernal 1.1.1 package [369]. tRNAs with dubious identity were aligned against the appropriate covariance models using cmsearch with the "–A" switch. For visualization and editing of sequence alignments, we used Jalview [370] and Inkscape v0.48 (https://inkscape.org). Transfer RNA secondary structure diagrams were created using R2R [371] and further edited using Inkscape v0.48.

5.4. Results

5.4.1. UAG is decoded as alanine and not leucine in *Neochloris aquatica*

Stop codon reassignments have been previously reported in Chlorophyta. In *P.minor*, Turmel *et al.* [356] have shown that UGA codons are decoded as tryptophan by a Trp-tRNA(UCA). The same group later reported the use of UGA for tryptophan in the mtDNA of *P. provasolii* [121]. Due to the absence of the tRNA(UCA) expected to decode the codon, they suggested that it is read instead by the canonical mtDNA-encoded Trp-tRNA(CCA).

From the multiple sequence alignment of the 13 mitochondrial protein, we have recovered these known cases of UGA reassignment to tryptophan (see Table 5. I and Supplemental Figure S1A for an illustration of the sequence analysis on the Nad1 alignment).

From previous analyses, the "Chlorophycean Mitochondrial Genetic Code", a deviant genetic code in which UAG is used as a sense codon, has also been established. This code was determined by Hayashi-Ishimaru *et al.* [117] who showed, by analyzing a multiple sequence alignment of cox1 genes, that UAG codes for alanine in Hydrodictyaceae, whereas it is decoded as leucine in Scenedesmaceae. Decoding of UAG as leucine in Scenedesmaceae, was later corroborated by a thorough analysis of the mitochondrial genome of *Tetradesmus obliquus* [118, 119]. In more recent analyses of Sphaeropleales mtDNA, Fučíková *et al.* have suggested that UAG is decoded as leucine and not alanine in Hydrodictyaceae, but also in *Neochloris aquatica* [130, 372]. Although we did not include any Hydrodictyaceae in our analysis, we have examined *Tetradesmus* and *Neochloris* and confirmed the use of UAG as sense codons in these two genomes.

In *Tetradesmus* we have inferred a UAG(Stop → Leu). As shown in Table 5. I, the tRNA(CUA) cognate to the codon shares several characteristics with Leu-tRNAs, including identity determinants (see Supplemental Figure S2), and was further classified under a Leu-tRNA identity by TFAM (see Supplemental Table S2). On the other hand, both sequences and tRNA analyses unambiguously suggest that UAG is decoded as alanine in *Neochloris* (see Table 5. I), contrasting with the results reported by Fučíková and colleagues. As illustrated on the Nad1 multiple sequence alignment in Figure S1A, UAG is used exclusively by *N. aquatica* in positions where alanine is predominant in other Sphaeropleales. Furthermore, the predicted mt-tRNA(CUA) was classified as Ala-tRNA by TFAM (Supplemental Table S2), and displays all major Ala-tRNA identity determinants, while missing the main Leu-tRNA characteristics such as the elongated variable loop (see Figure 5.3).

5.4.2. Several sense-to-sense mitochondrial codon reassignments in Chlorophyta

Aside from confirming known stop-to-sense codon reassignments, our framework has also uncovered new and previously unknown sense-to-sense codon reassignments in seven green algal mtDNA. From the analysis of the 13 standard mitochondrial genes in the selected 36 Viridiplantae, CoreTracker predicted a total of 12 sense-to-sense codon reassignments, including five AGG(Arg → Ala), three AGG(Arg → Ser), one AGG(Arg → Leu), one AGG(Arg → Met), one CGG(Arg, Leu) and one AUA(Ile → Met) (Figure 5.2). It is noteworthy that all the predicted reassignments are in Chlorophyta, and only in those mitochondrial genomes that have been classified as intermediately derived.

Table 5. I. Predicted codon reassignments in Sphaeropleales and arguments supporting the predictions.

Codon Reassignments	mtDNAs	Predictive Probability	Evidence from tRNA analysis
AGG(Arg→Ala)	*T. obliquus*	0.98	tRNA(CCU) with major Ala-tRNA identity determinants[a] - G3:U70 invariant wobble pair, part of conserved G^1GGC^4 - A73 at the discriminator position TFAM classification [b]
	C. incus	0.57	
	N. aquatica	0.98	
	B. minor	0.66	
	B. aerius	0.99	
AGG(Arg→Ser)	*T. obliquus*	0.79	No support
	B. aerius	0.94	
	B. minor	0.59	
AGG(Arg→Leu)	*C. zofingiensis*	0.96	No support
AGG(Arg→Met)	*C. zofingiensis*	0.98	tRNA(CCU) sharing high sequence and structure similarity with Met-tRNA. TFAM classification
CGG(Arg→Leu)	*C. zofingiensis*	0.99	tRNA(CCG) sharing high sequence and structure similarity with Leu-tRNA. Presence of multiple elements characterizing Leu-tRNAs[c] - long variable arm - A14 in a Reverse-Hoogsteen interaction with U8 - A73 at the discriminator position TFAM classification
ATA(Ile→Met)	*P. provasolii*	0.99	Divergent Met-tRNA(CAU) with potential post-transcriptional modification of C34 to U34
UGA(Stop→Trp)	*P. provasolii*	-	Divergent Trp-tRNA(CCA) with potential post-transcriptional modification of C34 to U34
UAG(Stop→Leu)	*T. obliquus*	-	tRNA(CUA) sharing high sequence and structure similarity with Leu-tRNA Presence of Leu-tRNAs determinants TFAM classification
UAG(Stop→Ala)	*N. aquatica*	-	tRNA(CUA) with major Ala-tRNA determinants TFAM classification
UGA(Stop→Trp)	*P. minor*	-	tRNA(UCA) with Trp-tRNA identity determinants [d] - A1:U72, G2:C71 base pairs - G73 base at the discriminator position - anticodon nucleotides C:35, A:36

[a] [138, 358, 373–375] [b] TFAM respectively predicted a Trp-tRNA and a Met-tRNA identity for tRNA(CCU) in *B. minor* and *T. obliquus*, see Table S2 of the supplementary material [c] [138, 376–378] [d] [379, 380]

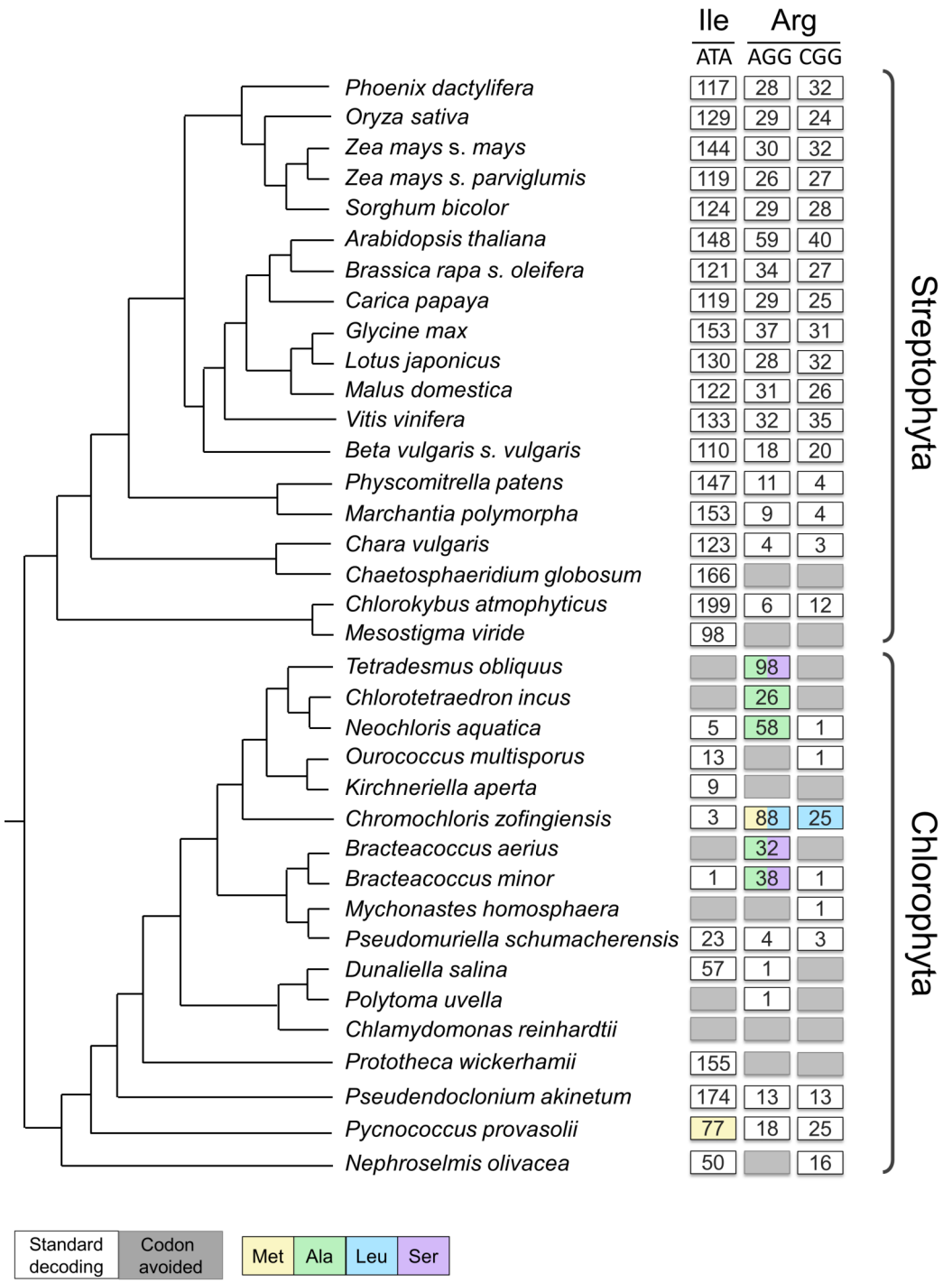


Figure 5.2. Sense-to-sense codon reassignments predicted by CoreTracker in Viridiplantae. The species tree on the left is obtained using the combined information from multiple sources (see methods). Numbers in rectangle indicate, for each mitochondrial genome, the usage of the corresponding codon in all protein-coding genes, ignoring intronic regions. Empty grey rectangles indicate that the codon is entirely avoided in the corresponding genome and colored rectangles indicate the decoding of the codon.

More precisely, except for AUA(Ile → Met) in *Pycnococcus provasolii*, all remaining reassignments involve arginine codons (AGG and CGG) and only occur in Sphaeropleales. These predicted reassignments, and the evidence supporting them, are summarized in Table 5. I.

5.4.2.1. *AGG is not decoded as arginine in most Sphaeropleales.*

In several Sphaeropleales, our framework predicted the reassignment of AGG from arginine to other amino acids. Analysis of AGG codon usage also revealed that the codon is either missing or strictly avoided in most Sphaeropleales mtDNA where it is not reassigned (Figure 5.2). This indicates that AGG reassignment was triggered by a single event preceding the emergence of all Sphaeropleales, which first led to the disappearance of the codon. AGG reassignment is also correlated with the presence of tRNAs with anticodon CCU, only found in Sphaeropleales and not sharing any apparent sequence or structure similarity with the natural Arg-tRNAs found in most chlorophytes mtDNA. Two of these mt-tRNAs(CCU), predicted in the two *Bracteacoccus*, displayed an atypical secondary structure with a reduced D-arm having only 3 nucleotides in the stem and 4-5 nucleotides in the D-loop (Figure 5.3). Due to nucleotide deletion in the D-arm, the reverse-Watson–Crick (WC) base pair 15:48 required in tertiary interactions is seemingly translocated to G14:C43 in *B. aerius* and G14:U43 in *B. minor*. We theorize that both tRNAs are still able to maintain a proper L-shaped tertiary and have conserved their functionalities. In fact, such tRNAs, with a non-standard structure yet fully functional, have been reported numerous times in mtDNA (reviewed in [73] and [359]).

In *Tetradesmus obliquus*, *Chlorotetraedron incus*, *Neochloris aquatica*, *Bracteacoccus minor* and *Bracteacoccus aerius*, CoreTracker predicted the decoding of AGG arginine codons as alanine. The codon was additionally predicted as serine in *T. obliquus*, *B. minor,* and *B. aerius*. Although ambiguous decoding of AGG codons as either alanine or serine is conceivable, as seen with CUG in some yeast nuclear genomes [382], support for AGG(Arg → Ala) is stronger (see Table 5. I). Furthermore, both TFAM classification and secondary structure analysis of tRNAs(CCU) are only in favor of an Ala-tRNA identity (see Table 5. I and Figure 5.3). Further investigation revealed that the standard alanine codons are often found in serine-conserved positions where AGG is used (Figure S4), suggesting that AGG(Arg → Ser) is indeed a false positive that is caused by the frequent substitution of serine by alanine.

In *Chromochloris zofingiensis*, AGG codons are instead predicted to be reassigned to either methionine or leucine with a high probability for both predictions (p=0.981 and p=0.969 respectively). However, only AGG(Arg → Met) is supported by analysis of the secondary structure of the predicted tRNA(CCU) and by TFAM classification (see Table 5. I and

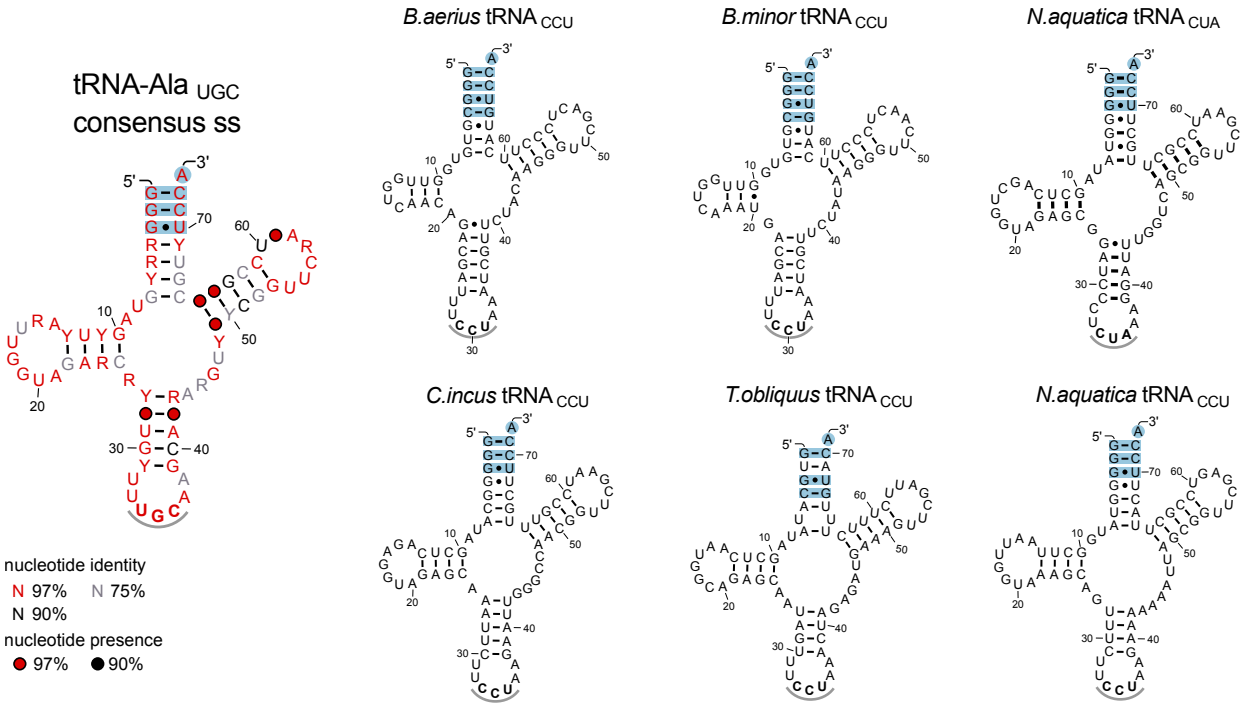


Figure 5.3. The consensus secondary structure of chlorophytes mitochondrial-encoded tRNA-Ala (UGC) is shown on the left. tRNA identity determinants for alanine are indicated by blue boxes and circle. On the right side, it is shown that several Sphaeropleales's tRNA(CCU) and *N. aquatica*'s tRNA(CUA) display multiple Ala-tRNA identity determinants. These results are consistent with the predicted AGG(Arg → Ala) in many Sphaeropleales mtDNA and with UAG(Stop → Ala) in *N. aquatica*. All tRNAs(CCU) are also missing the extra arm that characterizes Ser-tRNA [376, 381], confirming their Ala-tRNA identity.

Supplemental Figure S3). These observations are in favor of AGG(Arg → Met) and hint that, here again, frequent substitutions of leucine by methionine, due to their highly similar properties, could have caused the additional AGG(Arg → Leu) prediction. In fact, the mitochondrial protein sequence alignments revealed that *C. zofingiensis* use methionine in a total of 10 leucine-conserved compared to 20 methionine-conserved positions, implying the absence of strong selection against substitution of leucine by methionine in its genome.

5.4.2.2. *CGG is decoded as Leucine in Chromochloris*

In addition to AGG(Arg → Met), the mitochondrial genetic code of *C. zofingiensis* also differs from other Sphaeropleales mtDNA by the presence of CGG(Arg → Leu), making it one of the rare case in which two previously synonymous codons are reassigned to different

amino acids. The predicted decoding of CGG as leucine is confirmed by the presence of a Leu-tRNA(CCG) whose identity is confidently supported by TFAM prediction, as well as sequence and secondary structure analyses (see Table 5. I and Supplemental Figure S2).

5.4.2.3. *Decoding of AUA as methionine in Pycnococcus provasolii*

In *Pycnococcus provasolii*, AUA codons are predicted to be reassigned from isoleucine to methionine with high support (p=0.988). In fact, in highly conserved positions of the mitochondrial protein alignment, 55.4% of *P. provasolii*'s AUA codons were found in methionine-predominant positions versus only 5.3% in isoleucine-predominant positions. Consequently, we observed a significant improvement of the overall protein sequence alignment (p-value $= 1.23e-06$ for a Wilcoxon's signed-rank test) when all AUA codons were translated as methionine in *P. provasolii* mitochondrial genes. Therefore, sequence analyses strongly suggest that AUA is indeed read as Met in *P. provasolii* mitochondria. Although a tRNA with the exact corresponding anticodon was not found, we have identified a tRNA(CAU) with an uncertain identity that could decode the codon as methionine (please refer to section 5.4.3.5).

5.4.3. Origin of the mutant tRNA in Chlorophyta

As it would be expected when codons are reassigned, none of the mutant tRNAs group with their expected isoacceptors under the standard decoding. They are instead scattered in the tRNA phylogeny, which point out their different origin, and their recent acquisition. Below, we infer the evolutionary history of these tRNAs (see Table 5. III for a summary).

5.4.3.1. *Ala-tRNA(CCU) has distinct origin in Sphaeropleales*

According to our phylogenetic analysis (Figuree 5.4 and Supplemental Figure S4), the two *Bracteacoccus* Ala-tRNA(CCU) cluster with the Trp-tRNA(CCA) group. They also exhibit several conserved positions of Trp-tRNA(CCA) (see Supplemental Figure S5) and the analysis of gene order in both *B. minor* and *B. aerius* (Supplemental Figure S6) showed proximity on the genomic sequence between Ala-tRNA(CCU) and Trp-tRNA(CCA). These results suggest that Ala-tRNAs(CCU) in the *Bracteacoccus* lineage were derived from an ancestral tandem duplication of Trp-tRNAs(CCA), followed by tRNA remolding via the emergence of alanine identity elements. The Trp-tRNA origin hypothesis is further supported by TFAM respectively classifying *B. minor*'s tRNA(CCU) as Trp-tRNA and *B. aerius*'s tRNA(CCU) as Ala-tRNA, which is coherent given their inferred ancestral and current identities.

Figure 5.4. Unrooted phylogenetic network of 199 tRNAs from seven groups (Met, Arg, Tyr, Cys, Ala, Leu, and Trp) constructed using the neighbor-net method as implemented in SplitsTree v4.14.6. Each tRNA group is highlighted with a different color and sequences that do not cluster with their group are displayed using the group color. For clarity, the label of some leaves that cluster with their respective tRNA isoacceptors is not shown. Mutant tRNAs that may have been involved in codon reassignments are shown in white, with a black background.

On the other hand, the predicted Ala-tRNA(CCU) in *C. incus* and *N. aquatica* were found together, grouped with the Ala-tRNA(UGC) cluster in the phylogenetic network (Figure 5.4)

and tree (Supplemental Figure S7), indicating a shared common ancestor. It is likely that the mutant Ala-tRNAs in Neochloridaceae originated from an ancestral duplication of Ala-tRNA(UGC) in the lineage. However, we could not confirm this hypothesis from gene orders because evidence of a potential tandem duplication of Ala-tRNA(UGC) was not found in either of the genomes. It is possible that the duplication event was followed by subsequent mitochondrial genomic rearrangements which are frequent in Sphaeropleales and particularly in Neochloridaceae [130]. After the duplication, the new copy did not diverge much from its precursor, with most sequence mutations located in the anticodon arm changing the anticodon from UGC to CCU, while leaving the acceptor stem almost intact (see Figure 5.3 and Supplemental Figure S8). As alanine identity determinants were thereby conserved, tRNA(CCU) was able to decode AGG codons as alanine in both *N. aquatica* and *C. incus*.

Finally, the Ala-tRNA(CCU) identified in *Tetradesmus* groups with Trp-tRNA(CCA) in the phylogenetic tree (Supplemental Figure S4), whereas it is found inside a cluster containing Trp-tRNA(CCA) and a few Met-tRNA(CAT) in the network (see Figuree 5.4). In fact, Trp-tRNA(CCA) and Met-tRNA(CAT) are grouped in both phylogenies suggesting a common ancestry. Sequence similarity analysis between *T. obliquus* mt-tRNAs indicates that tRNA(CCU) is more similar to Met-tRNA(CAU) than Trp-tRNA(CCA) (62.5% vs 57.8% identity). Furthermore, TFAM also predicted a Met-tRNA identity, although with weak support (see Supplemental Table S2). These results suggest that Ala-tRNA(CCU) in *T. obliquus* originates from an alloacceptor remolding after duplication of Met-tRNA(CAU). The inferred Met-tRNA(CAU) origin of Ala-tRNA(CCU) and Trp-tRNA(CCA) in *T. obliquus* is also concordant with the fact that only one mutation is needed to switch the anticodon from one tRNA to another.

5.4.3.2. *Codon reassignments in Chromochloris are caused by recent tRNA isoacceptor remolding*

In *C. zofingiensis*, besides a Met-tRNA(CCU), a Leu-tRNA(CCG) able to decode CGG codons was also identified. These two tRNAs were respectively grouped with their corresponding tRNA isoacceptors in both the phylogenetic network and tree (see Figure 5.4 and Supplemental Figure S4 and S9). Furthermore, tRNA sequence analyses show an undeniable high similarity between each tRNA and their corresponding cluster (Supplemental Figure S2-S3). As tRNAs having similar sequences and the same anticodon were not found in the mtDNA of any of the related genomes, MLeu-tRNA(CCG) and Met-tRNA(CCU) very likely originated from recent duplications inside *C. zofingiensis* mitochondria. More precisely, Leu-tRNA(CCG) arises from a duplication of Leu-tRNA(UAG) (see Supplemental

Table 5. III. Evolutionary origin of the mutant mt-tRNAs decoding codons under a new identity in chlorophytes.

tRNA	mtDNA	Evolutionary origin
Ala-tRNA(CCU)	*Bracteaccocus*	Tandem duplication of Trp-tRNA(CCA) at early divergence of *Bracteaccocus* + alloacceptor remolding.
	Neochloris *Chlorotetraedron*	Duplication of Ala-tRNA(UGC) in an ancestral mtDNA of Neochloridaceae and isoacceptor remolding
	Tetradesmus	Alloacceptor remolding of possible Met-tRNA copy
Met-tRNA(CCU)	*Chromochloris*	Recent duplication of Met-tRNA(CAU) + isoacceptor remolding
Leu-tRNA(CCG)	*Chromochloris*	Recent duplication of Leu-tRNA + isoacceptor remolding
Ala-tRNA(CUA)	*Neochloris*	Duplication of Ala-tRNA(UGC) and isoacceptor remolding
Leu-tRNA(CUA)	*Tetradesmus*	Recent duplication of Leu-tRNA(CAA) and isoacceptor remolding
Trp-tRNA(UCA)	*Pedinomonas*	Tandem duplication of Cys-tRNA(UCC) and alloacceptor remolding
Trp-tRNA(CCA)	*Pycnococcus*	Divergent tRNA(CCA) of unknown origin
Met-tRNA(CAU)	*Pycnococcus*	Divergent tRNA(CAU) of unknown origin

Figure S2 and S6), followed by mutations in the anticodon. Likewise, Met-tRNA(CCU) originates from a recent isoacceptor remolding of a Met-tRNA(CAU) copy and reads AGG codons as methionine (Supplemental Figure S3).

5.4.3.3. *tRNA isoacceptor remolding allowed decoding of UAG as sense codon in Tetradesmus and Neochloris*

Stop codon UAG decoding in *Tetradesmus* and *Neochloris* is linked to the respective presence of a Leu-tRNA(CUA) and an Ala-tRNA(CUA) in the two mtDNAs. The Leu-tRNA(CUA) of *Tetradesmus* shares several features with other Leu-tRNA (Supplemental Figure S2), and also groups with them in the tRNA phylogenies (Figure 5.4 and Supplemental Figure S9). It is also located directly next to Leu-tRNA(CAA) in the genome (Supplemental Figure S6), suggesting that it emerges via tandem duplication of Leu-tRNA(CAA) followed

by sequence mutation in the anticodon that allowed recognition of UAG codons and their decoding as leucine.

Similarly, Ala-tRNA(CUA) in *Neochloris* shares sequence and structure similarity with the Ala-tRNA(UGC) and was also grouped with them in the tRNA phylogenies (Figure 5.4 and Supplemental Figure S7). In a same vein as with Ala-tRNA(CCU) in *Neochloris*, Ala-tRNA(CUA) likely originates from a duplication of Ala-tRNA(UGC), then accumulates mutations in its anticodon allowing UGA recognition.

5.4.3.4. *Trp-tRNA(UCA) originates from a tandem duplication of Cys-tRNA(GCA) in Pedinomonas*

In *P. minor*, a Trp-tRNA(UCA) decoding the UGA stop codons as tryptophan was predicted. This tRNA groups with the Cys-tRNA(GCA) and the Tyr-tRNA(GUA), found inside the genome, in the tRNA phylogenetic tree (see Supplemental Figure S1B), pointing out their common ancestry. Furthermore, the reduced mitochondrial genome of *P. minor* that is missing several tRNA genes has surprisingly two Tyr-tRNA(GUA) both adjacent to Cys-tRNA(GCA) and the Trp-tRNA(UCA) (see Figure S1D). This observation, taken together with the observed grouping in the phylogeny, suggests that both Tyr-tRNA(UCA) and Trp-tRNA(GCA) originate from a recent tandem duplication of Cys-tRNA(GCA) in the genome. Due to the high similarity that Trp-tRNA(UCA) shares with Cys-tRNA(GCA), TFAM even mistakenly classified it as a Cys-tRNA, but its identity is undeniably Trp-tRNA (see Table 5. I). Accordingly, we suggest that a recent duplication of Cys-tRNA(GCA), followed by novel functionalization of one copy due to relaxed selection, allowed reassignment of UGA codons to tryptophan in *P. minor*.

5.4.3.5. *Divergent tRNAs explain codon reassignments in Pycnococcus*

In *P. provasolii* mtDNA, sequence analysis predicted AUA(Ile → Met) and UGA(Stop→ Trp) codon reassignments. However, a tRNA with the corresponding canonical anticodon was not found in the genome, suggesting that the codons could be read by near-cognate tRNAs. In fact, in the standard code, AUA codons are usually recognized by an Ile-tRNA(CAU) that undergoes a post-transcriptional base modification at the wobble position [383]. To understand the decoding of AUA in *Pycnococcus*, we have undertaken a systematic analysis of its mt-tRNA(CAU).

In most genomes, three tRNA(CAU) with distinct identity (fMet, Met, and Ile) are usually found, but only two were predicted in *Pycnococcus*. Analysis of the phylogenetic network shows that one tRNA(CAU) is inside the green plants Met-tRNA cluster (including

both initiator and elongator), whereas the second is isolated, i.e., outside all well-established tRNA clusters (see Figuree 5.4). Its identity could not be confidently assessed according to the phylogenetic tree either, as tRNAs with anticodon CAU are scattered across the phylogeny, with no consistent groupings. Distinguishing between tRNA(CAU) identity in green plants as Met, fMet or Ile is further complicated by the existence of up to six copies of tRNA(CAU) genes in some lineages.

To determine the correct annotation of each mt-tRNA(CAU) in *P. provasolii*, we built a phylogeny of all tRNAs with CAU anticodon (excluding identical copies), then mapped each tRNA to its predicted identity by TFAM. The genes felt into three distinct groups that were mostly consistent with TFAM annotations (Supplemental Figure S10). Accordingly, one tRNA(CAU) was identified as a Met-tRNA initiator (fMet). However, doubts persist for the second tRNA(CAU) identity. Indeed, TFAM reported it as a Met-tRNA elongator, and there is high bootstrap support (0.914) for its grouping with Ile-tRNA(CAU) and Met-tRNA(CAU) elongator, but a more precise resolution was not obtained. We suggest that this peculiar tRNA(CAU) can decode both AUG and AUA codons as methionine in *P. provasolii*. Note that in all other Chlorophyta mtDNA, except *Pseudomuriella schumacherensis* where Ile-tRNA(CAU) is absent, AUA codons are merely avoided, suggesting that the standard Met-tRNA(CAU) is indded unable to recognize the codon.

P. provasolii mtDNA also exhibits a divergent Trp-tRNA(CCA) that do not cluster with any of the other mtDNA-encoded Trp-tRNA(CCA) in Chlorophyta, highlighting its different origin. Not considering the possibility of tRNA import from the nucleus, it is obvious that this tRNA(CCA) must be able to read its cognate UGG codon, as well as the near-cognate UGA codon via some sequence or structure modification. Indeed, a single base mutation of "G24" to "A24" in the D-arm, has been linked to insertion of tryptophan when reading UGA codons by a Trp-tRNA(CCA) in *E. coli* [384–386]. Because both UGA and UGG codons are almost equally used as tryptophan codons (45% vs. 55% reported in [121]), an efficient decoding of both codon is required. A post-transcriptional conversion of the anticodon 5' base C34 to U34 is a known efficient way that has been reported [73, 122].

5.4.4. Genetic code alterations and Chlorophyta phylogeny

Since both codon and amino acid usage bias have been shown to be responsible for erroneous topology inference [387], bias introduced by genetic code alteration should also be expected. To assess this impact, we have inferred a new Bayesian tree from the adequately

translated mitochondrial proteins sequences (Figure 5.5). As expected, some topological differences, mainly regarding the position of genera considered as *incertae sedis* within Sphaeropleales (Bracteacoccaceae, Mychonastaceae, Pseudomuriellaceae, Chromochloridaceae) were observed. In this new phylogeny, most chlorophytes groups were successfully recovered as monophyletic with several branches receiving equal or increased support.

Outside of Sphaeropleales, the observed branching is identical to the one reported in previous mitochondrial trees, with *Prototheca* diverging earlier than *Pycnococcus* [121]. *Pycnococcus* not affiliating with the other prasinophytes in phylogenies based on mitochondrial proteins is a well known artefact [121, 388, 389]. It appears that this artifact was not caused by the usage of a wrong genetic code, but is a result of the rapid mitochondrial sequence evolution in *Pycnococcus*, compared to the other investigated prasinophycean [121]. The inclusion of additional deep branching lineages is therefore required to understand the diversity of prasinophytes and their evolution.

Regarding the branching inside Sphaeropleales, we have recovered the Selenastraceae genera (*Monoraphidium*, *Ourococcus*, and *Kirchneriella*) as monophyletic and sister to Scenedesmaceae + Neochloridaceae, consistent with previous reconstructions based on the nuclear 18S rRNA and chloroplast genes [390–392]. The phylogeny subdivizes Sphaeropleales into two main lineages: (*Mychonastes*, *Bracteaccoccus*) + (*Pseudomuriella*, *Chromochloris*) and the group containing Selenastraceae + Scenedesmaceae + Neochloridaceae. Although the grouping of *Mychonastes* + *Bracteaccoccus* with *Pseudomuriella* + *Chromochloris* is weakly supported, it can be expected as, excluding *Mychonastes*, all the above-named species display the same cellular organization and are almost morphologically indistinguishable in such a way that they have been described as cryptic species [6]. It was hypothesized that their shared characteristics might be monophyletic within Sphaeropleales [6, 393], but phylogenetic reconstructions either only showed weak support or failed to recover it [130, 372]. Our new phylogeny using corrected protein sequences mainly corroborates that hypothesis, but also includes the morphologically distinct *Mychonastes* inside the clade. Placement of the latter has been challenging, with various unsupported and conflicting positions inferred in previous phylogenetic analyses [390]. There has been evidence suggesting that *Mychonastes* belong to the earliest-diverging clade of the Sphaeropleales, but more chlorophycean data and additional studies are required for the precise elucidation of its phylogenetic position.

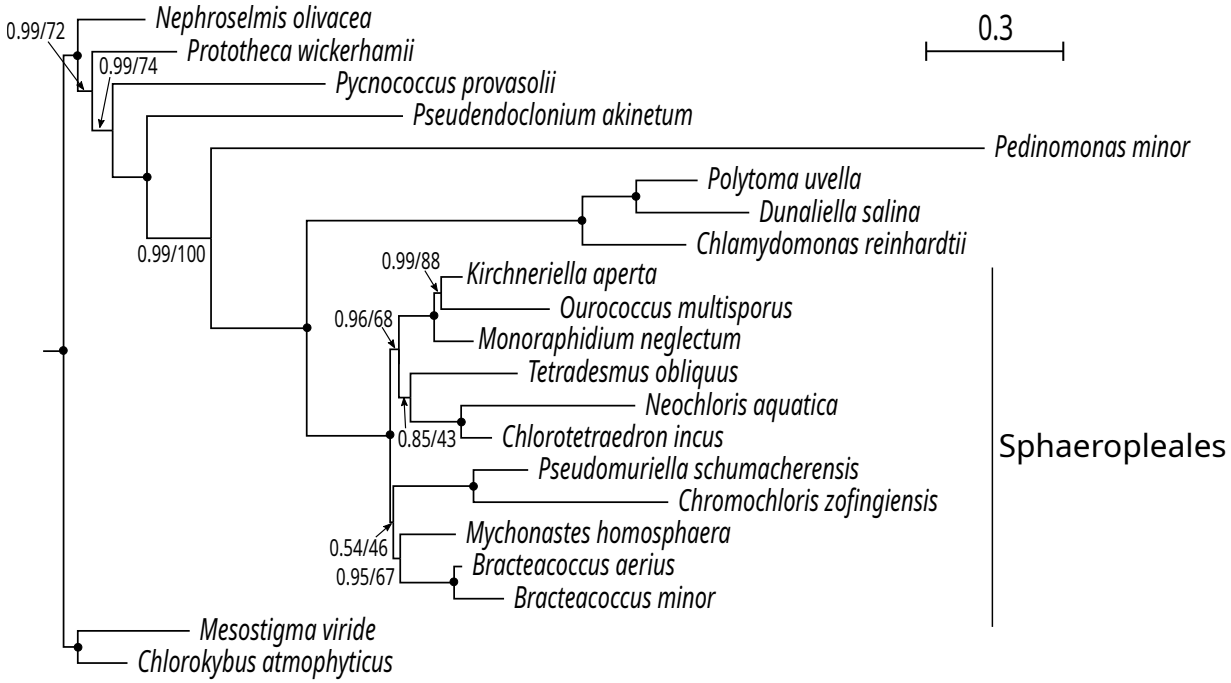


Figure 5.5. Phylogeny of 21 core chlorophytes inferred using a supermatrix built from the properly translated sequences of the 13 standard mitochondrial proteins. The tree presented here is the majority-rule posterior consensus tree inferred with PhyloBayes under the CAT-GTR + Γ model. Both posterior probability (PP) from the Bayesian analyses (left value) and bootstrap support from RAxML LG + Γ maximum likelihood reconstruction (right value) are shown. We use a black dot on an internal node to indicate that its parental branch received both a PP value of 1.0 and a bootstrap support greater than 95%. Branch lengths are shown and correspond to the estimated number of amino acid substitutions per site.

5.5. Discussion

5.5.1. Mitochondrial genetic code evolution in Chlorophyta facilitated by genome minimization

One of the most interesting outcomes of our study is the observation that changes in the genetic code only occurred in mitochondrial genomes that have undergone significant reduction (Sphaeropleales, *Pycnococcus,* and *Pedinomonas*), suggesting that reduced mitochondrial size could facilitate fixation of codon reassignments. The same conclusion was reached by [106] who uncovered a negative correlation between the mitochondrial proteome size and the number of observed genetic code alterations. They proposed that reduced "proteomic constraints" due to genome minimization allow changes in the genetic code to be more tolerated, and less likely to be lethal. This hypothesis differs from the genome streamlining

model of Kurland and colleagues [152, 153], in which shrinkage of the tRNA repertoire is the driving force for codon reassignments. Although we agree that circumstances provided by the reduced proteome size increase the likelihood of codon reassignments, we argue that a combination of several additional factors (e.g., codon usage patterns, GC fluctuation, genetic drift, gain/loss of tRNAs) is also needed. The importance of such factors is illustrated by the complete absence of codon reassignment in the otherwise severely reduced mitochondrial genomes of Chlamydomonodales.

5.5.2. Polyphyly of AGG decoding is driven by loss of ancestral tRNA(UCU) and codon disappearance

Our analyses suggest an independent reassignment of AGG codons during mitochondrial genome evolution of Sphaeropleales. We have indeed inferred a distinct evolutionary origin for each Ala-tRNA(CCU) in Scenedesmaceae, Neochloridaceae, and *Bracteacoccus*, whereas we have uncovered an entirely different decoding of the codon as methionine in *Chromochloris*. Since the clade comprising Selenastraceae + Scenedesmaceae + Neochloridaceae was recovered with high statistical support, and no alternative decoding of AGG codons was found in Selenastraceae, the polyphyly of AGG decoding seems supported. A similar polyphyletic decoding of CUG codons was reported in yeast nuclear genomes where the codon is reassigned from leucine to serine or alanine in some species [114, 115, 137]. The alternative decoding of CUG codons in these nuclear genomes was linked to the loss of the Leu-tRNA(CAG) decoding the codon [113, 155].

The reassignment of AGG codons in Sphaeropleales also appears to be correlated with the loss of a tRNA. Indeed, the ancestral Arg-tRNA(UCU) normally decoding AGR codons was lost in all Chlorophycean (Sphaeropleales and Chlamydomonodales) (Figure 5.6). It is, however, important to emphasize that the loss of tRNA(UCU) was not the sole condition for the reassignment. In angiosperms mtDNA for example, the tRNA(UCU) is also lost (see Figure 5.6), but rather than reassigning the corresponding codons, a new Arg-tRNA is imported from the nucleus [394–396].

AGG reassignment in Sphaeropleales cannot be explained by ambiguous decoding. Indeed, under this model, the codon would need to be simultaneously assigned to two tRNAs in the common ancestor of chlorophycean, before the reassignment occurred. However, this would require the independent loss of tRNA(UCU) in Sphaeropleales and Chlamydomonodales. Furthermore, because AGG codons are decoded by multiple extant tRNAs with distinct origins, one has to assume either the presence of all these tRNAs at once or transitions

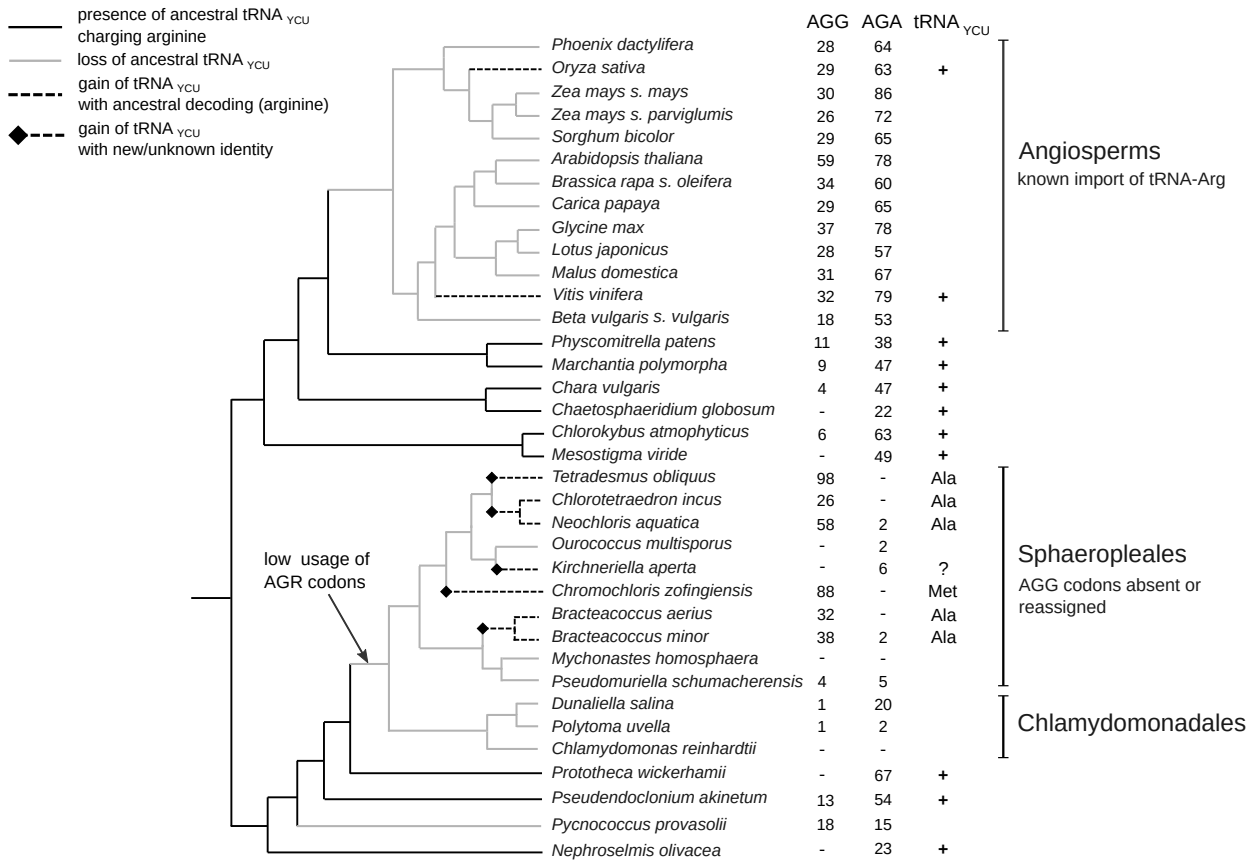


Figure 5.6. Evolutionary history of tRNA(YCU) in green plants. The usage of AGR codons is shown on the first two columns, with a dash used to indicate when the codon is missing in the coding regions of the considered genome. The presence/absence of tRNA(YCU) in each genome is also shown (third column). Moreover, the gain and loss history of tRNA(YCU) in each lineage during their mitochondrial evolution is indicated on the phylogeny.

through successive ambiguous decoding states. Neither of these scenarios seems parsimonious, as they require several tRNA losses in each lineage.

In contrast, under the codon capture hypothesis, the reassignment would need to be preceded by the disappearance of AGG codons in all Sphaeropleales. The AT-rich (53-70%) mitochondrial genomes of Chlorophytina (containing Chlorophyceae, Ulvophyceae, and Trebouxiophyceae) actually exhibit a preference for AGA over the synonymous AGG codon. In most chlorophycean, where reassignment of AGG was not predicted, the codon is either missing or altogether avoided. The additional absence of AGG in *Prototheca*, suggests that the disappearance of the codon could have predated the loss of tRNA(UCU), as required by the codon capture model. However, tRNA(UCU) translates the AGR block, and if it were deleted before AGA disappearance, there would likely be no alternative tRNA to translate

AGA codons. To avoid disrupting mRNAs translation, the codon capture hypothesis would require liberation of the whole AGR codon family before the reassignment. Severely reduced usage of AGA codons in Sphareropleales did indicate that AGA ultimately vanished before AGG reassignment, but the timing of that disappearance relative to the loss of tRNA(UCU) could not be established. We favor the hypothesis that the genome minimization process in chlorophycean might have not only caused the loss of tRNA(UCU) but also contributed to AGA usage reduction in a way minimizing the impact on structural and functional protein integrity. In fact, more than a third of AGA codons in the mitochondrial coding regions of *Prototheca* and *Pseudendoclonium*, fall into genes that are missing in Sphaeropleales mtDNA.

After the disappearance of AGG codons and loss of the cognate tRNA, new mutant tRNA(CCU)s with high specificity to AGG codons emerged, via tRNA duplication and remolding, in some Sphaeropleales. These new tRNAs were able to decode AGG under a new identity when it later reappeared. Interestingly, most of these tRNAs(CCU) have gained an Ala-tRNA identity, demonstrating a remarkable example of evolutionary convergence. The gain of the new tRNAs before the reappearance of the codon is supported by the presence of divergent tRNA(CCU) and tRNA(UCU) in *K. aperta* and *T. obliquus*, even though the cognate AGG and AGA codons are respectively missing in each genome.

5.5.3. Gain of a new tRNA caused CGG codon reassignment in *Chromochloris*

In *C. zofingiensis*, CGG codons are decoded as leucine by a Leu-tRNA(CCG) that originates from the isoacceptor remolding of a Leu-tRNA. This reassignment follows a mechanism different from the synonymous AGG codon reassignment. Although avoidance of CGG codons is observed in all Sphaeropleales, perhaps due to a high AT pressure and reduced specificity of the codon towards the Arg-tRNA(ACG) decoding the CGN block, the complete disappearance of the codon before its reassignment is not required. The codon capture model would also need the loss of the ancestral Arg-tRNA(ACG), which remains present in all Sphaeropleales. We propose that the reassignment occurred due to the gain of a new Leu-tRNA(CCG) with higher anticodon specificity for CGG codons. There could have been a transitional period of decoding ambiguity during which the codon was read by both Leu-tRNA(CCG) and Arg-tRNA(ACG). This decoding ambiguity would, however, had minimal impact on mRNA translation due to the low usage of CGG during that time. Upon relaxation of the mutation bias, allowing increased usage of the codon, CGG codons were fully captured by the new tRNA.

5.5.4. Mutant tRNAs decoding AUA and UGA in *Pycnococcus*

In *Pycnococcus*, we predicted an AUA(Ile → Met) and confirmed the UGA(Stop → Trp) previously identified by [121]. These reassignments are the most observed genetic code modification in mitochondrial genomes. AUA(Ile → Met) has been described in the mitochondrial genome of yeast, metazoa and Xanthophyceae [72, 107, 397]. Likewise, UGA(Stop → Trp) was reported in many mitochondrial genomes and some non-mitochondrial systems [106].

In the standard code, AUA is translated as isoleucine by an Ile-tRNA(K2CAU) with the cytidine at the wobble position modified to lysidine (K2C) [383, 398]. This modification allows Ile-tRNA(K2CAU) to pair with AUA, unlike the standard tRNA-Met(CAU) which only pairs with AUG. In all reported genomes in which AUA is reassigned to methionine, the canonical Ile-tRNA(K2CAU) is absent. Instead, a Met-tRNA(CAU) deciphering both AUA and AUG, often through some post-transcriptional modifications, is found. For example, it has been shown that modification of anticodon base C:34 to f5C:34 (5-formylcytidine) is essential for AUA recognition in most animal mitochondria [399–402]. UGA, on the other hand, is typically recognized by a release factor signaling the termination of translation. In most genomes where it is reassigned, a Trp-tRNA with anticodon UCA, in place of the standard CCA, is found. This tRNA is able to decode the UGR block as tryptophan. In some mitochondrial genomes (*Amoebidium*, *Crinipellis,* and *Schizophyllum*, kinetoplastids), the Trp-tRNA has retained the standard CCA anticodon and is assumed to be posttranscriptionally modified to translate UGA [72, 122].

In contrast with its closest relative (*Prototheca*, *Nephroselmis*, and *Pseudendoclonium*) where all three Met-tRNA(CAU), fMet-tRNA(CAU) and Ile-tRNA(CAU) are found, only the fMet-tRNA(CAU) and a divergent tRNA(CAU), visibly acquired recently are present in *Pycnococcus*. In support of the AUA(Ile → Met), it can be argued that the ancestral mtDNA-encoded Ile-tRNA(CAU) was either lost or possibly transferred to the nucleus, alongside other genes, during the mitochondrial genome streamlining process in *Pycnococcus* [121]. While Turmel *et al.* [121] suggested that AUA is decoded by a nuclear- or chloroplast-encoded Ile-tRNA, we conjecture instead, that the codon is decoded conjointly with AUG as methionine by the mutant tRNA(CAU). Given the high AT-content of *Pycnococcus* mtDNA and the frequent usage of AUA codons in its relatives, it is unlikely that AUA vanished before the loss of Ile-tRNA(CAU). Therefore, the codon capture mechanism cannot explain this reassignment. Some have suggested that AUA(Ile → Met) is usually initiated by the loss of Ile-tRNA(CAU) [72] and not by the disappearance of the codon. We favor this hypothesis for *Pycnococcus*. After the loss of Ile-tRNA(CAU), it is possible that translation

of AUA codons, was ensured by Ile-tRNA(GAU), albeit inefficiently [107] before the mutant Met-tRNA(CAU) was acquired.

In contrast with AUA reassignment, UGA (Stop → Trp) is entirely compatible with the codon capture mechanism. UGA codons are effectively missing from the mtDNA of *Nephroselmis*, *Ostreococcus*, *Prototheca*, where the AT-rich UAA stop codon is preferred. We suggest an evolutionary scenario in which UGA first disappears from the genome due to AT-pressure, followed by loss of the release factor's ability to recognize it. In parallel, a mutant Trp-tRNA(CCA) able to translate both UGA and UGG codons was acquired. Upon, re-appearance of UGA in the genome, the codon was then decoded as tryptophan. Reintroduction of UGA could have been facilitated by synonymous mutations of UGG to UGA, due to the sustained AT pressure that caused UGA disappearance [72].

It is unknown at which point during evolution the mutant tRNA(CAU) and tRNA(CCA) were acquired and by which mechanisms they decode non-cognate codons. We suspect that both mutant tRNAs were simultaneously acquired from the same source. Evidence of novel genes acquisition by gene transfer have been previously reported in some Chlorophyta [403] and the same could have occurred in *Pycnococcus*. In fact, principal component analysis and clustering of codon usage pattern in coding regions revealed a strong mitochondrial codon bias in *Pycnococcus*, not resembling any other chlorophytes mtDNA. An alternative scenario would assume a post-transcriptional modification of the two tRNAs. Because Turmel *et al.* [121] have reported that RNA editing does not occur in *Pycnococcus*, the most likely scenario remains a modification similar to Met-tRNA(f5CAU) in animals. Indeed, in *Pycnococcus* mtDNA, tRNA(CCA) and tRNA(CAU) are the only tRNAs with a cytidine at the wobble position of the anticodon and therefore, could both be modified by a post-transcriptional mechanism targeting the C:34 nucleotide. These explanations remain rather speculative. To understand how these tRNAs are able to decode non-canonical codons, RNA sequencing, including all potential modifications, of tRNAs and *in vivo* characterization of tRNA synthetases activities are required.

5.5.5. Stop codon capture in Chlorophyta

Similar to other known stop codon reassignments in metazoan mitochondria [72], UAG reassignment in *N. aquatica* and *T. obliquus* follows a codon capture mechanism [72, 336]. Since stop codons are initially rare, their disappearance is conceivable. In particular, besides UAA, all standard stop codons, namely UGA and UAG, are missing in Sphaeropleales, with UCA and UCC acting as translation termination signals instead [119, 130]. With the independent emergence, via isoacceptor remolding, of new tRNAs with anticodon CUA, the

codon was fully captured at its reintroduction, by Leu-tRNA(CUA) in *Tetradesmus* and by Ala-tRNA(CUA) in *Neochloris*.

UGA reassignment to tryptophan in *Pedinomonas* follows a similar mechanism, with a few differences. Because the standard Trp-tRNA(CCA) is missing, it is likely that the new Trp-tRNA(UCA) is able to decode both UGA and UGG codons as tryptophan, in accordance with the "U:R wobble" rule (e.g., [404, 405]). However, UGG codons only account for 3% of tryptophan positions in *Pedinomonas*, contrasting with *Pynococcus* where both codons are used in equal frequency. Nevertheless, the high AT mutation pressure in the genome and the absence of UGA in its close relatives [120], indicates a codon capture mechanism. Although disappearance of UGA codons was propitious for the reassignment, the critical factor was still the loss of several genes, including Trp-tRNA(CCA), during the severe mitochondrial genome reduction in *Pedinomonas*. To compensate for this loss, and ensure continued translation of genes, chloroplast or nuclear-encoded tRNAs could have been imported. Since tRNA import is not believed to be a dynamic process that could automatically adapt to new changes in the mitochondrial genome [406], it is possible that a Trp-tRNA was not efficiently imported to translate UGG codons, progressively driving them out of the genome. This is supported by the sporadic usage, in *Pedinomonas*, of other amino acids at conserved tryptophan positions (see Supplemental Figure S1A for an example in the Nad1 gene). The subsequent gain of a new Trp-tRNA(UCA) through tRNA remolding of a copy of Cys-tRNA(GCA), would later allow decoding of UGA as tryptophan upon re-introduction of the codon in the genome.

5.5.6. Usage of the appropriate genetic code might improve phylogenetic inference of Chlorophyta

Despite the increasing number of sequenced chlorophytes genomes, their evolution remains poorly understood. Recent studies have struggled to resolve the phylogenetic relationships among deep lineages [130, 364, 390, 407–409]. In particular, the evolution of the morphologically simple and similar, yet genetically divergent, Sphaeropleales is still not well-understood [6, 130, 390, 392, 409]. Previous attempts at inferring a phylogenetic tree for Sphaeropleales have revealed conflicting signals between mitochondrial, nuclear and chloroplast data, with the mitochondrial protein-based tree displaying stronger inconsistencies toward the other two [130, 390]. Similar erroneous phylogenetic placements, as a result of long-branch attraction, have also been observed for other chlorophytes when concatenated mitochondrial data are used [119, 120, 410]. These errors have mainly been attributed to systematic errors of phylogenetic reconstructions and the fast substitution rate of sequence evolution in chlorophytes mtDNA [364]. However, considering our new results, topology

incongruence could have also stemmed from undetected sense-to-sense codon reassignments introducing biases in the inferences. Indeed, the phylogenetic tree inferred using correctly translated mitochondrial proteins seems more in agreement with previous reconstructions based on nuclear and chloroplast data. Because the site-heterogeneous CAT-GTR model we used with PhyloBayes is useful in coping with long-branch attraction [411], it is hard to solely attribute artifacts to the usage of an incorrect genetic code, but the potential biases it can introduce should not be overlooked. This emphasizes, yet again, the importance of correctly identifying genetic code alterations, before any genomic analysis.

5.5.7. Why are sense-to-sense codon reassignments only present in the chlorophytes with an intermediate type of mtDNA ?

An important characteristic of the predicted deviations of sense codon decoding in chlorophytes is that they are only located in mtDNA which are thought to represent an intermediate stage during the streamlining process toward a reduced mitochondrial genome (see Figure 5.7). The observed great disparity in gene decoding between Sphaeropleales and Chlamydomonadales, their sister group that conserved the standard decoding, is astounding and suggests the existence of a link between the "intermediate state" of mtDNA in Sphaeropleales and their genetic code evolution.

The mechanisms responsible for mtDNA streamlining in chlorophytes are still not very well understood [119]. It was suggested that streamlining occurred because of competition among mtDNA molecules for faster replication time and that it promotes the transfer of genes to the nucleus [412]. Recent genome analyses have revealed abundant evidence for frequent recombination of mitochondrial genes into nuclear chromosomes, often to escape the higher rate of mildly deleterious mutations in the mitochondrial compartment [413–415].

Migration of several mitochondrial genes to the nuclear genome was also reported in Chlamydomonadales [416–419] and to a certain extent in *Tetradesmus* [416, 420, 421]. These observations have led to the current assumption that the mitochondrial streamlining process started in the chlorophycean ancestor of Sphaeropleales and Chlamydomonadales, but was later halted, for some reasons, in Sphaeropleales [119, 420, 422]. In light of our results, we propose that genetic code deviations in the mtDNA of Sphaeropleales could have contributed to their unusual genomic organization. Indeed, genes transferred to the nuclear genome are expected to be decoded by nuclear tRNAs then imported back into the mitochondria where they are functionally active. Decoding differences between the two compartments would, therefore, result in nonfunctional protein products, and can potentially be lethal. It is likely that the first genetic code alteration in the Sphaeropleales lineage was the reassignment

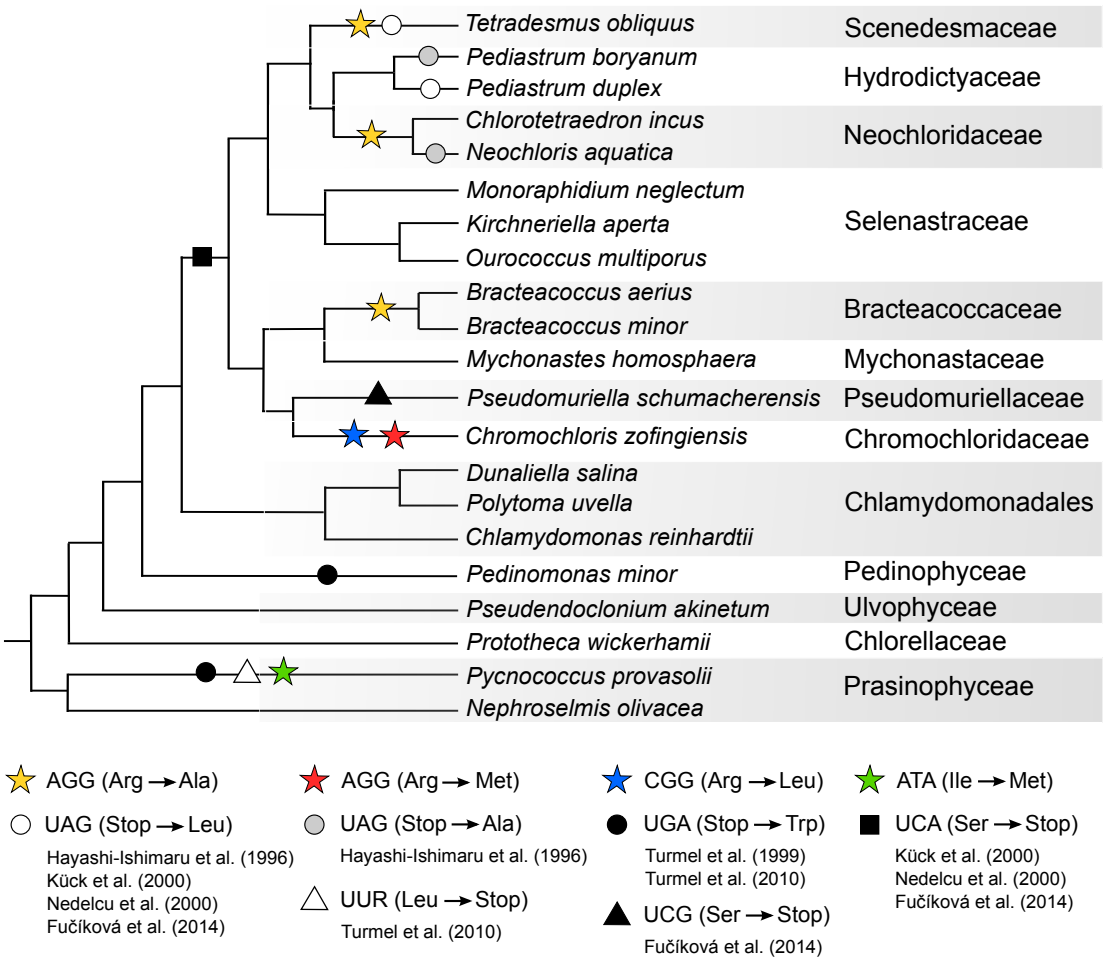


Figure 5.7. Known mitochondrial genetic code alteration in Chlorophyta.The root of the phylogeny assumes the standard genetic code and changes in decoding are displayed on branches of the phylogeny.

of serine UCA codons to stop, which appears to be shared by all Sphaeropleales [130]. Mitochondrial genes migration to the nucleus would have then been forcibly stopped, due to the alteration preventing correct translation of transferred genes. The resulting "intermediate type mtDNA" with reduced proteome size and already altered tRNA repertoire (e.g., loss of the Arg-tRNA(UCU)) would have therefore facilitated further changes in the genetic code, leading to the reassignment currently observed.

A similar scenario could also explain the predicted code alteration in *Pycnococcus*. However, analysis of additional chlorophytes mitochondrial genomes is required for a full understanding of the link between streamlining and genetic code alterations.

5.5.8. Concluding remarks

The evolution of chlorophytes mtDNA is characterized by the presence of multiple codon reassignments independently occurring in distinct lineages. Our results unveiled a significant variation in their mitochondrial genetic code, even between closely related species (Figure 5.7). Similar demonstrations of the mitochondrial genetic code evolving differently, and in parallel, in close genomes were previously given in yeast [110] and metazoan [133]. These results highlight why, despite the rarity of codon reassignments, it is crucial to determine the proper decoding in fast-evolving genomes and not assume it based on the code used in its relatives.

Our study also has brought forth an unexpected tolerance of a high number of deviations in chlorophytes mitochondrial genomes, hinting the existence of a tremendous diversity of mtDNAs in green algae and eukaryotes in general. Although the present study provides new information for understanding the mitochondrial genome evolution in chlorophytes, exploration of the mtDNA evolution of additional chlorophytes species, especially the early-diverging members, is still needed to gain a more general understanding and decipher the mechanisms responsible for such diversity.

Chapter 6

Efficient Gene Tree Correction Guided by Genome Evolution

Ce chapitre présente un travail publié dans *PLoS ONE* réalisé avec nos collaborateurs français de l'Université Lyon 1. Il décrit un pipeline de correction d'arbres de gènes qui comprend ProfileNJ , un algorithme rapide et déterministe ciblant les noeuds faiblement supportés ou non binaires des arbres de gènes. Le matériel supplémentaire de ce chapitre est disponible à l'annexe C.

Dans la section 3.4, nous avions énuméré quelques stratégies de correction d'arbre de gènes qui tirent avantage de la réconciliation entre arbre de gènes et arbre d'espèces. L'une de ces stratégies consiste à contracter les branches non supportées de l'arbre de gènes puis à résoudre les polytomies ainsi générées de façon à minimiser le score de réconciliation. Dans [29] (voir annexe E), nous avions décrit PolytomySolver, un algorithme de résolution de polytomies sous le modèle d'évolution DL, qui a une complexité linéaire, la meilleure obtenue pour ce problème. ProfileNJ est entièrement basée sur cet algorithme. La résolution de polytomies ne permet cependant pas de distinguer entre les copies de gènes dupliqués, ce qui fait que l'information sur les séquences est perdue à ce niveau. Pour pallier ce problème, ProfileNJ considère une contrainte supplémentaire et l'utilisation de l'algorithme *Neighbour Joining*(NJ) afin de choisir les bonnes copies de gènes, à chaque étape de la résolution. Nous montrons que de cette contrainte additionnelle améliore grandement la qualité des arbres corrigés. Un sommaire du fonctionnement de la méthode est fourni ci-après :

— **entrée:**

1. Un arbre d'espèce S binaire.
2. Un arbre de gènes G, potentiellement non enraciné et non binaire, mais possédant idéalement des supports statistiques sur chaque branche.

3. Une matrice de distance obtenue à partir de l'alignement des séquences de gènes présents dans G. Alternativement, si les longueurs de branches sont connues pour G, alors la matrice n'est pas requise.
4. Un seuil de support.

— **sorties:**

L'algorithme retourne un arbre corrigé de gènes dans lequel toutes les branches de l'arbre en entrée ayant un support au-delà du seuil choisi sont conservées, tandis que tous les sous-arbres se retrouvant sous une branche avec un support inférieur, correspondent à des résolutions minimisant le score pondéré de duplications et pertes. Par ailleurs, parmi toutes les résolutions possibles pour chaque sous-arbre, on choisit celles dont les distances induites entre gènes se rapprochent le plus de celles de la matrice en entrée (approximation faite avec NJ). ProfileNJ est capable de retourner toutes les solutions optimales partageant le même score de réconciliation.

La méthode présente deux avantages principaux : (1) elle est extrêmement rapide, ce qui permet de corriger toute la base de données Ensembl Compara (20519 familles de gènes au total) en quelques heures (~4h, pour 20 tâches simultanées, chacune utilisant 8 coeurs) sur des machines disposant d'un processeur Intel Xeon X5650 Westmere (2.67 GHz) et 24GB de RAM, disponibles sur le serveur Briarée de Calcul Québec; (2) elle ne modifie que les sous-arbres faiblement supportés par l'alignement de séquences. Cependant, sous certaines conditions, ce second avantage peut devenir une des faiblesses de ProfileNJ .

En premier lieu, le choix du seuil adéquat pour décider si une branche est supportée ou non n'est pas simple, d'autant plus que les différents supports (bootstrap, aLRT, probabilités *a posteriori*, etc.) ne communiquent pas forcément la même information. Des discussions au sein de la communauté scientifique persistent à cet égard, avec des recommandations variant entre 70 et 95%. Dans le cas de ProfileNJ , nous utilisons un seuil conservateur de 90% par défaut. Le problème majeur vient toutefois du fait qu'un support statistique élevé pour un sous-arbre ne veut pas dire que ce dernier reflète la vraie histoire évolutive de la famille de gènes. Il peut arriver que des relations erronées soient supportées par l'information des séquences et ne seront donc pas corrigées par ProfileNJ . Par ailleurs, les corrections locales effectuées par ProfileNJ ne garantissent pas que la vraisemblance de l'arbre final soit optimale.

Une solution potentielle à ces problèmes, que nous avons utilisée pour une collaboration publiée dans *Scientific Reports* [423] (voir annexe F), consiste à tester plusieurs seuils différents, en considérant toutes les résolutions de ProfileNJ , pour différents enracinements. Parmi toutes ces résolutions, nous ne considérons ensuite que celles qui sont statistiquement équivalentes à l'arbre ML selon un test AU. Ce procédé permet d'obtenir des arbres avec

une vraisemblance comparable à celle du meilleur arbre ML, mais qui sont meilleurs en ce qui concerne le score de réconciliation. Cette approche rappelle celle de TreeFix [284], mais la différence au niveau du temps d'exécution reste énorme et en faveur de ProfileNJ .

Bien que la seule autre méthode intégrative à laquelle nous avons comparé ProfileNJ soit TreeFix, d'autres méthodes telles que ecceTERA [295], qui n'était pas disponible à la soumission du manuscrit, et ALE [26] existent. Ces méthodes supportent, en plus des duplications et pertes de gènes, les évènements de transferts. Toutefois, contrairement à ProfileNJ et TreeFix, elles n'utilisent pas une stratégie de correction *a posteriori* des arbres de gènes, qui tente de préserver l'information des séquences. En particulier, ALE utilise l'amalgamation d'arbres et nécessite donc au préalable un échantillonnage des topologies d'arbres de gènes à partir de méthodes bayésiennes d'inférence d'arbres comme PhyloBayes et MrBayes.

Dans ce travail, nous proposons également un pipeline de correction d'arbres de gènes nommé RefineTree qui inclut ProfileNJ , ainsi que deux autres méthodes précédemment décrites [424, 425]. La première, appelée ParalogyCorrector, corrige les arbres de gènes en se basant sur des contraintes d'orthologie, *c.-à-d.* des paires d'orthologues déjà connues, pour construire un nouvel arbre avec une topologie similaire à celle de l'arbre en entrée tout en respectant les contraintes définies. La seconde, appelé Unduplicator, utilise l'information des synténies ancestrales inférée avec Deco [281] pour corriger les duplications douteuses causant des conflits d'adjacences. Nous montrons que l'application de ces méthodes de correction produit des arbres bien meilleurs que celle de la base de données Ensembl.

Contributions: Emmanuel Noutahi et Magali Semeria ont contribué de manière équivalente au projet décrit dans le manuscrit. L'étude et l'ensemble des expérimentations effectuées ont été conçues par Emmanuel Noutahi, Magali Semeria, Manuel Lafond, Bastien Bousseau, Nadia El-Mabrouk et Eric Tannier. ProfileNJ a été développé par Manuel Lafond, Nadia El-Mabrouk, Eric Tannier et Emmanuel Noutahi. Emmanuel Noutahi, Magali Semeria et Eric Tannier ont produit les résultats présentés dans le manuscrit. En particulier, Emmanuel Noutahi a implémenté ProfileNJ , l'a testé, puis a mesuré et comparé les performances de ProfileNJ , RAxML et TreeFix sur les données simulées. Magali Semeria, avec la participation de Eric Tannier, a produit les résultats basés sur la synténie et a appliqué ProfileNJ sur la base de données Ensembl. Jonathan Séguin a intégré les différents outils dans une application web (RefineTree). Les deux autres méthodes présentes dans RefineTree (ParalogyCorrector et Unduplicator) avaient déjà été décrites dans un travail précédent effectué par Manuel Lafond, Magali Semeria, Eric Tannier et Nadia El-Mabrouk. Laurent Guéguen, Bastien Bousseau et Nadia El-Mabrouk ont contribué à l'analyse des résultats. L'article a

été principalement rédigé par Eric Tannier et Nadia El-Mabrouk, avec la participation de Emmanuel Noutahi, Manuel Lafond, Magali Semeria et Bastien Boussau.

Efficient Gene Tree Correction Guided by Genome Evolution.

PLoS ONE 2016 **11**(8):e0159559 doi:10.1371/journal.pone.0159559

Emmanuel Noutahi [1], **Magali Semeria**[2], **Manuel Lafond**[1], **Jonathan Seguin**[1], **Bastien Boussau** [2], **Laurent Guéguen**[2], **Nadia El-Mabrouk**[1], **Eric Tannier**[2 3]

6.1. Abstract

Motivations. Gene trees inferred solely from multiple alignments of homologous sequences often contain weakly supported and uncertain branches. Information for their full resolution may lie in the dependency between gene families and their genomic context. Integrative methods, using species tree information in addition to sequence information, often rely on a computationally intensive tree space search which forecloses an application to large genomic databases.

Results. We propose a new method, called ProfileNJ , that takes a gene tree with statistical supports on its branches, and corrects its weakly supported parts by using a combination of information from a species tree and a distance matrix. Its low running time enabled us to use it on the whole Ensembl Compara database, for which we propose an alternative, arguably more plausible set of gene trees. This allowed us to perform a genome-wide analysis of duplication and loss patterns on the history of 63 eukaryote species, and predict ancestral gene content and order for all ancestors along the phylogeny.

Availability. A web interface called RefineTree, including ProfileNJ as well as other gene tree correction methods, which we also test on the Ensembl gene families, is available at: `http://www-ens.iro.umontreal.ca/~adbit/polytomysolver.html`. The code of ProfileNJ as well as the set of gene trees corrected by ProfileNJ from Ensembl Compara version 73 families are also made available.

1. DIRO, Université de Montréal, Canada
2. LBBE, UMR CNRS 5558, Université de Lyon 1, France
3. INRIA, Grenoble Rhône-Alpes, France

6.2. Introduction

Several gene tree databases from whole genomes are available, including Ensembl Compara [319], Hogenom [426], Phog [427], MetaPHOrs [428], PhylomeDB [429], Panther [430]. However, they are known to contain many errors and uncertainties, in particular for unstable families [428]. Their use for accurate ancestral genome inference, orthology detection, or the study of genome dynamics could lead to erroneous results. For example, Ensembl Compara trees, when reconciled with a species tree to annotate gene duplications and losses, systematically and unrealistically overestimate the number of genes in ancestral genomes, and lead to erroneous predictions of ancestral chromosome structures [25]. It is a known artifact, and a substantial number of nodes in the Ensembl gene trees are labeled as "dubious" [189].

Reasons for errors in gene trees are numerous. When constructed from multiple sequence alignments of homologous genes, they are dependent on gene annotations, gene family clustering or alignment quality, as well as on the accuracy of the models and algorithms used. But above all, gene sequences often do not contain enough substitutions to resolve all the branches of a phylogeny, or alternatively, too many such that the substitution history is saturated. Therefore *sequence based methods*, computing gene trees from sequence information (e.g. PhyML [216], RAxML [217], MrBayes [431], PhyloBayes [229]), are usually accompanied with measures of statistical support on their branches or *a posteriori* distributions of likely trees.

Another category of methods, designated here as *integrative methods*, use a species tree, in addition to a multiple sequence alignment, to model gene gains and losses inferred from the reconciliation between gene and species trees (e.g. TreeBeST [28], TreeFix [432], NOTUNG [433], PhylDog [25], ALE [26], GSR [250, 434], SPIMAP [282], Giga [27], MowgliNNI [425]). They all report gene trees with better accuracy compared with sequence-based methods. But they leave a large space for improvement, both on tree quality and on computing time. In terms of models, they often assume unrealistic loss/retention ratios [435]. In terms of computation strategy, most of them use tree space exploration based on small modifications or local moves on branches (typically NNI, SPR, TBR), usually proposed at random. Moves are accepted or rejected according to hill-climbing, Metropolis-like criteria, or other statistical or empirical arguments. Such exploration methods are computationally intensive and do not scale well as databases grow in size. Consequently, database construction pipelines such as TreeBeST (constructing the Ensembl Compara gene trees [319]) have to adopt compromises, exploring a limited tree space. Improving local exploration can be done by using some *correction* techniques, most of them based on the idea of selecting local moves reducing the

cost of reconciliation with a species tree (e.g.[287, 291, 296, 321, 324, 424, 425, 428, 433, 436]). But even with such improvements, it remains that most local search strategies have no guarantee, neither on running time nor on the quality of the solution.

In this paper, we propose a new gene tree correction method, called ProfileNJ , which can be directly used as a fast integrative method, without local search. It is a deterministic approach with a guaranteed time complexity. ProfileNJ takes as input a starting tree with supports on its branches, and outputs a set of rooted binary trees containing all well-supported branches of the starting tree, and minimizing the number of duplications and losses when reconciled with a given species tree. It is based on PolytomySolver, a previous algorithm developed by our group [296] for resolving a non-binary tree in a way minimizing a reconciliation cost. Its theoretical complexity has been shown to outperform previous algorithms for the same problem, namely NOTUNG [433] and Zheng and Zhang's algorithm [294]. ProfileNJ extends PolytomySolver by integrating Neighbor-Joining (NJ) principles to choose among the numerous optimal solutions. Among all trees with equal reconciliation cost exhibiting the same gene count on branches, a choice is made with NJ principles, based on a distance matrix computed from gene sequences. Other extensions are considered such as allowing different costs for duplications and losses, as well as unrooted trees as input. These extensions turn an algorithmic principle into a workable method suited for constructing trees from biological data.

We compare ProfileNJ with TreeFix [284]. Indeed, among all correction methods, TreeFix adopts the most similar evaluation strategy, by exploring neighboring trees which are statistically equivalent according to the sequences. Moreover, TreeFix is among the best available integrative methods, according to the quality of the output trees and running time. On simulations, both algorithms achieve results of comparable quality, but ProfileNJ is several times faster. We also ran ProfileNJ on the whole set of gene families from the Ensembl database, which is out of reach for competing methods with comparable quality. The trees for the whole database were processed by ProfileNJ in less than 48h sequential time. In the same amount of time, 0.5% of the database was corrected by TreeFix. The trees we propose compare very favorably with the trees stored in Ensembl. We then used the reconstructed trees to study genome evolution across the 63 eukaryotic species from the Ensembl database (release 73). A whole genome analysis of duplication patterns is provided, pointing at certain branches which seem to show acceleration of duplication or loss processes.

ProfileNJ is integrated into a modular interface called RefineTree that contains two other gene tree correction tools using information on extant and ancestral synteny [424, 425]. We could thus evaluate the results of a pipeline taking into account gene sequence, gene

content and chromosome structure evolution on the Ensembl database according to several criteria: (1) likelihood ratio based on the Ensembl alignments; (2) ancestral genome sizes based on reconciliation with the species tree; (3) linearity of ancestral chromosomal segments computed with DeCo [281]. We discuss the improvements brought by each type of method and the distance of their output to "true" gene trees, in the light of incomplete lineage sorting and gene conversion.

6.3. Description of ProfileNJ

The basic vocabulary of phylogenetic trees is taken from [313], and the reconciliation method between a rooted binary gene tree G and a rooted binary species tree S is recalled in the Method section. Just note that in a reconciled gene tree G, each node (representing an extant gene if at a leaf or an ancestral gene if at an internal node) is mapped to the node of S corresponding to the genome the gene belongs to. Edges of G are subdivided, adding extra vertices and pending edges so that the extremities of an edge map either to the same node or to two extremities of an edge of S. An internal node of G is a duplication if it maps to the same node of S as one of its children. The number of genes in a species $s \in V(S)$ induced by a reconciled gene tree G is defined as the number of nodes $x \in V(G)$ mapped to s, such that the parent of x does not map to s. The reconciliation cost is either the number of duplications and losses, or a linear combination of the two if different weights are given to the two kinds of events. Also note that when two trees G_1 and G_2 have the same genes at their leaves, we can say that a branch of G_1 is present in G_2 if the bipartition of the leaves induced by this branch in G_1 is also induced by a branch of G_2.

ProfileNJ is a gene tree correction algorithm that takes as input a gene tree (rooted or unrooted) G for a given gene family with supports on its branches, and improves it according to the available information, taken from a species tree, a distance matrix and a threshold number for statistical support. It can be viewed as a generalization of three different standard algorithms designed for evolutionary studies:

— The Wagner parsimony method applied to the inference of ancestral gene contents from the extant gene contents by minimizing a duplication and loss cost [437];

— The Neighbor-Joining [438] (NJ) method which constructs a tree from a distance matrix D between taxa;

— The reconciliation of a gene tree G with a species tree S [267].

Whereas these three methods do not have much in common *a priori*, they are all bricks of our solution, and each of them reduces to some particular case of our problem. ProfileNJ outputs a rooted binary gene tree G_c on the same gene family as the input gene tree G, where all branches of G with a support above the threshold are present in G_c. Among all such trees, ProfileNJ outputs those minimizing a duplication and loss cost when reconciled with the species tree with respect to the NJ criterion.

ProfileNJ is an extension of PolytomySolver, a previous algorithm developed by our group [293]. We first describe the principle of the latter and then describe the additions.

6.3.1. PolytomySolver

It takes as input a multifurcated rooted gene tree G (with non-binary nodes) and a binary rooted species tree S. It outputs a binary rooted gene tree containing all branches of G, that minimizes the number of duplications and losses when reconciled with S. It has been shown by [296] that each polytomy (multifurcated node) of G can be considered independently. Therefore, in the following, we restrict the presentation to a single polytomy P (s.f. polytomy P in Fig 6.1).

The algorithm, based on dynamic programming, computes a table M where, for each node (including leaves) s of S and each integer k (limits on k are discussed in [296]), $M(s,k)$ is the reconciliation cost of a gene tree with k genes in species s before any duplication in s. For example, in Fig 6.1, P has three genes belonging to genome b, and thus $M(b,1) = 2$ as any solution having one gene in b before any duplication in b means that two duplications must have occurred in b, while $M(b,4) = 1$ as having four genes induces one gene loss on b.

The final cost of a minimum refinement of the polytomy is given by $M(r,1)$, where r is the root of S. Using a backtracking approach, PolytomySolver then outputs a *count vector* V containing the number of genes per node of S. Notice that, by construction, two brother nodes of S (nodes with the same parent) have the same count. Then a gene tree G of minimum cost $M(r,1)$ is found, such that in the reconciliation of G with S there are exactly $V[s]$ genes in each $s \in V(S)$. For example, the final binary tree in Fig 6.1 has two maximal trees rooted at b, as required by the count vector V.

If the reconciliation cost is the number of duplications and losses (i.e., the same unit cost is attributed to each duplication or loss), the table M can be constructed in time linear to $|S|$ [296], leading to a linear-time algorithm for finding one optimal refinement of the polytomy. Moreover, we showed recently [29] that linearity can be extended to a whole gene tree involving multiple polytomies. For weighted operations, i.e., different costs for duplications and losses, the algorithm runs in quadratic time [29].

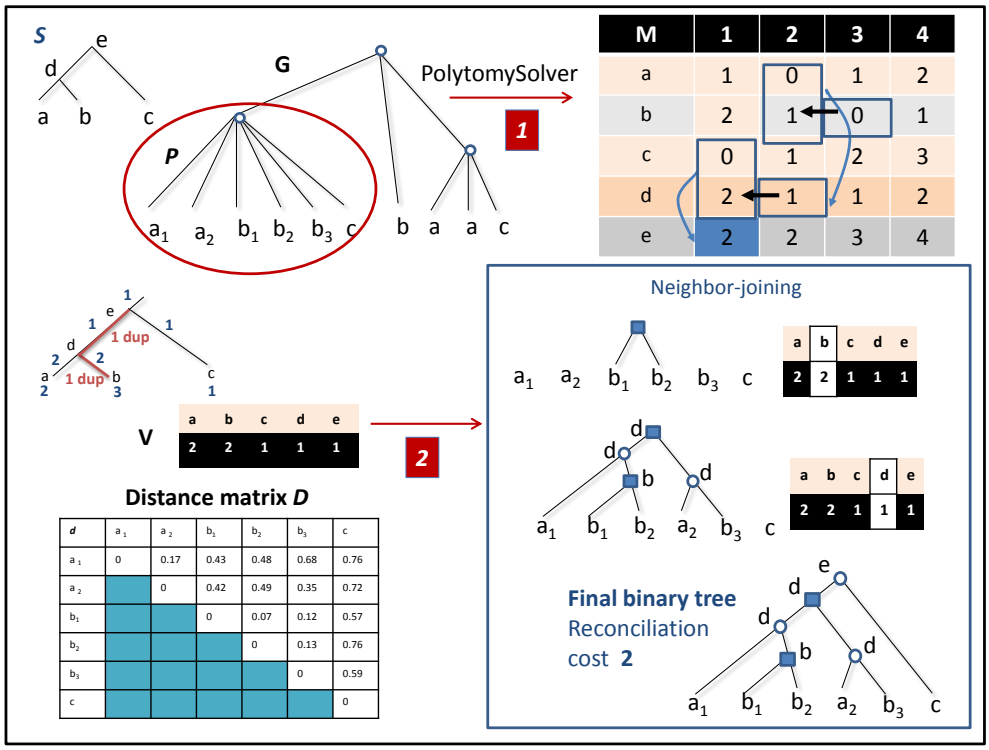


Figure 6.1. ProfileNJ at a glance. The input is a species tree S, multifurcated gene tree G and a distance matrix D. Each leaf x_i or x of G represents a gene belonging to genome x present as a leaf in S. Step (1) of ProfileNJ is PolytomySolver, which resolves each polytomy P of G independently. A dynamic programming table M is constructed. Step (2) of ProfileNJ takes as input a count vector V, here resulting from the backtracking path related by rectangles and arrows in table M, and a distance matrix d for the considered genes. A Neighbor-Joining (NJ) based procedure computes the gene tree in agreement with V that best reflects the distance matrix. The final completely refined tree is given bottom right. Duplication nodes are indicated by squares.

6.3.2. New extensions

ProfileNJ consists in contracting branches with low support in G, which leads to polytomies, and applying Polytomysolver. If G is not rooted, one node is chosen as the root. All nodes can be tried and one minimizing the cost can be chosen. The first phases of Polytomysolver are applied, until the construction of a count vector V.

Our main extension concerns a treatment of the multiplicity of solutions, as it can be exponential. Indeed, the backtracking procedure mentioned above may lead to many optimal count vectors, and for each count vector, there are possibly several gene trees in agreement with it. Therefore, exploring the set of optimal trees requires exploring the set of all gene

trees in agreement with each count vector. For example, the count vector of Fig 6.1 induces two duplications in b. However, this vector involves no information on which of the three genes b_1, b_2, b_3 should be joined first. Such information can be deduced from the pairwise alignment distance between gene sequences.

Suppose that a pairwise distance matrix D is available for the gene family. Then the problem can be seen as selecting, among all optimal solutions possibly output by PolytomySolver, given a vector V, the one best reflecting the distance matrix D. The problem of constructing a solution such that its induced distance is close to D according to a standard measure of metric spaces comparison is NP-complete. But it is also known to be empirically and, to a certain extent, theoretically, well approximated by Neighbor-Joining (NJ) [438, 439]. In ProfileNJ , we use such an NJ approach for choosing neighboring genes.

As in the NJ algorithm, a metric space E induced by D on the leaves of P, is progressively augmented with newly created genes. The algorithm proceeds by successively joining pairs of nodes (points of E), eventually leading to a full binary tree. For example, in Fig 6.1, the initial metric space E contains the nodes $\{a_1, a_2, b_1, b_2, b_3, c\}$. Joining the nodes b_1 and b_2 leads to the new set of nodes $\{a_1, a_2, b_3, b_4, c\}$. Nodes to be joined are selected according to the NJ criterion, namely we select from a node set of size n, the couple of genes x and y minimizing:

$$Q(x,y) = (n-2)D(x,y) - \sum_{t \neq x} D(x,t) - \sum_{t \neq y} D(y,t). \tag{6.3.1}$$

The metric space E is updated after each join $r = (x,y)$ by removing x and y, adding a new node r, and computing the distance between the newly created node r with each element t of E. When x and y are not created artificially (i.e. they are not loss nodes created with the last instruction of Algorithm 1), this is done using the NJ formula:

$$D(r,t) = \frac{1}{2}(D(x,t) + D(y,t) - D(x,y)) \tag{6.3.2}$$

Otherwise, if x is a loss, we set $D(r,t) = D(y,t)$ and if y is a loss, $D(r,t) = D(x,t)$.

A full pseudo-code of the extension part of ProfileNJ is written as Algorithm 1. It works on one polytomy P, assuming that all polytomies below have been resolved. It takes as input a count vector V, the species tree S and a distance matrix D defining the metric space E. It outputs a refinement of P in agreement with V, resulting from the performed joins on the nodes of E. Given a node s of S, denote by $E(s)$ the subset of E restricted to the genes belonging to s, and by $m(s) = |E(s)|$ the multiplicity of s in E. The tree S is processed bottom-up. For each internal node s, speciations are considered first by clustering, using the

NJ criterion, the genes from $E(s^l)$ with the genes from $E(s^r)$, where s^l and s^r are the two children of s. If the obtained multiplicity $m(s)$ of s is greater than the desired count $V[s]$, then duplications are performed, again using the NJ criterion for choosing the gene pairs in $E(s)$ to be joined. Otherwise, if $m(s)$ is lower than the desired count $V[s]$ of gene copies, then losses are predicted.

Algorithm 1 ProfileNJ (S,P,V,D)

Let E be the metric space with nodes corresponding to the leaves of P;
For each node s of S in a bottom-up traversal of S *Do*
  If s is an internal node of S with chidren s^l, s^r *Do*
    {By construction, $V[s^l] = V[s^r] = n$}
    For $i = 1$ *to* n *Do*
      Choose in $E(s^l) \times E(s^r)$ the gene pair (g^l, g^r) minimizing equation (1) and create the node $g = (g^l, g^r)$;
      Remove g^l and g^r from E and add g;
      Compute $D(g,g')$ for all $g' \in E$ using equation (2);
    End For
  End If
  If $m(s) > V[s]$ *Do*
    For $i = 1$ *to* $m(s) - V[s]$ *Do*
      {Perform $m(s) - V[s]$ duplications}
      Choose in $E(s) \times E(s)$ the gene pair (g_1, g_2) minimizing equation (1), and create the node $g = (g_1, g_2)$;
      Remove g_1 and g_2 from E and add g;
      Compute $D(g,g')$ for all $g' \in E$ using equation (2)
    End For
  End If
  Else If $m(s) < V[s]$ *Do*
    {Perform $V[s] - m(s)$ losses}
    Add $V[s] - m(s)$ artificial genes to $E(s)$, each with infinite distance to all elements of E;
  End If
End For

6.3.3. Complexity

Let G be the solution output by the algorithm, and suppose that G has n leaves after the inclusion of lost genes. Then exactly $n - 1$ NJ operations have been performed. Each join calculation is restricted to a subset of the genes, and so the time required to perform these joins is bounded by the time required to run the classical NJ algorithm on the n leaves of G, which is $O(n^3)$. Note that n can be as large as $|V(P)||V(S)|$, making the worst case running

time $O(|V(P)|^3|V(S)|^3)$. However, this worst case only occurs when $O(|V(S)|)$ losses are inserted on each branch of the solution. In practice n is in $O(|V(P)|)$.

6.3.4. A Multi-functional algorithm:

ProfileNJ is a phylogenetic tool that generalizes several usually unrelated standard methods. Indeed, if G is a binary rooted tree, then ProfileNJ can be seen as a reconciliation tool. If G is unrooted, then ProfileNJ can be used to choose an appropriate root according to the induced reconciliation cost. On the other hand, various ways of contracting branches can be considered. For example, an exploration scheme contracting the branches one by one and applying ProfileNJ can be considered, which would be equivalent to local modifications [440]. A more radical modification would be to contract all branches, leading to a star tree. In this case, ProfileNJ can be seen as a tool for computing ancestral gene content with Wagner parsimony, minimizing the cost of duplications and losses. If the star tree has all its genes belonging to a single species, ProfileNJ returns an NJ tree. Other kinds of contraction schemes can be imagined, as contracting branches around "Non Apparent Duplications" [441], or "Dubious duplications" stored in the Ensembl trees.

Notice that the pseudo-code for ProfileNJ has been given for a single count vector. For a full exploration of the tree space, all count vectors should be considered. This is what we do in the simulation section.

6.4. Efficiency of ProfileNJ

6.4.1. Efficiency of the NJ criterion

We ran ProfileNJ twice on the same data set of 20519 trees (the Ensembl Compara gene families), except that once the distance matrix was computed using the Ensembl nucleotide alignments with FastDist from the FastPhylo package [442], and once the distance matrix was random. The starting tree was computed for every family using PhyML on the nucleic alignments, and all branches with aLRT support < 0.95 were contracted. On average 55% of the branches were contracted. A histogram of the full distribution is shown in Fig A in S1 File.

Then we computed the likelihood of both trees for every family with PhyML. Among the trees for which the likelihood was different (55% of all tested trees), 76% were in favor of the trees built with the FastDist distance matrix. These 76% of trees account for 95% of the total sum of likelihood differences on all trees.

The comparisons are clearly in favor of the NJ criterion over no criterion at all, while quantitatively there remains a small but non-negligible part of the trees for which no criterion (the random distance matrix) gives an unexplained slightly, but significantly, better likelihood.

6.4.2. Efficiency of the tree space exploration strategy on simulated gene trees

We compared ProfileNJ with TreeFix, the most closely related tool, on simulated data. The principle of TreeFix is to randomly explore, by local moves, the space of trees that are statistically equivalent to the input tree, and report the one with the best reconciliation cost. Instead, we take a deterministic and more targeted approach by focusing on weakly supported branches of the tree, with a possibly deep modification of the tree. The comparison with TreeFix is therefore intended to compare these two space exploration strategies.

In [284], TreeFix has been compared with NOTUNG [433] and SPIMAP [282], showing better accuracy than NOTUNG, and a higher speed than SPIMAP. We perform a similar comparison on the same simulated dataset of 16 Fungi. This dataset consists of simulated gene families generated under the SPIMAP model and their corresponding nucleotide alignments, for four different rates of duplication and loss (DL) events: $(1 \times r_D,\ 1 \times r_L)$, $(2 \times r_D,\ 2 \times r_L)$, $(4 \times r_D,\ 4 \times r_L)$ and $(4 \times r_D,\ 1 \times r_L)$, where r_D and r_L are respectively the estimated duplication and loss rates for fungi. For instance, a $(2 \times r_D,\ 2 \times r_L)$-simulated gene family is expected to have, on average, two times more duplications and losses than a real gene family in fungi. Comparisons reported in this section are performed on 2575 simulated gene families randomly chosen from the four fungi datasets with different DL rates.

An initial maximum likelihood (ML) tree is constructed for each simulated gene family with RAxML v-8.1.2 [217], with the rapid bootstrap algorithm, under the GTR-Γ model and the majority rule consensus tree as bootstopping criterion. A randomly rooted tree is then provided as input to TreeFix (as TreeFix requires the input tree to be rooted), while a multifurcated unrooted tree obtained by contracting the branches with support lower than 95% is provided as input to ProfileNJ. We used default parameters for both programs. Among the set of all binary trees output by ProfileNJ (one for each count vector), the best statistically supported tree was selected using RAxML under the GTR-Γ model of nucleotide substitution.

For RAxML, TreeFix and ProfileNJ trees, we measured the Robinson-Foulds (RF) distance to true trees, compared the reconstructed tree with the true tree using site-wise likelihoods (see Fig G in S1 File), measured the accuracy of the duplication and loss scenarios

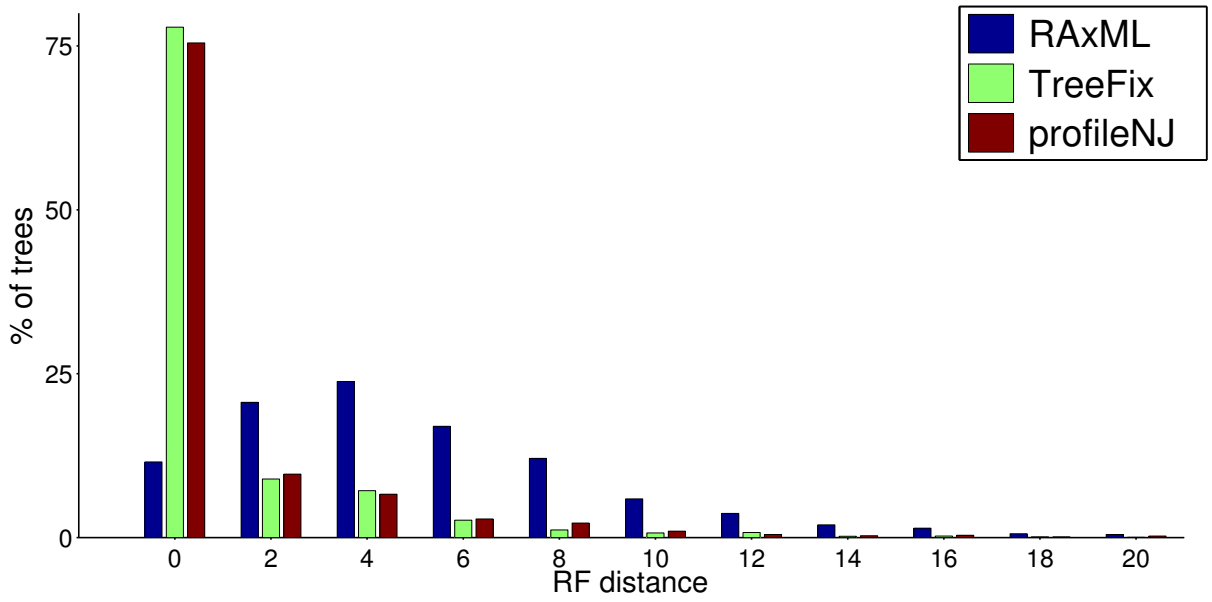


Figure 6.2. Topology accuracy of RAxML, TreeFix and ProfileNJ trees, measured by RF distance with the true tree, on $\sim$ 2500 simulated trees from the fungal dataset. We use a sample of trees simulated under four different DL rate : ($1r_D$ - $1r_L$), ($2r_D$ - $2r_L$), ($4r_D$ - $4r_L$) and ($4r_D$ - $1r_L$). Percentage of reconstructed trees (y-axis) with a given RF distance (x-axis) to the true tree. TreeFix and ProfileNJ have a similar reconstruction accuracy (75% of trees match the true trees) while the input trees (RAxML) have the lowest accuracy. The graph is cut on the right, but contains more than 99% of the data.

(Fig E in S1 File), the sensitivity of the accuracy to gene family size (Fig F in S1 File), the sensitivity to species tree errors (Fig H in S1 File), and the running time.

Fig 6.2 illustrates the results for the RF distance. It shows that sequence-only does not contain enough signal to lead to the true tree for our simulated dataset, and integrating additional information from the species tree actually improves reconstruction. Indeed, TreeFix and ProfileNJ reconstruct around 75% of true trees, compared with only 10% for RAxML. We investigated some cases where erroneous gene trees were inferred, and found that often, the true scenario was not parsimonious in terms of duplications and losses, while TreeFix and ProfileNJ chose duplications that are too recent in order to avoid losses. An example is given in Fig D in S1 File.

The performances of TreeFix and ProfileNJ are similar in terms of distance to the true tree. As for RAxML, it gives the best likelihood, which is not surprising as it is specifically designed for that. The returned likelihood is even usually higher than the likelihood of the true tree, but not significantly according to an AU test (Approximately unbiased [245], see

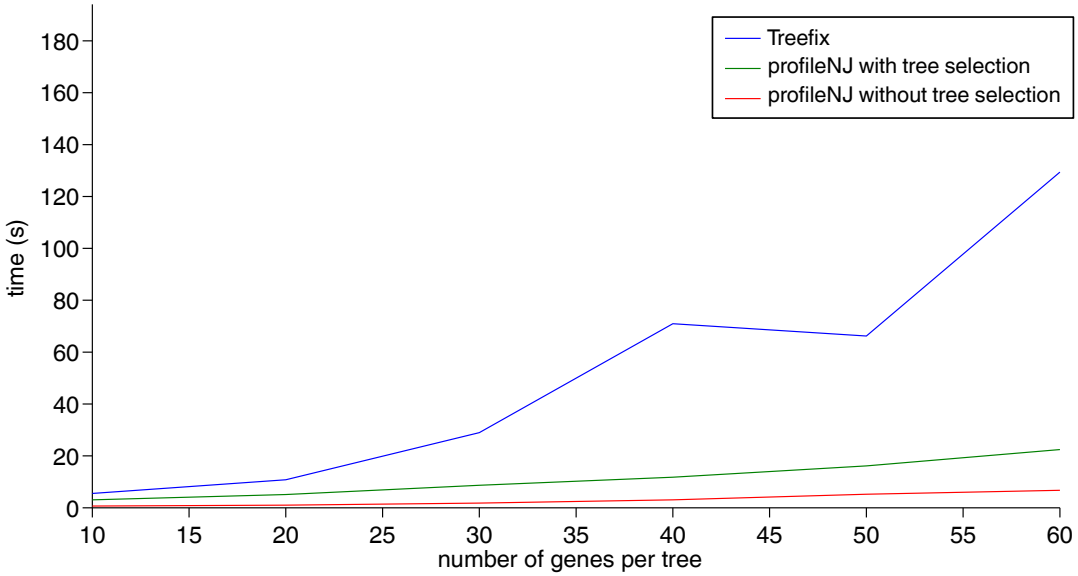


Figure 6.3. Run time of TreeFix and ProfileNJ for increasing size of gene tree. Run time was measured as the elapsed time from initial invocation of each method to completion of its output, with parallelism disabled for ProfileNJ. All experiments were performed on a system running Scientific Linux 6.3, with a single Intel Xeon X5650 Westmere (2.67 GHz) processor, available on compute nodes with 24GB internal memory on the Briaree server of Calcul Quebec.

Fig G in S1 File). TreeFix is designed to produce trees which are not significantly different from the ML tree, which we could check: 1.36% of the trees fail the AU test against the ML tree at $\alpha = 0.05$, while the proportion jumps to 9.17% for ProfileNJ. It is noticeable that this has no visible consequence on the distance to the true tree (Fig C in S1 File).

Fig 6.3 shows that ProfileNJ outperforms TreeFix in running-time, the gap between the two algorithms increasing with tree size. This figure also shows that the most time-consuming step in ProfileNJ is tree selection. For a tree of size 30, ProfileNJ is about four to seven times faster than TreeFix, and about 15 times faster if we discard statistical support evaluation and tree selection step with RAxML. This includes the construction of the distance matrix, but not the construction of the initial RAxML, as it is common to both methods.

Other analyses, including the sensitivity to gene family size and the number of duplications and losses, are reported in S1 File. They lead to the same conclusions: TreeFix and ProfileNJ have similar performance on all measures except running time for which ProfileNJ is significantly better.

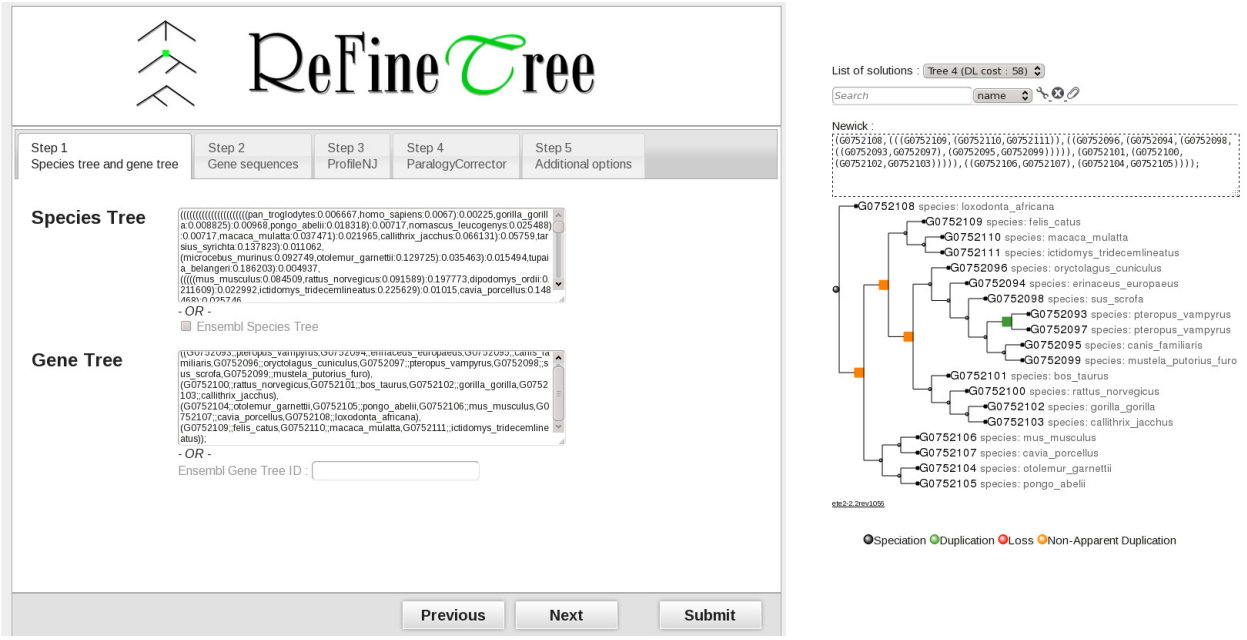


Figure 6.4. RefineTree web interface. The input is a species tree (or by default the Ensembl species tree) and a gene tree (or an Ensembl gene tree ID), gene sequences and additional options such as the branch contraction threshold, the request to test all roots, the maximum number of trees to be output by ProfileNJ and sorted by likelihood, etc. The integrated algorithms are ProfileNJ and ParalogyCorrector. Using this second algorithm requires, in addition, the input of a set of orthology constraints.

6.5. Application on biological data

6.5.1. Software: RefineTree

ProfileNJ is integrated into a new modular online software, called RefineTree, designed to combine a number of correction techniques, with an easy-to-use interface (see Fig 6.4). The present version includes, in addition to ProfileNJ , a tool called ParalogyCorrector [424] for correcting orthology relations. ParalogyCorrector takes as input a gene tree and a set of known pairwise orthology relations between genes, which would typically be derived from synteny comparisons, and constructs the the tree which is the closest to the input tree according to the RF distance, with the constraint that couples of putative orthologs must be orthologs in the reconciliation (see Method section).

RefineTree can be used in a modular way, according to the user's specifications. It has been designed to be easily extensible to other tools. For example instead of asking the user to input his own orthology relations, tools for inferring putative orthologs can be included.

6.5.2. Protocol

The low running time of ProfileNJ makes it possible to run it on large databases. Here, we exhaustively use ProfileNJ to correct all trees of the Ensembl Compara database, containing 63 whole eukaryotic genomes. Gene trees are constructed for 20519 families. In order to quantify the contribution of ProfileNJ and the contribution of methods using other kinds of information as synteny, we compared three sets of trees on the whole database.

— **Ensembl trees**: Trees stored in the Ensembl gene family database (see Method section);

— **ProfileNJ trees**: Trees output by ProfileNJ with unrooted PhyML trees as input (where branches with aLRT support < 0.95 are contracted) and FastDist distance matrices. A single solution is retained for the rooting leading to a minimum weighted reconciliation cost (see Method section);

— **Synteny trees**: Trees output by either ParalogyCorrector or Unduplicator [424] (the two are computed and the most likely according to the sequence is chosen) with ProfileNJ trees as input, using PhylDiag [443] and DeCo [281] to infer synteny constraints (see Method section and Fig 6.5)

6.5.3. Results on Ensembl gene trees

The PhyML trees for the 20519 gene families were processed by ProfileNJ in less than 48h sequential time. In the same amount of time, 0.5% of the database was corrected by TreeFix.

We evaluated the resulting trees (all made available at the web page indicated in the abstract) according to sequence likelihood, ancestral genome content and ancestral chromosome linearity. The ancestral genome content metric is based on the assumption that the distribution of ancestral gene content sizes should be close to that of extant genomes. Incorrect trees are known to require additional duplications to be reconciled with the species tree, which tends to increase the number of genes in ancestral genomes. The ancestral chromosome linearity metric is based on the assumption that the linearity of ancestral genomes is expected to be as close as possible to that of the extant genomes, with each gene having zero, one or two neighbors, with a peak at two (having genes with zero or one neighbor is usually due to partially assembled genomes).

Results are given in Fig 6.6. ProfileNJ trees show a better behavior than Ensembl trees according to the three measures: more than 2/3 of the trees have a better likelihood than Ensembl trees, ancestral genome content distribution is much closer to the extant one, and

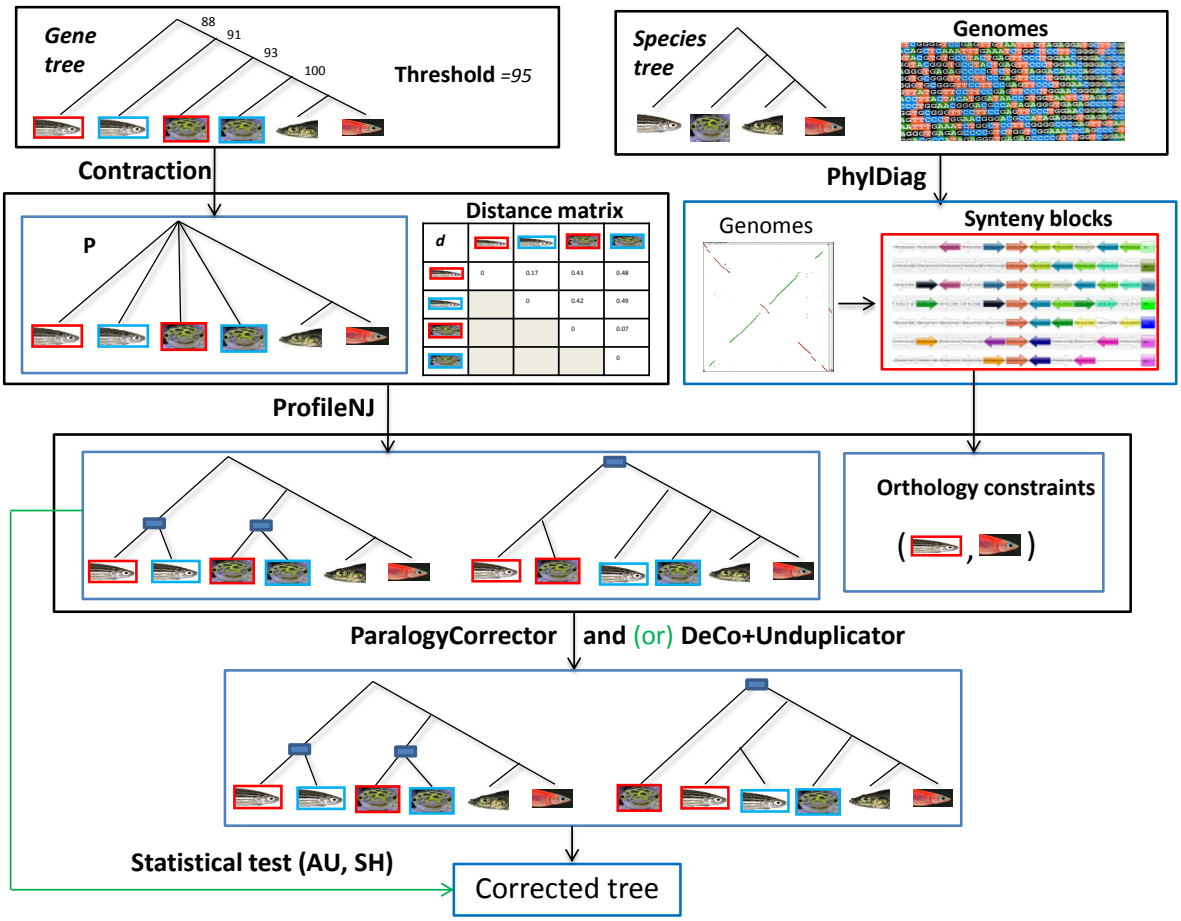


Figure 6.5. A general view on RefineTree when run on the Ensembl Compara gene families. An example is given for a species tree S of four fish species, a gene family of six genes (a gene is represented by the picture of the species it belongs to, and two paralogs belonging to the same species are distinguished by a different frame color), a rooted gene tree G (although it can be unrooted in general) with branch support, and a given threshold for branch contraction. Data framed in black are the input and those framed in blue are the output of the correction algorithm labeling the edge linking the considered frames. Black arrows depict the use we make of RefineTree on the Ensembl gene trees. The green arrow and the green "or" are alternative uses avoiding one or both of the correction tools ParalogyCorrector and Unduplicator. Any framed set of data can be alternatively provided to the pipeline as input. For example, orthology constraints obtained from various sources can be directly provided as input to ParalogyCorrector. The method for inferring orthology constraints from synteny blocks is described in the text.

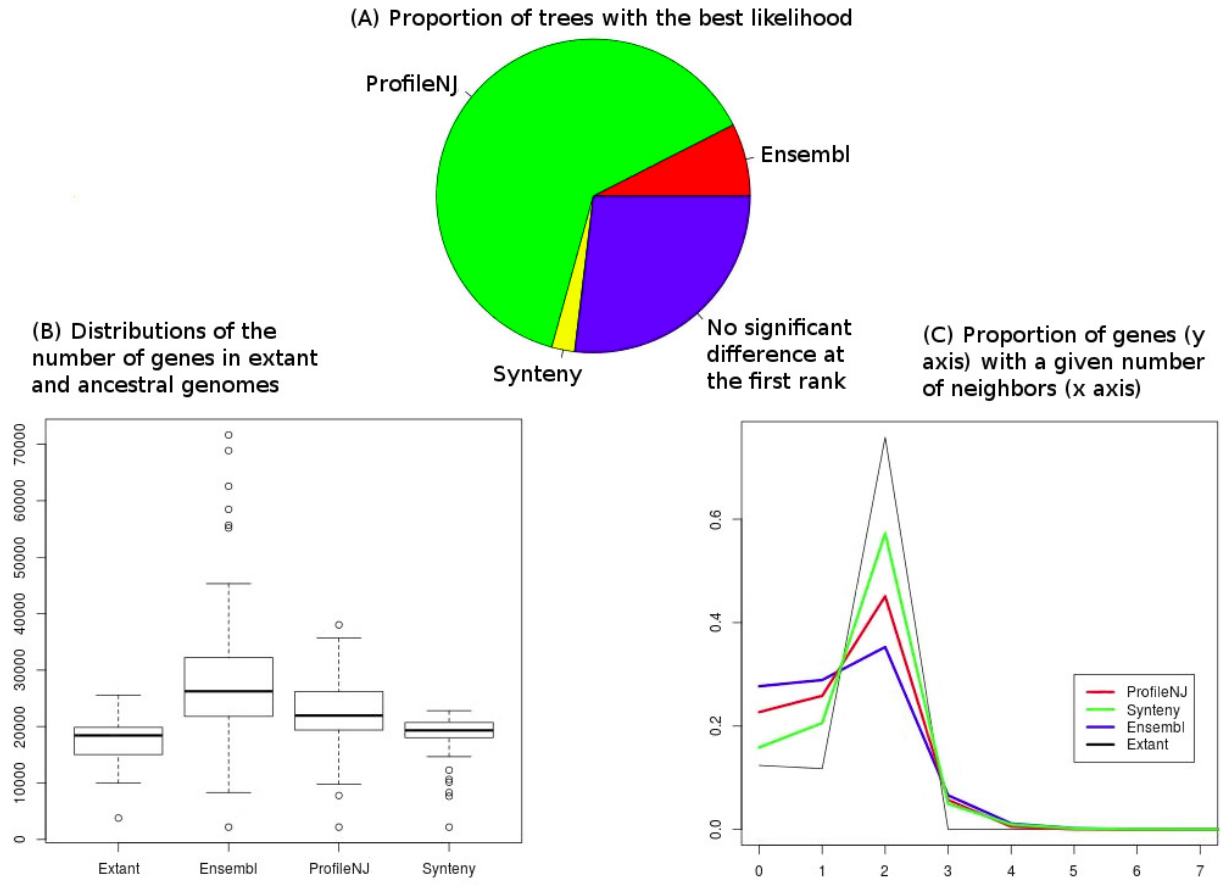


Figure 6.6. Sequence likelihood, ancestral genome content and ancestral chromosome linearity for ProfileNJ , Synteny and Ensembl trees: **(A)** Proportion of trees with a significantly better likelihood computed with PhyML. AU tests were computed for the three trees for each family, and if the tree at the first rank was significantly better than the second, it was stored as the best likelihood, and if not, it was stored as "no significant difference at the first rank". **(B)** Gene content computed with DeCo. Gene content has one value for each node of the phylogeny of 63 species, except for extant genomes, for which it has one value for each leaf. **(C)** Genome linearity computed with DeCo. Genome linearity is represented by a graph, whose x axis is the number of neighbors a gene can have, and the y axis shows the proportion of genes having this number of neighbors. Parameters from extant genomes are given as a reference in (B) and (C). Statistics for ancestral genomes are assumed better when close to the extant ones.

linearity of chromosomes is higher. Therefore this set of trees, achieving better performance according to sequence evolution, gene content evolution and chromosome evolution, is arguably a better dataset than the one stored in the Ensembl database.

However, results obtained when we include synteny information are less clear. Indeed, quality of synteny trees drops in terms of likelihood (Fig 6.6 (A)), but jumps in terms of the stability of gene content and the linearity of ancestral chromosomes (Fig 6.6 (B) and (C)).

6.5.4. Results on duplication and loss history in Eukaryotes

Partial patterns of duplications and losses in eukaryotes have been considered in previous studies, as for example by [25] in mammals with a subset of gene families, or by [444] in vertebrates with a subset of species. The ability of ProfileNJ to handle the whole Ensembl database allowed us to perform a more exhaustive study. In addition to gene trees, we reconstructed all ancestral gene contents and organizations. Gene content is computed according to reconciliation (see Methods), and genome organization, which consists in sets of links between consecutive genes, is inferred with DeCo. Genes are not always clustered into full linear genomes. Such non-linearity has diverse causes that we do not wish to mask with an *ad-hoc* linearization method. An interesting property of DeCo is to highlight genes or groups of genes evolving together in parts of the tree. For example, 8488 blocks of co-duplicated genes are inferred by DeCo on the considered eukaryote dataset. Most of them contain only a few number of genes (83% contain 2 genes). The largest blocks are found in the terminal branches leading to *Danio rerio* and *Caenorhabditis elegans*.

Fig 6.7 shows the result for the full genomes of the full phylogeny of the 63 Ensembl species. As seen in Fig 6.7, duplication rates are highly variable across branches of the phylogeny. Branches with a large number of duplications (hot branches) are those leading to vertebrates, which is in agreement with the two rounds of whole genome duplication hypothesis. Interestingly, the speciation event leading to *Petromyzon marinus*, which is usually thought to have diverged after these events [445], precedes the hot branches. This may be in agreement with recent results based on the analysis of Hox clusters in the Japanese lamprey [446]. Another hot branch leads to eutherian mammals, which was also found by two other studies [25, 444] with partial data. These two hottest internal branches are exactly the ones found by Mahmudi et al [444] using a probabilistic technique, but using only 9 species due to computational cost. Other hot branches are terminal, the hottest being those leading to *Caenorhabditis elegans* and *Danio rerio*. This is possibly due to ongoing dynamics of polymorphic copy number variations. The same tree showing the number of losses is provided in Fig I in S1 File.

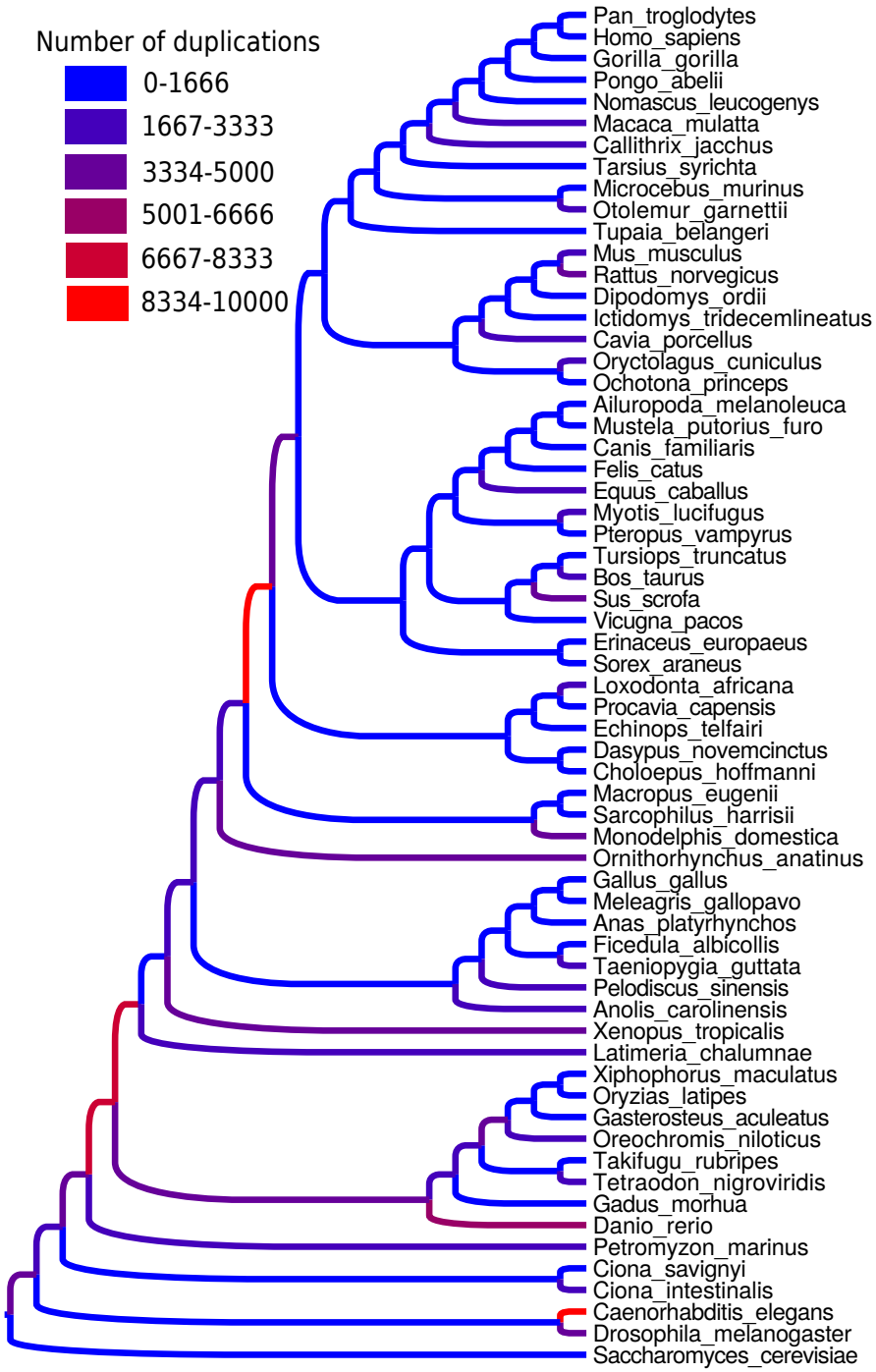


Figure 6.7. Numbers of duplications in the eukaryote phylogeny, estimated with reconciled ProfileNJ trees from PhyML starting trees on the whole Ensembl Compara database, version 73. Drawn with Figtree [447].

6.6. Discussion

ProfileNJ is a new gene tree correction method based on exploring a restricted tree space and choosing the most likely tree according to a species tree and a distance matrix on gene sequences. It is shown to be accurate and it outperforms in running time the most comparable existing correction methods. Efficiency in running time allowed us to apply ProfileNJ to the entire Ensembl database.

Trees obtained by correcting PhyML trees with ProfileNJ are arguably better than gene trees stored in Ensembl, according to sequence likelihood, ancestral genome content and ancestral chromosome linearity. We also corrected directly the Ensembl trees and the results (not shown) were similar, ProfileNJ giving better ancestral genomes and more likely trees than the starting trees. Based on such accurate trees, we have been able to perform an exhaustive study of the patterns of duplication and loss on the phylogeny of the 63 Eukaryote species included in Ensembl.

6.6.1. Accounting for synteny

In addition to sequence and phylogeny, we also used synteny information and orthology relations as correction criteria. As gene trees contain the most complete information about a gene family history, detecting orthologs or studying gene repertoire evolution should be achieved by interpreting trees. But due to the rate of errors in the current trees stored in databases, orthology is often assessed with a series of techniques including synteny [448] and Reciprocal Best Hits, while the evolution of gene repertoires is often studied with phyletic profile techniques [449]. What we presented here is a way of integrating those diverse techniques into a phylogenetic framework.

Unfortunately, integrating synteny information results in a drop of gene tree quality in terms of likelihood. One interpretation is that achieving orthology constraints may require breaking well-supported branches. Part of the likelihood drop could also be interpreted as an inadequacy of the sequence evolution models to appropriately account for gene families with a high rate of duplications. Further, as observed in our simulations, the true tree is not necessarily the ML tree. Finally, the likelihood is computed with an alignment that usually results from a guide tree, estimated using fast but crude approaches, and often different from the tree to evaluate. Some synteny trees might therefore be better trees even in cases where sequence likelihood disagrees, because sequence likelihood can be incorrect.

However there is a third interpretation. Synteny information describes the history of loci [270], while phylogenetic models describe the evolution of sequences. Loci and sequences

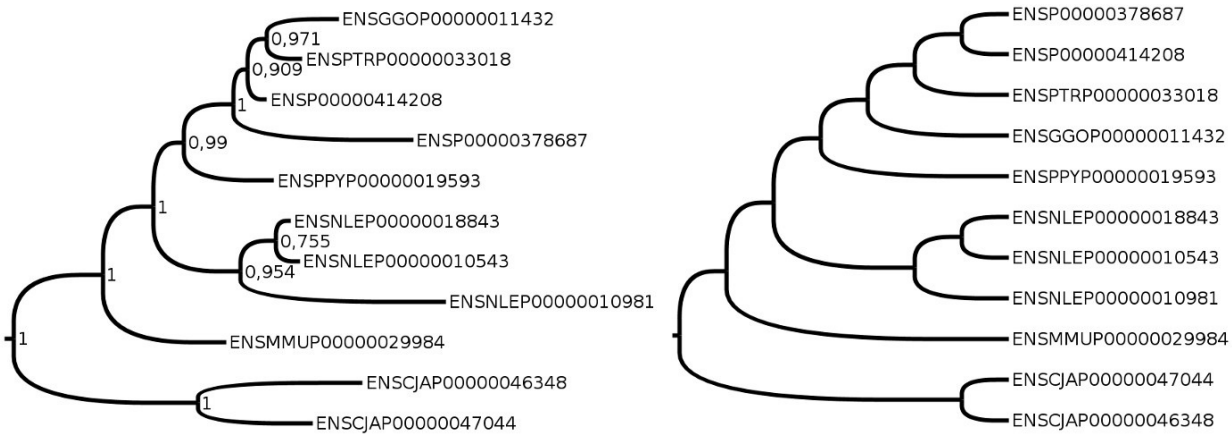


Figure 6.8. A probable example of ILS visible on a subtree of an Ensembl gene family. The monophyly of the chimpanzee and gorilla genes (ENSPTRP00000033018 and ENSGGOP00000011432) is well supported by the sequences (left tree, constructed by PhyML, with aLRT supports), while synteny argues for orthology of both with the human genes (ENSP00000414208 and ENSP00000378687) (right tree, constructed by ProfileNJ followed by ParalogyCorrector), so that a scenario of duplications and losses compatible with the left tree is unlikely.

often have the same history, but they may differ following gene conversion or incomplete lineage sorting (ILS).

In case of ILS or gene conversion, two different true versions of the gene history are concurrent. In Fig 6.8 the gene as a locus has a history depicted by the right tree, while the gene as a sequence has a history depicted by the left tree. None of the two are wrong, but they are significantly different. They highlight the ambiguity of the definition of a gene, which yields an ambiguity in its history. Sequence trees will have a high likelihood and mediocre results for gene contents and synteny when constructed from duplication and loss scenarios, while it is the opposite for locus trees. A probabilistic model that incorporates ILS in sequence and duplications and losses in loci has been proposed[270]. However, no model is currently able to handle conversion.

6.6.2. Not only the gene trees

Using genome evolution in the construction of the gene trees, we get ancestral genomes as a byproduct. They are made of genes and sets of gene adjacencies. They are still too big (in terms of gene number) and too non-linear to be fully trusted. This is partly due to incorrect gene trees in our output, or incorrect inferences from DeCo, but also to problems in sequencing, assembling, annotating genomes, clustering families or inferring the species

tree. Good methods for finding linear structures from a set of adjacencies exist [450]. Here we rather used non-linearity as a testimony of the flaws of the data and methods used to reconstruct genome evolution.

Although gene trees are "better" with our correction, they still could be improved. The likelihood drop for synteny correction is indeed surprising, as these corrections lead to ancestral genomes that are closer to gene content and gene neighborhoods of extant genomes. We would need better exploration schemes with integrated models to really trust gene trees on a whole genome database within a deep phylogeny.

6.7. Methods

Families, alignments and trees were taken from Ensembl Compara release 73, sept 2013. They were computed with a pipeline called TreeBest, but we simply call them the "Ensembl trees". There are 20529 rooted trees, each corresponding to a gene family, for a total of 1091891 genes taken from 63 species. 13128 of these trees have 3 leaves or more, and are thus treated by our phylogenetic study. Information on gene position on chromosomes, scaffolds or contigs is available at ftp://ftp.ensembl.org/pub/release-73/emf/ensembl-compara/homologies/.

6.7.1. Use of ProfileNJ on Ensembl

PhyML was used with default parameters to compute maximum likelihood trees from the protein multiple alignments taken from Ensembl, for families with 4 sequences or more. An aLRT support was computed, and only branches with support ≥ 0.95 were preserved by ProfileNJ . FastDist was run on DNA alignments to provide a distance matrix. Then ProfileNJ was run with the command (an example is given for the first family).

```
ProfileNJ -s Compara.73.species_tree  \\
-g data/famille_1.start_tree \\
-d data/famille_1.dist \\
-o data/famille_1.tree \\
-n -r best -c nj --slimit 1 \\
--plimit 1 --firstbest --cost 1 0.99999
```

We tested the sensitivity of the method to the choice of the threshold parameter for contracting unsupported branches. The threshold is a trade-off between the amount of change in a tree and the probability that the resulting tree is rejected. Values that are too high would avoid exploring a large space around the starting tree while small values would

lead to low likelihood trees. It has to be settled empirically. For example, .80 bootstrap threshold was considered an acceptable threshold in some genomic studies [451].

6.7.2. Ancestral genes and genomes with DeCo

DeCo [281] computes ancestral genes and gene neighborhoods from gene trees and extant gene neighborhoods. It was used to compare ancestral gene contents and orders with different sets of gene trees for the same set of gene families to generate Fig 6.6. It takes as input rooted gene trees and *adjacencies*, *i.e.* couples of extant genes that are consecutive on a chromosomes. Ancestral adjacencies are constructed according to a parsimony principle minimizing the number of gains and losses of adjacencies.

Adjacencies at different loci on chromosomes are supposed to evolve independently one from the other. As a consequence, although an extant gene should belong to at most two adjacencies (zero or one is frequent because of low assembly quality of some genomes), it is not necessarily the case for ancestral genes. As a consequence ancestral genomes are not necessarily linear arrangements of genes. It might be seen as a weakness of this ancestral genome reconstruction method, but for us it is a criterion to evaluate the quality of gene trees. Indeed, a high part of the non-linearity of ancestral genomes is not due to the inadequacy of the software itself, but to the quality of the input data. It has been remarked that a significant improvement in the linearity of ancestral genomes was obtained by constructing gene trees according to more integrative models [25, 452].

6.7.3. Information from extant synteny

Orthology constraints for synteny trees are inferred as follows. If several genes are found consecutive in one genome, and their homologs are also found consecutive in the other genome, the common linear arrangement was in the ancestor and the homologous genes are probably orthologous. This hypothesis is incorrect in at least three cases : (1) if the whole block of genes was duplicated, (2) if there is a tandem duplication of a gene followed by a differential loss in the two species, or (3) if a gene is converted by a paralog. To handle these cases, we require that (1) the majority of the homologous genes are indeed predicted as orthologs by phylogeny, (2) the common ancestor of two homologous genes does not lead to two paralogous descendants placed in tandem in one species. In case (3), we are in a situation where the loci are orthologous but not the sequences. In that case we construct the “locus tree” [270] and trust syntenic information over gene sequence information.

First we ran PhylDiag as follows, for each pair of genomes. Files genome_1, genome_2 and ancestral_genes respectively contain the ordered list of genes from each genome, and the list of families clustering the genes as in the Ensembl database.

```
phylDiag.py genome_1 genome_2 ancestral_genes \\
-gapMax=2 -pThreshold=0.00000005 \\
-filterType=InBothSpecies -multiprocess \\
-minChromLength=2 >syntenyblocks_1_2
```

The statistical threshold is calculated in order to minimize the number of false positives, taking into account the number (2211) of comparisons between pairs of species and an expected number (500, as an approximate average of the number of found synteny blocks) of synteny blocks for each comparison ($0.05/(2211 \times 500) \approx 5 \times 10^{-8}$).

For each synteny block found by PhylDiag, we kept only the genes that had one single copy in the two blocks from both species. We counted the number of such pairs of genes and referred to an LCA reconciliation of the output trees of ProfileNJ to check that most pairs are orthologs (their common ancestor is labeled by a speciation). We discarded the blocks that did not fit this condition. This discards possible block duplications.

For the remaining blocks, and for each couple of uniquely represented genes a and b, we required that the LCA node X of a and b in the reconciled ProfileNJ tree is not a supported duplication: let X_1 and X_2 be the two children of the node X labeled as a duplication (so X_1 and X_2 are in the same species as X), the genes a and b are not kept as putative orthologs if one of the branches XX_1 and XX_2 has a high support (> 0.95), and there are two genes, x_1 and x_2, which respectively descend from X_1 and X_2, which are located on the same genome. This discards possible tandem duplications in the block, followed by differential losses of copies.

The output trees from ProfileNJ as well as the filtered pairs of putative orthologs were given as input to ParalogyCorrector, which finds the tree that is as close as possible to the input tree in terms of RF distance, such that in an LCA reconciliation, all pairs of putative orthologs have an LCA node annotated as speciation.

6.7.4. Information from ancestral synteny

From the results of DeCo on the output gene trees produced by ProfileNJ , we used an "unduplication" principle as in [425] every time we found that an ancestral gene x had three neighbors a,b,c, two of them (say a,b) arising from a duplication node d in a single gene tree. In that case, we rearranged the four grand children of d so that the clade under d has an

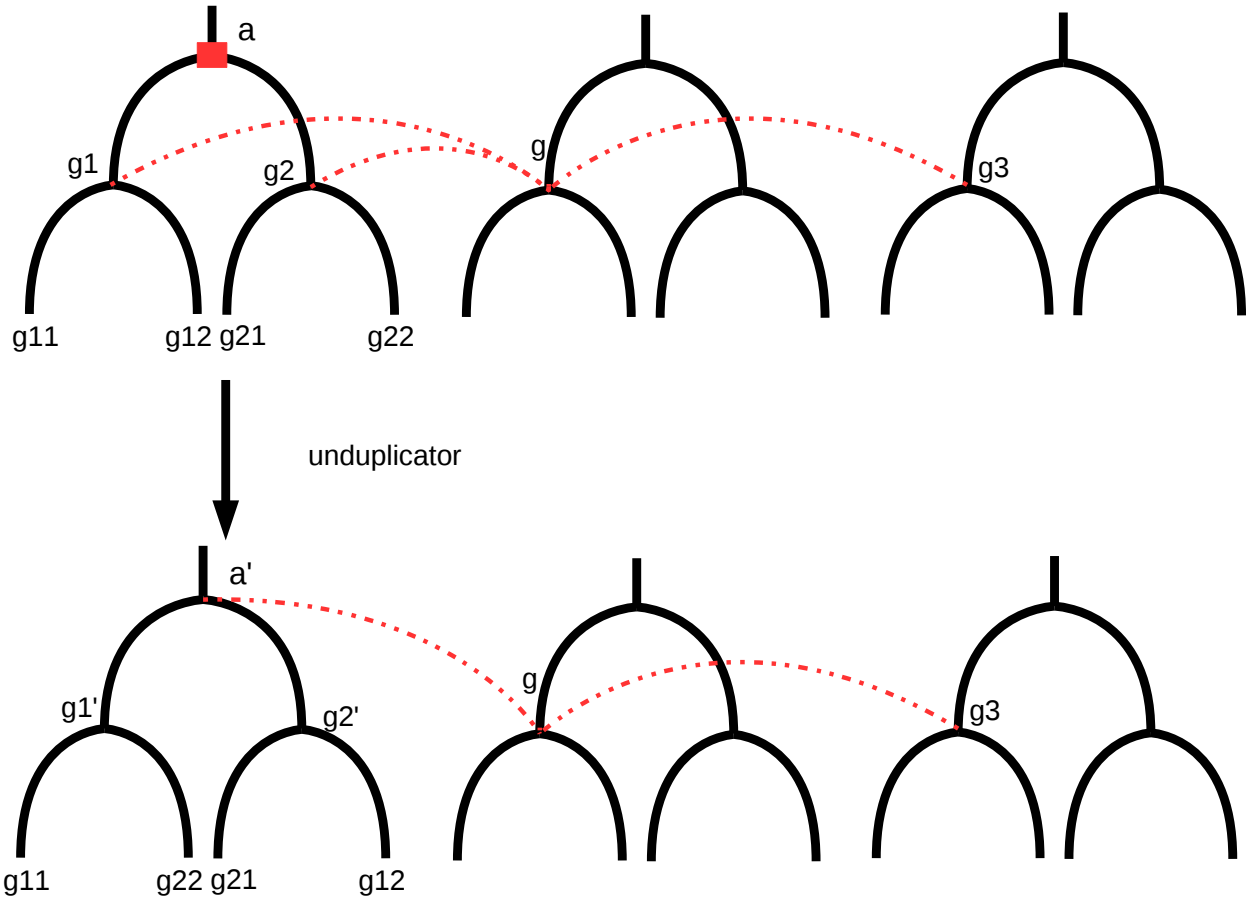


Figure 6.9. The unduplication principle (figure redrawn from [425]). A non linearity is detected in an ancestral genome (gene g has three neighbors). Two of its neighbors g_1 and g_2 are issued from a possibly dubious duplication labeled node. The tree is rearranged so that its root is labeled with a speciation instead of a duplication. In the resulting configuration g'_1 and g'_2 are in two different species, so that g can have only one neighbor in this family, and linearity is recovered.

LCA which is annotated as a speciation in the LCA reconciliation. See an insight into its functioning in Fig 6.9.

6.7.5. Likelihood ratio tests

We computed the likelihood of all trees according to the HKY85 model with PhyML on nucleotide alignments. To test the significance of a likelihood difference, we computed the AU tests (Approximately unbiased, [245] with RAxML and Consel [248]. Unless otherwise stated, we use "significantly" better for a tree with an AU value > 0.95.

6.7.6. Data and code availability

The program for ProfileNJ is available at https://github.com/UdeM-LBIT/profileNJ. The software is freely accessible for academic purpose, under a GPL license. We also provide the set of 20529 trees, as an output from ProfileNJ at ftp://pbil.univ-lyon1.fr/pub/datasets/PlosOne2016/. The 2575 simulated gene families used for our simulation represent a subset of the original SPIMAP simulated fungi datasets (see http://compbio.mit.edu/spimap/).

6.8. Supporting information

S1 File: Additional validity and robustness tests of ProfileNJ , followed by a representation of the number of losses along a phylogeny inferred from ProfileNJ trees. Figs A to I are included in S1 File.

Chapter 7

GATC: A Genetic Algorithm for gene Tree Construction under the Duplication-Transfer-Loss model of evolution

Cet article a été présenté à la conférence **APBC**(Asia Pacific Bioinformatics Conference) 2018, et est publié dans *BMC Genomics*. L'article présente GATC, un algorithme génétique pour la construction et la correction des arbres de gènes (voir l'annexe D pour les informations supplémentaires).

Dans le chapitre précédent, nous avions proposé ProfileNJ , une méthode de correction qui ciblait les branches faiblement supportées. Bien que cette méthode améliore grandement les arbres de gènes, ainsi que les histoires évolutives inférées, elle présente certaines limites, dont les plus pertinentes sont:

1. Elle est incrémentale dans le sens où les corrections appliquées, même en considérant le critère NJ additionnel, ne garantissent pas forcément que la solution finale soit optimale, à la fois pour les séquences et pour la topologie de l'arbre des espèces.
2. Le modèle d'évolution utilisé ne considère que les duplications et pertes, si bien que la méthode n'est pas efficacement applicable aux domaines du vivant où d'autres types d'évènements comme les transferts horizontaux sont fréquents.

Avec GATC nous considérons plutôt le problème comme celui de la construction d'un ensemble de topologies d'arbre de gènes qui devraient être simultanément optimales pour plusieurs critères. Nous montrons sur des données simulées et réelles que cet ensemble d'arbres contient presque toujours l'arbre dépeignant l'histoire évolutive la plus plausible.

GATC prend en entrée un arbre binaire et enraciné d'espèces, un alignement de séquences, et optionnellement un ensemble d'arbres de gènes. L'algorithme retourne un ensemble d'arbres de gènes enracinés qui sont optimaux pour une mesure combinant l'information

des séquences et celle de la réconciliation. Les différences entre ProfileNJ et GATC sont résumées dans la table suivante:

Table 7. I. GATC vs ProfileNJ .

	GATC	ProfileNJ
Modèle d'évolution	DTL	DL
Temps d'exécution	lent	rapide
Déterministe	✗	✓
Correction d'arbres	✓	✓
Construction d'arbres	✓	✗[1]
Arbres de gènes non enracinés	✓	✓
Critères d'optimisation ajustables	✓	✗
Arbres de gènes non binaires	✓	✓

1. possible sous forme de résolution d'un arbre étoile

Un défi intéressant rencontré lors de la conception de l'algorithme concerne la manière de combiner efficacement différentes sources d'information disponibles sur l'évolution des gènes. Lorsque les informations disponibles se limitent à celle sur les séquences et celle sur l'évolution des espèces, nous proposons deux façons intuitives de les combiner:

1. Lorsqu'une méthode probabiliste est utilisée pour calculer le score de réconciliation, nous pouvons combiner la vraisemblance de l'arbre des gènes G étant donnés les séquences et le modèle de substitution, et celle de G étant donné le modèle d'évolution et l'arbre des espèces. Cette combinaison, qui peut être pondérée, produit une mesure qui correspond à la vraisemblance jointe. C'est d'ailleurs la mesure utilisée par les méthodes bayésiennes comme ALE_mcmc [26] et PrIME-GSR [250]. Une mesure similaire est également utilisée par TERA [285]. En pratique, la réconciliation probabiliste est souvent difficile à faire, en raison des nombreuses exigences des modèles évolutifs sous-jacents. Par exemple, les "priors" sur les taux de duplications/pertes/transferts doivent être connus, et l'arbre d'espèces doit en général, être daté.
2. Lorsque la parcimonie est utilisée pour calculer le score de réconciliation, nous utilisons plutôt un algorithme d'optimisation multiobjectifs.

Le fonctionnement de GATC repose sur un algorithme génétique et diffère des méthodes bayésiennes par le fait qu'une *population* d'arbres est considérée à chaque génération, ce qui permet d'explorer un espace plus large de solutions, et ainsi de réduire les risques de tomber dans un optimum local. L'algorithme présente également l'avantage d'être facilement

adaptable pour considérer d'autres sources d'informations (conservation de fonction, ordre des gènes, etc.) sans avoir à formuler un modèle d'évolution tenant explicitement compte de ces informations.

Contributions: Emmanuel Noutahi a conçu l'étude, sous la supervision de Nadia El-Mabrouk. Emmanuel Noutahi a développé et implémenté l'algorithme, effectué les expériences, puis analysé les résultats présentés dans l'article. Le manuscrit a été rédigé par Emmanuel Noutahi et Nadia El-Mabrouk. Tous les auteurs ont lu et approuvé la version finale du manuscrit.

GATC: A Genetic Algorithm for gene Tree Construction under the Duplication-Transfer-Loss model of evolution.

BMC Genomics 2018 **19**(Suppl 2):102 doi:10.1186/s12864-018-4455-x

Emmanuel Noutahi [1], **Nadia El-Mabrouk**[1]

7.1. Abstract

Background. Several methods have been developed for the accurate reconstruction of gene trees. Some of them use reconciliation with a species tree to correct, *a posteriori*, errors in gene trees inferred from multiple sequence alignments. Unfortunately, the best fit to sequence information can be lost during this process.

Results. We describe GATC, a new algorithm for reconstructing a binary gene tree with branch length. GATC returns optimal solutions according to a measure combining both tree likelihood (according to sequence evolution) and a reconciliation score under the Duplication-Transfer-Loss (DTL) model. It can either be used to construct a gene tree from scratch or to correct trees inferred by existing reconstruction method, making it highly flexible to various input data types. The method is based on a genetic algorithm acting on a population of trees at each step. It substantially increases the efficiency of the phylogeny space exploration, reducing the risk of falling into local minima, at a reasonable computational time. We have applied GATC to a dataset of simulated cyanobacterial phylogenies, as well as to an empirical dataset of three reference gene families, and showed that it is able to improve gene tree reconstructions compared with current state-of-the-art algorithms.

Conclusion. The proposed algorithm can accurately reconstruct gene trees and is highly suitable for the construction of reference trees. Our results also highlight the efficiency of multi-objective optimization algorithms for the gene tree reconstruction problem. GATC is available on Github at `https://github.com/UdeM-LBIT/GATC`.

1. DIRO, Université de Montréal

7.2. Background

Most biological discoveries can only be made in the light of evolution. In particular, functional annotation of genes is usually deduced from the orthology and paralogy relation between genes, which is inferred from the comparison of a gene tree with a species tree. Therefore, phylogenetic tree reconstruction is an important component of most bioinformatic pipelines. In this paper, we focus on the reconstruction of gene trees.

Standard phylogenetic tools are based on maximum likelihood (ML) or bayesian methods reconstructing a tree from gene sequences (e.g. PhyloBayes [212], PhyML [216], RAxML [217], MrBayes [431]). However, for a variety of reasons due, not only to technical limitations but also to the data itself (sequences too close to each other or conversely too divergent), sequence-only methods often do not allow to confidently discriminate one tree from another.

To address this limitation, more recent gene tree reconstruction methods, designated here as *integrative methods*, include information from the species tree. The idea is to consider, in addition to a maximum likelihood value measuring the fitness of a tree to a sequence alignment (according to a model of sequence evolution), a measure reflecting the evolution of a whole gene family through gene gain and loss. A standard measure of fitness between a gene tree and a species tree is computed in terms of a "reconciliation" score. In a probabilistic framework, the reconciliation score corresponds to the probability density of the gene tree given some parameters (rates of events and branch lengths) under a birth-death and gain model of evolution. For the Most Parsimonious Reconciliation model (MPR), this score corresponds to the optimal number of gene gain and loss events, weighted by their costs, explaining the incongruence between a gene tree and a species tree.

Most integrative methods for gene tree reconstruction assume a simplified model of gene family evolution where gene gain events are reduced to gene duplication (e.g. Giga [27], TreeBeST [28], SPIMAP [282], TreeFix [284], NOTUNG [286], ProfileNJ [453]). In fact, the MPR problem for the Duplication-Loss (hereafter denoted DL) model of gene family evolution is linear-time solvable [256]. By introducing horizontal gene transfer (HGT) events, the Duplication-Transfer-Loss (DTL) model becomes NP-hard in general if time consistency is required for inferred events (unless the species tree is fully dated) [260, 262, 263]. However the MPR problem for the DTL model, with an undated species tree, can still be computed in polynomial time if the time consistency requirement is relaxed [259, 264, 454]. Due to this reasonable time-complexity, some recent phylogenetic softwares have extended the gene family evolution model to transfers (MowgliNNI [289], ecceTERA [295], TreeFix-DTL [290]).

Continuous effort is also made for developing fast probabilistic frameworks capturing HGT events (see [252] for a review of these models).

Integrative methods report gene trees with better accuracy compared with sequence-only methods [26, 251, 289, 290], but they still leave space for improvement, both on tree quality and on computation time. In fact, most of them rely on a two-steps approach, first computing a tree with the best fit to the sequences, and then exploring a tree space surrounding the initial tree to select one minimizing the considered reconciliation distance. From an accuracy point of view, this two-step methodology does not guarantee that the output tree optimizes both the likelihood given the sequence alignment, and the reconciliation measure, as the best fit to the sequences may be lost at the second step. Besides, by considering a single tree at a time, the risk of ignoring a large part of the tree space and falling into a local minimum is high. Other integrative methods (see for example PrIME-DLTRS [251]) compute the joint likelihood associated with a substitution model and DTL event rates, given a fixed, dated and ultrametric species tree and a gene family alignment. They use tree exploration heuristics similar to those found in sequence-only programs for phylogenetic tree reconstruction to explore the solution space, often in a Bayesian-MCMC framework. These methods come at a high computational cost, especially when HGT events are considered, and they are still subject to the risk of being stuck in a local optimum.

In this paper, we present *GATC* (*G*enetic *A*lgorithm for gene *T*ree *C*onstruction), a new software for gene tree reconstruction under the DTL model that can take as input completely unresolved, partially unresolved or fully resolved trees, and outputs a tree minimizing a measure combining both tree likelihood (according to sequence evolution) and a reconciliation score. In other words, it can either be used as a two-step correction method, when input trees are the output of other phylogenetic methods, or as a one-step method resolving a full polytomy (star tree) in a way optimizing fit to both the species tree and the sequences.

With GATC, we explore a new methodological framework based on a Genetic Algorithm (GA), a global search metaheuristic that mimics biological evolution [455]. The ability of GAs to find near-optimal solutions quickly, even for complex models and data makes them suitable for the problem of phylogenetic inference. In fact, the GA methodology has been previously applied to the phylogenetic inference problem, starting with Matsuda in 1996 [456] using a maximum likelihood criterion, Lewis [457] who introduced a subtree swap crossover operator, and other more recent algorithms (e.g., self-adaptive GA [458], Ga-mt [459], METAPIGA [460], GARLI [461]). However, all these algorithms are solely based on sequence information and, as discussed above, are often error-prone in the case of gene tree reconstruction. To our best knowledge, GAs have never been applied to species

tree-aware gene tree reconstruction, although the technique is suitable to the resolution of Multi-Objective Optimization Problems (MOOP).

To measure the performance of GATC, we compared it to current state-of-the-art softwares on a dataset of simulated cyanobacterial phylogenies. Our results show that GATC is more accurate than existing methods, suggesting that it substantially increases the efficiency of the phylogeny space exploration. We also evaluated GATC's ability to infer accurate homology relationships between genes on a standardized, manually curated, dataset of real trees. The predicted relationships were mostly in agreement with the ones inferred from a reference tree, highlighting the efficiency of the framework.

7.3. Methods

7.3.1. Notation on trees

All considered trees are rooted unless explicitly stated. A tree is *binary* if all its internal nodes have exactly two children, and *non-binary* otherwise. Unless stated differently, all trees are considered binary.

We denote by $V(T)$ the nodeset, by $E(T)$ the edgeset, by $\mathcal{L}(T)$ the leafset and by $r(T)$ the root of a tree T. An edge e of $E(T)$ is written as a pair (x, y) of two adjacent nodes where e is an outgoing edge of x. For $e = (x, y)$, x is the parent $p(y)$ of y, while y is a child of x. A node x is an *ancestor* of y, which is denoted $x <_T y$, if it is on the path from y to the root (excluding y). In this case, y is called a *descendant* of x. Similarly, an edge $e' = (x', y')$ is an ancestor of an edge $e = (x, y)$ if it is on the path from y to the root. Given a node x, $T[x]$ is the subtree of T rooted at x and $\mathcal{L}(x)$ the leafset of $T[x]$. Two subtrees $T[x]$ and $T[y]$ are *separated* in T if $x \neq y$, $x \not<_T y$ and $x \not>_T y$. In this case, $\mathcal{L}(x) \cap \mathcal{L}(y) = \emptyset$, if the leaves of T are uniquely labeled by the elements of $\mathcal{L}(T)$.

A *species tree* is a tree S with leaves uniquely labeled and $\mathcal{L}(S)$ being a set of species. Likewise, a *gene tree* is a tree G with leaves uniquely labeled and $\mathcal{L}(G)$ corresponding to a set of genes where each gene g belong to a genome $s(g)$. We denote by $\overline{G}$ the tree obtained from G by replacing each leaf label g_i by its genome $s(g_i)$. Notice that the mapping $s : \mathcal{L}(G) \to \mathcal{L}(S)$ does not have to be injective nor surjective. In particular, $\overline{G}$ may have several equally labeled leaves.

A *reconciliation* of G (or similarly $\overline{G}$) with S (see Figure 7.1) is an extension of s from $V(G)$ to $V(S)$ with additional labels on each internal node x of G, describing the type of evolutionary event that has led to $G[x]$ (duplication, speciation or transfer). G can be expanded to include lost genes.

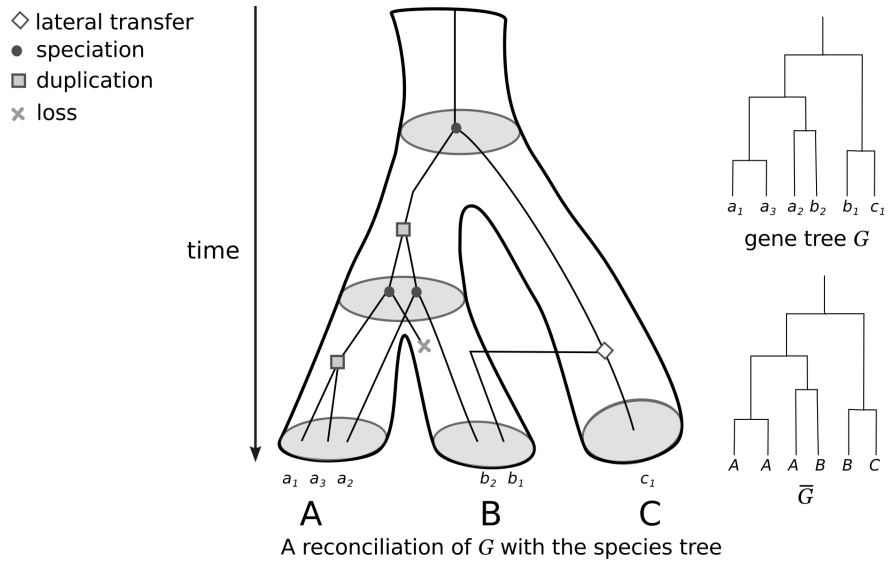


Figure 7.1. A reconciliation between a gene tree G and a species tree. The reconciliation represents a history of the gene family evolution through speciation, gene duplication, gene loss, and HGT, in a way that is consistent with the species tree. Both G and $\overline{G}$ are shown on the figure.

Finally, we refer to the process of removing a leaf l and its associated edge $\Big(p(l), l\Big)$ from a tree T as the *deletion of l from T*.

7.3.2. Vocabulary of Genetic Algorithms

A Genetic Algorithm (GA) is an algorithmic framework mimicking biological evolution. The vocabulary of a GA is filled with biological metaphors. It begins with a *population* of *individuals* whose *chromosomes or genomes* encode specific solutions to the problem of interest. Performance of these individuals in solving the problem is measured by their *fitness score*. To avoid confusion, throughout this paper the word "chromosome" will be used solely to designate the data structure of a genetic algorithm and the word "genome" will be used in its biological meaning to designate the macromolecules containing the genes under study.

At each step, starting from an initial population, a new population is generated using three operators: *selection*, *crossover* and *mutation* [455], which are defined according to the nature of the problem and the encoding of the solution. During selection, the fitness score is used to select individuals for reproduction. Selected candidates are combined using the crossover operator to create new individuals that are then modified by the mutation operator in order to introduce diversity and avoid local optima. With *elitism*, the less fit individuals of the newly obtained population are replaced by the best fit of the previous *generation*, in

order to conserve the best solutions found so far. The process described above is repeated through multiple generations, until an optimal solution (chromosome of the best individual) is obtained or a stop criterion is reached.

This natural selection process generally leads to the improvement of the average population fitness over generations. While GAs often converge to an optimal or near optimal solution, their performance mainly depends on the mechanism for balancing two potentially contradictory objectives: keeping the best solutions found so far, and at the same time efficiently exploring the search space for promising solutions.

7.3.3. The GATC algorithm description

In the rest of the manuscript, we will loosely refer to the tree likelihood given a multiple sequence alignment as *sequence likelihood*.

Given a sequence alignment D and a species tree S, our objective is to find the gene tree G or a set $\mathcal{G}$ of gene trees, with branch length, that are (near) optimal for both the sequence likelihood and the reconciliation score. To solve this problem, our fitness function should reflect both objectives. We will present different ways for computing the fitness score, either by a linear combination of the two scores or by trying to reach a *pareto optimality*. We start by presenting the general framework of the GA.

7.3.3.1. *Solution encoding*

A chromosome σ is defined as (G, θ) where G is a rooted binary gene tree and θ is the set of hyperparameters underlying the evolutionary model. Namely, $\theta = (\lambda, \delta, \tau, e, l, m)$, representing respectively the duplication rate, the loss rate, the transfer rate, the substitution rates across the gene tree edges, branch lengths and the substitution model. Some of these parameters might be kept fixed during the evolutionary process. For example, the substitution model m is usually fixed for all generations, whereas duplication, loss and transfer rates can vary when a probabilistic model is used to compute the reconciliation score. When parsimony is preferred, they correspond to fixed event cost. Branch lengths and edge substitution rates are usually optimized during sequence likelihood computation.

For the probabilistic model of reconciliation, the initial values of the hyperparameters λ, δ, τ and e are randomly drawn from a uniform distribution, unless explicitly provided. The default substitution model used for nucleotide sequences is the GTR model [462] with a gamma distribution to account for rate variation [463], whereas for proteins, the JTT model [211] with gamma-distributed rates is used.

7.3.3.2. *Gene trees in the initial population*

When starting trees are available (from any other tree construction method, integrative or not), they can be used as the population of the first generation in our GA. Otherwise, the trees of the initial population are generated, either randomly or according to a predefined procedure. GATC implementation allows generating the initial trees from a star tree, using PolytomySolver [29] which outputs the most parsimonious trees for the DL-reconciliation score (but not necessarily optimal for the DTL-reconciliation score or the sequence likelihood), or using bootstrapped trees obtained with RAxML [217]. These two methods should be preferred to the initialization with random trees, which can affect the algorithm's convergence.

Notice that two trees G_1, G_2 such that $\overline{G_1} = \overline{G_2}$ have the same reconciliation score, and thus if G_1 is a solution of PolytomySolver, minimizing the DL-reconciliation score, then G_2 is also a solution. Therefore, in this case, to increase the initial population of the GA, additional trees can be obtained by permutation of the genes at the leaves of G_1 in a way respecting the mapping function s.

7.3.3.3. *Computing the sequence likelihood and reconciliation score*

To evaluate the fitness of each chromosome σ_i , $1 < i < n$, in a population of size n, we first compute a vector $\vec{z_i}$ of two components, called the *raw score vector*, containing the sequence likelihood and the reconciliation score. Note that when the objective is to optimize only sequence likelihood, the second component corresponding to the reconciliation score is set to zero.

The sequence likelihood scores $p(D|G,l,m)$ can be computed using the Felsenstein algorithm [214] and the further computational enhancement described by Stamatakis et al. (2004) [464]. In fact, GATC use subroutines from RAxML to compute the sequence likelihood, thus benefiting from both its high computational speed and its large set of substitution models.

As for the reconciliation score, it can be computed under either the probabilistic or MPR model. For the MPR scoring model, we implemented the Bansal algorithm [264] which computes the DTL reconciliation cost between a binary gene tree G and a binary species tree S in time $O(|G||S|)$. Notice that, as explicit transfer pathways are not specified, a DTL scenario is not necessarily possible as it may violate temporal constraints [260]. In fact, a donating and a receiving species must have co-existed at the time of the transfer. Moreover,

in contrast to duplications and losses, HGTs are inter-dependent and can induce contradictory temporal constraints on ancestral species. However, as the reconciliation problem for DTL using undated species tree with the constraint of respecting temporal constraints is NP-hard, the Bansal algorithm remains a good alternative for computing a reasonable DTL reconciliation score. In the absence of HGTs, we compute the DL reconciliation score using a linear-time algorithm [256], to speed up calculations.

For the probabilistic scoring model, we have implemented the DTL model first described by Tofigh [320, 465] and used by PrIME-DLRS [251]. It is based on a birth-death model of evolution including rates for gene duplication, transfer and loss that requires discretization of a dated species tree and numerical resolution of ordinary differential equations. We refer the readers to the Supplementary Material of [251] for a thorough description of how the probability density of the reconciliation is computed.

Rather than minimizing the reconciliation score and maximizing likelihood, it is easier to minimize both measures simultaneously. For this reason, we take the negative log value when likelihood is used for any of the two scores. Therefore, it has to be understood that the best-adapted individuals will be those with the lowest fitness.

7.3.3.4. *Computing the fitness score*

Given a raw score vector $\vec{z_i}$ for a chromosome σ_i, a weight vector $\vec{w}$ and a scaling function ϕ, we define the fitness score f_i of σ_i as $f_i = \vec{w} \cdot \phi(\vec{z_i})$. In other words, f_i corresponds to the weighted sum of the two components of the raw score vector, scaled by a function ϕ. The simplest definition of ϕ is the identity function $\phi(\vec{z}) = \vec{z}$. An alternative is to standardize each score to a zero-minimum resulting in the following formulation : $\phi(z_i^k) = z_i^k - \min_i(z_i)^k$ for $1 \leq k \leq 2$ and $1 \leq i \leq n$. However, for this latter scaling function, fitness is not comparable between individuals of different generations.

Using the method described above for computing f_i transforms our problem into a single objective minimization problem and is suitable when both components of z_i are log likelihood values, since it is related to the joint weighted probability density for sequences data and reconciliation to the species tree.

When the reconciliation score is computed using parsimony, combining the two scores this way might not be optimal, since it is hard to scale one of the component of $\vec{(z)}$ to the same comparable range as the other. Furthermore, the biological interpretation of this combined score is not straightforward. Instead, we compute a set of *pareto optimal solutions* for this multi-objective optimization problem (MOOP). Several evolutionary based techniques have

Algorithm 2 Compute next generation population P_{k+1} from P_k

procedure COMPUTENEXTPOP(P_k)
  Compute P'_k, the offspring population of P_k
  $P^*_k \leftarrow P_k \cup P'_k$
  Evaluate z_i for all $\sigma_i \in P^*_k$
  Compute the dominance rank d_i for each $\sigma_i \in P^*_k$
  $w \leftarrow 1$
  while $\exists\, \sigma_i \in P^*_k \mid d_i = 0$ **do**
    $Wave_w \leftarrow \{\sigma_i \mid d_i = 0\}$
    Set a shared fitness for all $\sigma_i \in Wave_w$ as w
    $P^*_k \leftarrow P^*_k \setminus Wave_w$
    Compute the dominance rank d_i for each $\sigma_i \in P^*_k$
    $w \leftarrow w + 1$
  end while
  for $\sigma_i \in P^*_k$ **do**
    set the fitness of σ_i as $w + d_i$
  end for
  $P_{k+1} \leftarrow \bigcup_w Wave_w \cup P^*_k$
  return the first $|P_k|$ of P_{k+1} according to fitness
end procedure

been developed for MOOP [466]. Here we will use a technique similar to the widely known NSGA (Non-dominated Sorting Genetic Algorithm) [467].

A raw score vector $\vec{z_i} = (z_i^1, z_i^2)$ is said to *dominate* another vector $\vec{z_j} = (z_j^1, z_j^2)$, denoted as $\vec{z_i} \prec \vec{z_j}$, iff $\vec{z_i} \neq \vec{z_j}$ and $z_i^1 < z_j^1$, $z_i^2 < z_j^2$. We are interested in finding the set of non-dominated solutions called *pareto set* (PS) and denoted as :

$$PS = \{\sigma_i \mid \nexists\, \sigma_j, \vec{z_j} \prec \vec{z_i}\}$$

At the end of the GA's evolutionary process, the pareto set represents the set of pareto optimal solutions. In contrast with classical genetic algorithms, computing the pareto set requires to consider simultaneously a parent population P_i and its offspring P'_i, as optimal solutions from P_i can be lost if we use P'_i as the population P_{i+1} of the next generation.

Algorithm 2 illustrates the way fitness is computed for all individuals of a generation. It proceeds in a *wave* fashion, selecting the non-dominated individuals from the population $P^* = P_i \cup P'_i$, assigning them a shared fitness score, and then removing them from P^*. This process is repeated while increasing the fitness score for the individuals in the new waves, until the expected population size per generation is met or there are no non-dominated individuals anymore. In the latter case, the fitness of the remaining individuals is computed

as the sum of their *dominance rank* (number of individuals that dominates an individual) and the fitness of the last wave. This process ensures that individuals belonging to the same wave have the same fitness and as such the same probability to reproduce, since their are equally pareto-optimal. The n individuals with the best fitness constitute P_{i+1}. Selection, crossover and mutation operators can be applied to P_{i+1} resulting in offspring P'_{i+1}.

7.3.3.5. *Selection*

GATC implements multiple classical selection methods. Individuals can either be selected for crossover using the tournament selector [468] or by using the roulette wheel selector which chooses individuals with probability inversely proportional to their fitness values (recall that the best individuals have the smallest fitness value). Alternatively, the random uniform selector can be used, which gives an equal reproduction probability to all individuals regardless of their fitness. Selected individuals are used in the crossover operator to produce the individuals of the next generation.

7.3.3.6. *Crossover*

In the crossover operators implemented in GATC, two offsprings are created from two parent chromosomes. Each offspring inherits its hyperparameter θ from one of its parents, while its gene tree is obtained from the combination of the two parental trees.

Given trees G_i and G_j respectively from parent σ_i and σ_j, the first offspring is obtained with the subtree swap crossover operator [457], achieved by the following actions:

1. Select a subtree $G_i[x]$ from G_i (the root is excluded)
2. Delete all leaves from G_j that are also in $\mathcal{L}(x)$;
3. Regraft $G_i[x]$ to a random edge of G_j to obtain the offspring tree G'_j.

The second offspring tree G'_i, is obtained in a similar way by selecting a subtree from G_j and regrafting it in G_i. The crossover operator is illustrated on Figure 7.2A.

In the special case where the objective is to only optimize the sequence likelihood, under the hypothesis that the reconciliation score is already optimal, this crossover operator is not applicable as it does not preserve the reconciliation score. Instead, the offspring trees are created by exchanging two subtrees $G_1[x]$ and $G_2[y]$ such that $\overline{G_1[x]}$ and $\overline{G_2[y]}$ are isomorphic with respect to the labels at their leaves (see Figure 7.2B).

7.3.3.7. *Mutation*

For a chromosome $\sigma_i = (G_i, \theta_i)$, a mutation is performed either on the tree G_i or on the rates λ, δ, τ, e unless their values are fixed. Mutations on the rate parameters consist in

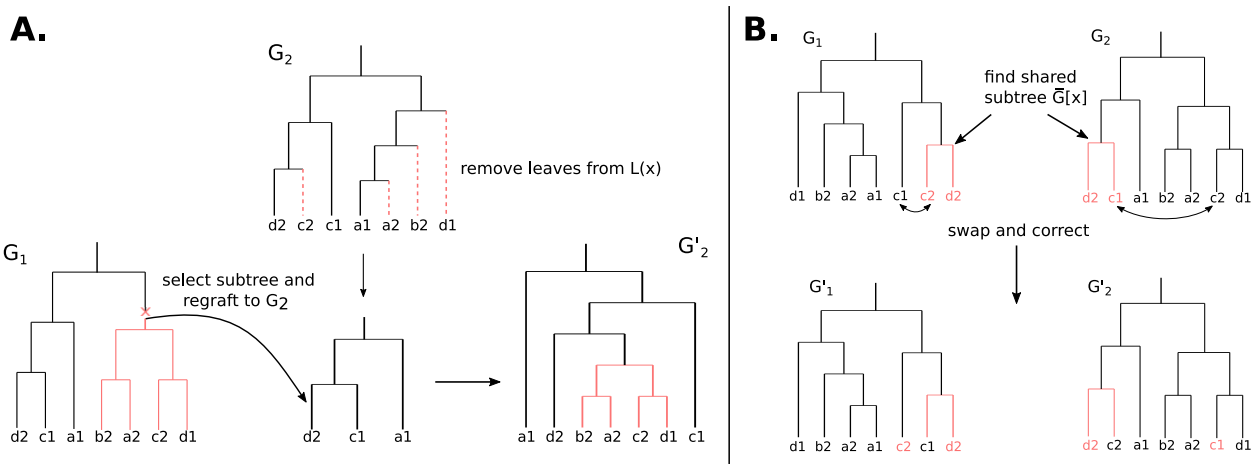


Figure 7.2. Crossover operator. A. Subtree swap. A subtree $G_1[x]$ (in red) is pruned from G_1 then regrafted to a random branch of G_2 after deleting from G_2 its leaves that also appear in $G_1[x]$ (shown in dotted lines). To obtain the second child, a similar operation is performed from G_2 to G_1 **B. Subtree swap preserving reconciliation**. Two subtrees $G_1[x]$ and $G_2[y]$, respectively from G_1 and G_2, such that $\overline{G_1[x]} = \overline{G_2[y]}$ are swapped and the remaining leaves are corrected to conserve the same leafset as the parent.

drawing a new value from their distribution. On the other hand, a mutation operates on G_i by applying a topological modification. GATC uses SPR (Subtree Pruning and Regrafting) and re-rooting operations (see Figure 7.3A-B) to generate a new tree topology. As with the crossover operator, when only sequence likelihood has to be optimized, reconciliation score should be preserved after mutations. For this purpose, mutations are performed by permuting the genes assignment to the leaves of G_i in such a way that only genes belonging to the same species are allowed to switch places (see Figure 7.3C).

7.3.3.8. *Stop criteria*

We proposed several criteria to stop the GA evolution. The simplest ones are to terminate when a maximum number of generations or a time limit are met, or when all individuals converge to a single fitness value. Aside from these criteria, we propose another simple termination criterion called *population-AU criterion* that is based on the use of a reference ML tree. Under this criterion, evolution is stopped when all the individuals in the current population are statistically equivalent to the known ML tree, according to the AU test [245]. This stop criterion allows for a good performance when the objective is restricted to the optimization of sequence likelihood.

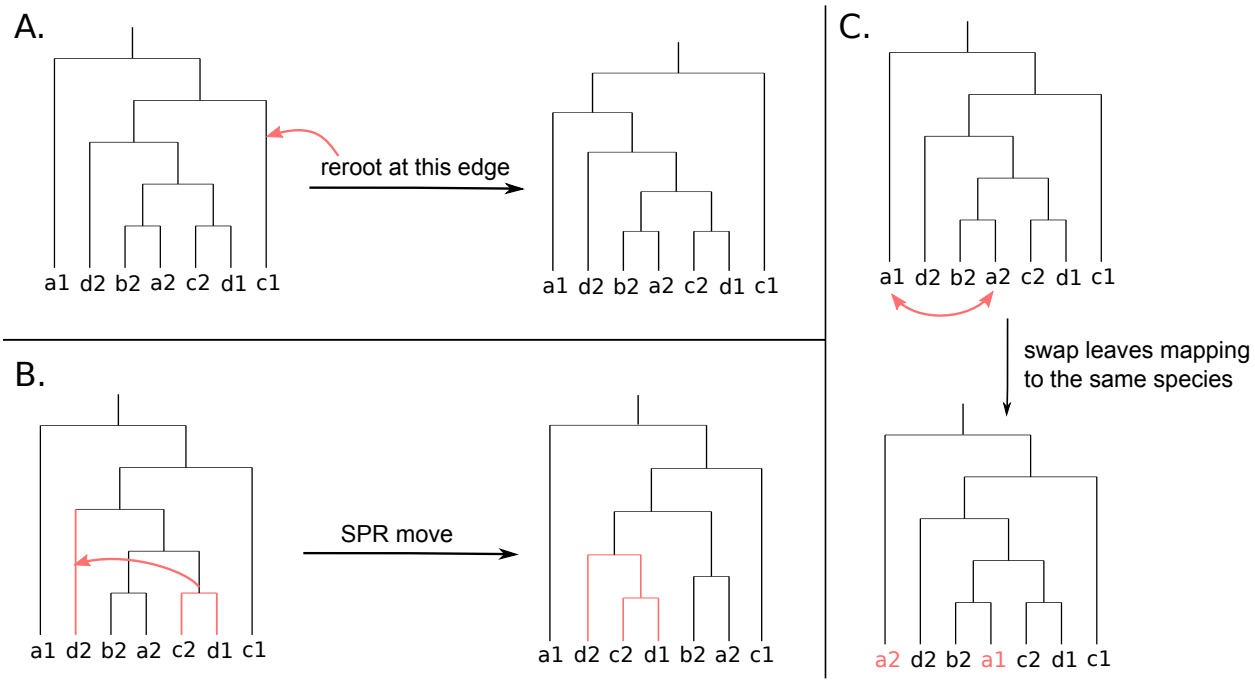


Figure 7.3. Mutation operator. A. Re-rooting. The tree is rerooted at a random edge. **B. SPR move**. A subtree is pruned from the tree and regrafted to another edge. **C. Mutation preserving reconciliation cost**. Two leaves l_1 and l_2 such that $s(l_1) = s(l_2)$ are swapped. This mutation only alter the sequence likelihood.

7.4. Results and discussion

To measure the efficiency of GATC in reconstructing accurate gene trees, we compared its performance, on a simulated dataset, to four different gene tree reconstruction methods: RAxML [217], MowgliNNI [289], TreeFix-DTL [290], ecceTERA [295] an ProfileNJ [453]. In contrast to RAxML which is a sequence-only method, the former four methods use both sequences and species tree information. We also used GATC to reconstruct the gene trees of three gene families for which reference trees have been proposed [469]. We will entirely focus on evaluating GATC's performance under the MPR model, as it is our main contribution and also because DTL-reconciliation scores can be computed significantly faster under this model.

7.4.1. Evaluation on a simulated Cyanobacteria histories dataset

We used the simulated cyanobacteria dataset of Szöllősi et al. (2013) [26] publicly available at `http://datadryad.org/resource/doi:10.5061/dryad.pv6df`. This dataset consists of 1099 gene families from 39 cyanobacteria species along with a well-resolved

dated species tree. To construct the dataset, the gene families were retrieved from HOGENOM [426] and multiple alignments were performed on these families with Muscle [193]. For each alignment, an MCMC sample of at least 3000 trees was obtained with PhyloBayes [229] and used to reconstruct an amalgamated tree with ALE [26]. These trees were used to simulate new multiple alignments of artificial sequences under the LG model with a gamma distribution. We refer to [26] for a more detailed description of the construction of the dataset.

From each of the 1099 simulated artificial sequence alignments, we reconstructed an initial tree using RAxML (LG + Gamma, 100 bootstraps). The RAxML trees (with bootstrap values) were used as input for all programs being compared against GATC.

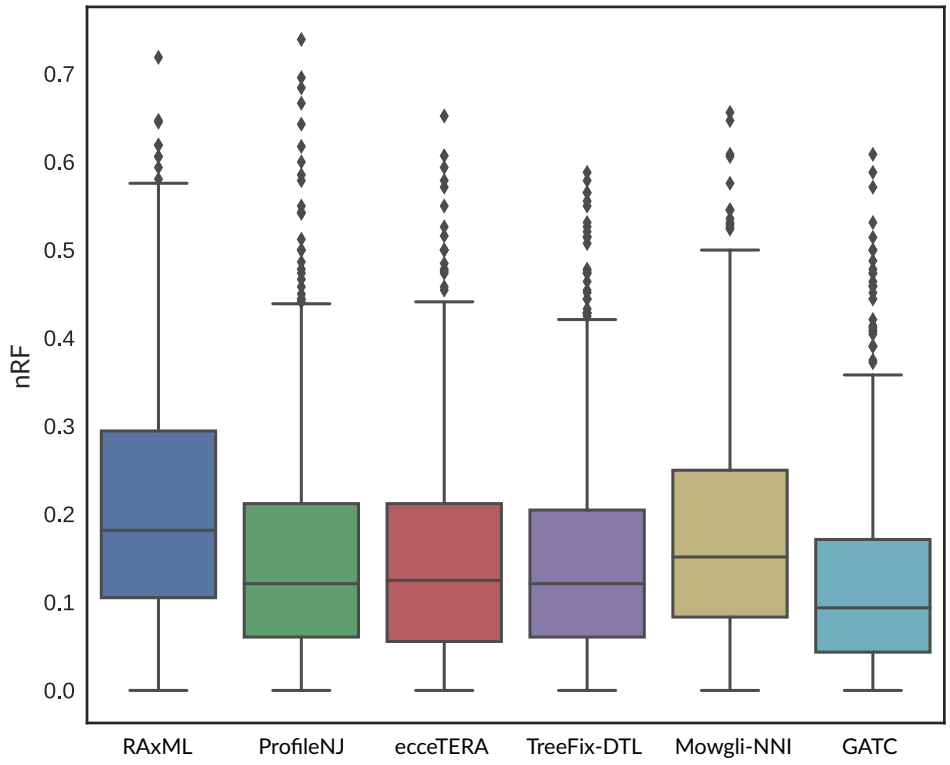


Figure 7.4. The accuracy of RAxML, ProfileNJ, ecceTERA, TreeFix-DTL, Mowgli and GATC on a dataset of simulated Cyanobacteria histories: we measure the normalized Robinson-Foulds distance of the reconstructed trees to the true gene trees for all 1099 gene families. GATC achieves the best accuracy on the simulated dataset, followed by TreeFix-DTL.

For all programs, we used fixed DTL rates ($\lambda = 2$, $\tau = 3$, $\delta = 1$) except for ProfileNJ which supports only a DL model of reconciliation and for which we took $\tau = \infty$. We ran TreeFix-DTL with default parameters and LG + Gamma for the evolutionary model. As it requires rooted trees, the input RAxML trees were rooted using the mid-point rooting method [172]. MowgliNNI, ecceTERA and ProfileNJ were run with a threshold of 0.7 for the contraction of for weak edges. Note that ProfileNJ and ecceTERA can output several solutions from which users can later select a tree according to some other measure (sequence likelihood for example). In our comparison, we only consider the first solution returned by both methods, as this selection process is not part of the methods, and also removes their running time advantage. We ran GATC with the following parameters: a maximum of 50 generations, a time limit of 90 minutes per gene family, LG + Gamma as the model of evolution and parsimony for DTL-reconciliation. We used the tournament selector as the selection operator and set the crossover and mutation rates to 0.8 and 0.5 respectively (see Additional file 1: section 1 and Figure S1, for a discussion on the effect of crossover and mutation rates on accuracy). To construct the initial population of the GA, we used PolytomySolver's resolutions of RAxML trees after contraction of edges with support less than 0.7. To keep the GA population size fixed at 30, we randomly removed or duplicated chromosomes from the initial population until its size became 30. We also used the population-AU as additional stopping criterion with the RAxML tree being the known best ML tree and a significance level $\alpha = 0.05$. When there were more than one tree in the pareto optimal set, the tree with the lowest DTL-reconciliation score was returned as GATC final solution.

We measured the accuracy, defined as the normalized Robinson-Foulds distance between each reconstructed tree and the true tree. As shown in Figure 7.4, trees reconstructed with species tree-aware algorithms were more accurate than RAxML's trees. This result was expected since it has been shown several times that integration of species tree information usually improves gene trees reconstruction. GATC, in particular, achieves a better accuracy than other methods, due to its improved tree space search efficiency. The algorithm also appears to be robust, in some extent, to errors in the species tree topology (see Additional file 1: section 2 and Figure S2). However, it should be noted that in order to obtain accurate results, there is a need to allocate a considerable time for the evolution of the GA. As such, the algorithm is much slower, in comparison to ProfileNJ and ecceTERA which can output solutions in a few seconds. To our surprise, ProfileNJ was almost as accurate as the second best method (TreeFix-DTL), although it only supports a DL model of reconciliation and HGTs were present in the dataset. It is possible that most edges with weak support were not involved in HGT events, which can explain the observed performance of ProfileNJ.

7.4.2. Evaluation on an empirical dataset

In an attempt to establish a benchmark for comparing orthology prediction methods, Boeckmann et al. (2011) [469] proposed manually curated "gold standard" gene trees for three well-conserved gene families : the Popeye-domain containing family (POP), the NOX 'ancestral-type' subfamily of NADPH oxidases (NOX) and the V-type ATPase beta subunit (VATP).

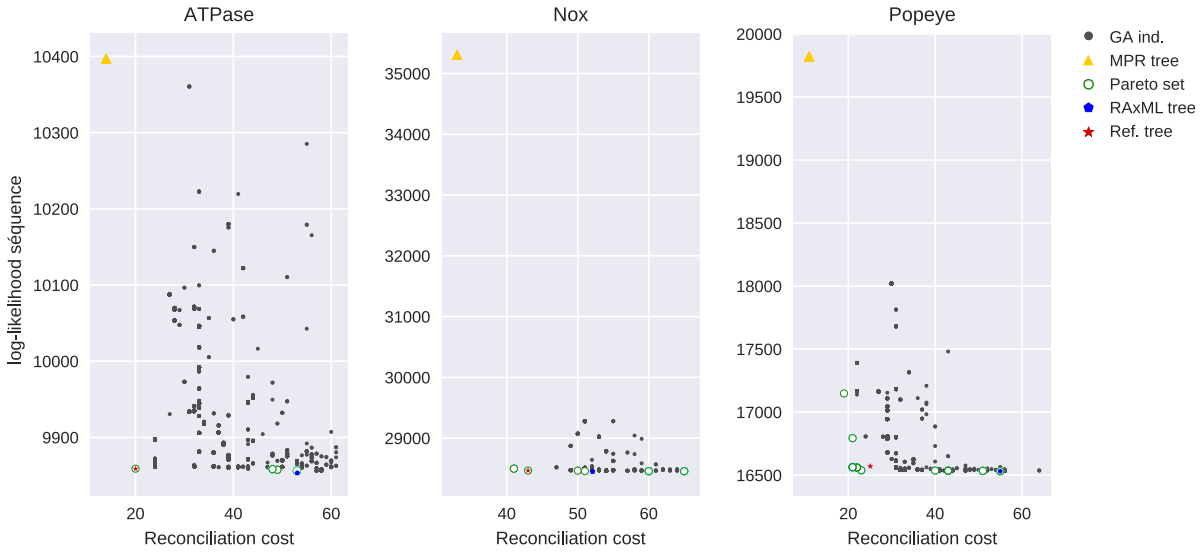


Figure 7.5. Distribution of individuals' raw scores during evolution on three "gold standard" gene families. The scores of the ML tree obtained with RAxML, the MPR tree for the DL score, and the reference gene tree of [469] are also shown. Note that for a fair comparison, the RAxML tree reconciliation score corresponds to the best rooting score, whereas the MPR tree sequence likelihood corresponds to the tree with the minimum negative log likelihood in the set of equivalent MPR trees. For the sake of visibility, we increased the size of each data point. The "best tree" is expected to be located in the lower left corner. For the ATPase and Nox families, the reference tree was present in the set of pareto optimal trees returned by GATC. For the Popeye gene families, the reference tree was located in the proximity of a cluster of pareto optimal solutions.

These gene families have been re-analyzed here to assess the performance of GATC on an empirical dataset. The reference species tree used was obtained from SwissTree [470]. Protein sequences from genomes not found in the species tree were removed and the remaining sequences aligned with Muscle [193]. GATC was used to reconstruct the corresponding trees for each gene family with the initial population of trees obtained from bootstrap replicates. We used the same parameters as above except for the DTL events cost, which was changed

to: ($\lambda = 1$, $\tau = \infty$, $\delta = 1$). Here, we prohibit HGT events since they are not expected in the dataset. We also set the maximum number of generations to 300 and the maximum time of evolution to 3h per gene family. For comparison, the average time needed by RAxML to obtain the best ML trees is 2.4h.

In order to measure the accuracy of GATC, we investigated how close the reference trees were to the set of pareto optimal trees. Figure 7.5 shows the distribution of individuals' scores, over generations, during the GA evolution for each gene family. We were able to retrieve the reference tree for the NOX and VATP gene families, whereas the reference tree for the POP family was located close to a cluster of pareto optimal trees. From the same figure, it can also be seen that even though the ML and MPR trees theoretically belong to the pareto optimal set of the complete tree space, they are often located far from the desired optimal result.

Table 7. II. Comparison between the reference tree of the Popeye family and the pareto optimal trees returned by GATC.

	normRF distance	**Orthologs**		**Paralogs**	
		Prec.	Rec.	Prec.	Rec.
Tree 1	0.260	0.763	0.942	0.971	0.871
Tree 2	0.260	0.765	0.941	0.971	0.873
Tree 3	0.087	0.902	0.983	0.992	0.894
Tree 4	0.109	0.829	0.866	0.940	0.922

Since the reference gene tree was not obtained for the POP family, we report the precision and recall of orthologous and parologous genes inference from the solutions returned by GATC, compared to the proposed reference tree (Table 7. II). Note that GATC only outputs ten trees from the 30 individuals of the final population resulting in four unique trees (see Figures S4-7 of Additional file 1: Supplementary material). We computed precision and recall for the two types of gene relationships as follows:

$$Precision = \frac{TP}{TP + FP} \quad , \quad Recall = \frac{TP}{TP + FN}$$

where TP corresponds to the number of shared pairs of orthologs/paralogs with the reference tree, FP corresponds to the number of predicted pairs of orthologs/paralogs not present in the reference tree, and FN to the number of missed orthologs/paralogs. As shown on Table 7. II, the precision and recall for the inferred gene relationships were high for all four solutions. Difference between GATC's solutions and the reference POP gene family tree (Figure S3

of Additional file 1: Supplementary material) can mostly be explained by the fact that duplication nodes were often placed lower in the solutions, resulting in a fewer number of losses and consequently lower reconciliation scores.

It is hard to argue whether the proposed reference tree represents the real evolutionary history of the gene family over our pareto optimal solutions. In fact, from Figure 7.5, it can be seen that some pareto optimal solutions were better than the reference POP gene tree for both scores, suggesting that they could be of higher quality. As the true evolutionary histories of gene families are hardly known, relying on high-quality phylogenetic gene trees for biological analyses is preferable.

In summary, our results on the empirical dataset demonstrate how a GA framework can be used for the inference of gene trees with high accuracy.

7.5. Conclusion

Algorithms for constructing gene trees from multiple sequence alignments are widely used. However when a reliable species tree is available, it is preferable to use species tree-aware methods which are often more accurate. In this work, we have presented a GA framework for the reconstruction of gene trees using both sequences and species tree information. From the comparison with existing methods, we have shown that this framework, implemented in a software called GATC, outputs more accurate gene trees.

As the true evolutionary history of a gene family does not always correspond to the most parsimonious one, GATC assumes instead that the true gene tree can, most likely, be found in the pareto optimal set of the search space. Therefore, given enough time, the algorithm will converge to a set of candidates containing that tree. Although this hypothesis was supported by our results on the empirical dataset, it does not necessarily hold for all gene families. For example, since our reconciliation model does not consider Incomplete Lineage Sorting (ILS), the efficiency of GATC is expected to decrease in presence of ILS. Indeed, signals of deep coalescence leading to incongruence between species and gene tree would be explained by DTL events, possibly resulting in incorrect trees. Moreover, another problem persists when there are several trees in the final pareto set, because alternative criteria for discriminating between these equivalent candidates are required. In its current implementation, GATC outputs solutions sorted by either the sequence likelihood or the reconciliation score.

Despite the good results we obtained by using GATC, one fundamental aspect that should be addressed in order to improve efficiency is the required evolution time. Indeed, running time cannot be accurately estimated, especially when the starting trees have poor quality. When ML or Bayesian trees have been inferred beforehand, it may be appropriate

to set the maximum evolution time to the time required to find the best ML tree. As the underlying idea behind GAs allows for easy parallelism, running time can be dramatically reduced. Balance between scalability to large datasets and search efficiency would likely be achieved by carefully selecting the different genetic operators and the stopping criteria. Finally, to avoid being trapped in local optima, multiple replicate searches, using different settings (such as DTL, crossover and mutations rates, population size and initialization) can be performed in parallel with exchange of information through a migration operator.

Chapitre 8

Conclusion

La phylogénie moléculaire, bien qu'ayant révolutionné l'étude de l'évolution des génomes et des familles de gènes, n'est pas sans limites. Pour garantir l'exactitude des analyses, il faudrait s'assurer que chacune des étapes, de l'acquisition des séquences à la reconstruction des phylogénies, en passant par l'annotation et l'alignement, soit effectuée de manière précise et avec le moins d'erreurs possible. Le besoin d'outils efficaces pour ces tâches se fait donc ressentir, d'autant plus que les progrès récents en biologie moléculaire ont facilité l'accès à de larges quantités de données.

Dans cette thèse, nous avons décrit plusieurs nouvelles méthodes algorithmiques qui ciblent individuellement certaines des problématiques liées à l'inférence de l'histoire évolutives des familles de gènes. Nos algorithmes peuvent être intégrés dans un même pipeline pour l'étude de l'évolution des génomes.

Plus précisément, nous avons développé des méthodes pour la prédiction de déviations du code génétique et d'autres pour la correction et la construction efficace des arbres de gènes. Nous avons montré l'efficacité de nos algorithmes aussi bien sur des données simulées que sur des données réelles. Notre travail ouvre la voie à de nouvelles investigations. Nous détaillons ci-dessous plusieurs perspectives pour des travaux futurs.

8.1. Nouvelles problématiques pour l'étude de l'évolution du code génétique.

Tout d'abord, nos résultats obtenus sur l'évolution du code génétique mitochondrial chez certaines algues vertes (chapitre 5) soulèvent de nouveaux questionnements sur les mécanismes d'évolution du code. Nous montrons que la réassignation des codons AGG chez les Sphaeropleales est initiée par la perte du Arg-ARNt(UCU) ancestral. Ce même ARNt est

également perdu chez les Angiospermes, mais une stratégie complètement différente, par importation d'ARNt nucléaires, est utilisée pour le décodage. On pourrait donc se demander pourquoi il existe une si grande disparité entre les stratégies adoptées par les génomes après la perte d'ARNt, et quels sont, s'ils existent, les facteurs qui contribuent à la détermination de la stratégie adéquate à utiliser. Par exemple, pourquoi de nouveaux ARNt mutants sont apparus chez les Sphaeropleales, alors que les Chlamydomonodales, leur clade soeur, importent plutôt les ARNt du cytosol ? De même, on pourrait se demander si l'évolution du code génétique a interrompu le processus de réduction de l'ADNmt chez les Sphaeropleales. Une étude systématique plus approfondie pourrait aider à établir des pistes de solutions pour ces nouvelles problématiques.

Un autre résultat de notre travail sur les génomes mitochondriaux d'algues vertes concerne l'évolution de l'ADNmt de *Pycnococcus provasolii*. Nous montrons en effet que les deux ARNt mitochondriaux responsables de la réassignation des codons AUA et UGA dans ce génome présentent certaines particularités en comparaison à leurs isoaccepteurs dans les autres chlorophytes. Leur origine demeure un mystère, raison pour laquelle nous avons favorisé la théorie du transfert horizontal de gènes dans le chapitre 5. En analysant l'usage des codons dans les régions codantes de l'ADNmt des algues vertes, nous observons effectivement, que le profile d'usage des codons de *P. provasolii* est bien différent (Figure 8.1). Cette différence est-elle due à une vitesse évolutive plus élevée ou est-elle plutôt causée, comme nous le suspectons, par l'acquisition de matériel génétique d'un autre génome ? La réponse à une telle question apporterait aussi une lumière sur l'évolution des premières espèces ayant divergé au cours de l'évolution des chlorophytes.

Dans le chapitre 4, nous avons discuté des forces et limites de CoreTracker, notre algorithme pour la prédiction des réaffectations de codons. Bien que le Random Forest de CoreTracker n'ait été entraîné que sur les séquences mitochondriales de métazoaires, nous montrons que l'algorithme est capable de prédire les modifications de code génétique dans d'autres génomes mitochondriaux (levures et algues vertes) et même dans les génomes nucléaires de levures (voir annexe A par exemple). Ces résultats confirment ainsi que certaines caractéristiques des réassignations de codons sont uniformes et indépendantes des mécanismes sous-jacents, et par conséquent peuvent être correctement capturées par les algorithmes de prédiction. Une extension logique de CoreTracker serait d'étendre, périodiquement, l'ensemble d'entraînement à de nouvelles réassignations afin de maintenir l'efficacité de la méthode.

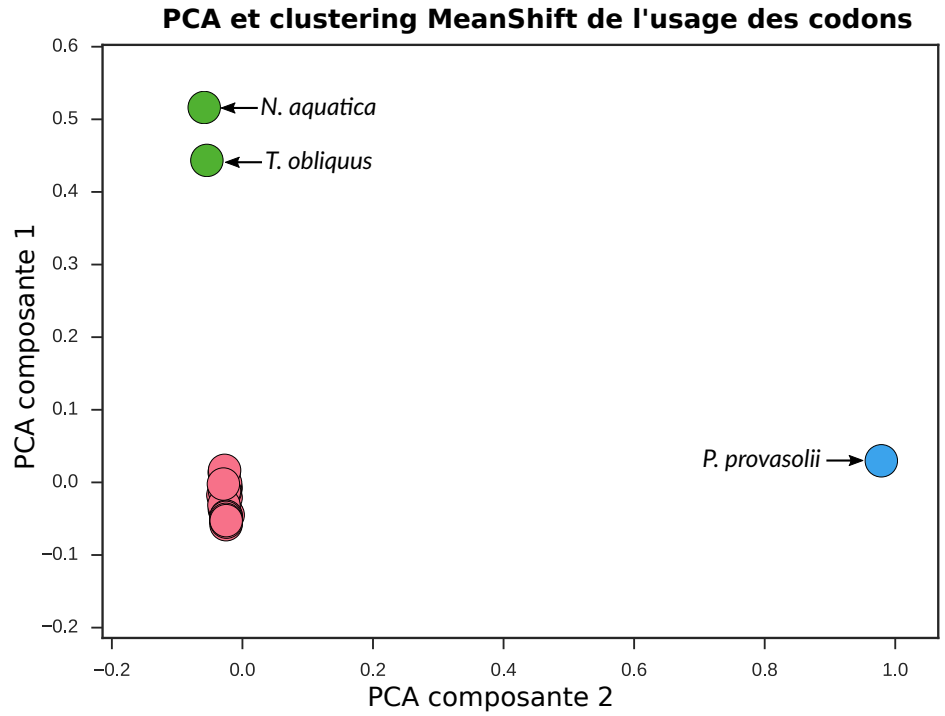


Figure 8.1. Usage de codons dans les séquences codantes des génomes mitochondriaux d'algues vertes. La figure montre un clustering par MeanShift, en fonction de leur usage en codons. Nous utilisons les deux premières composantes PCA, expliquant 87% de la variabilité de l'ensemble des données, pour faciliter la représentation. Certains génomes d'intérêt sont directement indiqués sur la figure (le cluster en rouge comprend tous les autres génomes : Chlorophyta + Streptophyta utilisés dans le chapitre 5).

Deux limites importantes de CoreTracker dont nous discutons dans le chapitre 4 sont l'absence de la prise en compte de l'évolution des ARNt, et la difficulté à différencier efficacement les altérations du code génétique des évènements de recodage. Dans le chapitre 5, nous abordons le premier problème en proposant un cadre d'étude intégrant l'analyse de l'évolution des ARNt, la difficulté majeure étant l'automatisation de cette étape. En effet, même si les gènes d'ARNt peuvent être efficacement identifiés [73, 365], une classification exacte en fonction de leur identité reste difficile [471]. Par ailleurs, les phylogénies construites, à large échelle, à partir des séquences d'ARNt sont souvent peu fiables, en raison du nombre limité de sites informatifs disponibles sur ces séquences (76 nucléotides seulement, dont plusieurs soumis à des contraintes structurales). Très souvent, on est contraint d'utiliser des algorithmes de clustering à partir d'une matrice de distance, en espérant que les alignements soient justes. Reconstruire l'histoire évolutive des ARNt est aussi un problème à part entière. Dans notre cas spécifique, nous ne ciblons que quelques ARNt (ceux potentiellement

impliqués dans une réassignation de codons), ce qui rend le problème beaucoup plus simple. Cependant, même sous ces conditions, il peut être difficile de retracer l'évolution des ARNt, en raison des évènements fréquents de duplications, pertes et réarrangements génomiques qui perturbent le contenu et l'ordre de ces gènes dans le génome. Par exemple, dans les génomes mitochondriaux de Sphaeropleales étudiés dans le chapitre 5, plusieurs réarrangements successifs ont eu lieu au sein de génomes proches, ce qui fait que l'ordre des ARNt dans ces génomes est très peu conservé, compliquant ainsi les analyses.

L'étude rigoureuse de l'évolution des ARNt nécessite, toutefois, de développer des algorithmes spécifiques permettant de suivre l'évolution des répertoires d'ARNt, étant donné une phylogénie d'espèces et un ordre actuel des ARNt dans des génomes d'intérêt. En d'autres termes, il s'agit d'une variante du problème "*small phylogeny*", bien étudié pour une variété de contraintes et de modèles évolutifs, et connu pour être difficile [472]. Une complication supplémentaire s'ajoute aussi dans notre cas, puisqu'il faut non seulement tenir compte des évènements de duplications, pertes, réarrangements, mais aussi de la possibilité d'un changement d'identité des ARNt au cours de l'évolution. Pour simplifier le problème, cette dernière contrainte pourrait être définie par une matrice qui donne pour chaque ARNt aux feuilles, une probabilité pour son identité, que l'on suppose variable le long des branches de la phylogénie. Le développement d'un tel algorithme constituerait une avancée scientifique intéressante pour la compréhension de l'évolution des ARNt.

En ce qui concerne la seconde limitation, nous pensons qu'il est en théorie possible de différencier entre recodage et modification du code génétique. Tout d'abord l'analyse des ARNt devrait fournir suffisamment d'informations pour permettre de distinguer entre les deux phénomènes. Mais, même sans cette analyse et juste en se basant sur les séquences codantes, il serait déjà possible de les distinguer. L'hypothèse supportant cette affirmation est basée sur la différence fondamentale entre recodage et modification du code génétique : contrairement à la modification du code génétique, le recodage n'affecte que quelques gènes du génome. On s'attendrait ainsi à voir les changements localisés dans quelques gènes uniquement, et puisque la distribution des changements à travers les gènes fait partie des prédicteurs utilisés par CoreTracker, une prédiction positive serait rarement obtenue. Il est à noter toutefois qu'il existe des évènements de recodage comme l'édition d'ARN, qui sont extrêmement répandus dans certains génomes.

L'édition d'ARN est une modification post-transcriptionnelle qui change la séquence des ARN par conversion, insertion ou délétion de nucléotides à certains sites spécifiques. Elle peut ainsi contribuer à l'existence de différences importantes entre les séquences de gènes et celles des protéines correspondantes, en modifiant l'interprétation des codons ADN. Dans

les génomes mitochondriaux et chloroplastiques de plantes terrestres, l'édition d'ARN par conversion C→ U et parfois U→ C est très fréquente. Dans certains de ces génomes, on estime même le nombre de sites édités à plus de 3000 [473], et ces sites sont distribués dans presque tous les gènes. La prédiction de réassignations dans ces génomes peut donc devenir compliquée.

Lorsque l'information sur les sites édités au sein des gènes est disponible, deux stratégies s'offrent naturellement. La première, celle utilisée dans le chapitre 5, consiste à corriger les séquences au préalable avant la prédiction. La seconde consiste à filtrer *a posteriori* les prédictions pour retirer celles compatibles avec l'édition d'ARN. Par exemple, en utilisant les séquences nucléotidiques originales des Angiospermes, CoreTracker prédit les réassignations CGG(Arg→Trp), UCU(Ser→Phe), et UCC(Ser→Phe), bien qu'aucune d'elles ne soit entièrement validée (elles ne passent pas les deux tests de validation). On se rend bien compte que ces prédictions sont parfaitement compatibles avec l'édition C → U chez les plantes.

Lorsqu'on n'a pas accès à l'information sur la présence d'édition au sein des génomes, le besoin de différencier entre réassignation et édition se fait ressentir. Idéalement, il faudrait être capable de prédire, non seulement les réassignations de codons, mais aussi l'édition d'ARN. Pour ce dernier problème, des algorithmes spécifiques à des types précis d'édition, qui comparent les données transcriptomiques d'ARNm matures aux séquences génomiques afin d'identifier les positions divergentes existent [474, 475]. De même des méthodes telles que PREPACT [476] et PREP-Mt [477] pour la prédiction d'édition C→U chez les plantes par comparaison avec des données de référence bien annotés rencontrent également du succès. Ces méthodes sont toutefois restrictives. Qu'arrive-t-il lorsqu'il n'existe pas de données de référence ou transcriptomiques pour les génomes d'intérêt ? Ou lorsque la distance évolutive les séparant des génomes de référence est trop grande pour une comparaison significative ? Des approches alternatives sont donc nécessaires. Une extension naturelle de CoreTracker, que nous pensons réalisable, serait de modifier la méthode pour qu'elle puisse différencier efficacement entre édition d'ARNm et réassignations de codons (voir Figure 8.2).

Nous proposons une première stratégie globale de prédiction par apprentissage machine (méthode 1 sur la Figure 8.2). Quelques études précédentes ont en effet répertorié une liste de caractéristiques disposant de puissance prédictive pour l'édition [478–480]. Alternativement, si nous faisons un parallèle entre l'édition et le problème d'identification de variants génétiques, il serait justifié d'explorer les réseaux de neurones convolutionnels et/ou récurrents, dont l'efficacité pour ce type de problème a été prouvée [481–487].

La deuxième stratégie que nous proposons exploite la capacité de CoreTracker à identifier les positions d'un alignement qui semblent compatibles avec un changement d'identité.

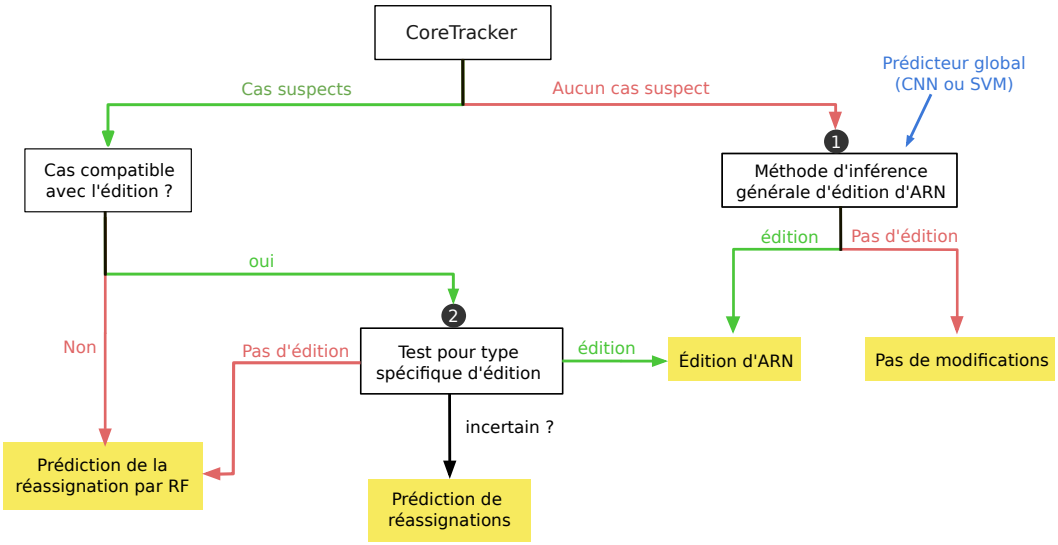


FIGURE 8.2. Pipeline qui étend CoreTracker afin de prédire l'édition d'ARN. 1. Algorithme général de prédiction de l'édition par apprentissage machine, puisque le type d'édition n'est pas connu. 2. Algorithme un peu plus spécifique qui cible uniquement un type précis d'édition puisque des candidats auront déjà été répertoriés par CoreTracker.

Comme vu avec l'exemple précédent chez les angiospermes, les sites édités feront partie de cette liste de candidats potentiels. Cette stratégie présente l'avantage de pouvoir restreindre les tests à une liste réduite de types d'édition. Dans ce cas, et sous les deux hypothèses cruciales suivantes, nous pouvons formuler un nouveau problème, qui peut être résolu avec les algorithmes d'optimisation.

1. L'édition a un effet correcteur sur les mutations génomiques potentiellement létales, et améliore de ce fait la similarité entre séquences orthologues ;
2. Les sites édités sont indépendants, aléatoires, et idéalement, uniformément répartis au sein des gènes ;

Problème (FindEditPos)**.** *: Étant donné un alignement de m gènes orthologues comprenant n positions, trouver une matrice binaire Q, de taille* $(m,n,3)$

$$Q_{i,j,k} = \begin{cases} 1, & \textit{si le nucléotide } k \textit{ du codon } j \textit{ de la séquence } i \textit{ est édité} \\ 0 & \textit{sinon.} \end{cases}$$

qui minimise soit la distance entre séquences, soit l'entropie de l'alignement, avec comme contraintes $\sum_k Q_{i,j,k} < 3$ (au plus deux sites édités par codon) et une pénalisation supplémentaire sur le nombre total de positions éditées pour assurer que Q soit creuse.

8.2. Limites et perspectives des méthodes intégratives d'inférence d'arbres de gènes.

Dans la seconde partie de cette thèse, nous avons décrit des méthodes intégratives efficaces de correction et de construction d'arbres de gènes, qui exploitent l'information de l'arbre des espèces pour améliorer la topologie des arbres de gènes. La critique la plus évidente de nos méthodes porte sur l'utilisation de la parcimonie pour la réconciliation, à la place des méthodes probabilistes basées sur des modèles évolutifs, certes complexes, mais explicites. Comme nous le notons dans le chapitre 3, le problème principal avec la parcimonie provient des hypothèses simplistes qu'elle fait, et de la non-prise en compte des autres histoires alternatives. Une inquiétude supplémentaire réside dans le choix objectif des coûts à donner aux différents évènements, et peut être sujet à controverse. Ceci étant dit, la parcimonie a été maintes fois prouvée compétitive par rapport aux méthodes probabilistes, tout en étant associé à un coût de calcul beaucoup plus faible. Elle présente également l'avantage d'être extrêmement flexible en ce qui concerne la nature des données en entrée (arbres enracinés ou non, datés ou non) et est souvent facilement extensible pour tenir compte de nouvelles sources d'information.

Ainsi, bien que les méthodes que nous présentons utilisent exclusivement la réconciliation pour corriger les arbres de gènes, elles peuvent, en théorie, être modifiées pour considérer d'autres contraintes évolutives sur les familles de gènes. Par exemple, l'ordre des gènes sur les segments génomiques peut fournir des informations précises sur les relations d'orthologie et de paralogie entre gènes. Il en est de même pour la préservation des fonctions au sein des familles de gènes, sous l'hypothèse que la conjecture des orthologues est vraie. Ces informations pourraient donc être exploitées lors de la reconstruction des arbres de gènes. Le défi principal provient de la manière de les combiner efficacement avec l'information des séquences et celles de la réconciliation, à l'intérieur d'un cadre d'étude unique et robuste.

Nos résultats obtenus avec GATC suggèrent qu'une telle combinaison est effectivement possible, en définissant de nouvelles contraintes, chacune associée à un objectif à optimiser. À titre d'illustration, considérons le problème de la construction d'un arbre de gènes G satisfaisant un ensemble de relations d'homologie $\mathbb{R}$ (que l'on suppose déterminées *a priori*

à partir de certaines évidences biologiques comme l'ordre des gènes), énoncé dans [323]. Nous pouvons définir un score simple qui mesure la capacité de G à satisfaire $\mathbb{R}$ et l'ajouter comme troisième objectif d'optimisation dans GATC. Nous pensons qu'une telle combinaison pourrait potentiellement être bénéfique à la reconstruction des phylogénies de gènes. Bien entendu une évaluation rigoureuse est requise, sans oublier que des conflits entre sources d'informations pourraient exister et que GATC demeure une heuristique.

Un autre défi important pour les méthodes d'inférence d'arbre de gènes concerne l'atteinte d'un équilibre entre, d'une part leur vitesse et d'autre part leur efficacité et leur puissance, c.-à-d. leur capacité à capturer une multitude d'évènements. Par exemple, notre algorithme ProfileNJ est restreint aux familles de gènes où les HGTs sont absents, mais bénéficie en retour d'une vitesse et d'une précision élevée.

D'un point de vue purement algorithmique, des progrès sont donc encore nécessaires dans le domaine et de nouvelles méthodes plus performantes et en mesure de considérer une variété d'évènements (duplications, pertes, transferts, recombinaison, ILS) sont requises.

Une extension simple de nos travaux sur l'inférence d'arbres de gènes serait de tenir compte des incertitudes au sein de l'arbre des espèces S. En effet, nos méthodes supposent que ce dernier est connu et est sans erreurs. Nous montrons bien que ProfileNJ et GATC sont tous les deux robustes aux erreurs topologiques dans S lorsqu'elles sont minimes, mais la considération explicite de ces erreurs au sein des méthodes est souhaitable. À notre connaissance, les seules approches existantes qui sont reliées au problème sont la coestimation de l'arbre des espèces et des arbres de gènes, et la résolution simultanée de noeuds non binaires au sein des deux arbres (espèces et gènes) [296]. Notons toutefois qu'une stratégie palliative couramment utilisée par les méthodes actuelles, y compris les nôtres, consiste à définir des poids pour la contribution des deux sources d'informations (alignements de séquences et réconciliation), ce qui permet de contrôler, quoique faiblement, l'effet des erreurs dans S sur la topologie des arbres de gènes corrigés.

Un travail connexe qui constituerait une excellente extension aux travaux de cette thèse serait de descendre d'un échelon évolutif pour considérer l'évolution des domaines protéiques. En effet, les algorithmes que nous décrivons ne ciblent que les évènements macro-évolutifs à l'échelle des gènes. Cependant, la duplication, la perte et le transfert de domaines sont des phénomènes bien connus qui modifient l'architecture, la fonction et la régulation des gènes et façonnent ainsi leurs évolutions. Il serait donc intéressant d'étudier cet aspect de l'évolution, qui fait d'ailleurs déjà l'objet de quelques études récentes [488, 489].

Finalement, une question importante en phylogénie moléculaire concerne l'évaluation des méthodes d'inférence. Puisque nous n'avons pas un accès direct au passé, il est actuellement

difficile de mesurer la précision des méthodes de reconstruction et donc de les comparer. Une stratégie fréquemment utilisée est d'évaluer la performance des méthodes sur des données simulées, sous des modèles évolutifs bien définis, pour lesquelles nous connaissons la "vraie histoire évolutive". Néanmoins, l'efficacité sur les données simulées n'est pas forcément transférable sur les données biologiques réelles. Comme alternative, l'utilisation de données de référence, constituées d'arbres de gènes d'excellente qualité construits sur des familles bien étudiées, est souvent préférée. Malheureusement, alors que des banques de données centralisées de référence existent pour plusieurs domaines de la génomique comparative, ils sont très rares pour les arbres de gènes. La seule initiative d'envergure pour fournir des arbres de références est le projet SwistTree [469]. Mais là encore, après sept ans, leur ensemble de référence ne contient qu'une vingtaine d'arbres, dont une majorité sont non binaires, et ne couvre pas du tout l'étendue de la complexité des modes d'évolution des familles de gènes. La construction d'arbres de référence est en général semi-automatisée, en se basant sur les inférences de méthodes fiables, suivie d'une vérification et correction manuelle à partir de connaissances biologiques pertinentes. Dans le chapitre 7, nous avons affirmé que GATC était approprié pour aider à la construction de tels arbres. En effet, l'ensemble *pareto*, d'arbres statistiquement et similairement plausibles retourné par GATC fournit une liste de solutions potentielles parmi lesquelles on pourrait faire un choix en fonction d'autres critères. La sélection de familles représentatives de gènes reste cependant une tâche centrale nécessitant la coopération de l'ensemble de la communauté.

Références

[1] M. S. Lee and A. Palci. Morphological phylogenetics in the genomic age. *Current Biology*, 25(19), p. R922–R929, 2015.

[2] E. Zuckerkandl and L. Pauling. Evolutionary divergence and convergence in proteins. In *Evolving genes and proteins*, p. 97–166. Elsevier, 1965.

[3] E. Zuckerkandl and L. Pauling. Molecules as documents of evolutionary history. *Journal of theoretical biology*, 8(2), p. 357–366, 1965.

[4] M. Dayhoff, R. Schwartz, and B. Orcutt. A model of evolutionary change in proteins. *In Atlas of Protein Sequences and Structure*, 5, p. 345–352, 1978.

[5] M. Dayhoff. The origin and evolution of protein superfamilies. *Fed. Proc.*, 35(10), p. 2132–2138, 1976.

[6] K. Fučíková and L. Lewis. Intersection of Chlorella, Muriella and Bracteacoccus : Resurrecting the genus Chromochloris Kol et Chodat (Chlorophyceae, Chlorophyta). *Fottea*, 12(1), p. 83–93, 2012.

[7] S. Ohno. *Evolution by Gene Duplication*. Springer Berlin Heidelberg, 1970.

[8] M. Goodman, J. Czelusniak, G. Moore, A. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Zoology*, 28, p. 132–163, 1979.

[9] R. L. Tatusov, E. V. Koonin, and D. J. Lipman. A genomic perspective on protein families. *Science*, 278(5338), p. 631–637, 1997.

[10] A. M. Altenhoff, R. A. Studer, M. Robinson-Rechavi, and C. Dessimoz. Resolving the ortholog conjecture : orthologs tend to be weakly, but significantly, more similar in function than paralogs. *PLoS computational biology*, 8(5), p. e1002514, 2012.

[11] M. Hahn. Bias in phylogenetic tree reconciliation methods : implications for vertebrate genome evolution. *Genome Biology*, 8(R141), 2007.

[12] W. R. Gilks, B. Audit, D. De Angelis, S. Tsoka, and C. A. Ouzounis. Modeling the percolation of annotation errors in a database of protein sequences. *Bioinformatics*, 18(12), p. 1641–1649, 2002.

[13] A. M. Schnoes, S. D. Brown, I. Dodevski, and P. C. Babbitt. Annotation error in public databases : misannotation of molecular function in enzyme superfamilies. *PLoS computational biology*, 5(12), p. e1000605, 2009.

[14] D. Frishman. Protein annotation at genomic scale : the current status. *Chemical reviews*, 107(8), p. 3448–3466, 2007.

[15] M. J. Bell. *Provenance, propagation and quality of biological annotation*. PhD thesis, Newcastle University, 2014.

[16] J. Felsenstein. Cases in which parsimony or compatibility methods will be positively misleading. *Systematic zoology*, 27(4), p. 401–410, 1978.

[17] F. Delsuc, H. Brinkmann, and H. Philippe. Phylogenomics and the reconstruction of the tree of life. *Nature Reviews Genetics*, 6, p. 361–375, 2005.

[18] R. M. Kliman, P. Andolfatto, J. A. Coyne, F. Depaulis, M. Kreitman, A. J. Berry, J. McCarter, J. Wakeley, and J. Hey. The population genetics of the origin and divergence of the Drosophila simulans complex species. *Genetics*, 156(4), p. 1913–1931, 2000.

[19] A. Suh. The phylogenomic forest of bird trees contains a hard polytomy at the root of Neoaves. *Zoologica Scripta*, 45(S1), p. 50–62, 2016.

[20] W. Maddison. Reconstructing character evolution on polytomous cladograms. *Cladistics*, 5(4), p. 365–377, 1989.

[21] J. P. Townsend, Z. Su, and Y. I. Tekle. Phylogenetic signal and noise : predicting the power of a data set to resolve phylogeny. *Systematic biology*, 61(5), p. 835–849, 2012.

[22] T. Garland Jr and R. Diaz-Uriarte. Polytomies and phylogenetically independent contrasts : examination of the bounded degrees of freedom approach. *Systematic Biology*, 48(3), p. 547–558, 1999.

[23] A. Elzanowski and J. Ostell. The Genetic Code, 2018.

[24] M. Kollmar and S. Mühlhausen. How tRNAs dictate nuclear codon reassignments : Only a few can capture non-cognate codons. *RNA Biology*, 14(3), p. 293–299, 2017.

[25] B. Boussau, G. Szöllősi, L. Duret, M. Gouy, E. Tannier., and V. Daubin. Genome-scale coestimation of species and gene trees. *Genome Research*, 23, p. 323-330, 2013.

[26] G. J. Szöllősi, W. Rosikiewicz, B. Boussau, E. Tannier, and V. Daubin. Efficient exploration of the space of reconciled gene trees. *Systematic biology*, 62(6), p. 901–912, 2013.

[27] P. Thomas. GIGA : a simple, efficient algorithm for gene tree inference in the genomic age. *BMC Bioinformatics*, 11, p. 312, 2010.

[28] F. Schreiber, M. Patricio, M. Muffato, M. Pignatelli, and A. Bateman. TreeFam v9 : a new website, more species and orthology-on-the-fly. *Nucleic acids research*, 42(D1), p. D922–D925, 2013.

[29] M. Lafond, E. Noutahi, and N. El-Mabrouk. Efficient Non-binary Gene Tree Resolution with Weighted Reconciliation Cost. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 54 of *27th Annual Symposium on Combinatorial Pattern Matching*, p. 14 :1-14 :12. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[30] E. Mayr. *Systematics and the Origin of Species*. Columbia Univ. Press, New York, 1942.

[31] W. B. Provine. Ernst Mayr : genetics and speciation. *Genetics*, 167(3), p. 1041–1046, 2004.

[32] J. Hey, W. M. Fitch, and F. J. Ayala. Systematics and the origin of species : An introduction. *Proceedings of the National Academy of Sciences*, 102(suppl 1), p. 6515–6519, 2005.

[33] A. Rich and U. RajBhandary. Transfer RNA : molecular structure, sequence, and properties. *Annual review of biochemistry*, 45(1), p. 805–860, 1976.

[34] E.-H. Im and S. S. Choi. Synonymous Codon Usage Controls Various Molecular Aspects. *Genomics & informatics*, 15(4), p. 123, 2017.

[35] J. B. Plotkin and G. Kudla. Synonymous but not the same : the causes and consequences of codon bias. *Nature Reviews Genetics*, 12(1), p. 32, 2011.

[36] T. E. Quax, N. J. Claassens, D. Söll, and J. van der Oost. Codon bias as a means to fine-tune gene expression. *Molecular cell*, 59(2), p. 149–161, 2015.

[37] Y. Lavner and D. Kotlar. Codon bias as a factor in regulating expression via translation rate in the human genome. *Gene*, 345(1), p. 127–138, 2005.

[38] S. Magadum, U. Banerjee, P. Murugan, D. Gangapur, and R. Ravikesavan. Gene duplication as a major force in evolution. *Journal of genetics*, 92(1), p. 155–161, 2013.

[39] S. Ohno, U. Wolf, and N. B. Atkin. Evolution from fish to mammals by gene duplication. *Hereditas*, 59(1), p. 169–187, 1968.

[40] J. Zhang. Evolution by gene duplication : an update. *Trends in ecology & evolution*, 18(6), p. 292–298, 2003.

[41] P. J. Keeling and J. D. Palmer. Horizontal gene transfer in eukaryotic evolution. *Nature Reviews Genetics*, 9(8), p. 605, 2008.

[42] C. Ku and W. F. Martin. A natural barrier to lateral gene transfer from prokaryotes to eukaryotes revealed from genomes : the 70% rule. *BMC biology*, 14(1), p. 89, 2016.

[43] E. G. Danchin. Lateral gene transfer in eukaryotes : tip of the iceberg or of the ice cube ? *BMC biology*, 14(1), p. 101, 2016.

[44] J. P. Demuth and M. W. Hahn. The life and death of gene families. *Bioessays*, 31(1), p. 29–39, 2009.

[45] C. Chothia and A. M. Lesk. The relation between the divergence of sequence and structure in proteins. *The EMBO journal*, 5(4), p. 823–826, 1986.

[46] H. Hegyi and M. Gerstein. The relationship between protein structure and function : a comprehensive survey with application to the yeast genome 1. *Journal of molecular biology*, 288(1), p. 147–164, 1999.

[47] D. Lee, O. Redfern, and C. Orengo. Predicting protein function from sequence and structure. *Nature Reviews Molecular Cell Biology*, 8(12), p. 995, 2007.

[48] M. Sadowski and D. Jones. The sequence–structure relationship and protein function prediction. *Current Opinion in Structural Biology*, 19(3), p. 357–362, 2009.

[49] A. Roesner, C. Fuchs, T. Hankeln, and T. Burmester. A globin gene of ancient evolutionary origin in lower vertebrates : evidence for two distinct globin families in animals. *Molecular Biology and Evolution*, 22(1), p. 12–20, 2004.

[50] T. Hankeln and T. Burmester. Neuroglobin and cytoglobin. In *The smallest biomolecules : diatomics and their interactions with heme proteins*, p. 203–218. Elsevier, 2008.

[51] R. C. Hardison. Evolution of hemoglobin and its genes. *Cold Spring Harbor perspectives in medicine*, 2(12), p. a011627, 2012.

[52] W. M. Fitch. Homology. A personal view on some of the problems. *TIG*, 16(5), p. 227-231, 2000.

[53] C. A. Darby, M. Stolzer, P. J. Ropp, D. Barker, and D. Durand. Xenolog classification. *Bioinformatics*, 33(5), p. 640–649, 2016.

[54] M. E. Peterson, F. Chen, J. G. Saven, D. S. Roos, P. C. Babbitt, and A. Sali. Evolutionary constraints on structural similarity in orthologs and paralogs. *Protein Science*, 18(6), p. 1306–1315, 2009.

[55] X. Chen and J. Zhang. The ortholog conjecture is untestable by the current gene ontology but is supported by RNA sequencing data. *PLoS computational biology*, 8(11), p. e1002784, 2012.

[56] N. L. Nehrt, W. T. Clark, P. Radivojac, and M. W. Hahn. Testing the ortholog conjecture with comparative functional genomic data from mammals. *PLoS computational biology*, 7(6), p. e1002073, 2011.

[57] C. W. Dunn, F. Zapata, C. Munro, S. Siebert, and A. Hejnol. Pairwise comparisons across species are problematic when analyzing functional genomic data. *Proceedings of the National Academy of Sciences*, p. 201707515, 2018.

[58] N. Kryuchkova-Mostacci and M. Robinson-Rechavi. Tissue-specificity of gene expression diverges slowly between orthologs, and rapidly between paralogs. *PLoS computational biology*, 12(12), p. e1005274, 2016.

[59] R. Díaz, C. Vargas-Lagunas, M. A. Villalobos, H. Peralta, Y. Mora, S. Encarnación, L. Girard, and J. Mora. argC Orthologs from Rhizobiales show diverse profiles of transcriptional efficiency and functionality in Sinorhizobium meliloti. *Journal of bacteriology*, 193(2), p. 460–472, 2011.

[60] M. V. Omelchenko, M. Y. Galperin, Y. I. Wolf, and E. V. Koonin. Non-homologous isofunctional enzymes : a systematic analysis of alternative solutions in enzyme evolution. *Biology direct*, 5(1), p. 31, 2010.

[61] A. Henschel, W. K. Kim, and M. Schroeder. Equivalent binding sites reveal convergently evolved interaction motifs. *Bioinformatics*, 22(5), p. 550–555, 2005.

[62] T. Gabaldón and E. V. Koonin. Functional and evolutionary implications of gene orthology. *Nature Reviews Genetics*, 14(5), p. 360, 2013.

[63] R. Chen and S.-S. Jeong. Functional prediction : identification of protein orthologs and paralogs. *Protein Science*, 9(12), p. 2344–2353, 2000.

[64] R. W. Holley, J. Apgar, G. A. Everett, J. T. Madison, M. J. Marquisee, S. H. Merrill, J. R. Penswick, and A. Zamir. Structure of a Ribonucleic Acid. *Science*, 147 3664, p. 1462–5, 1965.

[65] G. Gamow. Possible relation between deoxyribonucleic acid and protein structures. *Nature*, 173 (4398), p. 318–318, 1954.

[66] G. Gamow, A. Rich, and M. Yčas. The problem of information transfer from the nucleic acids to proteins. In *Advances in biological and medical physics*, volume 4, p. 23–68. Elsevier, 1956.

[67] F. Crick, L. Barnett, S. Brenner, and R. J. Watts-Tobin. General nature of the genetic code for proteins. *Nature*, 192, p. 1227 EP –, 1961.

[68] M. Nirenberg. Historical review : Deciphering the genetic code–a personal account. *Trends in biochemical sciences*, 29(1), p. 46–54, 2004.

[69] F. H. Crick. The origin of the genetic code. *Journal of molecular biology*, 38(3), p. 367–379, 1968.

[70] B. Barrell, A. Bankier, and J. Drouin. A different genetic code in human mitochondria. *Nature*, 282(5735), p. 189, 1979.

[71] T. D. Fox. Five TGA 'stop'codons occur within the translated sequence of the yeast mitochondrial gene for cytochrome c oxidase subunit II. *Proceedings of the National Academy of Sciences*, 76(12), p. 6534–6538, 1979.

[72] S. Sengupta, X. Yang, and P. G. Higgs. The mechanisms of codon reassignments in mitochondrial genetic codes. *Journal of molecular evolution*, 64(6), p. 662–688, 2007.

[73] B. F. Lang, D. Lavrov, N. Beck, and S. V. Steinberg. Mitochondrial tRNA structure, identity, and evolution of the genetic code. In *Organelle genetics*, p. 431–474. Springer, 2012.

[74] R. D. Knight, S. J. Freeland, and L. F. Landweber. Rewiring the keyboard : evolvability of the genetic code. *Nature Reviews Genetics*, 2(1), p. 49–58, 2001.

[75] L. Li, M. T. Boniecki, J. D. Jaffe, B. S. Imai, P. M. Yau, Z. A. Luthey-Schulten, and S. A. Martinis. Naturally occurring aminoacyl-tRNA synthetases editing-domain mutations that cause mistranslation in Mycoplasma parasites. *Proceedings of the National Academy of Sciences*, 108(23), p. 9378–9383, 2011.

[76] Y. I. Wolf and E. V. Koonin. On the origin of the translation system and the genetic code in the RNA world by means of natural selection, exaptation, and subfunctionalization. *Biology Direct*, 2 (1), p. 14, 2007.

[77] E. V. Koonin and A. S. Novozhilov. Origin and evolution of the genetic code : the universal enigma. *IUBMB life*, 61(2), p. 99–111, 2009.

[78] V. Chechetkin. Block structure and stability of the genetic code. *Journal of theoretical biology*, 222 (2), p. 177–188, 2003.

[79] S. J. Freeland, T. Wu, and N. Keulmann. The case for an error minimizing standard genetic code. *Origins of Life and Evolution of the Biosphere*, 33(4-5), p. 457–477, 2003.

[80] D. Haig and L. D. Hurst. A quantitative measure of error minimization in the genetic code. *Journal of molecular evolution*, 33(5), p. 412–417, 1991.

[81] I. Rumer. Systematization of the codons of the genetic code. *Doklady Akademii Nauk SSSR*, 183 (1), p. 225–226, 1968.

[82] M. Di Giulio. The origin of the genetic code : theories and their relationships, a review. *Biosystems*, 80(2), p. 175–184, 2005.

[83] J. L. King and T. H. Jukes. Non-darwinian evolution. *Science*, 164(3881), p. 788–798, 1969.

[84] D. Gilis, S. Massar, N. J. Cerf, and M. Rooman. Optimality of the genetic code with respect to protein stability and amino-acid frequencies. *Genome biology*, 2(11), p. research0049–1, 2001.

[85] C. R. Woese. *The genetic code : the molecular basis for genetic expression*. Harper & Row, New York, 1967.

[86] L. E. Orgel. Evolution of the genetic apparatus. *Journal of molecular biology*, 38(3), p. 381–393, 1968.

[87] C. Guerrier-Takada, K. Gardiner, T. Marsh, N. Pace, and S. Altman. The RNA moiety of ribonuclease P is the catalytic subunit of the enzyme. *Cell*, 35(3), p. 849–857, 1983.

[88] M. J. Fedor and J. R. Williamson. The catalytic diversity of RNAs. *Nature reviews Molecular cell biology*, 6(5), p. 399, 2005.

[89] C. R. Woese. Interpreting the universal phylogenetic tree. *Proceedings of the National Academy of Sciences*, 97(15), p. 8392–8396, 2000.

[90] I. Nitta, Y. Kamada, H. Noda, T. Ueda, and K. Watanabe. Reconstitution of peptide bond formation with Escherichia coli 23S ribosomal RNA domains. *Science*, 281(5377), p. 666–669, 1998.

[91] M. Barbieri. Evolution of the genetic code : the ribosome-oriented model. *Biological Theory*, 10(4), p. 301–310, 2015.

[92] G. E. Fox. Origin and evolution of the ribosome. *Cold Spring Harbor perspectives in biology*, p. a003483, 2010.

[93] H. F. Noller et al. Evolution of ribosomes and translation from an RNA world. *Cold Spring Harbor Monograph Series*, 43, p. 287, 2006.

[94] S. Osawa. *Evolution of the genetic code*. Oxford University Press on Demand, 1995.

[95] H. J. Cleaves II. The origin of the biologically coded amino acids. *Journal of theoretical biology*, 263(4), p. 490–498, 2010.

[96] D. A. Zaia, C. T. B. Zaia, and H. De Santana. Which amino acids should be used in prebiotic chemistry studies ? *Origins of Life and Evolution of Biospheres*, 38(6), p. 469–488, 2008.

[97] E. V. Koonin and A. S. Novozhilov. Origin and evolution of the universal genetic code. *Annual review of genetics*, 51, p. 45–62, 2017.

[98] S. Pelc and M. Welton. Stereochemical relationship between coding triplets and amino-acids. *Nature*, 209(5026), p. 868–870, 1966.

[99] C. R. Woese. On the evolution of the genetic code. *Proceedings of the National Academy of Sciences*, 54(6), p. 1546–1552, 1965.

[100] T. Sonneborn. Degeneracy of the genetic code : extent, nature, and genetic implications. In *Evolving genes and proteins*, p. 377–397. Elsevier, 1965.

[101] A. S. Novozhilov, Y. I. Wolf, and E. V. Koonin. Evolution of the genetic code : partial optimization of a random code for robustness to translation error in a rugged fitness landscape. *Biology Direct*, 2(1), p. 24, 2007.

[102] S. J. Freeland and L. D. Hurst. The genetic code is one in a million. *Journal of molecular evolution*, 47(3), p. 238–248, 1998.

[103] J. T.-F. Wong. A co-evolution theory of the genetic code. *Proceedings of the National Academy of Sciences of the United States of America*, 72(5), p. 1909, 1975.

[104] S. G. Andersson and C. G. Kurland. An extreme codon preference strategy : codon reassignment. *Molecular biology and evolution*, 8(4), p. 530–544, 1991.

[105] N. N. Ivanova, P. Schwientek, H. J. Tripp, C. Rinke, A. Pati, M. Huntemann, A. Visel, T. Woyke, N. C. Kyrpides, and E. M. Rubin. Stop codon reassignments in the wild. *Science*, 344(6186), p. 909–913, 2014.

[106] S. E. Massey and J. R. Garey. A comparative genomics analysis of codon reassignments reveals a link with mitochondrial proteome size and a mechanism of genetic code change via suppressor tRNAs. *Journal of molecular evolution*, 64(4), p. 399–410, 2007.

[107] S.-i. Yokobori, T. Suzuki, and K. Watanabe. Genetic code variations in mitochondria : tRNA as a major determinant of genetic code plasticity. *Journal of molecular evolution*, 53(4-5), p. 314–326, 2001.

[108] M. Li and A. Tzagoloff. Assembly of the mitochondrial membrane system : sequences of yeast mitochondrial valine and an unusual threonine tRNA gene. *Cell*, 18(1), p. 47–53, 1979.

[109] D. Su, A. Lieberman, B. F. Lang, M. Simonović, D. Söll, and J. Ling. An unusual tRNAThr derived from tRNAHis reassigns in yeast mitochondria the CUN codons to threonine. *Nucleic acids research*, p. gkr073, 2011.

[110] J. Ling, R. Daoud, M. J. Lajoie, G. M. Church, D. Söll, and B. F. Lang. Natural reassignment of CUU and CUA sense codons to alanine in Ashbya mitochondria. *Nucleic acids research*, 42(1), p. 499–508, 2014.

[111] T. Sugita and T. Nakase. Non-universal usage of the leucine CUG codon and the molecular phylogeny of the genus Candida. *Systematic and applied microbiology*, 22 1, p. 79–86, 1999.

[112] M. A. Santos, A. C. Gomes, M. C. Santos, L. C. Carreto, and G. R. Moura. The genetic code of the fungal CTG clade. *Comptes rendus biologies*, 334(8), p. 607–611, 2011.

[113] S. Mühlhausen, P. Findeisen, U. Plessmann, H. Urlaub, and M. Kollmar. A novel nuclear genetic code alteration in yeasts and the evolution of codon reassignment in eukaryotes. *Genome research*, 26(7), p. 945–955, 2016.

[114] S. Mühlhausen and M. Kollmar. Molecular Phylogeny of Sequenced Saccharomycetes Reveals Polyphyly of the Alternative Yeast Codon Usage. *Genome Biology and Evolution*, 6(12), p. 3222–3237, 2014.

[115] R. Riley, S. Haridas, K. H. Wolfe, M. R. Lopes, C. T. Hittinger, M. Göker, A. A. Salamov, J. H. Wisecaver, T. M. Long, C. H. Calvey, A. L. Aerts, K. W. Barry, C. Choi, A. Clum, A. Y. Coughlan, S. Deshpande, A. P. Douglass, S. J. Hanson, H.-P. Klenk, K. M. LaButti, A. Lapidus, E. A. Lindquist, A. M. Lipzen, J. P. Meier-Kolthoff, R. A. Ohm, R. P. Otillar, J. L. Pangilinan, Y. Peng, A. Rokas, C. A. Rosa, C. Scheuner, A. A. Sibirny, J. C. Slot, J. B. Stielow, H. Sun, C. P. Kurtzman, M. Blackwell, I. V. Grigoriev, and T. W. Jeffries. Comparative genomics of biotechnologically important yeasts. *Proceedings of the National Academy of Sciences of the United States of America*, 113(35), p. 9882–9887, 2016.

[116] E. Noutahi, V. Calderon, M. Blanchette, B. F. Lang, and N. El-Mabrouk. CoreTracker : accurate codon reassignment prediction, applied to mitochondrial genomes. *Bioinformatics*, 33(21), p. 3331–3339, 2017.

[117] Y. Hayashi-Ishimaru, T. Ohama, Y. Kawatsu, K. Nakamura, and S. Osawa. UAG is a sense codon in several chlorophycean mitochondria. *Current genetics*, 30(1), p. 29–33, 1996.

[118] U. Kück, K. Jekosch, and P. Holzamer. DNA sequence analysis of the complete mitochondrial genome of the green alga Scenedesmus obliquus : evidence for UAG being a leucine and UCA being a non-sense codon. *Gene*, 253(1), p. 13–18, 2000.

[119] A. M. Nedelcu, R. W. Lee, C. Lemieux, M. W. Gray, and G. Burger. The complete mitochondrial DNA sequence of Scenedesmus obliquus reflects an intermediate stage in the evolution of the green algal mitochondrial genome. *Genome Research*, 10(6), p. 819–831, 2000.

[120] M. Turmel, C. Lemieux, G. Burger, B. F. Lang, C. Otis, I. Plante, and M. W. Gray. The complete mitochondrial DNA sequences of Nephroselmis olivacea and Pedinomonas minor : two radically different evolutionary patterns within green algae. *The Plant Cell*, 11(9), p. 1717–1729, 1999.

[121] M. Turmel, C. Otis, and C. Lemieux. A deviant genetic code in the reduced mitochondrial genome of the picoplanktonic green alga Pycnococcus provasolii. *Journal of molecular evolution*, 70(2), p. 203–214, 2010.

[122] J. D. Alfonzo, V. Blanc, A. M. Estévez, M. A. T. Rubio, and L. Simpson. C to U editing of the anticodon of imported mitochondrial tRNATrp allows decoding of the UGA stop codon in Leishmania tarentolae. *The EMBO journal*, 18(24), p. 7056–7062, 1999.

[123] B. Hao, W. Gong, T. K. Ferguson, C. M. James, J. A. Krzycki, and M. K. Chan. A new UAG-encoded residue in the structure of a methanogen methyltransferase. *Science*, 296(5572), p. 1462–1466, 2002.

[124] G. Srinivasan, C. M. James, and J. A. Krzycki. Pyrrolysine encoded by UAG in Archaea : charging of a UAG-decoding specialized tRNA. *Science*, 296(5572), p. 1459–1462, 2002.

[125] L. Prat, I. U. Heinemann, H. R. Aerni, J. Rinehart, P. Odonoghue, and D. Soll. Carbon source-dependent expansion of the genetic code in bacteria. *Proceedings of the National Academy of Sciences*, 109(51), p. 21070–21075, 2012.

[126] S. U. Schneider, M. B. Leible, and X.-P. Yang. Strong homology between the small subunit of ribulose-1, 5-bisphosphate carboxylase/oxygenase of two species of Acetabularia and the occurrence of unusual codon usage. *Molecular and General Genetics MGG*, 218(3), p. 445–452, 1989.

[127] S. Horowitz and M. A. Gorovsky. An unusual genetic code in nuclear genes of Tetrahymena. *Proceedings of the National Academy of Sciences*, 82(8), p. 2452–2455, 1985.

[128] E. Cocquyt, G. H. Gile, F. Leliaert, H. Verbruggen, P. J. Keeling, and O. De Clerck. Complex phylogenetic distribution of a non-canonical genetic code in green algae. *BMC evolutionary biology*, 10(1), p. 327, 2010.

[129] P. J. Keeling and B. S. Leander. Characterisation of a non-canonical genetic code in the oxymonad Streblomastix strix. *Journal of molecular biology*, 326(5), p. 1337–1349, 2003.

[130] K. Fučíková, P. O. Lewis, D. Gonzalez-Halphen, and L. A. Lewis. Gene arrangement convergence, diverse intron content, and genetic code modifications in mitochondrial genomes of Sphaeropleales (Chlorophyta). *Genome biology and evolution*, 6(8), p. 2170–2180, 2014.

[131] J. Castresana, G. Feldmaier-Fuchs, and S. Pääbo. Codon reassignment and amino acid composition in hemichordate mitochondria. *Proceedings of the National Academy of Sciences of the United States of America*, 95 7, p. 3703–7, 1998.

[132] K. Tomita, T. Ueda, and K. Watanabe. The presence of pseudouridine in the anticodon alters the genetic code : a possible mechanism for assignment of the AAA lysine codon as asparagine in echinoderm mitochondria. *Nucleic acids research*, 27 7, p. 1683–9, 1999.

[133] F. Abascal, D. Posada, R. D. Knight, and R. Zardoya. Parallel evolution of the genetic code in arthropod mitochondrial genomes. *PLoS Biol*, 4(5), p. e127, 2006.

[134] M. F. Tuite and M. A. S. Santos. Codon reassignment in Candida species : an evolutionary conundrum. *Biochimie*, 78 11-12, p. 993–9, 1996.

[135] G. R. Moura, J. A. Paredes, and M. A. Santos. Development of the genetic code : insights from a fungal codon reassignment. *FEBS letters*, 584(2), p. 334–341, 2010.

[136] T. Suzuki, T. Ueda, and K. Watanabe. The 'polysemous' codon—a codon with multiple amino acid assignment caused by dual specificity of tRNA identity. *The EMBO Journal*, 16(5), p. 1122–1134, 1997.

[137] T. Krassowski, A. Y. Coughlan, X. X. Shen, X. Zhou, J. Kominek, D. A. Opulente, R. Riley, I. V. Grigoriev, N. Maheshwari, D. C. Shields, C. P. Kurtzman, C. T. Hittinger, A. Rokas, and K. H. Wolfe. Evolutionary instability of CUG-Leu in the genetic code of budding yeasts. *Nat Commun*, 9(1), p. 1887, 2018.

[138] R. Giegé, M. Sissler, and C. Florentz. Universal rules and idiosyncratic features in tRNA identity. *Nucleic acids research*, 26(22), p. 5017–5035, 1998.

[139] R. Giegé and M. Frugier. *Translation mechanism*, chapter Transfer RNA structure and identity. Austin (TX) : Landes Bioscience, 2003.

[140] S. Osawa and T. H. Jukes. Codon reassignment (codon capture) in evolution. *Journal of Molecular Evolution*, 28(4), p. 271–278, 1989.

[141] S. Osawa, D. Collins, T. Ohama, T. H. Jukes, and K. Watanabe. Evolution of the mitochondrial genetic code III. Reassignment of CUN codons from leucine to threonine during evolution of yeast mitochondria. *Journal of molecular evolution*, 30(4), p. 322–328, 1990.

[142] S. Osawa, T. Jukes, K. Watanabe, and A. Muto. Recent evidence for evolution of the genetic code. *Microbiological Reviews*, 56(1), p. 229–264, 1992.

[143] T. Ohama, Y. Inagaki, Y. Bessho, and S. Osawa. Evolving genetic code. *Proceedings of the Japan Academy, Series B*, 84(2), p. 58–74, 2008.

[144] D. W. Schultz and M. Yarus. Transfer RNA mutation and the malleability of the genetic code. *Journal of molecular biology*, 235(5), p. 1377–1380, 1994.

[145] D. W. Schultz and M. Yarus. On malleability in the genetic code. *Journal of molecular evolution*, 42(5), p. 597–601, 1996.

[146] M. A. Santos, C. Cheesman, V. Costa, P. Moradas-Ferreira, and M. F. Tuite. Selective advantages created by codon ambiguity allowed for the evolution of an alternative genetic code in Candida spp. *Molecular microbiology*, 31(3), p. 937–947, 1999.

[147] M. A. Santos, G. Moura, S. E. Massey, and M. F. Tuite. Driving change : the evolution of alternative genetic codes. *TRENDS in Genetics*, 20(2), p. 95–102, 2004.

[148] Y. Kawaguchi, H. Honda, J. Taniguchi-Morimura, and S. Iwasaki. The codon CUG is read as serine in an asporogenic yeast Candida cylindracea. *Nature*, 341(6238), p. 164, 1989.

[149] T. Ueda, T. Suzuki, T. Tokogawa, K. Nishikawa, and K. Watanabe. Unique structure of new serine tRNAs responsible for decoding leucine codon CUG in various Candida species and their putative ancestral tRNA genes. *Biochimie*, 76(12), p. 1217–1222, 1994.

[150] A. R. Bezerra, J. Simões, W. Lee, J. Rung, T. Weil, I. G. Gut, M. Gut, M. Bayés, L. Rizzetto, D. Cavalieri, et al. Reversion of a fungal genetic code alteration links proteome instability with genomic and phenotypic diversification. *Proceedings of the National Academy of Sciences*, 110(27), p. 11079–11084, 2013.

[151] R. M. Silva, J. A. Paredes, G. R. Moura, B. Manadas, T. Lima-Costa, R. Rocha, I. Miranda, A. C. Gomes, M. J. Koerkamp, M. Perrot, et al. Critical roles for a genetic code alteration in the evolution of the genus Candida. *The EMBO journal*, 26(21), p. 4555–4565, 2007.

[152] S. G. Andersson and C. G. Kurland. Genomic evolution drives the evolution of the translation system. *Biochemistry and cell biology*, 73(11-12), p. 775–787, 1995.

[153] S. G. Andersson and C. G. Kurland. Reductive evolution of resident genomes. *Trends in microbiology*, 6(7), p. 263–268, 1998.

[154] R. D. Knight, L. F. Landweber, and M. Yarus. How mitochondria redefine the code. *Journal of molecular evolution*, 53(4-5), p. 299–313, 2001.

[155] M. Kollmar and S. Mühlhausen. Nuclear codon reassignments in the genomics era and mechanisms behind their evolution. *BioEssays*, 39(5), p. 1600221–n/a, 2017.

[156] M. Kimura. The role of compensatory neutral mutations in molecular evolution. *Journal of Genetics*, 64(1), p. 7, 1985.

[157] I. Miranda, R. Silva, and M. A. Santos. Evolution of the genetic code in yeasts. *Yeast*, 23(3), p. 203–213, 2006.

[158] J. Ling, P. O' donoghue, and D. Söll. Genetic code flexibility in microorganisms : novel mechanisms and impact on physiology. *Nature Reviews Microbiology*, 13(11), p. 707, 2015.

[159] T. Mukai, A. Hayashi, F. Iraha, A. Sato, K. Ohtake, S. Yokoyama, and K. Sakamoto. Codon reassignment in the Escherichia coli genetic code. *Nucleic acids research*, 38(22), p. 8188–8195, 2010.

[160] Y. Huang, W. K. Russell, W. Wan, P.-J. Pai, D. H. Russell, and W. Liu. A convenient method for genetic incorporation of multiple noncanonical amino acids into one protein in Escherichia coli. *Molecular BioSystems*, 6(4), p. 683–686, 2010.

[161] R. E. Young and S. Purton. Codon reassignment to facilitate genetic engineering and biocontainment in the chloroplast of Chlamydomonas reinhardtii. *Plant biotechnology journal*, 2015.

[162] R. Krishnakumar and J. Ling. Experimental challenges of sense codon reassignment : An innovative approach to genetic code expansion. *FEBS letters*, 588(3), p. 383–388, 2014.

[163] M. J. Lajoie, A. J. Rovner, D. B. Goodman, H.-R. Aerni, A. D. Haimovich, G. Kuznetsov, J. A. Mercer, H. H. Wang, P. A. Carr, J. A. Mosberg, et al. Genomically recoded organisms expand biological functions. *science*, 342(6156), p. 357–360, 2013.

[164] A. Dumas, L. Lercher, C. D. Spicer, and B. G. Davis. Designing logical codon reassignment-Expanding the chemistry in biology. *Chemical Science*, 6(1), p. 50–69, 2015.

[165] J. M. O' Sullivan, J. B. Davenport, and M. F. Tuite. Codon reassignment and the evolving genetic code : problems and pitfalls in post-genome analysis. *TRENDS in Genetics*, 17(1), p. 20–22, 2001.

[166] J.-B.-P. Lamarck. *Philosophie zoologique*. 1809.

[167] C. Darwin. *On the origin of species, 1859*. Routledge, 2004.

[168] L. Nakhleh. Evolutionary phylogenetic networks : models and issues. In *Problem solving handbook in computational biology and bioinformatics*, p. 125–158. Springer, 2010.

[169] D. H. Huson, R. Rupp, and C. Scornavacca. *Phylogenetic networks : concepts, algorithms and applications*. Cambridge University Press, 2010.

[170] D. F. Robinson and L. R. Foulds. Comparison of phylogenetic trees. *Mathematical biosciences*, 53 (1-2), p. 131–147, 1981.

[171] P. Gorecki and O. Eulenstein. A Robinson-Foulds Measure to compare unrooted trees with rooted trees. In L. B. et al, editor, *ISBRA*, volume 7292 of *LNBI*, p. 115-126, 2012.

[172] D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M. Hillis. Phylogenetic inference. *Molecular Systematics, 2nd ed. Sinauer, Sunderland, MA*, 1996.

[173] J. Felsenstein. *Inferring phylogenies*, volume 2. Sinauer associates Sunderland, MA, 2004.

[174] M. S. Waterman and T. F. Smith. On the similarity of dendrograms. *Journal of Theoretical Biology*, 73(4), p. 789–800, 1978.

[175] M. Bordewich, O. Gascuel, K. T. Huber, and V. Moulton. Consistency of topological moves based on the balanced minimum evolution principle of phylogenetic inference. *IEEE/ACM transactions on computational biology and bioinformatics*, 6(1), p. 110–117, 2009.

[176] B. Boussau and V. Daubin. Genomes as documents of evolutionary history. *Trends in ecology & evolution*, 25(4), p. 224–232, 2010.

[177] F. Lutzoni, P. Wagner, V. Reeb, and S. Zoller. Integrating ambiguously aligned regions of DNA sequences in phylogenetic analyses without violating positional homology. *Systematic Biology*, 49 (4), p. 628–651, 2000.

[178] D. Sankoff, C. Morel, and R. J. Cedergren. Evolution of 5S RNA and the non-randomness of base replacement. *Nature*, 245(147), p. 232–234, 1973.

[179] M. A. Suchard and B. D. Redelings. BAli-Phy : simultaneous Bayesian inference of alignment and phylogeny. *Bioinformatics*, 22(16), p. 2047–2048, 2006.

[180] G. Lunter, I. Miklós, A. Drummond, J. L. Jensen, and J. Hein. Bayesian coestimation of phylogeny and sequence alignment. *Bmc Bioinformatics*, 6(1), p. 83, 2005.

[181] T. Warnow. Large-scale multiple sequence alignment and phylogeny estimation. In *Models and Algorithms for Genome Evolution*, p. 85–146. Springer, London, 2013.

[182] E. V. Koonin. Orthologs, paralogs, and evolutionary genomics. *Annu. Rev. Genet.*, 39, p. 309–338, 2005.

[183] C. Camacho, G. Coulouris, V. Avagyan, N. Ma, J. Papadopoulos, K. Bealer, and T. L. Madden. BLAST+ : architecture and applications. *BMC bioinformatics*, 10(1), p. 421, 2009.

[184] R. Overbeek, M. Fonstein, M. D'souza, G. D. Pusch, and N. Maltsev. The use of gene clusters to infer functional coupling. *Proceedings of the National Academy of Sciences*, 96(6), p. 2896–2901, 1999.

[185] L. B. Koski and G. B. Golding. The closest BLAST hit is often not the nearest neighbor. *Journal of molecular evolution*, 52(6), p. 540–542, 2001.

[186] K. P. O' brien, M. Remm, and E. L. Sonnhammer. Inparanoid : a comprehensive database of eukaryotic orthologs. *Nucleic acids research*, 33(suppl_1), p. D476–D480, 2005.

[187] L. Li, C. J. Stoeckert, and D. S. Roos. OrthoMCL : identification of ortholog groups for eukaryotic genomes. *Genome research*, 13(9), p. 2178–2189, 2003.

[188] H. Mi, X. Huang, A. Muruganujan, H. Tang, C. Mills, D. Kang, and P. D. Thomas. PANTHER version 11 : expanded annotation data from Gene Ontology and Reactome pathways, and data analysis tool enhancements. *Nucleic acids research*, 45(D1), p. D183–D189, 2016.

[189] P. Flicek, M. R. Amode, D. Barrell, K. Beal, K. Billis, S. Brent, D. Carvalho-Silva, P. Clapham, G. Coates, S. Fitzgerald, L. Gil, C. G. Girón, L. Gordon, T. Hourlier, S. Hunt, N. Johnson, T. Juettemann, A. K. Kähäri, S. Keenan, E. Kulesha, F. J. Martin, T. Maurel, W. M. McLaren, D. N. Murphy, R. Nag, B. Overduin, M. Pignatelli, B. Pritchard, E. Pritchard, H. S. Riat, M. Ruffier, D. Sheppard, K. Taylor, A. Thormann, S. J. Trevanion, A. Vullo, S. P. Wilder, M. Wilson, A. Zadissa, B. L. Aken, E. Birney, F. Cunningham, J. Harrow, J. Herrero, T. J. P. Hubbard, R. Kinsella, M. Muffato, A. Parker, G. Spudich, A. Yates, D. R. Zerbino, and S. M. J. Searle. Ensembl 2014. *Nucleic Acids Research*, 42(Database issue), p. D749–D755, 2014.

[190] L. P. Pryszcz, J. Huerta-Cepas, and T. Gabaldon. MetaPhOrs : orthology and paralogy predictions from multiple phylogenetic evidence using a consistency-based confidence score. *Nucleic acids research*, 39(5), p. e32–e32, 2010.

[191] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of computational biology*, 1(4), p. 337–348, 1994.

[192] M. A. Larkin, G. Blackshields, N. Brown, R. Chenna, P. A. McGettigan, H. McWilliam, F. Valentin, I. M. Wallace, A. Wilm, R. Lopez, et al. Clustal W and Clustal X version 2.0. *bioinformatics*, 23 (21), p. 2947–2948, 2007.

[193] R. C. Edgar. MUSCLE : multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5), p. 1792–1797, 2004.

[194] K. Katoh and D. M. Standley. MAFFT multiple sequence alignment software version 7 : improvements in performance and usability. *Molecular biology and evolution*, 30(4), p. 772–780, 2013.

[195] T. Warnow. Supertree Construction : Opportunities and Challenges. *arXiv preprint arXiv :1805.03530*, 2018.

[196] A. de Queiroz and J. Gatesy. The supermatrix approach to systematics. *Trends in ecology & evolution*, 22(1), p. 34–41, 2007.

[197] O. R. Bininda-Emonds. *Phylogenetic supertrees : combining information to reveal the tree of life*, volume 4. Springer Science & Business Media, 2004.

[198] Z. Yang and B. Rannala. Molecular phylogenetics : principles and practice. *Nature Reviews Genetics*, 13(5), p. 303, 2012.

[199] M. Steel, M. Hendy, and D. Penny. Loss of information in genetic distances. *Nature*, 336(6195), p. 118, 1988.

[200] K. Tamura, J. Dudley, M. Nei, and S. Kumar. MEGA4 : molecular evolutionary genetics analysis (MEGA) software version 4.0. *Molecular biology and evolution*, 24(8), p. 1596–1599, 2007.

[201] O. Gascuel. BIONJ : an improved version of the NJ algorithm based on a simple model of sequence data. *Molecular biology and evolution*, 14(7), p. 685–695, 1997.

[202] W. M. Fitch. Toward defining the course of evolution : minimum change for a specific tree topology. *Systematic Biology*, 20(4), p. 406–416, 1971.

[203] D. Sankoff and P. Rousseau. Locating the vertices of a Steiner tree in an arbitrary metric space. *Mathematical Programming*, 9(1), p. 240–246, 1975.

[204] J. Felsenstein. PHYLIP - Phylogeny Inference Package (Version 3.2). *Cladistics*, 5, p. 164–166, 1989.

[205] D. L. Swofford. PAUP : phylogenetic analysis using parsimony. 4th ed., Sinauer Associates, 2002.

[206] P. A. Goloboff, J. S. Farris, and K. C. Nixon. TNT, a free program for phylogenetic analysis. *Cladistics*, 24(5), p. 774–786, 2008.

[207] J. L. Thorne, H. Kishino, and J. Felsenstein. An evolutionary model for maximum likelihood alignment of DNA sequences. *Journal of Molecular Evolution*, 33(2), p. 114–124, 1991.

[208] T. H. Jukes, C. R. Cantor, et al. Evolution of protein molecules. *Mammalian protein metabolism*, 3(21), p. 132, 1969.

[209] M. Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of molecular evolution*, 16(2), p. 111–120, 1980.

[210] S. Tavaré. Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on mathematics in the life sciences*, 17(2), p. 57–86, 1986.

[211] D. T. Jones, W. R. Taylor, and J. M. Thornton. The rapid generation of mutation data matrices from protein sequences. *Bioinformatics*, 8(3), p. 275–282, 1992.

[212] N. Lartillot and H. Philippe. A Bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. *Molecular Biology and Evolution*, 21(6), p. 1095–1109, 2004.

[213] J. Aldrich. RA Fisher and the making of maximum likelihood 1912-1922. *Statistical science*, p. 162–176, 1997.

[214] J. Felsenstein. Evolutionary trees from DNA sequences : a maximum likelihood approach. *Journal of molecular evolution*, 17(6), p. 368–376, 1981.

[215] J. Adachi and M. Hasegawa. MOLPHY : Programs for molecular phylogenetics based on maximum likelihood, vers. 2.3. *Institute of Statistical Mathematics, Tokyo*, 1996.

[216] S. Guindon and O. Gascuel. A simple, fast and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52, p. 696-704, 2003.

[217] A. Stamatakis. RAxML-VI-HPC : maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, 22(21), p. 2688–2690, 2006.

[218] M. N. Price, P. S. Dehal, and A. P. Arkin. FastTree 2 - approximately maximum-likelihood trees for large alignments. *PloS one*, 5(3), p. e9490, 2010.

[219] L.-T. Nguyen, H. A. Schmidt, A. von Haeseler, and B. Q. Minh. IQ-TREE : a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular biology and evolution*, 32(1), p. 268–274, 2014.

[220] B. S. Gaut and P. O. Lewis. Success of maximum likelihood phylogeny inference in the four-taxon case. *Molecular Biology and Evolution*, 12(1), p. 152–162, 1995.

[221] J. P. Huelsenbeck. Performance of phylogenetic methods in simulation. *Systematic biology*, 44(1), p. 17–48, 1995.

[222] M. K. Kuhner and J. Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular biology and evolution*, 11(3), p. 459–468, 1994.

[223] J. S. Rogers. On the consistency of maximum likelihood estimation of phylogenetic trees from nucleotide sequences. *Systematic biology*, 46(2), p. 354–357, 1997.

[224] J. T. Chang. Full reconstruction of Markov models on evolutionary trees : identifiability and consistency. *Mathematical biosciences*, 137(1), p. 51–73, 1996.

[225] S. Roch and M. Steel. Likelihood-based tree reconstruction on a concatenation of aligned sequence data sets can be statistically inconsistent. *Theoretical population biology*, 100, p. 56–62, 2015.

[226] B. Larget and D. L. Simon. Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. *Molecular biology and evolution*, 16(6), p. 750–759, 1999.

[227] J. Hulsenbeck and F. Ronquist. MrBayes : Bayesian inference of phylogeny. *Bioinformatics*, 17, p. 754–755, 2001.

[228] A. J. Drummond, M. A. Suchard, D. Xie, and A. Rambaut. Bayesian phylogenetics with BEAUti and the BEAST 1.7. *Mol. Biol. Evol.*, 2012.

[229] N. Lartillot, T. Lepage, and S. Blanquart. PhyloBayes 3 : a Bayesian software package for phylogenetic reconstruction and molecular dating. *Bioinformatics*, 25(17), p. 2286–2288, 2009.

[230] Y. Suzuki, G. V. Glazko, and M. Nei. Overcredibility of molecular phylogenies obtained by Bayesian phylogenetics. *Proceedings of the National Academy of Sciences*, 99(25), p. 16138–16143, 2002.

[231] P. Erixon, B. Svennblad, T. Britton, and B. Oxelman. Reliability of Bayesian posterior probabilities and bootstrap frequencies in phylogenetics. *Systematic biology*, 52(5), p. 665–673, 2003.

[232] H. Philippe, F. Delsuc, H. Brinkmann, and N. Lartillot. Phylogenomics. *Annu. Rev. Ecol. Evol. Syst.*, 36, p. 541–562, 2005.

[233] A. R. Lemmon, J. M. Brown, K. Stanger-Hall, and E. M. Lemmon. The effect of ambiguous data on phylogenetic estimates obtained by maximum likelihood and Bayesian inference. *Systematic Biology*, 58(1), p. 130–145, 2009.

[234] J. J. Wiens and D. S. Moen. Missing data and the accuracy of Bayesian phylogenetics. *Journal of Systematics and Evolution*, 46(3), p. 307–314, 2008.

[235] J. J. Wiens. Incomplete taxa, incomplete characters, and phylogenetic accuracy : is there a missing data problem ? *Journal of Vertebrate Paleontology*, 23(2), p. 297–310, 2003.

[236] B. Roure, D. Baurain, and H. Philippe. Impact of missing data on phylogenies inferred from empirical phylogenomic data sets. *Molecular biology and evolution*, 30(1), p. 197–214, 2012.

[237] D. L. Swofford, P. J. Waddell, J. P. Huelsenbeck, P. G. Foster, P. O. Lewis, and J. S. Rogers. Bias in phylogenetic estimation and its relevance to the choice between parsimony and likelihood methods. *Systematic biology*, 50(4), p. 525–539, 2001.

[238] B. Kolaczkowski and J. W. Thornton. Performance of maximum parsimony and likelihood phylogenetics when evolution is heterogeneous. *Nature*, 431(7011), p. 980, 2004.

[239] E. Dobakova, P. Flegontov, T. Skalickỳ, and J. Lukeš. Unexpectedly streamlined mitochondrial genome of the euglenozoan Euglena gracilis. *Genome Biology and Evolution*, 7(12), p. 3358–3367, 2015.

[240] D. Posada and K. A. Crandall. Modeltest : testing the model of DNA substitution. *Bioinformatics (Oxford, England)*, 14(9), p. 817–818, 1998.

[241] J. Felsenstein. Confidence limits on phylogenies : an approach using the bootstrap. *Evolution*, 39 (4), p. 783–791, 1985.

[242] M. Anisimova and O. Gascuel. Approximate likelihood-ratio test for branches : a fast, accurate, and powerful alternative. *Systematic biology*, 55(4), p. 539–552, 2006.

[243] H. Kishino and M. Hasegawa. Evaluation of the maximum likelihood estimate of the evolutionary tree topologies from DNA sequence data, and the branching order in Hominoidea. *Journal of molecular evolution*, 29(2), p. 170–179, 1989.

[244] H. Shimodaira and M. Hasegawa. Multiple comparisons of log-likelihoods with applications to phylogenetic inference. *Molecular biology and evolution*, 16(8), p. 1114–1114, 1999.

[245] H. Shimodaira. An approximately unbiased test of phylogenetic tree selection. *Systematic biology*, 51(3), p. 492–508, 2002.

[246] N. Goldman, J. P. Anderson, and A. G. Rodrigo. Likelihood-based tests of topologies in phylogenetics. *Systematic biology*, 49(4), p. 652–670, 2000.

[247] K. Strimmer and A. Rambaut. Inferring confidence sets of possibly misspecified gene trees. *Proceedings of the Royal Society of London B : Biological Sciences*, 269(1487), p. 137–142, 2002.

[248] H. Shimodaira and M. Hasegawa. CONSEL : for assessing the confidence of phylogenetic tree selection. *Bioinformatics*, 17(12), p. 1246–1247, 2001.

[249] L. Y. Rusin, E. Lyubetskaya, K. Y. Gorbunov, and V. Lyubetsky. Reconciliation of gene and species trees. *BioMed research international*, 2014, 2014.

[250] Ö. Akerborg, B. Sennblad, L. Arvestad, and J. Lagergren. Simultaneous Bayesian gene tree reconstruction and reconciliation analysis. *Proceedings of the National Academy of Sciences USA*, 106 (14), p. 5714–5719, 2009.

[251] J. Sjöstrand, A. Tofigh, V. Daubin, L. Arvestad, B. Sennblad, and J. Lagergren. A Bayesian method for analyzing lateral gene transfer. *Systematic biology*, 63(3), p. 409–420, 2014.

[252] G. J. Szöllősi, E. Tannier, V. Daubin, and B. Boussau. The inference of gene trees with species trees. *Systematic biology*, 64(1), p. e42–e62, 2014.

[253] J.-P. Doyon, C. Chauve, and S. Hamel. Space of gene/species trees reconciliations and parsimonious models. *J. Comput. Biol*, 16(10), p. 1399-1418, 2009.

[254] J.-P. Doyon, S. Hamel, and C. Chauve. An efficient method for exploring the space of gene tree/species tree reconciliations in a probabilistic framework. *IEEE/ACM transactions on computational biology and bioinformatics*, 9(1), p. 26–39, 2012.

[255] R. D. Page. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Systematic Biology*, 43(1), p. 58–77, 1994.

[256] L. Zhang. On Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *Journal of Computational Biology*, 4, p. 177–188., 1997.

[257] C. M. Zmasek and S. R. Eddy. A simple algorithm to infer gene duplication and speciiation events on a gene tree. *Bioinformatics*, 17, p. 821–828, 2001.

[258] C. Chauve and N. El-Mabrouk. New perspectives on gene family evolution : losses in reconciliation and a link with supertrees. In *RECOMB 2009*, volume 5541 of *LNCS*, p. 46-58. Springer, 2009.

[259] A. Tofigh, M. Hallett, and J. Lagergren. Simultaneous identification of duplications and lateral gene transfers. *IEEE/ACM Trans.Comput.BiolBioinform.*, 8(2), p. 517-535, 2011.

[260] J.-P. Doyon, C. Scornavacca, K. Y. Gorbunov, G. J. Szöllősi, V. Ranwez, and V. Berry. An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers. In *RECOMB International Workshop on Comparative Genomics*, p. 93–108. Springer, 2010.

[261] J.-P. Doyon, V. Ranwez, V. Daubin, and V. Berry. Models, algorithms and programs for phylogeny reconciliation. *Briefings in bioinformatics*, 12(5), p. 392–400, 2011.

[262] M. T. Hallett and J. Lagergren. Efficient algorithms for lateral gene transfer problems. In *Proceedings of the fifth annual international conference on Computational biology*, p. 149–156. ACM, 2001.

[263] Y. Ovadia, D. Fielder, C. Conow, and R. Libeskind-Hadas. The cophylogeny reconstruction problem is NP-complete. *J. Comput. Biol.*, 18(1), p. 59-65, 2011.

[264] M. Bansal, J. Eric, and M. Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28(12), p. i283-i291, 2012.

[265] L. David and E. Alm. Rapid evolutionary innovation during an Archaean genetic expansion. *Nature*, 469, 2011.

[266] F. Rutschmann. Molecular dating of phylogenetic trees : a brief review of current methods that estimate divergence times. *Diversity and Distributions*, 12(1), p. 35–48, 2006.

[267] W. Maddison. Gene trees in species trees. *Syst. Biol.*, 46(3), p. 523–536, 1997.

[268] T. Wu and L. Zhang. Structural properties of the reconciliation space and their applications in enumerating nearly-optimal reconciliations between a gene tree and a species tree. In *BMC bioinformatics*, volume 12, p. S7. BioMed Central, 2011.

[269] D. Bork, R. Cheng, J. Wang, J. Sung, and R. Libeskind-Hadas. On the computational complexity of the maximum parsimony reconciliation problem in the duplication-loss-coalescence model. *Algorithms for Molecular Biology*, 12(1), p. 6, 2017.

[270] M. D. Rasmussen and M. Kellis. Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Research*, 22(4), p. 755–765, 2012.

[271] Y. Wu, M. Rasmussen, M. Bansal, and M. Kellis. Most parsimonious reconciliation in the presence of gene duplication, loss, and deep coalescence using labeled coalescent trees. *Genome Research*, 24, p. 475-486, 2014.

[272] M. Stolzer, H. Lai, M. Xu, D. Sathaye, B. Vernot, and D. Durand. Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. *Bioinformatics*, 28(18), p. i409-i415, 2012.

[273] B. Vernot, M. Stolzer, A. Goldman, and D. Durand. Reconciliation with non-binary species trees. *J Comput Biol*, 15, p. 981-1006, 2009.

[274] Y.-b. Chan, V. Ranwez, and C. Scornavacca. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *Journal of theoretical biology*, 432, p. 1–13, 2017.

[275] D. Hasic and E. Tannier. Gene tree species tree reconciliation with gene conversion. *arXiv preprint arXiv :1703.08950*, 2017.

[276] K. Liu and T. Warnow. Large-scale multiple sequence alignment and tree estimation using SATé. *Methods Mol. Biol.*, 1079, p. 219–244, 2014.

[277] J. Heled and A. J. Drummond. Bayesian inference of species trees from multilocus data. *Mol. Biol. Evol.*, 27(3), p. 570–580, 2010.

[278] C. C. Challis. *Bayesian Structural Phylogenetics*. PhD thesis, Duke University, 2013.

[279] D. A. Liberles, S. A. Teichmann, I. Bahar, U. Bastolla, J. Bloom, E. Bornberg-Bauer, L. J. Colwell, A. P. de Koning, N. V. Dokholyan, J. Echave, A. Elofsson, D. L. Gerloff, R. A. Goldstein, J. A. Grahnen, M. T. Holder, C. Lakner, N. Lartillot, S. C. Lovell, G. Naylor, T. Perica, D. D. Pollock, T. Pupko, L. Regan, A. Roger, N. Rubinstein, E. Shakhnovich, K. Sjolander, S. Sunyaev, A. I. Teufel, J. L. Thorne, J. W. Thornton, D. M. Weinreich, and S. Whelan. The interface of protein structure, protein biophysics, and molecular evolution. *Protein Sci.*, 21(6), p. 769–785, 2012.

[280] C. O. Wilke. Bringing molecules back into molecular evolution. *PLoS computational biology*, 8(6), p. e1002572, 2012.

[281] S. Bérard, C. Gallien, B. Boussau, G. J. Szöllősi, V. Daubin, and E. Tannier. Evolution of gene neighborhoods within reconciled phylogenies. *Bioinformatics*, 28(18), p. i382–i388, 2012.

[282] M. D. Rasmussen and M. Kellis. A Bayesian approach for fast and accurate gene tree reconstruction. *Molecular Biology and Evolution*, 28(1), p. 273–290, 2010.

[283] E. Noutahi and N. El-Mabrouk. GATC : a genetic algorithm for gene tree construction under the Duplication-Transfer-Loss model of evolution. *BMC genomics*, 19(2), p. 102, 2018.

[284] Y. Wu, M. Rasmussen, M. Bansal, and M. Kellis. TreeFix : Statistically informed gene tree error correction using species trees. *Systematic Biology*, 62(1), p. 110-120, 2013.

[285] C. Scornavacca, E. Jacox, and G. Szollosi. Joint amalgamation of most parsimonious reconciled gene trees. *Bioinformatics*, 31(6), p. 841-848, 2015.

[286] K. Chen, D. Durand, and M. Farach-Colton. Notung : Dating Gene Duplications using Gene Family Trees. *J.Comp.Biol.*, 7, p. 429–447, 2000.

[287] P. Gorecki and O. Eulenstein. Algorithms : simultaneous error-correction and rooting for gene tree reconciliation and the gene duplication problem. *BMC Bioinformatics*, 13(Supp 10), p. S14, 2011.

[288] P. Gorecki and O. Eulenstein. A linear-time algorithm for error-corrected reconciliation of unrooted gene trees. In *ISBRA*, volume 6674 of *LNBI*, p. 148-159. Springer-Verlag, 2011.

[289] T. H. Nguyen, V. Ranwez, S. Pointet, A.-M. A. Chifolleau, J.-P. Doyon, and V. Berry. Reconciliation and local gene tree rearrangement can be of mutual profit. *Algorithms for Molecular Biology*, 8(1), p. 12, 2013.

[290] M. Bansal, Y. Wu, E. Alm, and M. Kellis. Improved Gene Tree Error-Correction in the Presence of Horizontal Gene Transfer. *Bioinformatics*, 31(8), p. 1211-1218, 2015.

[291] D. Durand, B. V. Halldórsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *Journal of Computational Biology*, 13(2), p. 320–335, 2006.

[292] W. Chang and O. Eulenstein. Reconciling gene trees with apparent polytomies. In D. Chen and D. T. Lee, editors, *Proceedings of the 12th Conference on Computing and Combinatorics (COCOON)*, volume 4112 of *Lecture Notes in Computer Science*, p. 235–244, 2006.

[293] M. Lafond, K. Swenson, and N. El-Mabrouk. An Optimal Reconciliation Algorithm for Gene Trees with Polytomies. In *LNCS*, volume 7534 of *WABI*, p. 106-122, 2012.

[294] Y. Zheng and L. Zhang. Reconciliation with Non-binary Gene Trees Revisited. In *Lecture Notes in Computer Science*, volume 8394, p. 418-432, 2014. Proceedings of RECOMB.

[295] E. Jacox, C. Chauve, G. Szöllősi, Y. Ponty, and C. Scornavacca. ecceTERA : comprehensive gene tree-species tree reconciliation using parsimony. *Bioinformatics*, 32(13), p. 2056-2058, 2016.

[296] Y. Zheng, T. Wu, and L. Zhang. Reconciliation of gene and species trees With Polytomies. arXiv :1201.3995, 2012.

[297] P. Gorecki, O. Eulenstein, and J. Tiuryn. Unrooted tree reconciliation : A unified approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(2), p. 522-536, 2013.

[298] P. Górecki and O. Eulenstein. Algorithms for Unrooted Gene Trees with Polytomies ? 2013.

[299] M. Kordi and M. Bansal. On the complexity of Duplication-Transfer-Loss reconciliation with non-binary gene trees. *IEEE/ACM Transactions on Computational Biology and Bioin- formatics*, 2016.

[300] M. Kordi and M. Bansal. Exact Algorithms for Duplication-Transfer-Loss Reconciliation with Non-Binary Gene Trees. *IEEE/ACM Trans Comput Biol Bioinform.*, 2017.

[301] H. Lai, M. Stolzer, and D. Durand. Fast Heuristics for Resolving Weakly Supported Branches Using Duplication, Transfers, and Losses. In *RECOMB-CG*, p. 22 pages, 2017.

[302] E. Jacox, M. Weller, E. Tannier, and C. Scornavacca. Resolution and reconciliation of non-binary gene trees with transfers, duplications and losses. *Bioinformatics*, 33(7), p. 980–987, 2017.

[303] R. Cheng, M. Dohlen, C. Pekker, G. Quiroz, J. Wang, R. Libeskind-Hadas, and Y.-C. Wu. Reconciliation Feasibility of Non-binary Gene Trees Under a Duplication-Loss-Coalescence Model. In *International Conference on Algorithms for Computational Biology*, p. 11–23. Springer, 2018.

[304] H. Li. *Constructing the Treefam Database*. PhD thesis, The Institute of Theoretical Physics, Chinese Academic of Science, 2006.

[305] G. J. Szöllősi, E. Tannier, N. Lartillot, and V. Daubin. Lateral gene transfer from the dead. *Systematic biology*, 62(3), p. 386–397, 2013.

[306] S. Höhna and A. J. Drummond. Guided tree topology proposals for Bayesian phylogenetic inference. *Systematic biology*, 61(1), p. 1–11, 2011.

[307] B. Larget. The estimation of tree posterior probabilities using conditional clade probability distributions. *Systematic biology*, 62(4), p. 501–511, 2013.

[308] C. Scornavacca, L. van Iersel, S. Kelk, and D. Bryant. The agreement problem for unrooted phylogenetic trees is FPT. *J. Graph Algorithms Appl.*, 18(3), p. 385 -392, 2014.

[309] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *J. Classif.*, 9, p. 91 -116, 1992.

[310] A. Aho, S. Yehoshua, T. Szymanski, and J. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM J. Comput.*, 10(3), p. 405-421, 1981.

[311] M. Constantinescu and D. Sankoff. An efficient algorithm for supertrees. *J. Classif.*, 12, p. 101-112, 1995.

[312] M. Ng and N. Wormald. Reconstruction of rooted trees from subtrees. *Discrete Appl. Math*, 69, p. 19-31, 1996.

[313] C. Semple. Reconstructing minimal rooted trees. *Discrete Appl. Math.*, 127(3), 2003.

[314] N. Nguyen, S. Mirarab, and T. Warnow. MRL and SuperFine+MRL : new supertree methods. *Alg. Mol. Biol.*, 7(3), 2012.

[315] M. Lafond, A. Ouangraoua, and N. El-Mabrouk. Reconstructing a SuperGeneTree minimizing reconciliation. *BMC Genomics*, 16, p. S4, 2015. Special issue of RECOMB-CG 2015.

[316] M. Lafond, C. Chauve, N. El-Mabrouk, and A. Ouangraoua. Gene Tree Construction and Correction using SuperTree and Reconciliation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, PP(99), p. 12 pages, 2018.

[317] K. M. Swenson and N. El-Mabrouk. Gene Trees and Species Trees : Irreconcilable Differences. *BMC Bioinformatics*, 13((Suppl 19)), p. S15, 2012.

[318] C. Whidden, N. Zeh, and R. G. Beiko. Supertrees based on the subtree prune-and-regraft distance. *Systematic biology*, 63(4), p. 566–581, 2014.

[319] A. Vilella, J. Severin, A. Ureta-Vidal, L. Heng, R. Durbin, and E. Birney. EnsemblCompara gene trees : Complete, duplication-aware phylogenetic trees in vertebrates. *Genome Research*, 19, p. 327-335, 2009.

[320] J. Sjöstrand. *Reconciling gene family evolution and species evolution.* PhD thesis, Numerical Analysis and Computer Science (NADA), Stockholm University, 2013.

[321] A. Doroftei and N. El-Mabrouk. Removing Noise from Gene Trees. In *WABI*, volume 6833 of *LNBI/LNBI*, p. 76-91, 2011.

[322] R. Dondi and N. El-Mabrouk. Minimum Leaf Removal for Reconciliation : Complexity and Algorithms. In *CPM*, volume 7354 of *Lecture Notes in Computer Science*, p. 399-412. Springer, 2012.

[323] M. Lafond, R. Dondi, and N. El-Mabrouk. The link between orthology relations and gene trees : a correction perspective. *Algo. Mol. Biol.*, 11(1), p. 1, 2016.

[324] N. El-Mabrouk and A. Ouangraoua. A general framework for gene tree correction based on duplication-loss reconciliation. In *LIPIcs*, volume 88 of *Workshop on Algorithms in Bioinformatics (WABI)*, p. 8 :1-8 :14, 2017.

[325] F. Abascal, R. Zardoya, and D. Posada. GenDecoder : genetic code prediction for metazoan mitochondria. *Nucleic acids research*, 34(suppl 2), p. W389–W393, 2006.

[326] B. E. Dutilh, R. Jurgelenaite, R. Szklarczyk, S. A. van Hijum, H. R. Harhangi, M. Schmid, B. de Wild, H. G. Stunnenberg, M. Strous, M. S. Jetten, et al. FACIL : fast and accurate genetic code inference and logo. *Bioinformatics*, 27(14), p. 1929–1933, 2011.

[327] S. Mühlhausen and M. Kollmar. Predicting the fungal CUG codon translation with Bagheera. *BMC genomics*, 15(1), p. 411, 2014.

[328] L. Breiman. Random forests. *Machine learning*, 45(1), p. 5–32, 2001.

[329] M. W. Nirenberg, O. Jones, P. Leder, B. Clark, W. Sly, and S. Pestka. On the coding of genetic information. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 28, p. 549–557. Cold Spring Harbor Laboratory Press, Cold Spring Harbor Laboratory Press, 1963.

[330] D. Söll, E. Ohtsuka, D. Jones, R. Lohrmann, H. Hayatsu, S. Nishimura, and H. Khorana. Studies on polynucleotides, XLIX. Stimulation of the binding of aminoacyl-sRNA's to ribosomes by ribotrinucleotides and a survey of codon assignments for 20 amino acids. *Proceedings of the National Academy of Sciences*, 54(5), p. 1378–1385, 1965.

[331] K. Watanabe and S.-i. Yokobori. tRNA modification and genetic code variations in animal mitochondria. *Journal of nucleic acids*, 2011, 2011.

[332] P. J. Keeling. Genomics : Evolution of the Genetic Code. *Current Biology*, 26(18), p. R851–R853, 2016.

[333] S. Sengupta and P. G. Higgs. A unified model of codon reassignment in alternative genetic codes. *Genetics*, 170(2), p. 831–840, 2005.

[334] E. C. Swart, V. Serra, G. Petroni, and M. Nowacki. Genetic codes with no dedicated stop codon : context-dependent translation termination. *Cell*, 166(3), p. 691–702, 2016.

[335] S. S. Yadavalli and M. Ibba. Selection of tRNA charging quality control mechanisms that increase mistranslation of the genetic code. *Nucleic acids research*, 41(2), p. 1104–1112, 2013.

[336] J. Swire, O. P. Judson, and A. Burt. Mitochondrial genetic codes evolve to match amino acid requirements of proteins. *Journal of molecular evolution*, 60(1), p. 128–139, 2005.

[337] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[338] S. R. Eddy. Profile hidden Markov models. *Bioinformatics (Oxford, England)*, 14(9), p. 755–763, 1998.

[339] A. Adoutte, G. Balavoine, N. Lartillot, O. Lespinet, B. Prud' Homme, and R. De Rosa. The new animal phylogeny : reliability and implications. *Proceedings of the National Academy of Sciences*, 97(9), p. 4453–4456, 2000.

[340] K. M. Halanych. The new view of animal phylogeny. *Annu. Rev. Ecol. Evol. Syst.*, 35, p. 229–256, 2004.

[341] Y. Liu, D. M. Engelman, and M. Gerstein. Genomic analysis of membrane protein families : abundance and conserved motifs. *Genome biology*, 3(10), p. research0054–1, 2002.

[342] D. A. McClellan and K. G. McCracken. Estimating the influence of selection on the variable amino acid sites of the cytochrome b protein functional domains. *Molecular biology and evolution*, 18(6), p. 917–925, 2001.

[343] R. D. Rosengarten, E. A. Sperling, M. A. Moreno, S. P. Leys, and S. L. Dellaporta. The mitochondrial genome of the hexactinellid sponge Aphrocallistes vastus : evidence for programmed translational frameshifting. *BMC genomics*, 9(1), p. 33, 2008.

[344] K. M. Haen, B. F. Lang, S. A. Pomponi, and D. V. Lavrov. Glass sponges and bilaterian animals share derived mitochondrial genomic features : a common ancestry or parallel evolution ? *Molecular biology and evolution*, 24(7), p. 1518–1527, 2007.

[345] A. Jeyaprakash and M. A. Hoy. The mitochondrial genome of the predatory mite Metaseiulus occidentalis (Arthropoda : Chelicerata : Acari : Phytoseiidae) is unexpectedly large and contains several novel features. *Gene*, 391(1), p. 264–274, 2007.

[346] H. H. Rogers and S. Griffiths-Jones. tRNA anticodon shifts in eukaryotic genomes. *RNA*, 20(3), p. 269–281, 2014.

[347] F. Leliaert, D. R. Smith, H. Moreau, M. D. Herron, H. Verbruggen, C. F. Delwiche, and O. De Clerck. Phylogeny and molecular evolution of the green algae. *Critical Reviews in Plant Sciences*, 31(1), p. 1–46, 2012.

[348] L. A. Lewis and R. M. McCourt. Green algae and the origin of land plants. *American journal of botany*, 91(10), p. 1535–1556, 2004.

[349] G. Burger and A. M. Nedelcu. Mitochondrial genomes of algae. In *Genomics of Chloroplasts and Mitochondria*, p. 127–157. Springer, 2012.

[350] E. Rodríguez-Salinas, C. Remacle, and D. González-Halphen. Green Algae Genomics : A Mitochondrial Perspective. In *Advances in Botanical Research*, volume 63, p. 187–214. Elsevier, 2012.

[351] M. A. Buchheim, C. Lemieux, C. Otis, R. R. Gutell, R. L. Chapman, and M. Turmel. Phylogeny of the Chlamydomonadales (Chlorophyceae) : a comparison of ribosomal RNA gene sequences from the nucleus and the chloroplast. *Molecular Phylogenetics and Evolution*, 5(2), p. 391–402, 1996.

[352] E. M. Denovan-Wright, A. M. Nedelcu, and R. W. Lee. Complete sequence of the mitochondrial DNA of Chlamydomonas eugametos. *Plant molecular biology*, 36(2), p. 285–295, 1998.

[353] J. Kroymann and K. Zetsche. The mitochondrial genome of Chlorogonium elongatum inferred from the complete sequence. *Journal of molecular evolution*, 47(4), p. 431–440, 1998.

[354] D. R. Smith and R. W. Lee. Mitochondrial genome of the colorless green alga Polytomella capuana : a linear molecule with an unprecedented GC content. *Molecular biology and evolution*, 25(3), p. 487–496, 2008.

[355] G. Wolff, I. Plante, B. F. Lang, U. Kück, and G. Burger. Complete sequence of the mitochondrial DNA of the chlorophyte alga Prototheca wickerhamii : gene content and genome organization. *Journal of molecular biology*, 237(1), p. 75–86, 1994.

[356] M. Turmel, C. Otis, and C. Lemieux. The complete chloroplast DNA sequence of the green alga Nephroselmis olivacea : insights into the architecture of ancestral chloroplast genomes. *Proceedings of the National Academy of Sciences*, 96(18), p. 10248–10253, 1999.

[357] A.-P. Sibler, G. Dirheimer, and R. P. Martin. Nucleotide sequence of a yeast mitochondrial threonine-tRNA able to decode the CUN leucine codons. *FEBS letters*, 132(2), p. 344–348, 1981.

[358] M. E. Saks, J. Abelson, et al. The transfer RNA identity problem : a search for rules. *Science*, 263 (5144), p. 191–197, 1994.

[359] T. Salinas-Giegé, R. Giegé, and P. Giegé. tRNA biology in mitochondria. *International journal of molecular sciences*, 16(3), p. 4518–4559, 2015.

[360] D. H. Ardell and S. G. Andersson. TFAM detects co-evolution of tRNA identity rules with lateral transfer of histidyl-tRNA synthetase. *Nucleic acids research*, 34(3), p. 893–904, 2006.

[361] E. Picardi, T. M. R. Regina, A. Brennicke, and C. Quagliariello. REDIdb : the RNA editing database. *Nucleic acids research*, 35(suppl_1), p. D173–D177, 2007.

[362] B. R. Ruhfel, M. A. Gitzendanner, P. S. Soltis, D. E. Soltis, and J. G. Burleigh. From algae to angiosperms - inferring the phylogeny of green plants (Viridiplantae) from 360 plastid genomes. *BMC Evolutionary Biology*, 14(1), p. 23, 2014.

[363] D. E. Soltis, S. A. Smith, N. Cellinese, K. J. Wurdack, D. C. Tank, S. F. Brockington, N. F. Refulio-Rodriguez, J. B. Walker, M. J. Moore, B. S. Carlsward, et al. Angiosperm phylogeny : 17 genes, 640 taxa. *American journal of botany*, 98(4), p. 704–730, 2011.

[364] L. Fang, F. Leliaert, Z.-H. Zhang, D. Penny, and B.-J. Zhong. Evolution of the Chlorophyta : Insights from chloroplast phylogenomic analyses. *Journal of Systematics and Evolution*, 2017.

[365] T. M. Lowe and S. R. Eddy. tRNAscan-SE : a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic acids research*, 25(5), p. 955, 1997.

[366] S. Will, K. Reiche, I. L. Hofacker, P. F. Stadler, and R. Backofen. Inferring noncoding RNA families and classes by means of genome-scale structure-based clustering. *PLoS computational biology*, 3(4), p. e65, 2007.

[367] D. H. Huson and D. Bryant. Application of phylogenetic networks in evolutionary studies. *Molecular biology and evolution*, 23(2), p. 254–267, 2005.

[368] W. Li and A. Godzik. Cd-hit : a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13), p. 1658–1659, 2006.

[369] E. P. Nawrocki and S. R. Eddy. Infernal 1.1 : 100-fold faster RNA homology searches. *Bioinformatics*, 29(22), p. 2933–2935, 2013.

[370] A. M. Waterhouse, J. B. Procter, D. M. Martin, M. Clamp, and G. J. Barton. Jalview Version 2 - a multiple sequence alignment editor and analysis workbench. *Bioinformatics*, 25(9), p. 1189–1191, 2009.

[371] Z. Weinberg and R. R. Breaker. R2R-software to speed the depiction of aesthetic consensus RNA secondary structures. *BMC bioinformatics*, 12(1), p. 3, 2011.

[372] A. A. Farwagi, K. Fučíková, and H. A. McManus. Phylogenetic patterns of gene rearrangements in four mitochondrial genomes from the green algal family Hydrodictyaceae (Sphaeropleales, Chlorophyceae). *BMC genomics*, 16(1), p. 826, 2015.

[373] R. Giegé and G. Eriani. Transfer RNA recognition and aminoacylation by synthetases. *Encyclopedia of Life Sciences (ELS)*, p. 21, 2014.

[374] W. H. McClain and K. Foss. Changing the identity of a tRNA by introducing a GU wobble pair near the 3'acceptor end. *Science*, 240(4853), p. 793–796, 1988.

[375] K. Musier-Forsyth, N. Usman, S. Scaringe, J. Doudna, R. Green, and P. Schimmel. Specificity for aminoacylation of an RNA helix : an unpaired, exocyclic amino group in the minor groove. *Science*, 253(5021), p. 784–786, 1991.

[376] M. A. Tukalo, G. Yaremchuk, O. P. Kovalenko, I. A. Kriklivyi, and O. Gudzera. Recognition of tRNAs with a long variable arm by aminoacyl-tRNA synthetases. *Biopolymers and cell*, p. 311-323, 2013.

[377] H. Asahara, H. Himeno, K. Tamura, T. Hasegawa, K. Watanabe, and M. Shimizu. Recognition nucleotides of Escherichia coli tRNA-Leu and its elements facilitating discrimination from tRNA-Ser and tRNA-Tyr. *Journal of molecular biology*, 231(2), p. 219–229, 1993.

[378] B. Sohm, M. Frugier, H. Brule, K. Olszak, A. Przykorska, and C. Florentz. Towards understanding human mitochondrial leucine aminoacylation identity. *Journal of molecular biology*, 328(5), p. 995–1010, 2003.

[379] M. Pak, L. Pallanck, and L. H. Schulman. Conversion of a methionine initiator tRNA into a tryptophan-inserting elongator tRNA in vivo. *Biochemistry*, 31(13), p. 3303–3309, 1992.

[380] H. Himeno, T. Hasegawa, H. Asahara, K. Tamura, and M. Shimizu. Identity determinants of E. coli tryptophan tRNA. *Nucleic acids research*, 19(23), p. 6379–6382, 1991.

[381] S. Bilokapić, N. Ban, and I. Weygand-Đurašević. Seryl-tRNA Synthetases : Enzymes with Multiple Personalities. *Croatica Chemica Acta*, 82(2), p. 493–501, 2009.

[382] M. Santos, G. Keith, and M. F. Tuite. Non-standard translational events in Candida albicans mediated by an unusual seryl-tRNA with a 5'-CAG-3'(leucine) anticodon. *The EMBO journal*, 12 (2), p. 607, 1993.

[383] T. Muramatsu, S. Yokoyama, N. Horie, A. Matsuda, T. Ueda, Z. Yamaizumi, Y. Kuchino, S. Nishimura, and T. Miyazawa. A novel lysine-substituted nucleoside in the first position of the

anticodon of minor isoleucine tRNA from Escherichia coli. *Journal of Biological Chemistry*, 263 (19), p. 9261–9267, 1988.

[384] D. Hirsh. Tryptophan transfer RNA as the UGA suppressor. *Journal of molecular biology*, 58(2), p. 439–458, 1971.

[385] L. Cochella and R. Green. An active role for tRNA in decoding beyond codon : anticodon pairing. *Science*, 308(5725), p. 1178–1180, 2005.

[386] T. M. Schmeing, R. M. Voorhees, A. C. Kelley, and V. Ramakrishnan. How mutations in tRNA distant from the anticodon affect the fidelity of decoding. *Nature Structural and Molecular Biology*, 18(4), p. 432, 2011.

[387] C. J. Cox, B. Li, P. G. Foster, T. M. Embley, and P. Civáň. Conflicting phylogenies for early land plants are caused by composition biases among synonymous substitutions. *Systematic Biology*, 63 (2), p. 272–279, 2014.

[388] A. Satjarak, J. A. Burns, E. Kim, and L. E. Graham. Complete mitochondrial genomes of prasinophyte algae Pyramimonas parkeae and Cymbomonas tetramitiformis. *Journal of phycology*, 53(3), p. 601–615, 2017.

[389] Š. Hrdá, M. Hroudová, Č. Vlček, and V. Hampl. Mitochondrial Genome of Prasinophyte Alga Pyramimonas parkeae. *Journal of Eukaryotic Microbiology*, 64(3), p. 360–369, 2017.

[390] K. Fučíková, P. O. Lewis, and L. A. Lewis. Chloroplast phylogenomic data from the green algal order Sphaeropleales (Chlorophyceae, Chlorophyta) reveal complex patterns of sequence evolution. *Molecular phylogenetics and evolution*, 98, p. 176–183, 2016.

[391] L. Krienitz, C. Bock, P. Dadheech, and T. Pröschold. Taxonomic reassessment of the genus Mychonastes (Chlorophyceae, Chlorophyta) including the description of eight new species. *Phycologia*, 50(1), p. 89–106, 2011.

[392] N. P. Tippery, K. Fučíková, P. O. Lewis, and L. A. Lewis. Probing the monophyly of the Sphaeropleales (Chlorophyceae) using data from five genes. *Journal of Phycology*, 48(6), p. 1482–1493, 2012.

[393] K. Fučíková, V. R. Flechtner, and L. A. Lewis. Revision of the genus Bracteacoccus Tereg (Chlorophyceae, Chlorophyta) based on a phylogenetic approach. *Nova Hedwigia*, 96(1–2), p. 15–59, 2013.

[394] K. E. Glover, D. F. Spencer, and M. W. Gray. Identification and structural characterization of nucleus-encoded transfer RNAs imported into wheat mitochondria. *Journal of Biological Chemistry*, 276(1), p. 639–648, 2001.

[395] L. Marechal-Drouard, P. Guillemaut, A. Cosset, M. Arbogast, F. Weber, J.-H. Weil, and A. Dietrich. Transfer RNAs of potato (Solanum tuberosum) mitochondria have different genetic origins. *Nucleic acids research*, 18(13), p. 3689–3696, 1990.

[396] L. Marechal-Drouard, I. Small, J.-H. Weil, and A. Dietrich. Transfer RNA import into plant mitochondria. In *Methods in enzymology*, volume 260, p. 310–327. Elsevier, 1995.

[397] M. Ehara, Y. Hayashi-Ishimaru, Y. Inagaki, and T. Ohama. Use of a deviant mitochondrial genetic code in yellow-green algae as a landmark for segregating members within the phylum. *Journal of molecular evolution*, 45(2), p. 119–124, 1997.

[398] F. Weber, A. Dietrich, J.-H. Weil, and L. Maréchal-Drouard. A potato mitochondrial isoleucine tRNA is coded for by a mitochondrial gene possessing a methionine anticodon. *Nucleic acids research*, 18(17), p. 5027–5030, 1990.

[399] J. Moriya, T. Yokogawa, K. Wakita, T. Ueda, K. Nishikawa, P. F. Crain, T. Hashizume, S. C. Pomerantz, and J. A. McCloskey. A novel modified nucleoside found at the first position of the anticodon of methionine tRNA from bovine liver mitochondria. *Biochemistry*, 33(8), p. 2234–2239, 1994.

[400] S. Nakano, T. Suzuki, L. Kawarada, H. Iwata, K. Asano, and T. Suzuki. NSUN3 methylase initiates 5-formylcytidine biogenesis in human mitochondrial tRNA Met. *Nature chemical biology*, 12(7), p. 546, 2016.

[401] C. Takemoto, L. L. Spremulli, L. A. Benkowski, T. Ueda, T. Yokogawa, and K. Watanabe. Unconventional decoding of the AUA codon as methionine by mitochondrial tRNA Met with the anticodon f 5 CAU as revealed with a mitochondrial in vitro translation system. *Nucleic acids research*, 37(5), p. 1616–1627, 2009.

[402] K. Tomita, T. Ueda, S. Ishiwa, P. F. Crain, J. A. McCloskey, and K. Watanabe. Codon reading patterns in Drosophila melanogaster mitochondria based on their tRNA sequences : a unique wobble rule in animal mitochondria. *Nucleic acids research*, 27(21), p. 4291–4297, 1999.

[403] J.-S. Brouard, C. Otis, C. Lemieux, and M. Turmel. Chloroplast DNA sequence of the green alga Oedogonium cardiacum (Chlorophyceae) : unique genome architecture, derived characters shared with the Chaetophorales and novel genes acquired through horizontal transfer. *BMC genomics*, 9 (1), p. 290, 2008.

[404] N. C. Martin, H. D. Pham, K. Underbrink-Lyon, D. L. Miller, and J. E. Donelson. Yeast mitochondrial tRNATrp can recognize the nonsense codon UGA. *Nature*, 285(5766), p. 579, 1980.

[405] A. Sibler, R. Bordonne, G. Dirheimer, and R. Martin. Primary structure of yeast mitochondrial tryptophan-tRNA capable of translating the termination UGA codon. *Comptes rendus des seances de l'Academie des sciences. Serie D, Sciences naturelles*, 290(11), p. 695–698, 1980.

[406] T. Salinas, F. Duby, V. Larosa, N. Coosemans, N. Bonnefoy, P. Motte, L. Maréchal-Drouard, and C. Remacle. Co-evolution of mitochondrial tRNA import and codon usage determines translational efficiency in the green alga Chlamydomonas. *PLoS genetics*, 8(9), p. e1002946, 2012.

[407] C. Lemieux, C. Otis, and M. Turmel. Chloroplast phylogenomic analysis resolves deep-level relationships within the green algal class Trebouxiophyceae. *BMC evolutionary biology*, 14(1), p. 211, 2014.

[408] L. Sun, L. Fang, Z. Zhang, X. Chang, D. Penny, and B. Zhong. Chloroplast phylogenomic inference of green algae relationships. *Scientific reports*, 6, p. 20528, 2016.

[409] K. Fučíková, P. O. Lewis, and L. A. Lewis. Putting incertae sedis taxa in their place : a proposal for ten new families and three new genera in Sphaeropleales (Chlorophyceae, Chlorophyta). *Journal of phycology*, 50(1), p. 14–25, 2014.

[410] J.-F. Pombert, C. Otis, C. Lemieux, and M. Turmel. The complete mitochondrial DNA sequence of the green alga Pseudendoclonium akinetum (Ulvophyceae) highlights distinctive evolutionary trends in the Chlorophyta and suggests a sister-group relationship between the Ulvophyceae and Chlorophyceae. *Molecular biology and evolution*, 21(5), p. 922–935, 2004.

[411] N. Lartillot, H. Brinkmann, and H. Philippe. Suppression of long-branch attraction artefacts in the animal phylogeny using a site-heterogeneous model. *BMC evolutionary biology*, 7(1), p. S4, 2007.

[412] M.-A. Selosse, B. Albert, and B. Godelle. Reducing the genome size of organelles favours gene transfer to the nucleus. *Trends in ecology & evolution*, 16(3), p. 135–141, 2001.

[413] W. Martin and R. G. Herrmann. Gene transfer from organelles to the nucleus : how much, what happens, and why ? *Plant physiology*, 118(1), p. 9–17, 1998.

[414] M. Lynch and J. L. Blanchard. Deleterious mutation accumulation in organelle genomes. In *Mutation and Evolution*, p. 29–39. Springer, 1998.

[415] W. Martin. Gene transfer from organelles to the nucleus : frequent and in big chunks. *Proceedings of the National Academy of Sciences*, 100(15), p. 8612–8614, 2003.

[416] X. Perez-Martínez, A. Antaramian, M. Vázquez-Acevedo, S. Funes, E. Tolkunova, J. d'Alayer, M. G. Claros, E. Davidson, M. P. King, and D. González-Halphen. Subunit II of cytochrome c oxidase in Chlamydomonad algae is a heterodimer encoded by two independent nuclear genes. *Journal of Biological Chemistry*, 2000.

[417] X. Pérez-Martınez, M. Vázquez-Acevedo, E. Tolkunova, S. Funes, M. G. Claros, E. Davidson, M. P. King, and D. González-Halphen. Unusual Location of a Mitochondrial Gene subunit IIi of cytochrome c oxidase is encoded in the nucleus of chlamydomonad algae. *Journal of Biological Chemistry*, 275(39), p. 30144–30152, 2000.

[418] S. Funes, E. Davidson, M. G. Claros, R. Van Lis, X. Pérez-Martınez, M. Vázquez-Acevedo, M. P. King, and D. González-Halphen. The typically mitochondrial DNA-encoded ATP6 subunit of the F1F0-ATPase is encoded by a nuclear gene in Chlamydomonas reinhardtii. *Journal of Biological Chemistry*, 277(8), p. 6051–6058, 2002.

[419] P. Cardol, M. Lapaille, P. Minet, F. Franck, R. F. Matagne, and C. Remacle. ND3 and ND4L subunits of mitochondrial complex I, both nucleus encoded in Chlamydomonas reinhardtii, are required for activity and assembly of the enzyme. *Eukaryotic cell*, 5(9), p. 1460–1467, 2006.

[420] K. L. Adams, H. C. Ong, and J. D. Palmer. Mitochondrial gene transfer in pieces : fission of the ribosomal protein gene rpl2 and partial or complete gene transfer to the nucleus. *Molecular Biology and Evolution*, 18(12), p. 2289–2297, 2001.

[421] K. L. Adams and J. D. Palmer. Evolution of mitochondrial gene content : gene loss and transfer to the nucleus. *Molecular phylogenetics and evolution*, 29(3), p. 380–395, 2003.

[422] E. Rodríguez-Salinas, H. Riveros-Rosas, Z. Li, K. Fučíková, J. J. Brand, L. A. Lewis, and D. González-Halphen. Lineage-specific fragmentation and nuclear relocation of the mitochondrial cox2 gene in chlorophycean green algae (Chlorophyta). *Molecular phylogenetics and evolution*, 64(1), p. 166–176, 2012.

[423] J. S. Sabir, R. K. Jansen, D. Arasappan, V. Calderon, E. Noutahi, C. Zheng, S. Park, M. J. Sabir, M. N. Baeshen, N. H. Hajrah, et al. The nuclear genome of Rhazya stricta and the evolution of alkaloid diversity in a medically relevant clade of Apocynaceae. *Scientific reports*, 6, p. 33782, 2016.

[424] M. Lafond, M. Semeria, K. Swenson, E. Tannier, and N. El-Mabrouk. Gene tree correction guided by orthology. *BMC Bioinformatics*, 14 (supp 15)(S5), 2013.

[425] C. Chauve, N. El-Mabrouk, L. Guéguen, M. Semeria, and E. Tannier. Duplication, Rearrangement and Reconciliation : A Follow-Up 13 Years Later. In C. Chauve, N. El-Mabrouk, and E. Tannier, editors, *Models and Algorithms for Genome Evolution*, p. 47–62. Springer, London, 2013. ISBN 978-1-4471-5297-2.

[426] S. Penel, A.-M. Arigon, J.-F. Dufayard, A.-S. Sertier, V. Daubin, L. Duret, M. Gouy, and G. Perrière. Databases of homologous gene families for comparative genomics. *BMC bioinformatics*, 10(6), p. S3, 2009.

[427] R. Datta, C. Meacham, B. Samad, C. Neyer, and K. Sjölander. Berkeley PHOG : PhyloFacts orthology group prediction web server. *Nucleic Acids Research*, 37, p. W84-W89, 2009.

[428] L. Pryszcz, J. Huerta-Cepas, and T. Gabaldón. MetaPhOrs : orthology nd paralogy predictions from multiple phylogenetic evidence using a consistency-based confidence score. *Nucleic Acids Res.*, 39, p. e32, 2011.

[429] J. Huerta-Cepas, S. Capella-Gutierrez, L. Pryszcz, I. Denisov, D. Kormes, M. Marcet-Houben, and T. G. on. PhylomeDB v3.0 : an expanding repository of genome-wide collections of trees, alignments and phylogeny-based orthology and paralogy predictions. *Nucleic Acids Research*, 39, p. D556-D560, 2011.

[430] H. Mi, A. Muruganujan, and P. D. Thomas. PANTHER in 2013 : modeling the evolution of gene function, and other gene attributes, in the context of phylogenetic trees. *Nucleic acids research*, 41 (D1), p. D377–D386, 2012.

[431] F. Ronquist and J. Huelsenbeck. MrBayes3 : Bayesian phylogenetic inference under mixed models. *Bioinformatics*, 19, p. 1572-1574, 2003.

[432] R. Dondi, N. El-Mabrouk, and K. Swenson. Gene tree correction for reconciliation and species tree inference : complexity and algorithms. *Journal of Discrete Algorithms*, 25(submitted), p. 51–65, 2013.

[433] K. Chen, D. Durand, and M. Farach-Colton. NOTUNG : a program for dating gene duplications and optimizing gene family trees. *Journal of Computational Biology*, 7(3-4), p. 429–447, 2000.

[434] L. Arvestad, A. Berglund, J. Lagergren, and B. Sennblad. Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. In *RECOMB*, p. 326-335, 2004.

[435] D. Huson and C. Scornavacca. A survey of combinatorial methods for phylogenetic networks. *Genome Biol. Evol.*, 3, p. 23-35, 2011.

[436] A. Berglund-Sonnhammer, P. Steffansson, M. Betts, and D. Liberles. Optimal gene trees from sequences and species trees using a soft interpretation of parsimony. *Journal of Molecular Evolution*, 63, p. 240-250, 2006.

[437] A. G. Kluge and J. S. Farris. Quantitative phyletics and the evolution of anurans. *Syst. Zool.*, 18, p. 1–32, 1969.

[438] N. Saitou and M. Nei. The neighbor-joining method : a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4), p. 406–425, 1987.

[439] O. Gascuel and M. Steel. Neighbor-joining revealed. *Mol Biol Evol*, 23(11), p. 1997–2000, 2006.

[440] R. Chaudhary, J. Burleigh, and O. Eulenstein. Efficient error correction algorithms for gene tree reconciliation based on duplication, duplication and loss, and deep coalescence. *BMC Bioinformatics*, 13(Supp.10), p. S11, 2011.

[441] M. Lafond, C. Chauve, R. Dondi, Manuel, and N. El-Mabrouk. Polytomy refinement for the correction of dubious duplications in gene trees. *Bioinformatics*, 30(17), p. i519-i526, 2014.

[442] M. A. Khan, I. Elias, E. Sjölund, K. Nylander, R. V. Guimera, R. Schobesberger, P. Schmitzberger, J. Lagergren, and L. Arvestad. Fastphylo : fast tools for phylogenetics. *BMC Bioinformatics*, 14, p. 334, 2013.

[443] J. M. Lucas, M. Muffato, and H. Roest Crollius. PhylDiag : identifying complex synteny blocks that include tandem duplications using phylogenetic gene trees. *BMC Bioinformatics*, 15(1), p. 268, 2014.

[444] O. Mahmudi, J. Sjöstrand, B. Sennblad, and J. Lagergren. Genome-wide probabilistic reconciliation analysis across vertebrates. *BMC Bioinformatics*, 14 Suppl 15, p. S10, 2013.

[445] J. J. Smith, S. Kuraku, C. Holt, T. Sauka-Spengler, N. Jiang, M. S. Campbell, M. D. Yandell, T. Manousaki, A. Meyer, O. E. Bloom, et al. Sequencing of the sea lamprey (Petromyzon marinus) genome provides insights into vertebrate evolution. *Nature genetics*, 45(4), p. 415–421, 2013.

[446] T. K. Mehta, V. Ravi, S. Yamasaki, A. P. Lee, M. M. Lian, B.-H. Tay, S. Tohari, S. Yanai, A. Tay, S. Brenner, and B. Venkatesh. Evidence for at least six Hox clusters in the Japanese lamprey (Lethenteron japonicum). *Proceedings of the National Academy of Sciences*, 110(40), p. 16044–16049, 2013.

[447] A. Rambaut. Figtree. http ://tree.bio.ed.ac.uk/software/figtree/, 2006.

[448] E. L. L. Sonnhammer, T. Gabaldón, A. W. Sousa da Silva, M. Martin, M. Robinson-Rechavi, B. Boeckmann, P. D. Thomas, C. Dessimoz, and the Quest for Orthologs consortium. Big data and other challenges in the quest for orthologs. *Bioinformatics*, p. btu492, 2014.

[449] O. Cohen, H. Ashkenazy, D. Burstein, and T. Pupko. Uncovering the co-evolutionary network among prokaryotic genes. *Bioinformatics*, 28(18), p. i389–i394, 2012.

[450] J. Maňuch, M. Patterson, R. Wittler, C. Chauve, and E. Tannier. Linearization of ancestral multichromosomal genomes. *BMC Bioinformatics*, 13 Suppl 19, p. S11, 2012.

[451] S. S. Abby, E. Tannier, M. Gouy, and V. Daubin. Lateral gene transfer as a support for the tree of life. *Proceedings of the National Academy of Sciences USA*, 109(13), p. 4962–4967, 2012.

[452] M. Patterson, G. Szöllősi, V. Daubin, and E. Tannier. Lateral gene transfer, rearrangement, reconciliation. *BMC Bioinformatics*, 14 Suppl 15, p. S4, 2013.

[453] E. Noutahi, M. Semeria, M. Lafond, J. Seguin, L. Gueguen, N. El-Mabrouk, and E. Tannier. Efficient Gene Tree Correction Guided by Genome Evolution. *Plos.One*, 11(8), 2016.

[454] Z.-Z. Chen, F. Deng, and L. Wang. Simultaneous identification of duplications, losses, and lateral gene transfers. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 9(5), p. 1515–1528, 2012.

[455] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Inc. Boston, MA, USA, 1st edition, 1989. ISBN 0201157675.

[456] H. Matsuda. Protein phylogenetic inference using maximum likelihood with a genetic algorithm. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, p. 512-523, 1996.

[457] P. Lewis. Genetic algorithm for Maximum-Likelihood phylogeny inference using nucleotide sequence data. *Mol. Biol. Evol.*, 15(3), p. 277-283, 1998.

[458] A. Skourikhine. Phylogenetic tree reconstruction using self-adaptive genetic algorithm. In *Proceedings IEEE International Symposium on Bio-Informatics and Biomedical Engineering*, IEEE Int. Symp. Bio-Inf. and Biomed. Eng., p. 129-134, 2000.

[459] K. Katoh, K. Kuma, and T. Miyata. Genetic algorithm-based maximum- likelihood analysis for molecular phylogeny. *J. Mol. Evol.*, 53, p. 477-484, 2001.

[460] A. R. Lemmon and M. C. Milinkovitch. The Metapopulation Genetic Algorithm : An Efficient Solution for the Problem of Large Phylogeny Estimation. *Proc.Nat.Acad.Sci.USA*, 99(16), p. 10516-10521, 2002.

[461] D. Zwickl. *Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion*. PhD thesis, The University of Texas at Austin, 2006.

[462] C. Lanave, G. Preparata, C. Sacone, and G. Serio. A new method for calculating evolutionary substitution rates. *Journal of molecular evolution*, 20(1), p. 86–93, 1984.

[463] Z. Yang. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites : approximate methods. *Journal of Molecular evolution*, 39(3), p. 306–314, 1994.

[464] A. Stamatakis, T. Ludwig, and H. Meier. RAxML-III : a fast program for maximum likelihood-based inference of large phylogenetic trees. *Bioinformatics*, 21(4), p. 456–463, 2004.

[465] A. Tofigh. *Using trees to capture reticulate evolution : lateral gene transfers and cancer progression*. PhD thesis, KTH, 2009.

[466] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387332545.

[467] N. Srinivas and K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3), p. 221–248, 1994.

[468] B. L. Miller, D. E. Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3), p. 193–212, 1995.

[469] B. Boeckmann, M. Rechavi, I. Xenarios, and C. Dessimoz. Conceptual framework and pilot study to benchmark phylogenomic databases based on reference gene trees. *Briefings in bioinformatics*, 12(5), p. 423–435, 2011.

[470] B. Boeckmann, M. Marcet-Houben, J. A. Rees, K. Forslund, J. Huerta-Cepas, M. Muffato, P. Yilmaz, I. Xenarios, P. Bork, S. E. Lewis, et al. Quest for orthologs entails quest for tree of life : in search of the gene stream. *Genome biology and evolution*, 7(7), p. 1988–1999, 2015.

[471] D. H. Ardell. Computational analysis of tRNA identity. *FEBS letters*, 584(2), p. 325–333, 2010.

[472] O. Tremblay-Savard, B. Benzaid, B. F. Lang, and N. El-Mabrouk. Evolution of tRNA repertoires in Bacillus inferred with OrthoAlign. *Molecular biology and evolution*, 32(6), p. 1643–1656, 2015.

[473] B. Oldenkott, K. Yamaguchi, S. Tsuji-Tsukinoki, N. Knie, and V. Knoop. Chloroplast RNA editing going extreme : more than 3400 events of C-to-U editing in the chloroplast transcriptome of the lycophyte Selaginella uncinata. *RNA*, 2014.

[474] S. Alon, M. Erew, and E. Eisenberg. DREAM : a webserver for the identification of editing sites in mature miRNAs using deep sequencing data. *Bioinformatics*, 31(15), p. 2568–2570, 2015.

[475] Z. Wang, J. Lian, Q. Li, P. Zhang, Y. Zhou, X. Zhan, and G. Zhang. RES-Scanner : a software package for genome-wide identification of RNA-editing sites. *GigaScience*, 5(1), p. 37, 2016.

[476] H. Lenz and V. Knoop. PREPACT 2.0 : predicting C-to-U and U-to-C RNA editing in organelle genome sequences with multiple references and curated RNA editing annotation. *Bioinformatics and biology insights*, 7, p. BBI–S11059, 2013.

[477] J. P. Mower. PREP-Mt : predictive RNA editor for plant mitochondrial genes. *BMC bioinformatics*, 6(1), p. 96, 2005.

[478] M. P. Cummings and D. S. Myers. Simple statistical models predict C-to-U edited sites in plant mitochondrial RNA. *BMC bioinformatics*, 5(1), p. 132, 2004.

[479] P. Du, L. Jia, and Y. Li. CURE-Chloroplast : a chloroplast C-to-U RNA editing predictor for seed plants. *BMC bioinformatics*, 10(1), p. 135, 2009.

[480] J. Thompson and S. Gopal. Genetic algorithm learning as a robust approach to RNA editing site prediction. *BMC bioinformatics*, 7(1), p. 145, 2006.

[481] Z. Ouyang, F. Liu, C. Zhao, C. Ren, G. An, C. Mei, X. Bo, and W. Shu. Accurate identification of RNA editing sites from primitive sequence with deep neural networks. *Scientific reports*, 8(1), p. 6005, 2018.

[482] J. Zhou and O. G. Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10), p. 931, 2015.

[483] R. Poplin, D. Newburger, J. Dijamco, N. Nguyen, D. Loy, S. S. Gross, C. Y. McLean, and M. A. DePristo. Creating a universal SNP and small indel variant caller with deep neural networks. *BioRxiv*, p. 092890, 2017.

[484] T. Yue and H. Wang. Deep Learning for Genomics : A Concise Overview. *arXiv preprint arXiv :1802.00810*, 2018.

[485] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle. Deep learning for computational biology. *Molecular systems biology*, 12(7), p. 878, 2016.

[486] R. Luo, F. J. Sedlazeck, T.-W. Lam, and M. Schatz. Clairvoyante : a multi-task convolutional deep neural network for variant calling in Single Molecule Sequencing. *bioRxiv*, p. 310458, 2018.

[487] A. Telenti, C. Lippert, P.-C. Chang, and M. DePristo. Deep learning of genomic variation and regulatory network data. *Human molecular genetics*, 27(R1), p. R63–R71, 2018.

[488] L. Li and M. S. Bansal. An Integrated Reconciliation Framework for Domain, Gene, and Species Level Evolution. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2018.

[489] S. A. Muhammad, B. Sennblad, and J. Lagergren. Species tree-aware simultaneous reconstruction of gene and domain evolution. *bioRxiv*, p. 336453, 2018.

Annexe A

Supplementary Information for "CoreTracker : accurate codon reassignment prediction, applied to mitochondrial genomes"

Il s'agit de l'annexe de l'article présenté dans le chapitre 3 et publié dans *Bioinformatics*.

Supplementary Information

CoreTracker: accurate codon reassignment prediction, applied to mitochondrial genomes

Noutahi Emmanuel[1,†], Calderon Virginie[1,†], Blanchette Mathieu[3], Lang B. Franz[2] and El-Mabrouk Nadia[1,*]

[1]Département d'Informatique et de Recherche Opérationnelle (DIRO), Université de Montréal, CP 6128 Succursale Centre-Ville, Montréal, QC, Canada.

[2]Centre Robert Cedergren, Département de Biochimie, Université de Montréal, CP 6128 Succursale Centre-Ville, Montréal, QC, Canada.

[3]School of Computer Science, McGill University, McConnell Engineering Bldg., Rm. 318, 3480 Rue University, Montréal, Québec H3A 0E9 Canada.

*To whom correspondence should be addressed.

[†]The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors

1. Supplementary Methods

1.1 Protein sequence used as input for CoreTracker

Using the input protein sequences, CoreTracker constructs an initial alignment using Muscle (Edgar, 2004) or Mafft (Katoh and Standley, 2013), followed by a refinement step in which a HMM profile is constructed from the alignment and used to realign the sequences. The refinement step is executed a given number of times with the previous alignment as input, and the quality of each resulting alignment is evaluated based on the posterior probabilities of aligned residues. The best alignment found is retained and called the *raw* alignment. Alternatively, users can also provide their own protein alignment or HMM profile for refinement. Here we used pre-aligned mitochondrial sequences provided by GenDecoder (Abascal *et al*., 2009) to build a profile-HMM for each mitochondrial metazoan protein using HMMER *hmmbuild* (Eddy, 2001). This HMM was used for the alignment of metazoan mitochondrial sequences. For yeast sequences, the iterative pipeline described above was used.

CoreTracker computes the filtered alignment using conservation thresholds based on three criteria for each column: i) the gap percentage, ii) the information content (IC) and iii) the amino acid identity percentage (Figure 1A). The default parameters for this filtering step are 40% for gap, 30% for IC content and 50% for identity.

1.2 Identification of candidate genomes

For each genome G, the pairwise similarity score to the other genomes with respect to both filtered and X_i-filtered alignments are computed using the BLOSUM62 substitution matrix, defining respectively the average conservation $\mu(G)$ and the observed conservation $\mu_{X_i}(G)$. The Wilcoxon's signed-rank test (with α=0.05) is used to test the alternative hypothesis $\mu(G) > \mu_{X_i}(G)$ against the null hypothesis $\mu(G) = \mu_{X_i}(G)$.

1.3 Reconstruction of ancestral states

For each tested reassignment from amino acid X_i to X_j, nodes of the phylogeny are labeled with a binary encoding. A leaf corresponding to genome G is labeled X_j if G is in the set of candidates for a reassignment from X_i to X_j, and it is labeled X_i otherwise. The bottom-up phase of the Fitch's algorithm (Fitch, 1971) is used to label internal nodes: label a node as X_i (respectively X_j) if all children nodes are labeled X_i (respectively X_j), and as $\{X_i, X_j\}$ otherwise. Reconstruction of ancestral

nodes states allows the identification of the monophyletic group of species affected by the potential reassignment from X_i to X_j.

1.4 Identification of codons involved in a reassignment

For amino acids encoded by multiple codons, a generalized Fisher's exact test with mid-p correction was applied (significance level α=0.05) to evaluate the synonymous codon usage difference in type I columns, where X_j is prevalent, compared to type II columns, where X_i is prevalent (see Figure 1D of the main document).

Since the Fisher's exact test is not applicable for amino acids encoded by a single codon, in this latter case we use a binomial test instead, where our null hypothesis is the absence of bias in the substitution of X_i to X_j. The binomial test can be formulated as follows. Denote by p_{ij} the probability of a substitution of X_i to X_j, p_{ij} can be inferred from the alignment or with a substitution matrix ($p_{ij} = q_i q_j e^{\lambda s_{ij}}$), where q_i and q_j are respectively the frequency of X_i and X_j in the alignment and s_{ij} the substitution score of X_i by X_j. Let n_1 be the usage of codon C_i of X_i in column I and n the total codon usage of C_i, then we reject the null hypothesis: $\mu(x) = p_{ij}$ if $P(x \leq n_1) = Binomial(n_1, n, p_{ij}) > 1 - \alpha$.

1.5 Random Forest Classifier

For each possible reassignment $C(X_i, X_j)$ in each genome for which the codon C of interest is present in X_j prevalent columns, 12 variables were considered (see Table 1) to construct the RF training and testing datasets. Feature selection for the RF was performed using a combination of chi2-based univariate selection and mean decrease impurity, resulting in ten features being selected. The first (called Fitch) is the result of step C (Figure 1C) illustrating the phylogenetic context of the potential reassignment. The next five selected variables contain results of step D on interesting reassignments, namely codon usage of C in G in columns I and II (Figure 1D). The Subst.count, Codon.likelihood and Gene.fraction variables are used to evaluate the likeliness of a codon reassignment, based on the fact that a codon reassignment may be reflected by frequent substitutions occurring between two very divergent amino acids. Codon.likelihood corresponds to the likelihood score defined by Telford and coworkers (Telford *et al*., 2000). This score represents the relative probability that C codes for X_j given the amino acids aligned to C and a BLOSUM62 matrix. Gene.fraction reflects the proportion of genes, using C, that are concerned by the reassignment in G. If X_j prevalent columns are distributed in all genes using C, this will reinforce the prediction for reassignment. Conversely, X_j prevalent columns localized in a single gene may suggest other event such as mRNA editing. Two final

variables contain information on the length of the genomes and codon nature. Notice that more variables have been initially tested: one reflecting the genome GC content and another considering the translation redundancy of the third nucleotide of the codon, but they have not been retained as they were not informative.

The RF was trained on metazoan mitochondrial genomes using the most confirmed reassignments, resulting in 42 non-ambiguously identified codon reassignments, used as the set of true positives (see Table S1). The remaining set of 905 other $C(X_i, X_j)$ possibilities considered by CoreTracker form the true negative pool. In order to reduce the imbalance in data, this pool was randomly down-sampled to 420 reassignments, representing the true negative training set of the RF.

CoreTracker default filtering parameters, which are not too stringent (0.4 gap, 0.5 identity and 0.3 IC content) were used to compute all variables. Each row of the constructed dataset corresponds to a candidate reassignment in a genome, whether this reassignment is probable or not.

We use the RF implementation of the python package scikit-learn (Pedregosa *et al*., 2011) with 1000 trees, at least 100 leaves per tree and the square root of the number of features to train individual trees.

1.6 Validation

We implemented two validation steps in order to further confirm each predicted codon reassignment. The first one proceeds by validating the prediction in each clade obtained from the phylogenetic tree. For this purpose, we first predict the ancestral decoding of the reassigned codons, using parsimony. On the path from the root to the leaves, we consider the largest clade with an ancestral genome where the reassignment was predicted. Those clades are then validated one by one by assessing the improvement in quality of the alignment (restricted to genomes in the clade of interest and genomes where the reassignment was not predicted) when the appropriate decoding of the codon is used for the genomes in the clade. The second validation step proceeds by validating simultaneously all genomes with a predicted codon reassignment. It computes the Sum-of-Pairs (SP) score for each column from the filtered alignment affected by the reassignment with the initial and new genetic code. The difference in SP scores distribution is then assessed using a Wilcoxon's test. This validation step can be performed by either considering a codon reassignment at a time or all synonymous codons of an amino acid that were reassigned simultaneously. We chose the latter as there is often a bias in synonymous codon usage that causes a disparity in codon usage, resulting in not enough positions for the validation of certain codon reassignments.

2. Supplementary results

2.1 Comparison to Bagheera and FACIL on a yeast nuclear dataset

In order to demonstrate that CoreTracker is able to predict codon reassignments in nuclear genomes, we measured its performance on a dataset of *Saccharomycetes* nuclear genomes. In some yeast *Candida* species, the codon CUG is reassigned from leucine to serine(Santos *et al.*, 2011; Kawaguchi *et al.*, 1989) and in *Pachysolen tannophilus*, CUG is reassigned to alanine (Riley *et al.*, 2016; Mühlhausen *et al.*, 2016). We compared CoreTracker's performance on this nuclear dataset to FACIL (Dutilh *et al.*, 2011) and Bagheera (Mühlhausen and Kollmar, 2014), with the latter being specifically designed for CUG codon reassignments in yeast nuclear genomes.

To construct the dataset, we first download the cDNA and transcripts catalog of several publicly available yeast nuclear genomes assemblies from RefSeq (O'Leary *et al.*, 2016). Next, starting from 18 *S. cerevisiae S288C* proteins, we use TBLASTN (Altschul *et al.*, 1997; Camacho *et al.*, 2009) to search for their corresponding homologs in the other yeast genomes. The 18 proteins of the dataset consist of 16 motors and cytoskeletal proteins and two lipases. Detailed information on the dataset (species, accession, known codon reassignments and gene content in the supermatrix) can be found in supplementary Table S5.

For CoreTracker, we use the CDS of the 18 proteins as input. For FACIL we had to use all CDS as initial prediction with the 18 proteins lacked precision. We submitted the whole transcriptome (including non-coding RNAs) to Bagheera, since it also relies on tRNA identification. All programs were run with default parameters and the standard genetic code was set as reference code for CoreTracker and FACIL. Alignment of the translated sequences under this code was performed by CoreTracker using Mafft (see https://github.com/UdeM-LBIT/CoreTracker/tree/master/data for alignments and raw output). CUG decoding was selected for Bagheera based on "CTG position conservation" and "$tRNA_{CAG}$ identity prediction" when available.

Figure S3 show the predicted CUG decoding for each of the three programs compared to the known decoding of CUG codons in each genome. As shown on this Figure, Bagheera was able to predict all CUG (Leu, Ser) codon reassignment as expected, but failed to predict CUG (Leu, Ala) in *P. tannophilus.* Instead, it predicted the decoding of CUG as serine (23 positions vs 7 for the standard decoding) in that genome. FACIL failed to predict most codon reassignments but correctly reported the true decoding as the most frequent amino acid aligned to the query CUG codon, even in *P. tannophilus*. Unfortunately, in all those cases, its final prediction was the undetermined amino acid 'X'. Finally CoreTracker was able to accurately predict CUG decoding in most genomes. It only failed to predict CUG (Leu, Ser) in *Lodderomyces elongisporus NRLL YB-4239* and this was because

there were not enough CUG codons in conserved serine positions of the dataset. Our results on the yeast nuclear dataset show that CoreTracker is as accurate as Bagheera without being limited to CUG (Leu, Ser) in yeast nuclear genome. It thus highlights CoreTracker's efficiency and its applicability to both mitochondrial and nuclear dataset even with a small subset of genes.

2.2 Effect of errors in the dataset on prediction accuracy

We measured the impact of the quality of the input protein alignment on CoreTracker's performance using simulated data. We start by simulating a pure birth (Yule, 1925) rooted species tree of nine extant taxa, with branch length. From the root of this tree, the evolution of an ancestral protein sequence of 1000 amino acids, was simulated to obtain extant sequences at leaves. Sequences simulation was performed with constant rate of substitution (0.45) and deletion (0.3) per site. For each protein sequence a corresponding cDNA was generated with equal probability of codon usage. Two codons reassignments were inserted in the dataset: AUA(Ile, Met) in the ancestral genome leading to A and B and AAA(Lys, Asn) in genomes I and G (see Figure S4). Codon reassignments were designed in a way to not make them perfect. Using the simulated sequence history as the reference alignment, we create alternative alignments with different rates of errors. Errors were introduced to mimic misalignment. More specifically, we create a misalignment at position i by randomly selecting a subset S of taxa and appending a gap after i for all taxa in S. For the remaining taxa, the gap was added at the position just before i. For an error rate $e = 1.0$ and an initial alignment of length l a new alignment with a maximum of $e*l$ new positions will be created (since gap-only columns will be removed).

Figure S4 show the effect of alignment quality on precision and recall. On Figure S4-A, CoreTracker's prediction for increasing rate of errors is reported for the four genomes where a reassignment was introduced. Figure S4-B and C respectively show precision and recall using predictions that pass at least one validation step and standard results where both validation are required. Except for the pic of precision observed at $e = 4.0$ on Figure S4-C, precision and recall tend to decrease with increasing rate of errors. CoreTracker was however able to predict the true codon reassignments even for the unrealistic alignment obtained under the most extreme error rates (only one validation succeed in this case), showing its robustness towards errors in the protein alignment.

We have also measured the effect of errors in the phylogeny on prediction accuracy. For that purpose, we created alternative phylogeny by randomly swapping or contracting branches of the simulated tree. These alternative phylogenies (with an alignment error rate of 2.0) did not affect results. Indeed, reassignments were still predicted and validated but with just a slightly lower

prediction score when erroneous phylogenies were considered. We hypothesize that there were too many positions affected by the reassignments, making the topology of the tree not relevant for prediction. This will certainly not hold true for large phylogenies or reassignments where codon usage is a limitation.

3. Supplementary Figures

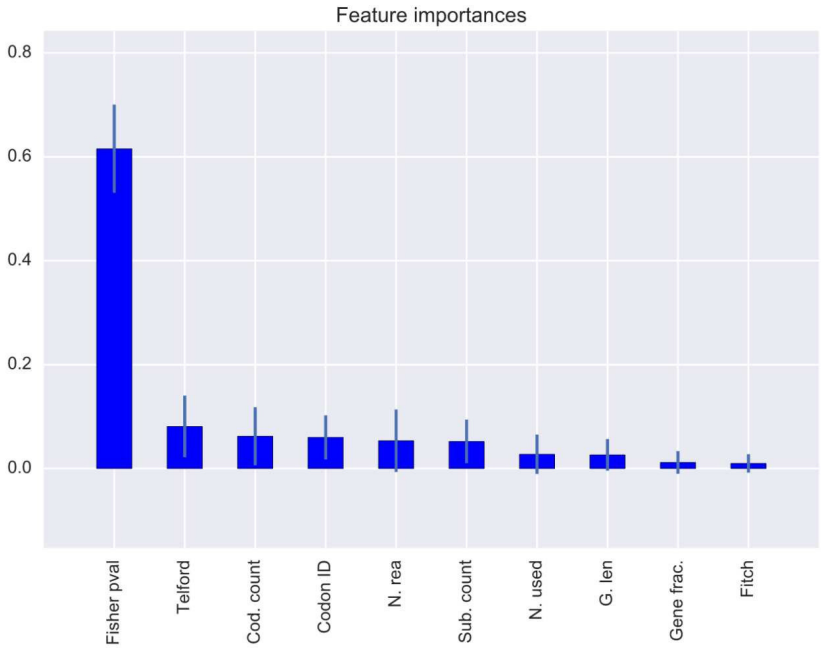


Figure S1: Importance of each feature used in the Random Forest. Feature importance was measured according to the Gini impurity index

A. **B.**

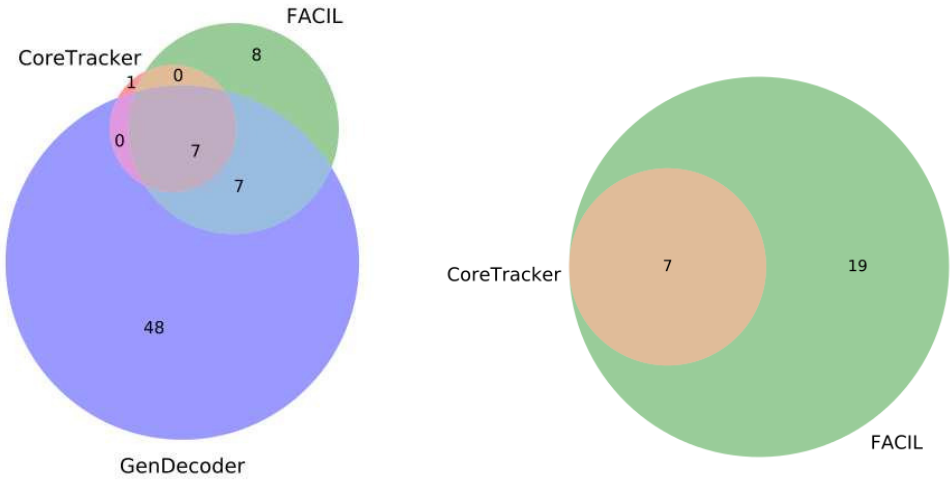


Figure S2: A. Comparison of CoreTracker, FACIL and GenDecoder predicted reassignment types on metazoan mitochondrial genome. CoreTracker and FACIL were run with default parameters. For GenDecoder, we selected *Metazoa* as the reference dataset and high site conservation (S < 1.0) for the entropy in order to increase selectivity. Seven codon reassignment types are shared by the three programs. These seven types correspond to known reassignment types in metazoan mitochondrial genomes. Only one unexpected reassignment type: AUA(Ile, Leu) was exclusively predicted by CoreTracker, whereas FACIL and GenDecoder predicted multiple unknown, but highly improbable reassignments. **B. Comparison of CoreTracker and FACIL predicted reassignment types on yeast mitochondrial genome.** Default parameters were used for both programs. All known reassignment types (subset of the shared set) were predicted in at least one genome by both programs. FACIL however also predicted multiple unknown, but highly improbable reassignments.

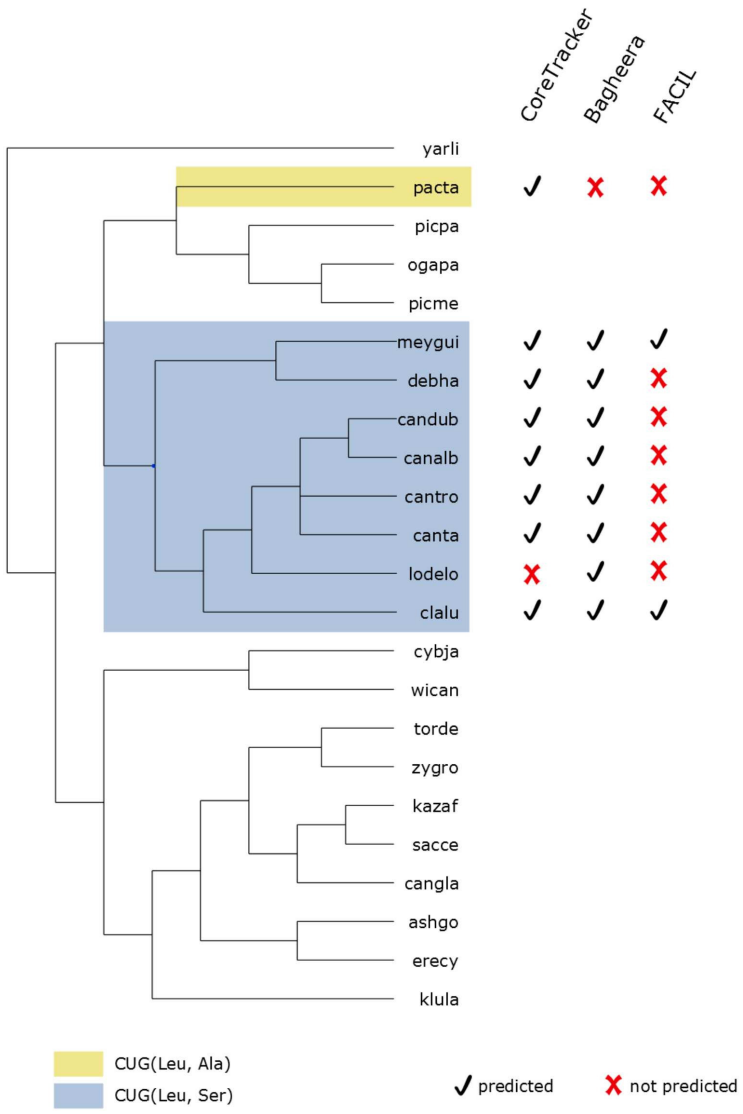


Figure S3: Comparison of CoreTracker, FACIL and Bagheera accuracy for the prediction of CUG reassignment in yeast nuclear genomes. Although CoreTracker was run without restricting predictions to a set of codons, only CUG(Leu, Ala) and CUG(Leu, Ser) were predicted.

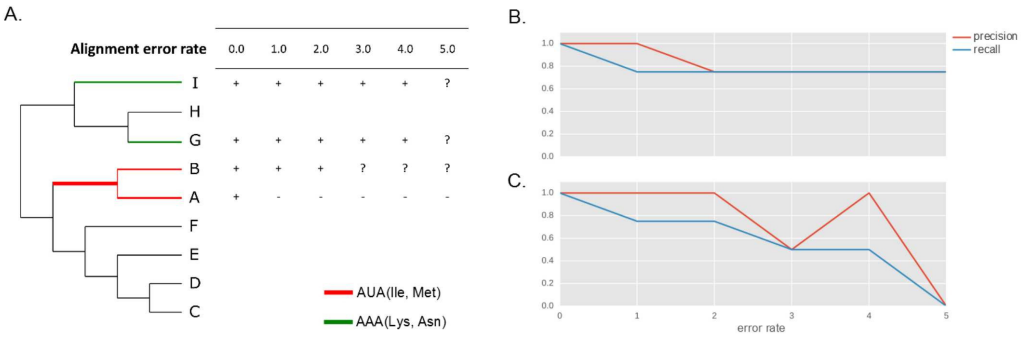


Figure S4: Effect of alignment quality on precision and recall. A. Predictions for the four genomes in which reassignments were introduced. A '+' indicates that the codon reassignment was predicted and validated, a '-' indicates that the reassignment was not predicted and '?' means the prediction has only passed the clade validation test. **B.** Precision and recall reported according to error rates for all predictions (including those in other taxa) that passes at least one validation test. **C.** Precision and recall considering predictions that were completely validated by CoreTracker.

4. Supplementary Tables

T**able S1**: List of true positives used to train the Random Forest

Genome	Codon	Xi	Xj	Statu	Number
Ciona	AGA	R	G	TP	
Ciona	AGG	R	G	TP	4
Halocynthia	AGA	R	G	TP	
Halocynthia	AGG	R	G	TP	
Limulus	AGG	R	K	TP	1
Caenorhabditis	ATA	I	M	TP	
Ciona	ATA	I	M	TP	
Daphnia	ATA	I	M	TP	
Drosophila	ATA	I	M	TP	
Epigonichthys	ATA	I	M	TP	
Homo	ATA	I	M	TP	12
Katharina	ATA	I	M	TP	
Limulus	ATA	I	M	TP	
Lumbricus	ATA	I	M	TP	
Myxine	ATA	I	M	TP	
Terebratulina	ATA	I	M	TP	
Trichinella	ATA	I	M	TP	
Fasciola	AAA	K	N	TP	
Paracentrotus	AAA	K	N	TP	4
Patiria	AAA	K	N	TP	
Schistosoma	AAA	K	N	TP	
Balanoglossus	AGA	R	S	TP	
Blanceolatum	AGA	R	S	TP	
Caenorhabditis	AGA	R	S	TP	
Caenorhabditis	AGG	R	S	TP	
Daphnia	AGA	R	S	TP	
Drosophila	AGA	R	S	TP	
Epigonichthys	AGA	R	S	TP	
Fasciola	AGG	R	S	TP	
Katharina	AGA	R	S	TP	
Katharina	AGG	R	S	TP	
Limulus	AGA	R	S	TP	21
Lumbricus	AGA	R	S	TP	
Paracentrotus	AGA	R	S	TP	
Paracentrotus	AGG	R	S	TP	
Patiria	AGA	R	S	TP	
Schistosoma	AGA	R	S	TP	
Schistosoma	AGG	R	S	TP	
Terebratulina	AGA	R	S	TP	
Terebratulina	AGG	R	S	TP	
Trichinella	AGA	R	S	TP	
Trichinella	AGG	R	S	TP	

Table S2: List of genomes used in this study

https://docs.google.com/spreadsheets/d/1EDcguFirZpoGwRs9CyLEMVCBw2ltmfNPq-WGCNYrqo4/edit?usp=sharing

Table S3: List of predicted codon reassignment by CoreTracker, FACIL and GenDecoder in metazoan and yeast mitochondrial genomes.

https://docs.google.com/spreadsheets/d/1pRiOH-qEHhiWCqvTmkbZVetFX_TirPCLUm2lnSd1YEA/edit?usp=sharing

Table S4: CoreTracker output in metazoan and yeast mitochondrial genomes

https://docs.google.com/spreadsheets/d/1RpozCQ2swnt70R94Zuu8XbNG_Sgl9BZ7NQUT5Te3X04/edit?usp=sharing

Table S5: Dataset used to evaluate CoreTracker on yeast nuclear genomes. All species were listed with their RefSeq or Genbank accession number. Known genetic code alterations are indicated and the length of the TBLASTN hit for the 18 proteins are reported for each genome. Gene absence was indicated by a length of 0.

https://docs.google.com/spreadsheets/d/1YSBxogo7WgBNtZ9MP3mq_571KyFvbjXf0ijqSsXEsV0/edit?usp=sharing

References

Abascal,F. *et al.* (2009) Genetic code prediction for metazoan mitochondria with GenDecoder. *Bioinforma. DNA Seq. Anal.*, 233–242.

Altschul,S.F. *et al.* (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.

Camacho,C. *et al.* (2009) BLAST+: architecture and applications. *BMC Bioinformatics*, **10**, 421.

Dutilh,B.E. *et al.* (2011) FACIL: fast and accurate genetic code inference and logo. *Bioinformatics*, **27**, 1929–1933.

Eddy,S.R. (2001) HMMER: Profile hidden Markov models for biological sequence analysis.

Edgar,R.C. (2004) MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, **5**, 113.

Fitch,W.M. (1971) Toward defining the course of evolution: minimum change for a specific tree topology. *Syst. Biol.*, **20**, 406–416.

Katoh,K. and Standley,D.M. (2013) MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Mol. Biol. Evol.*, **30**, 772–780.

Kawaguchi,Y. *et al.* (1989) The codon CUG is read as serine in an asporogenic yeast Candida cylindracea.

Mühlhausen,S. *et al.* (2016) A novel nuclear genetic code alteration in yeasts and the evolution of codon reassignment in eukaryotes. *Genome Res.*, **26**, 945–955.

Mühlhausen,S. and Kollmar,M. (2014) Predicting the fungal CUG codon translation with Bagheera. *BMC Genomics*, **15**, 411.
O'Leary,N.A. *et al.* (2016) Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Res.*, **44**, D733–D745.
Pedregosa,F. *et al.* (2011) Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, **12**, 2825–2830.
Riley,R. *et al.* (2016) Comparative genomics of biotechnologically important yeasts. *Proc. Natl. Acad. Sci. U. S. A.*, **113**, 9882–9887.
Santos,M.A. *et al.* (2011) The genetic code of the fungal CTG clade. *C. R. Biol.*, **334**, 607–611.
Telford,M.J. *et al.* (2000) Changes in mitochondrial genetic codes as phylogenetic characters: two examples from the flatworms. *Proc. Natl. Acad. Sci.*, **97**, 11359–11364.
Yule,G.U. (1925) A mathematical theory of evolution, based on the conclusions of Dr. JC Willis, FRS. *Philos. Trans. R. Soc. Lond. Ser. B Contain. Pap. Biol. Character*, **213**, 21–87.

Annexe B

Supplementary Information for "Rapid genetic code evolution in green algal mitochondrial genomes"

Il s'agit de l'annexe de l'article en préparation du chapitre 4 sur l'évolution du code génétique dans les génomes mitochondriaux d'algues vertes.

Supplementary Materials for "Rapid genetic code evolution in green algal mitochondrial genomes"

Emmanuel Noutahi[1], Virginie Calderon[2], Mathieu Blanchette[3], Nadia El-Mabrouk[1], and B Franz Lang[4]

[1]Département d'Informatique et de Recherche Opérationnelle, Université de Montréal
[2]Institut de Recherches Cliniques de Montréal
[3]School of Computer Science, McGill University
[4]Département de Biochimie, Centre Robert Cedergren, Université de Montréal

1 Supplementary information

Inference of genetic code evolution using cumulative information from protein sequence alignment and tRNA analyses

We have proposed a new framework for the prediction of genetic code alterations in the context of genome evolution. This framework takes into account the effect of changes in the genetic code by measuring the discrepancy between nucleotide sequence and their expected amino acid translation. It further searches for additional signals supporting inferred codon reassignments by considering changes in the tRNA repertoire that could have led to the observed alteration. The framework builds upon CoreTracker, a new algorithm for predicting codon reassignments that we have recently published and addresses two of its shortcoming, namely, the absence of tRNA evolution analysis and the restriction to sense-to-sense codon reassignments. Because the focus is on how changes in the tDNA sequences affect protein decoding, other potential modifications in the translation apparatus such as post-transcriptional tRNA modifications and mutations in aminoacyl-tRNA synthetase are not investigated. However, we believe that the framework is easily extensible to reassignments involving this particular cases, in particular when tDNA sequence analysis only failed to support codon reassignments predicted with high probabilities. The framework comprises of four independent yet complementary modules that we illustrate here on the UGA(Stop $\rightarrow$ Trp) codon reassignment in Pedinomonas minor mitochondrial genome (see Figure S1).

In the first module, potential codon reassignments are predicted by analyzing protein sequence alignment to detect unexpected codon usage at conserved amino acid positions. For sense-to-sense codon reassignment prediction, we use CoreTracker, which exhaustively explore reassignments across a group of related genomes, then uses statistical evaluations and a random forest classifier to predict the most likely ones. The method was proven to be more precise and robust than its competitors, thanks to its additional validation steps ensuring the correctness of predicted reassignments. CoreTracker concentrates on sense-to-sense reassignments because other reassignments involving stop codons are often uncovered during genome annotation due to the prediction of proteins with either premature termination or multiple additional C-terminal domains. Consequently, for sense-to-stop and stop-to-sense codon reassignments, we rely on current advances in gene annotation pipelines that allow detection of correct reading frames and gene structures. In practice, it is better to fine-tune this step of the framework for high sensitivity, even over precision, as the next steps will essentially filter predictions to only retain the most relevant. An illustration of the result of this step is shown on the Nad1 protein alignment in Figure S1A, where we can see that UGA stop codons are used by *Pedinomonas minor* at several tryptophan-conserved positions, suggesting their possible decoding as tryptophan.

In the second module, we search for the presence of a tRNA able to decode the reassigned codon and investigate its evolutionary story. To do this, we first need to identify the complete tRNA repertoire of each considered genome. Owing to the growth of genomics data, recent advances in computational analyses of RNA sequences have led to developments of reliable methods (tRNAscan-SE, ARAGORN, ERPIN, Infernal, RNAFinder) for finding tRNA genes. Results of these methods can further be combined to improve accuracy [1]. From the complete set of tRNAs in each genome, we can look for tRNAs with anticodon capable of decoding predicted reassigned codons. Indeed, most genetic code alterations are linked to tRNA identity switch. This switch in decoding can occur through tRNA remolding, by mutations in the acceptor stem or anticodon loop. A direct consequence of the latter case is an unexpected clustering inside the tRNA phylogeny, due to the mutant tRNA still grouping with its evolutionary-related tRNA family. As such, analysis of the tRNA phylogeny will not only reveal information about potential genetic code alterations but also help in determining the evolutionary history that led to these alterations. Although signal from phylogeny constructed with tDNA sequences is often low, in particular regarding lower branches resolution, the information about groupings is often sufficient when complemented with other clues. In the context of the UGA(Stop $\rightarrow$ Trp) codon reassignment in *P. minor*, Figure S1B shows that the corresponding mtDNA-encoded tRNA(UCA) cluster with Cys-tRNA(GCA) and Tyr-tRNA(GUA), suggesting a common ancestry. As tRNA(UCA) is not apparently related to the Trp-tRNA family, additional analyses to confirm its decoding of UGA codons as tryptophan are performed in the subsequent modules.

The third step of the framework consists of accurately determining the identity of tRNAs of interest. One way to perform such a task is by first inferring the secondary structure of the tRNA then search for major identity determinants and anti-determinants responsible for its specificity. Indeed, although the complete set of tRNA identity (anti-)determinants

is mostly unknown, major identity elements of some tRNAs are well characterized (Giegé et al. 1998; Salinas-Giegé et al. 2015), and there has been a growing effort to uncover the set of rules regulating tRNA identities. As shown in Figure S1C, application of this module to the identified mtDNA-encoded tRNA(UCA) reveals the presence of multiple elements, namely the 'G' base at the discriminator position, and the pairs A1:U71, G2:C70 which are similar to the known Trp-tRNA identity determinants in eubacteria [5, 4]. Furthermore, the UCA anticodon is highly similar to the canonic CCA anticodon of Trp-tRNA, suggesting that tryptophanyl-tRNA synthetase could be able to recognize tRNA(UCA), resulting in the decoding of UGA stop codons as tryptophan.

Finally, in the last module, we use information from genome evolution, conjointly with data obtained from previous modules, to infer the evolutionary scenario of the predicted codon reassignments. In the case of the reassignment of stop codon UGA in *P. minor*'s mtDNA, analysis of tRNA gene content, reveals the absence of several tRNAs including the standard Trp-tRNA(CCA), while gene order analysis shows proximity between the Cys-tRNA(GCA), tRNA(UCA) and the two Tyr-tRNA(GUA), supporting a recent tandem duplication (see Figure S1D). This result is coherent with the observed grouping inside the tRNA phylogeny and supports the emergence of tRNA(UCA) through duplication of Cys-tRNA(GCA) followed by sequence mutation. Because in most Chlorophyta UGA codons are missing, the codon could have been unassigned before the emergence of a new tRNA(UCA) originating from an alloacceptor remolding of a duplicated Cys-tRNA(GCA) gene. The new tRNA(UCA) was then able to decode UGA codons as tryptophan when they reappear in the genome. This process could have been further accelerated by the loss of several tRNAs which would have directly affected codon usage in the genome.

In summary, by reconciling the result of all four modules, not only are we able to infer changes in the genetic code, but we can also deduce the underlying evolutionary processes leading to the modification.

2 Supplemental Tables

Table S1: List of green plant mitochondrial genomes used in this study and their NCBI accession number

Species	**Accession**
Arabidopsis thaliana	NC_001284.2
Beta vulgaris subsp. vulgaris	NC_002511.2
Bracteacoccus aerius	NC_024755.1
Bracteacoccus minor	NC_024756.1
Brassica rapa subsp. oleifera	NC_016125.1
Carica papaya	NC_012116.1
Chaetosphaeridium globosum	NC_004118.1
Chara vulgaris	NC_005255.1
Chlamydomonas reinhardtii	NC_001638.1
Chlorokybus atmophyticus	NC_009630.1
Chlorotetraedron incus	NC_024757.1
Chromochloris zofingiensis	NC_024758.1
Dunaliella salina	NC_012930.1
Glycine max	NC_020455.1
Kirchneriella aperta strain SAG 2004	NC_024759.1
Lotus japonicus	NC_016743.2
Malus domestica	NC_018554.1
Marchantia polymorpha	NC_001660.1
Mesostigma viride	NC_008240.1
Mychonastes homosphaera	NC_024760.1
Neochloris aquatica	NC_024761.1
Nephroselmis olivacea	NC_008239.1
Oryza sativa Indica Group	NC_007886.1
Ourococcus multisporus	NC_024762.1
Phoenix dactylifera	NC_016740.1
Physcomitrella patens	NC_007945.1
Polytoma uvella	NC_026572.1
Prototheca wickerhamii	NC_001613.1
Pseudendoclonium akinetum	NC_005926.1
Pseudomuriella schumacherensis	NC_024763.1
Pycnococcus provasolii	NC_013935.1
Sorghum bicolor	NC_008360.1
Tetradesmus obliquus	NC_002254.1
Vitis vinifera	NC_012119.1
Zea mays subsp. mays	NC_007982.1
Zea mays subsp. parviglumis	NC_008332.1

Table S2: TFAM classification of 13 chlorophytes mtDNA-encoded tRNAs with dubious identities into seven tRNA families. A higher score indicates more support for the class. For each tRNA, the class with the highest score is shown in bold text.

	Covariance model						
	Ala	Met	Leu	Arg	Trp	Cys	Tyr
tRNA(CCT) *B. minor*	-12.88	-5.31	-41.03	-21.07	**6.83**	-20.77	-40.33
tRNA(CCT) *B. aerius*	**-2.58**	-5.44	-31.91	-24.95	-4.53	-6.46	-33.58
tRNA(CCT) *N. aquatica*	**4.89**	-6.47	-50.15	-38.47	-15.00	-21.14	-64.31
tRNA(CCT) *C. incus*	**18.07**	-0.28	-68.57	-32.17	-11.33	-26.87	-62.88
tRNA(CCT) *T. obliquus*	-32.59	**-5.14**	-61.90	-12.80	-10.69	-24.38	-58.17
tRNA(CCT) *C. zofingiensis*	-3.73	**17.64**	-67.63	-31.00	0.75	-14.94	-69.92
tRNA(CCT) *K. aperta*	-33.91	**-3.36**	-37.12	-22.84	-4.41	-11.84	-54.80
tRNA(CCG) *C. zofingiensis*	-34.28	-33.63	**41.01**	-26.42	-40.58	-43.15	-24.03
tRNA(CUA) *N. aquatica*	**11.59**	-9.71	-55.32	-38.30	-19.21	-29.13	-57.82
tRNA(CUA) *T. obliquus*	-46.37	-36.01	**38.90**	-45.13	-33.60	-44.40	-21.39
tRNA(CCA) *P. provasolii*	-34.39	-14.12	-34.41	-35.92	**-8.18**	-14.96	-71.62
tRNA(UCA) *P. minor*	-15.65	-8.61	-52.13	-23.53	-13.17	**-0.55**	-46.00
tRNA(UCU) *T. obliquus*	-26.46	**-7.96**	-62.08	-20.40	-11.74	-28.80	-51.60

3 Supplemental Figures

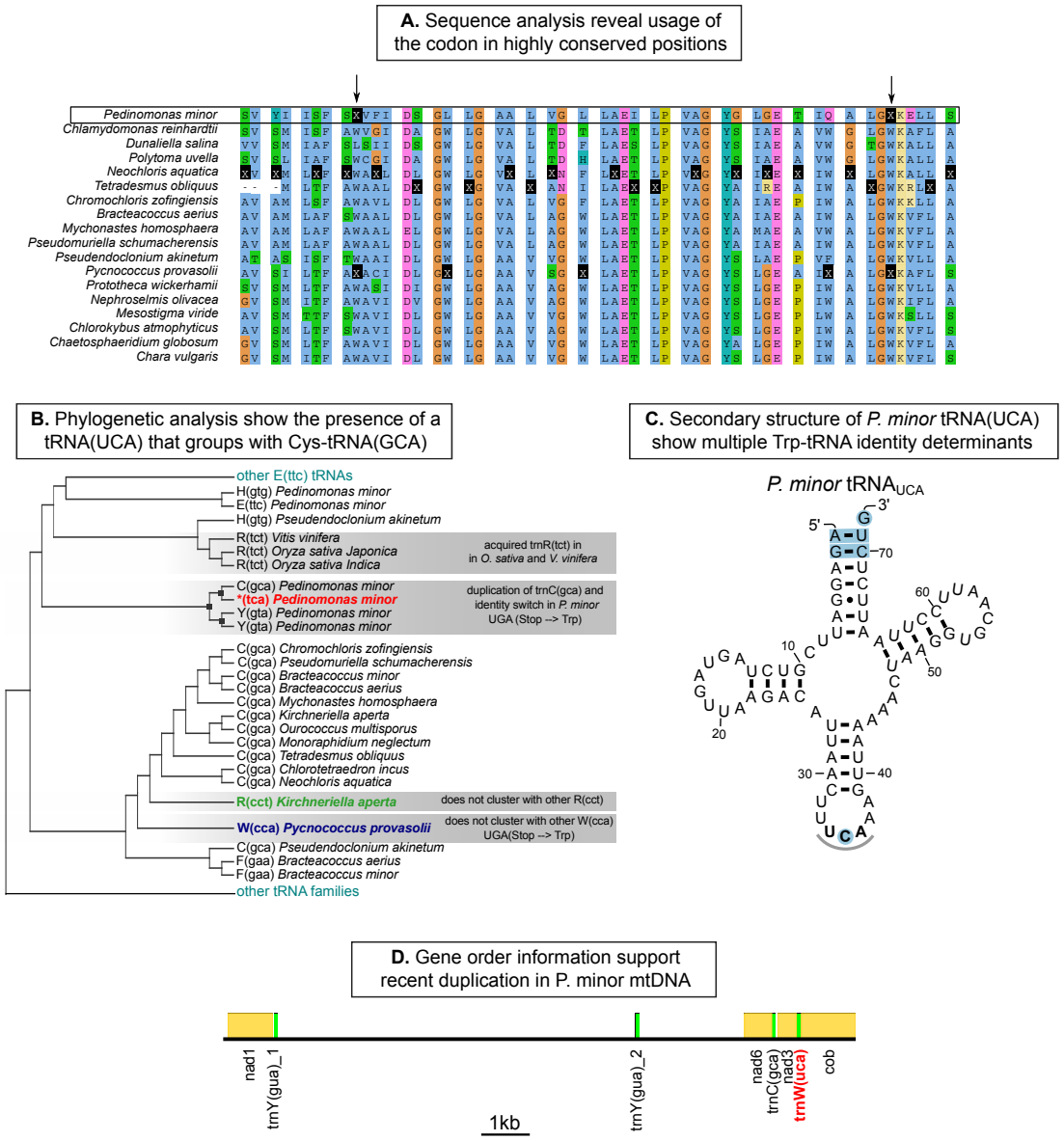


Figure S1: Overview of the workflow for inferring genetic code alteration illustrated on UGA(Stop→ Trp) in *Pedinomonas minor*. **(A)** Step 1 of the framework: Sequence analysis. In this module, codon reassignments are predicted from protein sequence alignment. Here, the nad1 protein alignment revealed the use of UGA stop codons in tryptophan-conserved positions by *P. minor* and *P. provasolii*, while UAG codons are respectively used in alanine and leucine conserved positions by *N. aquatica* and *T. obliquus*. **(B)** Step 2: Analysis of tRNA phylogeny. tRNA phylogenetic analysis revealed a shared common ancestor between Cys-tRNA(GCA), Tyr-tRNA(GUA) and tRNA(UCA), suggesting a common origin. **(C)** Step 3: Determination of tRNA identity. Multiple Trp-tRNA identity determinants are found in the secondary structure of tRNA(UCA) supporting its decoding of UGA codons as tryptophan. **(D)** Step 4: Analysis of tRNA gene order. The tRNA gene order in *P.minor*'s mtDNA supports the hypothesis of tRNA(UCA) originating from a recent duplication of Cys-tRNA(GCA).

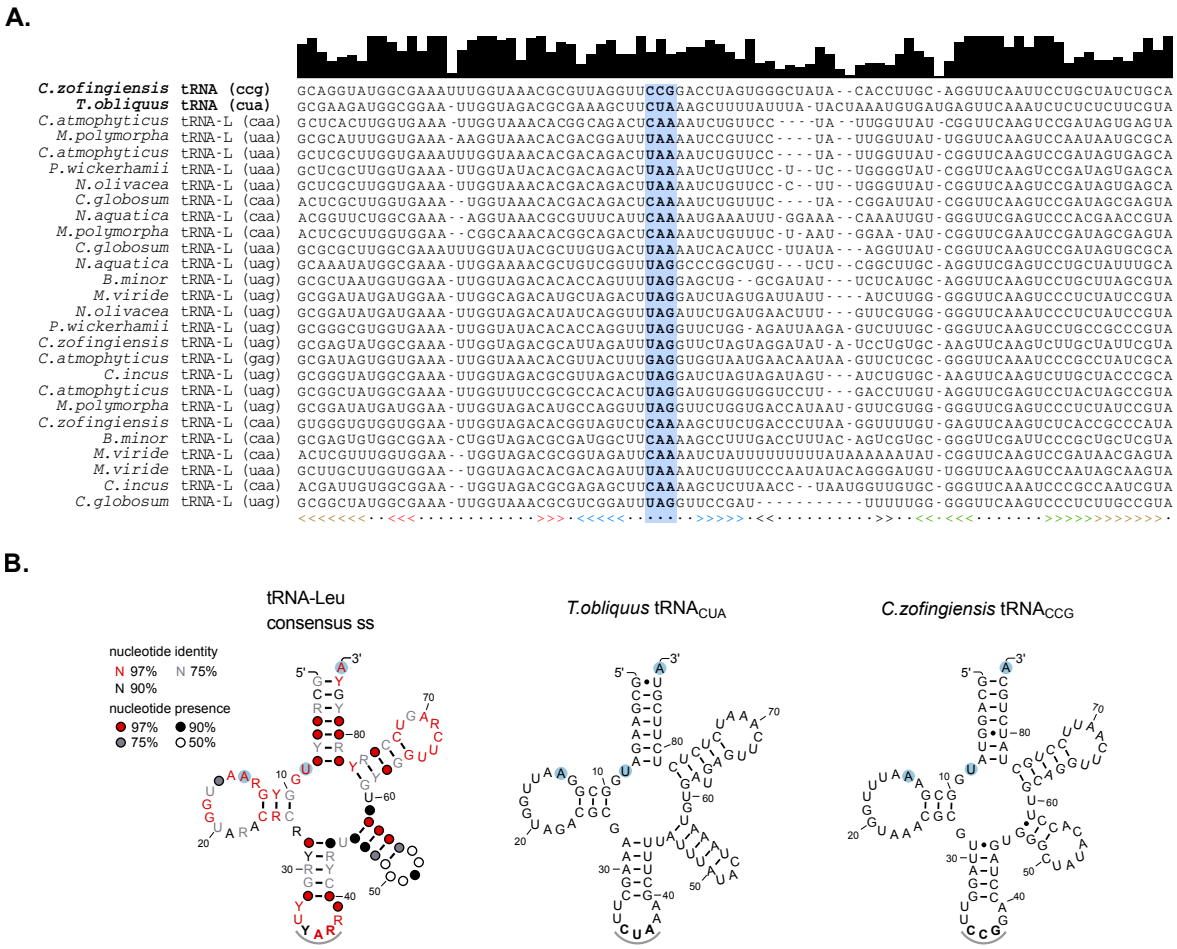


Figure S2: Primary and secondary structures of green plant mtDNA-encoded leucine tRNAs and the two divergent tRNA(CUA) and tRNA(CCG) of *T. obliquus* and *C. zofingiensis*. **(A)** Multiple sequence alignment of selected green plant mitochondrial Leu-tRNA with UAR or CAA anticodons. *C. zofingiensis* tRNA(CCG) and *T. obliquus* tRNA(CUA) are included in the alignment and show high sequence similarity with the Leu-tRNA group. Brackets are used to indicate the four standard helical regions in tRNAs. The anticodon position is highlighted in blue. **(B)** Secondary structure of the consensus green plant mtDNA-encoded Leu-tRNA (left), *T. obliquus* tRNA(CUA) (middle) and *C. zofingiensis* tRNA(CCG) (right). Minimum nucleotide identity and presence conservation are displayed according to the color-coding in the legend, Leu-tRNA identity determinants are highlighted in blue, and the anticodon nucleotides are shown in bold, with a grey outline. The secondary structures of the two divergent tRNAs display several characteristics of Leu-tRNA. Because the Leu-tRNA synthetase use no anticodon nucleotides for its recognition of Leu-tRNA[2] and the major Leu-tRNA identity determinants were preserved, tRNA(CUA) in *Tetradesmus* and tRNA(CCG) in *C. zofingiensis* are respectively able to decode UAG and CGG codons as Leu.

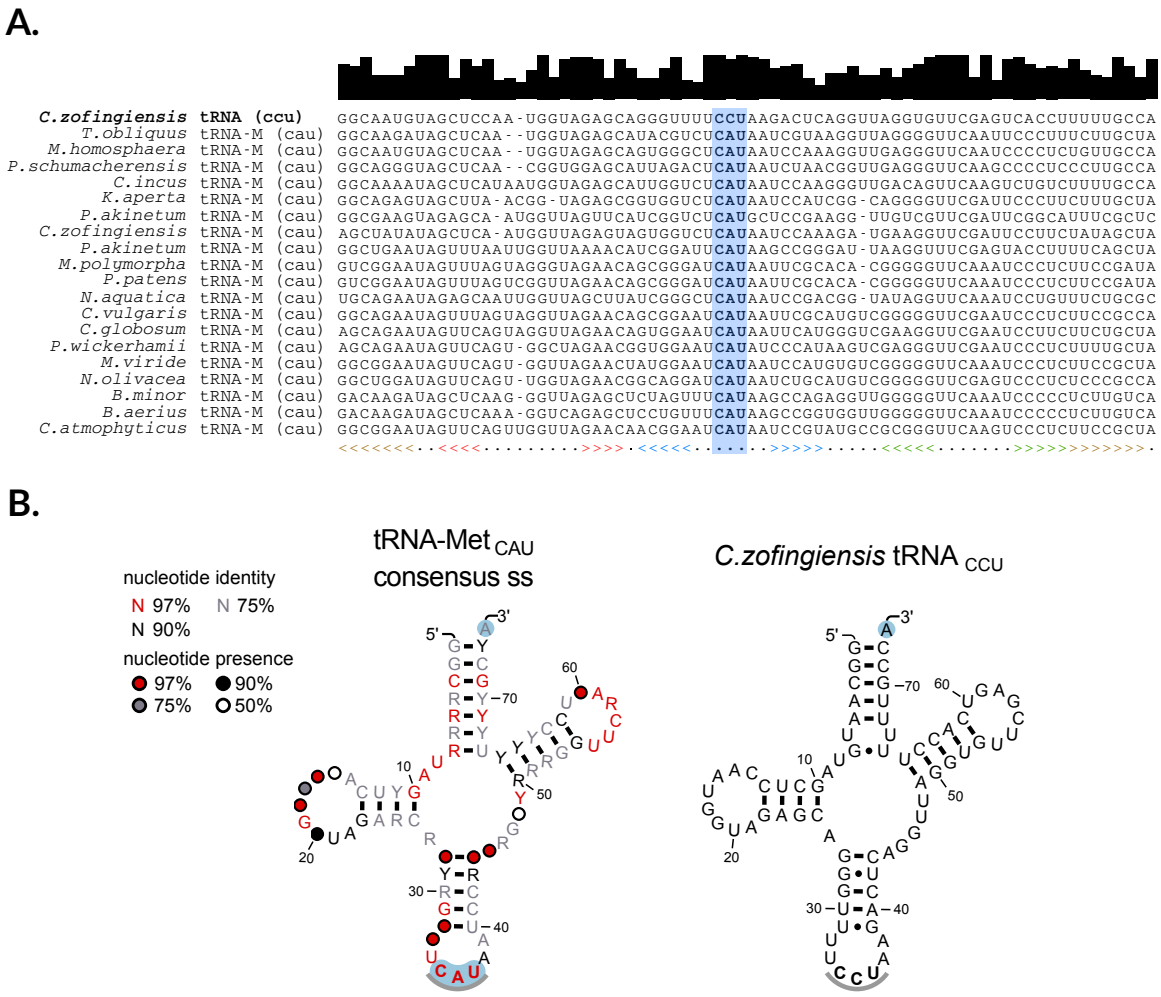


Figure S3: Primary and secondary structures of green plant mtDNA-encoded methionine tRNAs and the divergent *C. zofingiensis* tRNA(CCU). **(A)** Multiple sequence alignment of selected green plant mitochondrial Met-tRNA(CAU). *C. zofingiensis* tRNA(CCU) is included in the alignment and share several conserved positions with the Met-tRNA group. Brackets are used to indicate the four standard helical regions in tRNAs and the anticodon position is highlighted in blue. (**B**) Secondary structure of the consensus green plant mtDNA-encoded Met-tRNA(CAU) (left) and *C. zofingiensis* tRNA(CCU) (right). The legend follows the same scheme used in Figure 2 of the main text, with Met-tRNA identity determinants highlighted in blue. In contrast with some other tRNA families such as Ala-tRNA, Met-tRNA identity determinants are not well defined, and usually involve specific three-dimensional features[8] or even post-transcriptional modifications[9]. It is however accepted that the anticodon nucleotides and the discriminator base "A73" contribute to the recognition of the tRNA by the MetRS[4, 8]. Both determinants were found in the tRNA(CCU) except for base A35 of the anticodon being substituted by C35. Previous studies have however shown that such a substituion in the anticodon has limited effect on Met-tRNA functionality loss[3, 7].

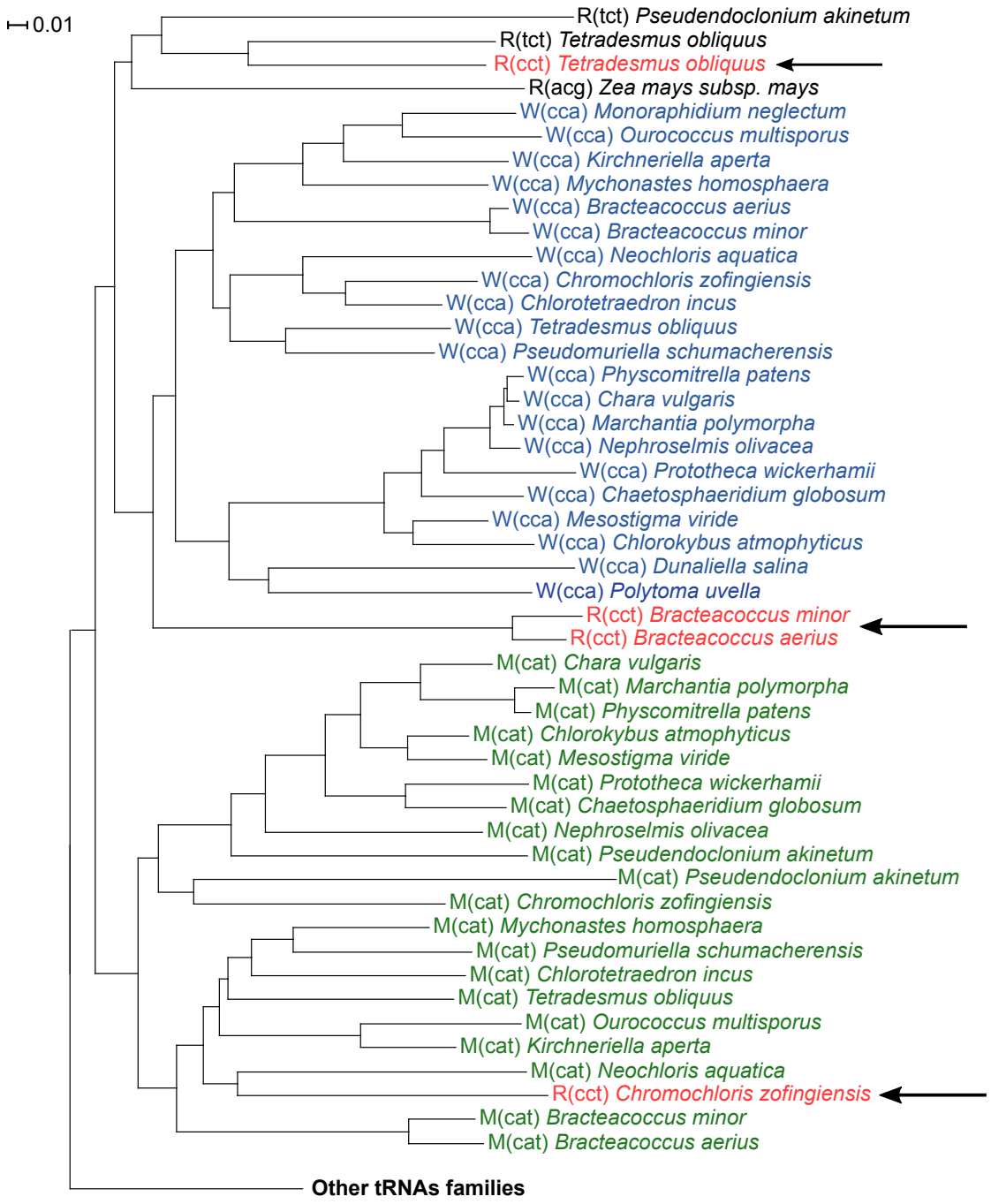


Figure S4: Phylogeny of green plant mitochondrial tRNAs with FastTree. The tree was restricted to the subtree showing Met-tRNA(CAU) (in green) and Trp-tRNA(CCA) (in blue). The phylogenetic analysis revealed unexpected positions of the tRNAs(CCU) from *B. minor*, *B. aerius*, *C. zofingiensis* and *T. obliquus* (in red).

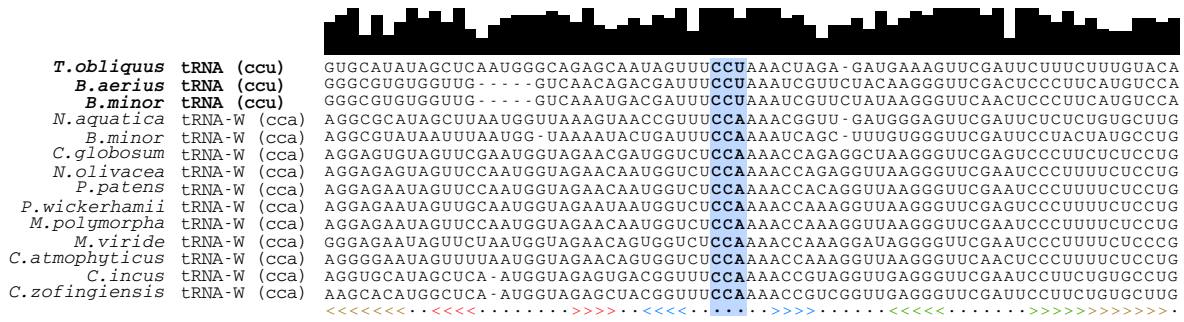


Figure S5: Sequence alignment between selected mtDNA-encoded Trp-tRNA(CCA) genes and tRNA(CCU) from *Bracteaccocus* and *Tetradesmus*. The alignment show high sequence similarity between the tRNA(CCU) and the Trp-tRNA(CCA) group. The anticodon position is highlighted in blue and brackets are used to indicate the four standard helical regions in tRNAs.

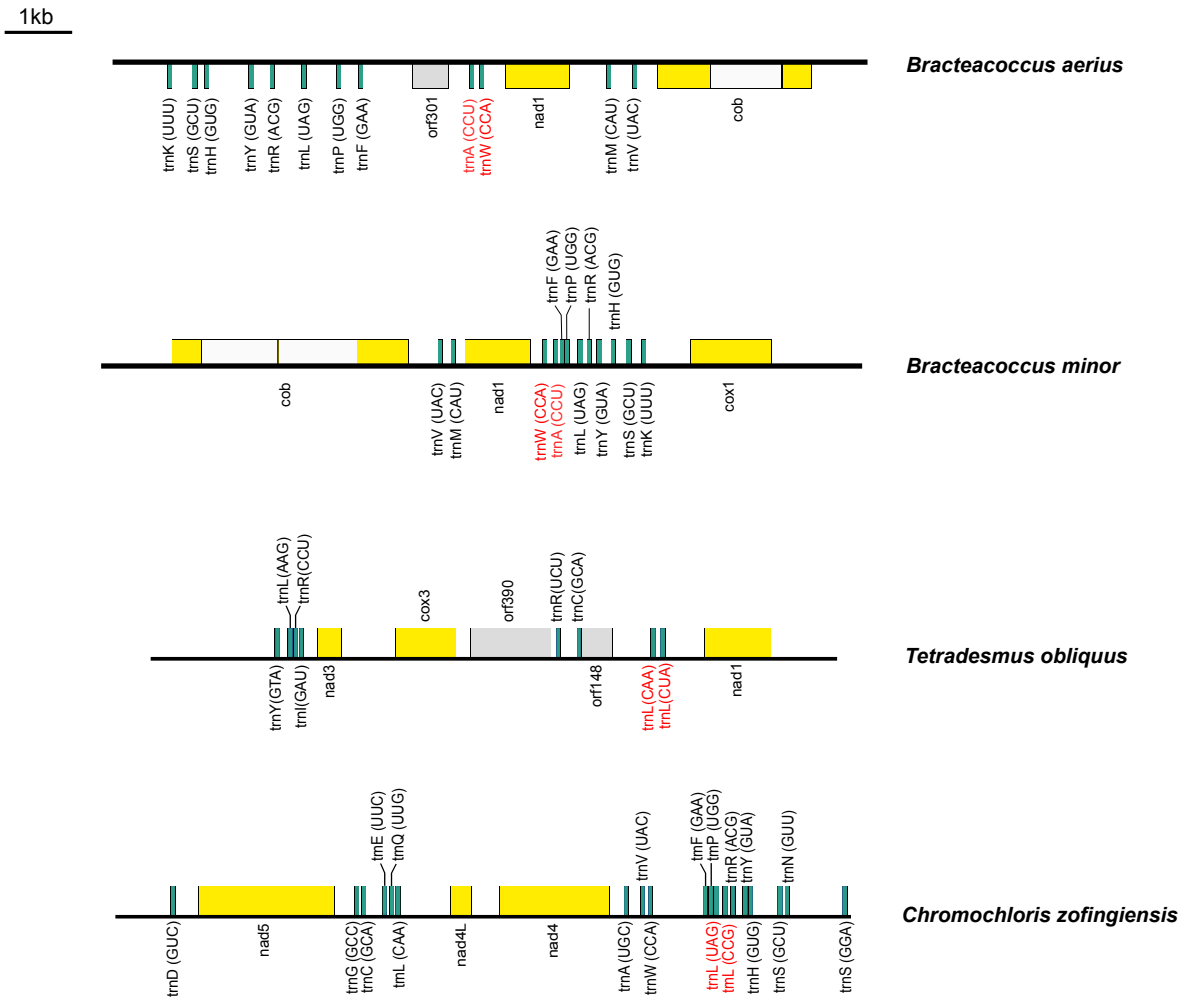


Figure S6: Gene order in *B. minor*, *B. aerius*, *T. obliquus* and *C. zofingiensis* mitochondrial genome. The genome maps were drawn with OGDraw [6] and were then modified using Inkscape to only show the part containing mutant tRNAs (highlighted in red) of interest. Yellow rectangles represent protein coding genes with introns indicated in white, gray rectangles correspond to ORFs, and green rectangles to tRNAs. In the two *Bracteacoccus* genomes, Ala-tRNA(CCU) is adjacent to Trp-tRNA(CCA). In *Tetradesmus*, the predicted Leu-tRNA(CUA) is found next to Leu-tRNA(CAA), suggesting a recent tandem duplication. The same is observed in *Chromochloris*, with the Leu-tRNA(CCG) and Leu-tRNA(UAG) adjacent to each other.

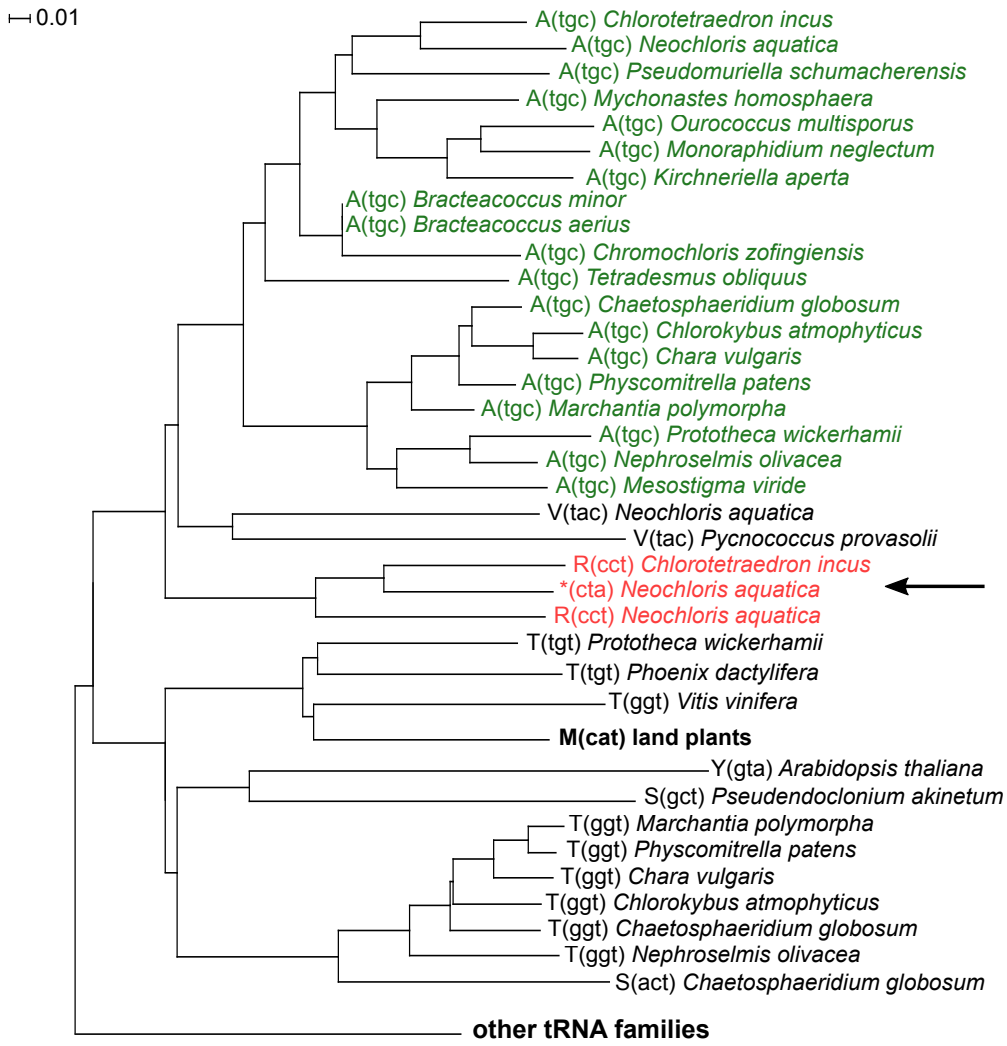


Figure S7: Phylogeny of green plant mitochondrial tRNAs with FastTree. The tree was restricted to the subtree showing Ala-tRNA(UGC) (in green) and revealed that Neochloridaceae mtDNA tRNA(CCU) and tRNA(CUA) (in red) cluster with the Ala-tRNA(UGC) group.

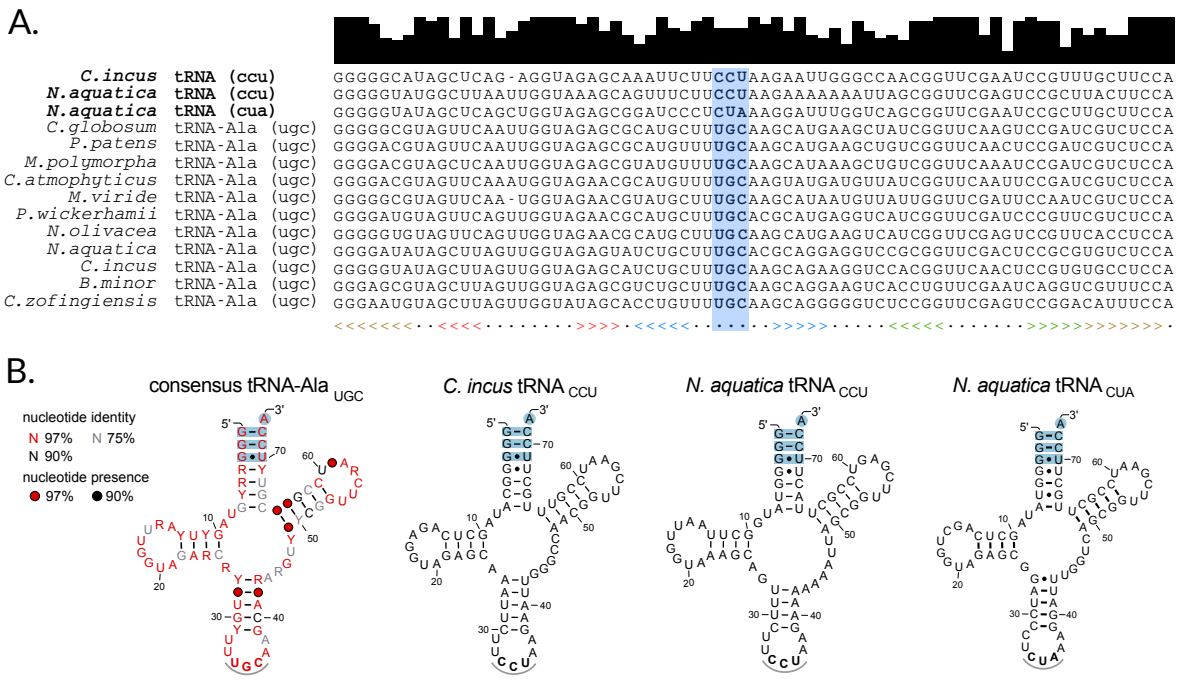


Figure S8: Primary and secondary structures comparison between green plant mtDNA-encoded Ala-tRNA and the tRNA(CCU) and tRNA(CCA) predicted in Neochloridaceae. **(A)** Sequence alignment between selected mtDNA-encoded Ala-tRNA(UGC) genes and the tRNA(CCU) and tRNA(CCA) from *N. aquatica* and *C. incus*. The alignment shows high sequence similarity between the tRNAs. The anticodon position is highlighted in blue, and brackets are used to indicate the four standard helical regions in tRNAs. **(B)** Secondary structure of the consensus green plant mtDNA-encoded Ala-tRNA(UGC) (left) followed by the Ala-tRNA(CCU) predicted in *C. incus* and the Ala-tRNA(CCU) and Ala-tRNA(CUA) predicted in *N. aquatica*. The legend follows the same scheme used in Figure 2 of the main text, with Ala-tRNA identity determinants highlighted in blue. Ala-tRNA identity elements are well known and consist of the invariant G3:U70 wobble pair, part of conserved G^1GGC^4, and the A73 at the discriminator position. These determinants were all found in the mutant tRNAs. Furthermore, each of mutant tRNA show extremely high similarity with the consensus Ala-tRNA(UGC), with most differences located in the central loop.

Figure S9: Phylogeny of green plant mitochondrial tRNAs with FastTree. Only the section of the tree containing Leu-tRNAs is shown. Both *C. zofingiensis* tRNA(CCG) and *T. obliquus* tRNA(CUA) (in red) are found inside the Leu-tRNA group pointing out their recent origin from a Leu-tRNA duplication.

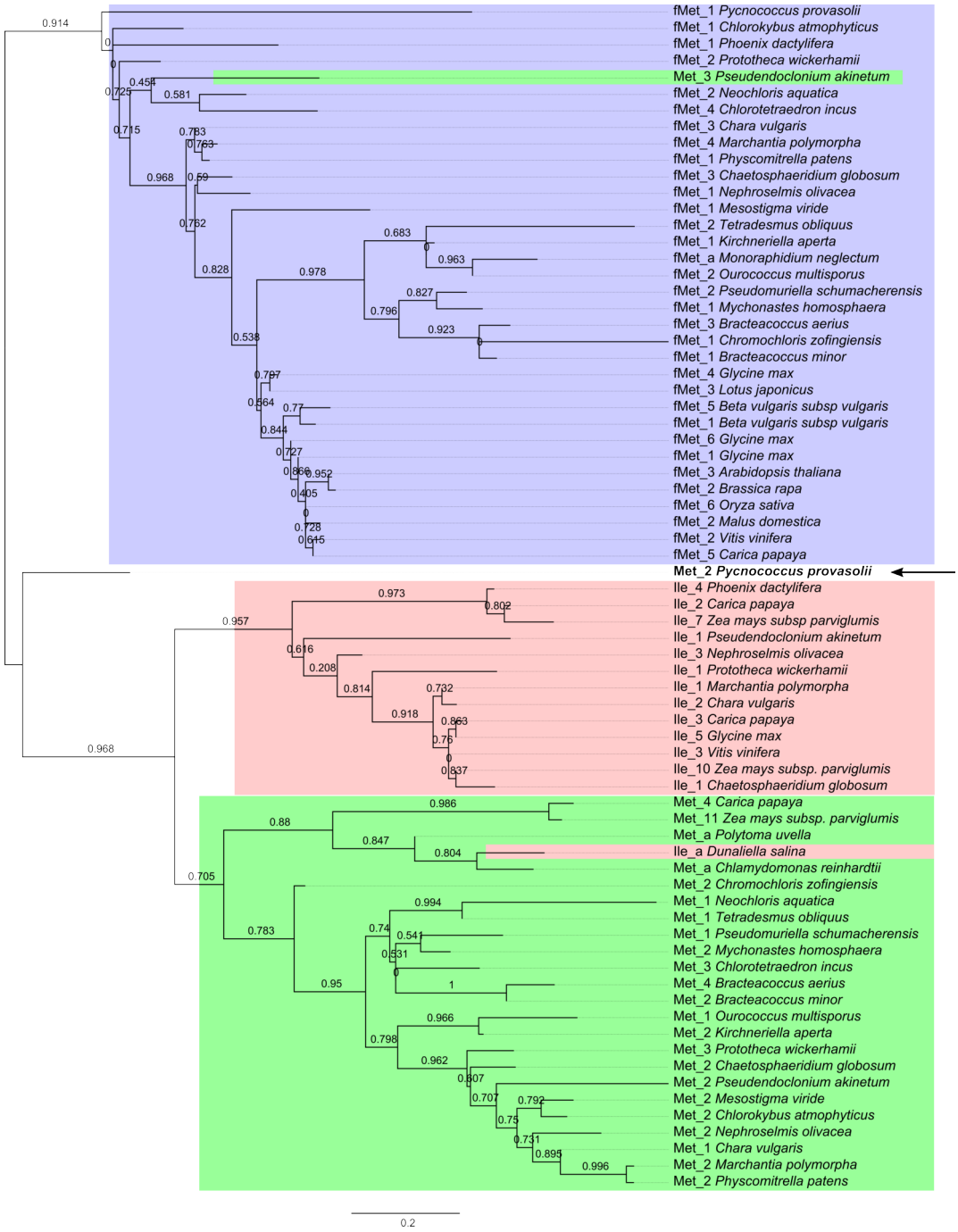


Figure S10: Phylogenetic tree computed with FastTree of the alignment of non-redundant green plant mitochondrial tRNAs with anticodon CAU. Branches are colored according to the classification returned by TFAM. The blue background indicates the Met-tRNA initiator, the green background correspond to Met-tRNA elongator, whereas the red background corresponds to Ile-tRNA(CAU). The arrow indicates the position of the divergent Met-tRNA(CAU) in *P. provasolii*. Note that the tree is not rooted, but is displayed in a way that group separation is obvious. The scale bar denotes substitutions per site.

References

[1] David H Ardell. Computational analysis of tRNA identity. *FEBS letters*, 584(2):325–333, 2010.

[2] Haruichi Asahara, Hyouta Himeno, Koji Tamura, Tsunemi Hasegawa, Kimitsuna Watanabe, and Mikio Shimizu. Recognition nucleotides of Escherichia coli tRNA-Leu and its elements facilitating discrimination from tRNA-Ser and tRNA-Tyr. *Journal of molecular biology*, 231(2):219–229, 1993.

[3] Harold J Drabkin, Melanie Estrella, and Uttam L Rajbhandary. Initiator-elongator discrimination in vertebrate tRnas for protein synthesis. *Molecular and cellular biology*, 18(3):1459–1466, 1998.

[4] Richard Giegé, Marie Sissler, and Catherine Florentz. Universal rules and idiosyncratic features in tRNA identity. *Nucleic acids research*, 26(22):5017–5035, 1998.

[5] Hyouta Himeno, Tsunemi Hasegawa, Haruichi Asahara, Koji Tamura, and Mikio Shimizu. Identity determinants of E. coli tryptophan tRNA. *Nucleic acids research*, 19(23):6379–6382, 1991.

[6] Marc Lohse, Oliver Drechsel, and Ralph Bock. Organellargenomedraw (ogdraw): a tool for the easy generation of high-quality custom graphical maps of plastid and mitochondrial genomes. *Current genetics*, 52(5-6):267–274, 2007.

[7] LaDonne H Schulman and Heike Pelka. Anticodon loop size and sequence requirements for recognition of formylmethionine tRna by methionyl-tRna synthetase. *Proceedings of the National Academy of Sciences*, 80(22):6755–6759, 1983.

[8] B Senger and F Fasiolo. Yeast tRnaMet recognition by methionyl-tRna synthetase requires determinants from the primary, secondary and tertiary structure: a review. *Biochimie*, 78(7):597–604, 1996.

[9] Chie Takemoto, Linda L Spremulli, Lisa A Benkowski, Takuya Ueda, Takashi Yokogawa, and Kimitsuna Watanabe. Unconventional decoding of the Aua codon as methionine by mitochondrial tRna Met with the anticodon f 5 Cau as revealed with a mitochondrial in vitro translation system. *Nucleic acids research*, 37(5):1616–1627, 2009.

Annexe C

Supplementary Materials for "Efficient gene tree correction guided by genome evolution"

Il s'agit de l'annexe de l'article présenté dans le chapitre 5 et publié dans *PLoS ONE*.

Efficient gene tree correction guided by genome evolution

Supplementary Data

Emmanuel Noutahi*[1], Magali Semeria*[2],
Manuel Lafond[1], Jonathan Seguin[1],
Bastien Boussau[2], Laurent Gueguen[2],
Nadia El-Mabrouk[1,4], Eric Tannier[2,3,4]

1 - Département d'Informatique (DIRO), Université de Montréal, H3C3J7, Canada;

2 - LBBE, UMR CNRS 5558, Université de Lyon 1, F-69622 Villeurbanne, France;

3 - INRIA Grenoble Rhône-Alpes, F-38334 Montbonnot, France

4 - Corresponding authors mabrouk@iro.umontreal.ca, Eric.Tannier@inria.fr

* - equal contribution

Unsupported branches

Figure A shows the distribution of unsupported branches in PhyML trees.

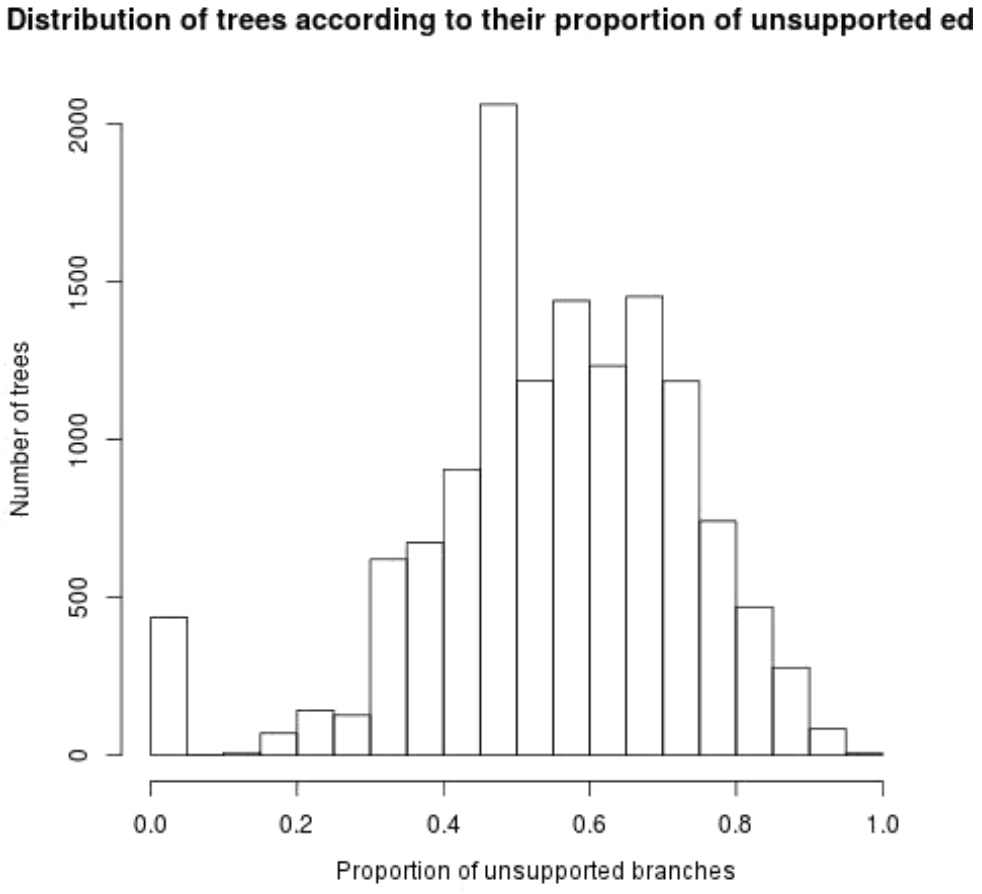


Figure A: Distribution of the number of unsupported (aLRT < 0.95) branches in PhyML trees on the 20529 Ensembl 73 gene families.

RefineTree interface

RefineTree is released with a web interface involving ProfileNJ and ParalogyCorrector. The user is requested to provide a species tree, or alternatively point to the Ensembl species tree, and a gene tree or an Ensembl gene tree ID. The distance matrix used in ProfileNJ can be provided or computed from the nucleotide or amino acids sequences input by the user. Such sequences are also required for ranking solutions by their likelihood using PhyML. A graphical representation of the corrected tree is displayed using the ETE2 Python Framework [1]. An example is given in Figure B.

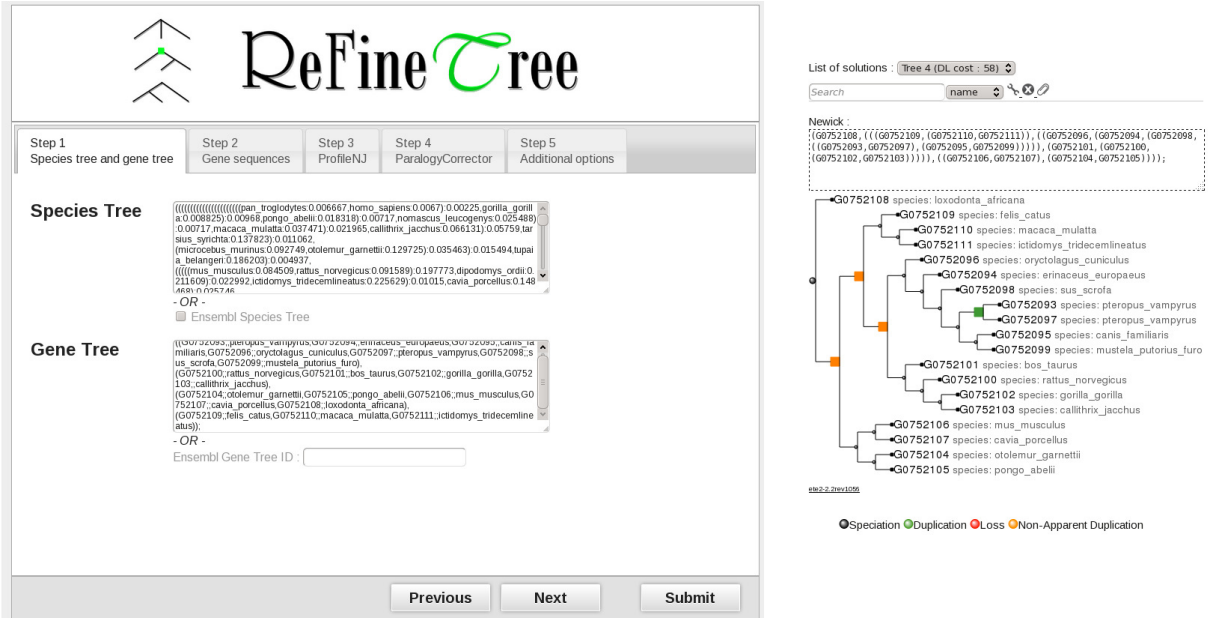


Figure B: RefineTree web interface. The main screen on the left, and an example of an output on the right with speciation and duplication nodes labelled differently.

Simulation

We provide the results of several analyses that show the behavior of ProfileNJ, compared to TreeFix, on simulations.

Topology Accuracy: Accuracy of tree topology is evaluated according to the Robinson-Foulds (RF) distance, i.e. the number of symmetric differences between the clade-sets, of the output tree and the true tree (obtained from simulations).

This difference between a reconciliation-based method versus a pure sequence-based method is probably related to the evolutionary model chosen for simulations, which appears to fit the parsimony criterion for duplications and losses, and thus favour a method based on the reconciliation cost. Indeed, among the simulated data-sets with true estimated DL rates ($1r_D$ - $1r_L$), TreeFix and ProfileNJ are able to recover about 93% of correct topologies. Moreover, a decrease in topology accuracy of both programs for increasing number of evolutionary events is observed (Figure C), TreeFix performing slightly better than ProfileNJ. This is expected as the most parsimonious reconciliation will always lead to the fewest number of events required to explain the data. For small trees with high DL events, ProfileNJ might not therefore be able to recover the correct tree. This artefact is not observed on simulated data-sets with true estimated DL rate.

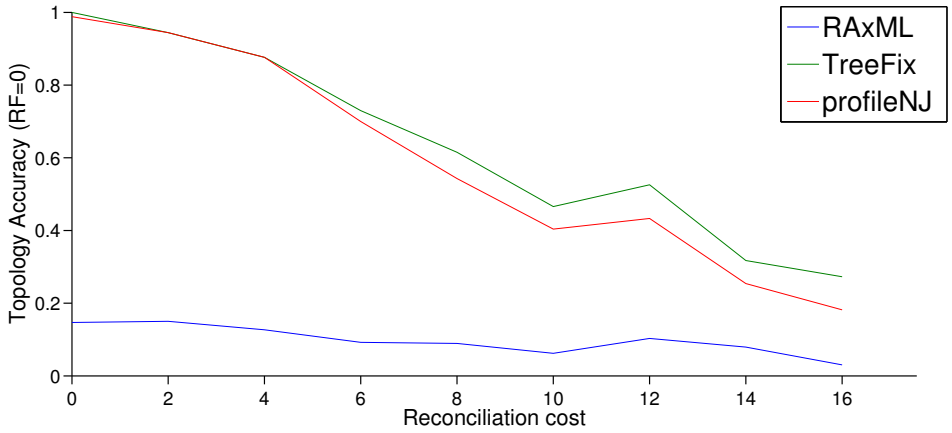


Figure C: Reconstruction accuracy of RAxML, TreeFix and ProfileNJ for increasing cost of reconciliation. Both TreeFix and ProfileNJ accuracy decrease when the number of evolutionary events increase, with TreeFix performing slightly better, but both still outperform RAxML.

Most cases of ProfileNJ errors in tree topologies (non-zero RF distance) are due to duplications that have been predicted lower on the tree. This is a known bias of reconciliation-based reconstruction methods, as lower duplications lead to less loss predictions. An example is given in Figure D.

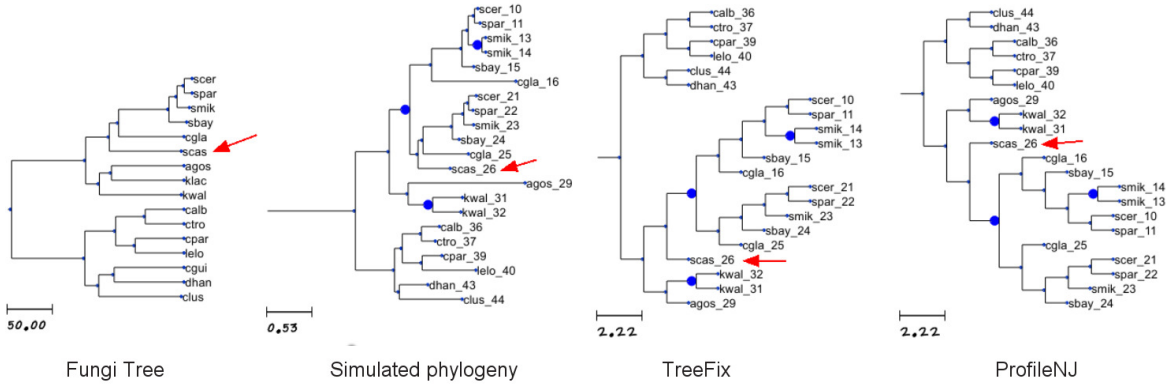


Figure D: An example of erroneously predicted topologies by TreeFix and ProfileNJ. Minimizing losses led both algorithms to place scas_26 above the predicted duplication, while the true (non parsimonious in terms of DL) scenario is a loss in scas.

Duplication and Loss Accuracy: Figure E illustrates accuracy of the duplication, loss and mutation (loss+duplication) costs, measured as the ratio of correct over all predictions, where a correct prediction is an output tree exhibiting the same cost as the true tree.

Again, ProfileNJ and TreeFix exhibit similar results, with ProfileNJ performing slightly better. High accuracy is obtained for all three measures, even though slightly lower for losses. This is due to the tendency of reconciliation-based methods to predict duplications lower in the tree, as illustrated in Figure D. On

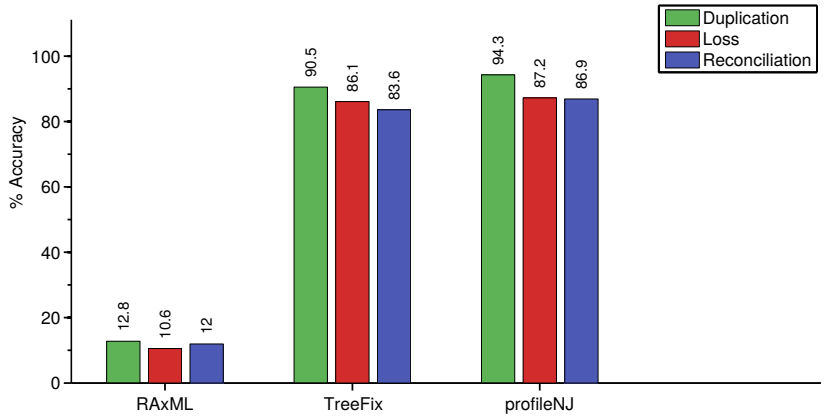


Figure E: Accuracy of duplications, losses and reconciliation cost of TreeFix, ProfileNJ and RaxML trees on $\sim$ 2500 simulated data from the fungi data-set.

the other hand, RAxML trees generally have a higher reconciliation cost (88% of RAxML trees have a reconciliation cost higher than the true cost, against 9.20% for TreeFix and 1.44% for ProfileNJ).

Gene Tree Size effect on Accuracy: To evaluate the scalability of the algorithms to gene trees of different sizes, we subdivided the set of trees into six classes according to their number of leaves: 0-10, 10-20, 20-30, 30-40, 40-50 and 50-60. Figure F shows that performance of all algorithms decrease with increase of tree size. However, ProfileNJ reconciliation cost accuracy is the less affected by the increase in tree size.

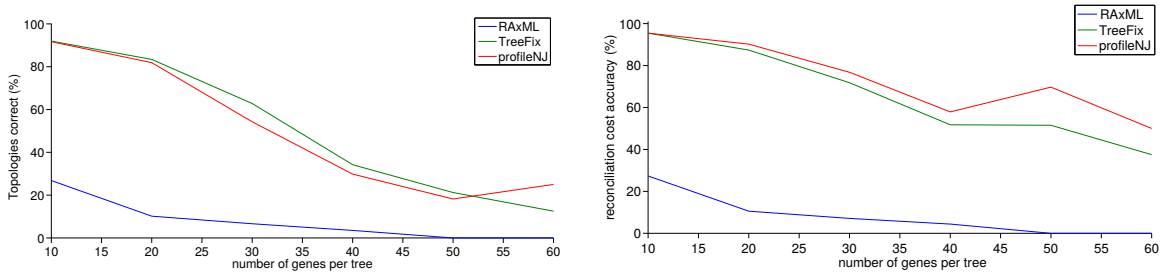


Figure F: Topological accuracy (left diagram) and reconciliation cost accuracy (right diagram) of TreeFix, ProfileNJ and RAxML for increasing size of gene tree from $\sim$ 2500 simulated fungal data-set.

Statistical Support: The AU test (using consel) has been used for evaluating the statistical support of trees according to sequence data. Figure G shows very similar results for the simulated tree and both TreeFix and ProfileNJ output trees, and much better results for the tree output by RAxML. This is not surprising as the RAxML tree is the ML tree. Here, for the large majority of data-sets, the simulated tree is not the ML tree, which invalidates the use of a sequence-based method such as RAxML for tree reconstruction. TreeFix performs better than ProfileNJ as the trees failing the AU test at $\alpha = 0.05$ represents 1.36% of all trees for

TreeFix and 9.165% of all trees for ProfileNJ. This is expected as TreeFix admits a corrected tree only if it is statistically equivalent to the input tree, while ProfileNJ outputs, among the optimal resolutions, the one best fitting the sequences.

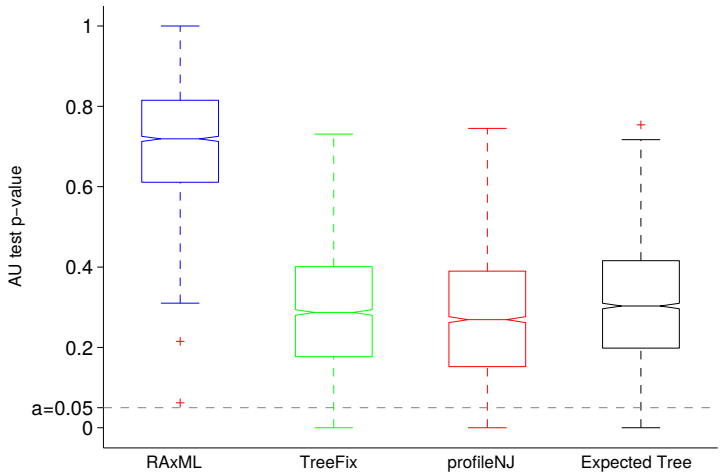


Figure G: Range of the p-value of an AU test using consel, between RAxML, TreeFix, ProfileNJ outputs and the simulated tree.

Robustness to Errors in the Species Tree: Reconstruction methods using information on the species tree are dependant upon the quality of the used tree. To measure this dependancy, we considered an alternative species tree for fungi, obtained by [2] (Figure I). Results on topology accuracy are given in Figure H. By comparing with Figure 2 (from the main text), it clearly appears that accuracy of both ProfileNJ and TreeFix is significantly affected by this erroneous species tree. This drop of accuracy is somewhat lower for ProfileNJ than for TreeFix. Moreover, RAxML outperforms both algorithms in this case. A pure sequence-based method could therefore be more appropriate when there is ambiguity in the species tree. This is a clear limitation of a reconciliation-based reconstruction method. We believe, however, that correcting only branches with low support leads to less dependency on the specie tree if the contraction threshold is cautiously chosen. This is supported by the result of Figure I which show improvement of ProfileNJ tree, on an incorrect specie tree, for lower contraction thresholds.

Modes of evolution in eukaryotes

Figure J gives the number of losses per branch of the phylogeny.

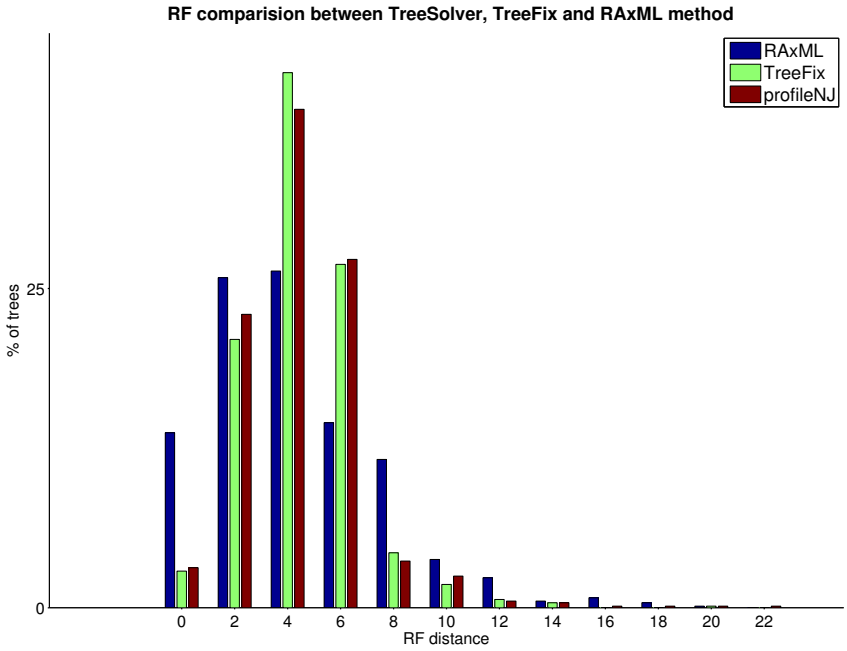


Figure H: Topology accuracy of RAxML, TreeFix and ProfileNJ, mesured by RF distance with the true tree, on $\sim$ 2500 simulated trees from the fungal data-set, using incorrect species tree topologies. TreeFix and ProfileNJ lost their high accuracies while no effect is seen for RAxML as its output is not affected by the species tree.

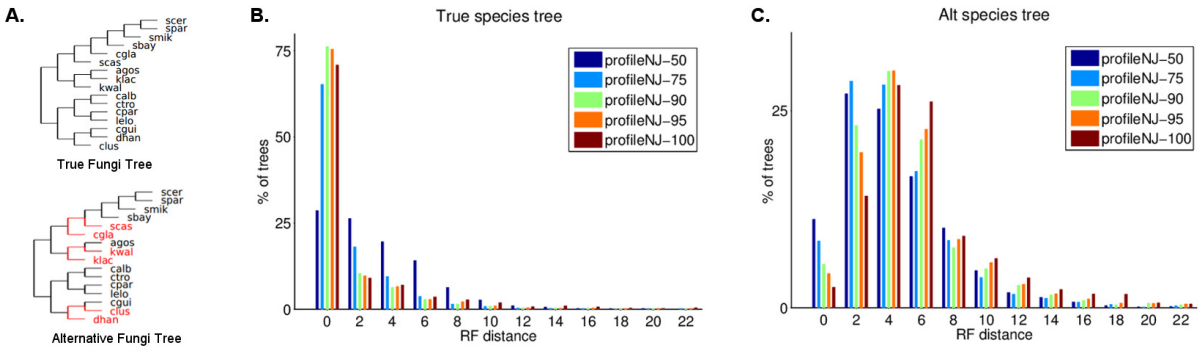


Figure I: Topology accuracy of ProfileNJ for different contraction threshold (α) using the correct and incorrect species tree. **(A)** True and alternative fungi specie trees. **(B)** Topology accuracy of ProfileNJ under the true specie tree. On those simulated data, we obtained the best performance when $\alpha = 90\%$ and $\alpha = 95\%$. Topology accuracy drop for lower values of α as incorrect branches are accepted and the tree space exploration is reduced. **(C)** Topology accuracy of ProfileNJ with an alternative specie tree. Here, lower values of α produce better tree, suggesting that α can be used as a way to control uncertainty in either the branch of the gene tree or the specie tree information.

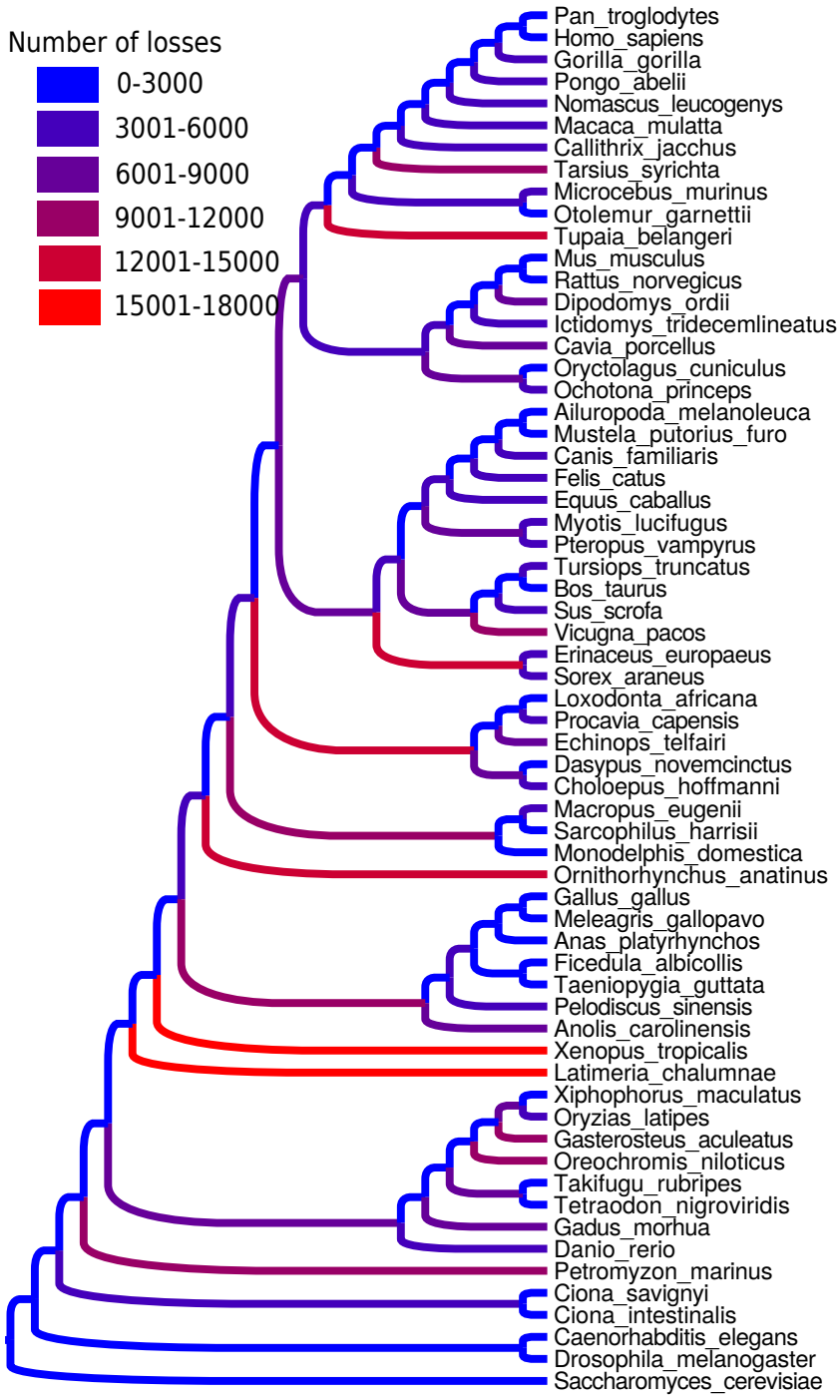


Figure J: Numbers of losses in the eukaryote phylogeny, estimated with ProfileNJ trees from PhyML starting trees on the whole Ensembl Compara database, version 73.

Reference

[1] Huerta-Cepas J, Dopazo J, Gabaldon T. ETE: a python Environment for Tree Exploration. BMC Bioinformatics. 2010;11(1):24. Available from: http://www.biomedcentral.com/1471-2105/11/24.

[2] Wu YC, Rasmussen MD, Bansal MS, Kellis M. TreeFix: Statistically informed gene tree error correction using species trees. Systematic Biology. 2013;62(1):110- 120.

Annexe D

Supplementary Materials for "GATC : A Genetic Algorithm for gene Tree Construction under the Duplication-Transfer-Loss model of evolution"

Il s'agit de l'annexe de l'article présenté dans le chapitre 6 et publié dans *BMC genomics*.

Supplementary Materials for GATC: A Genetic Algorithm for gene Tree Construction under the Duplication-Transfer-Loss model of evolution

Emmanuel Noutahi, Nadia El-Mabrouk

1 Effect of crossover and mutation rates on reconstruction accuracy

The performance of a genetic algorithm is strongly affected by the selected crossover and mutation rates. The optimal crossover and mutation rates usually vary with the problem of concern. In order to measure the effect of these rates on GATC's reconstruction accuracy, we compared GATC's results on a subset of 100 gene families from the dataset of simulated cyanobacterial phylogenies, under different rates of crossover (P_{cross}) and mutation (P_{mut}). We used the following parameters: a fixed population size of $N = 20$ individuals, a maximum number of 100 generations, LG + Gamma for the substitution model and $(\lambda, \tau, \delta) = (2, 3, 1)$ for the events cost. In total, 15 different rates of crossover and mutation were evaluated : $(P_{cross}, P_{mut}) \in \{0.2, 0.4, 0.6, 0.8, 1\} \times \{0, 0.5, 1\}$. Gene trees of the initial population were obtained using PolytomySolver. We measured reconstruction accuracy according to the error:

$$\Delta_E = \frac{\sum_i^N nRF(G_i, G_{true})}{N}$$

which corresponds to the mean normalized Robinson-Foulds distance between the gene trees in the last generation of evolution and the true phylogenetic tree for each simulated gene family.

A general tendency observed from the results depicted in Figure S1 is the decrease in the mean error Δ_E for increasing rate of crossover. Similarly, for the same crossover rate, higher rates of mutation usually yield better trees. Given the elitist nature of the algorithm, it is expected that high crossover rates would result in faster convergence. There is however a

risk of converging towards a local optimum due to loss of diversity in the latter generations. The use of high mutation rates can partly help solve this problem, but it can also prevent the solutions from converging. For the comparison between GATC and other methods we use $(P_{cross}, P_{mut}) = (0.8, 0.5)$, which gave reasonably good results with a balance between alternative gene tree topologies exploration and convergence speed.

2 Robustness to errors in the species tree topology

Since integrative methods for gene tree reconstruction rely on the accurate reconciliation between gene and species trees, they are sensitive to errors in the species tree [1, 2]. To measure the effect of these errors on reconstruction accuracy, we considered five alternative species tree topologies $(S_1, S_2, S_3, S_4, S_5)$ with increasing RF distance towards the assumed true cyanobacterial tree (S_{true}). The alternative topologies were obtained by performing SPR moves on S_{true}. We ran six instances of GATC, on 100 gene families, using, each time, a different species tree for the reconciliation framework. Trees of the initial population were obtained from bootstrap replicates and we used the same parameters as above with $(P_{cross}, P_{mut}) = (0.8, 0.5)$. Performance was again measured according to the mean error Δ_E between the trees of the last generation and the simulated gene tree. A comparison of GATC's results on each of the six instances and RAxML trees is shown on Figure S2. As expected, GATC's performance is heavily influenced by the error rates in the species tree, with alternative topologies that are most similar to S_{true} giving better results. GATC's average solutions were still more accurate than RAxML trees for species tree with only a few topological errors. Although integrative methods are dependent upon the quality of the species tree and should be avoided when the species tree is unreliable, they are generally robust against uncertainties on only a few branches of the species tree. One possible approach to reduce the effect of unreliable species trees could be to build consensus gene tree with bootstrap values from multiple runs with alternative, yet well supported species tree.

3 Supplementary Figures

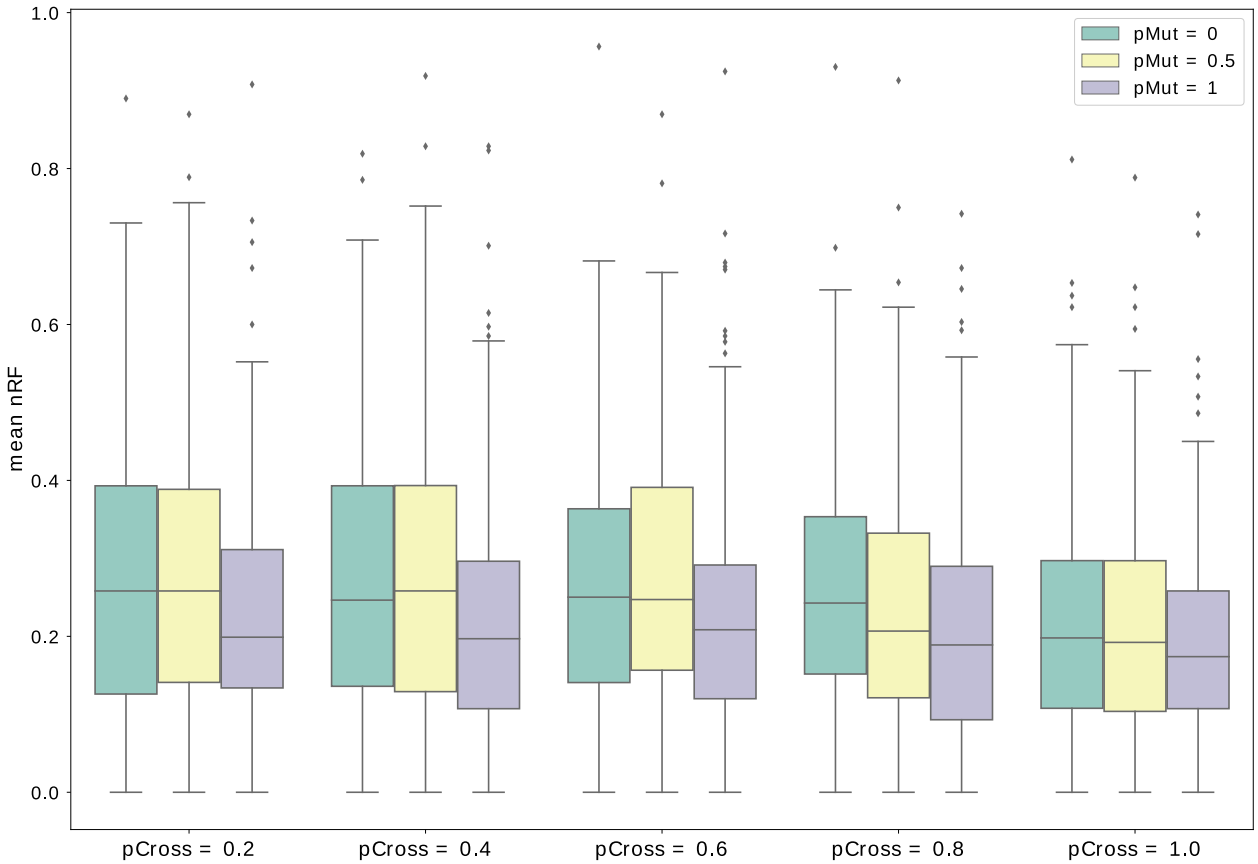


Figure S1: **Effect of various crossover and mutation rates on GATC's reconstruction accuracy**. A general tendency observed is the decrease in error rates for increasing rates of crossover and mutation.

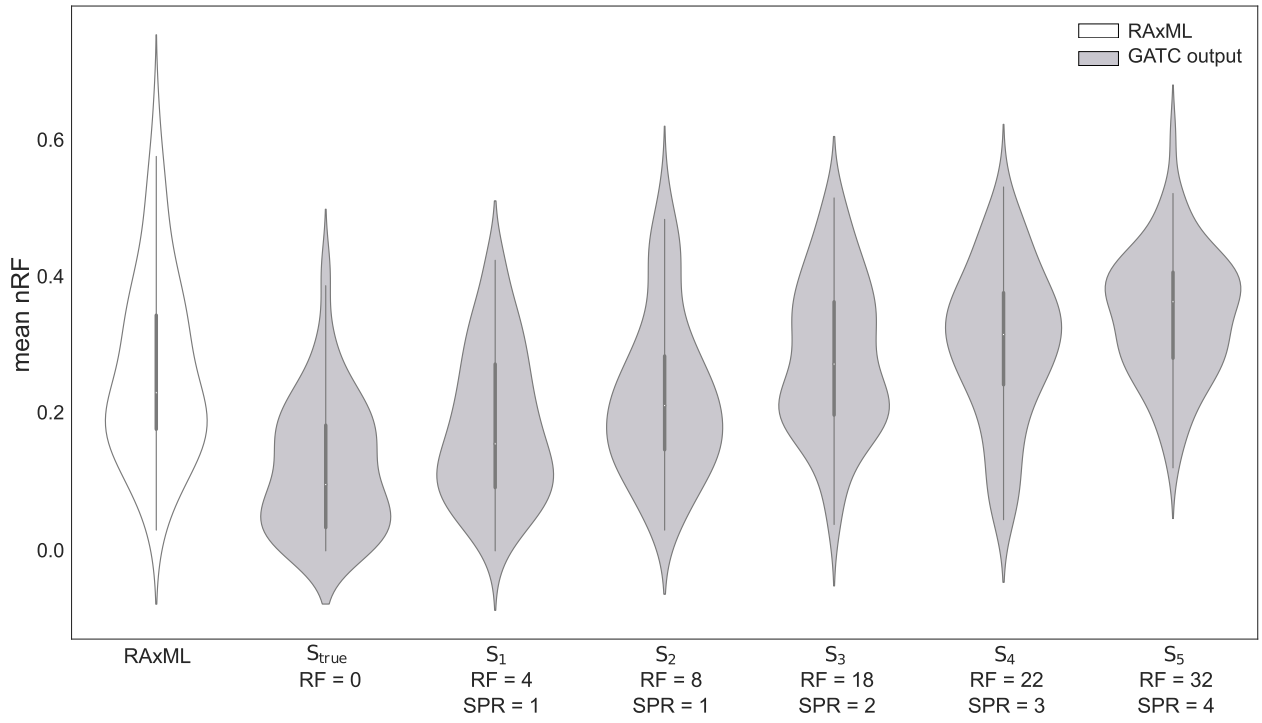


Figure S2: **Effect of using alternative species tree topologies on reconstruction accuracy.** The accuracy of GATC's gene tree reconstruction decreases for increasing rate of errors in the species tree used. Nevertheless, the algorithm is robust, to some extent, against minor errors in the species tree topologies.

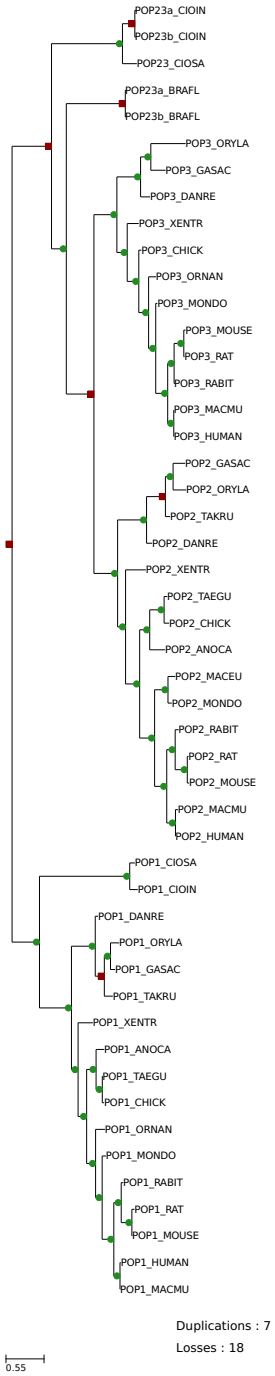


Figure S3: **Reference tree for the Popeye family.** Branch lengths are not shown. The gene tree was reconcilied with the species tree. Duplication nodes (leading to paralogs) are indicated by a red square, while speciation nodes (leading to orthologs) are indicated by a green circle. The total number of duplications and losses are shown at the bottom. Lost branches were not shown for clarity.

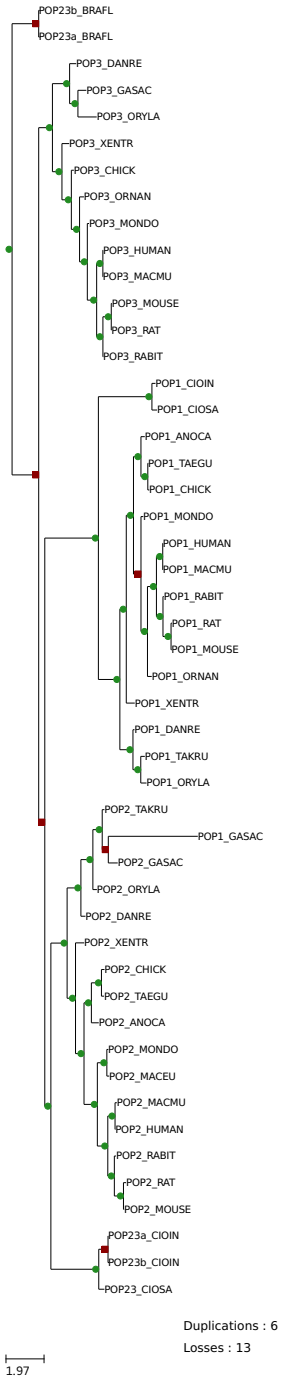


Figure S4: **Tree 1 return by GATC.** Branch lengths are not shown. The gene tree was reconcilied with the species tree. Duplication nodes (leading to paralogs) are indicated by a red square, while speciation nodes (leading to orthologs) are indicated by a green circle. The total number of duplications and losses are shown at the bottom. Lost branches are not shown for clarity.

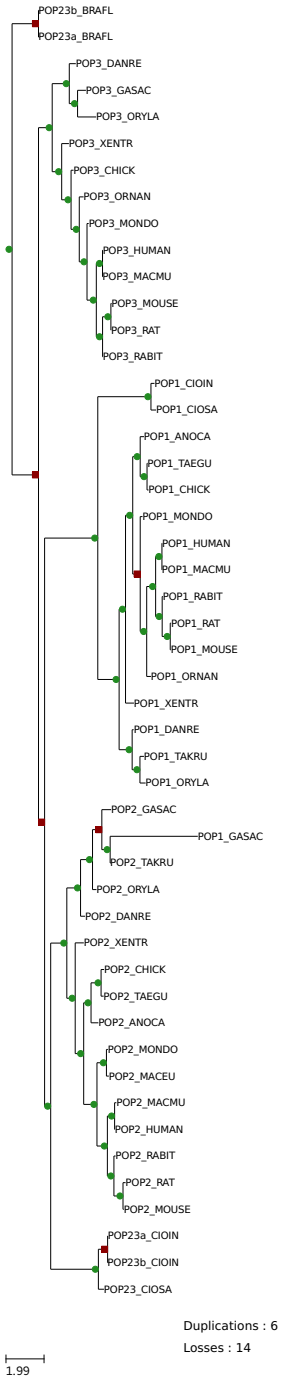


Figure S5: **Tree 2 return by GATC.** Branch lengths are not shown. The gene tree was reconcilied with the species tree. Duplication nodes (leading to paralogs) are indicated by a red square, while speciation nodes (leading to orthologs) are indicated by a green circle. The total number of duplications and losses are shown at the bottom. Lost branches are not shown for clarity.

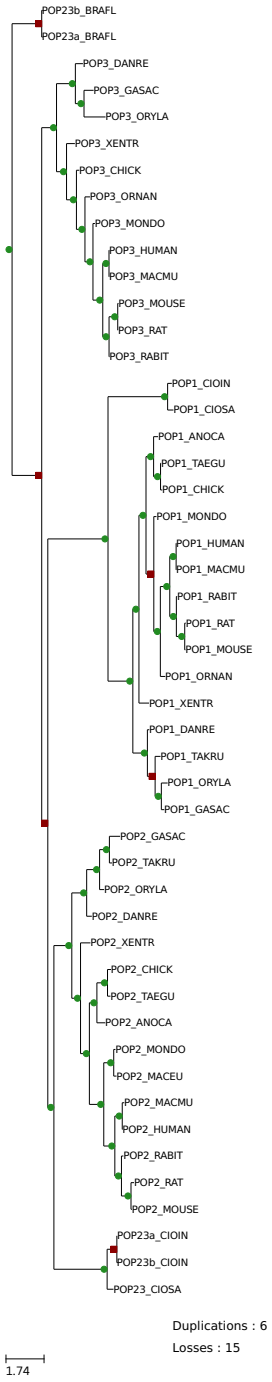


Figure S6: **Tree 3 return by GATC.** Branch lengths are not shown. The gene tree was reconcilied with the species tree. Duplication nodes (leading to paralogs) are indicated by a red square, while speciation nodes (leading to orthologs) are indicated by a green circle. The total number of duplications and losses are shown at the bottom. Lost branches are not shown for clarity.

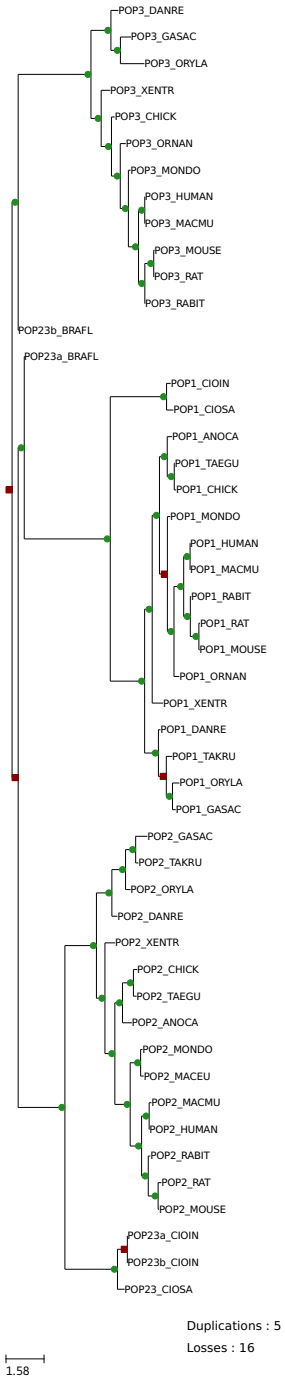


Figure S7: **Tree 4 return by GATC.** Branch lengths are not shown. The gene tree was reconcilied with the species tree. Duplication nodes (leading to paralogs) are indicated by a red square, while speciation nodes (leading to orthologs) are indicated by a green circle. The total number of duplications and losses are shown at the bottom. Lost branches are not shown for clarity.

References

[1] E. Noutahi, M. Semeria, M. Lafond, J. Seguin, L. Gueguen, N. El-Mabrouk, and E. Tannier. Efficient gene tree correction guided by genome evolution. *Plos.One*, 11(8), 2016.

[2] Y.C Wu, M.D. Rasmussen, M.S. Bansal, and M. Kellis. TreeFix: Statistically informed gene tree error correction using species trees. *Systematic Biology*, 62(1):110- 120, 2013.

Annexe E

Efficient Non-Binary Gene Tree Resolution with Weighted Reconciliation Cost

Manuel Lafond, Emmanuel Noutahi, Nadia El-Mabrouk

Ce manuscript décrit en profondeur l'algorithme de PolytomySolver. Il a été présenté à la conférence *Combinatorial Pattern Matching (CPM)* 2016 et est publié dans *LIPIcs-Leibniz International Proceedings in Informatics*.

Efficient Non-Binary Gene Tree Resolution with Weighted Reconciliation Cost

Manuel Lafond[1], Emmanuel Noutahi[2], and Nadia El-Mabrouk[3]

1 **DIRO, Université de Montréal, H3C 3J7, Canada**
`lafonman@iro.umontreal.ca`
2 **DIRO, Université de Montréal, H3C 3J7, Canada**
`noutahie@iro.umontreal.ca`
3 **DIRO, Université de Montréal, H3C 3J7, Canada**
`mabrouk@iro.umontreal.ca`

Abstract

Polytomies in gene trees are multifurcated nodes corresponding to unresolved parts of the tree, usually due to insufficient differentiation between sequences. Resolving a multifurcated tree has been considered by many authors, the objective function often being the number of duplications and losses reflected by the reconciliation of the resolved gene tree with a given species tree. Here, we present PolytomySolver, an algorithm accounting for a more general model allowing for costs that can vary depending on the operation, but also on the considered genome. The time complexity of PolytomySolver is linear for the unit cost and is quadratic for the general cost, which outperforms the best known solutions so far by a linear factor. We show, on simulated trees, that the gain in theoretical complexity has a real practical impact on running times.

1998 ACM Subject Classification Biology and genetics

Keywords and phrases gene tree, polytomy, reconciliation, resolution, weighted cost, phylogeny

Digital Object Identifier 10.4230/LIPIcs.CPM.2016.14

1 Introduction

Reconstructing gene trees is a fundamental task in bioinformatics and a prerequisite for most biological studies on gene function. Consequently, a plethora of phylogenetic methods have been developed, most of them integrating measures of statistical support (e.g. by bootstrapping or jackknifing), reflecting the confidence we have on the prediction. Some of them, such as bayesian methods [9, 11] lead to non-binary trees. Moreover, weakly supported branches are often contracted and also lead to non-binary trees. Thus, although unresolved nodes in a tree may reflect a true (or *hard* [12]) simultaneous speciation or duplication event leading to more than two gene copies, they are usually artifacts (called *soft*), due to methodological reasons or to a lack of resolution between sequences.

Information for the full resolution of a gene tree may rely on the weakly exploited link between gene and species evolution. The question of resolving a non-binary gene tree by minimizing the number of duplications and losses resulting from the reconciliation of the gene tree with the species tree has first been considered in NOTUNG [2] and later by Chang and Eulenstein [1]. In 2012 [8], we developed the first linear-time algorithm for resolving a polytomy (a single unresolved node), leading to a quadratic-time algorithm for a whole tree. Recently, algorithmic results extending linearity to a whole gene tree have been obtained by Zheng and Zhang [15]. These linearity results are however restricted to the case of a unit cost for duplications and losses. On the other hand, an algorithm allowing different costs for

27th Annual Symposium on Combinatorial Pattern Matching (CPM 2016).
Editors: Roberto Grossi and Moshe Lewenstein; Article No. 14; pp. 14:1–14:12

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Table 1 Time-complexity results for reporting a single optimal resolution of a whole gene tree G of size $|G|$ with a species tree S of size $|S|$, where Δ is the largest degree of a node in G, δ is the cost of a duplication and λ the cost of a loss. The last column refers to the case in which each species s has its own duplication cost δ_s and loss cost λ_s.

	$\delta = \lambda = 1$	$(\delta, \lambda) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$	$\{(\delta_s, \lambda_s)\}_{s \in V(S)}$
NOTUNG[6]	$O(\|S\|\|G\|\Delta^2)$	$O(\|S\|\|G\|\Delta^2)$	
Lafond[8]	$O(\|S\|\|G\|)$	$O(\|S\|\|G\|\Delta)$	
Zheng & Zhang[15]	$O(\|G\|)$	$O(\|G\|\Delta^2)$	
PolytomySolver	$O(\|G\|)$	$O(\|G\|\Delta)$	$O(\|G\|\|S\|\Delta)$

duplications and losses has been considered in NOTUNG [6], and further improved by Zheng and Zhang [15], using a compressed species tree idea.

In this paper, we present a new algorithm called PolytomySolver, which handles unit costs in linear time and improves the best complexity to date for more general duplication and loss cost model by a linear factor (complexity results are given in Table 1). Additionally, PolytomySolver is the first algorithm enabling to account for various evolutionary rates across the branches of a species tree, as it allows assigning each taxa its specific duplication and loss cost. This functionality may be used to reduce the effect of missing data by assigning a lower loss cost to species that are more likely to be concerned by such loss of information. It is also of practical use when biological evidence supports some particularly low or high gene duplication or loss rates in some species of interest [10]. In particular, fractionation following whole genome duplication (WGD) results in an excess of gene losses. In Section 6, we give an example showing that assigning appropriate costs to post-WGD genomes is important for an accurate inference.

The paper is subdivided as follows. First, in Section 3, we show how the linear-time algorithm developed previously by our group [8] for resolving a polytomy with unit duplication and loss cost can be extended to arbitrary costs, depending on the operation and on the genome affected by the operation. This extension is however not linear anymore but rather leads to a cubic-time algorithm. We then, in Section 4, show how using the ideas introduced by Zheng and Zhang [15] allows to reduce this time complexity to quadratic, which is the best obtained to date for the same problem. We also show how unit costs can be handled in linear time, and how PolytomySolver can be used to output all optimal resolutions, which is an advantage compared to Zheng and Zhang's algorithms. In Section 5, comparing our new algorithm with NOTUNG and Zheng and Zhang's algorithm, we show that the obtained gain in theoretical complexity actually leads to a significant gain in running times. For space reason, all proofs are given in Appendix, which is available online at `http://www-ens.iro.umontreal.ca/~lafonman/en/publications.php`.

2 Preliminary

All trees are considered to be rooted. Given a set X, a *tree T for X* has its leafset $\mathcal{L}(T)$ in bijection with X. Denote by $V(T)$ its set of nodes, $r(T)$ its root, and write $|T| = |V(T)|$. Given two nodes x and y of T, x is a *descendant* of y, and y is an *ancestor* of x, if y is on the (inclusive) path between x and $r(T)$. The *degree* $deg(x)$ of a node x is the number of edges incident to x. The maximum degree of T is $\Delta(T) = \max_{v \in V(T)} deg(v)$ (or just Δ when T is clear from the context). Given a set L of leaves, the *lowest common ancestor* of L in

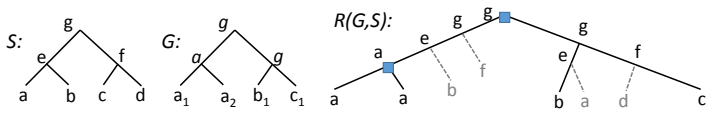


Figure 1 S is a species tree over $\Sigma = \{a, b, c, d\}$; G is a gene tree on the gene family Γ with two copies in genome a, one in genome b and one in genome c; $R(G, S)$ is a reconciliation of G with S with two duplications and four losses. Each node x of G and $R(G, S)$ is labeled by $s(x)$.

T, denoted $lca_T(L)$, is the common ancestor of L in T that is farthest from the root. A *polytomy* (or star tree) over a set L is a tree with a single internal node, which is of degree $|L|$, adjacent to each leaf of L. Finally, if x is a node of T, denote by T_x the subtree of T rooted at x, and by $T(x)$ the polytomy obtained by keeping only x and its children in T_x.

2.1 Gene Tree, Species Tree and Reconciliation

A species tree S for a set $\Sigma = \{\sigma_1, \cdots, \sigma_t\}$ of species represents an ordered set of speciation events that have led to Σ. Inside the species' genomes, genes undergo speciations when the species to which they belong do, but also duplications and losses (other events such as transfers can happen, but we ignore them here). A *gene family* is a set Γ of genes where each gene x belongs to a given species $s(x)$ of Σ. The evolutionary history of Γ can be represented as a *gene tree* G where $\mathcal{L}(G)$ is in bijection with Γ, and each internal node refers to an ancestral gene at the moment of an event (either speciation or duplication) belonging to the species $s(x) = lca_S(\{s(y) : y \in \mathcal{L}(G_x)\})$. We denote $\mathcal{S}(G) = \{s(y) : y \in \mathcal{L}(G)\}$ the set of species *represented by* G.

In this paper, we make no distinction between paralogous gene copies. In other words, a gene x is simply identified by the genome $s(x)$ it belongs to. A gene tree is therefore a tree where each leaf is labeled by an element of Σ, with possibly repeated leaf labels (Figure 1).

A *reconciliation* is an extension of the gene tree, obtained by adding lost branches, reflecting a history of duplications and losses in agreement with the species tree. Formally, an *extension* of G is a tree obtained from G by a sequence of graftings, where a *grafting* consists in subdividing an edge uv of G, thereby creating a new node w between u and v, then adding a leaf x with parent w. The new leaf x is mapped to a species $s(x)$ which is a node of S (internal or leaf). A formal definition follows (see Figure 1 for an example).

▶ **Definition 1** (Reconciled gene tree)**.** Let G be a binary gene tree and S be a binary species tree. A *reconciliation* $R(G, S)$ of G with S is an extension of G verifying: for each internal node x of $R(G, S)$ with two children x_l and x_r, either $s(x_l) = s(x_r) = s(x)$, or $s(x_l)$ and $s(x_r)$ are the two children of $s(x)$. The node x is a duplication in $s(x)$ in the former case, and a speciation node in $s(x)$ in the latter case. A grafted leaf x corresponds to a loss in $s(x)$.

Define δ_s as the duplication cost and λ_s as the loss cost assigned to a given species s. Then, the *reconciliation cost* of $R(G, S)$ is the sum of costs of the induced duplications and losses.

2.2 Problem statement

We consider a binary species tree S and a non-binary gene tree G. The goal is to find a *binary refinement* of G, as defined below.

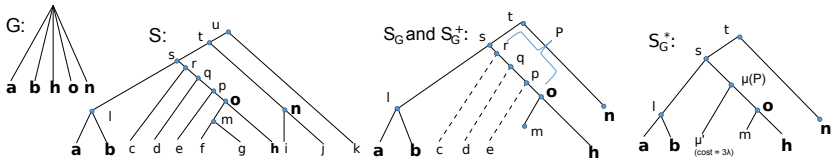


Figure 2 From left to right: a gene tree G; a species tree S; the species tree S_G linked to G is the tree illustrated by plain lines, and the augmented species tree S_G^+ linked to G is illustrated by plain and dotted lines; the compressed tree S_G^* linked to G as defined in Section 4. The leaf μ' of S_G^* has a special loss cost $\lambda_{\mu'} = 3$, as it results from the contraction of a path of length 3.

▶ **Definition 2** (binary refinement)**.** A *binary refinement* $B = B(G)$ of G is a binary tree such that $V(G) \subseteq V(B)$ and for every $x \in V(G)$, $\mathcal{L}(G_x) = \mathcal{L}(B_x)$.

The objective function taken for choosing among all possible binary refinements is the reconciliation cost.

▶ **Definition 3** (Resolution)**.** A *resolution* of G with respect to S is a reconciliation $R(B, S)$ between a binary refinement B of G and S. The set of all possible resolutions of a tree G is denoted $\mathcal{R}(G)$.

We are now ready to state our optimization problem.

Minimum Resolution Problem
Input: A binary species tree S and a non-binary gene tree G.
Output: A *Minimum Resolution* of G with respect to S (or simply *Minimum Resolution of* G), e.g. a resolution of G of minimum reconciliation cost with respect to S.

It has been previously shown [1] that each polytomy of G can be considered independently. In particular, a minimum resolution of G can be obtained by a depth-first procedure that solves each polytomy $G(x)$ iteratively, for each internal node x of G. Thus, in the following, we focus on a single polytomy $G = G(x)$.

Some parts of the species tree can be ignored in the process of refining G. Define the *species tree linked to* G, denoted by S_G, as the tree obtained from the subtree of S rooted at the lowest common ancestor of $\mathcal{S}(G)$, by removing all nodes that have no descendant in $\mathcal{S}(G)$ (Figure 2). The algorithms with the best known complexity results (Table 1) are obtained by using a compressed version S_G^* of this tree, which is defined in Section 4. We first begin, in Section 3, by describing the refinement strategy by using an *augmented species tree linked to* G, denoted S_G^+, obtained from S_G by adding to every node of degree two its missing child in S. It is known (c.f. [8, 15]) that resolving G with either S or S_G^+ leads to the same reconciliation cost. Intuitively, S_G^+ contains every node of S that may appear in a resolution of G, whether as a loss, a duplication or a speciation.

3 A dynamic programming approach

We present a dynamic programming approach for the Minimum Resolution Problem for a single polytomy G. It is a generalization of that presented in [8]. While the previous algorithm was developed for a unit cost of duplications and losses, the one we present here

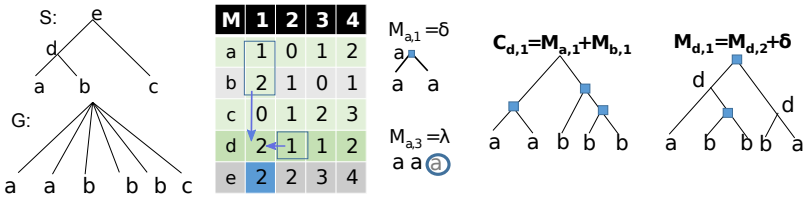


Figure 3 A polytomy G and a species tree S. The corresponding table M is obtained for $\delta_s = \lambda_s = 1$ for all species. Squares on trees illustrate duplications. To the right of table M, the forests corresponding to an $(a, 1)$ and $(a, 3)$-resolution are given, where the circled a illustrates a singleton loss. We illustrate the $(d, 1)$-resolution, rooted at a speciation node, corresponding to $C_{d,1} = 3$ (obtained from the vertical arrow in table M), and an optimal $(d, 1)$-resolution, obtained from a $(d, 2)$-resolution (horizontal arrow in M).

holds for a more general reconciliation cost, where each $s \in \Sigma$ has its own duplication cost δ_s and loss cost λ_s. In this section, we assume that $S = S_G^+$.

The recursion is made on the subtrees of S. Define the multiplicity $m(s)$ of $s \in V(S)$ in G as the number of times it appears in G, i.e. $m(s) = |\{x \in \mathcal{L}(G) : s(x) = s\}|$. An (s, k)-*resolution* of G is a forest of k reconciled gene trees $\mathcal{T} = \{T_1, \ldots, T_k\}$ such that, for each $1 \leq i \leq k$, $s(r(T_i)) = s$, and each leaf x of G with $s(x)$ being a descendant of s is present as a leaf of some tree of $\mathcal{T}$ (see Figure 3 for an example). All leaves of trees in $\mathcal{T}$ that are not in $\mathcal{L}(G)$ represent losses. Also, some trees of $\mathcal{T}$ may be restricted to a single node which is either a child x of $r(G)$ with $s(x) = s$, or a singleton loss in s. The cost of $\mathcal{T}$, denoted $c(\mathcal{T})$, is the sum of reconciliation costs of all T_is. Notice that since $S = S_G^+$, a resolution of G is an $(r(S), 1)$-*resolution*.

Denote by $M_{s,k}$ the minimum cost of an (s, k)-resolution for a given node s of S and a given integer $k \geq 1$ (and $M_{s,k} = \infty$ for $k < 1$). The final cost of a minimum resolution of G is given by $M_{r(S),1}$. The table M is computed, line by line, for all nodes of S, in a bottom-up traversal. For now, k is unlimited, but we show in the complexity section that there is no need to consider more than $|G| - 1$ columns.

The following lemma gives the base case for the leaves of S. It follows from the fact that, if k is larger than the number of available leaves, then additional leaves have to be added (called *singleton losses*); otherwise leaves have to be joined under duplication nodes. As an illustration, in Figure 3, this lemma is used to compute the three first lines of M.

▶ **Lemma 4** (Base case). *For a leaf node s of S, if $k > m(s)$ then $M_{s,k} = \lambda_s \cdot (k - m(s))$; otherwise $M_{s,k} = \delta_s \cdot (m(s) - k)$.*

The rest of this section focuses on the computation of a line M_s of M for an internal node s of S, from the lines M_{s_l} and M_{s_r}, where s_l and s_r are the two children of s in S. We require an intermediate cost table $C_{s,k}$, defined for internal nodes of S, accounting only for speciation events. That is, $C_{s,k}$ represents the minimum cost of an (s, k)-resolution in which every tree is rooted at a speciation node with two children (these two children may both be losses), or consists of a singleton node that is a child of $r(G)$ already mapped to s. For $k > m(s)$, such an (s, k)-resolution of cost $C_{s,k}$ can only be obtained from an $(s_l, k - m(s))$-resolution and an $(s_r, k - m(s))$-resolution by creating $k - m(s)$ speciation nodes, each joining a pair of (s_l, s_r) trees, then adding the $m(s)$ singleton trees mapped to s. No other scenarios are possible, since (s, k)-resolutions are reconciled trees, and each non-singleton root is a speciation in s

that must have genes mapped to s_l and s_r as children. See for example the $(d, 1)$-resolution corresponding to $C_{d,1}$ in Figure 3. Note that if instead $k \leq m(s)$, such an (s, k)-resolution cannot exist, since $m(s)$ trees are required for the children of $r(G)$ mapped to s, plus at least another tree containing the genes in a descendant of s. Thus we define:

$$C_{s,k} = M_{s_l,k-m(s)} + M_{s_r,k-m(s)} \text{ if } k > m(s) \text{ and } C_{s,k} = +\infty \text{ otherwise} \quad \textbf{(1)}$$

It is readily seen that $M_{s,k} \leq C_{s,k}$. A recurrence for computing $M_{s,k}$ follows.

▶ **Lemma 5.** *For an internal node s of S, $M_{s,k} = \min(M_{s,k-1} + \lambda_s, M_{s,k+1} + \delta_s, C_{s,k})$.*

This recurrence cannot be used as such to compute C and M, as it induces both a left and right dependency. That is, $M_{s,k}$ depends on $M_{s,k+1}$ and vice-versa, leading to a chicken-and-egg problem as to which value should be computed first. In the case of a unit cost $\delta_s = \lambda_s = 1$ for all s, we have shown in [8] that this dependency can be avoided by considering a strong property on lines of M. Indeed, each line M_s is characterized by two values k_1 and k_2 such that, for any $k_1 \leq k \leq k_2$, $M_{s,k}$ is minimum, for any $k \leq k_1$, $M_{s,k-1} = M_{s,k} + 1$, and for any $k \geq k_2$, $M_{s,k+1} = M_{s,k} + 1$. In other words, M_s has a slope of -1 until k_1, a slope of 0 until k_2, then a slope of 1. In particular, M_s can be treated as a convex function fully determined by k_1, k_2 and its minimum value γ. We then say M_s has a *minimum plateau* between k_1 and k_2. For example, line M_d in Figure 3 is fully determined by $k_1 = 2$ and $k_2 = 3$.

Here, we extend these results by first showing, in Lemma 7, that both C and M are still convex, albeit having less predictable changes in the slopes. Nevertheless, this allows to first compute the bounds k_1 and k_2 of the functions' minimum plateau, and then extend to the left and to the right from this plateau.

We first recall the formal definition of a discrete convex function, then state the convexity result for C and M and finally give the recurrences of the dynamic programming algorithm in Theorem 8.

▶ **Definition 6** (Convex function)**.** A discrete function f is convex if and only if, for any integer $n > 1$, the two following statements, which are equivalent, are true.

- $f(n+1) + f(n-1) - 2f(n) \geq 0$;
- for any integers $\epsilon_1, \epsilon_2 > 0$ and any integer $n > \epsilon_1$, $f(n - \epsilon_1) + f(n + \epsilon_2) - 2f(n) \geq 0$.

▶ **Lemma 7.** *Both M_s and C_s are convex.*

▶ **Theorem 8** (Recurrence 2)**.** *Let k_1 and k_2 be the smallest and largest values, respectively, such that $C_{s,k_1} = C_{s,k_2} = \min_k C_{s,k}$. Then,*

$$M_{s,k} = \begin{cases} C_{s,k} & \text{if } k_1 \leq k \leq k_2 \\ \min(C_{s,k}, M_{s,k+1} + \delta_s) & \text{if } k < k_1 \\ \min(C_{s,k}, M_{s,k-1} + \lambda_s) & \text{if } k > k_2 \end{cases}$$

Theorem 8 provides the way for computing a row M_s for an internal node s of S: for each k, compute $C_{s,k}$ using recurrence **(1)** and keep the two columns k_1 and k_2 setting the bounds of the convex function's plateau. Extend to the left of k_1 using $M_{s,k} = \min(C_{s,k}, M_{s,k+1} + \delta_s)$, and to the right of k_2 using $M_{s,k} = \min(C_{s,k}, M_{s,k-1} + \lambda_s)$. These recurrences, with the base case for S leaves given in Lemma 4, describe the dynamic programming algorithm, that we call PolytomySolver, for computing the cost $M_{r(S),1}$ of a minimum resolution of the polytomy G with respect to S. We refer the reader to [8] for the reconstruction of a solution from M in linear time, which is accomplished using a standard backtracking procedure.

Complexity

The following lemma states that there is no reason to explore more gene copies of a given species than the size of the polytomy, in other words, the size of a line of M can be bounded by $|G|$. This fact may seem obvious to the accustomed, but in [6] it was equally "obvious" that only $m^* = \max_{s \in V(S)} m(s)$ columns needed to be considered, which turns out to be wrong [1]. In fact, this Lemma requires a surprising amount of care in the details (see Appendix).

▶ **Lemma 9.** *Only the values of M and C for columns k between 1 and $|G|-1$ need to be computed.*

It follows from Lemma 4, Theorem 8 and Lemma 9 that each row of C and M can be computed in time $O(|G|)$, and the whole table in time $O(|S||G|)$.

Now suppose that H is a general tree with p polytomies, where Δ is the largest degree of a polytomy. According to the depth-first procedure described at the end of Section 2, G can be resolved in time $O(p|S|\Delta)$, which is less than $O(|H||S|\Delta)$. In the next section, we improve this to $O(|H|\Delta)$ in the case of distinct costs δ and λ that are shared across all species, and $O(|H|)$ in the case of equal costs $\delta = \lambda$.

4 A faster algorithm using species tree compression

Assume that all species have the same duplication cost δ and the same loss cost λ. We call it *unit cost* if $\delta = \lambda$, and *general cost* otherwise. Again we assume that G is a polytomy.

In the previous section, results have been obtained using the augmented linked species tree S_G^+. As observed by Zheng and Zhang [15], S_G^+ contains many "useless" nodes that do not provide any meaningful information with regards to the resolution of G. This idea allowed them to optimize their refinement algorithm for the unit cost, leading to a linear-time algorithm. However, their algorithm does not apply to the general cost. For such a cost, their optimisation idea was rather applied to the NOTUNG's algorithm, which is less efficient. Here, we use a similar idea to optimize PolytomySolver. More precisely, we show how a compressed version of the linked species tree S_G can be used to reduce the complexity for refining a general tree G to $O(|G|\Delta)$ for the general cost, and to $O(|G|)$ for the unit cost.

We first need some definitions. Let T be a tree. Call P a *path in* T if P is a sequence of non-root adjacent vertices of degree two in T. *Contracting* P in T consists in replacing P by a single node $\mu = \mu(P)$. Now, let U be the set of non-root vertices of degree two of S_G that are not in $\mathcal{S}(G)$. We call U the set of "useless nodes" of S_G. Notice that $S_G[U]$, the graph obtained from S_G by keeping only nodes of U and edges with both endpoints in U, corresponds to a set of disjoint paths in S_G. The *compressed tree* S_G^* is the tree obtained from S_G by contracting every path P of $S_G[U]$ to $\mu = \mu(P)$, then adding a leaf child μ' to every such μ (see Figure 2 for an example). Moreover, we set a special loss cost $\lambda_{\mu'} = \lambda|P|$ to μ' (and duplication cost δ as every other node). This special loss cost ensures that a loss in μ' is counted as a loss in every node in P. Notice that some internal nodes of S_G that are included in $\mathcal{S}(G)$ may still have only one child. Thus S_G^* is finally obtained by adding to each remaining node having only one child a new leaf child (duplication of cost δ and loss cost λ). The following Theorem ensures that S_G^* does not change the solution space.

[1] The complexity reported in Table 1 is not the one reported by NOTUNG, as dependency is not given on Δ but instead on m^*. However, it can be shown that considering m^* columns is not enough on some examples.

▶ **Theorem 10.** *Let T be a binary refinement of G. Then the reconciliation cost of T is the same whether we reconcile it with S_G^+ or S_G^* and their corresponding duplication/loss costs.*

Thus, using S_G^+ or S_G^* leads to the same minimum resolution for G. We show that using S_G^* leads to reduction in time complexity of the algorithm.

▶ **Theorem 11.** *Given a gene tree H,* PolytomySolver *can run in time $O(\Delta|H|)$.*

4.1 The case of a unit cost

In [8], we showed how, in the case of a unit cost $\delta = \lambda$, each line M_s of M can be computed in constant time. However, in order to take advantage of the compressed species tree $S = S_G^*$, we need to account for special leaves μ' with loss cost $\lambda_{\mu'} > 1$, since they make the cost not unitary anymore. The following theorem allows us to extend the result to this specific case. It leads to the computation of M in time $O(|S_G^*|) = O(|G|)$ for a polytomy G. The complexity for a gene tree H is thus reduced to $O(|H|)$, which results in a reduction of the previous complexity by a factor of Δ.

▶ **Theorem 12.** *Suppose $S = S_G^*$. Then for $s \in V(S)$,*

1. *if s is a leaf with loss cost $\lambda = 1$, then $M_{s,k} = |k - m(s)|$;*
2. *if s is a leaf with loss cost $\lambda_s > 1$, then $M_{s,k} = k \cdot \lambda_s$;*
3. *if s is an internal node, there exist 3 integers k_1, k_2 and γ_s such that*

$$M_{s,k} = \begin{cases} \gamma_s & \text{if } k_1 \leq k \leq k_2 \\ \gamma_s + k_1 - k & \text{if } k < k_1 \\ \gamma_s + k - k_2 & \text{if } k > k_2 \end{cases}$$

Moreover, k_1, k_2 and γ_s can be computed in constant time.

4.2 Constructing all minimum resolutions

After computing table M, it remains to compute $(r(S), 1)$-resolutions, i.e. all resolutions of minimum cost. Without any increase in the theoretical time complexity of the algorithm, a simple pass through table M leads to one minimum resolution (see [8] for the details). Here we rather show how to recover all minimum resolution.

Denote by $\mathcal{P}_{s,k}$ the set of all minimum (s,k)-resolutions of a polytomy G. By setting $s = r(S)$ and $k = 1$, we exhibit the following recursive algorithm that finds $\mathcal{P}_{r(S),1}$. To do so, we define three intermediate solution sets $\mathcal{P}_{s,k}^{dup}, \mathcal{P}_{s,k}^{loss}$ and $\mathcal{P}_{s,k}^{spec}$, which respectively correspond to (s,k)-resolutions containing a duplication root, a singleton loss and only speciation roots (it turns out that these three cases are disjoint).

We show in the Appendix that this algorithm eventually terminates, and does find every solution. The essential reason that this algorithm finishes is because of the convexity of M_s, which allows avoiding circular dependencies between say $\mathcal{P}_{s,k}$ and $\mathcal{P}_{s',k'}$.

It can be shown that this algorithm takes time $O(|S| \cdot |\mathcal{P}_{r(S),1}|)$, which may be exponential. Methods for outputting solutions iteratively, each in polynomial time, seem possible, but are not immediately obvious. Notice that Zheng and Zhang's algorithms [15] can only output a subset of $\mathcal{P}_{r(S),1}$. As for NOTUNG, it takes time $O(|S|\Delta \cdot (|\mathcal{P}_{r(S),1}| + \Delta))$ to construct every optimal solution [2].

procedure COMPUTE $\mathcal{P}_{s,k}$
  if s is a leaf and $m(s) = k$ **then**
    return k singleton trees mapped to s
  Let $\mathcal{P}^{dup}_{s,k} = \emptyset, \mathcal{P}^{loss}_{s,k} = \emptyset, \mathcal{P}^{spec}_{s,k} = \emptyset$
  if $M_{s,k} = M_{s,k+1} + \delta_s$ **then**
    Compute $\mathcal{P}_{s,k+1}$
    for every forest $\mathcal{T}$ in $\mathcal{P}_{s,k+1}$, and for every pair of distinct trees $T_1, T_2 \in \mathcal{T}$ **do**
      Add to $\mathcal{P}^{dup}_{s,k}$ the (s,k)-resolution obtained by joining $r(T_1)$ and $r(T_2)$
  if $M_{s,k} = M_{s,k-1} + \lambda_s$ **then**
    Compute $\mathcal{P}_{s,k-1}$
    for every forest $\mathcal{T}$ in $\mathcal{P}_{s,k-1}$ **do**
      Add to $\mathcal{P}^{loss}_{s,k}$ the (s,k)-resolution obtained adding a singleton loss in s in $\mathcal{T}$
  if s is an internal node with children s_1, s_2 and $M_{s,k} = M_{s_1,k-m(s)} + M_{s_2,k-m(s)}$ **then**
    Compute $\mathcal{P}_{s_1,k-m(s)}$ and $\mathcal{P}_{s_2,k-m(s)}$
    for each pair $(\mathcal{T}_1, \mathcal{T}_2)$ in $\mathcal{P}_{s_1,k-m(s)} \times \mathcal{P}_{s_2,k-m(s)}$, and for every bijection $f : \mathcal{T}_1 \longrightarrow \mathcal{T}_2$ **do**
      Add to $\mathcal{P}^{spec}_{s,k}$ the (s,k)-resolution $\mathcal{T}$ obtained by joining $r(T_1)$ with $r(f(T_1))$ for every
      $T_1 \in \mathcal{T}_1$, then adding the $m(s)$ children of G mapped to s as singleton trees
  Let $\mathcal{P}_{s,k} = \mathcal{P}^{dup}_{s,k} \cup \mathcal{P}^{loss}_{s,k} \cup \mathcal{P}^{spec}_{s,k}$, and **return** $\mathcal{P}_{s,k}$
end procedure

5 Results on simulated data

We compare the running time of our algorithm to Zheng and Zhang's algorithms [15] and NOTUNG, on simulated datasets for both cases of unit and general costs. We implemented PolytomySolver and Zheng and Zhang's algorithms in python and used the latest stable version (v2.6)[2] of NOTUNG. Our implementations are available at `https://github.com/UdeM-LBIT/profileNJ`. Run times are reported for single outputs of the algorithms.

We first simulated species trees with n leaves using a birth-death process. For each species tree, gene trees of fixed size $(1.5 \times n)$ and branch support picked from a standard uniform distribution, were simulated using a simple Yule process [13]. In order to mimic a gene family history with a high number of events (duplications and losses), we labeled each leaf of the gene tree with a uniformly chosen species from the set of leaves of the species tree. Non-Binary gene trees were then obtained by contracting edges of the gene trees with support lower than a fixed threshold r (0.2, 0.4, 0.6 and 0.8).

For each species tree and each algorithm, we measured the average running time on 40 non-binary trees (10 simulated gene trees for each contraction rate). All software were run on the same computer and with the same costs for duplications and losses.

We first considered the unit cost ($\lambda = \delta = 1$), for which both PolytomySolver and Zheng and Zhang's algorithm (LZZ) are linear. Figure 4a shows the results for values of n ranging from 500 to 10000, and Figure 4b shows results for n between 10000 and 100000. As expected, the two linear algorithms exhibit very similar run time in all cases, and are significantly

[2] Notice that an improved version of NOTUNG v2.8 became available after these tests were performed.

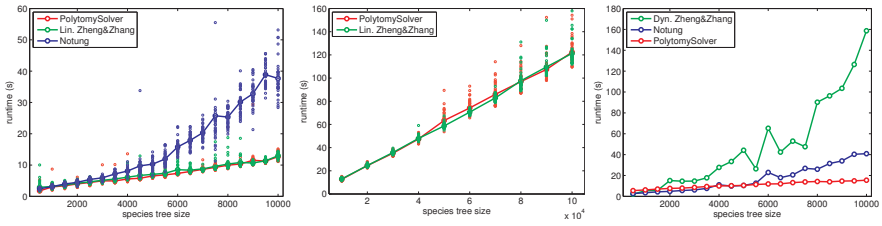


Figure 4 Running times comparisons between all algorithms for species trees of increasing size n and gene trees of size $1.5 \times n$. a Running times of PolytomySolver, LZZ (linear Zheng and Zhang's algorithm) and NOTUNG, using unit cost, for species trees of increasing size ranging from 500 to 10000. b Running times of PolytomySolver and LZZ for unit cost on larger species trees (n in the range of 10000 to 100000). c Running times of PolytomySolver, DZZ (Dynamic Zheng and Zhang's algorithm) and NOTUNG using $\delta = 3$ and $\lambda = 2$.

faster than NOTUNG, which could not be included in Figure 4b. Indeed, on those trees, NOTUNG took a considerable amount of time, and in some cases we could not get a result after many hours.

We then considered a non-unit cost, using $\delta = 3$ and $\lambda = 2$. Recall that PolytomySolver is quadratic in this case. As for the algorithm proposed by Zheng and Zhang for these costs, that we refer to by DZZ (for Dynamic Zheng and Zhang's algorithm), it is (essentially) cubic (see Table 1). Figure 4c gives the results for species trees of size ranging between 500 and 10000. As expected, PolytomySolver is faster than DZZ and NOTUNG. Surprisingly, NOTUNG turns out to be faster than DZZ, which rather expected to improve over NOTUNG as it uses the species tree compression idea. This could be due to the fact that NOTUNG is a well optimized program. Moreover, the error in NOTUNG of using m^* instead of Δ (see footnote in this Section 3), may accelerate the process, as m^* is usually much smaller than Δ.

6 A practical use of PolytomySolver

As handling species specific costs is one of the major contribution of this paper, we conclude our presentation by providing a biological example for which taking advantage of this flexibility of PolytomySolver leads to better accuracy.

We first downloaded the orthogroup of the yeast gene REG1, a regulatory subunit of type 1 protein phosphatase Glc7p, involved in negative regulation of glucose-repressible genes, from the Fungal Orthogroups Repository (http://www.broadinstitute.org/regev/orthogroups/). We then reconstructed the gene tree with PolytomySolver, using the same species tree as [14] and a unit cost for both λ and δ. Two equally parsimonious solutions with a reconciliation cost of 2 were obtained (Figures 5B, 5C).

It has been shown that the yeast *Saccharomyces cerevisiae* arose from an ancient whole-genome duplication (WGD) [4, 5, 7]. This WGD was immediately followed by a massive gene loss period, during which most of the duplicated gene copies were lost [7]. There is also evidence of lineage-specific loss of paralogous genes. In particular, *C. glabrata* and *S. castellii* appear to have lost several hundred paralogs [3, 5]. This is reflected in their total gene count, which are the lowest among the post-WGD genomes [14].

Whereas the solution shown in Figure 5C is in agreement with this WGD event, the alternative gene family history in Figure 5B places the duplication much lower in the tree, with

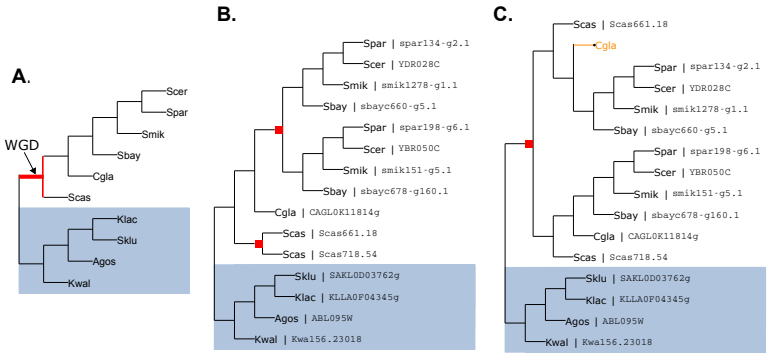


Figure 5 A. Phylogeny of ten Hemiascomycota fungi, including *S. cerevisiae (Scer), S. paradoxus (Spar), S. mikatae (Smik), S. bayanus (Sbay), C. glabrata (Cgla), S. castellii (Scas), K. waltii (Kwal), K. lactis (Klac), S. kluyveri (Sklu) and A. gossypii (Agos).* The whole-genome duplication (WGD) event in yeast is indicated. The species that did not went through the WGD are shadowed in light blue. **B.** and **C.** Two minimally resolved gene trees of the phosphatase Glc7p gene family. Duplication nodes are depicted by a red square and lost genes are shown in orange.

and additional duplication in *S. castellii* instead. By assigning to *C. glabrata* and *S. castellii* a loss cost lower than for all other species, the only solution returned by PolytomySolver is the one shown in Figure 5C. Using appropriate species dependant costs might therefore allow to filter the solution space with additional relevant information.

7 Conclusion

PolytomySolver is the most efficient algorithm to date for refining an unresolved gene tree. In contrast to previous methods, this algorithm is flexible enough to handle general reconciliation costs, allowing for instance to account for different costs over the branches of a species tree. Moreover, all topologies of optimal trees can be output by PolytomySolver. Notice that here we made no distinction between paralogous genes, which are simply referred to by their genome of origin. If we rather consider the specificity of each gene copy then, for a given topology obtained by PolytomySolver, an appropriate method shall be considered to distribute gene copies on leaves. We are presently investigating the possibility of introducing a Neighbor-Joining principle in the resolution process.

The gain in running time attained with PolytomySolver allows to perform exhaustive corrections of all trees contained in a large gene tree dataset such as Ensembl. Moreover, compared with NOTUNG, running time is independent upon the largest degree of a node, which makes the algorithm efficient enough to resolve highly unresolved trees. The next step will be to perform such a large scale gene tree dataset correction.

References

1 W.C. Chang and O. Eulenstein. Reconciling gene trees with apparent polytomies. In D.Z. Chen and D. T. Lee, editors, *Proceedings of the 12th Conference on Computing and Combinatorics (COCOON)*, volume 4112 of *Lecture Notes in Computer Science*, pages 235–244, 2006.

2 K. Chen, D. Durand, and M. Farach-Colton. Notung: Dating gene duplications using gene family trees. *Journal of Computational Biology*, 7:429–447, 2000.

3 Paul F Cliften, Robert S Fulton, Richard K Wilson, and Mark Johnston. After the duplication: gene loss and adaptation in saccharomyces genomes. *Genetics*, 172(2):863–872, 2006.

4 Fred S Dietrich, Sylvia Voegeli, Sophie Brachat, Anita Lerch, Krista Gates, Sabine Steiner, Christine Mohr, Rainer Pöhlmann, Philippe Luedi, Sangdun Choi, et al. The ashbya gossypii genome as a tool for mapping the ancient saccharomyces cerevisiae genome. *Science*, 304(5668):304–307, 2004.

5 Bernard Dujon, David Sherman, Gilles Fischer, Pascal Durrens, Serge Casaregola, Ingrid Lafontaine, Jacky De Montigny, Christian Marck, Cécile Neuvéglise, Emmanuel Talla, et al. Genome evolution in yeasts. *Nature*, 430(6995):35–44, 2004.

6 D. Durand, B.V. Haldórsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *Journal of Computational Biology*, 13:320–335, 2006.

7 Manolis Kellis, Bruce W Birren, and Eric S Lander. Proof and evolutionary analysis of ancient genome duplication in the yeast saccharomyces cerevisiae. *Nature*, 428(6983):617–624, 2004.

8 M. Lafond, K.M. Swenson, and N. El-Mabrouk. An optimal reconciliation algorithm for gene trees with polytomies. In *LNCS*, volume 7534 of *WABI*, pages 106-122, 2012.

9 Nicolas Lartillot and Hervé Philippe. A bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. *Molecular Biology and Evolution*, 21(6):1095–1109, Jun 2004. doi:10.1093/molbev/msh112.

10 Michael Lynch and John S Conery. The evolutionary demography of duplicate genes. *Journal of structural and functional genomics*, 3(1-4):35–44, 2003.

11 F. Ronquist and J.P. Huelsenbeck. MrBayes3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, 19:1572- 1574, 2003.

12 J.B. Slowinski. Molecular polytomies. *Molecular Phylogenetics and Evolution*, 19(1):114-120, 2001.

13 Mike Steel and Andy McKenzie. Properties of phylogenetic trees generated by yule-type speciation models. *Mathematical biosciences*, 170(1):91–112, 2001.

14 Ilan Wapinski, Avi Pfeffer, Nir Friedman, and Aviv Regev. Natural history and evolutionary principles of gene duplication in fungi. *Nature*, 449(7158):54–61, 2007.

15 Y. Zheng and L. Zhang. Reconciliation with non-binary gene trees revisited. In *Lecture Notes in Computer Science*, volume 8394, pages 418-432, 2014. Proceedings of RECOMB.

Annexe F

The nuclear genome of *Rhazya stricta* and the evolution of alkaloid diversity in a medically relevant clade of Apocynaceae

Jamal SM Sabir, Robert K Jansen, Dhivya Arasappan, Virginie Calderon, Emmanuel Noutahi, Chunfang Zheng, Seongjun Park, Meshaal J Sabir, Mohammed N Baeshen, Nahid H Hajrah, Mohammad A Khiyami, Nabih A Baeshen, Abdullah Y Obaid, Abdulrahman L Al-Malki, David Sankoff, Nadia El-Mabrouk, Tracey A Ruhlman

Ce manuscript présente une collaboration, publié dans *Nature Scientific Reports*, avec un groupe de recherche qui étudient l'évolution du génome de *Rhazia stricta*, une plante vénéneuse de la famille des *Apocynaceae*. Cette plante possède un intérêt particulier en médecine parce qu'on suspecte qu'elle est capable de produire des alcaloïdes utilisés pour le traitement du paludisme, mais aussi en chimiothérapie pour le traitement de la leucémie et du lymphome de Hodgkin. L'étude s'interesse en particulier à l'évolution de huit familles de gènes impliquées dans les voies métaboliques de ces alcaloïdes dans cinq génomes. Pour ce faire, l'histoire évolutive de ces gènes a été inférée en utilisant une méthode basée sur ProfileNJ .

SCIENTIFIC REPORTS

OPEN

The nuclear genome of *Rhazya stricta* and the evolution of alkaloid diversity in a medically relevant clade of Apocynaceae

Received: 19 February 2016
Accepted: 02 September 2016
Published: 22 September 2016

Jamal S. M. Sabir[1,*], Robert K. Jansen[1,2,*], Dhivya Arasappan[2], Virginie Calderon[3], Emmanuel Noutahi[3], Chunfang Zheng[4], Seongjun Park[2], Meshaal J. Sabir[1], Mohammed N. Baeshen[5], Nahid H. Hajrah[1], Mohammad A. Khiyami[6], Nabih A. Baeshen[7], Abdullah Y. Obaid[8], Abdulrahman L. Al-Malki[9], David Sankoff[4], Nadia El-Mabrouk[3] & Tracey A. Ruhlman[2]

Alkaloid accumulation in plants is activated in response to stress, is limited in distribution and specific alkaloid repertoires are variable across taxa. Rauvolfioideae (Apocynaceae, Gentianales) represents a major center of structural expansion in the monoterpenoid indole alkaloids (MIAs) yielding thousands of unique molecules including highly valuable chemotherapeutics. The paucity of genome-level data for Apocynaceae precludes a deeper understanding of MIA pathway evolution hindering the elucidation of remaining pathway enzymes and the improvement of MIA availability *in planta* or *in vitro*. We sequenced the nuclear genome of *Rhazya stricta* (Apocynaceae, Rauvolfioideae) and present this high quality assembly in comparison with that of coffee (Rubiaceae, *Coffea canephora*, Gentianales) and others to investigate the evolution of genome-scale features. The annotated *Rhazya* genome was used to develop the community resource, RhaCyc, a metabolic pathway database. Gene family trees were constructed to identify homologs of MIA pathway genes and to examine their evolutionary history. We found that, unlike *Coffea*, the *Rhazya* lineage has experienced many structural rearrangements. Gene tree analyses suggest recent, lineage-specific expansion and diversification among homologs encoding MIA pathway genes in Gentianales and provide candidate sequences with the potential to close gaps in characterized pathways and support prospecting for new MIA production avenues.

The power of genomics to illuminate evolutionary hypotheses lies in the availability of sequences from both early and late branching lineages for comparative analyses. Alkaloid accumulation in plant tissues is triggered in response to the range of stress encountered by plants; for humanity the result is a pharmacopeia ranging from caffeine to heroin to some of the most important chemotherapeutic and hypertensive treatment drugs available today[1]. Given that plant metabolic networks have followed a descent with modification pattern[2] and the unique and variable character of monoterpenoid indole alkaloid (MIA) production throughout the angiosperm order Gentianales, genome-scale comparative analyses within the order relative to more distant lineages could provide insight on the genetic factors involved in the evolution of this highly valuable aspect of plant secondary metabolism.

[1]Biotechnology Research Group, Department of Biological Sciences, Faculty of Science, King Abdulaziz University, Jeddah 21589, Saudi Arabia. [2]Department of Integrative Biology, University of Texas at Austin, Austin, 78712 Texas, USA. [3]Département d'informatique et de recherche opérationnelle, Université de Montréal, CP 6128 succ Centre-Ville, Montréal, H3C 3J7 Québec, Canada. [4]Department of Mathematics and Statistics, University of Ottawa, 585 King Edward Ave., Ottawa, K1N 6N5 Ontario Canada. [5]Department of Biological Sciences, Faculty of Science, University of Jeddah, Jeddah, Saudi Arabia. [6]King Abdulaziz City for Science and Technology, Riyadh 11442, Saudi Arabia. [7]Department of Biological Sciences, Faculty of Science, King Abdulaziz University, Jeddah 21589, Saudi Arabia. [8]Department of Chemistry, Faculty of Science, King Abdulaziz University, Jeddah 21589, Saudi Arabia. [9]Department of Biochemistry, Faculty of Science, King Abdulaziz University, Jeddah 21589, Saudi Arabia. *These authors contributed equally to the work. Correspondence and requests for materials should be addressed to T.R.A. (email: truhlman@austin.utexas.edu).

Alkaloid production in plants is limited in distribution and the specific alkaloid repertoire is variable across taxa. With the exception of the MIA quinolone derivatives, i.e. camptothecin, Gentianales are the exclusive plant producers of highly valuable MIA compounds, including the bisindole topoisomerase inhibitors vincristine and vinblastine. Regardless of their final form, more than 2500 identified MIAs begin with the condensation of tryptamine and secologanin to form strictosidine, the first committed molecule in the MIA pathway. The classification of MIAs into three broad groups was based on the arrangement of the fragmented cyclic monoterpene (secologanin), Iboga (catharanthine), Corynanthe-strychnos (ajmalicine) and Aspidosperma (stemmadenine). While all families of the order contain representatives that can accumulate MIA species with the Corynathe (type I) skeleton, Rauvolfioideae, a subfamily of Apocynaceae, represents a major center of structural expansion encompassing the diversity of skeletal arrangements and an astonishing number of downstream modifications yielding thousands of unique molecules[3].

Rhazya stricta Decne. (2n = 22; Apocynaceae, Rauvolfioideae) is an evergreen shrub abundant across Western and South Asia and is well adapted to harsh conditions[4]. Like other Rauvolfioideae, *Rhazya* accumulates MIAs. In fact, strictosidine was first isolated from *R. stricta* in 1968[5]. However recent study has focused mainly on *Catharanthus roseus* (L.) G. Don and, to a lesser extent, *Rauvolfia serpentina* (L.) Benth. ex Kurz and a handful of others in the family. There is an extensive literature describing chemical synthetic approaches to generate the kind of molecules produced by these species as, at least in the systems that have been studied, *in vivo* production is far below human demand[6]. Historical technological challenges to high quality assembly of complete nuclear genomes have resulted in fragmented drafts of limited use due to the exclusion of repetitive sequences and the lack of information on genome structure. Furthermore, the paucity of genome level data for Apocynaceae precludes a deeper understanding of MIA pathway evolution hindering the elucidation of remaining pathway enzymes and the potential to improve MIA availability *in planta* or *in vitro*. To address this, we sequenced the nuclear genome of *Rhazya stricta* and present this high quality assembly in comparison with that of coffee (Rubiaceae, *Coffea canephora* Pierre ex A. Froehner, Gentianales) and others to investigate the evolution of genome-scale features. Using genomic and transcriptomic sequences from Rauvolfioideae along with species in the order Gentianales and beyond, we identified homologs of characterized MIA pathway genes revealing the lineage specific nature of their evolution.

Results and Discussion

We sequenced three Illumina libraries yielding nearly 400 million reads to generate approximately 112X coverage of the *R. stricta* ~274 Mb genome. To achieve the highest quality draft with minimal fragmentation we added PacBio sequencing, (>550K reads; 10X coverage), and isolated nuclei for optical mapping (Supplementary Table S1). The final assembly contains 980 scaffolds with a maximum scaffold length of 16.4 Mb and N50 >5.5 Mb. Scaffolds of greater than 200 Kb (113) comprise 93% of the genome sequence (Supplementary Table S2, Supplementary Table S3). Comparison to assembly statistics reported for three recently published nuclear genomes demonstrates the high quality of the *Rhazya* draft (Supplementary Table S4). A comprehensive listing of statistics for 34 non-model plant nuclear genomes was recently reported by Unamaba *et al.*[7].

The transcriptome-guided annotation of the *R. stricta* genome identified 21,164 protein-coding genes (Supplementary Table S5), 60% of which were assigned UniProt identifiers. Comparison to a set of core eukaryotic sequences (CEGMA)[8] demonstrated that the *R. stricta* assembly captured 98% of the expected genes. The genome contains 16% transposable elements, of which 88% are long terminal repeat (LTR) retrotransposons comprising predominantly Copia and Gypsy superfamily members. Our list of repeated elements is not exhaustive and the actual value for repeat content is likely higher (Supplementary Table S6). All functionally annotated transcripts, as described in the methods section, were assigned gene names, putative functions, gene ontology (GO) terms and enzyme codes by homology search (BLAST) against the UniProt database.

A metabolic pathway database, RhaCyc, was created for *Rhazya* using the PathoLogic component of Pathway Tools (http://bioinformatics.ai.sri.com/ptools/). This program inferred metabolic pathways by comparing the annotated *Rhazya* genes to the MetaCyc 2.0 reference pathway database (http://www.metacyc.org). The RhaCyc database contains information on 3380 enzymes, 2828 compounds, and 458 pathways and is available as a community resource (http://rhacyc.icmb.utexas.edu:1555/). Links to UniProt identifiers and GO terms are embedded with each annotated sequence in RhaCyc.

The high quality draft of the *Coffea canephora* nuclear genome[9] was used to examine structural changes between the two Gentianales families and relative to the ancestral post-γ eudicot arrangement as represented by *Vitis*[10] (Fig. 1; Supplementary Fig. S1). As in *Coffea*, there was no evidence to suggest a subsequent whole-genome polyploidization in the *Rhazya* lineage. Comparison of adjacencies within each of the *Rhazya* scaffolds derived from different ancestral chromosomes and those within the *Coffea* chromosomes allowed us to identify most of the major gene order changes in the extant genomes. While the remaining *Rhazya* scaffolds each tend to reflect a part of only a single ancestral chromosome, they may contain some smaller rearrangements that went undetected. With the available data we detected limited rearrangement between the core eudicot ancestor and the Gentianales ancestor, and between the latter and *Coffea*. In the *Rhazya* lineage, however, many more structural rearrangements were inferred.

To explore the evolution of MIA pathway genes we assembled a translated subject database containing transcriptomic and genomic data for sequences from *Rhazya* and five additional species: two Rauvolfioideae (*C. roseus*, *R. serpentina*), *C. canephora* (Rubiaceae), a non-Gentianales asterid (*Solanum lycopersicum* L.) and an alkaloid producing rosid (*Theobroma cacao* L. cv Matina). For homolog identification we queried the database using previously characterized MIA protein sequences from *C. roseus* and *R. serpentina*, which produce alkaloids currently employed in cancer chemotherapy and as antiarrhythmic drugs, respectively. Putative homologs for each protein were aligned and used to generate phylogenies (Supplementary Fig. S2) for 21 genes (Supplementary Table S7). Of these, eight trees were reduced and restricted to the relevant clades as described in

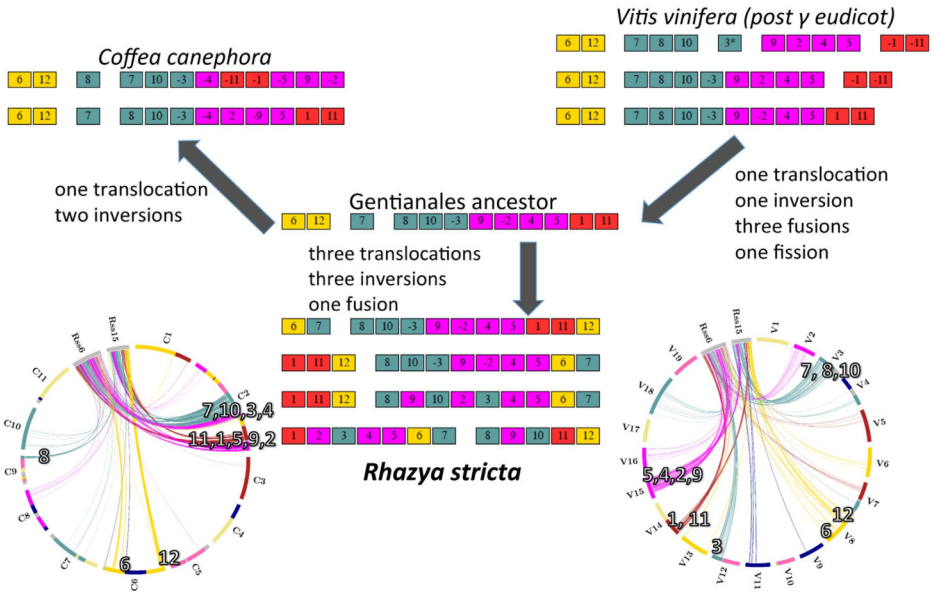


Figure 1. Divergent structural evolution in *Rhazya stricta*. Genome level rearrangements leading from the ancestral core eudicot, represented by *Vitis* (upper right hand corner) to *Rhazya* and *Coffea*. Each linear arrangement represents successive steps from the eudicot ancestor toward the Gentianales ancestor. An arrow leads from the last of these stages to the relevant portion of the Gentianales ancestor. From this ancestor, two arrows indicate the divergence of *Coffea* and *Rhazya*, and successive rearrangements within each lineage are again indicated by linear arrangements of colored blocks. Inferred rearrangements between lineages are given at the arrows. Extant genomes carry the species name. The circular diagrams at the lower left and right represent the current orthologies between *Rhazya* superscaffolds 6 and 15, and *Coffea* and *Vitis*, respectively. Syntenic blocks are colored according to the 21 ancestral core eudicot chromosomes[33] and do not represent biologically significant units.

methods and reconciled with the species tree, depicting the history of duplication and loss for each (Figs 2 and 3, Supplementary Fig. S3, Supplementary Table S8).

Generally, the families that include the eight characterized MIA genes underwent expansion as is seen in other gene families whose members participate in secondary metabolism[2] (Supplementary Table S8). Where genome level data, and therefore information about genetic loci, were available gene family expansion appears to have been facilitated by localized and/or tandem duplications. The two Gentianales taxa, *Coffea* and *Rhazya*, each contain proximally arrayed homologs. Within each species some homologs are grouped in distinct clades while others are dispersed across the phylogeny suggesting different evolutionary trajectories for these sequences. One example can be seen in *Coffea*. The full tree generated from sequences extracted using *C. roseus* Strictosidine β-glucosidase (SGD) as a query contains 157 β-glucosidase superfamily members (Supplementary Fig. S2), 17 of which are proximally arrayed on *Coffea* chromosome two. Of those, eight chromosome two sequences were retained in the reconciled subtree (Supplementary Fig. S3). The duplications giving rise to five glucosidases (Cc02_g27160, g27180, g27190, g27230, g27290; highlighted in brown) likely occurred prior to the divergence of Gentianales while more recent duplications gave rise to three homologs that form a clade containing *Coffea* sequences exclusively (Cc02_g30420, g30490, g30540; highlighted in beige).

For SGD, as for all of the experimentally confirmed MIA sequences analyzed, the *C. roseus* and/or *R. serpentina* sequences downloaded from Genbank were in derived positions in the phylograms. The clades that contained the known (query) sequences were restricted to the Rauvolfioideae, however some clades also included coffee sequences following the pattern of diversification within the order and reflecting the evolution of MIA diversity and complexity.

Our analyses placed two *Rhazya* sequences in the clade adjacent to the GenBank SGD sequences, along with a single sequence extracted from the *Rauvolfia* transcriptome (Fig. 2, Supplementary Fig. S2 and S3). However, SGD was not identified and annotated in the *Rhazya* genome. Looking more closely at these homologs, we aligned the *Rhazya* and *Rauvolfia* sequences within and between the clades. The *Rauvolfia* sequence grouping with the two *Rhazya* sequences was identified as Raucaffricine-O-β-D-glucosidase (RG; EC 3.2.1.125). The two *Rhazya* sequences and RG were 76% and 77% identical whereas identities to *Rauvolfia* SGD were 58% and 57% at the amino acid level.

Thus it appears that the *R. stricta* genome may not encode SGD, nonetheless our own mass spectrometry analysis (Supplementary Table S9) along with decades of reports demonstrate the production of MIAs that derive from the reactive strictosidine aglycone, a product of SGD activity. The presence of a close homolog of RG,

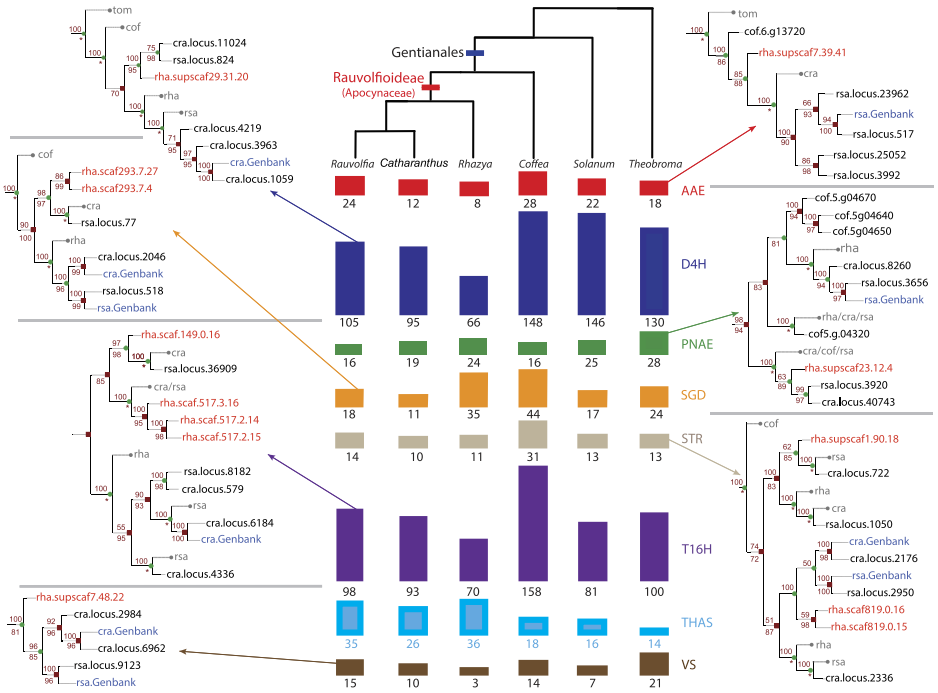


Figure 2. Expansion, loss and duplication of selected monoterpenoid indole alkaloid (MIA) sequences across Gentianales. The cladogram (top, center) depicts relationships among six eudicots based on the Angiosperm Phylogeny Group (http://www.mobot.org/MOBOT/research/APweb/). Histograms below each species indicate the number of sequences returned in BLAST searches to identify homologs for each of the eight gene families. The maximum likelihood trees along the left and right show subclades of reconciled gene phylogenies that include known homologs in either *Catharanthus* (cra.Genbank) or *Rauvolfia* (rsa.Genbank) for seven of the eight MIA genes (THAS is shown in Fig. 3). Complete reconciled phylogenies for each of the eight sequences are shown in Supplementary Fig. S3. cra, *Catharanthus roseus*; cof, *Coffea canephora*; rha, *Rhazya stricta*; tom, *Solanum lycopersicum*; rsa, *Rauvolfia serpentina*. Blue font indicates GenBank sequences and red font indicates *Rhazya stricta* sequences. Red square = gene duplication; green circle = speciation event; gray circles = gene loss. Values above branches indicate bootstrap support for the clade, numbers below nodes give bootstrap support for the event (duplication or speciation) at the node. Bootstrap values less than 50 are not shown; inferred loss events are indicated with an asterisk and do not have associated bootstrap values.

however, suggests another route to downstream intermediates and ultimately MIA end products. Raucaffricine, the major substrate of RG, is hydrolyzed to vomilenin, an intermediate in the ajmaline pathway[11]. While SGD is unable to hydrolyze raucaffricine, RG can catalyze the conversion of strictosidine[12], and may fill that role in *R. stricta* cells. Identification and annotation of the SGD gene in a low quality nuclear genome draft from *C. roseus* proved challenging[13], however, given the depth and completeness of our assembly it is unlikely that a *Rhazya* SGD sequence went undetected. Recently a similar approach used to search for MIA pathway genes in *Amsonia hubrichtii* (Apocynaceae, Rauvolfioideae) failed to detect a high identity homolog of the *C. roseus* SGD. A single transcript was designated as encoding a putative SGD with 57% amino acid identity to the *Catharanthus* sequence[14], suggesting that another β-glucosidase may fill the role of SGD in this species.

The incredible diversity of MIAs in the Rauvolfioidae is facilitated through the hydrolysis of strictosidine to its aglycone, which may then be converted into different structural classes and elaborated by myriad reactions, many of which have been characterized, at least for *C. roseus*. However the enzymatic activities responsible for directing which structural class of MIA will ultimately derive from the aglycone moiety were largely unknown, until recently. THAS activity (Tetrahydroalstonine synthase) converts the strictosidine aglycone to tetrahydroalstonine, the gateway molecule to other heteroyohimbine type alkaloids such as cathenamine[15,16]. We queried our database with the THAS amino acid sequence, extracted homologs for gene phylogenies, and produced trimmed, reconciled trees. Much like the other MIA pathway genes examined, THAS family members showed a pattern of lineage specific expansion in the Rauvolfioideae and included two sequences from *R. stricta* in the reduced tree (Fig. 3, Supplementary Fig. S2). While the two proximal sequences had only 60% amino acid identity to each other, their identities to the characterized *C. roseus* THAS was greater, at 71% and 75%.

Access to the high quality draft of the *Coffea* nuclear genome, along with drafts representing other lineages outside of Gentianales, has allowed us to use the *Rhazya* nuclear genome comparatively to evaluate lineage specific evolution of MIA pathway sequences. However, limiting these analyses to a single representative from each

F-v

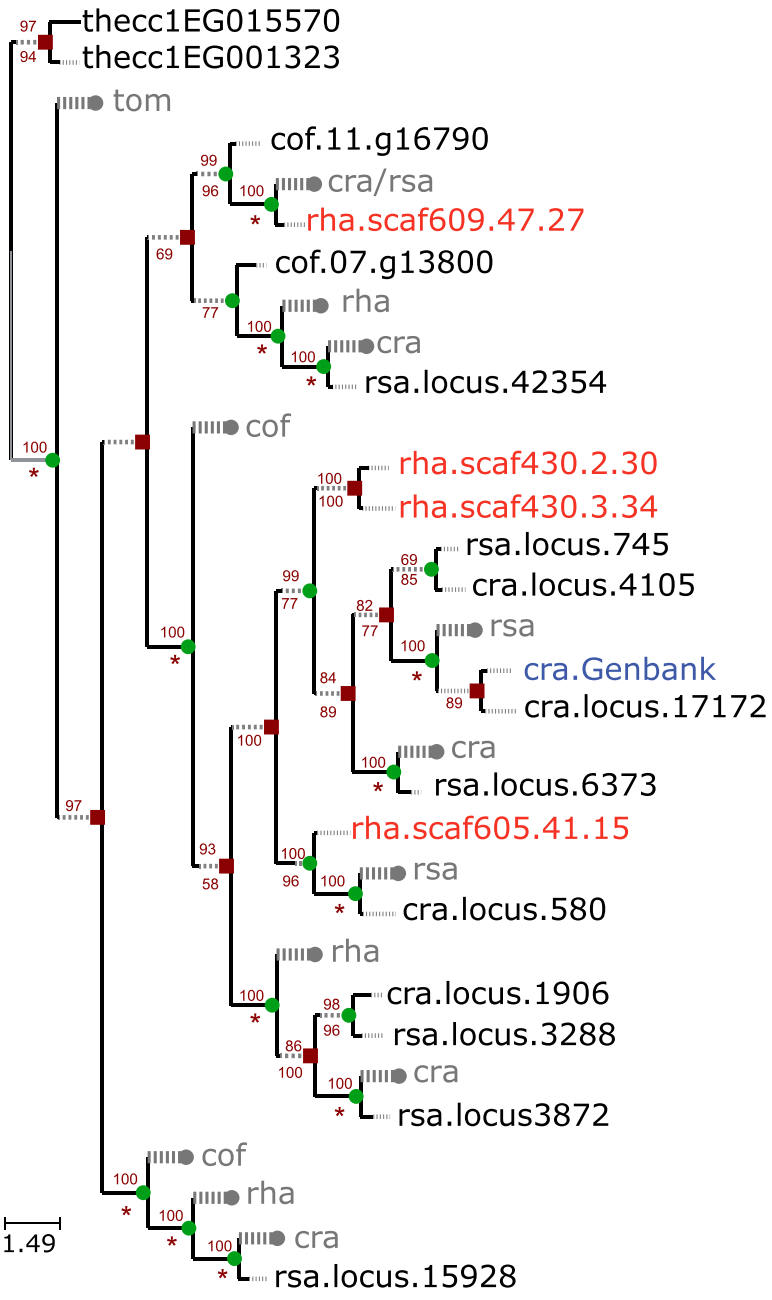


Figure 3. Duplication and loss of Tetrahydroalstonine synthase (THAS) across six angiosperms based on gene tree/species tree reconciliation. Internal and terminal branches in black reflect the amount of change along branch; dashed gray branch extensions are present for branches whose lengths were too short to enable uniform representation with branch support and length information. cra, *Catharanthus roseus*; cof, *Coffea canephora*; rha, *Rhazya stricta*; rsa, *Rauvolfia serpentina*; tom, *Solanum lycopersicum*; the, *Theobroma cacao*. Species indicated in blue font were downloaded from GenBank and those in red are from *Rhazya stricta*. Red square = gene duplication; green circle = speciation event; wide gray dashed line ending with a circle = gene loss. Values above branches indicate bootstrap support for the clade, numbers below nodes give bootstrap support for the event (duplication or speciation) at the node. Bootstrap values less than 50 are not shown; inferred loss events are indicated with an asterisk and do not have associated bootstrap values. Scale bar represents number of amino acid substitutions per site. The maximum likelihood (ML) tree shown is the tree with the best ML score.

lineage necessarily limits the inferences that can be drawn from comparative study. The highly fragmented draft of the *C. roseus* genome[13], while providing a useful resource for certain investigations, was not suitable for inclusion in the gene order comparisons. A more recent approach using single platform nuclear genome sequencing with PacBio generated a high quality draft for the grass genus *Oropetum*[17]. As sequencing technologies continue to improve in terms of both data quality and expense, less fragmented draft genomes for other Apocynaceae should become more tenable for future analyses.

Nonetheless, the gene phylogeny analyses facilitated by the *Rhazya* sequences combined with available data indicate lineage specific expansion resulting in distinct sets of MIA enzyme homologs in the Rauvolfioideae as well as in the Gentianales. The protein sequences clustering with characterized MIA homologs provide an evolutionary insight that could guide full characterization of MIA pathways in related species. One example may be found among the homologs of *C. roseus* Reticuline oxidase (RO, EC 1.21.3.3). This cytochrome P450 flavoprotein is a member of the BBE (Berberine bridge enzyme) family (pfam 08031) that catalyzes carbon-carbon bond formation via oxidative cyclization. An enzyme possessing such activity has been characterized in extracts from *R. serpentina*, but as yet the gene for SBE (Sarpagan bridge enzyme)[18] remains unknown. The inclusion of several *Rauvolfia* and *Rhazya* sequences in the clade containing *C. roseus* RO (Supplementary Fig. S2m) may indicate an avenue for future studies seeking to identify elusive MIA pathway genes.

As more high quality nuclear genome drafts become available we will be able to address genome scale hypotheses of evolution. The rearrangements in *Rhazya* but not *Coffea*, relative to their shared ancestor may suggest that genome level dynamics played a role in the evolution of chemical diversity in the Rauvolfioideae. A critical evaluation of this question would require more than a single representative from each lineage to draw meaningful conclusions.

Materials and Methods

Plant material, DNA and RNA isolation. *Rhazya stricta* seeds were obtained from natural populations collected in the Makkah Province, Saudi Arabia. Seeds were soaked in water overnight at 37 °C then transferred to Profile® Field & Fairway™ inorganic ceramic particles (Buffalo Grove, IL) in a growth chamber (16 h light, 8 h dark 38 °C) for germination. Young leaves were flash frozen in liquid nitrogen for DNA and RNA isolation and stored at −80 °C. Isolation of *R. stricta* nucleic acids was described in Park *et al.*[19]. Young leaves of *R. stricta* were collected at the same location and shipped to INDRAS Private Limited (Hyderabad India) for mass spectrometry analysis of monoterpene indole alkaloid content.

For optical mapping, a megabase DNA isolation protocol was used to obtain intact nuclei from *R. stricta* for embedding in agarose. The protocol was downloaded from the Arizona Genomics Institute at the University of Arizona (http://www.genome.arizona.edu/modules/publisher/item.php?itemid=24).

Genome Sequencing. Illumina (San Diego, CA) DNA libraries were sequenced on the Illumina HiSeq 2500 at the Genome Sequencing and Analysis Facility at the University of Texas at Austin (GSAF). Three different library types were prepared, as required by the AllPaths-LG assembler (http://www.broadinstitute.org/software/allpaths-lg/blog/)[20], an overlapping library with size range of 160–220 base pairs (bp), a mate-pair library ranging from 2–3 kilobases (kb), and a 6 kb mate-pair library. In addition, eight single molecule, real time (SMRT) cells of single-end PacBio RS reads were generated from a 10 kb library at the Interdisciplinary Center for Biotechnology Research, University of Florida, Gainesville. Supplementary Table S1 provides information about the data generated.

Transcriptome Sequencing. Total RNA isolation, library construction, and Illumina sequencing were performed according to Zhang *et al.*[21]. Duplex specific nuclease normalization (Evrogen, Moscow, Russia) of the RNA samples, Illumina RNAseq library construction, and sequencing were carried out at GSAF. Duplex specific nuclease normalization (Evrogen, Moscow, Russia) of the RNA samples, Illumina RNAseq library construction and sequencing were carried out at the GSAF. Raw read output from *R. stricta* RNAseq was deposited in the Small Read Archive (SRA) at the NCBI (accession number SRR1151604)[18]. The RNASeq data was assembled using Trinity (v2013_08_14)[22]. The assembly was annotated using Trinotate (https://trinotate.github.io). Trinotate collects annotation information into one database after running a number of annotation tools: homology search to SwissProt/Uniref90, protein domain identification using HMMER and PFAM, transmembrane domain prediction using tmHMM and comparison to functional categories such as Gene Ontology (GO) databases.

Genome assembly. A sequential pipeline of Illumina and PacBio tools were used to generate the assembly. An initial assembly was generated by Allpaths-LG[20] (release 44837), using reads from three Illumina libraries (fragment library and overlapping 3 kb and 6 kb libraries; described above). Allpaths-LG default parameters were used with ploidy set to two. The assembly comprised 1449 scaffolds with scaffold N50 of 1.0 Mb. PacBio reads were added to the assembly for scaffolding using AHA scaffolder v1.2.0, part of SMRT Analysis 2.0 suite (http://www.pacb.com/products-and-services/analytical-software/devnet/devnet-analysis-tools/). The AHA scaffolder uses BLASR[23] to map long PacBio reads to the assembly without gap filling. Blasr parameters required a minimum seed match of 10 bp (-minMatch 10), minimum identity of 70% (-minPctIdentity 10), reporting 10 best matches (-bestn 10); spliiting of subreads was not permitted (-noSplitSubreads). Parameters for the AHA hybrid assembly were varied iteratively and are provided in Supplementary Text S1. Further alignment of long PacBio reads with the draft assembly to close or improve gaps was carried out in PBJelly (v12.9.14; http://sourceforge.net/projects/pb-jelly/)[24]. The protocol file used to run all PBJelly stages (setup, mapping, support, extraction, assembly, output) is provided in Supplementary Text S1.

Rhazya nuclei embedded in agarose and the sequence of these 232 scaffolds were provided to OpGen (Gaithersburg, MD) for optical mapping. OpGen generated 11 high density Argus® MapCards using the BamHI

restriction enzyme to obtain single molecule restriction maps for the *Rhazya* genome. Utilizing the initial *R. stricta* assembly and OpGen's Genome Builder™ tool, OpGen compared the single molecule restriction data to the initial assembly to facilitate joining of scaffolds. If an overlap was detected when aligning the original *Rhazya* scaffolds to OpGen's extension, OpGen assigned join scores. A join score of 9.0 or above indicated reliable joins. The 232 scaffolds were reduced to 113 scaffolds and the scaffold N50 increased from 1.45 Mb to 5.55 Mb. The whole genome map statistics provided by OpGen under a fee-for-service contract are available in Supplementary Table S3.

Genome annotation. The MAKER2 annotation pipeline (v2.28b; https://www.biostars.org/p/45281/)[25] was employed for genome annotation. RepeatMasker (v3-3-0; http://www.repeatmasker.org)[26] was used to mask all repeat classes from Repbase (http://www.girinst.org/repbase/), and low complexity sequences. The RepeatRunner webserver (http://www.yandell-lab.org/software/repeatrunner.html)[27] was used to identify transposable elements based on the RepeatRunner protein database.

The Trinity-assembled transcripts were used to query the UniProt database (http://www.uniprot.org). Identified ESTs and protein sequences were aligned to the *Rhazya* genome using BLASTN and BLASTX (blast 2.2.27+)[28], respectively. Exonerate (v3-3-0; http://www.ebi.ac.uk/about/vertebrate-genomics/software/exonerate)[29] was used to realign sequences identified by BLAST around splice sites to more accurately detect gene models. Est2genome (http://emboss.sourceforge.net/apps/release/6.6/emboss/apps/est2genome.html) was used for polishing the EST hits and protein2genome, part of Exonerate, was used for polishing the protein hits.

SNAP *ab initio* gene predictor (v.2006-07-28; http://korflab.ucdavis.edu/software.html)[30] was used to predict gene models. SNAP is an easy to use, easy to train tool that is supported by the MAKER2 pipeline. The program was trained by running MAKER2 twice. In the first run, gene models were produced exclusively from EST and protein homology evidence. These gene models were used to train SNAP, and MAKER2 was rerun allowing for *ab initio* predictions using the newly trained gene predictor.

MAKER2 integrated all the EST alignments, protein alignments and *ab initio* predictions and updated information such as splice sites and UTR locations based on all the evidence to generate a filtered set of annotations. In addition to a final list of filtered and modified gene models, a list of rejected *ab initio* predictions were also generated for further detection of missing genes. All parameters used to run the various stages of MAKER2 annotation are provided in Supplementary Text S1

An additional 34,232 proteins were predicted by SNAP. These putative protein sequences were used to query a reference database consisting of annotated genes from *Arabidopsis thaliana*, *Solanum lycopersicum* (tomato), *Vitis vinifera* (grape), *Coffea canephora* (coffee), *Catharanthus roseus*, *Rauvolfia serpentina* and *Theobroma cacao* (>50% identity, evalue >0.000). The MAKER2 validated genes were also used to query the same reference database with the same identity and evalue requirements. If a SNAP predicted gene yielded a qualified hit to a gene in the reference database that had not been previously identified among the 17,041 validated genes, it was included in the final set of genes (4123 additional genes, Supplementary Table S5).

Verification of genome assembly and annotation. The assembled *Rhazya* genome was checked for completeness by using CEGMA 2.5 (Core Eukaryotic Genes Mapping Approach)[8]. CEGMA consists of a core set of eukaryotic genes that are highly conserved across eukaryotic taxa. The CEGMA genes were mapped to the *Rhazya* genome and a report indicating the completeness of the genome was generated.

Functional categorization of genome. All *Rhazya* transcripts were functionally annotated and were assigned gene names, functions, generic Gene Ontology (GO) terms and Enzyme codes (EC) by blasting to the UniProt protein database using BLAST settings (BLASTX, report 1 hit, maximum e-value $1E^{-6}$, percent identity greater than 40).

Metabolic pathway prediction. A metabolic pathway database was created for *Rhazya* using the PathoLogic component of Pathway Tools (v18.0; http://brg.ai.sri.com/ptools/)[31]. This program inferred metabolic pathways by comparing the annotated *Rhazya* genes to the MetaCyc reference pathway database (v.2.0; http://www.metacyc.org)[32]. MetaCyc contains experimentally elucidated, non-redundant metabolic pathways for multiple organisms and serves as a high quality reference for predicting pathways specific to an organism of interest. It also contains information about the enzymes, chemical compounds, reactions and genes related to these metabolic pathways.

The Pathologic Pathway Predictor algorithm consists of two phases. In the reactome inference phase, the predictor evaluates every annotated gene product in the genome and infers the reactions catalyzed by that gene product. The reactions catalyzed by a gene product are inferred from three information fields present in the input: the EC number, gene product name (enzyme name), and Gene Ontology (GO) terms. In the pathway inference phase, the tool infers the metabolic pathways present in the organism based on the set of catalyzed reactions. A pathway is inferred if it is not missing more than one reaction, or if a reaction that is unique to that pathway is present in the organism of interest. Nucleotide and amino acid sequence information is not considered by Pathologic in making these inferences. RhaCyc is available as a community resource at http://rhacyc.icmb.utexas.edu:1555/.

Structural evolution analysis. We used SynMap (with default parameters) in the CoGe platform (https://genomevolution.org/CoGe/; https://www.genomevolution.org/CoGe/SynMap.pl), to find syntenic blocks between *Rhazya* and *Coffea* (Rubiaceae, Gentianales). A whole genome triplication, discovered by Jaillon *et al.*[10] while sequencing the *Vitis vinifera* genome, is broadly accepted to have occurred at the root of the core eudicots. This polyploidization, known as γ(gamma), produced a twenty-one chromosome genome from the seven chromosome ancestor. It is not too difficult to reconstruct the general structure of the immediate post-γ chromosomes

since they differ from the nineteen-chromosome *Vitis* genome in only five or six large rearrangements. This baseline genome is the ultimate resource for deducing gene-order change in the core eudicots[33].

Within each pair of genomes among *Rhazya*, *Vitis* and *Coffea*, SynMap synteny blocks together with overall gene similarity and Ks scores were used to establish thresholds for distinguishing duplicate gene pairs due to speciation (orthologs) which were retained for our analysis, from the older, weaker and sparser gene pairs due to the gamma triplication ("out-paralogs"), which were discarded. The coloring of the blocks was according to the 21 ancestral core eudicot chromosomes, as published by Zheng *et al.*[33].

The circular display of homologies arrayed on the *Rhazya* scaffolds and *Coffea/Vitis* chromosomes were drawn with CIRCOS (http://circos.ca/)[34].

The Bergeron *et al.*[35] algorithm for double-cut-and-join (DCJ) was used to finding the shortest path between two genomes. Xu's 2008 median solver for DCJ[36] was employed to infer the Gentianales ancestor.

Gene family analysis. The subject database comprised protein sequences for six taxa. The species selected for inclusion in the MIA gene family analyses were required to meet certain criteria. The Apocynaceae taxa examined were required to produce MIAs, which restricted the sampling to the subfamily Rauvolfioideae[37]. The other three taxa included have complete genomes, which provided important information about isoforms so that only one isoform would be included in the gene tree analyses. In particular, coffee was selected because it is in the Gentianales, produces MIAs and its complete genome is available[9]. Tomato was included as an asterid outgroup to the Gentianales as its transcriptome and complete nuclear genome were available[38]. Likewise, *Theobroma cacao*, which produces non-MIA alkaloids, was included as a rosid outgroup to the asterid clade and had the required data available[39]. Protein sequences were downloaded for five species and combined with *Rhazya* sequences. Sequence data was accessed from the following sites: *Theobroma cacao* (rosid; http://www.cacaogenomedb.org/Tcacao_genome_v1.1), *Solanum lycopersicum* (non-Gentianales asterid; https://solgenomics.net/gb2/gbrowse/ITAG2.3_genomic/), *Coffea canephora* (Rubiaceae, Gentianales; http://coffee-genome.org), *Catharanthus roseus* and *Rauvolfia serpentina* (Apocynaceae, Gentianales; http://medicinalplantgenomics.msu.edu). Sequences for each of 21 characterized proteins involved in the MIA pathways of *Catharanthus roseus* (L.) and/or *Rauvolfia serpentina* (Supplementary Table S7) were used to query the database using BLAST (e-value $1e^{-3}$) and sequences with identities $\geq$ 35% were extracted. A single isoform was retained for each unique genetic locus and the sequences were aligned with the GenBank homolog (query) using MUSCLE[40] as implemented in Geneious (www.biomatters.com). Sequences covering $<$ 50% of the length of the GenBank homolog were removed. If the database sequence extended beyond the GenBank homolog sequence, the extra sequence was trimmed. Following removal of sequences, alignments were rerun with MUSCLE for maximum likelihood (ML) analysis with PhyML[41] in Geneious. Muscle settings included eight iterations with sequences grouped by similarity using the kmer 6_6 for the first iteration and pctid_kimura distance for all subsequent iterations and the neighbor joining clustering method. ML analyses used the Le Gascuel substitution model and Subtree Pruning and Regrafting topology search options.

A species tree was drawn in FigTree (see top Fig. 2) and follows APG (http://www.mobot.org/MOBOT/Research/APweb/) relationships. For reconciliation of gene trees with the species tree eight datasets were selected representing early and late stage enzymes in the MIA pathways of *C. roseus* and *R. serpentina*. For these eight (bold in Supplementary Table 7), an iterative series of alignment and ML tree generation was used to define a well-supported clade that included the GenBank homolog. For each gene family, sequences for the clade of interest were extracted and aligned with MUSCLE and a preliminary unrooted binary phylogenetic tree was computed using PhyML version 20131022 (GTR + Γ(4) model). PhyML parameters were set to optimize branch length, tree topology and substitution rate. A hundred non-parametric bootstraps were performed to assess clade support in the resulting tree.

The initial trees were refined, rooted and internal nodes labeled as duplication or speciation nodes for reconciliation with the species tree. Weakly supported branches, i.e. branches with bootstrap values under a given threshold, were removed leading to an unrooted non-binary tree. Different thresholds for the bootstrap value (20, 30, 40, 50, 60, 70, 80 and 90) were tested. For each obtained tree, all nodes were successively considered as potential roots. Then for each rooted non-binary tree, all binary refinements minimizing the reconciliation cost, i.e. the inferred number of duplication and loss operations according to the species tree, were computed using ProfileNJ[42]. Each tree output by ProfileNJ is a rooted binary tree with each internal node labeled as a duplication (red square) or speciation (green circle), and each lost gene reported on the tree by grafting a new branch (gray dashed branches) and leaf labeled with abbreviated names of the genomes affected by the loss. This was the set of trees used as replicates for node bootstrapping.

Per-site likelihood for all trees was computed with RaxML[43] version 8.1.3, and SH and AU tests were performed with CONSEL[44]. Then, manual filtering was done to retain only the best trees according to both the likelihood score and the reconciliation cost (Supplementary Table S8). A consensus tree was computed from all retained best trees using *consense,* part of the PHYLIP package (http://evolution.genetics.washington.edu/phylip.html), with extended majority rule for identifying most frequent subtrees. Among the best trees, the one sharing the closest topology to the consensus tree was chosen. As branch length and bootstrap values were lost for branches contracted with ProfileNJ, they were recomputed for each tree. Finally, a node support was computed as the frequency of replicates having that node, i.e. with the same incoming edge and the same inferred event (duplication or speciation).

Data Availability. This Whole Genome Shotgun project has been deposited at DDBJ/ENA/GenBank under the accession MEJB00000000. The version described in this paper is version MEJB01000000. All phylogenetic alignments are available upon request to TAR or RJK.

References

1. Luca, V. D., Salim, V., Atsumi, S. M. & Yu, F. Mining the biodiversity of plants: A revolution in the making. *Science* **336,** 1658–1661 (2012).
2. Chae, L., Kim, T., Nilo-Poyanco, R. & Rhee, S. Y. Genomic signatures of specialized metabolism in plants. *Science* **344,** 510–513 (2014).
3. Szabó, L. F. Rigorous biogenetic network for a group of indole alkaloids derived from strictosidine. *Molecules* **13,** 1875–1896 (2008).
4. Gilani, S. A., Kikuchi, A., Shinwari, Z. K., Khattak, Z. I. & Watanabe, K. N. Phytochemical, pharmacological and ethnobotanical studies of *Rhazya stricta* Decne. *Phytother. Res.* **21,** 301–307 (2007).
5. Smith, G. N. Strictosidine: a key intermediate in the biogenesis of indole alkaloids. *Chem. Commun. Lond.* 912–914 (1968).
6. Sears, J. E. & Boger, D. L. Total synthesis of vinblastine, related natural products, and key analogues and development of inspired methodology suitable for the systematic study of their structure–function properties. *Acc. Chem. Res.* **48,** 653 (2015).
7. Unamaba, C. I. N., Nag, A. & Sharma R. K. Next generation sequencing technologies: The doorway to the unexplored genomics of non-model plants. *Front. Plant Sci.* **6,** 1074 (2015).
8. Parra, G., Bradnam, K. & Korf, I. CEGMA: a pipeline to accurately annotate core genes in eukaryotic genomes. *Bioinformatics* **23,** 1061–1067 (2007).
9. Denoeud, F. *et al.* The coffee genome provides insight into the convergent evolution of caffeine biosynthesis. *Science* **345,** 1181–1184 (2014).
10. Jaillon, O. *et al.* The grapevine genome sequence suggests ancestral hexaploidization in major angiosperm phyla. *Nature* **449,** 463–467 (2007).
11. Ruppert, M., Panjikar, S., Barleben, L. & Stöckigt, J. Heterologous expression, purification, crystallization and preliminary X-ray analysis of raucaffricine glucosidase, a plant enzyme specifically involved in *Rauvolfia* alkaloid biosynthesis. *Acta Crystallograph. Sect. F Struct. Biol. Cryst. Commun.* **62,** 257–260 (2006).
12. Warzecha, H., Obitz, P. & Stöckigt, J. Purification, partial amino acid sequence and structure of the product of Raucaffricine-O-β -d-glucosidase from plant cell cultures of *Rauwolfia serpentina*. *Phytochemistry* **50,** 1099–1109 (1999).
13. Kellner, F. *et al.* Genome-guided investigation of plant natural product biosynthesis. *Plant J.* **82,** 680–692 (2015).
14. Xiao, M. *et al.* Transcriptome analysis based on next-generation sequencing of non-model plants producing specialized metabolites of biotechnological interest. *J. Biotechnol.* **166,** 122–134 (2013).
15. O'Connor, S. E. & Maresh, J. J. Chemistry and biology of monoterpene indole alkaloid biosynthesis. *Nat. Prod. Rep.* **23,** 532–547 (2006).
16. Stavrinides, A. *et al.* Unlocking the diversity of alkaloids in *Catharanthus roseus*: Nuclear localization suggests metabolic channeling in secondary metabolism. *Chem. Biol.* **22,** 336–341 (2015).
17. VanBuren, R. *et al.* Single-molecule sequencing of the desiccation-tolerant grass *Oropetium thomaeum*. *Nature* doi: 10.1038/nature15714 (2015).
18. Schmidt, D. & Stöckigt, J. Enzymatic formation of the sarpagan-bridge: A key step in the biosynthesis of sarpagine- and ajmaline-type alkaloids. *Planta Med.* **61,** 254–258 (1995).
19. Park, S. *et al.* Complete sequences of organelle genomes from the medicinal plant *Rhazya stricta* (Apocynaceae) and contrasting patterns of mitochondrial genome evolution across asterids. *BMC Genomics* **15,** 405 (2014).
20. Ribeiro, F. *et al.* Finished bacterial genomes from shotgun sequence data. *Genome Res.* **22,** 2270–7 (2012).
21. Zhang, J., Ruhlman, T. A., Mower, J. P. & Jansen, R. K. Comparative analyses of two Geraniaceae transcriptomes using next-generation sequencing. *BMC Plant Biol.* **13,** 228 (2013).
22. Grabherr, M. *et al.* Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nature Biotechnol.* **29,** 644–652 (2011).
23. Chaisson, M. & Tesler, G. Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC Bioinformatics* **13,** 238 (2012).
24. English A. C. *et al.* Mind the gap: upgrading genomes with Pacific Biosciences RS long-read sequencing technology. *PLoS One* **7,** e47768 (2012).
25. Holt, C. & Yandell, M. MAKER2: an annotation pipeline and genome database management tool for second-generation genome projects. *BMC Bioinformatics* **12,** 491 (2011).
26. Tarailo-Graovac, M. & Chen, N. Using RepeatMasker to identify repetitive elements in genomic sequences. *Curr. Protoc. Bioinformatics* **25,** 4.10.1–4.10.14 (2009).
27. Smith, C. D. *et al.* Improved repeat identification and masking in dipterans. *Gene* **389,** 1–9 (2007).
28. Camacho, C. *et al.* BLAST+: architecture and applications. *BMC Bioinformatics* **10,** 421 (2008).
29. Slater, G. S. & Birney, E. Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics*. **6,** 31 (2005).
30. Korf, I. Gene finding in novel genomes. *BMC Bioinformatics* **5,** 59 (2004).
31. Karp, P. D., Latendresse, M. & Caspi, R. The Pathway Tools pathway prediction algorithm. *Stand Genomic Sci* **5,** 424–429 (2011).
32. Caspi, R. *et al.* The MetaCyc Database of metabolic pathways and enzymes and the BioCyc collection of Pathway/Genome Databases. *Nucleic Acids Res*. **44,** D471–80 (2014).
33. Zheng, C., Chen, E., Albert, V. A., Lyons, E. & Sankoff, D. Ancient eudicot hexaploidy meets ancestral eurosid gene order. *BMC Genomics* **14,** S3 (2013).
34. Krzywinski, M. *et al.* Circos: An information aesthetic for comparative genomics. *Genome Res*. **19,** 1639–1645 (2009).
35. Bergeron, A., Mixtacki, J. & Stoye, J. In *Algorithms in Bioinformatics* (eds Bücher, P. & Moret, B. M. E.) **4175,** 163–173 (Springer Berlin Heidelberg, 2006).
36. Xu, A. W. & Sankoff, D. In *Algorithms in Bioinformatics* (eds Crandall, K. A. & Lagergren, J.) 25–37 (Springer Berlin Heidelberg, 2008).
37. St-Pierre, B. *et al.* Deciphering the evolution, cell biology and regulation of monoterpene indole alkaloids. *Adv. Bot. Res*. **68,** 73–109 (2013).
38. The Tomato Genome Consortium. The tomato genome sequence provides insights in to fleshy fruit evolution. *Nature*. **485,** 635–641 (2012).
39. Argout, X. *et al.* The genome of *Theobroma cacao*. *Nature Genet.* **43,** 101–108 (2011).
40. Edgar, R. C. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*. **32,** 1792–1797 (2004).
41. Guindon, S. & Gascuel, O. A simple, fast and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst. Biol.* **52,** 696–704 (2003).
42. Noutahi, E. *et al.* Efficient gene tree correction guided by species and synteny evolution. at https://hal.archives-ouvertes.fr/hal-01162963 (2015)
43. Stamatakis, A. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics* **30,** 1312–1313 (2014).
44. Shimodaira, H. & Hasegawa, M. CONSEL: for assessing the confidence of phylogenetic tree selection. *Bioinformatics* **17,** 1246–1247 (2001).

F-x

Acknowledgements

The authors acknowledge financial support from Vice President for Educational Affairs Prof. Dr. Abdulrahman O. Alyoubi at King Abdulaziz University (KAU) and from the Deanship of Scientific Research represented by the Unit of Strategic Technologies Research at KAU through project Number 431/008-D. We thank the Texas Advanced Computing Center (TACC) for access to supercomputers, the Plant Resources Center for storage of voucher specimens and Scott Hunicke-Smith at the Genome Sequence and Analysis Facility (GSAF) at the University of Texas at Austin.

Author Contributions

J.S.M.S., R.K.J., T.A.R., D.A., M.A.K. and N.A.B designed and conceived research; T.A.R germinated seeds and harvested plants, isolated DNA, RNA and nuclei, assisted in data analyses and figure production, and wrote the manuscript; D.A. assembled and annotated the genomic and transcriptomic data, performed blast comparisons for gene trees and developed RhaCyc; V.C., E.N. and N.E.-M. performed alignments and phylogenetic analyses on 8 selected MIA genes and gene tree/species tree reconciliation; S.P. assisted in transcriptome assembly and phylogenetic analyses; M.J.S., M.N.B., N.H.H., A.Y.O. and A.L.A.-M. acquired seed and leaf material of *Rhazya stricta*; C.Z. and D.S. performed gene order analyses; R.K.J. performed alignment and phylogenetic analyses of 21 MIA genes. All authors read and approved the final version of the manuscript.

Additional Information

Supplementary information accompanies this paper at http://www.nature.com/srep

Competing financial interests: The authors declare no competing financial interests.

How to cite this article: Sabir, J. S.M. *et al.* The nuclear genome of *Rhazya stricta* and the evolution of alkaloid diversity in a medically relevant clade of Apocynaceae. *Sci. Rep.* **6**, 33782; doi: 10.1038/srep33782 (2016).