Université de Montréal

Approche cartésienne pour le calcul du vent en terrain complexe avec application à la propagation des feux de forêt

par

Louis-Xavier Proulx

Département de mathématiques et de statistique

Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)
en mathématiques appliquées

Août 2010

Université de Montréal

Faculté des études supérieures

Ce mémoire intitulé

Approche cartésienne pour le calcul du vent en terrain complexe avec application à la propagation des feux de forêt

présenté par

Louis-Xavier Proulx

a été évalué par un jury composé des personnes suivantes :

Robert G. Owens

(président-rapporteur)

Anne Bourlioux

(directeur de recherche)

Michel Delfour

(membre du jury)

Mémoire accepté le:
13 janvier 2011

SOMMAIRE

La méthode de projection et l'approche variationnelle de Sasaki sont deux techniques permettant d'obtenir un champ vectoriel à divergence nulle à partir d'un champ initial quelconque. Pour une vitesse d'un vent en haute altitude, un champ de vitesse sur une grille décalée est généré au-dessus d'une topographie donnée par une fonction analytique. L'approche cartésienne nommée *Embedded Boundary Method* est utilisée pour résoudre une équation de Poisson découlant de la projection sur un domaine irrégulier avec des conditions aux limites mixtes. La solution obtenue permet de corriger le champ initial afin d'obtenir un champ respectant la loi de conservation de la masse et prenant également en compte les effets dûs à la géométrie du terrain. Le champ de vitesse ainsi généré permettra de propager un feu de forêt sur la topographie à l'aide de la méthode iso-niveaux. L'algorithme est décrit pour le cas en deux et trois dimensions et des tests de convergence sont effectués.

Mots clés : Approche cartésienne, feux de forêt, propagation, méthode de projection, conservation de la masse.

SUMMARY

The Projection method and Sasaki's variational technique are two methods allowing one to extract a divergence-free vector field from any vector field. From a high altitude wind speed, a velocity field is generated on a staggered grid over a topography given by an analytical function. The Cartesian grid Embedded Boundary method is used for solving a Poisson equation, obtained from the projection, on an irregular domain with mixed boundary conditions. The solution of this equation gives the correction for the initial velocity field to make it such that it satisfies the conservation of mass and takes into account the effects of the terrain. The incompressible velocity field will be used to spread a wildfire over the topography with the Level Set Method. The algorithm is described for the two and three dimensional cases and convergence tests are done.

Key words : Embedded boundary method, wildfires, spread, projection method, mass-consistent model.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES SIGLES ET ABRÉVIATIONS

ANAG : Applied Numerical Algorithms Group

CFD : Computational Fluid Dynamics

EBM : Embedded Boundary Method

FDM : Finite Difference Method

FEM : Finite Element Method

FVM : Finite Volume Method

MAC : Marker-and-Cell

MCM : Mass-Consistent Model

PDE : Partial Differential Equation

REMERCIEMENTS

Je suis sincèrement reconnaissant envers ma directrice de recherche, Anne Bourlioux, pour m'avoir proposé de travailler sur ce nouveau projet. J'aimerais aussi la remercier pour son support financier qui m'a permis de me consacrer pleinement à mes recherches ainsi que pour m'avoir donné l'opportunité de faire un stage de recherche à Berkeley.

J'aimerais aussi transmettre mes remerciements à Phillip Colella pour son acceuil chaleureux au Berkeley Lab et pour avoir mis à ma disposition les outils nécessaires pour poursuivre mes recherches sur ce projet. Je suis également redevable à plusieurs membres de l'équipe du ANAG, plus particulièrement à Terry Ligocki, Dan Graves, Hans Johansen et Jeffrey Johnson qui ont été d'un très grand secours lors de mon apprentissage de EBChombo et surtout lorsqu'il est venu le temps de trouver les *bugs*. Je remercie le Fonds québécois de la recherche sur la nature et les technologies (FQRNT) et le Centre de recherches mathématiques (CRM) pour avoir reconnu la valeur de ce projet et pour m'avoir accordé une bourse de stage international qui m'a assuré une sécurité financière lors de mon séjour à Berkeley.

Un merci spécial à Alexandre Desfossés-Foucault pour ses conseils sur Matlab, ses commentaires et ses corrections sur ce mémoire ainsi que son agréable compagnie au *labomat*. Finalement, un gros merci à tous ceux qui m'ont supporté de près ou de loin ces deux dernières années. Vos encouragements et nos discussions m'ont aidé à mener ce mémoire à terme.

INTRODUCTION

British-Columbia, Alberta and California face each year important forest fires on their territories. Fires are essential for maintaining the diversity and health of forest ecosystems, but they also bring a lot of drawbacks. Many mathematical models are currently used to predict the propagation of fires. More accurate simulations could prevent the negative impacts of wildfires on public health and safety of individuals, and could also decrease the loss of property and natural resources.

Simulation models for the spread of wildfires, such as PROMETHEUS and FARSITE, use the equations of fluid mechanics. Different factors must be taken into consideration when simulating the spread. The weather, the effects of the topography, the different type of fuels and obstacles are a few examples. Adding more parameters and variables to the model increases its accuracy but also the computational cost, hence choices must be made. Most current models tend to use the law of conservation of mass and conservation of momentum, but those models often neglect the effects of the terrain and of the fire as a dilation source term.

The aim of this thesis is to develop the first part of a new model to predict efficiently and accurately the spread of forest fires. The approach chosen here takes into account the topographic effects of the terrain on the wind direction, which is often neglected in current models. The tangential wind to the surface is non-zero since the effects of viscosity, more specifically the friction of the wind field with the terrain, are neglected. The final velocity vector field must only satisfy the law of conservation of mass. Two procedures can be used to transform an initial

vector field into an incompressible vector field. After a short discussion on mass-consistent models, chapter one presents Sasaki's variational technique followed by the projection method. The model will also construct the initial vector field from a high altitude wind. The different possibilities are discussed at the end of the first chapter.

In order to apply the constraint of incompressibility on the wind vector field, a Poisson equation is solved on an irregular domain. In fact, the shape of the topography is *embedded* in the computational domain using a Cartesian grid. This method is called the Embedded Boundary Method and was developed by Phillip Colella and his team at the Lawrence Berkeley National Laboratory (LBNL). This approach is based on the Finite Volume Method and is explained in chapter two for the two-dimensional case. Convergence of the solutions and error analysis are also studied.

In chapter three, the model and the algorithm are generalized to three dimensions. The implementation of the model was done with EBChombo, a software developed by the Applied Numerical Algorithms Group (ANAG) at the Berkeley Lab. The test cases used in the two-dimensional case are generalized for a domain in three dimensions in order of study the convergence of the solutions.

Chapitre 1

A MASS-CONSISTENT MODEL

Models for simulating wind flow fall into two main categories : prognostic models and diagnostic models. The first kind considers time-dependent hydrodynamic equations such as Navier-Stokes to forecast how the wind flow will evolve. These models also include many factors such as turbulence, moisture, momentum and heat. Elaborated models require precise data in order to deliver accurate predictions and such data is not always available. On the other hand, diagnostic models generate wind fields that satisfy specific physical constraints. Models that assure the conservation of mass are typically called mass-consistent models. These are simpler than prognostic models, they require less data and have the big advantage of having a low computational cost.

The goal of this thesis is to construct a predictive mass-consistent model for wildfire propagation. The model will generate a velocity field that will take into account the effects of the topography on the wind flow. A more accurate wind field will lead to better predictions of the fire front spread. Since only the constraint of conservation of mass is considered, the simulations will be faster than models using also the momentum equation and hence be useful for predictions.

Following a description of the problem, two methods which generate mass-consistent flow are presented. Then, a short discussion on the initialization of the vector field reveals the challenges for accurate predictions.

1.1. Formulation of the problem

The model is explained for the two-dimensional case, but it can be easily generalized to three dimensions. Consider a rectangular region in $\mathbb{R}^2$ of length L_x and height L_y. The topographical height above mean sea level is expressed by a function $H : \mathbb{R} \to \mathbb{R}$ depending on the horizontal coordinate x. The computational domain Ω is the region contained between the function $H(x)$ and the top of the rectangular region as shown in figure 1.1. The only information given for the wind is the magnitude of a high altitude horizontal wind $\mathbf{v}_g = (u^g, 0)$ from which an initial velocity vector field $\mathbf{v} = (u, v)$ must be constructed. This initial vector field might not be divergence-free, but since the wind is assumed to be inviscid (no viscosity) it must be tangential to the terrain surface, which means that it satisfies the slip condition $\mathbf{v} \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is the outward normal at the topography surface. We then look for a correction that will transform the vector field $\mathbf{v}$ in an incompressible vector field $\mathbf{v}_d$ which also fulfils the slip condition $\mathbf{v}_d \cdot \mathbf{n} = 0$. Boundary conditions for the sides and top of the domain have to be defined such that the flow will be allowed to go through.

1.1.1. Conservation of mass and incompressibility

In order to get an incompressible vector field $\mathbf{v}_d$, we must apply a constraint to the vector field. In fluid dynamics, the conservation of mass is expressed by the continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{1.1.1}$$

where the operator $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$, $\rho = \rho(x, y, z)$ is the density of the fluid and $\mathbf{v} = (u(x, y, z), v(x, y, z), w(x, y, z))$ where u, v and w are the velocity of the flow in the x, y and z directions. Notice that in our model, the variables ρ and $\mathbf{v}$ do not depend on time, since we are interested in a stationary flow, hence $\partial\rho/\partial t = 0$. The density of the fluid is assumed to be constant everywhere on the domain, so equation (1.1.1) becomes

$$\nabla \cdot \mathbf{v} = 0 \tag{1.1.2}$$

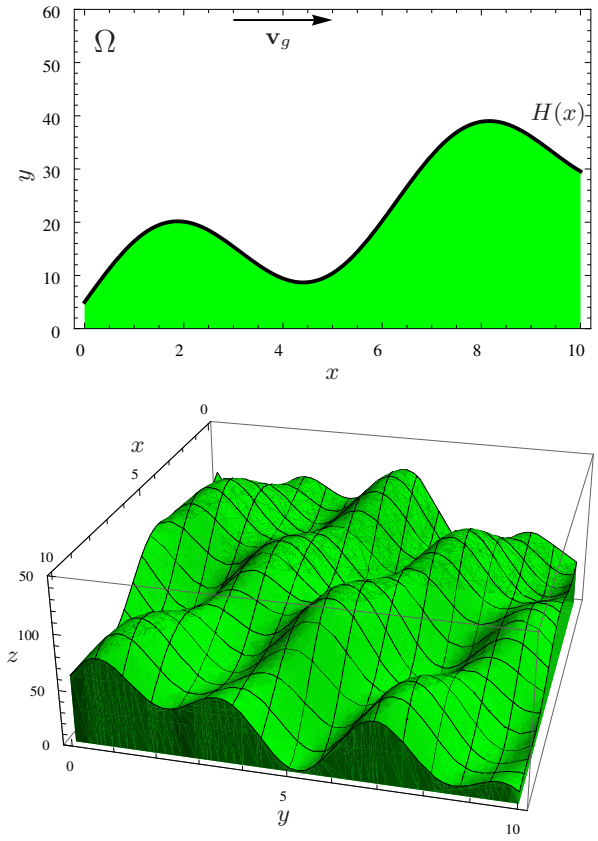


FIG. 1.1. Representation of the computational domain Ω in 2 and 3 dimensions, with the topography given by $H(x)$ and the high altitude wind $\mathbf{v}_g$.

where equation (1.1.2) is the incompressibility constraint on the flow. This condition expresses and guarantees the conservation of mass for our model. Mathematically, it also means that the vector field is divergence-free.

1.2. SASAKI'S VARIATIONAL TECHNIQUE

There are different approaches to apply the constraint of mass conservation on a given vector field. A quick look at the literature in atmospheric sciences shows that most models are based on a variational calculus method developed by Sasaki [**22**]. Ratto et al. [**20**] have reviewed these models which are adapted to take into account certain parameters and features of the flow, but the core of the technique remains the same.

1.2.1. Elliptic equation

Sasaki's variational technique is a natural approach in atmospheric sciences. Meteorologists construct their models with a variety of empirical data, such as wind velocity. The variational method allows them to find, for a set of observed wind data, a minimal correction to adjust the wind such that it will become divergence-free.

In fact, this method minimizes the variation between the adjusted values $\mathbf{v}_d = (u^d, v^d, w^d)$ and the initial values $\mathbf{v} = (u, v, w)$ in a generalized least squares formulation :

$$I(u^d, v^d, w^d) = \int_\Omega \left[\beta_1^2(u^d - u)^2 + \beta_2^2(v^d - v)^2 + \beta_3^2(w^d - w)^2\right] dV \tag{1.2.1}$$

where β_i $(i = 1, 2, 3)$ are the Gauss precision moduli. These weights are used for the calibration of the adjustments of the wind field $\mathbf{v}_d$ with the observed values $\mathbf{v}$. They will be explained in more details in section 1.2.3.

Since we are looking for an adjusted vector field $\mathbf{v}_d$ which fulfils the conservation of mass, we add the constraint given by equation (1.1.2) :

$$G(u_x^d, v_y^d, w_z^d) = \frac{\partial u^d}{\partial x} + \frac{\partial v^d}{\partial y} + \frac{\partial w^d}{\partial z} = 0 \tag{1.2.2}$$

to the functional I :

$$J = \int_\Omega E dV = I + \int_\Omega \lambda G dV \tag{1.2.3}$$

where $\lambda = \lambda(x, y, z)$ is a Lagrange multiplier. Writing explicitly the functional J in equation (1.2.3) we have :

$$J(u^d, v^d, w^d, \lambda) = \int_\Omega \Bigg[\beta_1^2(u^d - u)^2 + \beta_2^2(v^d - v)^2 + \beta_3^2(w^d - w)^2 + \lambda\left(\frac{\partial u^d}{\partial x} + \frac{\partial v^d}{\partial y} + \frac{\partial w^d}{\partial z}\right)\Bigg] dV \tag{1.2.4}$$

We now want to minimize the functional J under the strong constraint of conservation of mass. To achieve this goal, we must look at the first variation of J and find when it is equal to zero : $\delta J = 0$. From calculus of variations,

one knows that the solutions which minimize the functional J satisfy the Euler-Lagrange equations :

$$\frac{\partial E}{\partial f} - \sum_{i=1}^{3} \frac{\partial}{\partial x_i} \frac{\partial E}{\partial(\partial f/\partial x_i)} = 0 \tag{1.2.5}$$

where E is the integrand of the functional J, $f = (u^d, v^d, w^d, \lambda)$ and $(x_1, x_2, x_3) = (x, y, z)$. Solving with the Euler-Lagrange equations (1.2.5), we find

$$u^d = u + \frac{1}{2\beta_1^2}\frac{\partial \lambda}{\partial x} \tag{1.2.6}$$

$$v^d = v + \frac{1}{2\beta_2^2}\frac{\partial \lambda}{\partial y} \tag{1.2.7}$$

$$w^d = w + \frac{1}{2\beta_3^2}\frac{\partial \lambda}{\partial z} \tag{1.2.8}$$

$$\frac{\partial u^d}{\partial x} + \frac{\partial v^d}{\partial y} + \frac{\partial w^d}{\partial z} = 0. \tag{1.2.9}$$

Note that solution (1.2.9) corresponds to the incompressibility constraint (1.1.2). Ratto et al. [**20**] noticed that all mass-consistent models assume the Gauss precision moduli to be equal in the horizontal plane of the euclidean space :

$$\beta_1 = \beta_2. \tag{1.2.10}$$

In fact, important distinctions are seen between the horizontal components and the vertical component of the adjusted vector field with the initial vector field. There is no such big difference between the horizontal directions of the two velocity fields.

Differentiating equations (1.2.6), (1.2.7), (1.2.8) with respect to x, y, z respectively, and substituting the results in equation (1.2.9), this elliptic equation is obtained :

$$\frac{\partial^2 \lambda}{\partial x^2} + \frac{\partial^2 \lambda}{\partial y^2} + \left(\frac{\beta_1}{\beta_3}\right)^2 \frac{\partial^2 \lambda}{\partial z^2} = -2\beta_1^2 \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right). \tag{1.2.11}$$

Using the solution $\lambda(x, y, z)$ of equation (1.2.11), we can correct the initial vector field $\mathbf{v}$ with equations (1.2.6)-(1.2.8) and find the divergence-free vector field $\mathbf{v}_d$. In order to solve the elliptic equation, we need to specify the boundary conditions related to the model.

1.2.2. Boundary conditions

For the variational problem described above, Ishikawa [**13**] gives the associated boundary conditions as :

$$\oint_{\partial\Omega} \delta f \left[\sum_{i=1}^{3} \frac{\partial E}{\partial(\partial f/\partial x_i)} \right] \cdot \mathbf{n} dA = 0 \tag{1.2.12}$$

where $\mathbf{n}$ is the outward normal vector to the domain boundary $\partial\Omega$ and δf an arbitrary first variation of f. Solving the part between the brackets always leads to λ. The integral will be zero if the integrand is zero, hence the boundary conditions become

$$\lambda \delta \mathbf{v}_d \cdot \mathbf{n} = 0 \text{ on } \partial\Omega \tag{1.2.13}$$

where $\delta \mathbf{v}_d$ denotes the first variation of the velocity.

From equation (1.2.13), either the multiplier λ or the normal velocity component variation $\delta \mathbf{v}_d \cdot \mathbf{n}$ must be zero at a boundary. Sherman [**24**] claimed that only one of those two conditions must be imposed at a time, otherwise it would overdetermine the problem and the solution would not be unique. In fact, Núñez et al. [**19**] have shown that the sole properties of J guarantee the existence and uniqueness of a field $\mathbf{v}_d$ that minimizes globally J. Most authors of atmospheric sciences articles have followed the choice adopted by Sherman and her interpretation of the boundary conditions.

For *flow-through* or open boundaries, the appropriate boundary condition is the homogeneous Dirichlet condition,

$$\lambda = 0. \tag{1.2.14}$$

For this condition, the normal derivative of λ might not be equal to zero, which is generally the case. Hence, a non-zero adjustment of the initial velocity component normal to the boundary might occur. Moreover, a constant value of λ at an open boundary also implies that no correction is made for the velocity components in the non-normal direction, since the non-normal derivatives of λ are zero. This property is useful since we want to conserve the magnitude of the wind at the top boundary.

For *no-flow-through* or closed boundaries, the homogeneous Neumann boundary condition,

$$\frac{\partial \lambda}{\partial n} = 0 \tag{1.2.15}$$

is chosen. It implies that there is no adjustment in the normal velocity component, hence the variation of normal velocity is zero :

$$\delta \mathbf{v}_d \cdot \mathbf{n} = 0. \tag{1.2.16}$$

If the initial normal velocity component through the boundary is zero :

$$\mathbf{v} \cdot \mathbf{n} = 0 \tag{1.2.17}$$

the adjusted flow of mass across the boundary is also zero which means that the corrected vector field will satisfy the slip condition. The Neumann condition is then appropriate for the terrain surface boundary. According to Núñez et al. [**19**], a more appropriate boundary condition for the terrain surface would be

$$n_i \frac{1}{\beta_i^2} \frac{\partial \lambda}{\partial x^i} = -\mathbf{v} \cdot \mathbf{n}. \tag{1.2.18}$$

Notice that this boundary condition only coincides with the homogeneous Neumann boundary condition in two cases. First when the topography is flat and the initial velocity field is parallel to the terrain boundary : $\mathbf{v} \cdot \mathbf{n} = 0$. The second case is when all $\beta_i = 1$ and again the initial velocity field is parallel to the boundary. They also mentioned that the homogeneous Neumann condition is inconsistent when using the Finite Element Method, hence FDM should be used with this condition. From equation (1.2.18), we understand why Barnard et al. [**2**] and Ross et al. [**21**] mentioned that applying the closed boundary condition requires that the initial velocity field must respect the slip condition at the surface in order to satisfy the impenetrability constraint when solving the elliptic equation with the Finite Difference Method.

For our model, the following boundary conditions are selected :

$$\frac{\partial \lambda}{\partial n} = 0 \text{ for the terrain surface boundary} \tag{1.2.19}$$

$$\lambda = 0 \text{ for the top and sides boundaries} \tag{1.2.20}$$

1.2.3. Stability parameters

We now discuss the parametrization of the Gaussian precision moduli. It was noted that the coefficients $\beta_1 = \beta_2$ and β_3 play an important role in the correction of the velocity field. These weights can alter the flow pattern and the residual divergence of the adjusted flow field. Determining the right values for these parameters remains a major problem for wind models in atmospheric sciences. It has been noted through simulations of mass-consistent models that the codes are not directly sensitive to the values of β_1 and β_3 but to their ratio. This is why a new parameter was introduced :

$$\beta = \frac{\beta_1}{\beta_3}. \tag{1.2.21}$$

We recall the functional I in equation (1.2.1) now with $\beta_1 = \beta_2$:

$$I(u^d, v^d, w^d) = \int_\Omega \left[\beta_1^2(u^d - u)^2 + \beta_1^2(v^d - v)^2 + \beta_3^2(w^d - w)^2\right] dV. \tag{1.2.22}$$

One can see that large values of β_3 will imply minimal adjustments for w^d in $(w^d - w)^2$ since the functional is being minimized. The same argument is valid for small values of β_3 that will enforce bigger adjustments of w^d. The same reasoning can be used with the parameter β. For $\beta \gg 1$, flow adjustment in the vertical direction will predominate, so that wind is more likely to go over a terrain barrier rather than around it. For $\beta \ll 1$, flow adjustment will occur primarily in the horizontal plane, so the wind is more likely to go around a terrain barrier rather than over it.

Notice that when $\beta \to 0$, the adjustment is purely horizontal and when $\beta \to \infty$ the adjustment is strictly vertical. From a physical perspective, this last remark was used by the $\mathcal{WINDS}$ software [**11**] so that β could be defined as the atmospheric stability parameter, where the stratification obtained is unstable when $\beta \gg 1$, stable when $\beta \ll 1$ and neutral when $\beta = 1$.

The mass-consistent models usually adopt one of these two approaches when using the parameter β. It is either constant in the entire domain Ω and the values depend on atmospheric stability or it can be expressed as a function $\beta = \beta(x, y, z)$ and the values depend on the atmospheric stability and features of the topography.

1.3. Projection method

In the last section, we explained an approach widely used in atmospheric sciences to get a solenoidal vector field from a given vector field. Here, we introduce another method for solving problems in computational fluid dynamics. Chorin [**4**] has introduced this method based on the Hodge decomposition which allows one to extract a divergence-free vector field from any given vector field.

1.3.1. Hodge decomposition

Theorem 1.3.1 (Hodge decomposition)**.** *Let Ω be a simply connected domain with smooth boundary $\partial\Omega$. Any vector field $\mathbf{v}$ on Ω can be uniquely decomposed in the form*

$$\mathbf{v} = \mathbf{v}_d + \nabla\varphi \tag{1.3.1}$$

where $\nabla \cdot \mathbf{v}_d = 0$ in Ω and $\mathbf{v}_d \cdot \mathbf{n} = 0$ on $\partial\Omega$.

In other words, any vector field $\mathbf{v}$ can be decomposed into two orthogonal components; one divergence-free part $\mathbf{v}_d$ and a curl-free part expressed as the gradient of a scalar field φ . Applying the divergence operator on each side of equation (1.3.1) we have

$$\nabla \cdot \mathbf{v} = \underbrace{\nabla \cdot \mathbf{v}_d}_{=0} + \nabla \cdot \nabla\varphi \tag{1.3.2}$$

$$\nabla \cdot \mathbf{v} = \Delta\varphi \tag{1.3.3}$$

where Δ is the Laplacian operator in Cartesian coordinates. The solution of this elliptic equation given by $\varphi(x, y, z)$ will give the correction to be added to the initial vector field $\mathbf{v}$ such that we get the solenoidal vector field $\mathbf{v}_d$:

$$\mathbf{v}_d = \mathbf{v} - \nabla\varphi. \tag{1.3.4}$$

The boundary conditions required for solving the elliptic equation can be found by taking the normal component on both sides of equation (1.3.1) :

$$\mathbf{v} \cdot \mathbf{n} = \underbrace{\mathbf{v}_d \cdot \mathbf{n}}_{=0} + \nabla\varphi \cdot \mathbf{n} \tag{1.3.5}$$

$$\mathbf{v} \cdot \mathbf{n} = \nabla\varphi \cdot \mathbf{n} \tag{1.3.6}$$

which is the Neumann boundary condition :

$$\frac{\partial \varphi}{\partial n} = \mathbf{v} \cdot \mathbf{n} \text{ on } \partial\Omega. \tag{1.3.7}$$

1.3.2. Projection operator

Since $\mathbf{v}_d$ is uniquely determined, we can define a projection operator $\mathcal{P}$, such that

$$\mathcal{P}(\mathbf{v}) = \mathbf{v}_d. \tag{1.3.8}$$

Based on the procedure described above, we can define $\mathcal{P}$ as

$$\mathcal{P} = \mathbf{I} - \nabla \left(\nabla \cdot \nabla\right)^{-1} \nabla\cdot \tag{1.3.9}$$

The operator $\mathcal{P}$ defined this way is idempotent, $\mathcal{P}^2 = \mathcal{P}$, self-adjoint, $\mathcal{P} = \mathcal{P}^T$ and the norm of the operator is less than or equal to one, $\|\mathcal{P}(\mathbf{v})\|_2 \leq \|\mathbf{v}\|_2$. These properties are used to prove that the operator is stable.

1.3.3. Comparison with the variational method

Once again, the projection method requires the solution of the following Neumann problem :

$$\Delta\varphi = \nabla \cdot \mathbf{v} \text{ on } \Omega \tag{1.3.10}$$

$$\frac{\partial \varphi}{\partial n} = \mathbf{v} \cdot \mathbf{n} \text{ on } \partial\Omega. \tag{1.3.11}$$

Setting the parameters $\beta_1 = \beta_2 = \beta_3 = 1$ in equation (1.2.11) we have

$$\frac{\partial^2 \lambda}{\partial x^2} + \frac{\partial^2 \lambda}{\partial y^2} + \frac{\partial^2 \lambda}{\partial z^2} = -2\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \tag{1.3.12}$$

that is

$$\Delta\lambda = -2\nabla \cdot \mathbf{v} \tag{1.3.13}$$

It can be noticed that equation (1.3.13) is exactly the same as equation (1.3.10) up to the constant -2. The solution of both elliptic equations is a scalar field from which the gradient gives the correction to the initial vector field (equations (1.2.6)-(1.2.8) and (1.3.4)) that will make it divergence-free. We can therefore say that both methods are similar.

There are two main differences between these methods. First, the boundary conditions for solving the elliptic equations are not the same. In the projection method, only Neumann conditions are used unlike the mixed conditions (Dirichlet and Neumann) used in the variational method. The Hodge decomposition assumes that the divergence-free vector field $\mathbf{v}_d$ will be parallel to the domain boundary $\partial\Omega$ since $\mathbf{v}_d \cdot \mathbf{n} = 0$, which is definitely not the kind of vector field we are looking for.

It is also important to notice that the projection $\mathcal{P}$ is well-defined since $\mathbf{v}_d$ is uniquely determined as mentioned in the Hodge decomposition theorem. Unicity of the projection is guaranteed by the Neumann boundary conditions. In the current model, we are interested in using both Dirichlet and Neumann conditions. We will then lose the unicity of the projection. Hence, the corrected field $\mathbf{v}_d$ found with the projection $\mathcal{P}$ will be the *closest* vector field to $\mathbf{v}$ such that it is incompressible.

Finally, the main difference reside in the Gauss moduli. Since our model does not rely on experimental data, except the high altitude wind speed, there is no need for a calibration of the model with the weights β_i, hence they will be set equal to one. In order to avoid confusion with the notation, the scalar field will be noted by φ rather than λ and the problem that will be solved is the following :

$$\begin{cases} \Delta\varphi & = -2\nabla \cdot \mathbf{v} \text{ on } \Omega \\ \dfrac{\partial\varphi}{\partial n} & = 0 \text{ on terrain surface boundary} \\ \varphi & = 0 \text{ on side and top boundaries} \end{cases} \tag{1.3.14}$$

1.4. Vector field initialization

The way the initial vector field is initialized has a huge impact on the structure of the adjusted vector field. Remember that the corrected vector field will be the *closest* vector field to the initial vector field that satisfies the conservation of mass. Hence the initialization process is a very important step.

There are different types of data that are available to construct the initial wind vector field ; meteorological data from ground stations or towers stations, wind profilers, gradients or geostrophic wind. It has been observed that the quality

of this data is usually poor and that its density is more than often insufficient for resolving variations of the flow above complex topography. Moreover, the data gives only information for the vertical profile of the wind, never for the horizontal. These reasons encourage an initialization of the vector field with a high altitude wind, which is called the geostrophic wind in the literature.

As in $\mathcal{WINDS}$ [**11**], a vertical profile can be constructed from the geostrophic wind. Most of the models studied by Ratto et al. [**20**] split the atmosphere into two layers : the surface layer (SL) and the Planetary Boundary Layer (PBL). Over the PBL (1000m-2000m), the wind is assumed to be constant with height and given by the geostrophic wind. The surface layer takes into account friction/viscous effects and is usually contained in the first 100m of the atmosphere. This layer is not of great interest since the slip condition is applied at the surface of the terrain in this model. Between the two layers, different interpolation schemes can be used to get the vertical profile : linear, logarithmic or power law.

Barnard et al. [**2**] and Ross et al. [**21**] have also noticed that applying the closed boundary condition on the terrain surface requires that the initial velocity field must respect the slip condition at the terrain surface in order to satisfy the impenetrability constraint. Since there is usually no information on the horizontal profile, i.e. on the vertical component of the wind field, we will follow Barnard and set the vertical wind component to zero $w = 0$.

1.4.1. Potential flow and conformal coordinates

Barnard et al. [**2**] have observed that lack of verification is a major difficulty for mass-consistent models. Many papers used real data to calibrate and verify the results given by the model. Ross et al. [**21**] found out that if $\beta_1 = \beta_2 = \beta_3$ and that the right hand side of equation (1.3.14) vanishes, then φ represents a velocity potential. They also used simple terrain shapes (half-cylinder, hemisphere, ellipsoid) for which the analytic solution of the potential flow is known to generate a potential flow and verify the efficiency of their model. They use terrain-following

(conformal) coordinates to solve the Poisson equation :

$$\begin{cases} \xi & = x \\ \eta & = y \\ \sigma(x,y) & = \dfrac{h_t(x,y) - z}{h_t(x,y) - H(x,y)} \end{cases} \tag{1.4.1}$$

where $h_t(x,y)$ is the height of the domain which in our case is constant and where $H(x,y)$ is the height of the topography. The surface of the terrain is reached when $\sigma = 1$ and the top boundary when $\sigma = 0$. They still solve the elliptic equation with Cartesian coordinates but with a different Neumann boundary condition at the terrain surface in order to get the potential flow. They initialized their model with a uniform background wind in conformal coordinates and they found that the generated flow is in good agreement with the analytic potential flow. Even if the problem is solved in Cartesian coordinates, conformal coordinates are useful for constructing an initial velocity field tangent to the terrain and parallel to the top domain boundary.

Chapitre 2

SOLVING THE MODEL IN 2 DIMENSIONS

There exists many ways to solve the Poisson equation introduced in the last chapter. Differences between previous mass-consistent models come from numerical algorithms for solving elliptic equations and from the choice of the values used for the parameters. For instance, Sherman [**24**] used the Finite Difference Method (FDM) and discretized the topography in a *stair-step* fashion. Ishikawa [**13**] solved the elliptic equation with FDM as well but on a staggered grid. Ross et al. [**21**] and Barnard et al. [**2**] used conformal coordinates to solve the elliptic equation in order to get better integration of the terrain surface and ensure the application of the slip condition. Forthofer [**10**] used the Finite Element Method (FEM) to solve the PDE.

This chapter will consider the Embedded Boundary Method (EBM) for solving the elliptic equation with the appropriate boundary conditions for the two-dimensional case. This is the first time that EBM is applied to an empirical model used to simulate the spread of wildfires. The particular feature of this approach is that it embeds an irregular boundary into a Cartesian grid. Hence, the generated wind fields should be in better agreement with the terrain surface.

Since the EBM is based on the Finite Volume Method (FVM), this last method is first introduced before moving on to EBM. The convergence of the algorithm is tested for different terrain shapes and initial wind vector fields, and an error analysis is also conducted. Finally, the divergence-free vector fields are used to spread the fire over different terrain surfaces.

2.1. Finite Volume Method

The FVM is a useful approach for solving numerically partial differential equations (PDE). It allows the divergence operator to be discretized using the divergence theorem, such as in the Poisson equation. The finite-volume approach has some considerable advantages such as regular predictable memory access and higher accuracy for less computation. In this section, we explain how the FVM can solve our elliptic equation without topography embedded in the rectangular domain.

2.1.1. Discretization of the domain

Consider a rectangular domain $\Omega \subset \mathbb{R}^2$. The domain has length L_x and height L_y and is discretized using a Cartesian grid whose rectangular control volumes are defined as $\Upsilon_{i,j} = [(i-\frac{1}{2})h_x, (i+\frac{1}{2})h_x] \times [(j-\frac{1}{2})h_y, (j+\frac{1}{2})h_y]$ for $i = 1, \ldots, N_x$ and $j = 1, \ldots, N_y$. The number of horizontal and vertical cells, N_x and N_y, are used to define the horizontal and vertical mesh spacing $h_x = L_x/N_x$ and $h_y = L_y/N_y$. The method uses control volumes $V_{i,j} = \Upsilon_{i,j} \cap \Omega$.

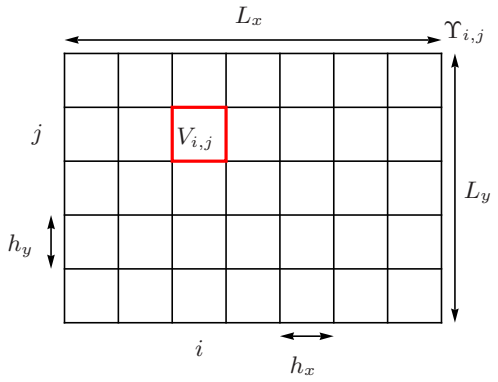


Fig. 2.1. Discretization of the two-dimensional domain for FVM.

2.1.2. Divergence operator

We are now looking to solve the following Poisson equation

$$\Delta\varphi = -2\nabla \cdot \mathbf{v} \tag{2.1.1}$$

where $\mathbf{v} = (u, v)$ is the initial vector field and φ is a scalar field. Equation (2.1.1) can be written in conservation form

$$\nabla \cdot \mathbf{F} = -2\nabla \cdot \mathbf{v} \tag{2.1.2}$$

where the flux $\mathbf{F} = \nabla\varphi$ is the conserved quantity.

To achieve a discretization of the divergence operator, we first recall the divergence theorem :

Theorem 2.1.1 (Divergence theorem)**.** *Let Ω be a compact region in $\mathbb{R}^n$ with a piecewise smooth boundary $\partial\Omega$. If $\mathbf{F}$ is a continuously differentiable vector field defined on a neighbourhood of Ω, then we have*

$$\int_{\Omega} \nabla \cdot \mathbf{F} dV = \oint_{\partial\Omega} \mathbf{F} \cdot \mathbf{n} dA. \tag{2.1.3}$$

Using the cell average value of the divergence of $\mathbf{F}$ and the divergence theorem in $\mathbb{R}^2$ we have ;

$$\nabla \cdot \mathbf{F} \approx \frac{1}{|V_{i,j}|} \int_{V_i} \nabla \cdot \mathbf{F} dV = \frac{1}{|V_{i,j}|} \oint_{\partial V_i} \mathbf{F} \cdot \mathbf{n} dA \tag{2.1.4}$$

$$= \frac{1}{|V_{i,j}|} \left[h_y F_{i+\frac{1}{2},j} + h_x F_{i,j+\frac{1}{2}} - h_y F_{i-\frac{1}{2},j} - h_x F_{i,j-\frac{1}{2}} \right] \tag{2.1.5}$$

where $\mathbf{n}$ is the outward unit normal to the control volume cell $V_{i,j}$ and $|V_{i,j}| = h_x h_y$ the volume of $V_{i,j}$. This corresponds to the midpoint rule discretization of the line integral. Hence, the discretized divergence operator $D^M(\mathbf{F})_{i,j}$ is

$$D^M(\mathbf{F})_{i,j} = \frac{F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j}}{h_x} + \frac{F_{i,j+\frac{1}{2}} - F_{i,j-\frac{1}{2}}}{h_y} \tag{2.1.6}$$

The fluxes are illustrated in figure 2.2. In the Poisson equation, a discretized

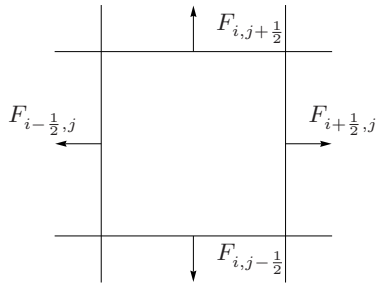


FIG. 2.2. Fluxes on every edge of control volume $V_{i,j}$.

divergence operator is also applied to the initial vector field $\mathbf{v}$. It would be desirable to use the same discretization of the operator. We also need a consistent discretization for the gradient operator since $\mathbf{F} = \nabla\varphi$. The MAC projection seems quite appropriate for this case.

2.1.3. Marker-and-Cell Projection

There are many ways to discretize mathematical operators. The discretization mostly relies on the kind of grid on which the computation is done. The Marker-and-Cell (MAC) projection introduced by Harlow and Welch [**12**] uses a staggered grid, where the components of the vector field are defined on the edges of the control cells and the scalar field is defined at the center as shown in Figure 2.3.

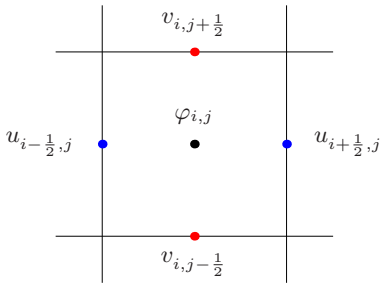


FIG. 2.3. Staggered grid with the cell-centered scalar field φ and the edge-centered components of the velocity field (u, v).

Let D^M and G^M be the discrete divergence and discrete gradient operators over the staggered grid. The divergence is defined using the divergence theorem on the control cell as before and the gradient by centered difference over the edges :

$$D^M(\mathbf{v})_{i,j} = \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h_x} + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{h_y} \tag{2.1.7}$$

and define

$$G^M(\varphi)_{i+\frac{1}{2},j} = \frac{\varphi_{i+1,j} - \varphi_{i,j}}{h_x} \tag{2.1.8}$$

$$G^M(\varphi)_{i,j+\frac{1}{2}} = \frac{\varphi_{i,j+1} - \varphi_{i,j}}{h_y} \tag{2.1.9}$$

Remember that our method uses a projection operator $\mathcal{P}$ defined in equation (1.3.9). Let $\mathbf{P}^M$ be the discretization of this projection. It is then defined as

$$\mathbf{P}^M = \mathbf{I} - G^M \left(D^M G^M\right)^{-1} D^M \tag{2.1.10}$$

and it is easy to verify that $D^M(\mathbf{P}^M(\mathbf{v})) = 0$. This means that the discretized projection operator is exact. See Appendix A for a discussion of the discretization of projections.

For the MAC projection, the discretization of the Laplace operator ($\Delta = \nabla \cdot \nabla$) is defined as $D^M(G^M(\varphi))_{i,j}$ which corresponds to the standard 5 point stencil Laplacian for interior control volume cells ($i = 2, ..., N_x - 1$ and $j = 2, ..., N_y - 1$) as shown in Figure 2.4.

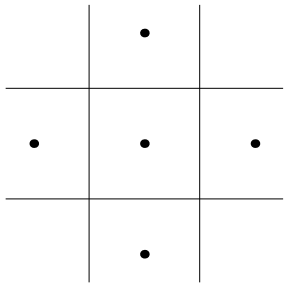


FIG. 2.4. Standard 5 point stencil Laplacian $D^M(G^M(\varphi))_{i,j}$.

Despite the fact that the components of the vector field are not collocated, the MAC projection has the advantage that the no-flow boundary condition can be set explicitly at walls for rectangular domains since the edges of the boundary cells match the domain boundaries.

2.1.4. Boundary fluxes

In order to use the divergence operator on the control volume cells on the boundaries of the domain, the fluxes at the boundaries must be specified. For the bottom boundary, the problem uses a Neumann condition $\frac{\partial \varphi}{\partial n} = 0$. In this case, the flux is simply zero, $F_{i,\frac{1}{2}} = 0$.

For the top and side boundaries, the Dirichlet condition $\varphi = 0$ is used. This type of condition does not prescribe directly a particular value for the flux at the boundary. Hence, we follow Johansen and Colella [**14**] and use a three-point gradient stencil in order to get a specific value of the flux on the boundary. The gradient formula is given by

$$\frac{\partial \varphi}{\partial n} = \frac{1}{d_2 - d_1} \left[\frac{d_2}{d_1} \left(\varphi^B - \varphi_1 \right) - \frac{d_1}{d_2} \left(\varphi^B - \varphi_2 \right) \right] \tag{2.1.11}$$

where φ^B is the value of the function φ on the boundary given by the Dirichlet condition, φ_1 and φ_2 are the closest grid point values of φ to φ^B. Those two grid points are respectively at distances d_1 and d_2 from the boundary. This stencil has a discretization error of order $\mathcal{O}(h^2)$. Figure 2.5 shows how the flux on the right boundary is calculated by

$$\frac{\partial \varphi}{\partial x} = F_{N_x+\frac{1}{2},j} = \frac{1}{h_x}\left[3\left(\varphi^B - \varphi_{N_x,j}\right) - \frac{1}{3}\left(\varphi^B - \varphi_{N_x-1,j}\right)\right] \tag{2.1.12}$$

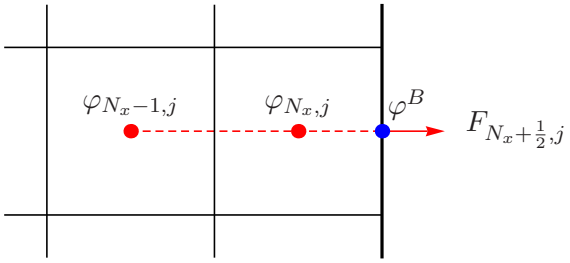


FIG. 2.5. Interpolation of the boundary flux $F_{N_x+\frac{1}{2},j}$ for applying the Dirichlet condition using φ^B, $\varphi_{N_x,j}$ and $\varphi_{N_x-1,j}$.

Note that this interpolation leads to an outward pointing gradient. With the fluxes at the boundary fixed, it is easy to see that the Laplacian operator will have 3 different stencils over the domain as pictured in Figure 2.6.

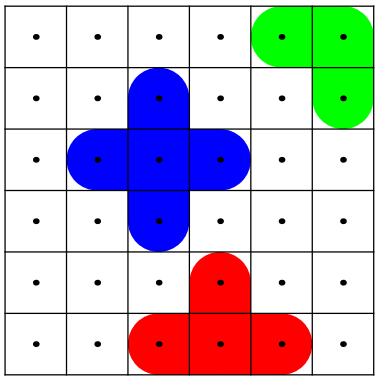


FIG. 2.6. The 3 different stencils for the Laplacian operator.

2.1.5. Symmetric matrix

Solving the Poisson equation with the FVM reduces to find a solution of a linear system $\mathbf{A}\boldsymbol{\varphi} = \mathbf{b}$. For the two-dimensional case, we use the *row-wise ordering*

to form the vector $\boldsymbol{\varphi}$ of the $\varphi_{i,j}$ defined at the center of the control volume cells :

$$\boldsymbol{\varphi} = (\varphi_{1,1}, \ldots, \varphi_{N_x,1}, \varphi_{1,2}, \ldots, \varphi_{N_x,2}, \ldots, \varphi_{1,N_y}, \ldots, \varphi_{N_x,N_y}) \tag{2.1.13}$$

The coefficients in front of the unknown values of $\varphi_{i,j}$ fill up the matrix $\mathbf{A}$ of size $N_xN_y \times N_xN_y$. More precisely, the main diagonal contains the coefficients of $\varphi_{i,j}$, the diagonals above and below the main diagonal the coefficients of $\varphi_{i\pm1,j}$ and the two extra diagonals, with $N_x - 2$ zero elements between the upper/lower diagonal, the coefficients of $\varphi_{i,j\pm1}$. Hence, the symmetric matrix $\mathbf{A}$ has the structure shown in figure 2.7 which is the same as that of the FDM.

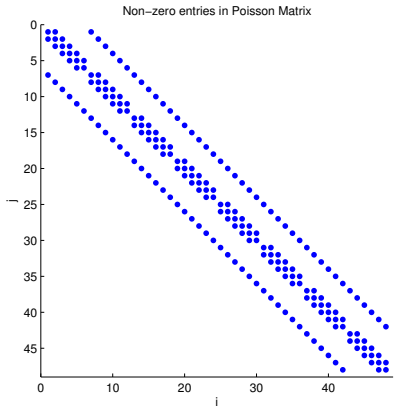


FIG. 2.7. Symmetric matrix of the linear system with $N_x = 6$ and $N_y = 8$.

All known values such as φ^B and the RHS of the Poisson equation, $D^M(\mathbf{v})_{i,j}$, are put in the vector $\mathbf{b}$. The linear system $\mathbf{A}\boldsymbol{\varphi} = \mathbf{b}$ is solved using the solver `mldivide` in MATLAB which uses a direct method.

2.1.6. Corrected vector field

Once we get the solution given by the vector $\boldsymbol{\varphi}$, we can compute the corrected vector field with equations (1.2.6)-(1.2.8) of chapter 1. The corrected field on the staggered grid is easily calculated with the MAC gradient operators :

$$u^d_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j} + \frac{1}{2}G^M(\varphi)_{i+\frac{1}{2},j} \tag{2.1.14}$$

$$v^d_{i,j+\frac{1}{2}} = v_{i,j+\frac{1}{2}} + \frac{1}{2}G^M(\varphi)_{i,j+\frac{1}{2}} \tag{2.1.15}$$

Notice that it is important to calculate $G^M(\varphi)_{i+\frac{1}{2},j}$ and $G^M(\varphi)_{i,j+\frac{1}{2}}$ on the boundaries with the same gradient stencil used for computing the fluxes on the boundaries. Otherwise, the final divergence will not be zero on the cells next to the boundaries.

2.2. Embedded Boundary Method

We will now focus on the Embedded Boundary Method for solving numerically the elliptic equation (2.1.1) on an irregular two-dimensional domain Ω. This approach uses a finite-volume discretization on the Cartesian grid on which the boundary of the topography is embedded. In this chapter, we will focus on the simple 2D case which is discussed by Johansen and Colella [**14**].

The rectangular domain Ω uses the same Cartesian grid as mentioned earlier. The topography which is given by the function $H(x)$ is represented as a piecewise linear function where the coordinate $x = (i \pm \frac{1}{2})h_x$ is defined on the control volume cell edges. The function $H(x)$ corresponds to the lower boundary of Ω. The geometry of the irregular domain Ω is represented with its intersection with the Cartesian grid. The method uses control volumes $V_{i,j} = \Upsilon_{i,j} \cap \Omega$ and their faces $A_{i\pm\frac{1}{2},j}$ and $A_{i,j\pm\frac{1}{2}}$ which are the intersection of $\partial V_{i,j}$ with the coordinate lines $\{x = (i \pm \frac{1}{2})h_x\}$ and $\{y = (j \pm \frac{1}{2})h_y\}$. The intersection of the boundary of the irregular domain and the Cartesian control volumes are the faces $A^B_{i,j} = \Upsilon_{i,j} \cap \partial\Omega$.

In order to construct an appropriate divergence operator, areas and volumes are written as nondimensional terms :

volume fractions : $\kappa_{i,j} = |V_{i,j}|(h_x h_y)^{-1}$

face apertures : $\alpha_{i\pm\frac{1}{2},j} = |A_{i\pm\frac{1}{2},j}|h_y^{-1}$ and $\alpha_{i,j\pm\frac{1}{2}} = |A_{i,j\pm\frac{1}{2}}|h_x^{-1}$

boundary apertures : $\alpha^B_{i,j}$.

It is assumed that those values can be calculated with an accuracy of order $\mathcal{O}(h^2)$. Notice that $\alpha, \kappa \in [0, 1]$. When $\kappa = 0$, the control volume cell is completely contained in the topography and when $\kappa = 1$, the cell is full, meaning that the topography does not cut the cell.

As before, we use the fact that the Laplacian operator $\Delta\varphi$ in LHS of equation (2.1.1) can be written on a conservative form as $\nabla \cdot \mathbf{F}$ with $\mathbf{F} = \nabla\varphi$. It is then

possible to discretize the divergence operator $\nabla\cdot$ using the divergence theorem :

$$D^M(\mathbf{F})_{i,j} = \frac{1}{\kappa_{i,j}}\left(\frac{\alpha_{i+\frac{1}{2},j}F_{i+\frac{1}{2},j} - \alpha_{i-\frac{1}{2},j}F_{i-\frac{1}{2},j}}{h_x} + \frac{\alpha_{i,j+\frac{1}{2}}F_{i,j+\frac{1}{2}} - \alpha_{i,j-\frac{1}{2}}F_{i,j-\frac{1}{2}}}{h_y} + \alpha^B_{i,j}F^B_{i,j}\right) \tag{2.2.1}$$

where the fluxes $F_{i\pm\frac{1}{2},j}$, $F_{i,j\pm\frac{1}{2}}$ and the flux on the embedded boundary F^B, defined at the centroid of every cell faces, are linear combinations of $\varphi_{i,j}$ and of the boundary values φ^B. This discretization takes into account the structure of the volume cell, particularly when we are calculating the divergence in a cut cell ($0 < \kappa < 1$). The left picture in Figure 2.8 gives an example of fluxes located at the face centroids in a cut cell, while the face centroids coincide with the face centers in a full cell.

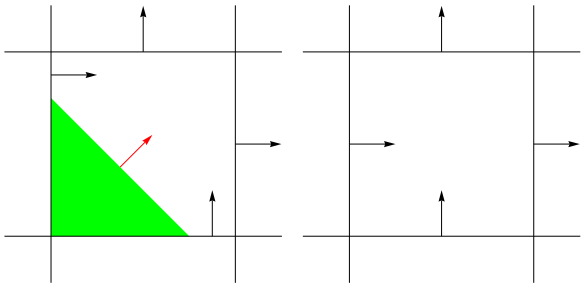


FIG. 2.8. Fluxes in a cut cell ($0 < \kappa < 1$) and in a full cell ($\kappa = 1$). The flux $F^B_{i,j} \equiv 0$ (red arrow) since a Neumann condition is applied at the terrain surface boundary.

Remember that a homogeneous Neumann condition $\frac{\partial\varphi}{\partial n} = 0$ is applied on the terrain surface boundary which implies that $F^B_{i,j} \equiv 0$. Hence, the divergence operator becomes

$$D^M(\mathbf{F})_{i,j} = \frac{1}{\kappa_{i,j}}\left(\frac{\alpha_{i+\frac{1}{2},j}F_{i+\frac{1}{2},j} - \alpha_{i-\frac{1}{2},j}F_{i-\frac{1}{2},j}}{h_x} + \frac{\alpha_{i,j+\frac{1}{2}}F_{i,j+\frac{1}{2}} - \alpha_{i,j-\frac{1}{2}}F_{i,j-\frac{1}{2}}}{h_y}\right) \tag{2.2.2}$$

It is important that the fluxes be defined at the centroid of the faces $A_{i\pm\frac{1}{2},j}$ and $A_{i,j\pm\frac{1}{2}}$ in order to keep a good approximation of the discretization of the integral. For fluxes defined on the vertical faces $A_{i\pm\frac{1}{2},j}$, a linear interpolation scheme is

used with the above cell. For instance, the flux $F_{i+\frac{1}{2},j}$ on the face $A_{i+\frac{1}{2},j}$ when $0 < \alpha_{i+\frac{1}{2},j} < 1$ can be calculated using the interpolation formula :

$$F_{i+\frac{1}{2},j} = \left(\frac{1+\alpha_{i+\frac{1}{2},j}}{2}\right) G^M(\varphi)_{i+\frac{1}{2},j} + \left(\frac{1-\alpha_{i+\frac{1}{2},j}}{2}\right) G^M(\varphi)_{i+\frac{1}{2},j+1} \tag{2.2.3}$$

This interpolation is illustrated in Figure 2.9. Note that for full faces (without embedded boundary), the face aperture $\alpha_{i+\frac{1}{2},j} = 1$, hence the regular centered difference is used :

$$F_{i+\frac{1}{2},j} = G^M(\varphi)_{i+\frac{1}{2},j}. \tag{2.2.4}$$

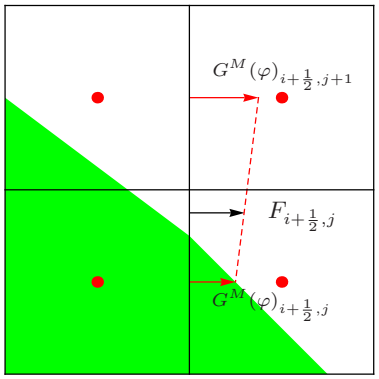


FIG. 2.9. Interpolation of the flux $F_{i+\frac{1}{2},j}$ on a cut cell edge using $G^M(\varphi)_{i+\frac{1}{2},j}$ and $G^M(\varphi)_{i+\frac{1}{2},j+1}$.

For the fluxes defined on the horizontal faces $A_{i,j\pm\frac{1}{2}}$, a linear interpolation scheme using the flux in either the right or left cell is required. It is important to use the appropriate interpolation so the computation makes sense. The flux $F_{i,j+\frac{1}{2}}$ on the face $A_{i,j+\frac{1}{2}}$ when $0 < \alpha_{i,j+\frac{1}{2}} < 1$ can be calculated using this interpolation formula :

$$\begin{aligned} F_{i,j+\frac{1}{2}} = & \left(\frac{1+\alpha_{i,j+\frac{1}{2}}}{2}\right) G^M(\varphi)_{i,j+\frac{1}{2}} \\ & + \left(\frac{1-\alpha_{i,j+\frac{1}{2}}}{2}\right) \begin{cases} G^M(\varphi)_{i+1,j+\frac{1}{2}} & \text{if terrain is on the left side} \\ G^M(\varphi)_{i-1,j+\frac{1}{2}} & \text{if terrain is on the right side} \end{cases} \end{aligned} \tag{2.2.5}$$

The two cases are shown in Figure 2.10.

When the divergence is calculated in a full cell ($\kappa_{i,j} = 1$) for which all face apertures $\alpha = 1$, we recover the MAC divergence operator given by equation

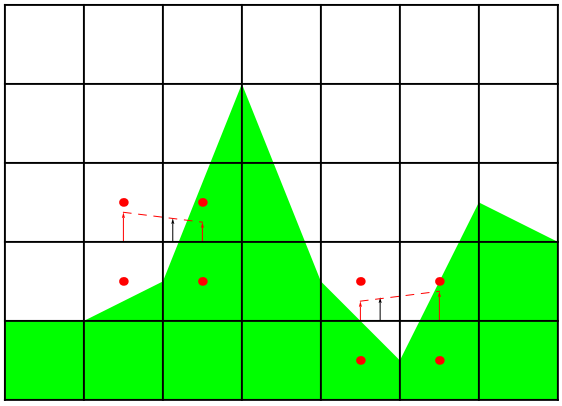


FIG. 2.10. Interpolation of the flux $F_{i,j+\frac{1}{2}}$ on a cut cell edge when the terrain is on the right hand side or left hand side.

(2.1.6) for the FVM :

$$D^M(\mathbf{F})_{i,j} = \frac{F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j}}{h_x} + \frac{F_{i,j+\frac{1}{2}} - F_{i,j-\frac{1}{2}}}{h_y} \tag{2.2.6}$$

A critical feature of the EBM, as explained by Johansen and Colella [**14**], is the assumption that the solution can be extended smoothly outside of Ω. As can be seen in Figure 2.10, some grid values φ are covered by the terrain. Johansen and Colella assume that there are solution values for them that are sufficiently smooth so that a truncation error analysis based on Taylor expansions will be valid.

For top and side boundaries which have Dirichlet conditions, the same gradient stencil is used as in the previous section. Johansen and Colella use one more constraint on the discretization of the domain that is related to this gradient formula : the interpolation stencil must not reach into cells with zero volume ($\kappa = 0$), hence the Cartesian grid must be fine *enough*.

The divergence on the right hand side of equation (2.1.1) is

$$D^M(\mathbf{v})_{i,j} = \frac{1}{\kappa_{i,j}} \left(\frac{\alpha_{i+\frac{1}{2},j} u_{i+\frac{1}{2},j} - \alpha_{i-\frac{1}{2},j} u_{i-\frac{1}{2},j}}{h_x} + \frac{\alpha_{i,j+\frac{1}{2}} v_{i,j+\frac{1}{2}} - \alpha_{i,j-\frac{1}{2}} v_{i,j-\frac{1}{2}}}{h_y} \right) \tag{2.2.7}$$

The interpolation of the vector field components in the middle of the faces in a cut cell use the same interpolation explained earlier

$$u^I_{i+\frac{1}{2},j} = \left(\frac{1+\alpha_{i+\frac{1}{2},j}}{2}\right) u_{i+\frac{1}{2},j} + \left(\frac{1-\alpha_{i+\frac{1}{2},j}}{2}\right) u_{i+\frac{1}{2},j+1} \tag{2.2.8}$$

$$\begin{aligned} v^I_{i,j+\frac{1}{2}} &= \left(\frac{1+\alpha_{i,j+\frac{1}{2}}}{2}\right) v_{i,j+\frac{1}{2}} \\ &+ \left(\frac{1-\alpha_{i,j+\frac{1}{2}}}{2}\right) \begin{cases} v_{i+1,j+\frac{1}{2}} & \text{if terrain is on the left side} \\ v_{i-1,j+\frac{1}{2}} & \text{if terrain is on the right side} \end{cases} \end{aligned} \tag{2.2.9}$$

2.2.1. Sparse matrix

As before, the solution of the Poisson equation is found by solving a linear system of the form $\mathbf{A}\boldsymbol{\varphi} = \mathbf{b}$ with the `mldivide` solver in MATLAB. Since the computational domain Ω is irregular, (the zero volume cells, $\kappa = 0$, are not taken into account in the computation), the matrix loses its symmetric property, but it is still a sparse matrix. For the purpose of computation, lines and rows corresponding to cells where $\kappa = 0$ are not removed, but instead, a fake value is added to the main diagonal element, so the matrix is not singular. See Figure 2.11 for the structure of the matrix $\mathbf{A}$ for the case of an exponential terrain surface.

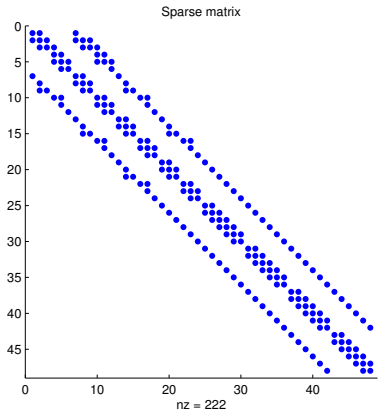


FIG. 2.11. Sparse matrix $\mathbf{A}$ for solving Poisson equation on an irregular domain with an embedded exponential hill with $N_x = 6$ and $N_y = 8$.

2.2.2. Corrected vector field

Once the vector $\boldsymbol{\varphi}$ is known, the corrected field on the staggered grid is easily calculated with the MAC gradient operators :

$$u^d_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j} + \frac{1}{2}G^M(\varphi)_{i+\frac{1}{2},j} \tag{2.2.10}$$

$$v^d_{i,j+\frac{1}{2}} = v_{i,j+\frac{1}{2}} + \frac{1}{2}G^M(\varphi)_{i,j+\frac{1}{2}} \tag{2.2.11}$$

when $\alpha_{i+\frac{1}{2},j}$ and $\alpha_{i,j+\frac{1}{2}}$ are not equal to zero. Once again, it is important to calculate $G^M(\varphi)_{i+\frac{1}{2},j}$ and $G^M(\varphi)_{i,j+\frac{1}{2}}$ on the domain boundaries with the same gradient stencil used for computing the fluxes on the boundaries.

2.3. Convergence and error analysis

In order to study the efficiency and accuracy of the Embedded Boundary Method, the algorithm will be used for solving constant or particular initial velocity fields over different geometries.

Two definitions of the usual norms were used for evaluating the convergence. In the case of values on the staggered grid, the $p-$norm is defined as :

$$\|\xi\|_p = \left(\sum_{(i,j)\in\Upsilon_{i,j}} |\xi_{i,j}|^p h^2 \right)^{1/p} \tag{2.3.1}$$

where $\Upsilon_{i,j}$ is the Cartesian grid over the total rectangular domain. Knowing that $h^2 = L^2/N^2$, we can rewrite (2.3.1) as :

$$\|\xi\|_1 = \frac{1}{N^2} \sum_{(i,j)\in\Upsilon_{i,j}} |\xi_{i,j}| \tag{2.3.2}$$

$$\|\xi\|_2 = \frac{1}{N} \left(\sum_{(i,j)\in\Upsilon_{i,j}} |\xi_{i,j}|^2 \right)^{1/2} \tag{2.3.3}$$

$$\|\xi\|_\infty = \max_{(i,j)\in\Upsilon_{i,j}} |\xi_{i,j}|. \tag{2.3.4}$$

up to a constant L. When the exact analytic solution for a given test case is unknown, a reference solution is used to check the convergence of the algorithm. The reference solution is a numerical solution for a problem on a finer grid. Then, we check at which rate the error between solutions on the coarser grids and the

reference solution diminishes when the grid resolution gets closer to the reference grid. In our case, we will compute the absolute error of the horizontal and vertical components of the corrected vector field $\mathbf{v}_d = (u^d, v^d)$ with the components of the reference solution. Then we will check if the errors converge for different norms above.

For instance, the absolute error for u^d is given by

$$e_N(u_N^d) = \left| u_{N_{ref}}^d - u_N^d \right|. \tag{2.3.5}$$

where N_{ref} is the number of cells of reference grid and N the number of cells of the actual grid.

In order to compute the absolute error e_N for the staggered components of the wind, we must compare the same grid nodes for each discretization of the grid. Since we are using a staggered grid for our vector field components, we must take the mean of the components of every grid except the coarsest grid, so we can compute the error at the same location on all grids. Figure 2.12 shows the vector component u^d for $N = 2$ and $N = 8$. Notice that the number of cells N in those

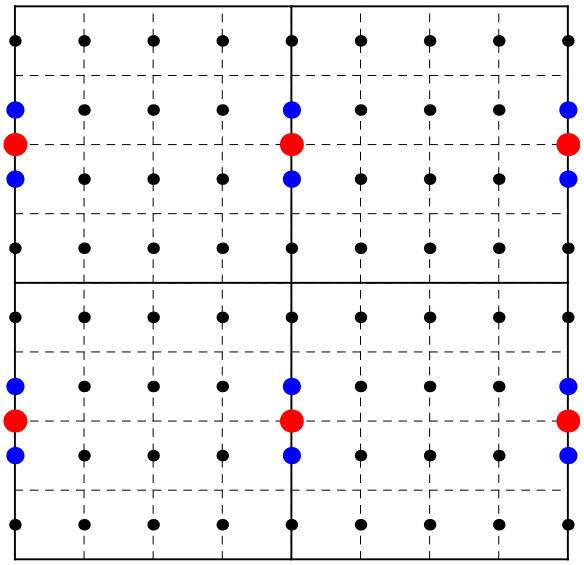


FIG. 2.12. The convergence of u^d on the red points of the grid $N = 2$ is done with the mean of the blue points of the grid $N = 8$.

norms will have to be replaced by N_{coar} which is the number of horizontal (resp.

vertical) components of the velocity of the coarsest grid on which the convergence tests are done. This corresponds to $N_{coar} = 6$ in our example in figure 2.12.

Since the discretization of the topography relies on the grid, each discretization changes the geometry of the domain. In order to avoid misleading errors, we will not consider the vector field components where $\alpha_{i+\frac{1}{2},j}$ and $\alpha_{i,j+\frac{1}{2}}$ are not equal to one for the coarsest discretization.

The second definition of the norm is a volume-weighted norm as used by Johansen and Colella [**14**] for cell-centered quantities. For a cell-centered variable ξ, the max norm is :

$$\|\xi\|_\infty = \max_{(i,j)\in\Omega} |\xi_{i,j}| \tag{2.3.6}$$

and the $p-$norm :

$$\|\xi\|_p = \left(\sum_{(i,j)\in\Omega} |\xi_{i,j}|^p \kappa_{i,j} h^2 \Big/ \sum_{(i,j)\in\Omega} \kappa_{i,j} h^2 \right)^{1/p} \tag{2.3.7}$$

where Ω is the computational domain and $\kappa_{i,j}$ the volume fraction of each control cell.

This definition of the norms is particularly useful when the exact solution of the equation is known. If it is not the case, we proceed as before using a reference solution which is the solution on a grid of very fine resolution. Then, we compute the error between the solutions of the coarsest grids and the reference fine grid. In our case, we are interested in the convergence of the velocity field. Since the norms are defined for cell-centered quantities, we average the edge-centered velocity field at the center of each cell. Then, the reference solution must be averaged to the coarse grid using a volume-weighted average :

$$Av(\xi^f) = \frac{1}{V^c} \sum_{v_f \in \mathcal{F}} V^f \xi_{v_f} \tag{2.3.8}$$

where $\mathcal{F}$ is the set of cells of the fine grid v_f contained in the cell of the coarse grid v_c and where V^f and V^c are the volume of cells v_f and v_c respectively. Figure 2.13 shows the cells v_f of the fine grid when $N = 8$ that are contained in the coarse cell v_c of the grid $N = 2$.

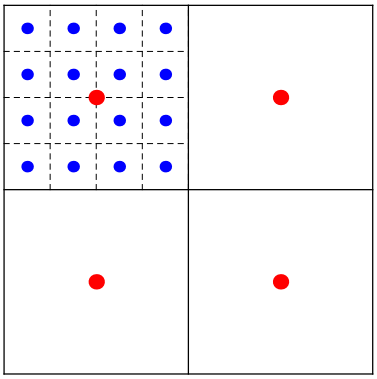


FIG. 2.13. The fine to coarse averaging of a cell-centered quantity using a volume-weighted average of values of the fine grid $N = 8$ to the coarse grid $N = 2$.

This average is useful since it allows us to use all the grid cells of any resolution grid size for comparing with the reference grid, which is not the case with the previous definition of the norm, since we were restricted to the number of values of the coarsest grid N_{coar}. Also, the volume-weighted average considers the geometry of the terrain in the two grids.

The absolute error for the component u^d is then given by

$$e_N(u_N^d) = \left| Av(u_{N_{ref}}^d) - u_N^d \right|. \tag{2.3.9}$$

where N_{ref} is the number of cells of reference grid and N the number of cells of a coarser grid. The absolute error is computed for the other components of the corrected vector field $\mathbf{v}_d = (u^d, v^d)$. Then, we check if the errors converge to zero for different volume-weighted norms. It is important to note that for the two-dimensional test cases, the volume-weighted norms were taken over all **full** volume cells only.

All convergence tests are performed on square domains :

$$L = L_x = L_y$$

$$N = N_x = N_y$$

$$h = h_x = h_y.$$

For all our cases, we set $L = 10$ and we use as a reference solution the results obtained on the grid with $N_{ref} = 512$ cells and the coarsest grid is $N = 8$.

The solutions of the different cases will be shown with the streamlines of the velocity field and the contour plots for the final divergence and velocity components. Those figures will be done using the following parameters for the discretization of the rectangular domain in all case tests :

N	L
64	10

TAB. 2.1. Parameters used for test cases in 2D.

2.3.1. Flat terrain case

The first test is done on a topography which is a flat line defined by

$$H_{flat}(x) = \frac{L_y}{3}. \tag{2.3.10}$$

This simple geometry does not show the powerful features of EBM but it allows us to check if the convergence rate of EBM for a rectangular domain is similar to the FVM. We also want to see the effects of the boundary conditions on the final velocity field. Even if we are just considering velocity fields for which the magnitude changes with height in our model, we define the following arbitrary initial velocity field for this test case :

$$(u, v)_{flat} = (100 + x^{\frac{5}{4}}, 0). \tag{2.3.11}$$

Note that for a constant horizontal velocity field over a flat terrain and even a horizontal velocity field with increasing amplitude over the $y-$axis, the initial divergence will be zero, meaning that the initial vector field is already mass-consistent. This will result in the trivial solution $\varphi = 0$ over the whole domain, hence the gradient giving the correction will be zero and the final vector field will be the same as the initial one. For that reason, we use an initial velocity field which depends on x.

Since the velocity increases with x, we can therefore expect an increasing divergence on the $x-$axis. The algorithm results in a zero-divergence velocity field as is shown in Figure 2.14.

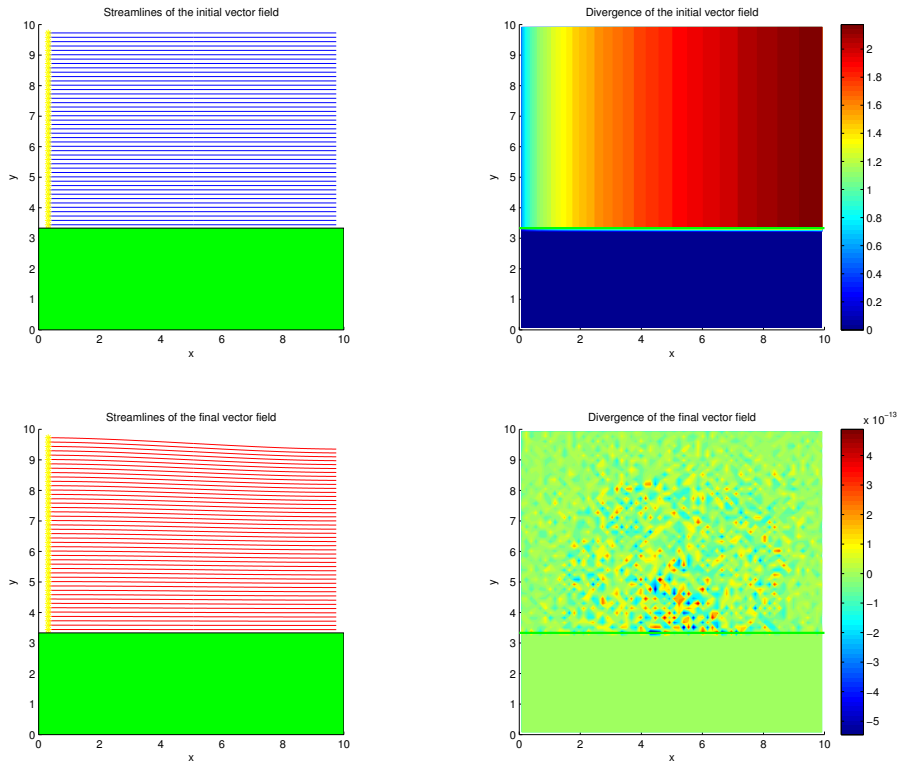


FIG. 2.14. Streamlines and divergence of initial and final vector fields $(u, v)_{flat}$ over the flat terrain surface H_{flat}.

Notice that the streamlines of the final vector field are not exactly straight as we could expect. Figure 2.15 shows the contour plot of the final velocity field components.

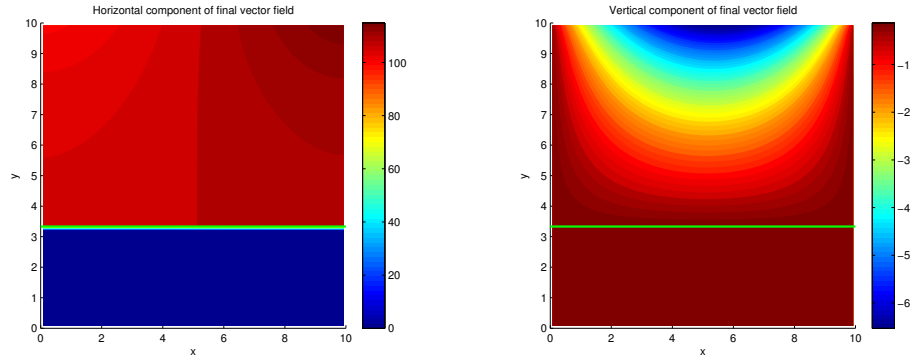


FIG. 2.15. Contour plot of the final velocity field components $(u^d, v^d)_{flat}$ over the flat terrain surface H_{flat}.

As we can see, even if the initial vertical velocity component v was zero, a correction was added to this component. This might be explained by the homogeneous Dirichlet boundary condition at the top of the domain which does not force the final vector field to remain horizontal at the top boundary.

In figure 2.16, we plot the different norms of the absolute errors defined above on a *log-log* scale as a function of the parameter N.

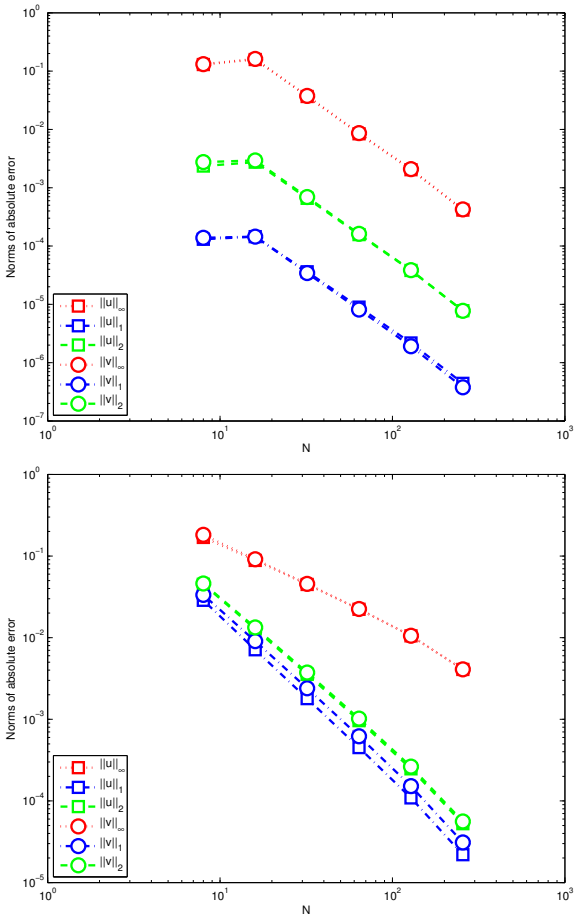


FIG. 2.16. Norms of the staggered absolute error e_N (top) and volume-weighted norms of the cell-centered absolute error e_N (bottom) of $(u^d, v^d)_{flat}$.

The convergence rate of the absolute error e_N in the different norms are shown in the following tables.

We notice a big difference between the max norm of the two tables even if their definition is the same. This is explained by the fact that in the case of the max norm of the absolute error of the staggered components, only a few number

e_N	L_∞	L_1	L_2
u	1.87	1.83	1.85
v	1.87	1.89	1.88

TAB. 2.2. Convergence rate of staggered absolute error e_N for $(u^d, v^d)_{flat}$.

e_N	L_∞	L_1	L_2
u	1.06	2.05	1.94
v	1.08	2.00	1.93

TAB. 2.3. Convergence rate of cell-centered absolute error e_N for $(u^d, v^d)_{flat}$.

of edges are used for computing the error, while in the volume-weighted norm, all cells are considered, especially those close to the embedded boundary, where the larger errors usually occur. Otherwise, for the L_1 and L_2 norms, the rate of convergence is about $\mathcal{O}(h^2)$.

2.3.2. Sinusoidal hills case

We now consider a slightly more complicated geometry in order to check the performance of the Embedded Boundary Method. The topography is defined by a sinusoidal function :

$$H_{sin}(x) = \frac{L_y}{30}\sin(x+10) + \frac{L_y}{8} \tag{2.3.12}$$

The arbitrary initial vector field is horizontal and constant and given by

$$(u,v)_{sin} = (1,0). \tag{2.3.13}$$

The initial velocity field does not satisfy the slip condition at the terrain boundary, hence we expect some divergence at the terrain surface, as shown by Figure 2.17. Notice that the streamlines of the initial vector field are only horizontal and that some streamlines are missing in the second cavity. This is due to the fact that the particles near the terrain surface that trace the streamlines reach the terrain surface at some point and remember that the velocity vector field is defined as zero under the topography. The vector field is well corrected since the streamlines of the final vector field are parallel to the topography and the final divergence is zero everywhere.

The convergence of the error for the two components of the final vector field is shown in Figure 2.18.

The convergence rates of the absolute error e_N in the different norms are shown in the following tables.

e_N	L_∞	L_1	L_2
u	2.21	1.96	2.05
v	2.06	2.09	2.06

TAB. 2.4. Convergence rate of staggered absolute error e_N for $(u^d, v^d)_{sin}$.

e_N	L_∞	L_1	L_2
u	–	1.82	1.57
v	–	1.81	1.12

TAB. 2.5. Convergence rate of cell-centered absolute error e_N for $(u^d, v^d)_{sin}$.

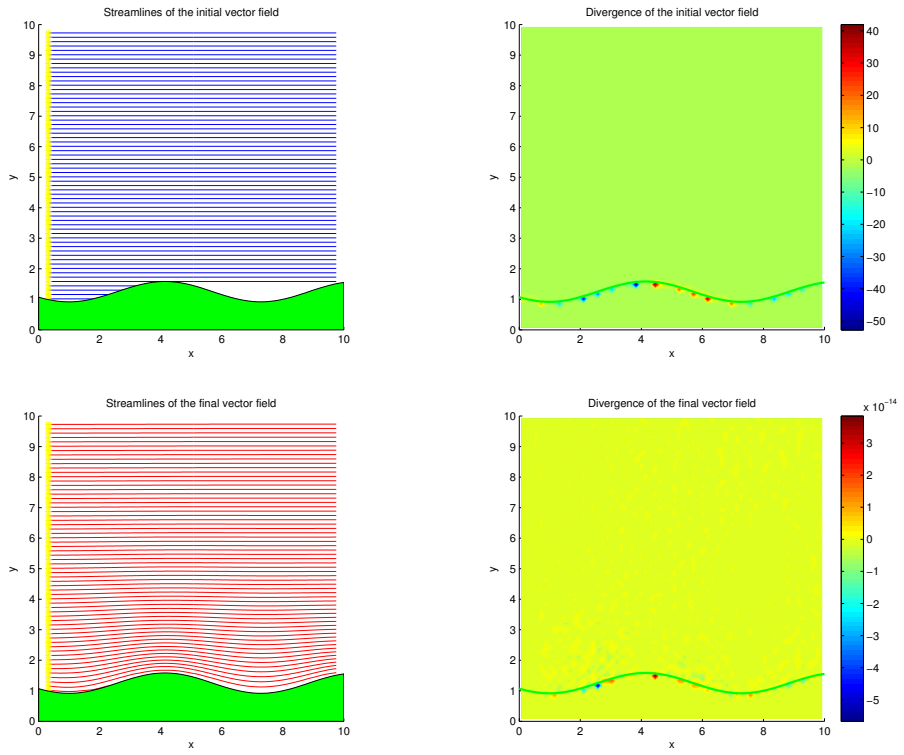


FIG. 2.17. Streamlines and divergence of initial and final vector fields $(u, v)_{sin}$ over the sinusoidal hills H_{sin}.

We notice that the max norm does not converge in the lower graph. The max norm is reached in a cell near the terrain surface. The most plausible explanation for the non-convergence in this norm is the change in the geometry of the topography for each discretization of the grid. Since the geometry is not exactly the same for two different discretizations this would affect the averaging of the reference solution on a coarser grid done with (2.3.8) and thus affect the convergence of the solution.

For the other volume-weighted norms, the convergence rate is a bit lower than what we have seen in the flat terrain case since the geometry is more complicated. The norms of the staggered errors are better though, which might be explained by the fact that the initial velocity field was constant rather than accelerating as in the flat terrain case.

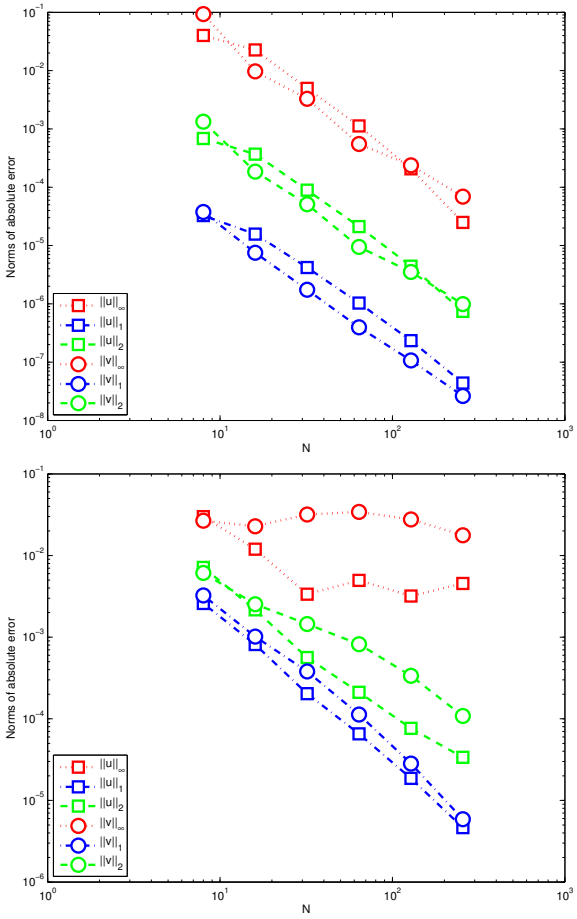


FIG. 2.18. Norms of the staggered absolute error e_N (top) and volume-weighted norms of the cell-centered absolute error e_N (bottom) of $(u^d, v^d)_{sin}$.

2.3.3. Exponential hill case

The two previous test cases used relatively smooth topographical shapes. We now want to test the EBM algorithm over a more complicated geometry. The topography is defined by

$$H_{exp}(x) = 4\exp\left(-\left(x - \frac{L_x}{2}\right)^2\right) + \frac{L_y}{10} \tag{2.3.14}$$

with the highest peak reaching half of the vertical domain length. Sherman [**24**] remarks that the topography should not reach the top boundary of the computational domain, which makes sense. Ratto et al. [**20**] adds that the top domain boundary should be *high enough* over the topography in order to obtain good results.

We once more use a constant horizontal initial vector field given by

$$(u, v)_{exp} = (1, 0). \tag{2.3.15}$$

As in the case of the sinusoidal hills, the region where the computed divergence of the initial velocity field is not zero will be near the terrain surface and the resulting streamlines will be horizontal. This is shown in Figure 2.19.

We remember that Ross et al. [**21**] and Barnard et al. [**2**] insisted that the slip condition should be applied on the initial velocity field with the homogeneous Neumann boundary condition at the terrain surface for satisfying the impenetrability constraint. Since we initialized our velocity field with a horizontal vector field, we should therefore observe a final vector field that is not tangent to the terrain surface. Figure 2.20 shows that it is not the case and that the slip condition is satisfied on the topography. This might be an advantage of the EBM over the FDM.

We now verify if the coarse shape of the geometry has some repercussions on the convergence of the final vector field. Convergence of the absolute error in the different norms is presented in Figure 2.21.

The convergence rates of the absolute error for different norms of figure 2.21 are shown in the following tables.

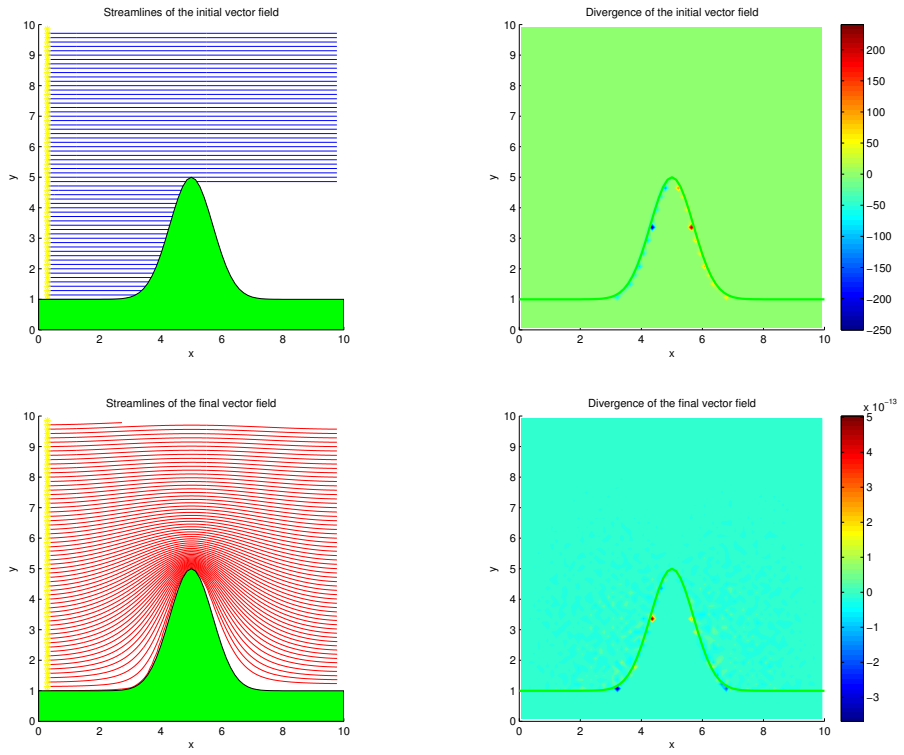


FIG. 2.19. Streamlines and divergence of the initial and final vector fields $(u, v)_{exp}$ over exponential hill H_{exp}.

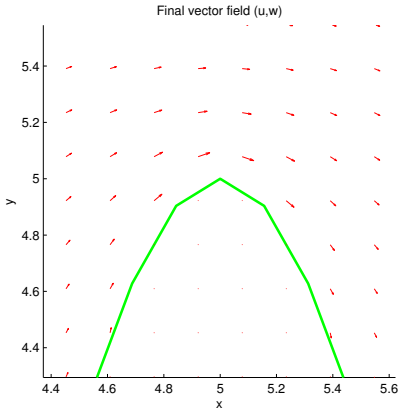


FIG. 2.20. Final vector fields $(u, v)_{exp}$ over the exponential hill H_{exp}.

The max norm still does not seem to converge when only full cells are taken into account. The non-convergence in this norm is not caused by small volume cells, since these are not used when the max norm is computed. It can be shown that even if those cells are not taken into account, the maximum of the error is

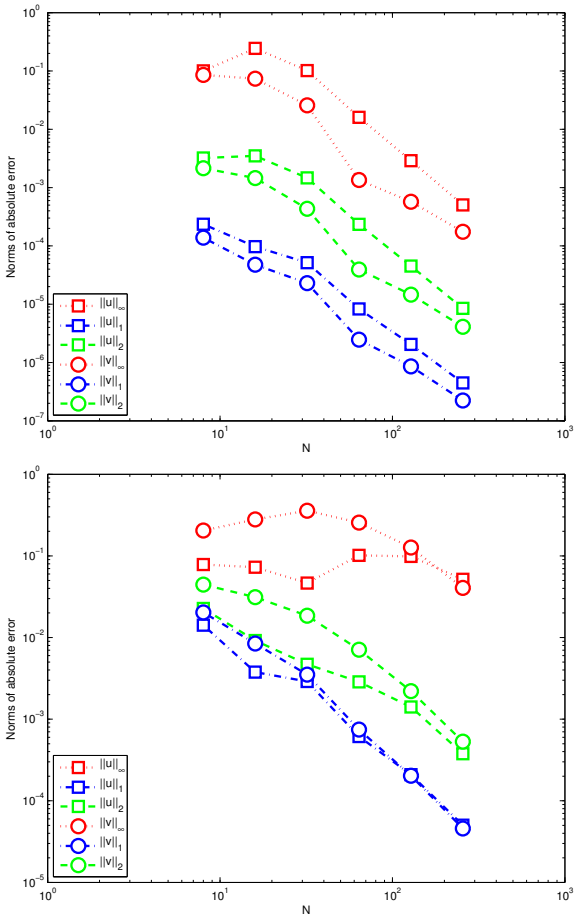


FIG. 2.21. Norms of the staggered absolute error e_N (top) and volume-weighted norms of the cell-centered absolute error e_N (bottom) of $(u^d, v^d)_{exp}$.

e_N	L_∞	L_1	L_2
u	0.85	1.82	1.57
v	0.59	1.81	1.12

TAB. 2.6. Convergence rate of staggered absolute error e_N for $(u^d, v^d)_{exp}$.

e_N	L_∞	L_1	L_2
u	–	1.62	1.12
v	–	1.80	1.35

TAB. 2.7. Convergence rate of cell-centered absolute error e_N for $(u^d, v^d)_{exp}$.

always reached in the cells close to the terrain surface. Even if the max norm does not converge, the generated velocity field is still incompressible since it has

zero-divergence and can be used for wildfire spread simulations. In actual models for wildfires spread, the wind velocity which is considered for the propagation of the fire is always taken from a certain height above the topography, hence we don't need to worry much about the fact that the max norm of the absolute error is not converging.

For the L_2-norm, the rate of convergence is now about $\mathcal{O}(h)$. The exponential topography has a greater impact on the final velocity field as was expected.

2.3.4. Half-cylinder hill case

We now want to check if our numerical solution converges to an exact solution. Exact solutions for the Poisson equation are known for very simple geometries such as the half-cylinder. Final velocity fields generated from uniform constant initial velocity fields are expected to generate a potential-like flow. Hence, from an initial vector field, we will try to generate a potential flow for which the analytic solution is known. Wang et al. [**25**] and Ross et al. [**21**] used a uniform velocity field in conformal coordinates to achieve this. We will see that good results can be obtained in Cartesian coordinates when solving with the EBM.

Our test case will be to find the potential flow around a half-cylinder. We use conventional polar coordinates (r, θ) to solve the following Laplace equation :

$$\Delta\varphi = 0 \tag{2.3.16}$$

with the following boundary conditions :
Far away from the cylinder of radius R, i.e. $r/R \gg 1$, the flow is assumed to be only horizontal :

$$\nabla\varphi = (u, 0) \tag{2.3.17}$$

and on the cylinder surface, i.e. $r = R$, the slip condition must be fulfilled :

$$\mathbf{v} \cdot \mathbf{n} = \nabla\varphi \cdot \mathbf{n} = 0 \tag{2.3.18}$$

$$\Rightarrow \frac{\partial\varphi}{\partial r} = 0. \tag{2.3.19}$$

The exact solution of this problem is given by the potential φ :

$$\varphi(r, \theta) = u\left(r + \frac{R^2}{r}\right)\cos\theta. \tag{2.3.20}$$

The vector field can be found first of all in polar coordinates :

$$v_r = \frac{\partial\varphi}{\partial r} = u\left(1 - \frac{R^2}{r^2}\right)\cos\theta \tag{2.3.21}$$

$$v_\theta = \frac{\partial\varphi}{\partial\theta} = -u\left(1 + \frac{R^2}{r^2}\right)\sin\theta \tag{2.3.22}$$

but since we are using Cartesian coordinates for our computation, we will use the vector field (u, v) :

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} v_r \\ v_\theta \end{bmatrix}. \tag{2.3.23}$$

Once again, the convergence will be tested in square domains. We will compare the vector field components of the numerical solution (u^d, v^d) with the exact solution (u^{exact}, v^{exact}) by calculating the absolute error of the difference. For example,

$$e_N(u^d) = \left| u_N^{exact} - u_N^d \right|. \tag{2.3.24}$$

Then we will take the volume-weighted norms of $e_N(u^d)$ defined earlier. All this is also done for v^d.

In the articles [**21**] and [**25**], the authors use a uniform background wind in conformal coordinates as the initial vector field. They say that since $\nabla \cdot \mathbf{v} = 0$, the Poisson equation will then be reduced to the Laplace equation $\Delta\varphi = 0$ so the numerical solution should converge to the solution of potential flow. Ross et al. use terrain-following coordinates to satisfy the slip condition on the terrain boundary and solve the Laplace equation. In our case, we use the initial vector field given by

$$(u, v)_{cyl} = (1, 0) \tag{2.3.25}$$

and the topography is defined by

$$H_{cyl}(x) = \sqrt{R^2 - \left(x - \frac{L_x}{2}\right)^2} \tag{2.3.26}$$

with $R = \dfrac{L_x}{8}$.

We already know that the divergence of the initial velocity field will not be zero everywhere since the vector field is not tangential to the half-cylinder boundary. This can be seen in Figure 2.22. In this figure, the streamlines of the final velocity field are drawn in red and those of the exact velocity field in black. As we can see, the profiles drawn by the streamlines are very similar, but this does not mean that the numerical solution converges to the analytic solution.

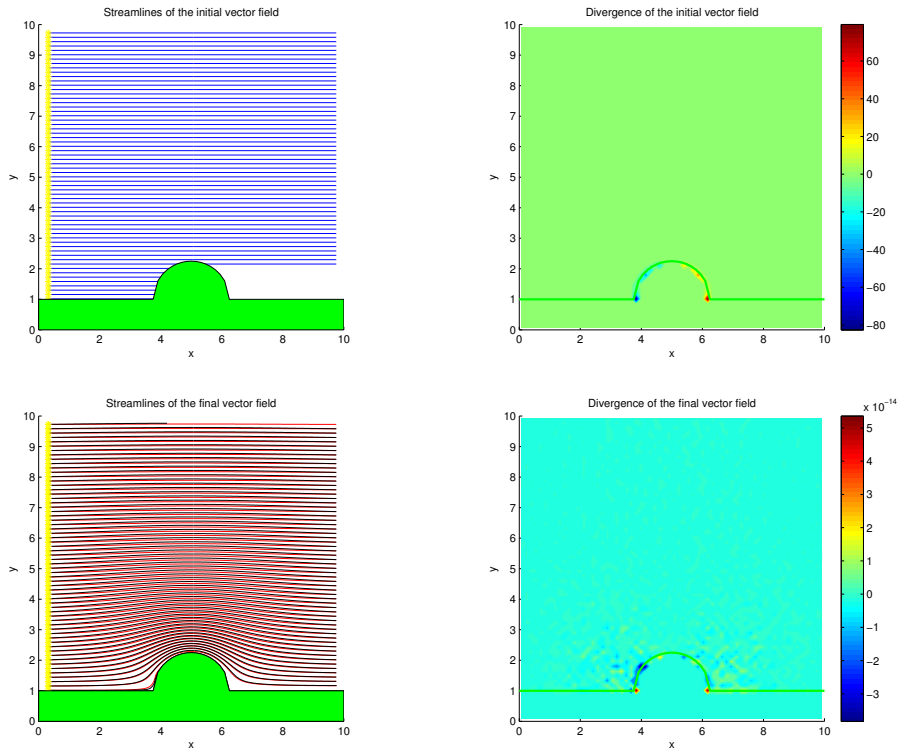


FIG. 2.22. Streamlines and divergence of the initial and final vector fields $(u, v)_{cyl}$ over half-cylinder hill H_{cyl}.

Figure 2.23 shows the absolute error of the numerical solution with the exact solution in different volume-weighted norms. We can therefore conclude that there is no convergence. This might occur for different reasons. First, the exact solution

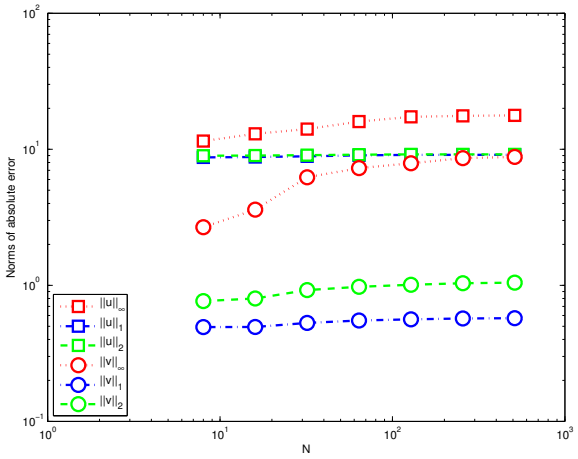


FIG. 2.23. Volume-weighted norms of the cell-centered absolute error e_N of $(u^d, v^d)_{cyl}$.

is found when solving the Laplace equation. In our case, we are solving a Poisson equation since the initial divergence is not zero over the whole domain. In the mathematical problem, one of the boundary conditions assumes that the initial velocity must be only horizontal at infinity. Our solution is found with Dirichlet boundary conditions on a finite domain. The size of the domain might be too small for the cylinder radius. Probably the two solutions would be even closer for a smaller cylinder radius. However, even if the solution does not converge, the resulting numerical flow looks very much like a potential flow.

Figure 2.24 shows the contour plot of the horizontal components u of the final vector field and exact vector field. The profile is very similar except in magnitude. This is even more obvious when looking at Figure 2.25 which shows the profile

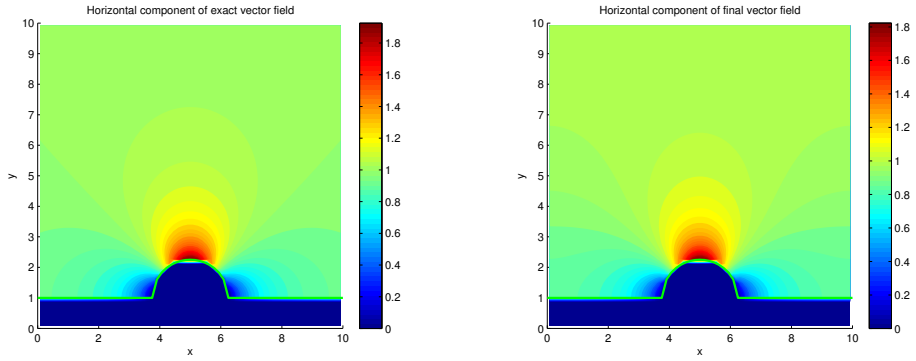


FIG. 2.24. Contour plot of horizontal component of numerical and exact velocity fields over a half-cylinder hill H_{cyl}.

of the components of the numerical vector field over the exact vector field. We see that the profiles have quite the same shapes but that the magnitude of the velocity is different, particularly for the component u.

Figure 2.26 shows the numerical and exact vector fields close to the surface of the half-cylinder. The vector fields do not match perfectly close to the terrain surface, which might contribute to the errors between the numerical and exact solutions. This difference might be an artefact due to the average of the numerical staggered velocity field since no extrapolation is done in zero volume cells when computing the average in the cut cells.

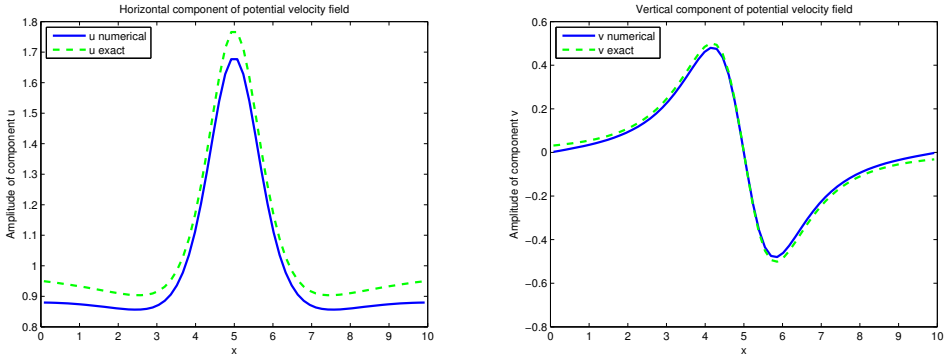


FIG. 2.25. Numerical and exact velocity magnitude at $y = 2.34$ over a half-cylinder hill H_{cyl}.

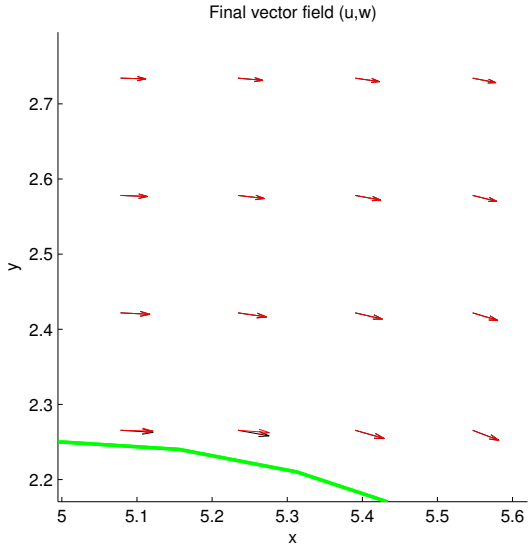


FIG. 2.26. Final vector field $(u^d, v^d)_{cyl}$ (red) and exact vector field (black) on half-cylinder hill given by H_{cyl}.

2.4. Fire spread by an incompressible flow

In this section, we explain how the fire is moved by a mass-consistent wind. First, a divergence-free vector field $\mathbf{v}_d$ is generated from a given initial wind vector field $\mathbf{v}$ which fulfils the slip condition on the terrain surface. The global velocity $\mathbf{v}_{tot}$ of the fire is given by

$$\mathbf{v}_{tot} = \mathbf{v}_d + \mathbf{v}_{fire}$$

where $\mathbf{v}_d$ is the divergence-free wind vector field and $\mathbf{v}_{fire}$ some constant velocity for the fire itself moving on the topography. In case of absence of wind, the fire would move at constant speed $\mathbf{v}_{fire}$. Since the vector field $\mathbf{v}_d$ is tangent to the topography, only the horizontal component u^d is needed.

$$u^{tot}_{i+\frac{1}{2},j} = u^d_{i+\frac{1}{2},j} + u^{fire}_{i+\frac{1}{2},j}$$

where $u^{fire}_{i+\frac{1}{2},j} = \text{proj}_{\mathbf{x}}\mathbf{v}_{fire}$ and where j is the lowest vertical index where $u_{i+\frac{1}{2},j}$ is non-zero.

Euler's method is used to simulate the fire propagation :

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}_{tot} \tag{2.4.1}$$

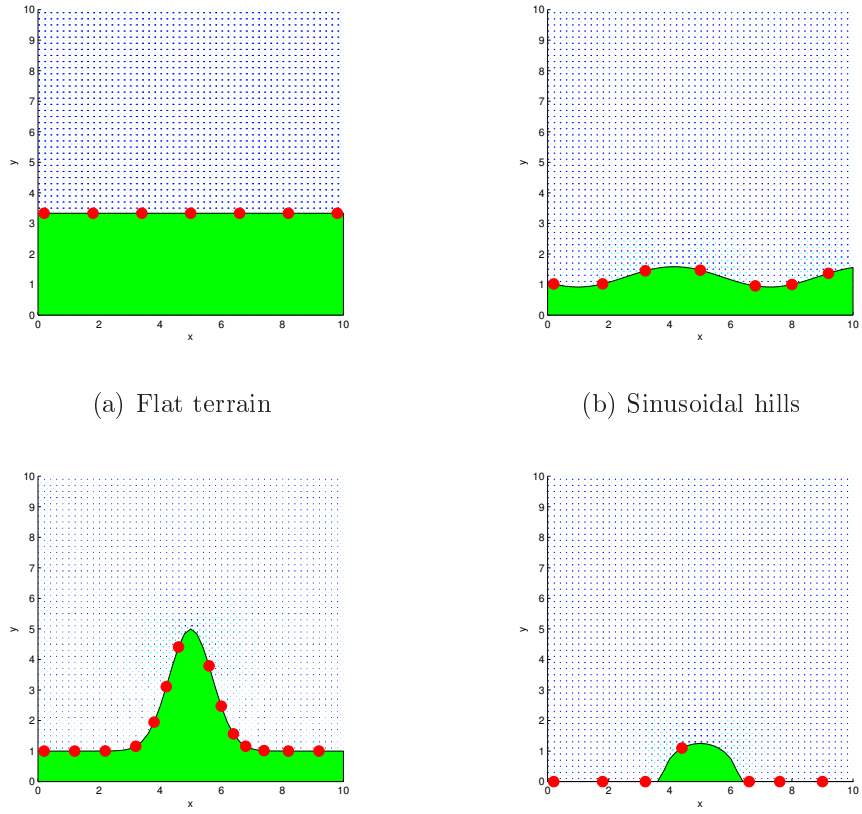
$$\frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t} = \mathbf{v}_{tot} \tag{2.4.2}$$

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \Delta t \mathbf{v}_{tot} \tag{2.4.3}$$

Figure 2.27 shows the evolution of the position of a fire represented by a point (•) and starting on the left side of the domain. The fire is pictured at every time step $\Delta t = 1$ for different terrain geometries on a 50×50 grid. The initial vector field is definded as $\mathbf{v} = (1, 0)$ and norm of the fire velocity $|\mathbf{v}_{fire}| = 0.5$.

2.4.1. Wind effect depending on its altitude

We now illustrate how the wind affects the fire spread depending on the height at which it is chosen. Here, we experiment with this effect using the sinusoidal and exponential hills. In both cases, the initial wind vector field is uniform, $\mathbf{v} = (1, 0)$, and the fire velocity is $|\mathbf{v}_{fire}| = 0.5$. The corrected wind vector field is calculated on a uniform 50×50 grid and $\Delta t = 1$. Figure 2.28 shows that the wind at different



(a) Flat terrain

(b) Sinusoidal hills

(c) Exponential hill

(d) Cylindrical hill

FIG. 2.27. Fire spreading over different terrain shapes (wind inflow is on the left side).

heights over the topography has little influence on the fire spread, which is not the case with the exponential topography in figure 2.29.

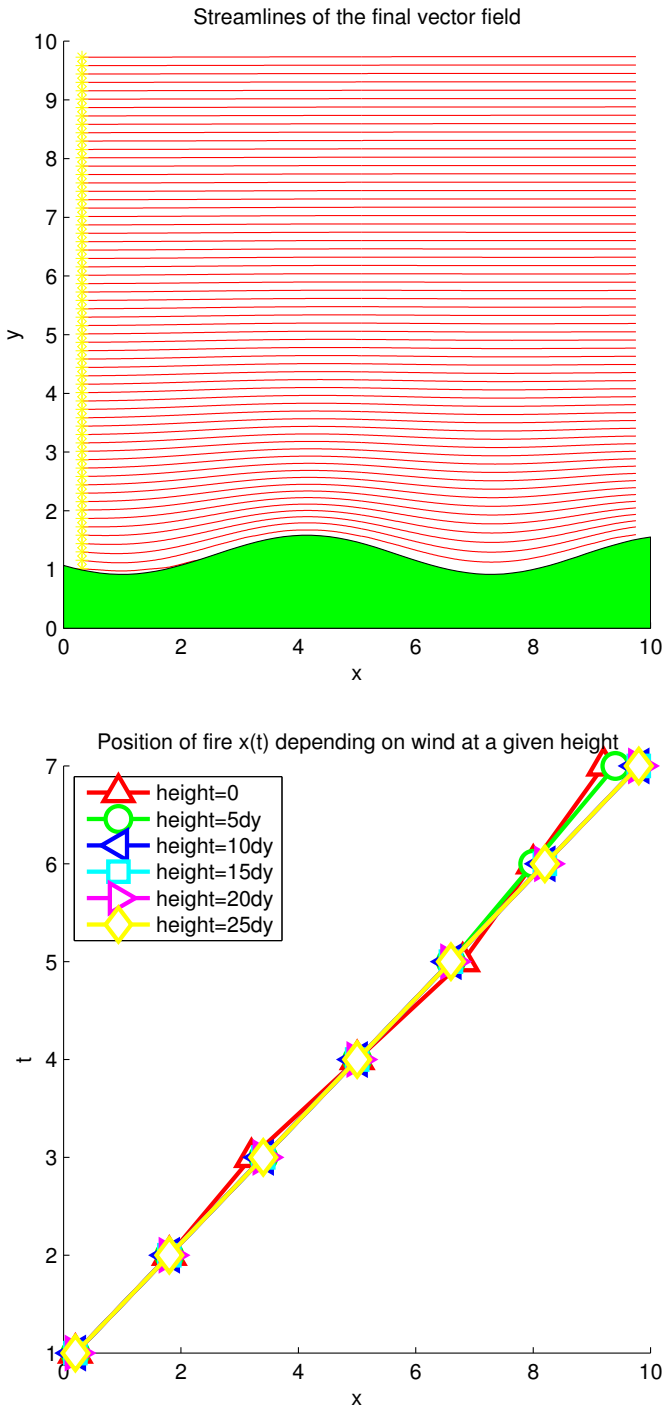


FIG. 2.28. Fire position $x(t)$ over sinusoidal hills with different heights of the wind.

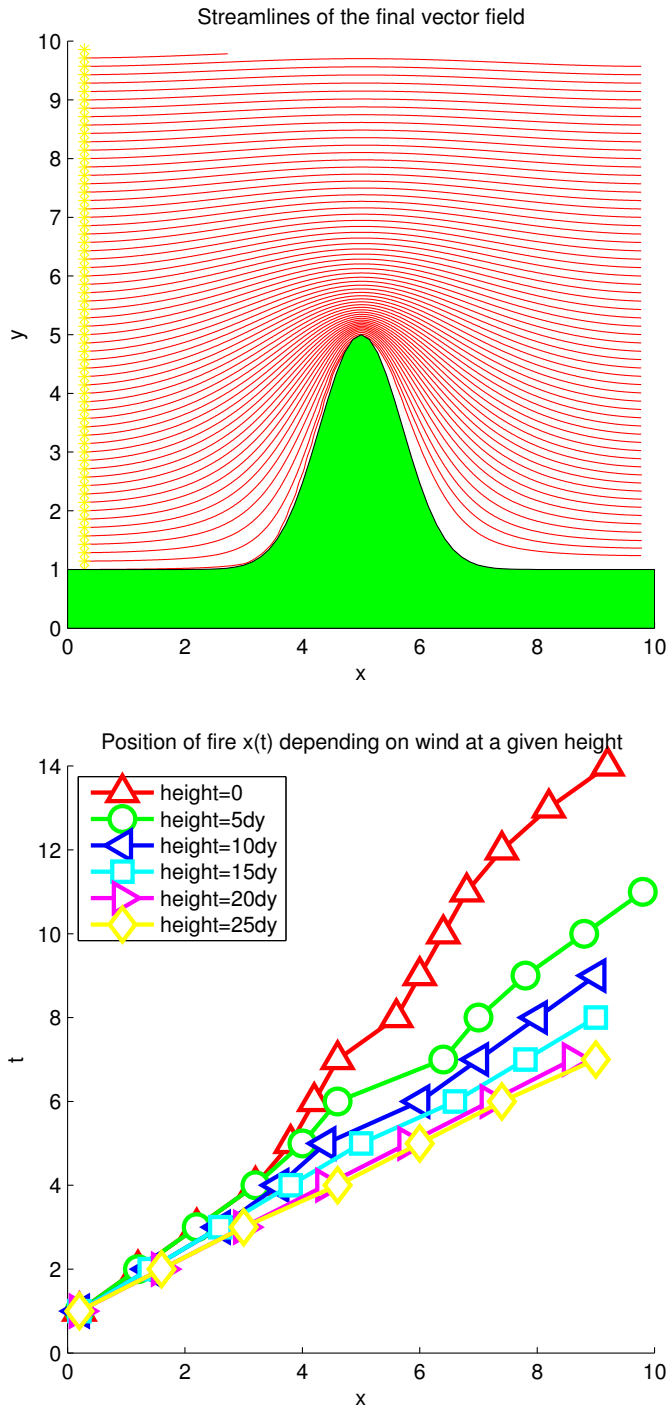


FIG. 2.29. Fire position $x(t)$ over an exponential hill with different heights of the wind.

Chapitre 3

SOLVING THE MODEL IN 3 DIMENSIONS

We now generalize the model for the three dimensional case. This part of the project was achieved with the use of EBChombo, a software developed by the ANAG team at the Lawrence Berkeley National Laboratory. EBChombo is a collection of *C++* classes with FORTRAN subroutines for the construction of numerical PDE algorithms in complex geometries using the Embedded Boundary Method. It is based on Chombo which also provides tools for solving PDE such as Adaptive Mesh Refinement (AMR) and allows the use of parallel computing.

We first explain how EBM is implemented in EBChombo and more precisely how the embedded boundary and the operators are discretized to 3D. Then the test cases of the previous chapter are generalized in 3D for testing the convergence of the algorithm.

3.1. Embedded Boundary Method

We recall that the aim of the Embedded boundary method here is to solve the elliptic equation :

$$\nabla \cdot \nabla \varphi = -2 \nabla \cdot \mathbf{v} \tag{3.1.1}$$

on an irregular domain in three dimensions with homogeneous Neumann condition at the embedded boundary and homogeneous Dirichlet condition at domain boundaries. This chapter generalizes the algorithm based on the finite volume method explained in chapter two. The general idea of the approach is well explained in the article of Schwartz et al. [**23**].

3.1.1. Discretization of the domain

The underlying discretization of space is given by rectangular cell-centered control volumes on a Cartesian grid $\Upsilon_{\mathbf{i}} = [(\mathbf{i} - \frac{1}{2}\mathbf{u})h_d, (\mathbf{i} + \frac{1}{2}\mathbf{u})h_d]$, $\mathbf{i} \in \mathbb{Z}^D$ where h_d is the mesh spacing in the $d-$direction, and $\mathbf{u} \in \mathbb{Z}^D$ is the vector with all components equal to one.

The geometry of the irregular domain Ω is represented by its intersection with the Cartesian grid. The method uses control volumes $V_{\mathbf{i}} = \Upsilon_{\mathbf{i}} \cap \Omega$ and faces $A_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d}$ which are the intersections of $\partial V_{\mathbf{i}}$ with the coordinate planes $\{\mathbf{x} : x_d = (i_d \pm \frac{1}{2})h_d\}$. Here $\mathbf{e}_d$ is the unit vector in the $d-$direction. The intersection of the boundary of the irregular domain and the Cartesian control volumes are the faces $A^B_{\mathbf{i}} = \Upsilon_{\mathbf{i}} \cap \partial\Omega$.

The discretized divergence operator requires some geometric data on the control volume cells such as areas and volumes written in nondimensional terms :

$$\text{volume fractions} : \kappa_{\mathbf{i}} = |V_{\mathbf{i}}| \prod_{d=1}^{D} \frac{1}{h_d}$$

$$\text{face apertures} : \alpha_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d} = |A_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d}| \prod_{d' \neq d} \frac{1}{h_{d'}}$$

$$\text{boundary apertures} : \alpha^B_{\mathbf{i}}.$$

It is also assumed that these values can be calculated with accuracy of $\mathcal{O}(h^2)$. Location of centroids and the average outward normal can also be computed with these explicit formulas :

$$\text{face centroid} : \mathbf{x}_{\mathbf{i}+\frac{1}{2}\mathbf{e}_d} = \frac{1}{|A_{\mathbf{i}+\frac{1}{2}\mathbf{e}_d}|} \int_{A_{\mathbf{i}+\frac{1}{2}\mathbf{e}_d}} \mathbf{x} dA$$

$$\text{boundary face centroid} : \mathbf{x}^B_{\mathbf{i}} = \frac{1}{|A^B_{\mathbf{i}}|} \int_{A^B_{\mathbf{i}}} \mathbf{x} dA$$

$$\text{outward normal} : \mathbf{n}^B_{\mathbf{i}} = \frac{1}{|A^B_{\mathbf{i}}|} \int_{A^B_{\mathbf{i}}} \mathbf{n}^B dA$$

where $\mathbf{n}^B$ is the outward normal to the boundary $\partial\Omega$ defined for each point on $\partial\Omega$. Again, the accuracy of these computed quantities is assumed to be $\mathcal{O}(h^2)$.

3.1.2. Embedded Boundary representation

EBChombo uses an implicit function $\phi : \mathbb{R}^D \to \mathbb{R}$ for embedding the irregular geometry in the Cartesian grid and defining the computation domain Ω :

$$\Omega = \{\mathbf{x} : \phi(\mathbf{x}) < 0\} \tag{3.1.2}$$

$$\partial\Omega = \{\mathbf{x} : \phi(\mathbf{x}) = 0\}. \tag{3.1.3}$$

This approach has been chosen by the ANAG team for different reasons, namely the easy computation of functions such as the normal at the boundary and representation of sophisticated geometries. Moreover, the moments which are used for computing the centroids can be found using the divergence theorem, Taylor expansions, least squares, recursion, and 1D root finding. An advantage is that an explicit representation of the irregular domain and its boundary is never needed nor computed. The article of Ligocki et al. [**16**] explains in more detail how this is achieved.

In our case, we assume in the 2D model that the topography was given by an analytic function $H : \mathbb{R}^{D-1} \to \mathbb{R}$ where $D = 2$. In order to represent such functions in EBChombo, we will use an implicit function of the form $\phi(x, y, z) = H(x, y) - z$.

3.1.3. Divergence operator

The divergence operator $\nabla\cdot$ is discretized using the divergence theorem as before. Let $\mathbf{F} = (F^1, \ldots, F^D)$ be a function of $\mathbf{x}$. By the midpoint rule we have :

$$\nabla \cdot \mathbf{F} \approx \frac{1}{|V_{i,j}|} \int_{V_i} \nabla \cdot \mathbf{F} dV = \frac{1}{|V_{i,j}|} \oint_{\partial V_i} \mathbf{F} \cdot \mathbf{n} dA \tag{3.1.4}$$

$$\approx \frac{1}{\kappa_{\mathbf{i}} h} \left[\left(\sum_{\pm=+,-} \sum_{d=1}^{D} \pm \alpha_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d} F^d(\mathbf{x}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d}) \right) + \alpha_{\mathbf{i}}^B \mathbf{n}_{\mathbf{i}}{}^B \cdot \mathbf{F}(\mathbf{x}_{\mathbf{i}}{}^B) \right]. \tag{3.1.5}$$

Once again, the homogeneous Neumann boundary condition is applied on the embedded boundary, meaning that $\mathbf{F}(\mathbf{x}_{\mathbf{i}}{}^B) = \mathbf{0}$. Hence, the discrete divergence operator becomes

$$D^M(\mathbf{F})_{i,j} = \frac{1}{\kappa_{\mathbf{i}} h} \left(\sum_{d=1}^{D} \alpha_{\mathbf{i}+\frac{1}{2}\mathbf{e}_d} F^d_{\mathbf{i}+\frac{1}{2}\mathbf{e}_d} - \alpha_{\mathbf{i}-\frac{1}{2}\mathbf{e}_d} F^d_{\mathbf{i}-\frac{1}{2}\mathbf{e}_d} \right) \tag{3.1.6}$$

3.1.4. Flux interpolation

In order to use the divergence operator on the volume control cells next to the boundaries of the domain, the fluxes on the boundaries must be specified. At the embedded boundary, the problem uses a homogeneous Neumann condition $\frac{\partial \varphi}{\partial n} = 0$, which means that the flux is simply zero, $\mathbf{F}(\mathbf{x_i}^B) = \mathbf{0}$.

For the top and side domain boundaries, the homogeneous Dirichlet condition $\varphi = 0$ is applied as in the 2D case. As we have seen before, this kind of condition does not prescribe a particular value for the flux at the boundary, hence the flux must be interpolated. We use the same three-point gradient stencil as in equation (2.1.11) of the two-dimensional problem in order to get a specific value of the flux on those boundaries.

For example, the flux on the right boundary in the direction $d = 1$ is given by

$$F_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1} = \frac{1}{h_1}\left[3\left(\varphi^B - \varphi_{\mathbf{i}}\right) - \frac{1}{3}\left(\varphi^B - \varphi_{\mathbf{i}-\mathbf{e}_1}\right)\right]. \tag{3.1.7}$$

where φ^B is the value of φ at the boundary.

The divergence operator requires fluxes to be defined on the face centroid of the volume cells. For cut cells, the flux must be specified at the centroid of the face cells. In order to achieve this, EBChombo applies a bilinear interpolation using the value of the fluxes in the neighbouring cells. A bilinear interpolation can be seen as a composition of two linear interpolations.

For instance, given a face with outward normal $\mathbf{e}_1$ with centroid $\mathbf{x}$, the flux $F^d_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1}$ in the $d-$direction for $d \neq 1$ is linearly interpolated by :

$$F^d_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1} = \eta\frac{(\varphi_{\mathbf{i}+\mathbf{e}_1} - \varphi_{\mathbf{i}})}{h} + (1-\eta)\frac{(\varphi_{\mathbf{i}+\mathbf{e}_1\pm\mathbf{e}_d} - \varphi_{\mathbf{i}\pm\mathbf{e}_d})}{h} \tag{3.1.8}$$

$$\eta = 1 - \frac{|\mathbf{x}\cdot\mathbf{e}_d|}{h} \tag{3.1.9}$$

$$\pm = \begin{cases} + & \mathbf{x}\cdot\mathbf{e}_d > 0 \\ - & \mathbf{x}\cdot\mathbf{e}_d \leq 0 \end{cases}. \tag{3.1.10}$$

This part is illustrated by the red dashed lines in Figure 3.1. Then, the flux is interpolated at the centroid using a linear interpolation of the two previous

linearly interpolated fluxes, which results in the bilinear interpolation :

$$F_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1} = \eta F^d_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1} + (1-\eta) F^d_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1 \pm \mathbf{e}_{d'}} \tag{3.1.11}$$

$$\eta = 1 - \frac{|\mathbf{x}\cdot\mathbf{e}_{d'}|}{h} \tag{3.1.12}$$

$$\pm = \begin{cases} + & \mathbf{x}\cdot\mathbf{e}_{d'} > 0 \\ - & \mathbf{x}\cdot\mathbf{e}_{d'} \leq 0 \end{cases} . \tag{3.1.13}$$

where $d' \neq d$ and $d' \neq 1$. This last step is pictured by the blue dashed line in Figure 3.1 which gives an example of the bilinear interpolation of the flux $F_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1}$ at the centroid in the case where $d = 2$ and $d' = 3$.

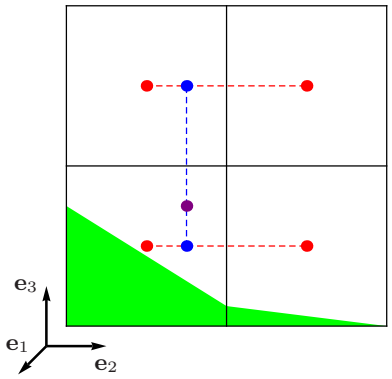


FIG. 3.1. Interpolation of the flux $F_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1}$ at the centroid of the cut face cell edge using bilinear interpolation. The scheme here is for the case where $d = 2$ and $d' = 3$.

Note that for cut faces where the distance between the face centroid and the face center only depends on one of the two tangential components of the face, the bilinear interpolation reduces to a simple linear interpolation as shown in Figure 3.2.

As mentioned by Schwartz et al. [**23**], this is a nontrivial choice for computing the fluxes on cut face cells. In fact, experiments have shown that this choice of bilinear interpolation assures the stability of the method for all test cases, especially for some configurations of adjacent small control volumes, which was not true when using a more simple linear interpolation.

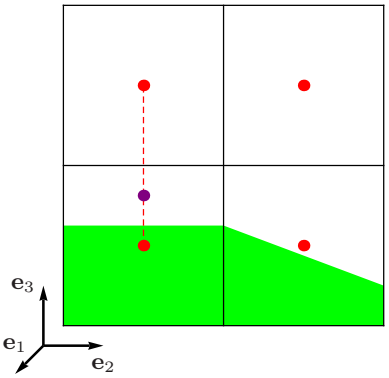


FIG. 3.2. Interpolation of the flux $F_{\mathbf{i}+\frac{1}{2}\mathbf{e}_1}$ at the centroid of the cut cell face using linear interpolation when the distance between the face centroid and the face center only varies in the tangential direction $\mathbf{e}_3$.

3.1.5. MAC Projection

We give some details on the class `EBCompositeMACProjector` which was used to obtain the divergence-free velocity field from an initial velocity field using the MAC projection and the Embedded Boundary Method. EBChombo is built so that it takes advantage of the adaptive mesh refinement (AMR) multigrid algorithm developed by Martin and Cartwright [**17**]. Since the 2D version has not been implemented with AMR, the tests in 3D use only one level, meaning that none of the multigrid features were used here.

The functions in the class first compute the right hand side of equation (3.1.1) $\rho = -2\nabla \cdot \mathbf{v}$. Then the elliptic equation $\Delta\varphi = \rho$ is solved with the Embedded Boundary Method on a Cartesian grid. Finally, the projection is completed by doing the correction :

$$\mathbf{v}^d = \mathbf{v} + \frac{1}{2}\nabla(\Delta^{-1}\nabla \cdot \mathbf{v}). \tag{3.1.14}$$

Some parameters must be defined in order to use the class correctly. For all simulations, a number of 40 iterations were done for pre-conditioning. The Gauss-Seidel method was used as the relaxation method and the minimum residual method for solving the elliptic equation.

3.2. Convergence and error analysis

As in chapter two, we study the efficiency and accuracy of the Embedded Boundary Method. The algorithm will be used for solving constant or particular initial velocity fields over different geometries.

When the exact solution is unknown, we study the convergence of the algorithm using a reference solution which is solution on a grid of very fine resolution. Then, we compute the error between the solutions of the coarsest grids and the reference fine grid. In our case, we are interested by the convergence of the velocity field. Since the norms are defined for cell-centered quantities, we average the edge-centered velocity field at the center of each cell. Then, the reference solution must be averaged to the coarse grid using a volume-weighted average available in the `EBCoarseAverage` class :

$$Av(\xi^f) = \frac{1}{V^c} \sum_{v_f \in \mathcal{F}} V^f \xi_{v_f} \tag{3.2.1}$$

where $\mathcal{F}$ is the set of cells of the fine grid v_f contained in the cell of the coarse grid v_c and where V^f and V^c are the volume of cells v_f and v_c respectively. Figure 3.3 shows the cells v_f of the fine grid $N = 8$ that are contained in the coarse cell v_c of the grid $N = 2$.

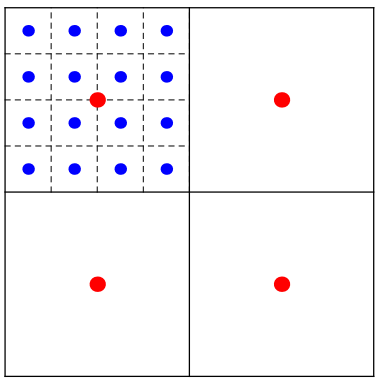


Fig. 3.3. The fine to coarse averaging of a cell-centered quantity using a volume-weighted average of values of the fine grid $N = 8$ to the coarse grid $N = 2$.

The absolute error for the component u is given by

$$e_N(u_N) = \left| Av(u_{N_{ref}}) - u_N \right|. \tag{3.2.2}$$

where N_{ref} is the number of cells of the reference grid and N the number of cells of a coarser grid. The absolute error is computed for the other components of the corrected vector field $\mathbf{v}_d = (u^d, v^d, w^d)$. Then, we check if the errors converge for different norms. We use the volume-weighted norm defined in the Johansen and Colella article [**14**], which is available in the `EBArith` class. For a cell-centered variable ξ, the max norm is :

$$\|\xi\|_\infty = \max_{(i,j,k)\in\Omega} |\xi_{i,j,k}|. \tag{3.2.3}$$

and the $p-$norm :

$$\|\xi\|_p = \left(\sum_{(i,j,k)\in\Omega} |\xi_{i,j,k}|^p \kappa_{i,j,k} h^D \Big/ \sum_{(i,j,k)\in\Omega} \kappa_{i,j,k} h^D \right)^{1/p} \tag{3.2.4}$$

where Ω is the computational domain and $\kappa_{i,j,k}$ the volume fraction of each control cell.

For the 3D test cases, the norms of the absolute error were taken over all **non-zero** volume cells.

All convergence tests are performed on square domains :

$$L = L_x = L_y = L_z$$

$$N = N_x = N_y = N_z$$

$$h = h_x = h_y = h_z.$$

and where the vertical axis is given by y. For all our tests, $L = 10$ and we use as a reference solution the results obtained on the grid $N_{ref} = 128$ and the coarsest grid is $N = 4$.

The solutions of the different cases will be shown with the streamlines of the velocity field and the contour plots for the final divergence and velocity components. Those figures will be done using the following parameters for the discretization of the rectangular domain in all test cases :

N	L
64	10

TAB. 3.1. Parameters used for test cases in 3D.

3.2.1. Flat terrain case

The flat terrain is a simple topography for which the Embedded Boundary Method should not be so different from the Finite Volume Method. The topography is a plane surface defined by

$$H_{flat}(x, y) = \frac{L_y}{3}. \tag{3.2.5}$$

For the tests, we use a horizontal initial velocity field which is accelerating in the $x-$direction :

$$(u, v, w)_{flat} = (100 + x^{\frac{5}{4}}, 0, 0). \tag{3.2.6}$$

Figure 3.4 shows the divergence of the corrected velocity field on three planes crossing the computational domain Ω, the grey horizontal plane being the flat topography.

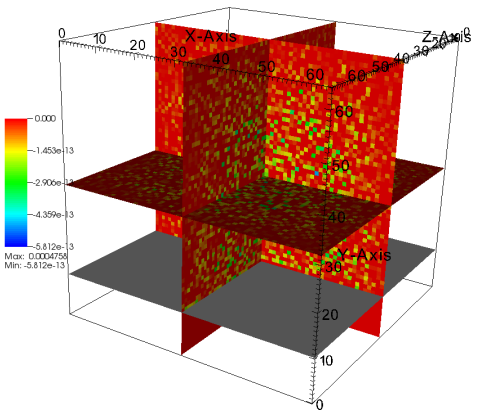


FIG. 3.4. Divergence of the final vector field $(u, v, w)_{flat}$ over the flat terrain H_{flat}.

The convergence of the corrected velocity field with a reference solution is shown in figure 3.5.

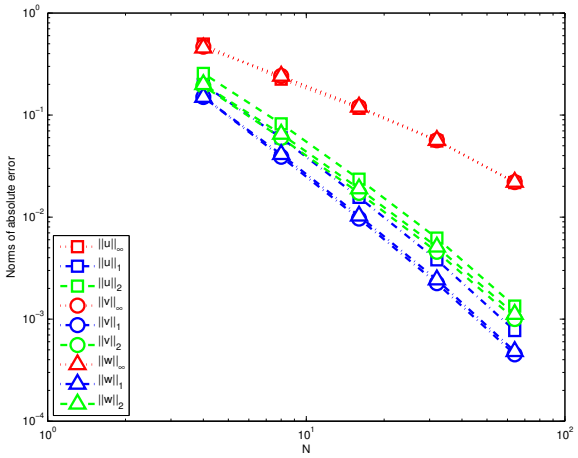


FIG. 3.5. Volume-weighted norms of the cell-centered absolute error e_N of $(u^d, v^d, w^d)_{flat}$.

Using the `polyfit` function in Matlab, we find that the rate of convergence for the three components of the velocity field presented in the following tables. The rate of convergence is about $\mathcal{O}(h^2)$ for the L_1-norm and the L_2-norm. For

e_N	L_∞	L_1	L_2
u	1.11	2.00	1.89
v	1.10	2.09	1.89
w	1.09	2.07	1.87

TAB. 3.2. Convergence rate of $(u, v, w)_{flat}$ in different norms.

the $L_\infty-$norm, the error convergence rate is $\mathcal{O}(h)$. This might be explained by the fact that the largest errors are contained in partial volume cells.

3.2.2. Sinusoidal hills case

As in chapter two, we try our 3D algorithm over a geometry which has a few more features. The topography is defined by the sinusoidal function :

$$H_{sin}(x, y) = \frac{L_y}{30}\sin(x + 10) + \frac{L_y}{8} \tag{3.2.7}$$

and the initial velocity field is horizontal and constant in the $x-$direction :

$$(u, v, w)_{sin} = (1, 0, 0). \tag{3.2.8}$$

The results are similar to those obtained in 2D. We take a closer look to the convergence which is shown in figure 3.6.

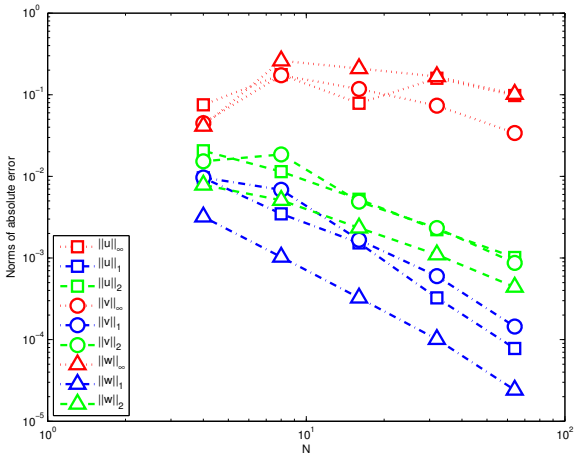


FIG. 3.6. Volume-weighted norms of the cell-centered absolute error e_N of $(u^d, v^d, w^d)_{sin}$.

In this case, we already notice that the rate of convergence in $L_\infty-$norm does not converge. Remember that in the 3D case, the partial volume cells are kept in the computation of the volume-weighted norms, which is not the case in 2D. Since the largest errors occur near the terrain surface, this might explain why the errors don't converge in the $L_\infty-$norm. Moreover, as explained for this test case in chapter two, the geometry of the terrain surface changes with each discretization of the grid which affects the averaging of the reference solution on a coarser grid. This could also explain the non-convergence in this norm.

e_N	L_∞	L_1	L_2
u	–	1.76	1.11
v	–	1.61	1.22
w	–	1.75	1.07

TAB. 3.3. Convergence rate of $(u, v, w)_{sin}$ in different norms.

The rate of convergence in the other two norms is lower that what was seen in the flat terrain test case. In fact, the L_2-norm seems to be more 1^{st} order accurate rather than 2^{nd}. Compared to the results obtained for u and v in the two-dimensional case, it is quite similar but still lower order. Again, this might be caused by the values of the velocity field in the partial volume cells when computing these norms.

3.2.3. Exponential hill case

The last test case aims to challenge the algorithm with a more complex topography. The exponential hill is defined by

$$H_{exp}(x, y) = 4 \exp\left(-\left(x - \frac{L_x}{2}\right)^2\right) + \frac{L_y}{10}. \tag{3.2.9}$$

We use the same initial vector field as before :

$$(u, v, w)_{exp} = (1, 0, 0). \tag{3.2.10}$$

The exponential topography is represented by the grey surface in Figure 3.7 while the 3 other planes show the remaining divergence after correcting the initial vector field. Once again, we notice that the final divergence is zero everywhere.

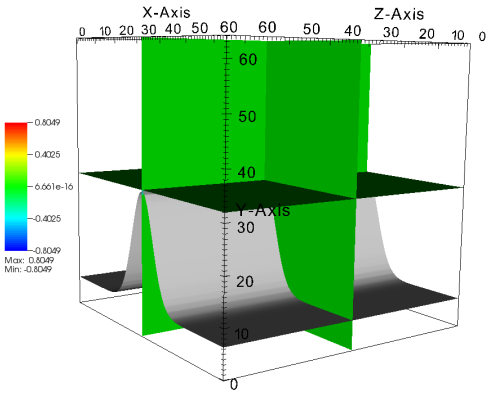


FIG. 3.7. Divergence of the final vector field $(u, v, w)_{exp}$ over exponential hill H_{exp}.

The constant horizontal velocity field is corrected to become the divergence-free velocity field tangent to the topography where wind speed-up can be observed at the top of the hill. This is pictured by the contour plot of the velocity component u in Figure 3.8.

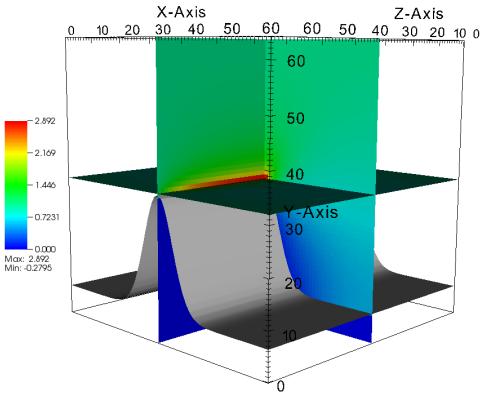


FIG. 3.8. Contour plot of the component u of the final vector field.

Figure 3.9 shows the final vector field over the exponential hill.

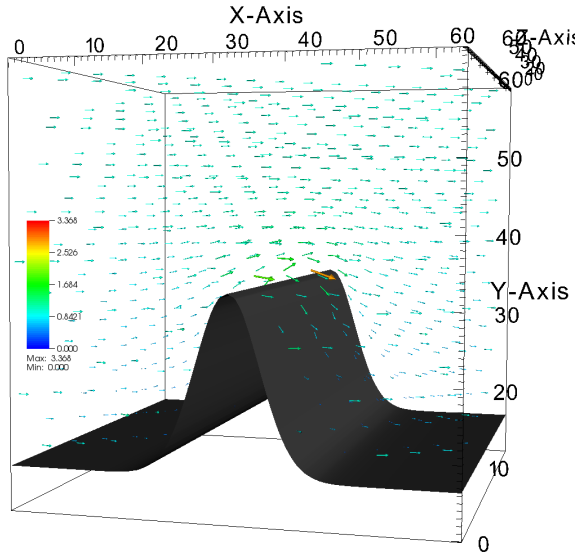


FIG. 3.9. Vector plot of the final vector field $(u, v, w)_{exp}$ over exponential hill H_{exp}.

The convergence of the algorithm for this test case is graphed in Figure 3.10.

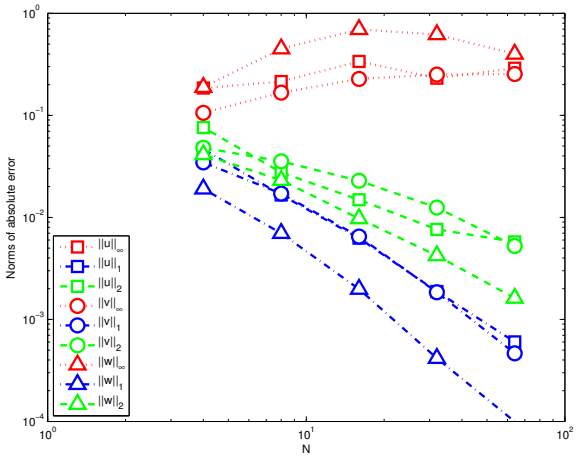


FIG. 3.10. Volume-weighted norms of the cell-centered absolute error e_N over exponential hill H_{exp}.

Once again, nothing seems to converge in the $L_\infty-$norm. The convergence rate in the L_2-norm is even lower than in the sinusoidal hills case. This is probably caused by the fact that the geometry is more prominent than the small sinusoidal hills and takes up a bigger region of the domain.

e_N	L_∞	L_1	L_2
u	–	1.57	0.96
v	–	1.59	0.82
w	–	1.93	1.19

TAB. 3.4. Convergence rate of $(u, v, w)_{exp}$ in different norms.

CONCLUSIONS

The goal for this project was to implement a new approach for modelling wildfire spread. The model is based on mass-consistent models that were developed over the last 30 years for wind modelling in atmospheric sciences. We have shown that Sasaki's variational technique was very similar to the projection method. This last approach was used for extracting a divergence-free velocity field from any initial vector field.

The model was formulated in such a way that it can be initialized with a high altitude wind which can be interpolated over the whole domain down to the surface. The geometric features of the topography which accounts for most of the final velocity profile are now treated efficiently with the Embedded Boundary Method. This technique based on the Finite Volume Method was used for modelling wind in two and three dimensions.

Error analysis has shown that the EBM algorithm converges for all cases except in the max norm. This norm is more sensitive to the error occurring in cut cells than the other cells of the computational domain. In 2D, the half-cylinder case test has shown that the resulting mass-consistent wind velocity is very similar to the potential flow even if our numerical solution does not converge to the exact solution.

Further research will implement the inclusion of the fire feedback as a dilation source term in the computation of the wind, more precisely when solving the Poisson equation. This additional step will lead to a mass-consistent and fire-induced flow that could be more representative of the actual wind in a wildfire region, especially for high intensity fires. New numerical methods will have to be developed in order to address the numerical issues related to the inclusion of the

source term in the projection operators. It will also be interesting to determine the regimes where the coupled fire-atmosphere mass-consistent model should be used prior to the use of the conservation of momentum equation.

BIBLIOGRAPHIE

[1] Almgren A.S., Bell J. B. and Szymczak W. G., A Numerical Method for the Incompressible Navier-Stokes Equations Based on an Approximate Projection, *SIAM J. Sci. Comp.*, **17**, 358-369, 1996.

[2] Barnard J. C. et al., Improving the performance of mass consistent numerical models using optimization techniques, *J. Climate Appl. Meteor.*, **26**, 675-686, 1987.

[3] Bell J. B., Colella P. and Glaz H. M., A second-order projection method of the incompressible Navier-Stokes equations, *J. Comp. Phys.*, **85**, 257-283, 1989.

[4] Chorin A. J. and Marsden J. E., *A Mathematical Introduction to Fluid Mechanics*, Springer-Verlag, 1990.

[5] Colella P. and Pao K., A Projection Method for Low Speed Flows, *J. Comp. Phys*, **149**, 1999.

[6] Colella P. et al., Embedded boundary algorithms and software for partial differential equations, *J. Phys. : Conf. Ser.*, **125**, 2008.

[7] Colella P. et al., *Chombo Software Package for AMR Applications Design Document*, Applied Numerical Algorithms Group.

[8] Colella P. et al., *EBChombo Software Package for Cartesian Grid, Embedded Boundary Applications*, Applied Numerical Algorithms Group.

[9] Drikakis D. and Rider W., *High-Resolution Methods for Incompressible and Low-Speed Flows*, Springer, 2005.

[10] Forthofer J. M., Modeling Wind in Complex Terrain for Use in Fire Spread Prediction, *Master's thesis*, Colorado State University, 2007.

[11] Georgieva E., et al., $\mathcal{WINDS}$ *: User's guide*, 2002.

[12] Harlow F. H. and Welch J. E., Numerical Calculation of time-dependent viscous incompressible flow of fluid with a free surface, *Physics of Fluids*, **8**, 2182-2189, 1965.

[13] Ishikawa H., Mass-consistent wind model as a meteorological preprocessor for tracer transport models, *J. Appl. Meteor.*, **33**, 733-743, 1994.

[14] Johansen H. and Colella P., A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, *J. Comp. Phys.*, **147**, 60-85, 1998.

[15] Lai M. F., *A Projection Method for Reacting Flow in the Zero Mach Number Limit*, Ph.D. thesis, UC Berkeley, 1994.

[16] Ligocki T. J. et al., Embedded boundary grid generation using the divergence theorem, implicit functions, and constructive solid geometry, *J. Phys. : Conf. Ser.*, **125**, 2008.

[17] Martin D. F. and Cartwright K. L., Solving Poisson's Equation using Adaptive Mesh Refinement, *U.C. Berkeley Electronics Research Laboratory*, **UCB/ERL M96/66**, 1996.

[18] Minion M. L., A Projection Method for Locally Refined Grids, *J. Comp. Phys.*, **127**, 158-178, 1996.

[19] Núñez M. A. et al., A study of hydrodynamic mass-consistent models, *J. Comp. Meth. Sci. Eng.*, **6**, 365-385, 2006.

[20] Ratto R. et al., Mass-Consistent models for wind fields over complex terrain : The state of the art, *Environ. Software*, **9**, 247-268, 1994.

[21] Ross D. G. et al., Diagnostic wind field modeling for complex terrain : model development and testing, *J. Appl. Meteor.*, **27**, 785-796, 1988.

[22] Sasaki Y., Some basic formalism in numerical variational analysis, *Monthly Weather Review*, **98**, 875-883, 1970.

[23] Schwartz P. et al., A Cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions, *J. Comp. Phys.*, **211**, 531-550, 2006.

[24] Sherman C. A., A mass-consistent model for wind fields over complex terrain, *J. Appl. Meteor.*, **17**, 312-319, 1978.

[25] Wang Y. et al., Application of a Multigrid Method to a Mass-Consistent Diagnostic Wind Model, *J. Appl. Meteo.*, **44**, 1078-1089, 2005.

[26] http ://firemodels.fire.org/

Annexe A

PROJECTIONS

The projection method which has been introduced in chapter one was used throughout the thesis for solving the Poisson Equation. We have chosen a discretization of the operators on a MAC grid for our algorithm. We show here two other ways for discretizing the projection operators, the divergence operator $\nabla\cdot$, the gradient operator ∇ and the projection operator $\mathcal{P}$.

Cell-Centered Projection

The cell-centered projection uses a grid where the components of the vector field $\mathbf{v}$ and the scalar field φ are collocated, i.e. defined at the center of the control cells. Let D^o and G^o be the discrete divergence and discrete gradient operators over the cell-centered grid. The natural way to define those operators is using the centered difference approximations :

$$D^o(\mathbf{v})_{i,j} = \frac{u_{i+1,j} - u_{i-1,j}}{2h_x} + \frac{w_{i,j+1} - w_{i,j-1}}{2h_y} \tag{A.0.11}$$

and

$$G^o(\varphi)_{i,j} = \left(\frac{\varphi_{i+1,j} - \varphi_{i-1,j}}{2h_x}, \frac{\varphi_{i,j+1} - \varphi_{i,j-1}}{2h_y} \right). \tag{A.0.12}$$

The projection operator is then defined as

$$\mathbf{P}^o = \mathbf{I} - G^o \left(D^o G^o\right)^{-1} D^o \tag{A.0.13}$$

and $D^o(\mathbf{P}^o(\mathbf{v})) = 0$.

Unfortunately, solving the Poisson equation is more complicated since the Laplacian $D^o(G^o(\varphi))_{i,j}$ is different from the standard Laplacian. The stencil is like

the standard 5 point Laplacian but it is expanded as it can be seen in Figure A.1. This decouples a two-dimensional grid into four distinct subgrids which makes

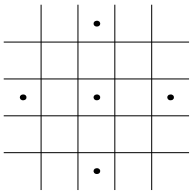


FIG. A.1. Expanded Laplacian $D^o(G^o(\varphi))_{i,j}$

the linear algebra more complicated to solve the Poisson equation. Oscillations in the solution might also occur but it can be corrected with filters. Note also that the subgrids can be recoupled with the boundary conditions.

MAC PROJECTION

The Marker-and-Cell (MAC) projection introduced by Harlow and Welch [**12**] uses a staggered grid, where the components of the vector field are defined on the edges of the control cells and the scalar field is defined at the center as shown in Figure A.2.

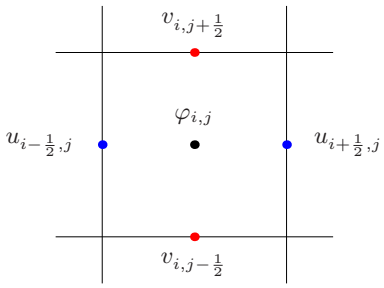


FIG. A.2. Staggered grid with the cell-centered scalar field φ and the edge-centered components of the velocity field (u, v).

Let D^{MAC} and G^{MAC} be the discrete divergence and discrete gradient operators over the staggered grid. The divergence is defined using the divergence theorem on the control cell and the gradient by centered differences over the edges :

$$D^{MAC}(\mathbf{v})_{i,j} = \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h_x} + \frac{w_{i,j+\frac{1}{2}} - w_{i,j-\frac{1}{2}}}{h_y} \tag{A.0.14}$$

and

$$G^{MAC}(\varphi)_{i+\frac{1}{2},j} = \frac{\varphi_{i+1,j} - \varphi_{i,j}}{h_x} \tag{A.0.15}$$

$$G^{MAC}(\varphi)_{i,j+\frac{1}{2}} = \frac{\varphi_{i,j+1} - \varphi_{i,j}}{h_y} \tag{A.0.16}$$

The projection operator is then defined as

$$\mathbf{P}^{MAC} = \mathbf{I} - G^{MAC}\left(D^{MAC}G^{MAC}\right)^{-1} D^{MAC} \tag{A.0.17}$$

and $D^{MAC}(\mathbf{P}^{MAC}(\mathbf{v})) = 0$.

For this projection, the Laplacian defined as $D^{MAC}(G^{MAC}(\varphi))_{i,j}$ is the standard 5 point stencil Laplacian as shown in Figure A.3.

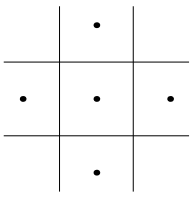


FIG. A.3. Standard 5 point stencil Laplacian

Despite the fact that the components of the vector field are not specified at the same place on the grid, the MAC projection has the advantage that the slip boundary condition can be set explicitly at walls for rectangular domains.

CELL-CENTERED APPROXIMATE PROJECTION

The two previous projections are called exact projections since the divergence of the divergence-free vector field is exactly zero. Some properties of the MAC and cell-centered projections would be desirable in the same projection. For instance, we might want the components of the vector field to be collocated. Also, it would be easier to use the standard Laplacian rather than the expanded one. These properties are included in the cell-centered approximate projection. The vector and scalar fields are defined at the center of the control cells, so the operators G^o and D^o can be used. To avoid difficulties with the expanded Laplacian, we use the standard Laplacian defined by G^{MAC} and D^{MAC} when solving the Poisson

equation. Hence, we are solving

$$D^{MAC}(G^{MAC}(\varphi))_{i,j} = D^o(\mathbf{v})_{i,j}. \tag{A.0.18}$$

The operator $\tilde{\mathbf{P}}$ is then defined as

$$\tilde{\mathbf{P}} = \mathbf{I} - G^o \left(L\right)^{-1} D^o \tag{A.0.19}$$

where $L \neq D^o G^o$. Hence, the projection $\tilde{\mathbf{P}}$ is non-idempotent $\tilde{\mathbf{P}} \neq \tilde{\mathbf{P}}^2$ but $\tilde{\mathbf{P}} \approx \tilde{\mathbf{P}}^2$. Also, $D^o(\tilde{\mathbf{P}}(\mathbf{v})) = \mathcal{O}(h^2)$, which means that $\mathbf{v}_d$ is not exactly divergence-free and $\tilde{\mathbf{P}}$ is then called an approximate projection operator.