2

Mass–Consistent Wind Field Models: Numerical Techniques by L2–Projection Methods

L. Héctor Juárez[1], María Luisa Sandoval[1], Jorge López[2]
and Rafael Reséndiz[1]
[1]*Departamento de Matemáticas, Universidad Autónoma Metropolitana Iztapalapa, México City*
[2]*División de Ciencias Básicas, Universidad Juárez Autónoma de Tabasco México*

1. Introduction

For several meteorological problems and a large number of applications, the knowledge of the 3–D wind field over a region is required. Examples include prediction of the transport, diffusion and dispersion of air pollutants in the atmosphere (Finardi et al., 2010; Sherman, 1978), realization of wind maps for the design of different urban and general projects (Castino et al., 2003), and the effect of wind on structures and fire spreading (Potter & Butler, 2009), among others. Moreover, meteorological wind fields are also required inputs for air quality models. In practice, usually limited horizontal wind field measurements are available, and therefore the calculation of the vertical motion must be predicted or calculated. Several methods and strategies, with various levels of complexity, have been proposed to address this problem. They can be included into two general model types: prognostic models and diagnostic models. Prognostic models are complex time–dependent hydrodynamic models governing air flow, including thermal effects, density variation and turbulent interaction. While these models are "realistic", they are expensive to operate, need extensive computer facilities, and require specialized training for their operation. On the other hand, diagnostic wind models do not require the integration of the non–linear hydrodynamic equations. Instead, available interpolated data is used to generate wind fields, which satisfy some physical or dynamical constraints. For instance, to assure mass conservation, a simplified steady–state version of the continuity equation is imposed, and the resulting model is then called a mass–consistent model. A review of these models is available in Ratto et al. (1994) and Ratto (1996).

We focus in a variational mass–consistent model which is based in the original formulation by Sasaki (Sasaki, 1958). This approach has been used for a variety of meteorological problems (Castino et al., 2003; Pennel, 1983; Sherman, 1978; Wang et al., 2005). Mass–consistent models are attractive because of their simplicity, and because they are easy and economical to operate. In some applications, these models outperform the more sophisticated and expensive dynamical models (Ratto et al., 1994). However, mass–consistent models have some disadvantages, because they are based on incomplete or idealized models and have difficulty

representing flows accurately in data–sparse regions as mountains or oceans. Despite these limitations, mass–consistent models are a valuable tool for air quality applications and consequently several developments have taken place over last decades (Ferragut et al., 2010; Ratto, 1996; Ratto et al., 1994; Ross et al., 1988; Wang et al., 2005). Most of the results presented in this chapter has been published in the last few years (Flores et al., 2010; Núñez et al., 2007; 2006), but we also include some additional ideas and recent results.

The variational method proposed by Sasaki uses the continuity equation $\nabla \cdot \mathbf{u} = 0$, where $\mathbf{u}$ is the wind velocity vector field on a given domain Ω. The method is based on the minimization of the functional L defined by

$$L(\mathbf{u}, \lambda) = \frac{1}{2} \int_{\Omega} \left\{ S \left(\mathbf{u} - \mathbf{u}^{\mathrm{I}} \right) \cdot \left(\mathbf{u} - \mathbf{u}^{\mathrm{I}} \right) + \lambda \left[\nabla \cdot \mathbf{u} \right] \right\} \, dV \,, \tag{1}$$

where $\mathbf{u}^{\mathrm{I}}$ is an initial observed wind field, λ is a Lagrange multiplier and S is a diagonal matrix with weighting parameters $\alpha_i > 0$, $i = 1, 2, 3$, called Gaussian precision moduli, related to the scales of the respective components of the velocity field. The vertical component of the initial wind field is taken as zero because meteorological stations usually do not measure this component. The Euler–Lagrange equations of (1) are:

$$\mathbf{u} = \mathbf{u}^{\mathrm{I}} + S^{-1} \nabla \lambda, \tag{2}$$

Usually $\mathbf{u}$ is obtained from (2), after λ is computed. Since $\nabla \cdot \mathbf{u} = 0$, then from (2) we obtain the elliptic equation $-\nabla \cdot \left(S^{-1} \nabla \lambda \right) = \nabla \cdot \mathbf{u}^{\mathrm{I}}$, from which λ is obtained. To complement (close) this equation, two types of boundary conditions are commonly used: homogeneous Dirichlet boundary conditions, $\lambda = 0$, for open or "flow through" boundaries (like truncated boundaries), and Neumann boundary conditions, $\partial \lambda / \partial \mathbf{n} = 0$, for closed or "no flow through" boundaries (like the surface terrain or topography). Many authors have been used and recommend these boundary conditions (Kitada et al., 1986; 1983; Ratto et al., 1994; Sherman, 1978). However they are physically and mathematically inconsistent as we will show in this work. Even though, there have been several sophisticated developments in the numerical simulations of this model as, for instance, the application of multigrid methods (Wang et al., 2005), and the application of genetic algorithms to estimate parameters (Montero et al., 2005), it seems that the analysis of boundary conditions has not attracted the attention of the community in meteorology.

In this work we study how boundary conditions affect solutions of the elliptic equation for λ. We show that the application of incorrect boundary conditions may degrade the solutions several orders of magnitude, and we propose some strategies to overcome this problem. In particular, we introduce a new approach based on the saddle–point formulation of the constrained least squares formulation of the problem, which allows the introduction of successful techniques from computational fluid dynamics. This new approach does not require boundary conditions for the multiplier. It produces much better results, and it also helps us to establish more consistent boundary conditions on truncated nonphysical boundaries. We also explore other boundary conditions for the multiplier better suited for artificial truncated boundaries. Furthermore, we present some preliminary numerical results using a meshfree method based on a radial basis function collocation method.

2. Mathematical formulation of the problem

Let Ω be an open, simply connected and bounded region in $\mathbb{R}^d$ ($d = 2$ or 3) with Lipschitz boundary $\partial\Omega = \Gamma_N \cup \Gamma_D$, where Γ_N is the part of the boundary associated to the surface terrain (topography), and Γ_D is the rest of the boundary (artificial vertical boundaries and top boundary), as shown in Figure 1. Given an initial vector field $\mathbf{u}^{\mathrm{I}}$ in Ω (which can be obtained

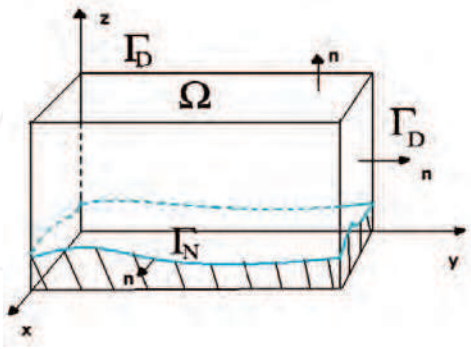


Fig. 1. Bounded region Ω.

by interpolating atmospheric data, or by other means), our goal is to generate a solenoidal field $\mathbf{u}$ –called adjusted field– as close to $\mathbf{u}^{\mathrm{I}}$ as possible in a sense that will be clarified below, such that $\mathbf{u} \cdot \mathbf{n} = 0$ on Γ_N.
We define the following vector function spaces: $\mathbf{L}_2(\Omega) = L_2(\Omega)^d$ and $\mathbf{H}(div;\Omega) = \{\, \mathbf{v} \in \mathbf{L}_2(\Omega) : \nabla \cdot \mathbf{v} \in L_2(\Omega) \,\}$. Then, the adjusted wind field $\mathbf{u}$ must belong to the normed closed space

$$\mathbf{V} = \{\, \mathbf{v} \in \mathbf{H}(div;\Omega) \;:\; \nabla \cdot \mathbf{v} = 0 \quad \text{and} \quad \mathbf{v} \cdot \mathbf{n} = 0 \quad \text{on} \quad \Gamma_N \,\}, \tag{3}$$

with the norm $\|\cdot\|_{S,\Omega}$ associated to the inner product $\langle \mathbf{u}, \mathbf{v} \rangle_S = \int_\Omega S\mathbf{u} \cdot \mathbf{v}\, d\mathbf{x}$, where $\mathbf{v} \cdot \mathbf{w} = \sum_1^d v_i\, w_i$ is the usual scalar product in $\mathbb{R}^d$. We can now formulate the problem as a least squares projection problem. For this purpose, we define a convex quadratic functional $J : \mathbf{V} \to \mathbb{R}$ as

$$J(\mathbf{v}) = \frac{1}{2} \parallel \mathbf{v} - \mathbf{u}^{\mathrm{I}} \parallel^2_{S,\Omega} = \frac{1}{2} \int_\Omega S(\mathbf{v} - \mathbf{u}^{\mathrm{I}}) \cdot (\mathbf{v} - \mathbf{u}^{\mathrm{I}})\, d\mathbf{x}. \tag{4}$$

Then, our problem can be stated as follows:

$$\text{Given} \quad \mathbf{u}^{\mathrm{I}} \in \mathbf{L}_2(\Omega), \quad \text{find} \quad \mathbf{u} \in \mathbf{V} \quad \text{such that} \quad J(\mathbf{u}) \leq J(\mathbf{v}), \quad \forall\, \mathbf{v} \in \mathbf{V}. \tag{5}$$

Due to the properties of this functional, $\mathbf{u} \in \mathbf{V}$ is a minimizer of J if and only if it is a stationary point of J:

$$\frac{\partial}{\partial \epsilon} J(\mathbf{u} + \epsilon\, \mathbf{v})|_{\epsilon=0} = \int_\Omega S(\mathbf{u} - \mathbf{u}^{\mathrm{I}}) \cdot \mathbf{v}\, d\mathbf{x} = 0, \quad \forall\, \mathbf{v} \in \mathbf{V}. \tag{6}$$

The Lax–Milgram theorem guaranties that this equation has a unique solution.

3. The traditional approach. Advantages and difficulties

3.1 Derivation of the elliptic problem

The first approach is based on a Helmholtz–type decomposition of the Hilbert vector space $\mathbf{L}_2(\Omega)$, and it reduces to the traditional approach used by meteorologists.

Proposition 1 *The orthogonal complement in* $\mathbf{L}_2(\Omega)$ *of the closed subspace* $\mathbf{V}$ *is*

$$\mathbf{V}^{\perp} = \{\ \nabla q\ :\ q \in H^1(\Omega),\quad q = 0\quad \text{on}\quad \Gamma_D\ \}.$$

An argument very similar to that given by Girault and Raviart (Girault & Raviart, 1986), shows that this decomposition is valid (details are given in (Núñez et al., 2007)). Therefore we get from (6) that $S\left(\mathbf{u} - \mathbf{u}^{\mathrm{I}}\right) = \nabla\lambda$, with λ in

$$H^1_D(\Omega) \equiv \{\ q \in H^1(\Omega)\ :\ q = 0 \text{ on } \Gamma_D\ \}. \tag{7}$$

With the above properties, we obtain a saddle–point problem for $\mathbf{u}$ and λ (left), as well as the correspondent elliptic problem for λ (right):

$$S\mathbf{u} - \nabla\lambda = S\mathbf{u}^{\mathrm{I}},\quad \text{and}\quad \nabla\cdot\mathbf{u} = 0\ \text{ in } \Omega, \qquad -\nabla\cdot\left(S^{-1}\nabla\lambda\right) = \nabla\cdot\mathbf{u}^{\mathrm{I}}\ \text{ in }\ \Omega, \tag{8}$$

$$\lambda = 0\quad \text{on } \Gamma_D, \qquad \lambda = 0\ \text{ on }\ \Gamma_D, \tag{9}$$

$$\mathbf{u}\cdot\mathbf{n} = 0\quad \text{on } \Gamma_N. \qquad -S^{-1}\nabla\lambda\cdot\mathbf{n} = \mathbf{u}^{\mathrm{I}}\cdot\mathbf{n}\ \text{ on }\ \Gamma_N. \tag{10}$$

To obtain the elliptic problem, we eliminate $\mathbf{u}$ from the saddle–point problem using that $\mathbf{u}$ belongs to $\mathbf{V}$. Once λ is calculated from (8)–(10), the adjusted field is recovered from (2).

Equation (8) has traditionally been used by meteorologists. However, this equation is generally introduced from a discussion in which it is not clear how to establish the proper boundary conditions for λ. The crucial argument in our study is the decomposition of $\mathbf{L}_2(\Omega)$ in orthogonal subspaces $\mathbf{V}$ and $\mathbf{V}^{\perp}$, from which the boundary conditions for λ arises in a natural way, from the mathematical point of view. We would like to mention that the boundary condition (10) has already been used in recent research (Ferragut et al., 2010).

3.2 Finite element solution of the elliptic problem

The variational formulation of the elliptic problem (8)–(10) is

$$\int_\Omega S^{-1}\nabla\lambda\cdot\nabla q\, d\mathbf{x} = -\int_\Omega \mathbf{u}^{\mathrm{I}}\cdot\nabla q\, d\mathbf{x},\quad \forall\, q \in H^1_D(\Omega). \tag{11}$$

Here, we consider the two–dimensional case. Let $\mathcal{T}_h$ be a finite element triangulation of $\overline{\Omega} \subset \mathbb{R}^2$ (Ciarlet, 2002), where h is taken as the space discretization step. Let's denote by P_1 the space of polynomials of degree less or equal than 1. Then, $\mathbf{L}_2(\Omega)$ and $H^1_D(\Omega)$ are approximated by the finite dimensional spaces

$$\mathbf{L}_h = \left\{\ \mathbf{v}_h \in \mathcal{C}^0(\bar{\Omega})^2\ :\ \mathbf{v}_h|_T \in P_1\times P_1,\, \forall\, T \in \mathcal{T}_h\ \right\}, \tag{12}$$

$$H_h = \left\{\ q \in \mathcal{C}^0(\bar{\Omega})\ :\ q|_T \in P_1,\, \forall\, T \in \mathcal{T}_h,\, q = 0 \text{ on } \Gamma_D\ \right\}, \tag{13}$$

respectively. Thus, the finite element algorithm is: Given $\mathbf{u}_h^{\mathrm{I}} \in \mathbf{L}_h$, find $\lambda_h \in H_h$ such that

$$\int_\Omega S^{-1} \nabla \lambda_h \cdot \nabla q \, d\mathbf{x} = -\int_\Omega \mathbf{u}_h^{\mathrm{I}} \cdot \nabla q \, d\mathbf{x}, \quad \forall\, q \in H_h, \tag{14}$$

where $\mathbf{u}_h^{\mathrm{I}} \in \mathbf{L}_h$ is the interpolant of the given initial velocity field $\mathbf{u}^{\mathrm{I}}$. We obtain λ_h after solving the resulting system of linear equations, and the numerical approximation $\mathbf{u}_h$ of $\mathbf{u}$ is computed by the weak version of (2) as follows: Find $\mathbf{u}_h \in \mathbf{L}_h$ with $\mathbf{u}_h \cdot \mathbf{n} = 0$ on Γ_N such that

$$\int_\Omega (S\mathbf{u}_h) \cdot \mathbf{v} \, d\mathbf{x} = \int_\Omega (S\mathbf{u}_h^{\mathrm{I}}) \cdot \mathbf{v} \, d\mathbf{x} - \int_\Omega \lambda_h \nabla \cdot \mathbf{v} \, d\mathbf{x}, \quad \forall\, \mathbf{v} \in \mathbf{L}_h, \quad \mathbf{v} \cdot \mathbf{n} = 0 \quad \text{on} \quad \Gamma_N. \tag{15}$$

From now on, we identify the algorithm (14)–(15) as the *E1–algorithm*.

Example 1. We consider the solenoidal vector field $\mathbf{u}(x, z) = (x, -z)$ defined in $\Omega = (1, 2) \times (0, 1)$, so that $\mathbf{u} \in \mathbf{V}$. Assuming that we have $\mathbf{u}^{\mathrm{I}}(x, z) = (x, 0)$ as an initial horizontal wind field, we want to apply the *E1–algorithm* to see how much we can recover of the vertical component of $\mathbf{u}$. For this numerical calculation, Ω is divided into a 80×80 triangular mesh, and we choose the following values for the Gaussian Precision moduli: $\alpha_1 = 1$ and $\alpha_3 = 0.001$. Figure 2 shows the exact field in red and the computed adjusted field in blue. Both fields agree fairly well almost everywhere, except on the vertical artificial boundaries $x = 1$ and $x = 2$.

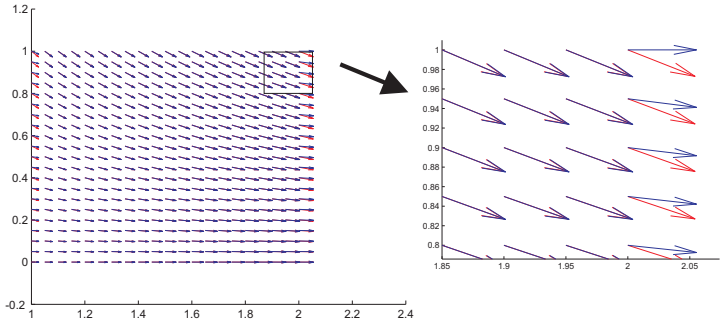


Fig. 2. Exact field $\mathbf{u} = (x, -z)$ in red, adjusted field obtained by the *E1–algorithm* in blue.

The relative error and the mean divergence of the computed solution are defined as

$$e_r = \frac{||\mathbf{u} - \mathbf{u}_h||_2}{||\mathbf{u}||_2}, \quad \text{and} \quad mdiv = \underset{\mathbf{x}_i}{\text{mean}}\{\, \nabla \cdot \mathbf{u}_h(\mathbf{x}_i) \mid \mathbf{x}_i \text{ is a interior vertex}\}, \tag{16}$$

respectively. The point–wise divergence is computed in a weak sense, as follows

$$\nabla \cdot \mathbf{u}_h(\mathbf{x}_i) = -\int_\Omega \mathbf{u}_h \cdot \nabla \phi_i \, d\mathbf{x}\,, \tag{17}$$

where ϕ_i is the piece–wise linear base function associated to vertex node $\mathbf{x}_i$. For the present example, we obtain $e_r = 1.9 \times 10^{-2}$ and $mdiv = 4.1 \times 10^{-2}$.

The values for the Gaussian precision moduli were chosen based on numerical performance. Table 1 shows the behavior of e_r and $mdiv$ for different values of α_3 when α_1 is kept constant

and equal to one. Clearly the best results were obtained with $\alpha_3 = 0.001$. We will explain this behavior later on. But, for the moment, we want to emphasize that this algorithm produces satisfactory results almost everywhere, except on the boundary Γ_D, where homogeneous Dirchlet boundary conditions were imposed.

$\boldsymbol{\alpha_3}$	0.001	0.01	0.1	1	100	1000
e_r	1.9×10^{-2}	9.6×10^{-2}	1.4×10^{-1}	5.2×10^{-1}	6.4×10^{-1}	9.8×10^{-1}
mdiv	4.1×10^{-2}	-6.1×10^{-2}	2.9×10^{-1}	5.4×10^{-1}	7.8×10^{-1}	9.8×10^{-1}

Table 1. Numerical performance of *E1–algorithm* for different values of α_3.

We can say that the main advantage of this traditional way to solve the problem is its simplicity, since it only involves the solutions of an elliptic partial differential equation (PDE). On the other hand, one of its major drawbacks is that inconsistent or incorrect boundary conditions, on truncated artificial boundaries, degrade the accuracy of the solution. In the rest of the chapter, we introduce some alternatives to overcome these problems.

4. A saddle–point formulation and the conjugate gradient algorithm

4.1 Derivation of the formulation

The second approach to solve the problem (5), or equivalently problem (6), is based on the usual methodology to solve constrained optimization problems. That is, we introduce the space of vector functions

$$\mathbf{V}_N = \{\, \mathbf{v} \in \mathbf{H}(div;\Omega) \;:\; \mathbf{v}\cdot\mathbf{n} = 0 \;\text{ on }\; \Gamma_N \,\}, \tag{18}$$

together with the Lagrangian L defined on $\mathbf{V}_N \times L_2(\Omega)$ as

$$L(\mathbf{v},q) \equiv J(\mathbf{v}) + \langle q, \nabla\cdot\mathbf{v}\rangle = \frac{1}{2}\int_\Omega S(\mathbf{v}-\mathbf{u}^{\mathrm{I}})\cdot(\mathbf{v}-\mathbf{u}^{\mathrm{I}})\,d\mathbf{x} + \int_\Omega q\nabla\cdot\mathbf{v}\,d\mathbf{x}.$$

A stationary point $(\mathbf{u},\lambda)$ of L solves the following saddle–point problem

$$\int_\Omega S\mathbf{u}\cdot\mathbf{v}\,d\mathbf{x} + \int_\Omega \lambda\nabla\cdot\mathbf{v}\,d\mathbf{x} = \int_\Omega S\mathbf{u}^{\mathrm{I}}\cdot\mathbf{v}\,d\mathbf{x}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N, \tag{19}$$

$$\int_\Omega q\nabla\cdot\mathbf{u}\,d\mathbf{x} = 0, \quad \forall\, q\in L_2(\Omega), \tag{20}$$

where λ **need not satisfy boundary conditions**. The solution $\mathbf{u}$ is the minimizer of J, and now it is obtained from the enlarged space $\mathbf{V}_N$ where free divergence is not required. Instead, the condition $\nabla\cdot\mathbf{u} = 0$ is relaxed by the introduction of the Lagrange multiplier λ so that $\mathbf{u}$ must satisfy the weaker condition (20). To solve (19)–(20) we introduce a method which has shown to be very effective for solving Stokes problems in computational fluid dynamics (Glowinski, 2003). The idea is as follows: assuming that $(\mathbf{u},\lambda)$ is a solution of the problem (19)–(20), the vector field $\mathbf{u}$ is decomposed as $\mathbf{u} = \mathbf{u}^{\mathrm{I}} + \mathbf{u}_\lambda$, where $\mathbf{u}^{\mathrm{I}}$ is the given initial vector field, and $\mathbf{u}_\lambda \in \mathbf{V}_N$ solves

$$\int_\Omega S\mathbf{u}_\lambda\cdot\mathbf{v}\,d\mathbf{x} = -\int_\Omega \lambda\nabla\cdot\mathbf{v}\,d\mathbf{x}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N. \tag{21}$$

Furthermore, $\mathbf{u}_\lambda$ must satisfy (20) which has the following equivalent strong version

$$-\nabla\cdot\mathbf{u}_\lambda = \nabla\cdot\mathbf{u}^{\mathrm{I}}. \tag{22}$$

A key point is that problem (21)–(22) can be formulated as a functional equation. For this we introduce the linear operator A from $L_2(\Omega)$ into $L_2(\Omega)$ defined by

$$A\,q = -\nabla\cdot\mathbf{u}_q\,, \tag{23}$$

where $\mathbf{u}_q \in \mathbf{V}_N$ is the solution of

$$\int_\Omega S\mathbf{u}_q\cdot\mathbf{v}\,d\mathbf{x} = -\int_\Omega q\nabla\cdot\mathbf{v}\,d\mathbf{x}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N. \tag{24}$$

With this definition, it is clear, from (21)–(22), that the multiplier λ satisfies the functional equation

$$A\lambda = \nabla\cdot\mathbf{u}^{\mathrm{I}}. \tag{25}$$

4.2 Conjugate gradient algorithm

Operator A is selfadjoint, and strongly elliptic, since from (23) and (24) we have

$$\int_\Omega q'\,A\,q\;d\mathbf{x} = -\int_\Omega q'\,\nabla\cdot\mathbf{u}_q = \int_\Omega S\,\mathbf{u}_{q'}\cdot\mathbf{u}_q\;d\mathbf{x} \qquad \forall\,q,\;q'\in L_2(\Omega),$$
$$\int_\Omega q\,A\,q\;d\mathbf{x} = \int_\Omega S\,\mathbf{u}_q\cdot\mathbf{u}_q > c\,\|\mathbf{u}_q\|^2_{L_2(\Omega)} \qquad \forall\,q\neq 0 \quad (0<c<\min\{\alpha_i\})$$

Therefore, the following iterative conjugate gradient algorithm may be used to solve the infinite dimensional problem (25):

1. Given $\lambda^0\in L_2(\Omega)$, solve for $\mathbf{u}^0\in\mathbf{V}_N$

$$\int_\Omega S\,\mathbf{u}^0\cdot\mathbf{v}\,d\mathbf{x} = \int_\Omega S\,\mathbf{u}^I\cdot\mathbf{v}\,d\mathbf{x} - \int_\Omega \lambda^0\nabla\cdot\mathbf{v}\,d\mathbf{x}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N.$$

   Set $g^0 = -\nabla\cdot\mathbf{u}^0$ and $d^0 = -g^0$.

2. For $k\geq 0$, assuming we know λ^k, g^k, d^k, $\mathbf{u}^k$, find λ^{k+1}, g^{k+1}, d^{k+1}, $\mathbf{u}^{k+1}$, doing the following: Solve for $\underline{\mathbf{u}}^k\in\mathbf{V}_N$

$$\int_\Omega S\,\underline{\mathbf{u}}^k\cdot\mathbf{v}\,d\mathbf{x} = -\int_\Omega d^k\nabla\cdot\mathbf{v}\,d\mathbf{x}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N.$$

   Set $w^k = -\nabla\cdot\underline{\mathbf{u}}^k$ and compute $\alpha_k = \dfrac{\langle g^k,\,g^k\rangle}{\langle d^k,\,w^k\rangle}$.

   Compute $\lambda^{k+1} = \lambda^k + \alpha_k\,d^k$, $\mathbf{u}^{k+1} = \mathbf{u}^k + \alpha_k\,\underline{\mathbf{u}}^k$, $g^{k+1} = g^k + \alpha_k\,w^k$.

3. If $\langle g^k, g^k\rangle \leq \varepsilon\langle g^0, g^0\rangle$, take $\lambda = \lambda^{k+1}$ and $\mathbf{u} = \mathbf{u}^{k+1}$ and stop. Otherwise, compute

$$d^{k+1} = -g^{k+1} + \beta_k\,d^k \quad \text{where} \quad \beta_k = \frac{\langle g^{k+1},\,g^{k+1}\rangle}{\langle g^k,\,g^k\rangle}.$$

Do $k = k + 1$ and return to 2.

Above, $\langle \cdot, \cdot \rangle$ indicates the usual scalar product in $L_2(\Omega)$. Observe that the adjusted field $\mathbf{u}$ is also computed as an intermediate step in the algorithm. In this algorithm, no boundary conditions are imposed on λ, contrary to what it was done in the first approach. This fact has a very important effect in the numerical calculation.

4.3 A mixed finite element method

To approximate the functions in $\mathbf{V}_N$ and $L_2(\Omega)$, considered in the previous algorithm, we use the *Bercovier–Pironneau* finite element approximation (Bercovier & Pironneau, 1979). Functions in $L_2(\Omega)$ are approximated by continuous piecewise linear polynomials over a triangulation $\mathcal{T}_h$ of Ω, while the elements in $\mathbf{V}_N$ are also approximated by linear polynomials but now over a twice finer triangulation $\mathcal{T}_{h/2}$ of Ω. The fine triangulation $\mathcal{T}_{h/2}$ is obtained from a regular subdivision of each triangle $T \in \mathcal{T}_h$. Then, the functional spaces $\mathbf{V}_N$ and $L_2(\Omega)$ will be approximated, respectively, by the finite dimensional spaces

$$\mathbf{V}_{Nh} = \left\{ \mathbf{v}_h \in C^0(\bar{\Omega})^2 \ : \ \mathbf{v}_h|_T \in P_1 \times P_1, \quad \forall\, T \in \mathcal{T}_{h/2}, \quad \mathbf{v}_h \cdot \mathbf{n} = 0 \text{ on } \Gamma_N \right\},$$

$$L_h = \left\{ q_h \in C^0(\bar{\Omega}) \ : \ q_h|_T \in P_1, \quad \forall\, T \in \mathcal{T}_h \right\},$$

We apply this mixed method, particularly to solve the integral equations in steps 1 and 2, as well as for the calculation of the weak divergence to obtain g^0 in step 1 and w^k in step 2. Those calculations require this mixed method, or any other stable finite element pair, to avoid instabilities in the numerical solution. Actually, the main cost of this algorithm is the solution at each iteration of the integral equation to get $\underline{\mathbf{u}}^k$ and the calculation of w^k. However, if the trapezoidal rule is applied to approximate the left hand side of the integral equations, we obtain a system of algebraic equations with diagonal matrix, and the cost to solve them is just a vector multiplication. We call this new algorithm the *CG–algorithm*.

Example 2. We consider again the initial horizontal field $\mathbf{u}^{\mathrm{I}} = (x, 0)$, as in Example 1 to test the performance of the *CG–algorithm*. In order to compare the numerical results with those obtained with the *E1–algorithm*, we chose $h = 1/40$ and $h/2 = 1/80$ in this case. To stop the iterations we choose $\varepsilon = 10^{-8}$ at step 3. Figure 3 shows the exact and the adjusted wind fields. The agreement is excellent this time, even at the vertical boundaries $x = 1$ and $x = 2$. The relative error and the average divergence are $e_r = 5.9 \times 10^{-4}$ and $mdiv = -5.3 \times 10^{-12}$, respectively. Note that we got a significant improvement: nearly two orders of magnitude better on the relative error, and about ten orders of magnitude better on the average divergence. The improvement of the relative error is mainly due to the reduction of the error on truncated boundaries, while the enhancing of average divergence is mainly due to the iterative method, because it stops when it reaches the tolerance (i.e. when the norm of the divergence is small enough).

To test further the *CG–algorithm* we consider two, more "realistic", additional examples. The first one includes a domain with a topography of a cosine–shape, and the second one includes a domain with a real topography. In both cases, the "exact" wind field was obtained with a Stokes solver using the methodology described in (Glowinski, 2003). The initial wind field $\mathbf{u}^I$ was obtained dropping the vertical component of the "exact" one in both cases. Then, the vector wind field is recovered using the same discretization parameters as in example 2.

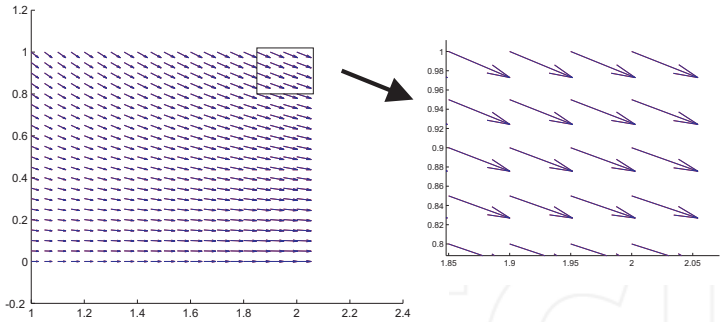


Fig. 3. Exact field $\mathbf{u} = (x, -z)$ in red, adjusted field obtained by the *CG–algorithm* in blue.

Example 3. Cosine–shape topography. In this case, we define the domain as follows

$$\Omega = \left\{ (x,y) \in \mathbb{R}^2 \ : \ 0 < x < 10, \ \ \frac{1}{2}\cos\frac{3\pi x}{10} + \frac{1}{2} < y < 10 \right\}.$$

The "exact" wind field satisfies $\nabla \cdot \mathbf{u} = 1.2 \times 10^{-16}$. Figure 4 shows the adjusted and "exact" wind fields.

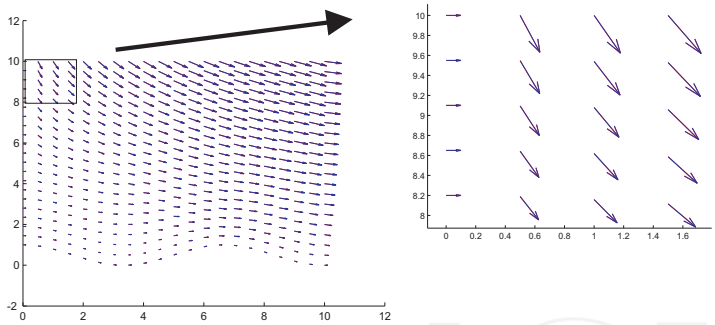


Fig. 4. "Exact" field for cosine topography in red, adjusted field obtained by the *CG–algorithm* in blue.

Example 4. Terrain elevation from real data. In this case, the domain is defined as

$$\Omega = \left\{ (x,y) \in \mathbb{R}^2 \ : \ 0 < x < 10, \ \ h(x) < y < 10 \right\},$$

where $h(x)$ is a function constructed via cubic splines, which interpolate discrete data over 10 Km of real topography of a certain region in Mexico, contained in a database (GTOPO, 1997). The "exact" wind field satisfies $\nabla \cdot \mathbf{u} = 6.1 \times 10^{-16}$. Figure 5 shows the adjusted and "exact" wind fields. We have an excellent agreement in all cases, even on truncated artificial boundaries. The relative error and the computed mean divergence are about the same order as in example 2. Table 2 shows a summary of the numerical results obtained with

the *CG-algorithm*. All numerical calculations were performed in a DELL Latitude D610 2.13 GHz laptop with an Intel Pentium M processor and 2 GB of RAM.

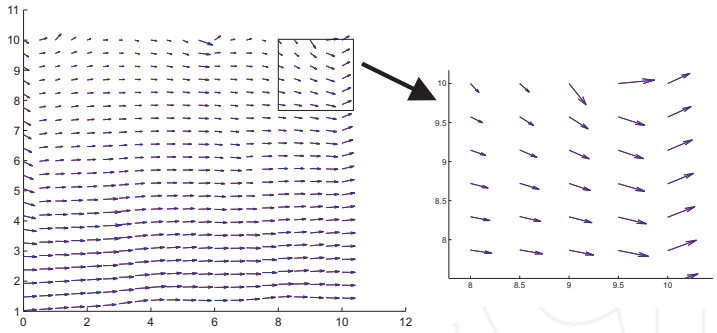


Fig. 5. "Exact" field for real topography in red, adjusted field obtained by the *CG–algorithm* in blue.

Example	Case with	e_r	*mdiv*	No. iters.	CPU–time (sec.)
2	u(x,z) = (x,-z)	5.9×10^{-4}	-5.3×10^{-12}	1214	3.9
3	cosine topography	3.6×10^{-6}	9.8×10^{-9}	955	3.3
4	real topography	4.1×10^{-6}	5.7×10^{-11}	1000	3.7

Table 2. Performance of the *CG–algorithm* for three different cases.

4.4 Preconditioned conjugate gradient method

The CPU time to solve the problem with the *CG–algorithm*, at the level of accuracy shown in Table 2, is about twice the CPU time needed to solve the problem with the *E1–algorithm*. In order to make this algorithm more reliable we need to speed up the iterative algorithm to get at least a comparable computational efficiency. Fortunately, we have found a good preconditioner for the iterative algorithm. This preconditioner is an optimal one, and we are presently working in its computer implementation, so we only describe here the main ideas without presenting numerical results yet.

Let $B : L_2(\Omega) \to L_2(\Omega)$ be an operator defined by

$$B\,q = \phi_q, \quad \text{where} \quad \phi_q \text{ solves} : \int_\Omega \left(S^{-1}\nabla\phi_q\right)\cdot\nabla\psi\,d\mathbf{x} = \int_\Omega q\,\psi\,d\mathbf{x} \qquad \forall\,\psi \in H^1(\Omega) \tag{26}$$

Operator B is self-adjoint and elliptic, and satisfies $A\,(B\,q) = q$, inside Ω, for every $q \in L^2(\Omega)$. An easy way to see these properties is considering the differential form of operators A and B:

$$A\,q = -\nabla\cdot\mathbf{u}_q = -\nabla\cdot(S^{-1}\nabla\,q), \quad \text{since} \quad S\,\mathbf{u}_q = \nabla q \quad \text{in} \quad \Omega, \tag{27}$$

$$B\,q = \phi_q = -[\nabla\cdot(S^{-1}\nabla)]^{-1}\,q, \quad \text{since} \quad -\nabla\cdot(S^{-1}\nabla\phi_q) = q \quad \text{in} \quad \Omega. \tag{28}$$

Then, from (27)–(28) we obtain

$$A\,(B\,q) = A\,\phi_q = -\nabla\cdot(S^{-1}\nabla\phi_q) = q. \tag{29}$$

This shows that B can be used as an optimal preconditioner. Therefore, the additional cost of the preconditioned conjugate gradient algorithm is the solution of an elliptic problem at each iteration. However, this additional cost is offsetted by two nice properties: a) the preconditioning must reduce drastically the number of iterations (from about 1000 to less than 20, based on previous experience in CFD); b) there is a significant reduction of degrees of freedom in the discrete version of the elliptic problem associated to operator B. This elliptic problem is solved in a coarse mesh, and it is four times smaller than the elliptic problem for the multiplier λ in the 2–D case, and about eight times smaller in 3–D problems.

5. Some extensions and future research

In this section, we present some additional alternatives to look at the problem. We first consider a different set of boundary conditions on vertical truncated boundaries for the multiplier λ, and we show that it produces better results than the traditional approach. We also show that if we introduce ghost nodes on the truncated artificial boundaries, we get even a better improvement. Finally, we introduce radial basis functions to solve the elliptic problems for the multiplier, and show that this is a promising alternative for 3-D wind fields.

5.1 Alternative boundary conditions for the elliptic problem

From equations (19)–(20), we obtain

$$\int_\Omega \left(S\mathbf{u} - \nabla\lambda - S\mathbf{u}^{\mathrm{I}}\right)\cdot\mathbf{v}\,d\mathbf{x} = \int_{\Gamma\setminus\Gamma_N}\lambda\,\mathbf{v}\cdot\mathbf{n}\,d\mathbf{\Gamma}, \quad \forall\,\mathbf{v}\in\mathbf{V}_N, \tag{30}$$

$$\int_\Omega q\nabla\cdot\mathbf{u}\,d\mathbf{x} = 0, \quad \forall\,q\in L_2(\Omega). \tag{31}$$

The boundary integral in (30) vanishes in two cases, namely: when $\mathbf{v}\cdot\mathbf{n} = 0$ or when $\lambda = 0$ on $\Gamma\setminus\Gamma_N$. The first case is not possible since it holds only on Γ_N, and the second case is not a good choice on vertical boundaries as we have already seen in Section 3. However, there is a possibility: decompose Γ_D as the union of the vertical boundaries, Γ_V, and the top boundary, Γ_T. Now, at Γ_T we still impose $\lambda = 0$, and on Γ_V we impose the new boundary condition $\mathbf{u}\cdot\mathbf{n} = \mathbf{u}^{\mathrm{I}}\cdot\mathbf{n}$. This new boundary condition is reasonable, since we assume that $\mathbf{u}^{\mathrm{I}}$ is the horizontal part of $\mathbf{u}$. Therefore, with this choice, we obtain the saddle–point problem (left) and its corresponding elliptic problem (right):

$$S\mathbf{u} - \nabla\lambda = S\mathbf{u}^{\mathrm{I}}, \quad \text{and} \quad \nabla\cdot\mathbf{u} = 0 \ \text{ in } \Omega, \qquad -\nabla\cdot\left(S^{-1}\nabla\lambda\right) = \nabla\cdot\mathbf{u}^{\mathrm{I}} \ \text{ in } \ \Omega, \tag{32}$$

$$\lambda = 0 \quad \text{on } \Gamma_T, \qquad \lambda = 0 \ \text{ on } \ \Gamma_T, \tag{33}$$

$$\mathbf{u}\cdot\mathbf{n} = \mathbf{u}^{\mathrm{I}}\cdot\mathbf{n} \quad \text{on } \Gamma_V, \qquad -S^{-1}\nabla\lambda\cdot\mathbf{n} = 0 \ \text{ on } \ \Gamma_V, \tag{34}$$

$$\mathbf{u}\cdot\mathbf{n} = 0 \quad \text{on } \Gamma_N. \qquad -S^{-1}\nabla\lambda\cdot\mathbf{n} = \mathbf{u}^{\mathrm{I}}\cdot\mathbf{n} \ \text{ on } \ \Gamma_N. \tag{35}$$

The finite element algorithm for the elliptic problem is: Given $\mathbf{u}_h^{\mathrm{I}} \in \mathbf{L}_h$, find $\lambda_h \in H_h$ such that

$$\int_\Omega S^{-1}\nabla\lambda_h \cdot \nabla q \, d\mathbf{x} = -\int_\Omega \mathbf{u}_h^{\mathrm{I}} \cdot \nabla q \, d\mathbf{x} + \int_{\Gamma_V} q \, \mathbf{u}_h^{\mathrm{I}} \cdot \mathbf{n} \, d\Gamma, \quad \forall\, q \in H_h, \tag{36}$$

where H_h is defined as in (13), but with $q = 0$ on Γ_T instead of Γ_D. Equation (36) differs from equation (14) only by the boundary integral on Γ_V. We call (36), together with (15), the *E2–algorithm*.

Example 5. Let us consider one more time the problem introduced in example 1 and in example 3, with the same discretization parameter, $h = 1/80$. The recovered wind field for both cases is better than the one obtained with the *E1–algorithm*, since the vertical component is recovered fairly well, not only in the interior of the domain but also at the vertical boundaries. Figure 6 shows the exact and recovered wind fields. Table 3 shows a summary of the results obtained in examples 1, 2, 3 and 5. For the case with exact wind field $\mathbf{u} = (x, -z)$ the immediate effect of this improvement is the reduction of the relative error by two orders of magnitude. However, we do not obtain a comparable reduction of the mean divergence. For the problems with cosine topography occurs the opposite. Actually, the numerical results show that the most effective algorithm to reduce both, the relative error and the average divergence is the *CG–algorithm*.

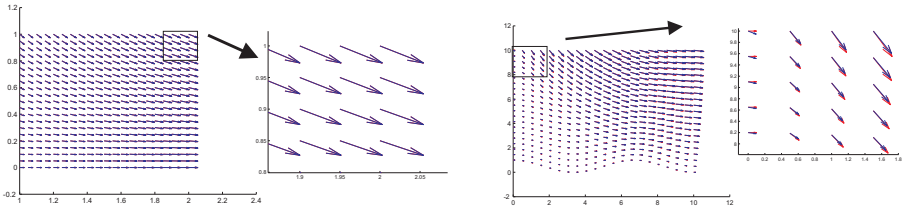


Fig. 6. Exact field (red) and adjusted field obtained by the *E2–algorithm* (blue). Left: case with exact field $\mathbf{u} = (x, -z)$. Right: case with cosine topography.

Ex.	Case with	Algorithm	e_r	$mdiv$	No. iters.	CPU time(s)
1	$\mathbf{u} = (x, -z)$	*E1–algorithm*	1.9×10^{-2}	4.1×10^{-2}	—	1.78
2	$\mathbf{u} = (x, -z)$	*CG–algorithm*	5.9×10^{-4}	-5.3×10^{-12}	1214	3.90
5	$\mathbf{u} = (x, -z)$	*E2–algorithm*	4.0×10^{-4}	1.8×10^{-2}	—	1.78
3	cosine topography	*CG–algorithm*	3.6×10^{-6}	9.8×10^{-9}	955	3.30
5	cosine topography	*E2–algorithm*	9.2×10^{-2}	3.6×10^{-4}	—	2.08

Table 3. Summary of the numerical results obtained in examples 1, 2, 3 and 5.

5.2 Ghost nodes

Given that $\mathbf{u}$ belongs to $\mathbf{V}$ and satisfies (2), then λ satisfies the equations

$$-\nabla \cdot (S^{-1}\nabla\lambda) = \nabla \cdot \mathbf{u}^{\mathrm{I}}, \quad \text{in} \quad \Omega, \tag{37}$$

$$-(S^{-1}\nabla\lambda) \cdot \mathbf{n} = \mathbf{u}^{\mathrm{I}} \cdot \mathbf{n}, \quad \text{on} \quad \Gamma_N. \tag{38}$$

Instead of looking for boundary conditions on Γ_D, we may enforce mass conservation by asking any solution of (37)–(38) to satisfy

$$\int_{\Gamma} \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \int_{\Gamma_D} (\mathbf{u}^{\mathrm{I}} + S^{-1} \nabla \lambda) \cdot \mathbf{n} \, d\Gamma = 0. \tag{39}$$

Equations (37)–(39) imply the identity $\int_{\Omega} \nabla \cdot \mathbf{u}^{\mathrm{I}} \, d\mathbf{x} = \int_{\Gamma} \mathbf{u}^{\mathrm{I}} \cdot \mathbf{n} \, d\Gamma$. Actually, this is the compatibility condition associated to the the above Poisson–Neumann–like problem. Therefore, this problem has a unique solution $\lambda \in H^1(\Omega)/\mathbb{R}$, and its variational formulation is: Given $\mathbf{u}^{\mathrm{I}} \in \mathbf{L}_2(\Omega)$, find $\lambda \in H^1(\Omega)/\mathbb{R}$ such that

$$\int_{\Omega} (S^{-1} \nabla \lambda) \cdot \nabla q \, d\mathbf{x} = -\int_{\Omega} \mathbf{u}^{\mathrm{I}} \cdot \nabla q \, d\mathbf{x} + \int_{\Gamma_D} q \, (\mathbf{u}^{\mathrm{I}} + S^{-1} \nabla \lambda) \cdot \mathbf{n} \, d\Gamma, \quad \forall \, q \in H^1(\Omega)/\mathbb{R}. \tag{40}$$

Observe that when $q = 1$ in $H^1(\Omega)/\mathbb{R}$, we recover (39). However, the computational solution of this problem is not trivial, since the matrix associated to the discrete version is semidefinite. On the other hand, the symmetry of the matrix is lost because the boundary integral in the right–hand side has the unknown λ. A way to overcome this computational problem is to introduce "ghost nodes", around and beyond of the nonphysical truncated boundary Γ_D. Then, we may impose $\lambda_h = 0$ and/or $\partial \lambda_h / \partial \mathbf{n} = 0$ on the outer layer of those ghost nodes. At the end, we discard the solution on the ghost nodes, and we only keep the solution values on the actual nodes. Actually, this is a well–known way to deal with differential equations in domains with truncated boundaries.

Example 6. We consider one more time the problem from example 1 with the same discretization parameters. We incorporate two layers of **ghost nodes** and impose $\lambda = 0$ on the outer layer. The recovered wind field obtained is such that the relative error and average weak divergence are $e_r = 2.1 \times 10^{-5}$ and $mdiv = 1.6 \times 10^{-6}$, respectively. The figure with the comparison of the adjusted wind field and the exact wind field is not shown, because it is very similar to Figure 5. Instead, we summarize in Table 4 the results for this example with the different algorithms.

Case	*E1–algortihm*	*E2–algortithm*	*Ghost-Nodes*	*CG–algorithm*
e_r	1.9×10^{-2}	4.0×10^{-4}	2.1×10^{-5}	5.4×10^{-4}
mdiv	4.1×10^{-2}	1.8×10^{-2}	1.6×10^{-6}	-5.2×10^{-12}

Table 4. Comparison of numerical solutions obtained with different algorithms.

Table 4 shows how boundary conditions degrade numerical calculations. It is observed that the solution improves each time the Dirichlet boundary condition $\lambda = 0$ is applied to a smaller section of the non-physical boundary. This is not surprising, since this boundary condition introduces a large artificial gradient, mainly on vertical truncated boundaries, when calculating the term $\nabla \lambda$ in order to get $\mathbf{u} = \mathbf{u}^{\mathrm{I}} + S^{-1} \nabla \lambda$ at the corresponding boundary nodes. At this time, and taking in account the performance of every algorithm, we may recommend to use either the classical approach with ghost nodes or the saddle point problem approach with the conjugate gradient algorithm, specially if we do not have enough information at truncated boundaries.

5.3 Approximation with radial basis functions

A function $\Phi : \mathbb{R}^d \to \mathbb{R}$ is called a radial basis function (RBF) if $\Phi(\mathbf{x}) = \phi(\|\mathbf{x}\|)$ where the kernel ϕ is a scalar function $\phi : \mathbb{R}^+ \to \mathbb{R}$, and d the spatial dimension. Usually $\|\mathbf{x}\|$ is denoted by r, and typical functions used in applications are, among others:

1. Multiquadrics, $\phi(r) = \sqrt{c^2 + r^2}$.
2. Gaussians, $\phi(r) = e^{-c\,r^2}$.
3. Thin plate splines, $\phi(r) = r^2 \ln(r)$.
4. Inverse multiquadrics, $\phi(r) = 1/\sqrt{c^2 + r^2}$.

The constant value c es called the *shape parameter*. The radial basis function method was first introduced in the 1970s for multivariate scattered data approximation, (Hardy, 1971). This interpolation problem is defined as follows:
Given a set of points $\{\mathbf{x}_j\}_{j=1}^n \subset \Omega \subset \mathbb{R}^d$, approximate the function $f(\mathbf{x})$ from the set of values $f_j = f(\mathbf{x}_j)$. A simple form is to define

$$s(\mathbf{x}) = \sum_{j=1}^{k} \lambda_j \, \phi(\|\mathbf{x} - \mathbf{x}_j\|) + p(\mathbf{x}). \tag{41}$$

where p is a polynomial which depends on the specific RBF. Then the interpolation condition

$$s(\mathbf{x}_i) = \sum_{j=1}^{k} \lambda_j \, \phi(\|\mathbf{x}_i - \mathbf{x}_j\|) + p(\mathbf{x}_i) = f_i, \quad i = 1, \ldots, n, \tag{42}$$

gives an algebraic system of equations for $\lambda = \{\lambda_i\}_{j=1}^k$. However, the corresponding matrix could be ill conditioned and, in some cases, even rank deficient, and special techniques are needed, like preconditioning and least squares (Buhmann, 2003), (Wendland, 2005).
In the last two decades, the main focus of the applications seems to have slowly shifted from scattered data approximation to the numerical solution of PDE. Radial basis function collocation methods for solving PDE are truly meshfree algorithms, in the sense that collocation points can be chosen freely and no connectivity between the points is needed or used (Kansa, 1990), (Narcowich & Ward, 1994). The main attraction of RBF collocation method to solve PDE is that it can be extended directly to solve 3–D problems. Moreover, due to the absence of a grid, these techniques are better suited than classical methods to cope with problems having complex boundaries. So, we think that the RBF collocation method is a good choice to study our problem, and we want to explore this alternative.
Let us denote by $\mathcal{L}$ the linear elliptic differential operator for the multiplier λ, and $\mathcal{B}$ the boundary operator. Suppose that we want to solve the problem

$$\mathcal{L}\,\lambda = -\nabla \cdot (S^{-1} \nabla \lambda) = \nabla \cdot \mathbf{u}^{\mathrm{I}} \quad \text{in} \quad \Omega, \tag{43}$$

$$\mathcal{B}\,\lambda = g \quad \text{on} \quad \Gamma_N. \tag{44}$$

We consider a set of collocation points $\{\mathbf{x}_i\}_{i=1}^n$, with n_i points in the interior and n_b points on the boundary, so that $n = n_i + n_f$. We look for an approximate solution $\lambda_h(\mathbf{x}) =$

$\sum_{j=1}^{n} \lambda_j \phi(\|\mathbf{x} - \mathbf{x}_j\|)$, where the unknown vector $\{\lambda_i\}_{i=1}^{n}$ satisfies the system of equations

$$\mathcal{L}\,\lambda_h(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j \,\mathcal{L}\,\phi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \nabla\cdot\mathbf{u}^{\mathrm{I}}(\mathbf{x}_i), \quad i = 1, \ldots, n_i, \tag{45}$$

$$\mathcal{B}\,\lambda_h(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j \,\mathcal{B}\,\phi(\|\mathbf{x}_i - \mathbf{x}_j\|) = g(\mathbf{x}_i), \quad i = n_i + 1, \ldots, n. \tag{46}$$

Therefore the recovered wind field $\mathbf{u}_h$ is given by

$$\mathbf{u}_h(\mathbf{x}) = \mathbf{u}^{\mathrm{I}}(\mathbf{x}) + \sum_{j=1}^{n} \lambda_j \, S^{-1}\nabla\phi(\|\mathbf{x} - \mathbf{x}_j\|). \tag{47}$$

Example 8. As a last example we include a 3–D numerical calculation using radial basis functions. The exact syntectic wind field for this example is $\mathbf{u} = (x, y, -2z)$. We dropped the vertical component so that $\mathbf{u}^{\mathrm{I}} = (x, y, 0)$. A multiquadric kernel with $c = 11.33$ was used, and we chose $\alpha_1 = \alpha_2 = \alpha_3 = 1$. The collocation points were obtained by a $5 \times 5 \times 5$ regular subdivision of $\Omega = (1,2) \times (0,1) \times (0,1)$. Figure 7 shows the collocation points and the comparison between the exact and recovered wind field. The agreement is excellent, we obtained a relative error $e_r = 1.98 \times 10^{-4}$ and mean divergence $mdiv = -5.59 \times 10^{-6}$.

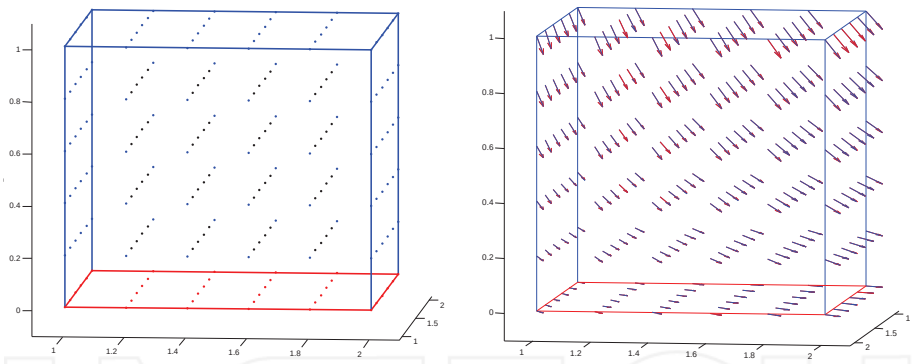


Fig. 7. Collocation points (left), and comparison of exact and recovered wind fields (right).

6. Conclusions

We studied the problem of generating an adjusted wind field from horizontal wind data by different numerical techniques. We have shown that boundary conditions can significantly affect numerical solutions depending of how we treat artificial truncated boundaries. The usual methodology (*E1–algorithm*) does not produce satisfactory results close to the vertical boundaries due to the high gradients introduced by the term $S^{-1}\nabla\lambda$ in (2), when homogeneous Dirichlet boundary conditions are imposed there. The formulation of the problem as a saddle–point one, with a functional equation that has a self–adjoint and strongly elliptic operator, allows the use of an iterative conjugate gradient algorithm (the *CG–algorithm*). This new methodology, in the context of mass consistent models, produces

much better results, it does not involve the solution of differential equations, and it does not require boundary conditions for the multiplier. An optimal preconditioner for the conjugate gradient algorithm was introduced, and we hope, based on previous experience with the solution of the Stokes problem, to reduce the number of iterations of nearly a thousand to a few tens.

In an attempt to improve the numerical results obtained with the traditional approach, we introduced new boundary conditions for the multiplier on vertical boundaries. These boundary conditions are demonstrated to be physically and mathematically consistent. The numerical reconstruction of the wind field was improved by two orders of magnitude, but a comparable reduction on the weak divergence is not always obtained. However, the introduction of "ghost nodes" produces more satisfactory results, reducing both the relative error and the mean weak divergence by two or more orders of magnitude. On the other hand, we have just started to explore meshfree methods. In particular, radial basis collocation methods seem to be a very simple reliably alternative to the reconstruction of three–dimensional wind fields, according to the preliminary numerical results shown in this work.

The application of the different alternatives and methodologies presented here to the more realistic three–dimensional case is a continuation of the present work. Another interesting issue is the potential extension and application of these methodologies to other experimental fields, such as fluid dynamics and computer vision. In particular, the reconstruction of solenoidal velocity fields from experimental data, obtained through the *particle image velocimetry* technique, is an important issue, (Adrian, 2005). Its relation with computer vision is established by *optical flow* estimation, (Ruhnau & Schnorr, 2007).

7. Acknowledgements

Authors wish to express our deep appreciation to InTech for the kind invitation to contribute with this chapter. No doubt this was a great motivation. Special thanks to Ms. Tajana Jevtic and Jana Sertic, InTech process managers, for their guidance, patience and kindness throughout the editorial process.

8. References

Adrian, R. J. (2005). Twenty years of particle image velocimetry, *Exp. Fluids* 39(2): 159–169.

Bercovier, M. & Pironneau, O. (1979). Error estimates for the finite element method solution of the stokes problem in the primitive variables, *Numer. Math.* 33: 211–224.

Buhmann, M. D. (2003). *Radial basis functions: theory and implementations*, Cambridge University Press, United Kingdom.

Castino, F., Rusca, L. & Solari, G. (2003). Wind climate micro–zoning: a pilot application to liguria region (north–western italy), *J. Wind. Eng. Ind. Aerodyn.* 91(11): 1353–1375.

Ciarlet, P. G. (2002). *The Finite Element Method for Elliptic Problems. Re–edited as Vol. 40, SIAM*, North–Holland Amsterdam (1970), Philadelphia, PA.

Ferragut, L., Montenegro, R., Montero, G., Rodríguez, E., Asensio, M. L. & Escobar, J. M. (2010). Comparison between 2.5-d and 3-d realistic models for wind field adjustmen, *J. Wind. Eng. Ind. Aerodyn.* 98(10–11): 548–558.

Finardi, S., Tinarelli, G., Nanni, A., Brusasca, G. & Carboni, G. (2010). Evaluation of a 3–d flow and pollutant dispersion modelling system to estimate climatological ground level concentrations in complex coastal sites, *Int. J. Environ. Pollut.* 16(1–6): 472–482.

Flores, C. F., Juárez, L. H., Nuñez, M. A. & Sandoval, M. L. (2010). Algorithms for vector field generation in mass consistent models, *J. Numer. Methods for PDE* 26(4): 826–842.

Girault, V. & Raviart, P. A. (1986). *Finite Element Methods for the Navier–Stokes Equations: Theory and Algorithms*, Springer–Verlag, Berlin.

Glowinski, R. (2003). *Numerical Methods for Fluids (Part 3), Handbook of Numerical Analysis, volume IX*, North–Holland, Amsterdam.

GTOPO, . (1997). Gtopo30 documentation, section 7, *U. S. Geological Survey* pp. 211–224. URL: *www.scd.ucar.edu/ dss/datasests/ds758.0hmtl*

Hardy, R. L. (1971). Multiquadric equations of topography and other irregular surfaces, *J. Geophys. Res.* 76(8): 1905–1915.

Kansa, E. J. (1990). Multiquadrics a scattered data approximation scheme with applications to computational fluid dynamics ii: Solutions to parabolic, hyperbolic and elliptic partial differential equations, *Comput. Math. Appl.* 19(8/9): 147–161.

Kitada, T., Igarashi, K. & Owada, M. (1986). Numerical analysis of air pollution in a combined field of land/sea breeze and mountain/valley wind, *J. Clim. Appl. Meteorol.* 25(6): 767–784.

Kitada, T., Kaki, A., H., U. & K., P. L. (1983). Estimation of the vertical air motion from limited horizontal wind data–a numerical experiment, *Atmos. Environ* 17(11): 181–2192.

Montero, G., Rodríguez, E., Montenegro, R., Escobar, J. M. & González-Yuste, J. M. (2005). Genetic algorithms for an improved parameter estimation with local refinement of tetrahedral meshes in a wind model, *Adv. Eng. Softw.* 36: 3–10.

Narcowich, F. J. & Ward, J. D. (1994). Generalized hermite interpolation via matrix-valued conditionally positive definite functions, *Math. Comp.* 63: 661–687.

Núñez, M. A., Flores, C. & Juárez, H. (2007). Interpolation of hydrodynamic velocity data with the continuity equation, *J. Comput. Meth. Sci. Eng.* 7(1): 21–42.

Núñez, M. A., Flores, C. & Juárez, L. H. (2006). A study of hydrodynamic mass–consistent models, *J. Comput. Meth. Sci. Eng.* 6: 1078–1089.

Pennel, W. T. (1983). *An evaluation of the role of numerical wind field models in wind turbine siting. Technical Report PNL–SA–11129*, Battelle Memorial Institute, Pacific Northwest Laboratory, Richland, Washington.

Potter, B. & Butler, B. (2009). Using wind models to more effectively manage wildfire, *Fire management today* 69(2): 40–46.

Ratto, C. F. (1996). An overview of mass-consistent models, *in* D. P. Lalas & C. F.Ratto (eds), *Modeling of Atmosphere Flow Fields*, World Scientific Publications, Place of publication, pp. 379–400.

Ratto, C. F., Festa, R., Romeo, C., Frumento, O. A. & Galluzzi, M. (1994). Mass–consistent models for wind fields over complex terrain: The state of the art, *Environ. Software* 9(4): 247–268.

Ross, D. G., Smith, I. N., Manins, P. C. & Fox, D. G. (1988). Diagnostic wind field modeling for complex terrain: Model development and testing, *J. Appl. Meteor.* 27: 785–796.

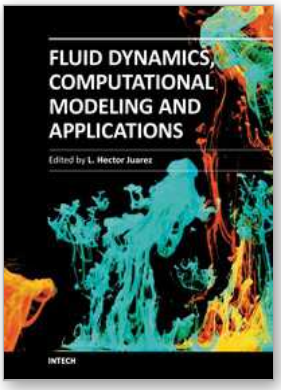
Ruhnau, P. & Schnorr, C. (2007). Optical stokes flow estimation: an imaging–based control approach, *Exp. Fluids* 42(1): 61–78.

Sasaki, Y. (1958). An objective analysis based on the variational method, *Journal Met. Soc. Japan* 36: 77–88.

Sherman, C. A. (1978). A mass–consistent model for wind fields over complex terrain, *J. Appl. Meteor.* Vol. 17(3): 312–319.

Wang, Y., Williamson, C., Garvey, D., Chang, S. & Cogan, J. (2005). Application of a multigrid method to a mass–consistent diagnostic wind model, *J. Appl. Meteor.* 44(7): 1078–1089.

Wendland, H. (2005). *Scattered data approximation*, Cambridge University Press, United Kingdom.



Fluid Dynamics, Computational Modeling and Applications

Edited by Dr. L. Hector Juarez

ISBN 978-953-51-0052-2
Hard cover, 660 pages
Publisher InTech
Published online 24, February, 2012
Published in print edition February, 2012

The content of this book covers several up-to-date topics in fluid dynamics, computational modeling and its applications, and it is intended to serve as a general reference for scientists, engineers, and graduate students. The book is comprised of 30 chapters divided into 5 parts, which include: winds, building and risk prevention; multiphase flow, structures and gases; heat transfer, combustion and energy; medical and biomechanical applications; and other important themes. This book also provides a comprehensive overview of computational fluid dynamics and applications, without excluding experimental and theoretical aspects.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

L. Héctor Juárez, María Luisa Sandoval, Jorge López and Rafael Reséndiz (2012). Mass–Consistent Wind Field Models: Numerical Techniques by L2–Projection Methods, Fluid Dynamics, Computational Modeling and Applications, Dr. L. Hector Juarez (Ed.), ISBN: 978-953-51-0052-2, InTech, Available from: http://www.intechopen.com/books/fluid-dynamics-computational-modeling-and-applications/mass-consistent-wind-field-models-numerical-techniques-by-l2-projection-methods

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821