Large-Scale Air Pollution Simulations with gridMathematica™

Anton Antonov
Wolfram Research, Inc.

Introduction

Generally in all mathematics, the most difficult problems are inverse problems. Air-pollution models are typically used for some strategy or policy for the sake of a society. In other words, by taking meteorological measurements and monitoring air pollutants (emissions) for some period of time (e.g., a month or year) in some region (e.g., Europe or California), using air pollution models we can point out where the emissions of the air pollutants have to be reduced, according to some cost function, in order to get the level of air pollution within some prescribed bounds. This is an optimizational problem; hence, it is an inverse problem. The forward problem is to calculate the concentrations of the pollutants over the region of interest according to the meteorological measurements and given emissions.

The inverse problem is too big, and it cannot currently be solved directly. More precisely, we attempt to answer its questions by solving a number of forward problems, taken and enumerated by some strategy according to the experts' knowledge in the field. Even the solutions of the forward problems are tedious and slow, since their region is very big (e.g., 5000km $\times$ 5000km covered by a 500×500 grid), and, say, between 30–120 pollutants are considered.

The ambitious task of large-scale air pollution simulation imposes different trade-offs that should be addressed by a flexible programming implementation of an air-pollution model. This paper illustrates how this can be accomplished using the environment for technical computing grid*Mathematica.* The sections of the paper that follow describe:

- the mathematical air pollution model
- the numerical method for the advection scheme
- choice and solution of chemical reaction scheme
- parallel implementation of the model
- performance and profiling test data

The model we have implemented follows and is similar to the Danish Eulerian Model (DEM) (see [A02] and [Z95]).

Mathematical Model

Temporal and spatial variations of the concentrations and/or the depositions of various harmful air pollutants can be studied [Z95] by solving the system (1) of partial differential equations (PDEs):

$$\frac{\partial c_s}{\partial t} = -\frac{\partial (u\, c_s)}{\partial x} - \frac{\partial (v\, c_s)}{\partial y} - \frac{\partial (w\, c_s)}{\partial z} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right) + E_s + Q_s(c_1, c_2, \ldots, c_q) - (k_{1\,s} + k_{2\,s})\, c_s, \qquad (1)$$

$$s = 1, 2, \ldots, q.$$

The different quantities that are involved in the mathematical model have the following meanings:

- The concentrations are denoted by c_s.

u, v, and w are wind velocities.
- K_x, K_y, and K_z are diffusion coefficients.
- The emission sources are described by E_s.
- $k_{1\,s}$ and $k_{2\,s}$ are deposition coefficients.
- The chemical reactions are described by the nonlinear functions $Q(c_1, c_2, \ldots, c_q)$.
- The number of equations q is equal to the number of species in the model.

It is difficult to treat the system of PDEs (1) directly. This is the reason for using different kinds of splitting. A simple splitting procedure, based on ideas discussed in [M75] and [Mc84], can be defined, for $s = 1, 2, \ldots, q$, by the following sub-models:

$$\frac{\partial c_s}{\partial t} = -\frac{\partial(u\,c_s)}{\partial x} - \frac{\partial(v\,c_s)}{\partial y},$$

$$\frac{\partial c_s}{\partial t} = \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right),$$

$$\frac{\partial c_s}{\partial t} = E_s + Q_s(c_1, c_2, \ldots, c_q),$$

$$\frac{\partial c_s}{\partial t} = -(k_{1\,s} + k_{2\,s})\,c_s,$$

$$\frac{\partial c_s}{\partial t} = -\frac{\partial(w\,c_s)}{\partial z} + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right),$$

$$s = 1, 2, \ldots, q.$$

In practice, the last system is often skipped.

The 2D simulations use the following splitting:

$$\frac{\partial c_s}{\partial t} = -\frac{\partial(u\,c_s)}{\partial x} - \frac{\partial(v\,c_s)}{\partial y} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right), \qquad (2)$$

$$\frac{\partial c_s}{\partial t} = E_s + Q_s(c_1, c_2, \ldots, c_q), \qquad (3)$$

$$s = 1, 2, \ldots, q.$$

The deposition is not simulated.

The 3D simulations use the following splitting:

$$\frac{\partial c_s}{\partial t} = -\frac{\partial(u\,c_s)}{\partial x} - \frac{\partial(v\,c_s)}{\partial y} - \frac{\partial(w\,c_s)}{\partial z} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right), \qquad (4)$$

$$\frac{\partial c_s}{\partial t} = E_s + Q_s(c_1, c_2, \ldots, c_q), \qquad (5)$$

$$s = 1, 2, \ldots, q.$$

The deposition is not simulated.

We assume that the diffusion coefficients in (1) are positive constants, sufficiently small, so that the advection dominates over the diffusion. Then equations (2) and (4) become

$$\frac{\partial c_s}{\partial t} = -\frac{\partial (u\, c_s)}{\partial x} - \frac{\partial (v\, c_s)}{\partial y} + K_x \frac{\partial^2 c_s}{\partial x^2} + K_y \frac{\partial^2 c_s}{\partial y^2}, \tag{6}$$

and

$$\frac{\partial c_s}{\partial t} = -\frac{\partial (u\, c_s)}{\partial x} - \frac{\partial (v\, c_s)}{\partial y} - \frac{\partial (w\, c_s)}{\partial z} + + K_x \frac{\partial^2 c_s}{\partial x^2} + K_y \frac{\partial^2 c_s}{\partial y^2} + K_z \frac{\partial^2 c_s}{\partial z^2}, \tag{7}$$

respectively.

Numerical Scheme for the Advection Equations

Here we fix s in (6), and instead of c_s use c:

$$\frac{\partial c}{\partial t} = -\frac{\partial (u\, c)}{\partial x} - \frac{\partial (v\, c)}{\partial y} + K_x \frac{\partial^2 c}{\partial x^2} + K_y \frac{\partial^2 c}{\partial y^2}. \tag{8}$$

We consider the problem of finding the solution of the equation (8) for a fixed s over the rectangular domain $\Omega = [0, X] \times [0, Y]$ in the interval $[t_0, t_1]$ with given initial conditions $c(x, y, 0) = c_0(x, y)$, $(x, y) \in \Omega$. To specify the two types of boundary conditions we denote the outward normal vector to $\partial\Omega$ with $\vec{n}$, and let $\vec{w} = \begin{bmatrix} u(x, y) \\ v(x, y) \end{bmatrix}$. We have an "outflow" boundary when the $\vec{n} \cdot \vec{w} > 0$, and an "inflow" boundary when $\vec{n} \cdot \vec{w} < 0$. At the outflow boundary we should not specify any boundary conditions (the PDE prevails here). At the inflow boundary we impose the condition $\partial c \big/ \partial \vec{n} = 0$.

For deeper discussion of the problem and the boundary conditions see [GS98].

Finite Element Spacial Discretization

The particular finite element used is Galerkin Finite Element Method (GFEM) with piecewise linear basis functions $N_i(x, y)$, where $i \in I$, I being the set of the vertices of the graph used to discretize the spatial domain of the model (see Figure 1).

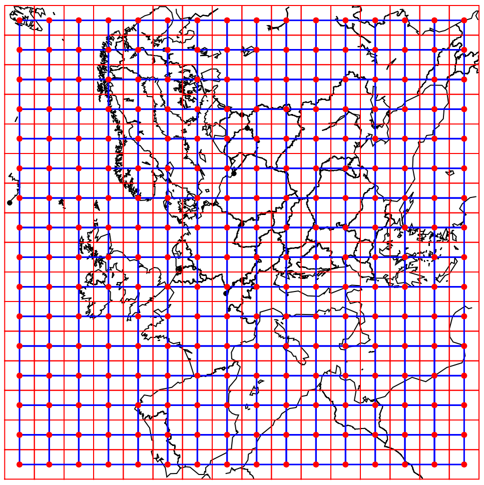

Figure 1. Example of a rectangular simulation grid over a geographical region (Europe).

These types of methods are discussed in detail in [M82] and [GS98]. The function $c(x, y, t)$ in (8) is approximated in the form $c(x, y, t) = \Sigma g_i(t) N_i(x, y)$ and, using the Galerkin principle, we have the equation

$$\sum_{i \in I} \frac{\partial g_i(t)}{\partial t} \int\int_\Omega N_i(x, y) N_j(x, y) \, dx \, dy =$$
$$\sum_{i \in I} g_i(t) \int\int_\Omega \left(- \frac{\partial (u(x, y, t) N_i(x, y))}{\partial x} N_j(x, y) - \frac{\partial (v(x, y, t) N_i(x, y))}{\partial y} N_j(x, y) + \right. \tag{9}$$
$$\left. K_x \frac{\partial N_i(x, y)}{\partial x} \frac{\partial N_j(x, y)}{\partial x} + K_y \frac{\partial N_i(x, y)}{\partial x} \frac{\partial N_j(x, y)}{\partial x} \right) dx \, dy, \;\; j \in I.$$

This equation can be written in operator form,

$$P \frac{\partial g(t)}{\partial t} = A g(t), \tag{10}$$

where $g(t)$ is the vector $g(t) = \{g_i(t)\}_{i=1}^{|I|}$, $g(t) \in \mathbb{R}^{|I|}$.

The application of θ-method (see [L72]) to handle numerically (10) gives

$$g^{k+1} = g^k + \Delta t \left(\theta P^{-1} A g^{k+1} + (1 - \theta) P^{-1} A g^k \right). \tag{11}$$

We apply the θ-method with $\theta = 1/2$, which results in the well-known Crank–Nicholson method (see again [L72]).

Three-dimensional discretization is done analogously.

Chemistry Reactions Simulation

We implemented a simple chemistry scheme taken from [PS98] (see Figure 2 and Figure 3).

$$\begin{aligned} CO + OH &\xrightarrow{O_2} CO_2 + HO_2 \\ HO_2 + NO &\longrightarrow NO_2 + OH \\ NO_2 + h\nu &\longrightarrow NO + O \\ O + O_2 + M &\longrightarrow O_3 + M \end{aligned}$$

Figure 2. Carbon monoxide atmospheric oxidation.

$$NO_2 \underset{H_2O}{\overset{h\nu}{\rightleftharpoons}} NO + O_3$$

$$CO + OH \underset{NO}{\longrightarrow} HO_2$$

Figure 3. Two cycles fast and slow.

The implementation is straightforward.

1. The reaction coefficients are programmed as time-dependent functions.
2. The chemical equations are converted to ordinary differential equations (ODEs).
3. The system of ODEs is given to *Mathematica*'s `NDSolve`.

Figure 4 shows the code at step **2.**

$$\text{species} = \{\text{"}O_3\text{"}, \text{"}NO_2\text{"}, \text{"NO"}, \text{"CO"}\};$$

$$a = \frac{1}{1 + \frac{k_{5.22}\, c[\text{"M"}]}{k_{5.23}\, c[\text{"}H_2O\text{"}, RH]}}$$

$$eq[\text{"}O_3\text{"}] = c[\text{"}O_3\text{"}, t] == \frac{k_{5.1}[t]\; c[\text{"}NO_2\text{"}, t]}{k_{5.3}\, c[\text{"NO"}, t] + k_{5.21\,b}\; a}$$

$$eq[\text{"}NO_2\text{"}] = D[c[\text{"}NO_2\text{"}, t], t] == \frac{k_{5.1}[t]\; k_{5.21\,b}\; a\; (2\, k_{5.24}\, c[\text{"CO"}, t] / k_{5.26} - 3\, c[\text{"}NO_2\text{"}, t])}{k_{5.3}\, c[\text{"NO"}, t] + k_{5.21\,b}\; a}$$

$$eq[\text{"NO"}] = D[c[\text{"NO"}, t], t] == \frac{j_{5.1}\; k_{5.21\,b}\; a\; (c[\text{"}NO_2\text{"}, t] - 2\, k_{5.24}\, c[\text{"CO"}, t] / k_{5.26})}{k_{5.3}\, c[\text{"NO"}, t] + k_{5.21\,b}\; a}$$

$$eq[\text{"CO"}] = D[c[\text{"CO"}, t], t] == \frac{2\, k_{5.1}[t]\; k_{5.21\,b}\; k_{5.24}\; a\; c[\text{"CO"}, t] / k_{5.26}}{k_{5.3}\, c[\text{"NO"}, t] + k_{5.21\,b}\; a}$$

$$(*\ N_2 + O_2\ \text{ppm}\ *)\, c[\text{"M"}] = 780\,840 + 209\,460;$$

Figure 4. Equations for `NDSolve`.

Parallel Implementation

As can be seen in Figure 1, the chemistry reactions happen within the boundaries of the red cells and species concentrations are redistributed by the advection with the blue grid. Obviously the chemical reactions` simulation of the model is trivially parallelized. The challenge of the parallel implementation of the model is to ensure proper advection simulation.

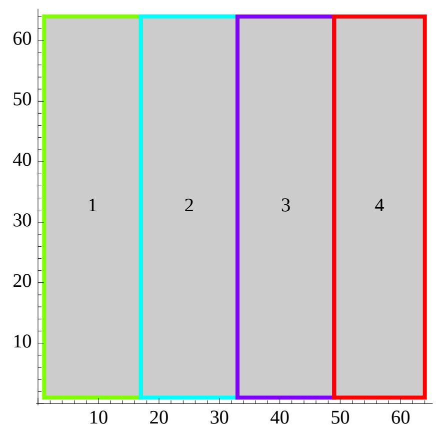

Figure 5. Domain of 64×64 nodes partitioned among 4 processors across the x axis.

Consider a computer with 4 processors. A simple way to partition the simulation domain is to divide it along one of the axes. On each processor we can have GFEM simulation. The processors overlap if placed in the original domain. In this way we basically run four GFEM advection simulations in parallel. We may say that the parallelism is at the GFEM level. Partitioning along one of the axes, though, might bring certain ill-conditioning (see Figure 6). A better approach is to partition along both axes, as shown in Figure 7. In our simulations we used both partitionings.

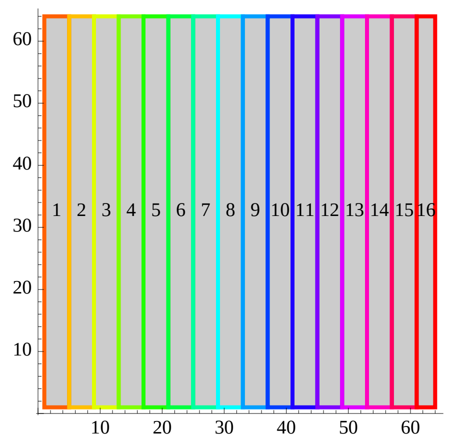

Figure 6. Domain of 64×64 nodes partitioned among 16 processors across the x axis.

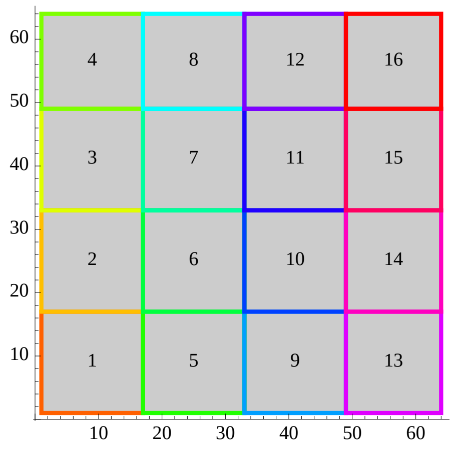

Figure 7. Partitioning on both axes leads to better conditioned space domains.

If instead of partitioning the physical domain we partition in the pseudo-domain generated by equation (9), it would be easier to make well-balanced parallel distributions of locally refined grids [A02]. If GFEM is used, this would mean that only one GFEM is used for the advection simulation. The parallelism is pushed at the linear algebra level.

Parallel Runs

Good speed-up results with the described parallel implementation were achieved using Microsoft CCS 2003 with 8, 16, and 32 cores of the Tyan Personal Supercomputer, which is built upon Intel quad-core Xeon processors coupled with Mellanox InfiniBand.

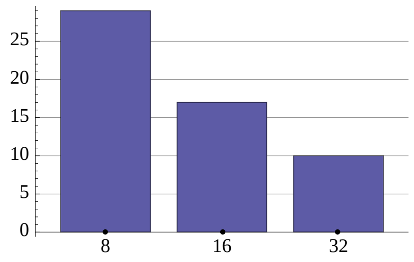

Figure 8. Timing for one step on 8, 16 and 32 processors.

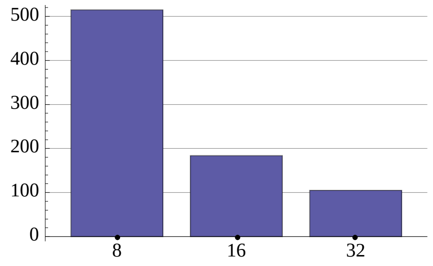

Figure 9. Timing for 10 steps on 8, 16 and 32 processors.

References

[A02] A. Antonov, "Object-Oriented Framework for Large-Scale Air Pollution Models," PhD thesis, The Technical University of Denmark, 2002. http://www.imm.dtu.dk/ ▮uniaaa/OODEM.

[GS98] P. M. Gresho and R. L. Sani, *Incompressible Flow and Finite Element Method*, New York: John Wiley & Sons, 1998.

[L72] J. D. Lambert, *Computational Methods in Ordinary Differential Equations*, New York: John Wiley, 1973.

[M75] G. I. Marchuk, *Methods of Numerical Mathematics*, 2nd ed., New York: Springer-Verlag, 1982.

[Mc84] G. J. McRae, W. R. Goodin, and J. H. Seinfeld, "Numerical Solution of the Atmospheric Diffusion Equation for Chemically Reacting Flows," J. Comp. Phys., 45(1), 1982 pp. 1–42.

[SP98] J. H. Seinfeld and S. N. Pandis, *Atmospheric Chemistry and Physics*, New York: John Wiley & Sons, 1998.

[V82] R. Vichnevetsky and J. B. Bowles, *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*, SIAM, 1982.

[Z95] Z. Zlatev, *Computer Treatment of Large Air Pollution Models*, Kluwer, 1995.