Numerical Methods for Parallel Simulation of Diffusive Pollutant Transport from a Point Source

Tim Brown[1], Arjun Pandya[2], Noah Sienkiewicz[3],
faculty mentor: Matthias K. Gobbert[1],

[1]Department of Mathematics and Statistics, UMBC
[2]Department of Information Systems, UMBC
[3]Department of Physics, UMBC

CyberTraining: Big Data + High-Performance Computing + Atmospheric Sciences
University of Maryland, Baltimore County, Spring 2018
`cybertraining.umbc.edu`

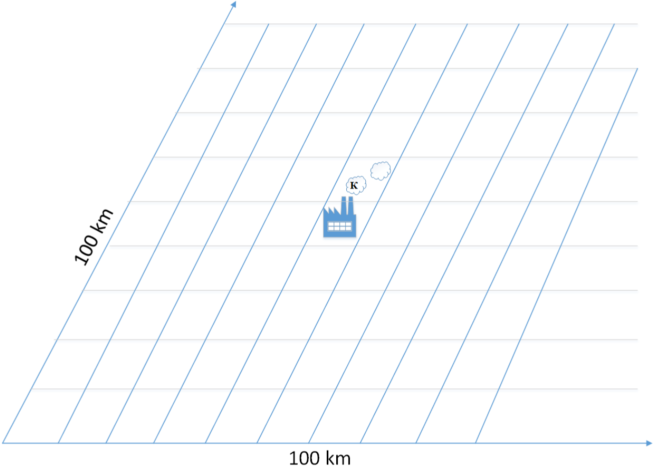
Acknowledgments: NSF, UMBC, HPCF

Outline

- Physical Model: Pollution from a factory at center of a region
- Mathematical Model: Diffusion of mass from point source (factor)
- Numerical Methods: Finite Volume approximation
  - Implicit time discretization
  - Parallelizable linear solver (conjguate gradient)
- Results:
  - Numerical study of test problem with known solution
  - Diffusion of pollution in closed system
  - Diffusion of pollution in open system
  - Parallel performance study
- Conclusions and Future Work

Point Pollution Source at Center of Domain

Our physical model involves a polluting factory at the center of a mesh.



Physical Model

We consider a problem that blends **Atmospheric Physics** and **HPC**:

- Diffusion with diffusivity D of aerosols emitted from a pollution source in a stable environment (no wind), modeled as a point source in the center $(x, y) = (50, 50)$ of a two-dimensional region of size 100 km by 100 km
- The source pollutes with κ kg per hour for the duration of a typical work day of 8 hours, then shuts off; the simulation runs for 24 hours, beginning when the pollution starts.
- The environment is modeled as closed system, with the two-dimensional region large enough that the pollution does not reach the boundary.
- Spatial region can be increased, allowing for sub domains to model regions of open boundaries
- Simulation starts with no pollution present throughout region.

The example represents, for instance, a factory in the center of a region on the scale of a city or county.
Big Data could be applied to ouput if the parameter κ or the location is not assumed to be known.

Math Note: Dirac Delta vs Kronecker Delta

The Dirac delta distribution is defined by the properties

$$\delta(x-a) = \begin{cases} 0 & x \neq a \\ \infty & x = a \end{cases} \text{ and } \int_{-\infty}^{\infty} \delta(x)dx = 1$$

The Kronecker Delta $\delta_{i,j}$ is a function of the 2 arguments i and j If i and j are the same value (i.e. $i = j$) then the function $\delta_{i,j}$ is equal to 1. Otherwise the Kronecker Delta is equal to zero. Formally this is written:

$$\delta_{i,j} = \begin{cases} 0 & for\ i \neq j \\ 1 & for\ i = j \end{cases}$$

Mathematical Model

Find concentration $u(x,y,t)$ of pollution in units of kg/km^2 such that

$$\begin{aligned} u_t - D\,\nabla\cdot(\nabla u) &= f(x,y,t) && \text{for } (x,y)\in\Omega \text{ and } t>0,\\ \mathbf{n}\cdot\nabla u &= 0 && \text{for } (x,y)\in\partial\Omega \text{ and } t>0,\\ u &= u_{ini}(x,y) && \text{for } (x,y)\in\overline{\Omega} \text{ at } t=0, \end{aligned}$$

i. spatial domain region $\Omega = (-50,50)\times(-50,50)\subset\mathbb{R}^2$ in units of km and time domain $0\le t\le 24$ in units of hours h
ii. diffusivity coefficient D in km^2/h is constant and
iii. source $f = \kappa\,\delta_{(0,0)}(x,y)\,\chi_{[0,8]}(t)$ with rate κ in kg km^{-2} h^{-1}; Dirac delta distribution in 2-D $\delta_{(0,0)}(x,y) := \delta(x-0)\,\delta(y-0)$; indicator function $\chi_{[0,8]}(t) = 1$ for $0\le t\le 8$ and 0 otherwise
iv. $\mathbf{n} = \mathbf{n}(x,y)$ outward unit normal vector at $(x,y)\in\partial\Omega$
v. initial concentration of pollution $u_{ini}\equiv 0$

Finite Volume Method (FVM): Idea

Primal mesh defined by points $(x_i, y_j) \in \overline{\Omega}$, $x_i = (i-1)h$, $y_j = (j-1)h$, $i, j = 1, \ldots, N_0$, with uniform mesh spacing $h = 1/(N-1)$. Assume N_0 odd integer, such that center point $(0,0)$ is guaranteed mesh point. Define dual mesh $\mathcal{T}_h = \cup_{(i,j)} \Omega_{ij}$ of cells around each mesh point (x_i, y_j). The FVM starts from the PDE converted to conservative integral form

$$\iint_{\Omega_{ij}} \frac{\partial u}{\partial t} \, dx \, dy - \iint_{\Omega_{ij}} D \, \nabla \cdot (\nabla u) \, dx \, dy = \iint_{\Omega_{ij}} f \, dx \, dy$$

by integration over each cell $\Omega_{ij} \in \mathcal{T}_h$.

Advantages of the FVM:

1. mass conservation of the discrete concentrations $u_{ij}(t) \approx u(x_i, y_j, t)$
2. ability to handle point sources, for instance, $\kappa \iint_{\Omega_{ij}} \delta_{(0,0)}(x, y) \, dx \, dy \, \chi_{[0,8]}(t) = \kappa \, \delta_{(0,0)}(x_i, y_j) \, \chi_{[0,8]}(t)$ with (generalized) Kronecker delta function $\delta_{(0,0)}(x_i, y_j) = 1$ if $(x_i, y_j) = (0,0)$ and 0 otherwise.

FVM for a Boundary Cell: Semi-Discretization

By divergence theorem applied to the diffusion term in the conservative integral form, the diffusion term explicitly tracks the flow through the boundary $\partial\Omega_{ij}$ of Ω_{ij} in

$$\iint_{\Omega_{ij}} \frac{\partial u}{\partial t}\, dx\, dy - D \int_{\partial\Omega_{ij}} \mathbf{n} \cdot (\nabla u)\; dS = \iint_{\Omega_{ij}} f\, dx\, dy,$$

for each cell $\Omega_{ij} \in \mathcal{T}_h$, $i, j = 1, \ldots, N_0$, of the dual mesh.
For a boundary cell at bottom boundary of $\partial\Omega$, $i = 2, \ldots, N_0 - 1$, $j = 1$, $\Omega_{ij} = (x_i - \frac{h}{2}, x_i + \frac{h}{2}) \times (0, \frac{h}{2})$, has size $h^2/2$, and the first and last integrals approximate to:

$$\iint_{\Omega_{ij}} \frac{\partial u(x,y,t)}{\partial t}\, dx\, dy \approx \int_0^{h/2} \int_{x_i - h/2}^{x_i + h/2} 1\, dx\, dy\, \frac{du(x_i, y_j, t)}{dt} \approx \frac{h^2}{2} \frac{du_{ij}(t)}{dt},$$

$$\iint_{\Omega_{ij}} f\, dx\, dy \approx \int_0^{h/2} \int_{x_i - h/2}^{x_i + h/2} 1\, dx\, dy\, f(x_i, y_j, t) = \frac{h^2}{2} f(x_i, y_j, t).$$

The surface integral over the diffusion term is calculated by considering each of the four linear segments of $\partial\Omega_{ij}$, one of which is a subset of the domain boundary $\partial\Omega$.

FVM for a Boundary Cell: Semi-Discretization

The segment $x_i - \frac{h}{2} \le x \le x_i + \frac{h}{2}$, $y_j = 0$ with $\mathbf{n} = (0,-1)^T$ lies on the domain boundary $\partial\Omega$, **thus by boundary condition $\mathbf{n} \cdot \nabla u = 0$**

$$\int_{x_i - h/2}^{x_i + h/2} \mathbf{n} \cdot \nabla u \, dx = 0$$

One of the other segments is at $x = x_i - \frac{h}{2}$ for $0 \le y \le \frac{h}{2}$ with outward unit normal vector $\mathbf{n} = (-1,0)^T$, then

$$\int_0^{h/2} \mathbf{n} \cdot \nabla u \, dy = \int_0^{h/2} (-1,0) \begin{pmatrix} u_x \\ u_y \end{pmatrix} dy = -\int_0^{h/2} u_x(x_i - \tfrac{h}{2}, y) \, dy$$

$$\approx -\int_0^{h/2} 1 \, dy \, \frac{u_{ij} - u_{i-1j}}{h} = -\frac{h}{2} \frac{u_{ij} - u_{i-1j}}{h} = -\tfrac{1}{2}(u_{ij} - u_{i-1j}) = \tfrac{1}{2}(u_{i-1j} - u_{ij})$$

Analogous derivations for the other segments of $\partial\Omega_{ij}$ yield

$$\int_{\partial\Omega_{ij}} \mathbf{n} \cdot (\nabla u) \; dS \approx \tfrac{1}{2}\left(u_{i-1j} - u_{ij}\right) + \tfrac{1}{2}\left(u_{i+1j} - u_{ij}\right) + \left(u_{ij+1} - u_{ij}\right)$$

$$= \tfrac{1}{2} u_{i-1j} - 2u_{ij} + \tfrac{1}{2} u_{i+1j} + u_{ij+1}$$

FVM for a Boundary Cell: Semi-Discretization

Insert all approximations into the integral equation

$$\frac{h^2}{2}\,\frac{du_{ij}(t)}{dt} - D\Big(\tfrac{1}{2}u_{i-1j}(t) - 2u_{ij}(t) + \tfrac{1}{2}u_{i+1j}(t) + u_{ij+1}(t)\Big) = \frac{h^2}{2}\,f(x_i, y_j, t)$$

Divide by h^2 to get the semi-discretization

$$\frac{1}{2}\frac{du_{ij}(t)}{dt} + \frac{D}{h^2}\Big(-\tfrac{1}{2}u_{i-1j}(t) + 2u_{ij}(t) - \tfrac{1}{2}u_{i+1j}(t) - u_{ij+1}(t)\Big) = \frac{1}{2}\,f(x_i, y_j, t)$$

for all boundary mesh points (x_i, y_j), $i = 2, \ldots, N_0 - 1$, $j = 1$.

FVM for a Boundary Cell: Full Discretization

Evaluate the semi-discretization at time $t = t_{n+1} = t_n + \Delta t$ and use backward Euler as approximation for the time derivative to get

$$\frac{1}{2}\frac{u_{ij}(t_{n+1}) - u_{ij}(t_n)}{\Delta t} + \frac{D}{h^2}\Big(-\tfrac{1}{2}u_{i-1j}(t_{n+1}) + 2u_{ij}(t_{n+1}) - \tfrac{1}{2}u_{i+1j}(t_{n+1}) - u_{ij+1}(t_{n+1})\Big)$$

$$\approx \tfrac{1}{2}f(x_i, y_j, t_{n+1}), \quad n = 0, 1, \ldots$$

Multiplying by Δt, introducing approximation $u_{ij}^{(n)} \approx u_{ij}(t_n)$, organizing terms with unknowns $u^{(n+1)}$ on the left, and using the short-hand notation $f_{ij}^{(n)} := f(x_i, y_j, t_n)$ yields the full discretization

$$-\frac{D}{2}\frac{\Delta t}{h^2}u_{i-1j}^{(n+1)} + \left(\frac{1}{2} + 2D\frac{\Delta t}{h^2}\right) u_{ij}^{(n+1)} - \frac{D}{2}\frac{\Delta t}{h^2}u_{i+1j}^{(n+1)} - D\frac{\Delta t}{h^2}u_{ij+1}^{(n+1)}$$

$$= \tfrac{1}{2}u_{ij}^{(n)} + \tfrac{\Delta t}{2}f_{ij}^{(n+1)}, \quad n = 0, 1, \ldots$$

for all boundary mesh points (x_i, y_j), $i = 2, \ldots, N_0 - 1$, $j = 1$.

FVM: All Types of Full Discretizations

Assemble the three possible full discretizations — namely for: corner point, boundary point (shown), interior point —, after multiplying each to make the diagonal coefficients the same value of $1 + 4D\frac{\Delta t}{h^2}$ for all three:

$$\left(1 + 4D\frac{\Delta t}{h^2}\right)\, u_{ij}^{(n+1)} - 2D\frac{\Delta t}{h^2} u_{i+1j}^{(n+1)} - 2D\frac{\Delta t}{h^2} u_{ij+1}^{(n+1)}$$

$$= u_{ij}^{(n)} + \Delta t\, f_{ij}^{(n+1)}, \quad i = 1, \quad j = 1,$$

$$-D\frac{\Delta t}{h^2} u_{i-1j}^{(n+1)} + \left(1 + 4D\frac{\Delta t}{h^2}\right)\, u_{ij}^{(n+1)} - D\frac{\Delta t}{h^2} u_{i+1j}^{(n+1)} - 2D\frac{\Delta t}{h^2} u_{ij+1}^{(n+1)}$$

$$= u_{ij}^{(n)} + \Delta t\, f_{ij}^{(n+1)}, \quad i = 2, \dots, N_0 - 1, \quad j = 1,$$

$$-D\frac{\Delta t}{h^2} u_{ij-1}^{(n+1)} - D\frac{\Delta t}{h^2} u_{i-1j}^{(n+1)} + \left(1 + 4D\frac{\Delta t}{h^2}\right)\, u_{ij}^{(n+1)} - D\frac{\Delta t}{h^2} u_{i+1j}^{(n+1)} - D\frac{\Delta t}{h^2} u_{ij+1}^{(n+1)}$$

$$= u_{ij}^{(n)} + \Delta t\, f_{ij}^{(n+1)}, \quad i = 2, \dots, N_0 - 1, \quad j = 2, \dots, N_0 - 1.$$

FVM: Full Discretization in Matrix Form

Define the column-vector $\mathbf{u}^{(n)} = (\mathbf{u}_k^{(n)})$, $k = 1, \ldots N$ with $N := N_0^2$, with components $\mathbf{u}_k^{(n)} = u_{ij}^{(n)}$, $k = i + N_0\,(j-1)$ for $i, j = 1, \ldots, N_0$; also analogously $\mathbf{f}^{(n)} = (\mathbf{f}_k^{(n)})$ with $\mathbf{f}_k^{(n)} = f_{ij}^{(n)}$.
Assembling all types of full discretizations yields then in matrix form

$$\left(I + D\frac{\Delta t}{h^2}\,A\right)\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \Delta t\,\mathbf{f}^{(n+1)}, \quad n = 0, 1, \ldots$$

with $A = I \otimes T + T \otimes I \in \mathbb{R}^{N\times N}$ computed as sum of Kronecker products between the identity matrix $I \in \mathbb{R}^{N_0\times N_0}$ and the tri-diagonal matrix

$$T = \begin{bmatrix} 2 & -2 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -2 & 2 \end{bmatrix} \in \mathbb{R}^{N_0\times N_0}$$

FVM: Example of Matrix A for $N_0 = 4$

$$
A = \left[\begin{array}{cccc:cccc:cccc:cccc}
4 & -2 & & & -2 & & & & & & & & & & & \\
-1 & 4 & -1 & & & -2 & & & & & & & & & & \\
& -1 & 4 & -1 & & & -2 & & & & & & & & & \\
& & -2 & 4 & & & & -2 & & & & & & & & \\
\hdashline
-1 & & & & 4 & -2 & & & -1 & & & & & & & \\
& -1 & & & -1 & 4 & -1 & & & -1 & & & & & & \\
& & -1 & & & -1 & 4 & -1 & & & -1 & & & & & \\
& & & -1 & & & -2 & 4 & & & & -1 & & & & \\
\hdashline
& & & & -1 & & & & 4 & -2 & & & -1 & & & \\
& & & & & -1 & & & -1 & 4 & -1 & & & -1 & & \\
& & & & & & -1 & & & -1 & 4 & -1 & & & -1 & \\
& & & & & & & -1 & & & -2 & 4 & & & & -1 \\
\hdashline
& & & & & & & & -2 & & & & 4 & -2 & & \\
& & & & & & & & & -2 & & & -1 & 4 & -1 & \\
& & & & & & & & & & -2 & & & -1 & 4 & -1 \\
& & & & & & & & & & & -2 & & & -2 & 4
\end{array}\right]
$$

Outline of Results

Outline of results:

- Numerical study of test problem with known solution
- Diffusion of pollution in closed system
- Diffusion of pollution in open system
- Parallel performance study

Test Problem with Smooth Right-Hand Side

$$
\begin{aligned}
u_t - D\,\nabla\cdot(\nabla u) &= f\,(x,y,t) && \text{for } (x,y)\in\Omega \text{ and } t>0,\\
\mathbf{n}\cdot\nabla u &= 0 && \text{for } (x,y)\in\partial\Omega \text{ and } t>0,\\
u &= u_{ini}\,(x,y) && \text{for } (x,y)\in\overline{\Omega} \text{ at } t=0,
\end{aligned}
$$

i spatial domain region $\Omega = (-50,50)\times(-50,50)\subset\mathbb{R}^2$

ii diffusivity $D = 10$, source given function with $\tau = 8$:
$f(x,y,t) = (2t/\tau^2)e^{-t^2/\tau^2}\cos^2(\pi x/100)\cos^2(\pi y/100)$
$+D(1-e^{-t^2/\tau^2})((-2\pi^2/100^2)\cos(2\pi x/100)\cos^2(\pi y/100)$
$+(-2\pi^2/100^2)\cos^2(\pi x/100)\cos(2\pi y/100))$

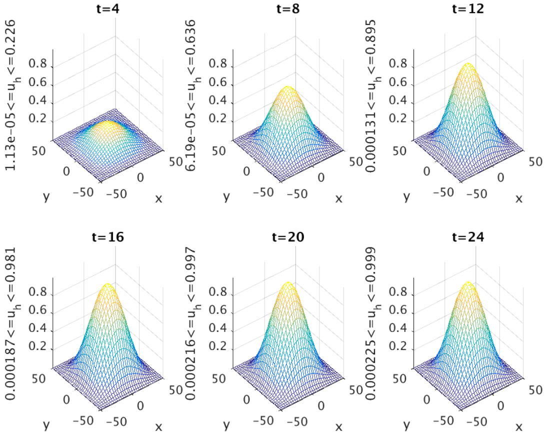
iii $\mathbf{n} = \mathbf{n}\,(x,y)$ outward unit normal vector at $(x,y)\in\partial\Omega$

iv $u_{ini}\equiv 0$

This test problem with smooth source admits known true solution
$u(x,y,t) = (1-e^{-t^2/\tau^2})\cos^2(\pi x/100)\cos^2(\pi y/100)$.

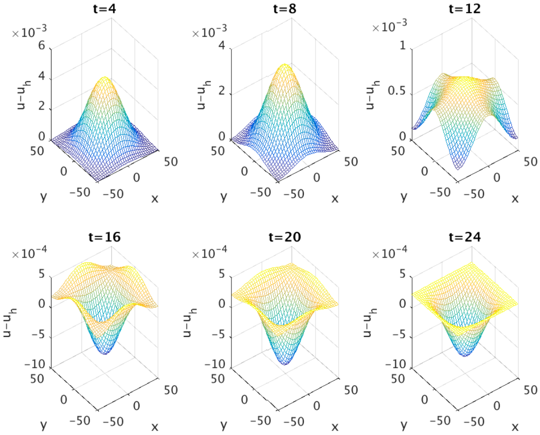
Test Problem: Matlab Solution

Diffusivity $D = 10$, $\tau = 8$, snapshots at $t = 4, 8, \ldots, 24$.
Matlab solution for a 129×129 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}.



Test Problem: Matlab Error

Diffusivity $D = 10$, $\tau = 8$, snapshots at $t = 4, 8, \ldots, 24$.
Matlab solution for a 129×129 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}.



Test Problem: Matlab Output

$D = 10$, $\tau = 8$, 129×129 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}; behavior of $\min(u)$, $\max(u)$, total mass $\iint_\Omega u\, dx\, dy$, and error norm agree with true solution at times $t = 4, 8, \ldots, 24$:

```
kappa =  1.000000e+01 D =  1.000000e+01 tau =  8.000000e+00
xmax =  5.000000e+01 xmaxout =  5.000000e+01 t_fin =  2.400000e+01
N0 =  129 N = 16641 h =  7.812500e-01
dt =  1.000000e-01 Nt =  2.400000e+02
tol =  1.000000e-09 maxit =  999
  n  t_n it cumit     min(U(:))    max(U(:))     mass(U(:))       enorminf
 39  4.0  7   273  1.131256e-05 2.256441e-01  5.651271e+02  4.444927e-03
 79  8.0  7   561  6.185924e-05 6.356576e-01  1.591709e+03  3.537069e-03
119 12.0  8   850  1.314239e-04 8.953037e-01  2.241353e+03  7.029653e-04
159 16.0  7  1140  1.866904e-04 9.810095e-01  2.455282e+03  6.748332e-04
199 20.0  5  1420  2.164897e-04 9.972224e-01  2.495258e+03  8.470974e-04
239 24.0  5  1619  2.250605e-04 9.991729e-01  2.499638e+03  7.037356e-04
```

Tests with mesh size confirm spatial convergence of the FVM.

Tests with time step Δt confirm convergence of time discretization.

Cost of implementation is represented by cumulative number of CG iterations.

Pollution Problem

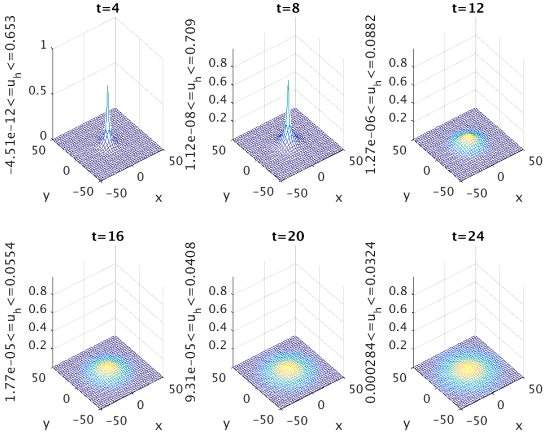
Now apply method to the problem of factory pollution.
Goals:

- Show that method conserves mass within the numerical domain.
- Demonstrate ability to simulate boundaries with and without pollution flow.
- Explore the relationship of mesh resolution and domain size and how it invites use of parallel computing.

First, plot entire domain on a modest mesh to examine first two properties.

Pollution Problem for $\kappa = 10$ on $(-50, 50) \times (-50, 50)$

Diffusivity $D = 10$ km^2 h^{-1}, $\kappa = 10$ kg km^{-2} h^{-1}, snapshots at $t = 4, 8, \ldots, 24$. Matlab solution for a 129×129 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}. Calculation on $(-50, 50) \times (-50, 50)$ km, plotted on $(-50, 50) \times (-50, 50)$ km



Pollution Problem for $\kappa = 10$ on $(-50, 50) \times (-50, 50)$

Output with parameters:

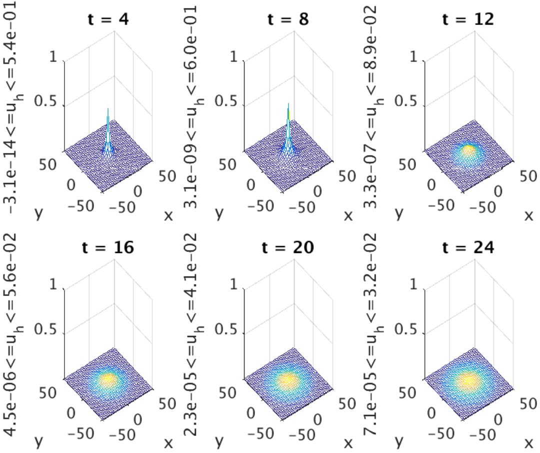
- Resolution is $h = 1.5625$ km
- Increasing mass until shutoff time, then constant
- Decreasing maximum after shutoff time

```
kappa =   1.000000e+01 D =   1.000000e+01 tau =   8.000000e+00
xmax =   5.000000e+01 xmaxout =   5.000000e+01 t_fin =   2.400000e+01
N0 =   129 N = 16641 h =   7.812500e-01
dt =   1.000000e-01 Nt =   2.400000e+02
tol =   1.000000e-09 maxit =   999
  n   t_n  it  cumit      min(U(:))      max(U(:))     mass(U(:))       enorminf
 39   4.0  12    755  -4.510617e-12  6.534136e-01  4.000000e+01  6.534136e-01
 79   8.0  11   1182   1.117690e-08  7.091521e-01  8.000000e+01  7.091521e-01
119  12.0  11   1926   1.272000e-06  8.819662e-02  8.000000e+01  8.819662e-02
159  16.0  10   2357   1.767457e-05  5.544710e-02  8.000000e+01  5.544710e-02
199  20.0  10   2753   9.312632e-05  4.080380e-02  8.000000e+01  4.080380e-02
239  24.0   8   3135   2.842079e-04  3.236277e-02  8.000000e+01  3.236277e-02
```

Next increase numerical boundaries while maintaing plot boundaries to simulate pollution flow at edges.

Pollution for $\kappa = 10$ on $(-800, 800) \times (-800, 800)$

Diffusivity $D = 10$ km^2 h^{-1}, $\kappa = 10$ kg km^{-2} h^{-1}, snapshots at $t = 4, 8, \ldots, 24$. Matlab solution for a 1025×1025 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}. Calculation on $(-800, 800) \times (-800, 800)$ km plotted on $(-50, 50) \times (-50, 50)$ km



Pollution for $\kappa = 10$ on $(-800, 800) \times (-800, 800)$

Output with parameters:

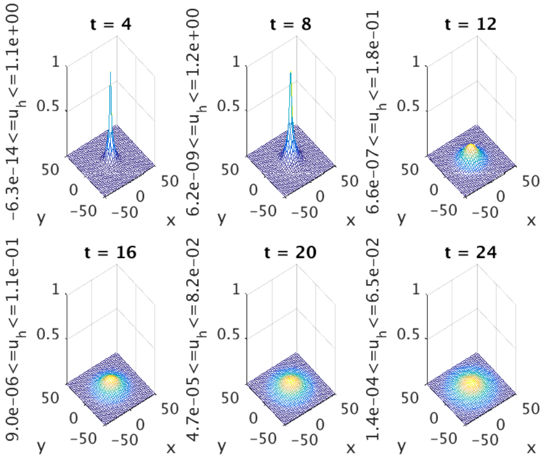
- Resolution is $h = 0.78125$ km (half of previous example)

```
kappa =   1.000000e+01 D =   1.000000e+01 tau =   8.000000e+00
xmax =   8.000000e+02 xmaxout =   5.000000e+01 t_fin =   2.400000e+01
N0 = 1025 N =1050625 h =   1.562500e+00
dt =   1.000000e-01 Nt =   2.400000e+02
tol =   1.000000e-09 maxit =   999
  n   t_n  it  cumit       min(U(:))      max(U(:))     mass(U(:))       enorminf
 39   4.0   7    440   -4.406830e-13   5.426228e-01   4.000000e+01   5.426228e-01
 79   8.0   6    676   -4.294251e-13   5.986023e-01   8.000000e+01   5.986023e-01
119  12.0   6   1100   -1.667979e-13   8.851597e-02   8.000000e+01   8.851597e-02
159  16.0   6   1340   -8.390670e-14   5.556418e-02   8.000000e+01   5.556418e-02
199  20.0   6   1561   -7.811444e-14   4.086560e-02   8.000000e+01   4.086560e-02
239  24.0   5   1776   -3.884732e-13   3.240026e-02   8.000000e+01   3.240026e-02
```

Number of cells has increased from 129 to 1025, but resolution has only halved. This is because of larger calculation domain, necessary to prevent pollution flowing back into plotting domain.

Pollution for $\kappa = 20$ on $(-800, 800) \times (-800, 800)$

Diffusivity $D = 10$ km^2 h^{-1}, $\kappa = 20$ kg km^{-2} h^{-1}, snapshots at $t = 4, 8, \ldots, 24$. Matlab solution for a 1025×1025 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}. Calculation on $(-800, 800) \times (-800, 800)$ km plotted on $(-50, 50) \times (-50, 50)$ km



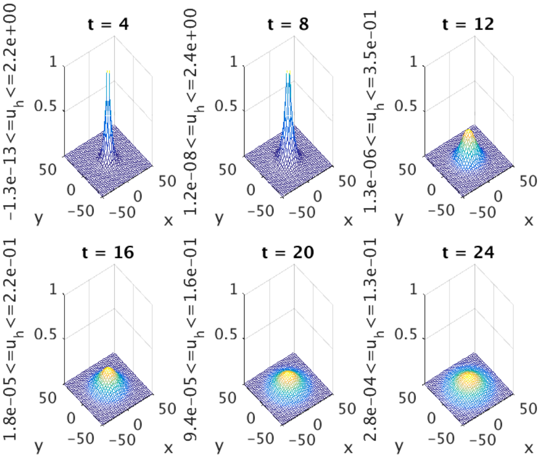
Pollution for $\kappa = 20$ on $(-800, 800) \times (-800, 800)$

Output with parameters:

```
kappa =  2.000000e+01 D =  1.000000e+01 tau =  8.000000e+00
xmax =  8.000000e+02 xmaxout =  5.000000e+01 t_fin =  2.400000e+01
N0 =  1025 N =     1050625 h =  1.562500e+00
dt =  1.000000e-01 Nt =  2.400000e+02
tol =  1.000000e-09 maxit =  999
    n  t_n  it cumit   min(U(:))   max(U(:))  mass(U(:))    enorminf
   39  4.0   7   440  -8.8137e-13  1.0852e+00  8.0000e+01  1.0852e+00
   79  8.0   6   676  -8.5885e-13  1.1972e+00  1.6000e+02  1.1972e+00
  119 12.0   6  1100  -3.3360e-13  1.7703e-01  1.6000e+02  1.7703e-01
  159 16.0   6  1340  -1.6781e-13  1.1113e-01  1.6000e+02  1.1113e-01
  199 20.0   6  1561  -1.5623e-13  8.1731e-02  1.6000e+02  8.1731e-02
  239 24.0   5  1776  -7.7695e-13  6.4801e-02  1.6000e+02  6.4801e-02
```

Pollution for $\kappa = 40$ on $(-800, 800) \times (-800, 800)$

Diffusivity $D = 10$ km^2 h^{-1}, $\kappa = 40$ kg km^{-2} h^{-1}, snapshots at $t = 4, 8, \dots, 24$. Matlab solution for a 1025×1025 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}. Calculation on $(-800, 800) \times (-800, 800)$ km plotted on $(-50, 50) \times (-50, 50)$ km



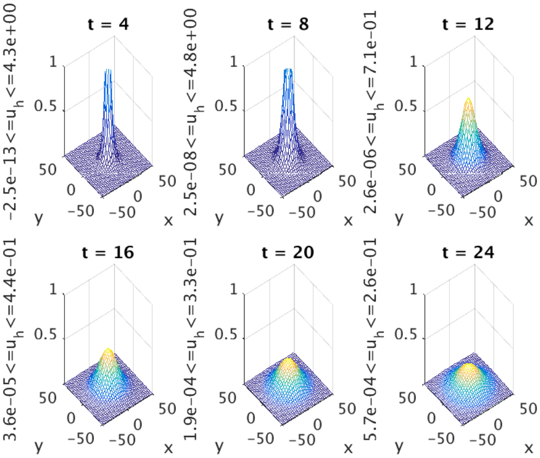
Pollution for $\kappa = 40$ on $(-800, 800) \times (-800, 800)$

Output with parameters:

```
kappa =  4.000000e+01 D =  1.000000e+01 tau =  8.000000e+00
xmax =  8.000000e+02 xmaxout =  5.000000e+01 t_fin =  2.400000e+01
N0 =  1025 N =     1050625 h =  1.562500e+00
dt =  1.000000e-01 Nt =  2.400000e+02
tol =  1.000000e-09 maxit =  999
    n  t_n  it cumit   min(U(:))   max(U(:))  mass(U(:))   enorminf
   39  4.0   7   440  -1.7627e-12  2.1705e+00  1.6000e+02  2.1705e+00
   79  8.0   6   676  -1.7177e-12  2.3944e+00  3.2000e+02  2.3944e+00
  119 12.0   6  1100  -6.6719e-13  3.5406e-01  3.2000e+02  3.5406e-01
  159 16.0   6  1340  -3.3563e-13  2.2226e-01  3.2000e+02  2.2226e-01
  199 20.0   6  1561  -3.1246e-13  1.6346e-01  3.2000e+02  1.6346e-01
  239 24.0   5  1776  -1.5539e-12  1.2960e-01  3.2000e+02  1.2960e-01
```

Pollution for $\kappa = 80$ on $(-800, 800) \times (-800, 800)$

Diffusivity $D = 10$ km^2 h^{-1}, $\kappa = 80$ kg km^{-2} h^{-1}, snapshots at $t = 4, 8, \ldots, 24$. Matlab solution for a 1025×1025 mesh, $\Delta t = 10^{-1}$, CG tolerance 10^{-9}. Calculation on $(-800, 800) \times (-800, 800)$ km plotted on $(-50, 50) \times (-50, 50)$ km



Pollution for $\kappa = 80$ on $(-800, 800) \times (-800, 800)$

Output with parameters:

```
kappa =  8.000000e+01 D =  1.000000e+01 tau =  8.000000e+00
xmax =  8.000000e+02 xmaxout =  5.000000e+01 t_fin =  2.400000e+01
N0 =  1025 N =     1050625 h =  1.562500e+00
dt =  1.000000e-01 Nt =  2.400000e+02
tol =  1.000000e-09 maxit =  999
    n  t_n  it cumit   min(U(:))   max(U(:))  mass(U(:))   enorminf
   39  4.0   7   440  -3.5255e-12  4.3410e+00  3.2000e+02  4.3410e+00
   79  8.0   6   676  -3.4354e-12  4.7888e+00  6.4000e+02  4.7888e+00
  119 12.0   6  1100  -1.3344e-12  7.0813e-01  6.4000e+02  7.0813e-01
  159 16.0   6  1340  -6.7125e-13  4.4451e-01  6.4000e+02  4.4451e-01
  199 20.0   6  1561  -6.2492e-13  3.2692e-01  6.4000e+02  3.2692e-01
  239 24.0   5  1776  -3.1078e-12  2.5920e-01  6.4000e+02  2.5920e-01
```

Parallel Performance

As we can see:

- Use of open boundaries requires large mesh to prevent mass returning to domain of interest.
- To maintain fine resolution with large mesh we need to increase $N0$ beyond what is reasonable for Matlab.

At this point, we translate the code created in Matlab to code in C using MPI. Here we can divide vectors and arrays up between processes and perform many calculations at once.
To gauge the efficiency of our result, we'll perform a parallel performance study of the test problem used previously using multiple arrangements of nodes and processes per node.

Parallel Performance: Wall Clock Time in Seconds

512 × 512 mesh	1 node	2 nodes	4 nodes	8 nodes
1 proc. per node	23	13	7	4
2 proc. per node	12	7	4	3
4 proc. per node	6	4	3	3
8 proc. per node	3	3	3	3
16 proc. per node	2	3	5	7
1024 × 1024 mesh	1 node	2 nodes	4 nodes	8 nodes
1 proc. per node	251	99	52	28
2 proc. per node	100	51	27	16
4 proc. per node	53	27	16	11
8 proc. per node	29	16	11	10
16 proc. per node	17	12	12	16
2048 × 2048 mesh	1 node	2 nodes	4 nodes	8 nodes
1 proc. per node	3043	1727	586	222
2 proc. per node	1236	541	239	112
4 proc. per node	909	305	118	62
8 proc. per node	904	214	71	43
16 proc. per node	446	178	81	44

Conclusions and Future Work

Conclusions:

- Finite Volume method allows for good handling of mass conservative physical problems.
- Use of open boundaries requires large meshes but fine resolution, making computation requirements ever increasing.
- Problem is easily parallelized by balancing mesh over computer nodes, but cost of computer node arrangements cancels out increases in computation time if too many nodes are used.

Future Work:

- Utilize output meshes for big data analysis of source location and characteristics.
- Examine nature of diffusion steady state for more complex source arrangements.