# Large-scale Air Pollution Modelling. Rotational Test.

K. Georgiev

## 1 Introduction

The task for finding reliable and robust control strategies for keeping the air pollution under certain safe levels is of great importance for the modern society. The air pollution is not restricted to the regions where there are big emissions sources, but also in the surrounding areas and even further. In order to estimate and predict the pollution levels and to take efforts for reducing the air pollution to some acceptable levels the output results of the air pollution models used have to be as reliable as possible. Large, complex and advanced air pollution models are used to meet this requirement. All physical and chemical processes including in the phenomenon, which is well known under the name "*Long-Range Transport of Air Pollution*" (LRTAP), are adequately described in them. LRTAP consists of: emission, transport of the air pollutants due to the wind and transformations during the transport (diffusion, deposition and chemical reactions). Such a model, e.g., is the *Danish Eulerian Model* (DEM) for long transport of air pollutants, which is developed at the National Environmental Research Institute (Roskilde, Denmark) ([12, 3]). The computer simulations with large air pollution models are very much time consuming mainly because the model domain is very large and the size of the systems of ordinary differential equations that arise after the space discretization is enormous. As the air quality models are used for the evaluation of emission control strategies and planning for the control of air pollution episodes, the computations should be kept tractable.

The Eulerian approach in which the behaviour of the species is described relative to a fixed coordinate system is a common way of treating heat and mass transfer. The Eulerian statistics are readily measurable and the mathematical expressions are directly applicable to the situations in which the chemical reactions take place. Let us denote with $q$ the number of the pollutants studied, and let $\Omega$ be the model domain. The Eulerian models are described by systems of partial differential equations (PDE's) like the following one, which is used in DEM:

$$\begin{aligned}\frac{\partial c_s}{\partial t} &= -\frac{\partial(uc_s)}{\partial x} - \frac{\partial(vc_s)}{\partial y} - \frac{\partial(wc_s)}{\partial z} + \\ &+\frac{\partial}{\partial x}\left(K_x\frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y\frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z\frac{\partial c_s}{\partial z}\right) + \qquad (1)\\ &+E_s + Q_s(c_1, c_2, \ldots, c_q) - (k_{1s} + k_{2s})\, c_s, \qquad s = 1, 2, \ldots, q,\end{aligned}$$

where $c_s = c_s(x, y, z, t)$ are the concentrations of the pollutants; $u = u(x, y, z, t)$, $v = v(x, y, z, t)$ and $w = w(x, y, z, t)$ are the wind velocities; $K_x$, $K_y$ and $K_z$ are the diffusion coefficients; the emission sources are described by $E_s$; $k_{1s}$ and $k_{2s}$ are the deposition coefficients, and finally, the chemical reactions are described by $Q_s$ (this term couples the system of PDE's and moreover it introduces nonlinearity in the model). As a rule in the atmospheric models, an operator splitting is used. The main advantage of this approach is that when each process in (1) is treated separately the most efficient numerical integration technique can be chosen. This leads as well to limitation of the total computation time. The splitting procedure proposed in [6, 5] which is used in DEM, leads to five submodels representing the horizontal advection (2), the horizontal diffusion (3), the chemistry and the emissions (4), the depositions (dry and wet) (5) and the vertical exchange (6).

$$\frac{\partial c_s^{(1)}}{\partial t} = -\frac{\partial(uc_s^{(1)})}{\partial x} - \frac{\partial(vc_s^{(1)})}{\partial y} \qquad (2)$$

$$\frac{\partial c_s^{(2)}}{\partial t} = \frac{\partial}{\partial x}\left(K_x\frac{\partial c_s^{(2)}}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y\frac{\partial c_s^{(2)}}{\partial y}\right) \qquad (3)$$

$$\frac{dc_s^{(3)}}{dt} = E_s + Q_s(c_1^{(3)}, c_2^{(3)}, \ldots, c_q^{(3)}) \qquad (4)$$

$$\frac{dc_s^{(4)}}{dt} = -(\kappa_{1s} + \kappa_{2s})c_s^{(4)} \qquad (5)$$

$$\frac{dc_s^{(5)}}{dt} = -\frac{\partial(wc_s^{(5)})}{\partial z} + \frac{\partial}{\partial z}\left(K_z\frac{\partial c_s^{(5)}}{\partial z}\right). \qquad (6)$$

These five systems of PDE's are solved successively at each time step. The concentrations obtained for each submodel at given time step are used as initial conditions for the next submodel in row. So, solving (6) at the time step $t$ the initial conditions for (2) at the time step $(t + \Delta t)$ are obtained.

In the two-dimensional version of DEM which will be considered later on in this study, the vertical exchange submodel (6) is skipped and $c_s = c_s(x, y, t)$, $u = u(x, y, t)$ and $v = v(x, y, t)$.

# 2 Rotational test and its discrete formulation

The *rotational test* ([1, 8]) is the most popular tool among the researchers in the fields of meteorology and environmental modelling for testing the numerical algorithms used in the large air pollution models. From the mathematical point of view, it is a pure advection problem which creates difficulties in the numerical treatment.

Let us consider the hyperbolic first order partial differential equation

$$Lc \equiv \frac{\partial c(x,y,t)}{\partial t} + u(x,y,t)\frac{\partial c(x,y,t)}{\partial x} + v(x,y,t)\frac{\partial c(x,y,t)}{\partial y} \qquad (7)$$

with the following initial and boundary conditions

$$\begin{aligned} c(x,y,0) &= c_0(x,y) \\ c(x,y,t) &= 0 \qquad (x,y) \in \partial\Omega, \end{aligned}$$

where $\Omega$ is a two-dimensional domain bounded by lines parallel to the coordinate axes and $0 < t \leq T$.

Let $\omega_h$ be a uniform mesh covering the domain $\Omega$ with meshsizes $h_x$ and $h_y$ respectively, and let $\tau$ be the time step.

Let us consider three difference schemes for space discretization of (7) - **central difference scheme (CD), upwind scheme of 1st order (UW1)** and **upwind scheme of 2nd order (UW2)**.

The implementation of CD-scheme together with a 2nd order discretization in time leads to the following discrete problem:

$$\begin{aligned} L_h c \equiv (\widehat{c}_{i,j} - c_{i,j}) \quad &+ \frac{\tau}{4h_x} u_{ij}(\widehat{c}_{i+1,j} - \widehat{c}_{i-1,j}) + \frac{\tau}{4h_y} v_{ij}(\widehat{c}_{i,j+1} - \widehat{c}_{i,j-1}) \\ &+ \frac{\tau}{4h_x} u_{ij}(c_{i+1,j} - c_{i-1,j}) + \frac{\tau}{4h_y} v_{ij}(c_{i,j+1} - c_{i,j-1}), \end{aligned} \qquad (8)$$

where $c_{i,j} = c(x_i, y_j, t) \equiv c(x_i, y_j, t^n)$ and $\widehat{c}_{i,j} = c(x_i, y_j, t+\tau) \equiv c(x_i, y_j, t^{n+1})$. The following theorem (see e.g. [9]) is hold:

**Theorem 1** *The local approximation error of the difference operator $L_h$ satisfies the estimation*

$$\varepsilon = |Lc - L_h c| = O(h^2 + \tau^2).$$

The implementation of UW1-scheme together with a 2nd order discretization in time will leads to the following discrete problem:

$$\begin{aligned} L_h c \equiv (\widehat{c}_{i,j} - c_{i,j}) \quad & + \frac{\tau}{4h_x}\left[(|u_{ij}| + u_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i-1,j}) + (|u_{ij}| - u_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i+1,j})\right] \\ & + \frac{\tau}{4h_y}\left[(|v_{ij}| + v_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i,j-1}) + (|v_{ij}| - v_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i,j+1})\right] \\ & + \frac{\tau}{4h_x}\left[(|u_{ij}| + u_{ij})(c_{i,j} - c_{i-1,j}) + (|u_{ij}| - u_{ij})(c_{i,j} - c_{i+1,j})\right] \\ & + \frac{\tau}{4h_y}\left[(|v_{ij}| + v_{ij})(c_{i,j} - c_{i,j-1}) + (|v_{ij}| - v_{ij})(c_{i,j} - c_{i,j+1})\right] \end{aligned} \tag{9}$$

The following theorem (see e.g. [9]) is hold:

**Theorem 2** *The local approximation error of the difference operator $L_h$ satisfies the estimation*

$$\varepsilon = |Lc - L_h c| = O(h + \tau^2).$$

*Pintchukov* introdused in [10] a correction algorithm for constructing high accuracy conservative scheme considering one dimensional problems. The implementation of his approach to the hyperbolic equation (7) leads to the following finite difference scheme

$$\begin{aligned} L_h c \equiv (\widehat{c}_{i,j} - c_{i,j}) \quad & + \frac{\tau}{4h_x}\left[(|u_{ij}| + u_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i-1,j}) + (|u_{ij}| - u_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i+1,j})\right] \\ & + \frac{\tau}{4h_y}\left[(|v_{ij}| + v_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i,j-1}) + (|v_{ij}| - v_{ij})(\widehat{c}_{i,j} - \widehat{c}_{i,j+1})\right] \\ & - \frac{\tau}{4h_x}\left[(|u_{ij}| - u_{ij})(c_{i,j} - c_{i-1,j}) + (|u_{ij}| + u_{ij})(c_{i,j} - c_{i+1,j})\right] \\ & - \frac{\tau}{4h_y}\left[(|v_{ij}| - v_{ij})(c_{i,j} - c_{i,j-1}) + (|v_{ij}| + v_{ij})(c_{i,j} - c_{i,j+1})\right] \end{aligned} \tag{10}$$

The following two theorems summarize the properties of this scheme.

**Theorem 3** *The local approximation error of the difference operator $L_h$ satisfies the estimation*

$$\varepsilon = |Lc - L_h c| = O(h^2 + \tau^2).$$

**Theorem 4** *Let $\omega_h$ be a given mesh. There exists $\tau_0$ such that, if $\tau \leq \tau_0$ then the scheme (10) is monotone.*

It follows from Theorem 2 that, if $c_0(x, y) \geq 0$ and if the time step $\tau$ is small enough, then the approximate sollution remains nonnegative during the computations. The number

$$K = \tau \ \max\left(\frac{|u(x,y)|}{h_x}, \frac{|v(x,y)|}{h_y}\right)$$

has a meaning of *Courant* number. i.e. the CFL (*Courant, Friedrichs* and *Lewy*) condition (necessary and sufficient condition for any explicit scheme to be stable) can be read as $K \leq 1$

The monotonicity of (10) is especially important in air pollution modelling because an uncertain choice of the spatial discretization may lead to a non-physical solution, i.e. negative concentrations.

The governing system of partial differential equations for the rotational test (one pollutant only, i.e. $c_s = c$) is:

$$\frac{\partial c}{\partial t} = -(1 - y)\frac{\partial c}{\partial x} - (x - 1)\frac{\partial c}{\partial y}, \tag{11}$$

where $0 \leq x \leq 2, \quad 0 \leq y \leq 2$. The initial cone is given by $u_0(x, y) = 100\left(1 - \frac{r^*}{r}\right)$ for $r^* < r$ and $u_0(x, y) = 0$ otherwise, where

$$x_0 = 0.5, \quad y_0 = 1, \quad r = 0.25, \quad r^* = \sqrt{(x - x_0)^2 + (y - y_0)^2}$$

The motion of the cone described by (11) (see e.g. Fig. 1) is circular and therefore after one full rotation the exact sollution should be the same as $u_0(x, y)$.

# 3 MATLAB sollution

In this section a complete MATLAB program for the implementation of the algorithms discussed above is presented. The set of routines consists of:

- **RT_main.m** - main program;

- **RT_CD_matrix.m, RT_UW1_matrix.m, RT_UW2_matrix.m** - subroutines for generating the *coefficient matrix* of the system of linear equations has to be solved for CD, UW2 and UW2-schemes;
- **RT_CD_RHS.m, RT_UW1_RHS.m, RT_UW2_RHS.m** - subroutines for generating the *right-hand-side vector* of the system of linear equations has to be solved for CD, UW2 and UW2-schemes;
- **RT_init.m** - subroutine for calculating the vector of the *initial condition*;
- **RT_vecf.m, RT_u.m, RT_v.m** - subroutines for calculating the *wind field*.

```
% Rotational test problem
% Central differences, 1st and 2nd order upwind schemes
% Dirichlet b.c. everywhere
%
% ---------------- Input data -------------------
disp('Choose the type of discretization:')
disp('0 - central differences;')
disp('1 - 1st order upwind;')
disp('2 - 2-nd order upwind')

  scheme=input('Enter, please  ')
  nx=input('Choose the number of the grid points in X-direction  ');
  ny=input('Choose the number of the grid points in Y-direction  ');
  N_tau=input('Choose the number of the time steps  ');
  nA=nx*ny;                       % number of nodes;
  hx=2.0/(nx-1); hy=2.0/(ny-1);   % stepsize in x- and y-direction;
  T_rot=2*pi;    tau=T_rot/N_tau; % time stepsize;

% ---------------- Create the matrix A ----------
  A=sparse([]);
  if scheme==0, [Ah]=RT_CD_matrix(nx,ny,hx,hy,tau); end

  if scheme==1, [Ah]=RT_UW1_matrix(nx,ny,hx,hy,tau);end

  if scheme==2, [Ah]=RT_UW2_matrix(nx,ny,hx,hy,tau);end

% ---------------- Initial condition ------------
  [c_old]=RT_init(nx,ny,hx,hy);
```

```
  c_init_max=max(max(c_old))
  c_plot1=reshape(c_old,nx,ny);
  figure(4),clf
  mesh(c_plot1);
  title('Initial cone')
  print -depsc initial.eps
  figure(5),clf
  contour(c_plot1)
  title('Initial cone - contour')
  print -depsc in_contour.eps

% --------------- Start in time ----------------

  for itau=1:N_tau

% -- Compute the corresponding right-hand-side vector

    if scheme==0, [b]=RT_CD_RHS(nx,ny,hx,hy,tau,c_old); end

    if scheme==1, [b]=RT_UW1_RHS(nx,ny,hx,hy,tau,c_old);end

    if scheme==2, [b]=RT_UW2_RHS(nx,ny,hx,hy,tau,c_old);end

% -- Solve the system of equations

    c_new=Ah\b';

% -- Upgrade the right-hand-side

    for i=1:nA
        c_old(i)=c_new(i);
    end
  end
  c_1280_max=max(max(c_old))
  c_1280_min=min(min(c_old))

  c_plot=reshape(c_old,nx,ny);

  if scheme==0
     figure(1),clf;
     mesh(c_plot);
```

```
    title('Cone after full rotation')
    print -depsc CD_1280.eps
  end
  if scheme==1
    figure(2),clf;
    mesh(c_plot);
    title('Cone after full rotation')
    print -depsc UW1_1280.eps
  end
  if scheme==2
    figure(3),clf;
    mesh(c_plot);
    title('Cone after full rotation')
    print -depsc UW2_1280.eps
  end
%==============================================

% Generate 2D finite difference matrix for the rotational test problem
%
% Central differences scheme (CD) for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%----------------------------------------------
function [A]=RT_CD_matrix(nx,ny,hx,hy,tau)
n=nx*ny;
A=sparse([]);
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

    A(ind,ind)=1;

    if ind-ny>=1, A(ind,ind-ny)=-cu*tau/(4*hx);end

    if ind+ny<=n, A(ind,ind+ny)= cu*tau/(4*hx);end
```

```
    if l~=1,      A(ind,ind-1) =-cv*tau/(4*hy);end

    if l~=ny,     A(ind,ind+1) = cv*tau/(4*hy);end

  end
end
%==============================================

% Generate 2D finite difference matrix for the rotational test problem
%
% 1st order upwind scheme (UW1)for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%----------------------------------------------
function [A]=RT_UW1_matrix(nx,ny,hx,hy,tau)
n=nx*ny;
A=sparse([]);
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

                  A(ind,ind)   =1+abs(cu)*tau/(2*hx)+abs(cv)*tau/(2*hy);

    if ind-ny>=1, A(ind,ind-ny)=-(abs(cu)+cu)*tau/(4*hx);end

    if ind+ny<=n, A(ind,ind+ny)=-(abs(cu)-cu)*tau/(4*hx);end

    if l~=1,      A(ind,ind-1) =-(abs(cv)+cv)*tau/(4*hy);end

    if l~=ny,     A(ind,ind+1) =-(abs(cv)-cv)*tau/(4*hy);end

  end
end
%==============================================
```

```
% Generate 2D finite difference matrix for the rotational test problem
%
% 2nd order upwind scheme (UW2) for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%----------------------------------------------
function [A]=RT_UW2_matrix(nx,ny,hx,hy,tau)
n=nx*ny;
A=sparse([]);
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

                  A(ind,ind)   =1+abs(cu)*tau/(2*hx)+abs(cv)*tau/(2*hy);

    if ind-ny>=1, A(ind,ind-ny)=-(abs(cu)+cu)*tau/(4*hx);end

    if ind+ny<=n, A(ind,ind+ny)=-(abs(cu)-cu)*tau/(4*hx);end

    if l~=1,      A(ind,ind-1) =-(abs(cv)+cv)*tau/(4*hy);end

    if l~=ny,     A(ind,ind+1) =-(abs(cv)-cv)*tau/(4*hy);end

  end
end

%==============================================

% Generate right-hand-side vector for the rotational test problem
%
% Central differences scheme (CD) for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
```

```
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%-----------------------------------------------
function [rhs]=RT_CD_RHS(nx,ny,hx,hy,tau,c_old)
n=nx*ny;
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

                  rhs(ind)=c_old(ind);

    if ind-ny>=1, rhs(ind)=rhs(ind)+cu*tau/(4*hx)*c_old(ind-ny);end

    if ind+ny<=n, rhs(ind)=rhs(ind)-cu*tau/(4*hx)*c_old(ind+ny);end

    if l~=1,      rhs(ind)=rhs(ind)+cv*tau/(4*hy)*c_old(ind-1); end

    if l~=ny,     rhs(ind)=rhs(ind)-cv*tau/(4*hy)*c_old(ind+1); end

  end
end
%===============================================

% Generate right-hand-side vector for the rotational test problem
%
% 1st order upwind scheme (UW1) for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%-----------------------------------------------
function [b]=RT_UW1_RHS(nx,ny,hx,hy,tau,c_old)
n=nx*ny;
for k=1:nx
  for l=1:ny
```

```
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

                  b(ind)=(1-tau/(2*hx)*abs(cu)-tau/(2*hy)*abs(cv))*c_old(ind);

    if ind-ny>=1, b(ind)=b(ind)+(abs(cu)+cu)*tau/(4*hx)*c_old(ind-ny);end

    if ind+ny<=n, b(ind)=b(ind)+(abs(cu)-cu)*tau/(4*hx)*c_old(ind+ny);end

    if l~=1,      b(ind)=b(ind)+(abs(cv)+cv)*tau/(4*hy)*c_old(ind-1);end

    if l~=ny,     b(ind)=b(ind)+(abs(cv)-cv)*tau/(4*hy)*c_old(ind+1);end

  end
end
%=============================================

% Generate right-hand-side vector for the rotational test problem
%
% 2nd order upwind scheme (UW2) for space discretization;
%
%  du/dt + u(x,y)du/dx + v(x,y)du/dy = 0
%
% \Omega=[0,2]x[0,2]
% Zero Dirichlet b.c. on all boundaries
% nx, ny - number of gridpoints in x- and y-directions
% column-wise ordering of the nodepoints
%--------------------------------------------
function [b]=RT_UW2_RHS(nx,ny,hx,hy,tau,c_old)
n=nx*ny;
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l; x = k*hx; y = l*hy;
    [cu,su]=RT_vecf(1,x,y); [cv,sv]=RT_vecf(2,x,y);

    b(ind)=(1+tau/(2*hx)*abs(cu)+tau/(2*hy)*abs(cv))*c_old(ind);

    if ind-ny>=1, b(ind)=b(ind)-(abs(cu)-cu)*tau/(4*hx)*c_old(ind-ny);end

    if ind+ny<=n, b(ind)=b(ind)-(abs(cu)+cu)*tau/(4*hx)*c_old(ind+ny);end
```

```
    if l~=1,      b(ind)=b(ind)-(abs(cv)-cv)*tau/(4*hy)*c_old(ind-1);end

    if l~=ny,     b(ind)=b(ind)-(abs(cv)+cv)*tau/(4*hy)*c_old(ind+1);end

  end
end
%==============================================

% Initial condition for the 2D rotational test problem
%
%----------------------------------------------
function [c_init]=RT_init(nx,ny,hx,hy)
n=nx*ny;
x0=0.5; y0=1.0; r0=0.25;
for k=1:nx
  for l=1:ny
    ind=(k-1)*ny+l;
    c_init(ind)=0;
    x = k*hx; y = l*hy;
    r = sqrt((x-x0)^2+(y-y0)^2);

    if r<r0, c_init(ind)=100*(1-r/r0);end

  end
end
%==============================================

% 2D Rotational test problem
% Variable vector field: depending on the sign of
% the field in a given point (x,y)
%
function [val,sign]=RT_vecf(who,x,y)

if who==1, val=RT_u(x,y); else val=RT_v(x,y); end

if val<0, sign=-1; else sign=1; end
%==============================================

%  2D Rotational test problem
%
%  Variable vector field: u(x,y)
```

```
function u=RT_u(x,y)
u = (1-y);
%==============================================

%  2D Rotational test problem
%
%  Variable vector field: v(x,y)
function v=RT_v(x,y)
v = (x-1);
%==============================================
```

It is well seen that that the main objective of the code is efficiency and simplicity but not luxury of the input and output. A lot of input/output improvements can be done from the reader. The standart *matrix division* is used in order to find the solution of the system of linear equations. In order to improve the algorithm some iterative algorithms like *Conjugate Gradient* (for symmetric coefficient matrices as when CD-scheme is applied) or *BiCGStab* (for non symmetric coefficient matrices as when UW1- and UW2-schemes are applied) can be used.

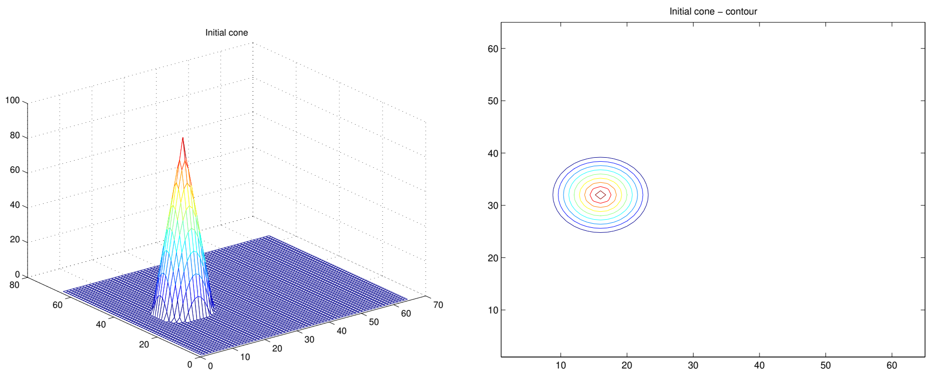
# 4 Results from the numerical tests

Results from some numerical tests performed are presented and discussed in this section. Let us consider two space discretizations: $(64 \times 64)$ and $(128 \times 128)$ gris squares. In all tests the time interval is divided in 1280 time steps. The initial cone and its contour is given on Fig. 1. Results about the accuracy of the three difference schemes are presented in Table 1.

Table 1: Accuracy of CD, UW1 and UW2 schemes

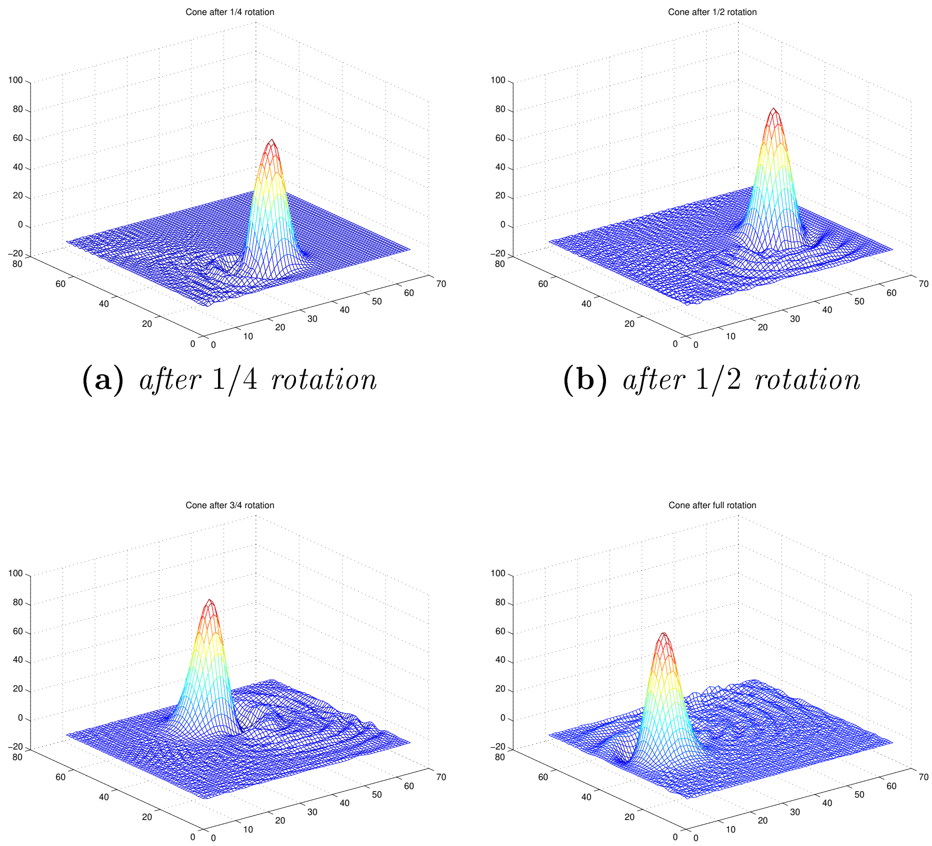
| Scheme | $n_x \times n_y$ | 1/4 rotation | | 1/2 rotation | | 3/4 rotation | | Full rotation | |
|---|---|---|---|---|---|---|---|---|---|
| | | $c_{min}$ | $c_{max}$ | $c_{min}$ | $c_{max}$ | $c_{min}$ | $c_{max}$ | $c_{min}$ | $c_{max}$ |
| CD | $64 \times 64$ | -5.97 | 85.97 | -10.09 | 87.67 | -9.28 | 85.70 | -12.69 | 82.04 |
| CD | $128 \times 128$ | -3.62 | 91.85 | -5.89 | 86.65 | -6.06 | 84.30 | -5.67 | 86.36 |
| UW1 | $64 \times 64$ | 0 | 41.06 | 0 | 41.05 | 0 | 32.14 | 0 | 15.24 |
| UW1 | $128 \times 128$ | 0 | 56.41 | 0 | 41.06 | 0 | 32.15 | 0 | 26.39 |
| UW2 | $64 \times 64$ | 0 | 86.41 | 0 | 86.93 | 0 | 84.76 | 0 | 81.75 |
| UW2 | $128 \times 128$ | 0 | 91.56 | 0 | 83.81 | 0 | 85.51 | 0 | 86.96 |

It is well senn from the first and the second row of Table 1 the negative values of the sollution which have no physical meaning. At the same time the height of the cone is kept well. Looking at the next two rows, three and four, one can see that there are not already negative concentrations but the height of the cone at the the end (after one full rotation) is not reasonable. The best situation can be seen when 2nd order upwind scheme is used - no negative concentration and good preserving of the height of the cone in the end of the time interval. The corresponding MATLAB plots can be seen in the following three figures.




**(a)** **(b)**

Figure 1: **(a)** *Cone in the beginning; height* $= 100$; **(b)** *Cone in the beginning - contour*



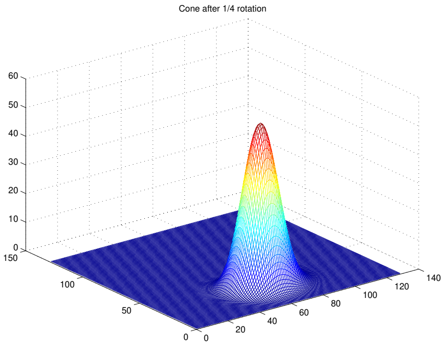

**(a)** *after* 1/4 *rotation*

**(b)** *after* 1/2 *rotation*

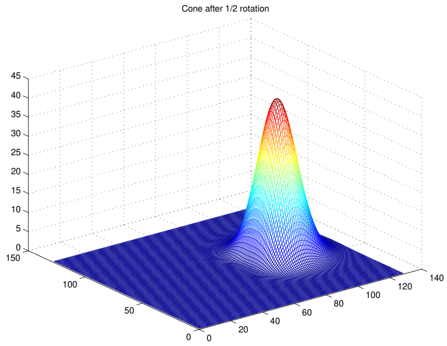
**(c)** *after* 3/4 *rotation*

**(d)** *after full rotation*

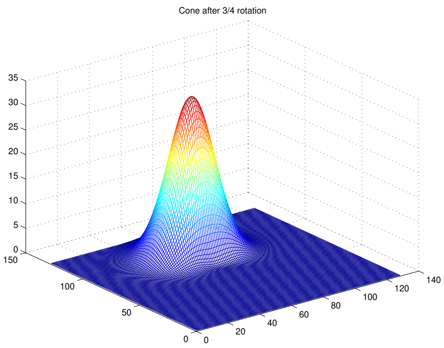
Figure 2: *Central diference scheme* ($n_x = n_y = 128$, $\tau = 2\pi/1280$)



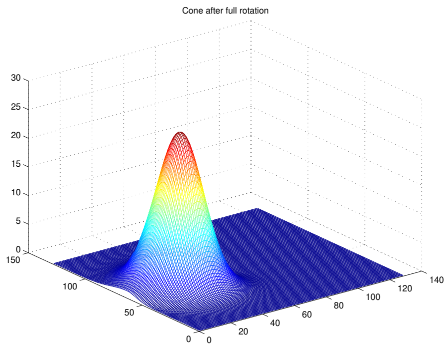

**(a)** *after* 1/4 *rotation*




**(b)** *after* 1/2 *rotation*

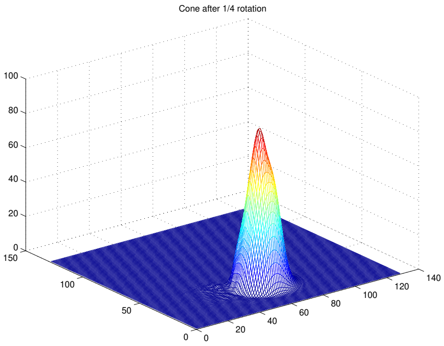



**(c)** *after* 3/4 *rotation*

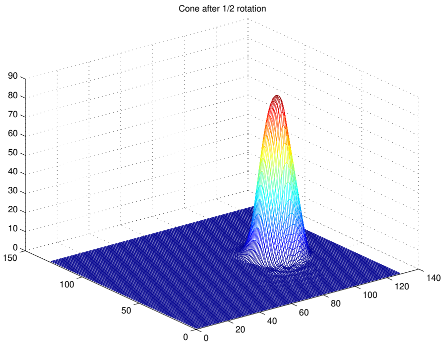



**(d)** *after full rotation*

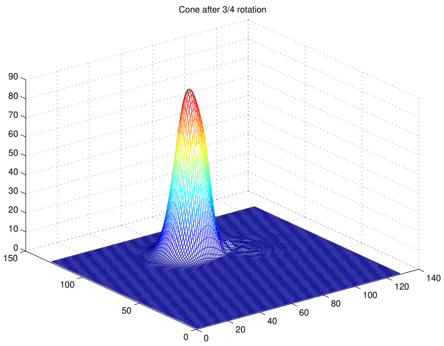
Figure 3: *1st order upwind scheme* ($n_x = n_y = 128$, $\tau = 2\pi/1280$)



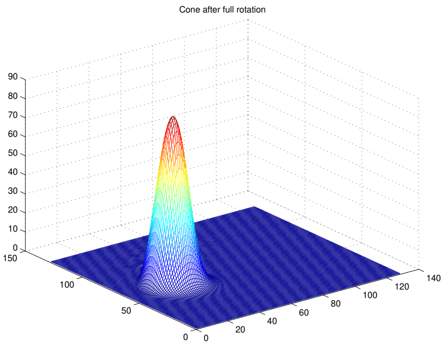

**(a)** *after* 1/4 *rotation*




**(b)** *after* 1/2 *rotation*




**(c)** *after* 3/4 *rotation*




**(d)** *after full rotation*

Figure 4: *2nd order upwind scheme* ($n_x = n_y = 128$, $\tau = 2\pi/1280$)

# References


[1] Crowley, W. P. (1968), *Numerical advection experiments*, Monthly Weather Review **96**, pp. 1-11

[2] Georgiev, K. (1996), *On an iterative solution of the convection-diffusion problems*, IMACS Ser. in Comp. and Appl. Math., **3**, pp. 80-89

[3] Georgiev, K., Zlatev, Z., (1999), *Parallel sparse matrix algorithms for air pollution models*, Parallel and Distributed Computing Practices, **2, No.4**, pp. 429-442

[4] Lirkov, I.; Margenov, S. (1993), *Circulant preconditioning of second order up-wind approximation of advection problems*, in: Scientific Computation and Mathematical Modelling, S. Markov (Ed.), DATECS Publ., Sofia, Bulgaria, pp. 73-76

[5] McRae, G. J., W. R., Goodin, W. R. and Seinfeld, J. H. (1984), *Numerical solution of the atmospheric diffusion equations for chemically reacting flows*, Journal of Computational Physics, **45**, pp. 1-42

[6] Marchuk, G. I. (1985), *Mathematical modelling for the problem of the environment*, North-Holland, Amsterdam

[7] *MATLAB Reference Guide*, The Mathworks Inc, 1996

[8] Molenkampf, C. R. (1968), *Accuracy of finite-difference methods applied to the advection equation*, Journal of Applied Meteorology, **7**, pp. 160-167

[9] Morton, K. W.; Mayers, D. F. (1994), *Numerical solution of partial differential equations*, Cambridge Univ. Press

[10] Pintchukov (1991), *On construction of monotone predictor-corrector schemes of arbitrary order of approximation*, Mathematical Modelling **3.9**, pp. 95-103 (in Russian)

[11] Quarteroni, A.; Sacco, R.; Saleri, F. (2000), *Numerical mathematics*, Springer-Verlag

[12] Zlatev, Z. (1995), *"Computer treatment of large air pollution models*, Kluwer Academic Publishers, Dordrecht-Boston-London