Interactive simulation of an ash cloud of the volcano Grímsvötn

1 MATHEMATICAL BACKGROUND

Simulating flows in the atmosphere, being part of CFD, is on of the research areas considered in the working group of Prof. Kröner. Transport of atmospheric species/like volcano ash are part of this topic. Air flows involve many different parameters, not all of them are relevant in any case. However, all of these flows are characterized by certain equations which depend on time and space. A physical basis of these equations are the conservation laws of mass, momentum and energy. Since we are interested in the time evolution of the species, the equations obviously have to include the time derivative. This results in partial differential equations (PDE) (cf. [Krö97]).
In the simplest model of the PDEs for atmosphere, a **advection-diffusion-equation** (the terms advection and diffusion will be explained shortly) in two space-dimensions is derived. The atmosphere is three-dimensional, which means that the ash movement is simulated at a fixed height level, parallel to the ground.
The ash is being carried by the air, while the air movement is induced by the imbalance of physical parameters, such as air temperature, density and pressure. The air movement can happen on small scale and on large scale.

- On small scale, the process is called **diffusion**. The motion is then caused by the molecular movement along the gradient of the quantity.
- **Advection** is a motion on a large scale. It occurs if a particle in the air is moved by the wind. The wind is the most important consequence

of the Sun's radiation and it can be calculated by equations as well. These are quite complicated, therefore we simplify our calculations by determining the velocity: We assume it to be constant and we want to know how it influences the air movement.

The molecular movement in the diffusion process is expressed by the divergence of a quantity's gradient. Thus the diffusion is $\nabla \cdot \nabla c = \Delta c$. Advection is the scalar product of the wind vector w and the gradient of the quantity we are considering, in our case the concentration of the ash, which we denote by $c(t, x)$. Thus the advection term is $w \cdot \nabla c$.
We are interested in the movement of volcano ash over hundreds of kilometers. That means that the movement on the molecular level is very small and happens slowly in comparison with the wind. Therefore we can leave out the molecular processes with good approximation. But the general air motion usually differs locally. These variations have different reasons, in particular there is one effect that occurs everywhere in the atmosphere: the so-called micro-turbulences. They can be pictured as small vortices and are responsible for air movement on a much smaller scale than the general wind system. Here, a simplification for our model takes place: Micro-turbulent effects on very large scales act similarly to diffusion effects on quite small scales (for example our pot of water). Therefore we take diffusion into account, and as these micro-turbulent effects can be larger or smaller depending on the circumstances, the user can choose how strong this effect will be, this is realized by a constant number ε with which the diffusion term is multiplied.
The PDE will be presented below. Apart from the advection and diffusion terms there are two further important mathematical terms:

- The first is the time derivative of the concentration $\partial_t c$. It describes how the other effects make the concentration change in time.
- The second modelizes the volcano itself, its eruption and the ash thrown into the atmosphere. This is realized by a function that is growing quickly in the beginning and abiding slowly till vanishing almost completely. As the volcano eruption is a local event, this function depends also on the space coordinate $\vec{x}$. It is called $v(t, \vec{x})$. We assume that we know when and how much ash is thrown into the atmosphere.

With all four parts, the PDE looks like follows:

$$\partial_t c + \nabla \cdot (v\, c) - \varepsilon \Delta c = v(t, \vec{x}),$$

We solve this equation using a numerical scheme which consists of a **Discontinuous Galerkin (DG)** method for the space discretization and a

Runge-Kutta method for the time discretization. The DG method follows the principle of a Finite Volume scheme: A computational grid divides the model space (encompassing the Europe map in our case) into parts, the so-called simplices, entities or grid cells. On each of these entities the values are computed by calculating the flow to or from the neighbour cells. In Finite Volume Schemes, these values consist of one constant value per entity, that is, the solution is **approximated** by a function that is piecewise constant, i.e. constant on each entity. The DG methods can be of higher order, which means that the function is now no longer constant, but can be, for example, a linear or quadratic function, or a polynomial of higher order. The function must be calculated in such a manner that it approaches the exact solution when the grid is being refined.
As the function in one entity differs from the ones in its neighbours, there are discontinuities at the edges between them (hence "Discontinuous" Galerkin scheme). This needs to be considerated when calculating the flux from one cell to another. Then follows the main step of this scheme: calculating values for the diffusion and advection term. The source equation is determined in the beginning and can easily be evaluated. As soon as we have obtained all those values, our equation is reduced to ordinary differential equations (ODE) that depends on time. ODEs can be solved numerically as well, there are many well-established methods to achieve this, differing in accuracy and computation time. We take Runge-Kutta methods of the same order as in the DG scheme.
Keep in mind that CFD uses approximations. For most PDE of this kind, the existence of a solution cannot even be proven, let alone computed exactly. However, in practice, these approximations often give good results and are used for example for weather forecast and engineering, but also flood prediction etc., in other words they are used in everything that involves fluid motion.

2 DOWNLOAD AND INSTALLATION

The installation process of this application is very simple. It contains two steps. First, you have to download the image and burn it on a blank CD. In the second step, you have to reboot you system from the CD. The interactive simulation will start immediately.

Download and burning

Download the live-cd image

dune.mathematik.uni-freiburg.de/dune-ash/image.iso

it has a size of about 200 MB, so it might take some time to download this file. Use your favorite CD burn software to burn the image on a blank CD. Depended on your operating system different burning programs can be used.

Run the program

Reboot you PC from the just burned CD. Depending on you system you have to change the boot order in the BIOS settings of you PC. After booting from CD the program will start immediately.
When closing the program, the user is asked whether he wants to reboot the PC to get back to the operating system or he wants to restart the application.

This live-cd is also able to run in a virtual environment (e.g. virtual machine emulator). In this case, the full performance is not gained.

3 PROGRAM USAGE

After booting from the CD the user will see the graphical interface. It provides input and output in an intuitive manner. The whole program consists of three parts: the graphical user interface and two background parts in which the calculations are realized. The graphical user interface gathers and stores information needed in the four steps of a program cycle.
One program cycle always starts with the positioning of the volcano. By touching the screen, clicking the mouse on the desired position the position of the erupting volcano can be chosen. If the desired position is reached, one can move on to the next step clicking the 'next' button.
In the next step, the user can draw some lines, which represent streamlines of the wind. Based on these lines a wind field is computed using the first background part.
Clicking once more on the 'next' button the computations of the wind field are performed, which might take a few seconds. After that the wind field is visualized on the screen, illustrated by blue arrows. Their shade of blue indicates the speed: They range from darker blue (high speed) to paler blue

(lower speed).
After the computation of the wind field ends, the user can decide how much ash shall be dispersed by so-called micro-turbulences: There is a slider where the marked position is interpreted as the strength of the dispersion. If the user has chosen the strength of the dispersion and moves to the next step, the second background part starts its work. It computes the actual solution of the equation in the background. This may take some time and while the parallel computation is performed. The graphical user interface shows the progress of the ash on the Europe map. The progress of the simulation can be seen in the slider in the lower right. The slider also shows the actual shown frame, in this simulation 100 frames are generated using the second background part. This frames can be used to start an animation of the ash spreading over Europe. The animation is started/paused or stopped using the 'start/pause' or 'stop' buttons.
In the third and fourth step of one program cycle, the user may switch between different aspects of the velocity field with the 'arrow' button and also the grid cells can be made visible with the 'grid' button.
Whenever informations are needed the i-button gives a help screen for the program. Further information on each aspect of this simulation can be accessed using the 'Learn more about ...' links in the right hand panel, marked in blue.

4 ADDITIONAL INFORMATIONS

Additional informations, e.g. on the volcano or the mathematics, can be obtain in the interactive simulation using the i-button.

References

[Krö97] **D. Kröner**
Numerical Schemes for Conservation Laws
Wiley & Teubner, Stuttgart, 1997

[De10] **A. Dedner, R. Klöfkorn, M. Nolte, M. Ohlberger** *A generic interface for parallel and adaptive scientific computing: Abstraction principles and the DUNE-FEM module* Computing 90, No. 3-4, pp. 165-196 (2010)

[No11] **M. Nolte** *Efficient Numerical Approximation of the Effective Hamiltonian*, Doctoral Dissertation, University of Freiburg (2011)