3D Modelling of Heat Flow in Sedimentary Basins - the Impact of Rifted Basement Topography

Stephan Zeipel

Abstract

Oil-Companies wish to explore hydrocarbon deposits in the North Sea at the lowest possible cost, therefore it is necessary to know the thermal history of the area. In this project a three dimensional heat flow model has been set up and parallelised on the Cray T3D. As an extension to a former project by D. Carstens, it is now possible to calculate water as a boundary condition. The program is written in C using MPI. This model has been verified with previous case studies.

Contents

Chapter 1

Introduction

The aim of this project is to set up a 3-dimensional model which can describe the heat flow in sedimentary basins in the north sea. It is part of the Summer Scholarship Programme (SSP) 1998 of the Edinburgh Parallel Computing Centre (EPCC). It is an extension of a former project done by D. Carstens. [1]

Oil industry drilling as well as scientific drilling projects have allowed people to gain insights into the geological structure of the earth. In regions unaffected by volcanic intrusions and tectonic deformations one is likely to find a formation made up out of well-defined, sedimentary layers. The 'core' extracted by a test drilling can be studied to obtain information of material properties and explicitly thermal properties [2] like the thermal conductivity of the different types of rocks being present.

Taking these empirical results into account it is possible to set up a model for the heat flow through these sedimentary layers and the earth crust. Determining the thermal history of individual formations, and especially layers containing a mixture of sedimentary rock and organic material, is of great importance for the oil industry. The above mentioned type of rock is often referred to as *source rock* and it is the main originator of hydrocarbons. If the source rock is heated to temperatures ranging from $80°C$ up to approx. $160°C$ for a long enough period of time a conversion of the primary organic material to hydrocarbons, oil and gas, can take place. For a correct assessment of the status of this so-called maturation process an accurate simulation of the heating process is required. Subsequently predictions of possible locations of hydrocarbon deposits can then be made.

The aim of this project is to set up a model, that can simulate this heating process.

Chapter 2

Background

2.1 Geological Background

The model is concerned with the lithosphere of the earth, which ranges to a depth up to 100km and contains the earth's crust and the uppermost part of the mantle. This implies that we will deal with matter having solid and rigid properties and the simulation will only take heat conduction into account.

The situation would be different in the asthenosphere (100 - 350km depth). Here the pressure and temperature have increased to such an extent that matter has become soft and plastic [3]. Under these circumstances it would be unavoidable to include heat convection, i.e. heat transfer via material currents, into the model. The transport of heat may also be governed by radiation [4]. But this process may be neglected for the temperature range one expects.

2.2 Physical Background

2.2.1 Three Forms of Heat Transport

It is known that as a substance is heated, its temperature increases, thus heat is a kind of energy. This was demonstrated by the British scientist J. P. Joule (1818-1889) in a famous experiment. Heat energy travels from one place to another in several ways. Heat always tends to flow from a region of high temperature to a region of low temperature. There are three basic methods by which heat transfer may take place.

If one puts one end of a metal bar into fire, you will eventually notice that the other end of the bar becomes warmer. Heat travelled through the metal. This form of heat transfer is called **conduction**. The molecules of a hot substance become more energetic and pass on some of their energy to adjacent molecules, which in turn pass some of their energy to the next molecules, and so on.

In a fluid, such as a gas or a liquid, heat is transferred by a process called **convection**. The fluid in contact with a heat source expands and becomes less dense than the cold fluid surrounding it, and therefore rises. The cold fluid moves in to take the place and in turn expands and rises. Thus, heat is transferred by the movement of the warm fluid in a so called convection current. If the fluid moves just because of the density difference, the mechanism is called **free convection**. If one uses e.g. a pump to transport the warmed fluid to another location, it is called **forced convection**.

Heat reaches us from the sun, but there is no substance in space to conduct or convect its heat to us. The heat energy reaches us by **radiation**. In this case, heat is just another form of electromagnetic radiation. [5]

Due to the solid structure of the earth's crust, the examined model takes only heat conduction into account.

2.2.2 Heat Conduction

Imagine a wall of thickness b and area A both sides are kept on constant, but different temperatures T_1 and T_2, $T_1 > T_2$. Fourier's law is valid and the amount of heat, that flows through the wall is:

$$Q = \frac{k}{b} A \left(T_2 - T_1\right) \tag{2.1}$$

With k the *thermal conductivity*, a property of the substance of which the wall consists. The dimension of k is: J/msK

The amount of heat penetrating a unit area of the surface per unit time is called the *heat flux, q*. For this the following equation is valid:

$$q = \frac{Q}{A} = \frac{k}{b} \left(T_2 - T_1\right) \tag{2.2}$$

If the thermal conductivity is constant, the temperature will decrease linearly from the value T_1 to the value T_2. However, this linear relationship does not exist in unsteady heating or cooling process and generally heat flux varies locally and in time. Equation (2.2) may then be applied only to parts of the body of infinitely small dimensions. Therefore the temperature difference $(T_2 - T_1)$ in equation (2.2) is to be replaced by the differential ∂T; the place of the wall thickness b is taken by the differential ∂n, where n is the line normal to the plane of the surface element dA. In this way, for the rate of heat flow dQ through the surface dA, the equation

$$dQ = -kdA\frac{\partial T}{\partial n}$$

is obtained. The minus sign in this equation indicates that the heat flow proceeds in the direction of the temperature drop, that is, in the negative direction of the temperature gradient $\frac{\partial T}{\partial n}$. For the heat flux q_n through the differential element, the equation

$$q_n = -k\frac{\partial T}{\partial n} \tag{2.3}$$

is obtained. Equation (2.3) is the mathematical expression of the basic principle of heat conduction, namely Fourier's Law which states that heat is transferred is proportional to the gradient of the temperature.

For a solid described by a Cartesian co-ordinate system, we can then write the heat fluxes in x, y and z direction respectively as follows:

$$\begin{aligned} q_x &= -k\frac{\partial T}{\partial x} \\ q_y &= -k\frac{\partial T}{\partial y} \\ q_z &= -k\frac{\partial T}{\partial z} \end{aligned} \tag{2.4}$$

2.2.3 Governing Equations of Heat Conduction

To find the equation for heat conduction in an isotropic material we consider a 2D element as in figure (2.1).

If the heat flowing in the direction of the x and y axes per unit length and in a unit time is denoted by q_x and q_y respectively, the difference between outflow and inflow for an element of size $dxdy$ is given as:

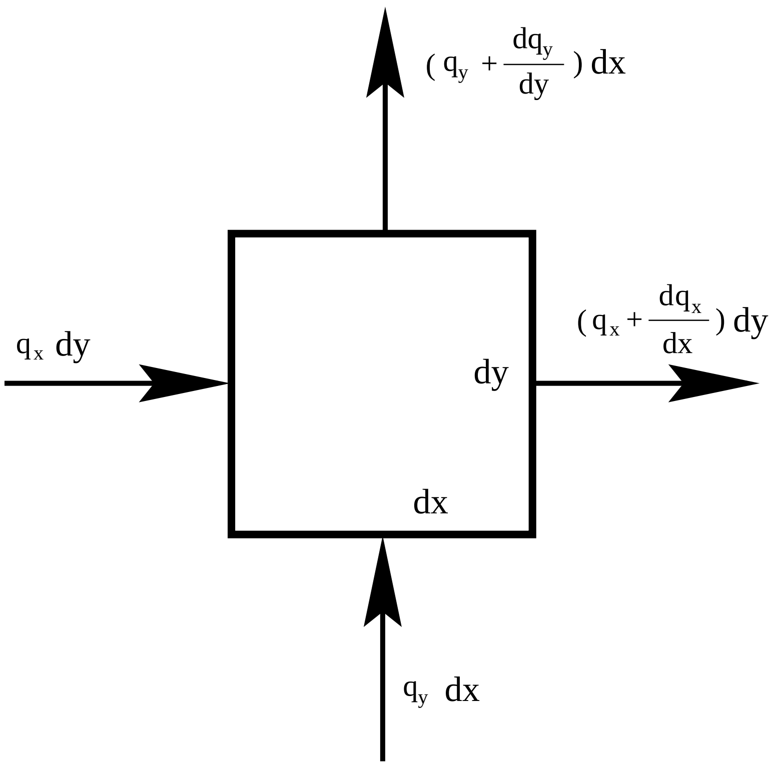


Figure 2.1: Heat Flow in a 2D element

$$dy\left(q_x + \frac{\partial q_x}{\partial x}dx - q_x\right) + dx\left(q_y + \frac{\partial q_y}{\partial y}dy - q_y\right)$$

For conservation of heat, this quantity must be equal to the sum of the heat generated in the element in unit time, say, $Qdxdy$ and the heat gained in a unit time due to the temperature change, namely, $-\rho c\frac{\partial T}{\partial t}dxdy$ (where c is the specific heat, ρ is the density and $T(x,y,t)$ is the temperature distribution). Clearly, this requirement of equality leads to the differential relationship:

$$\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} - Q + \rho c\frac{\partial T}{\partial t} = 0 \tag{2.5}$$

Substitution of equation(2.4) into equation (2.5) produces a higher order differential equation in a single dependant variable:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + Q - \rho c\frac{\partial T}{\partial t} = 0 \tag{2.6}$$

In a similar way the three dimensional equation of heat conduction can be obtained:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right)+Q-\rho c\frac{\partial T}{\partial t}=0 \tag{2.7}$$

It may also be written in vector notation as [5]:

$$\nabla(k\nabla T)+Q=\rho c\frac{\partial T}{\partial t} \tag{2.8}$$

In a steady state approach for the solution the differential $\frac{\partial T}{\partial t}$ is zero. Therefore we obtain the following equation:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right)=-Q \tag{2.9}$$

After applying the product rule for the derivative, this equation can also be written as:

$$\frac{\partial k}{\partial x}\frac{\partial T}{\partial x}+\frac{\partial k}{\partial y}\frac{\partial T}{\partial y}+\frac{\partial k}{\partial z}\frac{\partial T}{\partial z}+k\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right)=-Q \tag{2.10}$$

2.3 Initial Conditions

In order to solve the partial differential equation of conduction, one needs to specify the initial temperature for the interior nodes and conditions for the boundary nodes. In this model the initial temperature for the interior nodes is set to a constant value, which can be specified by the user. It would be possible to implement another initial temperature distribution for some special problems. One could then achieve a faster convergence, but it would not affect the solution.

2.4 Boundary Conditions

There are two typical boundary conditions involved. In the first condition, the values of temperature at the boundary are specified. These values may be constant (for steady state and transient solution) or be allowed to vary with time (for the transient solution only). A boundary condition of this form is frequently referred to as a *Dirichlet* or essential boundary condition.

In the second condition the values of heat flow in the direction n normal to the boundary are prescribed as $\overline{q}(x, y, z, t)$. Then we can write:

$$-k\frac{\partial T}{\partial n} = \overline{q} \qquad\qquad \textit{on boundary} \tag{2.11}$$

This type of boundary condition is often called a *Neumann* or natural boundary condition.

Chapter 3

The Model

3.1 Discretisation of the Equation for heat conduction

To solve the equation numerically, it is necessary to transform the continuous problem to a finite difference problem. This is done by calculating the solution not for every point in the material, but only for discrete points.

In the introduced model the number of grid points can be free chosen in each direction by the user, as well as the spacing between the points. One obtains then a discretised model as shown in figure (3.1).

The partial derivatives are approximated in the following way:

$$\frac{\partial k}{\partial x} = \frac{k_{i+1,j,k} - k_{i-1,j,k}}{2h_x} \tag{3.1}$$

$$\frac{\partial T}{\partial x} = \frac{T_{i+1,j,k} - T_{i-1,j,k}}{2h_x} \tag{3.2}$$

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i+1,j,k} + T_{i-1,j,k} - 2T_{i,j,k}}{h_x^2} \tag{3.3}$$

h_x is the spacing between the grid nodes. The approximations for the other dimensions can be obtained in a similar way.

Substituting these approximations into equation (2.10) and solving for $T_{i,j,k}$ we get the final equation for calculating the steady state solution for the interior nodes:

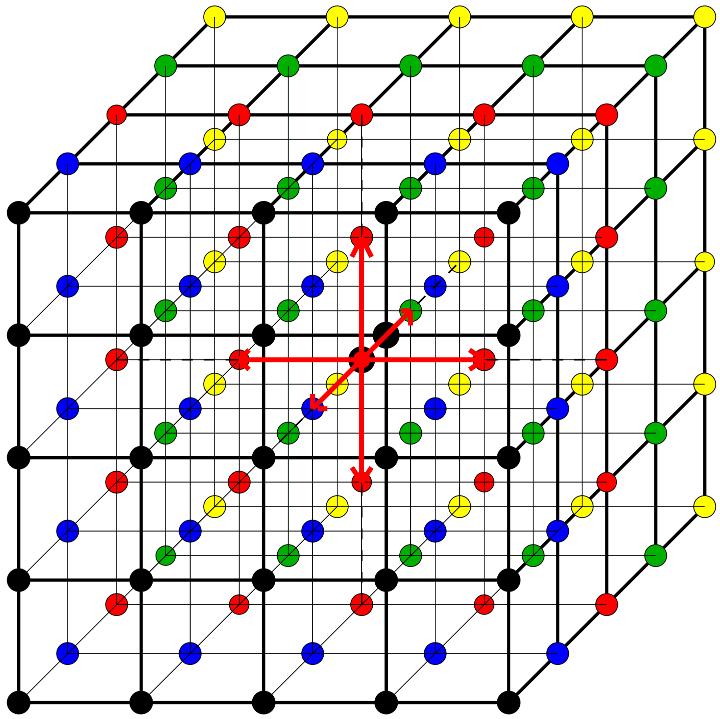


Figure 3.1: Discretised model

$$
\begin{aligned}
T_{i,j,k} \quad = \quad & \frac{h_x^2 h_y^2 h_z^2}{2k_{i,j,k}\left(h_y^2 h_z^2 + h_x^2 h_z^2 + h_x^2 h_y^2\right)}[Q \\
& + \frac{(k_{i+1,j,k} - k_{i-1,j,k})(T_{i+1,j,k} - T_{i-1,j,k})}{4h_x^2} \\
& + \frac{(k_{i,j+1,k} - k_{i,j-1,k})(T_{i,j+1,k} - T_{i,j-1,k})}{4h_y^2} \\
& + \frac{(k_{i,j,k+1} - k_{i,j,k-1})(T_{i,j,k+1} - T_{i,j,k-1})}{4h_z^2} \\
& + \frac{k(T_{i+1,j,k} - T_{i-1,j,k})}{h_x^2} \\
& + \frac{k(T_{i,j+1,k} - T_{i,j-1,k})}{h_y^2} \\
& + \frac{k(T_{i,j,k+1} - T_{i,j,k-1})}{h_z^2}]
\end{aligned} \tag{3.4}
$$

3.2 Discretisation for Boundaries

In this program it is possible to specify either a constant temperature as boundary condition, or a constant heat flux. One can specify a different constant value for each side of the model. However, for the bottom side of the model (x, y-plane, $z = 0$) it is possible to specify a heat flux distribution by naming a file in the input file. If the heat flux boundary condition is set, a temperature for the boundary nodes is calculated. The discretisation for *Fourier's Law* (equation (2.3)) in x-direction and for $i = 0$ is:

$$q_x = -k_{i,j,k} \frac{T_{i+1,j,k} - T_{i,j,k}}{h_x} \tag{3.5}$$

Solving the above equation for $T_{i,j,k}$ we get:

$$T_{i,j,k} = \frac{q_x h_x}{k_{i,j,k}} + T_{i+1,j,k} \tag{3.6}$$

With this one can update the temperature on the boundary nodes and then calculate the new temperatures for the interior nodes as usual.

3.3 Convergence

For the iteration the *Jacobi* Algorithm and the equation (3.4) are used. In the Jacobi Algorithm every grid point is updated with data from the previous iteration. One could use the *Gauss-Seidel* Algorithm instead. In that algorithm the latest calculation data is used to update the other nodes. This results in a faster convergence and I would implement this algorithm in the next version of the program.

The iteration process will be continued, until the solution has converged against a certain, user specified ϵ, or until a maximum number of iterations is reached. Convergence is reached, if the difference of the temperature of two time steps is less then ϵ for each node ($\Delta T = T^{n+1} - T^n < \epsilon$). This is tested every 10th iteration.

Chapter 4

The Program: A Users Guide

The basic programming language for this model is C. To solve large problems within a reasonable time it is necessary to parallelise the code. For parallelisation the MPI (Message Passing Interface) library was used. It makes the program very variable. It is portable to other systems and the program can even be run on workstation clusters connected via an Ethernet network. Data is send with the help of messages from one process to an other. [6]

4.1 Programming Techniques

For the description of the model a cartesian coordinate system is used, as it is easy to describe rectangular geometries in this. The axes of the model are seen in figure (4.1). The numbers of the elements of the x, y and z-axes run from 0 to $l-1$, $m-1$ and $n-1$, respectively. The coordinates in the in- and output-files are numbered from 1 to l, m and n, with l, m and n as the maximum number of elements in each direction.

To parallelise the code, the cube is divided into layers and each of these layers is calculated by one process. Each process needs data from the neighbouring processes for the calculation. This data is transferred between the processes by MPI and stored in a *halo*. Partitioning in layers has the advantage, that every number of processes is possible.

For each process two arrays of the size $l \cdot m \cdot (n+2)$ are allocated. The two extra layers in z-direction are used for the halo.

One step of calculation works as follows: First the boundary temperatures are updated, in case they are heat flux type. No data from the neighbour processes is necessary for that. In a second step the required data from

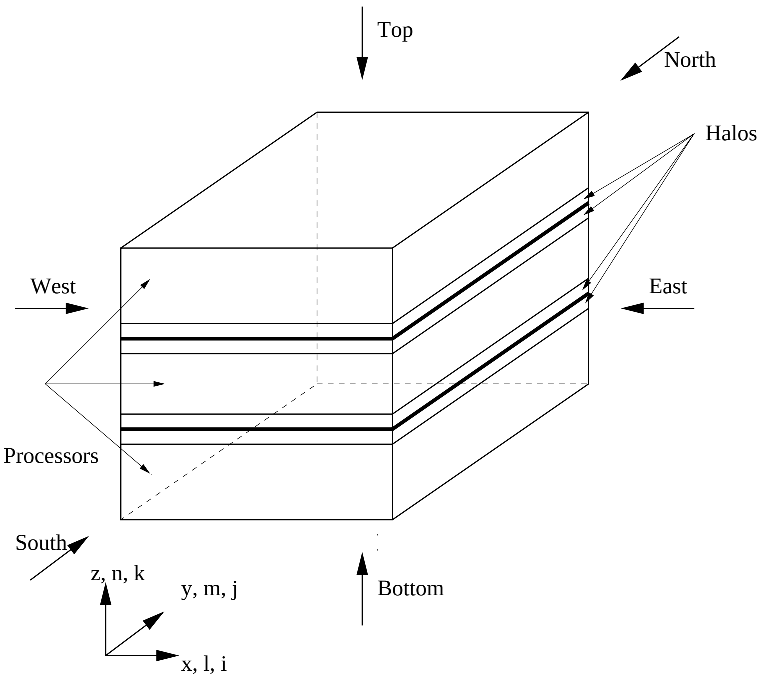


Figure 4.1: Layered structure of the model

the processes is copied to the halos of the other processes. Therefore a non blocking communication is used. Next the new temperatures for each interior node are calculated and stored in the second array. Every 10th iteration the convergence is checked, by subtracting the old temperature from the new one (see chapter 3.3). In a last step the first array is updated by copying the data from the second array to the first. Finally a **MPI_Barrier** call is made, in order to synchronise the processes.

This iteration step is repeated until the system has converged, or a certain number of steps is reached. The maximum number of steps can be specified by the user.

4.2 Data In- and Output

4.2.1 Specification of the input file

All the information necessary for the calculation is provided in the input file. A typical input file looks like this:

```
Inputfile for calculation of heat conduction!
```

```
Number of elements in x direction: 100
Number of elements in y direction: 100
Number of elements in z direction: 64

Spacing in x direction: 1.000000
Spacing in y direction: 1.000000
Spacing in z direction: 1.000000

Epsilon: 0.010000

Coefficient of heat conduction (value or filename):
./inputdata/k_coefficients

Initial temperature for interior nodes: 323.000000

Water (set to zero for no water or specify filename
for waterdistribution): ./inputdata/waterdistribution

Heat generation (set to zero or specify filename
for heat generation): ./inputdata/heatgeneration


Boundary conditions!
To specify heat flux, set temperature to zero!

Surface bottom!
Temperature: 373.000000
Heat flux (value or filename): 0

Surface top!
Temperature: 273.000000
Heat flux: 0.000000

Surface north!
Temperature: 0.000000
Heat flux: 0.000000

Surface south!
Temperature: 0.000000
Heat flux: 0.000000
```

```
Surface west!
Temperature: 0.000000
Heat flux: 0.000000

Surface east!
Temperature: 0.000000
Heat flux: 0.000000

Outputfilename: ./results/heatgen_100x100x64
Output format: AVS
```

First the number of elements in each direction is specified. The number of elements in the z-direction must be dividable by the number of processors one is going to use, without rest.

Then the spacing between the nodes in each direction is set. It may be different for each direction. The program is calculating in SI-units, therefore the spacing is in metres.

Next the convergence criterion is set. (see chapter 3.3)

In the next four lines some parameters are set for the material. If one wishes a constant coefficient of heat conduction, one just has to specify a constant value here. However, it is possible to have any distribution of of heat conduction coefficients (e.g. for more than one material). For the desired distribution one then has to give a relative path and name for the file to be read. The file must be a simple ASCII file, in which each line represents one node of the grid. The first three values in each line are integer number for the x, y and z-coordinates. The next value should be a float number for the coefficient of heat conduction. It is important, that the file contains as much lines, as there are nodes in the model to be calculated!

After that the initial temperature is set. (see chapter 2.3)

For specifying a water boundary condition, one has to name a file here. The presence of water is represented by a integer value of 1 in the file, the nodes, which are material have to be set to integer 0. The structure of the file is the same than for the heat coefficient file. If one does not want to simulate the water boundary condition at all, the value is to be set to 0 instead of specifying a file name.

It is also possible to simulate heat generation inside the material. For that purpose it is necessary to specify the next file. In this file also the first three values represent the coordinates and the fourth value gives the rate of heat generation inside the material for the specified node. The same rules, as for the heat conduction coefficient file have to be obeyed.

Then the boundary temperatures and boundary heat fluxes are specified, respectively. One can either give a fixed temperature or a fixed heat flux. For the latter the temperature value has to be set to 0. For the bottom it is possible to specify a non-uniform heat flux distribution. One therefore has to give an input file, in which the first two values of a line represent the coordinates on the bottom and the third value gives the heat flux for the node.

In the last two lines the output path and filename are specified, as well as the output format.

4.2.2 The Two Output Formats

Two output formats are possible. Either an output for a simple java interface, in this the three coordinates and the value for the temperature are given for each node in a separate line. Or, for a more sophisticated output, an AVS (*Advanced Visualisation System*) output file. For the orthogonal lattice grid AVS does not need to know the coordinates, as long as the values are in the row-minor ordering. Therefore these values are not given in this output format, in order to save disk space. In the first place in a line always the temperature for a node is given. In the next places the values for heat conduction, water distribution and heat generation for each node are given, but only if a matching input file was read.

4.3 Speedup curve

The speedup for a linear temperature distribution for problem size of 20x20x128 is shown in figure (4.2)

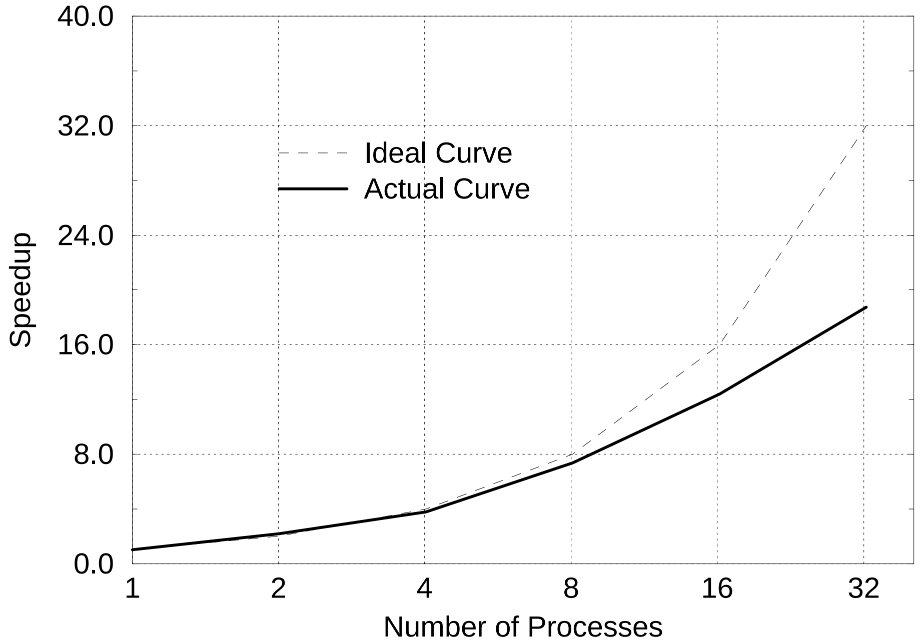


Figure 4.2: Speedup curve

Chapter 5

Results

5.1 Simple Case Studies

In this section the results of some simple case studies will be discussed. The size of the cube is 40x40x40 elements for all examples and the spacing is $1m$ in each direction.

The first thing to be tested was a linear temperature distribution. To achieve that, the bottom of the model was held on fixed temperature of $373K$ and the top on a temperature of $273K$. The other boundaries were set to zero heat flux. One would then expect a linear temperature distribution. This is shown in figure (5.1)

A similar result can be obtained by applying a constant heat flux on the bottom of the model. In the next case (figure (5.2)) the heat flux was $10W/m^2$.

Figure (5.3) shows the temperature distribution for two different materials. On the left are the different heat conduction coefficients. The coefficient of heat conduction in the middle is $5W/m$ and the one of the surrounding material is $1W/m$. Due to the higher conductivity the temperature in the middle is higher than on the outer parts of the cube.

The next example (figure (5.4)) shows a heat generation on the bottom of $5W/m^2$ and a heat generation in the inside of the cube of $20W/m^3$. The block with heat generation in the inside is shown on the left.

In the last figure (5.5) the implementation of the water boundary is shown, combined with a heat generation in the inside. The water is on the top of the cube.

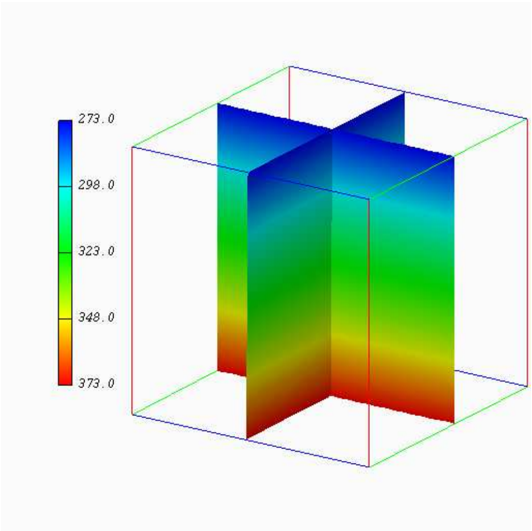


Figure 5.1: Linear temperature distribution in a cube

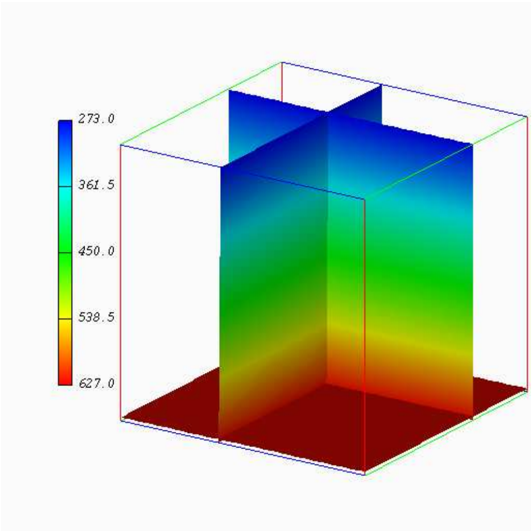


Figure 5.2: Constant heat flux on the bottom of the cube

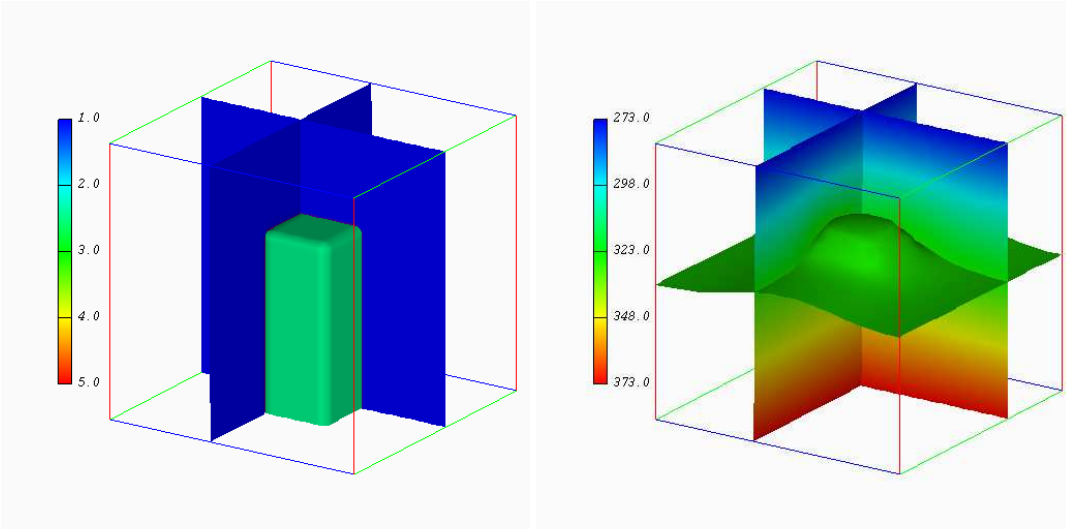


Figure 5.3: Two different heat conduction coefficients

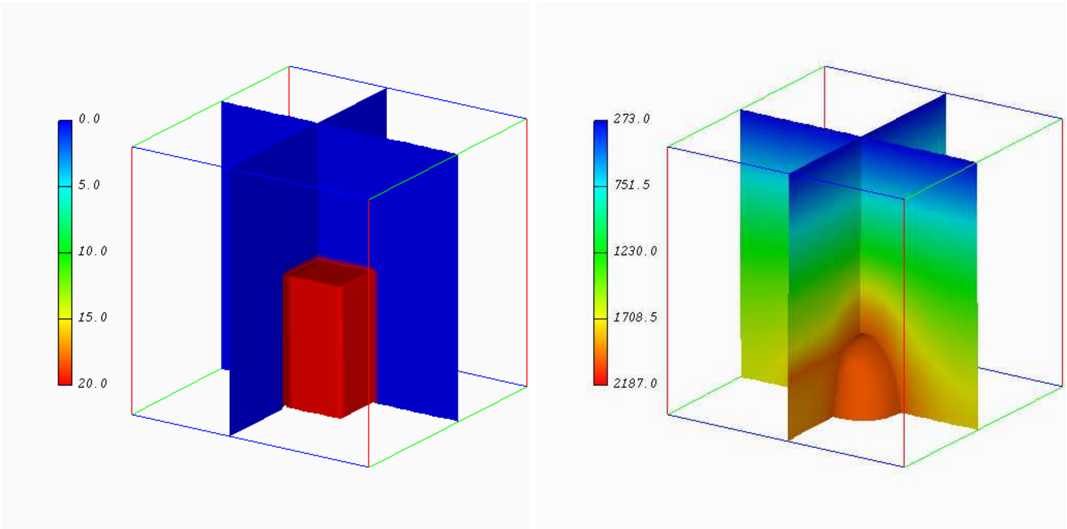


Figure 5.4: Heat generation on the bottom and inside the material

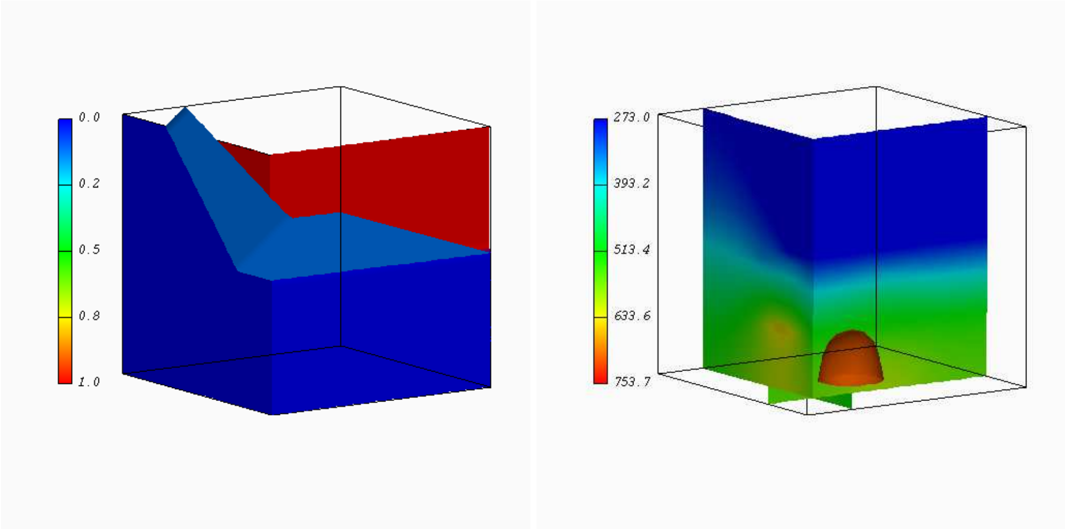


Figure 5.5: Water and heat generation in combination

5.2 Real Case Studies

Unfortunately there was no real data available on time for some real geological case studies.

Chapter 6

Conclusion

In this project a model for simulating the heat transfer in sedimentary basins has been set up successfully. It is possible to simulate different coefficients of heat conduction as well as heat generation inside the sediments and water above. Both, the Dirichlet and the Neumann boundary condition can be modelled.

It is possible to simulate any rectangular parallelepiped geometry with different spacings between the nodes in each direction.

The model has been verified with some simple case studies. For the future it is easily possible to set up some more complicated models by specifying some input files with real geological data. For that purpose it would be nice to program a tool to make it easier to generate input files for more complicated geometries.

Bibliography

[1] Dirk Carstens: *3D Modelling of Heat Flow in Sedimentary Basins* EPCC-SS92-02, 1997

[2] C. C. Clauser and Ernst Huenges: *Thermal Conductivity of Rocks and Minerals*, 105pp. American Geophysical Union, Reference Shelf 3 - Rock Physics and Phase Relations, 1995

[3] W. K. Hamblin and E. H. Christiansen: *Earth Dynamic Systems.* Prentice Hall, 1995

[4] H. U. Fuchs: *The Dynamics of Heat*, 301pp. Springer Verlag, 1996

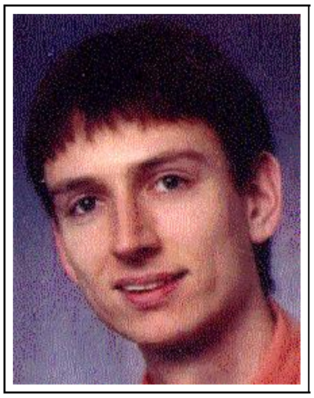
[5] Hou-Cheng Huang and Asif S. Usmani: *Finite Element Analysis for Heat Transfer* Springer-Verlag, London, Berlin, Heidelberg, New York, 1994

[6] *MPI 1.1 Standard* http://www.mpi-forum.org/docs/mpi-11-html/mpi-report.html

[7] Neil MacDonald, Elspeth Minty, Joel Malard, Tim Harding, Simon Brown, Mario Antonioletti: *MPI Programming on the Cray T3D, A Two Day Course on MPI Usage* epcc Version 1.8

[8] Peter MacCallum, Jean-Christophe Desplat, John Fisher, Anna Hondroukakis, Elspeth Minty: *Scientific Visualisation: a Practical Introduction, A two-day course* epcc Version 3.2

[9] André Willms: *C-Programmierung, Programmiersprache, Programmiertechik, Datenorganisation* Addison-Wesley Longman Verlag GmbH, 1996



Biography for Stephan Zeipel:
I was born in Göttingen, Germany. In 1993 I started studying Mechanical Engineering in Hanover, Germany and will finish with my Diploma in March 1999.

Acknowledgements:

I would like to thank my supervisors for all their support during my project. Special thanks to Elson Mourão, who helped me to solve a major problem with my program and Jens Hoerner, who always was able to give me some good advice.

The supervisors for this project were:

Jon Turner, Dept of Geology and Geophysics, University of Edinburgh

Jean-Christophe Desplat, EPCC

Douglas Smith, EPCC