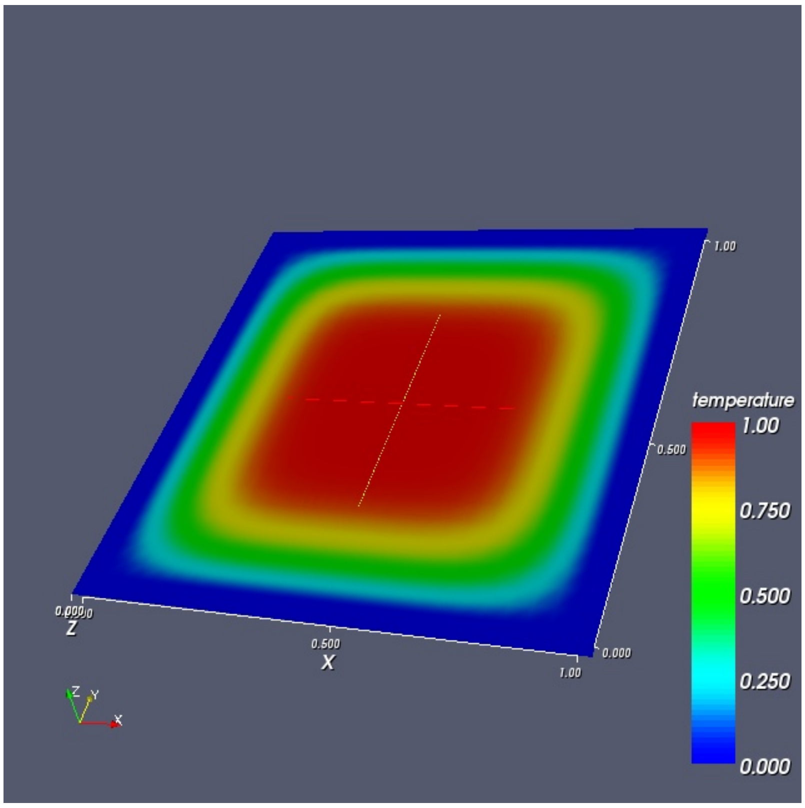
Time Integration Methods for the Heat Equation

Tobias Köppl - JASS March 2008

Heat Equation:

$$\partial_t u - \Delta u = 0$$



Preface

This paper is a short summary of my talk about the topic: Time Integration Methods for the Heat Equation, I gave at the Euler Institute in Saint Petersburg.
The goal of this talk was first to present Time integration methods for ordinary differential equations and then to apply them to the Heat Equation after the discretization of the Laplacian operator.
Moreover accuracy of the Time integration methods and stability conditions for our algorithms were discussed.
The picture above shows the solution of the Heat Equation at a certain time on the unit square, in which the solution of the Heat Equation was said to be zero at the boundary of the unit square.

1. Time Integration Methods

1.1. Implicit and explicit Onestep Methods

Before we talk about solution methods for the Heat Equation, we want to construct solution methods for ordinary differential equations (ODE). So for the moment our goal is to find numerical approximations of functions $x \in C^1([t_0, T], \mathbb{R}^d)$, which are solutions of an initial value problem (IVP):

$$\frac{d}{dt}x(t) = f(t, x)$$
$$x(t_0) = x_0$$

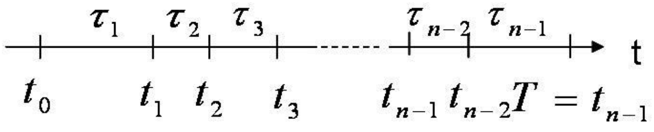
x_0 is a vector in $\mathbb{R}^d$, f ist a smooth function, which maps from $[t_0, \infty) \times \mathbb{R}^d$ into $\mathbb{R}^d$.

The first step for the construction of an numerical solution method for ODE is to divide the continous intervall $[t_0, T]$ by $n+1$ discrete timepoints:

$$t_0 < t_1 < ...t_n = T$$

This set of timepoints forms a grid Δ on $[t_0, T]$: $\Delta = \{t_0, t_1, ..., t_n\}$

For further discussions it is useful to define the expression stepsize of an grid. The stepsize τ_Δ of our Grid Δ is defined as: $\tau_\Delta := \max\{\tau_j = t_{j+1} - t_j | 0 \leq j < n\}$.



Now we want to approximate the solution of the IVP at the gridpoints t_j by a gridfunction $x_\Delta : \Delta \to \mathbb{R}^d$. So x_Δ should fullfill the following: $x_\Delta(t) \approx x(t)$ for all t $\in \Delta$.

In additon to that it should be possible to compute a certain value $x_\Delta(t_{j+1})$ by recursion, using only $x_\Delta(t_j)$. Such Time Integration Methods are called Onestep Methods.

$$x_\Delta(t_0) \to x_\Delta(t_1) \to ... \to x_\Delta(t_n)$$

Of course one does not need to use only $x_\Delta(t_j)$ in order to compute $x_\Delta(t_{j+1})$, but one can also use several values of the gridfunctions, which were computed in former steps (See literature e.g. [DB II]).

Two popular Onestep Methods are the explicit and the implicit Euler Method:

$$x_\Delta(t_0) = x_0$$

$$\textit{explicit EulerMethod}:$$

$$x_\Delta(t_{j+1}) = x_\Delta(t_j) + \tau_j f(t_j, x_\Delta(t_j))$$

$$\textit{implicit EulerMethod}:$$

$$x_\Delta(t_{j+1}) = x_\Delta(t_j) + \tau_j f(t_{j+1}, x_\Delta(t_{j+1}))$$

Considering this two Onestep Methods, it becomes clear, why it is necessary to distinguish between implicit and explicit Onestep Methods. Using the explicit Euler Method, $x_\Delta(t_{j+1})$ can be computed directly. But if you use the implicit Euler Method, $x_\Delta(t_{j+1})$ can in general only be computed by the solution of an in general non linear equation. Now one may ask why implicit Onestep Methods are considered at all, because of the computional effort, which is required in order to solve the mentioned equation. The answer is given in the next section, in which the convergence theory of Onestep Methods is treated.

1.2. Convergence theory for Onestep Methods

In this section our goal is to derive conditions under which a Onestep Method converges towards the exact solution of a given IVP. But before we can treat this problem in detail, we first have do define several expressions, which help us to set up our convergence theorem.

Definition (local discretization error)
The **local discretization error** $l(\Delta)$ of a grid function $x_\Delta : \Delta \to \mathbb{R}^d$ for a grid Δ on the intervall $[t_0, T]$ is defined as:

$$l(\Delta) = \max_{0 \le j < n}\{\left|x_\Delta(t_{j+1}) - x^{(j)}(t_{j+1})\right|\}$$

$x^{(j)}$ ist the solution of the IVP:

$$\frac{d}{dt}x(t) = f(t,x)$$
$$x(t_j) = x_\Delta(t_j)$$

Definition
A Onestep Method is called **consistent**, if: $l(\Delta) \to 0$ for $\tau_\Delta \to 0$

Theorem
The explicit and the implicit Euler Method are consistent.

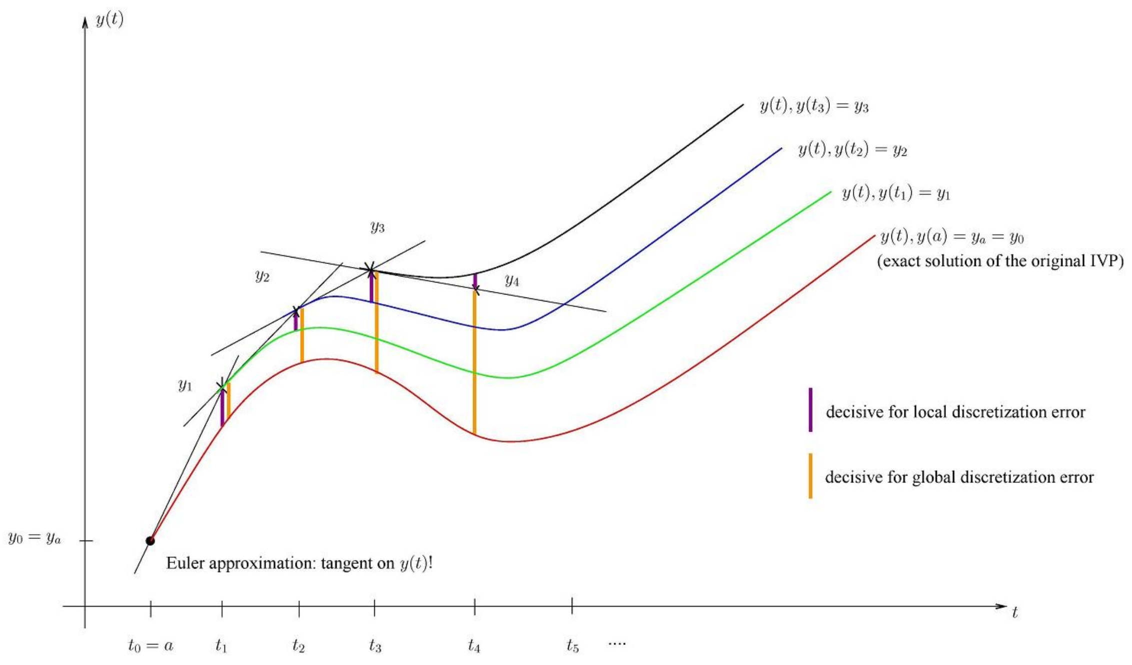
Proof: See e.g. [DB II] Chapter 4.

<u>Definition</u> (global discretization error)
The **global discretization error** $e(\Delta)$ is the maximum error between the computed approximations $x_\Delta(t_j)$ and the corresponding values of the exact solution $x(t_j)$.

$$e(\Delta) = max\{|x_\Delta(t_{j+1}) - x(t_{j+1})|\}$$
$$0 \leq j < n$$

<u>Definition</u> (convergence)
A Onestep Method is called **convergent** towards the exact solution on an IVP, if:
$e(\Delta) \rightarrow 0$ for $\tau_\Delta \rightarrow 0$

The following graph helps us to get a better imagination of the local and the global discretization error:



Now we have all the expressions, which are necessary to formulate our convergence theorem.

<u>Maintheorem of Numerics</u>
A Onestep Method is called convergent if and only if it is consistent.

Now we know the condition under which our Onestep Method is convergent. But the above Theorem is only ture, if we can chose an arbitrary small stepsize. The problem is that we can not choose an infinitly small stepsize for our computations with the help of a computer. In addition to the Maintheorem of Numerics our algorithm has to have a further property called stability, in order to achieve good numerical results with an acceptable small stepsize.

Definition (stability)
A numerical algorithm is called **stable**, if for all permitted input data perturbed in the size of computional accuracy $O(\epsilon)$ acceptable results are produced under the influence of rounding and method errors.

Example
If we apply the implicit and the explicit Euler Method to the following IVP (Dahlquist's testequation):

$$\frac{d}{dt}x(t) = \lambda x(t)$$
$$x(0) = 1, \lambda \in \mathbb{R}$$

one gets the following result:

Example
The implicit Euler Method is stable for any stepsize τ_Δ.
The explicit Euler Method is only stable, if $\tau_\Delta \leq \left|\frac{2}{\lambda}\right|$.

2. The Heat Equation

2.1. Discretization of the Laplacian operator

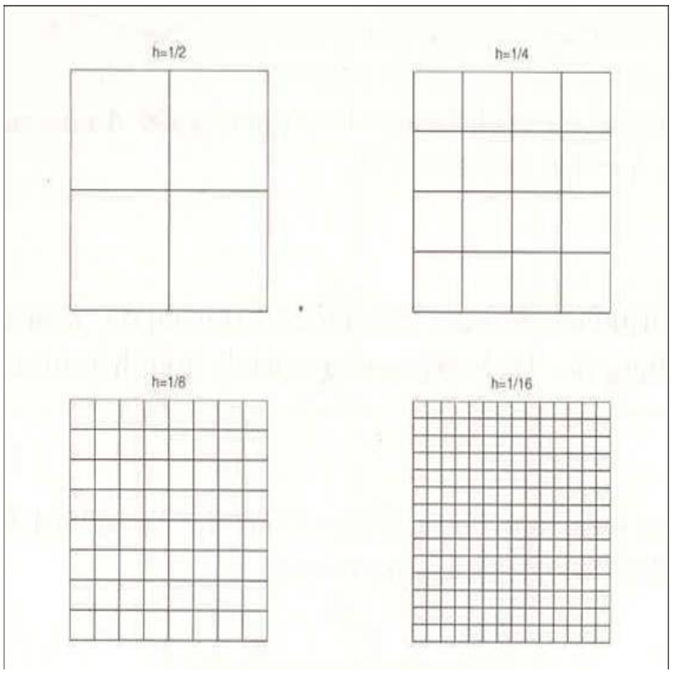
Before we can solve the Heat Equation, we have to think about solution methods for the Poisson equation (PE), for simplicity we consider only the two dimensional case:

$$-\Delta u = f$$
$$\Omega = [0,1]^2, u|_{\partial\Omega} = 0$$
$$f : \Omega \to \mathbb{R}$$

In order to solve the Poission equation, we transfer the partial differential equation into a system of linear equations. Here we replace differential operators by difference operators and discretize our domain Ω by an uniform grid with gridparameter h. h is the distance between two neighboured nodes of the grid in x- or y-direction. Thus we have the following discretization points:

$$(x_i, y_j) \in \Omega$$
$$x_i = ih, y_j = jh$$
$$0 \leq i, j \leq \frac{1}{h}$$

The picture below shows some possible discretizations or our domain Ω by an uniform grid.



Before we discretize the twodimensional Laplacian operator, we have to introduce some notation:

$u_{i,j} = u(x_i, y_j)$ (x_i, y_j) is an interior point of Ω.
$f_{i,j} = f(x_i, y_j)$

The next step is to expand $u(x_i + h, y_j)$ and $u(x_i - h, y_j)$ into a Taylor series up to order 4 around $x_i = ih$.

Furthermore one needs to expand $u(x_i, y_j + h)$ and $u(x_i, y_j - h)$ into a Taylor series up to order 4 around $y_j = jh$.

After that we add the Taylor series of $u(x_i + h, y_j)$ and $u(x_i - h, y_j)$ and we get an new equation. Next we solve this equation for the second derivative with respect to x and get:

$\partial_{xx}u(x_i, y_j) = \frac{1}{h^2}(u(x_i + h, y_j) - 2u(x_i, y_j) + u(x_i - h, y_j)) + O(h^4)$
Then one has to do the same with $u(x_i, y_j + h)$ and $u(x_i, y_j - h)$ and gets:
$\partial_{yy}u(x_i, y_j) = \frac{1}{h^2}(u(x_i, y_j + h) - 2u(x_i, y_j) + u(x_i, y_j - h)) + O(h^4)$

If you neglect the term $O(h^4)$, you get the following expression of the twodimensional Laplacian operator:

$$-\Delta u_{i,j} \approx \frac{1}{h^2}(4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1})$$

The Discretization error is in the order of $O(h^4)$.

In order to get a numerical solution of the Poisson equation on each gridpoint, one has to solve the following system of linear equations:

$$\frac{1}{h^2}(-4u_{i,j} + u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) = -f_{i,j}$$

$$0 \leq i, j \leq \frac{1}{h}$$

Matrix-Vector notation:

$$Au = -f$$

A has the following structure:

$$A := \frac{1}{h^2}\left(\begin{array}{cccc|cccc|cccc|cccc}
-4 & 1 & & & 1 & & & & & & & & & & & \\
1 & \ddots & \ddots & & & \ddots & & & & & & & & & & \\
 & \ddots & \ddots & 1 & & & \ddots & & & & & & & & & \\
 & & 1 & -4 & & & & 1 & & & & & & & & \\
\hline
1 & & & & -4 & 1 & & & \ddots & & & & & & & \\
 & \ddots & & & 1 & \ddots & \ddots & & & \ddots & & & & & & \\
 & & \ddots & & & \ddots & \ddots & 1 & & & \ddots & & & & & \\
 & & & 1 & & & 1 & -4 & & & & \ddots & & & & \\
\hline
 & & & & \ddots & & & & \ddots & & & & 1 & & & \\
 & & & & & \ddots & & & & \ddots & & & & \ddots & & \\
 & & & & & & \ddots & & & & \ddots & & & & \ddots & \\
 & & & & & & & \ddots & & & & \ddots & & & & 1 \\
\hline
 & & & & & & & & 1 & & & & -4 & 1 & & \\
 & & & & & & & & & \ddots & & & 1 & \ddots & \ddots & \\
 & & & & & & & & & & \ddots & & & \ddots & \ddots & 1 \\
 & & & & & & & & & & & 1 & & & 1 & -4
\end{array}\right)$$

It can be easily seen that A is a sparse matrix, thus one should use fast iterative solvers, in order to solve the system of linear equations, which is given above.

2.2. Application of Time Integration Methods

In this section our goal is to find numerical approximations of functions u, solving the homogenous Heat Equation:

$$\partial_t u - \Delta u = 0$$

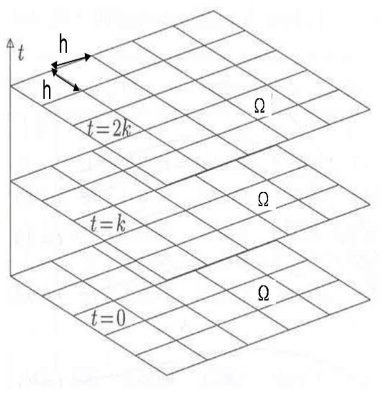
$$\Omega = [0,1]^2, u|_{\partial\Omega} = 0$$

$$g : (0,1)^2 \to \mathbb{R}, u(0,x) = g(x)$$

$$t \in [t_0, T]$$

It is clear that the solution of the Heat Equation depends on space and time. Thus discretization of time and space is necessary in order to get a discrete system of linear equations.

Both time and space can be discretized by an uniform grid, as it can be seen in the picture below:



We can discretize our unit square Ω again by an uniform grid with gridparameter h.
This yields the same discretizations points as in section 2.1.:

$$(x_i, y_j) \in \Omega$$

$$x_i = ih, y_j = jh$$

$$0 \leq i, j \leq \frac{1}{h}$$

Moreover we discretize the intervall $[t_0, T]$ by an onedimensional grid with stepsize k and n discrete timepoints:

$$t_m = km, 0 \leq m < n$$

We denote $u(t_m, x_i, y_j)$ with: $u_{m,i,j}$

Now one can take care about the construction of an solver of the Heat Equation:

$$\partial_t u - \Delta u = 0 \Leftrightarrow \partial_t u = \Delta u$$

The next step is to construct local initial value problems for every interior point (x_i, y_j) $0 \leq i, j \leq \frac{1}{h}$(IVP(ij)):

$$\frac{d}{dt} u(t, x_i, y_j) = \Delta u(t, x_i, y_j)$$

$$u(0, x_i, y_j) = g(x_i, y_j)$$

$$t \in [t_0, T]$$

IVP(ij) can be solved for example by the implicit Euler Method. Applying the implicit Euler Method to IVP(ij), we get the following formula:

$$u_{0,i,j} = g(x_i, y_j)$$

$$u_{m+1,i,j} = u_{m,i,j} + k\Delta u_{m+1,i,j}$$

After that we insert the discretization of the two dimensional Laplacian operator:

$$-\Delta u_{m+1,i,j} \approx \frac{1}{h^2}(4u_{m+1,i,j} - u_{m+1,i+1,j} - u_{m+1,i-1,j} - u_{m+1,i,j+1} - u_{m+1,i,j-1})$$

and this yields the following system of linear equations, whích is to be solved in every timestep, for example by a fast iterative solver (Jacobi Method, Gauss-Seidel Method):

$$(4 + \frac{h^2}{k})u_{m+1,i,j} + u_{m+1,i+1,j} + u_{m+1,i-1,j} + u_{m+1,i,j+1} + u_{m+1,i,j-1} = \frac{h^2}{k}u_{m,i,j}$$

$$0 \leq i, j \leq \frac{1}{h}$$
$$0 \leq m < n$$

2.3. Courant-Friedrichs-Levy condition

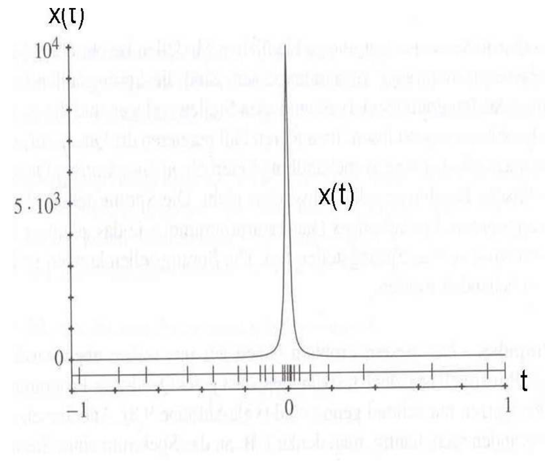
IVP(i,j) can also be solved by the explicit Euler Method. But remember: In section 1.2. we realized that the explicit Euler Method is only stable for small stepsizes k. We also know that stability is an essential condition for getting qualitatively correct solutions, when using practical stepsizes.

In 1928 Courant, Friedrichs and Levy found a condition under which the explicit Euler Method is a stable solver for the Heat Equation:

Theorem(CFL - condition)
The explicit Euler Method is a stable solver for the Heat Equation, if:

$$\left|4\frac{k}{h^2}\right| < 1$$

3. Outlook



Let $x(t)$ be a solution of an IVP. It is clear, that an uniform discretization of the time axis would lead to a slow convergence. Thus adaptive algorithms with a good errormeasurement are required in order to get a better convergence.
Further things, that would improve our numerical algorithm would be the optimization of the algorithms solving the sparse linear system of equations, with respect to storage and number of floating point operations, constructing an algorithm, which helps us to traverse our grid efficiently (peano curves) or the preconditioning of the Matrix representing the system of linear equations.

4. Literature

DH I: Numerische Mathematik I, Deuflhard/Hohmann, 2002, 3. edition
DB II: Numerische Mathematik II, Deuflhard/Bornemann, 2002, 2. edition
Jun: Lecture on Numerische Mathematik, Junge, 2007
SK: Numerische Mathematik, Schwarz/Köckler, 2006, 6. edition
QSS: Numerische Mathematik 2, Quateroni/Sacco/Saleri, 2002, 2. edition
Wa: Lecture on Finite Elements, Wall, 2007, 2. edition
Hu: Numerics for computer sience students, Huckle/Schneider, 2002, 3. edition
Bun: Lecture on numerical programming, Bungartz, 2007
El: Finite Elements and Fast Iterative Solvers, Elman/Silvester/Wathen, 2005, 2. ed.
Fo I: Analysis I, Forster, 1976, 6. edition
Fo II: Analysis II, Forster, 1976, 6. edition
Fei: Introduction into the theory of partial differential equations, 2000
CFL: Über die partiellen Differentialgleichungen der mathematischen Physik, Courant, Friedrichs, Levy, 1928