Technische Universität München
Lehrstuhl für Informatik V
Dr. T. Neckel, D. Pflüger

SS 2010
Solution 3a
7.5.2010

Algorithms of Scientific Computing (Algorithmen des Wissenschaftlichen Rechnens)

Fast Poisson Solver – Solution

Derivation of the System of Linear Equations – For the Sake of Completeness

In the lecture we derived the system of equations from a diskrete modell. However, the same systems of equations show up during the numerical solution of (partial) differential equations.

The examined one-dimensional Heat Transfer Problem is modelled by the so called *Poisson equation*. With appropriate *boundary conditions*, this looks like shown here:

$$\begin{aligned} -\tfrac{\partial^2}{\partial x^2}u(x) &= f(x) \quad \text{for} \quad x \in (0,1) \\ u(0) = u(1) &= 0 \end{aligned} \tag{1}$$

For a numerical solution we look for a solution $u(x)$ on the discrete points $x_n := nh$, with $n = 0, \ldots, N$ and $h := \frac{1}{N}$. So, in the equation (1) we substitute the partial derivative by an adequate difference quotient and get the following approximation on the points x_n:

$$\begin{aligned} -\frac{u(x_{n+1}) - 2u(x_n) + u(x_{n-1})}{h^2} &\approx f(x_n) \quad \text{for} \quad n = 1, \ldots, N-1 \\ u(x_0) = u(0) &= 0 \\ u(x_N) = u(1) &= 0 \end{aligned} \tag{2}$$

With $f_n := h^2 f(x_n)$ we get a system of linear equations for computing the approximation $u_n \approx u(x_n)$:

$$\begin{aligned} -u_{n+1} + 2u_n - u_{n-1} &= f_n \quad \text{for} \quad n = 1, \ldots, N-1 \\ u_0 = u_N &= 0 \end{aligned} \tag{3}$$

In the two-dimensional case the Poisson equation derives to

$$\begin{aligned} -\tfrac{\partial^2}{\partial x^2}u(x,y) - \tfrac{\partial^2}{\partial y^2}u(x,y) = f(x,y) \quad &\text{für} \quad x \in \Omega = (0,1)\times(0,1) \\ u(x,y) = 0 \quad &\text{if} \quad x \in \{0,1\} \quad \text{oder} \quad y \in \{0,1\}. \end{aligned} \tag{4}$$

We are now looking for the approximations $u_{n,m} \approx u(x_n, y_m)$, where $x_n := nh$ and $y_m := mh$ for $n, m = 0, \ldots, N$. So, we use a regular *Grid* with points $x_n, y_m)$, where we use the same number of points in x- and y-direction. The partial derivation in equation (4) we approximate analog to the one-dimensional case by means of the difference quotients

$$\frac{\partial^2}{\partial x^2}u(x_n, y_m) \approx \frac{u_{n+1,m} - 2u_{nm} + u_{n-1,m}}{h^2} \quad \text{and} \quad \frac{\partial^2}{\partial y^2}u(x_n, y_m) \approx \frac{u_{n,m+1} - 2u_{nm} + u_{n,m-1}}{h^2}. \tag{5}$$

If we set $f_{nm} := h^2 f(x_n, y_m)$, we get the following system of linear equations:

$$\begin{aligned} -u_{n,m+1} - u_{n+1,m} + 4u_{nm} - u_{n-1,m} - u_{n,m-1} = f_{n,m} \quad &\text{for} \quad n, m = 1, \ldots, N-1 \\ u_{0,m} = u_{n,0} = 0 \quad &\text{for} \quad n, m = 0, \ldots, N \end{aligned} \tag{6}$$

Excercise 1

We insert the transformations

$$u_n = 2\sum_{k=1}^{N-1} U_k \sin\frac{\pi nk}{N} \qquad \text{and} \qquad f_n = 2\sum_{k=1}^{N-1} F_k \sin\frac{\pi nk}{N} \tag{7}$$

into the system of linear equations (3):

$$\begin{aligned} -u_{n+1} + 2u_n - u_{n-1} = f_n \quad &\text{für} \quad n = 1, \ldots, N-1 \\ u_0 = u_N = 0, \end{aligned}$$

So, for $n = 1, \ldots, N-1$, we get:

$$-2\sum_{k=1}^{N-1} U_k \sin\frac{\pi(n+1)k}{N} + 4\sum_{k=1}^{N-1} U_k \sin\frac{\pi nk}{N} - 2\sum_{k=1}^{N-1} U_k \sin\frac{\pi(n-1)k}{N} = 2\sum_{k=1}^{N-1} F_k \sin\frac{\pi nk}{N} \tag{8}$$

Note that the homogenous boundary conditions $u_0 = 0$ and $u_N = 0$ are necessary to get a correct conversion for $n = 1$ and $n = N-1$, since here one of the terms of the sum become zero.

By using the theorems of addition

$$\sin(A+B) = \sin A\cos B + \cos A\sin B \quad \text{und} \quad \sin(A-B) = \sin A\cos B - \cos A\sin B$$

equation (8) can be converted to:

$$\begin{aligned}
&- \quad 2\sum_{k=1}^{N-1} U_k \left(\sin\frac{\pi nk}{N}\cos\frac{\pi k}{N} - \cos\frac{\pi nk}{N}\sin\frac{\pi k}{N}\right) + 4\sum_{k=1}^{N-1} U_k \sin\frac{\pi nk}{N} \\
&- \quad 2\sum_{k=1}^{N-1} U_k \left(\sin\frac{\pi nk}{N}\cos\frac{\pi k}{N} + \cos\frac{\pi nk}{N}\sin\frac{\pi k}{N}\right) = 2\sum_{k=1}^{N-1} F_k \sin\frac{\pi nk}{N} \\
\Leftrightarrow &\quad 4\sum_{k=1}^{N-1} U_k \left(\sin\frac{\pi nk}{N} - \sin\frac{\pi nk}{N}\cos\frac{\pi k}{N}\right) = 2\sum_{k=1}^{N-1} F_k \sin\frac{\pi nk}{N} \\
\Leftrightarrow &\quad 2\sum_{k=1}^{N-1} U_k \sin\frac{\pi nk}{N}\left(1 - \cos\frac{\pi k}{N}\right) = \sum_{k=1}^{N-1} F_k \sin\frac{\pi nk}{N}
\end{aligned}$$

This is true, if

$$2U_k\left(1 - \cos\frac{\pi k}{N}\right) = F_k.$$

holds for all $k = 1, \ldots, N-1$.

From this we can immediately derive the required dependency

$$U_k = \frac{F_k}{2 - 2\cos\frac{\pi k}{N}}. \tag{9}$$

Algorithm:

Equation (9) provides the following method for solving the system of equations:

1. Compute the coefficients F_k by a Fast Sine Transform.
2. Compute all coefficients U_k from the F_k by equation (9) for all $k = 1, \ldots, N-1$.
3. Compute the u_n from the U_k by means of a Inverse Fast Sine Transform.

Both of the Sine Transforms take $\mathcal{O}(N\log N)$ operations each, while step 2 needs only $\mathcal{O}(N)$ operations. In total the system of equations can be solved by this algorithm in $\mathcal{O}(N\log N)$ operations.

So, it is slower than solving the tri-diagonal system of equations directly, since this takes only $\mathcal{O}(N)$ operations. However, the algorithm provides a benefit in the n-dimensional case.

Excdercise 2: Two-Dimensional Fast Poisson Solver

We insert the transformations

$$u_{nm} = 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl}\sin\frac{\pi nk}{N}\sin\frac{\pi ml}{N} \qquad \text{and} \qquad f_{nm} = 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} F_{kl}\sin\frac{\pi nk}{N}\sin\frac{\pi ml}{N} \tag{10}$$

into the system of equations

$$-u_{n,m+1} - u_{n+1,m} + 4u_{nm} - u_{n-1,m} - u_{n,m-1} = f_n \qquad \text{for} \quad n, m = 1, \ldots, N-1 \tag{11}$$

and get

$$\begin{aligned}
- & \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi(n+1)k}{N} \sin\frac{\pi ml}{N} - \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi(n-1)k}{N} \sin\frac{\pi ml}{N} \\
- & \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi(m+1)l}{N} - \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi(m-1)l}{N} \\
+ & \; 4\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N} = \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} F_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N}
\end{aligned}$$

As in the one-dimensional case we can convert as follows:

$$\begin{aligned}
& \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi(n+1)k}{N} \sin\frac{\pi ml}{N} + \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi(n-1)k}{N} \sin\frac{\pi ml}{N} \\
= & \; 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \cos\frac{\pi k}{N} \sin\frac{\pi ml}{N},
\end{aligned}$$

and

$$\begin{aligned}
& \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi(m+1)l}{N} + \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi(m-1)l}{N} \\
= & \; 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N} \cos\frac{\pi l}{N}.
\end{aligned}$$

So, we get

$$\begin{aligned}
- & \; 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N} \cos\frac{\pi k}{N} \\
- & \; 2\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N} \cos\frac{\pi l}{N} \\
+ & \; 4\sum_{k=1}^{N-1}\sum_{l=1}^{N-1} U_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N} = \sum_{k=1}^{N-1}\sum_{l=1}^{N-1} F_{kl} \sin\frac{\pi nk}{N} \sin\frac{\pi ml}{N}
\end{aligned}$$

This holds, if

$$U_{kl}\left(4 - 2\cos\frac{\pi k}{N} - 2\cos\frac{\pi l}{N}\right) = F_{kl},$$

holds for all $k, l = 1, \ldots, N-1$.

From this we get the dependency

$$U_{kl} = \frac{F_{kl}}{4 - 2\cos\frac{\pi k}{N} - 2\cos\frac{\pi l}{N}}. \tag{12}$$

Algorithm:

The two-dimensional algorithms is virtually identically to the one-dimensional:

1. Compute the coefficients $F_{k,l}$ by means of a (two-dimensional) Fast Sine Transform.
2. Compute the coefficients $U_{k,l}$ as shown in equation (12) for all $k, l = 1, \dots, N-1$.
3. Compute the unknown $u_{n,m}$ out of the coefficients $U_{k,l}$ with a (two-dimensional) Inverse, Fast Sine Transform.

Both of the Sine Transforms take $\mathcal{O}(N^2 \log N)$ operations each, while step 2 needs only $\mathcal{O}(N^2)$ operations. In total the system of equations can be solved by this algorithm in $\mathcal{O}(N^2 \log N)$ operations.

Some observations and remarks:

- The 2d system of equations can not be written in a from, so that the associated matrix is narrow band matrix (or even a tri-diagonal matrix). So, it cannot be solved directly in $\mathcal{O}(N^2)$ operations.
  - Usually the system of equations is written, so that the associated matrix is a band matrix of width N. So, a direct solver (Gauß elimination) takes $\mathcal{O}(N^4)$ operations.
  - The *nested dissection* gives a matrix, which can be solved by a Gauß elimination with $\mathcal{O}(N^3)$ operations.

  So, both methods have a worse complexity than the algorithm based on the Sine Transform.
- The derivation of the 2d case shows that this method can be transfered easily to the 3d case and higher dimensional cases. The complexity is $\mathcal{O}(N^d \log N)$ in each case.