Project 1

Thomas Aarflot Storaas and Erik Fagerås Skaar
Department of Chemistry
University of Oslo

13. september 2017

Vi har desverre tullet det til med tiden. Vi har gjort ferdig koden og er på god vei, men rapporten er ikke helt der. For en oppdatert versjon av rapporten og koden gå til githuben vår(https://github.com/erikfsk/Project-1).

1 Abstract

In this article the numerical approximation to a one dimensional Poisson equation is studied. Three approaches are taken. One with a straight forward Gaussian elimination in a general form. one in a spesified gaussian elimination form. The third approach are done with LU decomposition using the libarary "Armadillo". The numericaL accuracy is then analysed and the three methods are then compared with respect to FLOPS and time used.

2 Underlying theory

The Poisson's equation is a classical equation from electromagnetism, which states that $\nabla^2\Phi = -4\pi\rho((\mathbf{r}))$. The electrostatic potential Φ is a sentro- symmetrical potential, which makes it possible to rewrite the potential as a one dimentional equation with respect to the distance from origo, $\mathbf{r}$.

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\Phi}{\partial r}\right) = -4\pi\rho(r)$$

Introducing the substitutions $\Phi(r) = \phi(r)/r$ gives:

$$\frac{\partial^2\phi}{\partial r^2} = -4\pi r\rho(r) \tag{1}$$

Further manipulation of the expression with letting $x \to u$ and $r \to x$ gives:

$$-u''(x) = f(x)$$

For this project the Poisson equation will be solved with Dirichlet boundary conditions and source function $f(x) = 100\mathrm{e}^{-10x}$, in the interval $x \in [0, 1]$, which gives.

$$u(0) = u(1) = 0 \tag{2}$$

For this specific source function a numerical solution can be found by doing a double integration, and using the Dirichlet boundary conditions.

$$\begin{aligned} f(x) &= 100\mathrm{e}^{-10x} \\ \int f(x)\,\mathrm{d}^2x &= c_1x + c_2 + e^{-10x} \\ &= 1 - \left(1 - \mathrm{e}^{-10}\right)x - \mathrm{e}^{-10x} \end{aligned}$$

The discretized approximation to u is v_i, where the step size is defined as $h = 1/(n+1)$ with grid points(sample step) $x_i = ih$. The discretized approximation from 2 gives $v_0 = v_{n+1} = 0$. The second derivative approximation of u are

$$f_i = -\frac{v_{i+1} + 2v_i - v_{i-1}}{h^2} \qquad i = 1, 2...n$$

For each step x_i a equation can be found:

$$\begin{aligned} f_2 &= (-v_1 + 2v_2 - v_3)/h^2 \\ f_3 &= (-v_2 + 2v_3 - v_4)/h^2 \\ f_4 &= (-v_3 + 2v_4 - v_5)/h^2 \end{aligned}$$

Note here that the end points are not included. This system of equations can be organized in a matrix, by defining $\tilde{b}_i = h^2 f_i$.

$$\begin{matrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{matrix}$$

A row reduction of $A\mathbf{x} = f$ will give the solution to all the equations.

3 Method

General brute force Gaussian elimination

The matrix equations we need to solve is the following:

$$\mathbf{A} = \begin{bmatrix} b_1 & c_1 & 0 & \dots & \dots & \dots \\ a_1 & b_2 & c_2 & \dots & \dots & \dots \\ & a_2 & b_3 & c_3 & \dots & \dots \\ & \dots & \dots & \dots & \dots & \dots \\ & & & a_{n-2} & b_{n-1} & c_{n-1} \\ & & & & a_{n-1} & b_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ \dots \\ \dots \\ v_n \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \dots \\ \dots \\ \dots \\ \tilde{b}_n \end{bmatrix}$$

Applying a general approach to the gaussian elimination of a 4×4 matrix gives an expression which can be generalized:

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 \\ a_1 & b_2 & c_2 & 0 \\ 0 & a_2 & b_3 & c_3 \\ 0 & 0 & a_3 & b_4 \end{bmatrix} \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \tilde{b}_3 \\ \tilde{b}_4 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & c_1/b_1 & 0 & 0 \\ 0 & b_2 - (c_1/b_1)a_2 & c_2 & 0 \\ 0 & a_2 & b_3 & c_3 \\ 0 & 0 & a_3 & b_4 \end{bmatrix} \begin{bmatrix} \tilde{b}_1/b_1 \\ \tilde{b}_2 - (\tilde{b}_1/b_1)a_1 \\ \tilde{b}_3 \\ \tilde{b}_4 \end{bmatrix}$$

4 Result

5 Discussion

6 Conclusion

The specific solution is the fastet, but if the matrix have a slight change it does not solve the problem we have. The general is almost as fast, but it can solve a wider range of tridiagonal matrices. The LU solution is the slowest. It is useful no matter what matrix you have, but it has a significant drawback in speed. By our calculations the error is the same for every method.