FYS3150 - Project 1

Git Repository: <https://github.com/henrikx2/FYS3150>

**Abstract**

This report reveals how ordinary differential equations can be solved numerically using a matrix equation. It shows how the matrix equation can be simplified and specialized through algorithms to obtain less experimental errors and faster calculation than i.e. a solution with LU-decomposition or a standard matrix multiplication.

1. **Introduction**

This experiment aims to solve the one-dimensional Poisson equation with Dirichlet boundary conditions. Poisson’s equation is written as:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

And the one-dimensional equation to solve is in this case expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

To solve the equation, we will use a numerical approximation to the second derivative and express it as a matrix equation with a tridiagonal matrix. We will then solve the matrix equation generally by rewriting it as three one-dimensional vectors and specialize it afterwards. In the end we will use LU-decomposition to solve the matrix equation and look at the difference number of floating points, CPU-time and relative error.

1. **Methods and approximations**
   1. **Discretized values**

To do integration on a computer, we have to work with discretized variables. In this case the main function, , is a function of , and we therefore need to be discrete. In other words;

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Where is the distance (step size) between each discretized and is the number of -values on the interval . In this explicit case we define and , this means that there is steps from to and that we may express the step size as:

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Furthermore, we denote the discretized approximation to as and .

* 1. **Second derivative approximation**

To calculate the second derivative numerically, we use an expression which evolves from tweaking two different taylor-expansions. Equation (5) shows the general expression of a taylor-expansion around a point :

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Now, knowing that is discretized, we can calculate one step forward and one step backward using equation (5).

|  |  |  |
| --- | --- | --- |
|  |  | (6) |
|  |  | (7) | |

We also denote the discrete values as follows:

To find an expression for the 2nd derivative we add equation (6) and (7) together and get equation (8):

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

Equation (8) gives the possibility to calculate the 2nd derivative by knowing the function values at each step .

* 1. **Tridiagonal matrix equation**

Now, by using equation (8) to describe equation (2) we get:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

Which we rewrite as:

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

This equation consists of three trailing -values no matter what we use. We can show how this will look when evolves from to by the following equation set:

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

We know that and that in every equation there is one multiplied with 2 and one and one multiplied with -1. This gives rise to a tridiagonal matrix of the type:

such that we can write the -dimensional equation system as a matrix equation as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

Where:

If we make the diagonals in matrix vectors , and , we can write the matrix equation in a general way as:

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

It’s important to note that the endpoints of the vectors are not included in this matrix, because we already know from the Dirichlet boundary conditions what they are.

* 1. **The analytical solution**

The source term gives the solution of equation (2) as

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

This can be shown by derivation:

This will be used to compare the numerical solutions.

* 1. **The General Algorithm**

To solve the matrix equation (13) we will use the following algorithm

* 1. **The Special Algorithm**

In the case when we know that all the elements at each diagonal is the same, the calculations can be done in another way.

* 1. **LU-Decomposition**

The last way to calculate the 2nd derivative will be by using the LU-decomposition.

1. **Results**
2. **Conclusion**
3. **Appendix**

**References**