Documentation weighting field

February 1, 2022

1 Weighting field

1.1 Documentation

Refer to 18/01/22
Refer to 21/01/22

1.1.1 Domain and dimensions

This part of the project involves the solution of Lapalace's equation for a configuration of electrodes as shown in P.1 of the paper notes.

The domain consisist of 11 electrodes.
The central electrode is set to 1, the other electrodes are set to 0, far away from the configuration 0 value boundary conditions are used.

P.1 also shows the dimensions for a typical element of the chain of electrodes.

1.1.2 Equation

P.2 shows the equation to be solved, its discretisation and the sources involved.

Discretised equation The discretised system of equations will form a pentadigonal matrix with entries:
[1, 1, -4, 1, 1]

1.1.3 Structuring the mesh

P.3 shows how the mesh was set up, the mesh size, the number of grid points etc.
Unit is microns.

The height of a block is indicated by N__y = (height/mesh_size).
The width of a block is indicated by N__x = (pitch/mesh_size).

They will differ between top and bottom blocks.

Mesh organisation The mesh was divided into two types of blocks:
1) Block with electrode segment;
2) Block without electrode segment.

(1) consisits of a block of size: pitch = 100, height = 900, mesh size = 0.5, an electrode is located at position y = 300, x_1 = 40, x_2 = 60.

(2) consists of a block of size: pitch = 100, height = 900, mesh size = 1, no electrodes just an empty rectangle.

(3) is used to describe the system in its poximity, (2) is used to describe the system further away where the W.F. is expected to quickly drop to 0, so much less variation is expected.

P.3 shows the naming convention.

Block naming Central Central_t
Central has electrode set to 1
Central_t is the free space expansion above the central block

To the left of the central electrode

Left_n Left_n_t
Left_n is electrode block with electrode set to 0
Left_n_t is the fress space extension above the block
n is a number from 1 to n w.r.t. the central block

B_left B_left_t
These electrodes have the left_most boundary set to 0 by default

To the right of the central electrode

Right_n Right_n_t
Right_n is electrode block with electrode set to 0
Right_n_t is the fress space extension above the block
n is a number from 1 to n w.r.t. the central block

B_right B_right_t
These electrodes have the right_most boundary set to 0 by default

Cross talk between blocks

As it can be seen from the drawing at P.3 not all blocks have a fixed value at the boundary (or not at all boundaries), the blocks generally share a boundary of variable values.

I shall call such cells just next to the boundaries between blocks transition cells.
P.4. shows the naming convention.
As it will be explained later, the transition cells belonging to the blocks surrounding a certain block will enter the system of equations for the cells inside such a block as sources.

1.1.4 The solution algorithm

P.5. shows the flow chart for the solution of Laplace's equation for the domain given.

Here:
SOR_one_iter() is a function that performs 1 iteration of the Gauss-Seidel SOR algorithm over all cells of a block;
global_residual() is a function that computes the global residual for a given block.

More on the algorithm later.

1.1.5 The mathematics problem

1.1.6 Solving the system of discretised equations

At P.7 you can find the algorithm that is implemented: Gauss-Seidel with over-relaxation.

The algorithm clearly shows that you must correctly map coefficients to entries in the potential vector.
This can be done by establishing a numbering convention for the cells in the block.

N.B.: the coefficients are fixed for all cells, so they can be generated by appending repeated elements to lists.

Numbering convention P.8 shows the numbering convention for the cells in the block.

Matrix of coefficients and matrix of positions For each block we define a matrix of coefficcients and a matrix of positions.

MATRIX OF COEFFICIENTS

```
            column
            S    W    P    E    N
row cell    1    1   -4    1    1
```

this in a matrix that is N_x*N_y big in rows.

MATRIX OF POSITIONS

```
            column
            S            W            P            E            N
row cell    P - N_x      P - 1       cell          P + 1        P + N_x
```

this in a matrix that is N_x*N_y big in rows.
Simple expressions are needed to map a cell at a specific location w.r.t. a given cell to the entries of the potential vector.

From the matrix of positions we can extract the cells that surround a given cell.
We can then associate each coefficent of the discretised equation to the corresponding cell and perform the iteration.

The functions block_with_electrode and block_without_electrode create such matrices specifically for the two block topologies.

block_without_electrode Simple rectangualar domain

block_with_electrode

Rectangular domain, however the coefficients corresponding to the electrode cells are all set to 0.
A conditional statement in the SOR_one_iter function checks if the the P coeffcient is 0.

If a_P = 0, continue, else the iteration is executed for the given cell.
Since the elecrode cells are not updated by the SOR algorithm they will be left at their initial value (either 1 for the central electrode or 0)

Creating the source terms The source terms must be contantly updated since the boundaries of the domain are given by variable cells, only a limited numebr of boundaries correspond to the actual domain boundaries.

Two functions are used:

block__with__electrode__source This function takes the right transition cells of the block to the left of the given block, the transition cells to the left of the block to the right, and the transition cells at the bottom of the block in the top to produce the source term.

block__without__electrode__source Does the same, but uses the transition cells at the top of the block at the bottom of the given block.

The transition cells must be contantly updated after each iteration, so too must the source terms before each iteration of a block.

The transition cells corresponding to the boundary at infinity of the domain must not be updated from their 0 value.

The solution method The matrix of coefficients and the matrix of positions are defined so that only the active cells have an equation associated to them.
Since some of the potential cells inside the domain are set to 0 by BC, these don't have to be updated, but defining them inside the potential vector and not as a source is useful for 2 reasons:
1) No reason to worry about defining additional source terms, less code;
2) When we plot the soution the BC is already included.
In order to pull this off we only need to properly define the matrix of positions so that the SOR solve can access these cells, but not modify them.
This is also the reason why we need a number of active cells.

1.1.7 The solver module

1.1.8 SOR__one__iter

This function takes the matrix of coefficients, the matrix of positions, the source, an over-relaxation parameter (omega or w at P.7) and the number of active cells and performs 1 iteration of the Gauss-Seide SOR algorithm.

The number of active cells is needed because the for loop in this program runs over all the rows of the matrix of ceofficients and matrix of positions, and such matrices are created only for the cells that must be updated by the algorithm, that is all the block cells except for the bottom row cells (block with electrode) or top row cells (block without electrodes) that are set to 0 by BC.

This algorithm checks if a_P = 0, if it is (electrode) then the iteration is not executed.

1.1.9 Global__residual

P.9 for the definition of global residual.

It is implemented like that.

This is a parameter required to determine if we can stop the iterative algorithm.
A tolerance is set. When this tolerance is matched the algorithm is stopped and the solution is reached.

2 Observations

2.1 Number of grid points

Electrode block: pitch/0.5 = 200 height/0.5 = 1800 total number of cells = 360,000

Block without electrode: pitch/1 = 100 height/1 = 900 total number of cells = 90,000

Relatively few cells are added to enforce the boundary at infinity condition more effectively.

Total numnber of cells = (360,000 + 90,000)*11 = (450,000)*11 = 4,950,000

The large number of grid points implies a pretty long running time, of the order of many hours.

2.2 Diffusion of the BC

Check P.9 of paper notes

2.3 CG solver

Refer to notes of date 20/01/22 for explanation of Conjugate Gradient method.
Refer to 21/01/22 for notes about software.

The functions are defined in the usual module: residual_solvers

They have the same arguments of the SOR algorithm with two differences:
1) They both take the start cell besides the number of active cells, this is needed because if the number of active cells is less then the size of the total number of cells we then have to modify the source term so that it has the same size as the search vector;
2) CG_one_iter takes also the gradient and search vectors at step r-1, they are needed to iterate the system.

The convention used in the numbering of the cells is the usual one (see above)

Important note When the sytem has an electrode, which has a fixed potential, these cells, despite the fact that enter the calculation, are not updated since their matrix elements are all zero, this means that their residual is always 0, hence the CG algorithm never updates them.

Theory The theory is in the notes of date 20/01/22.