SPICE like solver for PEEC formulated frequency domain problems

Report nr.7, L'Aquila

February 4, 2002

Introduction

A SPICE like solver for PEEC formulated frequency domain problems has been developed using MVC++. The objective is (1)to be able to exclude the external SPICE program and thus save time and workload that arise when input files for SPICE has to be created. (2) To have a controlled environment where the input, ie the matrices, can be easily manipulated and modified.

The Solver

Input

The solver require a 'plain text' inputfile that describes:

- The metallic structure and corresponding resistivity.
- The discretization of the structure.

The solver perfoms the discretization and calculates:

- Partial inductances, static or frequency dependent and stores the values in the *lpijkc or* L-matrix.
- Partial coefficients of potential, ie capacitances, static or frequency dependent and stores the values in the *pijkc or* P-matrix.
- Volume cell resistivity, stored in the *resist or* R-matrix.
- Connectivity matrix, *CM*, describing the connection of the partial inductances.
- Reduced node matrix, Rv, see Report nr. 5 *Short report on the Rv matrix computation in C++.* Used to remove common nodes.

The matrices are then used to obtain the solution for selected frequencies using the *Admittance matrix method.*

The Admittance matrix method

In this method the matrices are arranged according to eq.1 to obtain the solution, V, describing the node voltages.

$$\left[CM^T(R + j\omega L)^{-1}CM + j\omega P^{-1}\right] V = I_s \tag{1}$$

Where the expression inside the brackets correspond to the systems admittance matrix, Y, so that eq.1 can be written $Y\ V = I_s$. The frequencies in eq.1, $j\omega$-vector, are so far defined in the C++ code. The 'input' to the system is specified in the I_s- vector as currents injected or extracted from one or several nodes, also done in the C++ code. Additional lumped components, R, L and C, can be added between nodes using the following C++ syntax respectively.

Y = insertR(+node, -node, Ωvalue, Y)
Y = insertL(+node, -node, Hvalue, freq, Y)
Y = insertC(+node, -node, Fvalue, freq, Y)

Output

The required output, specified in the C++ code, is written to an output file in plain text format suitable for example for Matlab comparision and visualization. All matrices involved in the computation can also be written to output files and if wanted a SPICE-input file can be created.

Example

The resonance frequency for a simple $\frac{\lambda}{2}$dipole, approx. 20 cm, is simulated with the developed C++ solver and compared with a Spice3f4 solution. The dipole is discretized into, only, 2 x 1 cells for each arm of the dipole.

Input

The input file for the 2 x 1 discretized dipole is:

```
numero barre 2

ndiva 2 ndivb 1 ndivc 0 thinthickness yes ndiva 2 ndivb 1 ndivc 0 thinthickness yes

.cs sigma 574e6 debug1 yes debug2 no .bz 0 0 0 9.9 0 0 0 0.1 0 9.9 0.1 0 0 0 2e-3 9.9 0 2e-3 0 0.1 2e-3 9.9 0.1 2e-3

.cs sigma 574e6 debug1 yes debug2 no .bz 10.1 0 0 20 0 0 10.1 0.1 0 20 0.1 0 10.1 0 2e-3 20 0 2e-3 10.1 0.1 2e-3 20 0.1 2e-3
```

Output

The (1)voltage-difference over the current source and (2)the frequency-vector is written to an output file for the C++ test. The C++ code also produces an SPICE syntax input file that is given to a Spice 3f4 version as input. The resonance frequencies for the two solutions are displayed in Figure 1.

Concluding remarks

- To be remembered, this report desribes the version 0.0 of the developed frequency domain PEEC solver. This means that the solver works but the operation of it is subjected to some 'fixing' in the C++ code.

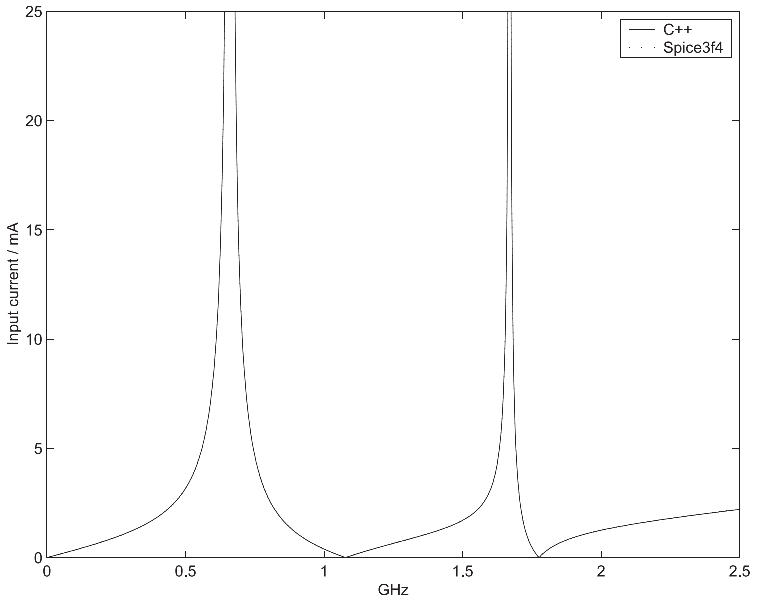

Figure 1: Resonance frequencies for dipole