Stamping a netlist

Historically the most well know circuit simulator is Spice, which had its origin in a Carnege Mellon University Graduate class taught by Ron Rohrer in 1968 (and UC Berkley in 1969-1970). Early versions were named CANCER, which was changed for obvious reasons.

The first Spice program was written by Larry Nagel as the basis for his Ph.D. dissertation. Spice, a public domain program, was later released by University of California in 1971.

In this homework you will learn to write a circuit simulator use Python. It won't be as functional or fast as the public domain and commercial versions available, but you will be able to add your on code which could be useful for special projects.

At the end of this exercise you will be able to:

- Read in a data file
- Use a stamping procedure to fill components into a matrix
- Solve these matrices
- Plot your result

Useful Reference:

Electronic Circuit and System Simulation Methods, Lawrence T. Pillage, Ronald A. Rohrer, and Chandramouli Visweswariah, McGraw Hill, New York, 1995.

As in circuit class we can use nodal analysis to solve a circuit to determine unknowns. Let's start with a DC solution. At DC inductors become wires and capacitors become opens, to first order.

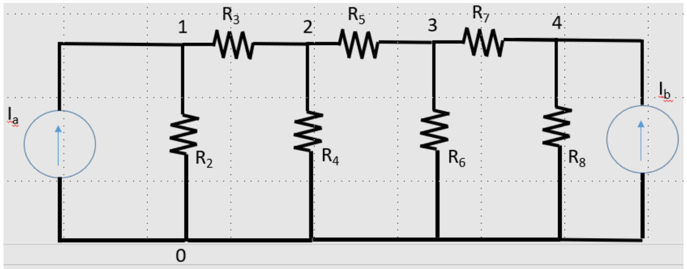


Figure 1

Doing nodal analysis of this circuit, the sum of the currents at each node must be 0:

$$-I_a + \frac{V_1}{R_2} + \frac{V_1 - V_2}{R_3} = 0$$
$$\frac{V_2 - V_1}{R_3} + \frac{V_2}{R_4} + \frac{V_2 - V_3}{R_5} = 0$$
$$\frac{V_3 - V_2}{R_5} + \frac{V_3}{R_6} + \frac{V_3 - V_4}{R_7} = 0$$
$$\frac{V_4 - V_3}{R_7} + \frac{V_4}{R_8} - I_b = 0$$

Figure 2

These equations can be grouped on current and placed in matrix form which gives:

$$\begin{bmatrix} \frac{1}{R_2}+\frac{1}{R_3} & \frac{-1}{R_3} & 0 & 0 \\ \frac{-1}{R_3} & \frac{1}{R_3}+\frac{1}{R_4}+\frac{1}{R_5} & \frac{-1}{R_5} & 0 \\ 0 & \frac{-1}{R_5} & \frac{1}{R_5}+\frac{1}{R_6}+\frac{1}{R_7} & \frac{-1}{R_7} \\ 0 & 0 & \frac{-1}{R_7} & \frac{1}{R_7}+\frac{1}{R_8} \end{bmatrix} \cdot \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} I_a \\ 0 \\ 0 \\ I_b \end{bmatrix}$$

Figure 3

An interesting pattern has developed. Each diagonal in the matrix contains the total conductance in each branch leading to that node. The off-diagonal terms contain the conductances between nodes. We immediately realize that we could write this matrix by inspection without doing the circuit analysis which suggests a Python program could do it. Instead let's develop an algorithm based on a netlist. As you remember netlists are just a list of components and their nodes. Here we use node numbers though names are also accepted in modern circuit simulators. This netlist describes the circuit fully. The first term is an instance name. If the instance name begins with R, it represents a resistor. If it begins with I, it represents a current source. Following the instance name are the nodes between which the element sits and then its value. We have made all the values 1 for this demonstration. (Note that for current and voltage sources, the first node is the **positive** node and the second node is the **negative** node! Similarly, the first node is the **from** node and the second node is the **to** node. This nomenclature will be important below.)

ISa 0 1 1
R2 1 0 1
R3 1 2 1
R4 2 0 1
R5 2 3 1
R6 3 0 1
R7 3 4 1
R8 0 4 1
ISb 0 4 1

We can build the matrix from this netlist by “stamping” the components into the matrix. The netlist is typically placed in a text file which is read by the circuit simulator. We can then read this netlist and fill in the matrix. This is called **stamping the components from the netlist into the matrix**.

To stamp a resistor from node i to node j into the admittance matrix, these terms are added to admittance matrix elements i,i, i,j j,i and j,j.

$$\begin{array}{cc} & \begin{array}{cc} \scriptstyle from\ i & \scriptstyle to\ j \end{array} \\ \begin{array}{c} from\ i \\ to\ j \end{array} & \begin{bmatrix} \frac{1}{R} & \frac{-1}{R} \\ \frac{-1}{R} & \frac{1}{R} \end{bmatrix} \end{array}$$

Figure 4

Independent current source I from node i, to node j are posted like this

$$\begin{array}{cc} \begin{array}{c} from\ i \\ to\ j \end{array} & \begin{bmatrix} -I \\ I \end{bmatrix} \end{array}$$

Figure 5

Note stamps are **added** to matrix and vector elements, they do not replace the element that is already there.

So taking the first line of the netlist which is a current, it is posted to the current column as $-I_a$ in row 0 and a I_a in row 1. The second line of the netlist is a resistor, it is posted to the matrix as $1/R_2$ in ij=00 and ij=11, and as $-1/R_2$ in ij=01 and ij=10. The rest of the matrix is filled out accordingly.

$$\begin{bmatrix} \frac{1}{R_2}+\frac{1}{R_4}+\frac{1}{R_6}+\frac{1}{R_8} & \frac{-1}{R_2} & \frac{-1}{R_4} & \frac{-1}{R_6} & \frac{-1}{R_8} \\ \frac{-1}{R_2} & \frac{1}{R_2}+\frac{1}{R_3} & \frac{-1}{R_3} & & \\ \frac{-1}{R_4} & \frac{-1}{R_3} & \frac{1}{R_3}+\frac{1}{R_4}+\frac{1}{R_5} & \frac{-1}{R_5} & \\ \frac{-1}{R_6} & & \frac{-1}{R_5} & \frac{1}{R_5}+\frac{1}{R_6}+\frac{1}{R_7} & \frac{-1}{R_7} \\ \frac{-1}{R_8} & & & \frac{-1}{R_7} & \frac{1}{R_7}+\frac{1}{R_8} \end{bmatrix} \cdot \begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} -I_a-I_b \\ I_a \\ 0 \\ 0 \\ I_b \end{bmatrix}$$

Figure 6

This is very similar to the matrix we had when we did the nodal analysis by hand. Note that choosing V_0 as ground, that is making all voltages referenced to V_0 we can eliminate row 0 (the top row) and column 0 (the first column) and the top entry in the column vectors. (For example, we can eliminate the first column since all elements of the column are multiplied by 0 and, therefore, don't contribute to the solution.)

This results in the same matrix as the hand nodal analysis.

$$\begin{bmatrix} \frac{1}{R_2}+\frac{1}{R_3} & \frac{-1}{R_3} & 0 & 0 \\ \frac{-1}{R_3} & \frac{1}{R_3}+\frac{1}{R_4}+\frac{1}{R_5} & \frac{-1}{R_5} & 0 \\ 0 & \frac{-1}{R_5} & \frac{1}{R_5}+\frac{1}{R_6}+\frac{1}{R_7} & \frac{-1}{R_7} \\ 0 & 0 & \frac{-1}{R_7} & \frac{1}{R_7}+\frac{1}{R_8} \end{bmatrix} \cdot \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} I_a \\ 0 \\ 0 \\ I_b \end{bmatrix}$$

Figure 7

This equation is of the form G V = I. Now, we simply need to supply a value for each of the components and known sources and then solve for voltage vector V by V = G^{-1} I. We can easily use Python to build the matrix and solve.

For AC analysis we may use Z, the impedance of a component, in the place of the resistance using the same procedure.

For an **independent voltage source**, the stamp is actually sort of complex. Consider V_{kl} where k is the positive node and l is the negative node.

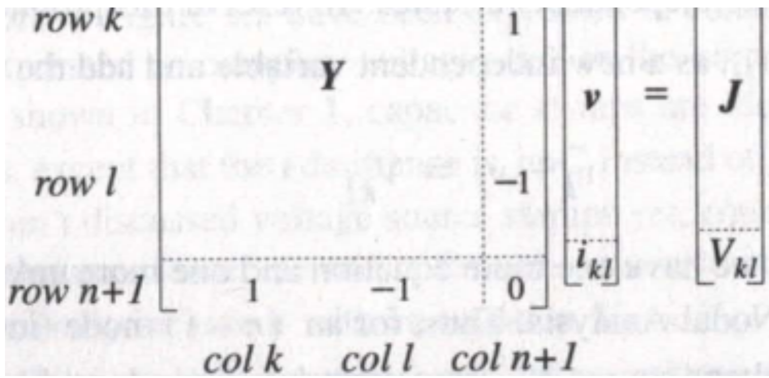
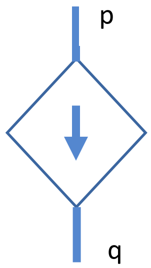


Figure 8

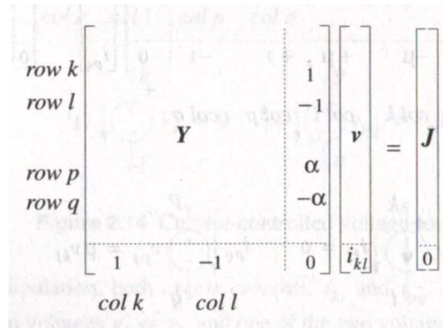
Ordinarily, this 0 on the diagonal might suggest a singular matrix, but the 0 can be eliminated by a row exchanges. If you do not eliminate node 0, the ground node, then voltages are relative to ground, which is set to value of 0. If you solve the matrix in figure 6 the voltage at node 0 is solved as well and the other voltages are relative to that value. Of course, this requires more space and time and doesn't add anything.

Note that if the row or column into which an entry should go is row or column 0, then that particular entry is not made.

For a **current-controlled current source**, the stamp is more involved.

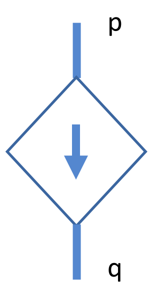


$i_{pq} = \alpha\, i_{kl}$

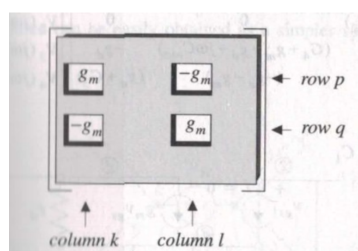


This diagram shows where terms are added to the Admittance matrix and to vectors V and J. Note that an additional column and row are added to the matrix, and an additional element to the v and J vectors. **Note that the assumption of a current controlled current source is that v_k-v_l = 0, which may not be the case in your problem, so you will have to add a low impedance series element to the circuit to detect the current.**

For a **voltage-controlled current source**, the stamp is as follows.



$i_{pq} = g_m\, V_{kl}$



This diagram shows where the value g_m is stamped into the Admittance matrix. Remember that when elements are stamped into the admittance matrix they are added to the element values already there.