**Matrix Multiplier by Prithvi P Rao , EE22B024**

## Algorithm for generating the matrix

The input program parses the .ckt file to generate a dictionary of dictionaries called connections. When we take connectionc[Node1][Node2] we get the details of all the elements between those 2 Nodes. Let us assume that the given .ckt file has `N` nodes including GND and `NV` be the number of Current Sources . We then generate 2 Martrices `A` and `B`, `A` being of a square matrix dimensions (`N`+`NV-1`) and `B` being a column matrix of the same size. The first `N-1` rows of `A` are obtained using Nodal analysis at all the nodes apart from `GND` . The equations and calculations involving `GND` are not part of the matrix as we are taking `GND` to be at 0 Volts always. The following `NV` equations are obtained by applying KCL around each Voltage source.

Nodal Analysis :- We parse through the connections made from the given node to take care of the R values. We initialise an array of zeroes of length (`N`+`NV-1`) .We subtract `1/R` from the `ith` entry wherein the `ith` entry is connected to the given node with a resistance `R`.For the term where the `ith` node is the given node itself , we again parse through all connections to the node and add up the resistances. Voltage sources are ignored here and taken care of in the current analysis. Current sources have appropriate coefficients added.

Current Analysis :- We initialise an array of zeroes length (`N`+`NV-1`) . The nodes which are at the end of this current source are replaced with 1 and -1 according to the direction of the current. If connection is with `GND` node , then no change needs to be brought about in the array.

This method of solving the matrices was possible due to my discussions with my friends Jatin , Bhuvan and Nikhil .

I used my own code for solving the matrices , which is sufficiently explained in the comments of the evalSpcie.py file Since it does give an innacuracy of order 10ˆ-7 in certain cases , I have used numpy.linAlgSolve as a primary method for solving the equations , but to handle the cases linAlgSolver can't handle , I've put a try expect block and the exception uses my algorithm for the same.

## Error Handling

test_invalid_file :- try except block to open the file test_invalid_element :- While parsing through .ckt file , if else statement is used test_malformed :- if we go out of bounds of the file while parsing through , it implies malformed circuit . Implemented using try except test_voltage_loop :- Each Node-Node pair is analysed for multiple voltage sources test_current_node :- Each Node-Node pair is analysed for multiple Current sources

Other test cases that I have handled include rejecting files which are not of the form `.ckt` . A pair of extra functions at the end , get currents and validate cross checks the credibility of the circuit by applying current analysis and ensuring that KCL is true at every Node , otherwise it implies that the circuit has no possible solution. I could not get it working perfectly due to time crunch though and have not used it.