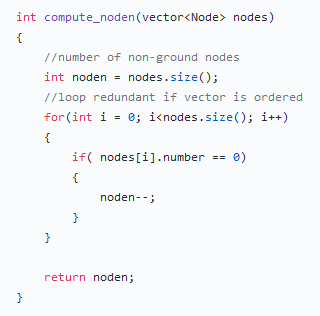
Sry if this is incoherent lol its 2 am

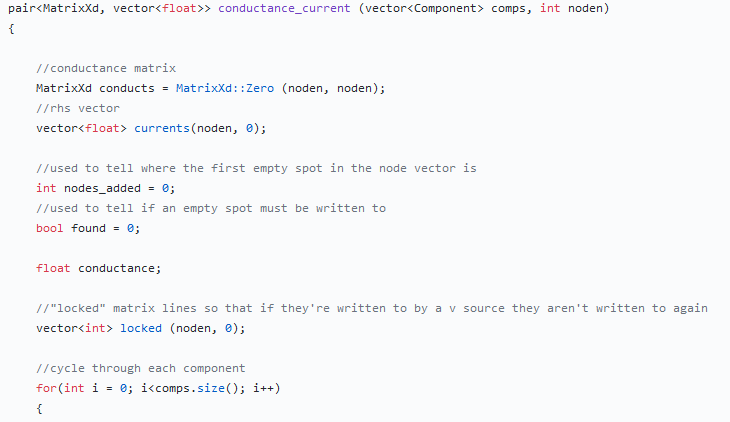
KCL equations: conductance matrix and right-hand side vector

For operating point and transient DC analysis as well as transient initialisation, we establish an equation Ax = b where x is the vector of mostly unknown node values, A is a matrix containing the variable terms of the equations describing the node voltages and b is the vector of known current and voltage source values. This equation is then solved using Jason’s matrixSolve function to yield the voltage at each node.

The left-hand side conductance matrix and the right-hand side current vector must be populated in order to create a series of equations describing the circuit. The dimensions of the square matrix and vectors are the number of non-grounded nodes, calculated with the Noden function.



The two matrix terms are produced by a conductance\_current function that cycles through each component and from each value contributes to the equation in accordance with the type.



Design one: processing resistors and voltage sources

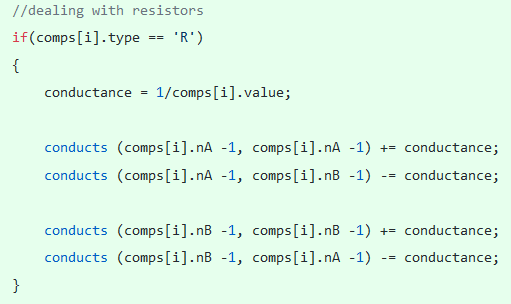
As explained previously, each row or column in the matrix and vectors corresponds to a non-grounded node in the circuit. As each node has a number, the row or column number is that of the node minus one. Each matrix row therefore contains either the coefficients of each node in a Kirchhoff’s current law equation or coefficients 1, 0 and -1 to describe the voltage of a node at the positive terminal of a voltage source.

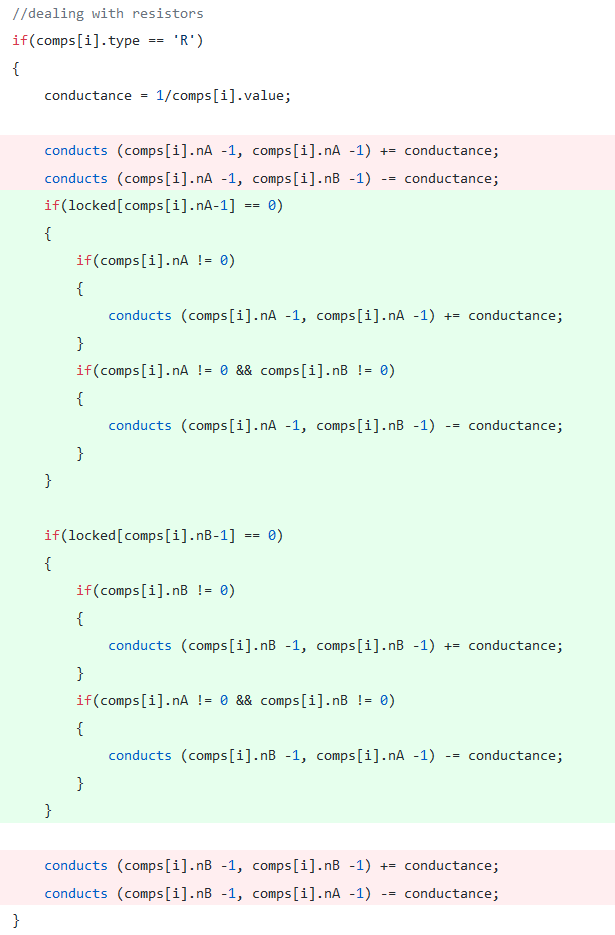
The coefficients of the node voltage in a given KCL equation are derived from the conductances of the resistors involved: the coefficient of the node at which nodal analysis is performed is the sum of all resistor conductances and the others have a coefficient of minus the conductance of the resistor separating them from the first node.

After initialising a zero-matrix using the Eigen library, we can simply read each resistor value and add or subtract its conductance to an element in the matrix. The row of the element is that corresponding to the node at which nodal analysis is being performed and the column that corresponding to the node whose coefficient is being added or subtracted.

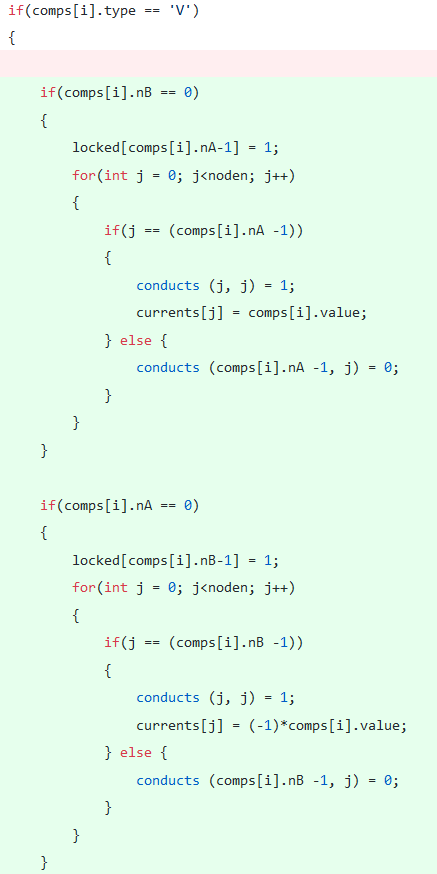
After initialising a zero-matrix using the Eigen library, we can simply cycle through each resistor and add or subtract its conductance to an element in the matrix. The row of the element is that corresponding to the node at which nodal analysis is being performed and the column that corresponding to the node whose coefficient is being added or subtracted.

As each resistor is connected to two nodes, the conductance can theoretically be added and subtracted to two rows (red). In practice, as the voltage at ground is already known there is no matrix row or column corresponding to node 0. Therefore, each modification to the matrix is subject to the node number no being 0.



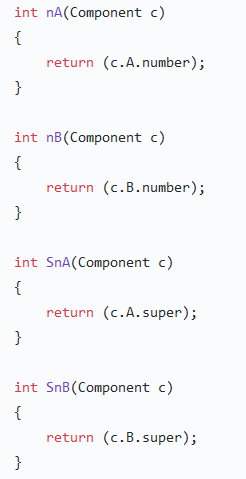


The A node of a voltage source always corresponds to the positive terminal, so the source can be described with the equation A = B + V where V is the source voltage. In our circuit equation this translates to a row matrix row 1\*Va + (-1)\*Vb + 0\*others and V in the same row of the rhs vector (current vector). To prevent this row from being written over by the resistor code, a vector of bools represents which rows are locked, blocking them to further editing as seen above. If a terminal is grounded, then only one matrix element is written to, as implemented below.

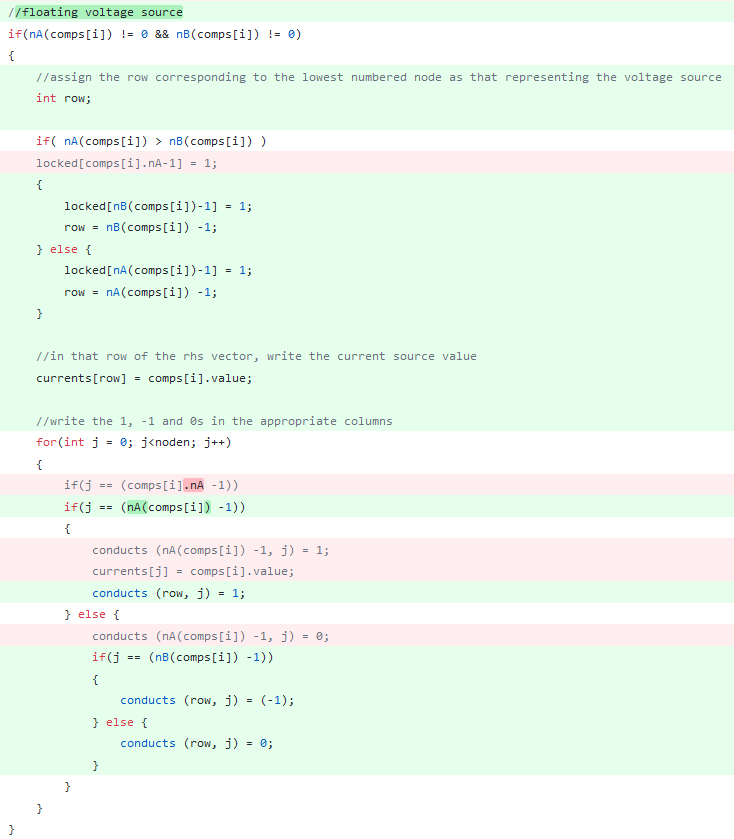


In the case of grounded voltage sources, a node value is already known so there is no point in performing nodal analysis on the node across the voltage source from ground. This is not the case for floating voltage sources which force a relationship between node voltages, creating a super-node, but do not a node voltage value. We’ve decided to implement this by using one row to represent the relationship between nodes (rephrase) and one to represent the super-node KCL equation, or rather the sum of the terms of the equations for the two nodes of the source.

We’ve arbitrarily decided that the higher numbered row will be the former and the lower valued row the latter. Jason has implemented this by adding a node and super-node value, equal if there is no floating voltage source, to each node struct. For simplicity, these are accessed with the functions below:



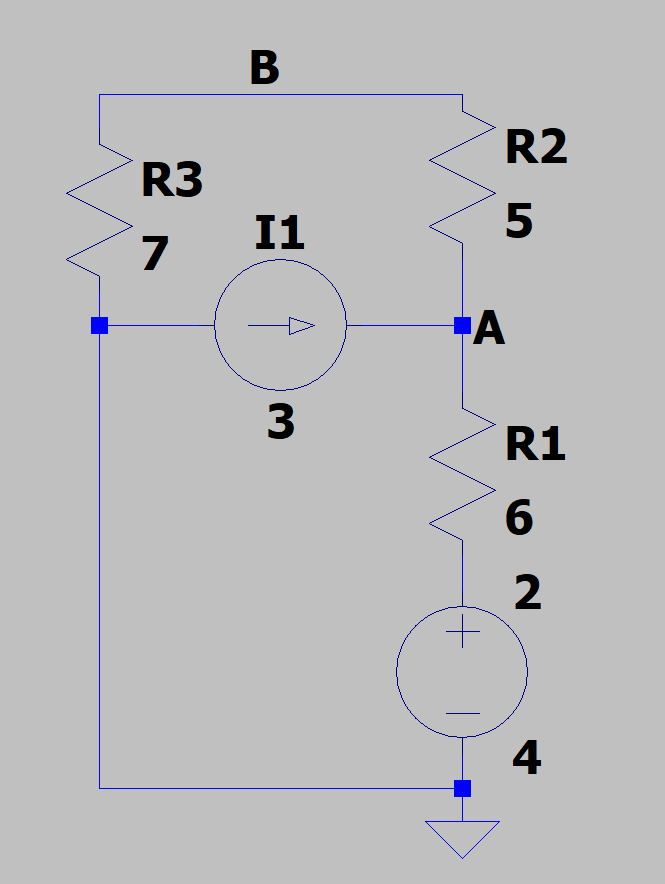
This arbitrary convention and its implementation in the Node struct allow us to obtain the required matrix with relative ease. Floating voltage sources function roughly the same as their grounded equivales but write -1 to one matrix element. As the super-node number of a node is equal to the regular node for non-super-nodes and to the row to which we write the KCL equations for super-nodes, we simply substitute the node number for the super-node number in the row of the element to which conductances are added or subtracted.





Design two: processing current sources

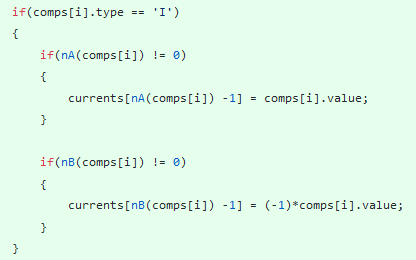
So far, we’ve only used the right-hand side, or current, vector to represent source voltages as our KCL equations terms were all multiples of node voltages. However, nodes connected to current sources have in their KCL equations constant terms of magnitude equal to the source currents:



Node A of this circuit yields the KCL equation:

(VA - VB)/5 + VA/6 – 3 = 0 => (VA - VB)/5 + VA/6 = 3

This is easily implemented by an if statement that adds and subtracts the source current value to the rows corresponding to the non-ground nodes the source is connected to.



Note: this early version incorrectly assigned instead of adding currents and the negative sign was applied to the wrong row. A locked condition was also omitted.