Currents:

Our program must compute the current through each component, both to satisfy the functional requirements and later to model the behaviour of capacitors, which are required to function as current dependant voltage sources during AC operation. We have seen how, with the functionality implemented up to this point, we can for a given operating point obtain the voltages at each node of a circuit made exclusively of resistors and current and voltage sources. We will therefore use these node values and component information to compute the current through each component at a given operating point. These values will be returned as a VectorXd, the vector of doubles type provided by the Eigen library.

For the purpose of calculating currents, we divide components into two types. The first is made up of those with currents that can be directly read or immediately computed from node voltages and component characteristics. This group comprises components functioning as current source and resistors. The second type of component corresponds to those functioning as voltage sources. The current through these components is dependant on that flowing in the rest of the circuit.

(insert recursive currents)

(insert easy components)

As per Kirchhoff’s current law, the sum of currents flowing in or out of a node is 0. This means that we can deduce the current flowing through a voltage source recursively by looking at those in other components sharing a node with the component. (do we have a diagram for this from the slides?) The function “recursive\_basecase” used for this will need to return a float current value; and if necessary, determine which component shares a node with the relevant component, find their currents, and add them up with the correct sign.

(insert more complicated components)

As the number of voltage source-like components in a circuit increases, so does the odds that recursion will be employed multiple times, possibly to read the same currents repeatedly. To avoid wasting time doing this, we use a vector called “computed”, passed as a reference to recursive\_basecase, which tracks which component currents have already been computed, allowing us to simply read them after their computation. To be legible, both this vector and the vector of currents need to be ordered identically to the vector of components. However, as the function is called recursively, we do not immediately know the index of all components it is called on. As such, we will need a function that parses through the vector of components to find the component’s index. For this measure to work, we need the current vector comp currents to also be passed as a reference so that currents calculated recursively can be read later one. Further simplification can be obtained by not computing a current in recursive\_currents if computed is true at that index.

To avoid infinite recursion from two voltage sources facing each other and repeatedly attempting to read each others’ current, a “used node” is passed as an argument of recursive\_basecase. This is the node A around which the function will look for attacked components, and when it is recursively called on another component, it is called with the other node B of the adjacent component.

Transients:

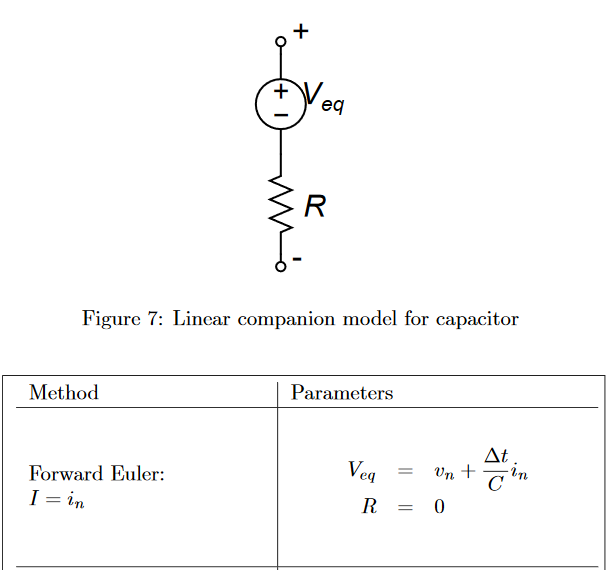
Transients in our simulations are handled as a series of operating points, with sinusoidal voltage source values computed at each time step. Under AC conditions, reactive components behave differently than in DC operating points where capacitors form open circuits, and are therefore ignored in the conductance matrix, and inducts are open circuits or 0 value voltage sources. As we treat node voltages discretely at each time step rather than as time dependant functions, we must approach reactive components with numerical methods rather than the analytical (?) ones as seen in lectures.

As reactive components are modeled as sources, only the right-hand side vector of the node voltage equation changes with each timestep. For the sake of efficiency, we create a single matrix with the function MatrixUpdate and then use the function VectorUpdate at each timestep handled by the transient function to update the equation before solving for node voltages. It is worth noting that the transient function assumes AC operation in the way it treats reactive components. For this reason, the program calculates a first timestep with the function no\_prior\_change, which uses the function conductance\_current and therefore assumes that all node voltages have previously been at 0 for an infinite duration of time.

(insert screenshots)

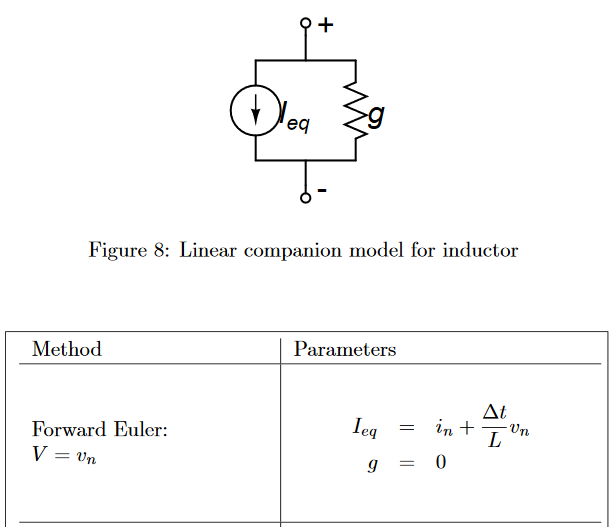
Initially, we used the Forward Euler method, which is convenient because it functions with 0 value resistors and thus allows to save time with a somewhat quicker construction of the conductance matrix.

Forward Euler for capacitors:



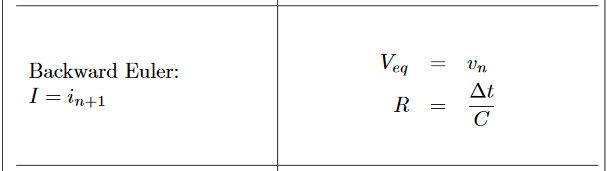
From <http://dev.hypertriton.com/edacious/trunk/doc/lec.pdf>

Forward Euler for inductors:

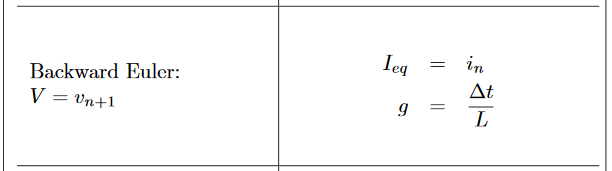


However, Forward Euler is also relatively inaccurate. Although this method provides adequate (show or reference tests) values for inductor or capacitor circuits, in LC circuits (show or reference tests) the error compounds and values shoot off to infinity. Modelling DC inductors or AC capacitors as simple voltage sources poses another problem as reactive components can be placed in parallel, unlike real voltage sources. Such a configuration triggers infinite recursion as neither component’s current is defined solely as a sum of knowable values. The solution is to switch to Backwards Euler, increasing accuracy and introducing resistors that allow for currents across capacitors to be instantly calculated. (show)

Backwards Euler for capacitors



Backwards Euler for inductors



Placing resistors in series with capacitors requires the creation of new node, which is accomplished by the input functions. (is this expanded on later?)

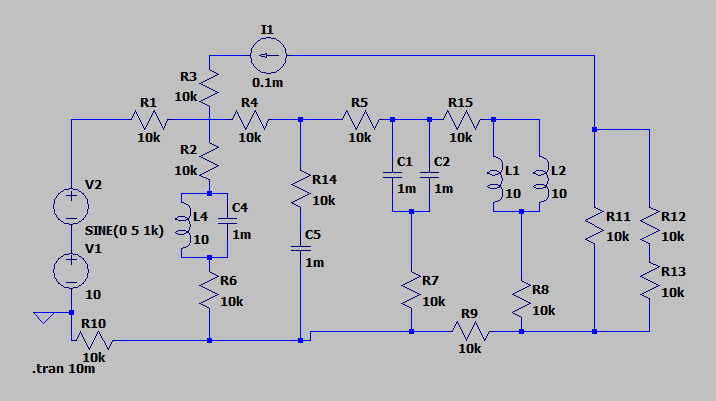
To resolve infinite recursion caused by inductors in DC conditions, we treat them as resistors. Tests performed in LTSpice (show) show that under DC conditions inductors behave like 1mΩ resistors. This new representation is implemented in conductance\_current and recursive\_basecase and gives accurate figures in operating point simulations. (show)

Final change to currents:

These changes to the representation of reactive components means that only true voltage sources require recursion to compute their currents. As the use of the computed vector requires functions that parse the component vector, which can be quite long for large circuits, and circuits rarely contain many voltage sources; tracking whether each current has been computed is unlikely to be a good design decision.

Two versions of the recursive\_basecase have been elaborated, with and without tracking, and tested on a circuit. The chrono library is used to measure the execution time of functions. Five tests are performed each time to account for variation in the time taken to execute the program.

Test circuit:



|  |  |  |
| --- | --- | --- |
| Test | With tracking | Without tracking |
| 1 | op duration: 609  tran duration: 5290216  main duration: 5292786 | op duration: 549  transient duration: 5014194  main duration: 5016483 |
| 2 | op duration: 643  tran duration: 5444769  main duration: 5447404 | op duration: 606  transient duration: 5127834  main duration: 5130410 |
| 3 | op duration: 533  tran duration: 5324080  main duration: 5326489 | op duration: 705  transient duration: 5200117  main duration: 5203535 |
| 4 | op duration: 561  tran duration: 5411750  main duration: 5414247 | op duration: 497  transient duration: 5031532  main duration: 5033857 |
| 5 | op duration: 509  tran duration: 5210722  main duration: 5212990 | op duration: 443  transient duration: 5015776  main duration: 5018199 |

Results of the timed tests

As we can see, despite the circuit containing series voltage sources, entailing double recursion, the version of the program that uses tracking is consistently slower than the version that does not. In consequence, the tracking feature is removed from the program. (perhaps make a better test where the voltage sources are connected to more components.)