Convergence of Saturated Magnetic Models in Gnucap Circuit Simulator

Telford Tendys
<telford@triode.net.au>

1 Introduction

This article gives a brief summary of nonlinear modeling in the "gnucap" circuit simulator and examines an alternative convergence strategy which improves stability for sharply saturating circuit components.

1.1 Some Circuit Simulation Background

The history of computer modeling of electronic circuits probably goes starts at [1] which might have been the first attempt to systematically apply matrix analysis in an attempt to design a universal circuit solver. Early attempts to use digital computers to simulate analog circuits go back to the early 1960's. However it was the Spice simulator that became the most well known and widely available tool [2], [3].

Over the years, other simulators have moved into the arena. Many of them are based on the original Berkeley Spice with more or less tweaks and modifications, some additional features and extended component libraries. Fancy graphical front-ends have proliferated in an attempt to make the tools easier to use but these are unrelated to the actual circuit solving algorithm.

For the most part, these offshoots of the original Spice code have been absorbed into proprietary products with secretive source code and undocumented internals. These may prove useful in the short term for a user who wants to concentrate on circuit development and who doesn't really care what goes on inside the simulator but in the long term, continuing development requires that researchers have the opportunity to view and modify the simulator source code.

1.2 Other Simulators (e.g. APLAC and others)

The APLAC simulator is an example of a simulator that developed in parallel to but independent of SPICE. It started in 1972, was originally designed for RF circuits, then later extended to support generic simulation. Development of APLAC has been centred around Finland. Unfortunately, source code for APLAC is not generally available.

PSIM is another program, in principle it is similar to SPICE but it uses different internal algorithms and was developed independently to SPICE. Once again, source code to PSIM is not generally available.

1.3 NGSPICE: Modern Descendent of Berkeley Spice

The "ngspice" project has taken the last official spice source released by Berkeley (i.e. spice3f5) and collected various snippets and patches to the original Spice code, merging those into the Spice base to make a more up-to-date codebase. This code is still being released under a BSD style license but the developers live in the hope that eventually they can convince the spice3f5 copyright holders to remove the "advertising clause" from the BSD license so that ngspice may finally be released as a GPL package.

1.4 Gnucap the GPL Circuit Simulator

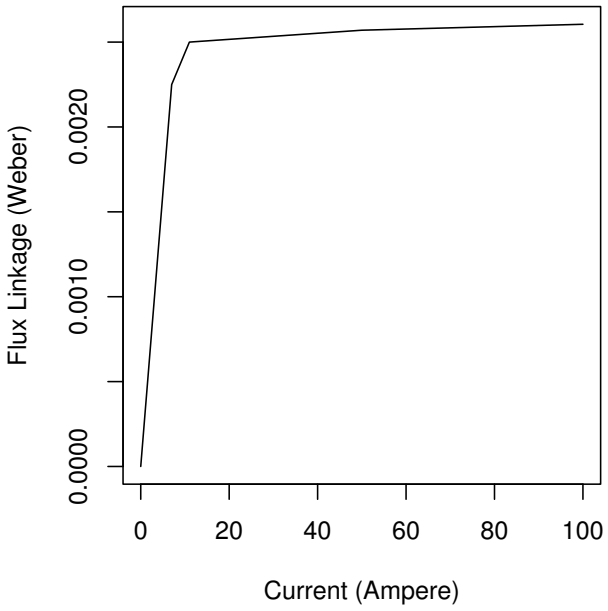
An alternative project is "gnucap" (previously known as "acs" or "Al's Circuit Simulator"). This is a largely Spice-compatible simulator that is a completely new codebase and which uses more flexible internal algorithms to the original Spice. The guncap simulator is licensed under GPL which prevents the source code from ever being stolen and thus guarantees that all future gnucap versions and updates will be available for research and teaching use as well as for commercial circuit development. The gnucap source code is always available for modification and improvements as well as for study.

Using patches available from `<http://www.triode.net.au/~telford/gnucap/>`, a modified version of gnucap may be built containing modifications which add the "PHI()" probe for inductors and the "STRATEGY=KNEECHORD" option which improves the numerical stability of the gnucap matrix algorithm. The standard gnucap simply uses Newton's method (i.e. use the first derivative of the curve) which is fast to calculate and converges rapidly but in some situations it is very unstable.

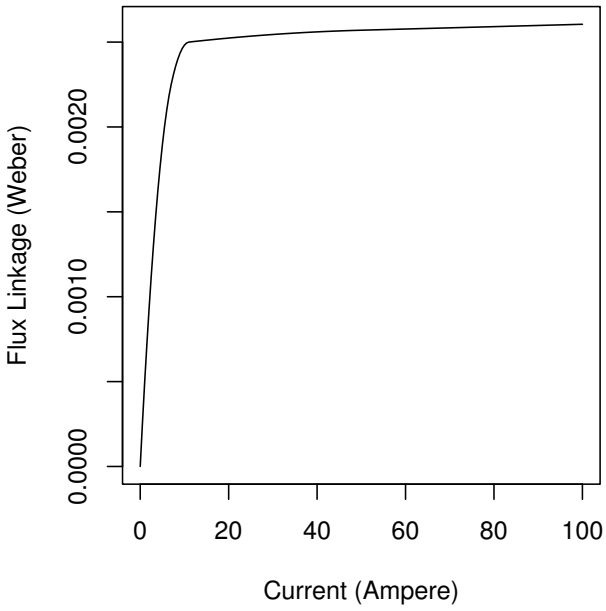
2 Nonlinear Components in Gnucap

In the common Spice versions the simple base components (i.e. resistor, capacitor, inductor) are linear. Gnucap provides an alternative approach by allowing a component value to be either a constant (i.e. a linear component) or a function. Specifying a function allows the user to define a relationship between voltage and current in a resistor, voltage and charge in a capacitor or flux-linkage and current in an inductor.

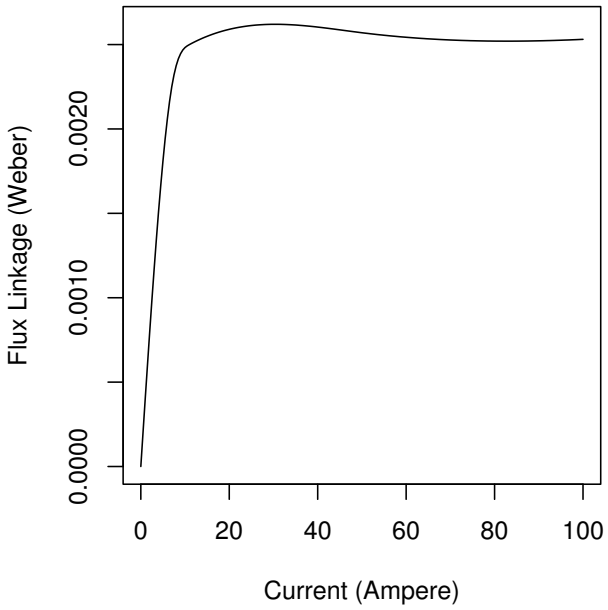
Various types of function may be defined, in this article the "FIT" function is examined since this is probably the easiest to use and most flexible. Any sample points may be given and a piecewise curve fit is used to model the component behaviour.



(a) Inductor Model, Linear Interpolation



(b) Inductor Model, Quadratic Interpolation



(c) Inductor Model, Cubic Interpolation

Figure 1: Sweep of Nonlinear Inductor

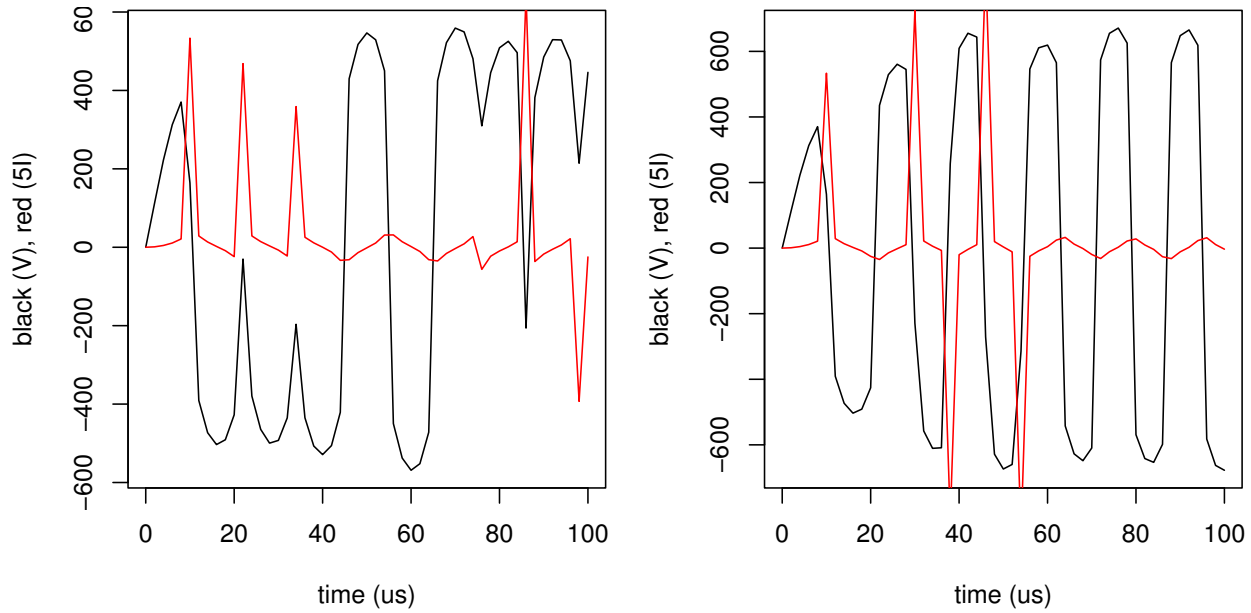
2.1 Sweep of a Nonlinear Inductor

Thi very simple circuit is a current source driving a nonlinear inductor. Using the DC sweep function results in a plot of the inductor characteristic and demonstrates the capability to model nonlinear components by merely providing a series of characteristic values.

```
SWEEP OF NONLINEAR INDUCTOR
ISUP    0       1       1
R1      1       2       1
L1      2       0       FIT -150,-2640u
+                           -50,-2570u
+                           -11,-2500u
+                            -7,-2250u
+                             0, 0
+                             7, 2250u
+                            11, 2500u
+                            50, 2570u
+                           150, 2640u ORDER=2
.print dc I(L1) PHI(L1)
.dc Isup 0 100 0.5 > sweep.dat QUIET
.end
```

Anyone familiar with SPICE will recognise most of the circuit description, it is not very complicated. Note that the use of the "FIT" function allows L1 to be defined as nonlinear.

When placing probes on a component "$I()$" gives the current through the component and "$PHI()$" will give the flux-linkage of an inductor.



(a) Standard Strategy

(b) Kneechord Strategy

Figure 2: Nonlinear Simulated Resonance, 2000ns Steps

Another gnucap feature that is a bit different to SPICE is the redirection of the output to a data file, making it easy to post-process the results.

However, this style of curve fitting has some limitations. In particular, piecewise cubic splines will often overshoot as can be seen in figure 1. The user may select the interpolation by using the "ORDER=" parameter, for the later examples quadratic interpolation was chosen (ORDER=2) because it avoids overshoot while still providing a reasonably smooth curve.

3 Resonant Circuit with a Nonlinear Inductor

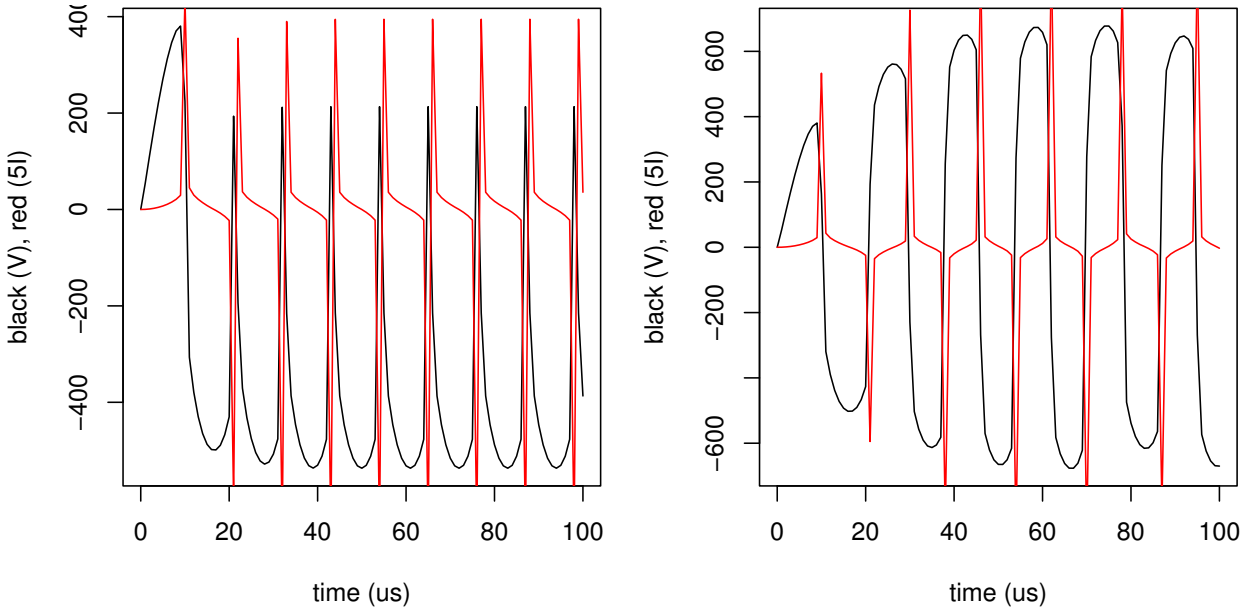
This example shows the same inductor as part of a second order filter. It demonstrates the effect of inductor saturation (the voltage wave becomes closer to a square wave while the current wave gets spiky peaks).

3.1 Circuit Text

```
SECOND ORDER RESONANT CIRCUIT DRIVEN BY CURRENT SOURCE
ISUP    0       1       GENERATOR(1)
R1      3       0       0.5
R2      2       0       0.1
L1      1       3       FIT -150,-2640u \
                             -50,-2570u \
                             -11,-2500u \
                              -7,-2250u \
                               0, 0     \
                               7, 2250u \
                              11, 2500u \
                              50, 2570u \
                             150, 2640u ORDER=2
C1      1       2       100n
.option STRATEGY=KNEECHORD
.generator FREQUENCY=0 MAX=6 MIN=0 RISE=0 FALL=0 INIT=0 DELAY=100n WIDTH=9u
.print tran V(C1) I(L1)
.tran 0 100u 2000n >res.dat QUIET
.end
```

The ".generator" directive invokes a general purpose function generator with various settings given, in this case it is used merely to generate a single pulse. Note that the unix-shell-style line continuation technique is acceptable as well as the more traditional FORTRAN style used above.

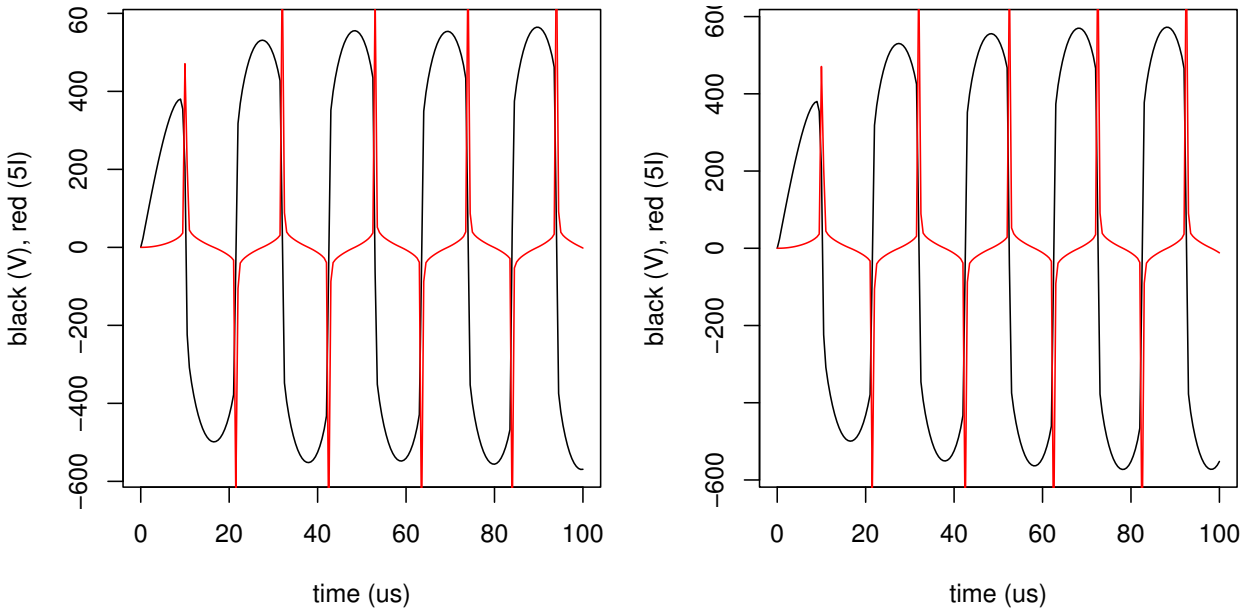
The "STRATEGY=KNEECHORD" is a special addition that activates an alternative convergence strategy. Details of how the "Kneechord" algorithm works are explained in the source code but in brief it involves picking a pair of points and drawing a chord between them. This is similar to the common secant method but the two points are recalculated with each iteration.



(a) Standard Strategy

(b) Kneechord Strategy

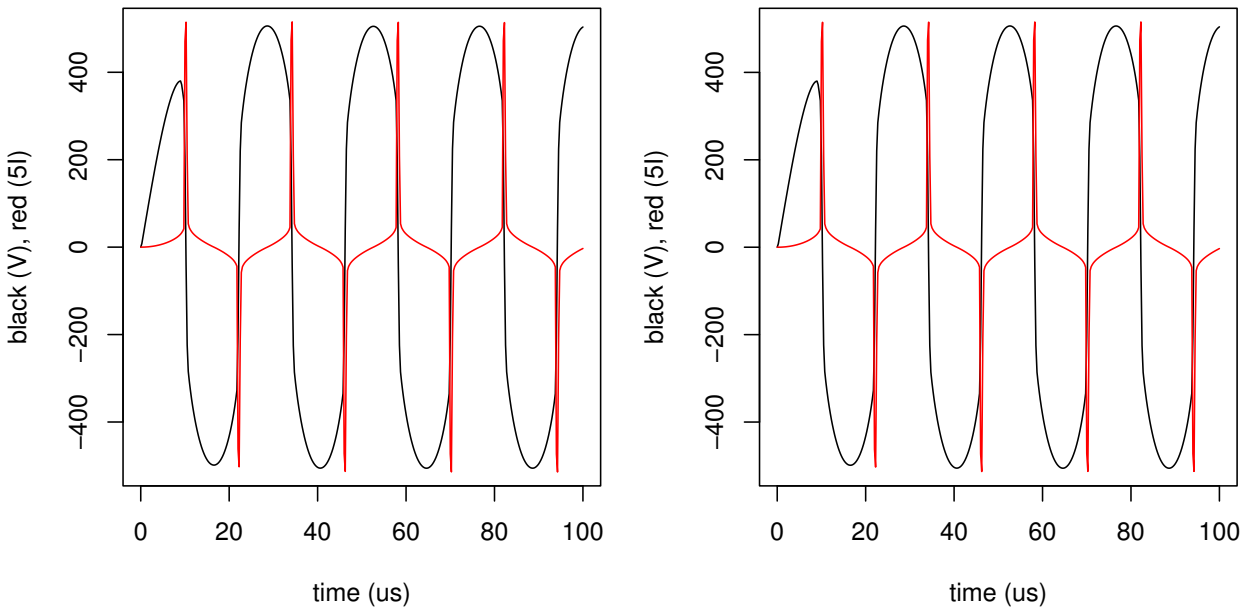
Figure 3: Nonlinear Simulated Resonance, 1000ns Steps



(a) Standard Strategy

(b) Kneechord Strategy

Figure 4: Nonlinear Simulated Resonance, 500ns Steps



(a) Standard Strategy

(b) Kneechord Strategy

Figure 5: Nonlinear Simulated Resonance, 250ns Steps

3.2 Results of Resonant Circuit

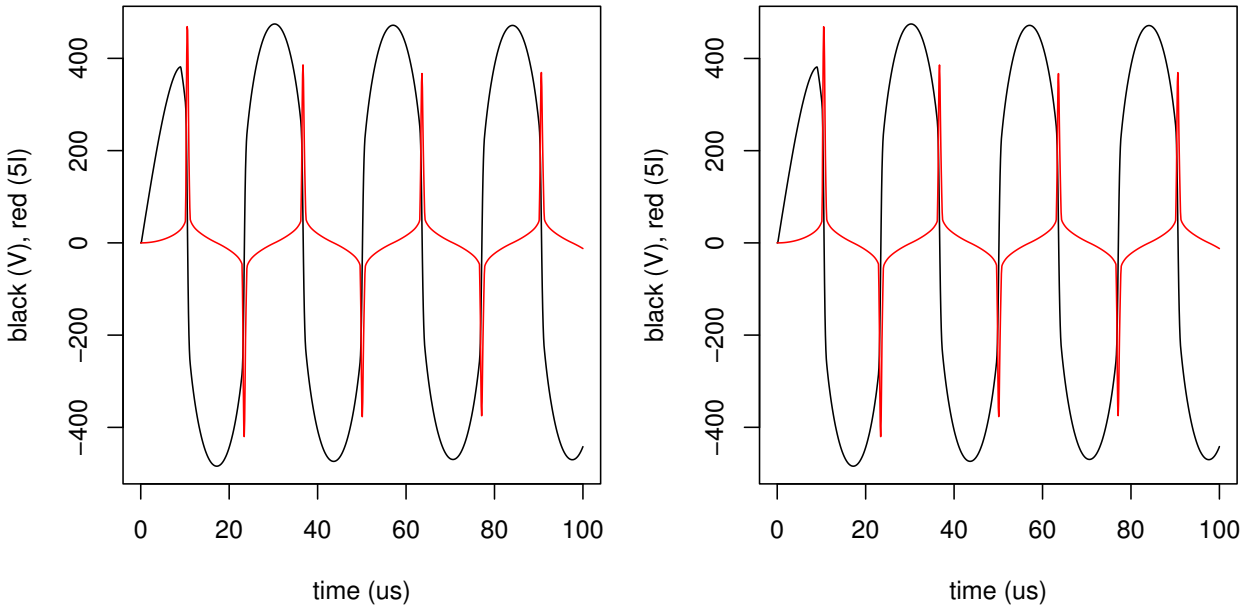
A series of tests was done with the "Kneechord" strategy and the standard strategy for the same circuit with smaller and smaller stepsize (see figure 2 through to figure 6).

From this it can be observed that the standard strategy works well under good conditions (with small step-size, the operating point is never far from the nonlinear component model so the system only needs to converge a small distance). However, if large steps are taken, the standard strategy might give results that are not even qualitatively similar to the true values. This demonstrates the lack of stability under difficult conditions.

In comparison, the "Kneechord" strategy always gives something reasonable, even though it is less accurate at larger step sizes (only to be expected because integration accuracy is reduced) it never produces completely unexpected artifacts.

3.3 CPU Usage Penalty

Unfortunately the "Kneechord" strategy requires additional CPU cycles. In the best possible case, it reverts to Newton's Method anyhow if the operating point converges in one cycle. Even in this case it requires one extra evaluation of the component modeling function. In worse cases it may require many evaluations and thus impose a considerable overhead in terms of extra CPU cycles. It should be pointed out that this overhead is ONLY imposed on nonlinear components and in any realistic circuit only a small fraction of the components will require nonlinear models. In addition, the option may be switched off once the user is confident that the simulation is producing reasonable results. Finally, there may exist situations where the standard algorithm is not sufficiently robust even at small step size.



(a) Standard Strategy

(b) Kneechord Strategy

Figure 6: Nonlinear Simulated Resonance, 100ns Steps

4 Conclusions

The gnucap circuit simulator contains some interesting features but the stability of its convergence is less than ideal. Because gnucap is open source software, it attracts researchers and developers who wish to experiment with advanced techniques, thus patches and improvements are gradually incorporated into the gnucap codebase.

One such contribution is the “Kneechord” alternative strategy for converging nonlinear component models. It has been demonstrated to be more stable than the standard strategy (Newton’s Method).

References

[1] G. Kron, *Tensors for Circuits*. Dover Publications, 1959.

[2] L. W. Nagel and D. O. Pederson, “Dc, ac, and transient analysis, nonlinear models,” in *Proc. Sixteenth Midwest Symposium on Circuit Theory, Waterloo, Canada, Memorandum No. ERL-M382*, Electronics Research Lab., Univ. of Cal., Berkeley., Apr. 1973.

[3] L. W. Nagel, “Spice2: A computer program to simulate semiconductor circuits,” *Tech. Rep. ERL M520, Electronics Research Laboratory Report, University of California, Berkeley, Berkeley, California*, May 1975.