Simulation of Guitar Effect Circuits

Johnathon Caguiat, Phillip Christian, Xiao Han Dong, and Ciaran Geaney

***Abstract*—Circuit solvers that incorporate both transient and nonlinear elements are of great importance in the area of modeling guitar effects. In this paper we are creating a circuit solver that is able to simulate an input wave file to an output wave file that will have similar results to that of an actual guitar effect if the same circuit were used. We accomplish this by implementing techniques that are relevant to solving transient and nonlinear circuits. Our circuit solver was able to effectively model certain guitar effects, however problems arose depending on the circuit schematic -specifically with regard to the number and nature of nonlinear elements within. Fixes were found to manage a limited number of nonlinear elements, however further research is needed to effectively model circuits that have many interacting nonlinear elements.**

I. Introduction

CIRCUIT modeling has had widespread use in the area of analog electronic audio component development. Modeling analog audio components enables the creation of software that is capable of simulating vintage effects such that devices can be made at a lower cost. These programs can also be embedded into portable devices such as a phone or a laptop, and used in real-time with actual instruments. Though, there are several musicians who believe that digital implementations of these effects sound inferior to the original analog pedals. Research into accurate simulations has allowed for minimal differences between the digital models, and the actual effect pedal. Special guitar timbre can also be created synthetically using various guitar effect pedals since circuitry of these effects are well known.

In this paper, we report our attempt to create simulation software to digitally simulate the guitar effects; an input sound file would create an output sound file with the sound enhanced by the guitar effects. A circuit solver capable of solving time-varying circuits with nonlinear components, such as diodes and transistors, is developed for this purpose. Additionally, variable time-stepping, a common technique used in commercial SPICE software, is implemented.

II. Circuit Solver

A. Systems of Equations

The matrix formulation of a circuit with dynamic and nonlinear elements can be found from Modified Nodal Analysis (MNA) to be

$$Gx(t) + C\dot{x}(t) + F'(x,t) = Bu(t) \tag{1}$$

$$y(t) = L^T x(t) \tag{2}$$

where x is the vector of voltages at each node as well as currents flowing through voltage sources and inductors, G is the nodal matrix of passive linear circuit elements. C is the nodal matrix of capacitors (and when relevant, inductors), F' is the vector of nonlinear elements, and $Bu(t)$ is the input excitation. In this paper, only diodes and BJT transistors using Ebers-Moll model are considered.

$F'(x)$ contains each rows nonlinear components. Alternatively, $F'(x)$ can be re-written as:

$$F'(x) = Hg(x) \tag{3}$$

Where H is a matrix containing the stamps of individual nonlinear element (i.e. $F'(x)$ using just ones within a column associated with each element) and $g(x)$ contains the nonlinear functions repsective to the columns in H. This alternative is helpful for coding, where $F'(x)$ allows the derivations to be more concise. B contains the stamp of the inputs nodes of inputs sources, voltage or current, where their values are entered in $u(t)$ and modified for each time step as necessary.

B. Merging Dynamic and Nonlinear Elements

To solve the system of equations, we need to combine Trapezoidal Rule and Newtons method. The first step is to derive the Trapezoidal rule for the system that we have which includes the nonlinear elements. The second would then be the application of Newtons method to manage the nonlinear elements.

The Trapezoidal rule relies on the following approximation of the first derivative:

$$\frac{x_{n+1}(t) - x_n(t)}{\Delta t} = \frac{\dot{x}_{n+1}(t) + \dot{x_n}(t)}{2} \tag{4}$$

Adding the $x_{n+1}(t)$ and $x_n(t)$ to (1):

$$\begin{aligned} G(x_{n+1}(t) + x_n(t)) + C(\dot{x}_{n+1}(t) + \dot{x}_n(t)) + \\ F(x_{n+1}(t) + F'(x_n(t) = B(u_{n+1}(t) + u_n(t)) \end{aligned} \tag{5}$$

Substituting the approximation of the first derivative and dividing the equation by 2:

$$\begin{aligned} \left(\frac{G}{2} + \frac{C}{\Delta t}\right) x_{n+1} + \left(\frac{G}{2} - \frac{C}{\Delta t}\right) x_n + \frac{F(x_{n+1}(t)) + F'(x_n(t)}{2} \\ = \frac{B(u_{n+1}(t) + u_n(t))}{2} \end{aligned} \tag{6}$$

The consequence is that we now have a nonlinear equation where $x_{n+1}(t)$ cannot be analytically solved given any $x_n(t)$

In order to use the general form of Newton's method to continue, we need to rearrange the trapezoidal rule into a format that can be solved using Newton's. Recall Newton's:

$$x_{k+1} = x_k - [J(F(x_k))]^{-1} F(x_k)] \tag{7}$$

Rearranging (6) such that it is some $F(x) = 0$, $F(x)$ would be:

$$F(x_k) = \left(\frac{G}{2} + \frac{C}{\Delta t}\right) x_k + \left(\frac{G}{2} - \frac{C}{\Delta t}\right) x_n + \frac{F(x_k(t)) + F'(x_n(t))}{2} - \frac{B(u_{n+1}(t) + u_n(t))}{2} = 0 \quad (8)$$

Now for the Jacobian, $J(F(x_k))$:

$$J(F(x_k)) = \left(\frac{G}{2} + \frac{C}{\Delta t}\right) + \frac{J(F'(x_k))}{2} \quad (9)$$

Because x_n, $F'(x_n)$ and $u_{n+1} + u_n$ are constants at a given time step.

C. Circuit Elements and their Stamps

1) Diodes: Diodes are electrical elements that enable current to flow, almost exclusively in one direction. As a consequence, attention to the orientation of a diode in a circuit is necessary to understand its behaviour in a circuit. For the purposes of modeling, the standard netlist input requires specific inputs for the nodes, like with voltage and current sources:

```
D[name] N+ N- MODName
```

Where MODName is a text qualifier for a given set of parameter values needed to calculate the nonlinear relationship between nodes n_1 (N_+) and n_2 (N_-). The current flows from n_1 to n_2. Consequently, the diode adds current to n_1 and subtracts from n_2, as per the algebraic definition of current direction in the Nodal Analysis paradigm. The current for any given time follows the following relationship:

$$i = I_0(e^{\frac{V_+ - V_-}{\eta V_t}} - 1) = I_0(e^{\frac{v_2 - v_1}{\eta V_t}} - 1) \quad (10)$$

Where I_0 is the saturation current and η is the quality factor, both of which are properties of the diode. V_t is the thermal voltage kT/q where k is the Boltzmann constant, T is the ambient temperature, and q is the charge on an electron. V_t is approximately $26\,\mathrm{mV}$ at room temperature. All the values but v_1 and v_2 are constants. From the netlist file, the following default values are used for the diode constants: I_0 = $1 * 10^-14$,η= 1, and $V_T = 25.85\,\mathrm{mV}$.

The diode stamp in $F'(x)$ is shown in (11). Alternatively, the stamps in $Hg(x)$ would look like (12).

$$\begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{c} \end{array} \\ \begin{bmatrix} i_D(v_1, v_2) \\ -i_D(v_1, v_2) \end{bmatrix} \end{array} \quad (11)$$

$F'(x)$

In order to solve the system, with elements like diodes, it is necessary to know the stamp of the Jacobian as well. The stamps for $F'(x)$ and $Hg(x)$ are the same as a Jacobian can only be applied to a column matrix of functions. (13) is the Jacobian of $F'(x)$ and $Hg(x)$

$$\begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} +1 & \end{array} \\ \begin{bmatrix} & \\ -1 & \end{bmatrix} \end{array} \begin{bmatrix} i_D(v_1, v_2) \end{bmatrix} \quad (12)$$

H g

$$\begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{bmatrix} \frac{\partial i_D}{\partial v} & -\frac{\partial i_D}{\partial v} \\ -\frac{\partial i_D}{\partial v} & \frac{\partial i_D}{\partial v} \end{bmatrix} \end{array} \quad (13)$$

$J_{F'}$

2) Transistors: The transistor model that was used for our circuit solver is the Ebers-Moll model, which attempts to recreate an electrical model of a transistor using two diodes whose currents are determined by (10).

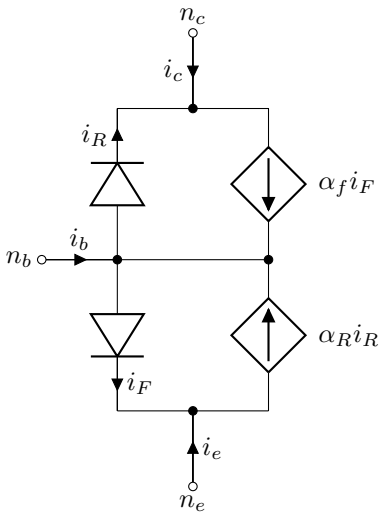

Fig. 1: Ebers-Moll model of the npn BJT transistor

The current through the diodes are as follows and follow 10

$$i_F = I_{Se}(e^{\frac{v_{be}}{v_{Te}}} - 1) \quad (14)$$

$$i_R = I_{Sc}(e^{\frac{v_{bc}}{v_{Tc}}} - 1) \quad (15)$$

The KCL equations we obtain are:

$$i_b = (1 - \alpha_F)i_F + (1 - \alpha_R)i_R \quad (16)$$

$$i_c = \alpha_F i_F - i_R \quad (17)$$

$$i_e = -i_F + \alpha_R i_R \quad (18)$$

The KCL equations can then be used to form the stamp for a transistor in the nonlinear part of the system of equations:

$$\begin{array}{c} n_b \\ n_c \\ n_e \end{array} \begin{bmatrix} (1-\alpha_F) & (1-\alpha_R) \\ \alpha_F & -1 \\ -1 & \alpha_R \end{bmatrix} \begin{bmatrix} i_F(v_b, v_e) \\ \\ i_R(v_b, v_c) \end{bmatrix} \tag{19}$$
$$H \qquad\qquad g$$

Similar to the diode treatment, these current equations can also be differentiated analytically to pre-determine the contribution to the Jacobian of a transistor:

$$\begin{array}{c} \\ \\ \\ \end{array} \begin{array}{ccc} n_c & n_b & n_e \end{array}$$
$$\begin{bmatrix} 0 & \frac{\partial i_F}{\partial v} & -\frac{\partial i_F}{\partial v} \\ -\frac{\partial i_R}{\partial v} & \frac{\partial i_R}{\partial v} & 0 \end{bmatrix} \tag{20}$$
$$J_g$$

Again, like the diode, the netlist input requires specific inputs for the nodes as shown:

```
Q[name] nc nb ne
```

The Ebers-Moll model is useful for large-signal, steady-state models (which are used for describing nonlinear devices, i.e. transistors). Another model is called the Gummel-Poon charge-control model which is intended to explain the transistor dynamics in greater detail than the Ebers-Moll model. There are also small-signal models, that reduce a transistor to a linearized equivalent circuit around the operating point.

III. Variable Time Stepping

If a constant-step finite difference method is applied to a system that contains both periods of constant change and periods of high complex changes, either a very small time step is required to capture the small changes, in which case the time stepping over the area of constant change is inefficient, or a large time step is used to speed up computation but at the cost of losing small-scale details. Variable time stepping is used commonly in SPICE software because of its ability to automatically distribute time points so that few time steps are used in regions of relatively constant change, and more time steps are focused in time periods of large change. A variable time-step trapezoidal rule finite difference method is implemented in this section.

After every time-step iteration, a relative error term is generated. If the error is larger than an user-defined maximum error, the time-step is recalculated using a smaller time-step size. If the error is smaller than the maximum error, the result at the current time is accepted and the size of the time step is increased for the next iteration.

To be able to generate the relative error, two different time stepping methods are used at each time step. The norm of the difference will then be the relative error. The methods implemented in this paper are trapezoidal rule and 2-step trapezoidal rule, which is just trapezoidal rule with 2 steps, each time step being only half as large as the 1-step trapezoidal rule. The L2 norm is used for computing the magnitude of the error.

A maximum allowable time step Δt_{max} is defined to prevent the solution from becoming too coarse. In this case T/50 is used, where T is the total period of time. After each iteration, the time steps are adjusted according to [2]

$$\Delta t_{new} = \min(0.9\sqrt{\frac{\epsilon_{max}}{\epsilon}}\Delta t_n, \Delta t_{max}) \tag{21}$$

where 0.9 is a safety factor. If the error is larger than the allowable error, the new time step for recalculating the current iteration would become smaller. If the error is smaller than allowable error, the time step for the next iteration becomes smaller.

Since two methods are used for every iteration, and if the resulting error is too large then the iteration need to be recalculated, it is not obvious that there would be savings in the computation time. A number of experiments are conducted, and their results recorded in Table I. The test circuit is the binary tree RLC model of a cable as seen in Assignment 5 with an input that periodically switches between the on and off states. This circuit is ideal for testing because the source changes rapidly in a short amount of time, but after the change there is a large period of constant input.

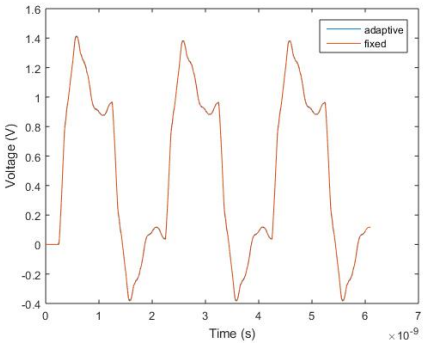

Fig. 2: comparison of the result between the adaptive stepping method with max error of 3×10^{-5} and fixed method with 60000 time steps. The two results are almost identical

For reference, the results of constant time-step trapezoidal rule for the same circuit is recorded in Table II. Note that the error terms are calculated as the absolute error; the trapezoidal rule result with 60000 time-steps have been assumed to have negligible error.

The computation time improved dramatically with adaptive time stepping. This is chiefly due to an optimal the number of time steps being taken. Comparing the computation time of the two methods for roughly the same number of total time steps, trapezoidal performance is faster. This is expected since at each step, adaptive method is computing two trapezoidal

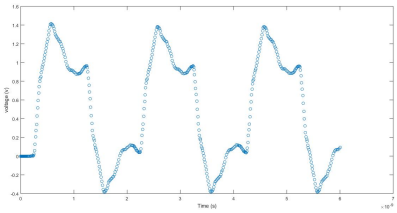

Fig. 3: Variable time stepping result which also shows the distribution of time points during calculation. Our implementation performs exactly as expected: concentrate time points closer together to catch small details when the function changes abruptly, and spread out time points when the function is very smooth.

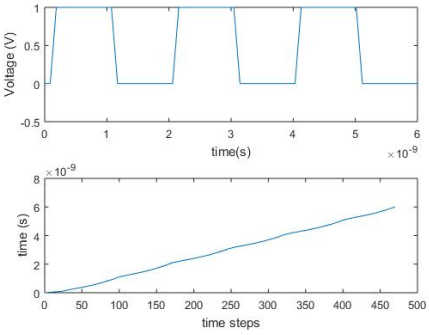

Fig. 4: The input signal is plotted as a function of time at the top. At the bottom, the time reached during time stepping as a function of iteration number is plotted.

Computation Time (s)	Error	Total Steps
1.09	0.0196	86
2.47	0.0136	218
3.75	0.0076	339
5.12	0.0005	470

TABLE I: Variable Time Stepping performance benchmark.

Total Steps	Computation Time (s)	Error
600	0.94	0.0905
1200	1.87	0.0434
6000	9.24	0.0088
12000	18.53	0.0048
60000	92.52	0

TABLE II: Trapezoidal Rule performance benchmark.

method results, as well as any recalculations if error is too large.

Because the time steps are variable and could not be predicted ahead of time, in general they will not match with the time values of the input excitation signal. Since Trapezoidal Rule uses a linear discretization scheme, Linear Interpolation is used to interpolate the value of the drive signal at the time point used by variable time stepping.

IV. Testing and Results

To verify the operation of the developed simulator before moving onto complex guitar effect circuits, simple circuits with well known behaviour were tested. The first of these

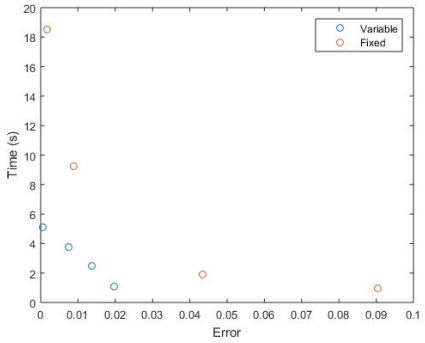

Fig. 5: The computation time savings of variable stepping method over fixed step method for trapezoidal rule. Y axis is in seconds.

if a half-wave rectifier, shown in Figure 6. The results from our simulator are shown in Figure 7. It is visible, when no capacitor was placed on the load resistance, that the output only allows the positive peaks of the input sine wave to pass through while losing $0.7\,\mathrm{V}$ of the amplitude. This is consistent with the known behaviour of this circuit. When a load capacitor was added the output was smoothed, which is also to be expected.

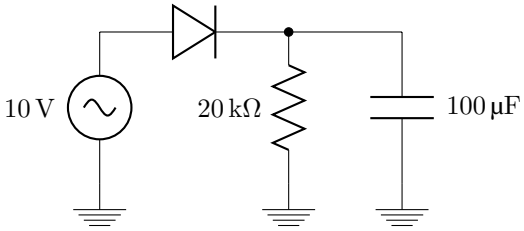

Fig. 6: Half-wave rectifier

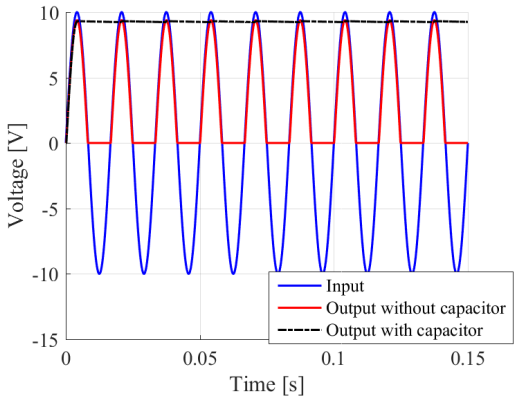

Fig. 7: Results for half-wave rectifier

To check that the solver could work with a more complicated circuit, a bridge rectifier was tested next, the schematic is shown in Figure 8). The results are visible in Figure 9. It is visible from the results with no load capacitor that the circuit rectifies both the positive and negative cycles of the input with a voltage drop of around $1.4\,\mathrm{V}$, which is two diode

voltage drops. When a load capacitor was added, its charging and discharging are observable. All of these observations are characteristic of a bridge rectifier.

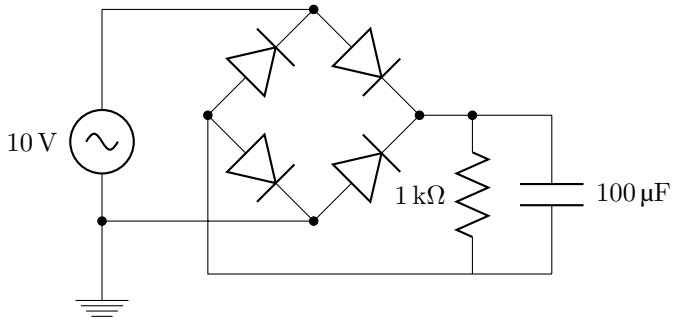

Fig. 8: Bridge rectifier circuit schematic

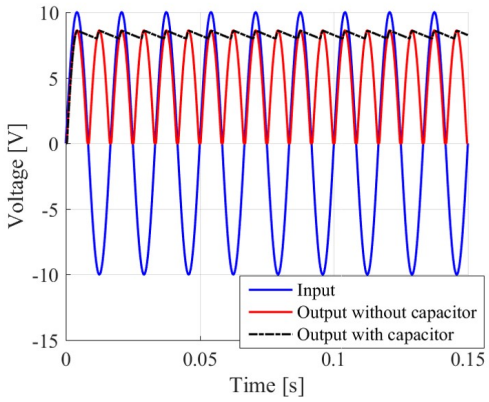

Fig. 9: Results for bridge rectifier

Finally, as the goal of this project was to simulate transistor circuits, a simple BJT common-emitter amplifier was simulated. The schematic for this circuit is visible in Figure 10, and the simulation results are shown in Figure 11. As the expected behaviour of this circuit was not as obvious as that of the rectifiers, the results from our solver were compared with those from a commercial SPICE simulator, LTSPICE. It is visible that the shapes of the output waveforms are very similar, but there is a slight difference in magnitude. This may be due to the fact that a very generic transistor model is being used in our simulator, but LTSPICE uses a specific transistor model which may also be much more complicated than the Ebers-Moll model. Based on all of these test results it was concluded that our circuit simulator is functional.

The audio file that was used for this effect was a guitar wave sample that had a sampling rate of 48000Hz.The guitar circuits and their results are seen from Figures 12 to 15 [4].

V. Discussion

The integration of both, Newtons Method and the Trapezoidal Rule, would seem to be a very simple process, however it was mentioned in Section II that the Trapezoidal implementation will change when nonlinearities are present. In this section we will focus on some ways in which we have worked around the problems that were faced, and also ways that can improve the solver.

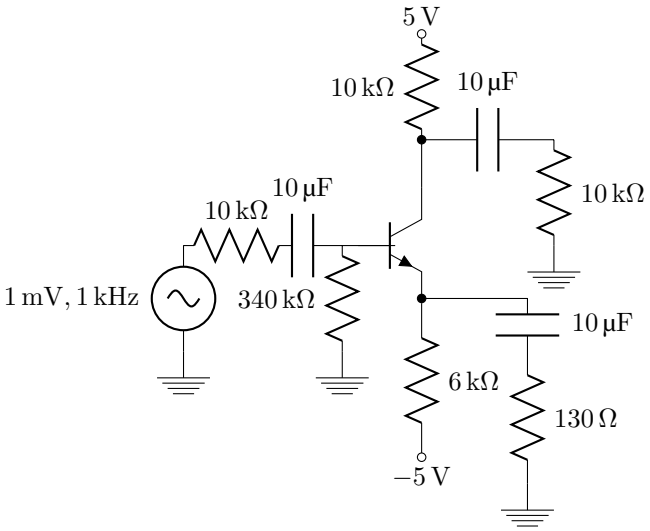

Fig. 10: BJT common-emitter amplifier

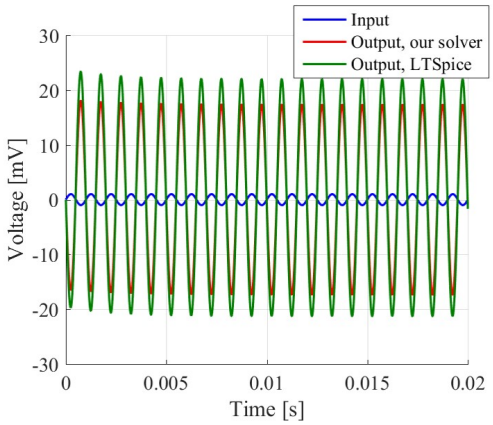

Fig. 11: Results for common-emitter amplifier

A. Nonlinear Circuit Elements

As one can see upon observing the results of the circuit solver for the solutions of the voltage nodes, the inverse of the Jacobian does not remain invertible in several cases and the reasons are due to the nonlinear components in the circuit.

1) Diodes and Transistors: The current through diodes do, on occasion, produce values that are very large, which either

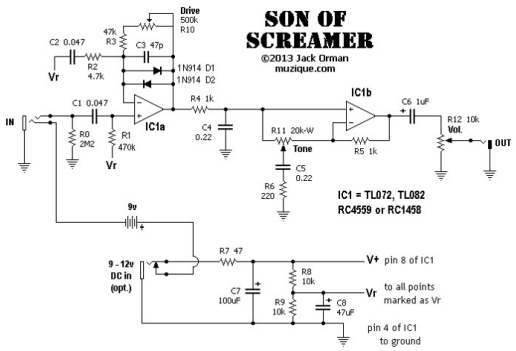

Fig. 12: tubescreamer effect circuit

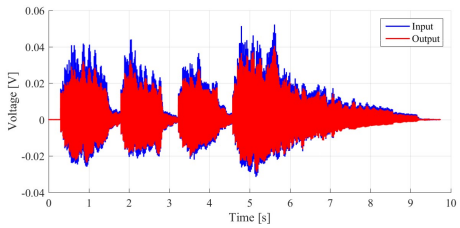

Fig. 13: The output of the tubescreamer (red) is overlayed over input signal(blue)

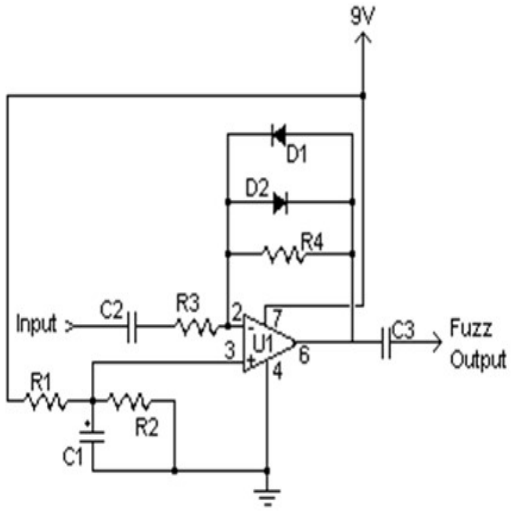

Fig. 14: Fuzz effect circuit

will produce a singular (exceeding precision) inverse-Jacobian, or a poorly scaled non-invertible Jacobian. This is due to the fact that the I-V curve has an exponential which can have large deviations with minute changes in adjacent node voltages. The resulting elements are then what make the Jacobian either diverge greatly from element to element, or exceed precision.

The transistor model that we used (Ebers-Moll model) also has similar problems to that of diodes, as it is includes two diodes. This led to varying degrees of difficulty when operating within a circuit model, but it was possible to alleviate the problems consistent with the Jacobian at times. This will be discussed in the following subsection.

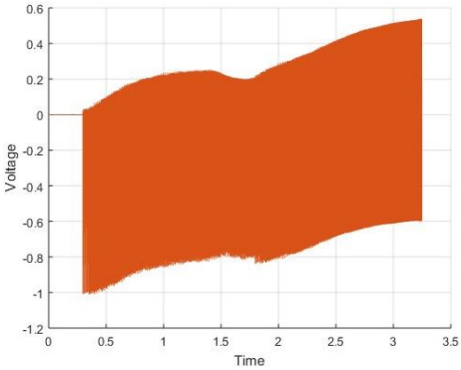

Fig. 15: Results for the fuzz effect circuit using the input guitar wave file for the first 3 seconds

B. Fixing Jacobian Singularity

This section reports on a number of methods used to fix the issue of the poorly-conditioned Jacobian.

Using the converged result of the previous time step as the initial guess of Newton's Method in the current time step greatly increased robustness. Since time step should be very small in order to approximate the instantaneous derivative, the final converged Newton's Method solution should not be drastically different unless there was an instantaneous change in the source, such as in switching. Using the previous step as the guess should decrease the number of iterations required for convergence, thereby reducing the chance of moving into a region o unstable Jacobians. For the first time step, the initial guess is obtained by performing a DC analysis of the circuit since Newton's Method seems to perform better for time-independent circuits.

Variable time stepping may be helpful in dealing with cases of input with abrupt changes. A line of code could be added to the solver where if Newton's Method failed to converge or resulted in near-singular Jacobian, then the time step can be reduced. If $\Delta t = 0$, Newton's Method would converge in one step since the initial guess, which is the final result from the previous step, is also the answer at the current step. Ideally if the time step is small, the voltage drop across the nonlinear elements should not change significantly from the previous time step, and Newton's Method should converge quickly. The fact that it does not converge at all may be because the time step is too large. While there are other dominant contributing factors such as the extreme voltage-current curve of a diode, variable time stepping may mitigate their effects.

The input sources were slightly delayed. During this period of 0 input, any transient behaviour due to initialization would have time to disappear. However, there does not seem to be significant initialization issues from any of our circuits, and this procedure did not bring any benefit.

Often the voltage drop and current flow across the diodes and transistors become extremely small in the middle of Newton's Method computations. Depending on the topology, entire rows of the incidence matrix F could become close to 0. Large grounding resistors are added to the SPICE netlist to reduce the chance of this error. These resistors need to be several orders of magnitude larger than the largest impedances used in the cicuit to prevent noticable perturbations to circuit behaviour. Improvement is obtained for certain circuits such as the bridge rectifier. If the circuit already contains elements with large resistances, the improvement is minimal. Commerical SPICE models use sophisticated algorithms such as Gmin Method to optimize the value and placement of these grounding resistors and can be further explored.[1]

One way to reduce large resistances is to scale the elements. For example, if all impedances are expressed in $k\Omega$ instead of Ω, the correct results can be obtained if voltages are entered in volts and currents in milliamps. In fact milliamps is more common in typical applications. To scale down, voltages

would be in millivants and currents in nanoamps, and scaling up would be similar. The I-V expression would also be scaled, so the problem inherent to diodes would remain.

The exponential behaviour of the diode I-V curve is a major cause of instability. One could constrain input amplitudes to a small range of several tens of millivolts about 0V, where the I-V curve has a moderate slope and the Jacobian is well-behaved, but this requirement is too stringent for an useful SPICE software.

An ad hoc method limiting the maximum allowable Newton step was devised and achieved some improvement. At every Newton iteration k, the solution is updated by:

$$x_{k+1} = x_k + dx \tag{22}$$

where dx is proportional to the derivative of the diode curve. For standard diodes, at voltage values significantly smaller than 0.7V, the derivative is extremely flat and dx may become large enough to cause numerical issues. Therefore, a limit of the magnitude dx_{max} is imposed on the maximum allowable dx. If the computed dx is greater than dx_{max}, the correction

$$dx' = \frac{dx_{max}}{|dx|} dx \tag{23}$$

is applied. There is no theoretical basis for this method, yet it seems to achieve some improvement for several multi-diode circuits.

The other numerical issue related to the diode curve is when the slope is too large. This is the region of high forward bias, when the diode essentially behaves as a short circuit. Perhaps a small series resistance or large parallel resistance can be added to the circuit to reduce the theoretical current flow. An actual diode or transistor also has a physical limit to the maximum possible bias voltage and current, for both forward and reverse biases, which could be hard-coded into Newton's Method to avoid blowing up.

If the goal is to simply simulate a guitar circuit, it may be possible to replace BJTs with MOSFETs since in most operating modes its I-V relation is at most quadratic [3], which would result in a contribution to the Jacobian that is much less sensitive to small changes in node potentials, possibly enabling inversion where we had not been able to previously.

VI. Conclusion

In conclusion, we have obtained a circuit solver which was able to handle certain circuit schematics that contained both transient and nonlinear components, however there were convergence issues with other circuit schematics that resulted due to the effects of how diodes and transistors interacted with the solver. We did find ways of improving the solver by having a case-by-case setup to the solver, which depended on the circuit that was being tested. Methods such as variable time-stepping, ways to fix the Jacobian singularity problem, and various other methods that involved the transient and nonlinear components of the solver were discussed in detail, and it has shown that.

References

[1] F.N. Najm, *Circuit Simulation*, 1st ed. Hoboken, NJ: John Wiley & Sons, 2010.

[2] B. Wetton, *Introduction to Scientific Computation: Time Stepping Methods for Initial Value Problems*, Math 405/607 Course Notes. Department of Mathematics, University of British Columbia

[1] Emarketer.com, 'Social Networking Reaches Nearly One in Four Around the World', 2014. [Online]. Available: http://www.emarketer.com/Article/Social-Networking-Reaches-Nearly-One-Four-Around-World/1009976. [Accessed: 23- Jun- 2014].

[3] A.S. Sedra, K.C. Smith *Microelectronic Circuits*, 5th ed. New York: Oxford, 2004

[4] aaroncake.net, 'Guitar Fuzz Effect', Available: http://www.aaroncake.net/circuits/fuzz.asp [Accessed: 20-Apr-2016].