ECSE 343

Team 12

Andrew Pate

Chenyi Xu

Soumik Kumar

Youjung Kang

Contents

Introduction

The primary objective of this project is to develop a software program that utilizes Modified Nodal Analysis (MNA) equations to simulate the transient response of the half wave rectifier provided. The team has discussed three different approaches to tackle this problem: Forward Euler, Backward Euler, and Trapezoidal method. Each of these methods will be explained in detail, and the final method that the team has chosen will be justified.

Before implementing the method, the team has simulated the circuit on LT Spice which provides us a way to verify the accuracy of our approaches. The plot is shown in ***Figure 1*** in the appendix section.

The Newton-Raphson Method

Newton-Raphson method is an iterative method that starts with an initial estimate of the root, followed by applying a sequence of linear approximations at every iteration to refine the initial estimation. The approximation is repeated until the difference between two successive guesses decreases to a predetermined tolerance limit, signaling that the solution has been attained. Newton-Raphson will be used along with the three approaches Forward Euler, Backward Euler, and Trapezoidal Method.

In order to apply the Newton-Raphson method to this project, a multivariable approach should be taken. An initial guess of X_n is set and $J\Delta X = -f(X_n)$ must be solved for ΔX. Then, X is updated by $X_{n+1} = X_n + \Delta X$. If the value of X does not converge, $X_n \leftarrow X_{n+1}$. This process is repeated until X converges. The Newton-Raphson method takes $X_n=X_0=[0\ 0\ 0\ 0]^T$ and $X_{n+1}= X_n+ \delta$, where δ is a small increment [1].

Approaches

Forward Euler Method

Forward Euler Method is an explicit method used to solve ordinary differential equations (ODEs). This method calculates the solution at each time step directly based on the solution at the previous time step, without using the derivative at the current time point being computed. It is simple and direct. The Forward Euler approximation is shown in ***Equation 1*** [2].

$$X'_n = \frac{X_{n+1} - X_n}{\Delta t} \tag{1}$$

The accuracy of the Forward Euler Method is dependent on the size of the time step h. The smaller time steps result in more accurate solutions, but the use of a large number of time discretization may lead to a significant increase in computational time. Therefore, a trade-off between accuracy and computational efficiency must be considered when selecting the time step size. Additionally, it is important to note that Forward Euler method is not A-stable which means that it is only stable for some values of h.

To find the solution to the circuit, the team substitute the values into ***Equation 1*** and derived ***Equation 2*** as shown below. The Newton-Raphson method can be applied to the new MNA equation modified through the Forward Euler method.

$$F(X_n, X_{n+1}) = GX_n + \frac{C}{\Delta t}(X_{n+1} - X_n) + D(X_n) - B_n = 0 \qquad (2)$$

The function has 5 inputs which are x0, Xguess, tol, t, and dt respectively and returns Xout. A while loop is used and continues until the norm od deltaX is less than tolerance or a maximum number of iterations is reached. Inside the loop, the Jacobian is first calculated. Then, deltaX is calculated by having -J\F where J represents Jacobian and F is equation 2. The output Xout is equal to x0 + deltaX.

The output Xout comprises three rows, with the first row labeled V1 and illustrated in blue, the second row labeled V2 and displayed in red, and the third row labeled V3 and depicted in green. The output in ***Figure 2*** appears to have the same shape as the spice simulation in ***Figure 1***, but it is slightly off. Further improvement in accuracy is still necessary.

Backward Euler Method

Backward Euler Method is an implicit method used to solve ordinary differential equations (ODEs). This method calculates the solution at each time step by solving an equation involving the derivative at the current time point being computed and the solution at the next time step. Unlike the Forward Euler Method, it is more computationally expensive since it requires solving a nonlinear equation at each time step. However, it is more stable and accurate for stiff ODEs, where the Forward Euler Method may fail. The Backward Euler approximation is shown in ***Equation 3*** [3].

$$X'_{n+1} = \frac{X_{n+1} - X_n}{\Delta t} \qquad (3)$$

To find the solution to the circuit, the team substitute the Backward Euler method and derived a new MNA equation as shown below in ***Equation 4***. The Newton-Raphson method can be applied to the new MNA equation modified through the Backward Euler method.

$$F(X_{n+1}) = G * (X_{n+1}) + C * \left(\frac{X_{n+1} - X_n}{\Delta t}\right) + D(X_{n+1}) - B_{n+1} = 0 \qquad (4)$$

The function takes five inputs which are X0 (the initial condition), Xguess (initial guess), tol (tolerance), t (time), and dt (step size) respectively and returns an output variable Xout. The while loop is used for the iterative process until the norm of delta X is less than the tolerance level or the counter value reaches 1000, indicating that the solution did not converge. Inside the loop, delta X is calculated by solving the system of linear equations -F/J, where F is represented in ***Equation 4*** and J represents the Jacobian matrix. The output variable Xout is updated at every iteration by adding itself with delta X.

After plugging the required values into the function, the resulting output is plotted in ***Figure 3***. V1 which is the first row of Xout is shown in blue, V2 which is the second row of Xout is shown in red, and V3 which is the third row of Xout is shown in green. The plot matches with the simulation from Spice in ***Figure 1***, which confirms that the team has obtained the correct result.

Trapezoidal Method

Trapezoidal method is an implicit second order accurate method in solving ODEs. The error between the numerical solution and the exact solution is proportional to h^2. It is more accurate than the Euler's. However, it has a bigger computation cost. The formula for trapezoidal rule is shown below in ***equation 5*** [4].

$$X_{n+1} = X_n + \frac{h}{2}(X'_n + X'_{n+1}) \text{ or } X'_n + X'_{n+1} = \frac{2}{\Delta t}(X_{n+1} - X_n) \quad (5)$$

Applying the trapezoid method to the given MNA, the following equation can be found:

$$GX_n + CX'_n + D(X_n) - B_n + GX_{n+1} + CX'_{n+1} + D(X_{n+1}) - B_{n+1} = 0$$

$$G(X_n + X_{n+1}) + C(X'_n + X'_{n+1}) + D(X_n) + D(X_{n+1}) - B_n - B_{n+1} = 0$$

$$F(X_n, X_{n+1}) = G(X_n + X_{n+1}) + \frac{2}{\Delta t}C(X_{n+1} - X_n) + D(X_n) + D(X_{n+1}) - B_n - B_{n+1} = 0 \quad (6)$$

The Newton-Raphson method can be applied to the new MNA equation modified through the trapezoid method shown in ***Equation 6***.

The function takes 5 input variables which are X0, Xguess, tol, t, and dt and an output variable Xout. As with the previous implementation, the while loop iterates until the norm of delta X is below the specified tolerance or the counter value reaches 1000, indicating that a convergence could not be reached. Delta X which is the difference between two successive runs is calculated by - J/F where J is Jacobian, and F is shown in ***Equation 6***.

Upon inputting the necessary parameters into the function, the resulting output was graphed and is displayed in ***Figure 4***. The output Xout comprises three rows, with the first row labeled V1 and illustrated in blue, the second row labeled V2 and displayed in red, and the third row labeled V3 and depicted in green. Notably, the graph is consistent with the simulation conducted using Spice shown in ***Figure 1***, thereby confirming that the team has indeed achieved the intended outcome.

Final Approach Chosen

After conducting a thorough analysis, we have identified the Backward Euler Method as the optimal approach for this assignment. Our evaluation focused on accuracy and computational cost, comparing the Forward Euler, Backward Euler, and Trapezoidal Methods.

To begin, we eliminated the Forward Euler Method due to its inferior accuracy when compared to the other two methods. Subsequently, we compared the Backward Euler and Trapezoidal Methods and found that both provide a certain level of accuracy. However, we found that Backward Euler method is more efficient. It provides a great balance between accuracy and computation cost.

Team Member Contribution

Everyone in the group made active contributions to the project. Andrew, Youjung, and Soumik were responsible for implementing the three approaches that were discussed above, while Chenyi focused on writing the report, creating the presentation video, and conducting the Spice simulation. All team members participated in the presentation. As a team, we are extremely proud of the progress that we have made so far.

Appendix

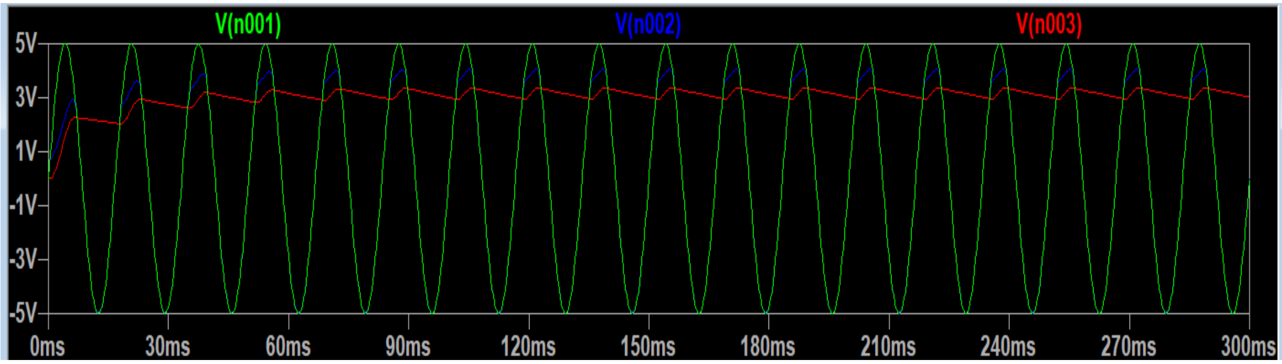

Figure LT Spice Simulation

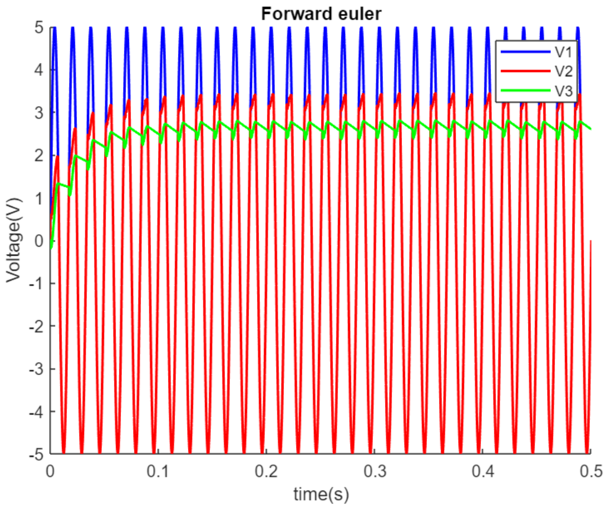

Figure 1 Forward Euler Method

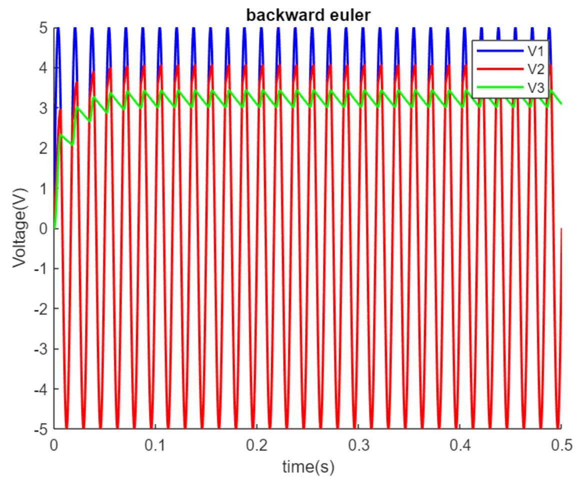

Figure 2 Backward Euler

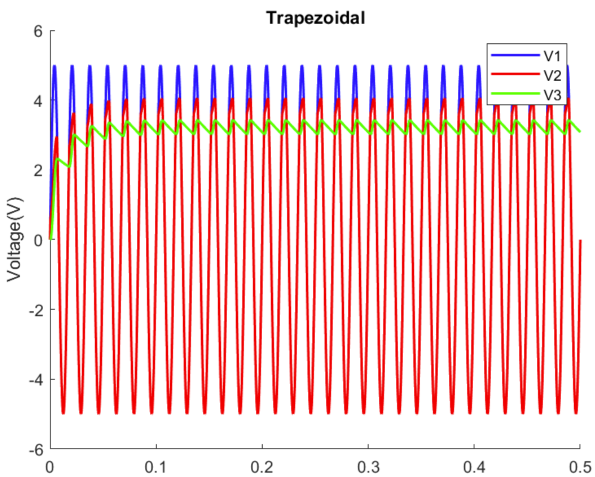

Figure 3 Trapezoidal Method

Reference

[1] Ariel Gershon, Edwin Yung, Jimin Khim, "Newton Raphson Method," Brilliant, 04 2023. [Online]. Available: https://brilliant.org/wiki/newton-raphson-method/#:~:text=The%20Newton%2DRaphson%20method%20(also,straight%20line%20tangent%20to%20it.. [Accessed 04 2023].

[2] S. Brorson, "Forward Euler Method," 26 07 2022. [Online]. Available: https://math.libretexts.org/Bookshelves/Differential_Equations/Numerically_Solving_Ordinary_Differential_Equations_(Brorson)/01%3A_Chapters/1.02%3A_Forward_Euler_method#:~:text=The%20forward%20Euler%20method%20is,%E2%88%92yn)%2Fh.. [Accessed 04 2023].

[3] S. Brorson, "Backward Euler Method," 22 07 2022. [Online]. Available: https://math.libretexts.org/Bookshelves/Differential_Equations/Numerically_Solving_Ordinary_Differential_Equations_(Brorson)/01%3A_Chapters/1.03%3A_Backward_Euler_method#:~:text=The%20backward%20Euler%20method%20is,%2B1)%3Dyn.. [Accessed 04 2023].

[4] U. o. Alberta, "Numerical Integration: Trapezoidal Rule," 2020. [Online]. Available: https://engcourses-uofa.ca/books/numericalanalysis/numerical-integration/trapezoidal-rule/. [Accessed 04 2023].