Achieving Accurate Results With a Circuit Simulator

Ken Kundert and Ian Clifford
Cadence Design Systems
Analog Division
San Jose, Calif 95134

Outline

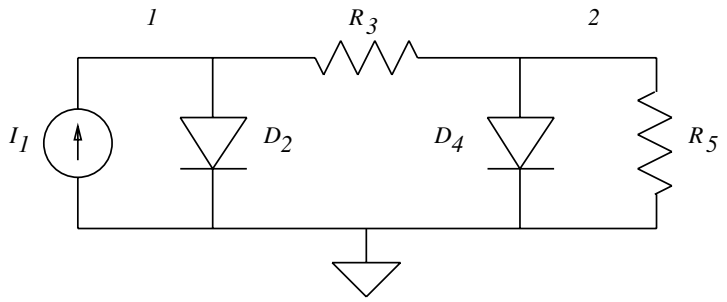
- Solving Nonlinear Systems of Equations
  - Convergence Issues
  - Solving Ordinary Differential Equations
  - Models
  - Examples
  - Circuit Issues

Newton's Method

A iterative methods for solving nonlinear algebraic systems of equations.

Newton's method is used in the DC analysis, and at each step of a transient analysis.

Nodal Analysis



Write Kirchhoff's Current Law

$$-I_1 + I_s(e^{\frac{V_1}{V_T}} - 1) + \frac{V_1 - V_2}{R_3} = 0$$

$$\frac{V_2 - V_1}{R_3} + I_s(e^{\frac{V_2}{V_T}} - 1) + \frac{V_2}{R_3} = 0$$

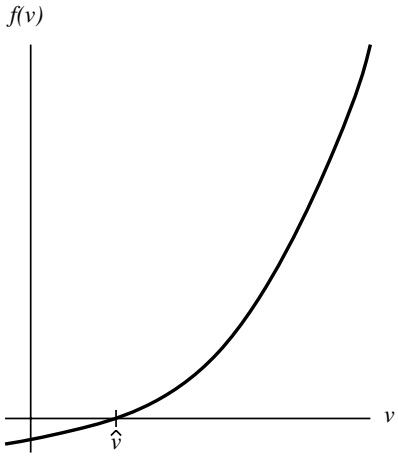
Or $f(v) = 0$.

We need to find $v = [V_1, V_2]$ such that $f(v) = [f_1(v), f_2(v)] = 0$

Newton's Method

Cannot solve nonlinear equations directly.
So guess a starting point, linearize,
solve linear system, and iterate.

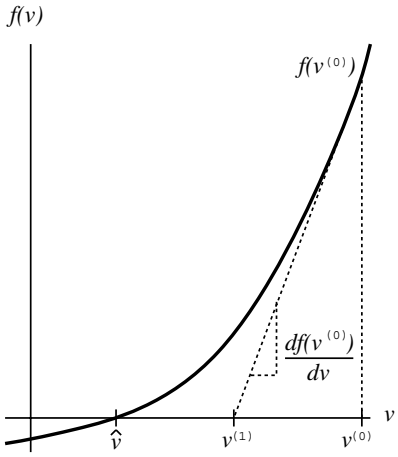
Algorithm for finding $\hat{v}$ such that $f(\hat{v}) = 0$:



Step 0: Initialize

set $k \leftarrow 0$

choose $v^{(0)}$



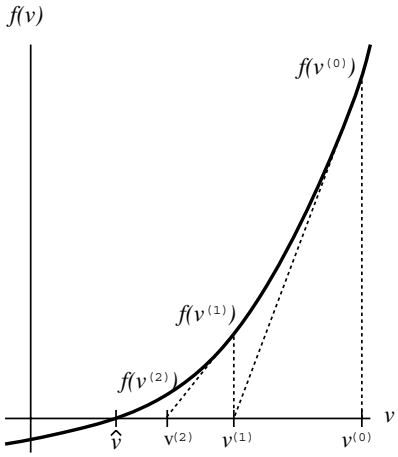
Step 1: Linearize about $v^{(k)}$

$J_f(v^{(k)}) = \frac{\partial f(v^{(k)})}{\partial v}$

where J_f is the Jacobian of f.

Step 2: Solve the linearized system

$v^{(k+1)} \leftarrow v^{(k)} - J_f^{-1}(v^{(k)}) f(v^{(k)})$



Step 3: Iterate

set $k \leftarrow k + 1$

if not converged, go to step 1.

Newton Convergence Criteria

The Newton-Raphson algorithm is used to solve the large systems of nonlinear equations that result when performing DC and transient analyses. It is an iterative algorithm that terminates when the convergence criteria are satisfied. The accuracy of the final solution is determined by the convergence criteria.

Newton's Method:

Solve $f(v) = 0$ by iterating

$$v^{(k)} = v^{(k-1)} - \frac{df(v^{(k-1)})}{dx} f(v^{(k-1)})$$

until the following criteria are satisfied.

SPICE

$$|v_n^{(k)} - v_n^{(k-1)}| < \text{RELTOL} \max(|v_n^{(k)}|, |v_n^{(k-1)}|) + \text{VNTOL}$$

$$|f_{nj}(v^{(k)}) - f_{nj}(v^{(k-1)})| < \text{RELTOL} \max(|f_{nj}(v^{(k)})|, |f_{nj}(v^{(k-1)})|) + \text{ABSTOL}$$

Spectre

$$|v_n^{(k)} - v_n^{(k-1)}| < \text{RELTOL} \max(|v_n^{(k)}|, |v_n^{(k-1)}|) + \text{VNTOL}$$

$$|f_n(v^{(k)})| < \text{RELTOL} \max_j(|f_{nj}(v^{(k)})|) + \text{ABSTOL}$$

Since SPICE does not verify KCL, it is subject to *false convergence.*

Unlike Spectre, SPICE never actually verifies that Kirchoff's current law (KCL) is satisfied. This can result in false convergence, which occurs when the iteration is terminated prematurely with KCL far from being satisfied because progress on one iteration is slow.

In the above, k is the iteration index, n is the node index, and j is the branch index for node n.

Selecting Newton Convergence Tolerances

- RELTOL – relative Newton tolerance

  Primary means of controlling accuracy In general, if you need more accuracy, tighten RELTOL.

  Controls how well KCL is satisfied, and sets how well truncation error is controlled.
- ABSTOL – absolute Newton tolerance for current
  ABSTOL is called `iabstol` in Spectre.

  Tighten if you are interested in very small currents

  Loosen if your circuit exhibits very large currents
- VNTOL – absolute Newton tolerance for voltage
  VNTOL is called `vabstol` in Spectre.

  Tighten if you are interested in very small voltages

  Loosen if your circuit exhibits very large voltages

Remedies for DC Accuracy Problems

- Assume computed solution is correct
  - Circuit may have unanticipated problems
    - Debug circuit as if on lab bench
  - Circuit may have multiple solutions
    - Use NODESET to select solution
- Check topology
- Check models
- Check component values
- Check power supplies
- Tighten RELTOL
- Check GMIN

Outline

- ○ Solving Nonlinear Systems of Equations
- ● Convergence Issues
  - Requirements for Convergence
  - Nodesets
  - Continuation Methods
  - Suggestions
- ○ Solving Ordinary Differential Equations
- ○ Models
- ○ Examples
- ○ Circuit Issues

Newton's Method will Always Converge if ...

- $f(v)$ and its derivatives are continuous.
- Solutions must be isolated ($J_f(\hat{v})$ is nonsingular).
- Initial guess $v^{(0)}$ is sufficiently close to solution $\hat{v}$.

The last condition is the hardest to satisfy.

Nodesets

Allows user to provide a good initial guess.

- Nodesets can be used in both DC and transient analyses.
- Nodesets do not change the solutions of the circuit. However, if the circuit has multiple solutions, nodesets may change the solution that the simulator finds.

How SPICE forces nodesets

Step 1:

Force all set nodes with voltage source in series with 1Ω resistor.

Compute DC solution with set nodes forced.

Step 2:

Delete forcing circuits and compute DC solution using results from step 1 as starting point.

Continuation Methods

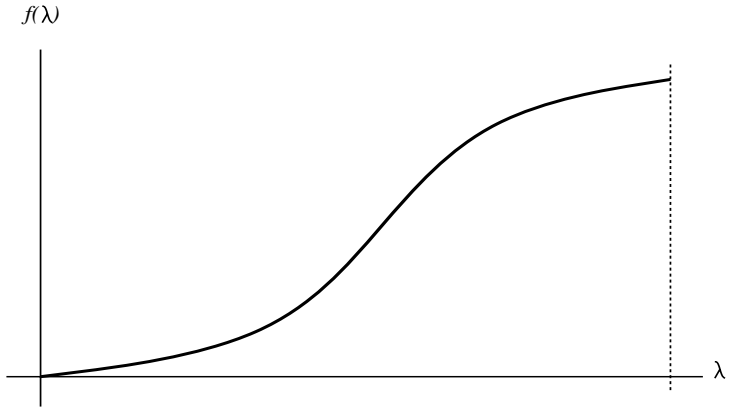
Used to compute a good initial guess.

Write the circuit equations in the form

$$f(v(\lambda), \lambda) = 0$$

and choose λ as a natural or contrived parameter of you circuit such that

- For $\lambda = 0$, the solution $v(0)$ is known in advance or is easy to find.
- For $\lambda = 1$, $f(v(1), 1) = f(v)$.
- $v(\lambda)$ is a continuous function of λ.



Procedure:

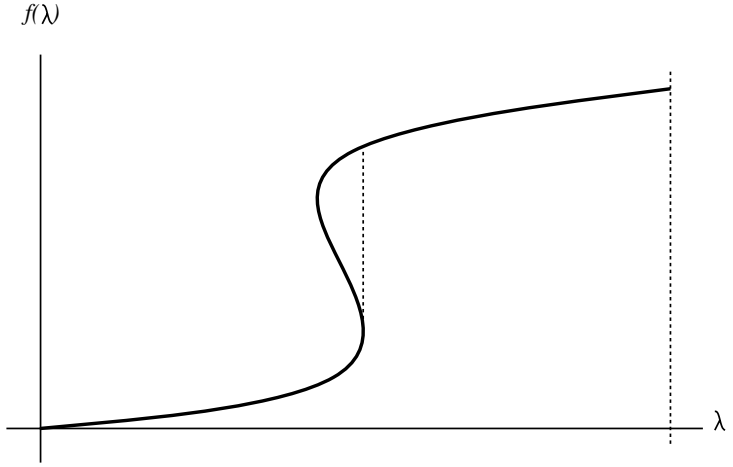
Slowly step λ from 0 to 1 computing $v(\lambda)$ at every step. Use $v(\lambda_k)$ as the initial guess of the solution $v(\lambda_{k+1})$.

If $|\lambda_{k+1} - \lambda_k|$ is small, and if $v(\lambda)$ is continuous, then $v(\lambda_k)$ will be in the region of convergence for $v(\lambda_{k+1})$ and so convergence will be guaranteed.

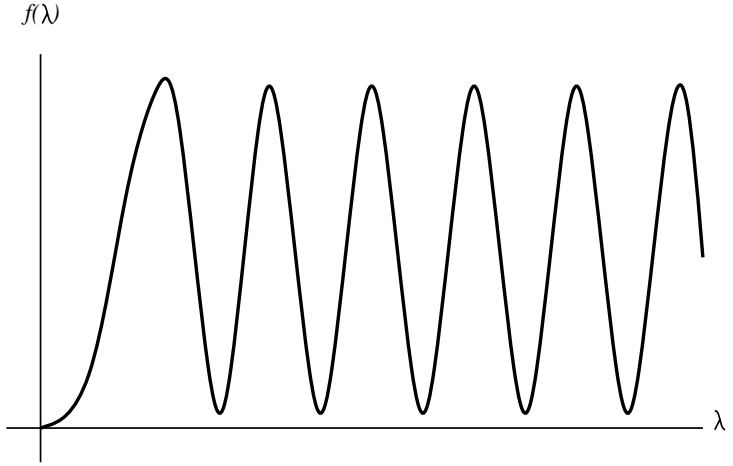
Examples:
Source Stepping, GMIN Stepping, and Pseudo-Transient

Why Continuation Methods Fail

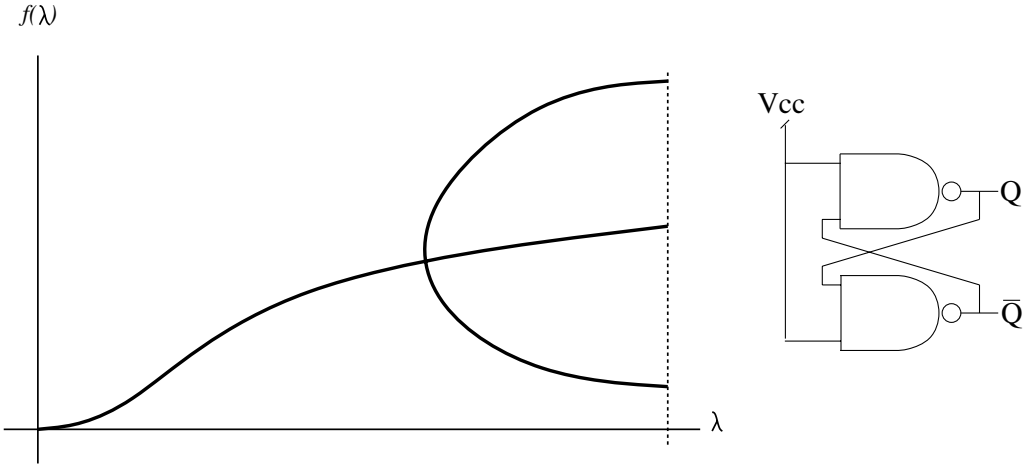
Folds plague source stepping and GMIN stepping.



Oscillations plague pseudo-transient.



Bifurcations result from symmetry.



Reasons for DC Convergence Problems

- Fundamentally a hard problem.
- Degenerate circuit.
- Unreasonable parameter values.
- Excessively tight tolerances
- Errors in model equations.

Remedies for DC Convergence Problems

- Heed warnings
- Check topology
- Check component parameter values
- Avoid small floating resistors

  Use 0-volt voltage source as current probe

  Avoid small parasitic resistors

  Do not set $R_{B_{\min}} \approx 0$ on BJT
- Use Nodesets to give estimate of solution
- Loosen Tolerances (particularly ABSTOL)
- Enable continuation methods

  Set ITL6 in SPICE
- Simplify models

  Try to eliminate second-order effects and parasitic resistors (especially nonlinear base resistors. Use the simplified circuit to generate a nodeset file for the original circuit.
- Disable oscillators and use transient analysis

  As a last resort, Spectre used a pseudo-transient analysis to compute the DC solution. This analysis will not find the DC solution if the circuit oscillates.

Reasons for Transient Convergence Problems

- Instantaneous jumps in signal value

  Jumps are generally caused by local positive feedback on nodes without capacitors. Circuits that can exhibit jumps include latches, Schmitt triggers, and relaxation oscillators.

- Errors in device charge models

Remedies for Transient Convergence Problems

- Avoid simplified models

  Use a complete capacitance model

  Specify nonzero output conductance

  Specify overlap capacitors

- Specify nonzero source and drain areas for all MOSFETs
  This results in the junction capacitors being modeled.

- Identify which node is having difficulties, and add a small capacitor from that node to ground.

- With Spectre, set CMIN nonzero. CMIN causes Spectre to add small capacitors from every node in the circuit to ground.

Outline

- ○ Solving Nonlinear Systems of Equations
- ○ Convergence Issues
- ● Solving Ordinary Differential Equations
  - Integration Methods
  - Truncation Error
  - Initial Conditions
  - Suggestions
- ○ Models
- ○ Examples
- ○ Circuit Issues

Transient Analysis

Formulate equations using Nodal Analysis

$$\begin{aligned} i(v(t)) + \frac{d}{dt} q(v(t)) &= 0 \\ v(0) &= v_0 \end{aligned}$$

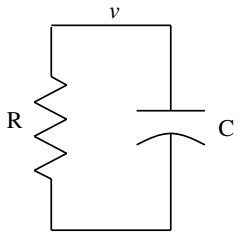
Where

i is current exiting the nodes from resistors.

q is current exiting the nodes from capacitors.

Example:

$$\begin{aligned} C\dot{v}(t) + \frac{v(t)}{R} &= 0 \\ v(0) &= v_0 \end{aligned}$$



In general, it is not possible to solve explicitly a nonlinear ordinary differential equation.
Approach:
Replace $\frac{d}{dt}$ operator by discrete-time approximation.

Integration Methods

Multistep Methods:
Form discrete time approximation to $\frac{d}{dt}$ by assuming trajectory is a low order polynomial.
Assume time is discretized uniformly and that $h = t_{k+1} - t_k$

Backward Euler (Linear Polynomial — One Step)

$$\dot{v}(t_{k+1}) = \frac{1}{h}(v(t_{k+1}) - v(t_k))$$

Trapezoidal Rule (Quadratic Polynomial — One Step)

$$\dot{v}(t_{k+1}) = \frac{2}{h}(v(t_{k+1}) - v(t_k)) - \dot{v}(t_k)$$

Second-order Gear (Quadratic Polynomial — Two Step)

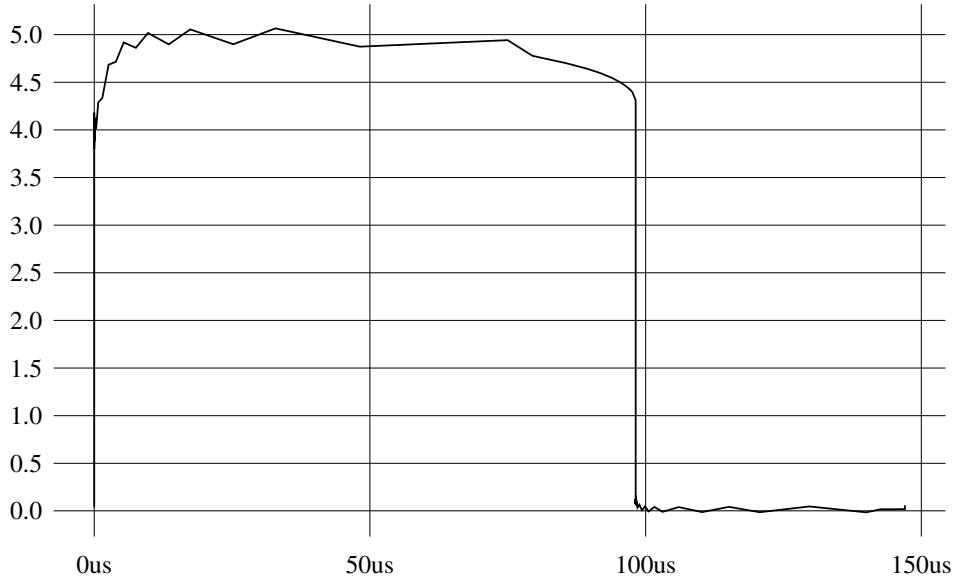
$$\dot{v}(t_{k+1}) = \frac{1}{h}\left(\frac{3}{2}v(t_{k+1}) - 2v(t_k) + \frac{1}{2}v(t_{k-1})\right)$$

Example:
Applying backward Euler to circuit equations:

$$\begin{aligned} q(v(t_{k+1})) - q(v(t_k)) + hi(v(t_{k+1})) &= 0 \\ v(t_0) &= v_0 \end{aligned}$$

Trapezoidal-Rule Rings

Trapezoidal Rule:

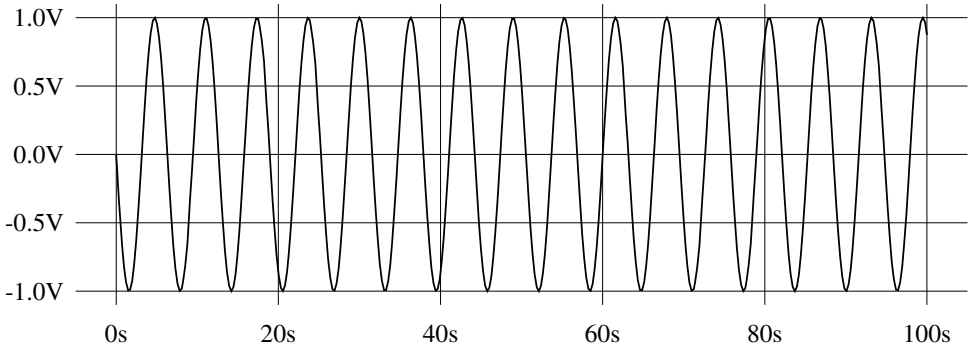


Point-to-point ringing is an artifact of the trapezoidal rule, switch to Gear to avoid it.

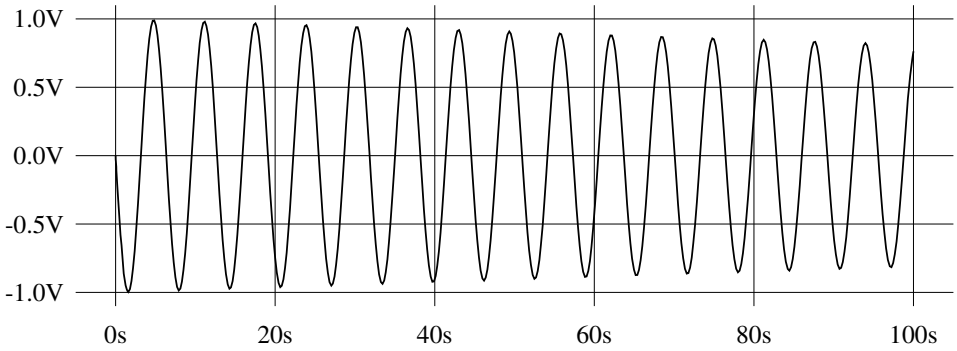
Gear Damps

Apply various integration methods to lossless linear LC oscillator.

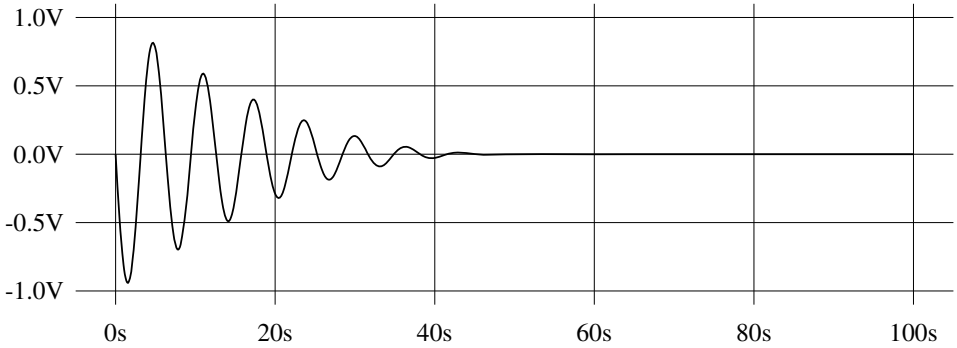
Trapezoidal Rule:



Gear's Second-Order Backward-Difference Formula:



Backward Euler:



Artificial numerical damping is an artifact of Gear and backward-Euler, switch to trapezoidal to avoid it.

Truncation Error

Definitions:

Truncation Error:

Error made by replacing time-differentiation operator with its discrete-time approximation. In general, the smaller the time step, the smaller the truncation error.

Local Truncation Error (LTE):

The truncation error made at a single step assuming all previous steps are accurate.

Global Truncation Error:

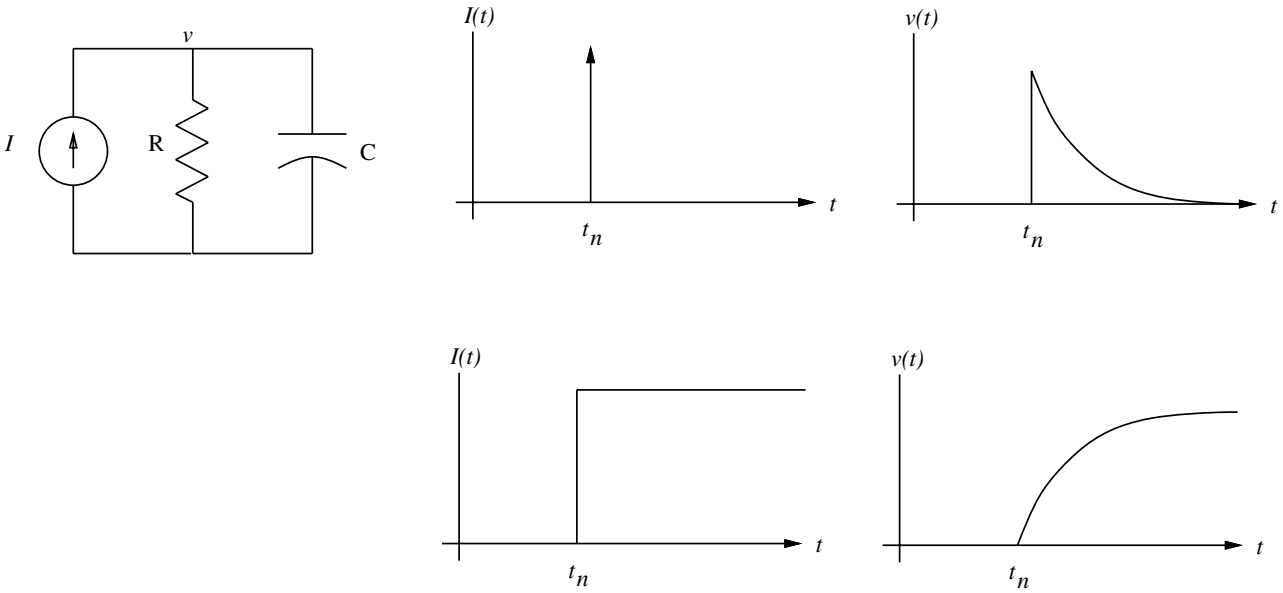
Maximum accumulated truncation error. Related to accumulated LTE and the tendency of the circuit to accumulate or dissipate errors.

In general, it is difficult to estimate global truncation error, so LTE is used to control the time step.

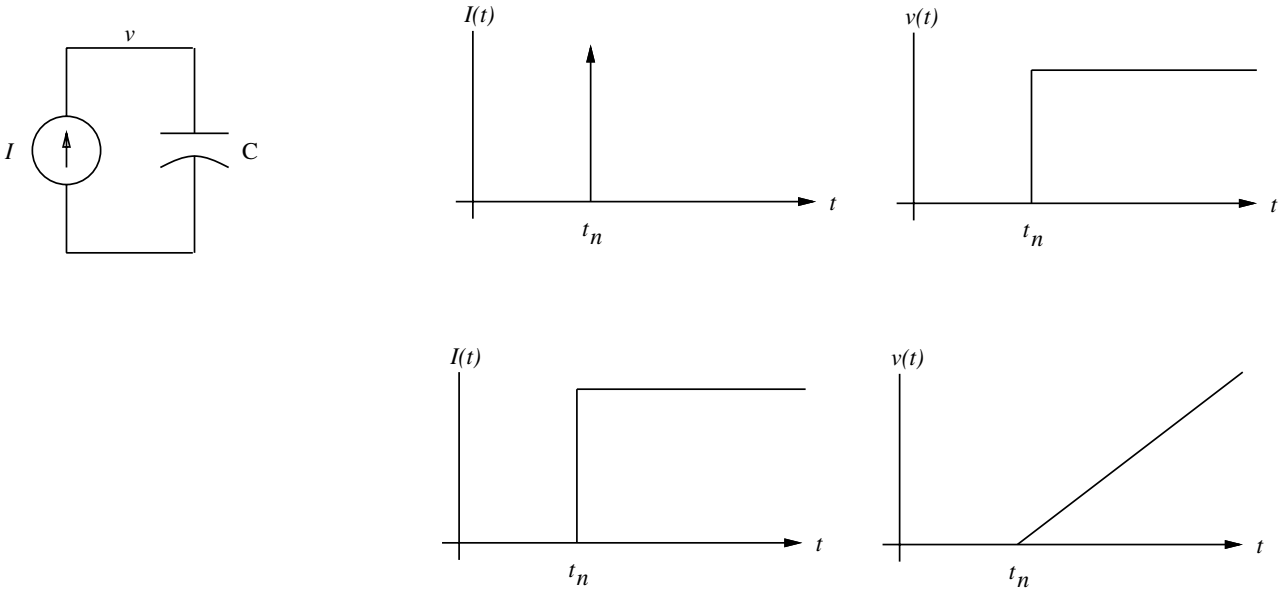
Circuit Determines How Errors Accumulate

Use superposition to consider effect of error separately.
Assume i represents error current injected by not satisfying KCL exactly.

Example 1 — RC Circuit

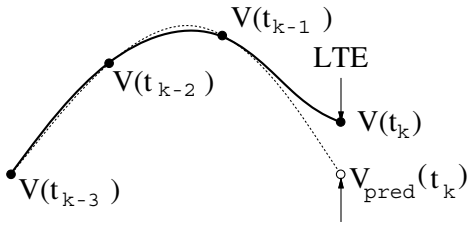


Example 2 — C Circuit



Local-Truncation Error

Integration method is exact if trajectory is low-order polynomial. Thus, LTE is proportional to difference between actual trajectory and low-order polynomial.



The local truncation error (LTE) is the error made on each time-step that results from discretizing time. The local truncation error is the difference between the computed solution and the polynomial extrapolation from the previous few time-steps.

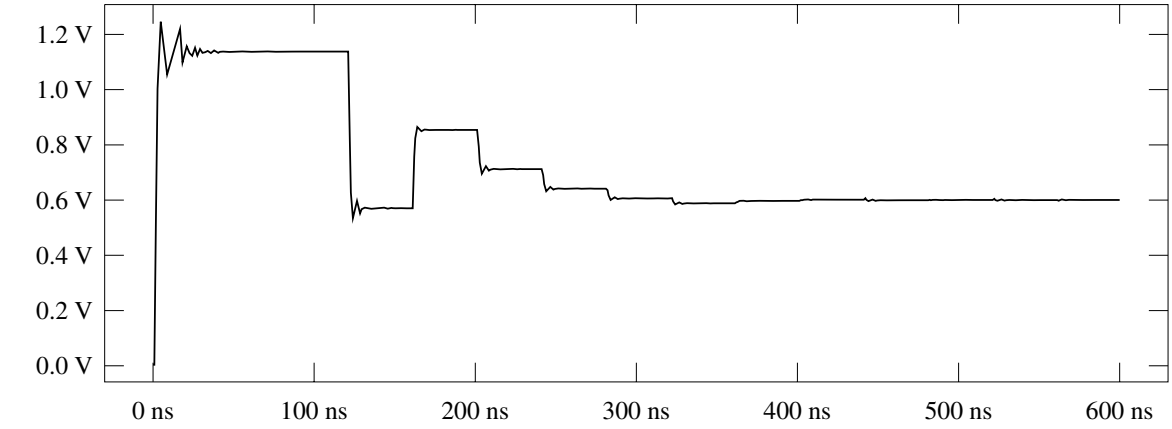
Choose time step such that $v(t_k) - v_{\text{pred}}(t_k) < \text{LTE}_{\text{max}}$

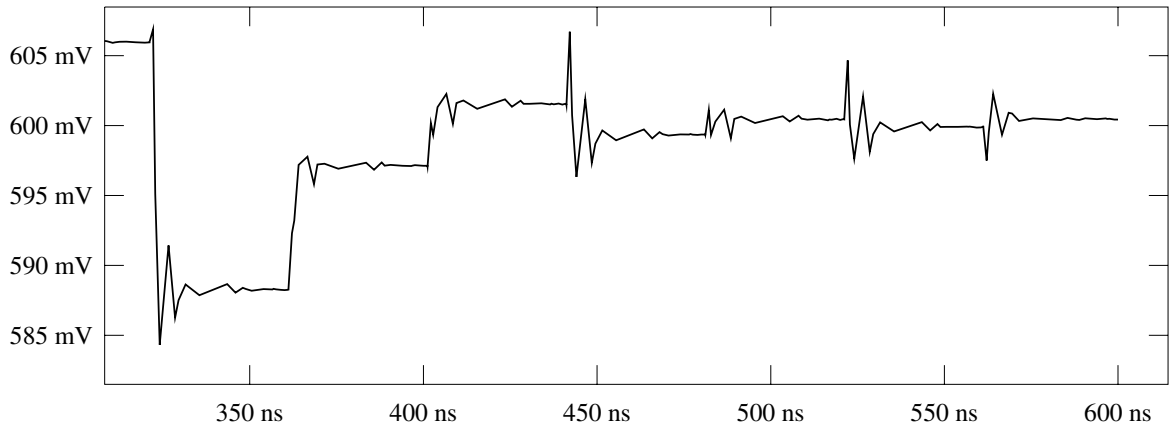
Charge-based Local-Truncation Error Criterion

SPICE — Controls Error in Charge

$$|q(v(t)) - q(v_{pred}(t))| < \text{TRTOL}[\text{RELTOL}|q(v(t))| + \text{CHGTOL}]$$

Use of CHGTOL results in SPICE ignoring large voltage errors on small capacitors.
CHGTOL should be reduced as device capacitances decrease.

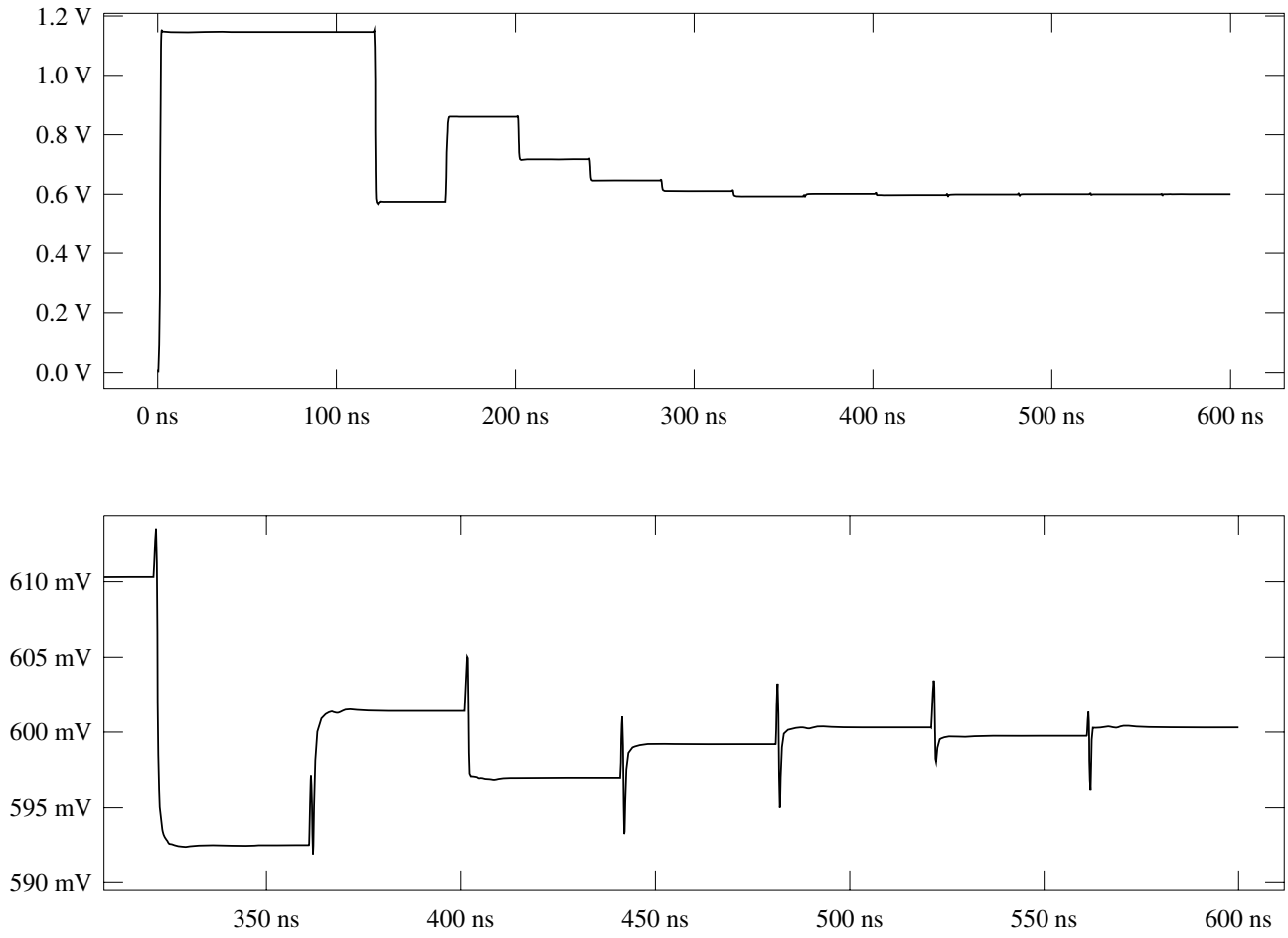




Voltage-based Local-Truncation Error Criterion

Spectre — Controls Error in Voltage

$$|v(t) - v_{pred}(t)| < \text{TRTOL}[\text{RELTOL}|v(t)| + \text{VNTOL}]$$



One can either control the local truncation error in charge or voltage. SPICE-based simulators control the error on the charge, hence the use of the cumbersome tolerance parameter CHGTOL. Controlling the charge error is not a good idea for two reasons. First, users are not interested in charge, and controlling charge error does not guarantee accurate voltages. Second, charge scales with the size of the nonlinear devices. Thus, *CHGTOL* should be reduced every time a new smaller semiconductor process is introduced. Instead, Spectre controls error on voltage, which gives more accurate answers and avoids the charge scaling problem.

Selecting Truncation Error Tolerances

Truncation error tolerances only affect accuracy of transient analysis.

All Newton convergence tolerances are also used as truncation error tolerances. In addition there are the following parameters.

- TRTOL – relative truncation error tolerance

  TRTOL is equivalent to LTERATIO in Spectre.

  Sets acceptable truncation error relative to RELTOL

  Allows independent control of KCL and truncation errors.

  Do not set TRTOL < 2.

  In general, if you need more accuracy, tighten RELTOL. Tightening TRTOL is not a good idea because it causes the simulator to shrink the time step in order to accurately follow the errors that result from not completely converging the Newton iteration. Generally, TRTOL is used to allow the Newton criteria to be set much tighter than the truncation error criteria (this is useful when trying to reduce charge conservation error to negligible levels). Loosening TRTOL by the same factor that RELTOL is tightened results in the Newton convergence criteria being tightened while keeping the truncation error criteria the same.

- CHGTOL – absolute charge tolerance

  Spectre has no equivalent to CHGTOL.

  Should be tightened as devices shrink.

  CHGTOL should be set such that roughly CHGTOL = VNTOL $\times C_{\min}$ where $C_{\min}$ is the smallest interesting capacitor in the circuit. If you are having problems with trapezoidal-rule ringing, tighten CHGTOL.

- TMAX – maximum time step

  Use if simulator misses events or for onset of oscillation.

  Should not normally be used to control error because

  1. Convergence criteria not affected.
  2. Only shrinks large steps.

  In general, tightening RELTOL is a better way to control error than tightening TMAX. TMAX is a poor way to control error for several reasons. First, tightening TMAX does not tighten the Newton convergence criteria. Second, tightening TMAX only shrinks large time steps. Thus tightening TMAX only has an effect when signals are not moving much, and not when signals are moving rapidly. TMAX is the preferred way to control error if signals are very small compared to their DC levels. This happens during the build-up phase of an oscillation. In this situation, the signals are generally too small to produce enough truncation error to shrink the time-step.

- LVLTIM – Time-step control method selection

  Can be used to disable truncation error time step control.

This is very dangerous and is not recommended!

Initial Conditions

Used in a transient analysis to start the circuit from some point other than the DC solution.

- Initial conditions are used only in transient analysis.
- Initial conditions directly affect the computed results by changing the starting point.

How SPICE forces initial conditions

UIC:

Set $v(0)$ equal to initial condition and go. Any unspecified initial conditions are taken to be zero.

No UIC:

Force all nodes with initial conditions with voltage sources in series with 1Ω resistors.

Compute DC solution with initial conditions forced. Set $v(0)$ equal to result and go.

Remedies for Transient Accuracy Problems

- Check initial operating point.
- Check topology.
- Check models.
- Check component values.
- Check power supplies.
- Check stimulus waveforms.
- Tighten RELTOL.
- Check GMIN.
- If charge conservation problem, use charge-conserving models.
- If point-to-point ringing problem, use Gear's method.
- If excessive loss, use trapezoidal rule.
- If simulating an oscillator, set TMAX for 10-25 time steps per period.

Outline

- ○ Solving Nonlinear Systems of Equations
- ○ Convergence Issues
- ○ Solving Ordinary Differential Equations
- ● Models
  - MOS Models
  - Charge Conservation in Models
- ○ Examples
- ○ Circuit Issues

Model Accuracy

Spice Current Model

Model	Output Conductance	Velocity Saturation	Subthresh Region	Impact Ionization
MOS-1	poor	none	none	no
MOS-2	poor	poor	poor	no
MOS-3	poor	poor	poor	no
BSIM-1	poor	okay	okay	no
BSIM-2	good	good	good	no

Spectre Current Model

Model	Output Conductance	Velocity Saturation	Subthresh Region	Impact Ionization
MOS-1	poor	none	none	yes
MOS-2	poor	poor	poor	yes
MOS-3	poor	poor	poor	yes
BSIM-1	poor	okay	okay	yes
BSIM-2	good	good	good	yes

Model Accuracy

Spice Charge Models

Charge Model	Available In Model	Conserves Charge	Velocity Saturation
Meyer	MOS-1,2,3	no	none
Ward-Dutton	MOS-2	yes	good
BSIM	BSIM-1,2	yes	good

Spectre Charge Models

Charge Model	Available In Model	Conserves Charge	Velocity Saturation
Modified-Meyer	MOS-1,2,3, BSIM-1,2	yes	none
Yang-Chatterjee	MOS-1,2,3, BSIM-1,2	yes	good
BSIM	MOS-1,2,3, BSIM-1,2	yes	good

Model Accuracy

Model Minimum Feature Size

Model	Min Gate Length	Min Gate Width	Min Oxide Thickness
MOS-1	10μm	10μm	60nm
MOS-2	2μm	2μm	30nm
MOS-3	0.8μm	2μm	20nm
BSIM-1	0.8μm	1μm	20nm
BSIM-2	0.2μm	1μm	5nm

Spectre Model Evaluation Time

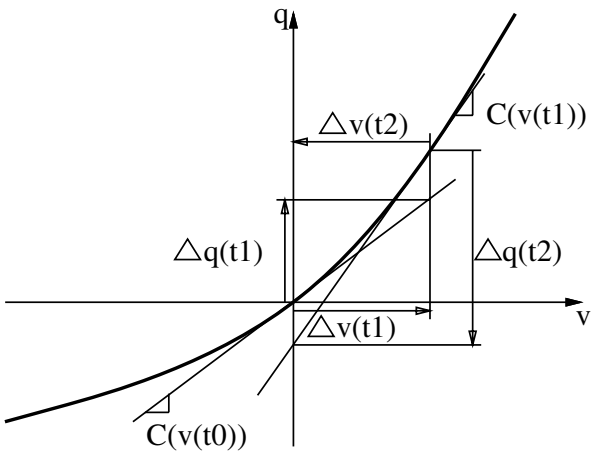
Model	Evaluation Time
MOS-1	0.44ms
MOS-2	1.4ms
MOS-3	0.58ms
BSIM-1	0.59ms
BSIM-2	0.93ms

BSIM2 is the only commercially available MOS model suitable for very small feature size MOSFETs or MOSFETs used in analog circuits.

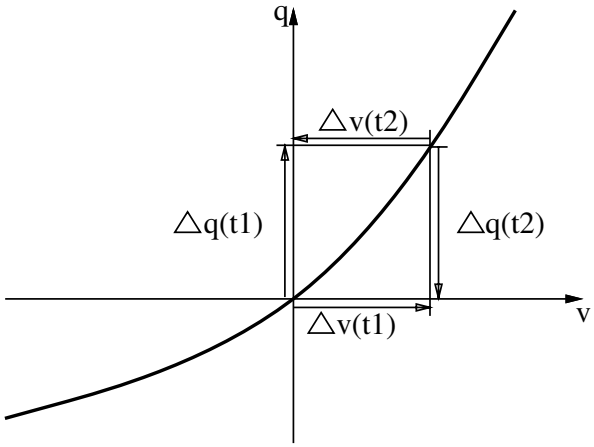
Charge Conservation

Capacitance-Based Models

Do not conserve charge because charge is computed from linearized model.



Charge-Based Models



Outline

- ○ Solving Nonlinear Systems of Equations
- ○ Convergence Issues
- ○ Solving Ordinary Differential Equations
- ○ Models
- • Examples

  Oscillator:
  Suffers from phase drift due to truncation error.

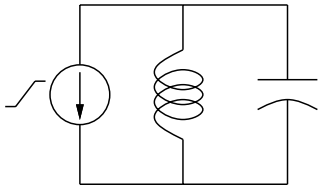
  AC-Coupled Comparator:
  Suffers from charge conservation problems.

  Sample and Hold:
  Suffers droop caused by GMIN.

- ○ Circuit Issues

Lossless Linear LC Oscillator

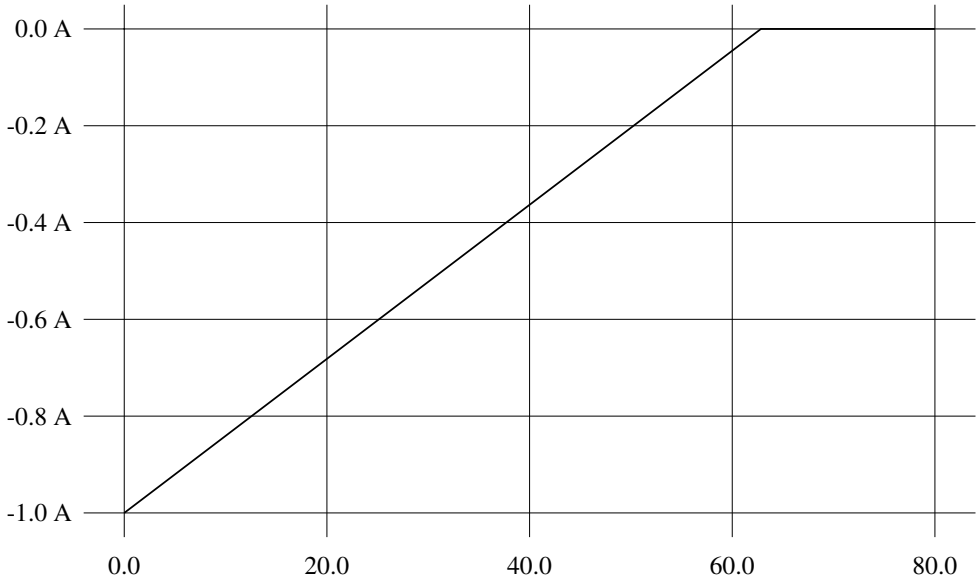
Example:



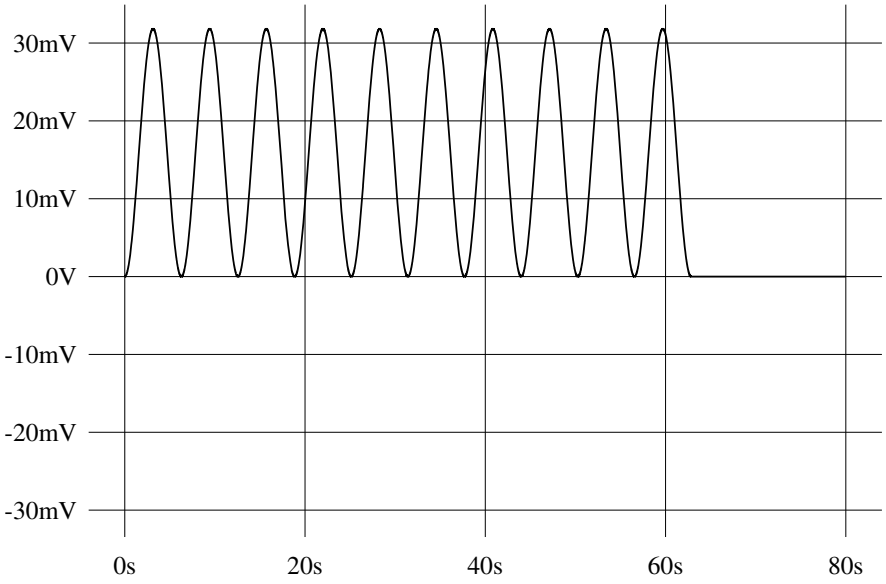
Accuracy determined largely by circuit.

- Resonator stores energy without loss.
- Resonator also stores error without loss.
- Small amount of error accumulates at each step.

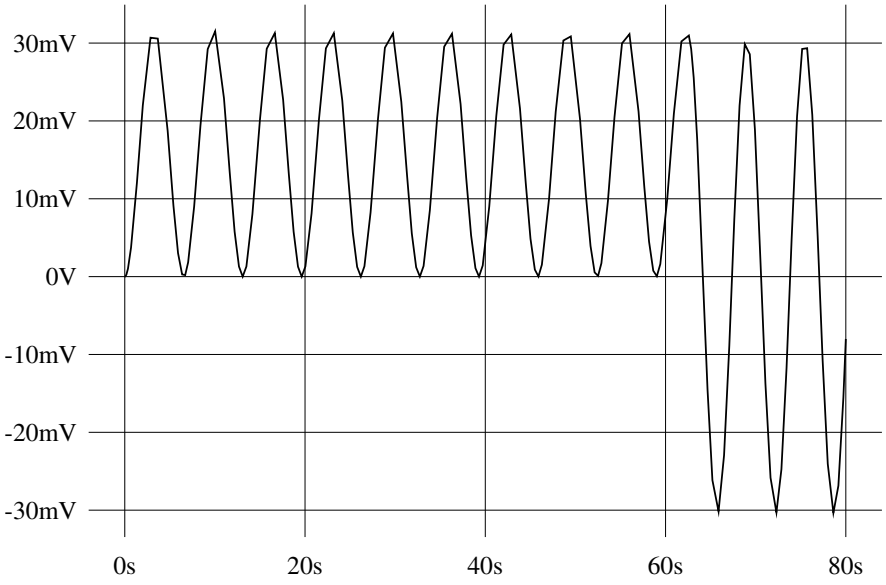
Stimulus:



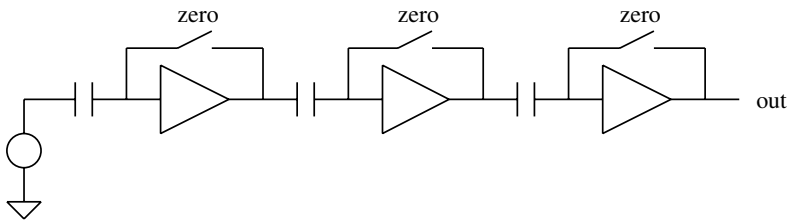
Correct Response:



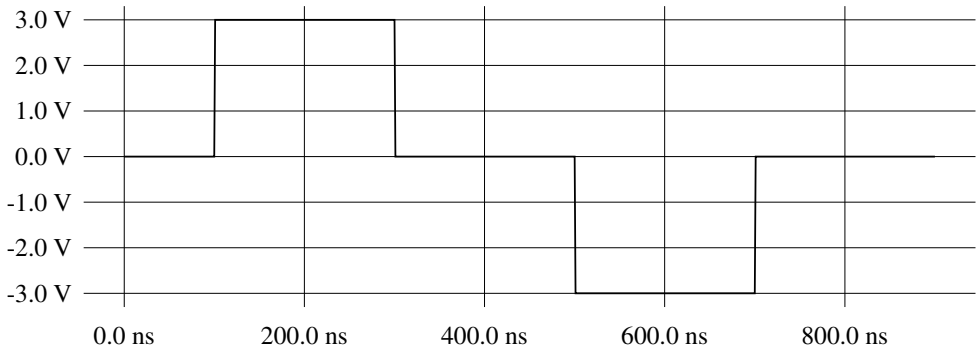
Simulated Response:



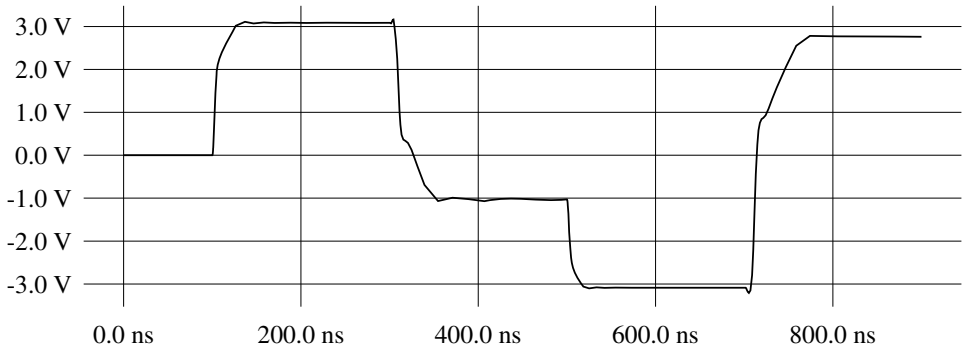
AC-Coupled Comparator with Non-Charge-Conserving Models



Stimulus: (Open switches at t=0)



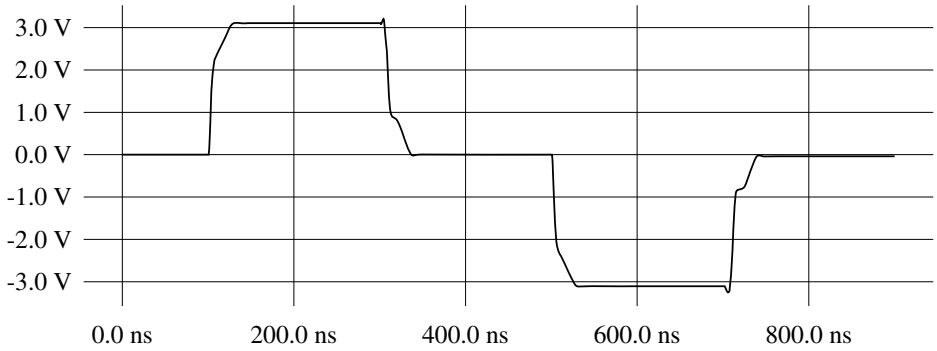
Response Computed with Non-Charge Conserving Models:



1. Non-charge conserving MOS switches inject a small amount of error charge at each time step.
2. Error charge is stored on interstage capacitors.
3. Amplifiers magnify error on capacitors.

AC-Coupled Comparator with Charge-Conserving Models

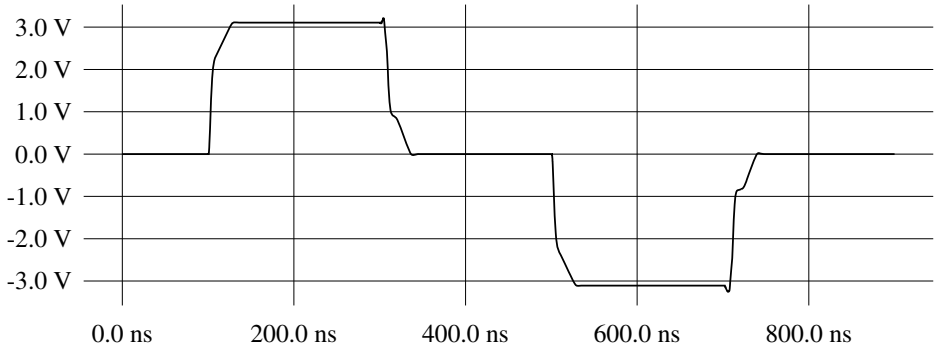
Response Computed with Charge-Conserving Models:



Convergence Criteria:

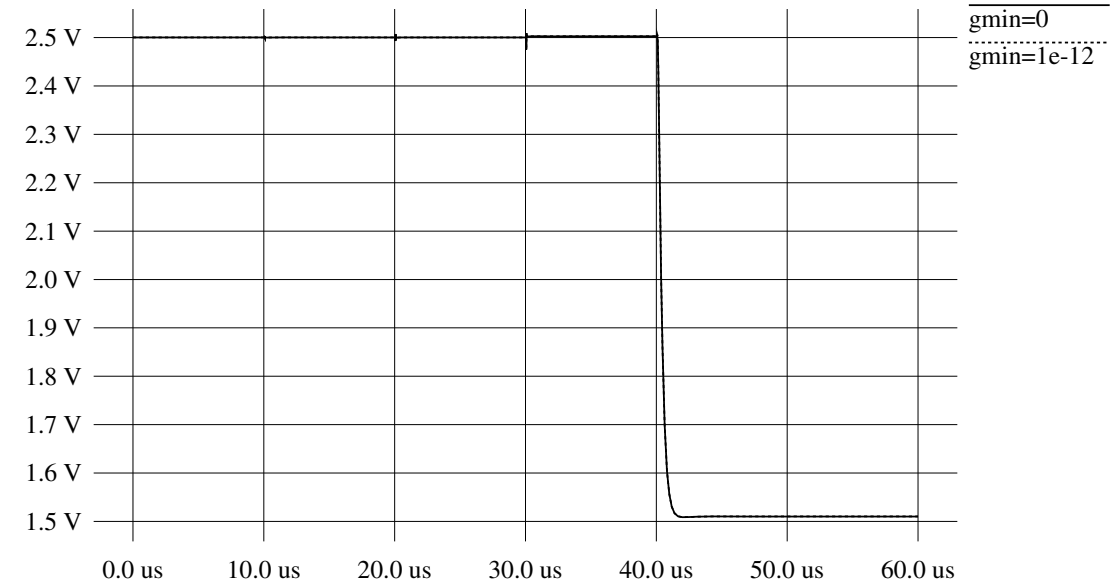
$$|i(v_n^{(j)}) - i(v_n^{(j-1)})| < \text{RELTOL}\max(|i(v_n^{(j)})|, |i(v_n^{(j-1)})|) + \text{ABSTOL}$$

Response Computed with Charge-Conserving Models and Tight Tolerances:

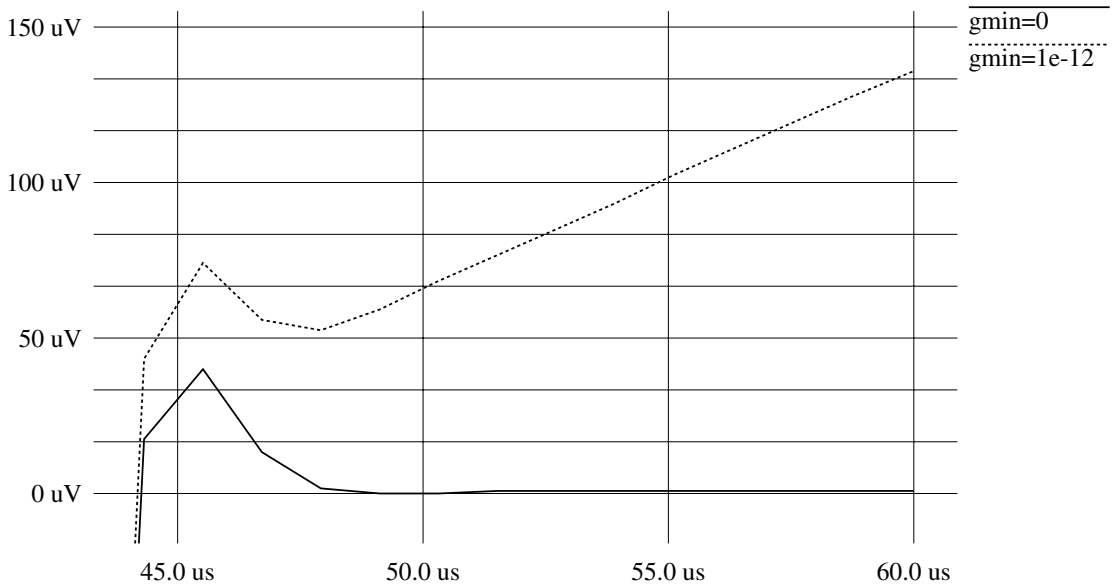


Sample and Hold

Response Computed with GMIN = 10^{-12} and GMIN = 0:

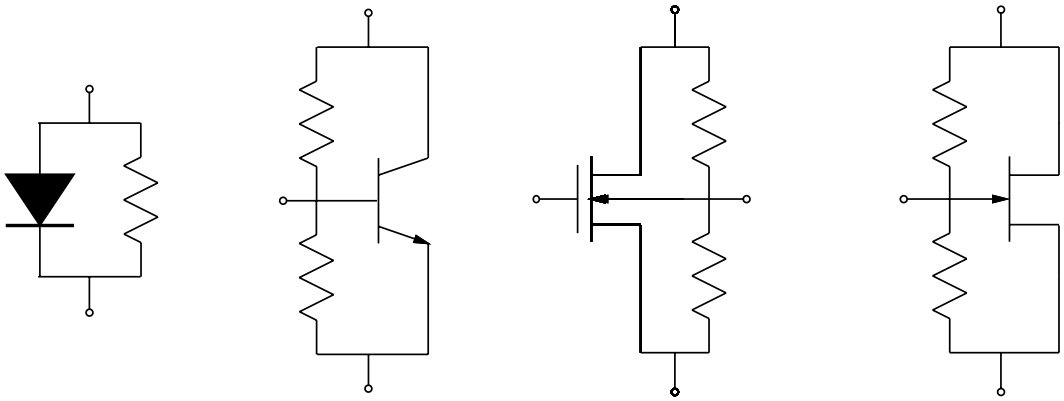


Zoom-in to compute drift:



With GMIN, drift is 8.5 V/s, without GMIN, drift is 0.

How SPICE installs GMIN:



Outline

- ○ Solving Nonlinear Systems of Equations
- ○ Convergence Issues
- ○ Solving Ordinary Differential Equations
- ○ Models
- ○ Examples
- ● Circuit Issues
  - Oscillators
  - Op-Amp Loop Gain Measurements

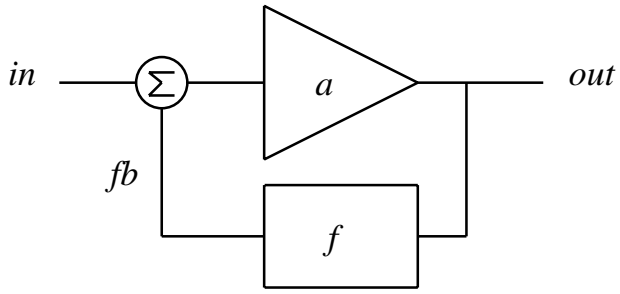
Oscillators

- Tighten RELTOL
  High-Q resonators store energy and error for long periods.
  Since oscillators accumulate phase error, is is a good idea to simulate oscillators with the RELTOL set tigher than you normally use on other circuits.

- Use initial conditions to start oscillation
  Oscillators do not start themselves.
  Initial conditions better than sources:
  Less error prone.
  Does not require modification of circuit.
  Use initial conditions to start an oscillator rather than an impulse stimulus. Initial conditions will more reliably start an oscillator without unexpected side effects.

- Set the maximum time step (TMAX) to accurately follow start-up.
  When simulating the start-up phase of the oscillation, the signals are so small that they are ignored by the time-step control algorithm and the time-step is not chosen small enough to accurately follow the signals. Generally TMAX should be set so that there are at least 10-25 time steps per period.

Op-Amp Loop Gain



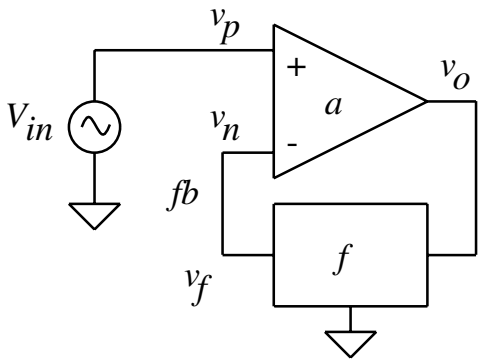
where

$$a = \text{open loop gain}$$
$$f = \text{feedback factor}$$
$$T = af = \text{loop gain}$$
$$A = \frac{1}{1+T} = \text{closed loop gain}$$

Conceptually, to measure loop gain, you break the loop at *fb*. Difficulties:

1. You must not change the operating point.
2. You must account for loading.

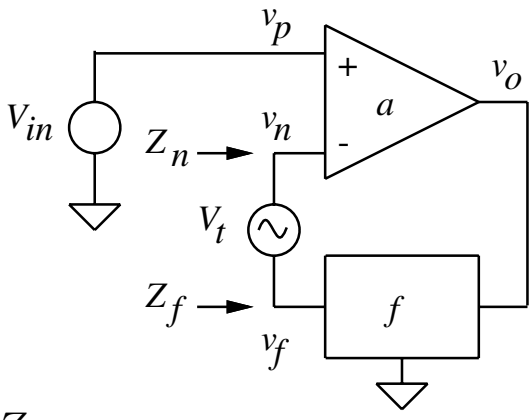
Measuring Voltage Loop Gain



Procedure:

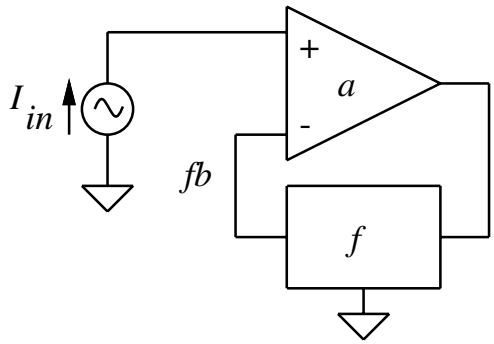
1. Insert floating voltage source with 0 DC value and unity AC magnitude at point *fb*.
2. Set AC magnitude of input source to 0.
3. Measure voltage loop gain as:

$$T_v = -\frac{v_f}{v_n}$$



$T = T_v$ if $Z_n \gg Z_f$

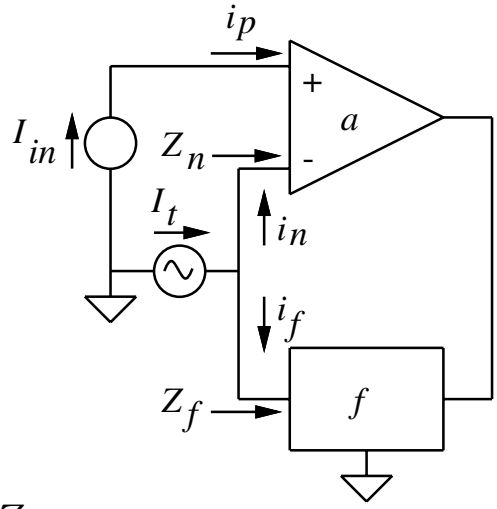
Measuring Current Loop Gain



Procedure:

1. Insert grounded current source with 0 DC value and unity AC magnitude at point *fb*.
2. Set AC magnitude of input source to 0.
3. Measure current loop gain as:

$$T_i = \frac{i_f}{i_n}$$



$T = T_i$ if $Z_n \ll Z_f$

Computing Total Loop Gain

If $T \neq T_v$ because $Z_n \gg Z_f$,
and $T \neq T_i$ because $Z_n \ll Z_f$, use

$$T = \frac{T_v T_i - 1}{2 + T_v + T_i}$$

This equation becomes inaccurate if $T \ll 1$.