Transient analysis – 2

Objective: Discuss the error control in SPICE

Outline: 1. Truncation and round-off errors
          2. “Optimal” computing of transients
          3. Estimation of LTE
          4. Control of integration step size in SPICE

Supplemental reading: Vladimirescu, The SPICE Book,
                       Chapters: 6, 9.4 - 9.5, 10.4
1. Truncation and round-off errors

Truncation error – caused by the algorithm (method).
Round-off error – caused by the finite word length in the computer representation of the number.
Both are important, but the round-off error will be discussed later in relation to “linear solvers”.
This lecture is exclusively dedicated to the discussion of truncation errors.

Notation (reminder):
\[ v(t_n) \] - theoretical value of \( v(t) \) at \( t = t_n \)
\[ v_n \] - approximation to \( v(t_n) \) computed numerically
GLOBAL ERROR (precisely Global Truncation Error - GTE)

After n-steps in computing the error is
\[ e_n = v(t_n) - v_n \]
and it is called GTE.

This error is of interest to a user but it is not computed and not available in general. User does not have direct control over GTE.

However, this error is important in applications!

LOCAL TRUNCATION ERROR (LTE)

Definitions
\[ e_{t_n} = v(t_n) - \bar{v}_n \]
is the LTE, which can be estimated and computed,
\[ \bar{v}_n \]
is a numerically computed approximation to \( v(t_n) \) with exact starting values.

LTE – is an error due to a numerical approximation in a single step calculation (error committed in one step of calculation).
Global Error – contains effects of LTE in all previous steps.
Example/Graphical Illustration of LTE

LTE is not what we directly want to know, but it is the only error that can be estimated and effectively controlled.
Discussion of LTE
Consider B-E:

\[ v_{n+1} = v_n + h_{n+1} \dot{v}_{n+1} \]

or else:

\[ v_n = v_{n-1} + h_n \dot{v}_n \]

Definition of LTE

\[ e_{t_n} = v(t_n) - \bar{v}_n \]

where

\[ \bar{v}_n = v(t_{n-1}) + h_n \dot{v}_n \]

Note: this is the exact starting value for the step \( n \).
Consider T-R:

\[ v_{n+1} = v_n + \frac{h_{n+1}}{2} (\dot{v}_n + \dot{v}_{n+1}) \]

The T-R formula can also be written (for a shifted subscript) as

\[ v_n = v_{n-1} + \frac{h_n}{2} (\dot{v}_{n-1} + \dot{v}_n) \]

To define LTE we need to compute

\[ \bar{v}_n = v(t_{n-1}) + \frac{h_n}{2} [\dot{v}(t_{n-1}) + \dot{v}_n] \]

These are the exact values – never known!
(except for a linear system)

LTE can only be computed for linear systems. But importantly, it can be estimated for nonlinear systems!
Exercise 1: Assume the RC circuit. USE B-E method for computation.

Assume:

\[ E = 5 = \text{const} \]
\[ v(0) = 0 \]

\[ h = 0.2RC = \text{const} \]

Compute first five steps:

\[ V_1, V_2, \ldots, V_5 \]

Calculate:

\[ e_{t_n}, e_n \] (you need the analytical solution for computing these errors).

Compare \( e_n \) and \( e_{t_n} \).
2. “Optimal” computing of transients

Two major factors are considered in computing the transients
- accuracy (errors in computing)
- CPU time.

Typically circuit models are with “built-in” errors, due to approximations in active device models or idealization of passive components. For this reason errors due to computing, or so-called numerical errors are tolerable if they do not exceed the effects of these built-in errors.

Reduction of numerical errors occurs at the expense of increased computing effort (increased CPU time). More accurate computations are more expensive. Consequently too accurate computation is not desirable. In other words we want to compute with errors, which do not exceed “built-in” errors.

In typical problems 0.1% or more often 1% numerical errors are acceptable.
“Optimal” computing occurs when numerical errors are just at the level of tolerance. This is called “optimal” because under those conditions the acceptable results are computed at a minimum CPU time.

In order to conduct “optimal” computing we need to be able to:

1. predict numerical errors,
2. obtain error tolerances/bounds,
3. control the computing process and errors.
3. Estimation of LTE

Numerical analysis supplies the following estimates:

B-E: \[ LTE_n \equiv -\frac{1}{4} h_n^2 v^{(2)}(t_n); \quad v^{(2)}(t) = \frac{d^2 v}{dt^2} \]

T-R: \[ LTE_n \equiv -\frac{1}{72} h_n^3 v^{(3)}(t_n); \quad \text{second order} \]

LMM: \[ LTE_n \equiv \frac{C_k}{(k+1)!} (h_n)^{k+1} v^{(k+1)}(t_n); \quad k\text{-th order} \]

where \( C_k \) - is the constant of the method

Derivatives: \[ v^{(k)}(t) = \frac{d^k v}{dt^k} \] are usually not available and must be computed numerically (approximated, estimated).

Derivatives (or precisely their approximations) are computed using Divided Differences (DD).
Numerical mathematics supplies following formulas for divided differences:

\[
DD_1(t_n) = \frac{v_n - v_{n-1}}{h_n} \equiv v^{(1)}(t_n)
\]

\[
DD_2(t_n) = \frac{DD_1(t_n) - DD_1(t_{n-1})}{h_n + h_{n-1}} \equiv \frac{1}{2} v^{(2)}(t_n)
\]

\[
DD_{k+1}(t_n) = \frac{DD_k(t_n) - DD_k(t_{n-1})}{\sum_{i=0}^{k} h_{n-i}} \equiv \frac{v^{(k+1)}(t_n)}{(k + 1)!}
\]
Substituting proper DD to LTE estimates we get:

**B-E:** \( LTE_n \approx -\frac{1}{2} h_n^2 DD_2(t_n) \)

**T-R:** \( LTE_n \approx -\frac{1}{2} h_n^3 DD_3(t_n) \)

**LMM:** \( LTE_n \approx C_k (h_n)^{k+1} DD_{k+1}(t_n) \).

**EXERCISE 2:** For RC circuit, use B-E as in EXERCISE 1 and compute \( DD \) and \( LTE \) estimates.
4. Step size control using $LTE$

Concept: the estimates of $LTE$ depend on step size $h_n$. Consequently $LTE$ can be controlled by the value of $h_n$.

Given an error bound, $EB_n$, for $LTE$ we state the step size control problem as follows

$$ |LTE|_n \leq EB_n $$

Using the $LTE$ estimates (for T-R and LMM) we obtain:

$$ |C_k (h_n)^{k+1} |DD_{k+1}(t_n)| \leq EB_n $$

Note: $C_k = -\frac{1}{12}$ for T-R, (2nd order method)

$$ C_k = -\frac{1}{2} $$ for B-E, (1st order method)

or else we can write

$$ h_n \leq \left( \frac{EB_n}{|C_k DD_{k+1}(t_n)|} \right)^{1/k+1} $$

which defines the UPPER STEP BOUND, $h_{nE}$. 
Comments: 1° computation of bound for the step size: $h_{nE}$ is expensive  
2° computation of $DD$ is error prone.

Step control in multivariable circuits
1. Each variable (indicated by the superscript “$i$”) has a bound $EB^i_n$ for $LTE$
   $$i = 1, 2, \ldots, I,$$  
   $I$ - is the number of variables.
2. Each variable has an estimate of $LTE$ in the form
   $$LTE^i_n \equiv \frac{C_k}{(k+1)!} (h_n)^{k+1} |v^{(k+1)}(t_n)|$$
   or introducing the expression for $DD$ of “$i$” variable we obtain
   $$LTE^i_n \equiv C_k (h_n)^{k+1} DD_{k+1}^{[i]}(t_n)$$
Using the condition

\[ |LTE_n^i| \leq EB^i \]

for every variable \((i = 1, 2, \ldots, I)\) we can write the following inequality for the step-size

\[ h_n \leq \min_i \left( \frac{EB_n^i}{C_k DD_{k+1}^{[i]}(t_n)} \right)^{\frac{1}{k+1}} \]

which defines the upper bound for step size in the form

\[ h_{nE} = \min_i \left( \frac{EB_n^i}{C_k DD_{k+1}^{[i]}(t_n)} \right)^{\frac{1}{k+1}} \].
5. Control of integration step size in SPICE

a) determination of error bounds

\[ EB_n^i = \varepsilon_a + RELTOL \max \{|x_n^{[i]}|, |x_{n-1}^{[i]}|\} \]

\[ \varepsilon_a = \begin{cases} ABSTOL \\ VNTOL \\ CHGTOLO \end{cases} \]

b) computation of step upper bound : \( h_{nE} \)

\[ h_{nE} = \min_i \left( EB^i \cdot TRTOL \right)^{1/(k+1)} \]

TRTOL – parameter (empirical)
Default : TRTOL = 7

c) step size selection: at each step we need to evaluate the size of \( h_n \) using the bound \( h_{nE} \).
STEP (ERROR) CONTROL OPTIONS IN SPICE

1. LTE Based Strategy (LTBS) for error control
   Uses $h_{nE}$ - bounds of step size. The error control is usually expensive, especially when many step changes are involved.

   The control is economical when a few step adjustments are needed.

   Illustration of a typical waveform in digital systems with the step adjustment process, where many calculations are rejected.
2. Iteration Count Based Strategy (ICBS)
   No computing of error estimates
   Inexpensive
   Suitable for circuits with many transients
   Available in some versions of SPICE
SPICE Error Bounds (after Vladimirescu)

For currents of capacitors and voltage across inductors

\[ \varepsilon_x = \varepsilon_a + \varepsilon_r \max \{ \left| \dot{x}_{n+1} \right|, \left| \dot{x}_n \right| \} \]

\( \dot{x}_{n+1}, \dot{x}_n \) - current of capacitor or voltage across inductor

\[ \varepsilon_a = \begin{cases} \text{ABSTOL} & \text{for current} \\ \text{VNTOL} & \text{for voltages} \end{cases} \]

\[ \varepsilon_r = \text{RELTOL} \]

Charge error (if the charge is used as a circuit variable)

\[ \varepsilon_x = \varepsilon_r \cdot \max \left\{ \left| x_{n+1} \right|, \left| x_n \right|, \varepsilon_{qa} \right\} \cdot \frac{1}{h_n} \]

\( \varepsilon_{qa} = \text{CHGTOL} \), \( x_{n+1}, x_n \) - charge (as a circuit variable).

Analogous formulation for NL inductor fluxes (if applicable) may be used.

Error Bound (EB)

\[ EB_{n+1} = \max \{ \varepsilon_x, \varepsilon_\dot{x} \} \]
Estimate of step size

\[ \hat{h}_{n+1} = \left( \frac{EB_{n+1}}{|C_k DD_{k+1}|} \right)^{\frac{1}{k+1}} \]

Experience showed that this formula is too conservative and as mentioned before a corrective parameter TRTOL yielding better estimate

\[ h_{n+1 \_E} = h_{n+1} \left( \frac{EB_{n+1} \text{ TRTOL}}{|C_k DD_{k+1}|} \right)^{\frac{1}{k+1}} \]

was introduced. Default : TRTOL = 7 \hspace{1cm} \text{(SPICE 2)}

The value of TRTOL can be reset using

.OPTION ...., TRTOL=#…,...
Selection of step-size control mechanism and integration method in SPICE

a) LTE Based Strategy (LTBS)-(available in all SPICE’s)
b) Iteration Count Based Strategy (ICBS) – SPICE2 only

Selection of the step control mechanism is done through the parameter LVLTIM available on .OPTION command. The LVLTIM is available only with the TR method.

LVLTIM=1 selects : ICBS
LVLTIM=2 (default) selects : LTBS

Selection of integration method is done through the parameter METHOD available on .OPTION command.

There are 2 methods available:
Trapezoidal (T-R) method is default one. It can be explicitly selected by setting METHOD = TRAP (default).
Gear’s method is selected by setting METHOD = GEAR

Additional parameter selecting the method order (number of steps) is: MAXORD = 2 <3,…,6>

This selection is available only with GEAR.
With GEAR the LTBS step control is used exclusively.
Details of LTBS control of step size in SPICE

\[ h_{n+1} = \frac{h_n}{8} \]
Implementation of Step Control

Solve at $\tilde{t}_{n+1}$
If ($\text{iter\_num} < \text{ITL4}$)
  Compute $\bar{h}_{n+1} = f(\text{LTE})$
  if ($\bar{h}_{n+1} < 0.9\bar{\tilde{h}}_{n+1}$) then
    Reject $\tilde{t}_{n+1}$
    $h_{n+1} = \bar{h}_{n+1}$; recompute
  else
    Accept $\tilde{t}_{n+1}$ and results
    $h_{n+2} = \min\{\bar{h}_{n+1}, 2h_n, T_{\text{max}}\}$
Else
  reject $\tilde{t}_{n+1}$, $h_{n+1} = h_n / 8$
  if ($h_{n+1} \geq h_{\text{min}}$) recompute
  else: TIME STEP TOO SMALL -- error message
ICBS – Strategy Based on Iteration Count

Given two integers: ITL3 (default = 4), ITL4 (default = 10)

Notation: \( I_n \) - N-R iteration count at step \( t_n \)
\( I_{nx} \) - iteration count after convergence

Strategy:
\[
\begin{align*}
I_{nx} &< \text{ITL3} \quad \text{then} \quad h_{n+1} = 2h_n \\
\text{ITL3} \leq I_{nx} \leq \text{ITL4} \quad \text{then} \quad h_{n+1} = h_n \\
I_n &> \text{ITL4} \quad \text{then} \quad h_{n+1} = \frac{h_n}{8}
\end{align*}
\]

Step is rejected and re-computation of values is necessary.

Check of the condition: \( h_{n+1} \geq h_{\text{min}} \) is also performed.