Time-domain Analysis of Linear and Nonlinear Circuits

Dr. José Ernesto Rayas-Sánchez

1

Introduction

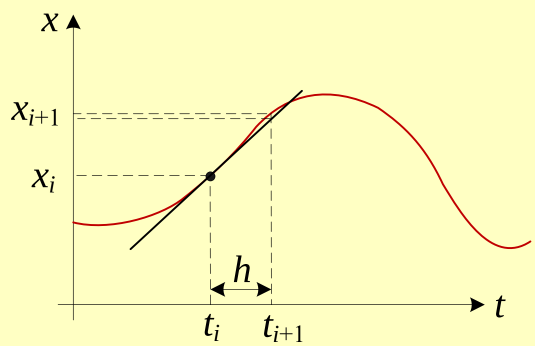
- Time domain analysis can be realized in the transient regime or in the steady-state regime
- Calculating the transient response of a circuit implies solving a system of differential equations
- A number of methods can be used to calculate the time response of electrical circuits:
  - Linear Multi-Step (LMS) formulae
  - Runge-Kutta (R-K) formulae
  - Harmonic-Balance (HB)

Dr. J.E. Rayas-Sánchez

2

Solving Differential Equations

$$x' = \frac{dx}{dt} = f(x,t) \qquad x(t_0) = x_0 \qquad x'_i = f(x_i, t_i)$$



$$h = t_{i+1} - t_i$$

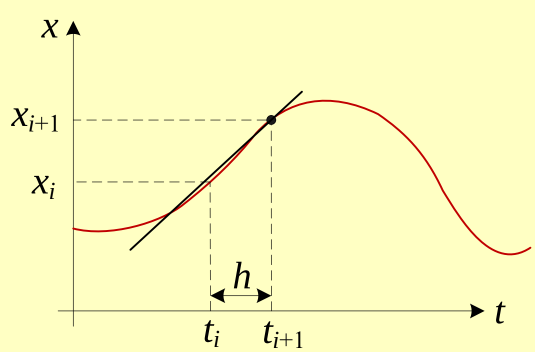
$$x'_i \approx \frac{x_{i+1} - x_i}{h}$$

$$x_{i+1} \approx x_i + hx'_i$$

Forward Euler formula

Dr. J.E. Rayas-Sánchez 3

Solving Differential Equations (cont.)



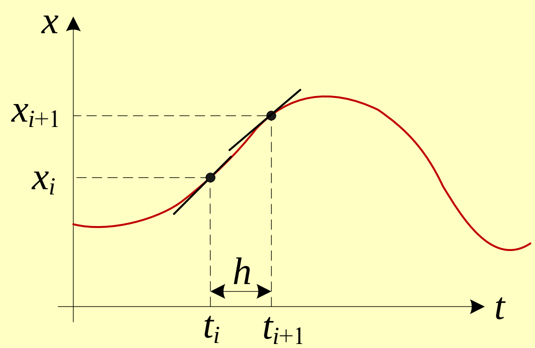
$$x'_{i+1} \approx \frac{x_{i+1} - x_i}{h}$$

$$x_{i+1} \approx x_i + hx'_{i+1}$$

Backward Euler formula

Dr. J.E. Rayas-Sánchez 4

Solving Differential Equations (cont.)



$$\frac{x'_{i+1} + x'_i}{2} \approx \frac{x_{i+1} - x_i}{h}$$

$$x_{i+1} \approx x_i + \frac{h}{2}(x'_{i+1} + x'_i)$$

Trapezoidal formula

Generalizing to Systems of Differential Eq.

$$\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, t) \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \qquad \boldsymbol{x}'_i = \boldsymbol{f}(\boldsymbol{x}_i, t_i)$$

$$\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{f} \in \Re^n$$

- Forward Euler:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{x}'_i$$

- Backward Euler:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{x}'_{i+1}$$

- Trapezoidal:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{x}'_{i+1} + \boldsymbol{x}'_i)$$

Using Predictors and Correctors

- Backward Euler and Trapezoidal formulae require $\boldsymbol{x}'_{i+1}$ to calculate $\boldsymbol{x}_{i+1}$, but this value is not known at the ith iteration
- Forward Euler can be used as a predictor for $\boldsymbol{x}_{i+1}$, which can be later inserted into a corrector using Backward Euler or Trapezoidal formulae

A Forward Euler Matlab Implementation

```
% This function solves a system of n first-order ordinary differential equations
% x'=f(x,t) using the Forward Euler formula.
%
% Usage: [t,x] = F_Euler(fun,x0,t0,tf,h)
%        t: Independent variable (row vector of length N).
%        x: Solution of the differential equations (an N by n matrix).
%        fun: name of the multidimensional vector function (string) that
%             evaluates the differential equations. This function takes two
%             arguments: a row vector x of length n and a scalar independent
%             variable t. It returns a row vector f of length n.
%        x0: initial condition (row vector, of length n). x0 = x(t=t0).
%        t0: initial value of the independent variable.
%        tf: final value of the independent variable.
%        h: step used for the independent variable.

function [t,x] = F_Euler(fun,x0,t0,tf,h)

N = round((tf-t0)/h); % Number of points.
n = length(x0);
x = zeros(N+1,n);
t = zeros(1,N+1);
x(1,:) = x0;
t(1) = t0;
i = 1;

while i<=N
   f = feval(fun,x(i,:),t(i));
   x(i+1,:) = x(i,:) + h*f;
   t(i+1) = t(i) + h;
   i = i+1;
end
```

A Backward Euler Matlab Implementation

```
% This function solves a system of n first-order ordinary differential equations
% x'=f(x,t) using the Backward Euler formula with predictors but no correctors.
%
% Usage: [t,x] = B_Euler(fun,x0,t0,tf,h)
%        t: Independent variable (row vector of length N+1).
%        x: Solution of the differential equations (an N+1 by n matrix).
%        fun: name of the multidimensional vector function (string) that
%             evaluates the differential equations. This function takes two
%             arguments: a row vector x of length n and a scalar independent
%             variable t. It returns a row vector f of length n.
%        x0: initial condition (row vector, of length n). x0 = x(t=t0).
%        t0: initial value of the independent variable.
%        tf: final value of the independent variable.
%        h: step used for the independent variable.

function [t,x] = B_Euler(fun,x0,t0,tf,h)

N = round((tf-t0)/h); % Number of points.
n = length(x0);
x = zeros(N+1,n);
t = zeros(1,N+1);
x(1,:) = x0;
t(1) = t0;
i = 1;
```

```
while i<=N
   f = feval(fun,x(i,:),t(i));
   xp = x(i,:) + h*f;
   t(i+1) = t(i) + h;
   f = feval(fun,xp,t(i+1));
   x(i+1,:) = x(i,:) + h*f;
   i = i+1;
end
```


A Trapezoidal Matlab Implementation

```
% This function solves a system of n first-order ordinary differential equations
% x'=f(x,t) using the Trapezoidal formula with predictors but no correctors.
%
% Usage: [t,x] = Trapezoidal(fun,x0,t0,tf,h)
%        t: Independent variable (row vector of length N+1).
%        x: Solution of the differential equations (an N+1 by n matrix).
%        fun: name of the multidimensional vector function (string) that
%             evaluates the differential equations. This function takes two
%             arguments: a row vector x of length n and a scalar independent
%             variable t. It returns a row vector f of length n.
%        x0: initial condition (row vector, of length n). x0 = x(t=t0).
%        t0: initial value of the independent variable.
%        tf: final value of the independent variable.
%        h: step used for the independent variable.

function [t,x] = Trapezoidal(fun,x0,t0,tf,h)

N = round((tf-t0)/h); % Number of points.
n = length(x0);
x = zeros(N+1,n);
t = zeros(1,N+1);
x(1,:) = x0;
t(1) = t0;
i = 1;
```

```
while i<=N
   f = feval(fun,x(i,:),t(i));
   xp = x(i,:) + h*f;
   t(i+1) = t(i) + h;
   fn = feval(fun,xp,t(i+1));
   x(i+1,:) = x(i,:) + (h/2)*(f+fn);
   i = i+1;
end
```

Example 1 - Forward Euler

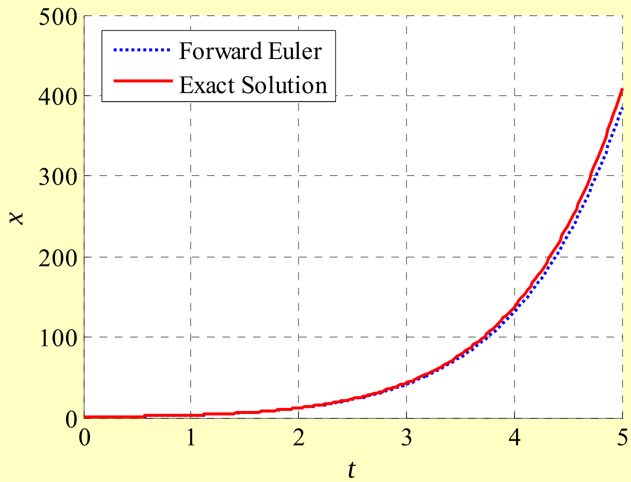
$x' = x + t^2$

$x_0 = 1$

$h = 0.025$

Exact solution:

$x = 3e^t - t^2 - 2t - 2$



Example 1 - Backward Euler

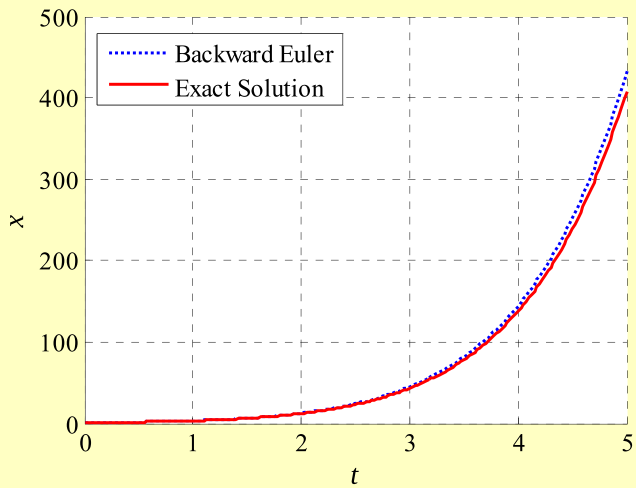
$x' = x + t^2$

$x_0 = 1$

$h = 0.025$

Exact solution:

$x = 3e^t - t^2 - 2t - 2$

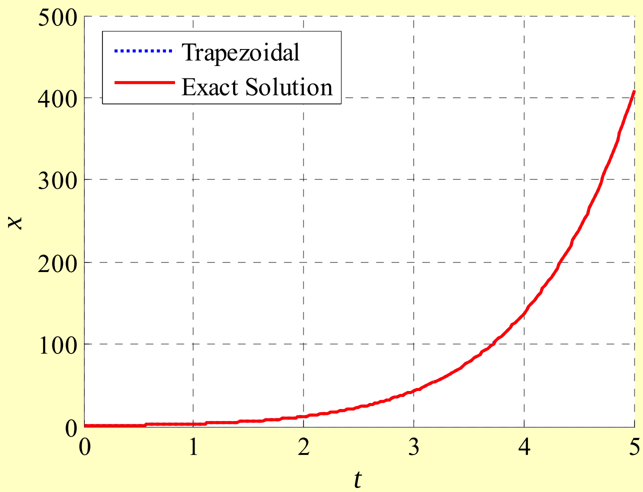


Example 1 - Trapezoidal

$$x' = x + t^2$$
$$x_0 = 1$$
$$h = 0.025$$

Exact solution:

$$x = 3e^t - t^2 - 2t - 2$$

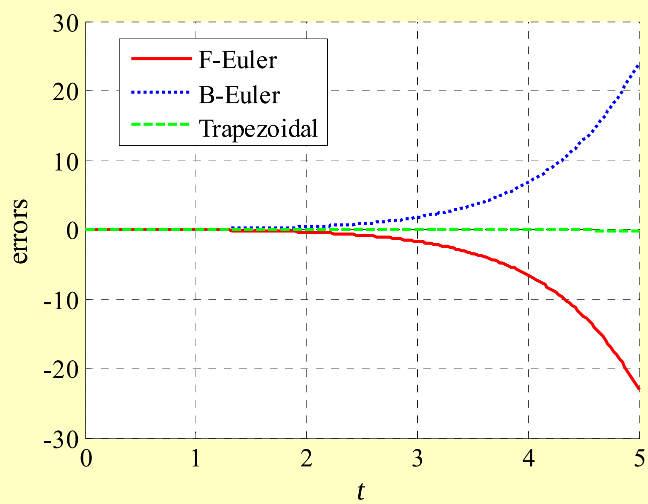


Example 1 - Comparison

$$x' = x + t^2$$
$$x_0 = 1$$
$$h = 0.025$$

Exact solution:

$$x = 3e^t - t^2 - 2t - 2$$



Example 1 - Comparison (Decreasing *h*)

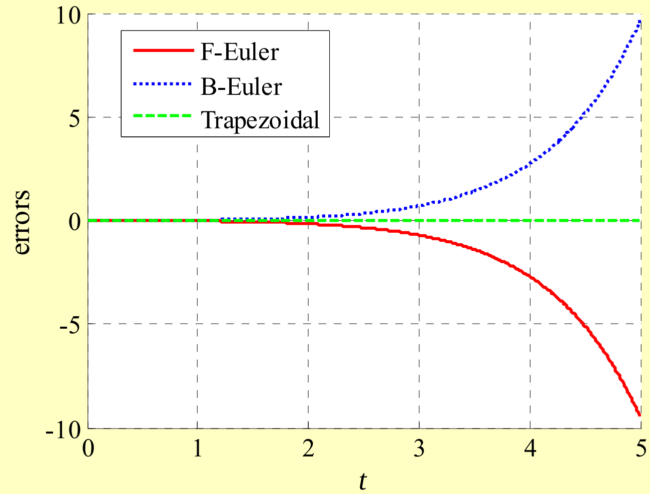
$x' = x + t^2$

$x_0 = 1$

$h = 0.01$

Exact solution:

$x = 3e^t - t^2 - 2t - 2$



Example 2 - Forward Euler

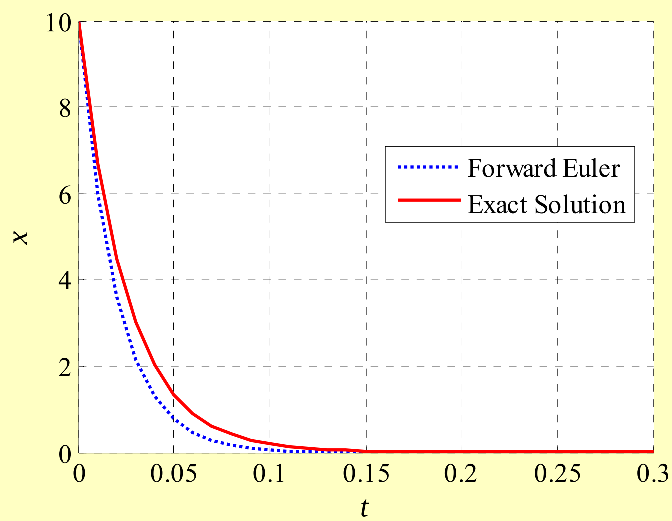
$x' = -40x$

$x_0 = 10$

$h = 0.01$

Exact solution:

$x = 10e^{-40t}$



Example 2 - Backward Euler

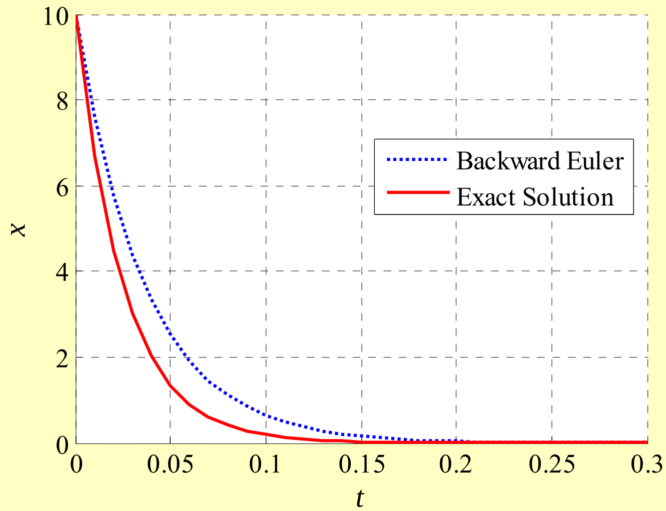
$x' = -40x$

$x_0 = 10$

$h = 0.01$

Exact solution:

$x = 10e^{-40t}$



Dr. J.E. Rayas-Sánchez 17

Example 2 - Trapezoidal

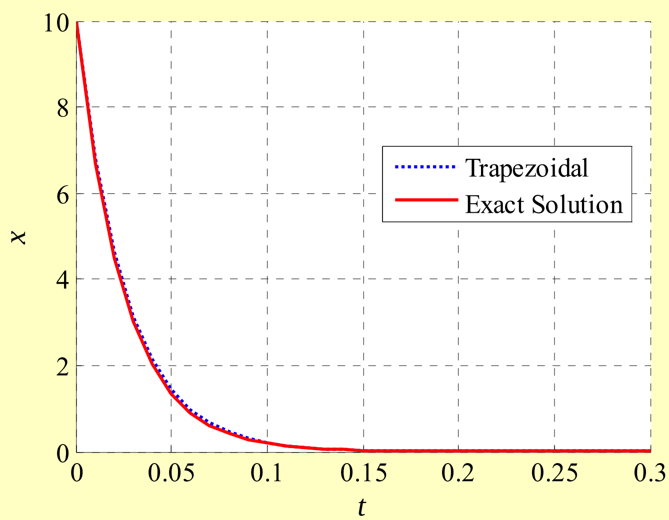
$x' = -40x$

$x_0 = 10$

$h = 0.01$

Exact solution:

$x = 10e^{-40t}$



Dr. J.E. Rayas-Sánchez 18

Example 2 - Comparison

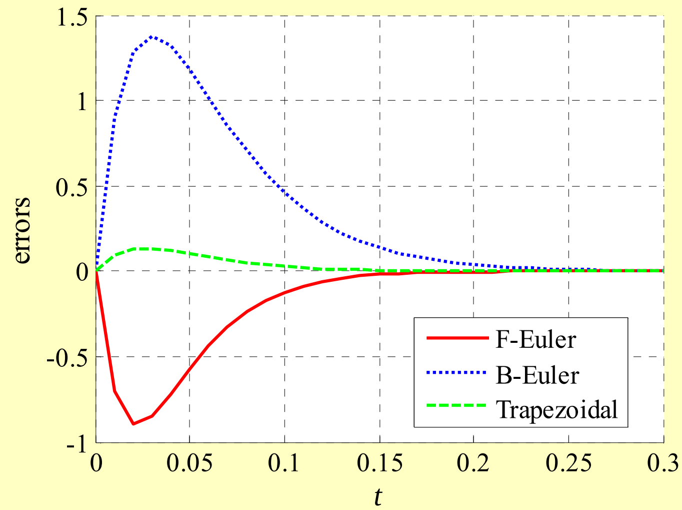
$x' = -40x$

$x_0 = 10$

$h = 0.01$

Exact solution:

$x = 10e^{-40t}$



Example 2 - Comparison (Decreasing *h*)

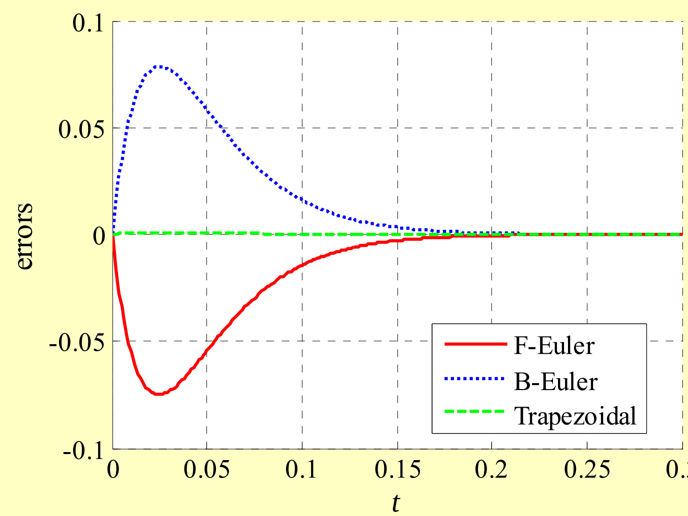
$x' = -40x$

$x_0 = 10$

$h = 0.001$

Exact solution:

$x = 10e^{-40t}$



Stability of Integration

- Test differential equation: $x' = \lambda x$ solution: $x = x_0 e^{\lambda t}$
- Forward Euler:

$$x_i = (1 + \lambda h)^i x_0 \qquad |1 + \lambda h| \le 1$$

- Backward Euler:

$$x_i = \left(\frac{1}{1 - \lambda h}\right)^i x_0 \qquad \left|\frac{1}{1 - \lambda h}\right| \le 1$$

- Trapezoidal:

$$x_i = \left(\frac{1 + \lambda h/2}{1 - \lambda h/2}\right)^i x_0 \qquad \left|\frac{1 + \lambda h/2}{1 - \lambda h/2}\right| \le 1$$

Dr. J.E. Rayas-Sánchez 21

Linear Systems of Differential Equations

$$\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, t) \qquad \boldsymbol{x}(t = 0) = \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \qquad \boldsymbol{x}'_i = \boldsymbol{f}(\boldsymbol{x}_i, t_i)$$

$$\boldsymbol{x}' = \boldsymbol{B}\boldsymbol{x} + \boldsymbol{w}(t)$$

Backward Euler:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{x}'_{i+1} = \boldsymbol{x}_i + h(\boldsymbol{B}\boldsymbol{x}_{i+1} + \boldsymbol{w}_{i+1})$$

$$(\mathbf{1} - h\boldsymbol{B})\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{w}_{i+1}$$

Trapezoidal:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{x}'_{i+1} + \boldsymbol{x}'_i) = \boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{B}\boldsymbol{x}_{i+1} + \boldsymbol{w}_{i+1} + \boldsymbol{B}\boldsymbol{x}_i + \boldsymbol{w}_i)$$

$$(\mathbf{1} - \frac{h}{2}\boldsymbol{B})\boldsymbol{x}_{i+1} = (\mathbf{1} + \frac{h}{2}\boldsymbol{B})\boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{w}_{i+1} + \boldsymbol{w}_i)$$

Dr. J.E. Rayas-Sánchez 22

Transient Solution of Linear Circuits

Let the system equations in the Laplace domain be

$$(\boldsymbol{H}_1 + s\boldsymbol{H}_2)\boldsymbol{X} = \boldsymbol{W}$$

Taking the inverse Laplace Transform,

$$\boldsymbol{H}_1\boldsymbol{x} + \boldsymbol{H}_2\boldsymbol{x}' = \boldsymbol{w}$$

$$\boldsymbol{H}_2\boldsymbol{x}' = \boldsymbol{w} - \boldsymbol{H}_1\boldsymbol{x}$$

Transient Solution of Linear Circuits (cont.)

$$\boldsymbol{H}_2\boldsymbol{x}' = \boldsymbol{w} - \boldsymbol{H}_1\boldsymbol{x}$$

Forward Euler:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{x}'_i$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + h\boldsymbol{H}_2\boldsymbol{x}'_i$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + h(\boldsymbol{w}_i - \boldsymbol{H}_1 x_i)$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = (\boldsymbol{H}_2 - h\boldsymbol{H}_1)\boldsymbol{x}_i + h\boldsymbol{w}_i$$

($\boldsymbol{H}_2$ might be singular)

Transient Solution of Linear Circuits (cont.)

$$\boldsymbol{H}_2\boldsymbol{x}' = \boldsymbol{w} - \boldsymbol{H}_1\boldsymbol{x}$$

Backward Euler:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{x}'_{i+1}$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + h\boldsymbol{H}_2\boldsymbol{x}'_{i+1}$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + h(\boldsymbol{w}_{i+1} - \boldsymbol{H}_1\boldsymbol{x}_{i+1})$$

$$(\boldsymbol{H}_2 + h\boldsymbol{H}_1)\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + h\boldsymbol{w}_{i+1}$$

Transient Solution of Linear Circuits (cont.)

$$\boldsymbol{H}_2\boldsymbol{x}' = \boldsymbol{w} - \boldsymbol{H}_1\boldsymbol{x}$$

Trapezoidal:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{x}'_{i+1} + \boldsymbol{x}'_i)$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{H}_2\boldsymbol{x}'_{i+1} + \boldsymbol{H}_2\boldsymbol{x}'_i)$$

$$\boldsymbol{H}_2\boldsymbol{x}_{i+1} = \boldsymbol{H}_2\boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{w}_{i+1} - \boldsymbol{H}_1\boldsymbol{x}_{i+1} + \boldsymbol{w}_i - \boldsymbol{H}_1\boldsymbol{x}_i)$$

$$(\boldsymbol{H}_2 + \frac{h}{2}\boldsymbol{H}_1)\boldsymbol{x}_{i+1} = (\boldsymbol{H}_2 - \frac{h}{2}\boldsymbol{H}_1)\boldsymbol{x}_i + \frac{h}{2}(\boldsymbol{w}_{i+1} + \boldsymbol{w}_i)$$

MNA Formulation for Transient Analysis

- The MNA equation

$$\begin{bmatrix} \boldsymbol{A}_1\boldsymbol{Y}_1\boldsymbol{A}_1^T & \boldsymbol{A}_2 \\ \boldsymbol{Y}_2\boldsymbol{A}_2^T & \boldsymbol{Z}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{V}_n \\ \boldsymbol{I}_2 \end{bmatrix} = \begin{bmatrix} -\boldsymbol{A}_1\boldsymbol{J}_1 \\ \boldsymbol{W}_2 \end{bmatrix} \qquad \boldsymbol{HX} = \boldsymbol{W}$$

  can be formulated without using oriented graphs or incidence matrices $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$

- $\boldsymbol{H}$ and $\boldsymbol{W}$ can be directly formulated by inspection, using stamps
- Once $\boldsymbol{H}$ is known, the L and C elements can be separated so that

$$(\boldsymbol{H}_1 + s\boldsymbol{H}_2)\boldsymbol{X} = \boldsymbol{W}$$

  which can be solved in the time domain using the previous methods