Time-Domain Macromodel Extraction

Bob Ross, Teraspeed Labs, USA
bob@teraspeedlabs.com

European Virtual IBIS Summit
(with Virtual SPI2021)
May 12, 2021

Agenda

- Goals
- Preliminaries and references
- Mathematical identities for time-domain extraction
- Laplace transform extraction flow
- Duality
- Last column mathematics for companion matrix functions
- Adding constraints
- Conclusions
- References

Teraspeed Labs

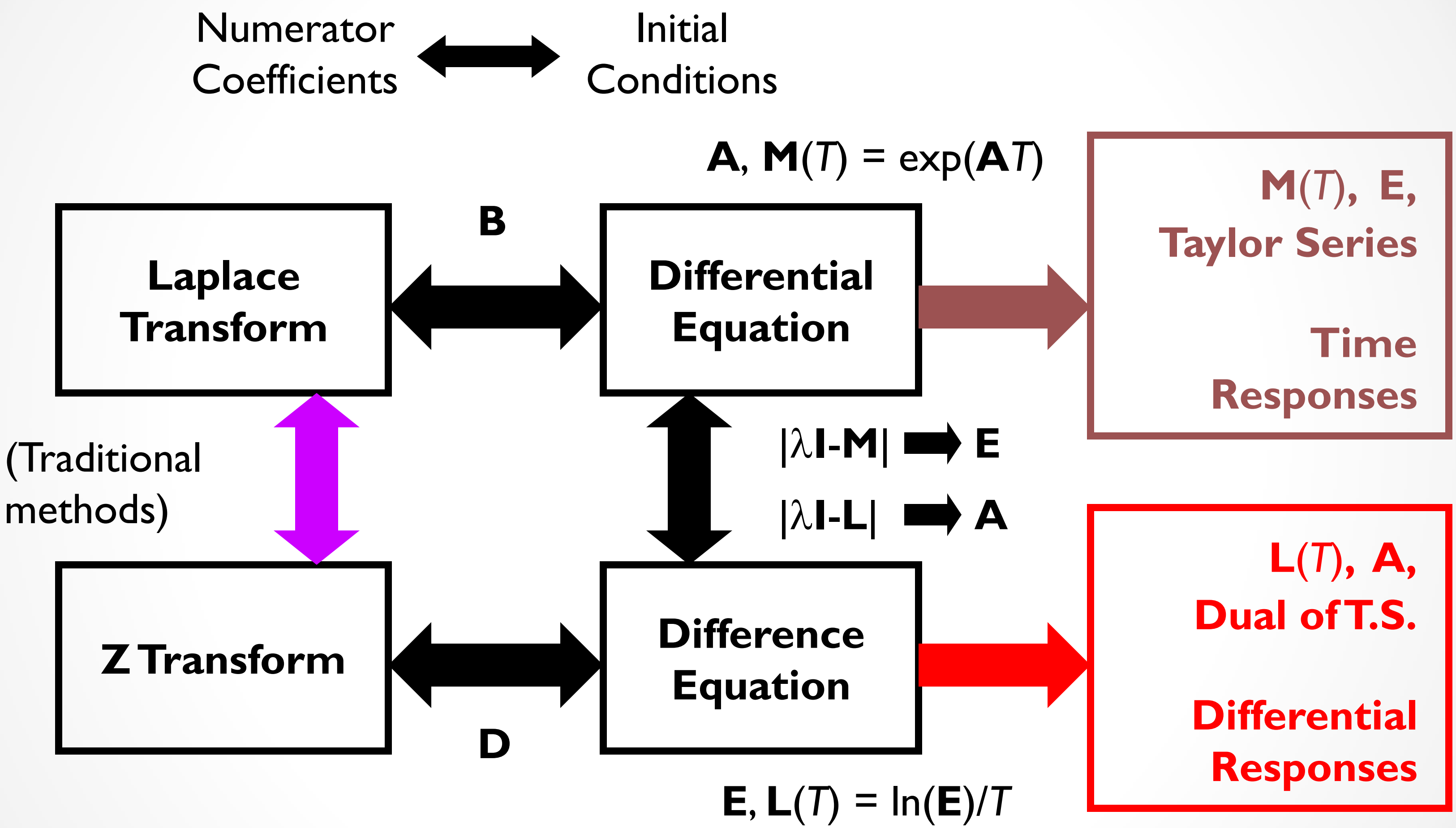
Goals

- Goal – Low-order Laplace transform network function extraction from time-domain measurements (or simulations) as a ratio of polynomials in s
  - Macromodel generation
  - Noisy measurements
  - Uncoupled networks
  - Least squared error steepest descent algorithm
- Show some not so well-known mathematical identities
  - Duality
  - Last column mathematics for functions of companion matrices
- Based on original correspondence 1969 – 1972 with Janez Valand (Yugoslavia/Croatia) and actual product implementation (1990's)
- Derivations and proofs not shown
  - Proofs based on power series expansions and companion matrix relationships

Special Notation - Equations

Laplace Transform	$X(s) = \dfrac{a_{n-1}s^{n-1} + \cdots + a_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_0},$
Differential Equation	$x^n(t) + b_{n-1}x^{n-1}(t) + \cdots + b_0 x(t) = 0$ initial conditions, $x(0), \cdots, x^{n-1}(0),$
Difference Equation	$x_n(t) + d_{n-1}x_{n-1}(t) + \cdots + d_0 x_0(t) = 0$ initial conditions, $x_0(0), \cdots, x_{n-1}(0),$
Z Transform	$Z(z) = \dfrac{z(c_{n-1}z^{n-1} + \cdots + c_0)}{z^n + d_{n-1}z^{n-1} + \cdots + d_0}.$

Conversions and Responses



Differential Equations

$d\mathbf{x}(t)/dt = \mathbf{A}\mathbf{x}(t)$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \\ -b_0 & -b_1 & \cdots & -b_{n-1} \end{bmatrix}$$

$\mathbf{x}(t+T) = \mathbf{M}\mathbf{x}(t)$

$\mathbf{M} = \exp(\mathbf{A}T)$

Difference Equations

$\mathbf{z}(t+T) = \mathbf{E}\mathbf{z}(t)$

$$\mathbf{E} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \\ -d_0 & -d_1 & \cdots & -d_{n-1} \end{bmatrix}$$

$d\mathbf{z}(t)/dt = \mathbf{L}\mathbf{z}(t)$

$\mathbf{E} = \exp(\mathbf{L}T)$

$\mathbf{L} = \ln(\mathbf{E})/T$

Companion Matrices

Teraspeed Labs

Differential Equations

$\mathbf{x}(t) = [x^0(t), x^1(t), \cdots, x^{n-1}(t)]^T$,

$\mathbf{a} = [a_{n-1}, \cdots, a_0]^T$,

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ b_{n-1} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ b_2 & b_3 & \cdots & 1 & 0 \\ b_1 & b_2 & \cdots & b_{n-1} & 1 \end{bmatrix}$$

$\mathbf{a} = \mathbf{B}\mathbf{x}(0)$

Difference Equations

$\mathbf{z}(t) = [x_0(t), x_1(t), \cdots, x_{n-1}(t)]^T$

$\mathbf{c} = [c_{n-1} \cdots c_0]^T$

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ d_{n-1} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ d_2 & d_3 & \cdots & 1 & 0 \\ d_1 & d_2 & \cdots & d_{n-1} & 1 \end{bmatrix}$$

$\mathbf{c} = \mathbf{D}\mathbf{z}(0)$

Teraspeed Labs

Recursive Taylor Series (Repeat b and c)

a) Initialize: $i = 1, \ldots, n-1$

$$x(0) = a_{n-1} \qquad x^{i}(0) = a_{n-1-i} - \sum_{j=0}^{i-1} b_{n-i-j}\, x^{j}(0)$$

b) Extend: $i = n, \ldots, p$

$$x^{i}(t) = -\sum_{j=0}^{n-1} b_j\, x^{i-n-j}(t)$$

c) Next time step: $i = 0, \ldots, n-1$ (Taylor series)

$$x^{i}(t+T) = \sum_{j=i}^{p} x^{j}(t)\frac{T^{j-i}}{(j-i)!}$$

R. I. Ross, "Evaluating the Transient Response of a Network Function," *Proc. IEEE*, vol.55, pp. 615-616, May 1967

Terasppeed Labs

Differential Eq'n Sensitivities

$$\left[\frac{\partial x(t)}{\partial a_0}, \frac{\partial x(t)}{\partial a_1}, \dots, \frac{\partial x(t)}{\partial a_{n-1}}\right]^T =$$

$$\left[\frac{\partial x(t)}{\partial a_0}, \frac{\partial x^1(t)}{\partial a_0}, \dots, \frac{\partial x^{n-1}(t)}{\partial a_0}\right]^T$$

$$\left[\frac{\partial x(t)}{\partial b_0}, \frac{\partial x(t)}{\partial b_1}, \dots, \frac{\partial x(t)}{\partial b_{n-1}}\right]^T =$$

$$\left[\frac{\partial x(t)}{\partial b_0}, \frac{\partial x^1(t)}{\partial b_0}, \dots, \frac{\partial x^{n-1}(t)}{\partial b_0}\right]^T$$

$$\frac{\partial x^i(t)}{\partial a_j \partial b_k} = \frac{\partial x^{i+j+k}(t)}{\partial a_0 \partial b_0}$$

Difference Eq'n Sensitivities

$$\left[\frac{\partial x(t)}{\partial c_0}, \frac{\partial x(t)}{\partial c_1}, \dots, \frac{\partial x(t)}{\partial c_{n-1}}\right]^T =$$

$$\left[\frac{\partial x(t)}{\partial c_0}, \frac{\partial x_1(t)}{\partial c_0}, \dots, \frac{\partial x_{n-1}(t)}{\partial c_0}\right]^T$$

$$\left[\frac{\partial x(t)}{\partial d_0}, \frac{\partial x(t)}{\partial d_1}, \dots, \frac{\partial x(t)}{\partial d_{n-1}}\right]^T =$$

$$\left[\frac{\partial x(t)}{\partial d_0}, \frac{\partial x_1(t)}{\partial d_0}, \dots, \frac{\partial x_{n-1}(t)}{\partial d_0}\right]^T$$

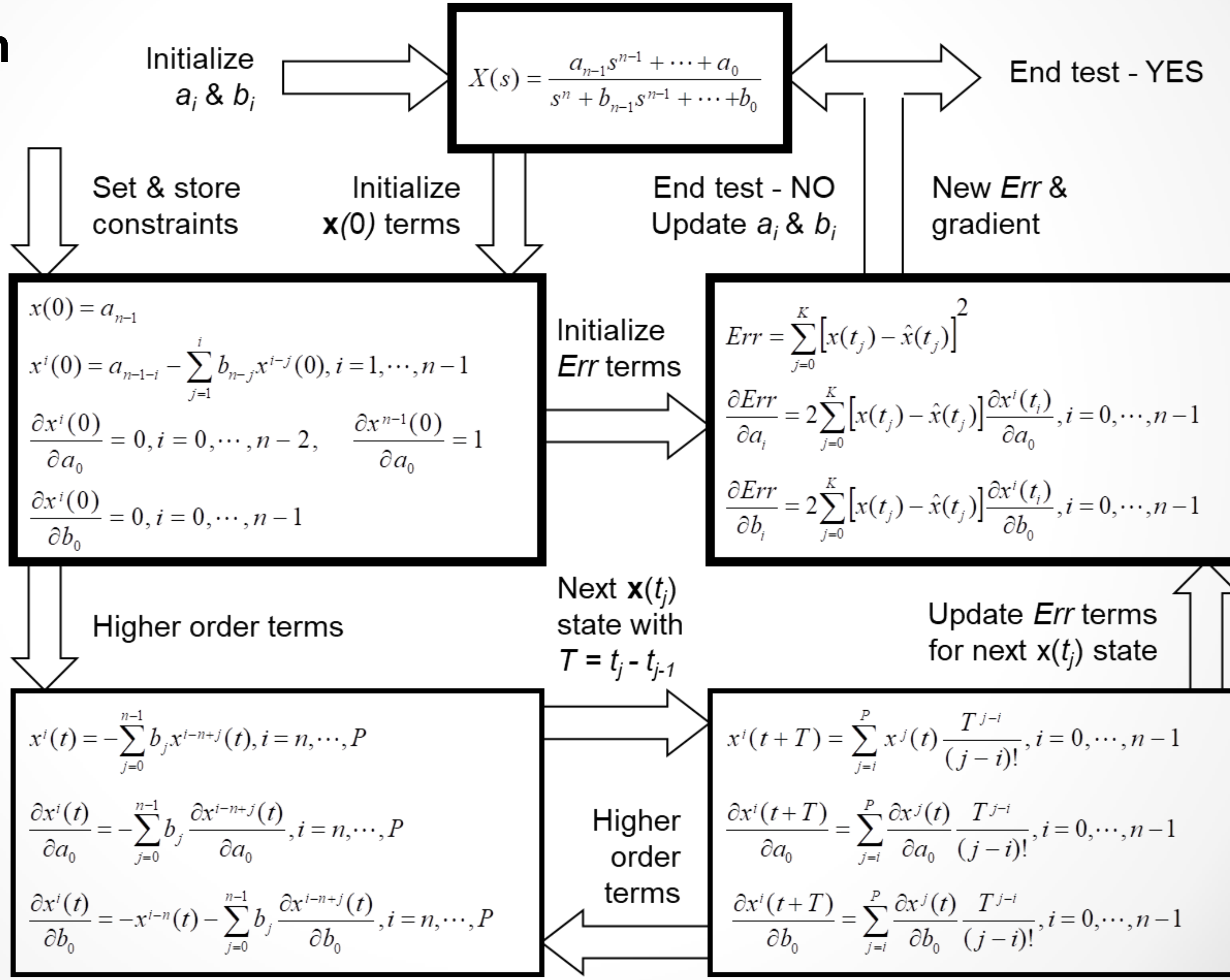
$$x_j(t) = x(t + jT), \quad j = 0, \dots, n-1$$

$$\frac{\partial x_i(t)}{\partial c_j \partial d_k} = \frac{\partial x_{i+j+k}(t)}{\partial c_0 \partial d_0}$$

Teraspeed Labs

Laplace Transform Extraction (T.S.)

Expanded on next slide



Initialize a_i & b_i

$$X(s) = \frac{a_{n-1}s^{n-1} + \cdots + a_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_0}$$

End test - YES

Set & store constraints

Initialize **x**(0) terms

End test - NO
Update a_i & b_i

New *Err* & gradient

$$x(0) = a_{n-1}$$

$$x^i(0) = a_{n-1-i} - \sum_{j=1}^{i} b_{n-j} x^{i-j}(0), i = 1, \cdots, n-1$$

$$\frac{\partial x^i(0)}{\partial a_0} = 0, i = 0, \cdots, n-2, \qquad \frac{\partial x^{n-1}(0)}{\partial a_0} = 1$$

$$\frac{\partial x^i(0)}{\partial b_0} = 0, i = 0, \cdots, n-1$$

Initialize *Err* terms

$$Err = \sum_{j=0}^{K} \left[x(t_j) - \hat{x}(t_j)\right]^2$$

$$\frac{\partial Err}{\partial a_i} = 2\sum_{j=0}^{K} \left[x(t_j) - \hat{x}(t_j)\right] \frac{\partial x^i(t_j)}{\partial a_0}, i = 0, \cdots, n-1$$

$$\frac{\partial Err}{\partial b_i} = 2\sum_{j=0}^{K} \left[x(t_j) - \hat{x}(t_j)\right] \frac{\partial x^i(t_j)}{\partial b_0}, i = 0, \cdots, n-1$$

Higher order terms

Next **x**(t_j) state with $T = t_j - t_{j-1}$

Update *Err* terms for next x(t_j) state

$$x^i(t) = -\sum_{j=0}^{n-1} b_j x^{i-n+j}(t), i = n, \cdots, P$$

$$\frac{\partial x^i(t)}{\partial a_0} = -\sum_{j=0}^{n-1} b_j \frac{\partial x^{i-n+j}(t)}{\partial a_0}, i = n, \cdots, P$$

$$\frac{\partial x^i(t)}{\partial b_0} = -x^{i-n}(t) - \sum_{j=0}^{n-1} b_j \frac{\partial x^{i-n+j}(t)}{\partial b_0}, i = n, \cdots, P$$

Higher order terms

$$x^i(t+T) = \sum_{j=i}^{P} x^j(t) \frac{T^{j-i}}{(j-i)!}, i = 0, \cdots, n-1$$

$$\frac{\partial x^i(t+T)}{\partial a_0} = \sum_{j=i}^{P} \frac{\partial x^j(t)}{\partial a_0} \frac{T^{j-i}}{(j-i)!}, i = 0, \cdots, n-1$$

$$\frac{\partial x^i(t+T)}{\partial b_0} = \sum_{j=i}^{P} \frac{\partial x^j(t)}{\partial b_0} \frac{T^{j-i}}{(j-i)!}, i = 0, \cdots, n-1$$

Initialization and Updates

- Similar flow for difference equations (Z-transform)
  - 2 cycles of unconstrained difference equation optimization to get Laplace transform starting coefficients
- Coefficient updates
  - Standard gradient methods by step size estimation: slow, inefficient
  - Other methods tried including Modified Gauss–Newton method and Linear fit algorithms documented in the literature gave faster convergence
- Last column mathematics used – next
- Constraints implemented in the best fit solution
- Last step – solve for poles and zeros (not needed during the optimization process)

Last Column Mathematics for Functions of the Companion Matrix

$$x(t+T) = M(T)x(t)$$

$$M(T) = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix}$$

$$x(t+T) = M(T)x(t) = M(T)B^{-1}a \ =$$

$$M(T)B^{-1}a = \begin{bmatrix} m_{n,n} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{2n-1,n} & \cdots & m_{n,n} \end{bmatrix} \begin{bmatrix} a_{n-1} \\ \vdots \\ a_0 \end{bmatrix}$$

$$m_{i,n} = -\sum_{j=0}^{n-1} b_j \, m_{i+j-n,n} \ , \ i = n+1, \cdots, 2n-1$$

Last column of state transition matrix becomes numerator sensitivity values: $\dfrac{\partial x(t)}{\partial a_{i-1}} = m_{i,n}$

Teraspeed Labs

Multiplication Algorithm

$$W(k+1) = W(k)V$$

V = last column vector calculated once

$$v_i = \sum_{j=i+1}^{n} b_j \, v_{j-i,n} \, , i = 0, \cdots, n-1$$

$$w_{i,n}(k) = -\sum_{j=0}^{n} b_j \;\; w_{i+j-n,n}(k) \, , i = n+1 \, , \cdots , 2n-1$$

Extend

$$w_{i,n}(k+1) = \sum_{j=0}^{n-1} v_j \;\; w_{i+j,n}(k) \, , i = 1, \cdots, n$$

New last column result

R. Ross, "Efficient Method to Multiply Successively Functions of the Companion Matrix, and Applying this Method to Evaluate Transient Response," Conference *Record, Fifth Asilomar Conference on Circuits and Systems*, Pacific Grove, California, pp. 261-265, Nov. 1971

Teraspeed Labs

Some Last Column Mathematics References

- W. E. Thomson, "Evaluation of Transient Response," *Proc. IEEE*, Nov. 1966, pp.1584
  - Several relationships between state transition matrix elements include last-column relationships
  - Relationships can be applied to any function of a companion matrix
- J. Valand, "Calculation of Transient Response," *Electron. Letters*, vol.4, June 28, 1969, p. 260.
  - A coefficients and last column of state transition matrix used to calculate transient response

Characteristic Equation

- Cayley-Hamilton Theorem – a matrix satisfies its own characteristic equation
- Computation of characteristic equations:
  - Based on built in mathematical functions
  - Or based on calculating traces (sum of diagonal terms) of powers of **M** or **L**

Teraspeed Labs

Differential Equation

$|\lambda \mathbf{I}-\mathbf{M}| =$

$$\lambda^n + d_{n-1}\lambda^{n-1} + \cdots + d_0 = 0$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \\ -b_0 & -b_1 & \cdots & -b_{n-1} \end{bmatrix}$$

Difference Equation

$$\mathbf{E} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \\ -d_0 & -d_1 & \cdots & -d_{n-1} \end{bmatrix}$$

$|\lambda \mathbf{I}-\mathbf{L}| =$

$$\lambda^n + b_{n-1}\lambda^{n-1} + \cdots + b_0 = 0$$

Trace Calculation: for Differential to Difference Equations

$$T_k = \sum_{j=1}^{n} j\, b_j\, m_{j,n}(kT)\ , k = 1, \cdots, n$$

$$\begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_n \end{bmatrix} = -\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ T_1 & 2 & \cdots & 0 & 0 \\ \vdots & T_1 & \ddots & \vdots & \vdots \\ T_{n-2} & \vdots & \ddots & n-1 & 0 \\ T_{n-1} & T_{n-2} & \cdots & T_1 & n \end{bmatrix} \begin{bmatrix} d_{n-1} \\ d_{n-2} \\ \vdots \\ d_0 \end{bmatrix}$$

Simplified by using last column mathematics
Similar mathematics for Difference to Differential Equations

Lofti Zedeh, Charles Desoer, *Linear System Theory: The State Space Approach*, 1963, pp. 304-305

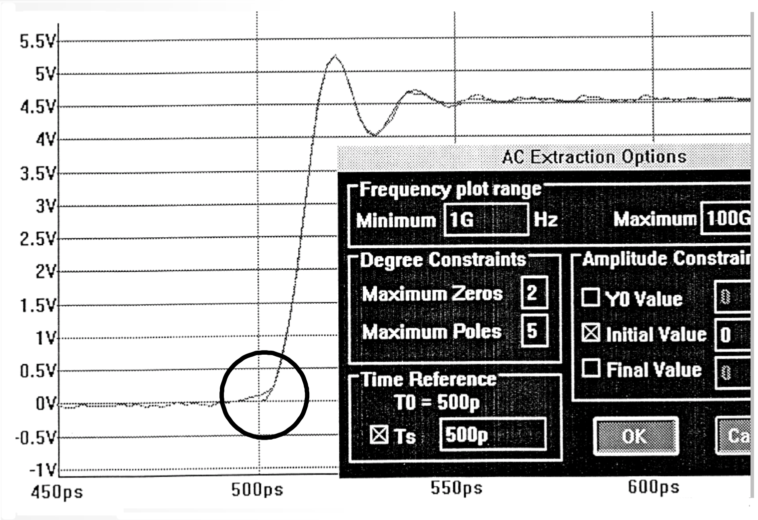
Maxime Böcher, *Introduction to Higher Algebra*, 1922, p. 297

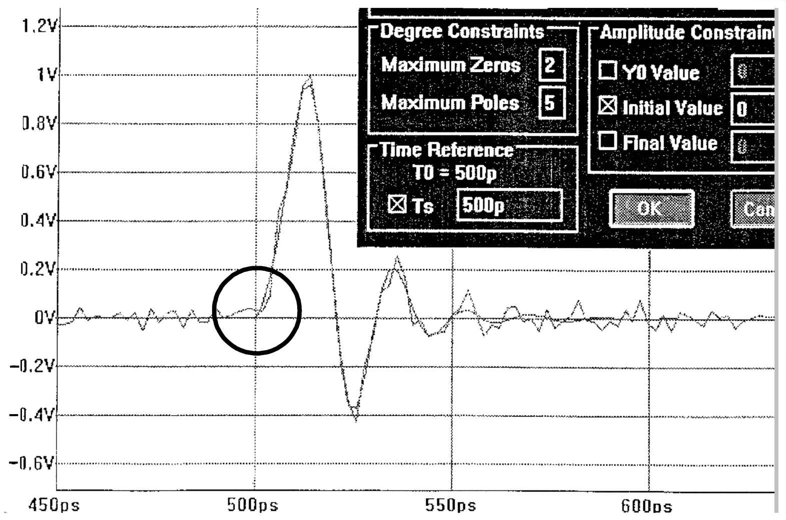
Teraspeed Labs

Available Constraints

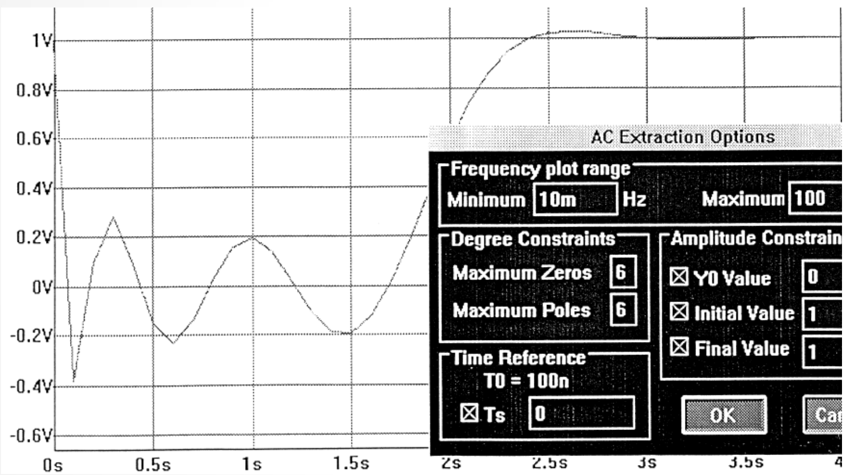
- **Denominator degree (number of poles)**
- **Numerator degree (number of zeros, where lower degree produces less leading edge ripple)**
- **Set t_S start value, DC offset (y_0)**
- **Minimum and maximum frequencies for log frequency domain plots**
- **Initial value**
- **Final value**

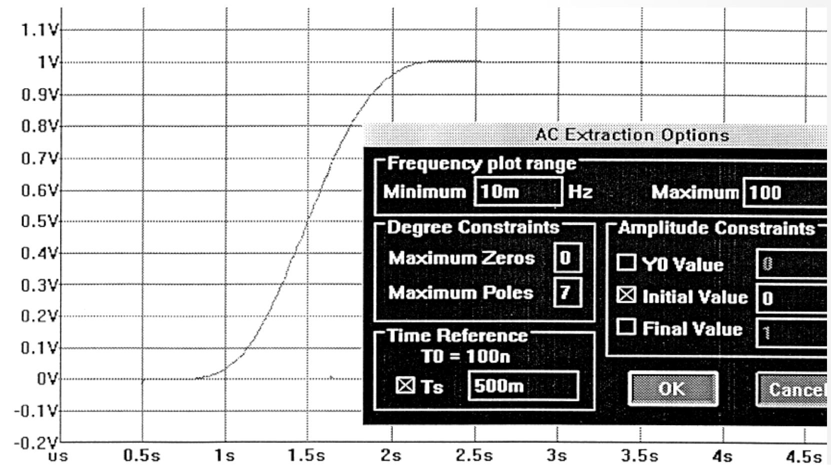
Constraints (Oscilloscope Step and Impulse Response Extractions)





Constraints – Maximally Flat Envelope Delay Networks





Conclusions

- Early (1970's) algorithm outlined for physical measurement-based time-domain extraction
- Most calculations done in place to save memory
- Most calculations used last-column matrix mathematics
- Many subroutines worked in both the difference equation and differential equation domains
- Laplace transform formulation allows practical constraints to be implemented
- Result is a Laplace transform as a polynomial ratio as N(s)/D(s) from a time-domain response

European IBIS Summits www.ibis.org/summits/

- J. E. Schutt-Ainé, “IBIS-Compatible Macromodel and Interconnect Simulation Techniques” (2018)
- T. Bradde, S. Grivet-Talocia, M. De Stefano, A. Zanco, “On Automated Generation of Behavioral Parameterized Macromodels, Part I: Algorithmic Aspects” (2018)
- M. De Stefano, S. Grivet-Talocia, T. Bradde, A. Zanco, “On Automated Generation of Behavioral Parameterized Macromodels, Part II: SPICE Equivalents and Applications” (2018)
- A. Zanco, E. Fevola, S. Grivet-Talocia, T. Bradde, M. De Stefano, “An Adaptive algorithm for Fully Automated Extraction of Passive Parameterized Macromodels” (2019)
- B. Ross, “Continuous and Discrete Modeling for IBIS-AMI (2011)
- B. Ross, “Time Response Utility” (2011 and .xls utility uploaded)

Other References and Implementations

- S. Grivet-Talocia, B. Gustavsen, *Passive Macromodeling, Theory and Applications,* Wiley, 2016
- J. Cadzow, "Recursive Digital Filter Synthesis via Gradient Based Algorithms," *IEEE Transactions on Audio and Electroacoustics*, Oct. 1976 (difference equation duality)
- B. Ross, "Taylor Series Duality," *Proc. 7th IEEE Workshop of Signal Propagation on Interconnects*, Siena, Italy, May 11-14, 2003, pp. 97-100

Teraspeed Labs

EPEPS2014 – Multnomah Falls, OR

Teraspeed Labs