Volterra Series Based RF Power Amplifier Behavioral Modeling

Anding Zhu, and Thomas J. Brazil

RF & Microwave Research Group
University College Dublin
Dublin 4, Ireland

Tel: +353-1-7161912
Fax: +353-1-2830921
E-mail: anding.zhu@ucd.ie
Outline

- Introduction
  - What is Volterra Series?
  - Why Volterra Series?
- Volterra Model Extraction
  - Discrete Time Domain
- Simplified Model Structures
  - Special Cases
  - Direct Pruning
  - Reformatting the Expansion
- Conclusions
Volterra Series in the Discrete Time Domain

**Linear system with memory**

\[ y(n) = \sum_{i=0}^{m-1} w(i) \times x(n-i) \]

\[ = w(0)x(n) + w(1)x(n-1) + \cdots \]

**Nonlinear system without memory**

\[ y(n) = \sum_{i=0}^{m-1} a_i[x(n)]^i \]

\[ = a_0 + a_1x(n) + a_2[x(n)]^2 + \cdots \]

**1st-order Volterra kernel**

\[ y(n) = \sum_{i=0}^{m-1} h_1(i) x(n-i) \]

**2nd-order Volterra kernel**

\[ = \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} h_2(i, j) x(n-i)x(n-j) \]

**3rd-order Volterra kernel**

\[ + \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} \sum_{k=0}^{m-1} h_3(i, j, k) x(n-i)x(n-j)x(n-k) + \cdots \]
Volterra Series in the Discrete Frequency Domain

Using the Discrete Fourier Transform (DFT), we could write the Volterra series in the frequency domain as

\[ Y_n(m) = H_1(m)X_n(m) + \sum_{k_1, k_2 = -M+1}^{M} H_2(k_1, k_2)X_n(k_1)X_n(k_2) \]

\[ + \sum_{k_1, k_2, k_3 = -M+1}^{M} H_3(k_1, k_2, k_3)X_n(k_1)X_n(k_2)X_n(k_3) + \cdots \]

Multi-dimensional DFT of Volterra Kernels
In wireless communication systems, a PA is normally used to transmit modulated signals where only the envelope carries useful information and even-order nonlinear components can be omitted. Therefore, the Volterra model can be written in the low-pass equivalent format as

\[
\tilde{y}(n) = \sum_{i=0}^{m-1} h_i(i) \tilde{x}(n-i) + \sum_{i_1=0}^{m-1} \sum_{i_2=i_1}^{m-1} \sum_{i_3=0}^{m-1} h_3(i_1, i_2, i_3) \tilde{x}(n-i_1) \tilde{x}(n-i_2) \tilde{x}^* (n-i_3) \\
+ \sum_{i_1=0}^{m-1} \sum_{i_2=i_1}^{m-1} \sum_{i_3=i_2}^{m-1} \sum_{i_4=i_3}^{m-1} \sum_{i_5=i_4}^{m-1} h_5(i_1, i_2, i_3, i_4, i_5) \tilde{x}(n-i_1) \tilde{x}(n-i_2) \tilde{x}(n-i_3) \tilde{x}^* (n-i_4) \tilde{x}^* (n-i_5) \\
+ \cdots
\]
Volterra Series Based Models

Advantages

- A firm mathematical foundation
  -- closed form expression.
- **Output is linear in relation to the coefficients**
  -- model can be extracted in a direct way by using linear optimization.
- Easily implemented in hardware
  -- the structure is the same as for linear filters, number of coefficients can be decided in advance.

Problems

- **Model structure too complicated**
  -- number of coefficients increases exponentially with the degree of nonlinearities and with memory length. (e.g., 4320 for 5th order with memory 8).
- **Model extraction quite difficult**
  -- involves complicated measurement procedures and complicated algorithms.
- **Convergence problem**
  -- cannot model discontinuities.
Outline

- Introduction
  - What is Volterra Series
  - Why Volterra Series
- Volterra Model Extraction
  - Discrete time domain
- Simplified Model Structures
  - Special Cases
  - Direct Pruning
  - Reformat the Expansion
- Conclusions
Several methodologies have been used for Volterra kernel extraction, including spectral analysis in the frequency domain, least squares in the time-domain, some solutions in the mixed-domain, and so on. However, in terms of excitation signals, they can be generally clustered in two groups:

1. Specialized Inputs (Deterministic Signals)
   - Use Two-/Multi-tone or Impulsive signals.
   - Power and frequency sweeps to cover multiple amplitude and frequency ranges.
   - Not experimentally efficient: involve complicated measurement procedures.

2. Arbitrary Inputs (Statistical Signals)
   - Use arbitrary sampled inputs and outputs in the discrete time domain.
   - Simple measurement configuration, only one measurement needed.
   - Can formulate the model extraction problem in a vector-matrix form, and then find the solution by employing least squares estimation.
   - A critical issue is that the matrix could be ill-conditioned.
   - These excitation inputs have to be relatively broadband to cover the entire range of frequencies of interest.
   - Has become more popular in recent years since practical test equipment is available.
Example: Test Bench for Low-pass Model Extraction
(Discrete Time Domain)

Send Test I/Q Signal Back to ADS

Send Simulated I/Q Signal to ESG

Generate I/Q Signal in ADS

Measured I/Q Signal Back in ADS

Arbitrary Signal Source, e.g., W-CDMA (3.84 MChips/Sec)
Example: Least Squares

Data gathered:

**input:** \( x(1), x(2), \ldots, x(N) \)

**output:** \( d(1), d(2), \ldots, d(N) \)

Formulate input and output vector:

\[
\begin{align*}
\mathbf{y} &= \begin{bmatrix} y(1) \\ y(2) \\ \vdots \end{bmatrix}_{N \times 1} \\
\mathbf{X} &= \begin{bmatrix} x_1(1), x_2(1), \ldots \\ x_1(2), x_2(2), \ldots \\ \vdots \end{bmatrix}_{N \times M}
\end{align*}
\]

Parameters to be extracted:

\[
\mathbf{\theta} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \end{bmatrix}_{M \times 1}
\]

then \( \mathbf{y} = \mathbf{X} \mathbf{\theta} \) and \( e(k) = d(k) - y(k) \)

\[
\min \quad J(\mathbf{\theta}) = \sum_{k=1}^{N} |e(k)|^2 = \mathbf{e}^H \mathbf{e}
\]

Solution \( \hat{\mathbf{\theta}}(n) = (\mathbf{X}^H \mathbf{X})^{-1} \mathbf{X}^H \mathbf{d} \)
Time-Shift Problem in Extraction

In some applications, e.g., predistortion, fast adaptive linear filtering algorithms are normally used to extract the nonlinear Volterra kernels, but we may encounter a time-shift problem as follows:

### Linear System

**Input vector stepped from time \((n)\) to \((n+1)\):**

\[
X_n \rightarrow X_{n+1}
\]

\[
\begin{bmatrix}
  x(n+1) \\
  x(n) \\
  x(n-1) \\
  \vdots \\
  x(n-m+2) \\
  x(n-m+1)
\end{bmatrix}
\]

\[
\begin{bmatrix}
  x(n+1) \\
  x(n) \\
  x(n-1) \\
  \vdots \\
  x(n-m+2) \\
  x(n-m+1)
\end{bmatrix}
\]

**time-shift kept**

### Volterra System

**Output vector stepped from time \((n)\) to \((n+1)\):**

\[
X_n \rightarrow X_{n+1}
\]

\[
\begin{bmatrix}
  x(n+1) \\
  x^3(n+1) \\
  x^2(n+1)x(n) \\
  \vdots \\
  x(n) \\
  x(n-1) \\
  x(n-2) \\
  x^3(n) \\
  x^3(n-1) \\
  x^3(n-2) \\
  \vdots \\
  x(n)x(n-1)x(n-2)
\end{bmatrix}
\]

\[
\begin{bmatrix}
  x(n+1) \\
  x(n) \\
  x(n-1) \\
  \vdots \\
  x(n+1)x(n)x(n-1) \\
  x(n-2) \\
  x^3(n-2) \\
  x^2(n)x(n-2) \\
  \vdots
\end{bmatrix}
\]

**time-shift lost**
New Volterra Model Structure
Using V-vector Algebra

A key advantage of this approach is that any fast algorithm for linear filter coefficient estimation (kernel extraction) can be used to extract the model.
Updating of Coefficients in Parallel
(Real-time system)

\[ x(n) \rightarrow \text{Sub-filter 1} \rightarrow y_1(n) \]
\[ \quad \text{Sub-filter 2} \rightarrow y_2(n) \]
\[ \quad \text{Sub-filter } k \rightarrow y_k(n) \]
\[ \quad \sum e_1(n) + e_2(n) + e_k(n) \]
\[ \quad y(n) - e(n) \rightarrow d(n) \]

\[ e(n) = d(n) - y(n) \]
\[ e_i(n) = d_i(n) - y_i(n) \]
\[ \min J_i(n) = \sum_{k=1}^{n} \lambda^{n-k} |e_i(n)|^2 \]

- Shorter time cycle for each iteration
- But… delayed convergence
- Trade-offs possible

[Zhu, IMS’03]
Sample Experimental Results

(Time Domain)

Model Fidelity Evaluation: **Normalized Mean Square Error**

\[
NMSE := 10 \log \left\{ \frac{\sum_{k=1}^{M} \left[ (y_{I,k}^{\text{meas}} - y_{I,k}^{\text{mod}})^2 + (y_{Q,k}^{\text{meas}} - y_{Q,k}^{\text{mod}})^2 \right]}{\sum_{k=1}^{M} \left[ (y_{I,k}^{\text{meas}})^2 + (y_{Q,k}^{\text{meas}})^2 \right]} \right\}
\]
Sample Experimental Results  
(Frequency Domain)

<table>
<thead>
<tr>
<th>Performance</th>
<th>Measurement Results</th>
<th>Volterra Model</th>
<th>AM/AM AM/PM Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain (dB)</td>
<td>13.19</td>
<td>13.17</td>
<td>13.16</td>
</tr>
<tr>
<td>ACPR (dBC) (+/-5MHz)</td>
<td>43.5 / 44.4</td>
<td>43.3 / 44.5</td>
<td>40.0</td>
</tr>
<tr>
<td>ACPR (dBC) (+/-10MHz)</td>
<td>56.5 / 57.4</td>
<td>56.4 / 57.0</td>
<td>54.6</td>
</tr>
</tbody>
</table>
Outline

- Introduction
  - What is Volterra Series
  - Why Volterra Series
- Volterra Model Extraction
  - Discrete time domain
- Simplified Model Structures
  - Special Cases
  - Direct Pruning
  - Reformat the Expansion
- Conclusions
Special Cases: Separate Nonlinearity and Memory

- **Two-Box Models**
  - Memoryless Nonlinearity
    - Linear Filter → AM/PM → AM/AM → Wiener Model
    - Memoryless Nonlinearity
      - AM/PM → AM/AM → Linear Filter → Hammerstein Model

- **Three-Box Model**
  - Memoryless Nonlinearity
    - Filter → AM/PM → AM/AM → Filter

- **Only measure Nonlinearity (AM/AM and AM/PM) at the center frequency.**
- **Linear filters capture the memory.**
- **Cannot predict interaction.**
PolySpectral Model

- Single frequency measurement instead of multi-dimension characterization.
- Difficult to construct for higher-orders.

Other Similar Models
- Parallel Wiener Model [Kenney, TransMTT’02]
- Parallel Hammerstein Model
- … …

- Memory and nonlinearities modeled separately.
- Interaction can not be fully characterized.
- Extraction effort increased for high orders.
Outline

- Introduction
  - What is Volterra Series
  - Why Volterra Series
- Volterra Model Extraction
  - Discrete Time Domain
- Simplified Model Structures
  - Special Cases
  - Direct Pruning
  - Reformat the Expansion
- Conclusions
Pruned Model (I): Memory Polynomial Model

\[
x^3(n) \quad 0 \\
x^3(n-1) \\
x^3(n-2) \\
0 \\
x^3(n-3)
\]

*All off-diagonal coefficients are set to zero.

\[
y[n] = \sum_{q=1}^{Q} \sum_{k=0}^{m} a_{k,q} [x(n-k)]^q
\]

Kim, EL’01

Lose fidelity of the model in many cases

Polynomial functions
Pruned Model (II): Near-Diagonality Reduction

\[ |i_m - i_n| \leq l \]

*keep some off-diagonals which are close to main diagonal.*

- Simplify model structure
- Keep high fidelity
- Trade-off possible

\[ x(n) x(n-1) x(n)x(n-2) x(n)x(n-3) \]
\[ x^n(n-1)x(n-2) x(n+1)x(n-3) \]
\[ x^2(n-2) x^2(n-3) \]
\[ x^2(n-1) \]

[Zhu, MWCL’04]
### Comparison of the pruned models

<table>
<thead>
<tr>
<th>$l$</th>
<th>Number of coefficients</th>
<th>NMSE (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>244</td>
<td>-36.1</td>
</tr>
<tr>
<td>2</td>
<td>133</td>
<td>-35.9</td>
</tr>
<tr>
<td>1</td>
<td>54</td>
<td>-35.5</td>
</tr>
<tr>
<td>0</td>
<td>12</td>
<td>-32.0</td>
</tr>
</tbody>
</table>

- **“Pruned model”**
- **“Full model”**
- **“Memory polynomial model”**
**Pruned Model (III): Deviation Reduction**

-- *Model Low-order Dynamics*

Set

\[ e(t, \tau) = x(t - \tau) - x(t) \]

The **Classic Volterra series** can be written as

\[ y(t) = y_S(t) + y_D(t) \]

where

\[ y_S(t) = \sum_{r=1}^{+\infty} a_r x^r(t) \]

\[ y_D(t) = \int \cdots \int g_r[x(t), \tau_1, \ldots, \tau_r] \cdot \left[ \prod_{i=1}^r e(t, \tau_i) \right] d\tau_i \]

Normally truncate \( y_D(t) \) to first order if \( e(t, \tau) \) small enough during \([-T_A, T_B] \)

\[ y_D(t) = \int_{-T_A}^{T_B} g_1[x(t), \tau] e(t, \tau) d\tau \]

**Extract separately**

**Complicated procedures**

[Filicori, EL'91]
[Mirri, TransCS’02]
[Ngoya, IMS’00]
In the finite discrete time domain

\[ y(n) = \sum_{p=1}^{P} a_p(n)x^p(n) + \sum_{p=1}^{P} x^{p-1}(n) \sum_{i=0}^{N-1} g_{p_i}(i)e(n,i) \]

Re-substitute \( e(n,i) = x(n-i) - x(n) \)

\[ y(n) = \sum_{p=1}^{P} h_p(n)x^p(n) + \sum_{p=1}^{P} x^{p-1}(n) \sum_{i=0}^{N-1} h_{p_i}(i)x(n-i) \]

In the same way, we can write the classic Volterra series as

\[ y(n) = \sum_{p=1}^{P} h_p(n)x^p(n) + \sum_{p=1}^{P} \left\{ \sum_{r=1}^{p} [x^{p-r}(n) \sum_{i_1=0}^{N-1} \cdots \sum_{i_r=i_{r-1}}^{N-1} h_{p_r}(i_1,\ldots,i_r) \prod_{j=1}^{r} x(n-i_j) ] \right\} \]

Conclusion: **Dynamic Volterra model is a truncated classic Volterra model, i.e., \( r=1 \).**
In many cases, the first-order truncation is not enough to capture the
dynamics of the system. More terms have to be added in to improve the
accuracy of the model.

\[ y(n) = \sum_{p=1}^{P} h_p(n)x_p(n) + \sum_{p=1}^{P} \left\{ \sum_{r=1}^{p} \left[ x^{p-r}(n) \sum_{i_1=0}^{N-1} \cdots \sum_{i_{r-1}=i_{r-1}}^{N-1} h_{p_{r}}(i_1, \ldots i_r) \prod_{j=1}^{r} x(n-i_j) \right] \right\} \]

**New Pruning Algorithm:** \( 1 \leq r \leq L \) “Deviation Reduction”

\[ r = 1 \quad x^{p-1}(n)x(n-i) \]

\[ r = 2 \quad x^{p-2}(n)x(n-i)x(n-j) \]

**Advantage**

- This model can be directly extracted from discrete time-domain
  measured input and output data by using least-squares estimation.
- More flexible: enable trade-off between accuracy and complexity.
Example:

<table>
<thead>
<tr>
<th>Index Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>h(0), h(0,0,0), h(0,0,0,0)</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>1 0 0 0 1 0 0 0 0 1 0 1</td>
</tr>
<tr>
<td>2 0 0 2 0 0 0 0 2 0 2 2</td>
</tr>
<tr>
<td>0 1 1 1 0 0 1 1 1 0 1 1</td>
</tr>
<tr>
<td>0 1 2 0 0 0 1 2 0 1 2 2</td>
</tr>
<tr>
<td>0 2 2 0 0 0 2 2 0 2 2 2</td>
</tr>
<tr>
<td>1 1 1 0 0 1 1 1 0 1 1 1</td>
</tr>
<tr>
<td>1 1 2 0 0 1 1 2 0 1 2 2</td>
</tr>
<tr>
<td>1 2 2 0 0 1 2 2 0 2 2 2</td>
</tr>
<tr>
<td>2 2 2 0 0 2 2 2 0 2 2 2</td>
</tr>
<tr>
<td>x(n) x^3(n) x^5(n) ...</td>
</tr>
<tr>
<td>r=0: Static part</td>
</tr>
<tr>
<td>x(n-i) x^2(n) x(n-i) x^4(n) x(n-i) ...</td>
</tr>
<tr>
<td>r=1: 1st-order dynamics</td>
</tr>
<tr>
<td>x(n) x(n-i_1) x(n-i_2) x^3(n) x(n-i_1) x(n-i_2) ...</td>
</tr>
<tr>
<td>r=2: 2nd-order dynamics</td>
</tr>
<tr>
<td>x(n-i_1) x(n-i_2) x(n-i_3), x^2(n) x(n-i_1) x(n-i_2) x(n-i_3), ...</td>
</tr>
<tr>
<td>r=3: 3rd-order dynamics</td>
</tr>
<tr>
<td>x(n) x(n-i_1) x(n-i_2) x(n-i_3) x(n-i_4), ...</td>
</tr>
<tr>
<td>r=4: 4th-order dynamics</td>
</tr>
<tr>
<td>x(n-i_1) x(n-i_2) x(n-i_3) x(n-i_4) x(n-i_5), ...</td>
</tr>
<tr>
<td>r=5: 5th-order dynamics</td>
</tr>
</tbody>
</table>

IMS-2006, June 11, 2006

University College Dublin
The experimental results show that increasing the order of the deviation does not improve much in accuracy but the number of coefficients increases dramatically, which means that the low-order dynamics dominate the nonlinear distortions generated by the PA. Therefore, it is reasonable to set $r$ to a small number, i.e., only keep low-order dynamics, so that we can reduce the model complexity but keep the model fidelity.
Outline

- Introduction
  - What is Volterra Series
  - Why Volterra Series

- Volterra Model Extraction
  - Discrete time domain

- Simplified Model Structures
  - Special Cases
  - Direct Pruning
  - Reformat the Expansion

- Conclusions
Reformat the Expansion: Laguerre-Volterra Model

-- Model Long-term Memory Effects

Classic Volterra Expansion

-- Based on Finite Impulse Response;
-- Number of delays directly relies on the actual memory length;
-- Not efficient to model “slow” memory.

Reformat the Expansion

-- Use more efficient Orthonormal Basis Functions, e.g., Laguerre functions, to replace the Dirac Impulse Responses;
-- Avoiding “ill-condition” since it is orthogonal;
-- But “a priori” knowledge required, e.g., find the pole.
Memory Dependent
-- Number of delays directly relies on the actual memory length.

Memory Independent
-- with feedback but fixed pole, so keep stable.

more efficient to model “long term” memory
\[ \tilde{y}(n) = \sum_{i_1=0}^{M-1} \hat{h}_1(i_1) \times \tilde{x}(n-i_1) + \sum_{i_1=0}^{M-1} \sum_{i_2=0}^{M-1} \sum_{i_3=0}^{M-1} \hat{h}_3(i_1, i_2, i_3) \times \tilde{x}(n-i_1) \tilde{x}(n-i_2) \tilde{x}^*(n-i_3) + \cdots \]

**Classic Volterra Model**

\[ l_k(n) = \sum_{m=0}^{+\infty} \varphi_k(m) \tilde{x}(n-m) = L_k(z, \lambda) \tilde{x}(n) \]

*Laguerre functions*

\[ \tilde{y}(n) = \sum_{k=0}^{L-1} c_1(k) l_k(n) + \sum_{k_1=0}^{L-1} \sum_{k_2=0}^{L-1} \sum_{k_3=0}^{L-1} c_3(k_1, k_2, k_3) l_{k_1}(n) l_{k_2}(n) l_{k_3}^*(n) + \cdots \]

**Laguerre-Volterra Model**

\[ L << M \quad \text{number of coefficients reduced} \]

[Zhu, IMS’05]
To model this power amplifier:

- HBT Class A, 2.14 GHz
- WCDMA signal source
- 12,000 sampling points

Nonlinearities: up to 5th Order

- Number of Laguerre Filters: L=3
- Total Coefficients: 81
- Pole: $\lambda = 0.2$
Measured Trajectory Diagram

Modeled Trajectory Diagram

Real Part

Imaginary Part

Time (usec)

In-Phase

Quadrature

Measured

Modeled
Model Fidelity Evaluation and Comparison

\[ \text{NMSE} := 10 \log \left( \frac{\sum_{k=1}^{M} \left( (y_{1,k}^{\text{meas}} - y_{1,k}^{\text{mod}})^2 + (y_{Q,k}^{\text{meas}} - y_{Q,k}^{\text{mod}})^2 \right)}{\sum_{k=1}^{M} \left( y_{1,k}^{\text{meas}} \right)^2 + \left( y_{Q,k}^{\text{meas}} \right)^2} \right) \]

No. of samples in each record (M) = 1000

AM/AM AM/PM Model (9 parameters)

Classic Volterra Model (81 parameters)

Laguerre-Volterra Model (81 parameters)

Laguerre-Volterra Model (20 parameters)

Laguerre-Volterra Model (605 parameters)

new model requires much smaller number of parameters to reach the same accuracy.
Conclusions

- Volterra models can be used to characterize both nonlinearity and memory effects of power amplifiers.
- The general Volterra model has to be simplified in practical applications.
- No “best-model” in general: model structure and pruning algorithm selection depends on the characteristics of the system and model fidelity requirements.
- Both the complexity of model structure and the feasibility of model extraction have to be considered. Simpler model structure sometimes means more difficult model extraction.
- Future work: Identify physical sources of nonlinearities and memory effects, then directly relate to the parameters of the model. *Parametric Model* is desirable.
References

Acknowledgements

Science Foundation Ireland

The EU FP6 Network of Excellence
TARGET
The best material model of a cat is another, or preferably the same, cat.

-- Norbert Wiener