Introduction to Compact Dynamical Modeling

IV – Reduction of Non-Linear Systems

Luca Daniel
Massachusetts Institute of Technology

Course Outline

- Quick Sneak Preview
- I. Assembling Models from Physical Problems
- II. Simulating Models
- III. Model Order Reduction for Linear Systems
- IV. Model Order Reduction for Non-Linear Systems
  - Some Examples
    - Model Order “Reduction” from PDE discretizations
    - Model “Generation” from input/output signals
- V. Parameterized Model Order Reduction

NonLinear devices in Systems-On-Chip

Example of a micro-electro-mechanical (MEM) switch

\[
C(v) \frac{dv}{dt} = -G(v) + Bv_m
\]

\[
EI \frac{d^4 u}{dx^4} - S \frac{d^2 u}{dx^2} = F_{elec} + \int_0^y (p - p_a) dy - \rho \frac{d^2 u}{dt^2}
\]

\[
\nabla \cdot ((1+6K)u^3 \nabla p) = 12 \mu \frac{\partial (pu)}{\partial t}
\]

\[
F_{elec} = -\varepsilon_0 MV_0^2 \frac{d}{2u^2}
\]

\[
\frac{dx}{dt} = f(x) + Bv
\]

\[
y = C^T x
\]
Example of model of a vein (with valves)

\[ C(v) \frac{dv}{dt} = -G(v) + Bi \]
\[ y = Hv \]

Projection Approach for Nonlinear Systems

**Large system of nonlinear ODEs**
\[ \frac{dx}{dt} = F(x) + bu \]
\[ \frac{d(U\hat{x})}{dt} = F(U\hat{x}) + bu \]

**System of q nonlinear ODEs**
\[ (V^T U) \frac{d\hat{x}}{dt} = V^T F(U\hat{x}) + (V^T b) \]

Is this a “reduced model”?
How fast can I simulate it?

Complexity of a Model
What is the cost of simulation?

**System of q linear ODEs**
\[ \frac{d\hat{x}}{dt} = (V^T A U) \hat{x} + (V^T b) \]

**System of q nonlinear ODEs**
\[ \frac{d\hat{x}}{dt} = V^T F(U\hat{x}) + (V^T b) \]

- **Explicit integration (FE)**
  - matrix-vector products on reduced system
- **Implicit integration (BE, Trap)**
  - linear system solve on reduced system
- **Simulation speed depends**
  - only on ORDER of REDUCED system
- **ODR Solver**
  (Time Integration or Periodic Steady-State)

What is the cost of this step?
- **Explicit integration (FE)**
  - Function evaluations on original system
- **Implicit integration (BE, Trap)**
  - Newton (need function evaluations and derivatives from original system for Jacobian)
- **Simulation speed depends on order AND NONLINEARITY of the LARGE ORIGINAL system**
Nonlinear Problem is MUCH Harder

1. In what basis should we project?
   - No simple frequency domain insight
   - No eigenmodes
   - No Krylov subspace
   - No Gramians

2. How do you represent the vector field?
   - “New” issue for nonlinear systems

---

MOR for nonlinear dynamic systems

- **Polynomial approximation (or Volterra Series) of nonlinear vector field (e.g. quadratic):**
  \[
  \frac{dx}{dt} = F(x_0) + J(x_0)(x-x_0) + W(x_0)((x-x_0) \otimes (x-x_0)) + Bu
  \]

- **Projection framework:**
  \[
  \frac{dx}{dt} = V^T F(x_0) + V^T J(x_0)(x-x_0) + V^T W(x_0)((x-x_0) \otimes (x-x_0)) + V^T Bu
  \]
  \[
  \frac{d\hat{x}}{dt} = V^T JU(\hat{x} - \hat{x}_0) + V^T WU \otimes U(\hat{x} - \hat{x}_0) \otimes (\hat{x} - \hat{x}_0) + \\
  + V^T F(x_0) + V^T Bu
  \]

 Computational Considerations

- **Size of Krylov spaces grows quickly with functional series order**
  - potentially large models

- **Practical for weak nonlinearities needing only few terms in functional series**
  - example: RF datapath, (low noise amplifiers)
RF mixer example

Model Order Reduction for NonLinear Systems

\[ \frac{dx}{dt} = F(x(t)) + bu(t) \]

- Representation of \( F(x) \) using a polynomial (e.g. Taylor's expansions, Volterra Series) [Phillips00]

\[ F(x) \approx F(x_0) + J(x-x_0) + W((x-x_0) \otimes (x-x_0)) + \ldots \]

- Representation of \( F(x) \) using several linearizations (Trajectory Piece-Wise Linear TPWL) [Rewienski01]

\[ F(x) \approx w_i(x)[F_i(x_i) + J_i(x-x_i)] + w_2(x)[F_2(x_2) + J_2(x-x_2)] + \ldots \]

- Representation of \( F(x) \) with several polynomials (PWP PieceWise Polynomial) [Dong03]

Piecewise-linear representation (TPWL) [M. Rewienski, J. White ICCAD01]

- Linearizations around \( x_i, i=0, \ldots, m-1 \)

\[ F(x) = F(x_i) + J_i(x-x_i) \]

- Weighted combination of the models:

\[ \frac{dx}{dt} = \sum_{i=0}^{m-1} w_i(x)F(x_i) + \sum_{i=0}^{m-1} w_i(x)J_i(x-x_i) + Bu \]

\[ \forall x \sum_{i=0}^{m-1} w_i(x) = 1 \]

- Project linearized models:

\[ \frac{dx}{dt} = \sum_{i=0}^{m-1} w_i(U\hat{x})V^T F(U\hat{x}) + \sum_{i=0}^{m-1} w_i(U\hat{x})V^T J_i(U(\hat{x} - \hat{x}_i)) + V^T Bu \]
TPWL: Picking Linearization Points

Use training trajectories to pick linearization points

\[ \epsilon = \text{Error > } \epsilon \]

Time Domain Simulation

Linearization at current state \( x_i \)

State Space

Background – TPWL

Reduction of the Linearized Systems

\[
\frac{dx}{dt} = w_1 [A_1 x(t) + K_1] + w_2 [A_2 x(t) + K_2] + \ldots + w_k [A_k x(t) + K_k] + bu(t)
\]

Model from linearization

Model from linearization 2

Model from linearization k

\[
U^\top \begin{bmatrix} A_1 \mid 0 \mid \ldots \mid 0 \mid \ldots \mid 0 \mid 0 \mid U \end{bmatrix} = 0
\]

Use moments from EACH linear model to construct \( U \)

\[ x \in \text{span} \{ b, A_1 b, A_1^2 b, \ldots, A_1^n b, A_2 b, \ldots, A_i b \} \]

\[ U = \begin{bmatrix} b & A_1 b & \ldots & A_1^n b & \ldots & A_i b \end{bmatrix} \]

TPWL: Weighting / Simulation

[Riewinski01, Tiwary05, Dong05]

Use weighting functions to combine linear models during simulation

\[
\frac{dx}{dt} = \sum_{i=1}^{k} w_i[A_i x(t) + K_i] + bu(t)
\]

TPWL - Constructing the projection matrix \( V \)

Linearization 1

Linearization 2

Linearization 3

C – poorly approximated

Well approximated

also well approximated

Current state
**Simulation with the piecewise-linear model**

\[
\frac{d\mathbf{x}}{dt} = f(\mathbf{x}) + Bv^2
\]

**How to choose the reduced basis subspace \( U \)?**

**Fast approximate simulation**

- Generate the reduced linear model
- Use the reduced linear system for simulation

**Example of a MEM micro-switch**

![MEM micro-switch diagram](image)

\[
EI \frac{\partial^4 w}{\partial x^4} - S \frac{\partial^2 w}{\partial t^2} = F_{\text{ext}} + \int_0^t (p - p_s) dy - \rho \frac{\partial^2 w}{\partial t^2}
\]

\[
\nabla \cdot ((1 + 6K)u^3 p \nabla p) = 12\mu \frac{\partial (pu)}{\partial t}
\]

\[
F_{\text{ext}} = -\frac{\varepsilon_0 wy^2}{2m^2}
\]
Approaches for picking $U$

- Use Eigenvectors of the linearized system matrices (modal analysis) [Ma88]
- Use Frequency Domain Data of the linearized systems
  - Compute $x(s_1), x(s_2), \ldots, x(s_k)$
  - Use the SVD to pick $q < k$ important vectors
- Use Time Series Data [Sirovich87, Wilcox Peraire91]
  - Compute $x(t_1), x(t_2), \ldots, x(t_k)$
  - Use the SVD to pick $q < k$ important vectors

II.2.b SVD Singular Value Decomposition
other names: POD Principal Orthogonal Decomposition
or KLD Karhunen-Lo`eve Decomposition
or PCA Principal Component Analysis
or Poor Man TBR

Approaches for picking $U$

- Use Eigenvectors of the linearized system matrices (modal analysis) [Ma88]
- Use Frequency Domain Data of the linearized systems
  - Compute $x(s_1), x(s_2), \ldots, x(s_k)$
  - Use the SVD to pick $q < k$ important vectors
- Use Time Series Data [Sirovich87, Wilcox Peraire91]
  - Compute $x(t_1), x(t_2), \ldots, x(t_k)$
  - Use the SVD to pick $q < k$ important vectors
- Use Krylov Subspace Vectors (Moment Matching) [Gunupudi99, Riewinski01, Tiwary05, Dong05]
- Use Singular Vectors of System Gramians Product (Truncated Balance Realizations) [Scherpen93, Vasilyev03],

Computational results
MEM micro-switch example
Computational results
MEM micro-switch example

Example of model of a vein (with valves)

Computational results

Input:
- Training input
- Testing input

Harmonics | Full nonlinear model | Reduced order TPWL | Error [%]
--- | --- | --- | ---
$dc (c_0)$ | 9.3736 | 9.4061 | 0.4
$1^{st} (c_1)$ | 3.9684-0.2625i | 3.9630-0.2641i | 0.2
$2^{nd} (c_2)$ | -0.2803+0.0617i | -0.3100+0.0578i | 10.5
$3^{rd} (c_3)$ | 0.0223-0.0106i | 0.0233-0.0138i | 13.5
### Computational results
#### model order reduction

<table>
<thead>
<tr>
<th>MOR Method</th>
<th>Model generation time* [s]</th>
<th>Simulation time* [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear MOR</td>
<td>44.8</td>
<td>1.2</td>
</tr>
<tr>
<td>Quadratic MOR</td>
<td>2756.5</td>
<td>31.5</td>
</tr>
<tr>
<td>Piecewise-linear MOR</td>
<td>80.7</td>
<td>8.0</td>
</tr>
</tbody>
</table>

*Matlab implementation

### Computational results
#### OpAmp example

**OpAmp:**
- 70 MOSFETs
- 13 resistors
- 9 capacitors
- 51 nodes

**Harmonics**

<table>
<thead>
<tr>
<th>Harmonics</th>
<th>Full nonlinear model</th>
<th>Reduced order TPWL model</th>
<th>Error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dc</td>
<td>-0.4967</td>
<td>-0.4921</td>
<td>-1.0</td>
</tr>
<tr>
<td>f2-f1</td>
<td>0.0072-0.0223i</td>
<td>0.0081-0.0226i</td>
<td>2.4</td>
</tr>
<tr>
<td>2f1-f2</td>
<td>0.0046-0.0037i</td>
<td>0.0056-0.0042i</td>
<td>18.8</td>
</tr>
<tr>
<td>f1</td>
<td>-0.0667+0.1116i</td>
<td>-0.0667+0.1108i</td>
<td>-0.6</td>
</tr>
<tr>
<td>f2</td>
<td>-0.0638-0.1003i</td>
<td>-0.0634-0.1005i</td>
<td>-0.1</td>
</tr>
<tr>
<td>2f2-f1</td>
<td>0.0013-0.0050i</td>
<td>0.0019-0.0048i</td>
<td>-0.3</td>
</tr>
<tr>
<td>2f1</td>
<td>0.0104+0.0008i</td>
<td>0.0108+0.0004i</td>
<td>3.7</td>
</tr>
</tbody>
</table>
Approaches for picking \( U \)

- Use Eigenvectors of the system matrix (modal analysis) [Ma88]
- Use Frequency Domain Data
  - Compute \( x(s_1), x(s_2), \ldots, x(s_k) \)
  - Use the SVD to pick \( q < k \) important vectors
- Use Time Series Data [Sirovich87, Wilcox Peraire91]
  - Compute \( x(t_1), x(t_2), \ldots, x(t_k) \)
  - Use the SVD to pick \( q < k \) important vectors
- Use Krylov Subspace Vectors (Moment Matching) [Gunupudi99, Riewinski01, Tiwary05, Dong05]
- Use Singular Vectors of System Gramians Product (Truncated Balance Realizations) [Scherpen93, Vasilyev03]

The key points of the algorithm:

- Use TPWL to handle nonlinearity

\[
f_{TPWL}(x^r) = \sum_{i=0}^{n} w_i(x^r)(V^T f(x_i) + V^T A U(x^r - W^T x_i))
\]

- Here we use TBR for projection matrices \( V \) and \( U \)

Reducing cost of TBR reduction - Combined Krylov-TBR algorithm

Initial Model: \((A B C), n\)

Intermediate Model: \((A_i B_i C_i), n_i\)

Reduced Model: \((A_{i,r} B_{i,r} C_{i,r}), q\)

Krylov reduction \((V_i, U_i)\):
\[
A_i = V_i^T A U_i, \quad B_i = V_i^T B, \quad C_i = C_i U_i
\]

TBR reduction \((V_{i,r}, U_{i,r})\):
\[
A_{i,r} = V_{i,r}^T A U_i, \quad B_{i,r} = V_{i,r}^T B_i, \quad C_{i,r} = C_i U_i
\]

Numerical results – RLC transmission line

Error in transient

TBR-based TPWL vs. Krylov-based TPWL

4-th order TBR TPWL reaches the limit of TPWL representation

Order of the reduced model
Micromachined device example

2 um of poly Si
0.5 um of poly Si deflection
y(t) = center point deflection
0.5 um SiN
2.3 um gap filled with air

$$\frac{\partial^4 u}{\partial x^4} - S \frac{\partial^2 u}{\partial x^2} = F_{\text{elec}} + \int_0^y (p - p_0) dy - \rho \frac{\partial^2 u}{\partial t^2}$$
$$\nabla((1 + 6K)u^3 \rho V p) = 12\mu \frac{d(pu)}{dt}$$

FD model

FD model

non-symmetric indefinite Jacobian

Comparing accuracies of Krylov TPWL method and TBR-based TPWL algorithm

<table>
<thead>
<tr>
<th>Accuracy in transient *</th>
<th>Order of the reduced model needed to achieve given accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Krylov-based TPWL</td>
</tr>
<tr>
<td>5%</td>
<td>12</td>
</tr>
<tr>
<td>1%</td>
<td>20</td>
</tr>
<tr>
<td>0.5%</td>
<td>&gt;30</td>
</tr>
</tbody>
</table>

5x reduction in order – 125x improvement in efficiency

*Testing input equal to training input

Questions yet to be answered:

- How should we pick training inputs?
- How “often” should we compute linearized models along the training trajectory if not checking the original system?
- What are the errors of representing a nonlinear systems as a trajectory piecewise linear system?
- What are “the best” reduced order bases for the trajectory PWL model?

Performance of Krylov- TBR TPWL MOR extraction procedures*

Model construction cost for 0.5% target accuracy

<table>
<thead>
<tr>
<th>Initial model size N</th>
<th>TBR TPWL, q=6</th>
<th>Krylov TPWL, q = 30</th>
<th>Krylov-TBR TPWL, q=6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1500</td>
<td>1268 s</td>
<td>26.34 s</td>
<td>30.57 s</td>
</tr>
<tr>
<td>800</td>
<td>181.8 s</td>
<td>7.75 s</td>
<td>8.57 s</td>
</tr>
<tr>
<td>400</td>
<td>23.75 s</td>
<td>3.03 s</td>
<td>2.73 s</td>
</tr>
</tbody>
</table>

Model Generation Cost of Krylov-TBR almost equals Krylov

*Matlab implementation
Outline

- Part 1: Model Order Reduction of Nonlinear systems
  - Introduction
  - Model Order “Reduction” from PDE discretizations
    - Weakly nonlinear systems
      - Volterra Series
    - Strongly nonlinear systems:
      - Trajectory PieceWise Linear (TPWL) and Polynomial (PWP)
      - with moment matching
      - with Truncated Balance Realization
  - Model “Generation” from input/output signals

- Part 2: Model Order Reduction of Parameterized Systems

- Part 3: Applications

System Identification: Existing Methods

- **Wiener-Hammerstein**
  - Linear Time Invariant (LTI) block with memory, followed by memory-less nonlinear block
  - restrictive structure

- **Volterra series**
  - fit polynomial
  - restrictive structure

- **Least squares fitting**
  - Identify coef. of basis functions by minimizing $F(y,u)$
  - Note:
    - $F(y,u) = \sum_{j} \phi_j(y_0, \ldots, y_m, u_0, \ldots, u_m)$
    - $\min_{\alpha} \|F(y,u)\|
    - small $\|F(\hat{y}, \hat{u})\|$, small $\|y(\hat{u}) - \hat{y}(\hat{u})\|$
    - need strong stability constraint
    - that is computationally easy to impose

Problem Setup

Given “training data” $\tilde{u}, \tilde{y}$
Identify a simple dynamical relation $F(y,u) = 0$
which minimizes the “output error” $\|e\| = \|y(u) - \tilde{y}(\tilde{u})\|
over all available training data

Incremental Stability

[Megretski, Bond Daniel CDC08, TCAD09]

1. Incremental stability implies standard internal stability
2. small $\|F(\tilde{y}, \tilde{u})\|$ = small $\|y(\tilde{u}) - \tilde{y}(\tilde{u})\|
3. Incremental stability can be re-written as quadratic constraint which can be easily imposed as sum-of-squares relaxation

Why do we like this kind of stability?

- Reference: A. Megretski, "Convex optimization in robust identification of nonlinear feedback ."
- In IEEE Conference on Decision and Control, December 2008.
Compact Dynamical Modeling as a Convex Optimization Program [Bond Daniel TCAD09]

\[ F(y, u) = 0 \]

\[ \begin{align*}
\min_{y, u} & \| F(y, u) \|_2 \\
\text{s.t.} & \quad \text{incremental stability}
\end{align*} \]

Can be cast as a convex program:

- unique solution
- efficient
- solvers available online


SYS-ID: Micro-Electro-Mechanical example using also projection of internal states

4th order CT model
7th degree polynomial
52 coefficients

\[ \dot{x} = f(x, u) \]
\[ \begin{align*}
\{x, u\} \\
\{\hat{x}, u\} \\
\dot{\hat{x}} = \hat{f}(\hat{x}, u)
\end{align*} \]

Low Noise Amplifier Example [Bond Daniel TCAD10]

Example: Discrete Time Compact Dynamical Model for system simulation

3rd order polynomial

\[ y_i = p_3(y_{i-1}, u_{i-1}, u_{i-2}) \]

2nd order polynomial

\[ y_i = q_2(y_{i-1}, u_{i-1}) \]

Example of analysis of Electronic Complex System with PMOR:

- e.g. RF or mm-wave distributed amplifier [Bond, Mahmood, Daniel 10]