PNLSS 1.0
A polynomial nonlinear state-space Matlab toolbox

Koen Tiels
Vrije Universiteit Brussel
Applications

\[ f(y) \]

loading

unloading

\[ y \]
Wiener-Hammerstein benchmark
Goal: Capture nonlinear dynamic system behavior
Goal: Capture nonlinear dynamic system behavior
Goal: Capture nonlinear dynamic system behavior
Goal: Capture nonlinear dynamic system behavior

- Input: $u(t)$
- System
- Output: $y_{meas}(t)$
- Model
- Output: $y(t)$
- Error signal: $\epsilon(t) = y_{meas}(t) - y(t)$
Goal: Capture nonlinear dynamic system behavior
Polynomial nonlinear state-space model (PNLSS)

\[ x(t + 1) = A \, x(t) + B \, u(t) \]

\[ y(t) = C \, x(t) + D \, u(t) \]

linear state-space model
Polynomial nonlinear state-space model (PNLSS)

\[
x(t + 1) = A x(t) + B u(t) + E \zeta(x(t), u(t))
\]

\[
y(t) = C x(t) + D u(t) + F \eta(x(t), u(t))
\]

linear state-space model

polynomials in \(x\) and \(u\)

with e.g. \(\zeta(x, u) = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 u \\ \vdots \\ x_2^2 u \\ u^3 \\ \vdots \end{bmatrix}\)
Identification of a PNLSS model

\[
x(t + 1) = Ax(t) + Bu(t) + E \zeta(x(t), u(t))
\]
\[
y(t) = Cx(t) + Du(t) + F\eta(x(t), u(t))
\]

linear state-space model

polynomials in \(x\) and \(u\)

\[
\theta = \begin{bmatrix}
\text{vec}(A) \\
\text{vec}(B) \\
\text{vec}(C) \\
\text{vec}(D) \\
\text{vec}(E) \\
\text{vec}(F)
\end{bmatrix}
\]

\[
\epsilon(k, \theta) = Y(k, \theta) - Y_{\text{meas}}(k)
\]

\[
V_{\text{WLS}}(\theta) = \sum_{k=1}^{N_F} \epsilon^H(k, \theta)W(k)\epsilon(k, \theta)
\]

\[
\hat{\theta} = \arg\min_{\theta} V_{\text{WLS}}
\]
Problem is nonlinear in the parameters

Nonlinear optimization
Starting values?

$V_{WLS}(\theta)$

$\theta$
Starting value: best linear approximation (BLA)

\[ x(t + 1) = A_{BLA} x(t) + B_{BLA} u(t) + E = 0 \]
\[ \zeta(x(t), u(t)) \]
\[ y(t) = C_{BLA} x(t) + D_{BLA} u(t) + F = 0 \]
\[ \eta(x(t), u(t)) \]

Nonlinear optimization

\[ x(t + 1) = A x(t) + B u(t) + E \]
\[ \zeta(x(t), u(t)) \]
\[ y(t) = C x(t) + D u(t) + F \]
\[ \eta(x(t), u(t)) \]
Collect data

\[
\text{load('WH_EstimationExample.mat')} \quad \% \text{ Multisine data}
\]
Best linear approximation and distortion levels

\[ N = 4096; \quad \% \text{ Number of samples} \]
\[ P = 2; \quad \% \text{ Number of periods} \]
\[ R = 9; \quad \% \text{ Number of realizations} \]
\[ m = 1; \quad \% \text{ Number of inputs} \]
\[ p = 1; \quad \% \text{ Number of outputs} \]
\[ \text{lines} = 2:787; \quad \% \text{ Excited frequency lines} \]
\[ \therefore \quad \% \text{ Reshape data} \]
\[ U = \text{fft}(u); \quad \% N \times m \times R \times P \]
\[ Y = \text{fft}(y); \quad \% N \times p \times R \times P \]
\[ U = U(\text{lines},:,:,:); \quad \% \text{ Input spectrum at excited frequencies} \]
\[ Y = Y(\text{lines},:,:,:); \quad \% \text{ Output spectrum at excited frequencies} \]

\% BLA, total distortion, and noise distortion
\[ [G, \text{covGtotal}, \text{covGnoise}] = \text{fCovarFrfs}(U,Y); \]
Best linear approximation and distortion levels
Linear state-space model

nAll = 2:8; % Model orders
maxr = 10; % Subspace dimensioning parameter
maxIter = 100; % Max. nbr. of Levenberg-Marquardt iterations
fs = 78125; % Sampling frequency
freq = (lines-1)*fs/N; % Frequency vector
forcestability = true; % Force stable model
showfigs = true; % Show overview and best model for each order

% Frequency domain subspace + nonlinear optimization
% McKelvey et al., 1996; Pintelon, 2002
models = fLoopSubSpace(freq,G,covGtotal,nAll,maxr,...
    maxIter,forcestability,showfigs,fs);

% Select best model on last period of validation data
[A,B,C,D] = models{6}{:}; % Select 6th-order model
[A,B,C] = dbalreal(A,B,C); % Balanced realization
6th-order linear state-space model

![Graph showing frequency vs. amplitude for different models and residuals.](image)
Starting value: best linear approximation (BLA)

\[ x(t + 1) = A_{BLA} x(t) + B_{BLA} u(t) + E = 0 \]
\[ y(t) = C_{BLA} x(t) + D_{BLA} u(t) + F = 0 \]

Nonlinear optimization

\[ x(t + 1) = A x(t) + B u(t) + E \]
\[ y(t) = C x(t) + D u(t) + F \]
Transient settings

% Concatenate all data
u = u(:); % N*P*R x m
y = y(:); % N*P*R x p

% Transient settings
NTrans = N; % Add a period before each realization
startReal = [1 4097 ... 32769]; % Starting indices realizations
T1 = [NTrans startReal]; % Transient setting periodic data
T2 = 0; % No nonperiododic transient handling
Linear model in PNLSS form

\[ x(t+1) = A_{BLA} x(t) + B_{BLA} u(t) + E = 0 \]
\[ y(t) = C_{BLA} x(t) + D_{BLA} u(t) + F = 0 \]
\[ \zeta(x(t),u(t)) \]
\[ \eta(x(t),u(t)) \]

% Settings nonlinear terms
nx = [2 3];  % Nonlinear degrees in \( \zeta \)
yx = [2 3];  % Nonlinear degrees in \( \eta \)

% Linear model in PNLSS form
model_init = fCreateNLSSmodel(A,B,C,D,nx,ny,T1,T2,0);

% Set all monomials free for optimization
model_init.xactive = fSelectActive('full',n,m,n,nx);
model_init.yactive = fSelectActive('full',n,m,p,ny);
Poor man’s stabilization method

% Simulate validation data during estimation
model_init.u_val = uval(:,);

% Set bound on modeled validation output
model_init.max_out = 1000*max(abs(yval(:)));
Nonlinear optimization

% Settings Levenberg-Marquardt (LM) optimization
nIter = 50; % Number of iterations
W = []; % No (frequency) weighting
lambda = 100; % Starting value LM damping factor

% Nonlinear optimization
[model, y_mod, models] = fLMnLssWeighted(u, y, model_init, ... nIter, W, lambda);

% Select the best model on the validation data
model = models(min_i);
Results on test data

% Change transient setting
model_pnlss_test = model;
model_pnlss_test.T1 = [16384 1];

% Compute rms error on multisine test data
e = rms(ytest - fFilterNLSS(model_pnlss_test,utest));

<table>
<thead>
<tr>
<th></th>
<th>multisine</th>
<th>swept sine</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear</td>
<td>0.03817</td>
<td>0.02319</td>
</tr>
<tr>
<td>PNLSS</td>
<td>0.03817</td>
<td>0.02258</td>
</tr>
</tbody>
</table>
Pros and cons of a PNLSS model

\[ x(t + 1) = Ax(t) + Bu(t) + E\zeta(x(t), u(t)) \]
\[ y(t) = Cx(t) + Du(t) + F\eta(x(t), u(t)) \]

- **Flexibility** ✓
- **Multiple input** ✓
- **Multiple output** ✓
- **Initial estimates** ✓
- **Software available** ✓
- **Number of parameters** ✗
- **Stability** ✗
- **Interpretability** ✗
- **Extrapolation** ✗
- **Process noise** ✗
What to expect in PNLSS 2.0?

Object-oriented

Modular components
(initialization, cost, Jacobian, optimization)

Beyond polynomials

Structure specification/retrieval
Success story: battery cell

Lithium Ion Polymer Battery (EIG-ePLB-C020, Li(NiCoMn))
3.65 V, 20 A h, AC impedance (1 kHz < 3 mΩ)

Nonparametric distortion analysis

modified by courtesy of Rishi Relan
Success story: battery cell

Time domain

Frequency domain

4 states
up to third-degree nonlinearities

modified by courtesy of Rishi Relan