LPV model identification: overview and perspectives

Marco Lovera
Dipartimento di Elettronica e Informazione, Politecnico di Milano
Outline

- Introduction and motivation
- LPV model classes:
  - state-space form
  - input-output form
- Issues in input-output to state-space conversion
- Overview of LPV model identification
- Perspectives and conclusions
Introduction

- Modern methods for robust and gain scheduled controller design call for advanced modelling and identification techniques

- Critical issue: deriving models in which the dependence from operating point information and/or uncertain parameters is explicit

- Linear Parametrically Varying (LPV) models: a useful modelling approach to bridge the gap between identification and controller design
Linear Parameter Varying systems are a class of linear time-varying systems.

In state space form they are described as

\[
\begin{align*}
x(t + 1) &= A(\delta(t))x(t) + B(\delta(t))u(t) \\
y(t) &= C(\delta(t))x(t) + D(\delta(t))u(t)
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= A(\delta(t))x + B(\delta(t))u \\
y &= C(\delta(t))x + D(\delta(t))u
\end{align*}
\]

“Time varying systems, the dynamics of which are functions of a measurable, time varying parameter vector $\delta$.\)”
Common assumptions on parameter vector $\delta$:

- Component-wise bounded
- Component-wise rate-bounded

The equations

\[
\begin{align*}
    x(t + 1) &= A(\delta(t))x(t) + B(\delta(t))u(t) \\
    y(t) &= C(\delta(t))x(t) + D(\delta(t))u(t)
\end{align*}
\]

\[
\begin{align*}
    \dot{x} &= A(\delta(t))x + B(\delta(t))u \\
    y &= C(\delta(t))x + D(\delta(t))u
\end{align*}
\]

describe a whole family of time-varying systems.

- A specific time-varying system is defined once a realisation $\delta(t)$ is chosen.

- A given LPV system can give rise to very different behaviours!
Motivation for LPV models:

- Models for LTI systems subject to time-varying uncertainty
  - robust control problems

- Models for LTV systems or linearizations of non linear systems along the trajectory of $\delta$
  - gain scheduling control problems
We will now focus on models of the type

\[ x(t + 1) = A(\delta(t))x(t) + B(\delta(t))u(t) \]
\[ y(t) = C(\delta(t))x(t) + D(\delta(t))u(t) \]

and define various model classes depending on how \( \delta \) enters the system matrices.

Finally we will compare the various structures and try to find similarities and transformations between them.
Structure of LPV models – state-space form

- **Affine parameter dependence (LPV-A):**
  \[ A(t) = A_0 + A_1 \delta_1(t) + \ldots + A_s \delta_s(t) \]

- **Input-affine parameter dependence (LPV-IA):**
  - only B and D are function of \( \delta \)
  - A and C are constant

- **Rational parameter dependence (LPV-R):**
  \[
  A(t) = \left[ A_{n0} + A_{n1} \delta_1(t) + \ldots + A_{ns} \delta_s(t) \right] \\
  \left[ I + A_{d1} \delta_1(t) + \ldots + A_{ds} \delta_s(t) \right]^{-1}
  \]
Structure of LPV models – state-space form

- Linear fractional transformation parameter dependence (LPV-LFT):

\[
\begin{align*}
x(t + 1) &= A x(t) + B_0 w(t) + B_1 u(t) \\
z(t) &= C_0 x(t) + D_{00} w(t) + D_{01} u(t) \\
y(t) &= C_1 x(t) + D_{10} w(t) + D_{11} u(t) \\
w(t) &= \Delta z(t) \\
\Delta &= \text{diag} \left( \delta_1 I_{r_1}, \ldots, \delta_s I_{r_s} \right)
\end{align*}
\]
LPV-IA and LPV-LFT models are related to each other. The compound state space matrix $M$ can be written as:

$$M(\delta) = \begin{bmatrix} A & B(\delta) \\ C & D(\delta) \end{bmatrix} = M_0 + M_1\delta_1 + \ldots + M_s\delta_s$$

where each of the $M_t$'s has the structure

$$M_t = \begin{bmatrix} 0 & B_t \\ 0 & D_t \end{bmatrix}$$

and is therefore a low rank ($r_t$) matrix. The value of $r_t$ can be recovered exactly using an SVD

$$M_t = \begin{bmatrix} 0 & B_t \\ 0 & D_t \end{bmatrix} = U_t V_t$$
Expressing each of the $M_t$'s by means of a rank $r_t$ decomposition as $M_t = U_t V_t$ one can write $M(\delta)$ as

$$M(\delta) = M_0 + U \Delta V,$$

$$U = [U_1 \ldots U_s], \quad V = [V_1^T \ldots V_s^T]^T$$

$$\Delta = \text{diag} \left( \delta_1 I_{r_1} \ldots \delta_s I_{r_s} \right)$$

The obtained form for the system matrices coincides with the one obtained in the special case of an LFT with $D_{00} = 0$:

$$M(\delta_t) = \begin{bmatrix} A & B_1 \\ C_1 & D_{11} \end{bmatrix} + \begin{bmatrix} B_0 \\ D_{10} \end{bmatrix} \Delta \begin{bmatrix} C_0 & D_{01} \end{bmatrix}$$
This does not hold anymore in the case of an LPV-A model, for which the $M_t$'s matrices turn out to be "full":

$$M_t = \begin{bmatrix} A_t & B_t \\ C_t & D_t \end{bmatrix}$$

so the LPV-A ! LPV-LFT conversion is likely to require approximations if one aims at a small size $\Delta$.

Similarly, if the $M_t$'s have been identified, the low rank condition will almost certainly not hold.
It will be necessary to truncate the computed factorisation.
We will consider input-output models of the type

\[ y(t) = - \sum_{i=1}^{n_a} a_i(\delta(t)) y(t - i) + \sum_{j=1}^{n_b} b_i(\delta(t)) u(t - j) \]

which are parameter-dependent extensions of discrete-time LTI input-output models.

As for state-space models, the \( a_i \)'s and \( b_j \)'s can be
- Affine
- Rational
- Linear Fractional
functions of the parameter vector \( \delta \).
Issues in input-output to state-space conversion

In the LTI case it is well known that if the state space model

$$[A, B, C, D]$$

corresponds to the input-output model

$$G(z) = C(zI - A)^{-1}B + D$$

then all the state space models defined by

$$[TAT^{-1}, TB, CT^{-1}, D]$$

where $T$ is square and non singular, are equivalent to the original one, in the sense that they give rise to the same input-output behaviour.
Issues in input-output to state-space conversion

In LTI system identification this motivates a number of methods which

1. Perform an initial (possibly nonparametric) estimation of the input-output behaviour

2. Refine the initial estimate
   - either directly in state-space form
   - or in input-output form, followed by conversion to state-space form
In the LPV case, the above notion of equivalence class does not hold anymore, as LPV models are *time-varying*.

In discrete-time:

\[
\begin{align*}
x(t + 1) &= A(t)x(t) + B(t)u(t) \\
y(t) &= C(t)x(t) + D(t)u(t)
\end{align*}
\]

\[
\tilde{x}(t) = T(t)x(t), \quad x(t) = T^{-1}(t)\tilde{x}(t)
\]

\[
\begin{align*}
\tilde{x}(t + 1) &= T(t + 1)A(t)T^{-1}(t)\tilde{x}(t) + T(t)B(t)u(t) \\
y(t) &= C(t)T^{-1}(t)\tilde{x}(t) + D(t)u(t)
\end{align*}
\]
In the LPV case, the above notion of equivalence class does not hold anymore, as LPV models are *time-varying*.

In continuous-time:

\[
\begin{align*}
\dot{x}(t) &= A(t)x(t) + B(t)u(t) \\
y(t) &= C(t)x(t) + D(t)u(t)
\end{align*}
\]

\[
\begin{align*}
\tilde{x}(t) &= T(t)x(t) \\
\dot{\tilde{x}}(t) &= \dot{T}(t)x(t) + T(t)\dot{x}(t)
\end{align*}
\]

\[
\begin{align*}
\dot{x}(t) &= (T(t)A(t)T^{-1}(t) + \dot{T}(t)T^{-1}(t))\tilde{x}(t) + T(t)B(t)u(t) \\
y(t) &= C(t)T^{-1}(t)\tilde{x}(t) + D(t)u(t)
\end{align*}
\]
What happens in the LPV case?

- The state transformation matrix $T$ will be parameter dependent.

- Therefore, classical LTI input-output and state-space equivalence notions should be used very carefully!

- In particular:
  - The change of state-space basis will depend \textit{locally} on the value of the parameters and on their rate of change.
  
  - The choice of input-output vs state-space models should be based on the eventual goal of the identification exercise, as conversion is not trivial.
Consider the switching system described by

\[
M(p) = \begin{cases} 
\begin{bmatrix} \alpha_0 & \beta_0 \\ 1 & 0 \end{bmatrix} & p = 0 \\
\begin{bmatrix} \alpha_1 & \beta_1 \\ 1 & 0 \end{bmatrix} & p = 1 
\end{cases}
\]

Which can be equivalently written in LPV form as

\[
M(\delta) = \sum_{i=0}^{1} \begin{bmatrix} \delta_i(p)\alpha_i & \delta_i(p)\beta_i \\ 0.5 & 0 \end{bmatrix}, \quad \delta_0(p_t) = 1 - p_t, \quad \delta_1(p_t) = p_t
\]

We can derive input-output models for the system using both representations.
Deriving input-output models for the individual modes and interpolating we get

\[ y_t = a_1(p_t)y_{t-1} + b_1(p_t)u_{t-1} \]

\[ a_1(p_t) = \sum_{i=0}^{1} \delta_i(p_t)\alpha_i \]

\[ b_1(p_t) = \sum_{i=0}^{1} \delta_i(p_t)\beta_i \]

while from the state-space form we arrive at

\[ y_{t+1} = a_1(p_t)y_t + b_1(p_t)u_t. \]

In this case only a delay is introduced.
The issue becomes more critical for higher order systems.
Two broad classes of methods can be defined:

- **Global approaches**
  - a single experiment: the parameter is also excited
  - a parameter-dependent model is directly obtained

- **Local approaches**
  - multiple experiments: constant parameter values
  - many LTI models are obtained, which have to be interpolated
Overview of the literature: global approach

- **Input/output models**
  - (Bamieh and Giarre’, 1999 & 2002)
  - (Previdi and Lovera, 2003 & 2004)
  - (Toth et al., 2007 & 2008)

- **State space models**
  - (Nemani et al., 1995)
  - (Lee and Poolla, 1997 & 1999)
  - (Lovera et al., 1998)
  - (Sznaier and Mazzaro, 2001 & 2003)
  - (Verdult and Verhaegen, 2002)
  - (Felici et al., 2007)
  - (van Wingerden and Verhaegen, 2008)
Model class

\[ y(t) = a_1(t)y(t - 1) + \cdots + a_{n_a}(t)y(t - n_a) + \]
\[ + b_1(t)u(t - 1) + \cdots + b_{n_b}(t)u(t - n_b) + e(t) \]

\[ a_i(t) = a_i^1 f_1(\delta_t) + \cdots + a_i^N f_N(\delta_t), \quad i = 1, \ldots, N \]
\[ b_i(t) = b_i^1 f_1(\delta_t) + \cdots + b_i^N f_N(\delta_t), \quad i = 1, \ldots, N \]

Parameter estimation: linear least squares

Main contribution: characterisation of persistency of excitation conditions for input-output LPV models
The model can be written in linear regression form by defining the parameter matrix and the extended regressor

\[
\Theta = \begin{bmatrix}
a_1 & \ldots & a_1^N \\
a_2 & \ldots & a_2^N \\
\vdots & \vdots & \vdots \\
a_{na} & \ldots & a_{na}^N \\
b_0 & \ldots & b_0^N \\
\vdots & \vdots & \vdots \\
b_{nb} & \ldots & b_{nb}^N
\end{bmatrix}
, \quad \Psi_t = \phi_t \pi_t = \begin{bmatrix}
-y_{t-1} \\
-\ldots \\
-\ldots \\
y_{t-na} \\
\ldots \\
u_{t-na_b}
\end{bmatrix}
\begin{bmatrix}
f_1(\delta_t) \\
f_2(\delta_t) \\
\ldots \\
f_N(\delta_t)
\end{bmatrix}
\]

So that \( y_t = \text{trace}(\Theta^T \Psi_t) + e_t \)

Parameters can be estimated recursively using LMS or RLS.
Main result: persistence of excitation.

- Assuming that the input is sufficiently rich to insure that $\phi_t$ is PE in the above sense, what is needed is that the trajectory of $\delta_t$ ’visit’ N distinct points infinitely many times.

- The rate at which $\delta_t$ revisits each of these limit points should not slow down.

- Thus, these revisits are sufficient to ergodically extract the correlation data of $\phi_t$. 
Previdi and Lovera 2003:
- NLPV model class, analogy with local model networks.

Previdi and Lovera 2004:
- NLPV model class, separable least squares estimation algorithm.

NLPV model class: feedback is dynamic and nonlinear in $\delta$. 
We start from the SISO time-varying ARX model

\[ y(t) = a_1(t)y(t-1) + \cdots + a_{n_a}(t)y(t-n_a) + \\
+ b_1(t)u(t-1) + \cdots + b_{n_b}(t)u(t-n_b) + e(t) \]

where the coefficients are given by

\[ a_i(t) = a_i^1 + a_i^2 z(t), \quad i = 1, \ldots, n_a \]
\[ b_j(t) = b_j^1 + b_j^2 z(t), \quad j = 1, \ldots, n_b \]

Variable \( z \) plays the role of a scheduling variable:

note that \( z \) is NOT measured, but estimated form data!
The scheduling variable $z$ is defined as

$$z(t, \eta) = f(\psi(t), \eta)$$

where $\eta$ is a suitable parameter vector and $\psi$ is a regression vector given by

$$\psi(t) = [y(t - 1), \ldots, y(t - n_y), u(t - 1), \ldots, u(t - n_u),$$

$$\delta(t - 1), \ldots, \delta(t - n_\delta)]'$$

Note that $\psi$ is also a function of the input scheduling variable $\delta$ (if available).
The NLPV identification problem

The overall model can be written as

\[ y(t) = \theta' \left( \varphi(t) \right) \]

where

\[ \theta = [a_1^1, \ldots, a_{n_a}^1, b_1^1, \ldots, b_{n_b}^1, a_1^2, \ldots, a_{n_a}^2, b_1^2, \ldots, b_{n_b}^2]' \]

\[ \varphi(t) = [y(t-1), \ldots, y(t-n_a), u(t-1), \ldots, u(t-n_b)]' \]

Identification problem: find \( \theta \) and \( \eta \) minimising

\[ J(\theta, \eta) = \sum_{i=1}^{N} \left( y_i - \sum_{j=1}^{n} \theta \phi_j(t_i, \eta) \right) \]  

\( \) Separable Cost Function
Besides the above mentioned issues with input-output to state-space conversion, it is argued that:

- An LPV system can be viewed as a collection of “local” behaviours (associated with constant parameter values)
- The overall behaviour of the system is given by an interpolation of the local behaviours
- The interpolation function is in general a function of time-shifted versions of the parameters
- A method based on OBFs is proposed, in the case of static interpolating functions (Wiener-LPV).
Overview of the literature: global approach

- Input/output models
  - (Bamieh and Giarre’, 1999 & 2002)
  - (Previdi and Lovera, 2003 & 2004)
  - (Toth et al., 2007 & 2008)
- State space models
  - (Nemani et al, 1995)
  - (Lee and Poolla, 1997 & 1999)
  - (Lovera et al., 1998)
  - (Sznaier and Mazzaro, 2001 & 2003)
  - (Verdult and Verhaeghen, 2002)
  - (Felici et al., 2007)
Identification of single input LPV-LFT models with a scalar parameter

Assume the state vector of the system is available for measurement

Both cases of noise free and noisy state measurement are taken into account, together with process noise in the state equation.

The problem is solved using RLS; the use of IV-RLS is also proposed to deal with non-white noise in the state measurements.
• Identification of MIMO LPV-A models

• No restrictive assumptions on the number of parameters

• Possibly noisy state vector measurement available

\[
x_{t+1} = (A_0 + A_1 \delta_{1,k} + \ldots + A_s \delta_{s,k}) x_t + (B_0 + B_1 \delta_{1,k} + \ldots + B_s \delta_{s,k}) u_t
\]

\[
y_t = x_t - e_t
\]

• Batch solution using IV least squares

\[
y_{t+1} + e_{t+1} = A_0(y_t + e_t) + A_1 \delta_{1,k} y_t + A_1 \delta_{1,k} e_t + \ldots + A_s \delta_{s,k} y_t + A_s \delta_{s,k} e_t + (B_0 + B_1 \delta_{1,k} + \ldots + B_s \delta_{s,k}) u_t
\]
A maximum likelihood (ML) algorithm for the identification of MIMO LPV-LFT models is proposed.

The algorithm is based on PEM and is strongly related to classical methods for the ML identification of ARMA and ARMAX models.

The computation of the gradient and of the hessian is performed by means of (LPV) filtering operations.

Major issue related to this algorithm: initialisation.
Consider a model class of the form

$$\mathcal{F}_u(G_p, \gamma) = \sum_{i=1}^{N_p} p_i \mathcal{F}_u(G_i, \gamma)$$

where

- the $N_p$ $G_i(z)$ transfer functions are known,
- $G_{np}(z)$ is a stable, norm-bounded operator
- $\eta$ is a bounded measurement noise.

An approach is provided which allows to test consistency (in the form of LMIs) of the a priori modelling information with the results (measured $y$ and parameters) of experiments.
Subspace-based methods

- Extensions to LPV systems of classical subspace model identification algorithms for LTI models

- Model class

\[
x_{k+1} = \left( A_0 + \sum_{i=1}^{s} [p_k]_i A_i \right) x_k + \left( B_0 + \sum_{i=1}^{s} [p_k]_i B_i \right) u_k + w_k
\]

\[
y_k = \left( C_0 + \sum_{i=1}^{s} [p_k]_i C_i \right) x_k + \left( D_0 + \sum_{i=1}^{s} [p_k]_i D_i \right) u_k + v_k
\]
Approach: use subspace techniques to estimate the state sequence from the available I/O data.

To this purpose we have to:

- Define the data equation for the state and the output of the LPV model
- Prove that we can estimate (at least approximately) the state sequence
Implementation issues: dimensionality

- The number of rows in the data matrices grows exponentially with the order of the system;

- This would limit the applicability of these techniques to SISO systems of small order;

- How can this be circumvented?

- Solutions available in the literature:
  - use the RQ factorisation to select the dominant rows in the data matrices and discard the rest
  - use kernel methods to compress the row spaces of the data matrices.
LPV model identification: an overview

Two broad classes of methods can be defined:

- **Global approaches**
  - a single experiment (the parameter is also excited)
  - a parameter-dependent model is directly obtained

- **Local approaches**
  - multiple experiments (constant parameter values)
  - many LTI models are obtained, which have to be interpolated
Local approach

- (Steinbuch et al 2003)
- (Paymans et al 2006)
- (Toth et al 2007)
- (Lovera Mercere 2007)
Issues with local approaches

Numerical accuracy:
- Local problems are often formulated using poorly conditioned canonical forms
- This may lead to ill-conditioning in the interpolation

Consistency of the interpolation procedure:
- Care must be taken when interpolating LTI identified models
  - Input/output form
    - Interpolating transfer function coefficients
  - State space form
    - Consistency of state space basis
The method of Steinbuch et al.

The algorithm (applicable to SISO or MISO models) can be summarised as follows:

Local experiments are performed and nonparametric estimates \( \hat{G}(j\omega) \) of the local frequency response are computed.

Transfer functions

\[
G(s) = \frac{\beta_{n-1}s^{n-1} + \beta_{n-2}s^{n-2} + \cdots + \beta_1s + \beta_0}{s^n + \alpha_{n-1}s^{n-1} + \cdots + \alpha_1s + \alpha_0} + \beta_n
\]

are fitted to the local frequency responses using nonlinear LS.
Each transfer function is converted to Canonical Controllability Form:

\[
A = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
\end{bmatrix},
B = \begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
1 \\
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
\beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{n-1} \\
\end{bmatrix},
D = \begin{bmatrix}
\beta_n \\
\end{bmatrix}
\]

The free parameters of the local models are interpolated.

The model is converted to LFT form.
The method of Steinbuch et al.: discussion

- Numerical issues: the CCF is ill conditioned, so the interpolation step will be numerically very sensitive.

- Method restricted to
  - low order models
  - without sensitive poles/zeros (lightly damped complex conjugate poles)

- State space interpolation: no guarantee that all local models are in the same state-space basis.
The method of Paijmans et al

- Local models are parameterised using poles, zeros and gain

\[ G_i(s) = \gamma_i \frac{(s + z_{i,1})(s + z_{i,2})\cdots(s + z_{i,n_z})}{(s + p_{i,1})(s + p_{i,2})\cdots(s + p_{i,n_p})}, \quad i = 1, \ldots, m \]

- and factored into first and/or second order subsystems

\[ G_i(s) = \gamma_i \prod_{\tau=1}^{\tau_1} F_i^\tau(s) \prod_{\tau=1}^{\tau_1} S_i^\tau(s), \quad i = 1, \ldots, m \]
The method of Paijmans et al

- Each local model is decomposed using the following rules:
  - A second order system is created for each pair of c.c.poles
  - All pairs of c.c. zeros are added to existing second order systems
  - For each remaining real pole a first order system is created
  - The remaining real zeros are added to the first and second order subsystems

- Parameter-dependent poles and zeros loci are optimised in order to fit the pole/zero maps of the local models.
The method of Paijmans et al: discussion

- Limited to SISO systems

- Interpolation step very critical – requires manual intervention

- Constraints introduce to preserve affine parameter dependence: $B^\tau$ and $D^\tau$ matrices of local models must be constant.
Consider the MIMO linear parametrically-varying system

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

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

Assume that the results of P identification experiments are available, associated with the operation of the system near different values of the parameter vector \( p \).

The problem consists in determining a set of parameter dependent matrices

\[ [\hat{A}(p), \hat{B}(p), \hat{C}(p), \hat{D}(p)] \]

which provide a good approximation of the system over the considered range of operating points.
The proposed method addresses numerical issues, as follows.

1. Linear discrete-time state space models are estimated for each operating point, using a frequency-domain SMI algorithm, e.g., (McKelvey 1995)

2. The identified models are balanced using the numerical algorithm of (Laub et al 1987)

3. If necessary, the balanced models are converted to continuous-time using a bilinear transformation

4. The p-dependent model is obtained by interpolation of the state-space matrices of the local models, made possible by the properties of balanced realisations

5. The model can be eventually converted to LFT form.
Balanced realisations: definitions

Definition: the matrices

\[ \mathcal{W}_o = \sum_{k=0}^{\infty} (A^k)^T C^T C A^k, \quad \mathcal{W}_c = \sum_{k=0}^{\infty} A^k B B^T (A^k)^T \]

are the observability and controllability Grammians for the discrete-time system

\[
\begin{align*}
x(t + 1) &= A x(t) + B u(t) \\
y(t) &= C x(t) + D u(t)
\end{align*}
\]

The state space realization is internally balanced if

\[ \mathcal{W}_o = \mathcal{W}_c = \Sigma = \text{diag} (\sigma_1 \cdots \sigma_n), \]

where \( \{\sigma_i\}_{i=1}^{n} \) are the singular values of \( \mathcal{W}_c \mathcal{W}_o \)
Definition: a state space realisation is \( qr \) internally balanced if

\[
O_q^T O_q = C_r C_r^T = \Sigma.
\]

The considered frequency-domain identification algorithm is such that

\[
O_q^T O_q = \hat{\Sigma}_s
\]

\[
C_r C_r^T = \left( I - A^{2M} \right) \hat{\Sigma}_s \left( I - A^{2M} \right)^T
\]

So for a stable system and large \( M \) the identified model is \( qr \) internally balanced.
Balanced realisations: useful properties

Some classical results on balanced realisations and the balancing transformation $T_B$ (Moore 1981, Laub et al 1987, Kabamba 1985)

- If the eigenvalues of the product of the reachability and observability Gramians are distinct, the eigenvectors are uniquely determined up to sign, so $T_B$ is essentially unique

- If the eigenvalues are distinct, if the system matrices are smooth functions of the parameter $p$, then so is $T_B$

  - Direct interpolation of the obtained state-space matrices is possible
- The balanced form is only unique up to a sign, so manual correction of sign switching might be necessary.

- Interpolation is a trivial exercise in LS estimation.

- Conversion to LFT form can be performed as previously illustrated.

- The LFT may be further optimised in order to compensate for SVD truncation.
Consider the parameter-dependent system

\[
A = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0.01p & -(p - 0.004) & -0.39
\end{bmatrix},
\]

\[
B = \begin{bmatrix}
0 & 0 & 1
\end{bmatrix}^T,
\]

\[
C = \begin{bmatrix}
0.02 & 0 & -0.5
\end{bmatrix}
\]

where

p \in [0.1, 0.95]
Numerical example: estimated matrix elements

### Estimated Matrix Elements of A

-0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0

### Estimated Matrix Elements of B

-0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0

### Estimated Matrix Elements of C

-0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0
Numerical example: Hankel singular values

![Graph showing numerical example of Hankel singular values]
Numerical example: transfer function coefficients

![Graph showing transfer function coefficients]
A fact: LPV model identification hardly used at all in practice.

There might be many reasons for this...

- The tools are not available in public domain
- The methods require unrealistic assumptions
- The obtained models do not match the existing design methods and tools: the “square peg-round hole” problem of system identification!
The modelling process as currently viewed in the field:

A smooth flow…

...from nonlinear simulation…

...to models ready for robust analysis and synthesis
It is often suggested to use data at the simulator level: “model calibration”

- Simulink
- Dymola
- “MoCaVa” (T. Bohlin)

...is this sensible?
Basic physical knowledge should be used to build an underlying model.

Only uncertainty structure should be fixed at this stage (Modelica annotations?).
The LFT extraction should emphasize:

- Measurable time-varying parameters
- Uncertain parameters to be refined using data.
Parameter estimation should be performed at this stage:

- To simplify the optimisation process
- To come closer to an analytically tractable problem.
What’s missing to complete the picture?

- Automated LFT extraction: many people working on this…
  - Solved a long time ago for a specific application (DLR)
  - Available for explicit models in Simulink (UC Berkeley)
  - Undergoing development in Open Modelica

- Identifiability analysis for structured LPV-LFT models
  - Results available
  - Not enough for complete and automated process

- Validation process
Concluding remarks

- An overview of the last 20 years in the field of LPV model identification has been provided.

- A discussion of the pros and cons of each approach has been offered.

- Some personal views on what the future of the area should be have been given.
Consider the problem of minimising

\[ J_1(a, p) = \| y - \Phi(p)g(a) \|_2^2 \]

SLS approach (Golub and Pereyra 1973):

- Minimise \( J_1 \) with respect to \( b = g(a) \); this gives
  \[ \hat{b} = \Phi^\dagger(p)y \]

- Define the Variable Projection Functional
  \[ J_2(p) = \| y - \Phi(p)\Phi^\dagger(p)y \|_2^2 \]

- And estimate \( p \) by minimising \( J_2 \).
SLS yields the optimal solution to the regression problem, under two assumptions:

- $g(a) = b$ has at least one solution for any $b$.
- $\Phi(p)$ has constant rank in a suitable open set containing the solution.

The partial linearity of the regression is exploited.

One then needs an efficient way of computing the gradients of $J_2$ with respect to $p$. 
The NLPV-SLS identification algorithm

Compute an estimate of $\eta$ as

$$\hat{\eta} = \arg \min_{\eta} \|P_\Phi^\perp(\eta)y\|^2_2$$

where

$$P_\Phi^\perp(\eta) = I - \Phi(\eta)\Phi^\dagger(\eta)$$

Compute an estimate of $\theta$ as

$$\hat{\theta} = \Phi^\dagger(\hat{\eta})y$$

Iterative minimisation of the cost function

Gauss-Newton-Marquardt algorithm.
The regression can be reformulated as

\[ Y_{t+1,t+N} + E_{t+1,t+N} = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} Y_{k,t+N-1}^{\delta} + E_{k,t+N-1}^{\delta} \\ U_{k,t+N-1}^{\delta} \end{bmatrix} \]

where

\[ Y_{k,t+N} = \begin{bmatrix} y_t & \cdots & y_{t+N} \end{bmatrix}, \quad E_{k,t+N} = \begin{bmatrix} e_t & \cdots & e_{t+N} \end{bmatrix} \]

\[ Y_{k,t+N-1}^{\delta} = \begin{bmatrix} y_t & y_{t+1} & \cdots & y_{t+N-1} \\ y_{t+1,k} & y_{t+1}^{\delta_{1,t+1}} & \cdots & y_{t+N-1}^{\delta_{1,t+N-1}} \\ \vdots & \vdots & & \vdots \\ y_{t+s,k} & y_{t+s}^{\delta_{s,t+s}} & \cdots & y_{t+N-1}^{\delta_{s,t+N-1}} \end{bmatrix} \]

\[ E_{k,t+N-1}^{\delta} = \begin{bmatrix} e_t & e_{t+1} & \cdots & e_{t+N-1} \\ e_{t+1,k} & e_{t+1}^{\delta_{1,t+1}} & \cdots & e_{t+N-1}^{\delta_{1,t+N-1}} \\ \vdots & \vdots & & \vdots \\ e_{t+s,k} & e_{t+s}^{\delta_{s,t+s}} & \cdots & e_{t+N-1}^{\delta_{s,t+N-1}} \end{bmatrix} \]
Consistent estimates can be computed using Instrumental Variables

\[
\begin{bmatrix}
\hat{A} & \hat{B}
\end{bmatrix} = Y_{t+1,t+N} W^T \left( \begin{bmatrix}
Y_{k,t+N-1}^\delta \\
U_{k,t+N-1}^\delta
\end{bmatrix} W^T \right)^{-1}
\]

where

\[
W = \begin{bmatrix}
Y_{k-1,t+N-2}^\delta \\
U_{k,t+N-1}^\delta
\end{bmatrix}.
\]