Indirect Adaptive Predictive Control in Tackling Real World Control Issues

Mihai Huzmezan and Guy A. Dumont
Department of Electrical and Computer Engineering
Pulp and Paper Centre
University of British Columbia,Vancouver, B.C. Canada

IASTED CA'99 - Banff, Canada, 25-29 July, 1999

Outline

1. **Introduction**
2. **System Identification**
   - **The need for modelling**
   - **Model structures**
   - **The Laguerre model**
   - **Least-squares identification**
   - **Open vs. closed-loop identification**
3. **MBPC**
   - **Why MBPC?**
   - **The methodology**
   - **Adaptive MBPC**
   - **Pros and cons**
   - **Conclusions**
4. **Applications**
   - **Batch reactor control**
   - **A commercial implementation at work**

System Identification

- **This is the field of building process models from experimental data**
- **It consists of the following:**
  - **Experimental planning**
  - **Selection of model structure**
  - **Parameter estimation**
  - **Validation**

Experimental Design

- **Experiments are difficult and costly**
- **Input signal should excite all modes of interest**
  - **Problematic in adaptive control with no active learning**
- **Closed-loop identifiability helped by:**
  - **Nonlinear, or time-varying feedback**
  - **Setpoint changes**

Models for Identification

- Models are really approximations of the real system
- There is no unique model to describe a given process
- Structure derived from prior knowledge and purpose of model
- Natural to use general linear system, called a black-box model
  - Impulse response model
  - Step response model
  - Transfer function model
  - General stochastic model
  - State-space model
  - Laguerre model

The Impulse Response Model

One of the simplest:

$$y(k) = \sum_{j=0}^{\infty} h_j u(k-j-1)$$

h_j is the jth element of the impulse response of the process. Assuming that the impulse response asymptotically goes to zero, it may be truncated after the n_hth element:

$$y(k) = \sum_{j=0}^{n_h} h_j u(k-j-1)$$

This is called a Finite Impulse Response (FIR) model and can only be used for stable processes.

The Step Response Model

Also one of the simplest:

$$y(k) = \sum_{j=0}^{n_s} s_j \Delta u(k - j - 1)$$

s_j **is the** j**th element of the step response of the process.** Δ **is the differencing operator:**

$$\Delta u(k) = u(k) - u(k-1) = (1 - q^{-1})u(k)$$

This is the Finite Step Response (FSR) model used in Dynamic Matrix Control (DMC), it can only be used for stable processes.

The Transfer Function Model

This is the classical discrete-time transfer function model:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})} q^{-d} u(k-1)$$

where d **is the time delay of the process in sampling intervals (**$d \geq 0$**) and the polynomials** A **and** B **are given by:**

$$A(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_{n_A} q^{-n_A}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \cdots + b_{n_B} q^{-n_B}$$

This can be used for both stable and unstable processes. Note that both FIR and FSR models can be seen as subsets of the transfer function model. This model requires less parameters than FIR or FSR, but assumptions about the orders n_A **and** n_B **and the time delay** d **must be made.**

The General Stochastic Model

This is better known as the Box-Jenkins model, and describes both the process and the disturbance:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})} q^{-d} u(k-1) + \frac{C(q^{-1})}{D(q^{-1})} e(k)$$

where $e(k)$ is a discrete white noise sequence with zero mean and variance σ_e.

The General Stochastic Model

The form most used in adaptive control is the so-called CARIMA, or ARIMAX model:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})} q^{-d} u(k-1) + \frac{C(q^{-1})}{A(q^{-1})\Delta} e(k)$$

Putting Δ in the numerator of the noise model means that the noise is nonstationary, ($e(t)/\Delta$ is known as a random walk). This forces an integration in the controller. This is the model used in Generalized Predictive Control (GPC).

The State-Space Model

Not used very often in adaptive control:

$$x(k) = A\,x(k-1) + B\,u(k-1)$$

$$y(k) = c^T x(k)$$

The ARMAX model can also be written in state-space form. One of the advantages of the state-space representation is that it simplifies the prediction. However, system identification is more complex for a state-space model than for a transfer function model.

Continuous Laguerre Functions

- Continuous Laguerre functions form a complete orthonormal set in $L_2[0, \infty)$
- Can be described by

$$F_i(s) = \sqrt{2p}\frac{(s-p)^{i-1}}{(s+p)^i} \qquad , \quad i = 1, .., N$$

where i is the order of the function $(i = 1, ..N)$, and $p > 0$ is the time-scale

Continuous Laguerre Functions

- **An open-loop stable system can be described by the stable, controllable and observable state-space model:**

$$l(t+1) = Al(t) + bu(t)$$

$$y(t) = c^T l(t)$$

with $l^T(t) = [\ l_1(t) \quad l_2(t) \quad \ldots \quad l_N(t)\ ]^T$**, and** $c^T = [c_1 \ \ c_2 \ldots c_N]$

Continuous Laguerre Functions

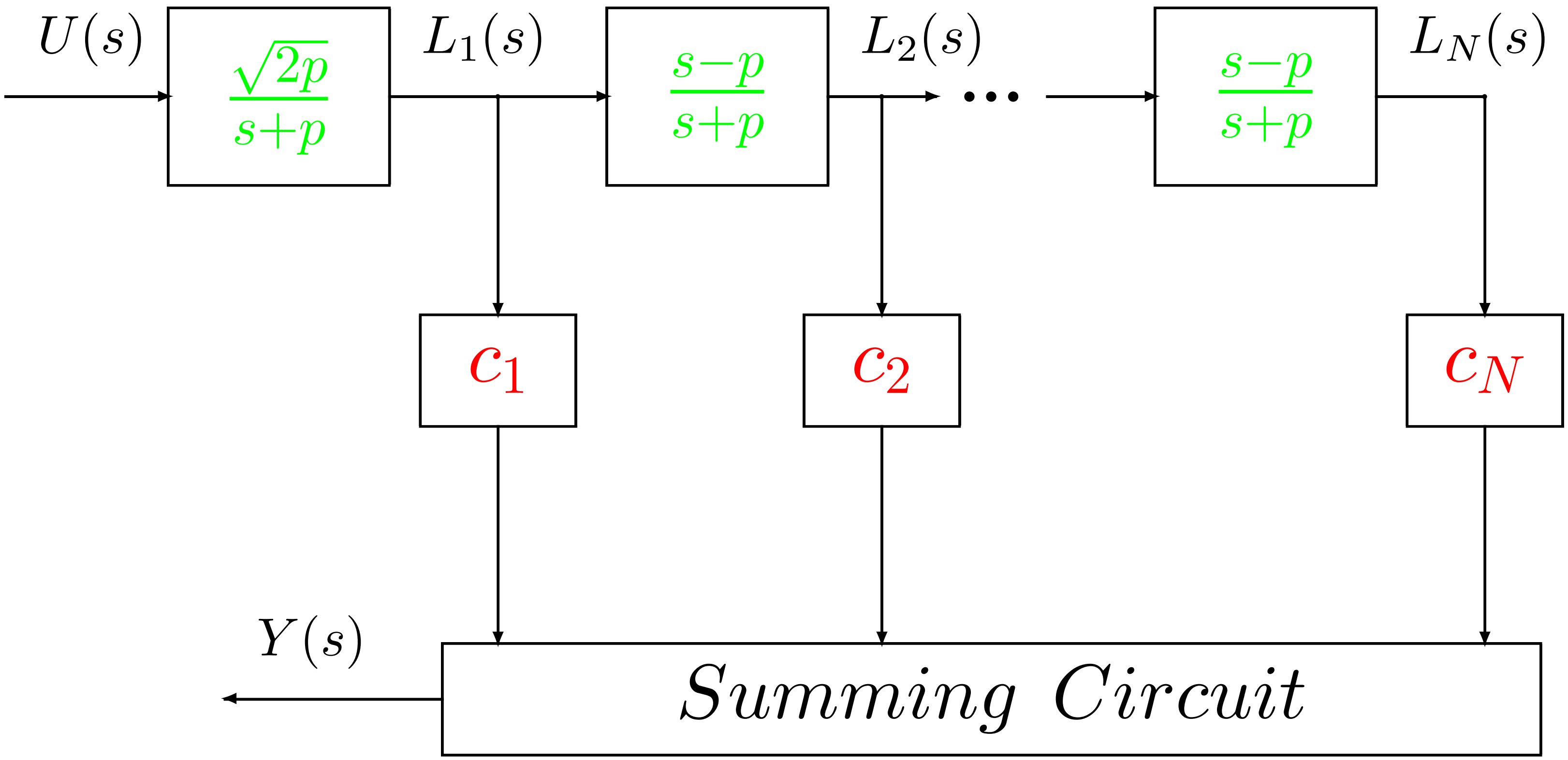
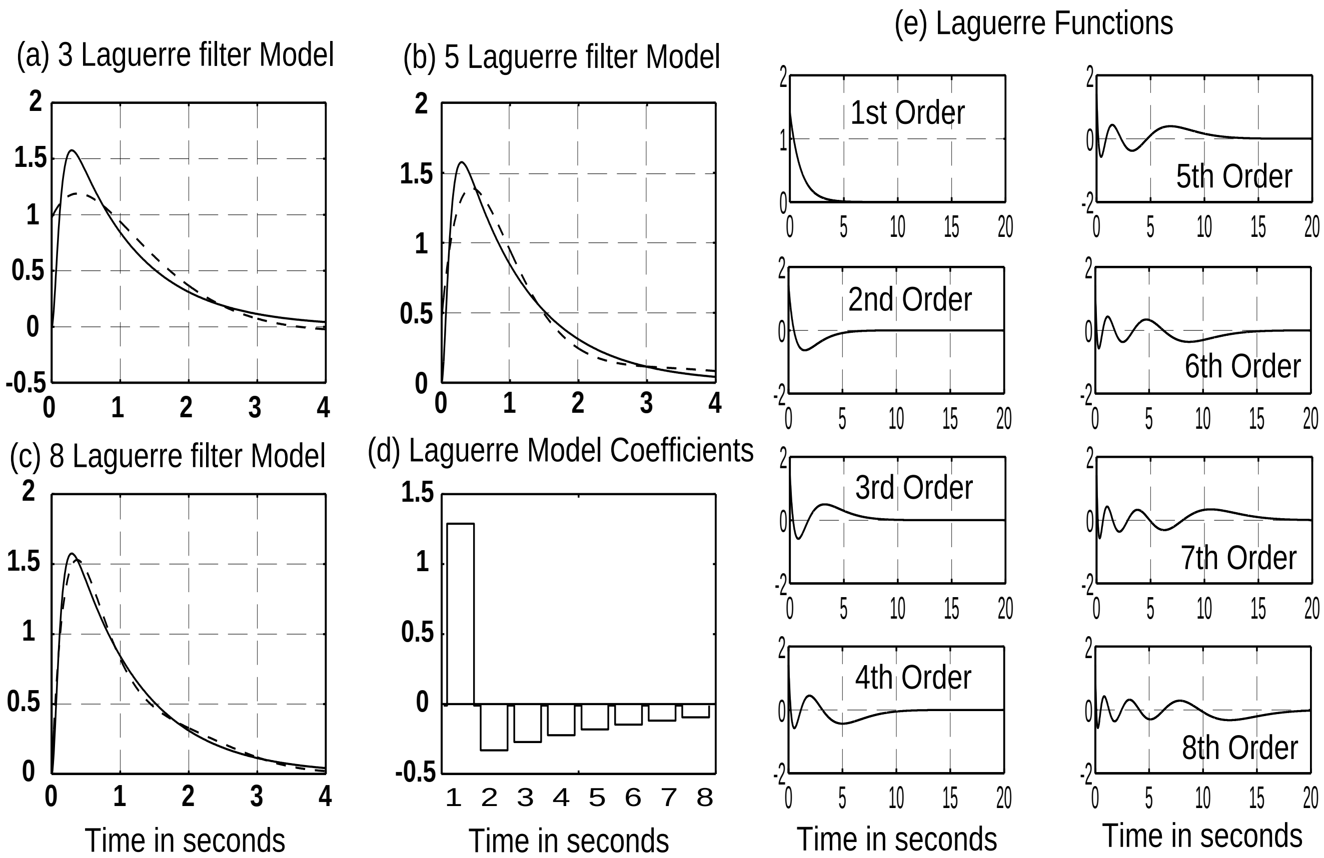


Figure 1: Representation of plant dynamics using a truncated continuous Laguerre ladder network

Example of Laguerre Modelling



Discrete Laguerre Models

- **An open loop stable linear system can be represented with arbitrary accuracy by:**

$$y(k) = \sum_{i=1}^{n} c_i \Phi_i(q^{-1}) u(k)$$

where $\Phi_i(q^{-1})$**, for** $i = 1, \cdots, n$**, are the discrete Laguerre functions:**

$$\Phi_i(q^{-1}) = \sqrt{1-p^2}\;\frac{q^{-1}(q^{-1}-p)^{i-1}}{(1-pq^{-1})^i}, \quad i = 1, \ldots, n$$

$$c_i = \sum_{k=0}^{\infty} h(k) \Phi_i(q^{-1}) \delta(k)$$

where $\delta(k)$ **is the unit impulse and** $h(k)$ **is the system impulse response**

Least-Squares Identification

- **Developed by Karl Gauss in his study of the motion of celestial bodies**
  " . . . the unknown parameters are chosen so that the sum of the squares of the differences between the observed and the computed values, multiplied by a number that measures the degree of precision, is a minimum"

- **Applied to a variety of problems in mathematics, statistics, physics, economics, signal processing and control**
- **Basic technique for parameter estimation**

Least-Squares Estimation

Assume that data generated by $y = bx + n$ where n is white measurement noise. The problem is to estimate b, from k data pairs (x, y).
The least–squares performance index is

$$J = \sum_{i=1}^{k} [y(i) - x(i)b]^2$$

Defining $\underline{y} = [y(1), \ldots, y(k)]^T$ and $\underline{x} = [x(1), \ldots, x(k)]^T$ one can write

$$J = [\underline{y} - b\underline{x}]^T [\underline{y} - b\underline{x}]$$

Least-Squares Estimation

The value that minimizes J is then found as

$$\frac{dJ}{db} = -\underline{x}^T[\underline{y} - b\underline{x}] = 0$$

The estimate $\hat{b}$ is the solution of

$$\underline{x}^T\underline{y} + \underline{x}^T\hat{b}\underline{x} = 0$$

or

$$\hat{b} = [\underline{x}^T\underline{x}]^{-1}\underline{x}^T\underline{y}$$

This is called the least–squares estimate of b.
This idea can easily be extended to dynamic systems.

Least-Squares Identification

Let a dynamic system be described by

$$y(t) = \sum_{i=1}^{n} -a_i y(t-i) + \sum_{i=1}^{n} b_i u(t-i) + w(t)$$

where $u(t)$ and $y(t)$ are respectively the input and ouput of the plant and $w(t)$ is the process noise.
Defining

$$\underline{\theta}^T = [\begin{array}{cccccc} a_1 & \cdots & a_n & b_1 & \cdots & b_n \end{array}]$$

and

$$\underline{x}^T = [\begin{array}{cccccc} -y(t) & \cdots & -y(t-n) & u(t-1) & \cdots & u(t-n) \end{array}]$$

the system above can be written as

$$y(t) = \underline{x}^T(t)\underline{\theta} + w(t)$$

Least-Squares Identification

With N observations, the data can be put in the following compact form:

$$\underline{Y} = X\underline{\theta} + \underline{W}$$

where $\underline{Y}^T = [\ y(1) \quad \cdots \quad y(N)]$ **,** $\underline{W}^T = [\ w(1) \quad \cdots \quad w(N)]$ **and**

$$X = \begin{bmatrix} \underline{x}^T(1) \\ \vdots \\ \underline{x}^T(N) \end{bmatrix}$$

Least-Squares Identification

For each point we define the modelling error as $\epsilon(t) = y(t) - \underline{x}^T(t)\underline{\theta}$ then the least-squares performance index to be minimized is:

$$J = \sum_{1}^{N} \epsilon^2(t)$$

or using the matrix notation above

$$J = [\underline{Y} - X\underline{\theta}]^T[\underline{Y} - X\underline{\theta}]$$

Differentiating with respect to $\underline{\theta}$ and equating to zero then yields $\hat{\underline{\theta}}$, the least-squares estimate of $\underline{\theta}$[1]:

$$\boxed{\hat{\theta} = [X^T X]^{-1} X^T Y}$$

[1] I am now tired of having to underline vectors, so I will stop doing so from now on . . .

Properties of the Least-Squares Estimates

The least-squares estimate can be expressed as

$$\hat{\theta} = [X^T X]^{-1} X^T [X\theta + W]$$

or

$$\hat{\theta} = \theta + [X^T X]^{-1} X^T W$$

Taking the expecting value gives

$$E(\hat{\theta}) = \theta + E\{[X^T X]^{-1} X^T W\}$$

Properties of the Least-Squares Estimates

Note that the second term in the RHS is a bias that is generally non–zero except when

1. $\{w(t)\}$ is zero mean and uncorrelated, or
2. $\{w(t)\}$ is independent of $\{u(t)\}$ and y does not appear in the regressor
   - FIR and step models
   - Laguerre model
   - Output-error models

Only in the above cases does the least–squares method give consistent estimates.

Properties of the Least-Squares Estimates

The estimate covariance is then given by

$$\text{cov } \hat{\theta} = E\{[X^T X]^{-1} X^T W W^T X [X^T X]^{-1}\}$$

which, if $\{w(t)\}$ is white, zero–mean gaussian with covariance σ_w^2, becomes

$$\boxed{\text{cov } \hat{\theta} = \sigma_w^2 [X^T X]^{-1}}$$

Identification of Laguerre Models

Consider the real plant described by

$$y(t) = \sum_{i=1}^{N} c_i L_i(q) + \sum_{i=N+1}^{\infty} c_i L_i(q) + w(t)$$

where $w(t)$ is a disturbance.
This model has an output-error structure, is linear in the parameters, and gives a convex identification problem.

- Even if $w(t)$ it coloured and non-zero mean, simple least-squares provide consistent estimates of the c_i's.
- The estimate of the nominal plant, i.e. of c_i, for $i = 1, \cdots, N$ is unaffected by the presence of the unmodelled dynamics represented by c_i, for $i = N+1, \cdots, \infty$.

Identification of Laguerre Models

- **The mapping $(1 + ae^{i\omega})/(e^{i\omega} + a)$ improves the condition number of the least-squares covariance matrix.**
- **The implicit assumption that the system is low-pass in nature reduces the asymptotic covariance of the estimate at high frequencies**
- **For least-squares, the mean square error of the transfer function estimate can be approximated by**

$$\hat{\pi}(e^{i\omega}) = \frac{1}{2}\left(N(1-\lambda)\frac{1-a^2}{|e^{i\omega}-a|^2}\frac{\Phi_v(e^{i\omega})}{\Phi_u(e^{i\omega})} + \frac{\mu^2}{1-\lambda} r_1(e^{i\omega}) \right)$$

- **The case $a = 0$ corresponds to a FIR model. The MSE is proportional to the number of parameters. Compared with a FIR model, an orthonormal series representation requires less parameters and thus gives a smaller MSE.**
- **The disturbance spectrum is scaled by**

$$\frac{1-a^2}{|e^{i\omega}-a|^2}$$

thus reducing the detrimental effect of disturbances at high frequencies.

Alternatives to Least-Squares

- **Need a method that gives consistent estimates in presence of coloured noise**
- **Generalized Least-Squares**
- **Instrumental Variable Method**
- **Maximum Likelihood Identification**

Recursive Identification

- **There are many situations when it is preferrable to perform the identification on-line, such as in adaptive control.**
- **Identification methods need to be implemented in a recursive fashion, i.e. the parameter estimate at time t should be computed as a function of the estimate at time $t - 1$ and of the incoming information at time t.**
- **Recursive least-squares**

Recursive Least-Squares (RLS)

We have seen that, with t observations available, the least-squares estimate is

$$\theta(\hat{t}) = [X^T(t)X(t)]^{-1}X^T(t)Y(t)$$

with

$$Y^T(t) = [\ y(1) \quad \cdots \quad y(t)]$$

$$X(t) = \begin{bmatrix} x^T(1) \\ \vdots \\ x^T(t) \end{bmatrix}$$

Assuming that one additional observation becomes available, the problem is then to find $\hat{\theta}(t+1)$ as a function of $\hat{\theta}(t)$ and $y(t+1)$ and $u(t+1)$.

Recursive Least-Squares (RLS)

Defining $X(t+1)$ **and** $\underline{Y}(t+1)$ **as**

$$X(t+1) = \begin{bmatrix} X(t) \\ x^T(t+1) \end{bmatrix} \qquad Y(t+1) = \begin{bmatrix} Y(t) \\ y(t+1) \end{bmatrix}$$

and defining $P(t)$ **and** $P(t+1)$ **as**

$$P(t) = [X^T(t)X(t)]^{-1} \qquad P(t+1) = [X^T(t+1)X(t+1)]^{-1}$$

one can write

$$P(t+1) = [X^T(t)X(t) + x(t+1)x^T(t+1)]^{-1}$$

$$\hat{\theta}(t+1) = P(t+1)[X^T(t)Y(t) + x(t+1)y(t+1)]$$

Recursive Least-Squares (RLS)

The use of the matrix-inversion lemma:

$$(A + BC)^{-1} = A^{-1} - A^{-1}B[I + CA^{-1}B]^{-1}CA^{-1}$$

and some simple matrix manipulations then gives the recursive least-squares algorithm:

$$\begin{aligned}
\hat{\theta}(t+1) &= \hat{\theta}(t) + K(t+1)[y(t+1) - x^T(t+1)\hat{\theta}(t)] \\
K(t+1) &= \frac{P(t)x(t+1)}{1 + x^T(t+1)P(t)x(t+1)} = P(t+1)x(t+1) \\
P(t+1) &= P(t) - \frac{P(t)x(t+1)x^T(t+1)P(t)}{1 + x^T(t+1)P(t)x(t+1)}
\end{aligned}$$

Recursive Least-Squares (RLS)

- The recursive least-squares algorithm is the exact mathematical equivalent of the batch least-squares.
- P is proportional to the covariance matrix of the estimate, and is thus called the covariance matrix.
- The algorithm has to be initialized with $\hat{\theta}(0)$ and $P(0)$.
- Generally, $P(0)$ is initialized as αI where I is the identity matrix and α is a large positive number. The larger α, the less confidence is put in the initial estimate $\hat{\theta}(0)$.

Identification in Closed Loop

- **The Identifiability Problem**

  Let a system be described by

$$y(t) + a \cdot y(t-1) = b \cdot u(t-1) + e(t)$$

with $u(t) = g \cdot y(t)$

The closed-loop system can be written as:

$$y(t) + (\hat{a} - \hat{b} \cdot g)y(t-1) = e(t)$$

Hence all estimates $\hat{a}$ **and** $\hat{b}$ **that satisfy**

$$\hat{a} - \hat{b} \cdot g = a - b \cdot g$$

will describe the closed-loop system equally well.

Identification in Closed Loop

All estimates such that

$$\hat{a} = a + k \cdot g$$
$$\hat{b} = b + k$$

will give a good description of the process.

- If the identification is performed using two feedback gains g_1 and g_2 or if the parameter a is fixed, then the system becomes identifiable, because we have as many equations as unknowns.

Ensuring Closed-Loop Identifiability

As a general rule, to ensure closed-loop identifiability, there should not be a linear, noise-free relationship between the process output y and the controller output u In practice, there are two ways to ensure identifiability in closed loop. For a single-input, single-output system these are:

- Switching between at least two independent controllers.
- Adding an external, known perturbation, typically on the reference or setpoint to the loop.

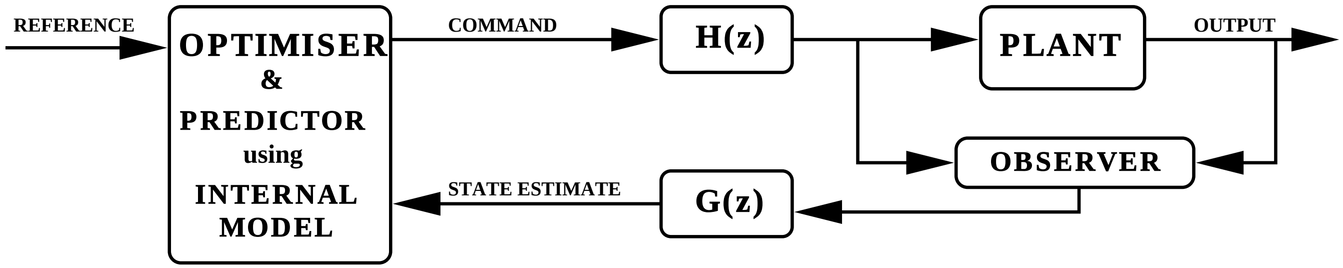
Why $MBPC$ (Model Based Predictive Control) for adaptive control?

- From the controller perspective:
  - It is a model based technique: good models mean good control.
  - MBPC has a simple mathematical formulation (process engineers can absorb this without effort).
  - The methodology is flexible and can be tested sequentially (firstly the control part, then the modelling/identification and ultimately the adaptive control).
  - Adaptive MBPC represents an easy to code algorithm relying on basic mathematical tools (matrix computation and recursive least square).
  - The possibility to include constraints that are acknowledged by the controller on line represents a very useful feature.

Why $MBPC$ (Model Based Predictive Control) for adaptive control? (Continuation)

- From the identification perspective:
  - For unknown models we can perform system identification and then control (INDIRECT ADAPTIVE CONTROL).
  - The need for modelling via identification is increased in process industries
  - Once a closed-loop model has been identified predictions are easily produced.
  - The identification will change the model used in the MBPC scheme only when required (i.e. with enough confidence in the new model).
  - The Laguerre model is a state space realization which couples perfectly with the MBPC state space approach.

The structure of $MBPC$ schemes



- **Observer - provides current state estimates.**
- **Internal model - represents the plant.**
- **Predictor - provides the optimiser with future values of states and outputs.**
- **Optimiser - contains the constrained cost function.**

General characteristics of $MBPC$

- **Explicit internal model gives predictions, assuming given control signals.**
- **Control signals chosen by optimising predicted behaviour.**
- **This is done repeatedly. The optimisation is done on-line.**

The Internal Model

- **Usually linear (Otherwise non-convex optimisation).**
- **All the usual (multivariable) types possible:**
  - **Convolution models (stable systems only!):**

$$y(k) = \sum_{i=-\infty}^{k} g_{k-i}\Delta u(i) + d(k)$$

  - **Transfer function models:**

$$A(z^{-1})y(k) = B(z^{-1})u(k) + n(k)$$

  - **State-space models :**

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + w(k) \\ y(k) &= Cx(x) + Du(k) + v(k) \end{aligned}$$

Getting predictions

- **Convolution model:**

$$\begin{aligned}
\hat{y}(k+j) &= \sum_{i=k+j-N}^{k+j} g_{k+j-i}\Delta u(i) \;+\; \hat{d}(k+j) \\
\hat{d}(k) &= y(k) - \hat{y}(k) \\
\hat{d}(k+j) &= \hat{d}(k) \quad \text{(Most common assumption)}
\end{aligned}$$

- **Transfer function model:**

$$n(k) = \frac{C(z^{-1})}{z-1} e(k) \quad \text{(Most common assumption)}$$

  Solve matrix Diophantine equation
  (Simpler alternatives possible)

- **State-space model: Stochastic w and v — Kalman filter**
  Other disturbance model — Other observer
  (Augment state to shape disturbance spectra etc.)

The problem fomulation

- Choose future control trajectory to minimise some cost function.

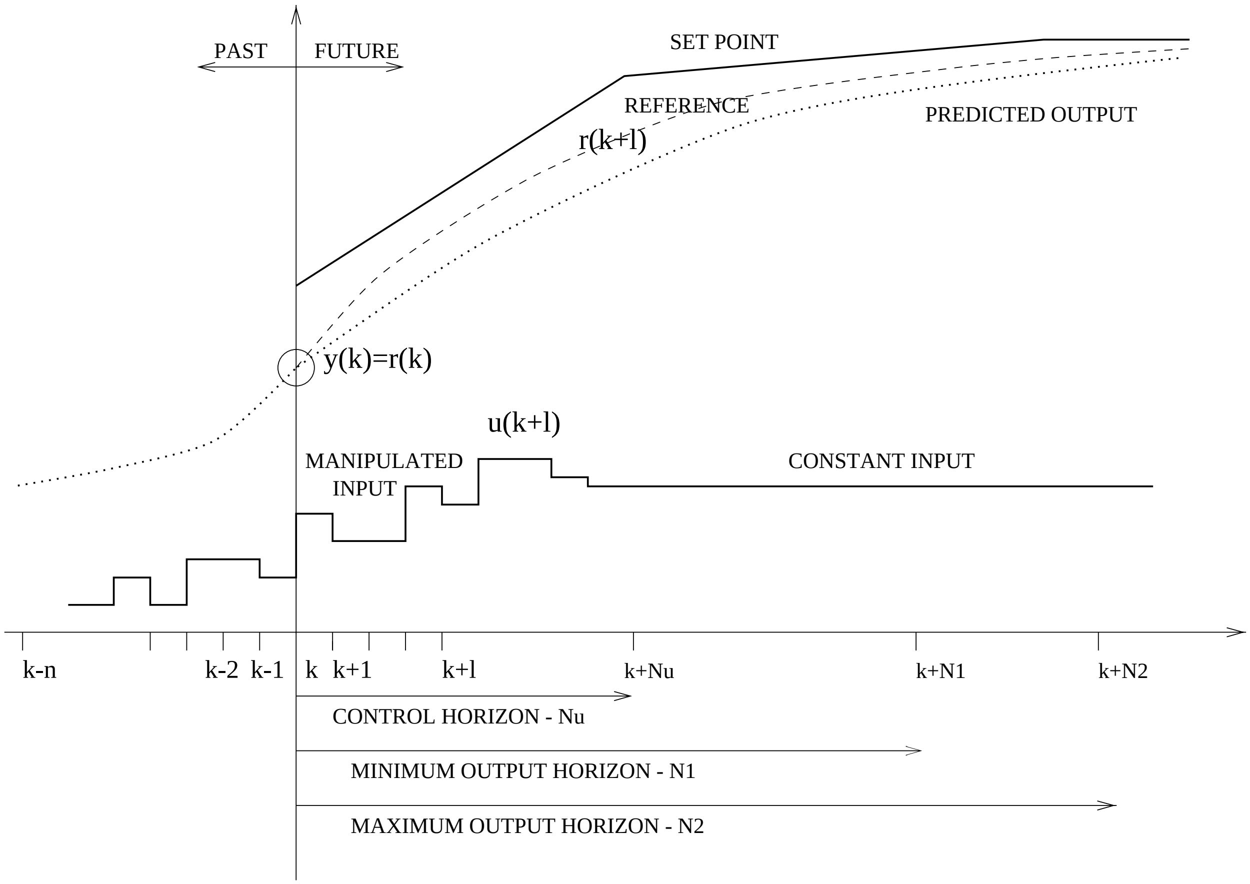
$$J(k) = \sum_{i=N_1}^{N_2} ||M\hat{x}(k+i|k) - r(k+i)||^2_{Q(i)} + \sum_{i=1}^{N_u} ||\Delta u(k+i)||^2_{R(i)}$$

- Subject to constraints such as

$$\begin{aligned} |\Delta u_j(k+i)| &\leq V_j \\ |u_j(k+i)| &\leq U_j \\ |(M\hat{x})_j(k+i|k)| &\leq X_j \end{aligned}$$

- The reference trajectory $r(k)$ can be passed through some filter.
- Note that *changes* in the control signals, $\Delta u(k)$, are penalised in the cost, not the signals themselves — because required values not known in advance.

The Receding Horizon Strategy



Alternative formulations

To ensure stability:

- **Add terminal constraints.**
- **Make horizons infinite**
  - **Difficult to keep constraints.**
  - **But possible using LMI approach (*Kothare & Morari*).**
- **Use Youla parametrisation to optimise over different degrees of freedom (*Kouvaritakis & Rossiter*).**

To get robustness:

- **Min-Max instead of quadratic cost (Harder problem).**
- **Optimise over polytope of models (LMI: *Kothare & Morari*).**

To get simpler problem:

- **Replace quadratic cost by absolute value — LP instead of QP.**

Solution techniques (I)

The constrained optimisation problem must be solved on-line, in real time.

- **Problem is usually a Quadratic Program. So solve using standard QP algorithm.**

- **Speed: Need something like 10^3 speed-up to allow use for high bandwidth control. Possible but not today. The method is feasible today for low bandwidth control.**

- **Does not exploit structure of Predictive Control problem (eg next solution usually very close to last one), or structure of particular application (eg flight control).**

- **Constraint formulation gives up if problem not feasible — need safety code.**

Solution techniques (II)

- **Mixed Weighted Least Squares algorithm (*Lawson*)**
  - **Solves QP problem if feasible.**
  - **Gives 'reasonable' solution if not feasible.**
  - **Adjusts weight to emphasise most-violated constraint.**
  - **Reaches feasible solution quickly.**

- **LMI algorithms**
  - **Can solve wider class of problems than QP.**
  - **Speed: Have reputation for being slow.**
  - **To be explored. Big potential. Currently trendy in Control.**

Controller Structure

- Controller structure not conventional.
- Complexity and behaviour of observer and predictor can be compared with conventional controllers.
- Complexity and behaviour of optimiser difficult to compare.

Controller Properties

- **Linear model & Quadratic cost implies a 'piecewise LTI' controller.**
  - **Linear Time Invariant (LTI) control law while set of active constraints remains unchanged.**
  - **Switches to different LTI law when set of active constraints changes.**
  - **Scope for analysis? Current research in 'intelligent', 'hybrid' and 'adaptive' systems emphasises piecewise-LTI systems. Results may be applicable here.**
- **Exhibits very nonlinear behaviour when necessary — eg when disturbance drives system up against constraint boundary.**
- **Standard disturbance model results in integral action. Can be generalised to reject ramps, sinusoids etc.**
- **Verification/certification difficult.**

Controller Tuning (I)

'Tuning knobs':

- **Horizons —** N_1**,** N_2**,** N_u**.**
- **Weights —** $Q(i)$**,** $R(i)$**, (**$i = 1, 2, \ldots$**).**
- **Prefilter — if 'reference'** $r(k)$ **different from set-point.**
- **Predictor — assumptions about disturbances and noise.**

Controller Tuning (II)

Specifications:

- **Time-domain specs**
  - **Enforce through constraints (sometimes).**
  - **But controller nonlinear — step response etc has limited meaning.**
- **Frequency-domain specs**
  - **Can be checked for a given choice of 'tuning knob' settings, using piecewise-LTI property.**
  - **No systematic way of tuning (yet).**
- **Stability**
  - **Several known ways of ensuring stability, even with constraints.**
  - **Proofs usually based on showing that cost function is a Lyapunov function.**

How to tune is not clear yet. But much progress is currently being made.

Controller characteristics

Bad features:

- **Heavy computational load limits control update rate (at present!).**
- **Analysis of resulting behaviour difficult.**

Good features:

- **Respects hard constraints:**
  - **Actuator constraints**
  - **Airframe constraints**
  - **Flight envelope constraints**
- **Reference management from data base is possible**

A bit more insight in the MBPC state space approach (I)

Considering a possible multi-model context, let index i represent the plant in use (this opens the adaptive control path):

$$
\begin{aligned}
x_i(k+1) &= A_i x_i(k) + B_{u_i} u_i(k) + B_{d_i} d_i(k) + B_{w_i} w_i(k) \\
y_i(k) &= C_i x_i(k) + \nu(k)
\end{aligned}
\tag{1}
$$

where:

- $x_i(k) \in \mathbb{R}^n$ **is the system state vector**
- $u_i(k) \in \mathbb{R}^m$ **is the system control input vector**
- $y_i(k) \in \mathbb{R}^p$ **is the vector of outputs**
- **All variables being measured at time** k.

A bit more insight in the MBPC state space approach (II)

We express the state vector in terms of the change in the manipulated variable (at time k):

$$\begin{bmatrix} \Delta x_i(k+1) \\ y x_i(k+1) \end{bmatrix} = \begin{bmatrix} A_i & 0 \\ C_i A_i & I \end{bmatrix} \begin{bmatrix} \Delta x_i(k) \\ y x_i(k) \end{bmatrix} + \begin{bmatrix} B u_i \\ C_i B u_i \end{bmatrix} \Delta u_i(k) + \begin{bmatrix} B_{d_i} & B w_i \\ C_i B_{d_i} & I \end{bmatrix} \begin{bmatrix} \Delta d_i(k) \\ \Delta w_i(k) \end{bmatrix}$$

$$y_i(k) = \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta x_i(k) \\ y x_i(k) \end{bmatrix} + \nu(k)$$

where:

- $y_{x_i}(k+1)$ corresponds to a vector of outputs free from measurement noise.
- $\Delta u_i(k) = u_i(k+1) - u_i(k)$ changes in the manipulated variable
- $\Delta d_i(k) = d_i(k+1) - d_i(k)$ the state disturbance
- $\Delta w_i(k) = w_i(k+1) - w_i(k)$ the output disturbance

A bit more insight in the MBPC state space approach (III)

The vector of future state estimates may be written as:

$$\hat{X}_i(k) = \begin{bmatrix} \hat{x}_i(k+1) \\ \vdots \\ \hat{x}_i(k+N_2) \end{bmatrix} = \mathcal{F}_i(k)\hat{x}_i(k) + \mathcal{G}_i \Delta U_i(k) + \mathcal{H}_i(k) y_i(k) \qquad (2)$$

In the above equations:

- $\Delta U_i(k) = \left[\Delta u_i(k)^T, \ldots, \Delta u_i(k+N_u-1)^T\right]^T$ denotes the vector of future manipulated variable increments
- $\hat{X}_i(k) = \left[\hat{x}_i(k+1)^T, \ldots, \hat{x}_i(k+N_2)^T\right]^T$ denotes the vector of the change of state estimates which is dependent only upon $\hat{x}_i(k)$ (the state estimate at time k).
- It is necessary to have a state estimator to determine current state estimate $\hat{x}_i(k)$.

A bit more insight in the MBPC state space approach (IV)

It is possible to obtain the solution of the optimisation problem as a result of the following QP problem:

$$\min_{\Delta U_i(k)} \{\Delta U_i(k)^T \mathcal{A}_i \Delta U_i(k) +$$

$$\left[\hat{x}_i(k)^T, y_i(k)^T, \mathcal{S}(k)^T\right] \mathcal{B}_i \Delta U_i(k)\}$$

subject to the constraints:

$$\mathcal{D}_i \Delta U_i(k) \leq \mathcal{E}_i \begin{bmatrix} u_i(k-1) \\ \hat{x}_i(k) \\ y_i(k) \\ c_i(k) \end{bmatrix}$$

The robust tracking idea

Internal model principle **states that, in order to have a disturbance rejected or a reference tracked, its corresponding model has to be included in the controller.**

Our approach is to augment the plant model with the disturbance model and then use the augmented model as a prediction model to construct the optimiser cost function. This formulation allows to describe a wide variety of signals such as: steps, ramps or sinusoids.

The disturbance and/or reference signal satisfies the following difference equation:

$$n(k+1) + \alpha_1 n(k) + \alpha_2 n(k-1) = 0 \tag{3}$$

$$\begin{aligned} \begin{bmatrix} n(k) \\ n(k+1) \end{bmatrix} &= \begin{bmatrix} 0 & I \\ -\alpha_2 I & -\alpha_1 I \end{bmatrix} \begin{bmatrix} n(k-1) \\ n(k) \end{bmatrix} \\ n(k) &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} n(k-1) \\ n(k) \end{bmatrix} \end{aligned}$$

The state space representation including the disturbance model

$$\begin{bmatrix} \xi(k+1) \\ \eta(k+1) \\ y_x(k) \\ y_x(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 & 0 & 0 \\ 0 & A_n & 0 & 0 \\ 0 & 0 & 0 & I \\ CA & 0 & -\alpha_2 I & -\alpha_1 I \end{bmatrix} \begin{bmatrix} \xi(k) \\ \eta(k) \\ y_x(k-1) \\ y_x(k) \end{bmatrix} + \begin{bmatrix} B \\ 0 \\ 0 \\ CB \end{bmatrix} \psi(k) \tag{4}$$

$$\tilde{y}(k) = \begin{bmatrix} 0 & C_n & 0 & I \end{bmatrix} \begin{bmatrix} \xi(k) \\ \eta(k) \\ y_x(k-1) \\ y_x(k) \end{bmatrix}$$

where: $y_x(k)$ **– vector of outputs free of disturbance.**

$\psi(k) = u(k) + \alpha_1 u(k-1) + \alpha_2 u(k-2)$ **– the generalised input.**

$\xi(k) = x(k) + \alpha_1 x(k-1) + \alpha_2 x(k-2)$ **– the generalised state.**

$\zeta(k) = \tilde{y}(k) + \alpha_1 \tilde{y}(k-1) + \alpha_2 \tilde{y}(k-2)$ **– the corresponding output.**

Algorithm related conclusions

- **An advanced model based predictive controller $MBPC$ developed for use on processes with an integrating response exibiting long dead time and time constants is now available.**
- **The controller is fairly easy to apply and configure without doctoral level of knowledge.**
- **The controller provides good performance (see last part of the course - applications). The robustness issue can be addressed via the adaptation mechanism.**
- **MBPC represents an alternative to PID control especially for difficult processes.**
- **Including the disturbance model together with the prediction model ensures offset-free control even when there is a significant mismatch between the model and the actual plant.**

Application related conclusions

- **Routinely used in petrochemicals — mature technology in this area.**
- **Increasingly used in other process sectors.**
- **High-bandwidth electromechanical application examples:**
  - **S.Abu el Ata-Doss, P.Fiani and J.Richalet,** *Handling input and state constraints in predictive functional control*, **Proc. CDC, Brighton UK, 1991.**
  - **A.J.Beaumont, A.D.Noble and A.S.Mercer,** *Predictive Control of Transient Engine Testbeds*, **Proc. Control 88, Oxford UK, 1988.**
- **Aerospace application examples:**
  - **S.N.Singh, M.Steinberg and R.D.DiGirolamo,** *Nonlinear Predictive Control of Feedback Linearizable Systems and Flight Control System Design*, **AIAA Jnl. Guidance, Control and Dynamics, 1995.**
  - **D.G.Ward and R.L.Barron,** *A self-designing receding horizon optimal flight controller*, **Proc. ACC, 1995.**