PFC
Predictive Functional Control

Prof. Cesar de Prada
Dpt. of Systems Engineering and Automatic Control
University of Valladolid, Spain
prada@autom.uva.es
Outline

✔ As simple as possible
  – Motivation
  – PFC main ideas
  – An introductory example
Motivation

✓ Predictive control is a widely used control technique in the process industry combined with an optimization layer

✓ Most of the implementations use internal linear models (step responses) and LP optimizers
DMC

**DMC no constraints**
- Computes MVs
- Sets optimal MVs

**LP, economic optimizer**
- Set target

**Past** → **Future**
- Set point
- Output prediction

**Time**
- MV, OP
- CV, PV

**SP**
- u(t)
- t, t+1, t+2, ...
Motivation

- When a significant non-linear process is faced, a non-linear controller based on a non-linear plant model is required.
- Current approaches based on first-principles models, NN, volterra series, etc. lead to a non-linear optimization problem that must be solved on-line.
- This is a heavy burden, both from the implementation and computational load.
Non-linear Predictive Control

Optimizer

Predictor

Process

\[ \min_{\Delta u} J = \sum_{j=1}^{N_2} \left[ \hat{y}(t + j) - w(t + j) \right]^2 + \sum_{j=0}^{N_u-1} \left[ \beta \Delta u(t + j) \right]^2 \]

\[ \dot{x} = f(x(t), u(t)) \]

\[ y(t) = g(x(t), u(t)) \]

\[ y \leq y(t + j) \leq \bar{y} \]

\[ _{\underline{u}} \leq u(t + j) \leq \bar{u} \]
Solving the dynamic NLP problem

Simulation from $t$ to $t+N2$ for computing $J(u,x(t))$

Optimizer

Process

State constraints

Gradients
The aim of Parametric Predictive Control (PPC) is to facilitate the implementation of the controller while retaining the main non-linear characteristics in its internal model.

Good compromise between speed of execution, easy of implementation and performance.

Target: Embedded controller in a DCS.
**PPC Main ideas**

- Combines first principles models with MPC ideas and generates a simplified solution that is updated every sampling time.
- **Parametric Predictive Control (PPC)** (J. Richalet, 1996) was developed for, and successfully applied to, temperature control of batch reactors.
- A. Assandri, A. Rueda, PhD students
- Three steps:
  - Basic ideas presented in the **linear** case
  - Extended to the **non-linear** case with a chemical reactor CSTR
  - **Industrial application** to the bottom temperature control of a distillation column heated with a furnace-reboiler.
Basic concepts

Aim: Decrease the error in the future until a certain percentage of the current error $w - y(t)$

This implies to change the process output by $\Delta p$

Use the model to compute the control $u(t)$ that provides a change in the model output $\Delta m (u) = \Delta p$

Key design equation: $\Delta_m (u) = \Delta_p$
Basic concepts

The future error at \( t+N \) must be a fraction of the current error \( w - y(t) \)

\[
E_d = \lambda^N (w - y_p(t)) \quad 0 < \lambda < 1
\]

\[
\Delta_p = (1 - \lambda^N)(w - y_p(t))
\]

Example: First order model

\[
\tau \frac{dy(t)}{dt} + y(t) = ku(t)
\]
Basic concepts (Monoreg)

Example: First order model

\[ \tau \frac{dy(t)}{dt} + y(t) = k u(t) \]

Assuming that \( u(t) \) is kept constant along the prediction horizon, that is \( N_u = 1 \), and with initial condition \( y_p(t) \):

- \( T_s \) sampling time
- \( N \) prediction (or coincidence) horizon

\[
\Delta y_m = \left( e^{-\frac{N T_s}{\tau}} - 1 \right) y_p(t) - k u(t) \left( e^{-\frac{N T_s}{\tau}} - 1 \right)
\]

\[
y_m(t + N T_s) = e^{-\frac{N T_s}{\tau}} y_p(t) + k u(t) \left( 1 - e^{-\frac{N T_s}{\tau}} \right)
\]
Design equation

\[ \Delta_p = (1 - \lambda^N)(w - y_p(t)) \]

\[ \Delta y_m = \left( e^{-\frac{NT_s}{\tau}} - 1 \right)y_p(t) - k u(t) \left( e^{-\frac{NT_s}{\tau}} - 1 \right) \]

design equation: \( \Delta_m(u) = \Delta_p \)

Explicit solution for the control signal

\[ u(t) = \frac{\left( e^{-\frac{NT_s}{\tau}} - 1 \right)y_p(t) - \left(1 - \lambda^N\right)\left[w(t) - y_p(t)\right]}{K \left( e^{-\frac{NT_s}{\tau}} - 1 \right)} \]
Tuning parameters

A discrete first order system with pole $\lambda$ will give a free response as the one desired at $t+N\tau$

\[
\frac{1}{1-\lambda q^{-1}}
\]

$N$ prediction horizon (number of sampling periods required to decrease the current error by $\lambda^N$)

$\lambda$ Reduction factor ($0,..,1$)
Example. Ideal case

Model = Process

\[ G(s) = \frac{1}{5s + 1} \]

\[ K = 1 \]

\[ \tau = 5 \text{ min.} \]

\[ N = 20 \]

\[ \lambda = 0.8 \]
Extension to $N_u > 1$

Example: First order model

$$\frac{dy(t)}{dt} = \frac{y(t)}{\tau} + k u(t)$$

$N_u = 2$

Predictions are more complex

$$y_m(t + T_s) = e^{\frac{-T_s}{\tau}} y_p(t) + k u(t) \left( 1 - e^{\frac{-T_s}{\tau}} \right)$$

$$y_m(t + N T_s) = e^{\frac{-(N-1)T_s}{\tau}} y_m(t + T_s) + k u(t + T_s) \left( 1 - e^{\frac{-(N-1)T_s}{\tau}} \right)$$

Two unknowns: $u(t)$, $u(t+T_s)$, then two coincidence points are required.
Robustness

10% change in the parameters

Model ≠ Process

Model:

\[ G(s) = \frac{1}{5.5s + 1} \]

\[ N = 20 \]

\[ \lambda = 0.8 \]

A bit slower
Robustness

Model ≠ Process

Model:

$$G(s) = \frac{0.9}{5s + 1}$$

$$N = 20$$

$$\lambda = 0.8$$

Solution: error model or explicit integrator added
Incorporating errors

Model:

\[
\tau \frac{dy(t)}{dt} + y(t) = k u(t) + v
\]

\(v\) disturbance

Assuming also that \(v\) does not change along the prediction horizon:

\[
y(t + NT_s) = e^{-\frac{NT_s}{\tau}} y_p(t) + K u(t) \left(1 - e^{-\frac{NT_s}{\tau}}\right) + v \left(1 - e^{-\frac{NT_s}{\tau}}\right)
\]

\[
\Delta y_m = \left(e^{-\frac{NT_s}{\tau}} - 1\right) y_p(t) + K u(t) \left(1 - e^{-\frac{NT_s}{\tau}}\right) + v \left(1 - e^{-\frac{NT_s}{\tau}}\right)
\]
Design equation

\[ \Delta_p = (1 - \lambda^N)(w - y_p(t)) \]

\[ \Delta y_m = \left( e^{-\frac{NT_s}{\tau}} - 1 \right) y_p(t) + K u(t) \left( 1 - e^{-\frac{NT_s}{\tau}} \right) + v \left( 1 - e^{-\frac{NT_s}{\tau}} \right) \]

design equation: \( \Delta_m(u) = \Delta_p \)

Controller equation.

\( v \) is not known and needs to be estimated every sampling time
Estimating \( v \)

\( v \) is estimated from the process model in order to cancel the difference between the measured process output at time \( t \) and the prediction made with values at \( t - T_s \).

\[
y_p(t) = e^{-\frac{T_s}{\tau}} y_p(t - T_s) + Ku(t - T_s) \left(1 - e^{-\frac{T_s}{\tau}}\right) + v \left(1 - e^{-\frac{T_s}{\tau}}\right)
\]

\[
y_p(t) - e^{-\frac{T_s}{\tau}} y_p(t - T_s) - Ku(t - T_s) \left(1 - e^{-\frac{T_s}{\tau}}\right)
\]

\[
\hat{v} = \frac{y_p(t) - e^{-\frac{T_s}{\tau}} y_p(t - T_s) - Ku(t - T_s) \left(1 - e^{-\frac{T_s}{\tau}}\right)}{\left(1 - e^{-\frac{T_s}{\tau}}\right)}
\]
Example

The error estimation compensates the modelling errors

Model:

\[ G(s) = \frac{0.9}{5.5s + 1} \]

Process:

\[ N = 20 \]

\[ \lambda = 0.8 \]
Adding an integrator

\[ i(t) = \frac{k_p}{T_i} \int_{0}^{t} e(\tau) d\tau \]

\[ i(t) = i(t - T_s) + \frac{T_s}{T_i} e(t - T_s) = i(t - T_s) + \frac{T_s}{T_i} (w(t - T_s) - y_p(t - T_s)) \]

\[ u(t) = \frac{\left( e^{-\frac{N T_s}{\tau}} - 1 \right) y_p(t) - (1 - \lambda^N) \left[ w(t) - y_p(t) \right] + i(t - T_s) + \frac{T_s}{T_i} e(t - T_s) }{K \left( e^{-\frac{N T_s}{\tau}} - 1 \right)} \]
Example

Model ≠ Process

Model:

\[ G(s) = \frac{0.9}{5.5s + 1} \]

N = 20
λ = 0.8
\( T_i = 0.1 \text{ min} \)
\( T_s = 20 \text{ sec.} \)
First order + delay

Model:

\[ \tau \frac{dy(t)}{dt} + y(t) = k u(t - d) + v \]

\( y(t) \) disturbance
\( d \) delay
\( d = DT_s \)

\[ \Delta_p = (1 - \lambda^{N-D})[w(t) - y_p(t)] \]
First order + delay

\[ \Delta_p = \left(1 - \lambda^{N-D}\right)[w(t) - y_p(t)] \]

\[ \Delta y_m = \left( e^{-\frac{NT_s}{\tau}} - 1 \right) y_p(t) + e^{-\frac{NT_s}{\tau}} \sum_{m=t}^{t+D-1} u(m-D)e^{\frac{(m-t)T_s}{\tau}} + \]

\[ + u(t)K\left(1-e^{-\frac{(N-D)T_s}{\tau}}\right) + v\left(1-e^{-\frac{NT_s}{\tau}}\right) \quad N > D \]

\[ u(t) = \left[ K\left(1-e^{-\frac{(N-D)T_s}{\tau}}\right) \right]^{-1} \left\{ \left(1 - \lambda^{N-D}\right)[w(t) - y_p(t)] - \left( e^{-\frac{NT_s}{\tau}} - 1 \right)y_p(t) - \right. \]

\[ \left. - e^{-\frac{NT_s}{\tau}} K\left( e^{\frac{T_s}{\tau}} - 1 \right) \sum_{m=t}^{t+D-1} u(m-D)e^{\frac{(m-t)T_s}{\tau}} - v\left(1-e^{-\frac{NT_s}{\tau}}\right) \right\} \]
Estimating $v$

$v$ is estimated from the process model in order to cancel the difference between the measured process output at $t$ and the prediction made with values at $t - T_s$.

$$\hat{v} = \frac{y_p(t - T_s) - K u(t - (D+1)T_s) \left( 1 - e^{-\frac{T_s}{\tau}} \right)}{\left( 1 - e^{-\frac{T_s}{\tau}} \right)}$$

+ smoothing filter
Example. Ideal case

Model =
Process

\[ G(s) = \frac{e^{-10s}}{5s + 1} \]

- \( K = 1 \)
- \( \tau = 5 \text{ min.} \)
- \( d = 10 \text{ min.} \)
- \( N = 40 \)
- \( \lambda = 0.8 \)
- \( T_s = 1 \text{ min.} \)
Robustness

The error estimation compensates the modelling errors

Model:
\[ G(s) = \frac{0.9e^{-9s}}{5.5s + 1} \]

\[ N = 40 \]

\[ \lambda = 0.8 \]