Model Predictive Control

WebCAST Seminar
May 7, 2007

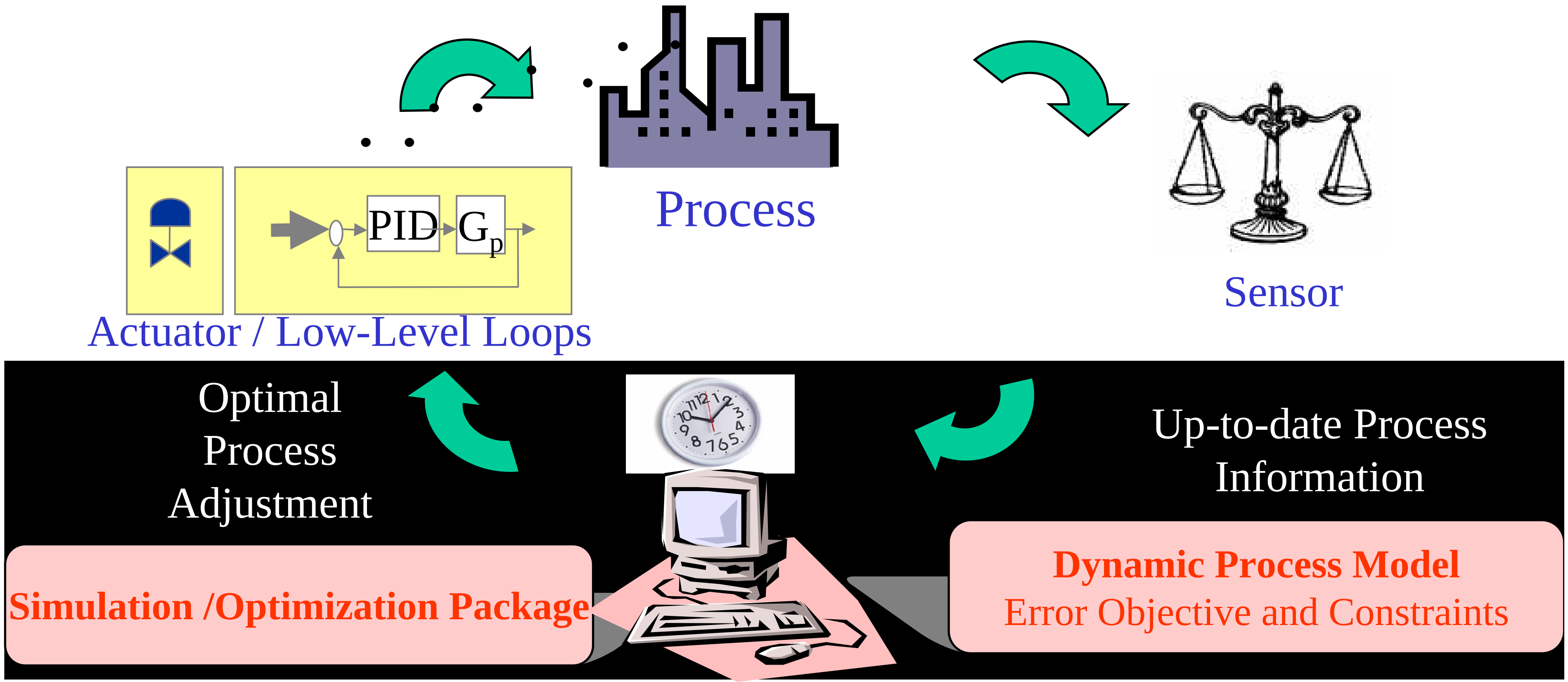
Professor Jay H. Lee

School of Chemical and Biomolecular Engineering

Georgia Institute of Technology

©Jay H. Lee, 2007

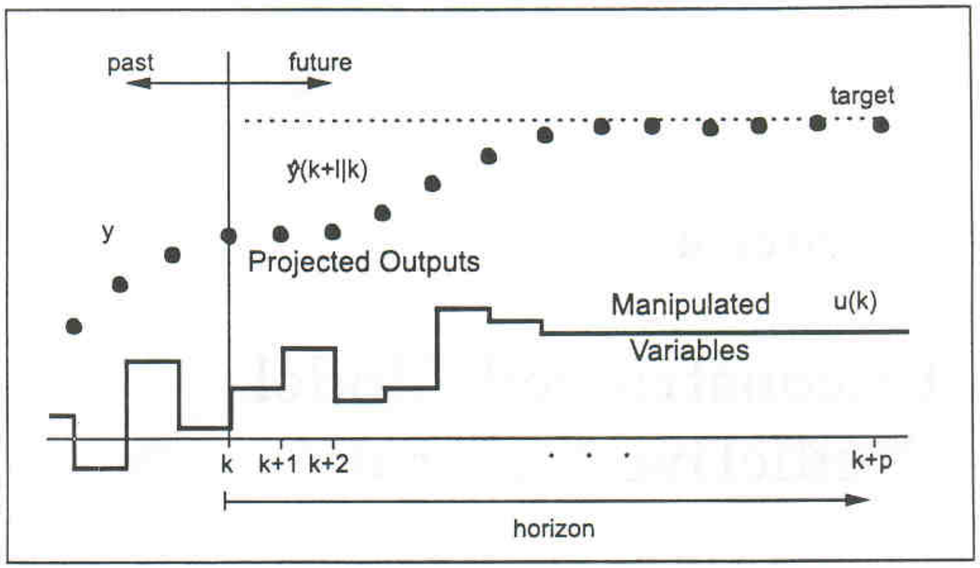
Overview



- Popularized in the late 70s and early 80s in refineries
- Standard APC method for refineries and petrochemical plants
- 4500+ reported industrial applications (Yr. 2000)
- Many vendors marketing software and engineering service
  - –Aspen Tech, Honeywell, Invensys, etc.
- Strong theoretical basis and systematic design for stability and performance

Some Key Features

- **Computer based**: Sampled-data control
- **Model based**: Requires a dynamic process model (fundamental or empirical)
- **Feedback Update:** Model updated using on-line measurements.
- **Predictive**: Makes explicit prediction of the future time behavior of CVs within a chosen window.



Some Key Features(2)

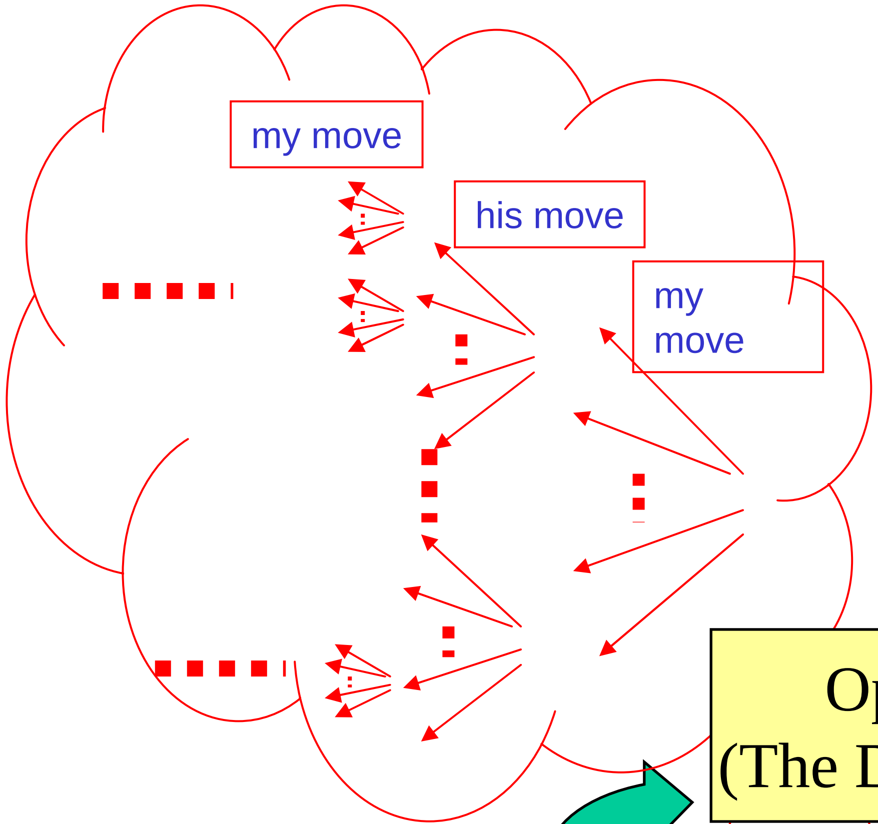
- **Optimization Based**: Performs optimization (numerical search) on-line for optimal control adjustments.

$$\min_{u_i} \sum_{i=0}^{p} \phi_i(x_i, u_i)$$
$$g_i(x_i, u_i) \geq 0 \quad \xrightarrow{?} \quad u_0 = \mu(x_0)$$
$$x_{i+1} = F(x_i, u_i) \qquad \text{HJB Eqn.}$$

No *explicit* form of control law – just model, objective function, and constraints are specified.

- **Integrated** constraint handling and economic optimization with regulatory and servo control.
- **Receding Horizon Control**: Repeats the prediction and optimization at each sample time step to update the optimal input trajectory after a feedback update.

Analogy to Chess Playing



Opponent
(The Disturbance)

The Opponent's
Move

My
Move

I
(The Controller)

New State

Industrial Use of MPC

Industrial Use of MPC

- Some trial of computer based control during 50s-60s (e.g., Standard Oil / IBM).
- Reappeared at Shell Oil and other refineries during late 70s and early 80s. – easier, cheaper implementation enabled by advances in microprocessors.
- Various commercial software
- Tens of thousands of worldwide installations
- Predominantly in the oil and petrochemical industries but the range of applications is expanding.
- Models used are predominantly <u>empirical</u> models developed through plant testing.
- The technology is not only for multivariable control, but for most economic operation within constraint boundaries.

Result of a Survey in 1999 (Qin and Badgwell)

Area	Aspen Technology	Honeywell Hi-Spec	Adersa[1]	Invensys	SGS[2]	Total
Refining	1200	480	280	25		1985
Petrochemicals	450	80	-	20		550
Chemicals	100	20	3	21		144
Pulp and Paper	18	50	-	-		68
Air & Gas	-	10	-	-		10
Utility	-	10	-	4		14
Mining/Metallurgy	8	6	7	16		37
Food Processing	-	-	41	10		51
Polymer	17	-	-	-		17
Furnaces	-	-	42	3		45
Aerospace/Defense	-	-	13	-		13
Automotive	-	-	7	-		7
Unclassified	40	40	1045	26	450	1601
Total	1833	696	1438	125	450	4542
First App.	DMC:1985 IDCOM-M:1987 OPC:1987	PCT:1984 RMPCT:1991	IDCOM:1973 HIECON:1986	1984	1985	
Largest App	603x283	225x85	-	31x12	-	

Linear MPC Vendors and Packages

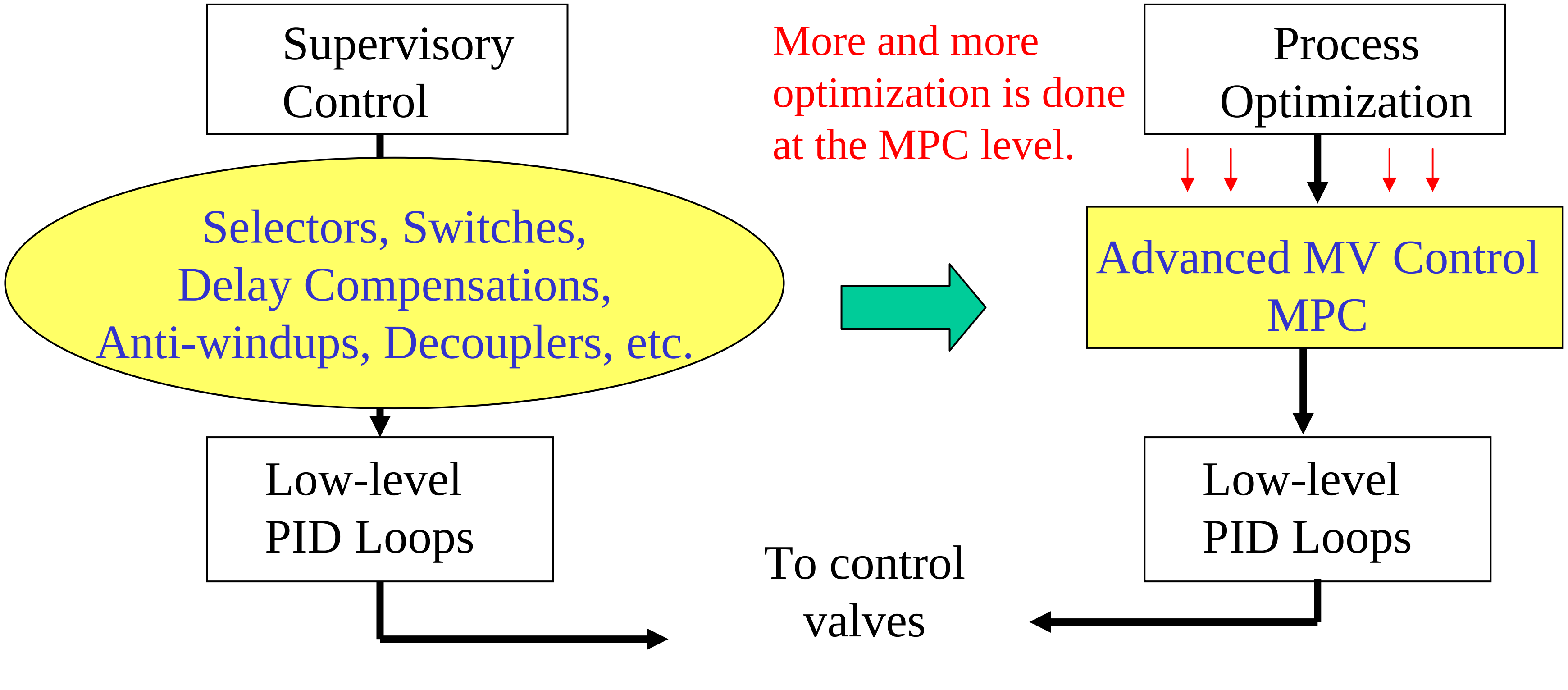
- Aspentech
  - DMCplus
  - DMCplus-Model
- Honeywell
  - Robust MPC Technology (RMPCT)
- Adersa
  - Predictive Functional Control (PFC)
  - Hierarchical Constraint Control (HIECON)
  - GLIDE (Identification package)
- MDC Technology (Emerson)
  - SMOC (licensed from Shell)
  - Delta V Predict
- Predictive Control Limited (Invensys)
  - Connoisseur
- ABB
  - 3d MPC

Result of A Survey for Nonlinear MPC (Qin and Badgwell)

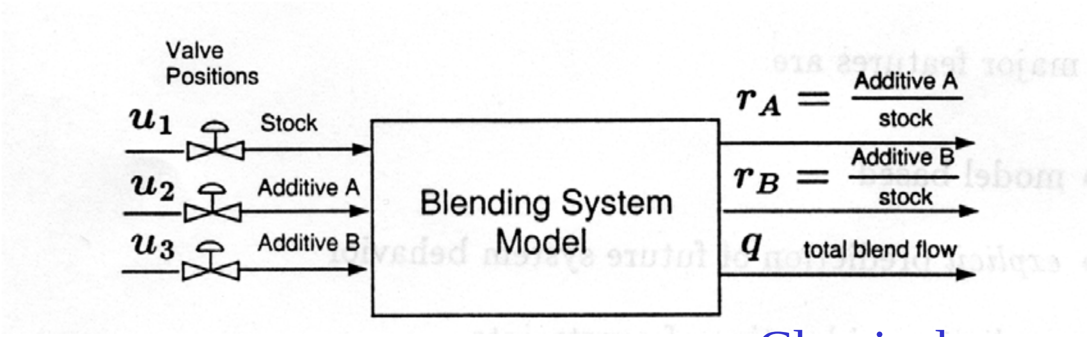
Area	Adersa	Aspen Technology	Continental Controls	DOT Products	Pavilion Technologies	Total
Air and Gas			18			18
Chemicals	2		15		5	22
Food Processing					9	9
Polymers		1		5	15	21
Pulp & Paper					1	1
Refining					13	13
Utilities		5	2			7
Unclassified	1		1			2
Total	3	6	36	5	43	93

Reason for Popularity(1)

- MPC provides a **systematic, consistent, and integrated solution** to process control problems with complex features:
  - Delays, inverse responses and other complex dynamics.
  - Strong interactions (e.g., large RGA)
  - Constraints (e.g., actuator limits, output limits)

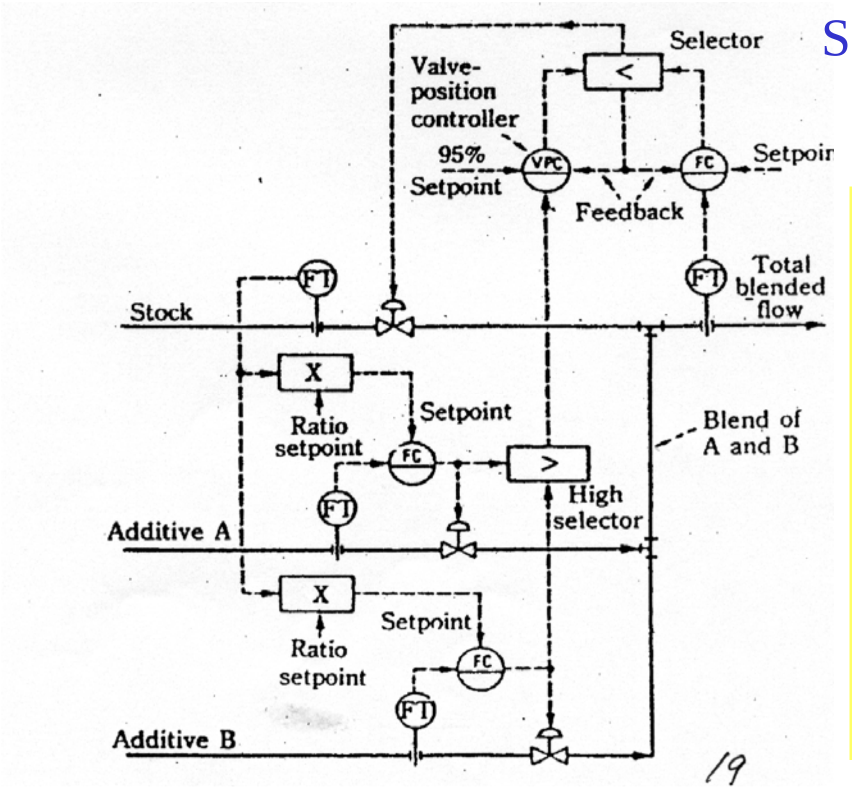


Example 1: Blending System Control



- Control r_A and r_B.
- Control q if possible.
- Flowrates of additives are limited.

Classical Solution



MPC: Solve at each time k

p=Size of prediction window

$$\min_{\substack{u_1(j),u_2(j),u_3(j)\\ j=k,\cdots,k+p-1}} \sum_{i=1}^{p} \Big(r_A(k+i\,|\,k)-r_A{*}\Big)^2 + \Big(r_B(k+i\,|\,k)-r_B{*}\Big)^2 + \gamma\Big(q(k+i\,|\,k)-q{*}\Big)^2$$

$$(u_i)_{\min} \le u_i(j) \le (u_i)_{\max},\ i=1,\cdots,3,$$

$$\gamma << 1$$

Advantages of MPC over Traditional APC

- Integrated solution
  - Automatic constraint handling
  - Feedforward / feedback control
  - No need for decoupler or delay compensation
- Efficient Utilization of degrees of freedom
  - Can handle nonsquare systems (e.g., more MVs and CVs)
  - Assignable priorities, ideal settling values for MVs
- Consistent, systematic methodology
- Realized benefits
  - Higher on-line times
  - Cheaper implementation
  - Easier maintenance

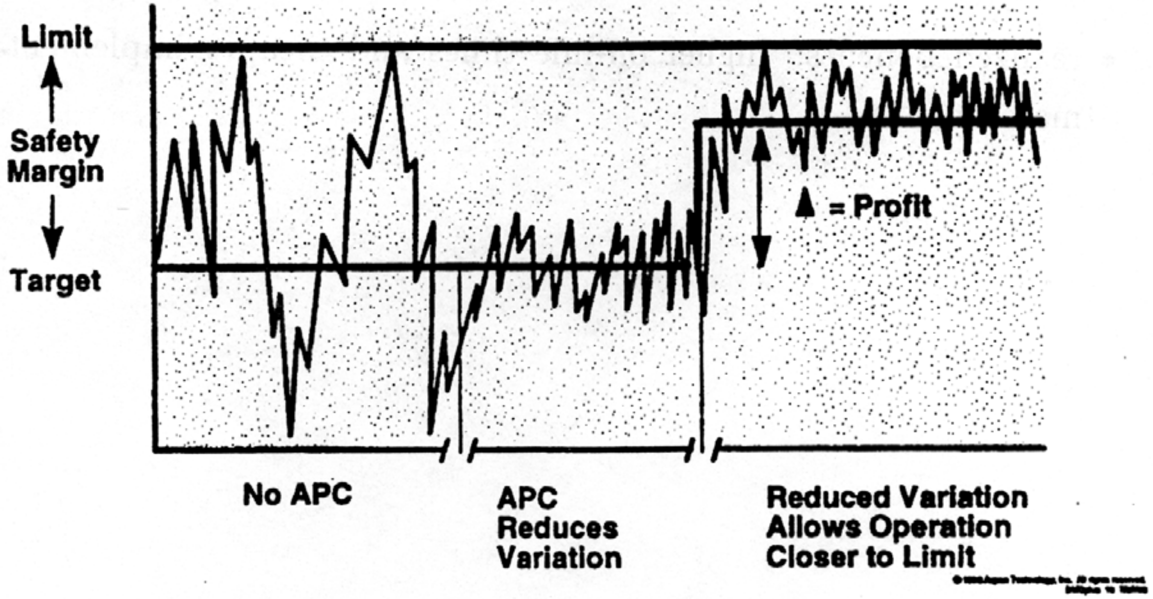
Reason for Popularity(2)

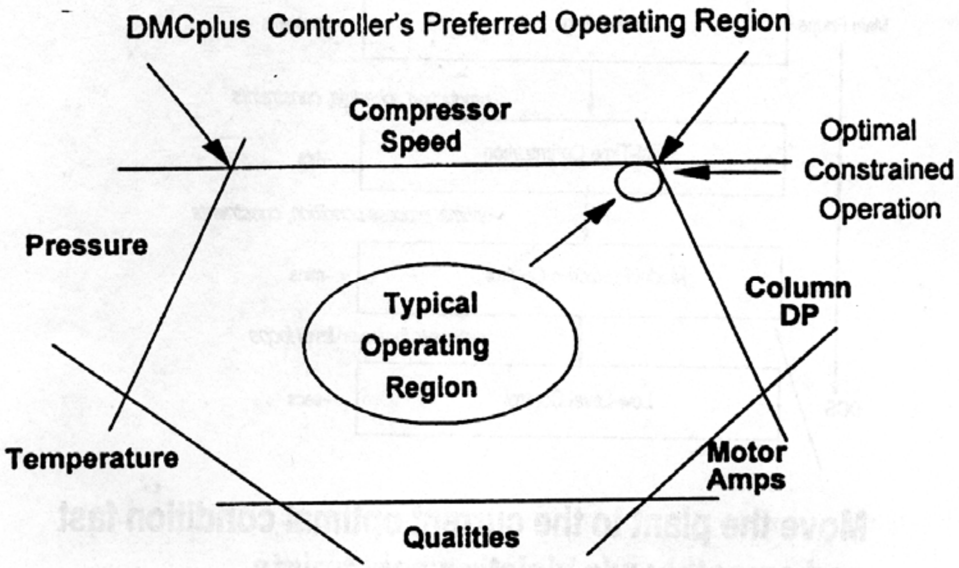
- Emerging popularity of **on-line optimization**
- Process optimization and control are often conflicting objectives
  - Optimization pushes the process to the boundary of constraints.
  - Quality of control determines how close one can push the process to the boundary.
- Implications for process control
  - High performance control is needed to realize on-line optimization.
  - Constraint handling is a must.
  - The appropriate tradeoff between optimization and control is time-varying and is best handled within a single framework

↓

Model Predictive Control

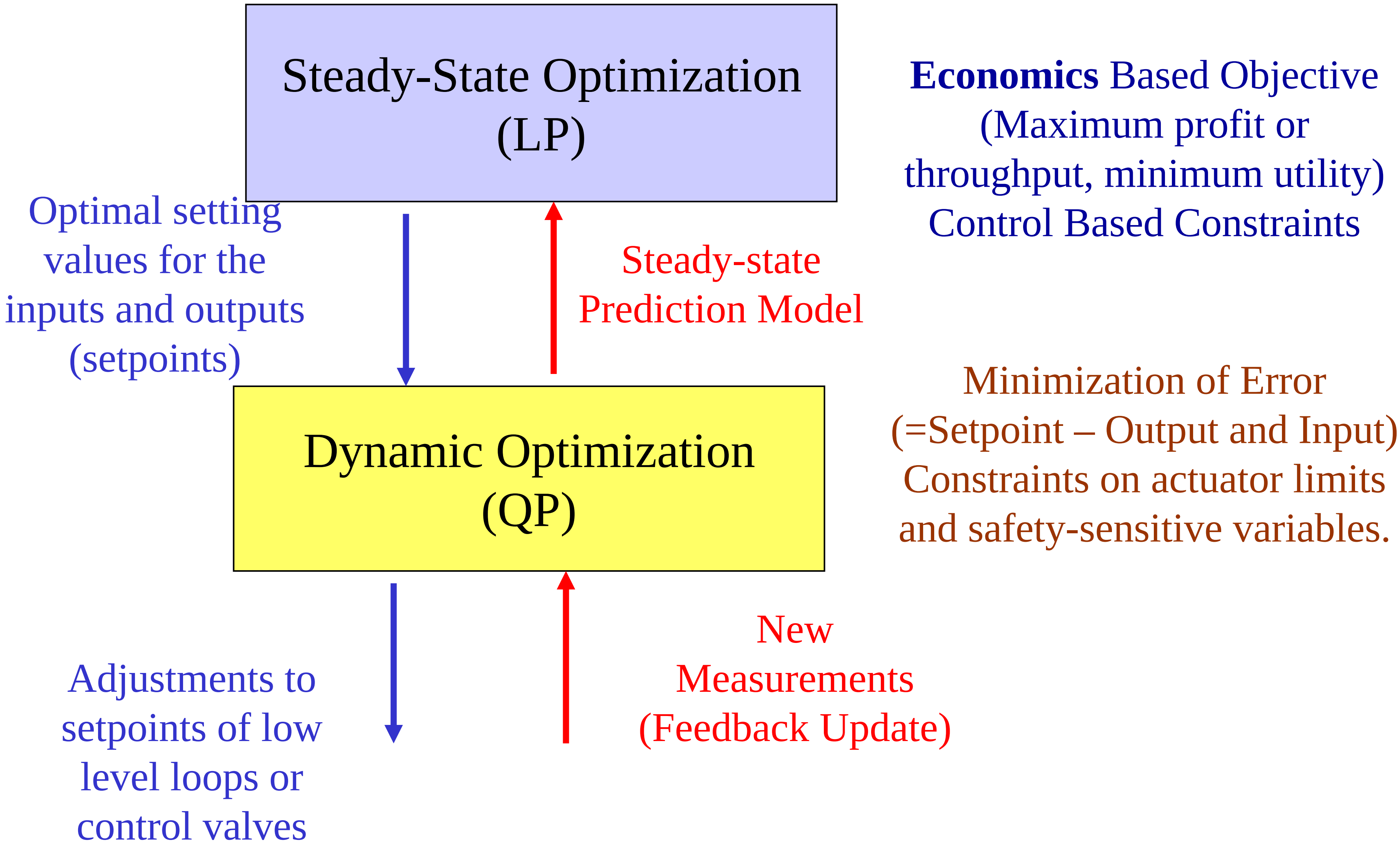
Conflict / Synergy Between Optimization and Control



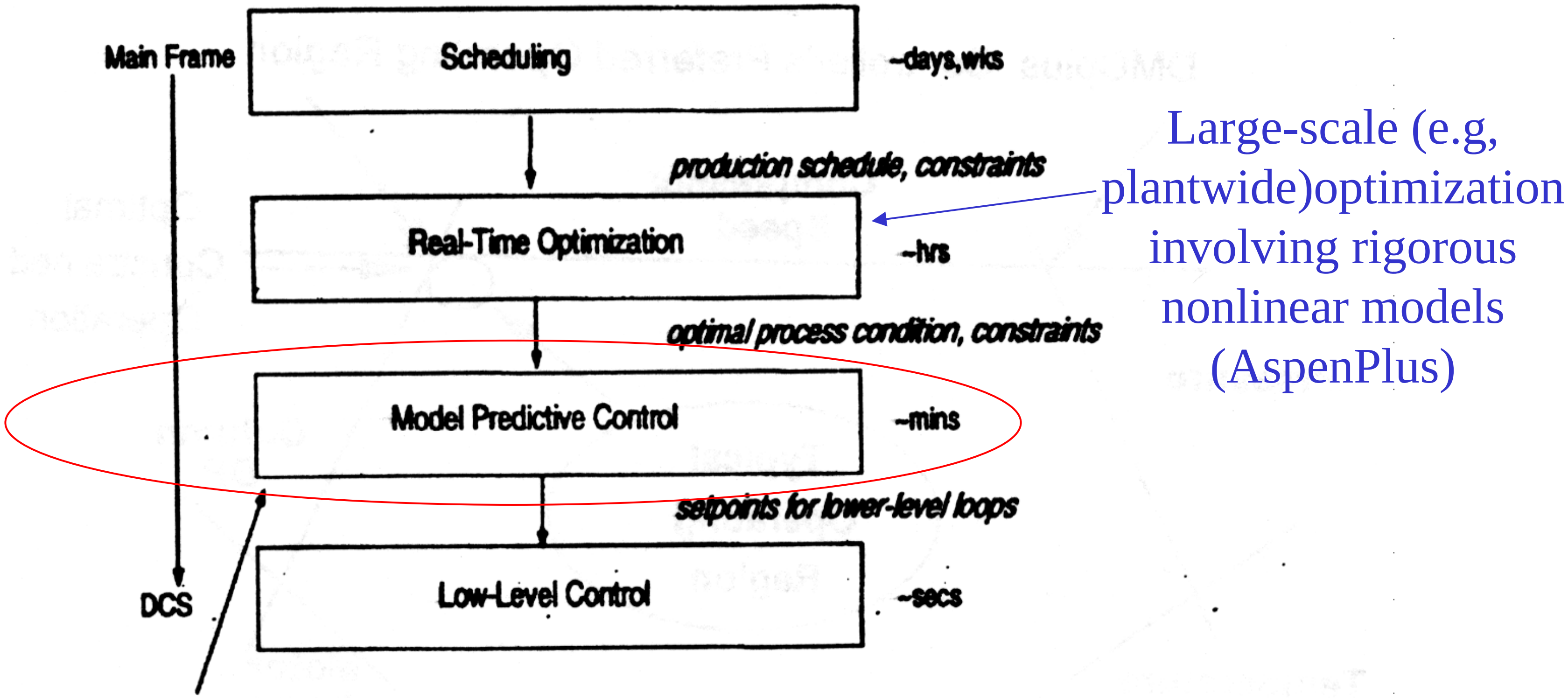


Acknowledgment:
Aspen Technology

Bi-Level Optimization Used in MPC



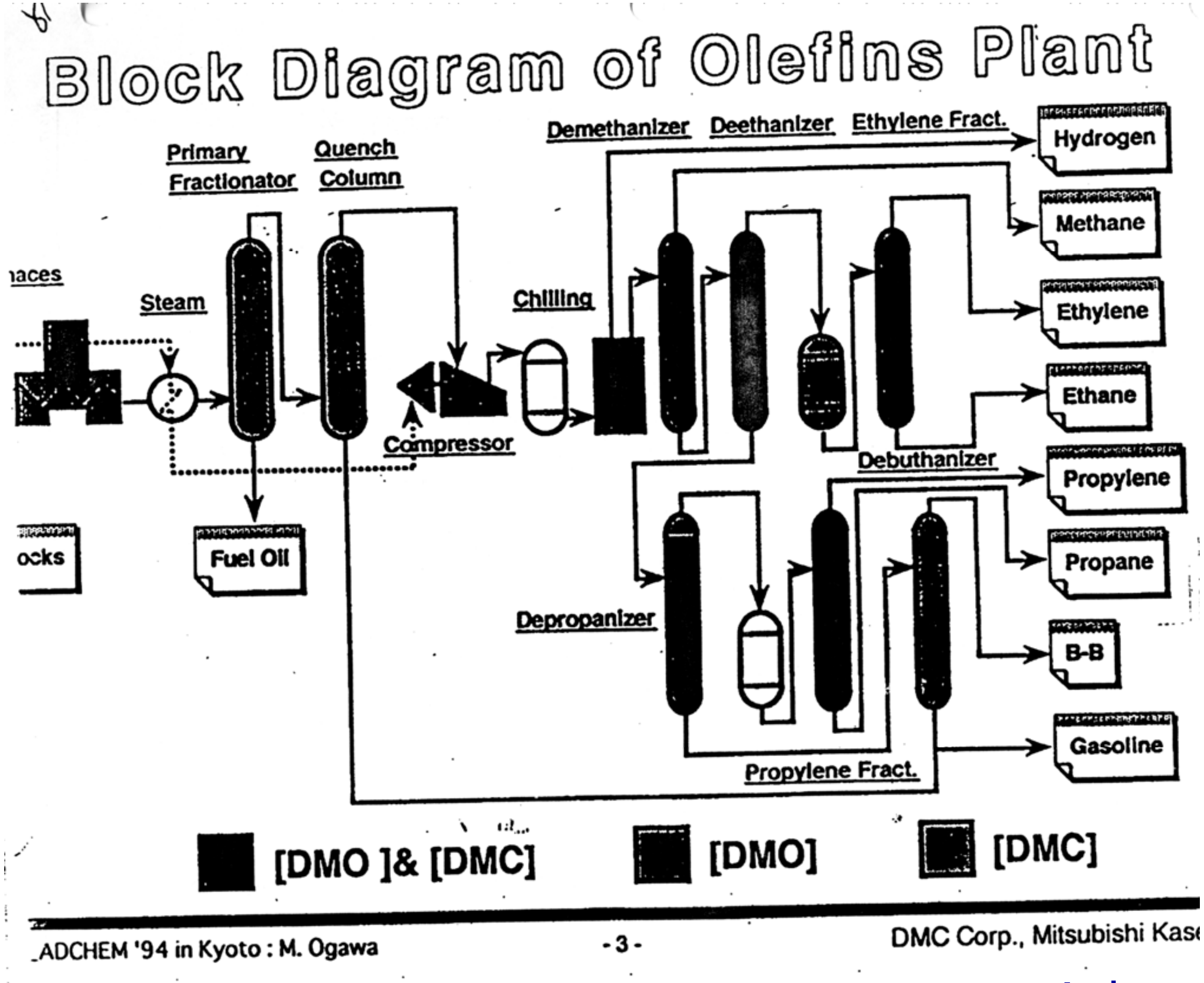
New Operational Hierarchy and Role of MPC



Move the plant to the current optimal condition fast and smoothly w/o violating constraints

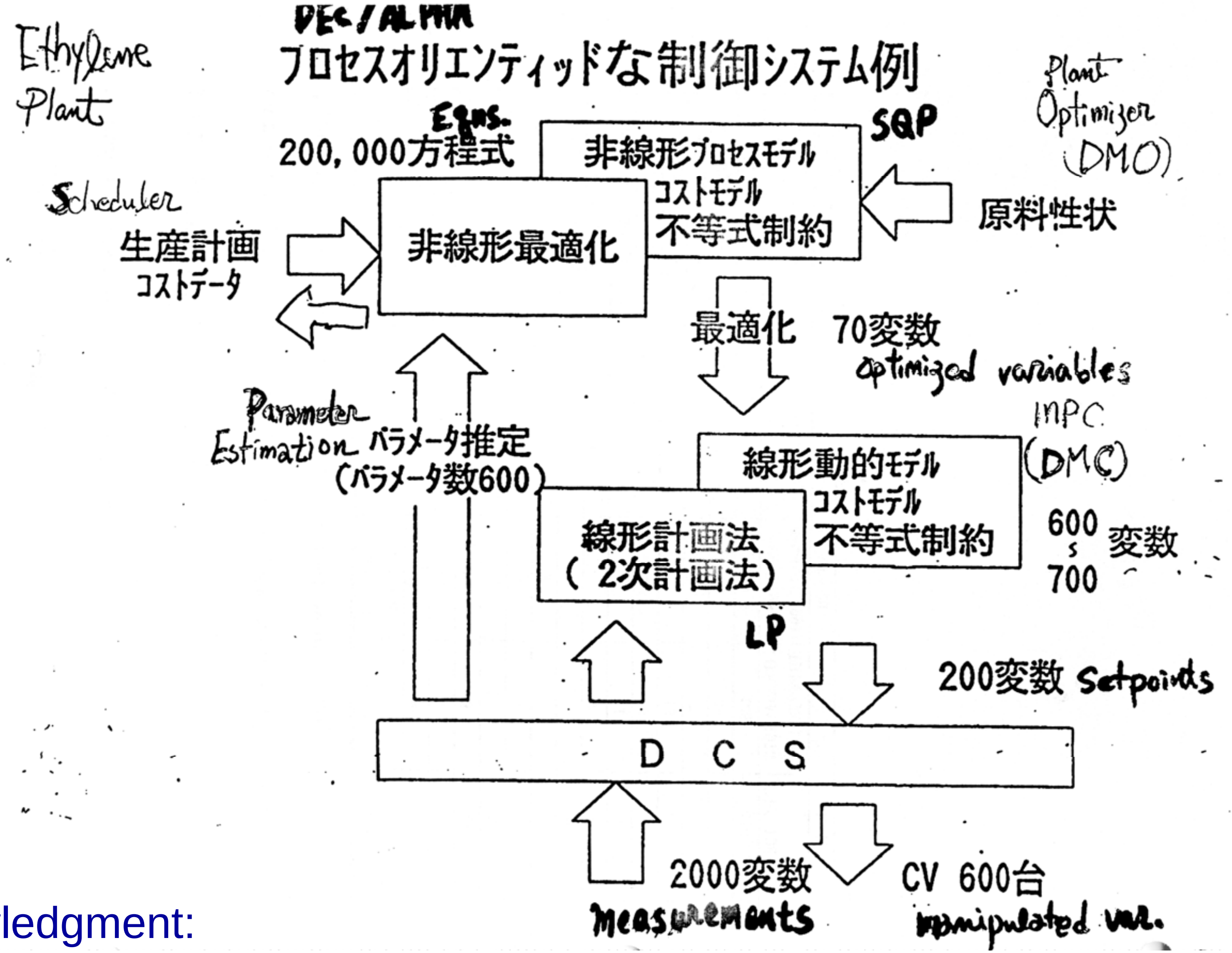
Local optimization + control

An Exemplary Application(1)



Acknowledgment:
Mitsubishi Chemicals

An Exemplary Application(2)



Acknowledgment:
Mitsubishi Chemicals

Linear MPC

Popular Linear Model Structures

- Finite Impulse Response Model

$$y(k) = h_1 u(k-1) + \cdots + h_N u(k-m)$$

- Truncated Step Response Model

$$x(k+1) = \underbrace{M_1}_{shift} x(k) + \underbrace{S}_{step\,response} \Delta u(k)$$

- Transfer Function Model

$$y(k) = a_1 y(k-1) + \cdots + a_n y(k-n) + b_1 u(k-1) + \cdots + b_m u(k-m) \quad \Rightarrow$$

$$G(q) = \frac{b_1 q^{-1} + \cdots + b_m q^{-m}}{1 - a_1 q^{-1} - \cdots - a_n q^{-n}}$$

- State Space Model

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

Key: Prediction Equation

- **Step Response Model**

Predicted future output samples

The "state" stored in dynamic memory

Future input moves (to be decided)

$$\begin{bmatrix} y_{k+1|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix} = \begin{bmatrix} y_1(k) \\ \vdots \\ y_p(k) \end{bmatrix} + \Omega^u \begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}$$

$$\Omega^d \begin{bmatrix} \Delta d(k) \\ \vdots \\ \Delta d(k+p-1) \end{bmatrix} + \begin{bmatrix} e(k) \\ \vdots \\ e(k) \end{bmatrix}$$

Dynamic Matrices (made of step response coefficients)

Feedback Error Correction

$$e(k) = y_m(k) - y(k)$$

Feedforward term: new measurement (Assume Δd(k+1)=…= Δ d(k+p-1)=0)

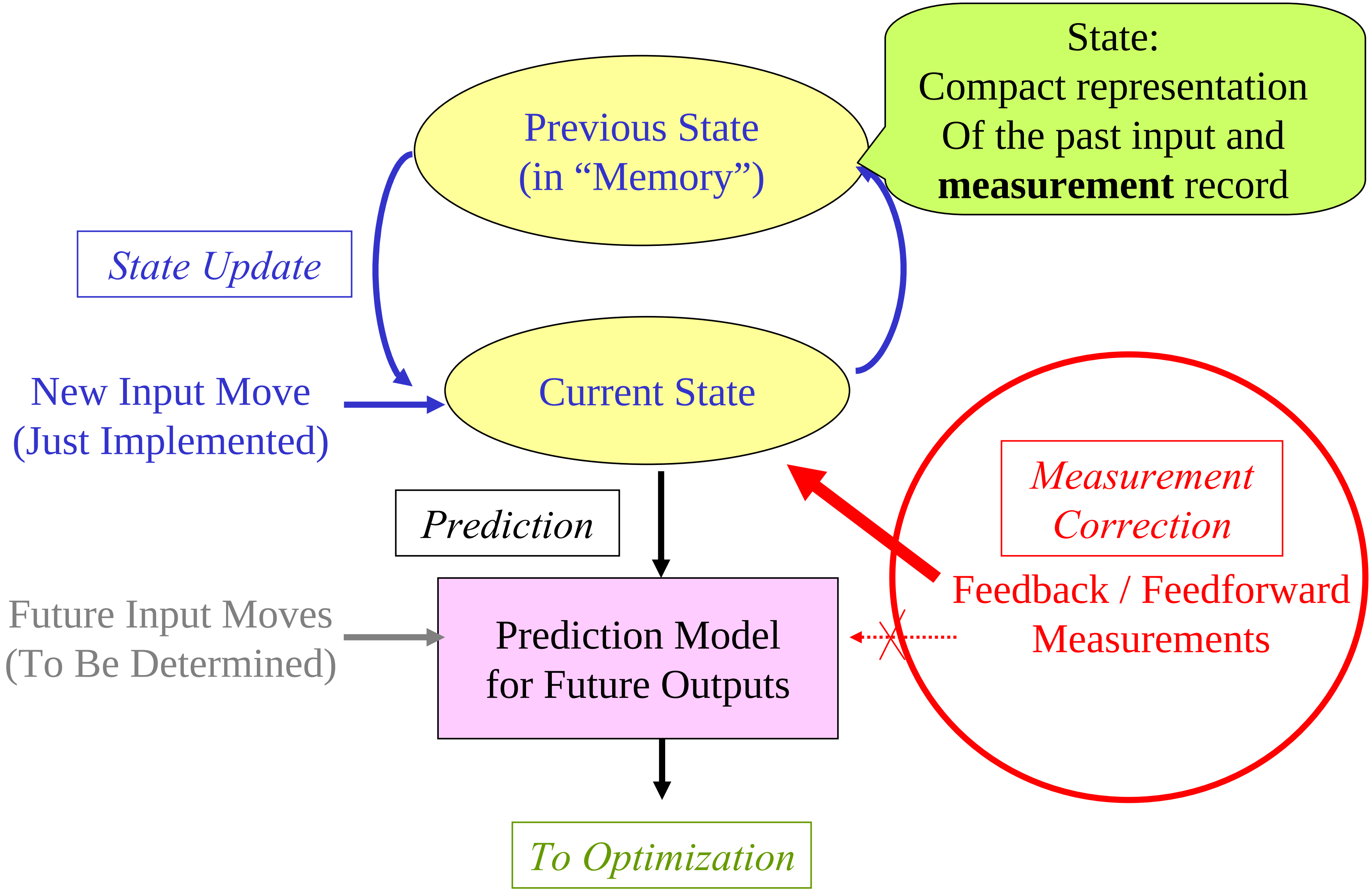
Prediction Equation: General

- Regardless of model form, one gets the prediction equation in the form of

$$\underbrace{\begin{bmatrix} y_{k+1|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix}}_{Y(k)} = \underbrace{L^x x(k) + L^d \Delta d(k) + L^e e(k)}_{known \equiv b(k)} + L^u \underbrace{\begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}}_{\Delta U(k)}$$

- Assumptions
  - Measured DV (d) remains constant at the current value of d(k)
  - Model prediction error (e) remains constant at the current value of e(k)

Measurement Correction of State



State Update Equation

$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma^u \Delta u(k) + \Gamma^d \Delta d(k)$$

$$+ K(y_m(k) - \Xi\hat{x}(k))$$

- K is the update gain matrix that can be found in various ways
  - *Pole placement*: Not so effective with systems with many states (most chemical processes)
  - *Kalman filtering*: Requires a **stochastic** model of form

$$x(k+1) = \Phi x(k) + \Gamma^u \Delta u(k) + \Gamma^d \Delta d(k) + w(k)$$

$$y(k) = \Xi x(k) + \nu(k)$$

White noises of known covariances
Effect of unmeasured disturbances and noise

Prediction Equation

Contains past feedback measurement corrections

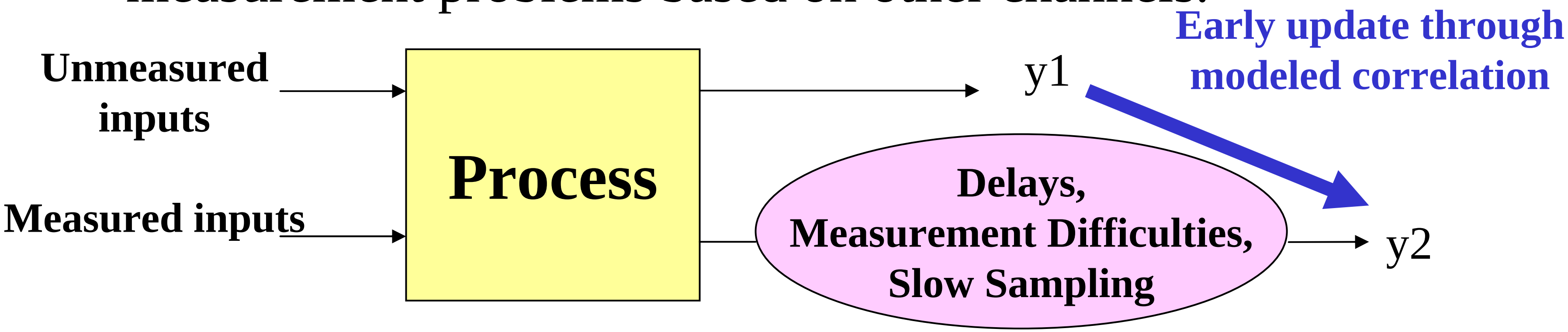
$$\begin{bmatrix} y_{k+1|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix} = \begin{bmatrix} \Xi\Phi \\ \vdots \\ \Xi\Phi^{p} \end{bmatrix} \hat{x}(k) + \Omega^{u} \begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}$$

$$\Omega^{d} \begin{bmatrix} \Delta d(k) \\ \vdots \\ \Delta d(k+p-1) \end{bmatrix} + \begin{bmatrix} e(k) \\ \vdots \\ e(k) \end{bmatrix}$$

Additional measurement correction NOT needed here!

What Are the Advantages of Using a State Estimator (Observer)?

- Can handle **unstable** processes
  - Integrating processes, run-away processes
- **Cross-channel** update
  - More effective update of output channels with delays or measurement problems based on other channels.



- Systematic handling of **multi-rate** measurements
- Optimal **extrapolation of output error** and **filtering of noise** (based on the given stochastic system model)

Objective Function

- Minimization Function: Quadratic cost (as in DMC)

$$V(k) = \sum_{i=1}^{p} (y_{k+i|k} - y^*)^T \Lambda^y (y_{k+i|k} - y^*) + \sum_{i=0}^{m-1} \Delta u^T (k+i) \Lambda^u \Delta u(k+i)$$

  - Consider only m input moves by assuming $\Delta u(k+j)=0$ for $j \geq m$
  - Penalize the tracking error as well as the magnitudes of adjustments

- V(k) is a quadratic function of $\Delta u(k+j)$, $j=0,\ldots,m-1$

Objective Function

degrees of freedom

$$V(k)=\sum_{i=1}^{p}(y_{k+i|k}-y^*)^T\Lambda^y(y_{k+i|k}-y^*)+\sum_{i=0}^{m-1}\Delta u^T(k+i)\Lambda^u\Delta u(k+i)$$

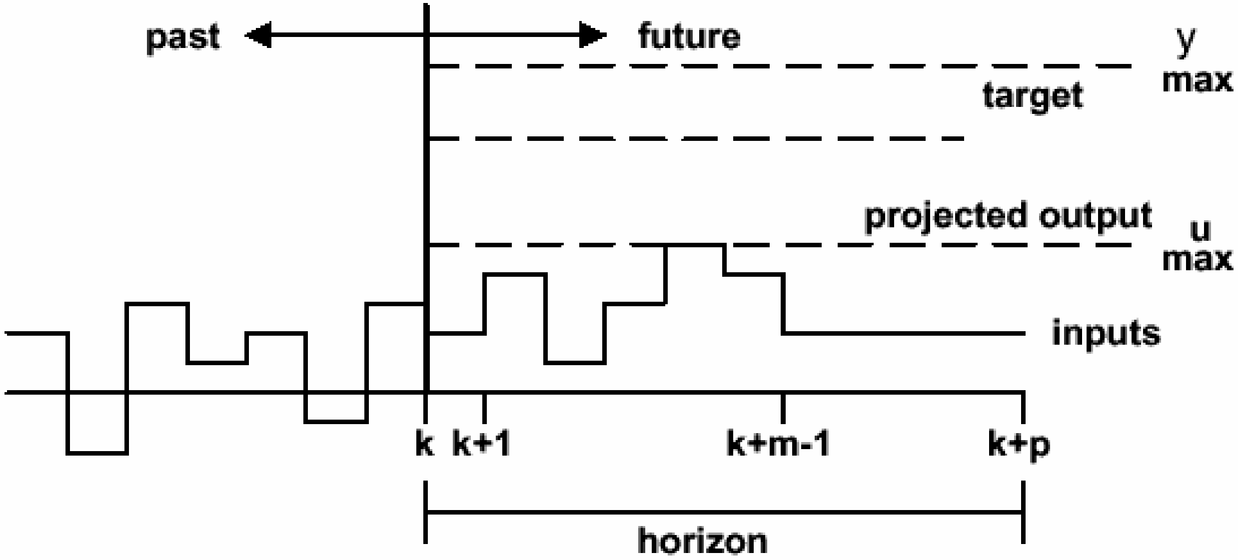
$$V(k)=\left(Y(k)-Y^*\right)^{\mathrm{T}}\mathrm{diag}(\Lambda^y)\left(Y(k)-Y^*\right)+\Delta U_m^T(k)\mathrm{diag}(\Lambda^u)\Delta U_m(k)$$

Substitute

$$\underbrace{\begin{bmatrix} y_{k+1|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix}}_{Y(k)}=\underbrace{L^x x(k)+L^d\Delta d(k)+L^e e(k)}_{known\equiv b(k)}+L^u\underbrace{\begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}}_{\Delta U(k)}$$

$$V(k)=\Delta U_m^T(k)H\Delta U_m(k)+g^T(k)\Delta U_m(k)+c(k)$$

Constraints



$$u_{\text{min}} \le u(k+\ell|k) \le u_{\text{max}}$$
$$|\Delta u(k+\ell|k)| \le \Delta u_{\text{max}}, \qquad \ell = 0, \cdots, m-1$$
$$y_{\text{min}} \le y(k+j|k) \le y_{\text{max}}, \quad j = 1, \cdots, p$$

Substitute the prediction equation and rearrange to

$$C\Delta U_m(k) \ge h(k)$$

Optimization Problem

- Quadratic Program

$$\min_{\Delta U_m(k)} \Delta U_m^T(k) H \Delta U_m(k) + g^T(k) \Delta U_m(k)$$

$$\text{such that } C \Delta U_m(k) \geq h(k)$$

- Unconstrained Solution

$$\Delta U_m(k) = -\frac{1}{2} H^{-1} g(k)$$

- Constrained Solution
  - Must be solved numerically.

Quadratic Program

- Minimization of a quadratic function subject to linear constraints.
- Convex and therefore *fundamentally tractable*.
- Solution methods
  - Active set method: Determination of the active set of constraints on the basis of the KKT condition.
  - Interior point method: Use of barrier function to “trap” the solution inside the feasible region, Newton iteration
- Solvers
  - Off-the-shelf software, e.g., QPSOL
  - Customization is desirable for large-scale problems.

Bi-Level Optimization

Steady-State Optimization (Linear Program)

$$\min_{u_s(k)} L(y_{\infty|k}, u_s(k))$$

$$C_s \begin{bmatrix} y_{\infty|k} \\ u_s(k) \end{bmatrix} \geq c_s(k)$$

$$u_s(k) = u(k-1) + \Delta u(k) + \cdots + \Delta u(k+m-1)$$

$$y_{\infty|k} = b_s(k) + L_s \Delta u_s(k)$$

Steady-State Prediction Eqn.

Optimal Setting Values (setpoints)

$$y^*_{\infty|k}, u^*_s(k)$$

State
Feedforward Measurement
Feedback Error

Dynamic Prediction Eqn.

To Dynamic Optimization (Quadratic Program)

Stability

Classical Optimal Control - LQR

- Quadratic objective

$$\sum_{i=0}^{p} x_i^T Q x_i + \sum_{i=0}^{m-1} u_i^T R u_i$$

Linear State Space System Model

$$x_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k$$

- Fairly general formulation:
  - State regulation, Output regulation, Setpoint tracking
- Unconstrained ∞ horizon problem has an analytical solution. $\rightarrow$ Linear state feedback law (Kalman's LQR)
- Stability guaranteed for <u>stabilizable</u> system
- Solution is smooth with respect to the parameters
- BUT, presence of inequality constraints $\rightarrow$ no analytical solution via Riccati equation.

Why Has Stability Analysis of MPC Been Difficult?

- MPC=Nonlinear state feedback control law
- Implicitly defined by an optimization
  - No explicit expression for the MPC control law
- Use of an observer
  - Lack of separation principle

Use of ∞ Prediction Horizon – Why?

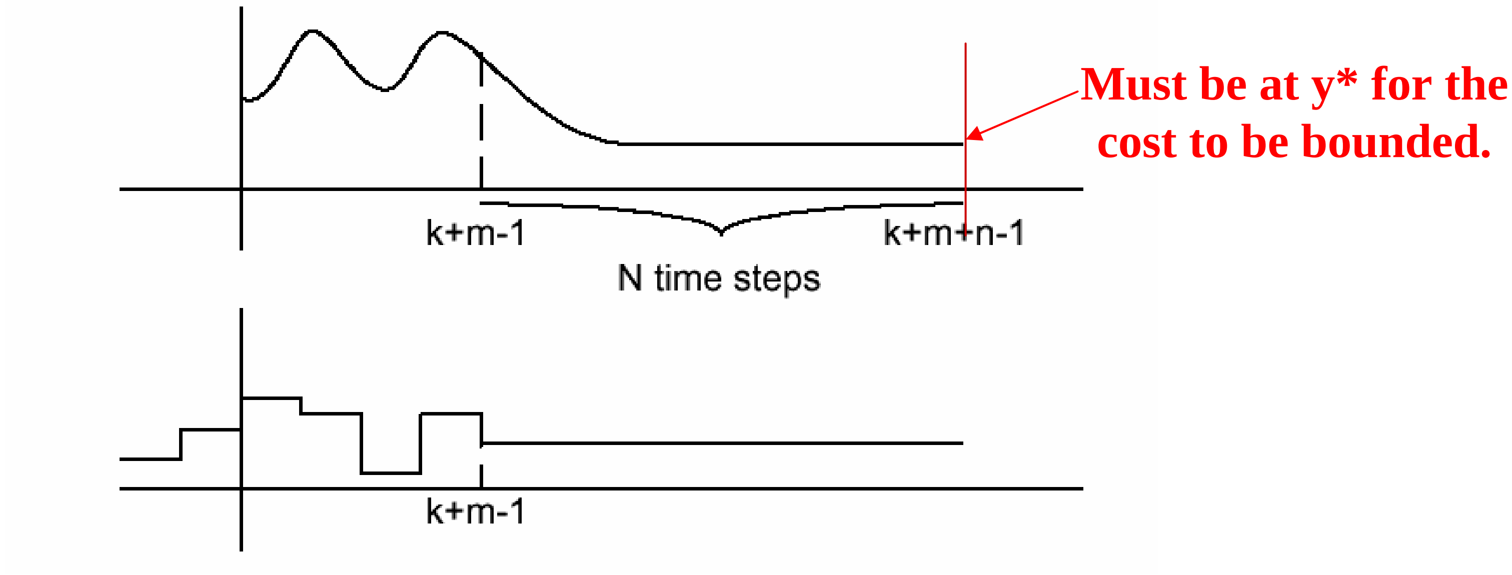
- Stability guarantee
  - The optimal cost function can be shown to be the control Lyapunov function.
- Less parameters to tune
- More consistent, intuitive effect of weight parameters
- Close connection with the classical optimal control methods, e.g., LQG control

Step Response Model Case

$$V(k) = \sum_{i=1}^{\infty} (y_{k+i|k} - y^*)^T \Lambda^y (y_{k+i|k} - y^*) + \sum_{i=0}^{m-1} \Delta u^T(k+i) \Lambda^u \Delta u(k+i)$$

$$V(k) = \sum_{i=1}^{m+n-1} (y_{k+i|k} - y^*)^T \Lambda^y (y_{k+i|k} - y^*) + \sum_{i=0}^{m-1} \Delta u^T(k+i) \Lambda^u \Delta u(k+i)$$

with extra constraint $y_{k+m+n-1|k} = y^*$



Additional Comments

- Previously, we assumed **finite** settling time.
- Can be generalized to general state-space models
  - More complicated procedure to turn the ∞-horizon problem into a finite horizon problem
  - Requires solving a Lyapunov equation to get the terminal cost matrix
  - Also, must make sure that output constraints will be satisfied beyond the finite horizon $\rightarrow$ construction of an output admissible set.
- Use of a *sufficiently large* horizon ($p \approx m+$ the settling time) should have a similar effect.
- Can we always satisfy the settling constraint?
  - $y=y^*$ may not be feasible due to input constraints or insufficient m. $\rightarrow$ use two-level approach.

Two-Level Optimization

Steady-State Optimization
(Linear Program or Quadratic Program)

Optimal Setting Values (setpoints)

$$y^*_{\infty|k}, u^*_s(k)$$

Dynamic Optimization (∞-horizon MPC)

Constraint $y_{k+m+n-1|k} = y^*_{\infty|k}$ is guaranteed to be feasible.

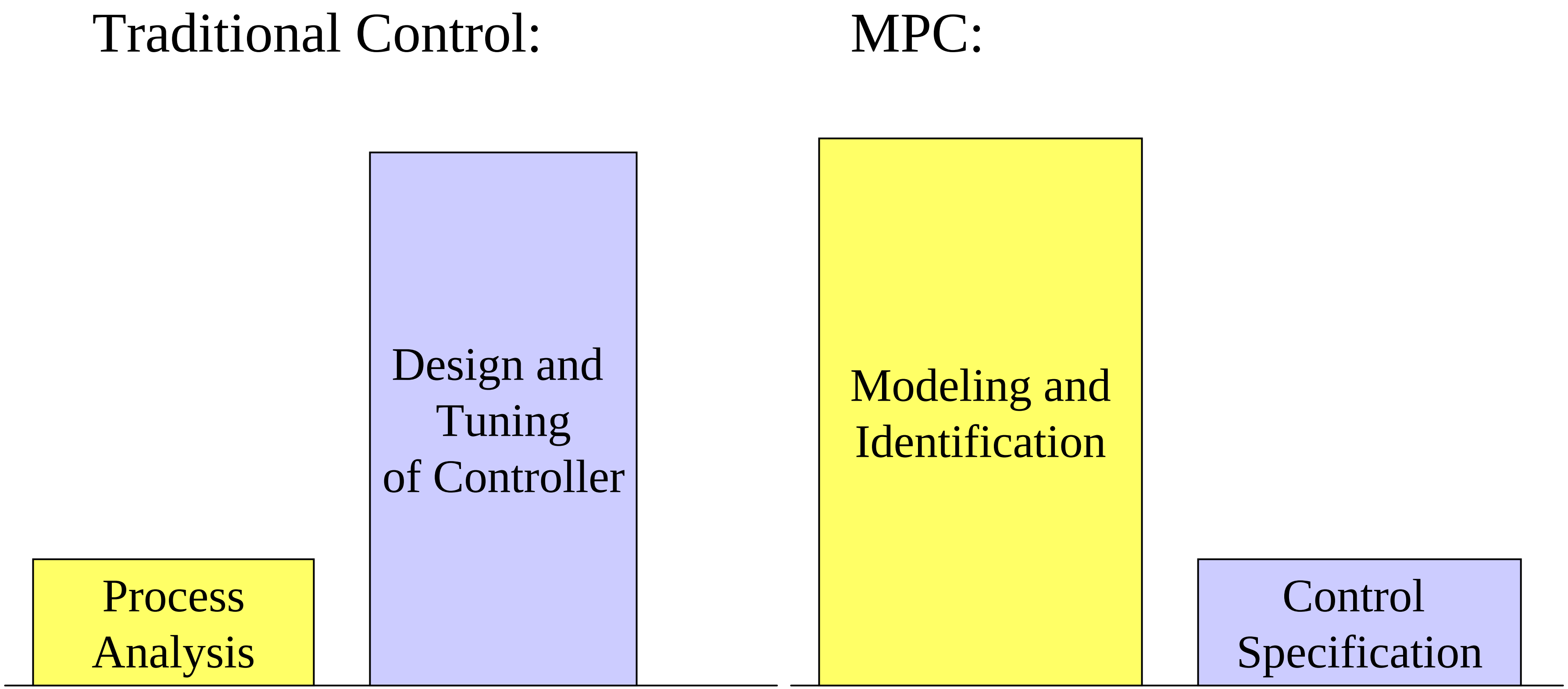
Constraint $\Delta u(k) + \cdots + \Delta u(k+m\text{-}1) = \Delta u^*_s \rightarrow y_{k+m+n\text{-}1|k} = y^*_{\infty|k}$.

Process Identification

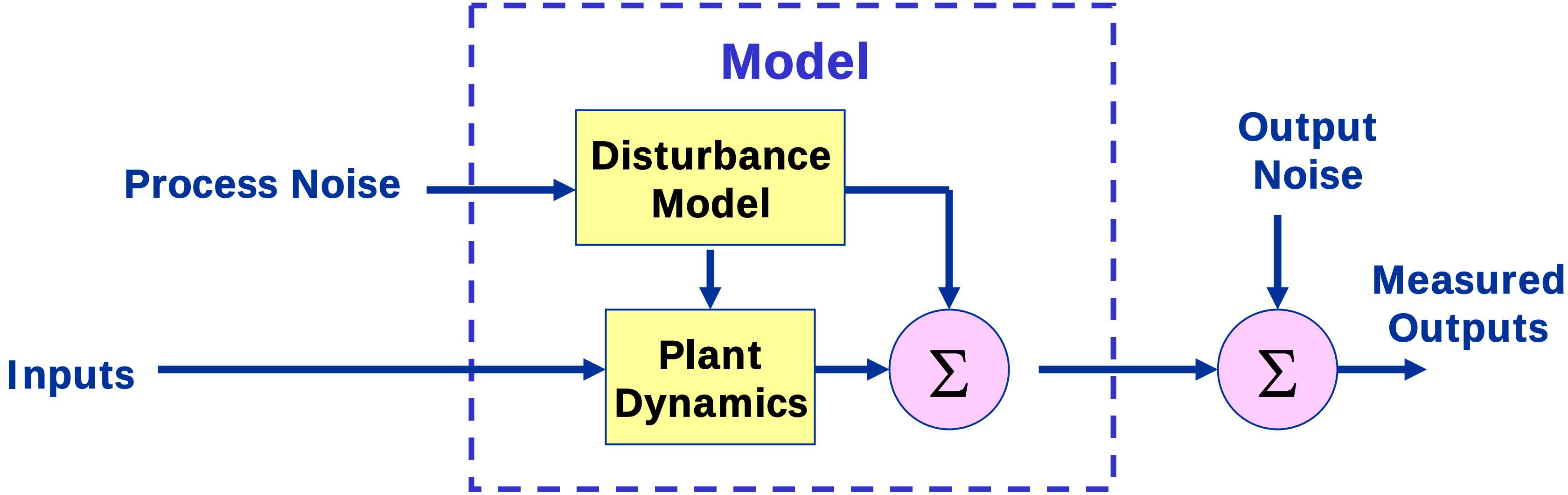
Importance of Modeling

- Almost all models used in MPC are typically empirical models "identified" through **plant tests** rather than first-principles models.
  - Step responses, pulse responses from plant tests.
  - Transfer function models fitted to plant test data.
- Up to **80% of time and expense** involved in designing and installing a MPC is attributed to modeling / system identification. $\rightarrow$ should be improved.
- Keep in mind that obtained models are **imperfect** (both in terms of structure and parameters).
  - Importance of <u>feedback</u> update of the model.
  - Penalize excessive input movements.

Design Effort



Model Structure (1)



- I/O Model

White noise sequence

$$y(k) = \underbrace{G(q)u(k)}_{\text{effect of inputs}} + \underbrace{H(q)\,e(k)}_{\text{effect of disturbances, noise}}$$

Models auto- and cross-correlations of the residual (not physical cause-effect)

Assume w.l.g. that H(0)=1

SISO I/O Model Structure(1)

- FIR (Past inputs only)

$$y(k) = h_1 u(k-1) + \cdots + h_N u(k-m) + e(k)$$

- ARX (Past inputs and outputs: "Equation Error")

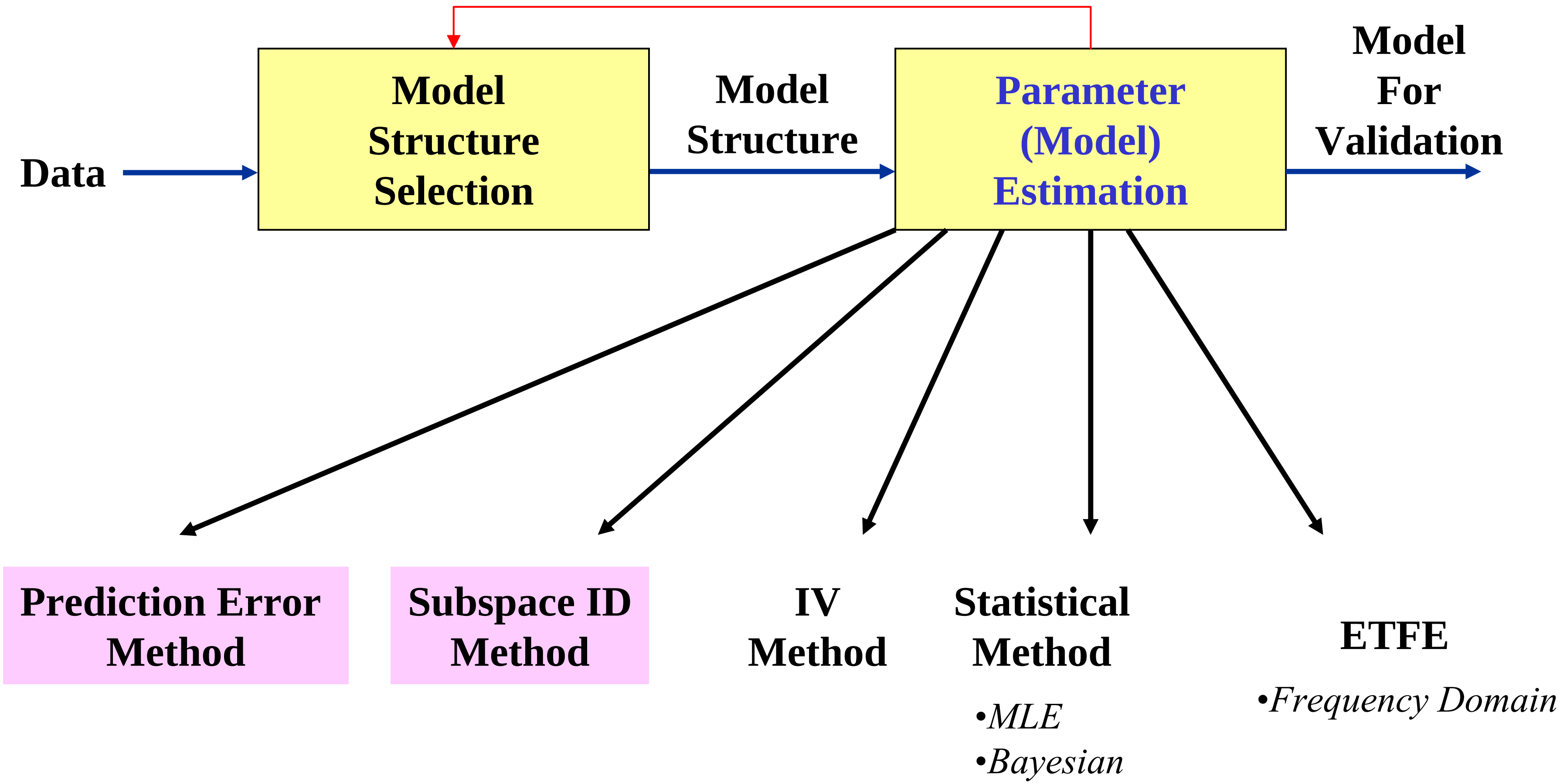
$$y(k) = a_1 y(k-1) + \cdots + a_n y(k-n) + b_1 u(k-1) + \cdots + b_m u(k-m) + e(k)$$

- ARMAX (Moving average of the noise term)

$$\begin{aligned} y(k) &= a_1 y(k-1) + \cdots + a_n y(k-n) + b_1 u(k-1) + \cdots + b_m u(k-m) \\ &\quad + e(k) + c_1 e(k-1) + \cdots + c_n e(k-n) \end{aligned}$$

- Output Error (OE), Box-Jenkins (BJ), etc.

Overview



Data
Model Structure Selection
Model Structure
Parameter (Model) Estimation
Model For Validation
Prediction Error Method
Subspace ID Method
IV Method
Statistical Method
•MLE
•Bayesian
ETFE
•Frequency Domain

Prediction Error Method

- Predominant method at current time
- Developed by Ljung and coworkers
- Flexible
  - Can be applied to any model structure
  - Can be used in recursive form
- Well developed theories and software tools
  - Book by Ljung, System ID Toolbox for MATLAB
- Computational complexity depends on the model structure
  - ARX, FIR $\rightarrow$ Linear least squares
  - ARMAX, OE, BJ $\rightarrow$ Nonlinear optimization

Prediction Error Method

- Put the model in the predictor form

$$y(k) = G(q,\theta)u(k) + H(q,\theta)e(k) \quad \rightarrow$$

$$y_{k|k-1} = G(q,\theta)u(k) + \underbrace{\left(I - H^{-1}(q,\theta)\right)}_{\text{Contains at least 1 delay}}\left(y(k) - G(q,\theta)u(k)\right)$$

$$e(k) = y(k) - y_{k|k-1} = H^{-1}(q,\theta)\left(y(k) - G(q,\theta)u(k)\right)$$

- Choose the parameter values to minimize the sum of the prediction error for the given N data points.

$$\min_{\theta}\left\{\frac{1}{N}\sum_{k=1}^{N}\left\|e(k)\right\|_2^2\right\} \qquad e(k) = H^{-1}(q,\theta)\left(y(k) - G(q,\theta)u(k)\right)$$

  - ARX, FIR $\rightarrow$ Linear least squares,
  - ARMAX, OE, BJ $\rightarrow$ Nonlinear least squares
- Not easy to use for identifying *multivariable* models.

MIMO I/O Model Structure

- Inputs and outputs are vectors. Coefficients are matrices.
- For example, ARX model becomes

$$
\begin{aligned}
y(k) &= A_1 y(k-1) + \cdots + A_n y(k-n) \\
&\quad + B_1 u(k-1) + \cdots + B_m u(k-m) + e(k)
\end{aligned}
$$

A_i is an $n_y \times n_y$ matrix. B_i is an $n_y \times n_u$ matrix.

- *Identification* is very difficult.
  - Different sets of coefficient matrices giving exactly same G(q) and H(q) through pole/zero cancellations. $\rightarrow$ Problems in parameter estimation $\rightarrow$ Requires special parameterization to avoid problem.

State Space Model

- Deterministic

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k) + e(k)$$

⟺ **Output Error Structure**

- Combined Deterministic / Stochastic

$$x(k+1) = Ax(k) + Bu(k) + Ke(k)$$
$$y(k) = Cx(k) + e(k)$$

⟺ **ARMAX Structure**

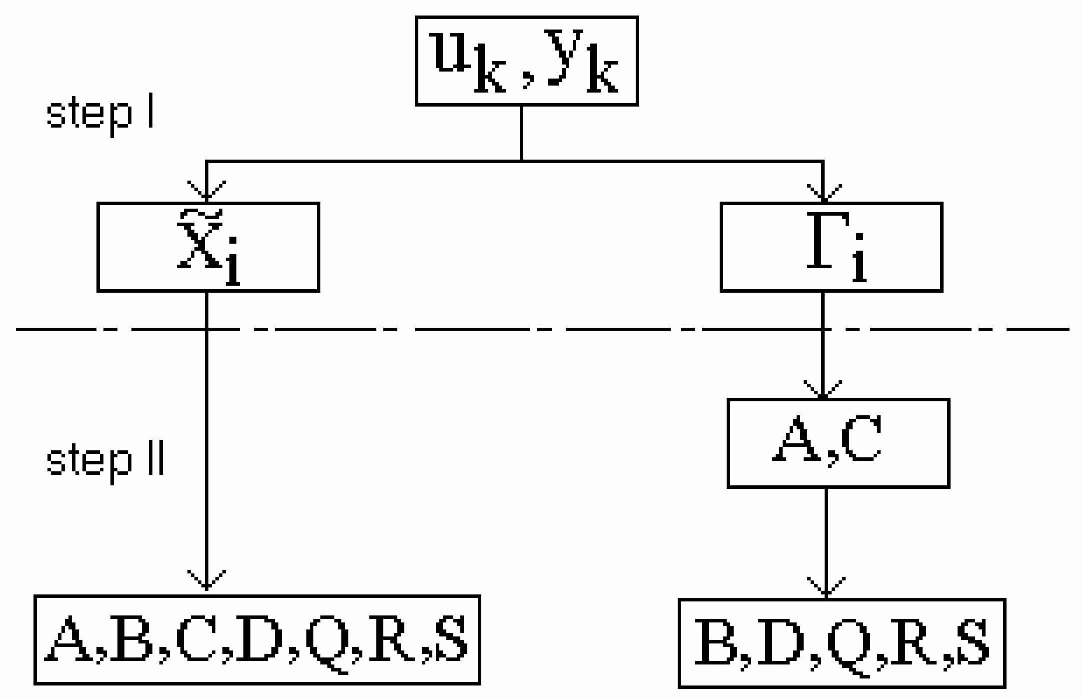
- *Identifiability* can be an issue here too
  - State coordinate transformation does not change the I/O relationship.

Subspace Method

- More recent development
- Dates back to the classical realization theories but rediscovered and extended by several people
- Identifies a state-space model
- Some theories and software tools
- Computationally simple
  - Non-iterative, linear algebra
- Good for identifying *multivariable* models.
  - No special parameterization is needed.
- Not optimal in any sense
- May need a lot of data for good results
- May be combined with PEM
  - Use SS method to obtain an initial guess for PEM.

SIM Procedures

SIM algorithms have two categories and contain two steps:



CVA, N4SID   MOESP, DSR

Acknowledgment:
Prof. Joe Qin

Use of Nonlinear Model

Difficulty (1)

$$\dot{x} = f(x,u,d)$$
$$y = g(x)$$

Discretization?

$$x(k+1) = F(x(k),u(k),d(k))$$
$$y(k) = g(x(k))$$

Orthogonal Collocation

$$y_{k+1|k} = g \circ F(x(k),u(k),d(k)) + e(k)$$
$$y_{k+2|k} = g \circ F\big(F(x(k),u(k),d(k)),u(k+1),d(k)\big) + e(k)$$
$$\vdots$$
$$y_{k+p|k} = g \circ F^{p}\big(x(k),u(k),\cdots u(k+p-1),d(k)\big) + e(k)$$

The prediction equation is nonlinear w.r.t. u(k), ……, u(k+p-1)

↓

Nonlinear Program (Not so nice!)

Difficulty (2)

State Estimation

$$\dot{x} = f(x,u,d) + w$$
$$y = g(x) + v$$

Extended Kalman Filtering

$$x(k+1) = \int_{k}^{k+1} f(x,u,d) + K(k)\big(y_m(k) - g(x(k))\big)$$

- Computationally more demanding steps, e.g., calculation of K at each time step.
- Based on linearization at each time step – not optimal, may not be stable.
- Best practical solution at the current time
- Promising alternative: Moving Horizon Estimation (requires solving NLP).
- Difficult to obtain with an appropriate stochastic system model (no ID technique)

Practical Algorithm

EKF

$x(k)$

Dynamic Matrix based on the linearized model at the <u>current state and input values.</u>

$$\begin{bmatrix} y_{k+1|k} \\ y_{k+2|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix} = \begin{bmatrix} \int_k^{k+1} f(x,u,d) \\ \int_k^{k+2} f(x,u,d) \\ \vdots \\ \int_k^{k+p} f(x,u,d) \end{bmatrix} + \underbrace{\Omega^u(k) \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}}_{\text{Linearized Effect of Future Input Adjustments}}$$

Model integration with constant input u=u(k-1) and d=d(k)

Linear prediction equaton

Linear or Quadratic Program

Additional Comments / Summary

- Some refinements to the “Practical Algorithm” are possible.
  - Use the previously calculated input trajectory (instead of the constant input) in the integration and linearization step.
  - Iterate between integration/linearization and control input calculation.
- “Full-blown” nonlinear MPC is still computationally prohibitive in most applications.
  - A lot of recent developments in SQP solver.
- Some promising directions
  - Tabulation
  - Simulation based approach (Approximate dynamic programming)

Remaining Challenges

- Efficient identification of **control-relevant** models
- Managing the sometimes exorbitant on-line **computational load**
  - Nonlinear models → Nonlinear Programs (NLP)
  - Hybrid system models (continuous dynamics + discrete events or switches, e.g., pressure swing adsorption) → Mixed Integer Programs (NLP)
  - Difficult to solve these reliably on-line for large-scale problems.
- How do we design model, estimator (of model parameters and state), and optimization algorithm as **an integrated system** - that are simultaneously optimized - rather than disparate components?
- Long-term **performance** of MPC.

Conclusion

- MPC is the established advanced multivariable control technique for the process industry. It is already an indispensable tool and its importance is continuing to grow.
- It can be formulated to perform some economic optimization and can also be interfaced with a larger-scale (e.g., plantwide) optimization scheme.
- Obtaining an accurate model and having reliable sensors for key parameters are key bottlenecks.
- A number of challenges remain to improve its use and performance.

Some References to Start With.

- Morari, M. and J. H. Lee, “Model Predictive Control : Past, Present and Future,” *Computers and Chemical Engineering*, 23, pp. 667-682 1999.

- Lee, J. H. and B. Cooley, “Recent Advances in Model Predictive Control and Other Related Areas,” *5th International Conference on Chemical Process Control*, edited by J. Kantor, C. Garcia, and B. Carnahan, *AIChE Symposium Series*, Vol. 91, No. 316, pp. 201-216, 1997.

- Qin, S. J. and T. Badgwell, “A survey of industrial model predictive control technology,” *Control Engineering Practice*, 11, 733–764, 2003.

- Rawlings, J. and K. Muske, “The Stability of Constrained Receding Horizon Control,” *IEEE Trans. Auto. Control*, 38, 1993

Extra Slides

State-Space Model (1)

$$z(k+1) = Az(k) + B^u u(k) + B^d d(k)$$
$$y(k) = Cz(k)$$

Discretization

State-Space Realization

$$y(z) = G^u(z)u(z) + G^d(z)d(z)$$

or

$$y(s) = G(s)u(s) + G^d(s)d(s)$$

State-Space Identification

$$\dot{z} = \tilde{A}z + \tilde{B}^u u + \tilde{B}^d d$$
$$y = \tilde{C}z$$

Linearization

I/O model Identification

$$\dot{z} = f(z, u, d)$$
$$y = g(z)$$

$$\{y(i), u(i), i = 1, \cdots N\}$$

Fundamental Model

Test Data

State-SpaceModel (2)

$$\begin{pmatrix} z(k+1) = Az(k) + B^u u(k) + B^d d(k) \\ y(k+1) = Cz(k+1) \end{pmatrix}$$

$$\begin{pmatrix} z(k) = Az(k-1) + B^u u(k-1) + B^d d(k-1) \\ y(k) = Cz(k) \end{pmatrix}$$

-

$$\Delta z(k+1) = A\Delta z(k) + B^u \Delta u(k) + B^d \Delta d(k)$$

$$\Delta y(k+1) = C\Delta z(k+1) \quad \rightarrow$$

$$y(k+1) = y(k) + C\left(A\Delta z(k) + B^u \Delta u(k) + B^d \Delta d(k)\right)$$

$$\begin{bmatrix} \Delta z(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \begin{bmatrix} \Delta z(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B^u \\ CB^u \end{bmatrix} \Delta u(k) + \begin{bmatrix} B^d \\ CB^d \end{bmatrix} \Delta d(k)$$

$$y(k) = \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta z(k) \\ y(k) \end{bmatrix}$$

State Update

$$x(k+1) = \Phi x(k) + \Gamma^u \Delta u(k) + \Gamma^d \Delta d(k)$$

$$y(k) = \Xi x(k)$$

State-Space Model (3)

- **Prediction**

Model prediction of y(k) ⟶ $$y(k) = \Xi x(k)$$

Model prediction error ⟶ $$e(k) = y_m(k) - y(k)$$

Predicted future output samples

The "state" stored in "memory"

Future input moves (to be decided)

$$\begin{bmatrix} y_{k+1|k} \\ \vdots \\ y_{k+p|k} \end{bmatrix} = \begin{bmatrix} \Xi\Phi \\ \vdots \\ \Xi\Phi^p \end{bmatrix} x(k) + \Omega^u \begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+p-1) \end{bmatrix}$$

$$\Omega^d \begin{bmatrix} \Delta d(k) \\ \vdots \\ \Delta d(k+p-1) \end{bmatrix} + \begin{bmatrix} e(k) \\ \vdots \\ e(k) \end{bmatrix}$$

Dynamic Matrix (made of step response coefficients)

Feedback Error Correction

Feedforward term: new measurement (Assume Δd(k+1)=…= Δ d(k+p-1)=0)