458.604 Process Dynamics & Control

Lecture 1a: MPC on Excel

Jong Min Lee
Chemical and Biological Engineering
Seoul National University

MPC – model predictive control

- Optimal controller is based on minimizing error from trajectory
- Basic version uses **linear** model, but there are many possible models
- Corrections for unmeasured disturbances, model errors are included
- Treats multivariable control, feedforward control

1st-order process

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

general solution?

Excel Implementation of MPC

$$3\frac{dy}{dt} + y(t) = 3u(t)$$

Model Predictive Control Example: 1st order process

Problem Configuration	
y0	0
time step	0.2

Model Parameters	
K	3
tau	3
theta	0

Target Trajectory Parameters	
Final Target	5
Time Constant (tau)	4
Delay (theta)	2

Minimize Either of These	
Sum of Squared Errors	124.04
Sum of Absolute Errors	62.34

t	Δu	u	y (model)	Target	abs(error)	error^2
0.0		0	0.0000	0.0000	0.0000	0.0000
0.2	0.00	0	0.0000	0.0000	0.0000	0.0000
0.4	0.00	0	0.0000	0.0000	0.0000	0.0000
0.6	0.00	0	0.0000	0.0000	0.0000	0.0000
0.8	0.00	0	0.0000	0.0000	0.0000	0.0000
1.0	0.00	0	0.0000	0.0000	0.0000	0.0000
1.2	0.00	0	0.0000	0.0000	0.0000	0.0000
1.4	0.00	0	0.0000	0.0000	0.0000	0.0000
1.6	0.00	1.6593E-05	0.0000	0.0000	0.0000	0.0000
1.8	0.00	1.6593E-05	0.0000	0.0000	0.0000	0.0000
2.0	0.00	1.6593E-05	0.0000	0.0000	0.0000	0.0000
2.2	0.00	1.6593E-05	0.0000	0.2439	0.2438	0.0595
2.4	0.00	1.6593E-05	0.0000	0.4758	0.4758	0.2264
2.6	0.00	1.6593E-05	0.0000	0.6965	0.6964	0.4850
2.8	0.00	1.6593E-05	0.0000	0.9063	0.9063	0.8214
3.0	0.50	0.50001659	0.0968	1.1060	1.0092	1.0186
3.2	0.00	0.50001659	0.1873	1.2959	1.1086	1.2291

Play with manual input and try the "solver" in Excel to compute the input profile

458.604 Process Dynamics & Control

Lecture3: Dynamic Matrix Control (DMC)

Jong Min Lee
School of Chemical and Biological Engineering
Seoul National University

In this lecture, we will discuss

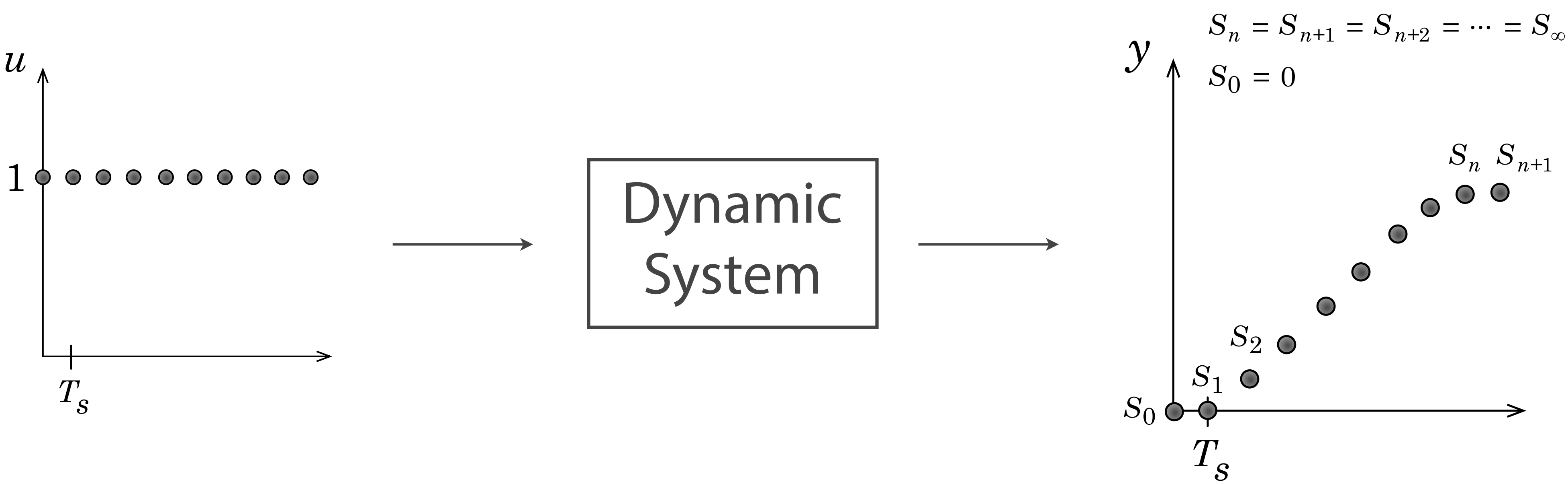
- Process representation: step response model
- Prediction (perfect model)
- Incorporation of "feedback"
- Optimization: unconstrained and constrained QP
- Implementation

Dynamic Matrix Control

- First appeared in the open literature in 1979 (Cutler and Ramaker; Prett and Gillette)
  - with notable success on several Shell processes for many years
- Reformulation as a quadratic program by Garcia and Morshedi in 1986 - "Quadratic Dynamic Matrix Control"
- AspenTech: DMCplus
- Prototype of commercial algorithms presently used in the process industry

Process representation

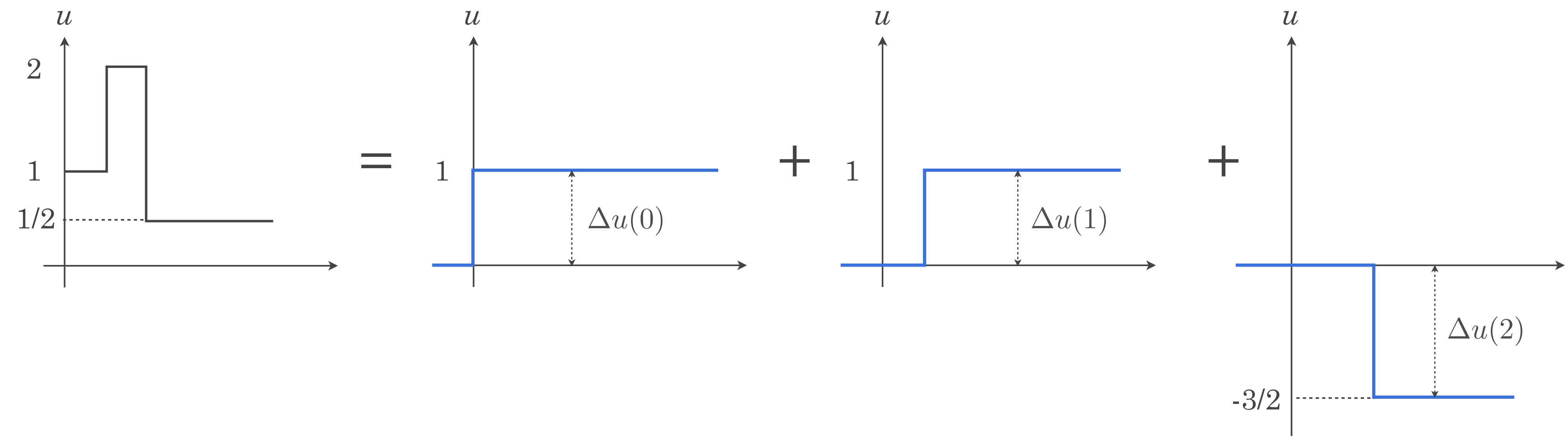
Stable, SISO:

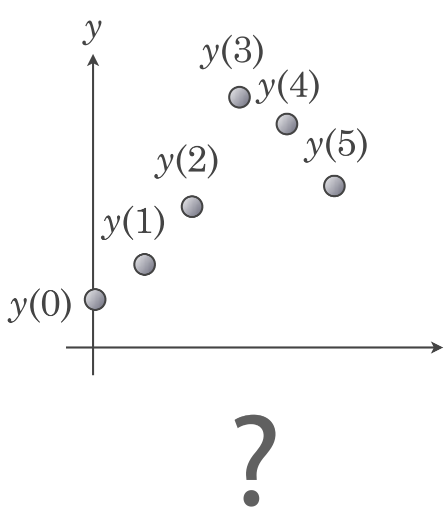


unit step-response function: $S = [S_1,\ S_2,\ S_3,\ \ldots,\ S_n]^T$

Complete description of the process requires n step response coefficients

Principle of superposition

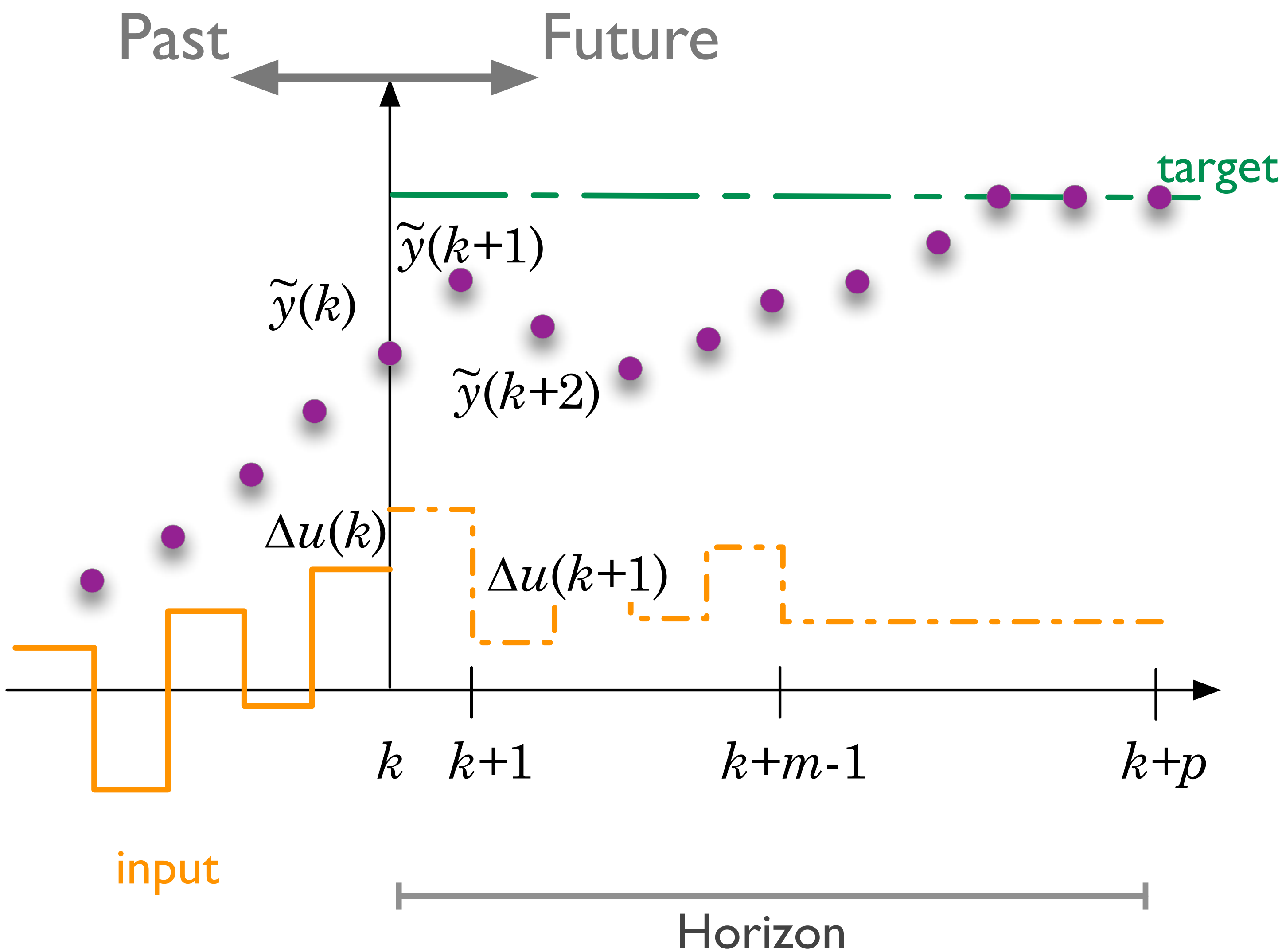




$$y(1) = y(0) + S_1 \Delta u(0)$$
$$y(2) = y(0) + S_1 \Delta u(1) + S_2 \Delta u(0)$$
$$\vdots \quad = \quad \vdots$$
$$y(k+1) = y(0) + \sum_{i=1}^{n-1} S_i \Delta u(k-i+1) + S_n \left\{ \Delta u(k-n+1) + \Delta u(k-n) + \cdots \Delta u(0) \right\}$$
$$= \boxed{y(0) + \sum_{i=1}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1)}$$

$$\Delta u(k-i+1) = u(k-i+1) - u(k-i)$$

Elements of DMC



Predictions

1. Prediction (stable, SISO)

At time k: we know $y(k)$ and need to compute $\Delta u(k)$, which we don't know yet.

$\hat{y}(k+1)$: prediction of $y(k+1)$ made at time k

Assume $y(0) = 0$

$$\hat{y}(k+1) = \sum_{i=1}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1)$$

$$\hat{y}(k+1) = \underbrace{S_1 \Delta u(k)}_{\text{Effect of current control action}} + \underbrace{\sum_{i=2}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1)}_{\text{Effect of past control actions}}$$

-
- Substitute k = k+1, and
-

$$\hat{y}(k+2) = \underbrace{S_1 \Delta u(k+1)}_{\text{Effect of future control action}} + \underbrace{S_2 \Delta u(k)}_{\text{Effect of current control action}} + \underbrace{\sum_{i=3}^{n-1} S_i \Delta u(k-i+2) + S_n u(k-n+2)}_{\text{Effect of past control actions}}$$

j-step ahead prediction

$$\hat{y}(k+j) = \underbrace{\sum_{i=1}^{j} S_i \Delta u(k+j-i)}_{\text{Effect of current and future control actions}} + \underbrace{\sum_{i=j+1}^{n-1} S_i \Delta u(k+j-i) + S_n u(k+j-n)}_{\text{Effect of past control actions}}$$

Effect of current and future control actions

Effect of past control actions

Let

$$\hat{y}^0(k+j) \triangleq \sum_{i=j+1}^{n-1} S_i \Delta u(k+j-i) + S_n u(k+j-n)$$

This is referred to as "predicted unforced response" with past inputs only

$$\mathbf{U} = [\ldots,\ u(k-2),\ u(k-1),\ 0,\ 0,\ 0,\ \ldots]^T \quad \text{for } j = 1$$

$$\boxed{\hat{y}(k+j) = \sum_{i=1}^{j} S_i \Delta u(k+j-i) + \hat{y}^0(k+j)}$$

Multiple predictions

$$\hat{\mathbf{Y}}(k+1) \triangleq [\hat{y}(k+1),\ \hat{y}(k+1),\ \cdots,\ \hat{y}(k+p)]^T$$

$$\hat{\mathbf{Y}}^0(k+1) \triangleq \left[\hat{y}^0(k+1),\ \hat{y}^0(k+2),\ \cdots,\ \hat{y}^0(k+p)\right]^T$$

$$\mathbf{U}(k) \triangleq [\Delta u(k),\ \Delta u(k+1),\ \cdots,\ \Delta u(k+m-1)]^T$$

p: prediction horizon, m: control horizon $\quad m \leq p \leq n+m$

In a matrix form:

$$\hat{\mathbf{Y}}(k+1) = \mathcal{S}\Delta\mathbf{U}(k) + \hat{\mathbf{Y}}^0(k+1)$$

$$\mathcal{S} \triangleq \begin{bmatrix} S_1 & 0 & 0 & \cdots & 0 \\ S_2 & S_1 & 0 & \cdots & 0 \\ S_3 & S_2 & S_1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ S_m & S_{m-1} & S_{m-2} & \cdots & S_1 \\ S_{m+1} & S_m & S_{m-1} & \cdots & S_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ S_p & S_{p-1} & S_{p-2} & \cdots & S_{p-m+1} \end{bmatrix}$$

Output feedback and bias correction

So far, we have not utilized the latest observation, $y(k)$.

The fact is that there is no perfect model.

Corrected prediction by adding a constant bias term.

$$\tilde{y}(k+j) \triangleq \hat{y}(k+j) + b(k+j)$$

$$b(k+j) = y(k) - \hat{y}(k)$$

$\hat{y}(k)$: one-step ahead prediction made at the previous time instance, k-1

$$\tilde{y}(k+j) = \hat{y}(k+j) + [y(k) - \hat{y}(k)]$$

⇩

$$\boxed{\tilde{\mathbf{Y}}(k+1) = \mathcal{S}\Delta\mathbf{U}(k) + \hat{\mathbf{Y}}^0(k+1) + [y(k) - \hat{y}(k)]\,\mathbf{1}}$$

$$\tilde{\mathbf{Y}}(k+1) = [\tilde{y}(k+1),\ \tilde{y}(k+2),\ \cdots,\ \tilde{y}(k+p)]^T$$

$$\mathbf{1} = [1,\ 1,\ ,\cdots,\ 1]^T$$

Recursive update of unforced response

For stable models, one can update the predicted unforced response after

$u(k)$ is computed.

works like a state; hence you need "n" not "p"

$$\hat{\mathbf{Y}}_n^0(k+1) = \mathbf{M}\hat{\mathbf{Y}}_n^0(k) + \mathcal{S}^*\Delta u(k)$$

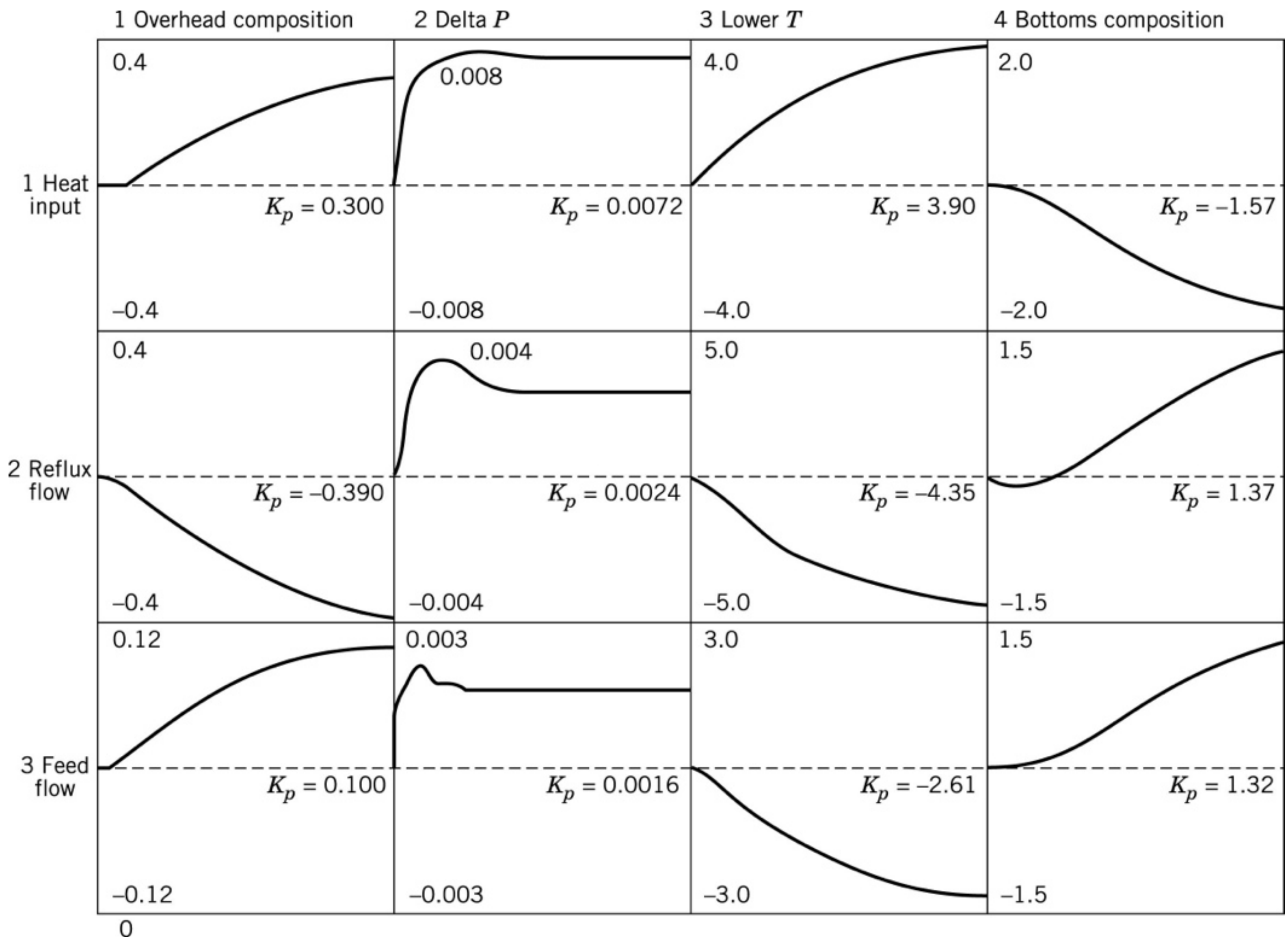
or

$$\begin{bmatrix} \hat{y}^0(k+1) \\ \hat{y}^0(k+2) \\ \vdots \\ \hat{y}^0(k+\cancel{p}\,n) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{y}^0(k) \\ \hat{y}^0(k+1) \\ \vdots \\ \hat{y}^0(k+\cancel{p}\,n-1) \end{bmatrix} + \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{\cancel{p}\,n} \end{bmatrix} \Delta u(k)$$

Why??

- To achieve good control performance
  - $\tilde{\mathbf{Y}}(k+1)$ should be close to the true open-loop output
  - This requires that n, the number of coefficient matrices in S* is chosen that $S_n = S_{n+1}$ (i.e., plant should be stable), otherwise $\mathbf{M}\hat{\mathbf{Y}}^0$ will be in error. It also requires the feedback term stays approximately constant. (step disturbance)

1. Prediction (stable, MIMO)



2-by-2 system

$$\hat{y}_1(k+1) = \sum_{i=1}^{n-1} S_{11,i}\Delta u_1(k-i+1) + S_{11,n}u_1(k-n+1)$$
$$+ \sum_{i=1}^{n-1} S_{12,i}\Delta u_2(k-i+1) + S_{12,n}u_2(k-n+1)$$

$$\hat{y}_2(k+1) = \sum_{i=1}^{n-1} S_{21,i}\Delta u_1(k-i+1) + S_{21,n}u_1(k-n+1)$$
$$+ \sum_{i=1}^{n-1} S_{22,i}\Delta u_2(k-i+1) + S_{22,n}u_2(k-n+1)$$

Vector notation

$$\mathbf{I}_p = \begin{bmatrix} \mathbf{I} \\ \vdots \\ \mathbf{I} \end{bmatrix} \quad pn_y\text{-}by\text{-}n_y$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_r \end{bmatrix}$$

$$\tilde{\mathbf{Y}}(k+1) = \mathbf{S}\Delta\mathbf{U}(k) + \hat{\mathbf{Y}}^0(k+1) + \mathbf{I}_p\left[\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right]$$

mp-by-1

$$\tilde{\mathbf{Y}}(k+1) = \begin{bmatrix} \tilde{\mathbf{y}}(k+1) \\ \tilde{\mathbf{y}}(k+2) \\ \vdots \\ \tilde{\mathbf{y}}(k+p) \end{bmatrix} \quad \hat{\mathbf{Y}}^0(k+1) = \begin{bmatrix} \hat{\mathbf{y}}^0(k+1) \\ \hat{\mathbf{y}}^0(k+2) \\ \vdots \\ \hat{\mathbf{y}}^0(k+p) \end{bmatrix} \quad \Delta\mathbf{U}(k) = \begin{bmatrix} \Delta\mathbf{u}(k) \\ \Delta\mathbf{u}(k+1) \\ \vdots \\ \Delta\mathbf{u}(k+m-1) \end{bmatrix}$$

Use up to p only out of n

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{S}_2 & \mathbf{S}_1 & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{S}_m & \mathbf{S}_{m-1} & \cdots & \mathbf{S}_1 \\ \mathbf{S}_{m+1} & \mathbf{S}_m & \cdots & \mathbf{S}_2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{S}_p & \mathbf{S}_{p-1} & \cdots & \mathbf{S}_{p-m+1} \end{bmatrix} \quad \mathbf{S}_i = \begin{bmatrix} S_{11,i} & S_{12,i} & \cdots & S_{1r,i} \\ S_{21,i} & \cdots & \cdots & S_{2r,i} \\ \vdots & \vdots & \vdots & \vdots \\ S_{m1,i} & \cdots & \cdots & S_{mr,i} \end{bmatrix}$$

Recursive update of unforced response

$$\hat{\mathbf{Y}}_n^0(k+1) = \mathbf{M}\hat{\mathbf{Y}}_n^0(k) + \mathbf{S}^*\Delta\mathbf{u}(k)$$

or

$$\begin{bmatrix} \hat{\mathbf{y}}^0(k+1) \\ \hat{\mathbf{y}}^0(k+2) \\ \vdots \\ \hat{\mathbf{y}}^0(k+\underset{n}{\cancel{p}}) \end{bmatrix} = \begin{bmatrix} 0 & \mathbf{I}_m & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{I}_m & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & \mathbf{I}_m \\ 0 & 0 & \cdots & 0 & \mathbf{I}_m \end{bmatrix} \begin{bmatrix} \hat{\mathbf{y}}^0(k) \\ \hat{\mathbf{y}}^0(k+1) \\ \vdots \\ \hat{\mathbf{y}}^0(k+\underset{n}{\cancel{p}}-1) \end{bmatrix} + \begin{bmatrix} \mathbf{S}_1 \\ \mathbf{S}_2 \\ \vdots \\ \mathbf{S}_{\underset{n}{\cancel{p}}} \end{bmatrix} \Delta\mathbf{u}(k)$$

Control calculations

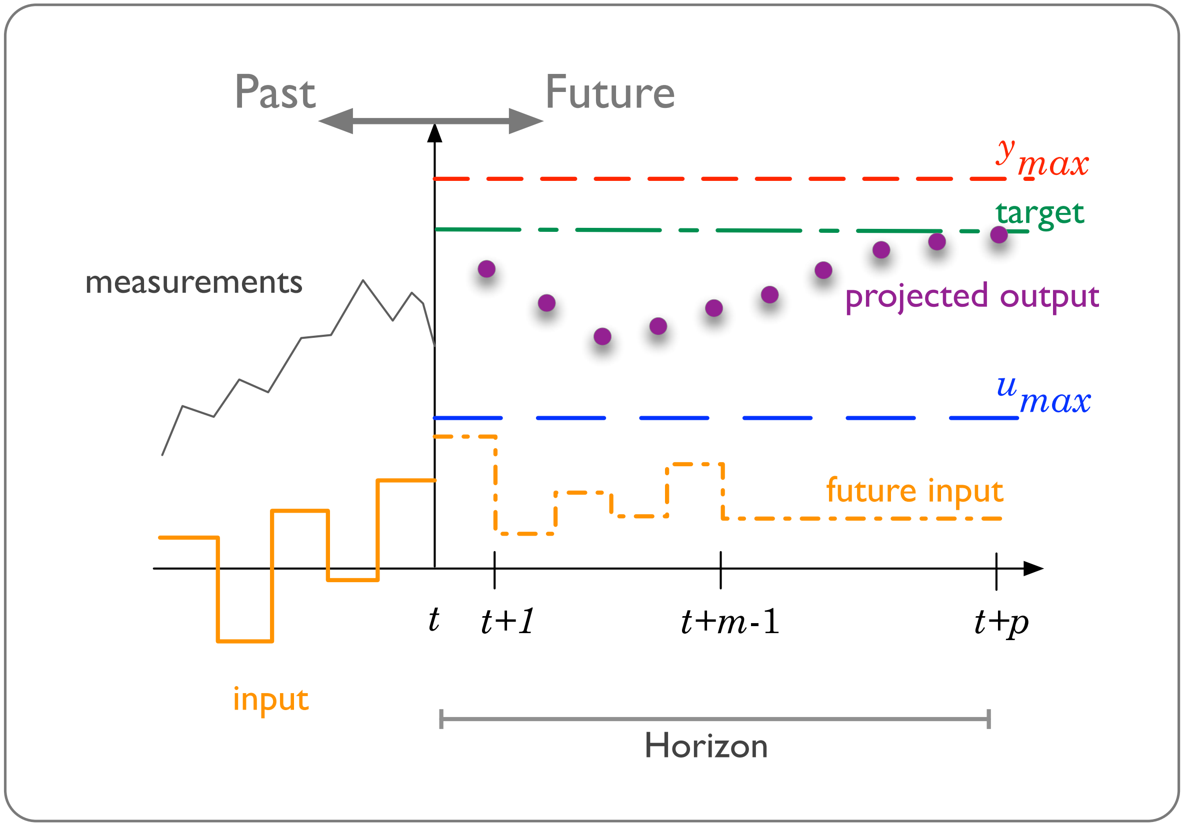
Objective function

At time k, minimize the predicted deviation of the output from the setpoint with some penalty on the input movement size measured in terms of the quadratic norm.

$$\min_{\Delta \mathbf{U}(k)} \left\{ \sum_{i=1}^{p} \left(\mathbf{y}_r(k+i) - \tilde{\mathbf{y}}(k+i)\right)^T Q \left(\mathbf{y}_r(k+i) - \tilde{\mathbf{y}}(k+i)\right) \right.$$
$$\left. + \sum_{\ell=0}^{m-1} \Delta \mathbf{u}^T(k+\ell) R \Delta \mathbf{u}(k+\ell) \right\}$$

Q, R : weighting matrices (diagonal)

Constraints



Input magnitude

$$\mathbf{u}_{\min} \leq \mathbf{u}(k+\ell) \leq \mathbf{u}_{\max}$$

Input rate

$$|\Delta\mathbf{u}(k+\ell)| \leq \Delta\mathbf{u}_{max}$$

Output magnitude

$$\mathbf{y}_{\min} \leq \tilde{\mathbf{y}}(k+i) \leq \mathbf{y}_{\min}$$

Solve: quadratic program

$$\min_{\Delta\mathbf{U}(k)} \left\{ \frac{1}{2} \Delta\mathbf{U}^T(k) \mathcal{H} \Delta\mathbf{U}(k) + \mathbf{f}^T \Delta\mathbf{U}(k) \right\}$$

$$\mathbf{A}\Delta\mathbf{U}(k) \leq \mathbf{b}$$

$\mathcal{H}$: Hessian matrix

$\mathbf{f}$: gradient vector

$\mathbf{A}$: constraint matrix

$\mathbf{b}$: constraint vector

$\Delta\mathbf{U}(k)$: decision variable

We need to convert the MPC objective and constraints to the standard QP form.

Unconstrained problem

$$\min_{\Delta\mathbf{U}(k)} \left\{ \frac{1}{2}\Delta\mathbf{U}^T(k)\mathcal{H}\Delta\mathbf{U}(k) + \mathbf{f}^T\Delta\mathbf{U}(k) \right\}$$

Take the gradient w.r.t. the input:

$$\mathcal{H}\Delta\mathbf{U}(k) + \mathbf{f} = 0$$

$$\Delta\mathbf{U}(k) = -\mathcal{H}^{-1}\mathbf{f}$$

Objective function in quadratic form

$$\min_{\Delta \mathbf{U}(k)} \left\{ \sum_{i=1}^{p} \left(\mathbf{y}_r(k+i) - \tilde{\mathbf{y}}(k+i)\right)^T Q \left(\mathbf{y}_r(k+i) - \tilde{\mathbf{y}}(k+i)\right) + \sum_{\ell=0}^{m-1} \Delta \mathbf{u}^T(k+\ell) R \Delta \mathbf{u}(k+\ell) \right\}$$

$$\begin{bmatrix} \mathbf{y}_r(k+1) - \tilde{\mathbf{y}}(k+1) \\ \mathbf{y}_r(k+2) - \tilde{\mathbf{y}}(k+2) \\ \vdots \\ \mathbf{y}_r(k+p) - \tilde{\mathbf{y}}(k+p) \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & & & \\ & \mathbf{Q} & & \\ & & \ddots & \\ & & & \mathbf{Q} \end{bmatrix} \begin{bmatrix} \mathbf{y}_r(k+1) - \tilde{\mathbf{y}}(k+1) \\ \mathbf{y}_r(k+2) - \tilde{\mathbf{y}}(k+2) \\ \vdots \\ \mathbf{y}_r(k+p) - \tilde{\mathbf{y}}(k+p) \end{bmatrix}$$

$$+ \begin{bmatrix} \Delta \mathbf{u}(k) \\ \Delta \mathbf{u}(k+1) \\ \vdots \\ \Delta \mathbf{u}(k+m-1) \end{bmatrix}^T \begin{bmatrix} \mathbf{R} & & & \\ & \mathbf{R} & & \\ & & \ddots & \\ & & & \mathbf{R} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}(k) \\ \Delta \mathbf{u}(k+1) \\ \vdots \\ \Delta \mathbf{u}(k+m-1) \end{bmatrix}$$

$$\Rightarrow \left(\mathbf{Y}_r(k+1) - \tilde{\mathbf{Y}}(k+1)\right)^T \bar{\mathbf{Q}} \left(\mathbf{Y}_r(k+1) - \tilde{\mathbf{Y}}(k+1)\right) + \Delta \mathbf{U}^T(k) \bar{\mathbf{R}} \Delta \mathbf{U}(k)$$

Not done yet!

$$\left(\mathbf{Y}_r(k+1) - \tilde{\mathbf{Y}}(k+1)\right)^T \bar{\mathbf{Q}} \left(\mathbf{Y}_r(k+1) - \tilde{\mathbf{Y}}(k+1)\right) + \Delta\mathbf{U}^T(k)\bar{\mathbf{R}}\Delta\mathbf{U}(k)$$

$$\tilde{\mathbf{Y}}(k+1) = \mathbf{S}\Delta\mathbf{U}(k) + \hat{\mathbf{Y}}^0(k+1) + \mathbf{I}_p\left[\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right]$$

This yields

$$\varepsilon^T(k+1)\bar{\mathbf{Q}}\varepsilon(k+1) - 2\varepsilon^T(k+1)\bar{\mathbf{Q}}\mathbf{S}\Delta\mathbf{U}(k) + \Delta\mathbf{U}^T(k)\left(\mathbf{S}^T\bar{\mathbf{Q}}\mathbf{S} + \bar{\mathbf{R}}\right)\Delta\mathbf{U}(k)$$

where

$$\varepsilon(k+1) = \mathbf{Y}_r(k+1) - \hat{\mathbf{Y}}^0(k+1) - \mathbf{I}_p\left[\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right]$$

is a known term.

Hessian (a constant matrix): $\mathcal{H} = \mathbf{S}^T\bar{\mathbf{Q}}\mathbf{S} + \bar{\mathbf{R}}$

gradient vector (must be updated at each time): $\mathbf{f}^T = -\varepsilon^T(k+1)\bar{\mathbf{Q}}\mathbf{S}$

Constraints in linear inequality form

$$\mathbf{u}_{\min} \leq \mathbf{u}(k+\ell) \leq \mathbf{u}_{\max}$$

$$|\Delta\mathbf{u}(k+\ell)| \leq \Delta\mathbf{u}_{max}$$

$$\mathbf{y}_{\min} \leq \tilde{\mathbf{y}}(k+i) \leq \mathbf{y}_{\min}$$

$$i = 1, \cdots, p$$

$$\ell = 0, \cdots, m-1$$

$$\Longrightarrow \quad \mathbf{A}\Delta\mathbf{U}(k) \leq \mathbf{b}$$

Input magnitude constraint

$$\mathbf{u}_{\min} \le \mathbf{u}(k+\ell) \le \mathbf{u}_{\max}, \quad \ell = 0, \ \cdots, \ m-1$$

$$-\mathbf{u}(k-1) - \sum_{i=0}^{\ell} \Delta\mathbf{u}(k+i) \le \mathbf{u}_{\min}$$

$$\mathbf{u}(k-1) + \sum_{i=0}^{\ell} \Delta\mathbf{u}(k+i) \le \mathbf{u}_{\max}$$

$\mathbf{I}_L$

$$\begin{bmatrix} -\begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \end{bmatrix} \\ \begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \end{bmatrix} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}(k) \\ \Delta\mathbf{u}(k+1) \\ \vdots \\ \Delta\mathbf{u}(k+m-1) \end{bmatrix} \le \begin{bmatrix} -\begin{bmatrix} \mathbf{u}_{\min} - \mathbf{u}(k-1) \\ \mathbf{u}_{\min} - \mathbf{u}(k-1) \\ \vdots \\ \mathbf{u}_{\min} - \mathbf{u}(k-1) \end{bmatrix} \\ \begin{bmatrix} \mathbf{u}_{\max} - \mathbf{u}(k-1) \\ \mathbf{u}_{\max} - \mathbf{u}(k-1) \\ \vdots \\ \mathbf{u}_{\max} - \mathbf{u}(k-1) \end{bmatrix} \end{bmatrix}$$

Input rate constraints

$$|\Delta \mathbf{u}(k+\ell)| \leq \Delta \mathbf{u}_{\max} \qquad \ell = 0, \cdots, m-1$$

$$-\Delta \mathbf{u}_{\max} \leq \Delta \mathbf{u}(k+\ell) \leq \Delta \mathbf{u}_{\max}$$

$$\begin{aligned} \Delta \mathbf{u}(k+\ell) &\leq \Delta \mathbf{u}_{\max} \\ -\Delta \mathbf{u}(k+\ell) &\leq \Delta \mathbf{u}_{\max} \end{aligned}$$

$\mathbf{I}$

$$\begin{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I} \end{bmatrix} \\ -\begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I} \end{bmatrix} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}(k) \\ \Delta \mathbf{u}(k+1) \\ \vdots \\ \Delta \mathbf{u}(k+m-1) \end{bmatrix} \leq \begin{bmatrix} \begin{bmatrix} \Delta \mathbf{u}_{\max} \\ \Delta \mathbf{u}_{\max} \\ \vdots \\ \Delta \mathbf{u}_{\max} \end{bmatrix} \\ \begin{bmatrix} \Delta \mathbf{u}_{\max} \\ \Delta \mathbf{u}_{\max} \\ \vdots \\ \Delta \mathbf{u}_{\max} \end{bmatrix} \end{bmatrix}$$

Output magnitude constraints

$$\mathbf{y}_{\min} \leq \tilde{\mathbf{y}}(k+i) \leq \mathbf{y}_{\max}, \quad i = 1, \cdots, p$$

$$\tilde{\mathbf{y}}(k+i) \leq \mathbf{y}_{\max}$$
$$-\tilde{\mathbf{y}}(k+i) \leq -\mathbf{y}_{\min}$$

$$\begin{bmatrix} \mathbf{S}\Delta\mathbf{U}(k) + \hat{\mathbf{Y}}^0(k+1) + \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \\ -\mathbf{S}\Delta\mathbf{U}(k) - \hat{\mathbf{Y}}^0(k+1) - \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \end{bmatrix} \leq \begin{bmatrix} \mathbf{Y}_{\max} \\ -\mathbf{Y}_{\min} \end{bmatrix}$$

$$\mathbf{Y}_{\max} = \begin{bmatrix} \mathbf{y}_{\max} \\ \mathbf{y}_{\max} \\ \vdots \\ \mathbf{y}_{\max} \end{bmatrix} \qquad \mathbf{Y}_{\min} = \begin{bmatrix} \mathbf{y}_{\min} \\ \mathbf{y}_{\min} \\ \vdots \\ \mathbf{y}_{\min} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{S} \\ -\mathbf{S} \end{bmatrix} \Delta\mathbf{U}(k) \leq \begin{bmatrix} \mathbf{Y}_{\max} - \hat{\mathbf{Y}}^0(k+1) - \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \\ -\mathbf{Y}_{\min} + \hat{\mathbf{Y}}^0(k+1) + \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \end{bmatrix}$$

In summary,

$$
\begin{bmatrix} -\mathbf{I}_L \\ \mathbf{I}_L \\ -\mathbf{I} \\ \mathbf{I} \\ -\mathbf{S} \\ \mathbf{S} \end{bmatrix} \Delta \mathbf{U}(k) \leq
\begin{bmatrix}
-\begin{bmatrix} \mathbf{u}_{\min} - \mathbf{u}(k-1) \\ \vdots \\ \mathbf{u}_{\min} - \mathbf{u}(k-1) \end{bmatrix} \\
\begin{bmatrix} \mathbf{u}_{\max} - \mathbf{u}(k-1) \\ \vdots \\ \mathbf{u}_{\max} - \mathbf{u}(k-1) \end{bmatrix} \\
\begin{bmatrix} \Delta \mathbf{u}_{\max} \\ \vdots \\ \Delta \mathbf{u}_{\max} \end{bmatrix} \\
\begin{bmatrix} \Delta \mathbf{u}_{\max} \\ \vdots \\ \Delta \mathbf{u}_{\max} \end{bmatrix} \\
\begin{bmatrix} \mathbf{Y}_{\max} - \hat{\mathbf{Y}}^0(k+1) - \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \\ -\mathbf{Y}_{\min} + \hat{\mathbf{Y}}^0(k+1) + \mathbf{I}_p\left(\mathbf{y}(k) - \hat{\mathbf{y}}(k)\right) \end{bmatrix}
\end{bmatrix}
$$

should be swapped ($-\mathbf{S}$ and $\mathbf{S}$)

$$\mathbf{A}\Delta \mathbf{U}(k) \leq \mathbf{b}$$

Solving QP

- Quadratic program: minimization of quadratic function subject to linear inequality constraints.
- QPs are convex and therefore fundamentally tractable.
- Off-the-shelf solvers (e.g., QPSOL, QUADPROG) are available but further customization is desirable (to exploit the structure in the Hessian and constraint matrices)
- Complexity of a QP is a complex function of the dimension/structure of Hessian, as well as the number of constraints.

- Active set method
- Interior point method
  - Barrier function

Real-time implementation

1. Initialization: Initialize the memory vector and the reference vector $\hat{\mathbf{Y}}(0)$ and the reference vector. Set $k = 0$.

2. Memory update: $\hat{\mathbf{Y}}^0(k+1) = \mathbf{M}\hat{\mathbf{Y}}^0(k) + \mathbf{S}^*\Delta\mathbf{u}(k)$

3. Reference vector update

4. Measurement intake: Take in new measurement $y(k)$ and $\Delta d(k)$

5. Calculation of the gradient vector and constraint vector

6. Solve QP

7. Implementation of input

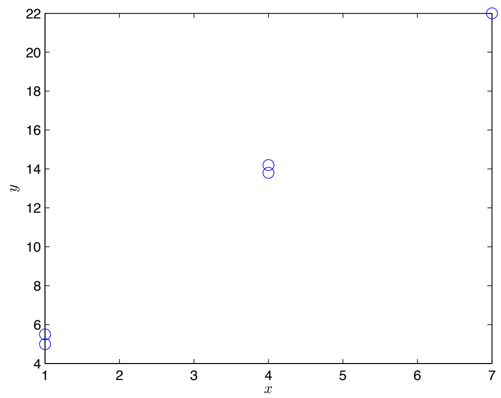
$$\mathbf{u}(k) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k)$$

8. Go back to step 2 after setting

Linear Least-Squares Regression

Suppose we want to fit a line through some experimental data.

x	y
1	5.5
7	22
4	14.2
1	5.0
4	13.8



We want to calculate the slope and offset of the line to maximize the accuracy of the predictions.

- Prediction means: $\hat{y} = f(x,\ \beta)$ when the true model is $y = f(x, \beta) + \epsilon$.
- Prediction accuracy:
  - many ways to measure
  - most common objective is to minimize $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$.

Least squares regression:

$$\min_{m,\ b} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

subject to

$$\begin{aligned}
\hat{y}_1 &= mx_1 + b \\
\hat{y}_2 &= mx_2 + b \\
\vdots &= \vdots \\
\hat{y}_n &= mx_n + b
\end{aligned}$$

or in vector form:

$$\min_{\beta}(\mathbf{y}-\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}})$$

subject to

$$\hat{\mathbf{y}} = \mathbf{X}\beta \qquad \text{and} \qquad \mathbf{y} = \mathbf{X}\beta + \epsilon$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix},\ \hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_n \end{bmatrix},\ \beta = \begin{bmatrix} b \\ m \end{bmatrix},\ \mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

To solve this problem:

1. Eliminate $\hat{y}$ by substitution:

$$\min_{\beta}(\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{y}-\mathbf{X}\beta)$$

2. Take the derivative (gradient) and set to zero

$$\nabla_\beta\left\{(\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{y}-\mathbf{X}\beta)\right\} = 0$$

yields

$$-2\mathbf{X}^T(\mathbf{y}-\mathbf{X}\beta) = 0$$

or

$$\mathbf{X}^T\mathbf{y} = \mathbf{X^T X}\beta$$

then,

$$\boxed{\hat{\beta} = (\mathbf{X^T X})^{-\mathbf{1}}\mathbf{X^T y}}$$

In our example,

$$\hat{\beta} = \begin{bmatrix} 2.5476 \\ 2.8095 \end{bmatrix} \Rightarrow \hat{y} = 2.5476 + 2.8095x$$

- How accurate are the estimates for the model predictions?

$$\sigma_\epsilon^2 = \text{var}(y - \hat{y}) \approx \frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n-p}(\mathbf{y} - \hat{\mathbf{y}})^T(\mathbf{y} - \hat{\mathbf{y}})$$

- How accurate are the estimates of the model parameters?

$$\text{cov}(\hat{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\sigma_\epsilon^2$$

Derivation:

Note that $y = X\beta + \epsilon$ and $\mathbb{E}(\hat{\beta}) = (X^TX)^{-1}X^T\mathbb{E}(X\beta + \epsilon) = \beta$.

$$\begin{aligned}
\text{cov}(\hat{\beta}) &= \mathbb{E}\left[\left\{(X^TX)^{-1}X^T(X\beta + \epsilon) - \beta\right\}\left\{(X^TX)^{-1}X^T(X\beta + \epsilon) - \beta\right\}^T\right] \\
&= \mathbb{E}\left[\left\{(X^TX)^{-1}X^T\epsilon\right\}\left\{(X^TX)^{-1}X^T\epsilon\right\}^T\right] \\
&= \mathbb{E}\left[(X^TX)^{-1}X^T\epsilon\epsilon^TX(X^TX)^{-1}\right] \qquad \because X^TX \text{ is a symmetric matrix.} \\
&= \sigma_\epsilon^2\mathbf{I}(X^TX)^{-1}
\end{aligned}$$

where $\sigma_\epsilon^2\mathbf{I}$ is the noise covariance matrix with the same variance term on the diagonal locations.

Nonlinear Regression

- Fitting nonlinear models (nonlinear in the parameters)
- Models of the form:

$$y = f(\mathbf{x}, \beta) + \epsilon$$

  where $\epsilon \in \mathcal{N}(0,\ \sigma_\epsilon^2)$.

- no analytical solution: numerical optimization
- has the form of

$$\min_\beta(\mathbf{y} - \hat{\mathbf{y}})^T(\mathbf{y} - \hat{\mathbf{y}})$$

subject to

$$\hat{\mathbf{y}} = f(\mathbf{X}, \beta)$$

- requires nonlinear programming
  - most common approach is Levenberg-Marquardt
  - Matlab has several possibilities
    * `fmincon` – SQP method
    * `lsqcurvefit`

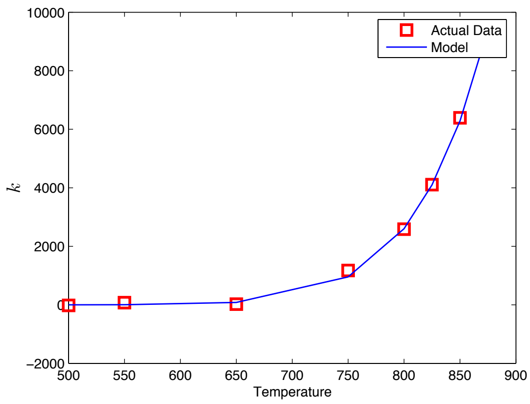
Example

The following rate data was collected to determine temperature dependence on kinetic rate constant.

$$k = \alpha e^{-\beta/T} = 8.5 \times 10^9 e^{-12000/T} + \epsilon$$

where $\epsilon \in \mathcal{N}(0, 10000)$.

T (°R)	k
500	-18.3500
550	75.4229
650	22.7654
750	1174.9
800	2586.5
825	4107.8
850	6390.2
875	9411.4



Use `lsqcurvefit`:

```
T= [500; 550; 650; 750; 800; 825; 850; 875];
k = [-18.35; 75.4229; 22.7654; 1174.9; 2586.5; 4107.8; 6390.2; 9411
xdata = T;
ydata = k;
```

```
beta = lsqcurvefit(@kineticf, [0; 0], xdata, ydata);

%kineticf.m
function F= kineticf(beta, xdata)
F = beta(1)*10^9*exp(-beta(2)*1000./xdata);
```

Note that some "trick" is needed for a better convergence: make all parameters ($\beta(1)$, $\beta(2)$) about the same magnitude. Matlab will give the following answer:

$$\begin{aligned} k &= 6.4569 \times 10^9 e^{-11758.8/T} \\ \hat{\sigma_\epsilon}^2 &= \frac{1}{8-2}\sum_{i=1}^{8}(y(i)-\hat{y}(i))^2 = 8855.5 \\ \hat{\sigma_\epsilon} &= 94.1 \end{aligned}$$

Common approach using linear regression – Is this valid?

For the same example, regress $\ln(k)$ vs. $1/T$:

$$\ln(k) = \ln(\alpha) - \beta\left(\frac{1}{T}\right)$$

Then,

$$\begin{bmatrix} \ln\alpha \\ \beta \end{bmatrix} = \left(\begin{bmatrix} 1 & \frac{1}{T_1} \\ \vdots & \vdots \\ 1 & \frac{1}{T_n} \end{bmatrix}^T \begin{bmatrix} 1 & \frac{1}{T_1} \\ \vdots & \vdots \\ 1 & \frac{1}{T_n} \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & \frac{1}{T_1} \\ \vdots & \vdots \\ 1 & \frac{1}{T_n} \end{bmatrix}^T \begin{bmatrix} \ln k_1 \\ \vdots \\ \ln k_n \end{bmatrix}$$

Note that we must throw away the 1st data point because k should be positive inside the logarithm. The solution is

$$\begin{aligned} k &= 9.0891 \times 10^7 e^{-8411.4/T} \\ \hat{\sigma_\epsilon}^2 &= \frac{1}{7-2}\sum_{i=2}^{8}(y(i)-\hat{y}(i))^2 = 1.497 \times 10^7 \\ \hat{\sigma_\epsilon} &= 1730.3 \end{aligned}$$

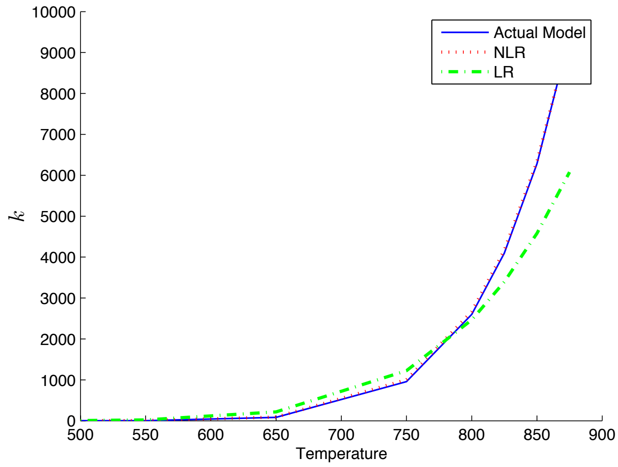


Figure 1: Comparison of LR and NLR for the nonlinear example

Comments

The problem is in transforming the rate equation. We have assumed that the model is of the form:

$$\ln k = \ln \alpha + \frac{\beta}{T} + \epsilon$$

Inverting the transform gives

$$k = e^{[\ln \alpha - \beta/T + \epsilon]}$$

or

$$k = \alpha e^{-\beta/T} \times e^{\epsilon}$$

but we know the model has the form:

$$k = \alpha e^{-\beta/T} + \epsilon$$

- Transforming to the linear form is fitting the wrong model.
- Fit the correct model.
- Why does it matter that transform + LLSR is technically incorrect if it gives you a “good enough answer"?

  - quick and dirty
  - gets into the ball park
  - example shows that it was really good at low temperature.

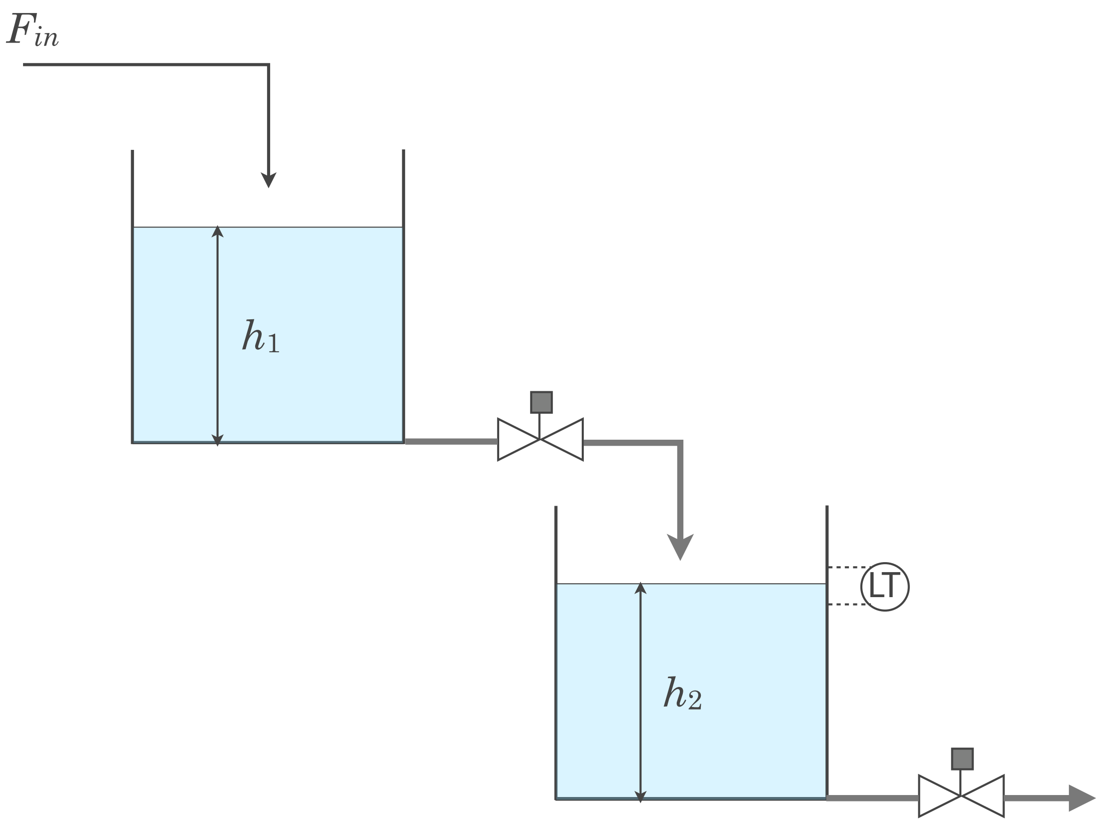
- Here are the issues:

  1. Inaccuracy was in the measurements.
     - Objective was to discover the “true" relationship between the variables.
  2. Wrong relationship gives poor designs.
  3. Looks “accurate" enough over some rage.
     - Can’t know where it will or won’t be accurate

Kalman Filter

Jong Min Lee
Chemical and Biological Engineering
Seoul National University

Non-interacting two-tank system



$$\mathbf{h}(k) = F[\mathbf{h}(k-1), F_{in}(k-1)]$$

$$\mathbf{x}(k) = F[\mathbf{x}(k-1), u(k-1)]$$

measurement eqn:

$$y(k) = \mathbf{C}\mathbf{x}(k)$$

$$\mathbf{C} = [0 \quad 1]$$

$$A\frac{dh_1}{dt} = F_{in} - F_{out,1} = F_{in} - C_{v1}\sqrt{h_1}$$

$$A\frac{dh_2}{dt} = F_{out,1} - F_{out,2} = C_{v1}\sqrt{h_1} - C_{v2}\sqrt{h_2}$$

Linearization and discretization yield:

$$\mathbf{x}'(k+1) = \mathbf{\Phi}\mathbf{x}'(k) + \mathbf{\Gamma}u'(k)$$
$$y'(k) = \mathbf{C}\mathbf{x}'(k)$$

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_{11} & 0 \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \qquad \mathbf{\Gamma} = \begin{bmatrix} \Gamma_1 \\ 0 \end{bmatrix} \qquad \mathbf{C} = [0 \;\; 1]$$

$A : 0.25 \ [\mathrm{m}^2]$ $\bar{h}_1 : 0.36 \ [\mathrm{m}]$ $\bar{h}_2 : 0.25 \ [\mathrm{m}]$

$\bar{F}_{in} : 0.3 \ [\mathrm{m}^3/\mathrm{min}]$ $C_{v1} : 0.5 \ [\mathrm{m}^{2.5}/\mathrm{min}]$ $C_{v2} : 0.6 \ [\mathrm{m}^{2.5}/\mathrm{min}]$

$$\dot{\mathbf{x}}' = \begin{bmatrix} -1.67 & 0 \\ 1.67 & -2.4 \end{bmatrix} \mathbf{x}' + \begin{bmatrix} 4 \\ 0 \end{bmatrix} u'$$
$$y' = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{x}'$$

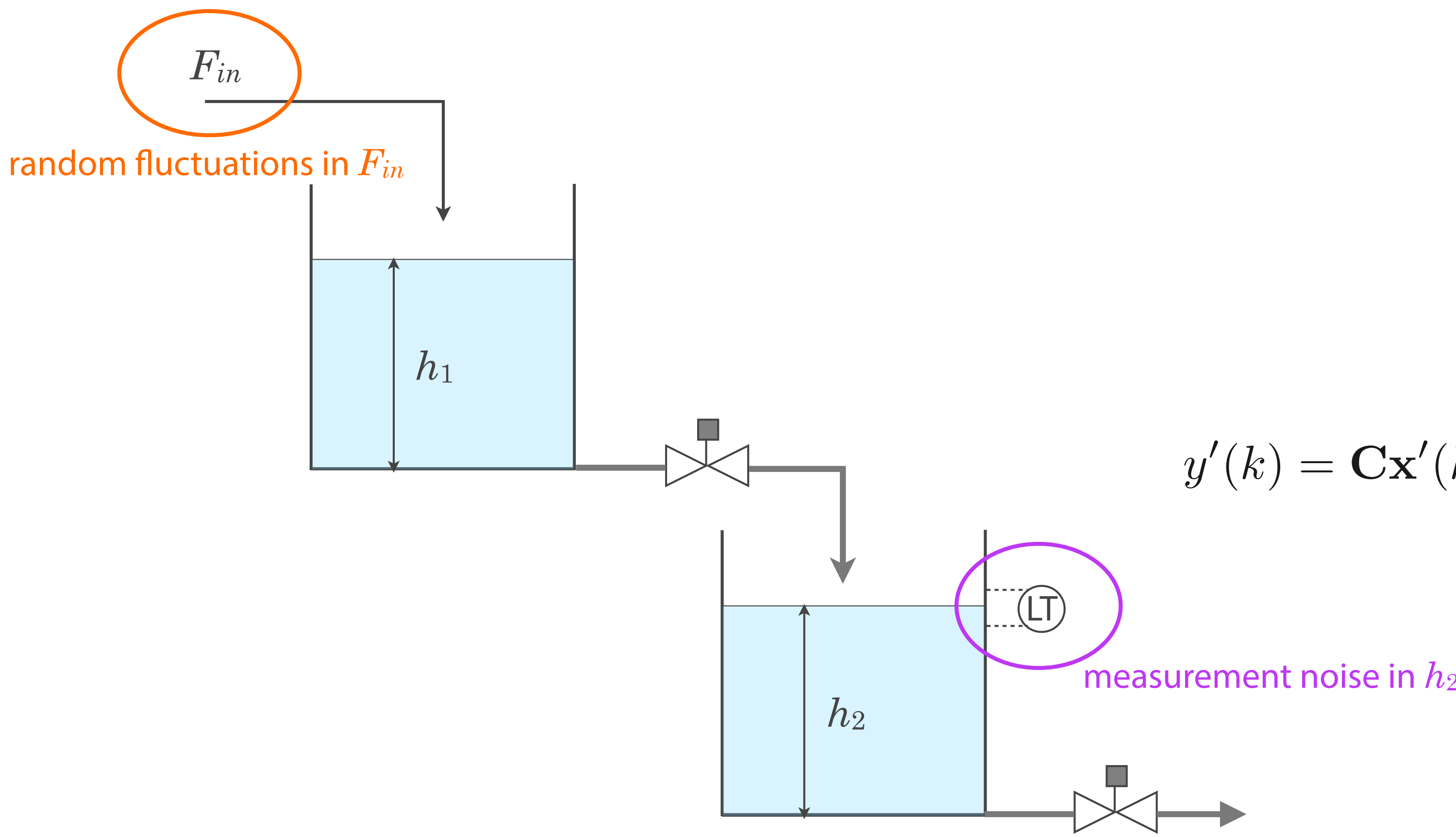
$$\xrightarrow[T_s = 0.1 \ [\mathrm{min}]]{}$$

$$\mathbf{x}'(k+1) = \begin{bmatrix} 0.8462 & 0 \\ 0.1363 & 0.7866 \end{bmatrix} \mathbf{x}'(k) + \begin{bmatrix} 0.3684 \\ 0.0292 \end{bmatrix} u'(k)$$
$$y'(k) = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{x}'(\mathbf{k})$$

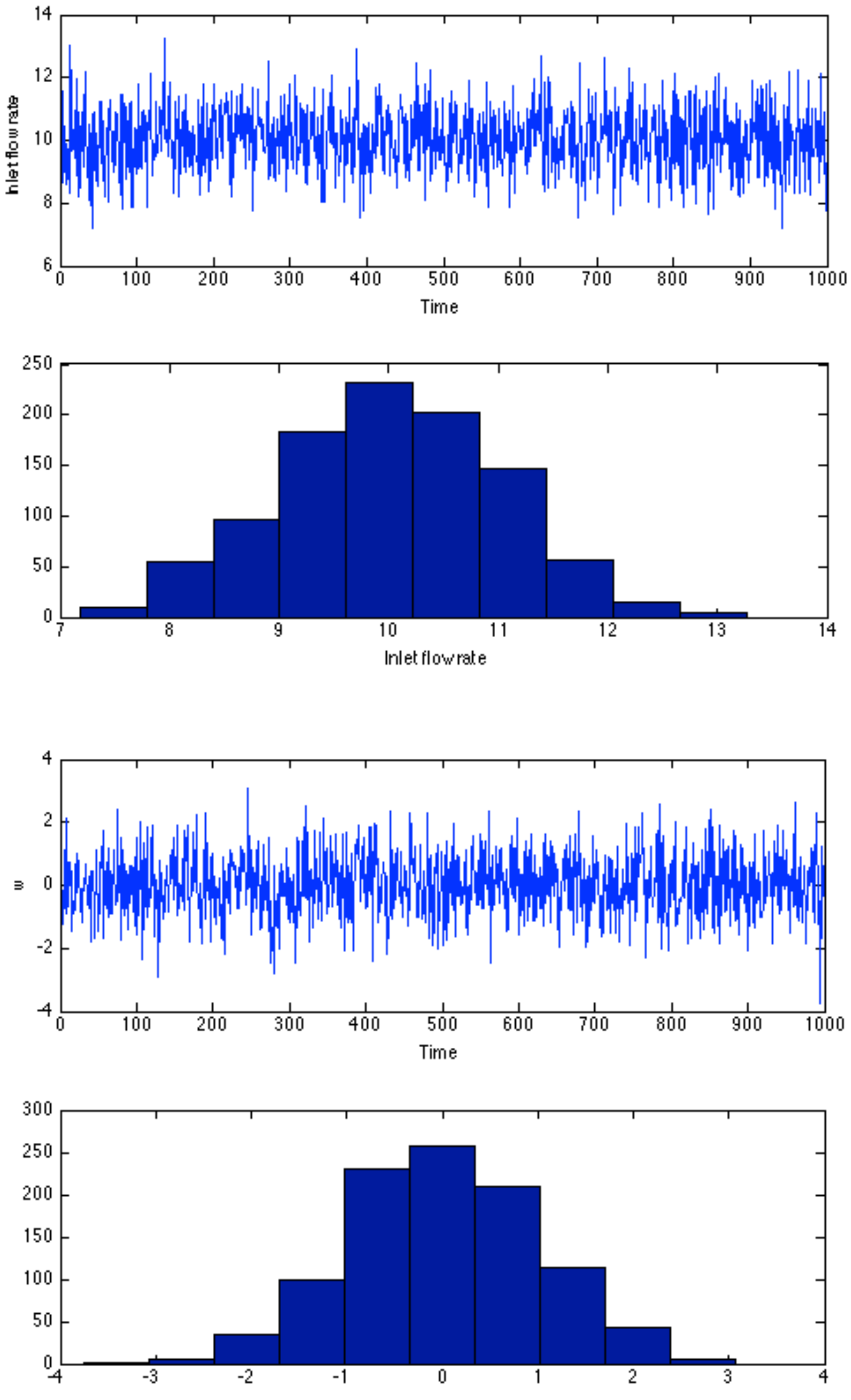
In reality,...

$F_{in} = F_{in}^d + F_{in}^s$

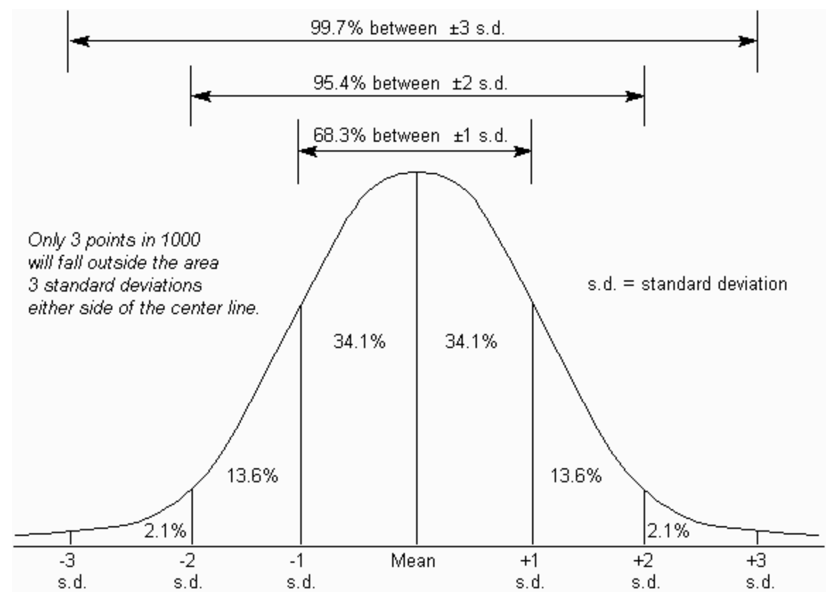
$$\mathbf{x}'(k+1) = \mathbf{\Phi}\mathbf{x}'(k) + \mathbf{\Gamma}u'(k) + \mathbf{\Gamma}w(k)$$



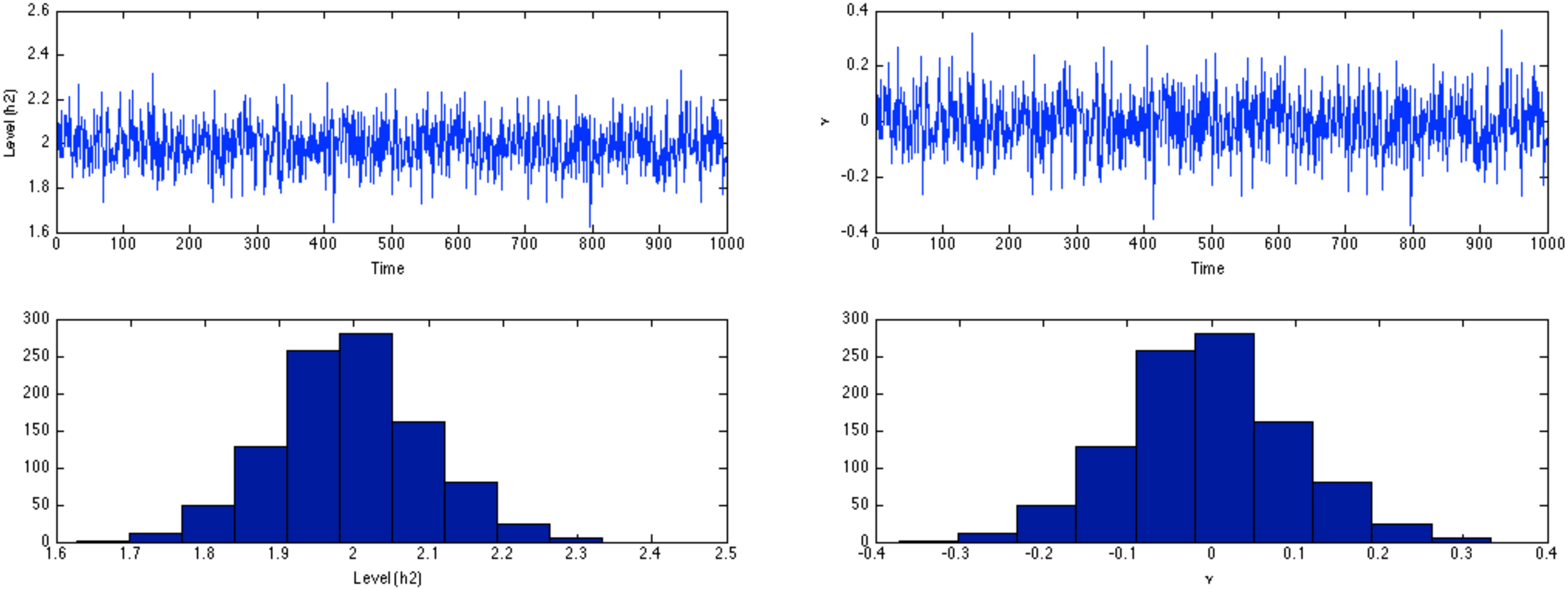
$$y'(k) = \mathbf{C}\mathbf{x}'(k) + \nu(k)$$



$$W \sim \mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left[\frac{(w-\mu)^2}{2\sigma^2}\right]}$$



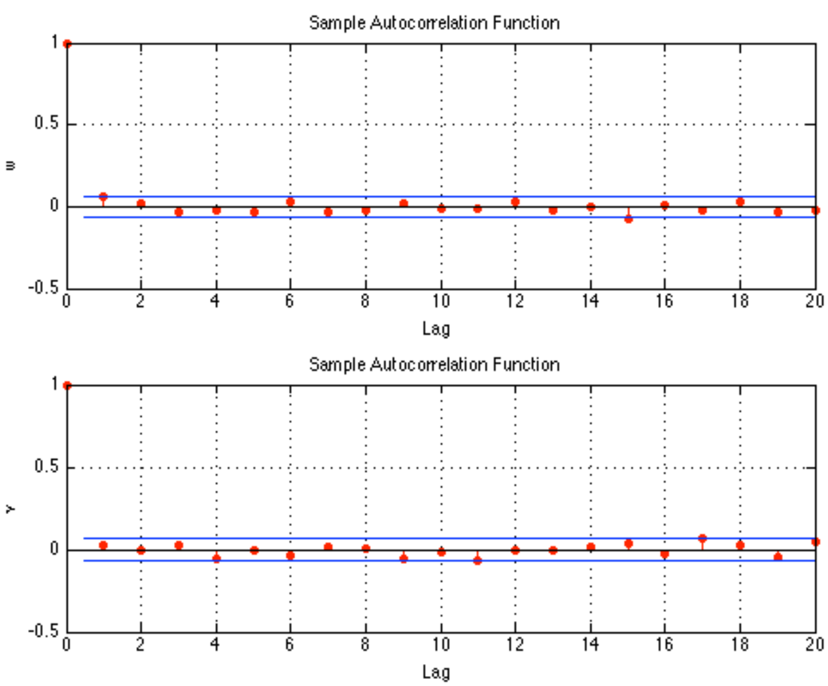
$$V \sim \mathcal{N}(\mu', \sigma'^2) = \frac{1}{\sqrt{2\pi\sigma'^2}} e^{-\left[\frac{(\nu - \mu')^2}{2\sigma'^2}\right]}$$

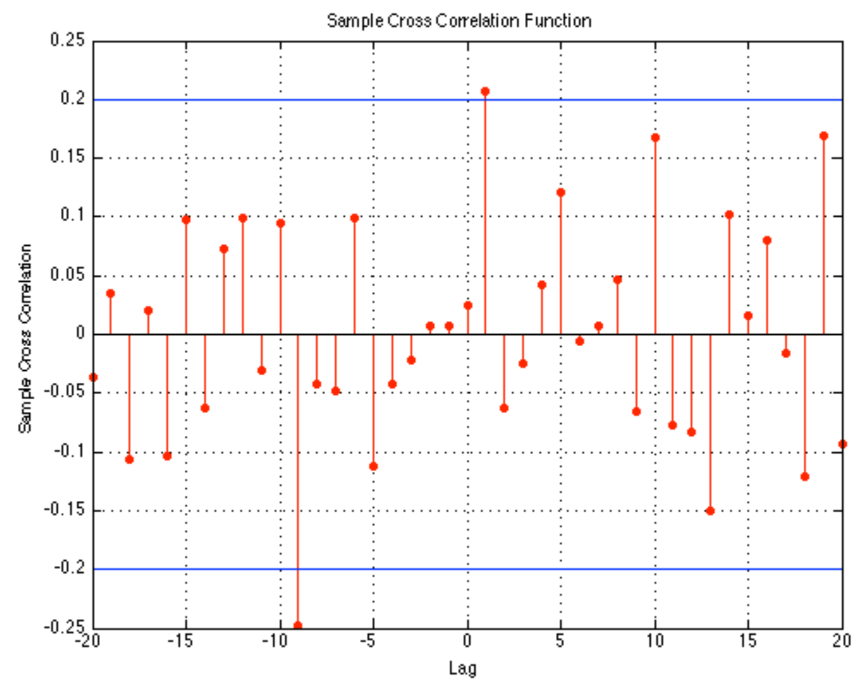


{w} and {v}: white noise sequence with zero-mean Gaussian

$$\mathbb{E}\left[w(k)\right] = 0$$
$$\mathbb{E}\left[\nu(k)\right] = 0$$
$$\text{cov}\left[w(k)\right] = Q$$
$$\text{cov}\left[\nu(k)\right] = R$$

$$\text{cov}\left[w(k), w(j)\right] = 0; j \neq k$$
$$\text{cov}\left[\nu(k), \nu(j)\right] = 0; j \neq k$$
$$\text{cov}\left[w(k), \nu(j)\right] = 0$$





Probability density function of vectors

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \mathcal{N}(\mu, P)$$

$$p(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left[\frac{-1}{2(1-\rho^2)}\,[z]\right]$$

$$z = \frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{(y-\mu_y)^2}{\sigma_y^2} - \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y}$$

$$\mu = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} \qquad P = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

Linear transformation

$$\mathbf{Z} = \mathbf{T}\mathbf{X}$$

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim \mathbf{N}(\mu, P)$$

$$P = E\Lambda E^T$$

$$\mathbf{Z} \sim \mathcal{N}(\mathbf{T}\mu,\ \mathbf{T}P\mathbf{T}^T)$$

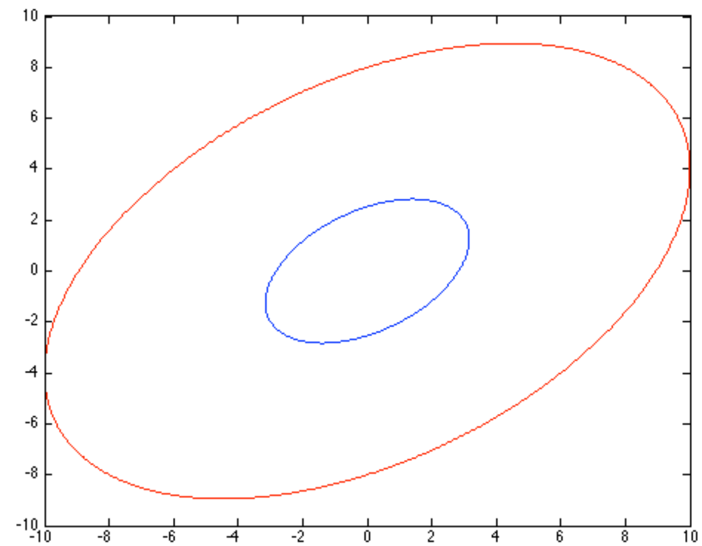
$$P = \begin{bmatrix} 100 & 40 \\ 40 & 80 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} 131.21 & 0 \\ 0 & 48.7 \end{bmatrix}$$

$$P = \begin{bmatrix} 10 & 4 \\ 4 & 8 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} 13.1 & 0 \\ 0 & 4.9 \end{bmatrix}$$

trace = 18 trace = 18

```
%%
% Drawing covariance ellipse
P1 = [100 40; 40 80];
P2 = [10 4; 4 8];
[E1, L1] = eig(P1);
[E2, L2] = eig(P2);

theta = 2*pi*[0:0.001:1];
for ii = 1:length(theta)
    y(:, ii) = [cos(theta(ii)); sin(theta(ii))];
    x1(:, ii) = E1*sqrt(L1)*y(:, ii);
    x2(:, ii) = E2*sqrt(L2)*y(:, ii);
end
figure
plot(x1(1, :), x1(2, :), 'r', 'LineWidth', 1);
hold on
plot(x2(1, :), x2(2, :), 'b', 'LineWidth', 1);
```



Drawing covariance ellipses

n-dim. multivariate-normal dist. $p(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\mathbf{P}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_x)^T\mathbf{P}^{-1}(\mathbf{x}-\mu_x)\right\}$

A particularly useful contour is 1-σ bound: $(\mathbf{x}-\mu_x)^T\mathbf{P}^{-1}(\mathbf{x}-\mu_x) = 1$

Assume zero-mean. $\mathbf{x}^T\mathbf{P}^{-1}\mathbf{x} = 1$ $\mathbf{x}^T(E\Lambda E^T)^{-1}\mathbf{x} = 1$

$$\mathbf{x}^T E\Lambda^{-1}E^T\mathbf{x} = 1$$

Orthonormality

$$\mathbf{x}^T E\Lambda^{-1/2}\Lambda^{-1/2}E^T\mathbf{x} = 1$$

$$\mathbf{x}^T\mathbf{K}\mathbf{K}^T\mathbf{x} = 1 \qquad \mathbf{K} = E\Lambda^{-1/2}$$

now for any point $\mathbf{y} = [x, y]^T$ which is on the unit circle, $\mathbf{y}^T\mathbf{y} = 1$, so

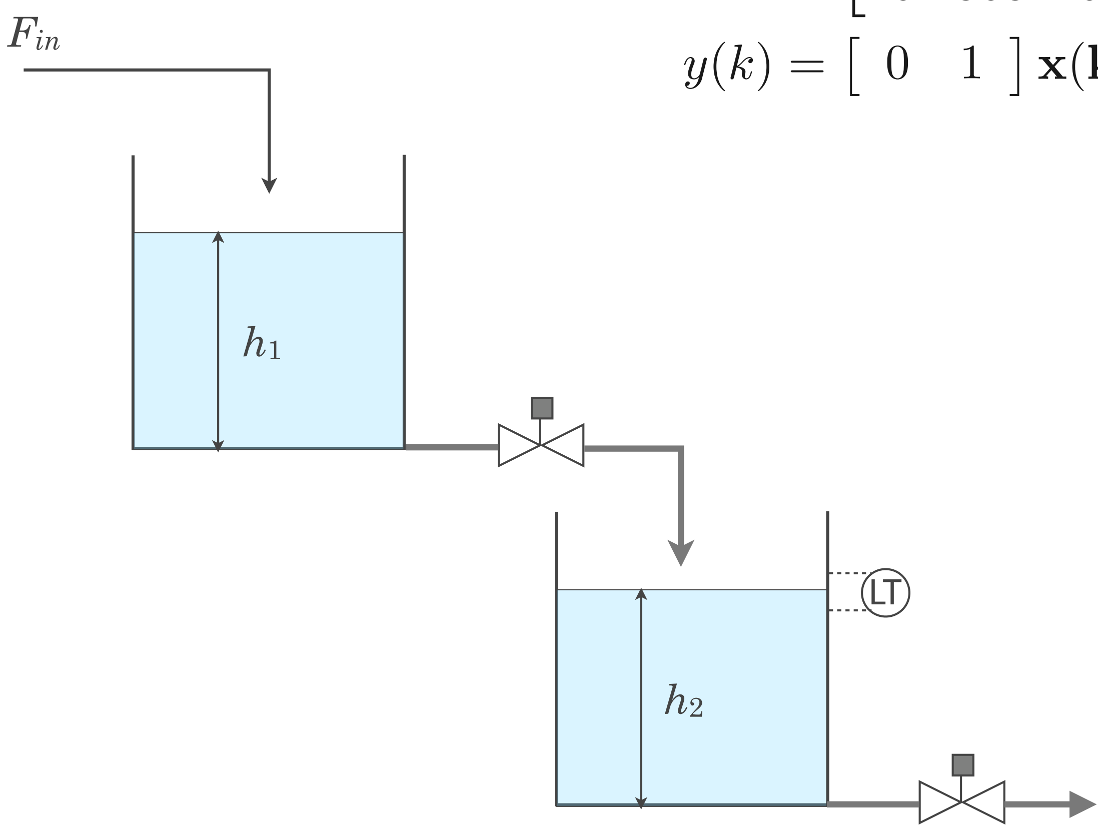
$$\mathbf{x}^T\mathbf{K}\mathbf{K}^T\mathbf{x} = \mathbf{y}^T\mathbf{y}$$

$$\mathbf{K}^T\mathbf{x} = \mathbf{y}$$

$$\Rightarrow \mathbf{x} = E\Lambda^{1/2}\mathbf{y}$$

$$\mathbf{x}(k+1) = \begin{bmatrix} 0.8462 & 0 \\ 0.1363 & 0.7866 \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} 0.3684 \\ 0.0292 \end{bmatrix} u(k) + \mathbf{w}(k)$$

$$y(k) = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{x(k)} + \nu(k)$$



Estimate of h_1

Filtered estimate of h_2

Can we extract information about h1 given only h2?

Observability

$$\text{rank of} \begin{bmatrix} \mathbf{C} \\ \mathbf{C\Phi} \\ \vdots \\ \mathbf{C\Phi}^{n-1} \end{bmatrix} = n = \text{state dimension}$$

Can we extract information about h1 given only h2?

$$\mathbf{C} = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_{11} & 0 \\ \Phi_{21} & \Phi_{22} \end{bmatrix}$$

$$\mathbf{C\Phi} = \begin{bmatrix} \Phi_{21} & \Phi_{22} \end{bmatrix}$$

$$O_b = \begin{bmatrix} 0 & 1 \\ \Phi_{21} & \Phi_{22} \end{bmatrix}$$

yes

Can we extract information about h2 given only h1?

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_{11} & 0 \\ \Phi_{21} & \Phi_{22} \end{bmatrix}$$

$$\mathbf{C\Phi} = \begin{bmatrix} \Phi_{11} & 0 \end{bmatrix}$$

$$O_b = \begin{bmatrix} 1 & 0 \\ \Phi_{11} & 0 \end{bmatrix}$$

no

Linear state estimation

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

$\{w(k)\}$ and $\{\nu(k)\}$ are stationary random processes with known statistical properties.

Objective: To find the conditional probability density function of the state (or) posterior density of the state

$$p\left[x(k)|Y^k\right]$$

Y^k : Set of all the available measurements up to time instant k

Non-linear state estimation

$$x(k+1) = F\left[x(k), u(k)\right] + w(k)$$
$$y(k) = H(x(k)) + \nu(k)$$

$\{w(k)\}$ and $\{\nu(k)\}$ are stationary random processes with known statistical properties.

Objective: To find the conditional probability density function of the state (or) posterior density of the state

$$p\left[x(k)|Y^k\right]$$

Y^k : Set of all the available measurements up to time instant k

Sequential estimation problem

$$P(H|D) = \frac{P(H) \times P(D|H)}{P(D)}$$

H: hypothesis, D: data

$p(H_t|H_{t-1}, D_t)$: this is what we are doing

Posterior density is estimated in two stages:

1. **Prediction step:** posterior density at previous time step is propagated into the next time step through the transition density

$$p\left[x(k)|Y^{k-1}\right] = \int p[x(k)|x(k-1)]p[x(k-1)|Y^{k-1}]dx(k-1)$$

2. **Update step:** involves application of Bayes' rule

$$p[x(k)|Y^k] \approx \underbrace{p[y(k)|x(k)]}_{\text{likelihood}} \times \underbrace{p[x(k)|Y^{k-1}]}_{\text{prior}}$$

Optimal state estimate

- Minimum mean square error (MMSE)
  - aimed at finding the conditional mean of x
- Maximum a posterior (MAP)
  - aimed at finding **mode** of posterior probability density function
- Measure of accuracy of a state estimate
  - covariance

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

$$p(w) = \mathcal{N}(0,\ R_w) \qquad p(\nu) = \mathcal{N}(0,\ R_\nu)$$

$$p\left[x(k-1)|Y^{k-1}\right] = \mathcal{N}\left[\hat{x}(k-1|k-1), P(k-1|k-1)\right]$$

- initial state distribution is assumed to be normal

$$p\left[x(k)|Y^{k-1}\right] = \mathcal{N}\left[\hat{x}(k|k-1), P(k|k-1)\right]$$

- prior

$$p\left[x(k)|Y^{k}\right] = \mathcal{N}\left[\hat{x}(k|k), P(k|k)\right]$$

- posterior are Gaussian w/ system being linear and noise being Gaussian

$$p\left[x(k)|Y^k\right] \propto p[y(k)|x(k)]p\left[x(k)|Y^{k-1}\right]$$

$$p\left[x(k)|Y^{k-1}\right] = \frac{1}{(2\pi)^{n/2}|P|^{1/2}} \exp\left[-\frac{1}{2}\left\{x(k) - \hat{x}(k|k-1)\right\}^T P(k|k-1)^{-1}\left\{x(k) - \hat{x}(k|k-1)\right\}\right]$$

$$\xi(k) \triangleq x(k) - \hat{x}(k|k-1)$$

$$p\left[x(k)|Y^{k-1}\right] = \frac{1}{(2\pi)^{n/2}|P|^{1/2}} \exp\left[-\frac{1}{2}\xi^T(k)P(k|k-1)^{-1}\xi(k)\right]$$

$$p\left[y(k)|x(k)\right] = \frac{1}{(2\pi)^{n/2}|R_\nu|^{1/2}} \exp\left[-\frac{1}{2}\left\{y(k) - Cx(k)\right\}^T R_\nu^{-1}\left\{y(k) - Cx(k)\right\}\right]$$

$$e(k) \triangleq y(k) - Cx(k)$$

$$p\left[y(k)|x(k)\right] = \frac{1}{(2\pi)^{n/2}|R_\nu|^{1/2}} \exp\left[-\frac{1}{2}e^T(k)R_\nu^{-1}e(k)\right]$$

$$p\left[x(k)|Y^k\right] \propto \exp\left[-\frac{1}{2}\left\{e^T(k)R_\nu^{-1}e(k) + \xi^T(k)P(k|k-1)^{-1}\xi(k)\right\}\right]$$

$$\min_{x(k)} J = \min_{x(k)} \frac{1}{2} \left[e^T(k) R_\nu^{-1} e(k) + \xi^T(k) P(k|k-1)^{-1} \xi(k) \right]$$

$$e(k) = y(k) - Cx(k) \qquad \xi(k) = x(k) - \hat{x}(k|k-1)$$

$$\boxed{\frac{\partial J}{\partial x(k)} = \left[-C^T R_\nu^{-1} e(k) + P^{-1} \xi(k) \right] = 0}$$

$$\left[-C^T R_\nu^{-1} \{ y(k) - Cx(k) \} + P^{-1} \{ x(k) - \hat{x}(k|k-1) \} \right] = 0$$

$$-C^T R_\nu^{-1} y + C^T R_\nu^{-1} C x + P^{-1} x - P^{-1} \hat{x} = 0$$

$$\boxed{\left[P^{-1} + C^T R_\nu^{-1} C \right] x = P^{-1} \hat{x} + C^T R_\nu^{-1} y}$$

$$\left[P^{-1} + C^T R_\nu^{-1} C \right] x = P^{-1} \hat{x} + C^T R_\nu^{-1} y - C^T R_\nu^{-1} C \hat{x} + C^T R_\nu^{-1} C \hat{x}$$

$$\left[P^{-1} + C^T R_\nu^{-1} C \right] x = \left[P^{-1} + C^T R_\nu^{-1} C \right] \hat{x} + C^T R_\nu^{-1} \left[y - C\hat{x} \right]$$

$$x = \hat{x} + \left[P^{-1} + C^T R_\nu^{-1} C \right]^{-1} C^T R_\nu^{-1} \left[y - C\hat{x} \right]$$

$$\boxed{\hat{x}(k|k) = \hat{x}(k|k-1) + K \{ y - C\hat{x}(k|k-1) \}}$$

$$K = \left[P^{-1} + C^T R_\nu^{-1} C \right]^{-1} C^T R_\nu^{-1}$$

$$K = \left[P^{-1} + C^T R_\nu^{-1} C\right]^{-1} C^T R_\nu^{-1}$$

$$[A + BCD]^{-1} = A^{-1} - A^{-1} B \left[C^{-1} + DA^{-1} B\right]^{-1} DA^{-1}$$

$$\left[P^{-1} + C^T R_\nu^{-1} C\right]^{-1} = P - PC^T \left[R_\nu + CPC^T\right]^{-1} CP$$

$$K = \left[P - PC^T \left[R_\nu + CPC^T\right]^{-1} CP\right] C^T R_\nu^{-1}$$

$$K = \underline{P(k|k-1)} C^T \left[CP(k|k-1)C^T + R_\nu\right]^{-1}$$

second moment

Note $z = \Phi X$ $X \sim \mathcal{N}(\mu, P)$ $z \sim \mathcal{N}(\Phi\mu, \Phi P \Phi^T)$

$$x(k) = A\underline{x(k-1)} + Bu(k-1) + w(k)$$

$$\mathcal{N}\left(\hat{x}(k-1|k-1), P(k-1|k-1)\right)$$

Prediction:

$$\therefore \quad \hat{x}(k|k-1) = A\hat{x}(k-1|k-1) + Bu(k-1) + 0$$

$$P(k|k-1) = AP(k-1|k-1)A^T + R_w$$

Update: posterior ≃ prior x likelihood

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

$$p\left[x(k)|Y^{k-1}\right] = \mathcal{N}\left[\hat{x}(k|k-1), P(k|k-1)\right]$$

$$\begin{aligned}\mathbb{E}\left[x(k)|Y^{k-1}\right] &= \hat{x}(k|k-1)\\ &= \mathbb{E}\left[\,Ax(k-1) + Bu(k-1) + w(k-1)|\,Y^{k-1}\right]\end{aligned}$$

$$\boxed{\hat{x}(k|k-1) = A\hat{x}(k-1|k-1) + Bu(k-1)}$$

$$P(k|k-1) = \operatorname{cov}\left[\xi(k|k-1)\right] = \mathbb{E}\left[\xi(k|k-1)\xi(k|k-1)^T\right]$$

$$\begin{aligned}\xi(k|k-1) &= x(k) - \hat{x}(k|k-1)\\ &= \left[Ax(k-1) + Bu(k-1) + w(k-1)\right] - \left[A\hat{x}(k-1|k-1) + Bu(k-1)\right]\\ &= \left[A\xi(k-1|k-1) + w(k-1)\right]\end{aligned}$$

$$
\begin{aligned}
\mathbb{E}\left[\xi(k|k-1)\xi(k|k-1)^T\right] &= \mathbb{E}\left[\left\{A\xi(k-1|k-1)+w(k-1)\right\}\left\{A\xi(k-1|k-1)+w(k-1)\right\}^T\right] \\
&= \mathbb{E}\left[A\xi(k-1|k-1)\xi^T(k-1|k-1)A^T\right] + \mathbb{E}\left[w(k-1)w(k-1)^T\right]
\end{aligned}
$$

$$
P(k|k-1) = AP(k-1|k-1)A^T + R_w
$$

Why is *K* a fcn of two covariance matrices?

$$\mathbb{E}[y(k)|Y^{k-1}] = \hat{y}(k|k-1) = CA\hat{x}(k-1|k-1) + CBu(k-1)$$

$$\begin{aligned} e(k|k-1) &= y(k) - \hat{y}(k|k-1) \\ &= [CAx(k-1) + CBu(k-1) + Cw(k-1) + \nu(k)] - [CA\hat{x}(k-1|k-1) + CBu(k-1)] \end{aligned}$$

$$e(k|k-1) = [CA\xi(k-1|k-1) + Cw(k-1) + \nu(k)]$$

$$e(k|k-1) = [C\xi(k|k-1) + \nu(k)]$$

$$\begin{aligned} \text{cov}\left[e(k|k-1)\right] &= \mathbb{E}\left[\{C\xi(k|k-1) + \nu(k)\}\{C\xi(k|k-1) + \nu(k)\}^T\right] \\ &= CP(k|k-1)C^T + R_\nu \end{aligned}$$

$$\begin{aligned} \text{cov}\left[\xi(k|k-1)e^T(k|k-1)\right] &= \mathbb{E}\left[\xi(k|k-1)\{C\xi(k|k-1) + \nu(k)\}^T\right] \\ &= P(k|k-1)C^T \end{aligned}$$

$$\begin{aligned} K &= \text{cov}\left[\xi(k|k-1)e^T(k|k-1)\right]\left[e(k|k-1)\right]^{-1} \\ &= P(k|k-1)C^T\left[CP(k|k-1)C^T + R_\nu\right]^{-1} \end{aligned}$$

$$p[x(k)|Y^k] = \mathcal{N}\left[\hat{x}(k|k), P(k|k)\right]$$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K\left[y(k) - C\hat{x}(k|k-1)\right]$$

$$P(k|k) = \text{cov}\left[\xi(k|k)\right] = \mathbb{E}\left[\xi(k|k)\xi(k|k)^T\right]$$

$$\begin{aligned}
\xi(k|k) &= x(k) - \hat{x}(k|k) \\
&= \left[Ax(k-1) + Bu(k-1) + w(k-1)\right] - \left[A\hat{x}(k-1|k-1) + Bu(k-1) + Ke(k|k-1)\right] \\
&= A\xi(k-1|k-1) + w(k-1) - Ke(k|k-1) \\
&= \xi(k|k-1) - Ke(k|k-1)
\end{aligned}$$

$$\begin{aligned}
P(k|k) &= \mathbb{E}\left[\xi(k|k)\xi(k|k)^T\right] \\
&= \mathbb{E}\left[\left\{\xi(k|k-1) - Ke(k|k-1)\right\}\left\{\xi(k|k-1) - Ke(k|k-1)\right\}^T\right] \\
&= P(k|k-1) + K\text{cov}\left[e(k|k-1)\right]K^T - PC^TK^T - KCP \\
&= P(k|k-1) + P(k|k-1)C^T\left[CP(k|k-1)C^T + R_\nu\right]^{-1}\left[CP(k|k-1)C^T + R_\nu\right]K^T - P(k|k-1)C^TK^T - KCP(k|k-1) \\
&= P(k|k-1) - KCP(k|k-1) = [I - KC]P(k|k-1)
\end{aligned}$$

Kalman filter

Prediction step $\mathcal{N}\left[\hat{x}(k|k-1), P(k|k-1)\right]$

$$\hat{x}(k|k-1) = A\hat{x}(k-1|k-1) + Bu(k-1)$$

$$P(k|k-1) = AP(k-1|k-1)A^T + R_w$$

Correction step $\mathcal{N}\left[\hat{x}(k|k), P(k|k)\right]$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + Ke(k|k-1)$$

$$P(k|k) = [I - KC]P(k|k-1)$$

$$K = P(k|k-1)C^T\left[CP(k|k-1)C^T + R_\nu\right]^{-1}$$

Kalman Gain Computation

$$e(k|k-1) = [y(k) - C\hat{x}(k|k-1)]$$

(innovations)

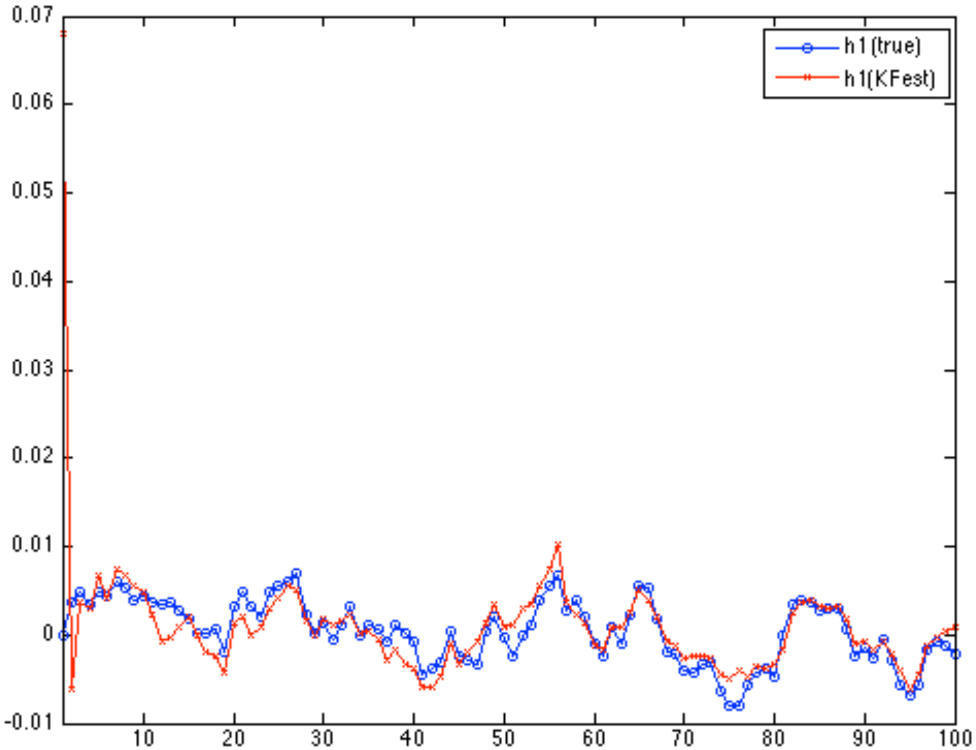
$$\xi(k) = x(k) - \hat{x}(k|k-1)$$

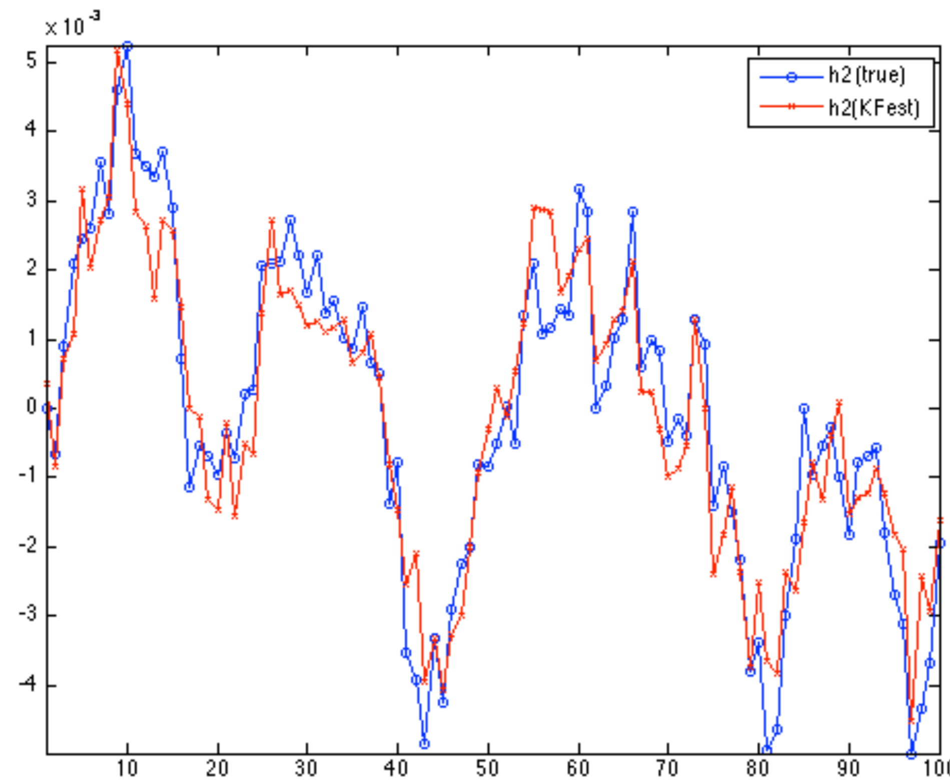
(estimation error)

$$K(k) = P_{\xi,e}\left[P_{e,e}(k)\right]^{-1}$$

(covariance matrices)

Simulation results





Issues in state estimation

- Generates the maximum likelihood (or minimum variance) estimates of the states when noises are Gaussian and the system is linear
- Model accuracy is critical to state estimation
- Noise model parameters: measurement and state noise covariance matrices (Q and R) are difficult to estimate. These matrices are typically used as tuning parameters

Difficulties with Kalman filter

- Linear perturbation model: range of applicability limited to a narrow region around an operating point
- Cannot be used for
  - strongly nonlinear systems or operation over wide operating range
  - batch/semi-batch processes
- State estimation using nonlinear dynamic model:
  - Extended Kalman filter (EKF)
  - Ensemble Kalman filter (EnKF)
  - Unscented Kalman filter (UKF)