

Lecture 9b

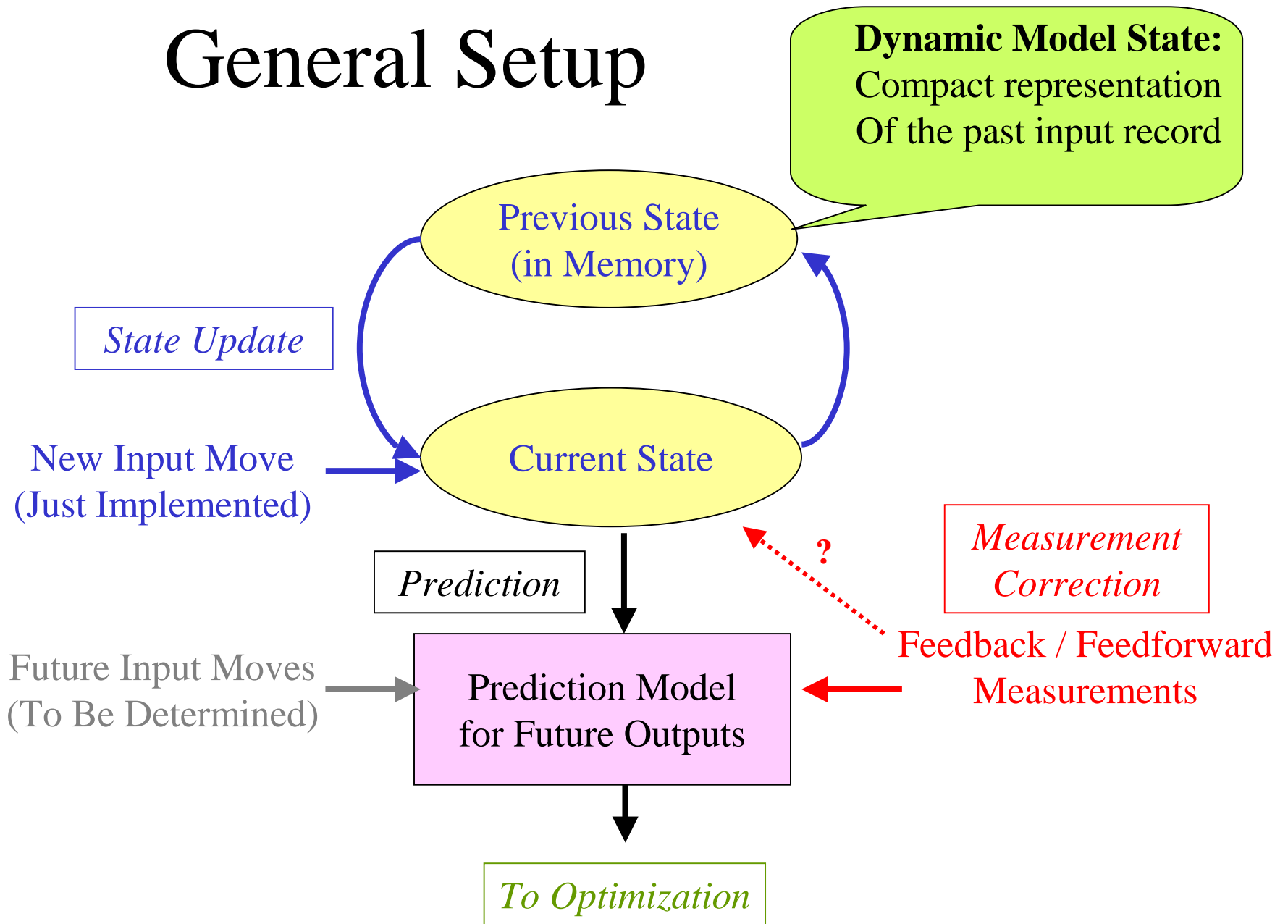
Some Details and Theories of Model Predictive Control

Professor Jay H. Lee
Georgia Inst. Technology
©Jay H. Lee, 2001

Important Ingredients of MPC Algorithm

- Dynamic Model \rightarrow Prediction Model
 - Predicted future outputs = Function of current “state” (stored in memory) + feedforward measurement + feedback measurement correction + future input adjustments
- Objective and Constraints
- Optimization Algorithm
- Receding Horizon Implementation

General Setup



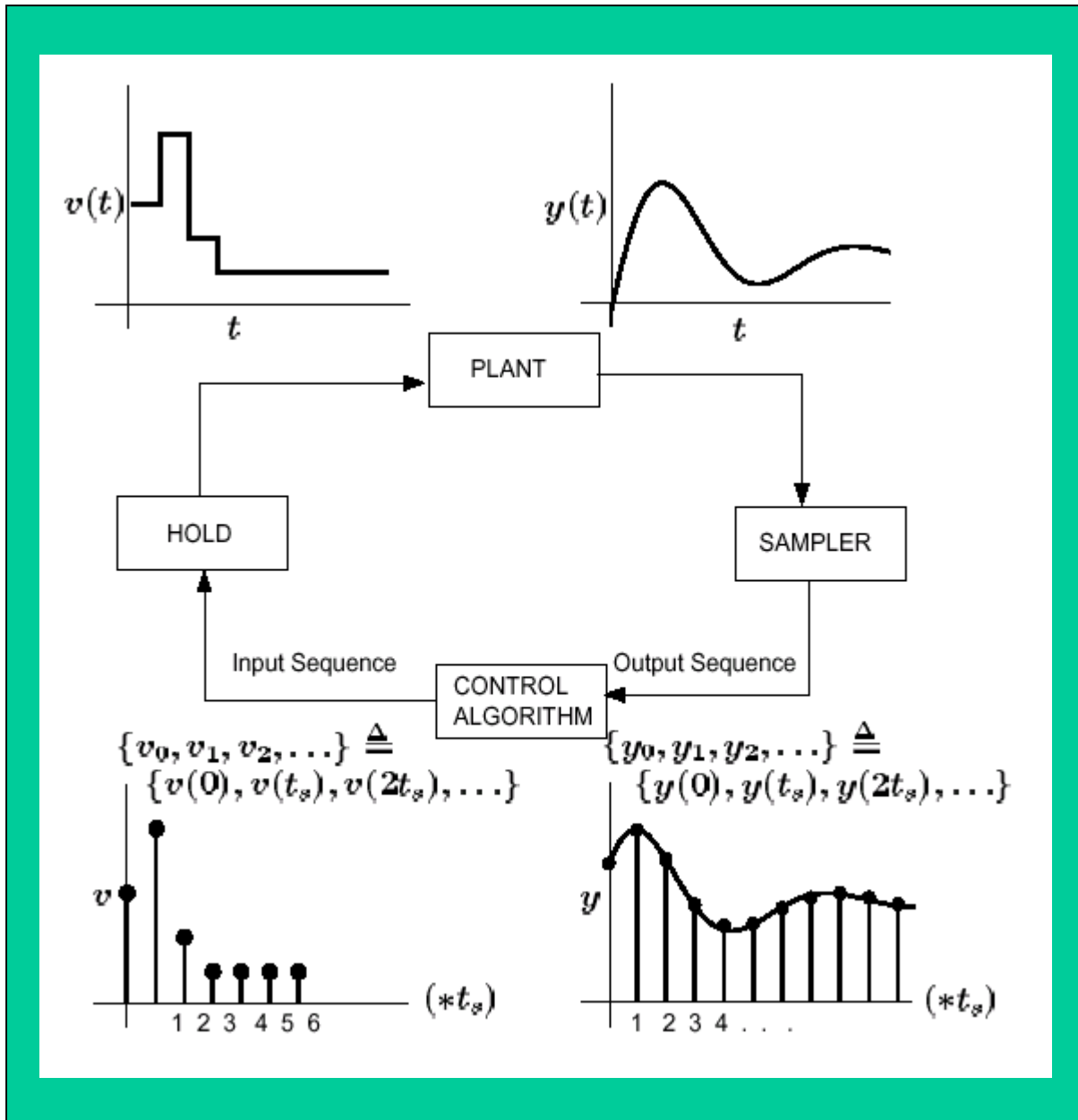
Options

- Model Types
 - Finite Impulse Response Model or Step Response Model
 - State-Space Model
 - Linear or Nonlinear
- Measurement Correction
 - To the prediction (based on open-loop state calculation)
 - To the state (through state estimation)
- Objective Function
 - Linear or Quadratic
 - Constrained or Unconstrained

Prediction Model for Different Model Types

- Finite Impulse Response Model
- Step Response Model
- State-Space Model

Sample-Data (Computer) Control



Model relates
input samples to
output samples

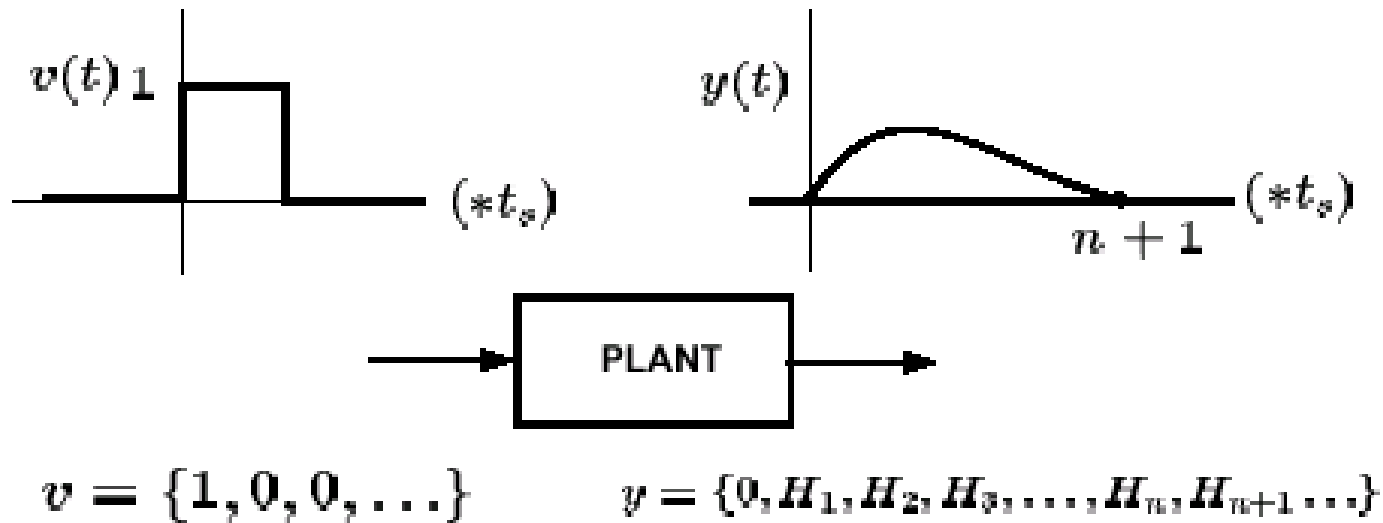
$$\{v_0, v_1, v_2, \dots\}$$

$$\downarrow$$

$$\{y_0, y_1, y_2, \dots\}$$

v can be a MV
(u) or a
Measured DV
(d)

Finite Impulse Response Model (1)

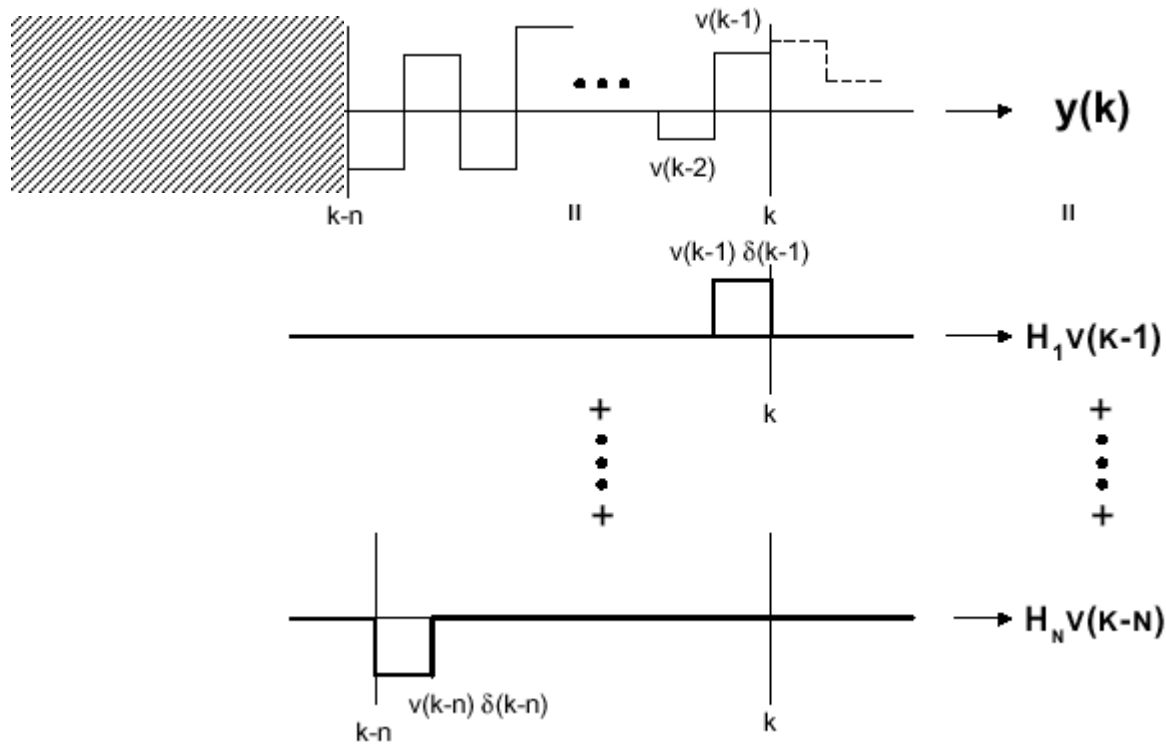


Assumptions:

- $H_0 = 0$: no **immediate** effect
- The response settles back in n steps s.t. $H_{n+1} = H_{n+2} = \dots = 0$: “Finite Impulse Response” (reasonable for stable processes).

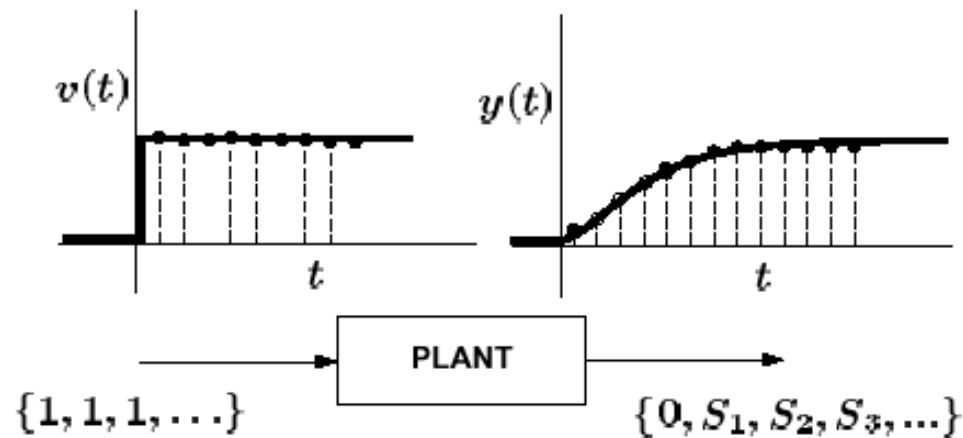
Finite Impulse Response Model (2)

Linear Model → “Superposition Principle”



$$y(k) = H_1 v(k-1) + \dots + H_n v(k-n)$$

Step Response Model (1)



Assumptions:

- $S_0 = 0$: no **immediate** effect
- The response settles in n steps s.t. $S_n = S_{n+1} = \dots = S_\infty$: the same as the finite impulse response assumption
- Relationship with the impulse response coefficients:

$$\begin{aligned} S_k &= \sum_{i=1}^k H_i \\ H_k &= S_k - S_{k-1} \end{aligned}$$

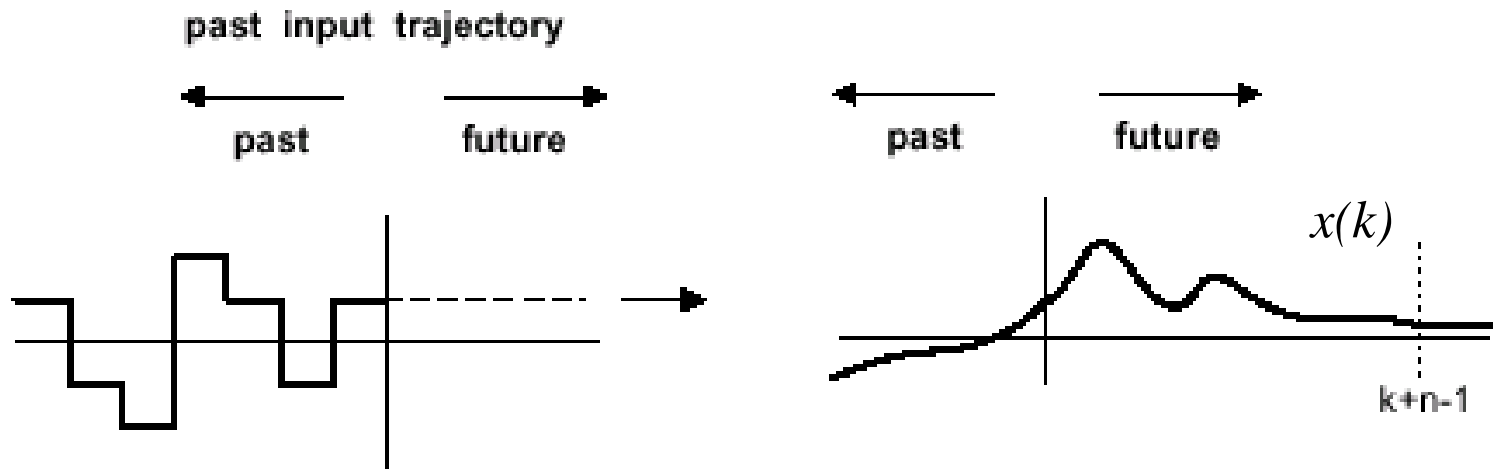
Step Response Model (2)

•State

$$x(k) = \left[y_0^T(k), \dots, y_{n-1}^T(k) \right]^T$$

n future outputs assuming the input remains constant at the most recent value.

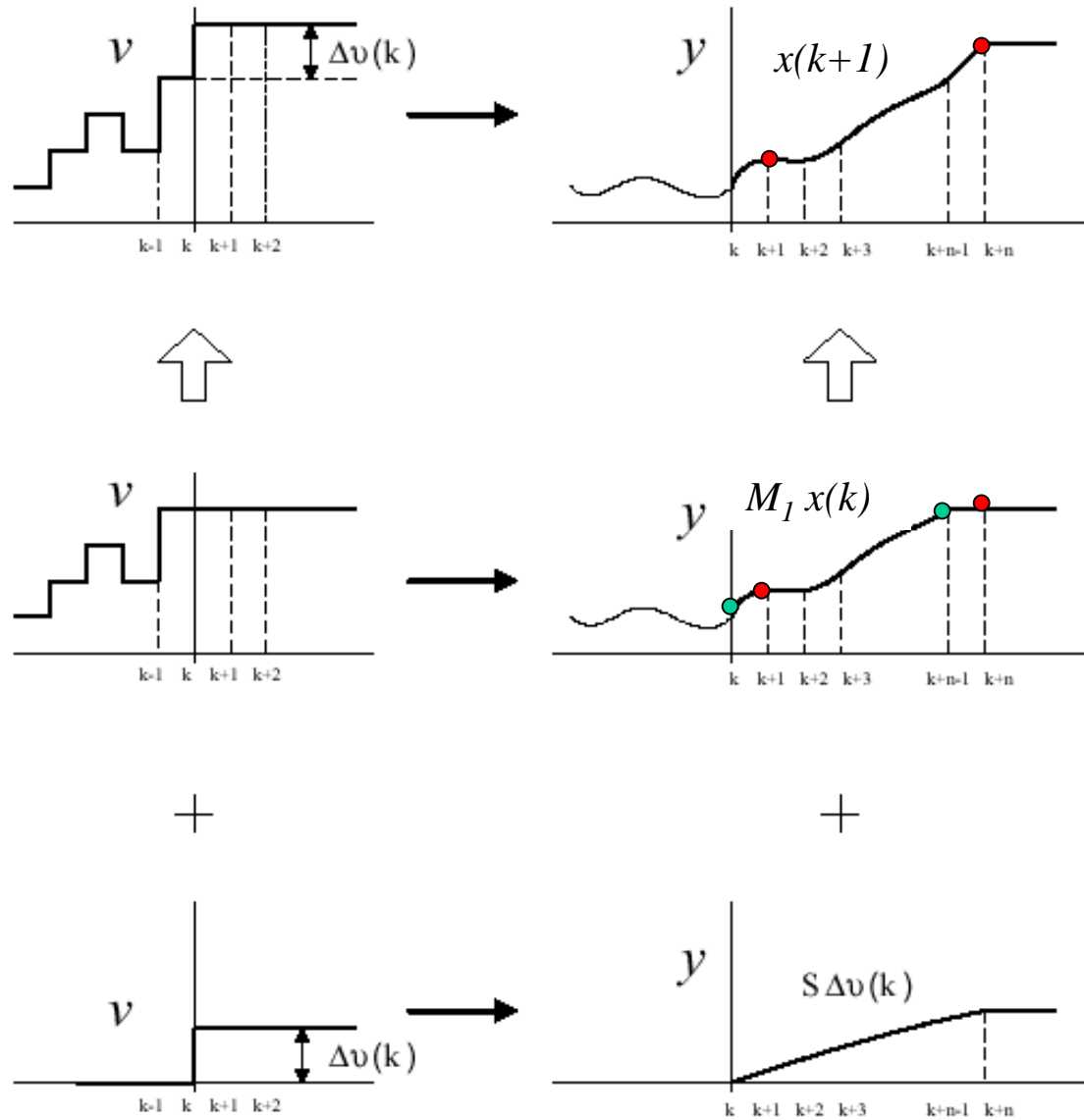
$$y_i(k) = y(k+i) \text{ w/ } \Delta u(k) = \Delta u(k+1) = \dots = 0$$



Note $y_{n-1}(k) = y_n(k) = \dots = y_\infty(k)$

Also $y(k) = y_0(k)$

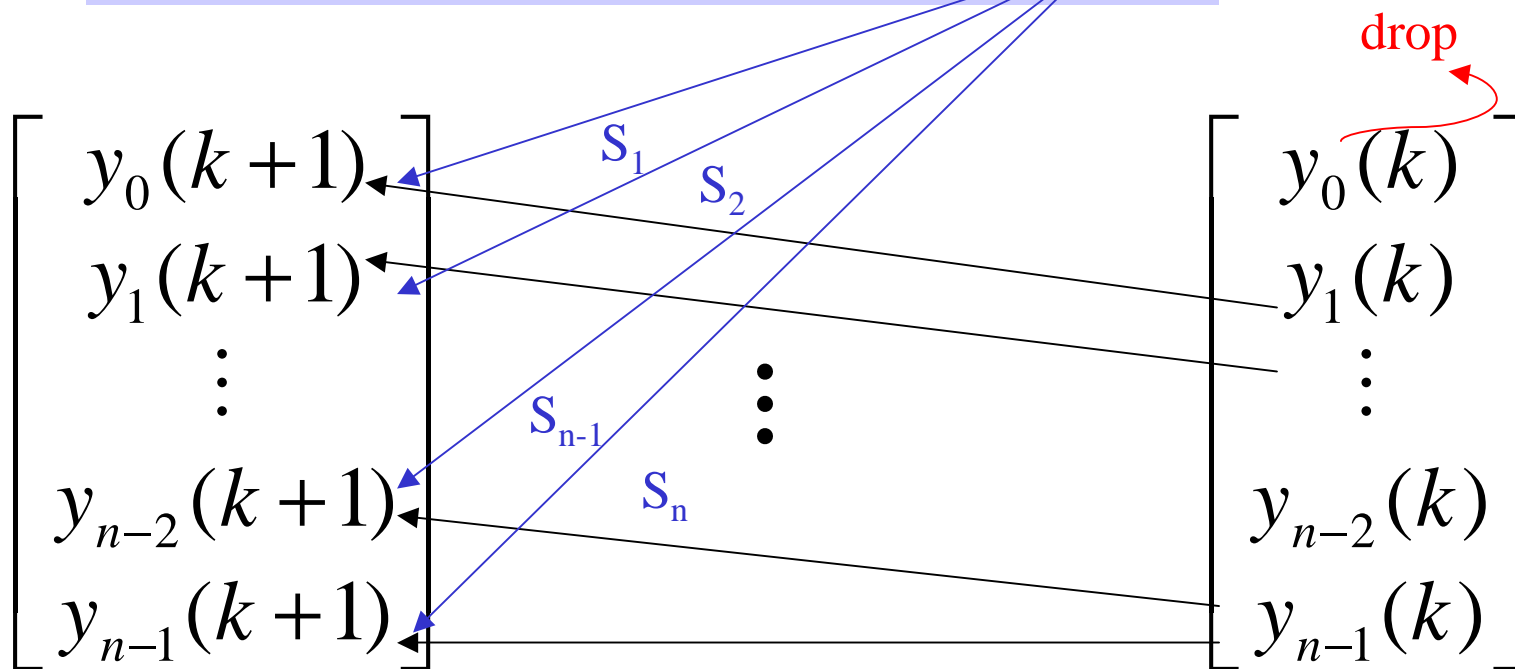
Pictorial Representation of The State Update



Step Response Model (3)

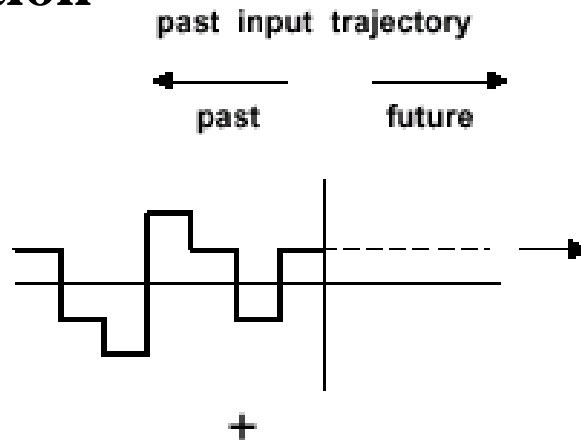
- **State Update**

$$x(k+1) = \underbrace{M_1}_{\text{shift}} x(k) + \underbrace{\sum}_{\text{step response}} \Delta v(k)$$

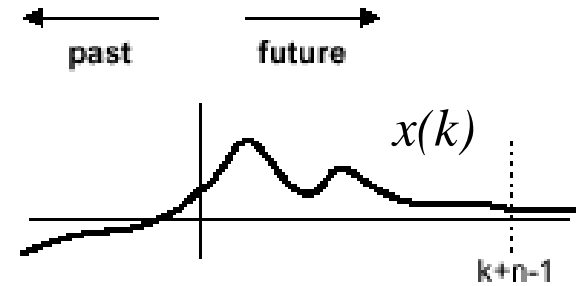
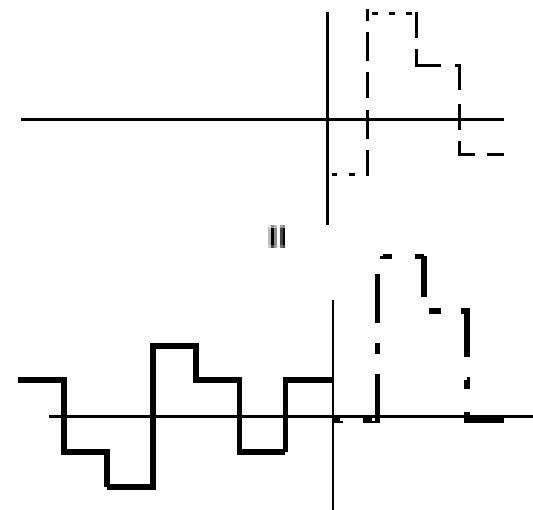


Step Response Model (4)

•Prediction

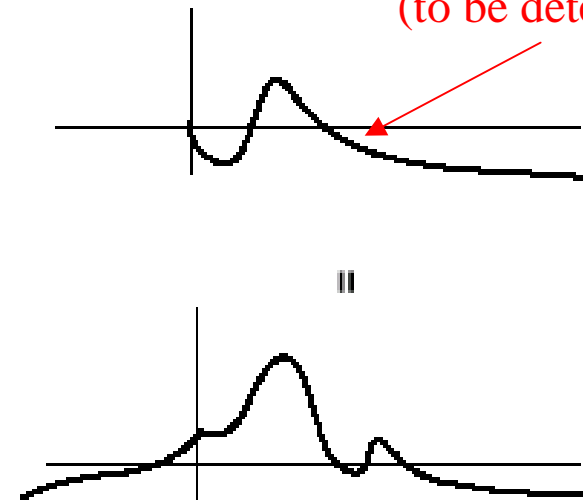


hypothesized future input trajectory



+

Effect of future input moves
(to be determined)

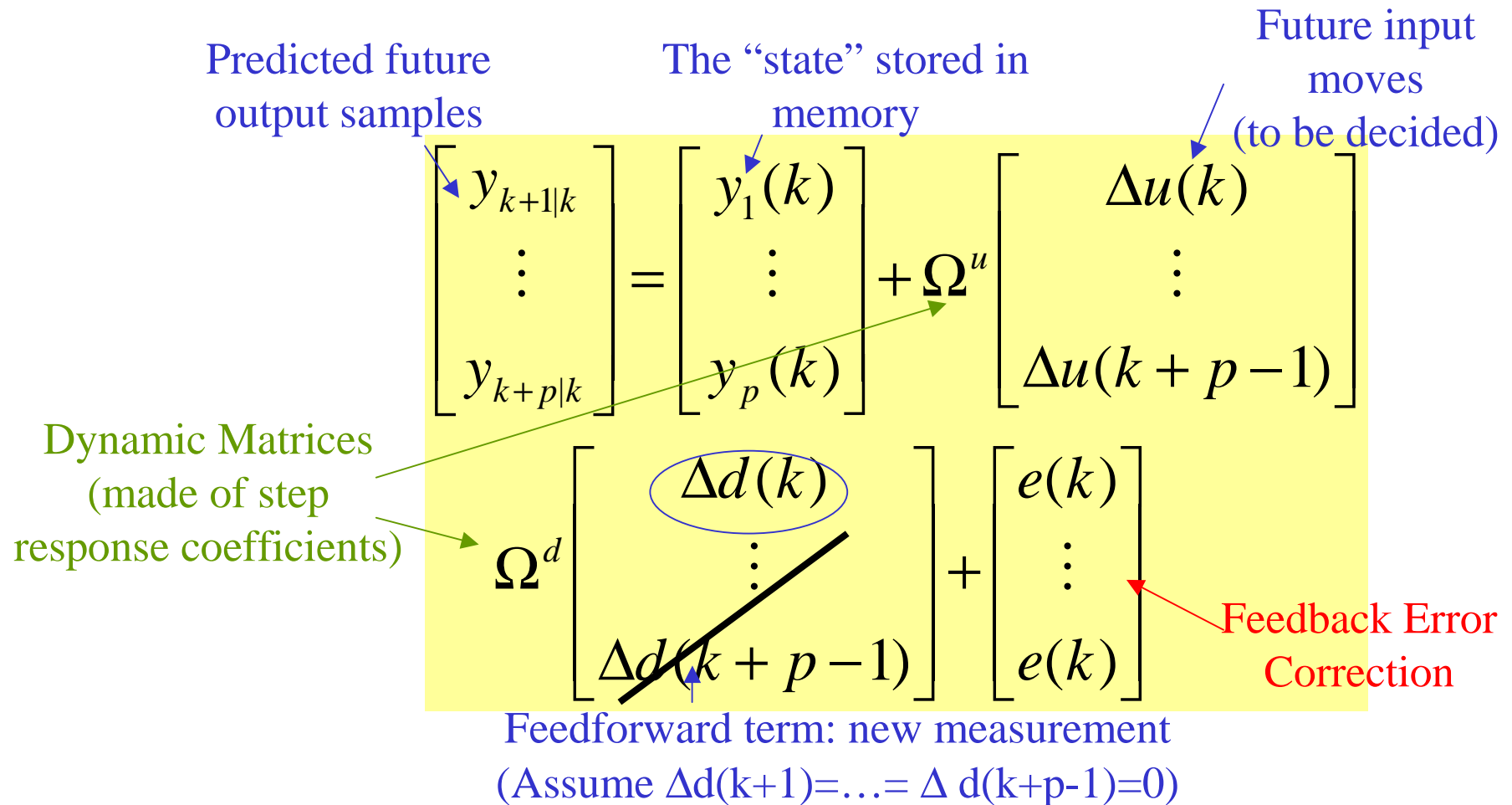


Step Response Model (4)

- Prediction**

Model prediction of $y(k) \rightarrow y(k) = y_0(k) = [1, 0, \dots, 0]x(k)$

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



Summary

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

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

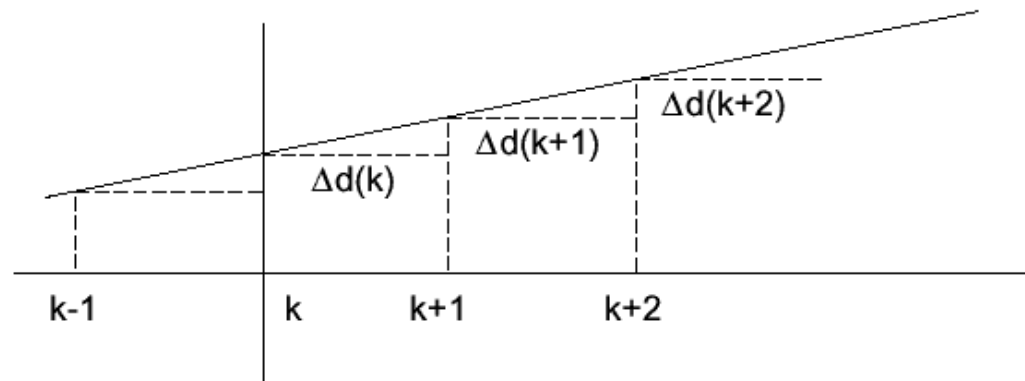
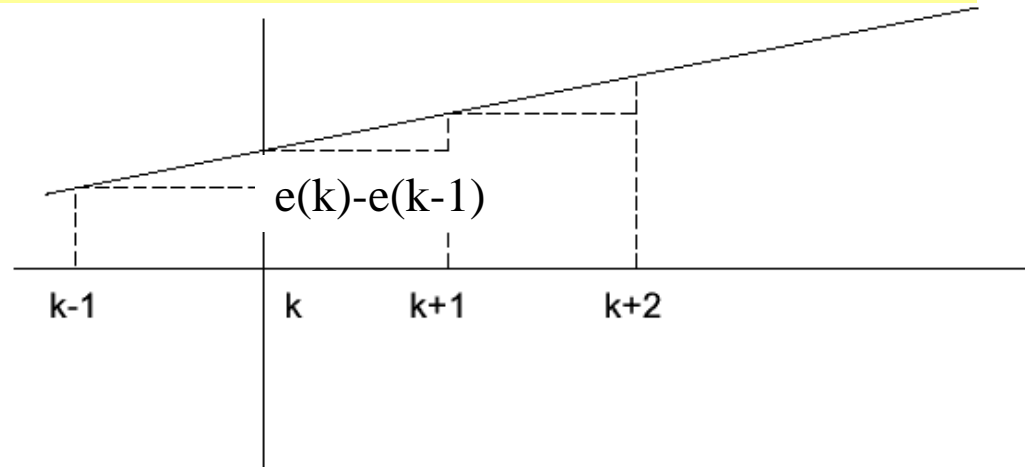
- Assumptions
 - Measured DV (d) remains constant at the current value of $d(k)$
 - Model prediction error (e) remains constant at the current value of $e(k)$

Ramp Type Extrapolation

- For Integrating Processes, Slow Dynamics

$$e(k+i) = e(k) + i(e(k) - e(k-1))$$

$$\Delta d(k) = \Delta d(k+1) = \dots = \Delta d(k+p-1)$$



Optimization

Objective Function

- Minimization Function: Quadratic cost (as in DMC)

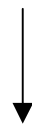
$$V(k) = \sum_{i=1}^p (y_{k+i|k} - y^*)^T \Lambda^y (y_{k+i|k} - y^*) + \sum_{i=0}^{m-1} \Delta u^T(k+i) \Lambda^u \Delta u(k+i)$$

- 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

- Use the prediction equation.

First m columns of L^u

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



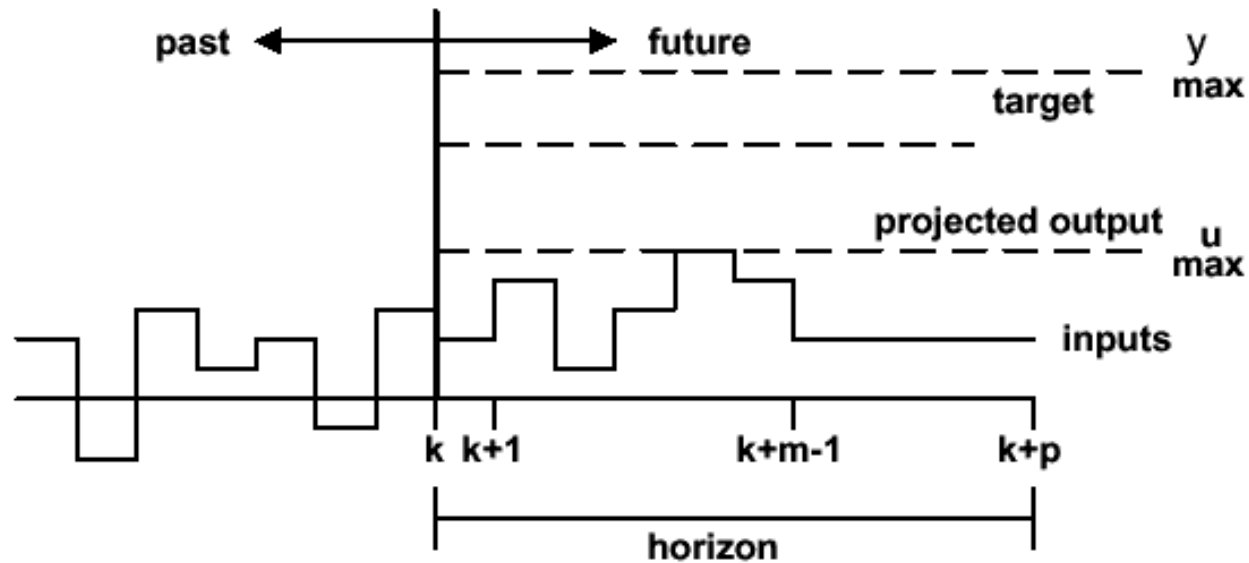
Substitute

$$Y(k) = b(k) + L_m^u \Delta U_m(k)$$

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

constant

Constraints



$$\begin{aligned}
 u_{\min} &\leq u(k + \ell | k) \leq u_{\max} \\
 |\Delta u(k + \ell | k)| &\leq \Delta u_{\max}, \quad \ell = 0, \dots, m - 1 \\
 y_{\min} &\leq y(k + j | k) \leq y_{\max}, \quad j = 1, \dots, p
 \end{aligned}$$

Substitute the prediction equation and rearrange to

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

Optimization Problem

- Quadratic Program

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

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.

Two-Level Optimization

Steady-State Optimization (Linear Program)

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

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

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

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

Optimal Setting Values (setpoints)

$$y_{\infty|k}^*, u_s^*(k)$$

(sometimes input deviations are included
in the quadratic objective function)

Steady-State
Prediction Eqn.

State
Feedforward Measurement
Feedback Error

To Dynamic Optimization (Quadratic Program)

Dynamic
Prediction Eqn.