QUIZ

The Linear Quadratic Regulator (LQR)

(a) Is a model free method

(b) Is applicable for continuous state spaces

(c) Can only be used for maintaining an equilibrium

(d) All of the above

Compared to RL algorithms, methods in classical control

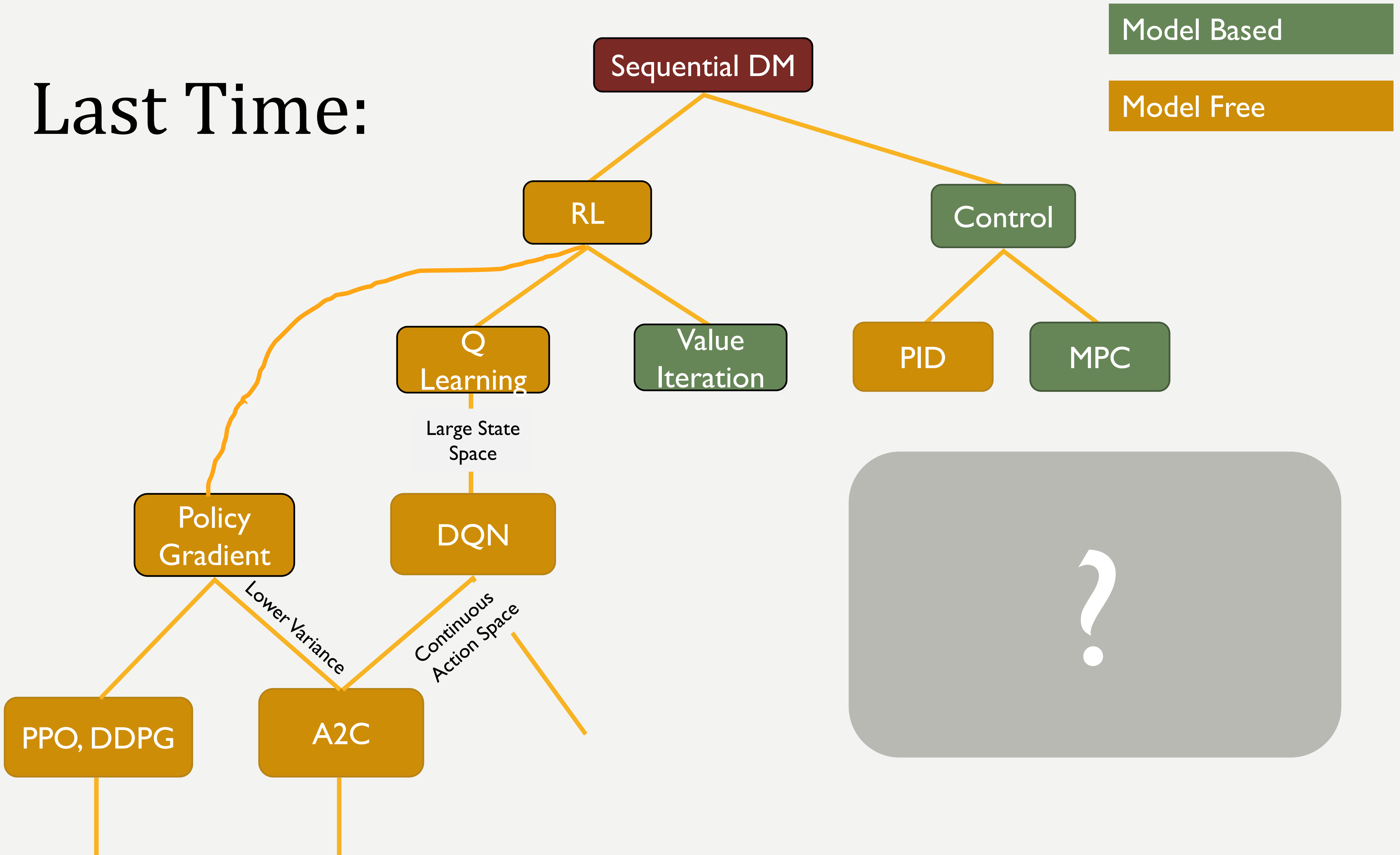
(a) Require more samples to achieve the same performance

(b) Cannot be applied to discrete time systems

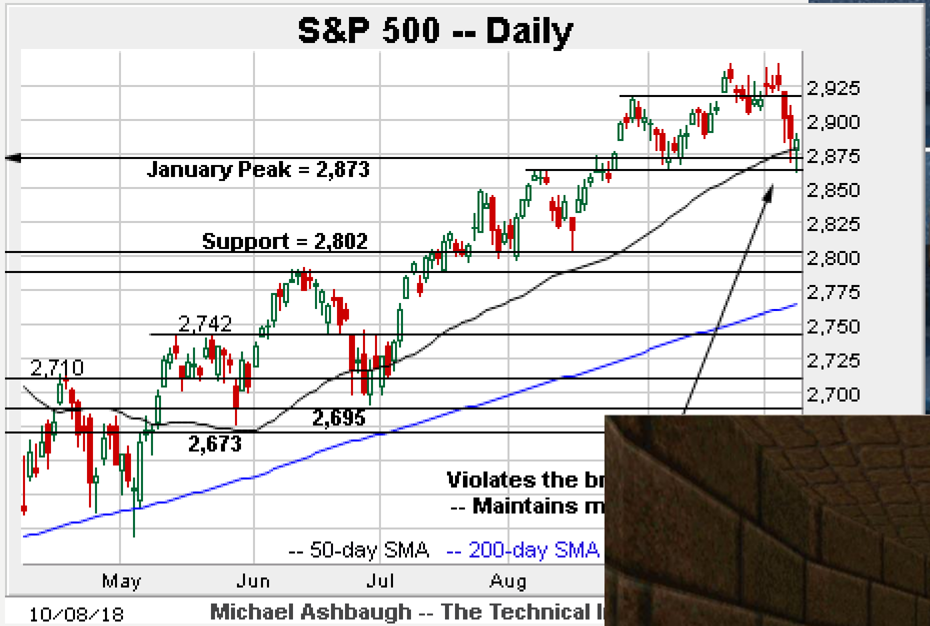
(c) All of the above

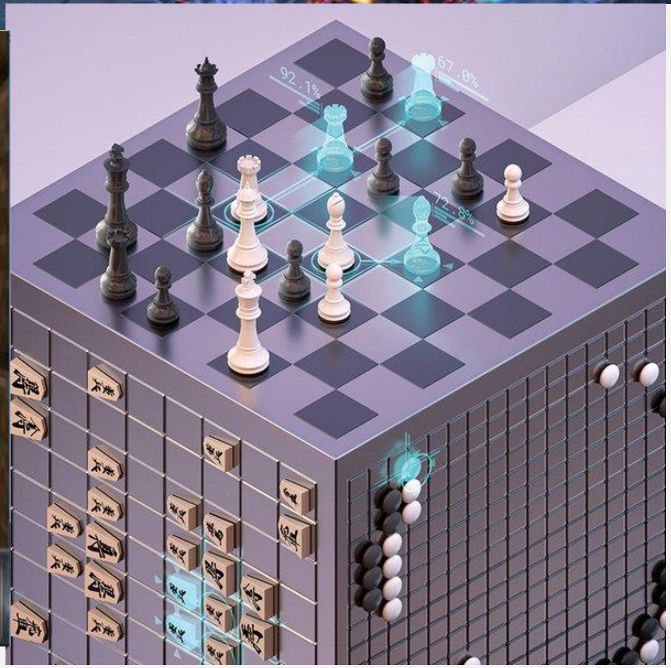
(d) None of the above

Last Time:

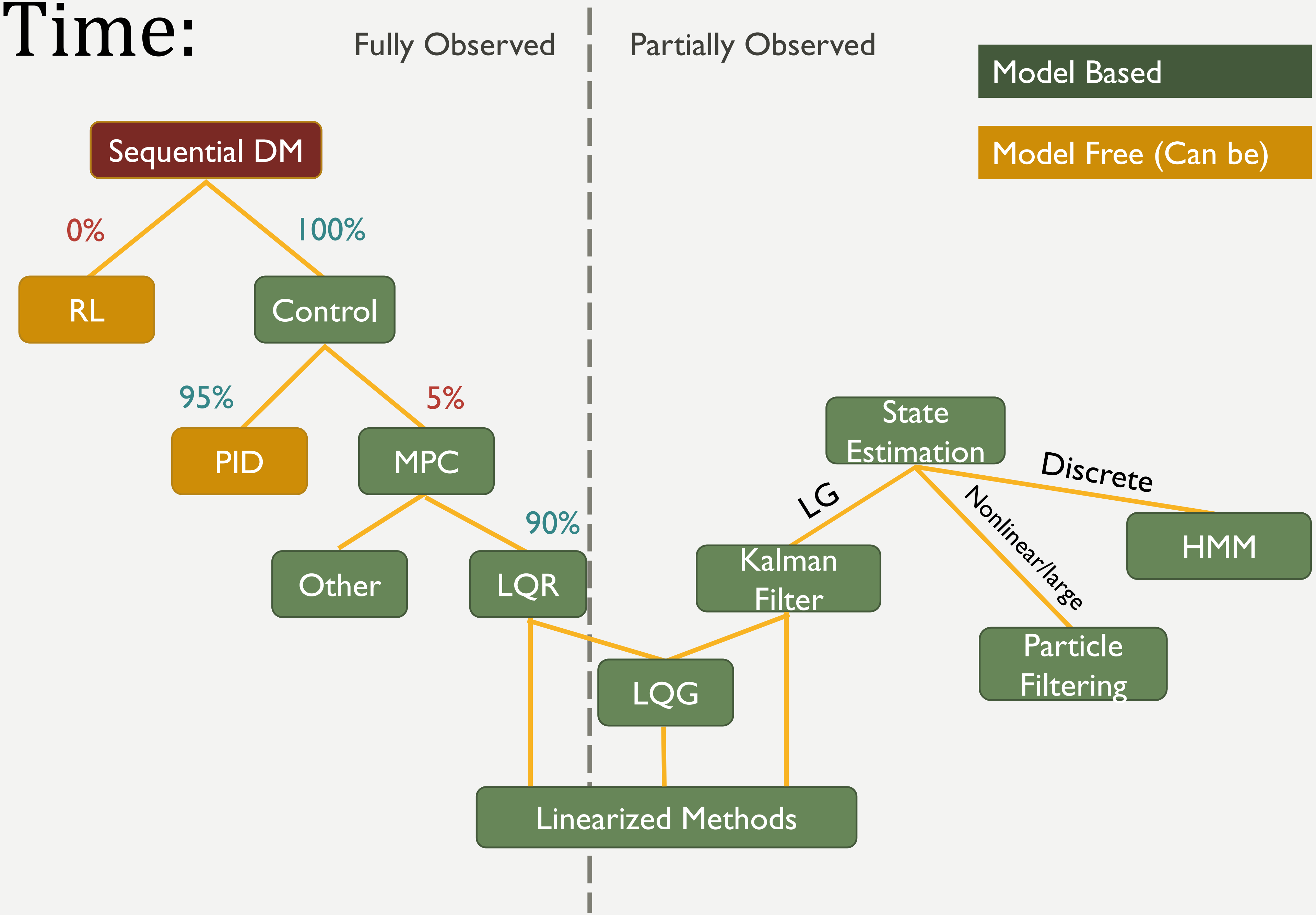

Sequential DM
RL
Control
Q Learning
Value Iteration
PID
MPC
Large State Space
Policy Gradient
DQN
Lower Variance
Continuous Action Space
PPO, DDPG
A2C
?
Model Based
Model Free

APPLICATIONS

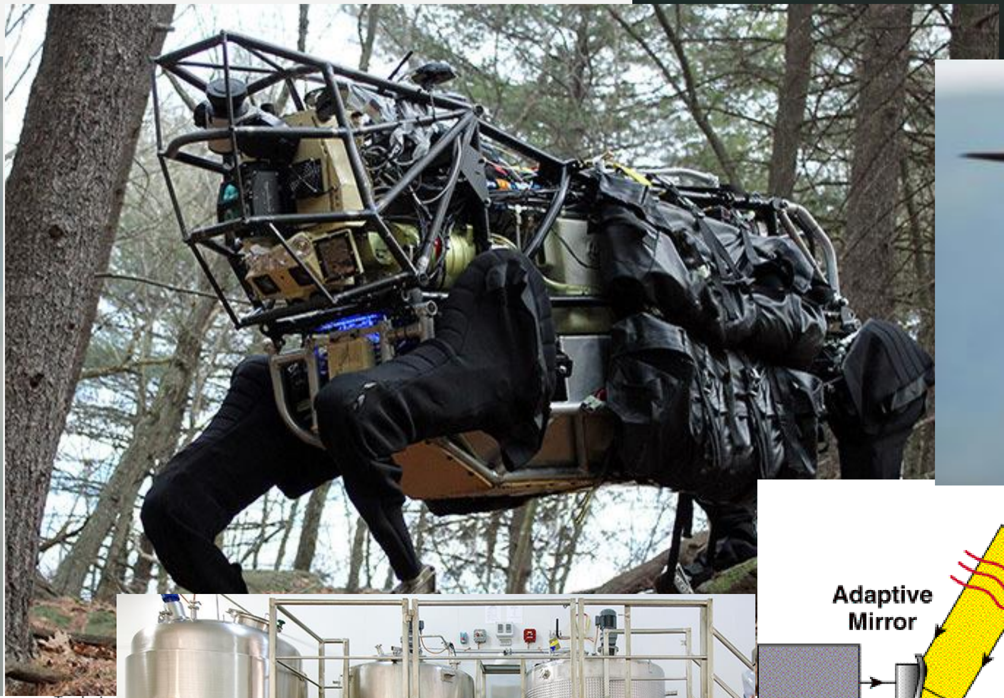


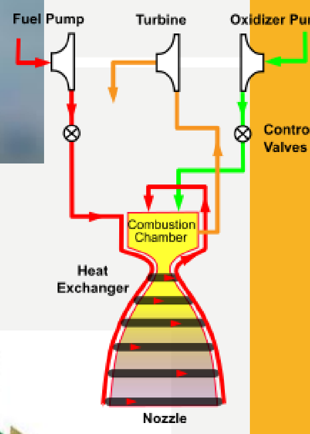


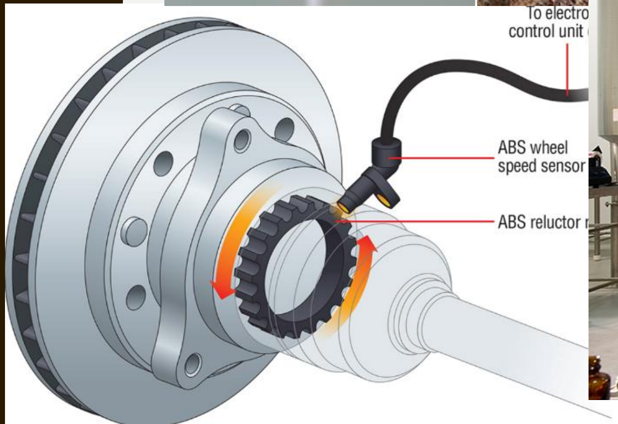
This Time:

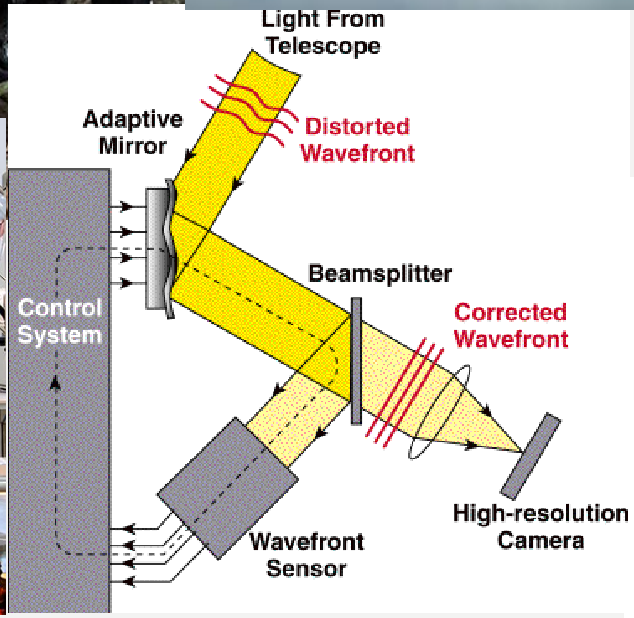

Fully Observed
Partially Observed
Model Based
Model Free (Can be)
Sequential DM
0%
100%
RL
Control
95%
5%
PID
MPC
90%
Other
LQR
State Estimation
LG
Discrete
Nonlinear/large
Kalman Filter
HMM
Particle Filtering
LQG
Linearized Methods

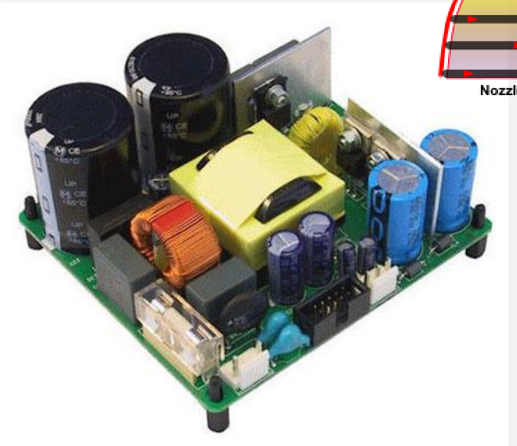
APPLICATIONS



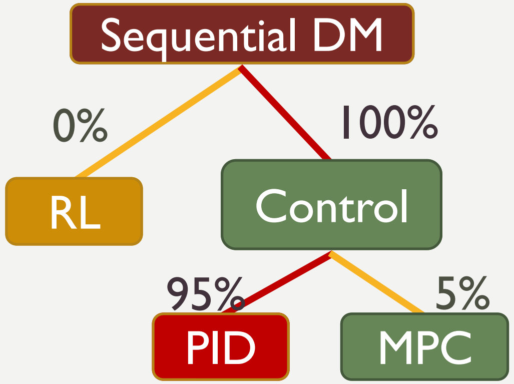






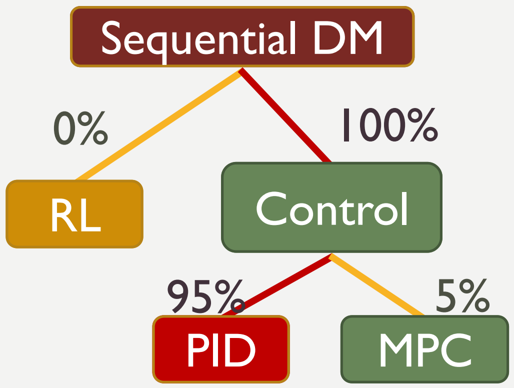


PID CONTROL



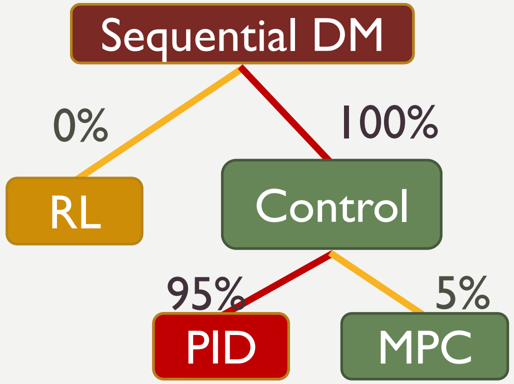
- $e(t) = y(t) - x(t)$
- $u(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \dot{e}(t)$

PID CONTROL



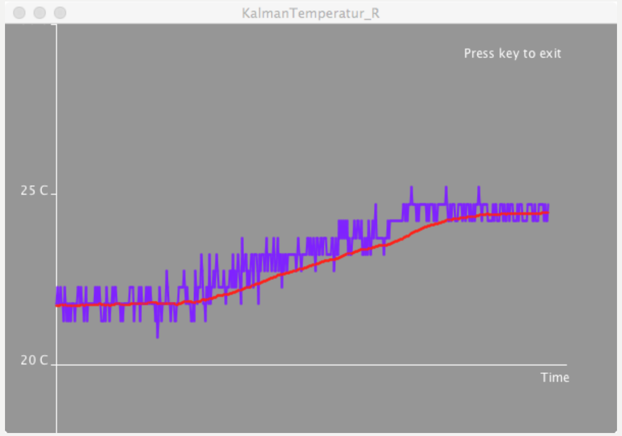
- $e(t) = y(t) - x(t)$
- $u(t) = k_p e(t) + k_i \int_0^t e(t)dt + \cancel{k_d \dot{e}}(t)$

PID CONTROL

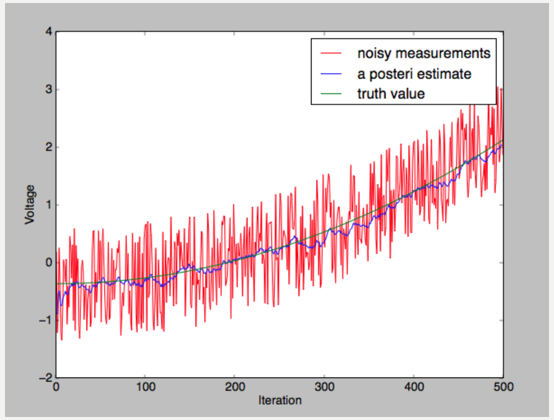


- $e(t) = y(t) - \ x(t)$
- $u(t) = k_p e(t) + k_i \int_0^t e(t)dt + \cancel{k_d \dot{e}(t)}$

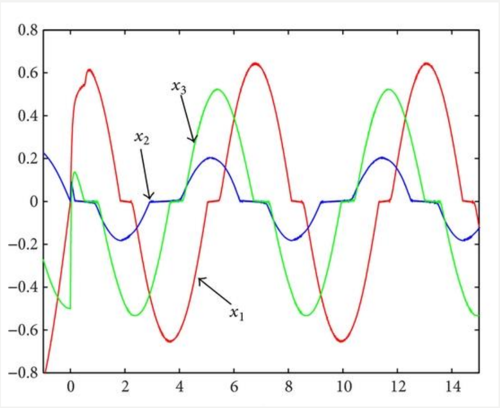
Sensor Quantization



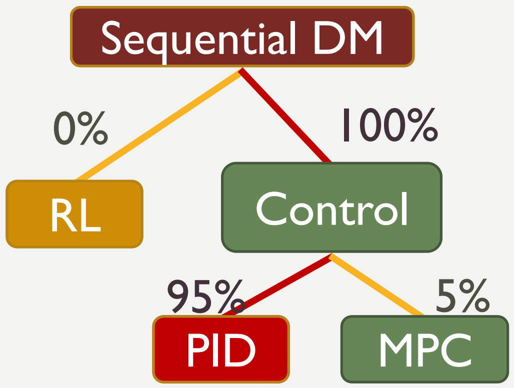
Noisy Measurements



Non differentiable dynamics

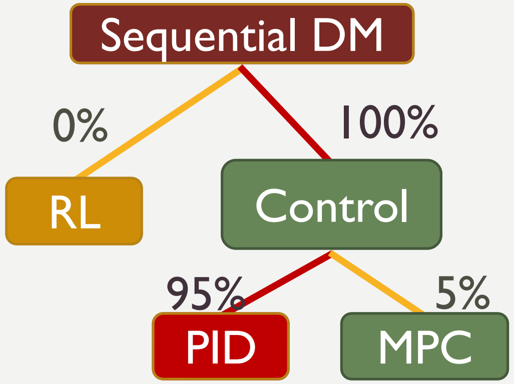


PI CONTROL

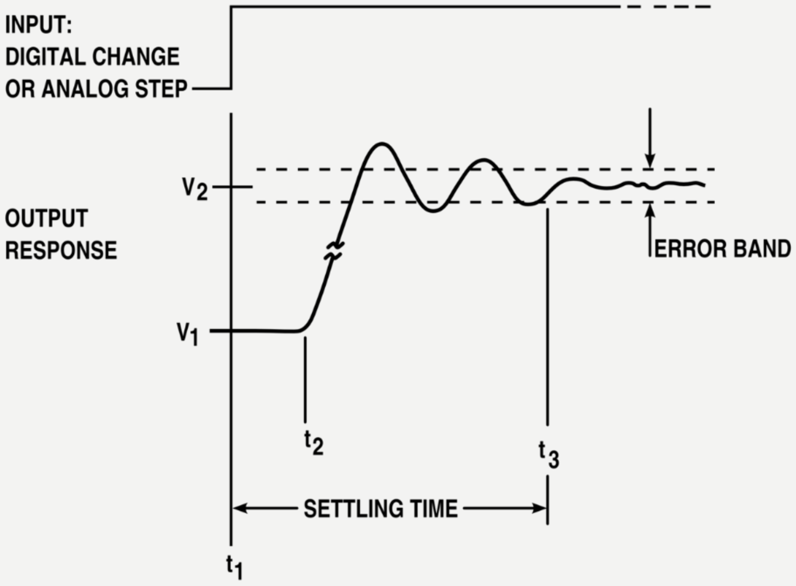


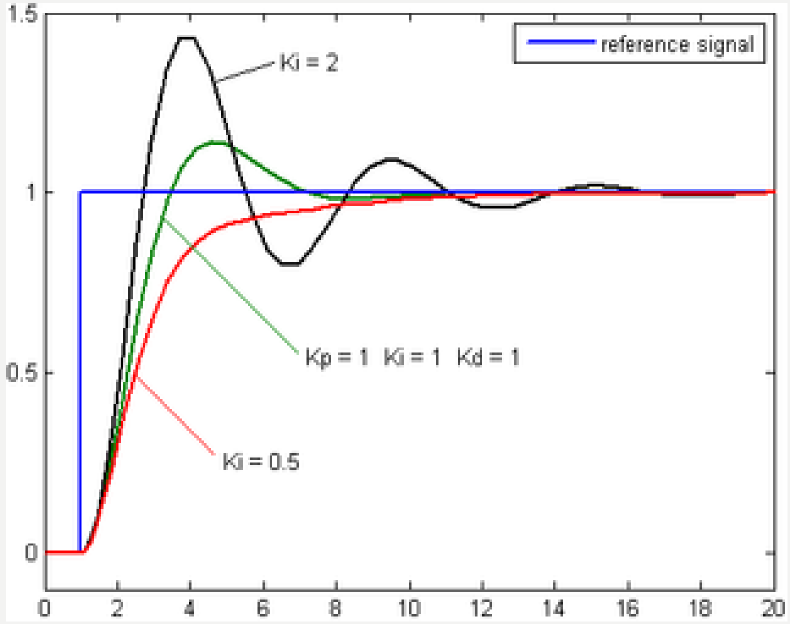
- $e(t) = y(t) - x(t)$
- $u(t) = k_p e(t) + k_i \int_0^t e(t)dt$
- Can actually be implemented in an analog circuit!
- Transfer function $U(s) = k_p + \frac{k_i}{s}$

CHOICE OF PARAMETERS

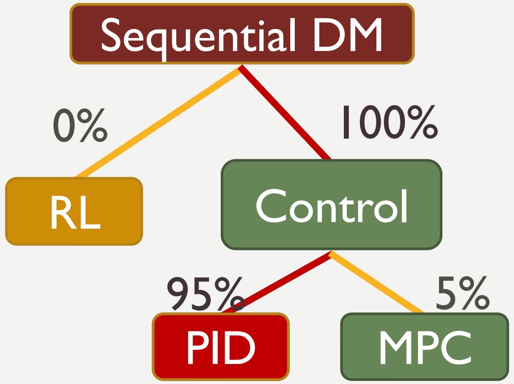


- Most easily analyzed in frequency space (Laplace transform)





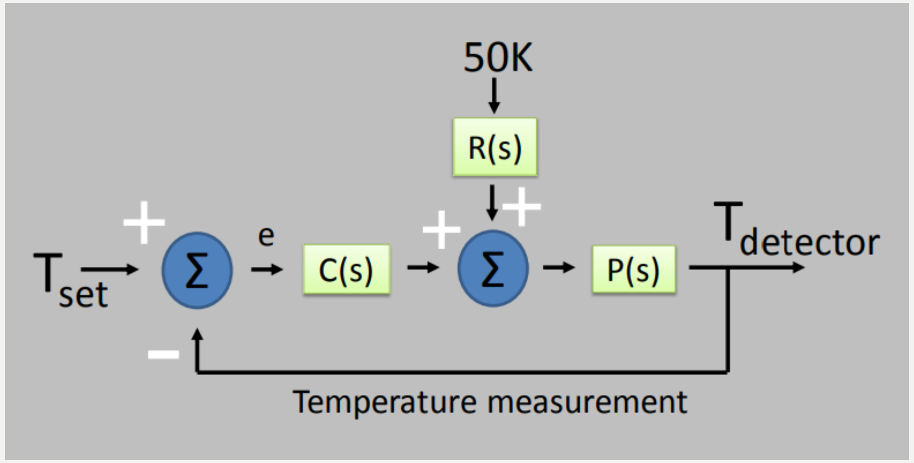
CASE STUDY (TEMPERATURE REGULATION)

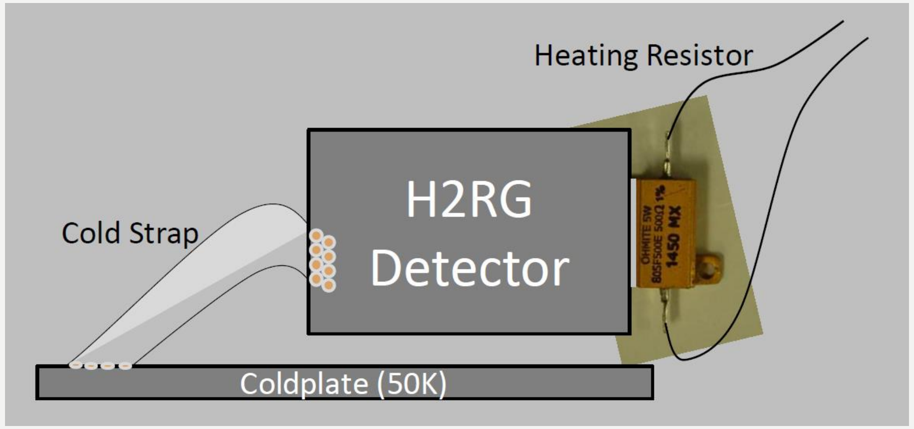


- Model information can be used to tune the coefficients
- Ex: Temperature Control

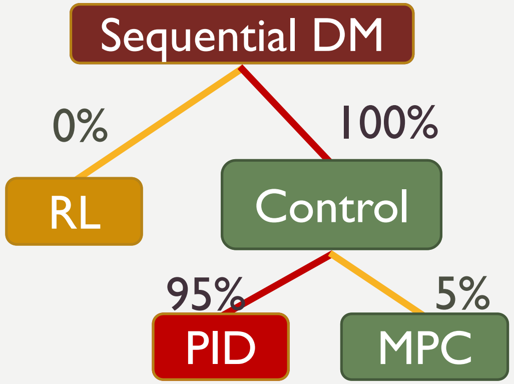
$$\begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix}' = \begin{bmatrix} -\frac{\alpha}{c_h} & \frac{\alpha}{c_h} \\ \frac{\alpha}{c_d} & -\frac{\alpha+\beta}{c_d} \end{bmatrix} \begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix} + \begin{bmatrix} \frac{q_r(t)}{c_h} \\ 50K\frac{\beta}{c_d} \end{bmatrix}$$

$$T_d(s) = \frac{T_{set}(s)C(s)P(s)+(50K)R(s)P(s)}{1+C(s)P(s)}$$

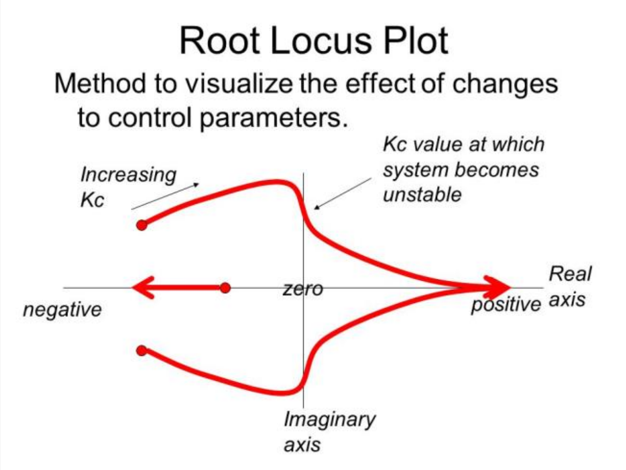


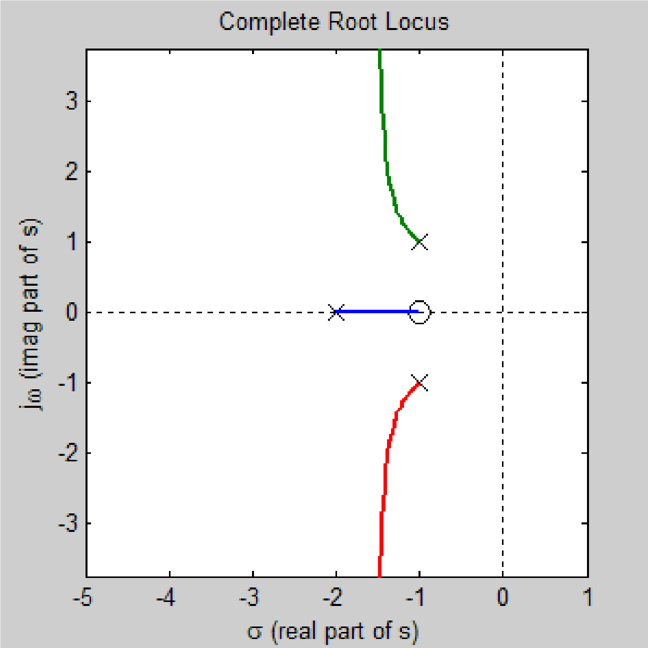


POLE PLACEMENT

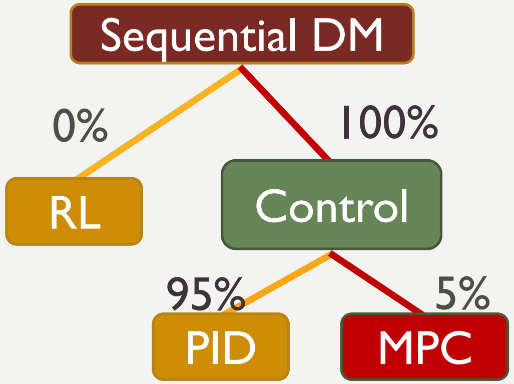


- Want stability of the closed loop system
- Look at poles of the closed loop transfer function





MODEL PREDICTIVE CONTROL



Do:

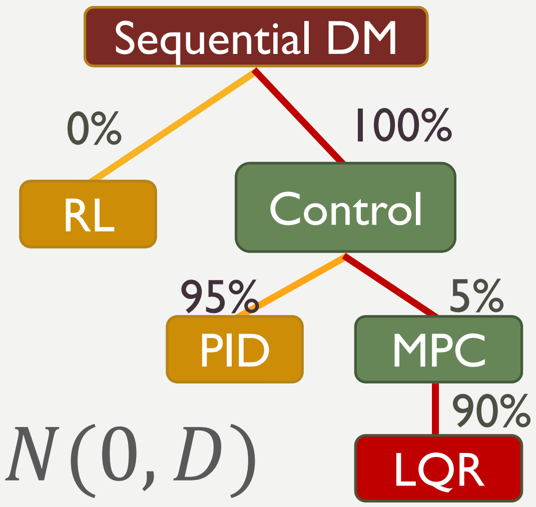
- Update current state estimate
- Rollout Trajectory to Finite Horizon T
- Optimize the cumulative reward with respect to the set of T controls on the trajectory
- Take first action in the sequence
- Repeat

- Can initialize with previous action sequence
- Robust to model misspecification & noise
- Can be too costly for real time [in general]

Algorithm F.1 MPC algorithm

Input: initial system state $\mathbf{x}^0$,
Input: randomly initialized sequence of actions $\{\mathbf{a}^t\}$.
Input: pretrained dynamics model M such $\mathbf{x}^{t_0+1} = M(\mathbf{x}^{t_0}, \mathbf{a}^{t_0})$
Input: Trajectory cost function L such $c = C(\{\mathbf{x}^t\}, \{\mathbf{a}^t\})$
for a number of iterations **do**
  $\mathbf{x}_r^0 = \mathbf{x}^0$
  for t in range(0, horizon) **do**
    $\mathbf{x}_r^{t+1} = M(\mathbf{x}_r^t, \mathbf{a}^t)$
  end for
  Calculate trajectory cost $c = C(\{\mathbf{x}_r^t\}, \{\mathbf{a}^t\})$
  Calculate gradients $\{\mathbf{g}_a^t\} = \frac{\partial c}{\partial \{\mathbf{a}^t\}}$
  Apply gradient based update to $\{\mathbf{a}^t\}$
end for
Output: optimized action sequence $\{\mathbf{a}^t\}$

LQR CONTROL (DISCRETE TIME)

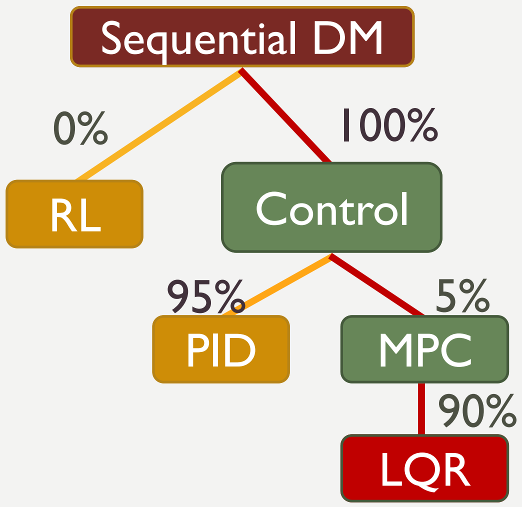


- Dynamics Model: $x_{t+1} = Ax_t + Bu_t + \epsilon, \quad \epsilon \sim N(0, D)$
- Cost (-Reward):

$$\mathrm{J}(\pi, x_0) = E[x_N^T S x_N + \sum_{t=0}^{N-1} x_t^T Q x_t + u_t^T R u_t]$$

- Conveniences:
  - Objective is convex in $\{u_t\}_0^N$ (and quadratic)
  - State is Gaussian
  - Noisy dynamics don't change the objective

LQR CONTROL

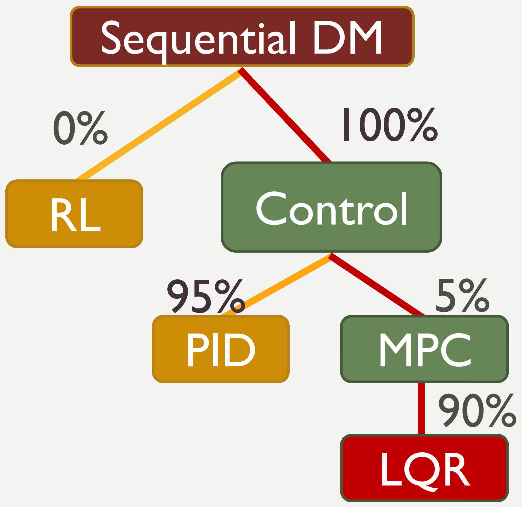


- Consider:

$$z_{t+1} = Az_t + Bu_t, \qquad z_0 = x_0$$

$$x_t = z_t + \sum_0^t A^t \epsilon_t = z_t + \xi_t, \text{ where } \xi_t \sim N\left(0, \left(\sum_0^t A^t\right)^T D \left(\sum_0^t A^t\right)\right)$$

LQR CONTROL



- Consider:

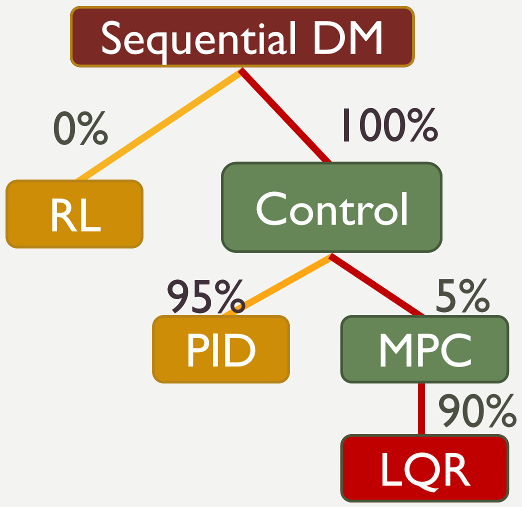
$$z_{t+1} = Az_t + Bu_t, \qquad z_0 = x_0$$

$$x_t = z_t + \sum_0^t A^t \epsilon_t = z_t + \xi_t, \quad \text{where} \quad \xi_t \sim N\left(0, \left(\sum_0^t A^t\right)^T D \left(\sum_0^t A^t\right)\right)$$

$$\text{J}(\pi, x_0) = E[x_N^T S x_N + \sum_{t=0}^{N-1} x_t^T Q x_t + u_t^T R u_t] =$$

$$= z_N^T S z_N + E[\xi_N^T S \xi_N] + \sum_{t=0}^{N-1} (z_t^T Q z_t + u_t^T R u_t + E[\xi_t^T Q \xi_t])$$

LQR CONTROL



- Consider:

$$z_{t+1} = Az_t + Bu_t, \qquad z_0 = x_0$$

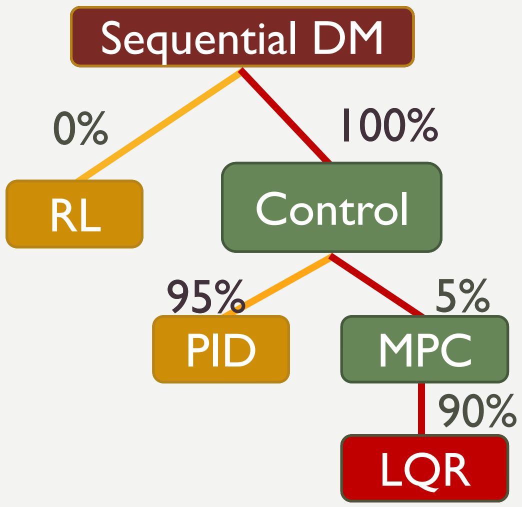
$$x_t = z_t + \sum_0^t A^t \epsilon_t = z_t + \xi_t, \text{ where } \xi_t \sim N\left(0, \left(\sum_0^t A^t\right)^T D \left(\sum_0^t A^t\right)\right)$$

$$\mathrm{J}(\pi, x_0) = E\left[x_N^T S x_N + \sum_{t=0}^{N-1} x_t^T Q x_t + u_t^T R u_t\right] =$$

$$= z_N^T S z_N + E[\xi_N^T S \xi_N] + \sum_{t=0}^{N-1} \left(z_t^T Q z_t + u_t^T R u_t + E[\xi_t^T Q \xi_t]\right)$$

$$J(\pi) = z_N^T S z_N + \sum_{t=0}^{N-1} \left(z_t^T Q z_t + u_t^T R u_t\right)$$

LQR CONTROL



Use Lagrange multipliers:

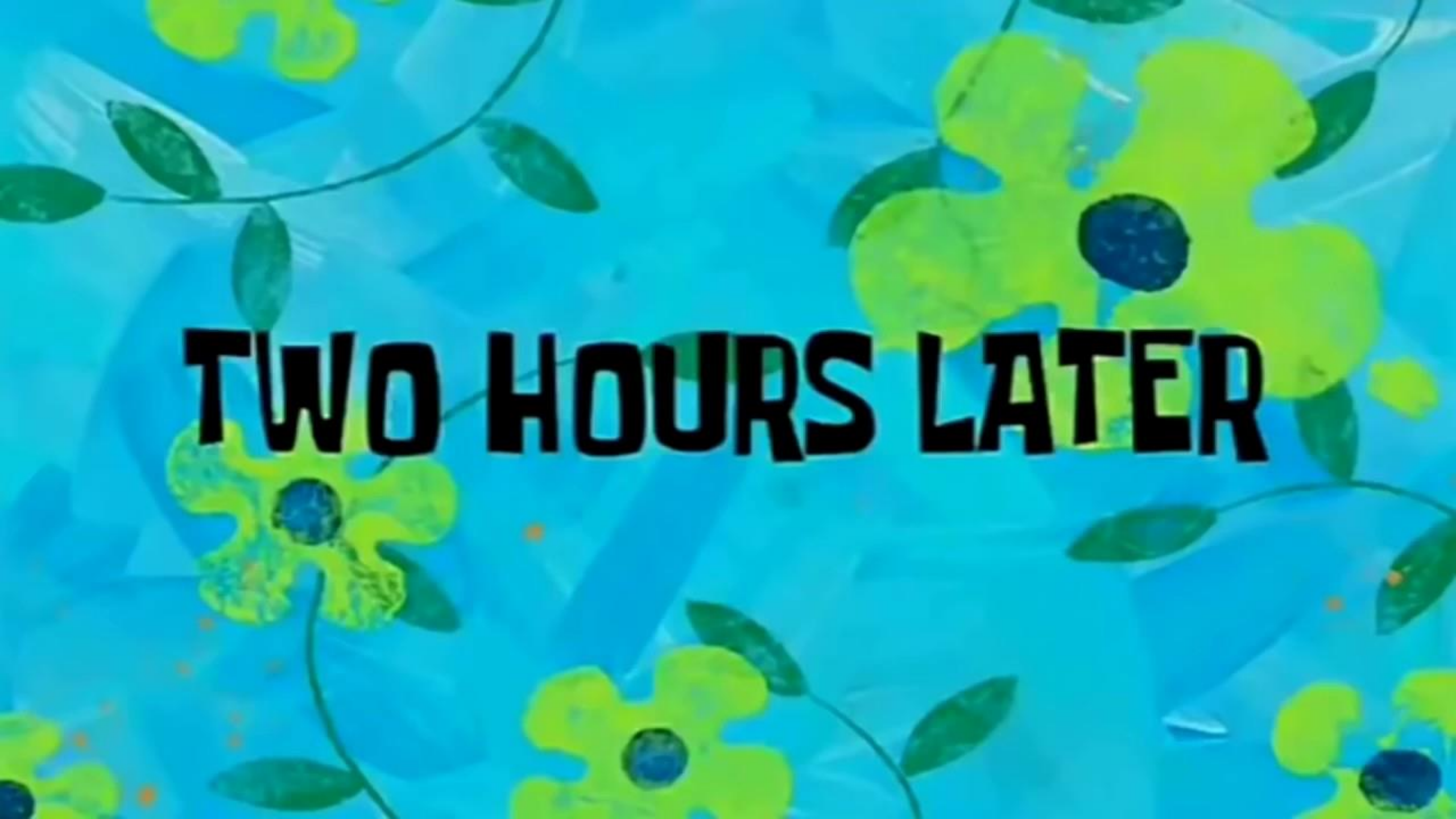
$$L(x,\lambda,u) = x_N^T S x_N + \sum_{t=0}^{N-1}(x_t^T Q x_t + u_t^T R u_t - \lambda_t^T (x_{t+1} - A x_t - B u_t))$$

$$\nabla_{x_N} L = S x_N - \lambda_N = 0$$

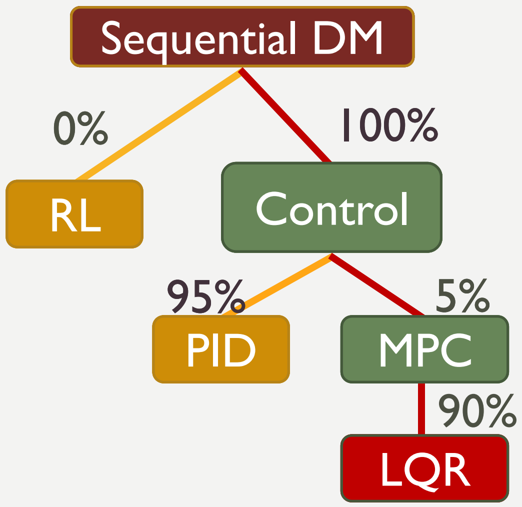
$$\nabla_{x_t} L = Q x_t - \lambda_t + A^T \lambda_{t+1} = 0$$

$$\nabla_{u_t} L = R u_t + B^T \lambda_{t+1} = 0$$

$$\nabla_{\lambda_t} L = A x_t + B x_t - x_{t+1} = 0$$


TWO HOURS LATER

LQR SOLUTIONS



$$u_t = -K_t x_t$$

where:

$$K_t = (R + B^T M_{t+1} B)^{-1} B^T M_{t+1} A$$

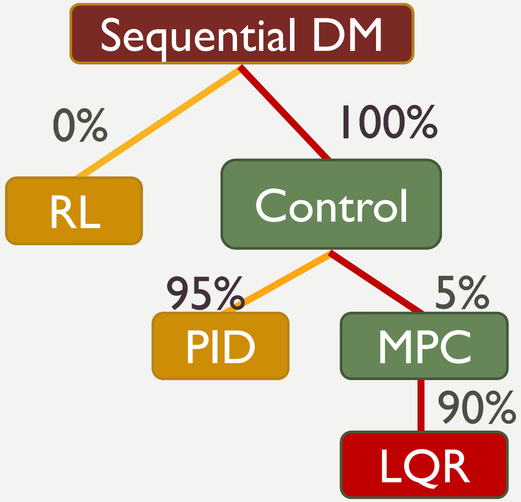
$$M_t = Q + A^T M_{t+1} A \ - (A^T M_{t+1} B)(R + B^T M_{t+1} B)^{-1}(B^T M_{t+1} A)$$

Backwards recursion starting with $M_N = S$

Infinite Horizon Case: set $M_t = M$

M = scipy.linalg.solve_discrete_are(A,B,Q,R)

LQR (CONTINUOUS TIME)



- Dynamics Model: $\dot{x}(t) = Ax(t) + Bu(t)$
- Cost (-Reward):

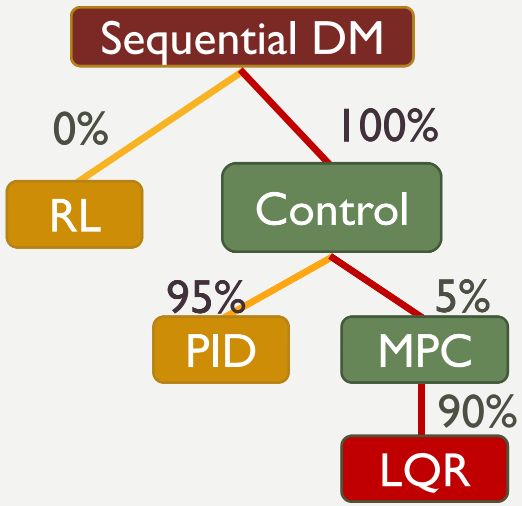
$$\mathrm{J}(u, x_0) = \int_0^{\infty} \left(x(t)^T Q x(t) + u(t)^T R u(t)\right) dt$$

- Solution

$$u(t) = -Kx(t)$$

$$K = R^{-1} B^T M, \qquad CARE(A, B, Q, R, M) = 0$$

COURSE MODEL ID

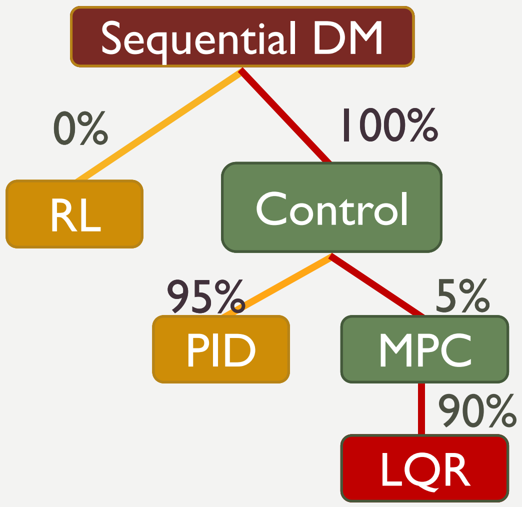


- What if we don't know A and B?

- Ex: Temp Controller

$$\begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix}' = \begin{bmatrix} -\dfrac{\alpha}{c_h} & \dfrac{\alpha}{c_h} \\ \dfrac{\alpha}{c_d} & -\dfrac{\alpha+\beta}{c_d} \end{bmatrix} \begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix} + \begin{bmatrix} \dfrac{q_r(t)}{c_h} \\ 50K\dfrac{\beta}{c_d} \end{bmatrix}$$

COURSE MODEL ID



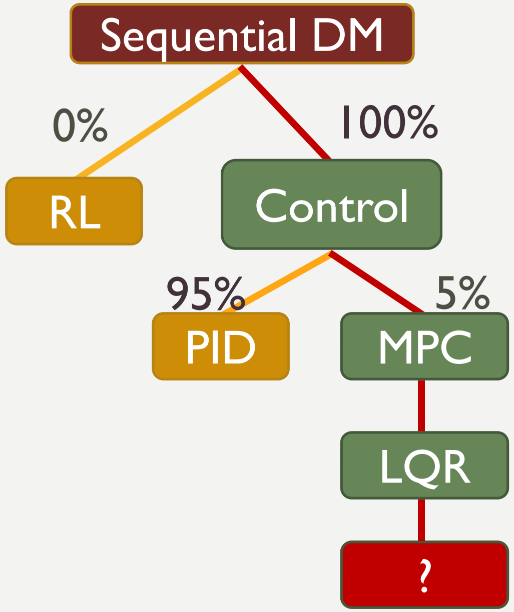
- What if we don't know A and B?
- Ex: Temp Controller

$$\begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix}' = \begin{bmatrix} -\frac{\alpha}{c_h} & \frac{\alpha}{c_h} \\ \frac{\alpha}{c_d} & -\frac{\alpha+\beta}{c_d} \end{bmatrix} \begin{bmatrix} T_h(t) \\ T_d(t) \end{bmatrix} + \begin{bmatrix} \frac{q_r(t)}{c_h} \\ 50K\frac{\beta}{c_d} \end{bmatrix}$$

- Fit the model

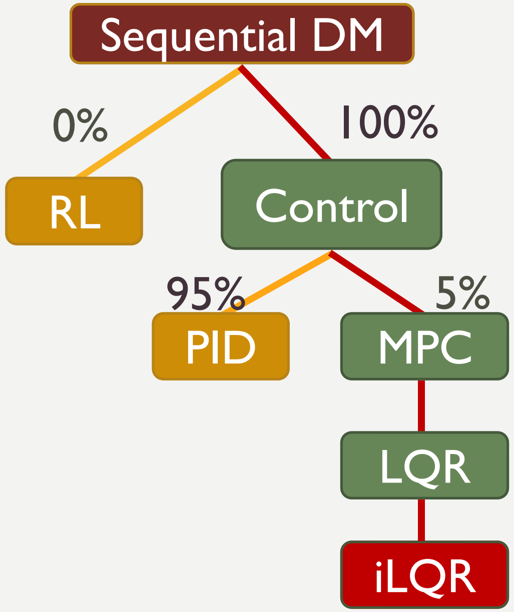
$$\min_{A,B} \sum_t ||x_{t+1} - Ax_t - Bu_t||^2$$

NONLINEAR DYNAMICS



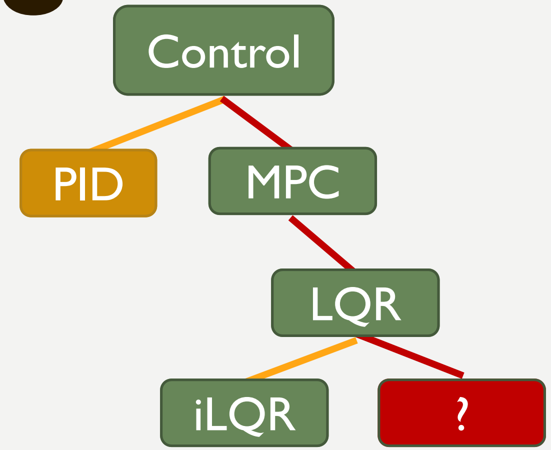
- What to do with nonlinear dynamics?
- $\dot{x}(t) = f(x, u)$
- Ex: (single pendulum) $\begin{bmatrix} \dot{\omega} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{g}{L}\sin(\theta) \\ \omega \end{bmatrix} + u \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

NONLINEAR DYNAMICS



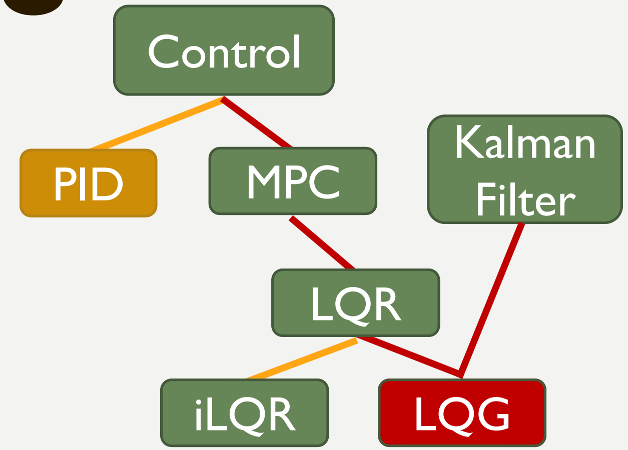
- What to do with nonlinear dynamics?
- $\dot{x}(t) = f(x, u)$ or $x_{t+1} = f(x_t, u_t)$
- Ex: (single pendulum) $\begin{bmatrix} \dot{\omega} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{g}{L}\sin(\theta) \\ \omega \end{bmatrix} + u \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
- Plan using local linear approximation of dynamics (iLQR) around a guessed trajectory $\tau = \{\tilde{x}_t, \tilde{u}_t\}$
- $x_{t+1} \approx f(\tilde{x}_t, \tilde{u}_t) + D_x f(\tilde{x}_t, \tilde{u}_t)(x_t - \tilde{x}_t) + D_u f(\tilde{x}_t, \tilde{u}_t)(u_t - \tilde{u}_t)$
- $u_t - \tilde{u}_t = K_t(x_t - \tilde{x}_t)$ and iterate

INCOMPLETE MEASUREMENTS



- What if state is not fully observed?
- $y_t = Mx_t + w, \qquad w \sim N(0, C)$
- Ex: Missile Guidance
- $\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}_{t+1} = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & -\gamma & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}_t + (u_t - g) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + N(0, D)$
- $y_t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}_t + N\left(0, \begin{bmatrix} 1m^2 & 0 \\ 0 & \left(\frac{0.01m}{s^2}\right)^2 \end{bmatrix}\right)$

INCOMPLETE MEASUREMENTS

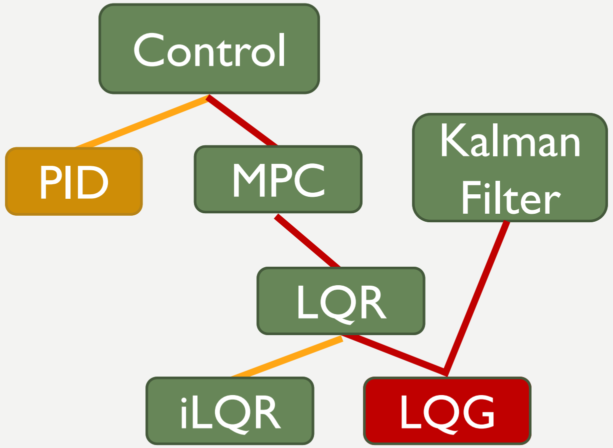


- What if state is not fully observed?
- $y_t = Mx_t + w, \qquad w \sim N(0, C)$
- Linear Measurements + Gaussian Noise

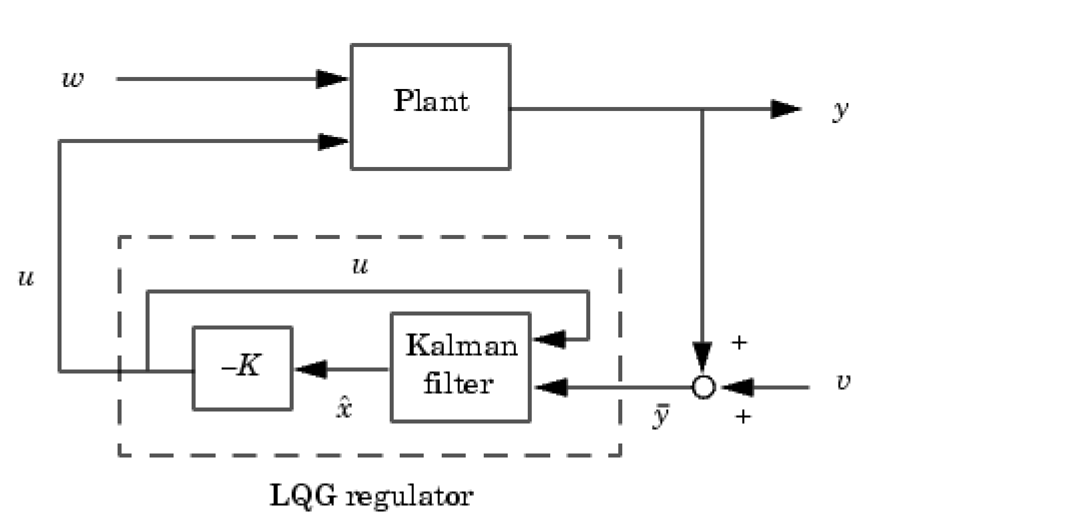
  = Kalman Filter

- Posterior over x_t is also Gaussian, we just need to update the mean μ_t and covariance Σ_t as new measurements come in.

LQG CONTROL

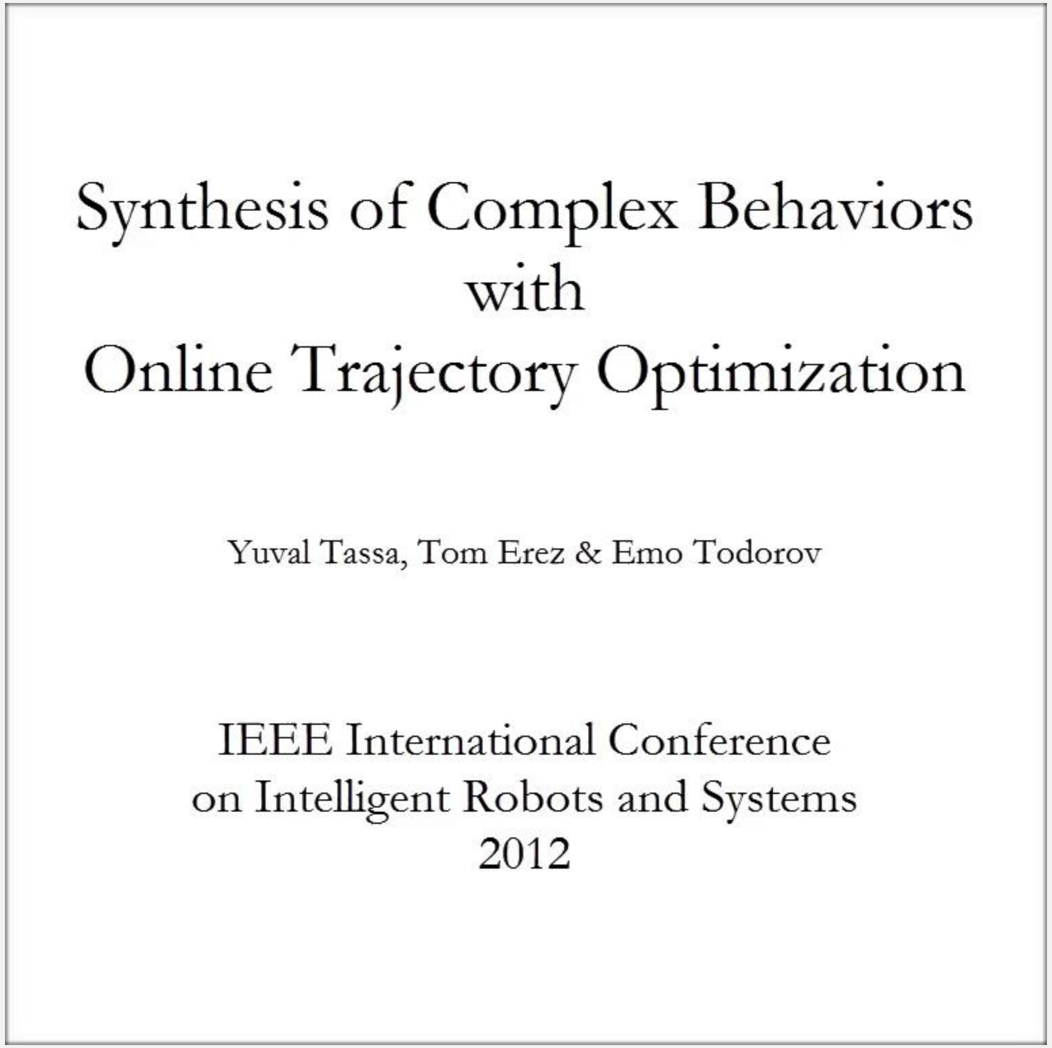


- $y_t = Mx_t + w, \qquad w \sim N(0, C)$
- Separation Principle
  - Optimal state estimation & control can be decoupled
- $u_t = -K\hat{x}_t$
- $\hat{x}_t = \mu_t$



https://youtu.be/8jk17dXETHs

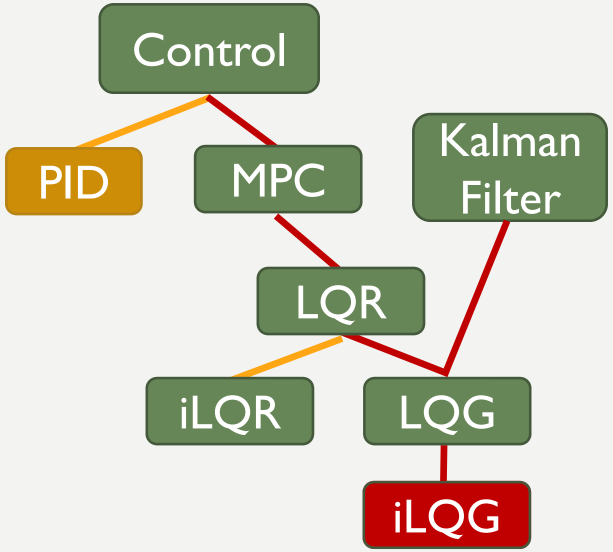
MUJOCO PAPER



Synthesis of Complex Behaviors
with
Online Trajectory Optimization

Yuval Tassa, Tom Erez & Emo Todorov

IEEE International Conference
on Intelligent Robots and Systems
2012



Can easily modify objective,
Unlike most RL algorithms

Many physical systems can be written in the form

$$\dot{p} = F(q, u)$$
$$\dot{q} = M(q)^{-1} p$$

CONCLUSIONS

- Control theory has many practical applications
- More hands on approach than RL, where models or coarse models are known, allows for more robust and sample efficient algorithms
- Linear Quadratic Regulator is very powerful and extensible special case of MPC.
- Many systems and sensors are designed with linearity and Gaussianity in mind.