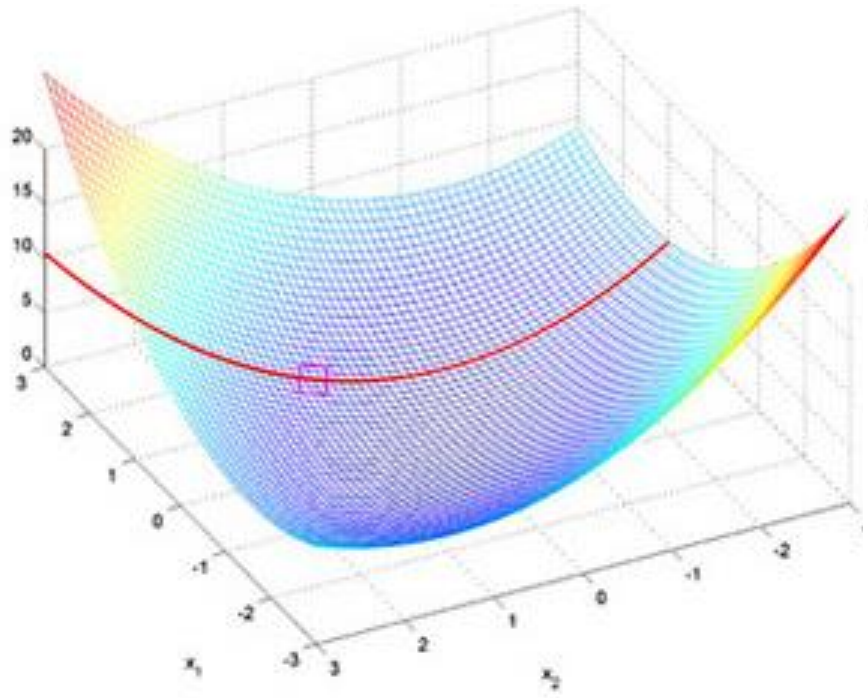
MEGR 7090/8090: Advanced Optimal Control

UNC CHARLOTTE



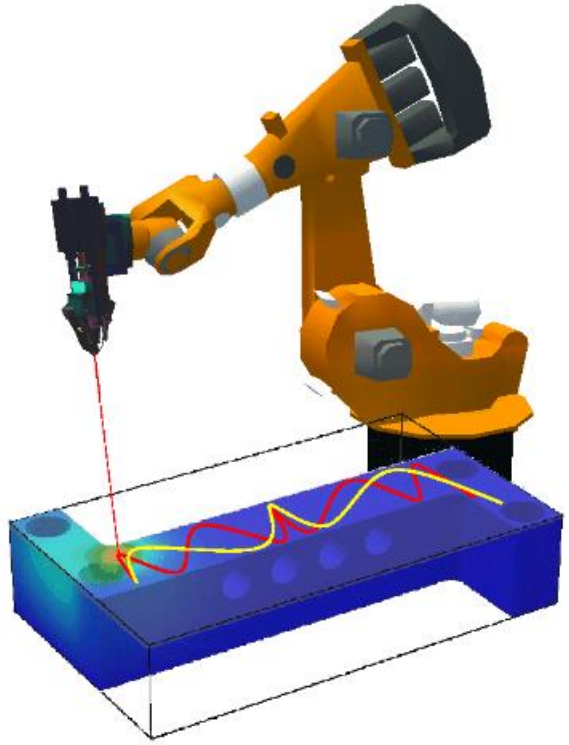
$$V_n(\mathbf{x}_n) = \min_{\{\mathbf{u}_n, \mathbf{u}_{n+1}, \cdots, \mathbf{u}_{N-1}\}} \left[\frac{1}{2} \sum_{k=n}^{N-1} \left(\mathbf{x}_k^T \mathbf{Q}_k \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k \right) + \frac{1}{2} \mathbf{x}_N^T \mathbf{Q}_N \mathbf{x}_N \right]$$

$$V_n(\mathbf{x}_n) = \min_{\{\mathbf{u}_n, \mathbf{u}_{n+1}, \cdots, \mathbf{u}_{N-1}\}} \left[\frac{1}{2} \sum_{k=n}^{N-1} \left(\mathbf{x}_k^T \mathbf{Q}_k \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k \right) + \frac{1}{2} \mathbf{x}_N^T \mathbf{Q}_N \mathbf{x}_N \right]$$

$$= \min_{\mathbf{u}_n} \left[\frac{1}{2} \left(\mathbf{x}_n^T \mathbf{Q}_n \mathbf{x}_n + \mathbf{u}_n^T \mathbf{R} \mathbf{u}_n \right) + \underbrace{\min_{\{\mathbf{u}_{n+1}, \cdots, \mathbf{u}_{N-1}\}} \left[\frac{1}{2} \sum_{k=n+1}^{N-1} \left(\mathbf{x}_k^T \mathbf{Q}_k \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k \right) + \frac{1}{2} \mathbf{x}_N^T \mathbf{Q}_N \mathbf{x}_N \right]}_{V_{n+1}(\mathbf{x}_{n+1})} \right]$$

$$= \min_{\mathbf{u}_n} \left[\frac{1}{2} \left(\mathbf{x}_n^T \mathbf{Q}_n \mathbf{x}_n + \mathbf{u}_n^T \mathbf{R} \mathbf{u}_n \right) + V_{n+1}(\mathbf{x}_{n+1}) \right]$$

$$V_n(\mathbf{x}_n) = \min_{\mathbf{u}_n} \left[\frac{1}{2} \left(\mathbf{x}_n^T \mathbf{Q}_n \mathbf{x}_n + \mathbf{u}_n^T \mathbf{R} \mathbf{u}_n \right) + V_{n+1}(\mathbf{x}_{n+1}) \right]$$



Lecture 3

August 29, 2017

Review of Basics from Lecture 2

UNC CHARLOTTE

Office hour: Tuesdays, 1-2pm

Recitation section (online): Sunday, ~~7-9~~ 6-8 pm

Doodle poll thereafter.

---

3 elements in optimal control problems:

1) Objective function: $J(u(t); \underline{x}(0))$
   - J: (cost)
   - $u(t)$: control trajectory (decision var.)
   - $\underline{x}(0)$: initial state

2) Dynamic model: $\dot{\underline{x}} = f(\underline{x}, u)$

3) Constraints.

Topics for today:

1) Deriving continuous-time state space representations
2) Sampling of signals
3) Discrete-time approximations.

Continuous-Time Optimal Control Framework - Reminder

UNC CHARLOTTE

Whether the control trajectory is optimized offline or online, every continuous-time optimal control problem will involve the following general framework:

$$u^*(t) = \arg\min_{u(t)} J(u(t); \mathbf{x}(0)) \quad \text{where} \quad J\big(u(t); \mathbf{x}(0)\big) = \int_0^{t_f} g(\mathbf{x}(t), u(t))dt + h(\mathbf{x}(t_f))$$

Total cost

Instantaneous cost

Terminal cost

Subject to:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$$
$$\mathbf{u}(t) \in U, \forall t: 0 \le t \le t_f$$
$$\mathbf{x}(t) \in X, \forall t: 0 \le t \le t_f$$
$$\mathbf{x}(t_f) \in X_f$$

Key question: What is the *dimension* of the trajectory ($\mathbf{u}$) to be optimized?

Hint: You need to optimize $u(0), u(0.1t_f), u(0.9t_f), u(0.99t_f)$, … (i.e., you need to optimize u at every time instant between $t = 0$ and $t = t_f$.

Optimal Control – General Discrete-Time Framework

Whether the control trajectory is optimized offline or online, every discrete-time optimal control problem will involve the following general framework:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} J(\mathbf{u}; \mathbf{x}(0)) \quad \text{where} \quad J(\mathbf{u}; \mathbf{x}(0)) = \sum_{i=0}^{N-1} g(\mathbf{x}(i), \mathbf{u}(i)) + h(\mathbf{x}(N))$$

Total cost

Stage cost

Terminal cost

Subject to:

$$\mathbf{x}(i+1) = f(\mathbf{x}(i), \mathbf{u}(i))$$
$$\mathbf{u}(i) \in U, i = 0 \dots N-1$$
$$\mathbf{x}(i) \in X, i = 0 \dots N-1$$
$$\mathbf{x}(N) \in X_f$$

Note: $\mathbf{u} = [\mathbf{u}(0) \quad \dots \quad \mathbf{u}(N-1)]^T$ …Now, what is the dimension of $\mathbf{u}$?

Key benefit of discrete-time representations for optimal control: The optimization problem is reduced to a finite-dimensional optimization!

Going from Continuous to Discrete Time

UNC CHARLOTTE

- Real physical systems evolve in continuous time (we live in continuous time)

- We need to ***approximate*** real systems in discrete time in order to apply discrete time optimization tools…making this approximation requires knowledge of:
  - How we represent the system in continuous time in the first place
  - Sampled signals and systems

Continuous-Time State Space Representations

UNC CHARLOTTE

Reminder (I hope): A state space representation is a set of n first-order differential equations that describe an n^{th} order system

General form: $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), u(t))$

$\mathbf{x}(t) = [x_1(t) \quad \dots \quad x_n(t)]^T, \mathbf{u}(t) = [u_1(t) \quad \dots \quad u_p(t)]^T$

Matrix form for linear systems:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t)$$

$\mathbf{x}(t) = [x_1(t) \quad \dots \quad x_n(t)]^T, \mathbf{u}(t) = [u_1(t) \quad \dots \quad u_p(t)]^T$

Review question: What is the size of the matrices A and B?

Continuous-Time State Space Representations

UNC CHARLOTTE

Key characteristic of a state ($\mathbf{x}$):

Given $\underline{x}(t_0)$, $u(t)$ for $t \geq t_0$, you can predict evolution of $\underline{x}(t)$, $t \geq t_0$

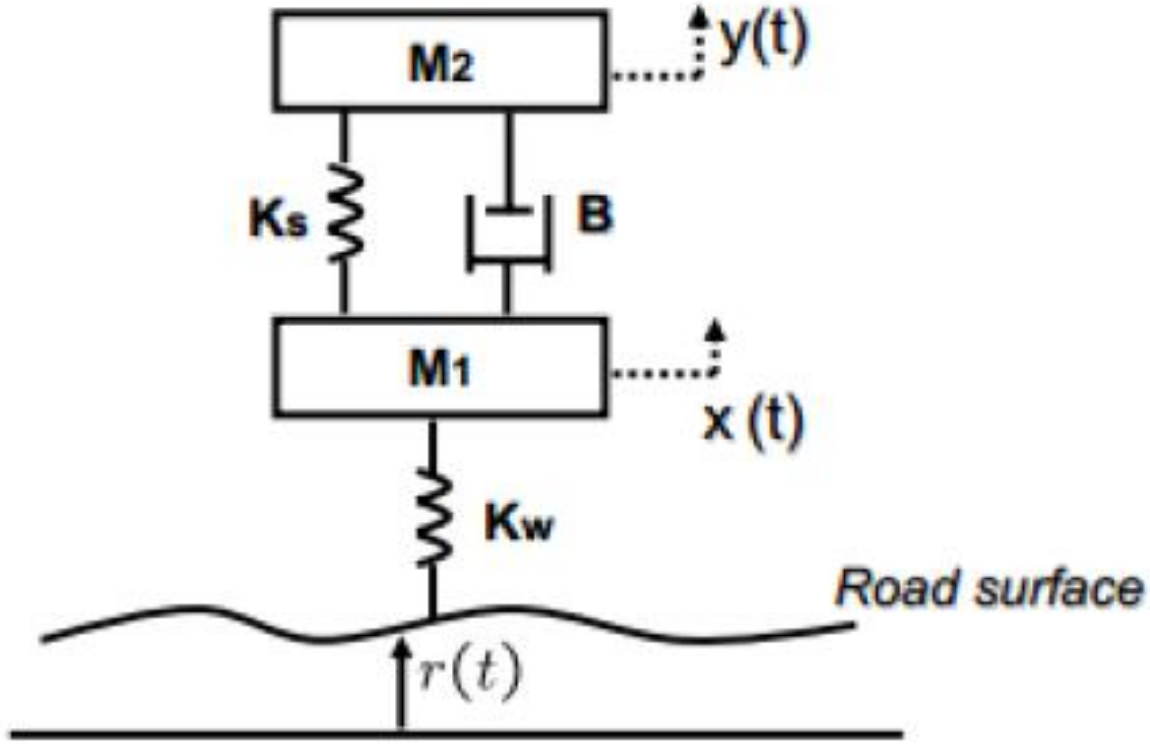
$A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times 1}$

Deriving a Continuous Time State Space Representation - Example

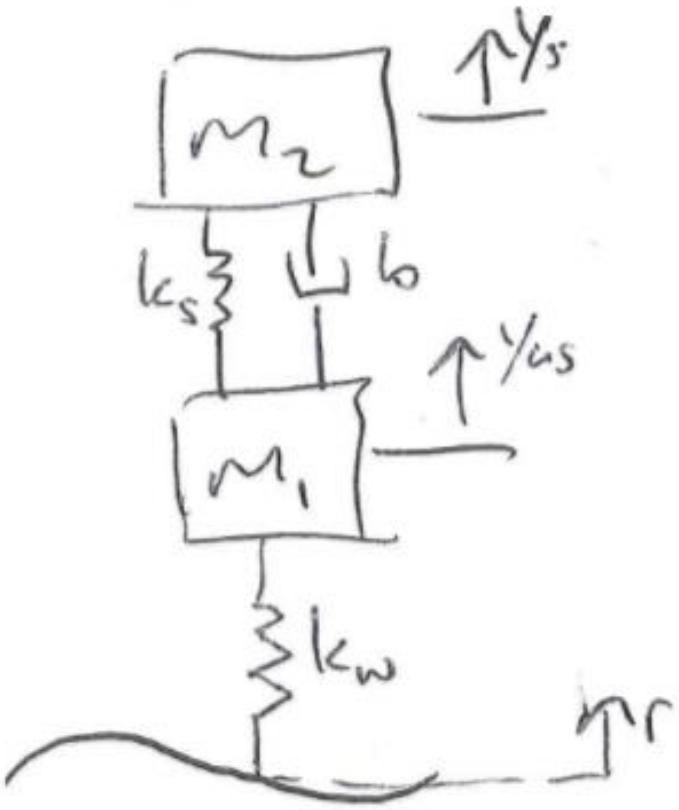
UNC CHARLOTTE

Class exercise: For the quarter car suspension model below, derive a state space representation of the form:

$$\dot{\mathbf{x}}(t) = f\big(\mathbf{x}(t), r(t)\big) = A\mathbf{x}(t) + Br(t)$$



Deriving a Continuous Time State Space Representation - Example



$$m_2 \ddot{y}_s = k_s(y_{us} - y_s) + b(\dot{y}_{us} - \dot{y}_s)$$

$$m_1 \ddot{y}_{us} = k_s(y_s - y_{us}) + b(\dot{y}_s - \dot{y}_{us}) + k_w(r - y_{us})$$

$$x_1 \triangleq y_s \qquad x_2 \triangleq \dot{y}_s$$

$$x_3 \triangleq y_{us} \qquad x_4 \triangleq \dot{y}_{us}$$

Deriving a Continuous Time State Space Representation - Example

UNC CHARLOTTE

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{1}{m_2}\left[k_s(y_{us} - y_s) + b(\dot{y}_{us} - \dot{y}_s)\right]$$

$$= \frac{1}{m_2}\left[k_s(x_3 - x_1) + b(x_4 - x_2)\right]$$

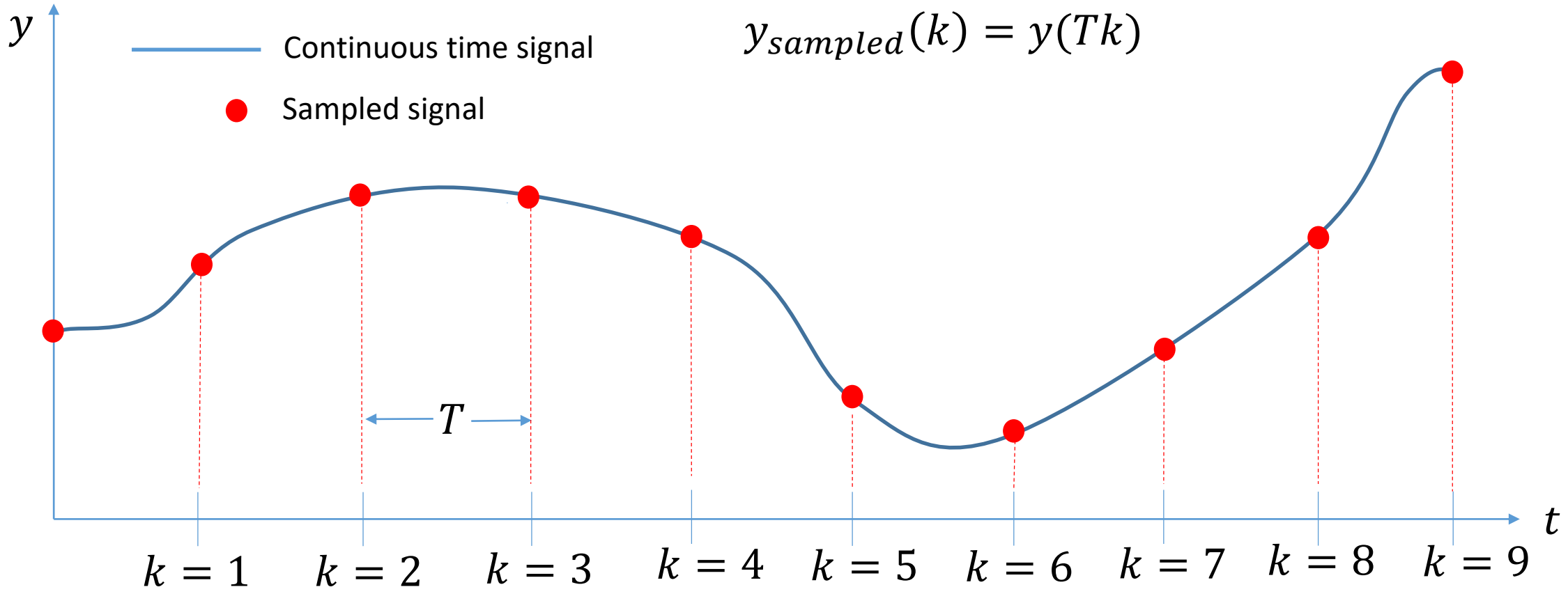
$$\dot{x}_3 = x_4$$

$$\dot{x}_4 = \frac{1}{m_1}\left[k_s(x_1 - x_3) + b(x_2 - x_4) + k_w(r - x_3)\right]$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_s}{m_2} & -\frac{b}{m_2} & \frac{k_s}{m_2} & \frac{b}{m_2} \\ 0 & 0 & 0 & 1 \\ \frac{k_s}{m_1} & \frac{b}{m_1} & -\frac{(k_w+k_s)}{m_1} & -\frac{b}{m_1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{k_w}{m_1} \end{bmatrix} r$$

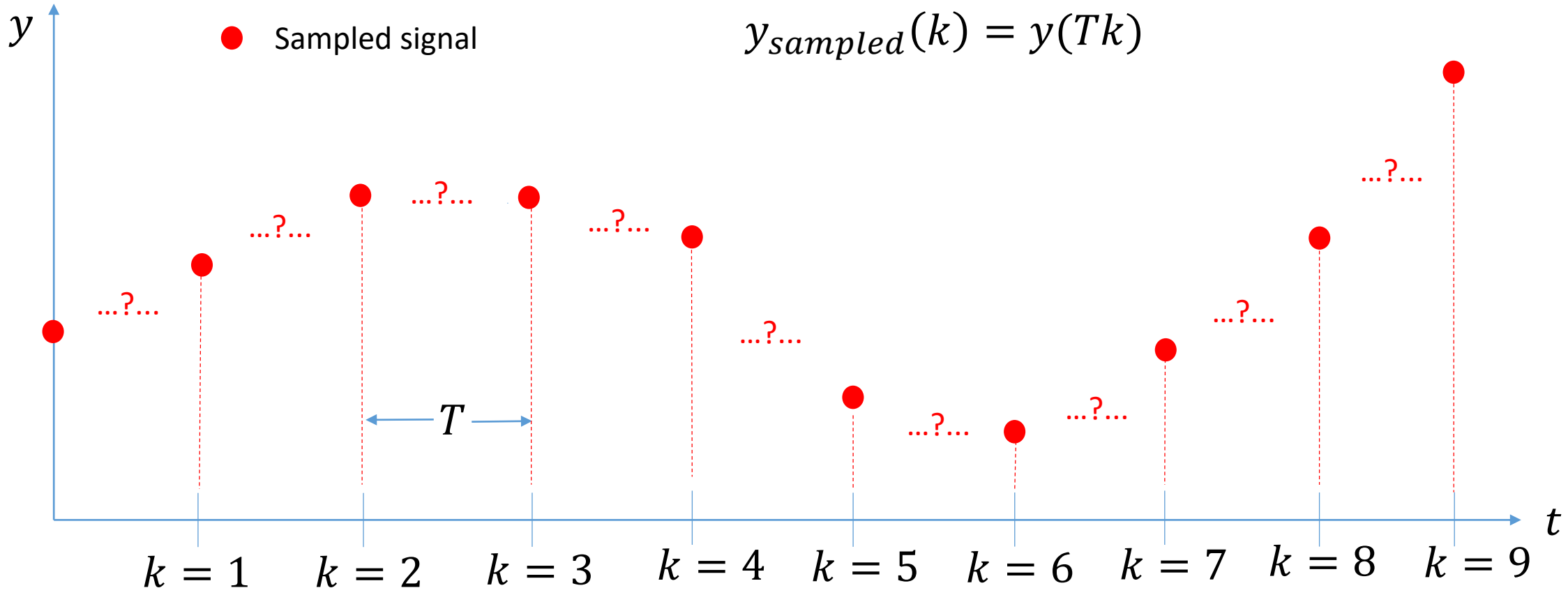
Converting to Discrete Time – Sampled Signals

Main idea: A **sampled** signal is obtained by evaluating a continuous signal at discrete, (usually) regularly-spaced time intervals of length T (T is called the **time step**). **Discrete-time system representations** model the evolution of this sampled signal over time. While the continuous-time signal is a function of time (t), the sampled signal is a function of time index (k)



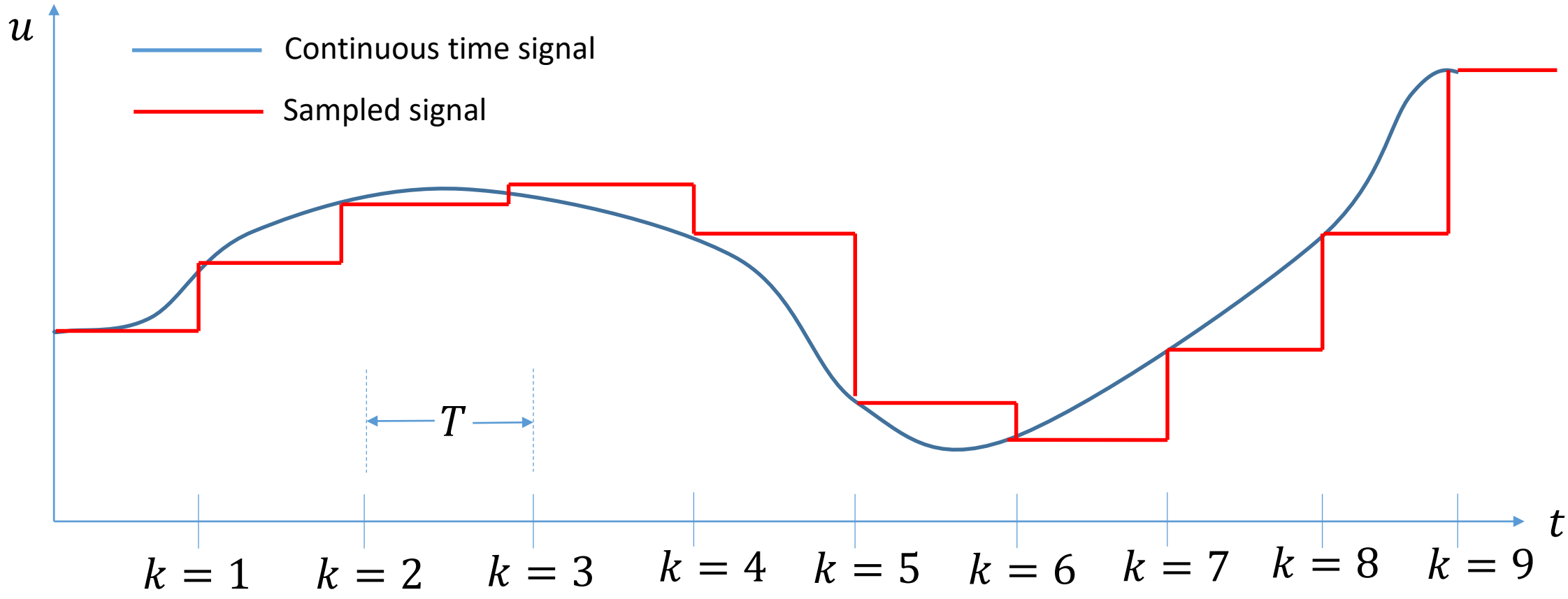
Converting to Discrete Time – Key Challenge

Main idea: The sampled signal does not provide any indication of what happens ***between*** sampling instances. ***Critical challenge:*** We need to predict the state at the next sampling instant based ***only*** on the state and control signal at the current sampling instant.



Converting to Discrete Time – Control Signals

Main idea: Unlike states ($\mathbf{x}$) and outputs (y), the control signal (u) is…well…under our control. So we can guarantee what happens ***during*** a sampling interval by ***forcing*** the control signal to remain constant over a sampling interval. Thus, $u(i)$ reflects the value of u from $t = Ti$ to $t = T(i+1)$.



Discrete-Time System Representations – The Forward Euler Approximation

UNC CHARLOTTE

Reminder: The continuous-time system model is given by $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), u(t))$

The forward Euler approximation is given by $\mathbf{x}(k+1) = \mathbf{x}(k) + T\dot{\mathbf{x}}(k) = \mathbf{x}(k) + f\big(\mathbf{x}(k), u(k)\big)T$

Linear system approximation:

Continuous-time system: $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t)$

Discrete-time approximation: $\mathbf{x}(k+1) = \mathbf{x}(k) + \big(A\mathbf{x}(k) + Bu(k)\big)T$

$= (I + AT)\mathbf{x}(k) + BTu(k)$

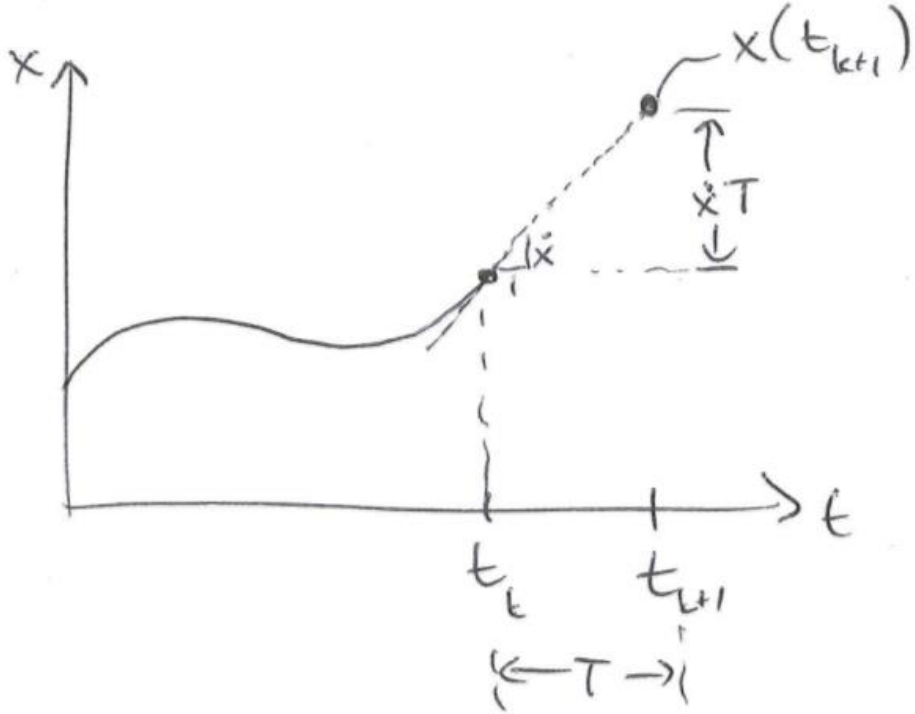
Discrete-Time System Representations – The Forward Euler Approximation

UNC CHARLOTTE

Key challenge: Given $\underline{x}(k)$ (which can evolve btwn t_k & t_{k+1}) and $u(k)$ (which remains constant btwn t_k & t_{k+1}), predict $\underline{x}(k+1)$

$\Rightarrow \underline{x}(k+1) = f(\underline{x}(k), u(k))$

Forward Euler (scalar x):



Discrete-Time System Representations – The Forward Euler Approximation

UNC CHARLOTTE

Linear system: $\dot{\underline{x}} = A_c \underline{x} + B_c \underline{u}$

want: $\underline{x}_{(k+1)} = A_d \underline{x}(k) + B_d u(k)$?

$$\underline{x}(k+1) = \underline{x}(k) + \dot{\underline{x}}(k) T$$

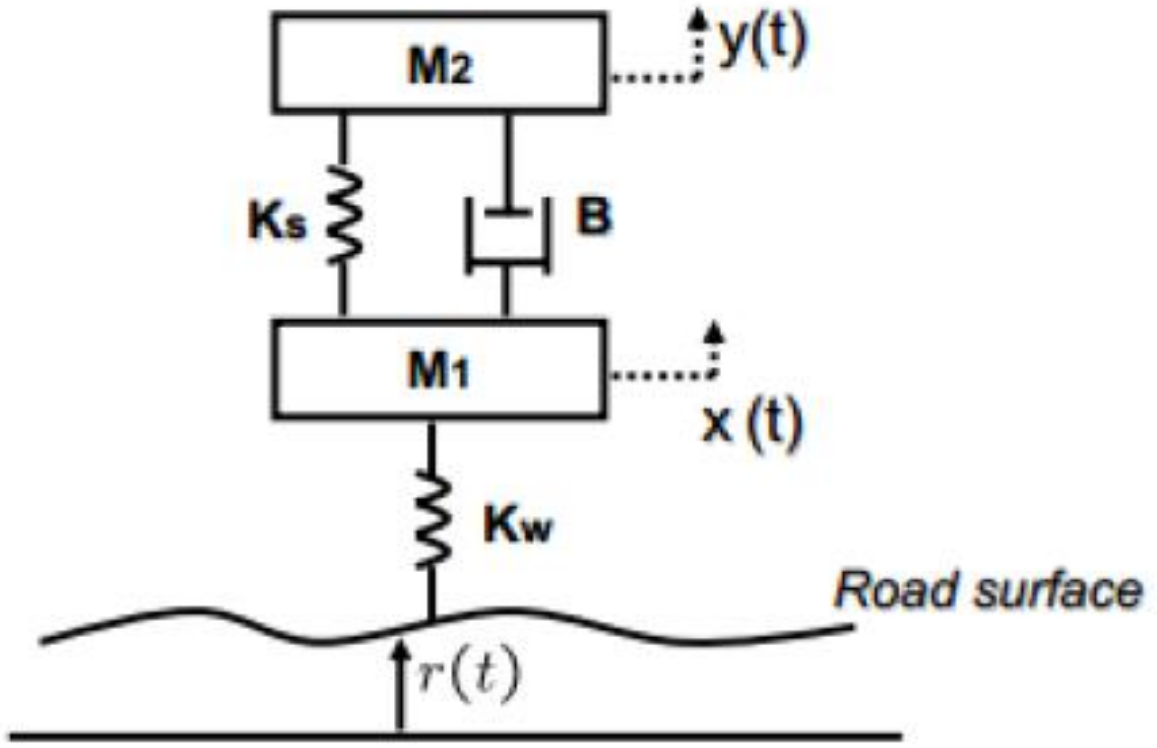
$$= \underline{x}(k) + T\left[A_c \underline{x}(k) + B_c u(k)\right]$$

$$= \underbrace{\left(\underline{I} + A_c T\right)}_{A_d} \underline{x}(k) + \underbrace{B_c T}_{B_d} u(k)$$

Deriving a Discrete-Time State Space Representation – Example

UNC CHARLOTTE

Class exercise: For the quarter car suspension model below, given the previously-derived continuous-time model, $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Br(t)$, derive a symbolic discrete-time approximation of the form $\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d r(k)$, as a function of the time step (T). Use a forward Euler approximation.



Discrete-Time System Representations for Linear Systems – The Zero Order Hold Approximation

UNC CHARLOTTE

Reminder: The continuous-time linear system model is given by $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t)$

Observation: The forward Euler approximation assumes $\mathbf{x}(t)$ remains constant until the next time step, whereas in reality, it evolves.

From your linear systems theory class (which may have been titled "Optimal Control I"), you were taught (I think) that the solution to the above system can be expressed in the following general form:

$$\mathbf{x}(t) = e^{A(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} e^{A(t-\tau)}Bu(\tau)d\tau$$

$e^{A(t-t_0)}$ is also known as the **state transition matrix**.

This allows you to determine $\mathbf{x}(k+1)$, assuming $u(t)$ is held constant over one step but **not** assuming that $\mathbf{x}(t)$ is constant over one step…this is called a **zero order hold approximation.**

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d u(k) \qquad \text{where:} \qquad A_d = e^{AT} \qquad B_d = \int_0^T e^{A(T-\tau)}Bd\tau$$

Discrete-Time System Representations for Linear Systems – The Zero Order Hold Approximation

UNC CHARLOTTE

$$\underline{x}(t_{k+1}) = e^{A(t_{k+1}-t_k)}\,\underline{x}(t_k) + \int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau)} B u(\tau)\, d\tau$$

$$= e^{A(t_{k+1}-t_k)}\,\underline{x}(t_k) + u(t_k)\int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau)} B\, d\tau$$

$$\underline{x}(k+1) = e^{AT}\,\underline{x}(k) + u(k)\int_0^T e^{A(T-\tau)} B\, d\tau$$

$$= \underbrace{e^{AT}}_{A_d}\,\underline{x}(k) + u(k)\underbrace{\left[e^{AT}\int_0^T e^{-A\tau} B\, d\tau\right]}_{B_d}$$

Calculating the Zero Order Hold Approximation by Hand

UNC CHARLOTTE

Key fact: The state transition matrix, e^{AT}, can be computed as:

$$e^{AT} = \mathcal{L}^{-1}\{(sI - A)^{-1}\}$$ ← **Follows from the Cayley-Hamilton Theorem**

Alternative method (generally easier): It also follows that e^{At} can be written as:

$$e^{AT} = \alpha_0 I + \alpha_1 A + \cdots + \alpha_{n-1} A^{n-1}$$ for scalar $\alpha_0 \ldots \alpha_{n-1}$

and: $e^{\lambda_i T} = \alpha_0 I + \alpha_1 \lambda_i + \cdots + \alpha_{n-1} \lambda_i^{n-1}$ for every eigenvalue, $\lambda_1 \ldots \lambda_n$

n equations for n unknowns

…It follows that e^{AT} can be calculated by (i) solving the n equations for n unknowns, then (ii) plugging the $\alpha_0 \ldots \alpha_{n-1}$ values into the equation above it

Calculating the Zero Order Hold Approximation by Hand

UNC CHARLOTTE

Remember: When λ's are complex, you can take:

$$e^{i\theta} = i\sin\theta + \cos\theta$$

Calculating the Zero Order Hold Approximation by Hand - Examples

UNC CHARLOTTE

Example: Compute the zero order hold approximation of the following continuous-time systems

System 1: $\dot{x} = ax + bu$

System 2: $\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -4 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$

Calculating the Zero Order Hold Approximation by Hand - Examples

UNC CHARLOTTE

Example 1:

$\dot{x} = \underbrace{a}_{A_c} x + bu$

$$A_d = e^{aT}$$

$$B_d = e^{aT} \int_0^T e^{-a\tau} b \, d\tau$$

$$= b e^{aT} \int_0^T e^{-a\tau} \, d\tau$$

$$= -b e^{aT} \left[\frac{1}{a} e^{-a\tau} \Big|_0^T \right]$$

$$= -b e^{aT} \left[\frac{1}{a} e^{-aT} - \frac{1}{a} \right]$$

$$= \frac{b}{a} e^{aT} \left[1 - e^{-aT} \right]$$

Calculating the Zero Order Hold Approximation by Hand - Examples

UNC CHARLOTTE

Example 2:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -4 & -5 \end{bmatrix}}_{A_c} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B_c} u$$

$$\lambda I - A_c = \begin{bmatrix} \lambda & -1 \\ 4 & \lambda+5 \end{bmatrix}$$

Char. eq: $\lambda(\lambda+5)+4=0$

$\Rightarrow (\lambda+1)(\lambda+4)=0$

$\lambda_1 = -1, \lambda_2 = -4$

$e^{-T} = \alpha_0 - \alpha_1$

$e^{-4T} = \alpha_0 - 4\alpha_1$

Calculating the Zero Order Hold Approximation by Hand - Examples

UNC CHARLOTTE

$$3\alpha_1 = e^{-T} - e^{-4T} \Rightarrow \alpha_1 = \frac{1}{3}\left[e^{-T} - e^{-4T}\right]$$

$$3\alpha_0 = 4e^{-T} - e^{-4T} \Rightarrow \alpha_0 = \frac{4}{3}e^{-T} - \frac{1}{3}e^{-4T}$$

$$\Rightarrow e^{AT} = A_d = \alpha_0 I + \alpha_1 A$$

$$= \begin{bmatrix} \frac{4}{3}e^{-T} - \frac{1}{3}e^{-4T} & 0 \\ 0 & \frac{4}{3}e^{-T} - \frac{1}{3}e^{-4T} \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{3}(e^{-T} - e^{-4T}) \\ -\frac{4}{3}(e^{-T} - e^{-4T}) & -\frac{5}{3}(e^{-T} - e^{-4T}) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{4}{3}e^{-T} - \frac{1}{3}e^{-4T} & \frac{1}{3}(e^{-T} - e^{-4T}) \\ -\frac{4}{3}(e^{-T} - e^{-4T}) & -\frac{1}{3}e^{-T} + \frac{4}{3}e^{-4T} \end{bmatrix}$$

Calculating the Zero Order Hold Approximation by Hand - Examples

UNC CHARLOTTE

$$B_d = \int_0^T e^{A_c(T-\tau)} B_c \, d\tau$$

$$= e^{A_c T} \int_0^T \underbrace{e^{-A_c \tau}}_{2\times 2} \underbrace{B_c}_{2\times 1} d\tau \quad \left(\text{note: } e^{-A_c\tau} = e^{A_c(-\tau)}\right)$$

✓

$$= e^{A_c T} \int_0^T \begin{bmatrix} \frac{1}{3}(e^{\tau} - e^{4\tau}) \\ -\frac{1}{3}e^{\tau} + \frac{4}{3}e^{4\tau} \end{bmatrix} d\tau$$

$$= e^{A_c T} \begin{bmatrix} \frac{1}{3}e^{\tau} - \frac{1}{12}e^{4\tau} \\ -\frac{1}{3}e^{\tau} + \frac{1}{3}e^{4\tau} \end{bmatrix}_0^T$$

$$= e^{A_c T} \begin{bmatrix} \frac{1}{3}e^{T} - \frac{1}{12}e^{4T} - \frac{1}{4} \\ -\frac{1}{3}e^{T} + \frac{1}{3}e^{4T} \end{bmatrix}$$

Discrete-Time System Representations for Linear Systems – Using MATLAB

UNC CHARLOTTE

Reminder: The continuous-time linear system model is given by $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t)$

The following MATLAB syntax can be used to derive the discrete-time representation:

```
>> sys_continuous = ss(A,B,C,D);
>> Ts = .1;
>> sys_discrete = c2d(sys_continuous,Ts,'zoh')
```

This is the method used for discretization…
'zoh' stands for "zero order hold."

Discrete Time Linear System Analysis

UNC CHARLOTTE

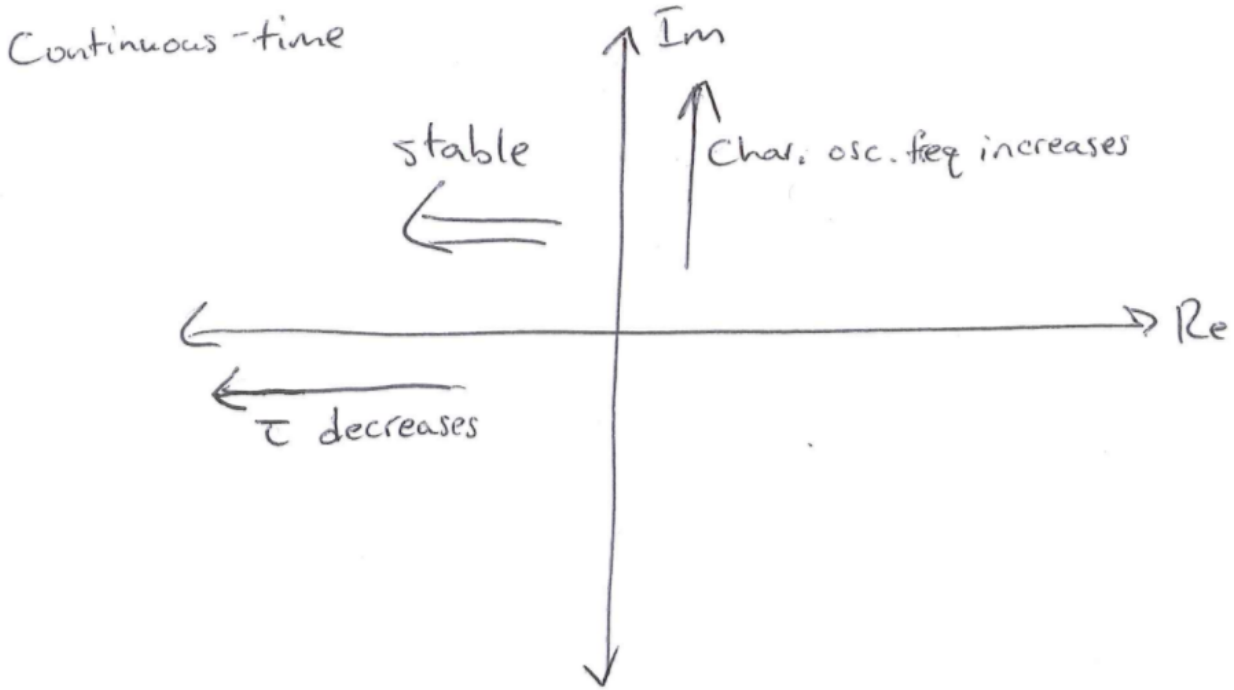
Stability: A discrete-time linear system, $\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)$, is ***asymptotically stable*** (i.e., the origin is asymptotically stable in the absence of an input) if and only if all eigenvalues of A lie within the **unit circle**, i.e., $\|\lambda_i(A)\| < 1, i = 1 \ldots n$

Time constants: An eigenvalue at $\lambda = a + bi$ has an associated magnitude ($\|\lambda\|$) and phase (ϕ), which determine the time constant and characteristic oscillation frequency, respectively…***note: Eigenvalue locations depend on the time step!***

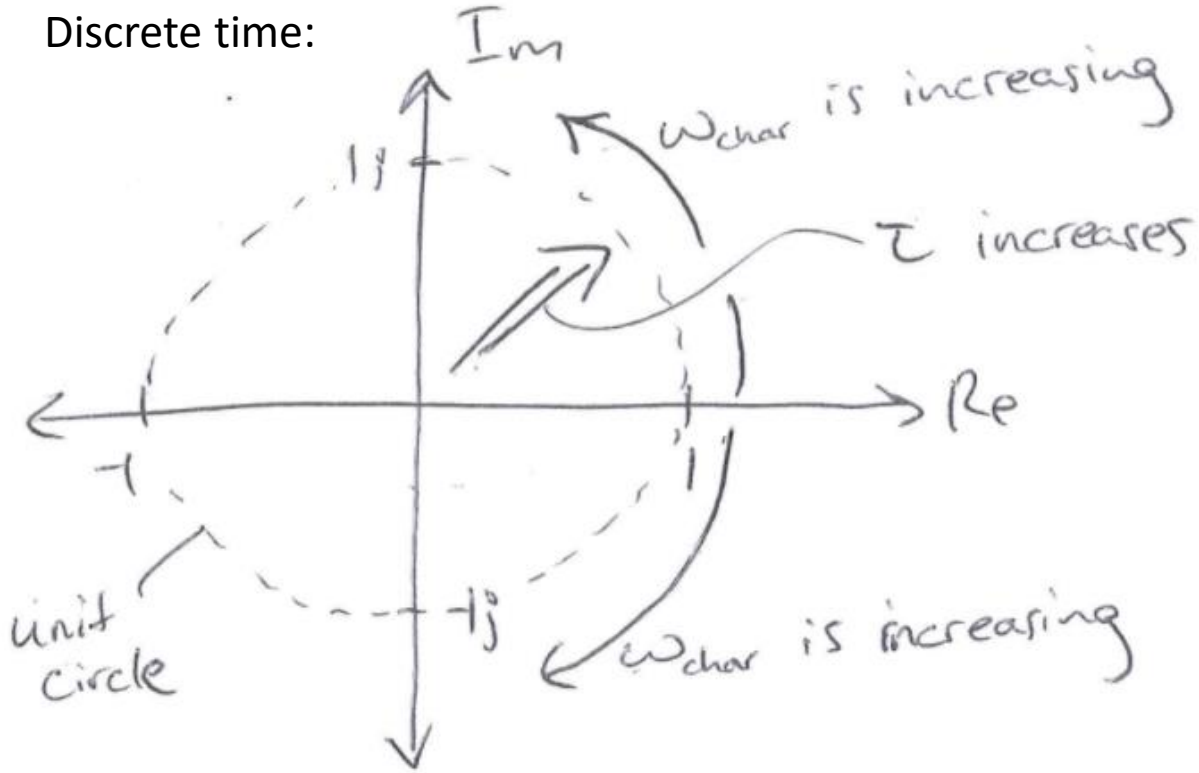
Time constants: Given a **stable** eigenvalue with magnitude $\|\lambda\|$, the associated time constant is given by $\tau = -\frac{T}{\ln(\|\lambda\|)}$

Characteristic oscillation frequencies: Given an eigenvalue with phase ϕ (or complex conjugate with phase $-\phi$), the associated characteristic oscillation frequency is given by $\omega = \frac{\phi}{T}$

Discrete Time Linear System Analysis

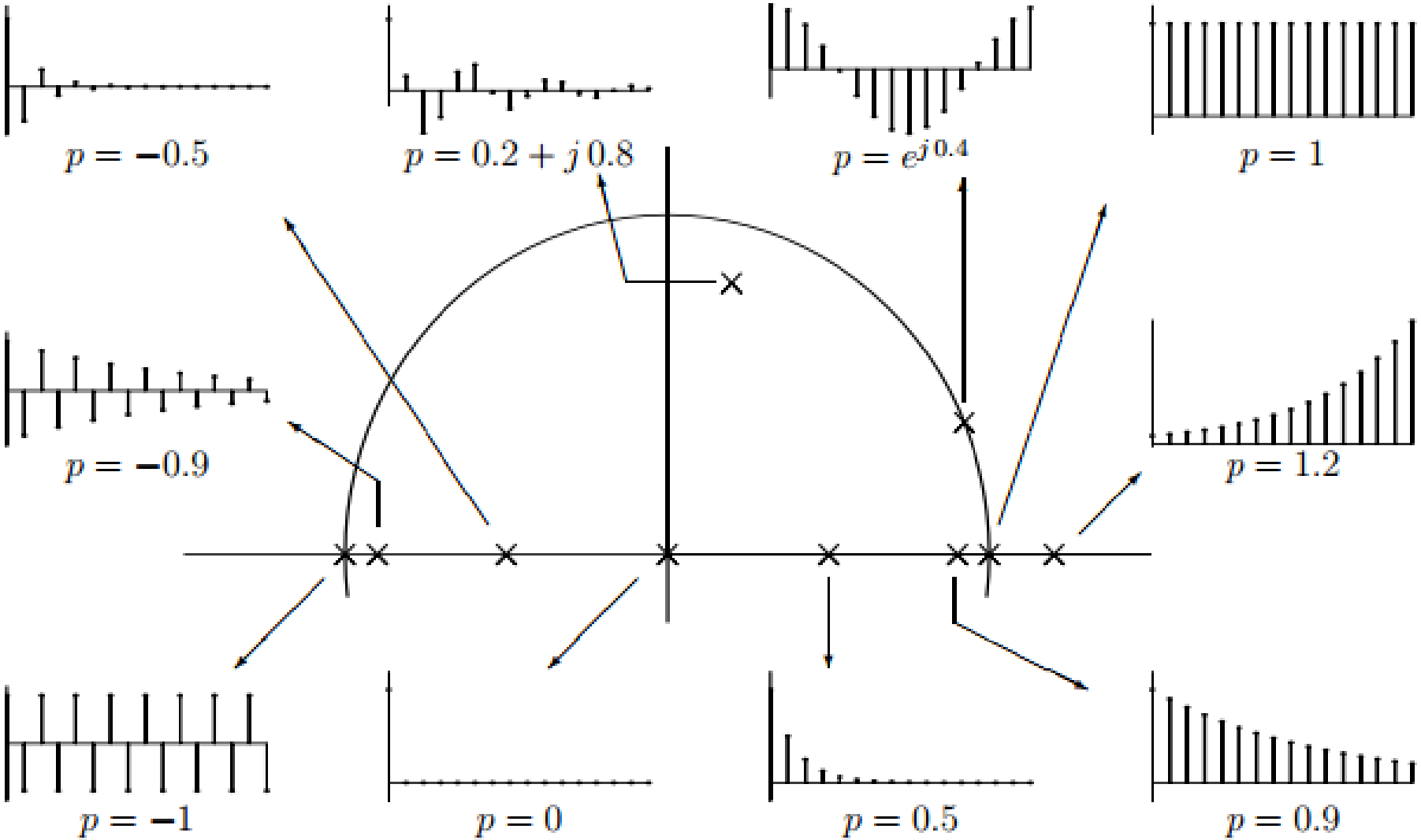


Discrete time:



Discrete Time Linear System Analysis – Graphical Interpretation

UNC CHARLOTTE



Preview of next lecture (and beyond)

UNC CHARLOTTE

Topics for lecture 4:

- Setting up and solving a simple finite-dimensional **design optimization** problem
- Drawing parallels between finite-dimensional **design optimization** and optimization of a **finite sequence of control signals**

Beyond next lecture, we will examine several different techniques for discrete-time optimal control (optimizing a finite sequence of control signals):

- Convex optimization techniques (gradient approaches, Newton's method, sequential quadratic programming (SQP))
- Dynamic programming for global optimization