Transformations between mathematical models of systems

1) ODE ↔ SS

- Any **linear nth order ODE with constant coefficients** can be expressed as a **system of n first-order linear ODEs.**
- In other words, any linear physical system has a non-unique linear state space representation.
- **Example (n state variables, no input)**

$$a_n x^{(n)} + a_{n-1}x^{(n-1)} + \cdots + a_2\ddot{x} + a_1\dot{x} + a_0 x = b, \qquad a_n \neq 0$$

$$\text{substitute: } \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_n \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \\ \dots \\ x^{(n-1)} \end{bmatrix} \Longrightarrow \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_n \end{bmatrix} = \begin{bmatrix} z_1 \\ \dot{z}_1 \\ \dots \\ \dot{z}_{n-1} \end{bmatrix}$$

$$\dot{z}_n = x^{(n)} = \frac{b - a_0 x - a_1\dot{x} - \cdots - a_{n-1}x^{(n-1)}}{a_n} = \frac{b - a_0 z_1 - a_1 z_2 - \cdots - a_{n-1}z_n}{a_n}$$

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dots \\ \dot{z}_n \end{bmatrix} = \begin{bmatrix} 0 & & \\ 0 & & \mathrm{I}_{(n-1)\,\mathrm{x}\,(n-1)} \\ \dots & & \\ -a_0/a_n & \dots & -a_{n-1}/a_n \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dots \\ b/a_n \end{bmatrix}$$

- **Inverse transfer:** Try representing one of the n equations using any z_i and its derivatives.

2) ODE ↔ TF

- **Used along with (3) to transform ODE with input to State Space.**

$$a_n x^{(n)} + a_{n-1}x^{(n-1)} + \cdots + a_0 x = b_m u^{(m)} + b_{n-1}u^{(m-1)} + \cdots + b_0 u, \qquad a_n \neq 0, \qquad b_n \neq 0$$

- **Take Laplace transform of both sides**

$$(a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0)X(s) = (b_m s^m + b_{m-1}s^{m-1} + \cdots + b_0)U(s)$$

- **Transfer function:**

$$H(s) = \frac{X(s)}{U(s)} = \frac{(b_m s^m + b_{m-1}s^{m-1} + \cdots + b_0)}{(a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0)}$$

- **Inverse transfer:** replace s^m with $u^{(m)}$ and s^n with $x^{(n)}$ and cross-multiply the last equation.

3) TF ↔ SS (Controllable Canonical Form)

- If the coefficient of s^n in the denominator is $a_n \notin \{0,1\}$, multiply $H(s)$ by $\frac{1/a_n}{1/a_n}$

$$H(s) = \frac{Y(s)}{U(s)} = \frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \Longrightarrow \begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

$$A = \begin{bmatrix} 0 & & \\ 0 & & \mathrm{I}_{(n-1)\mathrm{x}\,(n-1)} \\ \dots & & \\ -a_0 & \dots & -a_{n-1} \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \dots \\ b_0 \end{bmatrix}$$

$$C = [b_0 - a_0 b_n \quad b_1 - a_1 b_n \quad \dots \quad b_{n-2} - a_{n-2}b_n \quad b_{n-1} - a_{n-1}b_n], D = [b_n]$$

- **Inverse transfer**:
  - $\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \Longrightarrow \begin{cases} sIX - AX = BU \\ Y = CX + DU \end{cases} \Longrightarrow X = (sI - A)^{-1}BU \Longrightarrow Y = (C(sI - A)^{-1}B + D)U$
  - Therefore: $H(s) = C(sI - A)^{-1}B + D$

Lecture 1

- **Control system:** manages, directs, or regulates the behavior of other devices or systems using control loops
    - **Control loops:** physical components and control functions necessary to automatically adjust the value of a measured process variable (PV) to equal the value of a desired set-point (SP).
        - **Open loop control:** control action is independent from process output.
        - **Closed loop control:** control action from the controller is dependent on feedback from the process in the form of a process variable.
- **Control theory:** concerned with the strategies to select the appropriate input for a control system.
- **Dynamical system**
    - A system in which a function $x(t)$ describes how a point in a geometrical space changes position over time t.
    - Examples include the mathematical models that describe the swinging of a clock pendulum, the flow of water in a pipe, etc.
    - At any given time, the system has a **state (vector of real numbers)** that can be represented by a point in an appropriate **state space**.
        - **State space** of a system is the set of all possible configurations of the system.
- **ODEs are sometimes used to represent (continuous) dynamical systems.**
    - **Example 1:** $\dot{x} = f(x,t)$
        - $x = x(t)$ is the solution of the equation
        - t is a free variable (representing time).
        - $\{x\}$ is the state space of the system.
    - **Example 2:** $x^{(n)} = f(x^{(n-1)}, x^{(n-2)}, \dots, \ddot{x}, \dot{x}, x, t)$
        - $\{x^{(n-1)}, x^{(n-2)}, \dots, \ddot{x}, \dot{x}, x\}$ is the state space of the system.
    - **Example 3 (spring):** $F = ma \Leftrightarrow u(t) - kx = m\ddot{x}$
- **State Space representation**
    - Represents a dynamical system as a set of 1st order ODEs that can be analyzed using the methods of LA.
    - A linear system of p inputs (u), q outputs (y), and n state variables (x) have the following representation

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \Leftrightarrow \begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dots \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix} + \begin{bmatrix} b_{11} & \dots & a_{1p} \\ \dots & \dots & \dots \\ b_{n1} & \dots & a_{np} \end{bmatrix} \begin{bmatrix} u_1 \\ \dots \\ u_p \end{bmatrix} \\ \begin{bmatrix} y_1 \\ \dots \\ y_q \end{bmatrix} = \begin{bmatrix} c_{11} & \dots & c_{1n} \\ \dots & \dots & \dots \\ c_{q1} & \dots & c_{qn} \end{bmatrix} \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix} + \begin{bmatrix} d_{11} & \dots & d_{1p} \\ \dots & \dots & \dots \\ d_{q1} & \dots & d_{qp} \end{bmatrix} \begin{bmatrix} u_1 \\ \dots \\ u_p \end{bmatrix} \end{cases}$$

    - 1st equation describes the dynamics of the system, how new states are obtained from current states and an input.
        - Matrix $A_{n \times n}$ describes how the internal states are all connected to each other.
        - Matrix $B_{n \times p}$ describes how the input enters the system, which states it affects.
    - 2nd equation describes the output of the system, and how we can control.
        - Matrix $C_{q \times n}$ describes how the states are combined to get the output.
        - Matrix $D_{q \times p}$ accounts for the feedforward (discussed later).
- **Check ODE ↔ SS transformation (1st page)**

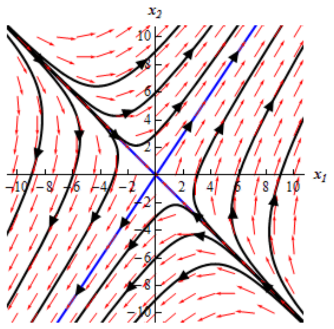
Lecture 2

- **Stability of dynamical systems**
  - Stability is a condition in which a slight disturbance in a system does not produce too disrupting effect on that system.
    - Consider applying small force to a pendulum, it returns to stable state quickly.
  - The node(s) or critical point(s) of a dynamical system is the set of its equilibrium states.
  - [Check] A dynamical system described by an ODE is called
    - **Stable**: if the solutions with different initial conditions don't diverge from the node.
      - **Example**: solution $x = 7$ to the system $\dot{x} = 0$
    - **Asymptotically stable**: if the solutions with different initial conditions converge to the node.
      - **Example**: solution $x = 0$ to the system $\dot{x} = -x$
        - Other solutions converge to $x = 0$ as $t \to \infty$
    - **Unstable**: Otherwise
      - **Example**: solution $x = 0$ to the system $\dot{x} = x$ (other solutions diverge).
- **Linear Time Invariant system (LTI):**
  - A dynamical system that has 2 properties:
    - **Linearity (homogeneity and superposition):**
      - $\left(x_1(t) \Rightarrow y_1(t) \wedge x_2(t) \Rightarrow y_2(t)\right) \Longrightarrow \left(ax_1(t) + bx_2(t) \Rightarrow ay_1(t) + by_2(t)\right)$
    - **Time invariance:** $x(t-a) \Rightarrow y(t-a)$
  - The output of such systems can be calculated by the formula $y(t) = x(t) * h(t)$ where '*' denotes the convolution operation and $h(t)$ is the impulse response function (output when taking a brief input).
    - This brief input (impulse) is called Dirac delta input: $x(t) = \delta(t) \Longrightarrow y(t) = h(t)$
    - $h(t)$ is the inverse Laplace transform of the system transfer function (check next lecture).
  - Since convolution is not easy to compute, Laplace transform is being used
    - $y(t) = L^{-1}\big(H(s)X(s)\big), X(s) = L(x(t)), H(s) = L(h(t))$
  - **Example:** $\dot{x} = Ax + Bu$
    - u represents input, $\dot{x}$ represents output, x is the state (lecture notation).
- **Autonomous LTI system** evolution depends only on the state of the system (has no input u): $\dot{x} = Ax$
  - Autonomous LTI is stable ⇔ real parts of eigenvalues of A are **non-positive.**
    - Eigenvalues (λ) can be computed by solving $det(A - \lambda I) = 0$
    - Eigenvectors are computed by solving $(A - \lambda I)x = 0$ for each eigenvalue.
  - Autonomous LTI is asymptotically stable ⇔ real parts of eigenvalues of A are **negative**.
  - Unstable otherwise
  - The system can be solved analytically (S is the matrix of linearly independent eigenvectors).

$$x(t) = e^{At}x(0) = \mathcal{L}^{-1}\{(sI - A)^{-1}\}\,x(0) = Se^{\Lambda t}S^{-1}x(0),$$

$$e^{\Lambda t} = \begin{pmatrix} e^{\lambda_1 t} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & e^{\lambda_n t} \end{pmatrix}, S = [v_1 \quad \cdots \quad v_n]$$

- **Phase portraits:** a geometric representation of a trajectory (integral curves) of a dynamical system.
  - **Example:** a system of two ODEs $\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ can be represented as a vector field, with x_1 on x-axis and x_2 on the y-axis.



  - This gives an impression about the stability of the system by checking the common trajectory of the vectors [Check graphs]
    - If they converge to the same point, the system is stable and asymptotically stable.
    - If they don't converge to the same point, but don't diverge to infinity, the system is stable.
    - If they diverge to infinity, the system is unstable.
    - If they form a closed elliptic (or circular) path, the system is (Lyapunov) stable.
      - Solutions starting near to the equilibrium remain near it forever.

Lecture 3

- **Transfer Function (TF)**
  - Laplace transform of the impulse response $h(t)$ of a LTI system when $x(0) = 0$
    - Impulse response is the output when input = $\delta(t)$
  - **Laplace transform (Table):**
    - $L\big(f(t)\big) = F(s) = \int_0^{+\infty} f(t)e^{-st}dt$
    - $L\big(\delta(t)\big) = 1$
    - $L(x'(t)) = sL\big(x(t)\big) = sx(s) - x(0)$
    - $L(x''(t)) = s^2L\big(x(t)\big) - sx(0) - x'(0) = s^2x(s) - sx(0) - x'(0)$
  - **Useful properties of LT:**
    - **Linearity**
      - $L(c_1y_1 + \cdots + c_ny_n) = c_1L(y_1) + \cdots + c_nL(y_n)$, same with $L^{-1}(y)$
    - **First shifting theorem**
      - $L\big(e^{at}f(t)\big) = F(s-a) \Longrightarrow L^{-1}\big(F(s-a)\big) = e^{at}L^{-1}\big(F(s)\big)$
      - The substitution $S = s - a$ in the transform corresponds to the multiplication of the original function by e^{at}
    - **Second shifting theorem**
      - $L(f(t-a)I_{t>a}) = e^{-as}\,L\big(f(t)\big) = e^{-as}F(s)$
      - Laplace transformation of a function with shifted parameter $T = t - a$ is equal to the transformation of the unshifted function multiplied by e^{-as}
    - **Change of scale property:**
      - $L\big(f(at)\big) = \frac{1}{a}F\left(\frac{s}{a}\right) \Longrightarrow L^{-1}\left(F\left(\frac{s}{a}\right)\right) = af(at)$
    - **Final value Theorem**
      - $f(\infty) = \lim_{s\to 0} sF(s)$
  - Transfer function is the (unique) linear mapping of the Laplace transform of the input (denoted as $U(s)$) to the Laplace transform of the output (denoted as $Y(s)$).
    - **Mathematically:** $H(s) = \frac{Y(s)}{U(s)}$
- **Check transformations (1st page).**

Lecture 4

- **Two classes of problems in control:**
  - **Stabilization (regulation):** the problem of controlling the system towards a static point.
  - **Trajectory tracking:** the problem of controlling the system to reach and follow a time-varying trajectory.
- **Stabilizing control problem**
  - The problem of finding such a control law u that makes a certain dynamical system $\dot{x} = f(x, u)$ **stable**.
    - **Stable** here means that all solutions converge to a certain solution x^* (for asymptotic stability) or don't diverge from x^* (Lyapunov stability)
      - If not specified, we want all solutions to converge to $x^* = 0$ (the trivial solution) .
  - **Linear control law**
    - For linear dynamical systems on the form $\dot{x} = Ax + Bu$, we choose $u = -Kx$
    - The system becomes $\dot{x} = (A - BK)x$ which is autonomous LTI that can be analyzed easily.
      - We choose such K that makes the real parts of eigenvalues of $(A - BK)$ negative.
        - **Methods to construct K? Check Pole Placement, LQR below.**
      - K may or may not exist, in the latter case, the system can't be stabilized.
  - **Affine control law**
    - For affine systems on the form $\dot{x} = Ax + Bu + c$ we choose $u = -Kx + u^*,\ u^* = const$
    - The system becomes $\dot{x} = (A - BK)x + Bu^* + c$
      - We choose such K, u^* that makes
        - The real parts of eigenvalues of $(A - BK)$ negative
        - $Bu^* = -c$
      - Both K, u^* may or may not exist.
- **Full state feedback:** assuming that we know (can measure) the state of the system at any point in time, we can feed it back to the plant after multiplying it by the "gain" matrix K and possibly shifting it, to get the system to converge the desired static point.
  - **Pole placement**
    - A useful technique for constructing the gain matrix K that stabilizes the system.
    - Works only for single input systems, (i.e., u is a column matrix), for multiple inputs, use LQR.
    - **Method**
      - Set values $\lambda_i < 0$ for the poles (eigenvalues of A), we usually choose values close to the imaginary axis (i.e., -1, -2, …).
      - We need to have $\prod_{i=1}^{n}(x - \lambda_i) = 0$, expand this to get a polynomial $P(\lambda)$
      - Solve $\det(A - BK - \lambda I) = P(\lambda)$ for the matrix K.
      - The matrix we obtained is one of the possible gains, but not necessarily the optimal one.
  - **Linear-Quadratic Regulator (LQR)** – **check lecture 9**
    - Finds the optimal K, by introducing a cost function, that adds up the weighted sum of performance and effort, then solving it for the gain matrix
    - Choosing the appropriate weights is more intuitive that choosing poles.
  - **Root locus**: a graphical representation of the eigenvalues of $(A - BK)$ varying one component of K.
- **Trajectory tracking control:**
  - We need to find such u that makes all solutions converge to some non-trivial solution $x^*(t) \neq 0$
  - In other words, we need to find the stabilizing control u knowing that $\dot{x}^* = Ax^* + Bu^*$ where u^* is some control law that makes x^* a solution (we call it the feedforward).
    - **Linear control:** $u^* = B^+(\dot{x}^* - Ax^*)$ where B^+ is the **pseudoinverse** of B
    - **Affine control:** $u^* = B^+(\dot{x}^* - Ax^* - c)$
  - We can reduce this problem to finding the stabilizing control for the **error dynamics**

Pseudoinverse for non-invertible matrices can be calculated as follows:

- If $A_{n\times m}$ has a full rank $(Rank(A) = \min\{n, m\})$
  - **Left inverse:** $A^+_{left} = (A^\top A)^{-1}A^\top \Longrightarrow A^+_{left}A = I$
  - **Right inverse:** $A^+_{right} = A^\top(AA^\top)^{-1} \Longrightarrow AA^+_{right} = I$
  - For square matrices $A^+_{left} = A^+_{right}$, otherwise, only one of them exists.

- Otherwise, use SVD
  - $A = U\Sigma V^\top \Longrightarrow A^+ = U\Sigma^+V^\top$
  - Σ^+ is the same as Σ with inverted diagonals (zeros are kept).

- **Error dynamics:**
    - **Control error $\boldsymbol{e}$:** the difference between the stable state (the solution we want to converge to) and the current state: $\boldsymbol{e} = \boldsymbol{x}^* - \boldsymbol{x}$
    - We can calculate the dynamics equation for the error:
        - $\begin{cases} \dot{x^*} = Ax^* + Bu^* \\ \dot{x} = Ax + Bu \\ e = x^* - x \end{cases} \implies \dot{\boldsymbol{e}} = A\boldsymbol{e} + B(u - u^*) \implies \dot{\boldsymbol{e}} = A\boldsymbol{e} + Bv$
        - Now the problem reduces to the trivial case, we set $v = -Ke$ and find such K that makes the real parts of eigenvalues of $(A - BK)$ negative, finally we get the control law:

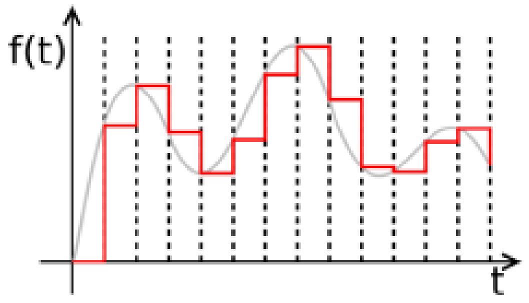
$$u = u_{FB} + u_{FF} = -K(x - x^*) + u^*$$

    - **Feed-Forward control** manipulates the input towards the goal state x^*.
        - Depends on the trajectory and equations of dynamics, but not the current state.
    - **Feed-Back control**, measures the output (or state) to feedback a manipulated input.
        - Depends on the control error.
- **Point-to-point control**
    - Moving the system from some initial state x to some desired state x^* that is not necessarily a solution.
    - We use the fact that $x = x^* \implies \dot{x} = 0$ provided that $x^* = const$
        - **Linear control:** $u^* = -B^+ A x^*,\ \ u = -K(x - x^*) + u^*$
        - **Pure state feedback control:** $u^* = -B^+(A - BK)x^*,\ \ u = -Kx + u^*$

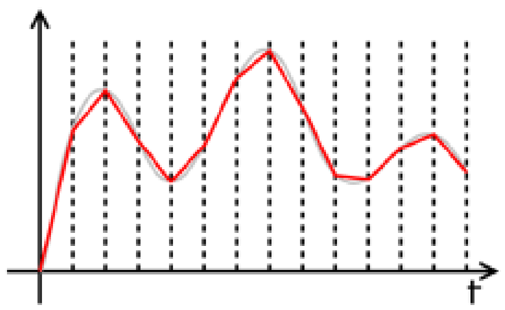
Lecture 5

- **Discrete dynamical systems** (in contrast with continuous systems): states are countable, no derivatives are involved (rather difference equations) and the systems are easily simulated.
  - **Linear discrete dynamical system**: $x_{i+1} = Ax_i + Bu_i$
- **Stability criterion for discrete autonomous systems**
  - The system $x_{i+1} = Ax_i$ is **stable** if the eigenvalues of A don't exceed 1 by absolute value.
    - Thus, for pole placement method, we place them such that they lie inside the unit circle.
    - **Mathematically,** $\forall \lambda_k \in eigenvalues\{A\}: \sqrt{Re(\lambda_k)^2 + Im(\lambda_k)^2} \leq 1$
      - If one of them = 1 and the rest are negatives, the system is Lyapunov stable.
- **Discretization:** going from the continuous s-domain to the discrete z-domain.
  - To start thinking in discrete case, we can define a constant unit difference **(sampling time)** Δt between states such that $x_0 = x(0), x_1 = x(\Delta t), \ldots, x_n = x(n\Delta t)$
    - In continuous case, Δt is an infinitesimal.
    - It's also usually denoted as T.
  - In this case, the derivative is analogous to the difference b/w two consecutive states divided by Δt
    - $\dot{x} \sim \frac{x(t+\Delta t)-x(t)}{\Delta t} = \frac{x_{i+1}-x_i}{\Delta t}$
  - **Continuous LTI system** $\dot{x} = Ax + Bu$ can be rewritten as $x_{i+1} = \bar{A}x_i + \bar{B}u_i$ where the definitions for $\bar{A}$ and $\bar{B}$ depends on the method of discretization used.
- **Discretizing analog (continuous) signal**
  - **Exact discretization:** represents the exact values of $\bar{A}$ and $\bar{B}$ that can then be evaluated at specific points in time to discretize. We say that the discretization is exact if the following holds:
    - $\bar{A} = e^{A\Delta t} = \mathcal{L}^{-1}\{(sI - A)^{-1}\}_{t=\Delta t}$
    - $\bar{B} = \left(\int_{t_0}^{t_0+\Delta t} e^{As} ds\right) B = A^{-1}(\bar{A} - I)B$
  - **Zero order hold:** holds the last measured value until a new one arrives.
    - $\bar{A} = I + TA$
    - $\bar{B} = B\Delta t$
  - **First-order hold:** constructs a piecewise-linear approximation to the signal being sampled.
  - **Bilinear (Tustin) transform.**
    - $\bar{A} = (I + 0.5TA)(I - 0.5TA)^{-1}$
    - $\bar{B} = B\Delta t$

Graphically, zero order hold is this:

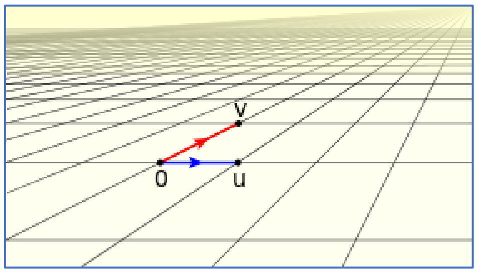


First order hold is this:



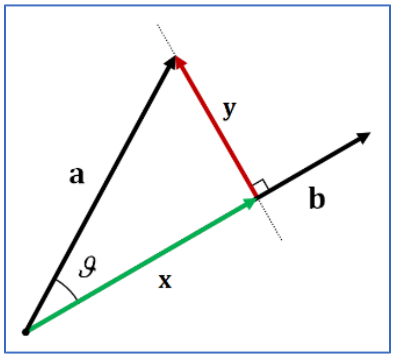
Lecture 6 - 7

- **Definitions from linear algebra.**
  - **Vector space V over a field F:** a set of vectors with two defined operations on these vectors, addition and scalar multiplication.
    - $v_i \in V, v_j \in V \Longrightarrow v_i + v_j \in V$
    - $a \in F, v \in V \Longrightarrow av \in V$
  - **Subspace** S of some vector space V is simply a subset of V.
  - **(Linear) span of a vector space:**
    - The set of all possible linear combinations of the vectors in the vector space.
  - **Basis B of a vector space V (or subspace):**
    - (Minimal) set of linearly independent vectors that span the whole vector space (i.e., any $v \in V$ can be expressed as a linear combination of vectors in B)
    - Dimension of a subspace is the number of vectors in its basis.
  - **Orthogonal** vectors = perpendicular vectors
  - **Orthonormal** vectors = perpendicular unit vectors.
    - **Orthonormal** matrix: matrix with orthonormal column vectors (and hence, rows).
    - **Orthogonal** matrix: square orthonormal matrix
    - Some sources write that they are the same.
  - **Properties of Q**
    - $QQ^\top = Q^\top Q = I \Longrightarrow Q^\top = Q^{-1}$
    - A vector v is said to be orthogonal to a subspace S if $v \cdot w = 0$ for each $w \in basis(S)$
      - In other words, the closest vector in w to v is the zero vector.
    - An orthogonal subspace to S, denoted as $S^\perp$ is the space satisfying
      - $\forall a \in S,\ \forall b \in S^\perp \Longrightarrow a \perp b$



Example:
- V = the coordinate space $\mathbb{R}^2$
- $S = \{\boldsymbol{u}, \boldsymbol{v}\}$ is a subspace of V over $\mathbb{R}$
- $span(S) = \{a\boldsymbol{u} + b\boldsymbol{v} \mid a, b \in \mathbb{R}\}$ = the depicted plane.
- If $u = (1,0), v = (0,1)$ then S forms a basis for V (linearly independent spanning set, since every vector in V can be expressed as a linear combination of elements in S)

- **Projection:** taking a vector v that may not belong to some space S and producing another one (its projection/shadow that lies in S), mathematically: $Proj_S(v) \in S$
  - Let $(\boldsymbol{x} \in \mathcal{L}, \boldsymbol{y} \in \mathcal{L}^\perp \Rightarrow \boldsymbol{y} \perp \boldsymbol{x}), \boldsymbol{a} = \boldsymbol{x} + \boldsymbol{y}$
    - Assume vectors are n-dimensional.
  - **We say:**
    - $\boldsymbol{x}$ is the projection of a onto subspace $\mathcal{L}$ $\left(\text{i.e. } \boldsymbol{x} = proj_{\mathcal{L}}(\boldsymbol{a})\right)$
    - $\boldsymbol{y}$ is the projection of a onto $\mathcal{L}^\perp$ $\left(\text{i.e. } \boldsymbol{y} = proj_{\mathcal{L}^\perp}(\boldsymbol{a})\right)$
    - $\boldsymbol{y} = \boldsymbol{0} \Rightarrow \boldsymbol{a} \in \mathcal{L} \Rightarrow proj_{\mathcal{L}}(\boldsymbol{a}) = \boldsymbol{a}$
  - Let L be an orthonormal basis of $\mathcal{L}$, then
    - $proj_{\mathcal{L}}(\boldsymbol{a}) = \boldsymbol{x} = \sum_{i=1}^{n}(\boldsymbol{a} \cdot L_i)L_i = LL^+\boldsymbol{a} = LL^\top\boldsymbol{a}$
    - $\boldsymbol{y} = (I - LL^\top)\boldsymbol{a}$
  - **Example:**
    - $\boldsymbol{x} = (0,1,0)^\top, \boldsymbol{y} = (0,0,1)^\top, \boldsymbol{a} = (0,1,1)^\top$
    - $L = \{(0,1,0)^\top, (1,0,0)^\top\} \equiv \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}$
    - $proj_{\mathcal{L}}(\boldsymbol{a}) = 1 * (0,1,0)^\top + 0 * (1,0,0)^\top = (0,1,0)^\top = \boldsymbol{x}$
    - $proj_{\mathcal{L}}(\boldsymbol{a}) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \boldsymbol{x}$

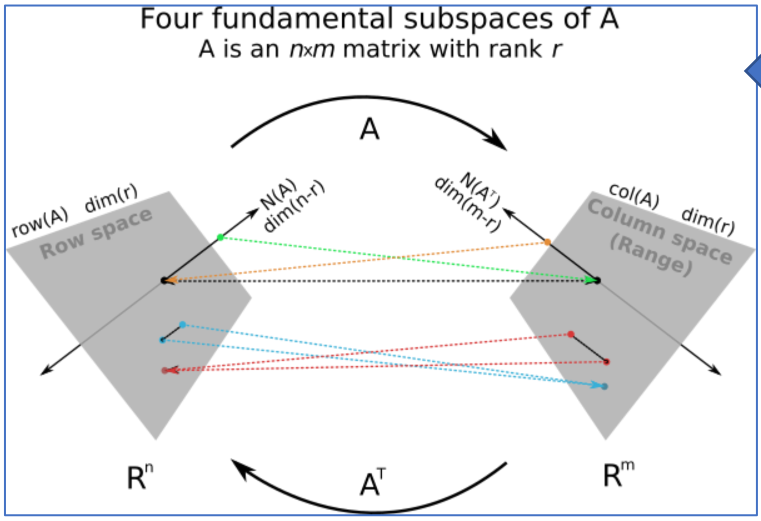


Only when L is orthonormal.

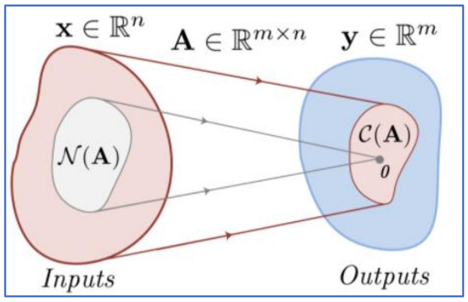
- **Four fundamental subspaces of a matrix $\boldsymbol{A}_{n\times m}$:**
  - **Null space (kernel)** $\mathcal{N}(A)$: set of all solutions to $A\boldsymbol{x} = 0$
    - Let N be an orthonormal basis that spans the null space of A
    - If x^* is a solution of $Ax = 0$, then $x^* = Nz$ where z represents the new coordinates of x^* in the null space of A.

If N is orthonormal, $N^+ = N^\top$

      - If x is in the null space spanned by N then it will have zero projection in the subspace that is orthogonal to the null space, mathematically $(I - NN^+)\boldsymbol{x} = \boldsymbol{0}$
  - **Left null space** = $\mathcal{N}(A^\top)$ = solutions to $\boldsymbol{x}^\top A = \boldsymbol{0}$
      - $proj_{\mathcal{N}(A^\top)}(v) = (I - AA^+)v$
  - **Column space (range, image)** $\mathcal{C}(A) = \mathcal{N}^\perp(A^\top)$: the span of all column vectors of A.
      - The domain of outputs for any possible inputs to the operator
          - Column space is the subspace containing all vectors $\boldsymbol{y}$ such that $A\boldsymbol{x} = \boldsymbol{y}$
          - Finding an orthonormal basis C for $C(A)$ is called matrix orthonormalization.
          - Outputs of A are then described as $Ax = Cz \ \forall z$
      - If A has a non-trivial null space, multiple different inputs will map to the same point in the column space (the zero point).
      - Projection of some vector v onto the column (or row) space of A doesn't require an orthonormal basis, only the matrix itself is needed.
          - $proj_{c(A)}(v) = AA^+v$
          - $proj_{\mathcal{R}(A)}(v) = A^+Av$
      - The column space is orthogonal to the left null space.
  - **Row space** $\mathcal{R}(A) = C(A^\top) = \mathcal{N}^\perp(A)$: the space of all inputs that produce non-zero output.
      - Row space is the span of non-zero rows of the (reduced) row echelon form
      - The row space is orthogonal to the null space.
      - If x lies in the row space spanned by $N^\perp$ then it should have a zero projection in the null space, mathematically: $NN^+x = 0$

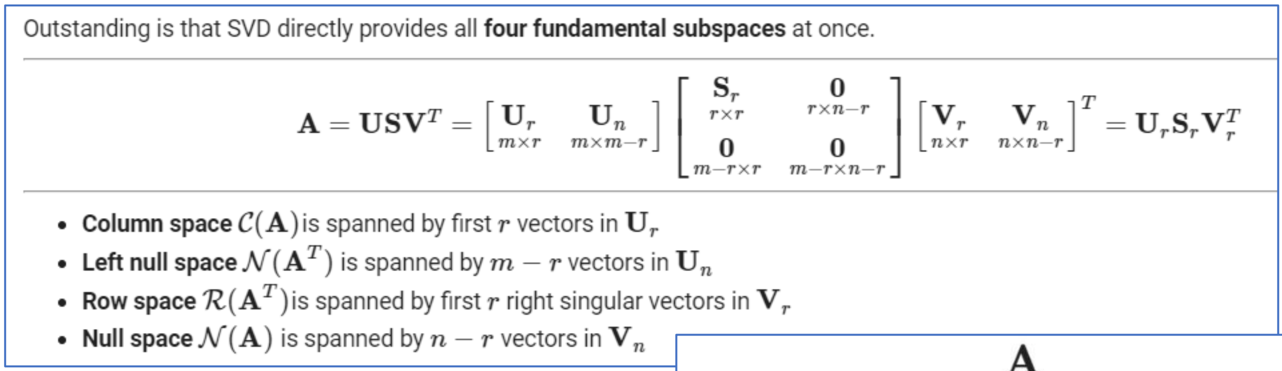


The colored dots represent vectors, and the lines show which space those vectors get mapped to through A.


$\mathbf{x} \in \mathbb{R}^n$
$\mathbf{A} \in \mathbb{R}^{m \times n}$
$\mathbf{y} \in \mathbb{R}^m$
$\mathcal{N}(\mathbf{A})$
$\mathcal{C}(\mathbf{A})$
0
Inputs
Outputs

- **All solutions to** $A\boldsymbol{x} = \boldsymbol{y}$ can be given by $x = \{x^p + x^N \mid x^N \in \mathcal{N}(A), x^p \notin \mathcal{N}(A)\}$
  - $x^p \in \mathcal{R}(A)$ is a (unique, partial) solution to the system of equations that doesn't lie in the null space of A
      - $x^p = A^+y$, where A^+ is the pseudoinverse of A
      - $x^p = (I - NN^+)x^*$, where x^* is any other solution in the null space of A.
  - Difference between any two solutions to $Ax = y$ lies in the null space of A
  - Alternative way to write the solutions is then $x = A^+y + Nz, \ \forall z$
- **Singular Value Decomposition (SVD)** $A = U\Sigma V^\top$ (exists for any A)
  - $U_{m\times m}$: unitary, orthogonal matrix of eigenvectors of $AA^\top$
  - $\Sigma_{m\times n}$: unitary diagonal rectangular matrix with non-negative elements $\sigma_i = \sqrt{\lambda_i}$
      - λ_i is the i^{th} eigenvalue of $A^\top A$ or $AA^\top$
      - $\sigma's$ are called the singular values of A
      - Number of non-zero $\sigma's$ = Rank(A)
      - $\det(A) = \prod_i \sigma_i$
  - $V_{n\times n}$: orthogonal matrix of eigenvectors of A^TA

Unitary matrix has $U^HU = UU^H = I$
H is the Hermitian (conjugate transpose)

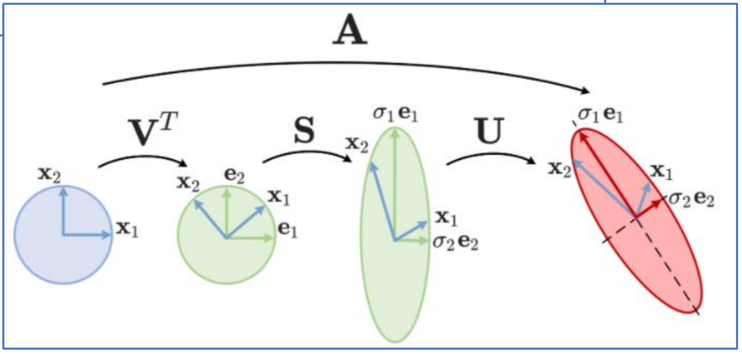


Outstanding is that SVD directly provides all **four fundamental subspaces** at once.

$$\mathbf{A} = \mathbf{USV}^T = \begin{bmatrix} \underset{m\times r}{\mathbf{U}_r} & \underset{m\times m-r}{\mathbf{U}_n} \end{bmatrix} \begin{bmatrix} \underset{r\times r}{\mathbf{S}_r} & \underset{r\times n-r}{\mathbf{0}} \\ \underset{m-r\times r}{\mathbf{0}} & \underset{m-r\times n-r}{\mathbf{0}} \end{bmatrix} \begin{bmatrix} \underset{n\times r}{\mathbf{V}_r} & \underset{n\times n-r}{\mathbf{V}_n} \end{bmatrix}^T = \mathbf{U}_r\mathbf{S}_r\mathbf{V}_r^T$$

- **Column space** $\mathcal{C}(\mathbf{A})$ is spanned by first r vectors in $\mathbf{U}_r$
- **Left null space** $\mathcal{N}(\mathbf{A}^T)$ is spanned by $m-r$ vectors in $\mathbf{U}_n$
- **Row space** $\mathcal{R}(\mathbf{A}^T)$ is spanned by first r right singular vectors in $\mathbf{V}_r$
- **Null space** $\mathcal{N}(\mathbf{A})$ is spanned by $n-r$ vectors in $\mathbf{V}_n$

- Geometrical interpretation of SVD
  - If we think of A as an operator that applies certain rotation/scaling on its operand x then SVD separates the operations to:
    - V^T rotates x to lie along the x axis
    - S scales x be a certain amount.
    - U rotates x back to its final position.



- **Condition number**: $\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$
  - Characterizes how much the output can change for a small change in input (sensitivity of the operator A)
  - $\sigma_{max}, \sigma_{min}$ are the max and min singular values of A.

Tasks

- **Finding fixed points:**
  - Find all equilibrium points of the LTI $\dot{x} = Ax + Bu$ given a constant control law.
  - LTI can be represented as an equation of block matrices $\dot{x} = [A \quad B]\begin{bmatrix} x \\ u \end{bmatrix}$
  - For an equilibrium point (that may or may not be stable) we have $\dot{x} = \mathbf{0}$, thus $\begin{bmatrix} x \\ u \end{bmatrix} = Nz;\ \forall z$
    - N is an orthonormal basis that spans the null space of $[A \quad B]$
- **Checking fixed points:**
  - Check if some constant x_d can be an equilibrium point for the system LTI $\dot{x} = Ax + Bu$, what is the feedforward that can be used to do it.
  - Start by placing poles, to find K that stabilizes the system.
    - Pole placement fails if at least one of the requested poles is repeated more than rank(B) times
    - If no such K exists, x_d cannot be an equilibrium point.
  - $u_d = -B^+(A - BK)x_d$
- **Correcting fixed points**
  - Find the closest equilibrium point x_f to some non-equilibrium point x_d under constant control law for the LTI $\dot{x} = Ax + Bu$
    - $x_f = NN^+x_d$ is the projection of x_d into the null space of $[A \quad B]$
- **Finding fixed points for affine systems**
  - Find all equilibrium points of the LTI $\dot{x} = Ax + Bu + c$ given a control law $u = -Kx + u^*$
  - LTI can be represented as an equation of block matrices $\dot{x} = [A - BK \quad B]\begin{bmatrix} x \\ u \end{bmatrix} + c$
  - For an equilibrium point (that may or may not be stable) we have $\dot{x} = \mathbf{0}$, thus
    - $\begin{bmatrix} x^* \\ u^* \end{bmatrix} = -[A - BK \quad B]^+c + Nz;\ \forall z$
    - N is the orthonormal basis that spans the null space of $[A - BK \quad B]$

Lecture 8

- **Lyapunov function** $V(\boldsymbol{x})$
  - A function on the state vector $\boldsymbol{x}$ of a dynamical system that has a **positive value** and a **negative time derivative** everywhere, except for the zero point (it has a value of 0)
  - Lyapunov functions are used to prove the stability for any dynamical system described by
    - **Continuous case:** $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$
    - **Discrete case:** $\boldsymbol{x}_{i+1} = f(\boldsymbol{x}_i)$.
  - **A valid Lyapunov function satisfy all the following conditions:**
    - $V(\boldsymbol{0}) = 0$
    - $V(\boldsymbol{x}) > 0,\ x \neq \boldsymbol{0}$
    - $\dot{V}(\boldsymbol{x}) = \nabla V \cdot \dot{\boldsymbol{x}} \leq 0,\ \boldsymbol{x} \neq 0$
      - Gradient ($\nabla = del$) of a multivariable function is the vector of all its partial derivatives.
      - For discrete case, this condition reduces to $V(x_{i+1}) - V(x_i) \leq 0$
- **Lyapunov stability criterion (direct method):**
  - The system $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ is (Lyapunov) stable if there exists a valid Lyapunov function $V(\boldsymbol{x})$
    - Asymptotic stability requires $\dot{V}(x) < 0$
    - Marginal stability requires $\dot{V}(x) \leq 0$
- **Lyapunov equation:** used to prove/disprove the stability of the systems on the form $\dot{x} = Ax$ (continuous) or $x_{i+1} = Ax_i$ (discrete).
  - For any $Q \succ 0$, there exists a unique solution $S \succcurlyeq 0$ to the Lyapunov equation if and only if the system is stable ($S \succ 0$ for asymptotic case).
  - Lyapunov function in this case can be chosen to be $V(x) = x^\top S x \geq 0$

Continuous Lyapunov equation

$$A^\top S + SA + Q = 0$$

Discrete Lyapunov equation

$$A^\top SA - S + Q = 0$$

- Jacobian linearization around a fixed point
  - The non-linear system $\dot{x} = f(x)$ has linear error dynamics near any fixed point ($f(x_e) = 0$)
  - The error dynamics are given by $\dot{e} = \mathcal{J}(x_e)e$ where:
    - $e = x_e - x$ (the deviation for equilibrium point)
    - $\mathcal{J}(x_e) = \frac{\partial f}{\partial x}|_{x_e}$
  - The stability can then be determined using the eigenvalues of $\mathcal{J}(x_e)$ since the stability of error dynamics imply stability of the original system.

Example for finding the Jacobian (2d case)

$$\dot{x} = f(x) \Leftrightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix}$$

$$\mathcal{J}(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}$$

Lecture 9

- **Cost function J: (in mathematical optimization)**
  - A function that maps an event or values of one or more variables onto a real number representing the cost of that event or value.
  - Minimizing a cost function is usually the goal in an optimization problem.
  - Applying this to the controlled system $\dot{x} = f(x,u)$, define $J\big(x_0, \pi(x,t)\big) = \int_0^\infty g(x,u)dt$ where:
    - x_0: initial state (vector) of the system
    - $\pi(x,t)$ defines the control policy (method) that we use, in other words, how the control law u is chosen based on feedforward and state feedback.
    - $g(x,u)$ describes the trajectory of the system as a function of state and control.
      - $g(u,x) = \frac{\partial J}{\partial t}$ is sometimes referred to as the cost function, however, what we need to minimize is the total cost J
  - Thus, knowing the initial state and the control law completely defines the trajectory $x(t)$ will take.
  - Minimizing J (finding J^*) is equivalent to choosing the best control policy.
    - $\dot{J}(x_0,\pi) = g(x,u) + \frac{\partial J}{\partial x} f(x,u)$
- **Hamilton-Jacobi-Bellman (HJB) equation**

$$\min_u \left[g(x,u) + \frac{\partial J^*}{\partial x} f(x,u) \right] = 0$$

  - The cost function $J(x_0,\pi)$ has a minimum value of 0 when using the best control law $u = \pi^*$, at which the derivative of J^* (in brackets) is zero.
  - Best control law can then be found by solving HJB (by finding $\arg\min \dot{J}$) equating the partial derivatives to 0 and solving for u.
  - For the LTI system $\dot{x} = Ax + Bu$
    - Solving HJB for u with a typical quadratic cost function $g(x,y) = x^\top Qx + u^\top Ru$ reduces to the problem of solving the Algebraic Riccati Equation (ARE) $A^\top P + PA - PBR^{-1}B^\top P + Q = 0$, the feedback controller that uses the solution for P in the control law $u = -R^{-1}B^\top Px$ is called the Linear Quadratic Regulator (LQR)
    - **(Extra) Solution to the ARE:** $A^\top P + PA - PBR^{-1}B^\top P + Q = 0$
      - $P = YX^{-1}$ where $\begin{bmatrix} X \\ Y \end{bmatrix} = [v_1 \quad \cdots \quad v_n]$ and v_i are the eigenvectors of $Z = \begin{pmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{pmatrix}$

- **Linear Quadratic Regulator (Recall)**
  - Finds the optimal gain matrix, by introducing a cost function, that adds up the weighted sum of performance Q and effort R, then solving it for the gain matrix $K = R^{-1}B^\top P$
  - Choosing the appropriate weights (matrices Q, R) is more intuitive that choosing poles.

Lecture 10

- **State observer**:
  - For a control law that feeds back the state such as $u = -Kx$ we cannot practically find the exact current state of the system, instead, we use state observer **measurements** that are not necessarily precise due to
    - Discrete time measurements.
    - Unpredicted events, unmodeled kinematics/dynamics.
    - Lack/bias of sensors.
  - Measured state $\hat{x}$ cannot usually be found directly, instead we need to have an observed output $y = Cx$
  - Dynamics for x should also hold for $\hat{x}$
  - **Open loop observer** builds the estimated states from the initial state (by solving the system directly) without considering the observed output, this may lead to divergence of the estimated state from the actual state.
  - **Closed loop observer** considers the output $y = Cx$ by introducing a correction term $L(y - C\hat{x})$ that compensate for the observation error.
- **Linear observer model (Luenberger observer)**
  - Assume the measurement error in the observed output is $y - \hat{y} = y - C\hat{x}$, then we can estimate the behavior of the system by the following equation.

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x})$$

  - Let the measurement error in x be $\varepsilon = \hat{x} - x$, we can find the error dynamics in the same way we did in Lecture 4.

$$\dot{\varepsilon} = (A - LC)\varepsilon$$

  - To have a stable observer, we choose L such that $A - LC \prec 0$ (i.e., has negative eigenvalues), this can be done through Pole placement or solving ARE for LQR as we did before.
- **Separation principle (of estimation and control):**
  - If the **observer** and the **controller** are independently stable, the overall system is stable too.
  - This can be proven by deriving the equations for the controller and the state measurement (estimation) error $e = x - \hat{x}$ as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix} \begin{bmatrix} x \\ e \end{bmatrix}$$

  - Notice that this state space matrix has eigenvalues $eig(A - BK) \cup eig(A - LC)$, all of which need to be negative for the system to be (asymptotically) stable.

Separation principle does not hold in general for non-linear systems.

Lecture 11

- **Controllability:** the ability to move the system around its entire configuration space (reach all possible states) using only certain manipulations (control laws)
- **Observability:** the ability to precisely determine internal system states given only its output.
- **Kalman tests** (note lecture notation: $\mathcal{B}$ for controllability matrix, $\mathcal{C}$ for observability matrix)

Controllability matrix for the LTI $\dot{x} = Ax + Bu$ $\mathcal{C} = [B \quad AB \quad \dots \quad A^{n-1}B]$	**Observability matrix for the system** $\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$ is $\mathcal{O} = \begin{bmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{bmatrix}$
Any state can be reached, and the system is said to be **controllable** if $rank(\mathcal{C}) = n$ (full row rank)	The observation error goes to 0 and the system is said to be observable if $rank(\mathcal{O}) = p$ (full column rank)

- Popov-Belevitch-Hautus (PBH) **test for controllability of the LTI $\dot{x} = Ax + Bu$**
  - Provides more information about the system such as the degree of controllability, input channels, unlike the Kalman test which only gives a yes/no answer.
  - The system is controllable if and only if:
    - $rank([A - \lambda_i I \quad B]) = n, \ \forall \lambda_i \in eigs(A)$
  - That is:
    - $A - \lambda_i I$ has independent rows $\forall \lambda_i \in eigs(A)$
    - $B \notin \mathcal{C}(A - \lambda_i I)$, equivalently, columns of B should be independent from each other and eigenvectors of A.
  - Input channels = number of actuators needed to control the system = number of repeated eigenvalues of A.
  - The same test can be used to determine stabilizability
    - $rank([A - \lambda_i I \quad B]) = n, \ \forall \lambda_i \in eigs(A), Re(\lambda_i) \geq 0$

A matrix is said to be rank deficient if it does not have a full rank, rank_deficiency$(A_{n\times m}) = rank(A) - min\,(n, m)$

A system is **stabilizable** if all states that cannot be reached decay to zero asymptotically.

- **For discrete systems:** $x_{i+1} = Ax_i + Bu_i$, we can prove $x_{n+1} = A^n x_1 + \sum_{k=0}^{n-1} A^k B u_{n-k}$ by construction, this can also be written in matrix form as:

$$x_{n+1} - A^n x_1 = [B \quad AB \quad \dots \quad A^{n-1}B] \begin{bmatrix} u_k \\ u_{k-1} \\ \dots \\ u_1 \end{bmatrix}$$

  - This can be interpreted as: if $x_{n+1} - A^n x_1$ lies in the column space of $\mathcal{B}$, then any state can be reached from any other state in n steps at maximum, giving the impression of "unlimited control".
  - Another example is, for any final state x_f and initial state x_1, we have x_f can be reached from x_1 in only one step if $x_f - Ax_1 = Bu$ (i.e., $x_f - Ax_1$ lies in the column space of B), since we can solve this equation analytically to find $u = B^+(x_f - Ax_1)$
  - This unlimited control is however not practically possible, as in real-life systems, not every input u can be achieved, usually the choice of u is restricted by equations such as $Du_i \leq d$, which are not analytically solvable.

Lecture 12

- **Newton's equations** $m_i \ddot{r}_i = f_i,\ i \in \{1, \ldots, m\}$
  - m_i – mass of particle i
  - r_i – position of particle i
  - f_i – force acting on particle i
- **Lagrange equations** $\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}}\right) - \frac{\partial T}{\partial q} = \tau$
  - T – kinetic energy, $T = 0.5\,\dot{r}^\top M \dot{r}$
  - q – generalized coordinates
  - τ – generalized forces
- **Linearization of a non-linear system** $\dot{x} = f(x, u)$
  - Using Taylor's expansion

$$f(x,u) \sim f(x_0,u_0) + \frac{\partial f}{\partial x}(x - x_0) + \frac{\partial f}{\partial u}(u - u_0)$$

  - Let $A = \frac{\partial f}{\partial x}, B = \frac{\partial f}{\partial u}, c = f(x_0,u_0) - Ax_0 - Bu_0$ we get the linearization $\dot{x} \sim Ax + Bu + c$
- **Manipulator equations:** $H\ddot{q} + C\dot{q} + g = \tau_n$
  - $H(q)$ – generalized inertia matrix
  - $C(q, \dot{q})$ – inertial force matrix
  - $g(q)$ – generalized gravity (and other conservative forces)
  - τ_n – generalized non-conservative forces (e.g., control inputs)
  - Let $\tau_n = Tu$, $x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$, we get $\ddot{q} = H^{-1}(Tu - C\dot{q} - g)$

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ \frac{\partial}{\partial q}\left(H^{-1}(Tu - C\dot{q} - g)\right) & -H^{-1}\left(C + \frac{\partial C}{\partial \dot{q}}\dot{q}\right) \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ H^{-1}T & 0 \end{bmatrix} u$$

Conservative force: work done is <u>independent</u> from path taken (e.g., gravity), depends only on the current position of the object.

Non-conservative: <u>dependent</u> (e.g., friction)

Generalized coordinates describe a state of the system (e.g., position) relative to some reference configuration (e.g., the angle that locates a point moving on a circle).

Lecture 13

- When modeling a dynamical system, we typically encounter **explicit constraints**:
  - Systems with contact interactions
  - Hybrid systems: two or more different dynamics which switch between one-another.
  - Nonholonomic constraints in the dynamics (dynamics of a unicycle, bicycle, etc.)
- To control such systems, we can
  - Reduce the system to a one with implicit constraints and control it.
    - A system with explicit constraints can be transformed to a system with implicit constraints by substituting the constraints into the equations and simplifying/reducing to get one equation.
  - Treat reaction forces as external forces.
  - Design control law based on the explicit representation of constraints.
- **Explicit constraints**
  - A **linear** dynamical system with explicit constraints can have the form:

$$\begin{cases} \dot{x} = Ax + Bu + F\lambda \\ G\dot{x} = 0 \end{cases}$$

    - G – constraint matrix
    - $\lambda \in \mathbb{R}^k$ – constraint reaction forces
    - F – reaction force Jacobian
  - An **affine** dynamical system with explicit constraints can have the form:

$$\begin{cases} \dot{x} = Ax + Bu + S\lambda + c \\ G\dot{x} = 0 \end{cases}, \qquad G = \begin{bmatrix} F & 0 \\ \dot{F} & F \end{bmatrix}$$

    - $\lambda \in \mathbb{R}^k$ constraint reaction forces
    - S – linearized constraint Jacobian
    - F – reaction force Jacobian
- **Minimal representation (realization)**
  - Dynamics are clearer to understand when using explicit constraints, however, minimal representation simplifies calculations by describing the same system with the smallest number of states.
  - A linear/affine system with explicit constraint $G\dot{x} = 0$ means that x lies in the null space of G, in other words, $x = Nz$ with N being an orthonormal basis in the null space of G
  - Substituting in original system we get ($c_N = 0$ in linear case)

$$\dot{z} = A_N z + B_N u + c_N, \qquad A_N = N^\top A N, \qquad B_N = N^\top B, \qquad c_N = N^\top c$$

  - We can stabilize the system as usual using LQR: $K_N = lqr(A_N, B_N, Q, R)$
- **Inverse kinematics (dynamics):**
  - Consider a robotic arm consisting of multiple connected joints, with an end-effector (e.g., hand), two mathematical processes exist related to control of such arm:
    - **Forward kinematics:** calculating the final position of the end-effector given the parameters (e.g., angles) of each joint.
    - **Inverse kinematics:** calculating the parameters of each joint, given the desired final position of the end-effector (e.g., driving the hand to a specific position).
      - An analytical solution to the inverse kinematics of any (constrained) LTI is possible if $\dot{x} - Ax - c$ lies in the column space of B, in other words $(I - BB^+)(\dot{x} - Ax - c) = 0$
      - The solution is then $u_{IK} = B^+(\dot{x} - Ax - c)$
  - Inverse kinematics for manipulator equations $H\ddot{q} + C\dot{q} + g = Tu + F^\top\lambda$

$$\begin{cases} u = (S_2 Q^\top T)^+ S_2 Q^\top (H\ddot{q} + Q^\top C\dot{q} + g) \\ \lambda = R^{-1} S_1 Q^T (H\ddot{q} + Q^\top C\dot{q} + g - Tu) \end{cases}$$

  - Where:
    - $F^\top = Q\begin{bmatrix} R \\ 0 \end{bmatrix}$ (the QR decomposition of $F^\top$ - check AGLAII notes)
    - $S_1 = [I \quad 0]$, $S_2 = [0 \quad I]$ (the switching variables)
  - We can then solve for the minimum u, λ that minimize $||u||$ given the system and any explicit constraints using quadratic programming (quadratic optimization problem)