MURAT ARCAK

EE16B NOTES SPRING'17

Control Systems

State Space Representation of Dynamical Systems

State variables are a set of variables that fully represent the state of a dynamical system at a given time, *e.g.*, capacitor voltages and inductor currents in a circuit, or positions and velocities in a mechanical system. We denote by $\vec{x}(t)$ the vector of these state variables and refer to it as the *state vector*.

Continuous-Time Systems

In a continuous-time system with n state variables, $\vec{x}(t) \in \mathbb{R}^n$ evolves according to a differential equation of the form

$$\frac{d}{dt}\vec{x}(t) = f(\vec{x}(t)) \tag{1}$$

where $f(\vec{x}(t))$ is an n-vector that dictates the derivatives of each state variable according to the current value of the states. The form of f depends on the system we are modeling as we will see in examples.

If the system has input variables we can manipulate (*e.g.* voltage and current sources in a circuit, or force and torque delivered to a mechanical system), we represent the system as

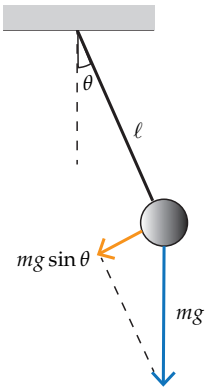
$$\frac{d}{dt}\vec{x}(t) = f(\vec{x}(t), \vec{u}(t))$$

where we refer to $\vec{u}(t)$ as the *control input*, since we can manipulate this input to influence the behavior of the system. Most of our examples will contain a single control input, but we write $\vec{u}(t)$ as a vector to allow for multiple control inputs.

Finally, we denote by $\vec{w}(t)$ other inputs that are not under our control, *e.g.* wind force in a flight control system, and add it to our model:

$$\frac{d}{dt}\vec{x}(t) = f(\vec{x}(t), \vec{u}(t), \vec{w}(t)).$$

The inputs contained in $\vec{w}(t)$ are often called *disturbances*.



Example 1: The motion of the pendulum depicted on the right is governed by the differential equation

$$m\ell\frac{d^2\theta(t)}{dt^2} = -k\ell\frac{d\theta(t)}{dt} - mg\sin\theta(t) \tag{2}$$

where the left hand side is mass×accelaration in the tangential direction and the right hand side is total force acting in that direction.

To bring this second order differential equation to state space form we define the state variables

$$x_1(t) \triangleq \theta(t) \qquad x_2(t) \triangleq \frac{d\theta(t)}{dt}$$

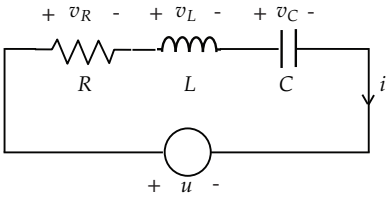
and note that they satisfy

$$\begin{aligned}\frac{dx_1(t)}{dt} &= x_2(t)\\ \frac{dx_2(t)}{dt} &= -\frac{k}{m}x_2(t) - \frac{g}{\ell}\sin x_1(t).\end{aligned} \tag{3}$$

The first equation here follows from the definition of $x_2(t)$, and the second equation follows from (2). In this state representation we have two first order differential equations, one for each state variable, instead of the second order differential equation (2) for one variable.

Here we did not consider disturbances or control inputs that could be applied (say, to balance the pendulum in the upright position) so the equations (3) have the form (1) with

$$f(\vec{x}(t)) = \begin{bmatrix} x_2(t) \\ -\frac{k}{m}x_2(t) - \frac{g}{\ell}\sin x_1(t) \end{bmatrix}.$$



Example 2: Consider the RLC circuit depicted on the right where u denotes the input voltage.

Since the capacitor and inductor satisfy the relations

$$\begin{aligned} C\frac{dv_c(t)}{dt} &= i(t)\\ L\frac{di(t)}{dt} &= v_L(t), \end{aligned} \tag{4}$$

we select $x_1 = v_C$ and $x_2 = i$ as the state variables, and eliminate $v_L(t)$ from the right hand side of (4) using KVL and Ohm's Law:

$$v_L = -v_C - v_R + u = -v_C - Ri + u.$$

Then the state model becomes

$$\begin{aligned}\frac{dx_1(t)}{dt} &= \frac{1}{C}x_2(t)\\ \frac{dx_2(t)}{dt} &= \frac{1}{L}\left(-x_1(t) - Rx_2(t) + u(t)\right).\end{aligned} \tag{5}$$

Discrete-Time Systems

In a *discrete*-time system, $\vec{x}(t)$ evolves according to a *difference* equation rather than a differential equation:

$$\vec{x}(t+1) = f(\vec{x}(t), \vec{u}(t), \vec{w}(t)) \quad t = 0, 1, 2, \ldots$$

Here $f(\vec{x}(t), \vec{u}(t), \vec{w}(t))$ is a vector that dictates the value of the state vector at the next time instant based on the present values of the states and inputs.

Example 3: Let $s(t)$ denote the inventory of a manufacturer at the start of the t-th business day. The inventory at the start of the next day, $s(t+1)$, is the sum of $s(t)$ and the goods $g(t)$ manufactured, minus the goods $w(t)$ sold on day t. Assuming it takes a day to do the manufacturing, the amount of goods $g(t)$ manufactured is equal to the raw material available the previous day, $r(t-1)$. The raw material $r(t)$ is equal to the order placed the previous day, $u(t-1)$, assuming it takes a day for the order to arrive.

The state variables $s(t)$, $g(t)$, $r(t)$, thus evolve according to the model

$$\begin{aligned} s(t+1) &= s(t) + g(t) - w(t) \\ g(t+1) &= r(t) \\ r(t+1) &= u(t). \end{aligned} \tag{6}$$

Note that the order $u(t)$ is an input that the manufacturer can control, but the amount of goods sold, $w(t)$, depends on the customers.

Example 4: Let $p(t)$ be the number of EECS professors in a country in year t, and let $r(t)$ be the number of industry researchers with a PhD degree. A fraction, γ, of the PhDs become professors themselves and the rest become industry researchers. A fraction, δ, in each profession leaves the field every year due to retirement or other reasons.

Each professor graduates, on average, $u(t)$ PhD students per year. We treat this number as a control input because it can be manipulated by the government using research funding. This means there will be $p(t)u(t)$ new PhDs in year t, and $\gamma p(t)u(t)$ new professors. The state model is then

$$\begin{aligned} p(t+1) &= (1-\delta)p(t) + \gamma p(t)u(t) \\ r(t+1) &= (1-\delta)r(t) + (1-\gamma)p(t)u(t). \end{aligned} \tag{7}$$

Linear Systems

When $f(\vec{x}, \vec{u}, \vec{w}) \in \mathbb{R}^n$ is linear in $\vec{x} \in \mathbb{R}^n$, $\vec{u} \in \mathbb{R}^m$, $\vec{w} \in \mathbb{R}^d$, we can rewrite it in the form

$$f(\vec{x}, \vec{u}, \vec{w}) = A\vec{x} + B_u\vec{u} + B_w\vec{w}$$

where A is a $n \times n$ matrix, B_u is $n \times m$, and B_w is $n \times d$. The state equations then take the form

$$\begin{aligned} \frac{d}{dt}\vec{x}(t) &= A\vec{x}(t) + B_u\vec{u}(t) + B_w\vec{w}(t) \\ \vec{x}(t+1) &= A\vec{x}(t) + B_u\vec{u}(t) + B_w\vec{w}(t) \end{aligned}$$

for a continuous- and discrete-time system, respectively. When we don't need to differentiate between control and disturbance inputs, we drop the subscripts u and w from B.

Note that the models in Examples 2 and 3 above are linear[1]. In particular we can write (6) in the matrix form:

[1] Why are Examples 1 and 4 nonlinear?

$$\underbrace{\begin{bmatrix} s(t+1) \\ g(t+1) \\ r(t+1) \end{bmatrix}}_{\vec{x}(t+1)} = \underbrace{\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} s(t) \\ g(t) \\ r(t) \end{bmatrix}}_{\vec{x}(t)} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{B_u} u(t) + \underbrace{\begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}}_{B_w} w(t).$$

Likewise, we rewrite (5) as:

$$\underbrace{\begin{bmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \end{bmatrix}}_{\frac{d}{dt}\vec{x}(t)} = \underbrace{\begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}}_{\vec{x}(t)} + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix}}_{B} u(t).$$

Changing State Variables

It is important to note that the choice of state variables is not unique. Given the state vector $\vec{x} \in \mathbb{R}^n$ any transformation of the form

$$\vec{z} \triangleq T\vec{x}, \tag{8}$$

where T is a $n \times n$ invertible matrix, defines new variables z_i, $i = 1, \ldots, n$, as a linear combination of the original variables $x_1, \ldots, x_n$.

To see how this change of variables affects the state equation

$$\vec{x}(t+1) = A\vec{x}(t) + B\vec{u}(t),$$

note that

$$\vec{z}(t+1) = T\vec{x}(t+1) = TA\vec{x}(t) + TB\vec{u}(t)$$

and substitute $\vec{x} = T^{-1}\vec{z}$ in the right hand side to obtain:

$$\vec{z}(t+1) = TAT^{-1}\vec{z}(t) + TB\vec{u}(t).$$

Thus the original A and B matrices are replaced with:

$$A_{\text{new}} = TAT^{-1}, \qquad B_{\text{new}} = TB.$$

The same change of variables brings the *continuous-time* system

$$\frac{d}{dt}\vec{x}(t) = A\vec{x}(t) + B\vec{u}(t)$$

to the form

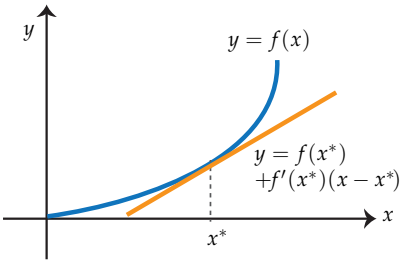
$$\frac{d}{dt}\vec{z}(t) = A_{\text{new}}\vec{z}(t) + B_{\text{new}}\vec{u}(t)$$

where A_{new} and B_{new} are as defined above.

In the suquel we will use particular choices of T to bring A_{new} and B_{new} to special forms that will make it easy to analyze properties such as *stability* and *controllability*.

Linearization

Linear models are advantageous because their solutions, stability properties, and stabilizing controllers can be studied using linear algebra. The methods applicable to nonlinear models are limited; therefore it is common practice to approximate a nonlinear model with a linear one that is valid around a desired operating point.



Recall that the Taylor approximation of a differentiable function f around a point x^* is:

$$f(x) \approx f(x^*) + \nabla f(x)|_{x=x^*}\,(x - x^*),$$

as illustrated on the right for a scalar-valued function of a single variable. When x and $f(x)$ are n-vectors as in our state models, $\nabla f(x)$ must be interpreted as the $n \times n$ matrix of partial derivatives:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f_1(x_1,\ldots,x_n)}{\partial x_1} & \frac{\partial f_1(x_1,\ldots,x_n)}{\partial x_2} & \cdots & \frac{\partial f_1(x_1,\ldots,x_n)}{\partial x_n} \\ \frac{\partial f_2(x_1,\ldots,x_n)}{\partial x_1} & \frac{\partial f_2(x_1,\ldots,x_n)}{\partial x_2} & \cdots & \frac{\partial f_2(x_1,\ldots,x_n)}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n(x_1,\ldots,x_n)}{\partial x_1} & \frac{\partial f_n(x_1,\ldots,x_n)}{\partial x_2} & \cdots & \frac{\partial f_n(x_1,\ldots,x_n)}{\partial x_n} \end{bmatrix}.$$

We linearize nonlinear state models by applying this approximation around an *equilibrium* point. For the continuous-time system

$$\frac{d}{dt}\vec{x}(t) = f(\vec{x}(t)), \tag{9}$$

$\vec{x}^*$ is called an *equilbrium* when $f(\vec{x}^*) = 0$ because, if the initial condition is $\vec{x}^*$, then $\frac{d}{dt}\vec{x}(t) = 0$ and $\vec{x}(t)$ remains at $\vec{x}^*$. If we define the deviation of $\vec{x}$ from $\vec{x}^*$ as:

$$\boxed{\tilde{x}(t) \triangleq \vec{x}(t) - \vec{x}^*} \tag{10}$$

then we see that

$$\frac{d}{dt}\tilde{x}(t) = f(\vec{x}(t)) \approx f(\vec{x}^*) + \nabla f(\vec{x})|_{\vec{x}=\vec{x}^*}\,\tilde{x}(t).$$

Substituting $f(\vec{x}^*) = 0$ and defining

$$\boxed{A \triangleq \nabla f(\vec{x})|_{\vec{x}=\vec{x}^*}} \tag{11}$$

we obtain the linearization of (9) around the equilibrium $\vec{x}^*$:

$$\boxed{\frac{d}{dt}\tilde{x}(t) \approx A\tilde{x}(t).}$$

In the discrete-time case

$$\vec{x}(t+1) = f(\vec{x}(t)),$$

$\vec{x}^*$ is an equilbrium point if $f(\vec{x}^*) = \vec{x}^*$. The vector $\tilde{x}(t)$ defined in (10) satisfies:

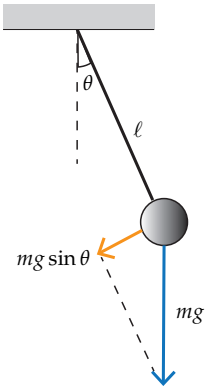
$$\tilde{x}(t+1) = \vec{x}(t+1) - \vec{x}^* = f(\vec{x}(t)) - \vec{x}^* \approx f(\vec{x}^*) - \vec{x}^* + \nabla f(\vec{x})|_{\vec{x}=\vec{x}^*}\, \tilde{x}(t).$$

Substituting $f(\vec{x}^*) - \vec{x}^* = 0$ and defining A as in (11), we get

$$\boxed{\tilde{x}(t+1) \approx A\tilde{x}(t).}$$

Example 5: Recall the pendulum model derived in Example 1:

$$\begin{aligned} \frac{dx_1(t)}{dt} &= x_2(t) \\ \frac{dx_2(t)}{dt} &= -\frac{k}{m}x_2(t) - \frac{g}{\ell}\sin x_1(t) \end{aligned} \tag{12}$$



where

$$x_1(t) \triangleq \theta(t) \quad \text{and} \quad x_2(t) \triangleq \frac{d\theta(t)}{dt}.$$

To find the equilibrium points note that

$$f(\vec{x}) = \begin{bmatrix} x_2 \\ -\frac{k}{m}x_2 - \frac{g}{\ell}\sin x_1 \end{bmatrix} = 0$$

when $x_2 = 0$ and $\sin x_1 = 0$. Thus the two distinct equilibrium points are the downward position:

$$x_1 = 0, \quad x_2 = 0, \tag{13}$$

and the upright position:

$$x_1 = \pi, \quad x_2 = 0. \tag{14}$$

Since the entries of $f(\vec{x})$ are $f_1(\vec{x}) = x_2$ and $f_2(\vec{x}) = -\frac{k}{m}x_2 - \frac{g}{\ell}\sin x_1$, we have

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f_1(x_1,x_2)}{\partial x_1} & \frac{\partial f_1(x_1,x_2)}{\partial x_2} \\ \frac{\partial f_2(x_1,x_2)}{\partial x_1} & \frac{\partial f_2(x_1,x_2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{\ell}\cos x_1 & \frac{-k}{m} \end{bmatrix}.$$

By evaluating this matrix at (13) and (14), we obtain the linearization around the respective equilibrium point:

$$A_{\text{down}} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{\ell} & \frac{-k}{m} \end{bmatrix} \qquad A_{\text{up}} = \begin{bmatrix} 0 & 1 \\ \frac{g}{\ell} & \frac{-k}{m} \end{bmatrix}. \tag{15}$$

Stability of Linear State Models

The Scalar Case

We first study a system with a single state variable $x(t)$ that obeys

$$x(t+1) = ax(t) + bu(t) \tag{16}$$

where a and b are constants. If we start with the initial condition $x(0)$, then we get by recursion

$$\begin{aligned}
x(1) &= ax(0) + bu(0) \\
x(2) &= ax(1) + bu(1) = a^2x(0) + abu(0) + bu(1) \\
x(3) &= ax(2) + bu(2) = a^3x(0) + a^2bu(0) + abu(1) + bu(2) \\
&\vdots \\
x(t) &= a^tx(0) + a^{t-1}bu(0) + a^{t-2}bu(1) + \cdots + abu(t-2) + bu(t-1),
\end{aligned}$$

rewritten compactly as:

$$\boxed{x(t) = a^tx(0) + \sum_{k=0}^{t-1} a^{t-1-k}bu(k) \quad t = 1, 2, 3, \ldots} \tag{17}$$

The first term $a^tx(0)$ represents the effect of the initial condition and the second term $\sum_{k=0}^{t-1} a^{t-1-k}bu(k)$ represents the effect of the input sequence $u(0), u(1), \ldots, u(t-1)$.

Definition. We say that a system is *stable* if its state $x(t)$ remains bounded for any initial condition and any bounded input sequence. Conversely, we say it is *unstable* if we can find an initial condition and a bounded input sequence such that $|x(t)| \to \infty$ as $t \to \infty$.

It follows from (17) that, if $|a| > 1$, then a nonzero initial condition $x(0) \neq 0$ is enough to drive $|x(t)|$ unbounded. This is because $|a|^t$ grows unbounded and, with $u(t) = 0$ for all t, we get $|x(t)| = |a^tx(0)| = |a|^t|x(0)| \to \infty$. Thus, (16) is unstable for $|a| > 1$.

Next, we show that (16) is stable when $|a| < 1$ is stable. In this case $a^tx(0)$ decays to zero, so we need only to show that the second term in (17) remains bounded for any bounded input sequence. A bounded input means we can find a constant M such that $|u(t)| \leq M$ for all t. Thus,

$$\left|\sum_{k=0}^{t-1} a^{t-1-k}bu(k)\right| \leq \sum_{k=0}^{t-1} |a|^{t-1-k}|b||u(k)| \leq |b|M\sum_{k=0}^{t-1} |a|^{t-1-k}.$$

Defining the new index $s = t-1-k$ we rewrite the last expression as

$$|b|M\sum_{s=0}^{t-1} |a|^s,$$

and note that $\sum_{s=0}^{t-1} |a|^s$ is a geometric series that converges to $\frac{1}{1-|a|}$ since $|a| < 1$. Therefore, each term in (17) is bounded and we conclude stability for $|a| < 1$.

Summary: The scalar system (16) is stable when $|a| < 1$, and unstable when $|a| > 1$.

When a is a complex number, a perusal of the stability and instability arguments above show that the same conclusions hold if we interpret $|a|$ as the modulus of a, that is:

$$|a| = \sqrt{\mathrm{Re}\{a\}^2 + \mathrm{Im}\{a\}^2}.$$

What happens when $|a| = 1$? If we disallow inputs ($b = 0$), this case is referred to as "marginal stability" because $|a^t x(0)| = |x(0)|$, which neither grows nor decays. If we allow inputs ($b \neq 0$), however, we can find a bounded input to drive the second term in (17) unbounded. For example, when $a = 1$, the constant input $u(t) = 1$ yields:

$$\sum_{k=0}^{t-1} a^{t-1-k} b u(k) = \sum_{k=0}^{t-1} b = bt$$

which grows unbounded as $t \to \infty$. Therefore, $|a| = 1$ is a precarious case that must be avoided in designing systems.

The Vector Case

When $\vec{x}(t)$ is an n-dimensional vector governed by

$$\vec{x}(t+1) = A\vec{x}(t) + Bu(t), \tag{18}$$

recursive calculations lead to the solution

$$\vec{x}(t) = A^t \vec{x}(0) + \sum_{k=0}^{t-1} A^{t-1-k} B u(k) \quad t = 1, 2, 3, \ldots \tag{19}$$

where the matrix power is defined as $A^t = \underbrace{A \cdots A}_{t \text{ times}}$.

Since A is no longer a scalar, stability properties are not apparent from (19). However, when A is diagonalizable we can employ the change of variables $\vec{z} \triangleq T\vec{x}$ and select the matrix T such that

$$A_{\text{new}} = TAT^{-1}$$

is diagonal. A and A_{new} have the same eigenvalues and, since A_{new} is diagonal, the eigenvalues appear as its diagonal entries:

$$A_{\text{new}} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}.$$

The state model for the new variables is

$$\vec{z}(t+1) = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \vec{z}(t) + B_{\text{new}} u(t) \tag{20}$$

which nicely decouples into scalar equations:

$$z_i(t+1) = \lambda_i z_i(t) + b_i u(t), \quad i = 1, \ldots, n \tag{21}$$

where we denote by b_i the i-th entry of B_{new}. Then, the results for the scalar case above imply stability when $|\lambda_i| < 1$ and instability when $|\lambda_i| > 1$.

For the whole system to be stable each subsystem must be stable, therefore we need $|\lambda_i| < 1$ for each $i = 1, \ldots, n$. If there exists at least one eigenvalue λ_i with $|\lambda_i| > 1$ then we conclude instability because we can drive the corresponding state $z_i(t)$ unbounded.

Summary: The discrete-time system (18) is stable if $|\lambda_i| < 1$ for each eigenvalue $\lambda_1, \ldots, \lambda_n$ of A, and unstable if $|\lambda_i| > 1$ for some eigenvalue λ_i.

Although we assumed diagonalizability of A above, the same stability and instability conditions hold when A is not diagonalizable. In that case a transformation exists that brings A_{new} to an upper-diagonal form with eigenvalues on the diagonal[2]. Thus, instead of (20) we have

[2] The details of this transformation are beyond the scope of this course.

$$\vec{z}(t+1) = \begin{bmatrix} \lambda_1 & \star & \cdots & \star \\ & \ddots & \ddots & \vdots \\ & & \ddots & \star \\ & & & \lambda_n \end{bmatrix} \vec{z}(t) + B_{\text{new}} u(t) \tag{22}$$

where the entries marked with '$\star$' may be nonzero, but we don't need their explicit values for the argument that follows. Then it is not difficult to see that z_n obeys

$$z_n(t+1) = \lambda_n z_n(t) + b_n u(t) \tag{23}$$

which does not depend on other states, so we conclude $z_n(t)$ remains bounded for bounded inputs when $|\lambda_n| < 1$. The equation for z_{n-1} has the form

$$z_{n-1}(t+1) = \lambda_{n-1} z_{n-1}(t) + [\star\, z_n(t) + b_{n-1} u(t)] \tag{24}$$

where we can treat the last two terms in brackets as a bounded input since we have already shown that $z_n(t)$ is bounded. If $|\lambda_{n-1}| < 1$ we conclude $z_{n-1}(t)$ is itself bounded and proceed to the equation:

$$z_{n-2}(t+1) = \lambda_{n-2} z_{n-2}(t) + [\star\, z_{n-1}(t) + \star\, z_n(t) + b_{n-2} u(t)]. \tag{25}$$

Continuing this argument recursively we conclude stability when $|\lambda_i| < 1$ for each eigenvalue λ_i.

To conclude instability when $|\lambda_i| > 1$ for some eigenvalue, note that the ordering of the eigenvalues in (22) is arbitrary: we can put them in any order we want by properly selecting T. Therefore, we can assume without loss of generality that an eigenvalue with $|\lambda_i| > 1$ appears in the nth diagonal entry, that is $|\lambda_n| > 1$. Then, instability follows from the scalar equation (23).

Stability of Continuous-Time Linear Systems

The solution of the scalar continuous-time system

$$\frac{d}{dt}x(t) = ax(t) + bu(t) \tag{26}$$

is given by

$$x(t) = e^{at}x(0) + b\int_0^t e^{a(t-s)}u(s)ds. \tag{27}$$

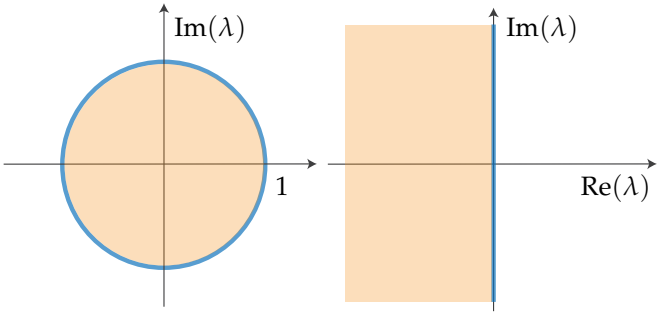
It follows that this system is stable when $a < 0$ (in which case $e^{at} \to 0$ as $t \to \infty$) and unstable when $a > 0$ (in which case $e^{at} \to \infty$).

Using a diagonalization argument as in the discrete-time case, we conclude that the vector continuous-time system

$$\frac{d}{dt}\vec{x}(t) = A\vec{x}(t) + B\vec{u}(t) \tag{28}$$

is stable if $\text{Re}\{\lambda_i\} < 0$ for each eigenvalue $\lambda_1, \ldots, \lambda_n$ of A, and unstable if $\text{Re}\{\lambda_i\} > 0$ for some eigenvalue λ_i.

The figures below highlight the regions of the complex plane where the eigenvalues must lie for stability of a discrete-time (left) and continuous-time (right) system.



Example 6: In Example 5 we derived continuous-time linearized models for the downward and upright positions of the pendulum,

and obtained:

$$A_{\text{down}} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{\ell} & \frac{-k}{m} \end{bmatrix} \qquad A_{\text{up}} = \begin{bmatrix} 0 & 1 \\ \frac{g}{\ell} & \frac{-k}{m} \end{bmatrix}. \tag{29}$$

The eigenvalues of A_{down} are the roots of $\lambda^2 + \frac{k}{m}\lambda + \frac{g}{\ell}$, which can be shown to have strictly negative real parts when $k > 0$. Thus the downward position is stable.

The eigenvalues of A_{up} are the roots of $\lambda^2 + \frac{k}{m}\lambda - \frac{g}{\ell}$, which are given by:

$$\lambda_1 = -\frac{k}{2m} - \frac{1}{2}\sqrt{\left(\frac{k}{m}\right)^2 + 4\frac{g}{\ell}}, \quad \lambda_2 = -\frac{k}{2m} + \frac{1}{2}\sqrt{\left(\frac{k}{m}\right)^2 + 4\frac{g}{\ell}}.$$

Since $\lambda_2 > 0$, the upright position in unstable. Note that making the length ℓ smaller increases the value of λ_2. This suggests that a smaller length aggravates the instability of the upright position and makes the stabilization task more difficult, as you would experience when you try to balance a stick in your hand.

Predicting System Behavior from Eigenvalue Locations

We have seen that the solutions of a discrete-time system are composed of λ_i^t terms where λ_i's are the eigenvalues of A. Thus, to predict the nature of the solutions (damped, underdamped, unbounded, etc.), it is important to visualize the sequence λ^t, $t = 1, 2, \ldots$ for a given λ. If we rewrite λ as $\lambda = |\lambda| e^{j\omega}$ where $|\lambda|$ is the distance to the origin in the complex plane, then we get

$$\lambda^t = |\lambda|^t e^{j\omega t} = |\lambda|^t \cos(\omega t) + j\,|\lambda|^t \sin(\omega t),$$

the real part of which is depicted in Figure 1 for various values of λ. Note that the envelope $|\lambda|^t$ decays to zero when λ is inside the unit disk ($|\lambda| < 1$) and grows unbounded when it is outside ($|\lambda| > 1$), which is consistent with our stability criterion.

Likewise, for a continuous-time system each eigenvalue λ_i contributes a function of the form $e^{\lambda_i t}$ to the solution. Decomposing λ into its real and imaginary parts, $\lambda = v + j\omega$, we get

$$e^{\lambda t} = e^{vt} e^{j\omega t} = e^{vt} \cos(\omega t) + j\, e^{vt} \sin(\omega t).$$

Figure 2 depicts the real part of $e^{\lambda t}$ for various values of λ. Note that the envelope e^{vt} decays when $v = \text{Re}(\lambda) < 0$ as in our stability condition.

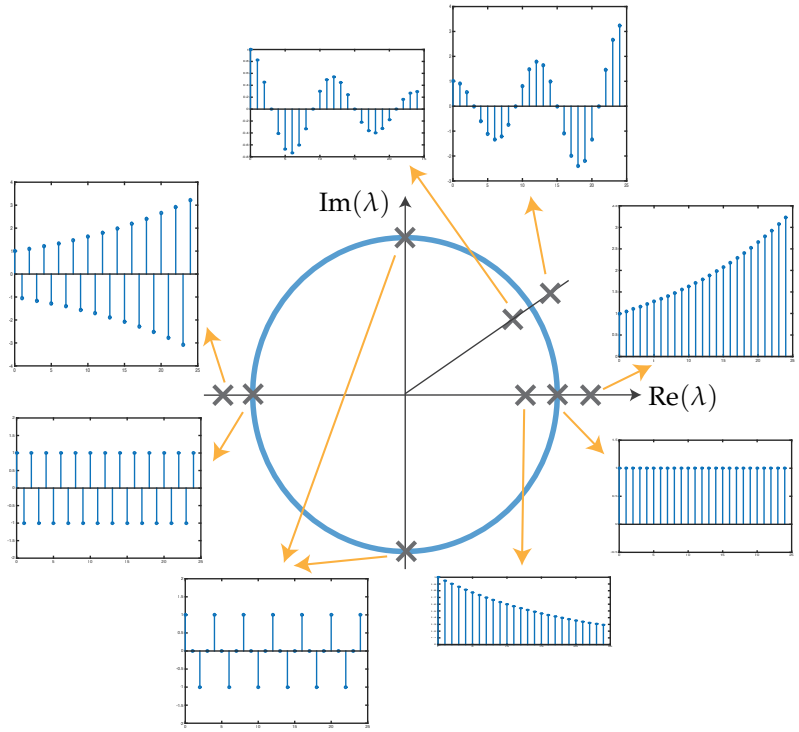


Figure 1: The real part of λ^t for various values of λ in the complex plane. It grows unbounded when $|\lambda| > 1$, decays to zero when $|\lambda| < 1$, and has constant amplitude when λ is on the unit circle ($|\lambda| = 1$).

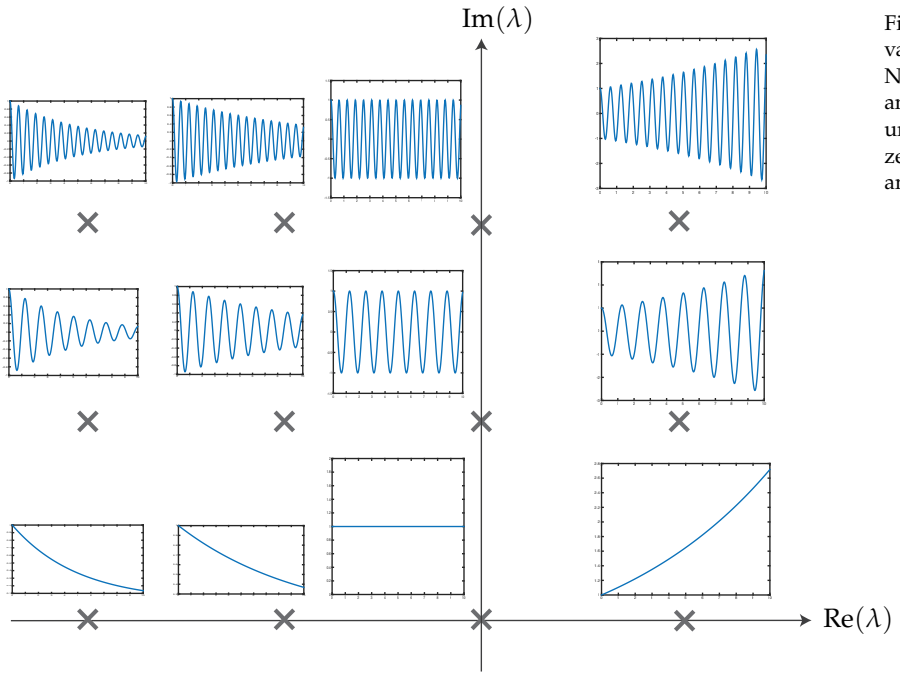


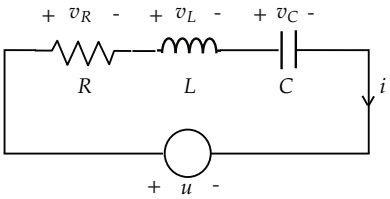
Figure 2: The real part of $e^{\lambda t}$ for various values of λ in the complex plane. Note that $e^{\lambda t}$ is oscillatory when λ has an imaginary component. It grows unbounded when $\mathrm{Re}\{\lambda\} > 0$, decays to zero when $\mathrm{Re}\{\lambda\} < 0$, and has constant amplitude when $\mathrm{Re}\{\lambda\} = 0$.

Example 7: In Example 2 we modeled the RLC circuit depicted on the right as

$$\frac{dx_1(t)}{dt} = \frac{1}{C}x_2(t)$$
$$\frac{dx_2(t)}{dt} = \frac{1}{L}\left(-x_1(t) - Rx_2(t) + u(t)\right)$$

where $x_1 = v_C$ and $x_2 = i$. Since this model is linear we can rewrite it in the form (28), with

$$A = \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix}.$$



Then the roots of

$$\det(\lambda I - A) = \lambda^2 + \frac{R}{L}\lambda + \frac{1}{LC}$$

give the eigenvalues:

$$\lambda_{1,2} = -\alpha \mp \sqrt{\alpha^2 - \omega_0^2} \quad \text{where} \quad \alpha \triangleq \frac{R}{2L},\ \omega_0 \triangleq \frac{1}{\sqrt{LC}}.$$

For $\alpha > \omega_0$ we have two real, negative eigenvalues which indicate a damped response. For $\alpha < \omega_0$, we get the complex eigenvalues

$$\lambda_{1,2} = -\alpha \mp j\omega \quad \text{where} \quad \omega \triangleq \sqrt{\omega_0^2 - \alpha^2},$$

indicating oscillations with frequency ω and decaying envelope $e^{-\alpha t}$.

Controllability

We have seen that the solution of the discrete-time state model

$$\vec{x}(t+1) = A\vec{x}(t) + Bu(t), \tag{30}$$

where $\vec{x}(t)$ is an n-dimensional vector, is given by

$$\vec{x}(t) = A^t\vec{x}(0) + A^{t-1}Bu(0) + A^{t-2}Bu(1) + \cdots + ABu(t-2) + Bu(t-1)$$

or, equivalently,

$$\vec{x}(t) - A^t\vec{x}(0) = \underbrace{\begin{bmatrix} A^{t-1}B & A^{t-2}B & \cdots & AB & B \end{bmatrix}}_{\triangleq R_t} \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(t-2) \\ u(t-1) \end{bmatrix}. \tag{31}$$

Can we find an input sequence $u(0), u(1), \ldots, u(t-1)$ that brings the state from $\vec{x}(0)$ to a desired value $\vec{x}(t) = \vec{x}_{\text{target}}$? The answer is yes

if $\vec{x}_{\text{target}} - A^t\vec{x}(0)$ lies in the range space of R_t because, then we can find appropriate values for $u(0), u(1), \ldots, u(t-1)$, *i.e.* an appropriate linear combination of the columns of R_t, so that the right hand side of (31) is $\vec{x}_{\text{target}} - A^t\vec{x}(0)$ and, thus, $\vec{x}(t) = \vec{x}_{\text{target}}$.

Now, if we want to be able to reach any $\vec{x}_{\text{target}}$ from any initial $\vec{x}(0)$ – a property we henceforth refer to as *controllability* – then the range of R_t must be the whole n-dimensional space, that is R_t must have n linearly independent columns for some t.

Since increasing t amounts to adding more columns to R_t, it may appear that we may eventually have n independent columns. However, this is not so: *if we don't have n independent columns in R_t at $t = n$, we never will for $t > n$.* This is a consequence of a result[3] in linear algebra which states that, if A is $n \times n$, then A^n can be written as a linear combination of $A^{n-1}, \ldots, A, I$:

$$A^n = \alpha_{n-1}A^{n-1} + \cdots + \alpha_1 A + \alpha_0 I \quad \text{for some} \quad \alpha_{n-1}, \ldots, \alpha_1, \alpha_0.$$

[3] This result is known as the Cayley-Hamilton Theorem and its details are beyond the scope of this course. You can consult the Wikipedia article if you are interested.

Multiplying both sides from the right by B we see that A^nB is itself a linear combination of $A^{n-1}B, \ldots, AB, B$. This means that the new columns in R_{n+1} are merely linear combinations of the columns of R_n, and the same argument extends to $R_{n+2}, R_{n+3}, \cdots$.

Thus, for controllability, we need R_n to have n linearly independent columns, which means rank $= n$:

$$\text{Controllability} \quad \Leftrightarrow \quad \text{rank}\begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} = n.$$

Example 8: The system

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}}_{A}\vec{x}(t) + \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{B}u(t)$$

is uncontrollable because the matrix

$$\begin{bmatrix} AB & B \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$$

has rank $= 1$ rather than $n = 2$. The reason for uncontrollability becomes clear if we write the equation for $x_2(t)$ explicitly:

$$x_2(t+1) = 2x_2(t).$$

The right hand side doesn't depend on $u(t)$ or $x_1(t)$, which means that $x_2(t)$ evolves independently and can be influenced neither directly by input $u(t)$, nor indirectly through the other state $x_1(t)$.

Example 9: Consider a vehicle moving in a lane with speed $v(t)$. We are able to change the acceleration, $u(t)$, every T seconds and it remains constant for the next T seconds. This suggests a discrete-time model which we obtain from the relationship

$$\frac{d}{dt}v(t) = u(t)$$

by integrating both sides from t to $t+T$ and keeping in mind that $u(t)$ is constant in this interval, that is $u(t+\tau) = u(t)$ for $\tau \in [0, T)$:

$$v(t+T) - v(t) = \int_0^T u(t+\tau)d\tau = Tu(t). \tag{32}$$

Next, we let $p(t)$ denote the position and note that

$$\frac{d}{dt}p(t) = v(t).$$

Integrating from t to $t+T$ and substituting $v(t+\tau) = v(t) + \tau u(t)$ we get

$$p(t+T) - p(t) = \int_0^T (v(t) + \tau u(t))d\tau = Tv(t) + \frac{1}{2}T^2u(t). \tag{33}$$

Finally we combine (32) and (33) into the state model:

$$\begin{bmatrix} p(t+T) \\ v(t+T) \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} p(t) \\ v(t) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{1}{2}T^2 \\ T \end{bmatrix}}_{B} u(t)$$

which is of the form (30) if we take the unit time to be T seconds.

The controllability condition above holds: the rank of

$$\begin{bmatrix} AB & B \end{bmatrix} = \begin{bmatrix} \frac{3}{2}T^2 & \frac{1}{2}T^2 \\ T & T \end{bmatrix}$$

is indeed $n = 2$. This confirms that we can move the vehicle to an arbitrary target location p_{target} and stop there ($v_{\text{target}} = 0$) by applying an appropriate input sequence obtained from (31).

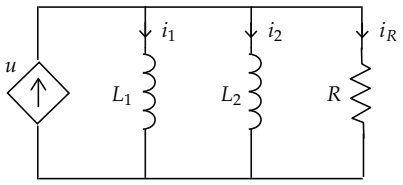
Controllability in Continuous-Time

The controllability condition for the continuous-time system

$$\frac{d}{dt}\vec{x}(t) = A\vec{x}(t) + Bu(t)$$

is exactly the same: $\text{rank}(R_n) = n$ where R_n is as defined above. We omit the derivation for this case but illustrate the result with a circuit example.

Example 10: For the circuit depicted on the right we treat the current source as the control $u(t)$, and the inductor currents $i_1(t)$ and $i_2(t)$ as the state variables.



Since the voltage across each capacitor is the same as the voltage across the resistor, we have

$$\begin{aligned} L_1 \frac{di_1(t)}{dt} &= Ri_R(t) \\ L_2 \frac{di_2(t)}{dt} &= Ri_R(t). \end{aligned} \tag{34}$$

Substituting $i_R = u - i_1 - i_2$ from KCL and dividing the equations by L_1 and L_2 respectively, we get

$$\begin{bmatrix} \frac{di_1(t)}{dt} \\ \frac{di_2(t)}{dt} \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{R}{L_1} & -\frac{R}{L_1} \\ -\frac{R}{L_2} & -\frac{R}{L_2} \end{bmatrix}}_{A} \begin{bmatrix} i_1(t) \\ i_2(t) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{R}{L_1} \\ \frac{R}{L_2} \end{bmatrix}}_{B} u(t).$$

Note that

$$AB = \begin{bmatrix} -\frac{R}{L_1}\left(\frac{R}{L_1} + \frac{R}{L_2}\right) \\ -\frac{R}{L_2}\left(\frac{R}{L_1} + \frac{R}{L_2}\right) \end{bmatrix} = -\left(\frac{R}{L_1} + \frac{R}{L_2}\right) B$$

which means that AB and B are linearly dependent. Thus

$$\operatorname{rank} \begin{bmatrix} AB & B \end{bmatrix} = 1,$$

and the model is *not* controllable.

To see the physical obstacle to controllability note that the two inductors in parallel share the same voltage:

$$L_1 \frac{di_1(t)}{dt} = L_2 \frac{di_2(t)}{dt}.$$

Thus,

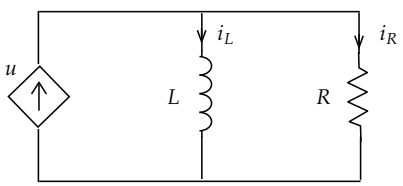
$$\frac{d}{dt}\left(L_1 i_1(t) - L_2 i_2(t)\right) = 0$$

which means that the difference between the two inductor fluxes, $L_1 i_1 - L_2 i_2$, remains constant no matter what u we apply.

Because of this constraint we can't control i_1 and i_2 independently. We can, however, control the total current $i_L = i_1 + i_2$ which obeys, from (34),

$$\frac{di_L(t)}{dt} = \left(\frac{1}{L_1} + \frac{1}{L_2}\right) Ri_R(t) = \frac{R}{L}(-i_L(t) + u(t))$$

where $L \triangleq \left(\frac{1}{L_1} + \frac{1}{L_2}\right)^{-1}$. Note that this is the governing equation for the circuit on the right where the two inductors are lumped into one.



State Feedback Control

Suppose we are given a single-input control system

$$\vec{x}(t+1) = A\vec{x}(t) + Bu(t), \quad u(t) \in \mathbb{R}, \tag{35}$$

and we wish to bring the solution $\vec{x}(t)$ to the equilibrium $\vec{x} = 0$ from any initial condition $\vec{x}(0)$.

To achieve this goal we will study a "control policy" of the form

$$u(t) = k_1 x_1(t) + k_2 x_2(t) + \cdots + k_n x_n(t) \tag{36}$$

where $k_1, k_2, \ldots, k_n$ are to be determined. Rewriting (36) as

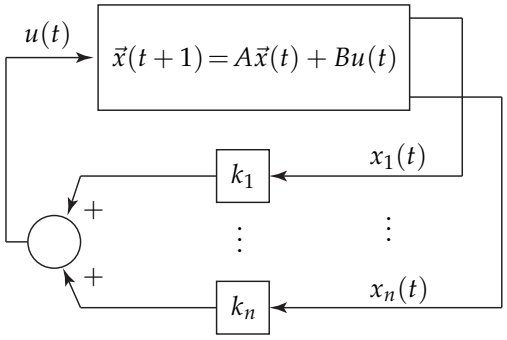
$$\boxed{u(t) = K\vec{x}(t)} \tag{37}$$

with row vector $K = [k_1\, k_2\, \cdots\, k_n]$, and substituting in (35), we get

$$\boxed{\vec{x}(t+1) = (A + BK)\vec{x}(t).} \tag{38}$$

Thus, if we can choose K such that all eigenvalues of $A + BK$ satisfy the stability condition $|\lambda_i(A + BK)| < 1$, then $\vec{x}(t) \to 0$ from any $\vec{x}(0)$.

We will see that if the system (35) is controllable, then we can arbitrarily assign the eigenvalues of $A + BK$ with the choice of K. Thus, in addition to bringing the eigenvalues inside the unit disk for stability, we can place them in favorable locations to shape the transients, *e.g.*, to achieve a well damped convergence.



We refer to (38) as the "closed-loop" system since the control policy (36) generates a feedback loop as depicted in the block diagram. The state variables are measured at every time step t and the input $u(t)$ is synthesized as a linear combination of these measurements.

Comparison to Open Loop Control

Recall that controllability allowed us to calculate an input sequence $u(0), u(1), u(2), \ldots$ that drives the state from $\vec{x}(0)$ to any $\vec{x}_{\text{target}}$. Thus, an alternative to the feedback control (36) is to select $\vec{x}_{\text{target}} = 0$, calculate an input sequence based on $\vec{x}(0)$, and to apply this sequence in an "open-loop" fashion without using further state measurements as depicted below.

$$u(0), u(1), u(2), \ldots \longrightarrow \boxed{\vec{x}(t+1) = A\vec{x}(t) + Bu(t)}$$

The trouble with this open-loop approach is that it is sensitive to uncertainties in A and B, and does not make provisions against disturbances that may act on the system.

By contrast, feedback offers a degree of robustness: if our design of K brings the eigenvalues of $A + BK$ to well within the unit disk, then small perturbations in A and B would not move these eigenvalues outside the disk. Thus, despite the uncertainty, solutions converge to $\vec{x} = 0$ in the absence of disturbances and remain bounded in the presence of bounded disturbances.

Eigenvalue Assignment by State Feedback: Examples

Example 11: Consider the second order system

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} 0 & 1 \\ a_1 & a_2 \end{bmatrix}}_{A} \vec{x}(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u(t)$$

and note that the eigenvalues of A are the roots of the polynomial

$$\det(\lambda I - A) = \lambda^2 - a_2\lambda - a_1.$$

If we substitute the control

$$u(t) = K\vec{x}(t) = k_1 x_1(t) + k_2 x_2(t)$$

the closed-loop system becomes

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} 0 & 1 \\ a_1 + k_1 & a_2 + k_2 \end{bmatrix}}_{A + BK} \vec{x}(t)$$

and, since $A + BK$ has the same structure as A with a_1, a_2 replaced by $a_1 + k_1$, $a_2 + k_2$, the eigenvalues of $A + BK$ are the roots of

$$\lambda^2 - (a_2 + k_2)\lambda - (a_1 + k_1).$$

Now if we want to assign the eigenvalues of $A + BK$ to desired values λ_1 and λ_2, we must match the polynomial above to

$$(\lambda - \lambda_1)(\lambda - \lambda_2) = \lambda^2 - (\lambda_1 + \lambda_2)\lambda + \lambda_1\lambda_2,$$

that is,

$$a_2 + k_2 = \lambda_1 + \lambda_2 \quad \text{and} \quad a_1 + k_1 = -\lambda_1\lambda_2.$$

This is indeed accomplished with the choice $k_1 = -a_1 - \lambda_1\lambda_2$ and $k_2 = -a_2 + \lambda_1 + \lambda_2$, which means that we can assign the closed-loop eigenvalues as we wish.

Example 12: Let's apply the eigenvalue assignment procedure above to

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}}_{A} \vec{x}(t) + \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{B} u(t).$$

Now we have

$$A + BK = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} = \begin{bmatrix} 1 + k_1 & 1 + k_2 \\ 0 & 2 \end{bmatrix}$$

and, because this matrix is upper diagonal, its eigenvalues are the diagonal entries:

$$\lambda_1 = 1 + k_1 \quad \text{and} \quad \lambda_2 = 2.$$

Note that we can move λ_1 with the choice of k_1, but we have no control over λ_2. In fact, since $|\lambda_2| > 1$, the closed-loop system remains unstable no matter what control we apply.

This is a consequence of the uncontrollability of this system, which was shown in Example 8. The second state equation

$$x_2(t+1) = 2x_2(t)$$

can't be influenced by $u(t)$, and $x_2(t) = 2^t x_2(0)$ grows exponentially.

Continuous-Time State Feedback

The idea of state feedback is identical for a continuous-time system,

$$\frac{d}{dt}\vec{x}(t) = A\vec{x}(t) + Bu(t), \quad u(t) \in \mathbb{R}.$$

To bring $\vec{x}(t)$ to the equilibrium $\vec{x} = 0$ we apply

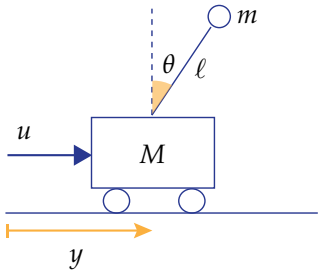
$$u(t) = K\vec{x}(t)$$

and obtain the closed-loop system

$$\frac{d}{dt}\vec{x}(t) = (A + BK)\vec{x}(t).$$

The only difference from discrete-time is the stability criterion: we must choose K such that $\mathrm{Re}\{\lambda_i(A + BK)\} < 0$ for each eigenvalue λ_i.

Example 13: Consider the inverted pendulum system depicted below.



The equations of motion are

$$\ddot{y} = \frac{1}{\frac{M}{m} + \sin^2\theta}\left(\frac{u}{m} + \dot{\theta}^2\ell\sin\theta - g\sin\theta\cos\theta\right)$$
$$\ddot{\theta} = \frac{1}{\ell(\frac{M}{m} + \sin^2\theta)}\left(-\frac{u}{m}\cos\theta - \dot{\theta}^2\ell\cos\theta\sin\theta + \frac{M+m}{m}g\sin\theta\right)$$

and linearization about the upright position $\theta = 0$, $\dot{\theta} = 0$ gives

$$\begin{aligned}\ddot{y} &= -\frac{m}{M}g\theta + \frac{1}{M}u \\ \ddot{\theta} &= \frac{M+m}{M\ell}g\theta - \frac{1}{M\ell}u.\end{aligned} \tag{39}$$

We write (39) in state space form as

$$\frac{d}{dt}\begin{bmatrix}\theta(t)\\ \dot{\theta}(t)\\ \dot{y}(t)\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1 & 0\\ \frac{M+m}{M\ell}g & 0 & 0\\ -\frac{m}{M}g & 0 & 0\end{bmatrix}}_{A}\begin{bmatrix}\theta(t)\\ \dot{\theta}(t)\\ \dot{y}(t)\end{bmatrix} + \underbrace{\begin{bmatrix}0\\ -\frac{1}{M\ell}\\ \frac{1}{M}\end{bmatrix}}_{B}u(t),$$

where we have omitted $y(t)$ from the state vector because we are interested in stabilizing the point $\theta = 0$, $\dot{\theta} = 0$, $\dot{y} = 0$, and we are not concerned about the final value of the position $y(t)$. If it is of interest to bring $y(t)$ to a specific position the state equations above can be augmented with $y(t)$, leading to a fourth order model.

We now design a state feedback controller,

$$u(t) = k_1\theta(t) + k_2\dot{\theta}(t) + k_3\dot{y}(t).$$

Substituting the values $M = 1$, $m = 0.1$, $l = 1$, and $g = 10$, we get

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 11 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}}_{A} + \underbrace{\begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}}_{B} \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 11 - k_1 & -k_2 & -k_3 \\ -1 + k_1 & k_2 & k_3 \end{bmatrix}.$$

The characteristic polynomial of this matrix is

$$\lambda^3 + (k_2 - k_3)\lambda^2 + (k_1 - 11)\lambda + 10k_3 = 0$$

and, as in Example 11, we can choose k_1, k_2, k_3, to match the coefficients of this polynomial to those of

$$(\lambda - \lambda_1)(\lambda - \lambda_2)(\lambda - \lambda_3)$$

where λ_1, λ_2, λ_3 are desired closed-loop eigenvalues.

Eigenvalue Assignment by State Feedback

Our goal in this section is to show that controllability allows us to arbitrarily assign the eigenvalues of $A + BK$ with the choice of K. We will do this by generalizing the special structure in Example 11:

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} 0 & 1 \\ a_1 & a_2 \end{bmatrix}}_{A} \vec{x}(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u(t), \tag{40}$$

where the eigenvalues of $A + BK$ are the roots of

$$\lambda^2 - (a_2 + k_2)\lambda - (a_1 + k_1)$$

and can be assigned to desired values λ_1 and λ_2 by matching the polynomial above to

$$(\lambda - \lambda_1)(\lambda - \lambda_2) = \lambda^2 - (\lambda_1 + \lambda_2)\lambda + \lambda_1\lambda_2.$$

Controller canonical form

We now generalize the special structure of A and B in (40) to $n > 2$:

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & 1 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \\ a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \end{bmatrix}. \tag{41}$$

The first benefit of this structure is the simple form of the polynomial

$$\det(\lambda I - A) = \lambda^n - a_n\lambda^{n-1} - a_{n-1}\lambda^{n-2} - \cdots - a_2\lambda - a_1$$

whose roots constitute the eigenvalues of A. The second benefit is that $A + BK$ preserves the structure of A, except that the entry a_i is replaced by $a_i + k_i$, $i = 1, \ldots, n$. Therefore,

$$\det(\lambda I - (A + BK)) = \lambda^n - (a_n + k_n)\lambda^{n-1} \cdots - (a_2 + k_2)\lambda - (a_1 + k_1)$$

and assigning the closed-loop eigenvalues to desired values $\lambda_1, \ldots, \lambda_n$ amounts to matching the coefficients of this polynomial to those of

$$(\lambda - \lambda_1)(\lambda - \lambda_2) \cdots (\lambda - \lambda_n). \tag{42}$$

Thus eigenvalue assignment is straightforward when A and B have the special form above, known as the *controller canonical form*.

Example 14: Suppose we want all eigenvalues of $A + BK$ to be 0 for

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 2 & 3 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

This means that we must match the polynomial

$$\det(\lambda I - (A + BK)) = \lambda^3 - (3 + k_3)\lambda^2 - (2 + k_2)\lambda - (1 + k_1)$$

to λ^3 which has three roots at 0. This is accomplished with the choice $k_1 = -1$, $k_2 = -2$, $k_3 = -3$. □

Controllability enables eigenvalue assignment

It turns out that we can bring any controllable, single-input system

$$\vec{x}(t+1) = \tilde{A}\vec{x}(t) + \tilde{B}u(t) \tag{43}$$

to the controller canonical form with a change of variables $\vec{z} = T\vec{x}$; that is, there exists T such that

$$T\tilde{A}T^{-1} = A \quad \text{and} \quad T\tilde{B} = B \tag{44}$$

where A and B are as in (41).

This means that we can design a state feedback $u = K\vec{z}$ to assign the eigenvalues of $A + BK$ using the procedure above for the controller canonical form. Since $\vec{z} = T\vec{x}$, $u = K\vec{z}$ is identical to $u = \tilde{K}\vec{x}$ where

$$\tilde{K} = KT \tag{45}$$

and, since $T(\tilde{A} + \tilde{B}\tilde{K})T^{-1} = A + BK$, the eigenvalues of $\tilde{A} + \tilde{B}\tilde{K}$ are the same as those of $A + BK$.

Conclusion: If the system (43) is controllable, then we can arbitrarily assign the eigenvalues of $\tilde{A} + \tilde{B}\tilde{K}$ with an appropriate choice of $\tilde{K}$.

How can we find a matrix T that satisfies (44)? Recall the matrix we use for checking controllability:

$$R_n = \begin{bmatrix} A^{n-1}B & \cdots & AB & B \end{bmatrix}. \tag{46}$$

If we substitue (44), we see that

$$\begin{aligned} R_n &= \begin{bmatrix} (T\tilde{A}^{n-1}T^{-1})(T\tilde{B}) & \cdots & (T\tilde{A}T^{-1})(T\tilde{B}) & (T\tilde{B}) \end{bmatrix} \\ &= T \underbrace{\begin{bmatrix} \tilde{A}^{n-1}\tilde{B} & \cdots & \tilde{A}\tilde{B} & \tilde{B} \end{bmatrix}}_{= \tilde{R}_n} \end{aligned} \tag{47}$$

which suggests the choice $T = R_n \tilde{R}_n^{-1}$.

$\tilde{R}_n$ is full rank, thus invertible, because (43) is controllable. Likewise, R_n has the lower diagonal form

$$R_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \star & 1 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ \star & \cdots & \star & 1 \end{bmatrix}$$

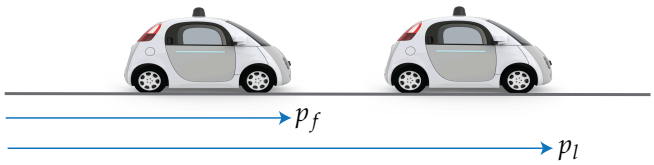
that has rank n regardless of the values of the entries marked as '$\star$'. Then $T = R_n \tilde{R}_n^{-1}$ is itself invertible, thus a viable choice for (44).

Summary

We used the controller canonical form to prove that controllability allows us to arbitrarily assign the eigenvalues of $A + BK$. In practice, however, it is *not* necessary to bring the system to the controller canonical form to assign its eigenvalues. We can simply calculate the characteristic polynomial of $A + BK$ and choose K to match its coefficients to those of the desired closed-loop polynomial (42). That's what we did in Example 13 for the inverted pendulum model, which was not in controller canonical form.

Case Study: Cooperative Adaptive Cruise Control (CACC)

Consider a vehicle following another as depicted below.



We denote by p_l the position of the leader and by p_f the position of the follower, and write the continuous-time models

$$\begin{aligned} \frac{d}{dt}p_l(t) &= v_l(t) \\ \frac{d}{dt}v_l(t) &= u_l(t) \end{aligned} \tag{48}$$

and

$$\begin{aligned} \frac{d}{dt}p_f(t) &= v_f(t) \\ \frac{d}{dt}v_f(t) &= u_f(t). \end{aligned} \tag{49}$$

To maintain a constant distance δ between p_l and p_f, we define

$$x_1 \triangleq p_l - p_f - \delta \quad x_2 \triangleq v_l - v_f \quad u \triangleq u_l - u_f, \tag{50}$$

and obtain from (48) and (49) the following model that describes the relative motion of the two vehicles:

$$\begin{aligned} \frac{d}{dt}x_1(t) &= x_2(t) \\ \frac{d}{dt}x_2(t) &= u(t). \end{aligned} \tag{51}$$

Then the task is to stabilize the equilibrium $x_1 = 0$, $x_2 = 0$ which means $p_l - p_f = \delta$ and $v_l = v_f$. This is accomplished with

$$u(t) = k_1 x_1(t) + k_2 x_2(t)$$

where k_1 and k_2 are selected such that the eigenvalues of

$$A + BK = \begin{bmatrix} 0 & 1 \\ k_1 & k_2 \end{bmatrix}$$

have negative real parts.

Recall from (50) that $u = u_l - u_f$. Thus, if the lead vehicle broadcasts its acceleration $u_l(t)$ to the follower via vehicle-to-vehicle wireless communication[4], then the follower can implement the controller

$$\begin{aligned} u_f(t) &= u_l(t) - u(t) \\ &= u_l(t) - k_1 x_1(t) - k_2 x_2(t) \\ &= u_l(t) - k_1(p_l(t) - p_f(t) - \delta) - k_2(v_l(t) - v_f(t)) \end{aligned} \tag{52}$$

[4] hence the term *cooperative* adaptive cruise control

using range sensors to obtain $p_l(t) - p_f(t)$ and $v_l(t) - v_f(t)$. In this implementation the lead vehicle chooses its own input $u_l(t)$ without regard to the follower, and the follower applies the controller (52) to automatically follow the leader with constant relative distance δ.

Outputs and Observability

In applications we may not have measurements of all state variables, but only an "output" vector

$$\vec{y}(t) = C\vec{x}(t).$$

If we have n states and p outputs, then C is a $p \times n$ matrix. For example, if we measure only the ith state variable, $y(t) = x_i(t)$, then C has a single row that consists of the ith unit vector.

Thus we augment our state model as

$$\begin{aligned} \vec{x}(t+1) &= A\vec{x}(t) + B\vec{u}(t) \\ \vec{y}(t) &= C\vec{x}(t). \end{aligned} \tag{53}$$

Then an important question is, *if we only monitor the output $\vec{y}(t)$ can we infer the full state $\vec{x}(t)$ with the help of this model?* If the answer is yes, we say that the system is *observable*.

Observability is equivalent to the ability to determine the initial state $\vec{x}(0)$ from a set of measurements $\vec{y}(0), \vec{y}(1), \ldots, \vec{y}(t)$. This is because, if we can determine $\vec{x}(0)$, then we can use the explicit solution of the state equation studied earlier to find $\vec{x}(t)$.

To see how we may determine $\vec{x}(0)$ from $\vec{y}(0), \vec{y}(1), \ldots, \vec{y}(t)$, assume for now $u(t) = 0$ for all t and note that

$$\begin{aligned} \vec{y}(0) &= C\vec{x}(0) \\ \vec{y}(1) &= C\vec{x}(1) = CA\vec{x}(0) \\ &\vdots \\ \vec{y}(t) &= C\vec{x}(t) = CA^t\vec{x}(0) \end{aligned} \tag{54}$$

or, equivalently,

$$\begin{bmatrix} \vec{y}(0) \\ \vec{y}(1) \\ \vdots \\ \vec{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^t \end{bmatrix}}_{\triangleq O_t} \vec{x}(0). \tag{55}$$

To *uniquely* determine $\vec{x}(0)$, which has n entries, we need the matrix O_t to have n linearly independent rows so that its null space is $\{0\}$.

It follows from an argument similar to the one we made for controllability that, if O_t doesn't have n independent rows at $t = n-1$, then it never will[5] for $t \geq n$.

[5] Try to prove this claim by adapting our argument for the columns of R_t in controllability to the rows of O_t.

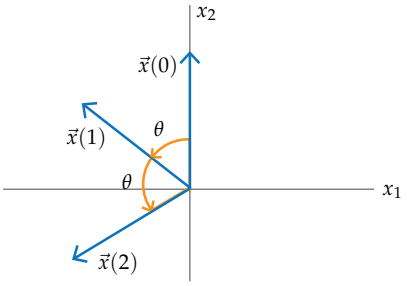
Thus, for observability, we need O_{n-1} to have n linearly independent rows, that is rank $= n$:

$$\text{Observability} \quad \Leftrightarrow \quad \text{rank} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} = n.$$

Although we assumed $u(t) = 0$ for all t above, adding inputs does not change this observability condition. The only change in this case is that the right hand side of (55) must be augmented with another term that depends on the history of the input $\vec{u}(0), \cdots, \vec{u}(t-1)$. But this term is known, since we know the control inputs we have applied, and can be subtracted from both sides of the equation. The rest of the arguments above are therefore unchanged.

Example 15: Consider the second order system

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{A} \vec{x}(t) \tag{56}$$



where the matrix A rotates the state vector by an angle of θ at each time instant, as depicted on the right. If the output is

$$y(t) = x_1(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \vec{x}(t) \tag{57}$$

we get

$$\begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \cos\theta & -\sin\theta \end{bmatrix}$$

which has rank $= n = 2$ when $\sin\theta \neq 0$, and rank $= 1$ if $\sin\theta = 0$. Thus, we lose observability for $\theta = n\pi$, $n = 1, 2$.

To see why observability is lost, suppose $\theta = \pi$ and $y(t) = x_1(t) = 0$ for each t. Then we can infer that $\vec{x}(0)$ points in the vertical direction, and $\vec{x}(t)$ oscillates back and forth between the positive and negative vertical axes. However, we have no information about the magnitude of this oscillation and, thus, can't determine $\vec{x}(0)$.

Observers

An *observer* is an algorithm that estimates the full state $\vec{x}(t)$ from measurements of $\vec{y}(t)$. Observers complement state feedback controllers by providing estimates for states that are not available for measurement.

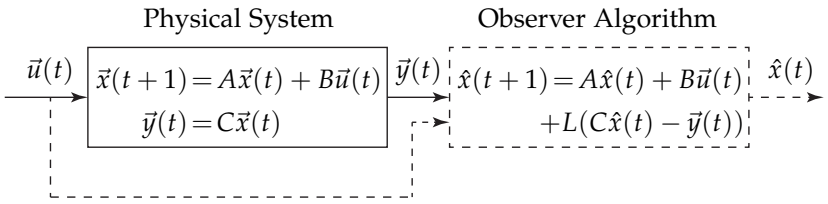
One common observer algorithm is to start with an initial guess $\hat{x}(0)$ and to update it at each time according to the equation

$$\hat{x}(t+1) = A\hat{x}(t) + L(C\hat{x}(t) - \vec{y}(t)) + B\vec{u}(t) \tag{58}$$

where L is a $n \times p$ matrix to be designed. Note that this update rule incorporates the model of the physical system and augments it with the correction term $L(C\hat{x}(t) - \vec{y}(t))$. If the state estimate $\hat{x}(t)$ is consistent with the output measurement $y(t)$ at that time, that is $C\hat{x}(t) = y(t)$, then the correction term is zero and the update rule (58) simply mimics the physical model

$$\vec{x}(t+1) = A\vec{x}(t) + B\vec{u}(t). \tag{59}$$

The block diagram below illustrates how the observer algorithm interacts with the physical system depicted on the left.



How should we design L to guarantee the convergence of $\hat{x}(t)$ to the correct state $x(t)$? To answer this question we define the estimation error

$$\vec{e}(t) \triangleq \hat{x}(t) - \vec{x}(t)$$

and note from (58) and (59) that

$$\begin{aligned} \vec{e}(t+1) &= A(\hat{x}(t) - \vec{x}(t)) + L(C\hat{x}(t) - C\vec{x}(t)) \\ &= (A + LC)\vec{e}(t). \end{aligned} \tag{60}$$

Thus, if we design L such that all eigenvalues of $A + LC$ are within the unit disk then we guarantee $e(t) = \hat{x}(t) - \vec{x}(t) \to 0$; that is, the estimate $\hat{x}(t)$ approaches the state $\vec{x}(t)$ asymptotically.

It turns out that if the system is observable then we can assign the eigenvalues of $A + LC$ with an appropriate choice of L. We prove

this by analogy to controllability. First note that taking the transpose of a matrix does not change its eigenvalues; therefore assigning the eigenvalues of $A + LC$ is equivalent to assigning the eigenvalues of

$$A^T + C^T L^T. \tag{61}$$

Now if we define $A_0 = A^T$, $B_0 = C^T$, $K_0 = L^T$, (61) takes the form $A_0 + B_0 K_0$ and we know that we can assign its eigenvalues by choice of K_0 if the pair (A_0, B_0) is controllable.

The next step, which we leave as an exercise, is to show that the columns of the controllability matrix for (A_0, B_0), when transposed, match the rows of the observability matrix for (A, C). Therefore they have identical ranks and observability of (A, C) implies controllability of (A_0, B_0). Then we can assign the eigenvalues of $A_0 + B_0 K_0$ with the design of K_0, and $L = K_0^T$ assigns the same eigenvalues to $A + LC$.

Example 16: Consider again the system

$$\begin{aligned} \vec{x}(t+1) &= \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{A} \vec{x}(t) \\ y(t) &= \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \vec{x}(t). \end{aligned} \tag{62}$$

If we take $\theta = \pi/2$, for which the system is observable, then

$$A + LC = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} l_1 \\ l_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} l_1 & -1 \\ l_2 + 1 & 0 \end{bmatrix}$$

whose eigenvalues are the roots of

$$\det \begin{bmatrix} \lambda - l_1 & 1 \\ -l_2 - 1 & \lambda \end{bmatrix} = \lambda^2 - l_1 \lambda + (l_2 + 1).$$

Note that, when $l_1 = l_2 = 0$, the eigenvalues are at $\pm j$ which are on the unit circle. To move these eigenvalues to desired values λ_1, λ_2 *inside* the unit disk we must match the coefficients of the polynomial above to those of

$$(\lambda - \lambda_1)(\lambda - \lambda_2) = \lambda^2 - (\lambda_1 + \lambda_2)\lambda + \lambda_1 \lambda_2.$$

This means $l_1 = \lambda_1 + \lambda_2$ and $l_2 = \lambda_1 \lambda_2 - 1$. For example, if we choose $\lambda_{1,2} = \pm 0.9j$ which are inside the unit disk, we get $l_1 = 0$, $l_2 = -0.19$.

Now repeat this example for $\theta = \pi$, in which case

$$A + LC = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} + \begin{bmatrix} l_1 \\ l_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} l_1 - 1 & 0 \\ l_2 & -1 \end{bmatrix}$$

and the eigenvalues are the roots of

$$\det \begin{bmatrix} \lambda + 1 - l_1 & 0 \\ -l_2 & \lambda + 1 \end{bmatrix} = (\lambda + 1 - l_1)(\lambda + 1).$$

Note that an eigenvalue at $\lambda = -1$ persists regardless of the choice of l_1, l_2, which is a result of unobservability in the case where $\theta = \pi$.

Example 17: Navigation is the task of identifying a vehicle's position, attitude, velocity, etc. relative to an inertial frame by integrating multiple measurements which may be inaccurate individually. To relate this task to observers consider the model of a vehicle moving in a lane from Example 9,

$$\begin{bmatrix} p(t+1) \\ v(t+1) \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} p(t) \\ v(t) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{1}{2}T^2 \\ T \end{bmatrix}}_{B} u(t),$$

where $p(t)$ is position, $v(t)$ is velocity, and $u(t)$ is acceleration.

Without further measurements, our observer would be a simple copy of the model above:

$$\begin{bmatrix} \hat{p}(t+1) \\ \hat{v}(t+1) \end{bmatrix} = A \begin{bmatrix} \hat{p}(t) \\ \hat{v}(t) \end{bmatrix} + Bu(t).$$

This primitive strategy is known as *dead reckoning* and leads to major errors over long distances because observer errors accumulate over time and are not dissipated due to the eigenvalues of A at 1.

Modern navigation systems use satellite-based measurements of position. However, since these measurements are noisy and intermittent, it is reasonable to combine them with the dead reckoning method above within the observer

$$\begin{bmatrix} \hat{p}(t+1) \\ \hat{v}(t+1) \end{bmatrix} = A \begin{bmatrix} \hat{p}(t) \\ \hat{v}(t) \end{bmatrix} + L(\hat{p}(t) - p(t)) + Bu(t)$$

where $p(t)$ is treated as the output; that is

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

The task is then to design L such that $A + LC$ has eigenvalues inside the unit disk, which is possible since the pair (C, A) is observable (show this).

With this architecture we make use of satellite measurements but do not rely exclusively on them; we also exploit the system model which can make accurate predictions in the short term.

A more elaborate form of the observer (58), where the matrix L is also updated at each time, is known as the Kalman Filter and is the industry standard in navigation. The Kalman Filter takes into account the statistical properties of the noise that corrupts measurements and minimizes the mean square error between $x(t)$ and $\hat{x}(t)$.

Figure 3: **Rudolf Kalman (1930-2016)** introduced the Kalman Filter as well as many of the state space concepts we studied, such as controllability and observability. He was awarded the National Medal of Science in 2009.

Singular Value Decomposition (SVD)

SVD separates a rank-r matrix $A \in \mathbb{R}^{m\times n}$ into a sum of r rank-1 matrices of the form $\vec{u}\vec{v}^T$ (column times row). Specifically, we can find:

1) orthonormal vectors $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^m$,

2) orthonormal vectors $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^n$,

3) real, positive numbers $\sigma_1, \ldots, \sigma_r$ such that

$$A = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_r \vec{u}_r \vec{v}_r^T. \tag{63}$$

The numbers $\sigma_1, \ldots, \sigma_r$ are called *singular values* and, by convention, we order them from the largest to smallest:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0.$$

In its original form A has mn entries to be stored. In the SVD form each of the r terms is the product of a column of m entries with a row of n entries; therefore we need $r(m+n)$ numbers to store. This is an advantage when r is small relative to m and n, that is $r(m+n) \ll mn$.

In a typical application the exact rank r may not be particularly small, but we may find that the first few singular values, say $\sigma_1, \ldots, \sigma_{\hat{r}}$, are much bigger than the rest, $\sigma_{\hat{r}+1}, \ldots, \sigma_r$. Then it is reasonable to discard the small singular values and approximate A as

$$A \approx \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_{\hat{r}} \vec{u}_{\hat{r}} \vec{v}_{\hat{r}}^T \tag{64}$$

which has rank $= \hat{r}$, thus $\hat{r}(m+n) \ll mn$ numbers to store.

Example 1 (Netflix): Suppose we have a $m \times n$ matrix that contains the ratings of m viewers for n movies. A truncated SVD as suggested above not only saves memory; it also gives insight into the preferences of each viewer. For example we can interpret each rank-1 matrix $\sigma_i \vec{u}_i \vec{v}_i^T$ to be due to a particular attribute, *e.g.*, comedy, action, sci-fi, or romance content. Then σ_i determines how strongly the ratings depend on the ith attribute, the entries of $\vec{v}_i^T$ score each movie

with respect to this attribute, and the entries of $\vec{u}_i$ evaluate how much each viewer cares about this particular attribute. Then truncating the SVD as in (64) amounts to identifying a few key attributes that underlie the ratings. This is useful, for example, in making movie recommendations as you will see in a homework problem.

Finding a SVD

To find a SVD of the form (63) we use either the $n \times n$ matrix A^TA or the $m \times m$ matrix AA^T. We will see later that these matrices have only *real eigenvalues*, r of which are positive and the remaining zero, and a complete set of *orthonormal eigenvectors*. For now we take this as a fact and propose the following procedures to find a SVD for A:

SVD procedure using A^TA

1. Find the eigenvalues λ_i of A^TA and order them from the largest to smallest, so that $\lambda_1 \geq \cdots \geq \lambda_r > 0$ and $\lambda_{r+1} = \cdots = \lambda_n = 0$.

2. Find orthonormal eigenvectors $\vec{v}_i$, so that

$$A^TA\,\vec{v}_i = \lambda_i \vec{v}_i \quad i = 1, \ldots, r. \tag{65}$$

3. Let $\sigma_i = \sqrt{\lambda_i}$ and obtain $\vec{u}_i$ from

$$A\vec{v}_i = \sigma_i \vec{u}_i \quad i = 1, \ldots, r. \tag{66}$$

Justification: To see that $\vec{u}_i$, $i = 1, \ldots, r$, obtained from (66) are orthonormal, multiply (66) from the left by $(A\vec{v}_j)^T = \sigma_j \vec{u}_j^T$:

$$(A\vec{v}_j)^T A\vec{v}_i = \sigma_j \sigma_i \vec{u}_j^T \vec{u}_i. \tag{67}$$

The left hand side is $\vec{v}_j^T A^T A \vec{v}_i = \vec{v}_j^T \lambda_i \vec{v}_i = \sigma_i^2 \vec{v}_j^T \vec{v}_i$, therefore

$$\sigma_i^2 \vec{v}_j^T \vec{v}_i = \sigma_j \sigma_i \vec{u}_j^T \vec{u}_i. \tag{68}$$

The vectors $\vec{v}_i$, $i = 1, \ldots, r$, are orthonormal by construction, which means $\vec{v}_j^T \vec{v}_i = 1$ if $i = j$, and 0 if $i \neq j$. Thus, (68) becomes

$$\sigma_j \sigma_i \vec{u}_j^T \vec{u}_i = \begin{cases} \sigma_i^2 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{69}$$

and $\sigma_j \sigma_i$ cancels with σ_i^2 when $i = j$, proving orthonormality of $\vec{u}_i$, $i = 1, \ldots, r$.

To see why σ_i, $\vec{u}_i$, $\vec{v}_i$ resulting from the procedure above satisfy (63), rewrite (66) in matrix form as:

$$A \underbrace{\begin{bmatrix}\vec{v}_1 \cdots \vec{v}_r\end{bmatrix}}_{\triangleq V_1} = \begin{bmatrix}\vec{u}_1 \cdots \vec{u}_r\end{bmatrix} \begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_r\end{bmatrix}.$$

Next, multiply both sides from the right by V_1^T:

$$AV_1V_1^T = \begin{bmatrix}\vec{u}_1 \cdots \vec{u}_r\end{bmatrix} \begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_r\end{bmatrix} \underbrace{\begin{bmatrix}\vec{v}_1^T \\ \vdots \\ \vec{v}_r^T\end{bmatrix}}_{V_1^T} \tag{70}$$

and note that the right hand side is indeed the decomposition in (63). We conclude by showing that the left hand side is equal to A. To this end define $V_2 = \begin{bmatrix}\vec{v}_{r+1} & \cdots & \vec{v}_n\end{bmatrix}$ whose columns are the remaining orthonormal eigenvectors for $\lambda_{r+1} = \cdots = \lambda_n = 0$. Then $V = \begin{bmatrix}V_1 & V_2\end{bmatrix}$ is an orthonormal matrix and, thus,

$$VV^T = \begin{bmatrix}V_1 & V_2\end{bmatrix} \begin{bmatrix}V_1^T \\ V_2^T\end{bmatrix} = V_1V_1^T + V_2V_2^T = I.$$

Multiplying both sides from the left by A, we get

$$AV_1V_1^T + AV_2V_2^T = A. \tag{71}$$

Since the columns of V_2 are eigenvectors of A^TA for zero eigenvalues we have $A^TAV_2 = 0$, and multiplying this from the left by V_2^T we get $V_2^TA^TAV_2 = (AV_2)^T(AV_2) = 0$. This implies $AV_2 = 0$ and it follows from (71) that $AV_1V_1^T = A$. Thus, the left hand side of (70) is A, which proves that σ_i, $\vec{u}_i$, $\vec{v}_i$ proposed by the procedure above satisfy (63). □

An alternative approach is to use the $m \times m$ matrix AA^T which is preferable to using the $n \times n$ matrix A^TA when $m < n$. Below we summarize the procedure and leave its justification as an exercise.

SVD procedure using AA^T

1. Find the eigenvalues λ_i of AA^T and order them from the largest to smallest, so that $\lambda_1 \geq \cdots \geq \lambda_r > 0$ and $\lambda_{r+1} = \cdots = \lambda_m = 0$.

2. Find orthonormal eigenvectors $\vec{u}_i$, so that

$$AA^T\vec{u}_i = \lambda_i\vec{u}_i \quad i = 1, \ldots, r. \tag{72}$$

3. Let $\sigma_i = \sqrt{\lambda_i}$ and obtain $\vec{v}_i$ from

$$A^T \vec{u}_i = \sigma_i \vec{v}_i \quad i = 1, \ldots, r. \tag{73}$$

Example 2: Let's follow this procedure to find a SVD for

$$A = \begin{bmatrix} 4 & 4 \\ -3 & 3 \end{bmatrix}.$$

We calculate

$$AA^T = \begin{bmatrix} 4 & 4 \\ -3 & 3 \end{bmatrix} \begin{bmatrix} 4 & -3 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 32 & 0 \\ 0 & 18 \end{bmatrix}$$

which happens to be diagonal, so the eigenvalues are $\lambda_1 = 32$, $\lambda_2 = 18$, and we can select the orthonormal eigenvectors:

$$\vec{u}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \vec{u}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{74}$$

The singular values are $\sigma_1 = \sqrt{\lambda_1} = 4\sqrt{2}$, $\sigma_2 = \sqrt{\lambda_2} = 3\sqrt{2}$ and, from (73),

$$\vec{v}_1 = \frac{1}{\sigma_1} A^T \vec{u}_1 = \frac{1}{4\sqrt{2}} \begin{bmatrix} 4 \\ 4 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$
$$\vec{v}_2 = \frac{1}{\sigma_2} A^T \vec{u}_2 = \frac{1}{3\sqrt{2}} \begin{bmatrix} -3 \\ 3 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

which are indeed orthonormal. We leave it as an exercise to derive a SVD using, instead, $A^T A$.

Note that we can change the signs of $\vec{u}_1$ and $\vec{u}_2$ in (74), and they still serve as orthonormal eigenvectors. This implies that SVD is not unique. However, changing the sign of $\vec{u}_i$ changes the sign of $\vec{v}_i$ in (73) accordingly, therefore the product $\vec{u}_i \vec{v}_i^T$ remains unchanged.

Another source of non-uniqueness arises when we have repeated singular values, as illustrated in the next example.

Example 3: To find a SVD for

$$A = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

note that AA^T is the identity matrix, which has repeated eigenvalues at $\lambda_1 = \lambda_2 = 1$ and admits any pair of orthonormal vectors as eigenvectors. We parameterize all such pairs as

$$\vec{u}_1 = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \quad \vec{u}_2 = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} \tag{75}$$

where θ is a free parameter. Since $\sigma_1 = \sigma_2 = 1$, we obtain from (73):

$$\vec{v}_1 = \frac{1}{\sigma_1} A^T \vec{u}_1 = \begin{bmatrix} \cos\theta \\ -\sin\theta \end{bmatrix} \qquad \vec{v}_2 = \frac{1}{\sigma_2} A^T \vec{u}_2 = \begin{bmatrix} -\sin\theta \\ -\cos\theta \end{bmatrix}. \tag{76}$$

Thus, (75)-(76) with $\sigma_1 = \sigma_2 = 1$ constitute a valid SVD for any choice of θ. You can indeed verify that

$$\begin{aligned} \vec{u}_1\vec{v}_1^T + \vec{u}_2\vec{v}_2^T = \begin{bmatrix} \vec{u}_1 & \vec{u}_2 \end{bmatrix} \begin{bmatrix} \vec{v}_1^T \\ \vec{v}_2^T \end{bmatrix} &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ -\sin\theta & -\cos\theta \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \end{aligned} \tag{77}$$

Geometric Interpretation of SVD

To develop a geometric interpretation of SVD, we first rewrite (63) as

$$A = U_1 S V_1^T \tag{78}$$

where $U_1 = \begin{bmatrix} \vec{u}_1 \cdots \vec{u}_r \end{bmatrix}$ is $m \times r$, $V_1 = \begin{bmatrix} \vec{v}_1 \cdots \vec{v}_r \end{bmatrix}$ is $n \times r$, and S is the $r \times r$ diagonal matrix with entries $\sigma_1, \ldots, \sigma_r$.

Next we form the $m \times m$ orthonormal matrix

$$U = [U_1 \; U_2]$$

where the columns of $U_2 = [\vec{u}_{r+1} \; \cdots \; \vec{u}_m]$ are eigenvectors of AA^T corresponding to zero eigenvalues. Likewise we define $V_2 = [\vec{v}_{r+1} \; \cdots \; \vec{v}_n]$ whose columns are orthonormal eigenvectors of A^TA for zero eigenvalues, and obtain the $n \times n$ orthogonal matrix

$$V = [V_1 \; V_2].$$

Then we write

$$\boxed{A = U \underbrace{\begin{bmatrix} S & 0_{r\times(n-r)} \\ 0_{(m-r)\times r} & 0_{(m-r)\times(n-r)} \end{bmatrix}}_{\triangleq \Sigma} V^T} \tag{79}$$

which is identical to (78) but exhibits square and orthonormal matrices U and V^T that enable the geometric interpretation below.

Note that multiplying a vector by an orthonormal matrix does not change its length. This follows because $U^TU = I$, which implies

$$\|U\vec{x}\|^2 = (U\vec{x})^T(U\vec{x}) = \vec{x}^TU^TU\vec{x} = \vec{x}^T\vec{x} = \|\vec{x}\|^2.$$

Thus we can interpret multiplication by an orthonormal matrix as a combination of operations that don't change length, such as rotations, and reflections.

Since S is diagonal with entries $\sigma_1, \ldots, \sigma_r$, multiplying a vector by Σ defined in (79) stretches the first entry of the vector by σ_1, the second entry by σ_2, and so on.

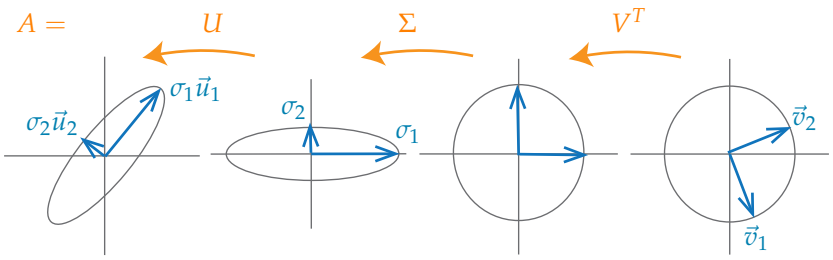
Combining these observations we interpret $A\vec{x}$ as the composition of three operations:

1) $V^T\vec{x}$ which reorients $\vec{x}$ without changing its length,

2) $\Sigma V^T\vec{x}$ which stretches the resulting vector along each axis with the corresponding singular value,

3) $U\Sigma V^T\vec{x}$ which again reorients the resulting vector without changing its length.

The figure below illustrates these three operations moving from the right to the left:



The geometric interpretation above reveals that σ_1 is the largest amplification factor a vector can experience upon multiplication by A:

if the length of $\vec{x}$ is $\|\vec{x}\| = 1$ then $\|A\vec{x}\| \leq \sigma_1$.

For $\vec{x} = \vec{v}_1$ we get $\|A\vec{x}\| = \sigma_1$ with *equality* because $V^T\vec{v}_1$ is the first unit vector which, when multiplied by Σ, gets stretched by σ_1.

Symmetric Matrices

We say that a square matrix Q is *symmetric* if

$$Q = Q^T.$$

Note that the matrices A^TA and AA^T we used to compute a SVD for A are automatically symmetric: using the identities $(AB)^T = B^TA^T$ and $(A^T)^T = A$ you can verify $(A^TA)^T = A^TA$ and $(AA^T)^T = AA^T$.

Below we derive important properties of symmetric matrices that we used without proof in our SVD procedures.

A symmetric matrix has real eigenvalues and eigenvectors.

Let Q be symmetric and let

$$Qx = \lambda x, \tag{80}$$

that is λ is an eigenvalue and x is an eigenvector. Let $\lambda = a + jb$ and define the conjugate $\bar{\lambda} = a - jb$. To show that $b = 0$, that is λ is real, we take conjugates of both sides of $Qx = \lambda x$ to obtain

$$Q\bar{x} = \bar{\lambda}\bar{x} \tag{81}$$

where we used the fact that Q is real. The transpose of (81) is

$$\bar{x}^T Q = \bar{\lambda}\bar{x}^T. \tag{82}$$

Now multiply (80) from the left by $\bar{x}^T$ and (82) from the right by x:

$$\begin{aligned} \bar{x}^T Qx &= \lambda \bar{x}^T x \\ \bar{x}^T Qx &= \bar{\lambda}\bar{x}^T x \end{aligned} \tag{83}$$

Since the left hand sides are the same we have $\lambda \bar{x}^T x = \bar{\lambda}\bar{x}^T x$, and since $\bar{x}^T x \neq 0$, we conclude $\lambda = \bar{\lambda}$. This means $a + jb = a - jb$ which proves that $b = 0$.

Now that we know the eigenvalues are real we can conclude the eigenvectors are also real, because they are obtained from the equation $(Q - \lambda I)x = 0$ where $Q - \lambda I$ is real. □

The eigenvectors can be chosen to be orthonormal.

We will prove this for the case where the eigenvalues are distinct although the statement is true also without this restriction[6]. Orthonormality of the eigenvectors means they are orthogonal and each has unit length. Since we can easily normalize the length to one, we need only to show that the eigenvectors are orthogonal.

[6] A further fact is that a symmetric matrix admits a complete set of eigenvectors even in the case of repeated eigenvalues and is thus diagonalizable.

Pick two eigenvalue-eigenvector pairs: $Qx_1 = \lambda_1 x_1$, $Qx_2 = \lambda_2 x_2$, $\lambda_1 \neq \lambda_2$. Multiply $Qx_1 = \lambda_1 x_1$ from the left by x_2^T, and $Qx_2 = \lambda_2 x_2$ by x_1^T:

$$\begin{aligned} x_2^T Qx_1 &= \lambda_1 x_2^T x_1 \\ x_1^T Qx_2 &= \lambda_2 x_1^T x_2. \end{aligned} \tag{84}$$

Note that $x_2^T Qx_1$ is a scalar, therefore its transpose is equal to itself: $x_1^T Qx_2 = x_2^T Qx_1$. This means that the left hand sides of the two equations above are identical, hence

$$\lambda_1 x_2^T x_1 = \lambda_2 x_1^T x_2.$$

Note that $x_1^T x_2 = x_2^T x_1$ is the inner product of x_1 and x_2. Since $\lambda_1 \neq \lambda_2$, the equality above implies that this inner product is zero, that is x_1 and x_2 are orthogonal. □

The final property below proves our earlier assertion that AA^T and $A^T A$ have nonnegative eigenvalues. (Substitute $R = A^T$ below for the former, and $R = A$ for the latter.)

If Q can be written as $Q = R^T R$ for some matrix R, then the eigenvalues of Q are nonnegative.

To show this let x_i be an eigenvector of Q corresponding λ_i, so that

$$R^T R x_i = \lambda_i x_i.$$

Next multiply both sides from the left by x_i^T:

$$x_i^T R^T R x_i = \lambda_i x_i^T x_i = \lambda_i \|x_i\|^2.$$

If we define $y = Rx_i$ we see that the left hand side is $y^T y = \|y\|^2$, which is nonnegative. Thus, $\lambda_i \|x_i\|^2 \geq 0$. Since the eigenvector is nonzero, we have $\|x_i\| \neq 0$ which implies $\lambda_i \geq 0$. □

Principal Component Analysis (PCA)

PCA is an application of SVD in statistics that aims to find the most informative directions in a data set.

Suppose the $m \times n$ matrix A contains n measurements from m samples, for example n test scores for m students. If we subtract from each measurement the average over all samples, then each column of A is an m-vector with zero mean, and the $n \times n$ matrix

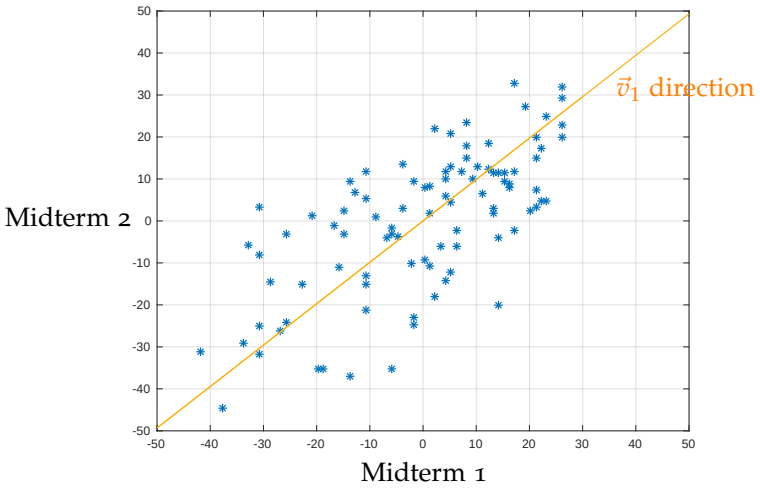
$$\frac{1}{m-1} A^T A$$

constitutes what is called the “covariance matrix" in statistics. Recall that the eigenvalues of this matrix are the singular values of A except for the scaling factor $m - 1$, and its orthonormal eigenvectors correspond to $\vec{v}_1, \ldots, \vec{v}_n$ in the SVD of A.

The vectors $\vec{v}_1, \vec{v}_2, \ldots$ corresponding to large singular values are called principal components and identify dominant directions in the data set along which the samples are clustered. The most significant direction is $\vec{v}_1$ corresponding to σ_1.

As an illustration, the scatter plot below shows $n = 2$ midterm scores in a class of $m = 94$ students that I taught in the past. The data points are centered around zero because the class average is subtracted from

the test scores. Each data point corresponds to a student and those in the first quadrant (both midterms ≥ 0) are those students who scored above average in each midterm. You can see that there were students who scored below average in the first and above average in the second, and vice versa.



For this data set the covariance matrix is:

$$\frac{1}{93}A^TA = \begin{bmatrix} 297.69 & 202.53 \\ 202.53 & 292.07 \end{bmatrix}$$

where the diagonal entries correspond to the squares of the standard deviations 17.25 and 17.09 for Midterms 1 and 2, respectively. The positive sign of the $(1,2)$ entry implies a positive correlation between the two midterm scores as one would expect.

The singular values of A, obtained from the square roots of the eigenvalues of A^TA, are $\sigma_1 = 215.08$, $\sigma_2 = 92.66$, and the corresponding eigenvectors of A^TA are:

$$\vec{v}_1 = \begin{bmatrix} 0.7120 \\ 0.7022 \end{bmatrix} \quad \vec{v}_2 = \begin{bmatrix} -0.7022 \\ 0.7120 \end{bmatrix}.$$

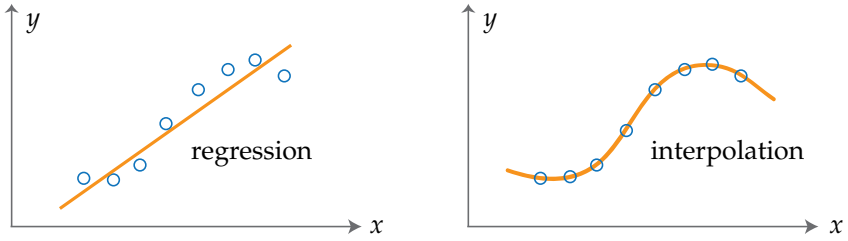
The principal component $\vec{v}_1$ is superimposed on the scatter plot and we see that the data is indeed clustered around this line. Note that it makes an angle of $\tan^{-1}(0.7022/0.7120) \approx 44.6^\circ$ which is skewed slightly towards the Midterm 1 axis because the standard deviation in Midterm 1 was slightly higher than in Midterm 2. We may interpret the points above this line as students who performed better in Midterm 2 than in Midterm 1, as measured by their scores relative to the class average that are then compared against the factor $\tan(44.6^\circ)$ to account for the difference in standard deviations.

The $\vec{v}_2$ direction, which is perpendicular to $\vec{v}_1$, exhibits less variation than the $\vec{v}_1$ direction ($\sigma_2 = 92.66$ vs. $\sigma_1 = 215.08$), but enough to convince you that you can do better on the final!

Sampling and Interpolation

Regression vs. Interpolation

Given data points (x_i, y_i), $i = 1, \cdots, n$, *interpolation* is the task of finding a function that exactly matches the data points as shown in the figure below (right). This differs from *regression* whose aim is to choose an approximate fit to data from among a class of functions as in the figure (left). The *least squares* method you studied in 16A is a commonly used form of regression.



Regression is meaningful when the data are inaccurate; for example when we have noisy measurements that cluster around a line rather than lying exactly on a line.

Interpolation is preferable when the data are accurate and we believe the variation is the result of the core phenomenon that the data represents rather than noise. For example, when zooming in on a digital image, an algorithm interpolates between existing pixels to fill in the pixels between them.

Interpolation is performed using a family of functions, such as polynomials, to predict what happens between the data points.

Polynomial Interpolation

If we have two data points (x_1, y_1), (x_2, y_2) we can find a line $y = a_0 + a_1 x$ that exactly matches these points, that is

$$a_0 + a_1 x_1 = y_1, \quad a_0 + a_1 x_2 = y_2.$$

To find a_0 and a_1, rewrite the two equations above in matrix form as

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{85}$$

and note that the matrix

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix}$$

is invertible when $x_1 \neq x_2$.

For three data points (x_1, y_1), (x_2, y_2), (x_3, y_3) we can find a quadratic polynomial $y = a_0 + a_1 x + a_2 x^2$ by solving for a_0, a_1, a_2 from

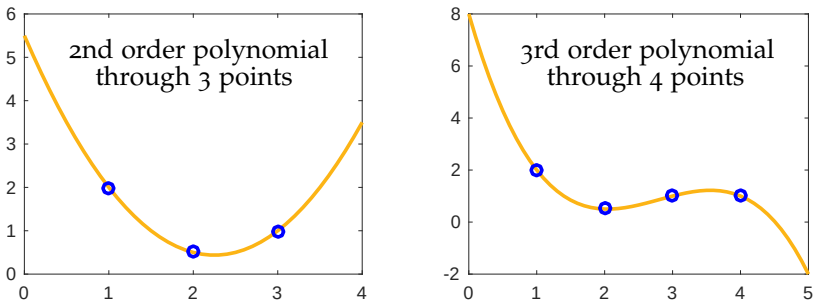
$$\begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}. \tag{86}$$

Similarly, for four data points (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), we look for a cubic polynomial $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ and solve for a_0, a_1, a_2, a_3 from

$$\begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ 1 & x_4 & x_4^2 & x_4^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}. \tag{87}$$

We will see below that the matrices in the two equations above are invertible if $x_i \neq x_j$ when $i \neq j$, that is if the x_i values are distinct.

The results of this procedure are illustrated in the figure below for three data points (left) and four data points (right).



Generalizing the arguments above to n data points we reach the following conclusion:

> Given (x_i, y_i), $i = 1, \cdots, n$, where $x_i \neq x_j$ when $i \neq j$, there is a unique polynomial of order $(n-1)$ passing through these points:
> $$y = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}.$$

We solve for the n coefficients $a_0, a_1, \ldots, a_{n-1}$ from

$$\underbrace{\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ 1 & x_3 & x_3^2 & \cdots & x_3^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix}}_{V} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \end{bmatrix}. \tag{88}$$

The matrix V is known as a Vandermonde matrix in linear algebra and its determinant is given by

$$\det(V) = \prod_{1 \leq i < j \leq n} (x_j - x_i), \tag{89}$$

which is $(x_2 - x_1)$ for $n = 2$, $(x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$ for $n = 3$, and so on[7]. Since we assumed $x_i \neq x_j$ when $i \neq j$, we conclude that the determinant (89) is nonzero and, thus, V is invertible.

[7] Verify this formula for $n = 2$ and $n = 3$ by calculating the determinant of the matrices in (85) and (86).

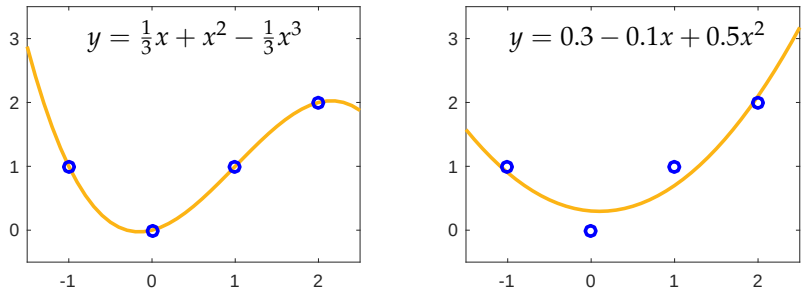
Example 1: To find a cubic polynomial $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ passing through the four points $(-1, 1), (0, 0), (1, 1), (2, 2)$, set up the equation (87):

$$\underbrace{\begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \end{bmatrix}}_{V} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \end{bmatrix}.$$

By inverting the matrix V we obtain

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \underbrace{\frac{1}{12} \begin{bmatrix} 0 & 12 & 0 & 0 \\ -4 & -6 & 12 & -2 \\ 6 & -12 & 6 & 0 \\ -2 & 6 & -6 & 2 \end{bmatrix}}_{V^{-1}} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1/3 \\ 1 \\ -1/3 \end{bmatrix}.$$

Thus interpolation gives the polynomial $y = \frac{1}{3}x + x^2 - \frac{1}{3}x^3$, as shown in the figure below (left).



As a comparison we next perform a quadratic *regression*, that is look for a quadratic polynomial $y = a_0 + a_1 x + a_2 x^2$ that passes close to the points $(-1,1), (0,0), (1,1), (2,2)$. If we attempt to exactly match these points we get the overdetermined equation

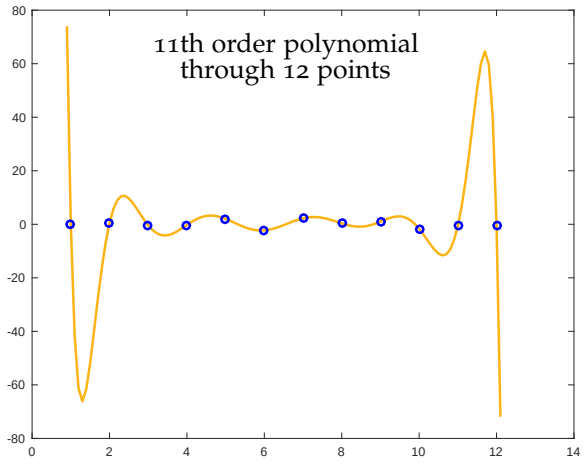
$$\underbrace{\begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix}}_{S} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \end{bmatrix}$$

which has no solution. Instead, we find the least squares solution that minimizes the magnitude of the error between the left- and right-hand sides of the equation above:

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = (S^T S)^{-1} S^T \begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0.3 \\ -0.1 \\ 0.5 \end{bmatrix}.$$

Thus quadratic regression gives the polynomial $y = 0.3 - 0.1x + 0.5x^2$, depicted in the figure above (right). □

A shortcoming of polynomial interpolation is that, as n gets larger, the polynomial has terms with large powers (up to $n-1$) that grow very fast. This makes the interpolation erratic and unreliable, especially near the end points. The figure below illustrates this behavior with a polynomial interpolation through $n = 12$ points.



Interpolation with Basis Functions

An alternative method is to assign to each x_i a function $\phi_i(\cdot)$ that satisfies:

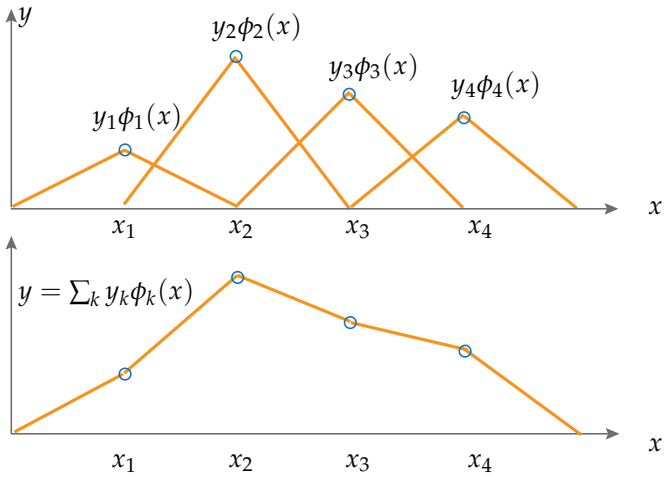
$$\phi_i(x_i) = 1 \quad \text{and} \quad \phi_i(x_j) = 0 \text{ when } j \neq i \tag{90}$$

and to interpolate between the data points (x_i, y_i) with the function:

$$y = \sum_k y_k \phi_k(x). \tag{91}$$

When $x = x_i$ this expression yields $y = y_i$ as desired, because $\phi_k(x_i) = 0$ except when $k = i$.

We refer to $\phi_i(x)$ as "basis functions" since our interpolation consists of a linear combination of these functions. Basis functions restrict the behavior of the interpolation between data points and avoid the erratic results of polynomial interpolation. The figure below uses triangular basis functions which lead to a linear interpolation.



For simplicity we henceforth assume x_i are in increasing order and evenly spaced:

$$x_{i+1} - x_i = \Delta \quad \text{for all } i.$$

This allows us to obtain basis functions $\phi_i(x)$ by shifting a single function $\phi(x)$:

$$\phi_i(x) = \phi(x - x_i). \tag{92}$$

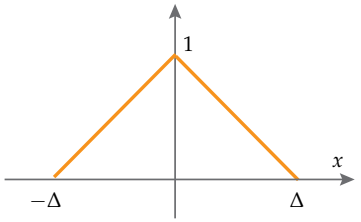
Note that (90) holds if we define $\phi(x)$ such that

$$\phi(0) = 1 \quad \text{and} \quad \phi(k\Delta) = 0 \text{ when } k \neq 0. \tag{93}$$

Linear Interpolation

When $\phi(x)$ is as depicted on the right, that is

$$\phi(x) = \begin{cases} 1 - \frac{|x|}{\Delta} & |x| \leq \Delta \\ 0 & \text{otherwise,} \end{cases}$$

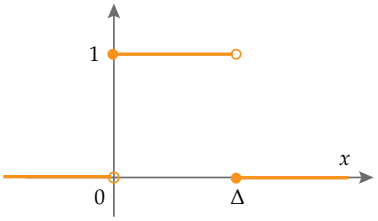


and the basis functions are obtained from (92), then the interpolation (91) connects the data points with straight lines. (See figure above.)

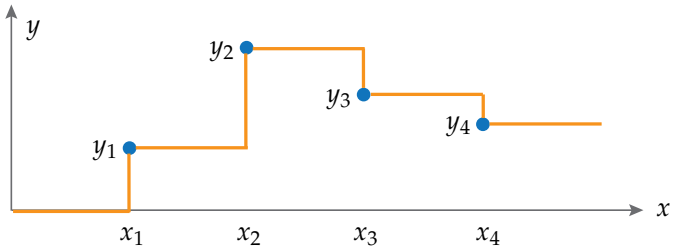
Zero Order Hold Interpolation

When

$$\phi(x) = \begin{cases} 1 & x \in [0, \Delta) \\ 0 & \text{otherwise} \end{cases}$$



as depicted on the right, the interpolation (91) keeps y constant between the data points:

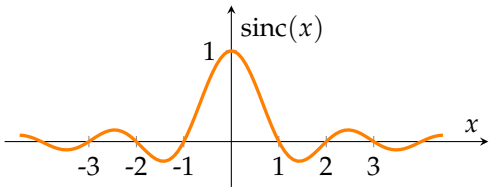


Sinc Interpolation

The *sinc* function is defined as

$$\operatorname{sinc}(x) \triangleq \begin{cases} \frac{\sin(\pi x)}{\pi x} & x \neq 0 \\ 1 & x = 0 \end{cases}$$

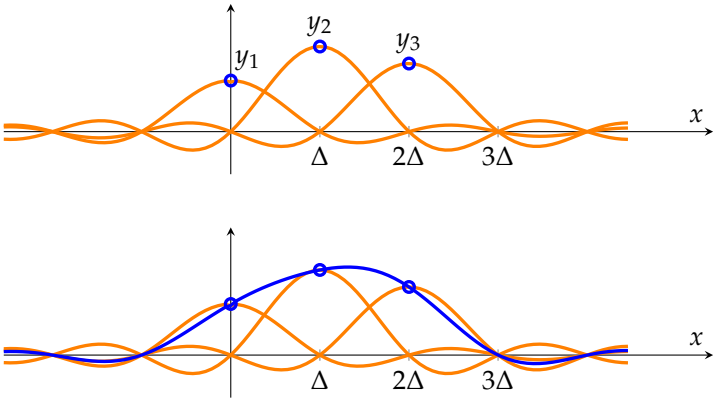
and depicted below. It is continuous since $\lim_{x \to 0} \frac{\sin(\pi x)}{\pi x} = 1$, and vanishes whenever x is a nonzero integer.



In sinc interpolation we apply (91) with

$$\phi(x) = \text{sinc}(x/\Delta).$$

To illustrate this, the first plot below superimposes $y_i\phi(x - x_i)$ for three data points $i = 1, 2, 3$. The second plot adds them up (blue curve) to complete the interpolation.



The interest in sinc interpolation is due to its smoothness – contrast the blue curve above with the kinks of linear interpolation and the discontinuities of zero order hold interpolation.

To make this smoothness property more explicit we use the identity

$$\text{sinc}(x) = \frac{1}{\pi}\int_0^\pi \cos(\omega x)d\omega \tag{94}$$

which you can verify by evaluating the integral. Viewing this integral as an infinite sum of cosine functions, we see that the fastest varying component has frequency $\omega = \pi$. Thus the sinc function can't exhibit variations faster than this component.

Functions that involve frequencies smaller than some constant are called "band-limited." This notion is made precise in EE 120 with continuous *Fourier Transforms*. For 16B it is sufficient to think of a band-limited signal as one that can be written as a sum or integral of sinusoidal components whose frequencies lie in a finite band, which is $[0, \pi]$ for the sinc function in (94).

Sampling Theorem

Sampling is the opposite of interpolation: given a function $f(\cdot)$ we evaluate it at sample points x_i:

$$y_i = f(x_i) \quad i = 1, 2, 3, \ldots$$

Sampling is critical in digital signal processing where one uses samples of an analog sound or image. For example, digital audio is often recorded at 44.1 kHz which means that the analog audio is sampled 44,100 times per second; these samples are then used to reconstruct the audio when playing it back. Similarly, in digital images each pixel corresponds to a sample of an analog image.

An important problem in sampling is whether we can perfectly recover an analog signal from its samples. As we explain below, the answer is yes when the analog signal is band-limited and the interval between the samples is sufficiently short.

Suppose we sample the function $f : \mathbb{R} \to \mathbb{R}$ at evenly spaced points

$$x_i = \Delta i, \quad i = 1, 2, 3, \ldots$$

and obtain

$$y_i = f(\Delta i) \quad i = 1, 2, 3, \ldots$$

Then sinc interpolation between these data points gives:

$$\hat{f}(x) = \sum_i y_i \phi(x - \Delta i) \tag{95}$$

where

$$\phi(x) = \operatorname{sinc}(x/\Delta),$$

which is band-limited by π/Δ from (94). This means that $\hat{f}(x)$ in (95) contains frequencies ranging from 0 to π/Δ.

Now if $f(x)$ involves frequencies smaller than π/Δ, then it is reasonable to expect that it can be recovered from (95) which varies as fast as $f(x)$. In fact the shifted sinc functions $\phi(x - \Delta i)$ form a basis for the space of functions[8] that are band-limited by π/Δ and the formula (95) is simply the representation of $f(x)$ with respect to this basis.

[8] for technical reasons this space is also restricted to square integrable functions

Sampling Theorem: If $f(x)$ is band-limited by frequency

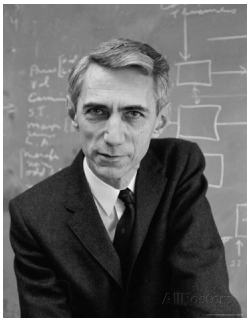
$$\omega_{\max} < \frac{\pi}{\Delta} \tag{96}$$

then the sinc interpolation (95) recovers $f(x)$, that is $\hat{f}(x) = f(x)$.

By defining the sampling frequency $\omega_s = 2\pi/\Delta$, we can restate the condition (96) as:

$$\omega_s > 2\omega_{\max}$$

which states that the function must be sampled faster than twice its maximum frequency. The Sampling Theorem was proven by Claude Shannon in the 1940s and was implicit in an earlier result by Harry Nyquist. Both were researchers at the Bell Labs.

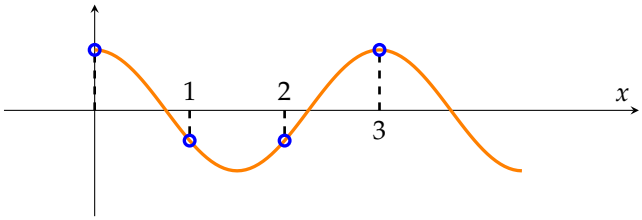

Claude Shannon (1916-2001)


Harry Nyquist (1889-1976)

Example 2: Suppose we sample the function

$$f(x) = \cos\left(\frac{2\pi}{3}x\right)$$

with period $\Delta = 1$. This means that we take 3 samples in each period of the cosine function, as shown in the figure below. Since $\omega_{\max} = \frac{2\pi}{3}$ and $\Delta = 1$, the criterion (96) holds and we conclude that the sinc interpolation (95) exactly recovers $f(x)$.



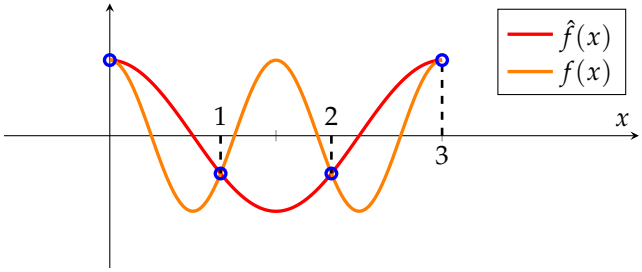
Example 3: Suppose now the function being sampled is

$$f(x) = \cos\left(\frac{4\pi}{3}x\right). \tag{97}$$

With $\omega_{\max} = \frac{4\pi}{3}$ and $\Delta = 1$, the criterion (96) fails. To see that the result of the sinc interpolation $\hat{f}(x)$ is now different from $f(x)$, note that this time we take 3 samples every *two* periods of the cosine function, as shown below. These samples are identical to the 3 samples collected in one period of the function in Example 2 above. Therefore, sinc interpolation gives the same result it did in Example 2:

$$\hat{f}(x) = \cos\left(\frac{2\pi}{3}x\right)$$

which does not match (97).



Aliasing and Phase Reversal

In Example 3 the low frequency component $2\pi/3$ appeared in $\hat{f}(x)$ from the actual frequency $4\pi/3$ of $f(x)$ that exceeded the critical value $\pi/\Delta = \pi$. The emergence of phantom lower frequency components as a result of under-sampling is known as "aliasing."

To generalize Example 3 suppose we sample the function

$$f(x) = \cos(\omega x + \phi) \tag{98}$$

with period $\Delta = 1$ and obtain

$$y_i = \cos(\omega i + \phi).$$

Using the identity $\cos(2\pi i - \theta) = \cos(\theta)$ which holds for any integer i, and substituting $\theta = \omega i + \phi$. we get

$$y_i = \cos(2\pi i - \omega i - \phi) = \cos((2\pi - \omega)i - \phi)$$

which suggests that the samples of the function

$$\cos((2\pi - \omega)x - \phi) \tag{99}$$

are identical to those of (98).

If $\omega \in (\pi, 2\pi]$ in (98) then sinc interpolation gives the function in (99) whose frequency is $2\pi - \omega \in [0, \pi)$. This function changes more slowly than (98) and the sign of the phase ϕ is reversed. These effects are visible in movies where a rotating wheel appears to rotate more slowly and in the opposite direction when its speed exceeds half of the sampling rate (18-24 frames/second).

Example 4: Suppose we sample the function

$$f(x) = \sin(1.9\pi x)$$

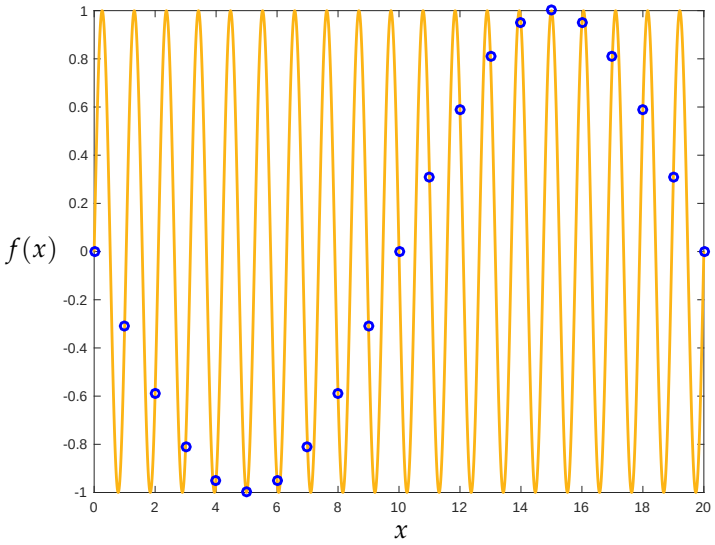
with $\Delta = 1$ as shown in the figure below. This function is of the form (98) with $\omega = 1.9\pi$ and $\phi = -\pi/2$ because

$$\sin(1.9\pi x) = \cos(1.9\pi x - \pi/2).$$

Thus, from (99), the sinc interpolation gives

$$\hat{f}(x) = \cos(0.1\pi x + \pi/2) = -\sin(0.1\pi x)$$

as evident from the samples in the figure. Note that the negative sign of $-\sin(0.1\pi x)$ is a result of the phase reversal discussed above.



Example 5: Note that the inequality in (96) is strict. To see that the Sampling Theorem does not hold when $\omega_{\max} = \frac{\pi}{\Delta}$ suppose $f(x) = \sin(\pi x)$ and $\Delta = 1$. Then the samples are

$$y_i = \sin(\pi i) = 0, \quad i = 1, 2, 3, \ldots$$

Since all samples are zero, sinc interpolation gives

$$\hat{f}(x) = 0$$

and does not recover $f(x) = \sin(\pi x)$.

Discrete-Time Control of Continuous-Time Systems

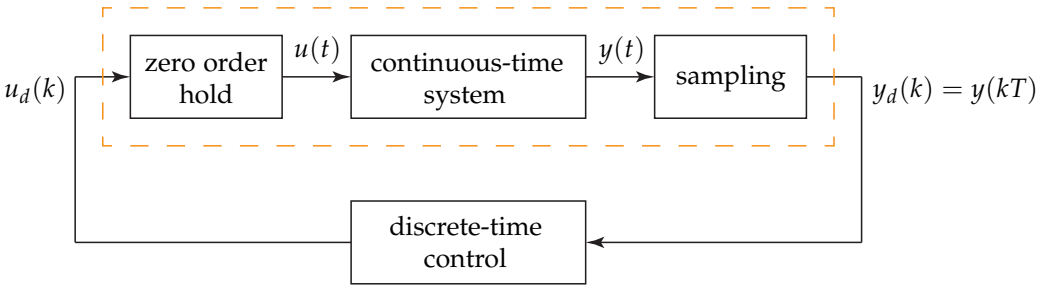
In a typical application the control algorithm for a continuous-time physical system is executed in discrete-time. Thus, the measured output must be sampled before being fed to the control algorithm. Conversely, the discrete-time control input generated by the algorithm must be interpolated into a continuous-time function, typically with a zero order hold, before being applied back to the system.

This scheme is depicted in the block diagram below. We let T denote the sampling period and represent the samples of the output $y(t)$ at $t = kT$, $k = 0, 1, 2, \ldots$ by

$$y_d(k) = y(kT)$$

where the subscript "d" stands for discrete-time. The control sequence generated in discrete time is denoted $u_d(k)$ and is interpolated by the zero order hold block to the continuous-time input

$$u(t) = u_d(k) \quad t \in [kT, (k+1)T). \tag{100}$$



To design a discrete-time control algorithm we need to represent the continuous-time system, combined with the zero order hold and sampling blocks, with a discrete-time model. This combination is depicted with the dashed box in the figure above, with input $u_d(k)$ and output $y_d(k)$.

Suppose the continuous-time system model is

$$\begin{aligned} \frac{d}{dt}\vec{x}(t) &= A\vec{x}(t) + Bu(t) \\ y(t) &= C\vec{x}(t) \end{aligned} \tag{101}$$

and we wish to obtain a discrete-time model

$$\begin{aligned} \vec{x}_d(k+1) &= A_d\vec{x}(k) + B_d u_d(k) \\ y_d(k) &= C_d\vec{x}_d(k) \end{aligned} \tag{102}$$

where $\vec{x}_d(k)$ is the value of the state $\vec{x}(t)$ at time $t = kT$. It follows that $C_d = C$ because

$$y_d(k) = y(kT) = C\vec{x}(kT) = C\vec{x}_d(k).$$

To see how A_d and B_d in (102) can be obtained let's first assume $A = a$ and $B = b$ are scalars. Recall that the solution of the differential equation (101) with initial condition $x(0)$ is

$$x(t) = e^{at}x(0) + b\int_0^t e^{a(t-s)}u(s)ds. \tag{103}$$

Since $u(s)$ is constant in the interval $s \in [0, T)$ and equal to $u_d(0)$, the solution at $t = T$ is

$$\underbrace{x(T)}_{x_d(1)} = e^{aT}\underbrace{x(0)}_{x_d(0)} + \frac{e^{aT}-1}{a}b\,u_d(0)$$

and a generalization of this argument shows

$$x_d(k+1) = e^{aT}x_d(k) + \frac{e^{aT}-1}{a}b\,u_d(k).$$

Thus we conclude that

$$A_d = e^{aT} \quad \text{and} \quad B_d = \frac{e^{aT}-1}{a} b.$$

Next assume A is not scalar but diagonal, and B is an arbitrary column vector:

$$A = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}.$$

Recall that any linear system where A is diagonalizable can be brought to this form with a change of state variables.

Then (101) decouples into scalar differential equations

$$\frac{d}{dt} x_i(t) = \lambda_i x_i(t) + b_i u(t) \quad i = 1, \ldots, n$$

and we conclude from the arguments above that

$$x_{d,i}(k+1) = e^{\lambda_i T} x_{d,i}(k) + \frac{e^{\lambda_i T}-1}{\lambda_i} b_i \, u_d(k).$$

Concatenating $x_{d,i}$, $i = 1, \ldots, n$, into the vector $\vec{x}_d$ we obtain the discrete-time model (102) with

$$A_d = \begin{bmatrix} e^{\lambda_1 T} & & \\ & \ddots & \\ & & e^{\lambda_n T} \end{bmatrix} \quad B = \begin{bmatrix} \frac{e^{\lambda_1 T}-1}{\lambda_1} b_1 \\ \vdots \\ \frac{e^{\lambda_n T}-1}{\lambda_n} b_n \end{bmatrix}.$$

A natural question to ask is whether stability of the continuous-time model implies stability of the discretized model. To see that the answer is yes note that an eigenvalue of A at

$$\lambda = v + j\omega$$

maps to an eigenvalue of A_d at

$$\lambda_d = e^{\lambda T} = e^{vT} e^{j\omega T}$$

whose magnitude is

$$|\lambda_d| = e^{vT}.$$

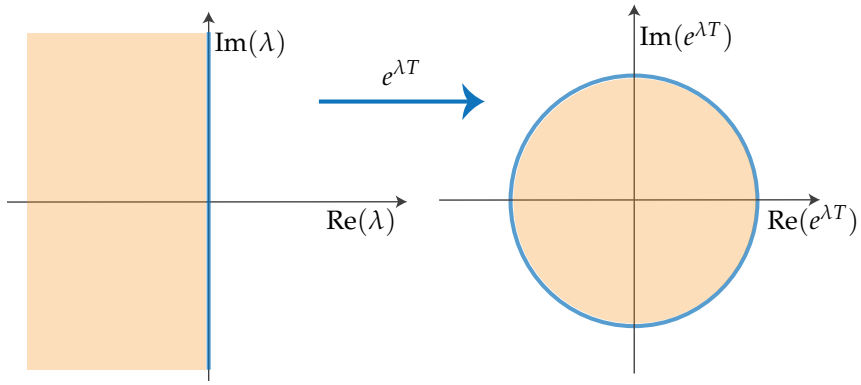
Thus, if λ satisfies the continuous-time stability criterion

$$\mathrm{Re}(\lambda) = v < 0$$

then λ_d satisfies the discrete-time stability criterion

$$|\lambda_d| = e^{vT} < 1.$$

Indeed, as depicted in the figure below, $\lambda \mapsto e^{\lambda T}$ maps the imaginary axis in the complex plane to the unit circle, and the left hand side of the imaginary axis to the interior of the unit circle.



Although stability is maintained, controllability and observability properties may not be preserved when we control and observe a system in discrete time. You will investigate the potential loss of observability in a homework problem.

Discrete Fourier Transform (DFT)

Discrete-Time Signals as Vectors

Consider a discrete-time signal $x[n]$ that consists of N samples, $n = 0, 1, \ldots, N-1$. To prepare for the Discrete Fourier Transform to be introduced later we interpret this signal as a vector,

$$\vec{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}.$$

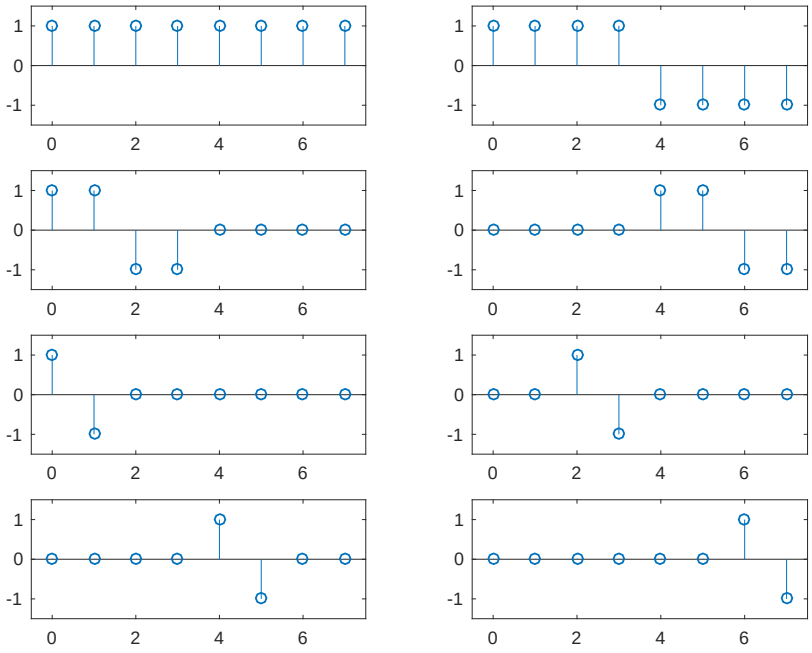
This viewpoint allows us to define the inner product of two signals $x[n]$ and $y[n]$ as

$$\vec{x}^T \vec{y} = \sum_{n=0}^{N-1} x[n]y[n]$$

and to declare two signals to be orthogonal when this inner product is zero. For example, you should be able to verify by inspection that the eight signals depicted below are orthogonal to each other.

Interpreting discrete time signals as vectors allows us to introduce new "basis signals" such as those depicted below, which provide an orthogonal basis for length-8 sequences. Special basis functions are commonly used in compressing images, videos, soundtracks, etc.

As an illustration consider a 10-second soundtrack recorded at 44.1 kHz, which contains 444,100 samples. Instead of viewing this signal as a linear combination of the standard basis signals $\{1, 0, 0 \ldots\}$, $\{0, 1, 0, \ldots\}$, $\{0, 0, 1, \cdots\}$, $\ldots$, each multiplied by the value of the corresponding sample, we can choose a different basis where some of the signals are less important for the sound quality than others, *e.g.* high frequency sinusoidal signals that are inaudible. This allows one to reduce the file size by either ignoring the coefficients of the insignificant basis signals or by quantizing them with fewer bits.



Example 1: The "Discrete Cosine Transform" (DCT) has been employed in standards for image and video compression, such as JPEG[9] and MPEG[10]. One type of DCT, called DCT-2, represents a signal as a linear combination of the basis signals

[9] Joint Photographic Experts Group

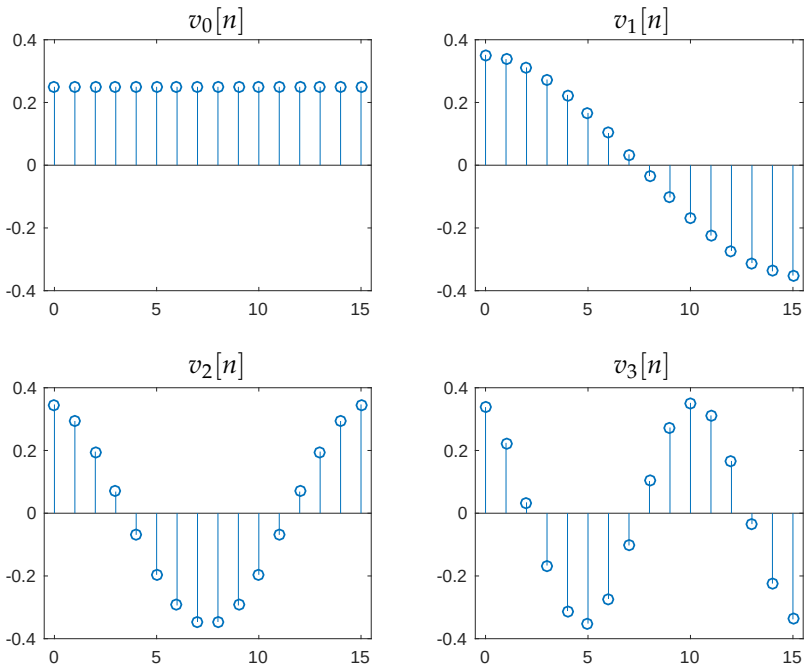
[10] Moving Picture Experts Group

$$v_0[n] = \sqrt{\frac{1}{N}}, \quad v_k[n] = \sqrt{\frac{2}{N}} \cos\left(\frac{\pi k}{2N}(2n+1)\right) \quad k = 1, \cdots, N-1,$$

where larger k indicates a higher frequency. The first four of these basis signals ($k = 0, 1, 2, 3$) are depicted below.

The following identity, whose proof is omitted, shows that the basis signals defined above are orthonormal:

$$\sum_{n=0}^{N-1} \cos\left(\frac{\pi k}{2N}(2n+1)\right) \cos\left(\frac{\pi m}{2N}(2n+1)\right) = \begin{cases} N & \text{if } k = m = 0 \\ \frac{N}{2} & \text{if } k = m \neq 0 \\ 0 & \text{if } k \neq m. \end{cases}$$



Given orthonormal basis vectors $\vec{v}_0, \ldots, \vec{v}_{N-1}$ we can express any vector $\vec{x}$ as a linear combination

$$\vec{x} = \alpha_0 \vec{v}_0 + \cdots + \alpha_{N-1} \vec{v}_{N-1} \tag{104}$$

and obtain the coefficients from the inner product

$$\alpha_k = \vec{v}_k^T \vec{x}.$$

This follows from orthonormality: if we multiply both sides of (104) by $\vec{v}_k^T$ all terms in the right hand side vanish, except $\alpha_k \vec{v}_k^T \vec{v}_k = \alpha_k$.

Similarly, given N orthonormal basis signals $v_0[n], \ldots, v_{N-1}[n]$ we can express any signal $x[n]$ as a linear combination

$$x[n] = \alpha_0 v_0[n] + \cdots + \alpha_{N-1} v_{N-1}[n] \qquad n = 0, 1, \ldots, N-1$$

and obtain the coefficients α_k from the inner product

$$\alpha_k = \sum_{n=0}^{N-1} v_k[n] x[n].$$

We can then store the coefficients $\alpha_0, \ldots, \alpha_{N-1}$ instead of the signal values $x[0], \ldots, x[N-1]$. This is the premise of compression algorithms that store only the coefficients corresponding to basis functions that are important for the quality of sound, image, etc.

The Discrete Fourier Transform (DFT) is similar in spirit to the Discrete Cosine Transform but, as we will see, it uses complex valued basis signals. Therefore, before proceeding to DFT, we adapt the definition of inner products to complex vectors.

Complex Inner Products

For complex valued vectors $\vec{x}$ and $\vec{y}$ the appropriate inner product is

$$\vec{x}^* \vec{y}$$

where $\vec{x}^*$ is the *conjugate transpose* which means that, in addition to transposing, we take the complex conjugate. As an illustration,

$$\vec{x} = \begin{bmatrix} 1 \\ j \end{bmatrix} \quad \Rightarrow \quad \vec{x}^T = \begin{bmatrix} 1 & j \end{bmatrix} \quad \vec{x}^* = \begin{bmatrix} 1 & -j \end{bmatrix}. \tag{105}$$

For a real-valued $\vec{x}$ there is no difference between $\vec{x}^*$ and $\vec{x}^T$, as the complex conjugate of a real number is itself. Note that

$$\vec{x}^* \vec{x} = \|\vec{x}\|^2$$

which follows because $\vec{x}^* \vec{x} = \sum_n x[n]^* x[n] = \sum_n |x[n]|^2$. For the example in (105), $\vec{x}^* \vec{x} = 1 - j^2 = 2$ which means $\|\vec{x}\| = \sqrt{2}$. By contrast, $\vec{x}^T \vec{x} = 1 + j^2 = 0$ which shows the necessity of conjugation when defining complex inner products.

Now if $\vec{v}_0, \ldots, \vec{v}_{N-1}$ are complex valued and orthonormal, that is

$$\vec{v}_k^* \vec{v}_m = \begin{cases} 1 & \text{if } k = m \\ 0 & \text{if } k \neq m, \end{cases}$$

then the coefficients in the decomposition (104) become

$$\alpha_k = \vec{v}_k^* \vec{x}. \tag{106}$$

Discrete Fourier Transform (DFT)

The DFT uses the basis signals

$$\boxed{u_k[n] \triangleq \frac{1}{\sqrt{N}} e^{jk\omega n}, \quad k = 0, 1, \ldots, N-1 \quad \text{where} \quad \omega = \frac{2\pi}{N}} \tag{107}$$

which we can rewrite as[11]

$$u_k[n] = \frac{1}{\sqrt{N}} \cos(k\omega n) + j \frac{1}{\sqrt{N}} \sin(k\omega n).$$

The DFT basis is similar to DCT in that it consists of sinusoids of varying frequencies, but differs due to its complex values. The interest in DFT is because of computational efficiency[12] and, as we will see later, because it facilitates the analysis of linear time-invariant systems.

[11] using the identity $e^{j\theta} = \cos\theta + j \sin\theta$

[12] A class of algorithms known as Fast Fourier Transforms has been developed to perform the DFT.

Finding the DFT of $x[n]$ means finding coefficients $X[k]$, $k = 0, 1, \ldots, N-1$, to represent $x[n]$ as a linear combination of $u_k[n]$:

$$\boxed{x[n] = \sum_{k=0}^{N-1} X[k] u_k[n].} \tag{108}$$

Example 2 ($N = 2$): Consider the signal

$$x[0] = 2, \quad x[1] = 3.$$

Since the duration is $N = 2$ we have $\omega = \pi$,

$$u_0[n] = \frac{1}{\sqrt{2}} e^{j0n} = \frac{1}{\sqrt{2}} \quad \text{and} \quad u_1[n] = \frac{1}{\sqrt{2}} e^{j\pi n} = \frac{1}{\sqrt{2}} (-1)^n.$$

We view $x[n]$, $u_0[n]$, $u_1[n]$ as length-two vectors whose entries are the values that each sequence takes at $n = 0, 1$:

$$\vec{x} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad \vec{u}_0 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \vec{u}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Then DFT becomes a change of basis

$$\vec{x} = X[0]\vec{u}_0 + X[1]\vec{u}_1$$

similar to (104). Since $\vec{u}_0$ and $\vec{u}_1$ are real and orthonormal, that is $\vec{u}_0^T \vec{u}_1 = 0$ and $\vec{u}_0^T \vec{u}_0 = \vec{u}_1^T \vec{u}_1 = 1$, we get

$$X[0] = \vec{u}_0^T \vec{x} = \frac{5}{\sqrt{2}} \qquad X[1] = \vec{u}_1^T \vec{x} = -\frac{1}{\sqrt{2}}.$$

Example 3 ($N = 4$): When the duration is $N = 4$ we have $\omega = \frac{2\pi}{N} = \frac{\pi}{2}$:

$$\begin{aligned} u_0[n] &= \frac{1}{2} e^{j0n} = \frac{1}{2} \\ u_1[n] &= \frac{1}{2} e^{j\frac{\pi}{2}n} = \frac{1}{2} (j)^n \\ u_2[n] &= \frac{1}{2} e^{j2\frac{\pi}{2}n} = \frac{1}{2} (-1)^n \\ u_3[n] &= \frac{1}{2} e^{j3\frac{\pi}{2}n} = \frac{1}{2} (-j)^n. \end{aligned}$$

We view $u_0[n], \ldots, u_3[n]$ as length-four vectors whose entries are the values that each sequence takes at $n = 0, 1, 2, 3$:

$$\vec{u}_0 = \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \vec{u}_1 = \frac{1}{2} \begin{bmatrix} 1 \\ j \\ -1 \\ -j \end{bmatrix} \quad \vec{u}_2 = \frac{1}{2} \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \quad \vec{u}_3 = \frac{1}{2} \begin{bmatrix} 1 \\ -j \\ -1 \\ j \end{bmatrix}.$$

To find $X[k]$, $k = 0, 1, 2, 3$, such that

$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix} = X[0]\vec{u}_0 + X[1]\vec{u}_1 + X[2]\vec{u}_2 + X[3]\vec{u}_3$$

we will again use the orthonormality of the basis vectors $\vec{u}_0, \ldots, \vec{u}_3$. You can indeed show that

$$\vec{u}_k^* \vec{u}_m = \begin{cases} 0 & k \neq m \\ 1 & k = m. \end{cases}$$

Using orthonormality as in (106) we can find the coefficients $X[k]$ from

$$X[k] = \vec{u}_k^* \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

and combine this equation for $k = 0, 1, 2, 3$ into

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

where the first row of is $\vec{u}_0^*$, the second row is $\vec{u}_1^*$, and so on.

As an illustration, for the sequence $x[0] = 1$, $x[1] = x[2] = x[3] = 0$, we find

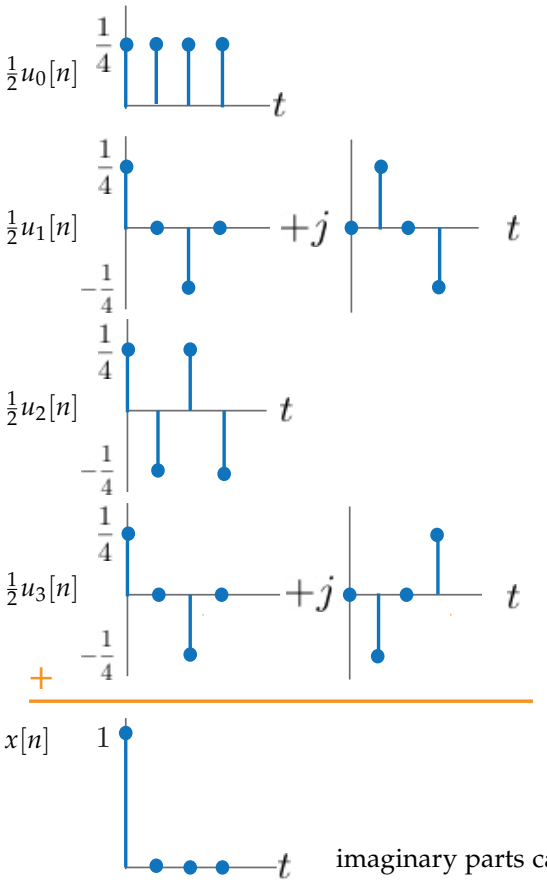
$$X[0] = X[1] = X[2] = X[3] = \frac{1}{2}.$$

The summation of $u_0[n], \ldots, u_3[n]$ with these weights indeed recovers $x[n]$ as shown in the figure below.

Orthonormality of the DFT Basis

In both examples ($N = 2$ and $N = 4$) we made use of the orthonormality of the DFT basis. We now show that this is a general property that holds for any N. To simplify the notation we define

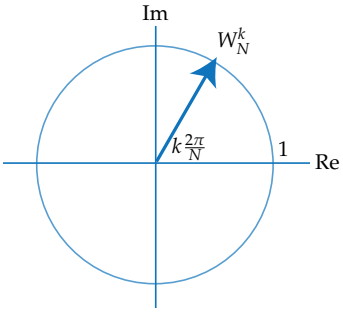
$$W_N \triangleq e^{j\frac{2\pi}{N}} \tag{109}$$



imaginary parts cancel out

and rewrite (107) as

$$u_k[n] \triangleq \frac{1}{\sqrt{N}} W_N^{kn}, \quad k = 0, 1, \ldots, N-1,$$



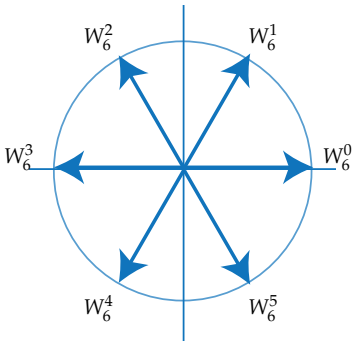
or as the vector

$$\vec{u}_k = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 \\ W_N^k \\ W_N^{2k} \\ \vdots \\ W_N^{(N-1)k} \end{bmatrix}.$$

To see that these vectors form an orthonormal basis, note that

$$\vec{u}_k^* \vec{u}_m = \frac{1}{N} \sum_{n=0}^{N-1} (W_N^{kn})^* W_N^{mn} = \frac{1}{N} \sum_{n=0}^{N-1} e^{j(m-k)\frac{2\pi}{N}n} = \frac{1}{N} \sum_{n=0}^{N-1} W_N^{(m-k)n}$$

where the second equality follows from $(W_N^{kn})^* = (e^{j\frac{2\pi}{N}kn})^* = e^{-j\frac{2\pi}{N}kn}$. Now, if $m = k$, $e^{j\frac{2\pi}{N}(m-k)n} = 1$ and the summation gives N. If $m \neq k$ the summation gives zero because the numbers $W_N^{(m-k)n}$, $n = 0, \ldots, N-1$ are spread evenly around the unit circle and add up to zero, as illustrated on the right for $m - k = 1$ and $N = 6$. Thus,



$$\vec{u}_k^* \vec{u}_m = \begin{cases} 0 & k \neq m \\ 1 & k = m. \end{cases} \tag{110}$$

DFT Analysis and Synthesis Equations

Recall that the DFT representation of the sequence $x[n]$ is

$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix} = X[0]\vec{u}_0 + X[1]\vec{u}_1 + \cdots + X[N-1]\vec{u}_{N-1}. \tag{111}$$

To determine $X[k]$ we multiply both sides of this equation by $\vec{u}_k^*$ and use the orthonormality property (110) to obtain:

$$\vec{u}_k^* \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix} = X[k]\vec{u}_k^*\vec{u}_k = X[k].$$

Thus, stacking up $X[0], \ldots, X[N-1]$ in a vector gives the formula:

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix} = \begin{bmatrix} \vec{u}_0^* \\ \vec{u}_1^* \\ \vec{u}_2^* \\ \vdots \\ \vec{u}_{N-1}^* \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix} \tag{112}$$

which we call the "analysis equation" since it amounts to analyzing the frequency content of the signal $x[n]$.

Likewise, we rewrite the equation (111) compactly as:

$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix} = \underbrace{\begin{bmatrix} \vec{u}_0 & \vec{u}_1 & \vec{u}_2 & \cdots & \vec{u}_{N-1} \end{bmatrix}}_{\triangleq F} \begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix}$$

and call it the "synthesis equation" since it synthesizes $x[t]$ from the basis signals.

We refer to the $N \times N$ matrix F as the Fourier Matrix. Note that the matrix in the analysis equation (112) corresponds to F^*, because its rows are conjugate transposes of the columns of F. Since the analysis equation $\vec{X} = F^* \vec{x}$ and synthesis equation $\vec{x} = F\vec{X}$ are inverses of each other, we conclude

$$F^{-1} = F^*$$

which means that F is a *unitary* matrix:

$$F^*F = FF^* = I.$$

Example 4: Recall from Example 3 that, when $N = 4$, we have

$$\vec{u}_0 = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \vec{u}_1 = \frac{1}{2}\begin{bmatrix} 1 \\ j \\ -1 \\ -j \end{bmatrix} \quad \vec{u}_2 = \frac{1}{2}\begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \quad \vec{u}_3 = \frac{1}{2}\begin{bmatrix} 1 \\ -j \\ -1 \\ j \end{bmatrix}.$$

Thus, for $N = 4$,

$$F = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \quad F^* = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}. \tag{113}$$

You can verify that $F^*F = FF^* = I$.

Some Properties of DFT

The first two properties below follow directly from (112).

1) Scaling:

If $x[n]$ has DFT coefficients $X[k]$ then, for any constant a, the sequence $ax[n]$ has DFT coefficients $aX[k]$.

2) Superposition:

If $x[n]$ and $y[n]$ have DFT coefficients $X[k]$ and $Y[k]$ respectively, then $x[n]+y[n]$ has DFT coefficients $X[k]+Y[k]$.

3) Parseval's relation: Given $x[n]$ and its DFT coefficients $X[k]$,

$$\sum_{n=0}^{N-1}|x[n]|^2=\sum_{k=0}^{N-1}|X[k]|^2$$

which means $\|\vec{x}\|^2=\|\vec{X}\|^2$. This follows because $\vec{x}=F\vec{X}$ where F is unitary ($F^*F=I$) as discussed above and, thus,

$$\|\vec{x}\|^2=\vec{x}^*\vec{x}=\vec{X}^*F^*F\vec{X}=\vec{X}^*\vec{X}=\|\vec{X}\|^2.$$

4) Conjugate symmetry:

When $x[n]$ is real-valued, the DFT coefficients satisfy

$$\boxed{X[N-k]=X[k]^* \qquad k=1,\dots,N-1.} \tag{114}$$

When $N=4$, this implies $X[3]=X[1]^*$ and $X[2]=X[2]^*$ which means $X[2]$ is real. You can indeed see from (112) and (113) that, when $x[n]$ is real, $X[2]=\frac{1}{2}\left(x[0]-x[1]+x[2]-x[3]\right)$ is real and

$$\begin{aligned} X(1) &= \frac{1}{2}\left(x[0]-x[2]\right)-j\frac{1}{2}\left(x[1]-x[3]\right)\\ X(3) &= \frac{1}{2}\left(x[0]-x[2]\right)+j\frac{1}{2}\left(x[1]-x[3]\right)\end{aligned}$$

are complex conjugates.

To see why (114) holds in general first note from (107) that

$$W_N^{N-k}=e^{j\frac{2\pi}{N}(N-k)}=e^{j2\pi}e^{-j\frac{2\pi}{N}k}=e^{-j\frac{2\pi}{N}k}={W_N^k}^*$$

and, similarly, $W_N^{(N-k)n}=(W_Nkn)^*$, $n=2,3,\dots$ It then follows that

$$X[k]=\vec{u}_k^*\vec{x}=\sum_{n=0}^{N-1}(W_N^{kn})^*x[n]=\sum_{n=0}^{N-1}(W_N^{(N-k)n})x[n].$$

Since $x[n]$ is real, we have $x[n]^*=x[n]$; thus taking the conjugate of both sides gives

$$X[k]^*=\sum_{n=0}^{N-1}(W_N^{(N-k)n})^*x[n]=\vec{u}_{N-k}^*\vec{x}=X[N-k].$$

Example 5: Let $x[n] = e^{j\omega_0 n}$, $n = 0, 1, \ldots, N-1$. If

$$\omega_0 = k_0 \frac{2\pi}{N} \quad \text{for some integer } k_0 \in \{0, 1, \ldots, N-1\} \tag{115}$$

then $x[n]$ is identical to the k_0-th basis function, except for the normalization factor $\sqrt{N}$:

$$x[n] = \sqrt{N} u_{k_0}[n].$$

This means that $x[n]$ is recovered from the k_0-th basis function alone:

$$X[k] = \begin{cases} \sqrt{N} & \text{if } k = k_0 \\ 0 & \text{otherwise.} \end{cases}$$

As a concrete example, if we have the duration $N = 100$ sequence $x[n] = e^{j0.2\pi n}$, $n = 0, 1, \ldots, 99$, then (115) holds with $k_0 = 10$. Thus $X[10] = \sqrt{100} = 10$ and all other $X[k]$ values are zero. Note that $X[90] \neq X[10]^*$ because the conjugate symmetry property (114) does not hold when $x[n]$ is complex-valued.

The assumption (115) implies that $x[n]$, interpreted as a vector $\vec{x}$, is exactly aligned with the k_0-th basis vector $\vec{u}_{k_0}$. When ω_0 is *not* exactly equal but close to $k_0 \frac{2\pi}{N}$, we can expect that $\vec{x}$ will be approximately aligned with $\vec{u}_{k_0}$ which means that $X[k]$ peaks around $k = k_0$ and is smaller away from k_0.

Example 6: Now let $x[n] = \cos(\omega_0 n)$ where ω_0 satisfies (115). Rewriting

$$\begin{aligned} x[n] &= \frac{1}{2} e^{j\omega_0 n} + \frac{1}{2} e^{-j\omega_0 n} \\ &= \frac{1}{2} e^{j\omega_0 n} + \frac{1}{2} e^{j(2\pi - \omega_0) n} \\ &= \frac{1}{2} e^{jk_0 \frac{2\pi}{N} n} + \frac{1}{2} e^{j(N-k_0)\frac{2\pi}{N} n} \\ &= \frac{\sqrt{N}}{2} u_{k_0}[n] + \frac{\sqrt{N}}{2} u_{N-k_0}[n] \end{aligned}$$

we conclude

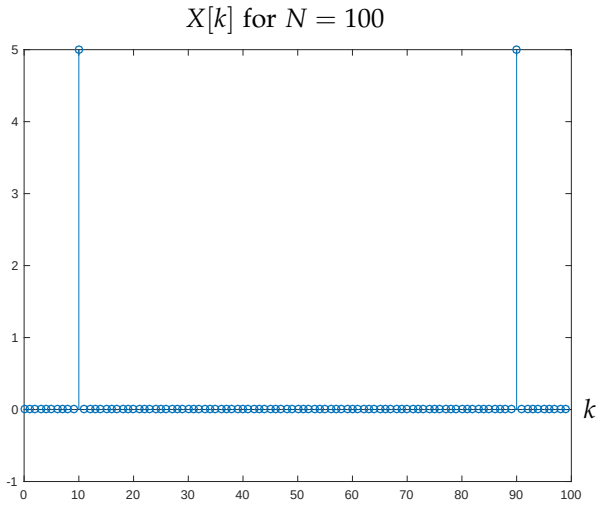
$$X[k] = \begin{cases} \frac{\sqrt{N}}{2} & \text{if } k = k_0 \text{ or } k = N - k_0 \\ 0 & \text{otherwise.} \end{cases} \tag{116}$$

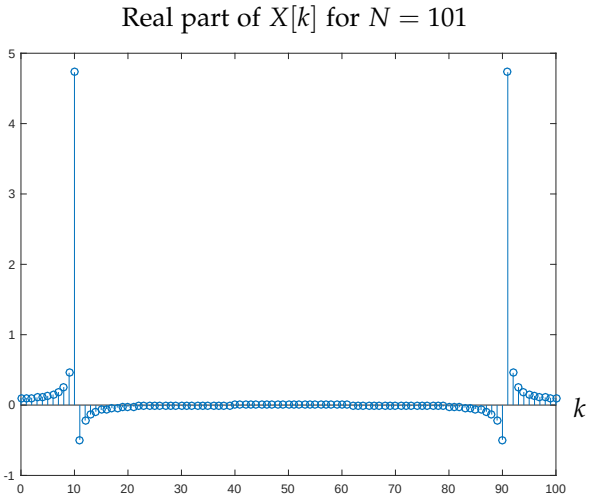
As an illustration, for

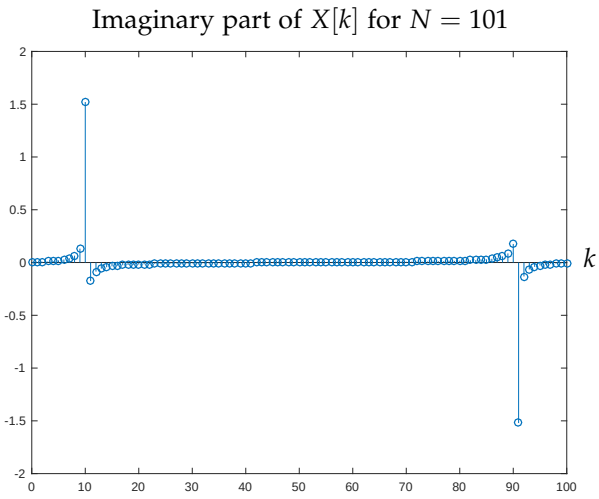
$$x[n] = \cos(0.2\pi n), \quad n = 0, 1, \ldots, N-1 \tag{117}$$

with $N = 100$ we get $X[10] = X[90] = 5$ and $X[k] = 0$ for all other k, as depicted below. The conjugate symmetry property (114) is now satisfied because $x[n]$ is real-valued.

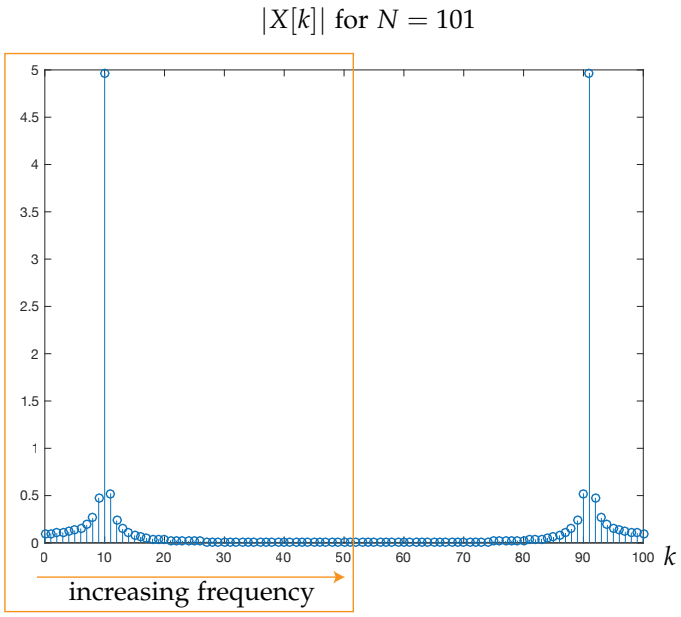
To illustrate what happens when (115) does not hold, we next take $N = 101$ and plot the real and imaginary parts of the resulting DFT. As expected, $X[k]$ peaks near $k = 10$ and $k = 90$.







Note that the real part of $X[k]$ is even symmetric and the imaginary part is odd symmetric about the midpoint, $N/2$, of the k axis. This is a result of the conjugate symmetry property $X[N-k] = X[k]^*$. The plot of $|X[k]|$, shown below, is also even symmetric.



We make two important observations about DFT plots:

1) The second half of the k axis beyond $N/2$ repeats the information contained in the first half due to conjugate symmetry.

2) The midrange of the k axis represents high frequencies. To see this assume, for simplicity, that N is even and note that the DFT basis

signal for $k = N/2$ is

$$u_{N/2}[n] = \frac{1}{\sqrt{N}} e^{j\frac{N}{2}\frac{2\pi}{N}n} = \frac{1}{\sqrt{N}} e^{j\pi n} = \frac{1}{\sqrt{N}}(-1)^n,$$

which varies faster than all other basis signals.

Therefore, the critical part of a DFT plot is the left half where lower values of k represent low frequencies and higher values approaching $N/2$ represent high frequencies.

Linear Time Invariant (LTI) Systems

Consider a system that takes in an input signal $u[n]$ and returns an output signal $y[n]$ as in the figure below.

We say that the system is *linear* if it satisfies the following conditions:

1. Scaling: If an input $u[n]$ generates[13] the output $y[n]$, then

$$au[n] \to ay[n] \quad \text{for any constant } a. \tag{118}$$

2. Superposition: If $u_1[n] \to y_1[n]$ and $u_2[n] \to y_2[n]$, then

$$u_1[n] + u_2[n] \to y_1[n] + y_2[n]. \tag{119}$$

[13] we will use the notation $u[n] \to y[n]$ as a shorthand for "$u[n]$ generates $y[n]$"

If the input to a linear system is 0 then the output must be 0 as well. This follows by choosing $a = 0$ in (118).

A system is called *time invariant* if a time shift in the input results is an identical time shift in the output:

$$u[n - n_0] \to y[n - n_0]. \tag{120}$$

This means that the system reacts the same way to an input applied later as it would have reacted if the input was applied now.

Example 7: Consider the discrete-time state space model

$$\begin{aligned} \vec{x}[n+1] &= A\vec{x}[n] + Bu[n] \\ y[n] &= C\vec{x}[n] \end{aligned} \tag{121}$$

and recall that the solution $\vec{x}[n]$ for $n = 1, 2, 3, \ldots$ is

$$\vec{x}[n] = A^n\vec{x}[0] + A^{n-1}Bu[0] + \cdots + ABu[n-2] + Bu[n-1].$$

Multiplying both sides from the left by C we get

$$y[n] = CA^n\vec{x}[0] + CA^{n-1}Bu[0] + \cdots + CABu[n-2] + CBu[n-1]$$

and, if the initial condition $\vec{x}[0]$ is zero, we conclude

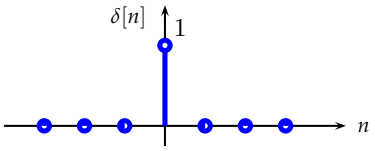
$$y[n] = CA^{n-1}Bu[0] + \cdots + CABu[n-2] + CBu[n-1]. \qquad (122)$$

This relationship satisfies the scaling and superposition properties; therefore the system is linear. It is also time invariant because A, B, C do not depend on time: if we apply the same input starting at time $n = n_0$ instead of $n = 0$, we get the same solution shifted in time by n_0.

Impulse Response of LTI Systems

Systems that are both linear and time invariant possess many useful properties. One such property is that, if we know the system's response to the *unit impulse* sequence

$$\delta[n] \triangleq \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{otherwise,} \end{cases}$$



depicted on the right, we can predict its response to any other input.

We denote the impulse response by $h[n]$, that is $\delta[n] \rightarrow h[n]$. To see how we can use $h[n]$ to predict the system's response to another input $u[n]$, rewrite $u[n]$ as

$$\begin{aligned} u[n] &= \ldots + u[0]\delta[n] + u[1]\delta[n-1] + u[2]\delta[n-2] + \ldots \\ &= \sum_k u[k]\delta[n-k]. \end{aligned}$$

Since $\delta[n] \rightarrow h[n]$, by time-invariance we have $\delta[n-m] \rightarrow h[n-m]$. Then, by linearity, $\sum_m u[m]\delta[n-m] \rightarrow \sum_m u[m]h[n-m]$, therefore

$$\boxed{y[n] = \sum_m u[m]h[n-m].} \qquad (123)$$

This expression is known as the *convolution sum* and shows how we can predict the output for any input using the impulse response $h[n]$.

Example 8: For the state space example above, note from (122) that

$$h[n] = CA^{n-1}B\delta[0] + \cdots + CAB\delta[n-2] + CB\delta[n-1] = CA^{n-1}B$$

for $n \geq 1$, and $h[0] = C\vec{x}[0] = 0$ since we assumed $\vec{x}[0] = 0$ to ensure linearity. Thus,

$$h[0] = 0,\ h[1] = CB,\ h[2] = CAB,\ h[3] = CA^2B,\ \ldots$$

and (122) can be interpreted as a convolution sum.

Example 9: For a LTI system whose output is

$$y[n] = \frac{u[n] + u[n-1]}{2}$$

the impulse response is zero except at $n = 0, 1$ where $h[0] = h[1] = \frac{1}{2}$.

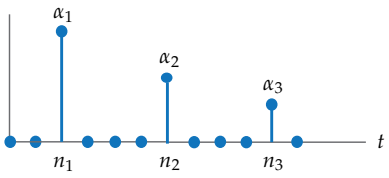
Example 10: Radio waves from a wireless transmitter may be deflected by obstacles, creating echoes of the transmitted signal that reach the receiver with different delays. For example, if there are three paths, the received signal can be written as

$$y[n] = \alpha_1 u[n - n_1] + \alpha_2 u[n - n_2] + \alpha_3 u[n - n_3]$$

which includes three copies of the transmitted signal $u[n]$, each with delay n_i and gain α_i, $i = 1, 2, 3$. In a discrete-time model n_i should be interpreted as the nearest number of sample periods. This input/output relationship is linear; it is also time invariant if we assume that α_i and n_i, $i = 1, 2, 3$ do not change over time. The impulse response is obtained by substituting the unit impulse for the input,

$$h[n] = \alpha_1 \delta[n - n_1] + \alpha_2 \delta[n - n_2] + \alpha_3 \delta[n - n_3],$$

and is depicted on the right.



Finite Impulse Response (FIR) Systems

We now restrict our attention to LTI systems whose impulse response is of finite length. Specifically we assume

$$h[n] = 0 \quad \text{when} \quad n \notin \{0, 1, \ldots, M\} \tag{124}$$

for some integer M, so that only $h[0], \ldots, h[M]$ can be nonzero. Likewise we assume a finite length input $u[n]$, $n = 0, 1, \ldots, L-1$.

It then follows from the convolution sum formula (123) that

$$y[n] = \sum_{m=0}^{L-1} u[m] h[n - m] \tag{125}$$

and it can be shown from (124) that

$$y[n] = 0 \quad \text{when } n \notin \{0, 1, \ldots, M + L - 1\}.$$

For $n \in \{0, 1, \ldots, M+L-1\}$ we write (125) in matrix form as

$$\underbrace{\begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ \vdots \\ \vdots \\ y[M+L-1] \end{bmatrix}}_{\triangleq \vec{y}} = \begin{bmatrix} h[0] & 0 & \cdots & 0 \\ h[1] & h[0] & \ddots & \vdots \\ \vdots & h[1] & \ddots & 0 \\ h[M] & \vdots & \ddots & h[0] \\ 0 & h[M] & & h[1] \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h[M] \end{bmatrix} \begin{bmatrix} u[0] \\ u[1] \\ \vdots \\ u[L-1] \end{bmatrix} \tag{126}$$

where each row in the matrix corresponds to a particular n in (125).

Note that if we apply the unit impulse as the input, then

$$\begin{bmatrix} u[0] \\ u[1] \\ \vdots \\ u[L-1] \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

and multiplication with the matrix in (126) gives

$$\begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ \vdots \\ \vdots \\ y[M+L-1] \end{bmatrix} = \begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[M] \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

which, as expected, is the impulse response.

To match the size of the input and output vectors in (126) we augment the input vector with M zeros,

$$\vec{u} \triangleq \begin{bmatrix} u[0] & u[1] & \cdots & u[L-1] & \underbrace{0 \cdots 0}_{M \text{ zeros}} \end{bmatrix}^T, \tag{127}$$

and complete the $[M+L] \times L$ matrix above into a square matrix by adding M columns. Although the choice of these columns is arbitrary (they get multiplied by the zero entries of $\vec{u}$) we preserve the structure and continue shifting the columns downwards and wrap around

the entries from the bottom back to the top:

$$\vec{y} = \underbrace{\begin{bmatrix} h[0] & 0 & \cdots & 0 & h[M] & \cdots & h[1] \\ h[1] & h[0] & \ddots & \vdots & 0 & \ddots & \vdots \\ \vdots & h[1] & \ddots & 0 & \vdots & \ddots & h[M] \\ h[M] & \vdots & \ddots & h[0] & 0 & \vdots & 0 \\ 0 & h[M] & & h[1] & h[0] & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & 0 & h[M] & h[M-1] & \cdots & h[0] \end{bmatrix}}_{\triangleq H} \vec{u}. \quad (128)$$

The resulting $(M+L)\times(M+L)$ square matrix H belongs to a class of matrices called "circulant."

Analyzing LTI Systems with DFT

Circulant Matrices and Diagonalization with DFT Basis

A circulant matrix has the general form

$$C = \begin{bmatrix} c[0] & c[N-1] & \cdots & c[2] & c[1] \\ c[1] & c[0] & c[N-1] & & c[2] \\ \vdots & c[1] & c[0] & \ddots & \vdots \\ c[N-2] & \vdots & \ddots & \ddots & c[N-1] \\ c[N-1] & c[N-2] & \cdots & c[1] & c[0] \end{bmatrix} \quad (129)$$

which encompasses H in (128) as a special case. Circulant matrices have the following useful property proven at the end of these notes:

The DFT basis diagonalizes all circulant matrices. Specifically,

$$F^* C F = \sqrt{N} \begin{bmatrix} C[0] & & & \\ & C[1] & & \\ & & \ddots & \\ & & & C[N-1] \end{bmatrix} \quad (130)$$

where F is the Fourier matrix defined earlier and $C[0], C[1], \ldots, C[N-1]$ are the DFT coefficients of the sequence $c[0], c[1], \ldots, c[N-1]$ obtained from the first column.

We now use this property to derive a relation between the input and output of a LTI system in the DFT basis. Multiplying both sides of (128) from the left with F^* we write:

$$F^*\vec{y} = F^* H \vec{u} = F^* H F F^* \vec{u}$$

where the second equality follows because $FF^* = I$. Since $F^*\vec{y}$ and $F^*\vec{u}$ give the vector of DFT coefficients $\vec{Y}$ and $\vec{U}$, this means

$$\vec{Y} = F^*HF\vec{U}.$$

Finally, adapting (130) to the matrix H in (128), we get

$$\begin{bmatrix} Y[0] \\ Y[1] \\ \vdots \\ Y[M+L-1] \end{bmatrix} = \sqrt{M+L} \begin{bmatrix} H[0] & & & \\ & H[1] & & \\ & & \ddots & \\ & & & H[M+L-1] \end{bmatrix} \begin{bmatrix} U[0] \\ U[1] \\ \vdots \\ U[M+L-1] \end{bmatrix}$$

where $H[0], H[1], \ldots, H[M+L-1]$ are the DFT coefficients of the length $M+L$ sequence obtained from the first column of H:

$$h[0]\ h[1]\ \cdots\ h[M]\ \underbrace{0\ \cdots\ 0}_{L-1 \text{ zeros}}\ .$$

Likewise, $U[0], U[1], \ldots, U[M+L-1]$ represent the DFT for the input in (127) which has been "padded" with M zeros so that it has the same length as the output and, thus, has as many DFT coefficients.

Because the matrix H in (128) is now diagonalized we simply multiply the DFT coefficients of the input and the DFT coefficients of the impulse response to obtain the DFT coefficients of the output:

$$\boxed{Y[k] = \sqrt{M+L}\, H[k]U[k] \quad k = 0, 1, \ldots, L+M-1.} \tag{131}$$

We can then reconstruct $y[n]$ from its DFT $Y[k]$. This method is preferable to the convolution sum performed in (128) since DFT is computed efficiently with FFT algorithms.

Summary: In the DFT basis a LTI system simply scales the DFT coefficients of the input. Each frequency component k is scaled by a factor $H[k]$ that comes from the DFT of the impulse response.

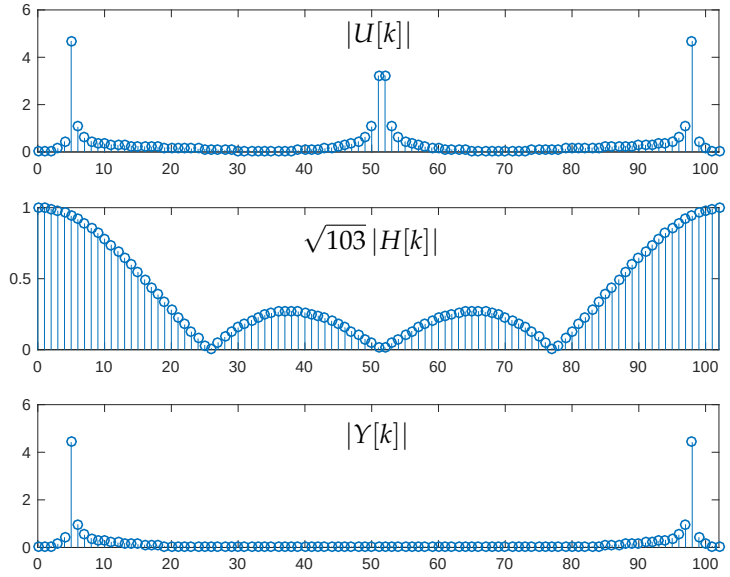
Example 11: Suppose we apply 100 samples of the signal

$$u[n] = \cos(0.1\pi n) + 0.5\cos(\pi n) \quad n = 0, 1, \ldots, 99$$

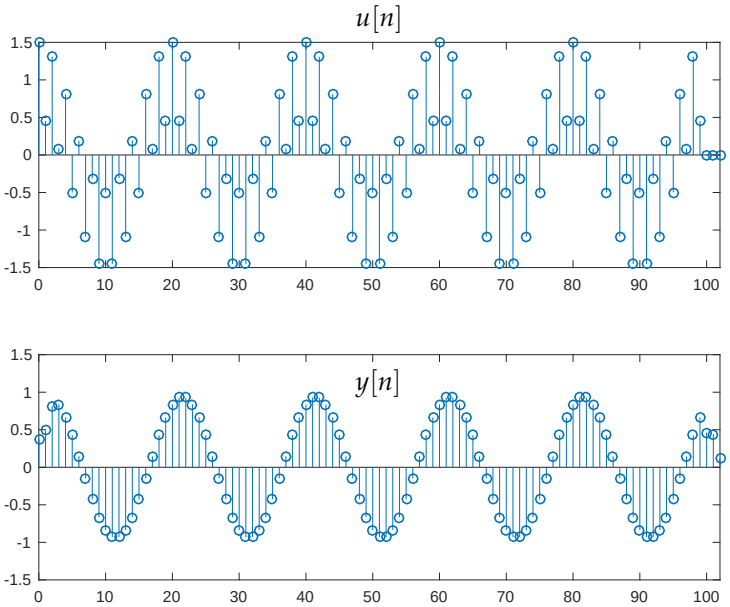
to a LTI system whose impulse response is

$$h[n] = \begin{cases} \frac{1}{4} & n = 0, 1, 2, 3 \\ 0 & \text{otherwise.} \end{cases}$$

To find the output $y[n]$, we pad three zeros to the input and take the DFT of the resulting length 103 sequence. Likewise we pad 99 zeros to the length four impulse response and calculate its DFT. The figure below shows $|U[k]|$ (top) and $\sqrt{103}\,|H[k]|$ (middle), as well as their product (bottom) which gives $|Y[k]|$ from (131).



We see that the spikes around $k = 51$ and 52 (corresponding to the high frequency π) are eliminated in the output because $H[k]$ is close to zero in that frequency range. This is also visible in the $y[n]$ plot below where the $\cos(\pi n)$ component of $u[n]$ has been filtered out.



Proof of (130)

We first rewrite (129) as the sum

$$C = c[0]I + c[1]S + c[2]S^2 + \cdots c[N-1]S^{N-1} \tag{132}$$

where the matrix S is defined as

$$S \triangleq \begin{bmatrix} 0 & \cdots & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}.$$

As an illustration, (132) decomposes a 3×3 circulant matrix as

$$\begin{bmatrix} c[0] & c[2] & c[1] \\ c[1] & c[0] & c[2] \\ c[2] & c[1] & c[0] \end{bmatrix} = c[0]\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + c[1]\underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{S} + c[2]\underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}}_{S^2}.$$

Next we claim that

$$\vec{u}_k^* S = {W_N^k}^* \vec{u}_k^* \tag{133}$$

where $W_N^k = e^{j\frac{2\pi}{N}k}$. To see this recall that

$$\vec{u}_k^* = \frac{1}{\sqrt{N}}\left[1 \; W_k^* \cdots ({W_N^k}^*)^{N-1}\right]$$

and note that multiplication by S gives

$$\vec{u}_k^* S = \frac{1}{\sqrt{N}}\left[{W_N^k}^* \cdots ({W_N^k}^*)^{N-1} \; 1\right] = \frac{1}{\sqrt{N}}\left[{W_N^k}^* \cdots ({W_N^k}^*)^{N-1} \; ({W_N^k}^*)^N\right]$$

where we substituted[14] $({W_N^k}^*)^N = 1$. Factoring out ${W_N^k}^*$ from each entry we get the expression (133). With a recursive application of (133) we also conclude that

[14] This follows because $W_N^k = e^{j\frac{2\pi}{N}k}$.

$$\vec{u}_k^* S^n = (W_N^{kn})^* \vec{u}_k^*.$$

Then, using (132),

$$\vec{u}_k^* C = \sum_{n=0}^{N-1} c[n]\vec{u}_k^* S^n = \sum_{n=0}^{N-1} c[n](W_N^{kn})^* \vec{u}_k^* = \sqrt{N}C(k)\vec{u}_k^* \tag{134}$$

where the last equality follows because

$$\sum_{n=0}^{N-1} c_n (W_N^{kn})^* = \left[1 \; {W_N^k}^* \cdots ({W_N^k}^*)^{N-1}\right]\begin{bmatrix} c[0] \\ c[1] \\ \vdots \\ c[N-1] \end{bmatrix} = \sqrt{N}\,\vec{u}_k^* \underbrace{\begin{bmatrix} c[0] \\ c[1] \\ \vdots \\ c[N-1] \end{bmatrix}}_{C[k]}.$$

Finally note from (**??**) that

$$\underbrace{\begin{bmatrix} \vec{u}_0^* \\ \vec{u}_1^* \\ \vec{u}_2^* \\ \vdots \\ \vec{u}_{N-1}^* \end{bmatrix}}_{F^*} C = \sqrt{N} \begin{bmatrix} C[0] & & & \\ & C[1] & & \\ & & \ddots & \\ & & & C[N-1] \end{bmatrix} \underbrace{\begin{bmatrix} \vec{u}_0^* \\ \vec{u}_1^* \\ \vec{u}_2^* \\ \vdots \\ \vec{u}_{N-1}^* \end{bmatrix}}_{F^*}.$$

Multiplying both sides from the right by F and substituting $F^*F = I$ on the right hand side, we obtain (130).