Outline

- $f = ma$ reviewed
- Ordinary Differential Equations
- Finite Difference Schemes
- Overview of Model Types
- Digitizing Models in State Space Form
- Summary, Overview, and Related Research

Two Approaches to Physical Modeling

1. “White Box” Modeling:
   (a) Find the describing differential equations from basic physical principles
   (b) Digitize the differential equations to obtain difference equations implemented in software

2. “Black Box” Modeling:
   (a) Measure the system response to a representative set of input signals
   (b) Fit a computational model to the measured input-output set
   (c) In the Linear, Time-Invariant (LTI) case, a Multi-Input, Multi-Output (MIMO) digital filter will suffice

This class blends white- and black-box approaches:

1. LTI sections become fast, accurate digital filters
2. Nonlinear or rapidly time-varying subsystems normally get a white-box approach (reeds, hammers, bows, . . .)
**Ordinary Differential Equations**

Ordinary Differential Equations (ODEs) typically result from Newton’s laws of motion:

\[ f(t) = ma(t) \quad \text{(Force = Mass times Acceleration)} \]

Acceleration \( a(t) \) relates to velocity \( v(t) \) and position \( x(t) \) by differentiation with respect to time \( t \):

\[
\begin{align*}
    a(t) & \triangleq \dot{v}(t) \triangleq \frac{d\dot{x}(t)}{dt} \triangleq \ddot{x}(t) \triangleq \frac{d^2x(t)}{dt^2}
\end{align*}
\]

**Physical Diagram:**

\[ a(t), v(t), x(t) \]

\[ x = 0 \]

\[ f(t) \rightarrow m \]

Force \( f(t) \) driving mass \( m \) along frictionless surface

---

**Equivalent Circuit for a Force-Driven Mass**

\[
\begin{align*}
    v(t) & \rightarrow \\
    f(t) & \rightarrow m \\
\end{align*}
\]

- Mass \( m \) is an **inductor** \( L = m \) Henrys
- Driving force \( f(t) \) is a **voltage source**
- Mass velocity \( v(t) \) is the **loop current**

The ODE is obtained from the equivalent circuit by summing all "voltages" around the current loop to zero to obtain

\[
-f(t) + m\dot{v} = 0
\]

The minus sign for \( f(t) \) occurs because the current arrow entered the minus side of the "voltage source"
Reference Directions in Equivalent Circuits

\[ m + f(t) + v(t) - f(t) + m \dot{v} = 0 \]

- “Reference directions” (±) on the voltage source and circuit elements may be chosen arbitrarily—just keep track and be consistent.

- When \( f(t) \) is positive, “current” is pushed from its + to its − terminal, i.e., \( v(t) \) will be positive if the rest of the circuit is just a wire or a resistor.

- The “force drop” across the mass \( m \) is positive when \( v(t) \) increases in the direction going from its + to − terminal. This can be interpreted as the inertial reaction force of the mass that opposed the external applied force (Newton’s first law of motion).

ODE for a Mass Sliding with Friction

Force \( f(t) \) driving mass \( m \) along surface with friction force \( \mu v(t) \):

\[ f(t) = m \ddot{x}(t) + \mu v(t) \]

\[ = m \ddot{x}(t) + \mu \dot{x}(t) \]

- Note that the friction force is positive to the left in this figure, i.e., it is a reaction force

- The inertial reaction force of the mass points to the left as well (not shown, but equal to \(-f(t)\))
**Force-Driven Mass with Friction**

*Diagram and Equivalent Circuit*

- Force driving an ideal mass and dashpot

**Mass-Spring ODE**

An *ideal spring* described by Hooke’s law

\[ f(t) = k x(t) = k \int_0^t v(\tau) \, d\tau \quad \leftrightarrow \quad \frac{V(s)}{s} \]

where \( k \) denotes the *spring constant*, \( x(t) \) denotes the *compressive* spring displacement from rest at time \( t \), and \( f(t) \) is the force required for displacement \( x(t) \).

If the force on a mass is due to a spring then, as discussed later, we may write the ODE as

\[ k x(t) + m \ddot{x}(t) = 0 \]

(Spring Force + Mass Inertial Force = 0)

**Physical diagram:**

\[ \dot{x}(t) \rightarrow \]

\[ m \]

\[ k \]

\[ x = 0 \]

\[ x(t) \rightarrow \]
Mass-Spring-Wall System

\[ f_{\text{ext}}(t) - f_m(t) - f_k(t) = 0 \]

\[ v(t) \rightarrow \]

- Driving force \( f_{\text{ext}}(t) \) is to the right on the mass
- Driving force + mass inertial force + spring force = 0
- Mass velocity = spring velocity
- This is a series combination of the spring and mass

If two physical elements are connected so that they share a common velocity, then they are said to be formally connected in series

Equivalent Circuit for Mass-Spring-Wall

The “series” nature of the connection becomes more clear when the equivalent circuit is considered:

\[ v_m(t) = v_k(t) \]

\( f_{\text{ext}}(t) \leftrightarrow \) voltage source

- The driving force is applied to the mass such that a positive force results in a positive mass displacement and positive spring displacement (compression)
- The common mass and spring velocity appear as a single current running through the inductor and capacitor that model the mass and spring, respectively
Mass-Spring-Dashpot ODE

If the mass is sliding with friction, then a simple ODE model is given by

\[ k \, x(t) + \mu \, \dot{x}(t) + m \, \ddot{x}(t) = 0 \]

(Spring + Friction + Inertial Forces = 0)

Physical diagram:

We will use such ODEs to model mass, spring, and dashpot elements, and their equivalent circuits

---

Difference Equations
(Finite Difference Schemes)

- There are many methods for converting ODEs to difference equations
- For example, we’ll use a very simple, order-preserving method which replaces each derivative with a finite difference:

\[ \dot{x}(t) \triangleq \frac{d}{dt} x(t) \triangleq \lim_{\delta \to 0} \frac{x(t) - x(t - \delta)}{\delta} \approx \frac{x(nT) - x((n-1)T)}{T} \triangleq \hat{x}(t) \]

for sufficiently small \( T \) (the sampling interval)

- This is formally known as the backward difference, or Backward Euler (BE) method for differentiation approximation
- We’ll look at a few others as well
- For a more advanced treatment of finite difference schemes, see, e.g., *Numerical Sound Synthesis* by Stefan Bilbao (2009, Wiley)
Difference Equation for a Force-Driven Mass

- Newton’s $f = ma$ can be written in terms of force and velocity as:
  $$f(t) = m \dot{v}(t)$$

- The backward-difference substitution gives:
  $$f(nT) \approx m \frac{v(nT) - v((n-1)T)}{T} = m \hat{\dot{v}}(nT)$$
  for $n = 0, 1, 2, \ldots$. Or, in a lighter notation,
  $$f_n \approx m \frac{v_n - v_{n-1}}{T} = m \hat{\dot{v}}_n, \quad n = 0, 1, 2, \ldots$$
  with $v_{-1} \triangleq 0$

- Note that $\hat{\dot{v}}_n$ is more accurately written as $\hat{\dot{v}}_{n-\frac{1}{2}}$.

- Solving for $v_n$ yields a difference equation (finite difference scheme):
  $$\hat{\dot{v}}_n = \hat{\dot{v}}_{n-\frac{1}{2}} + \frac{T}{m} f_n, \quad n = 0, 1, 2, \ldots$$
  with $\hat{\dot{v}}_{-\frac{1}{2}} \triangleq 0$

Summary of Backward Euler

- $v_n = v_{n-1} + T \hat{\dot{v}}_n$
- $\iff \hat{\dot{v}}_n = \frac{v_n - v_{n-1}}{T}$
- $\Downarrow = \Downarrow$
- $V(z) = z^{-1}V(z) + T \hat{V}(z)$
- $\Rightarrow \hat{V}(z) = \frac{1 - z^{-1}}{T} V(z)$
Filter Design Approach

We’ve been talking about the white-box approach: Find the describing differential equations (often in the form of an equivalent circuit)

Let’s also think about the black-box approach for the simple case of an integrator (force-driven mass with a velocity output), and differentiator (force-driven spring with a velocity output):

- **Ideal Digital Integrator**
  \[ H(e^{j\omega T}) = \frac{1}{j\omega}, \quad \omega \in [-\pi/T, \pi/T] \]

- **Ideal Digital Differentiator**:
  \[ H(e^{j\omega T}) = j\omega, \quad \omega \in [-\pi/T, \pi/T] \]

- Exact match is not possible in finite order

- Minimize \( \| H(e^{j\omega T}) - \hat{H}(e^{j\omega T}) \| \) where \( \hat{H} \) is the digital filter frequency response and \( \| E \| \) denotes some norm of \( E \)

- This is a digital filter design formulation

Ideal Differentiator Frequency Response

- Discontinuity at \( z = -1 \Rightarrow \) no exact solution (polynomial approximation over the unit circle)

- Need oversampling and a don’t-care band at high frequencies (e.g., 20 kHz to 22.05 kHz)

- The frequency response can be arbitrary between the upper limit of human hearing (20kHz) and \( f_s/2 \)

- A small increment in oversampling factor yields a large decrease in required filter order for a given spec
Backward Difference as a Conformal Map

(Back to white-box modeling)

\[ \hat{v}_n = \frac{v_n - v_{n-1}}{T} \]

\[ \hat{V}(z) = \left(\frac{1 - z^{-1}}{T}\right) V(z) \]

In the continuous time ("analog") case, we have

\[ \hat{V}_a(s) = s V_a(s). \]

Noting the algebraic correspondence

\[ s \leftrightarrow \frac{1 - z^{-1}}{T} \]

we may define

\[ V(z) \triangleq V_a \left(\frac{1 - z^{-1}}{T}\right) \]

\[ \hat{V}(z) \triangleq \hat{V}_a \left(\frac{1 - z^{-1}}{T}\right) = \left(\frac{1 - z^{-1}}{T}\right) \cdot V(z) \]

to obtain the Backward Euler finite difference scheme via
the substitution \( s \leftarrow (1 - z^{-1})/T \) in the frequency
domain (a conformal map from the \( s \)-plane to the \( z \)-plane).

Delay-Free Loops

Backward-Euler numerical integrator:

\[ v_n = v_{n-1} + T \hat{v}_n \]

Corresponding Backward Euler mass-model:

\[ \hat{v}_n = \hat{v}_{n-1} + \frac{T}{m} f_n \]

where \( \hat{v}_n \) is the \( n \)th sample of the estimated velocity, \( f_n \)
is the driving force at sample \( n \), \( m \) is the mass, and \( T \) is
the sampling interval.

- Note that a delay-free loop appears if \( f_n \) depends on
\( v_n \) (e.g., due to friction):

\[ \hat{v}_n = \hat{v}_{n-1} + \frac{T}{m} f_n(\hat{v}_n) \]

- In such a case, the difference equation is not
computable in this form

- Non-computable finite-differences schemes such as
this are said to be implicit

- We can address this by using a forward-difference
("Forward Euler") in place of a backward difference
Replacing Backward-Euler by Forward-Euler

- The backward difference was based on the usual left-sided limit in the definition of the time derivative:
  \[ \dot{x}(t) = \lim_{\delta \to 0} \frac{x(t) - x(t - \delta)}{\delta} \approx \frac{x_n - x_{n-1}}{T} \]

- The forward difference is based on the right-sided limit:
  \[ \dot{x}(t) = \lim_{\delta \to 0} \frac{x(t + \delta) - x(t)}{\delta} \approx \frac{x_{n+1} - x_n}{T} \]

- As \( T \to 0 \), the forward and backward difference approximations approach the same limit, because \( x(t) \) is assumed continuous and differentiable at \( t \).

- The forward difference gives an explicit finite difference scheme even if the driving force depends on current velocity:
  \[ \hat{v}_{n+1} = \hat{v}_n + \frac{T}{m} f_n, \quad n = 0, 1, 2, \ldots \]
  with \( v_0 \triangleq 0 \)

- We obtain essentially the same result by introducing an ad hoc delay in the driving force of the Backward Euler scheme to get \( \hat{v}_n = \hat{v}_{n-1} + (T/m) f_{n-1} \)

Centered Finite Difference

- No time delay or advance
- Cf. the Leapfrog integrator
- \( s \) to \( z \) mapping is
  \[ s = \frac{z - z^{-1}}{2T} \quad \Rightarrow \quad \frac{e^{j\omega T} - e^{-j\omega T}}{2T} = j \frac{\sin(\omega T)}{T} \approx j\omega \]
  at low frequencies, but note how it reaches a maximum at \( \omega T = \pi/2 \) and comes back down to 0 at \( \omega T = \pi \)
Explicit and Implicit Finite Difference Schemes

Explicit:
\[ y_{n+1} = x_n + f(y_n) \]

Implicit:
\[ y_{n+1} = x_n + f(y_{n+1}) \]

- A finite difference scheme is said to be explicit when it can be computed forward in time using quantities from previous time steps.
- We will associate explicit finite difference schemes with causal digital filters.
- In implicit finite-difference schemes, the output of the time-update (\(y_{n+1}\) above) depends on itself, so a causal recursive computation is not specified.
- Implicit schemes are generally solved using
  - iterative methods (such as Newton’s method) in nonlinear cases, and
  - matrix-inverse methods for linear problems.
- Implicit schemes are typically used offline (not in real time).

Semi-Implicit Finite Difference Schemes

- Implicit schemes can often be converted to explicit schemes (e.g., for real-time usage) by limiting the number of iterations used to solve the implicit scheme.
- These are called semi-implicit finite-difference schemes.
- Iterative convergence is generally improved by working at a very high sampling rate, and by initializing each iteration to the solution for the previous sample.
- See the 2009 CCRMA/EE thesis by David Yeh\(^1\) for semi-implicit schemes for real-time computational modeling of nonlinear analog guitar effects (such as overdrive distortion).
- Convex optimization methods can be used to develop powerful new semi-implicit finite-difference schemes: \[ \text{http://www.stanford.edu/~boyd/cvxbook/} \]

\(^{1}\) \url{http://ccrma.stanford.edu/~dtyeh}
ODE Laplace Transform Analysis

Recall the mass \( m \) sliding on friction \( \mu \):
\[
\begin{align*}
\dot{x}(t) &= x(0) + \int_0^t f(t) \, dt - \mu \int_0^t v(t) \, dt \\
m \ddot{x}(t) + \mu \dot{x}(t) &= m \ddot{x}(t) + \mu \dot{x}(t) \\
\end{align*}
\]

ODE:
\[
f(t) = m \ddot{x}(t) + \mu \dot{x}(t)
\]

Take the Laplace Transform of both sides and apply the differentiation theorem (three times):
\[
F(s) = m \left[ s^2 X(s) - s x(0) - \dot{x}(0) \right] + \mu \left[ s X(s) - x(0) \right]
\]
assuming zero initial conditions \( x(0) = \dot{x}(0) = 0 \).

Force-to-Velocity Transfer Function (often called the “admittance” or “mobility”):
\[
H(s) \triangleq \frac{V(s)}{F(s)} = \frac{s X(s)}{F(s)} = \frac{1}{ms + \mu}
\]

Bilinear Transform

The bilinear transform is a one-to-one mapping from the \( s \) plane to the \( z \) plane:
\[
s = \frac{c}{1 + z^{-1}}, \quad c > 0, \quad c = \frac{2}{T} \quad \text{(typically)}
\]
\[
\Rightarrow z = \frac{1 + s/c}{1 - s/c}
\]

Starting with a continuous-time transfer function \( H_a(s) \), we obtain the discrete-time transfer function \( H_d(z) \):
\[
H_d(z) \triangleq H_a \left( \frac{1 - z^{-1}}{1 + z^{-1}} \right)
\]
where “\( d \)” denotes “digital,” and “\( a \)” denotes “analog.”
Properties of the Bilinear Transform

The bilinear transform maps an $s$-plane transfer function $H_a(s)$ to a $z$-plane transfer function:

$$H_d(z) \triangleq H_a \left( \frac{1 - z^{-1}}{c + 1 + z^{-1}} \right)$$

We can observe the following properties of the bilinear transform:

- Analog dc ($s = 0$) maps to digital dc ($z = 1$)
- Infinite analog frequency ($s = \infty$) maps to the maximum digital frequency ($z = -1$)
- The entire $j\omega$ axis in the $s$ plane (where $s \triangleq \sigma + j\omega$) is mapped exactly once around the unit circle in the $z$ plane (rather than summing around it infinitely many times, or “aliasing” as it does in ordinary sampling)
- Stability is preserved (when $c$ is real and positive)
- Order of the transfer function is preserved
- Choose $c$ to map any particular finite frequency (such as a resonance frequency) from the $j\omega_a$ axis in the $s$ plane to a particular desired location on the unit circle $e^{j\omega_d}$ in the $z$ plane. Other frequencies are “warped”.

Bilinear Transform of Force-Driven Mass

We have, from $f = m\ddot{v} \leftrightarrow F(s) = ms\, V(s)$,

$$V(s) = \frac{1}{ms} F(s)$$

Setting $s = (2/T)(1 - z^{-1})/(1 + z^{-1})$ according to the bilinear transform yields

$$V_d(z) = \frac{T}{2m} \frac{1 + z^{-1}}{1 - z^{-1}} F_d(z)$$

where we defined

$$F_d(z) = F \left( \frac{2}{T} \frac{1 - z^{-1}}{1 + z^{-1}} \right)$$

$$V_d(z) = V \left( \frac{2}{T} \frac{1 - z^{-1}}{1 + z^{-1}} \right)$$

The resulting finite-difference scheme is then

$$v_d(n) - v_d(n - 1) = \frac{T}{2m} [f_d(n) + f_d(n - 1)]$$

i.e.,

$$v_d(n) = v_d(n - 1) + \frac{T}{2m} [f_d(n) + f_d(n - 1)]$$

We see that this is the same as the backward Euler scheme plus a new term $(T/2m)f_d(n - 1)$. 

Trapezoidal Rule for Numerical Integration

The velocity $v(t)$ can be written as

$$v(t) = v(0) + \left( \int_0^t \dot{v}(\tau) d\tau \right)$$

In particular,

$$v(nT) = v(0) + \int_0^{(n-1)T} \dot{v}(\tau) d\tau + \int_{(n-1)T}^{nT} \dot{v}(\tau) d\tau$$

$$= v[(n-1)T] + \int_{(n-1)T}^{nT} \dot{v}(\tau) d\tau$$

$$\approx v[(n-1)T] + T \frac{\dot{v}[(n-1)T] + \dot{v}(nT)}{2}$$

- This approximation replaces a one-sample integral by the area under the trapezoid having vertices $(n-1, 0), (n-1, \dot{v}_{n-1}), (n, 0), (n, \dot{v}_n)$
- In other words, $\dot{v}(t)$ is approximated by a straight line between time $n - 1$ and $n$
- This is a first-order approximation of $\dot{v}(t)$ in contrast to the zero-order approximation used by forward and backward Euler schemes
- Note that the bilinear transform is equivalent

Bilinear Transform Physically Derived

In Newton’s law $f = m\ddot{v}$, start with the Backward Euler approximation of the time-derivative:

$$f(t) = m\ddot{v} \approx m \frac{v(t) - v(t-T)}{T}$$

Notice that there is a 1/2 sample delay in the first-order difference on the right. This misaligns the force $f(t)$ and subsequent velocity by half a sample. A very simple delay compensation is to use a two-point average on the left:

$$\frac{f(n) + f(n-1)}{2} \approx m \frac{v(n) - v(n-1)}{T}$$

Taking the $z$ transform:

$$\frac{1 + z^{-1}}{2} F(z) \approx m \frac{1 - z^{-1}}{T} V(z)$$

or

$$F(z) \approx m \left( \frac{2 - z^{-1}}{T(1 + z^{-1})} \right) V(z)$$

which is the normal bilinear transform of $F(s) = ms V(s)$ using

$$s \mapsto \frac{2 - z^{-1}}{T(1 + z^{-1})}.$$
Frequency Warping is the Only Error

We just physically derived

\[ F(z) \approx m \left( \frac{2 + z^{-1}}{T + z^{-1}} \right) V(z) \]

and noted that it was the usual bilinear transform (trapezoidal integration in the time domain)

Let’s look along the unit circle in the \( z \) plane:

\[ \frac{F(e^{j\omega T})}{V(e^{j\omega T})} \approx m \left( \frac{2 + e^{-j\omega T}}{T + e^{-j\omega T}} \right) = m j \left( \frac{2}{T} \tan \left( \frac{\omega T}{2} \right) \right) \]

Since the exact formula is \( \frac{F(e^{j\omega T})}{V(e^{j\omega T})} = m j \omega \), we can push all of the error into a frequency warping:

\[ \omega_d = \frac{2}{T} \tan \left( \frac{\omega_a T}{2} \right) \]

- Frequency-warping is the only error over the unit circle when using the bilinear transform
- What started out as different gain errors on the left and right became the correct gains at warped frequency locations
- Frequency-warping implications should also be considered in the time domain

Hybrid Euler-Bilinear Mapping

We can easily interpolate between Backward Euler and Bilinear Transform:

\[ s \rightarrow \frac{1 + \alpha \cdot 1 - z^{-1}}{T \cdot 1 + \alpha \cdot z^{-1}} \]

- \( \alpha = 0 \) gives Backward Euler (high-frequency modes artificially damped)
- \( \alpha = 1 \) gives Bilinear Transform (high-frequency modes artificially squeezed in frequency)
- Intermediate \( \alpha \) allows optimization of another consideration, such as decay time
- Low-frequency response approximately invariant, dc maps to dc in every case

Example: Leaky Integrator

\[ H_a(s) = \frac{1}{s + \epsilon} \rightarrow H_d(z) = \frac{1}{\frac{1 + \alpha}{T} \cdot 1 - z^{-1} + \epsilon} \]

\[ = g \frac{1 + \alpha z^{-1}}{1 - pz^{-1}}, \quad p = \frac{1 - \alpha}{1 + \frac{\epsilon T}{1 + \alpha}}, \quad g = \frac{T}{1 + \alpha + \epsilon T} \]
Accuracy of Backward Euler

Suppose we take the backward-difference approximation
\[ f_n = \frac{m}{T}(v_n - v_{n-1}) \]
and expand \( v_{n-1} \) in Taylor series about \( v_n \). This yields:
\[
f_n = \frac{m}{T}\left(v_n - \left(v_n - T\frac{dv}{dt}\bigg|_{nT} + O(T^2)\right)\right)
= \frac{m}{T}\frac{dv}{dt}\bigg|_{nT} + O(T)
\]

- We say that the backward difference approximation has an error of order \( T \), written \( O(T) \).
- The order of the error tells us how fast the error approaches zero as the sampling rate \( f_s = 1/T \) approaches infinity.
- Backward Euler maps infinite frequency \( s = \infty \) to \( z = 0 \) (maximally damped), while trapezoidal rule (bilinear transform) maps \( s = \infty \) to \( z = -1 \) (no damping introduced).

Accuracy of Trapezoidal Rule

For the Trapezoid Rule (bilinear transform),
\[
f_n = m\frac{dv}{dt}\bigg|_{nT} + O(T^2)
\]
so it is second-order accurate in \( T \).
We will come back to this below.
Backward Difference Conformal Map

We saw that the backwards difference substitution can be seen as a conformal map taking the $s$ plane to the $z$ plane:

$$s \rightarrow \frac{1 - z^{-1}}{T}$$

Look at the image of the $j\omega$ axis under this mapping:

The continuous-time frequency axis, $s = j\omega$, is not mapped to the discrete-time frequency axis (unit circle):

- dc ($s = 0$) mapped to dc ($z = 1$)
- infinite frequency mapped to ($z = 0$)

This means artificial damping will be introduced for high-frequency system resonances

Laplace Analysis of Trapezoidal Rule

The $z$ transform of the trapezoid rule yields

$$F(z) = \frac{2m}{T} \frac{1 - z^{-1}}{1 + z^{-1}} V(z)$$

Since $F(s) = ms V(s)$, the $s$ to $z$ mapping has become

$$s \rightarrow \frac{2m}{T} \frac{1 - z^{-1}}{1 + z^{-1}}$$

which is of course the standard bilinear transform:

- $s = j\omega$ axis maps to the $|z| = 1$ unit circle where it belongs
- dc maps to dc
- Infinite frequency maps to half the sampling rate
- Frequency axis is warped, especially at high frequencies
- Stability preserved precisely
Trapezoidal Rule Frequency Mapping

Let’s look at the $s$ to $z$ mapping,

$$s = \frac{2 \frac{1}{1} - z^{-1}}{1 + z^{-1}}$$

on the unit circle, where $s = j\omega_a$ and $z = e^{j\omega d T}$:

$$j\omega_a = \frac{2 \frac{1}{1} - e^{-j\omega d T}}{1 + e^{-j\omega d T} T} = \frac{2 T}{T} \tan(\omega d T/2)$$

or

$$\frac{\omega a T}{2} = \tan\left(\frac{\omega d T}{2}\right)$$

- Near dc ($\omega_d = 0$), we have
  $$\omega_a = \frac{2 T}{T} \tan(\omega d T/2) = \omega_d + \mathcal{O}(T^3)$$
  where, since $\tan(\theta)$ is odd, there are no even-order terms in its series expansion

In general, the trapezoid rule is a second-order accurate approximation to a derivative, in the limit of small $T$ (i.e., near dc). Here, it is third-order accurate along the unit circle at dc.

Summary of Backward Euler vs. Trapezoidal Rule

For

$$f(t) = ma(t) = m \dot{v}(t) = \ddot{p}(t)$$

$$= \lim_{T \to 0} \frac{p(t) - p(t - T)}{T} \approx \frac{p(t) - p(t - T)}{T}$$

- Backward Euler (BE)
  $$f_n = \frac{1}{T} (p_n - p_{n-1})$$
  is $\mathcal{O}(T)$ (first-order accurate in $T$)

- Bilinear Transform, or Trapezoidal Rule (TR)
  $$f_n = \frac{2}{T} (p_n - p_{n-1}) - f_{n-1},$$
  is $\mathcal{O}(T^2)$ (second-order accurate in $T$)

- A continuum of transforms
  
  $$s = \frac{1 + \alpha - z^{-1}}{1 + \alpha z^{-1}}$$

exists between BE and TR and can be optimized for the application at hand (see Kurt Werner thesis and Germain and Werner DAFx-15 paper for details—Germain thesis coming soon)
Why Don’t We Always Use the Bilinear Transform?

- Backward Euler (BE) is still sometimes needed:
  - Damps out unwanted high-frequency oscillations (warped)
  - Avoids oscillations at half the sampling rate from a real exponential
  * TR warps high-frequency poles toward half the sampling rate:
    \[ s = g' \cdot \frac{(1 - z^{-1})}{(1 + z^{-1})} \]
    toward \( z = -1 \leftrightarrow (-1)^n \)
  * BE warps high-frequency poles toward \( z = 0 \) so it never introduces alternating-sign oscillations:
    \[ s = g \cdot (1 - z^{-1}) \]
  * Alternating-sign oscillations due to BLT can be problematic in nonlinear circuits such as those containing diodes (see Kurt Werner thesis for a real-world example)

- Recall also that Forward Euler (FE) can break a delay-free loop, and pairs well with BE in series

Physical Model Formulations

Below are names of various kinds of physical model representations we will consider:

- Ordinary Differential Equations (ODE)
- Partial Differential Equations (PDE)
- Difference Equations (DE)
- Finite Difference Schemes (FDS)
- (Physical) State Space Models
- Transfer Functions (between physical signals)
- Modal Representations (Parallel Second-Order Filters)
- Equivalent Circuits
- Impedance Networks
- Wave Digital Filters (WDF)
- Digital Waveguide (DW) Networks

We are mainly concerned with real-time computational physical models
State-Space Models

The state space formulation replaces an $N$th-order ODE by a vector first-order ODE.

Review of discrete-time case:

\[
\begin{align*}
\mathbf{x}(n+1) &= \mathbf{A}\mathbf{x}(n) + \mathbf{B}\mathbf{u}(n) \\
y(n) &= \mathbf{C}\mathbf{x}(n) + \mathbf{D}\mathbf{u}(n)
\end{align*}
\]

where

- $\mathbf{x}(n) \in \mathbb{R}^N = \text{state vector at time } n$
- $\mathbf{u}(n) = p \times 1$ vector of inputs
- $y(n) = q \times 1$ output vector
- $\mathbf{A} = N \times N$ state transition matrix
- $\mathbf{B} = N \times p$ input coefficient matrix
- $\mathbf{C} = q \times N$ output coefficient matrix
- $\mathbf{D} = q \times p$ direct path coefficient matrix

The state-space representation is especially powerful for

- multi-input, multi-output (MIMO) linear systems
- time-varying linear systems
  (every matrix can have a time subscript $n$)

Continuous-Time State Space Models:

In continuous time, we obtain a first-order vector ODE in which a vector of state time-derivatives is driven by linear combinations of state variables:

\[
\begin{align*}
\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\
y(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)
\end{align*}
\]

State-Space Advantages:

- State-space models are used extensively in advanced modeling applications
- Extensive support in Matlab, with many numerically excellent associated tools and techniques (such as the singular value decomposition, to name one)
- Analytically powerful for theory work
- Example: Solution of $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$ is $\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0)$, where the matrix exponential is defined as

\[
e^{\mathbf{A}t} \triangleq I + \mathbf{A}t + \frac{1}{2!}\mathbf{A}^2t^2 + \frac{1}{3!}\mathbf{A}^3t^3 + \ldots
\]

- We won’t do much with state-space modeling in this class, but you should know it exists and that it should be considered for larger, more complex systems than we will be dealing with
Digitizing State Space Models (Simplistically)

Starting with a continuous-time state-space model

\[
\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) + Du(t)
\end{align*}
\]

we can, e.g., apply Backward Euler, Trapezoidal Rule (Bilinear Transform), or anything in between:

\[
s = g \frac{1 - z^{-1}}{1 + \alpha z^{-1}}, \quad \alpha \in [0, 1]
\]

to get, letting \( g = (1 + \alpha)/T \) and defining \( x_n = x(nT) \),

\[
\begin{align*}
\frac{x_n - x_{n-1}}{T} &= A \left[ x_n + \alpha x_{n-1} \right] + B \left[ u_n + \alpha u_{n-1} \right] \\
y_n &= Cx_n + Du_n
\end{align*}
\]

for zero initial conditions \( x(0) = 0 \) \( \Rightarrow \)

\[
\begin{align*}
x_{n+1} &= \left( I - A \frac{T}{1 + \alpha} \right)^{-1} \left( I + A \frac{\alpha T}{1 + \alpha} \right) x_n \\
&\quad + \left( I - A \frac{T}{1 + \alpha} \right)^{-1} BT \left( z + \alpha \right) u_n
\end{align*}
\]

where \( z u_n \triangleq u_{n+1} \)

More sophisticated methods will digitize in a manner that conserves energy and/or momentum.
Recommended Related Courses at Stanford

- Math 226
- AA 214 A/B/C
- ME 300 A/B/C
- ME 335 A/B/C