Intro to Control Theory for LIGO People

G1600726-v3

Summary

- Lecture 1: Introduction
  - Part 1: Intro to controls
  - Part 2: System modeling
- Lecture 2: Basic Control Design
  - Part 1: Feedforward
  - Part 2: Feedback
  - Part 3: Sensor Blending
- Lecture 3: Digital Control

Background Assumptions

- These lectures assume no prior knowledge of controls.
- They do assume some prior exposure to
  - Ordinary differential equations
  - Fourier transform
  - Laplace transform

Useful References

- Digital Control of Dynamic Systems, 3rd Ed. Franklin, Powell, and Workman.
  - Main focus is on the control of digitally sampled systems, but also has a good review of continuous control, system identification, optimal control, and nonlinear systems.
- Modern Control Engineering, 5th Ed. Ogata
  - Standard introductory text to control

Lecture 1

Introduction to Controls & System Modeling

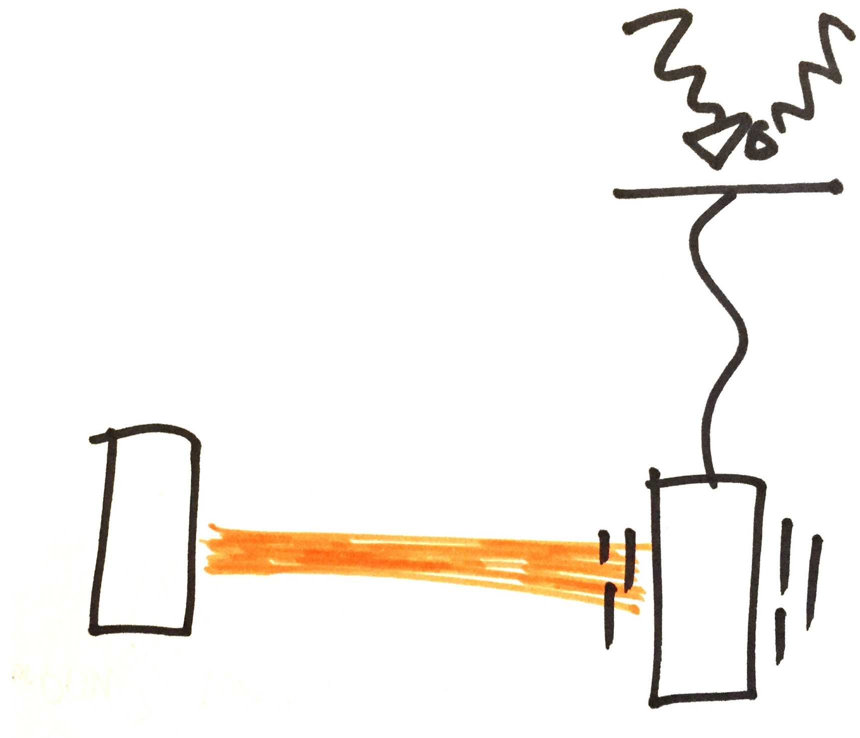
Lecture 1 - Part 1

Introduction to Controls

Why we need control

- The ground moves and disturbs our mirrors.
- We use control to keep the arm lengths at a constant 4 km +- 10^{-14} m

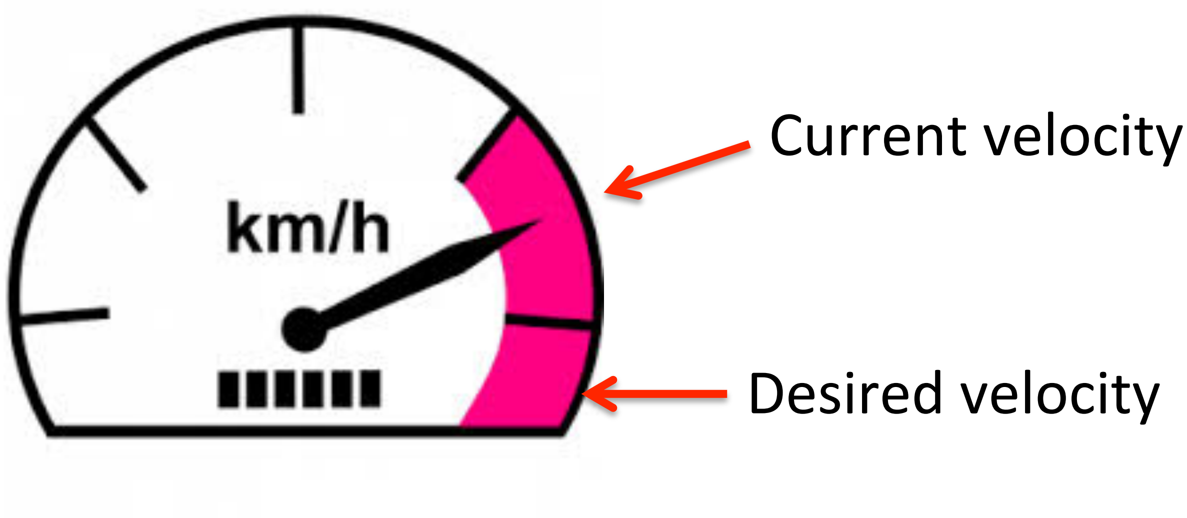
Fabry-Perot cavity

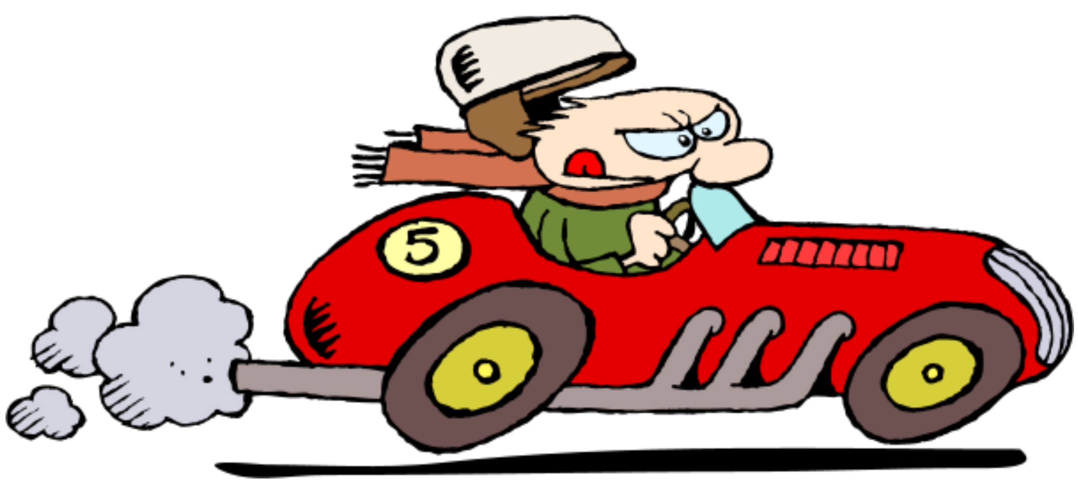


Ref G1600525

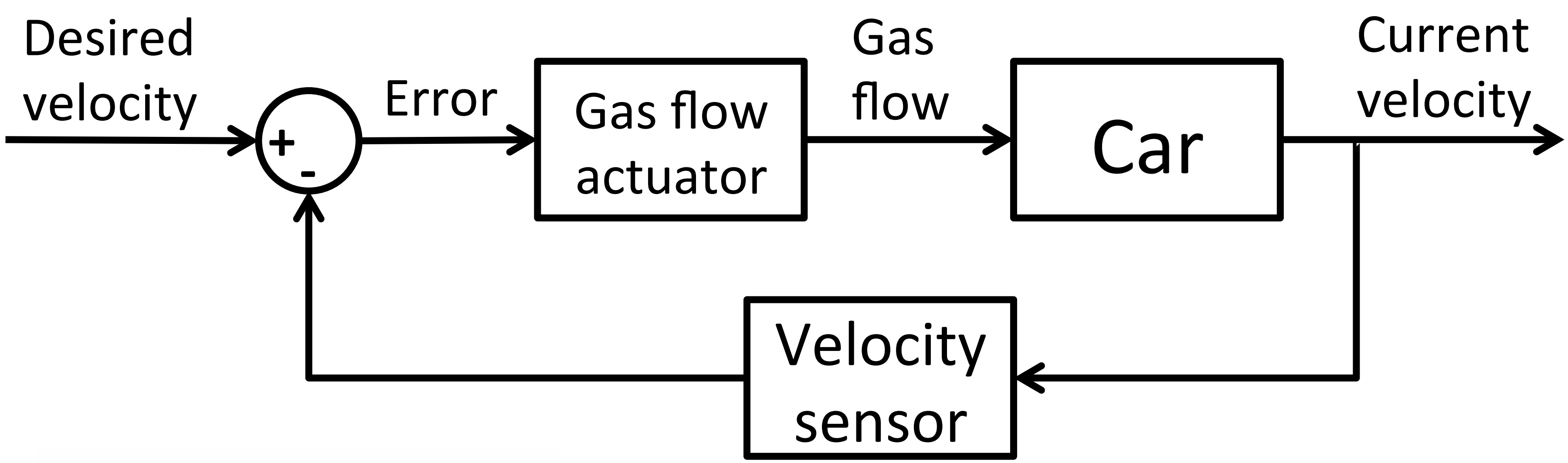
Example of a feedback loop

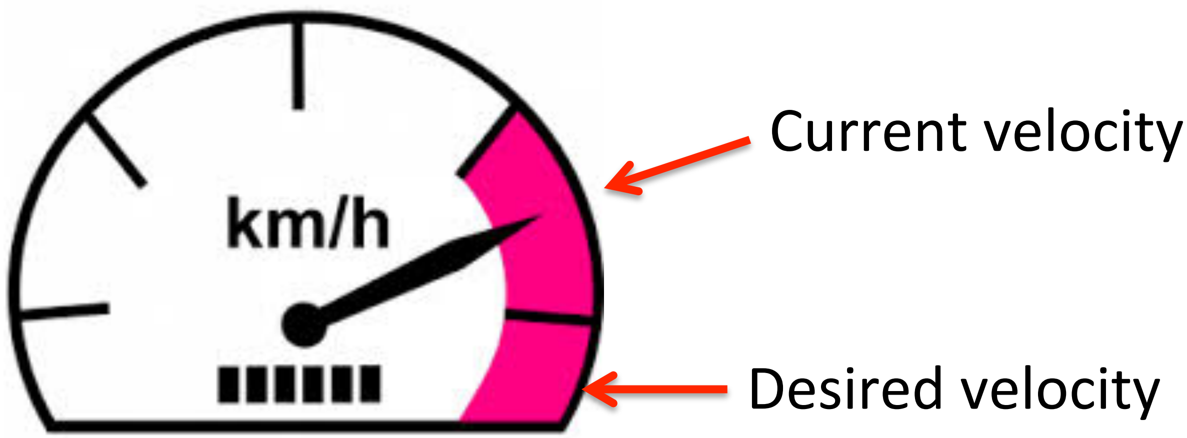
Cruise control



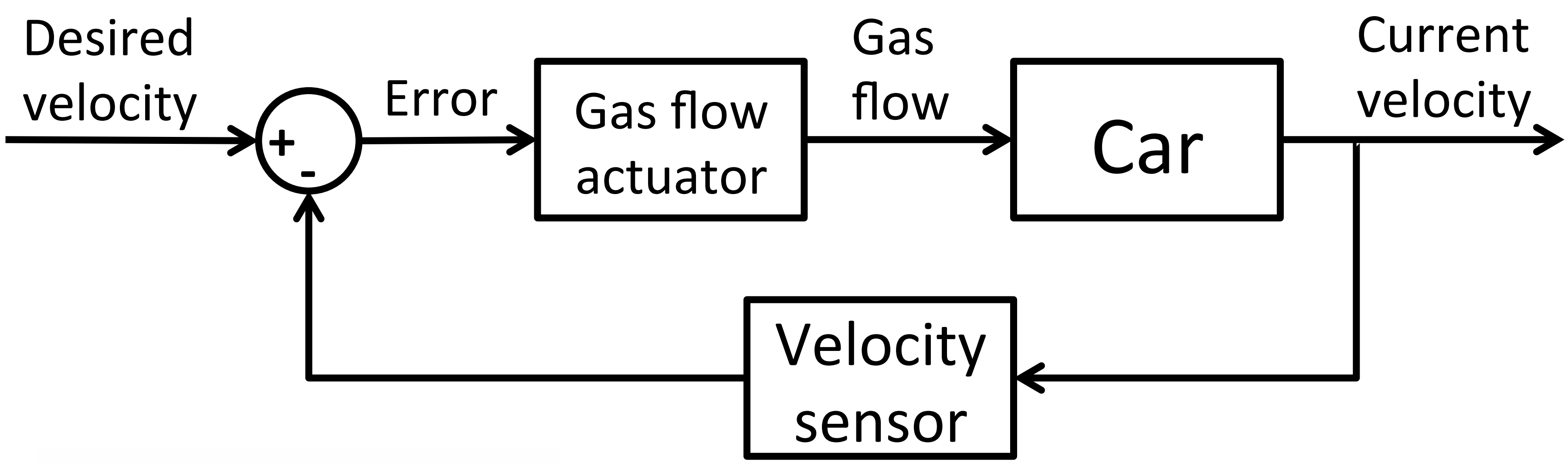


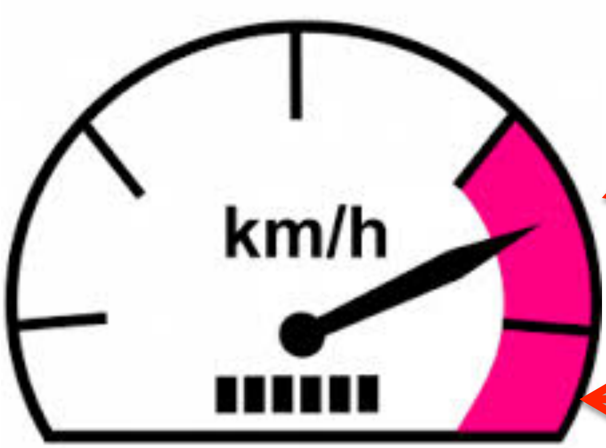
Feedback loop block diagram





LIGO Feedback loop block diagram

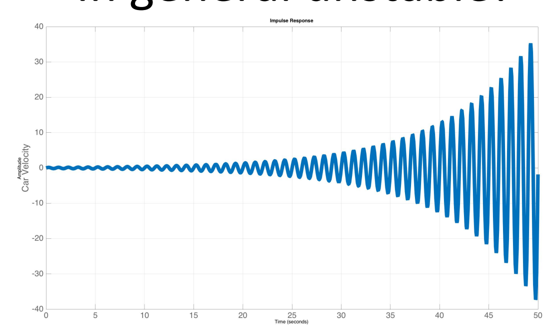




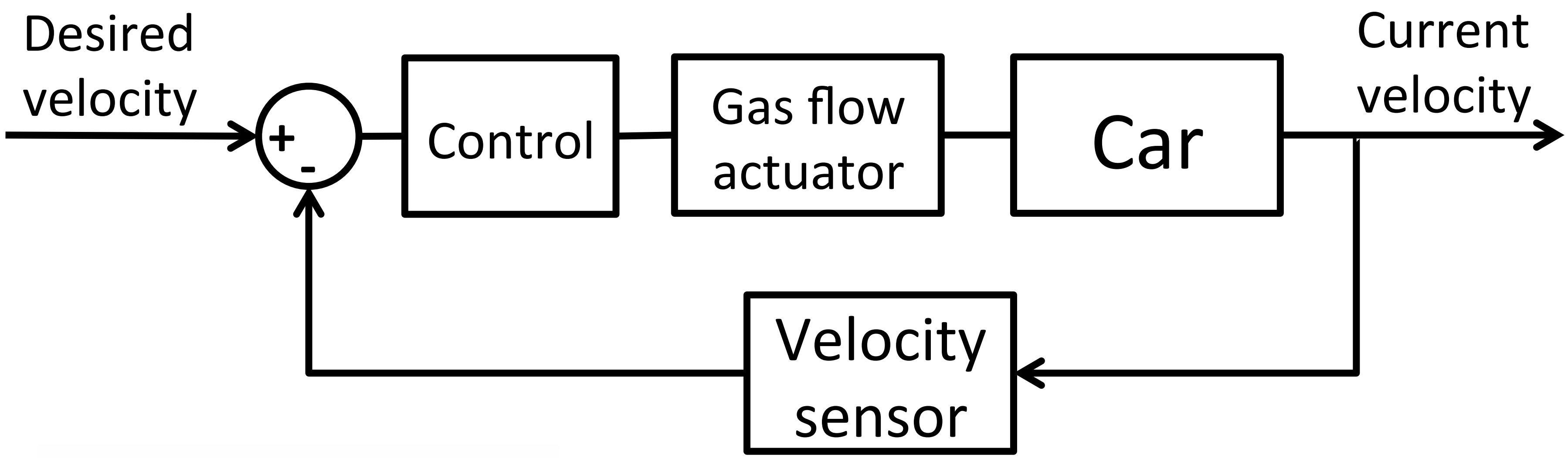
Current velocity

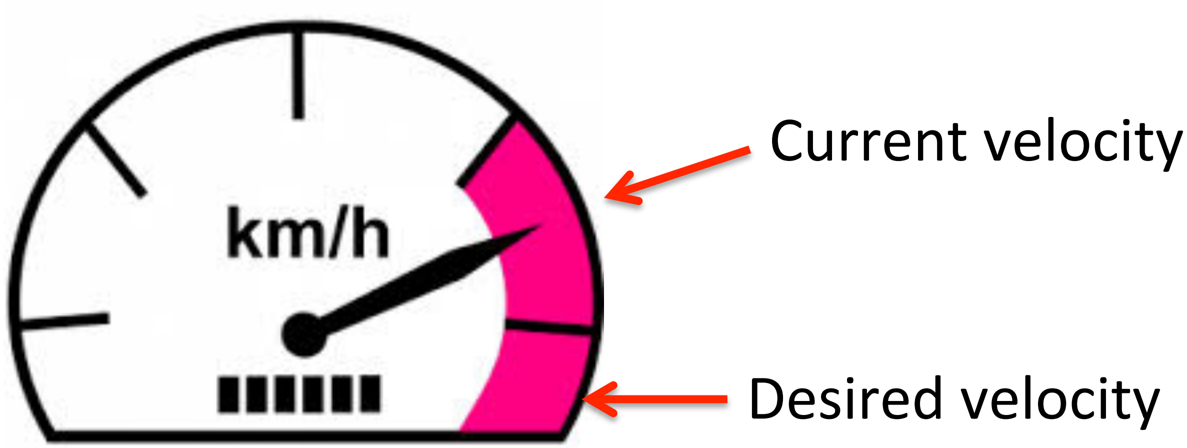
Desired velocity

In general unstable!



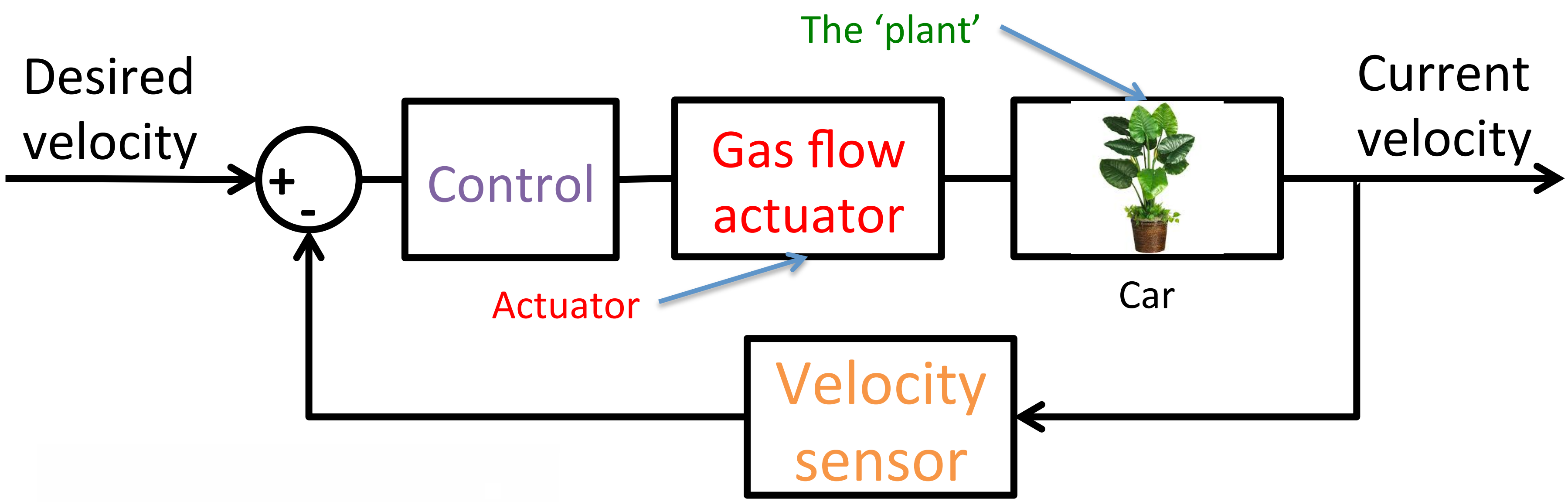
Feedback loop block diagram





- Stabilize with the addition of a control filter
- Also gives you parameters to tune the response

Basic system components



- Plant (car) – object to be controlled
- Sensor – measures the plant response
- Controller – sets the loop dynamics to achieve the desired behavior
- Actuator – drives the plant

Types of control

- Signal flow
  - Feedback
  - Feedforward

- Computation
  - Linear
  - Nonlinear

These will be defined on the next few slides.

Types of control

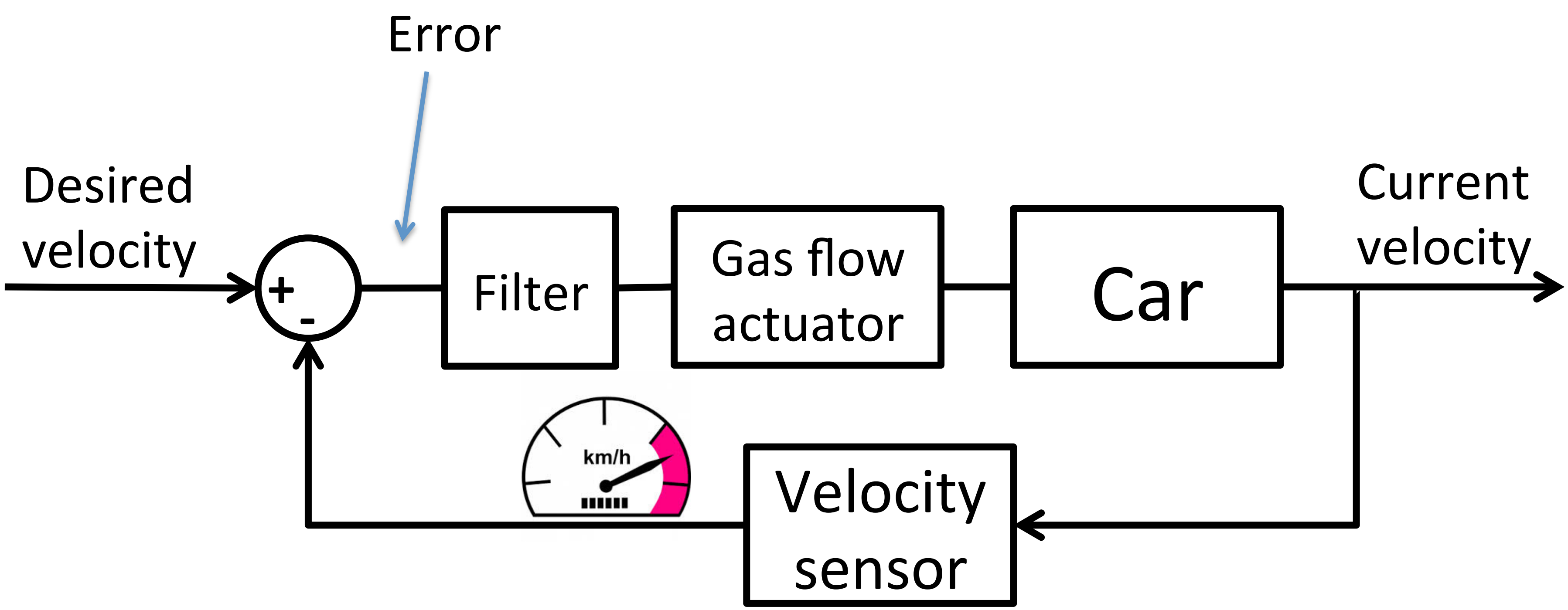
- Signal flow
  - Feedback
  - Feedforward

- Computation
  - Linear
  - ~~Nonlinear~~

Nearly all our controls are linear.
Some exceptions include:
- Acquiring cavity lock
- ESD actuation (F α V^2)
- ISI blend filter switching

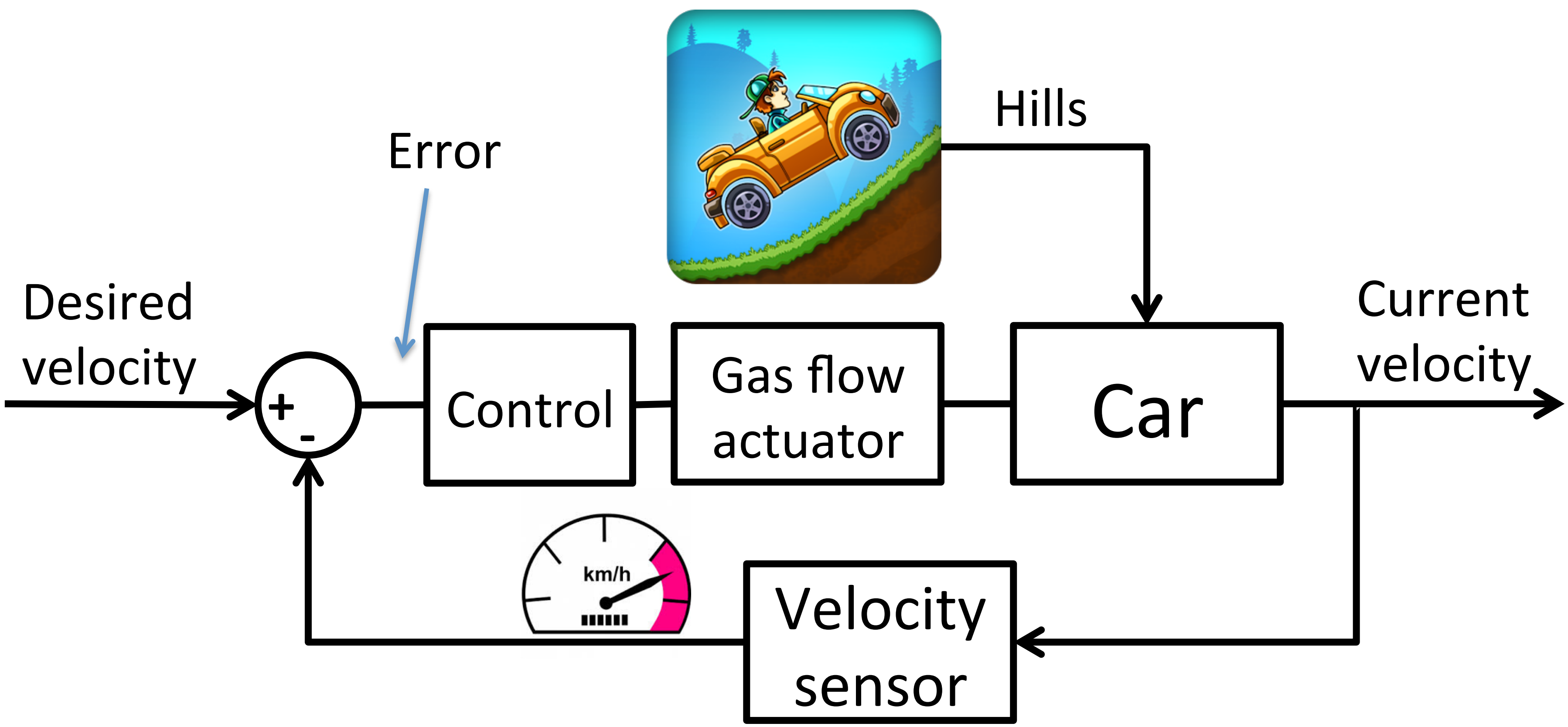
Types of control – feedback

- Feedback – **Reacts** to changes in the error after they occur.



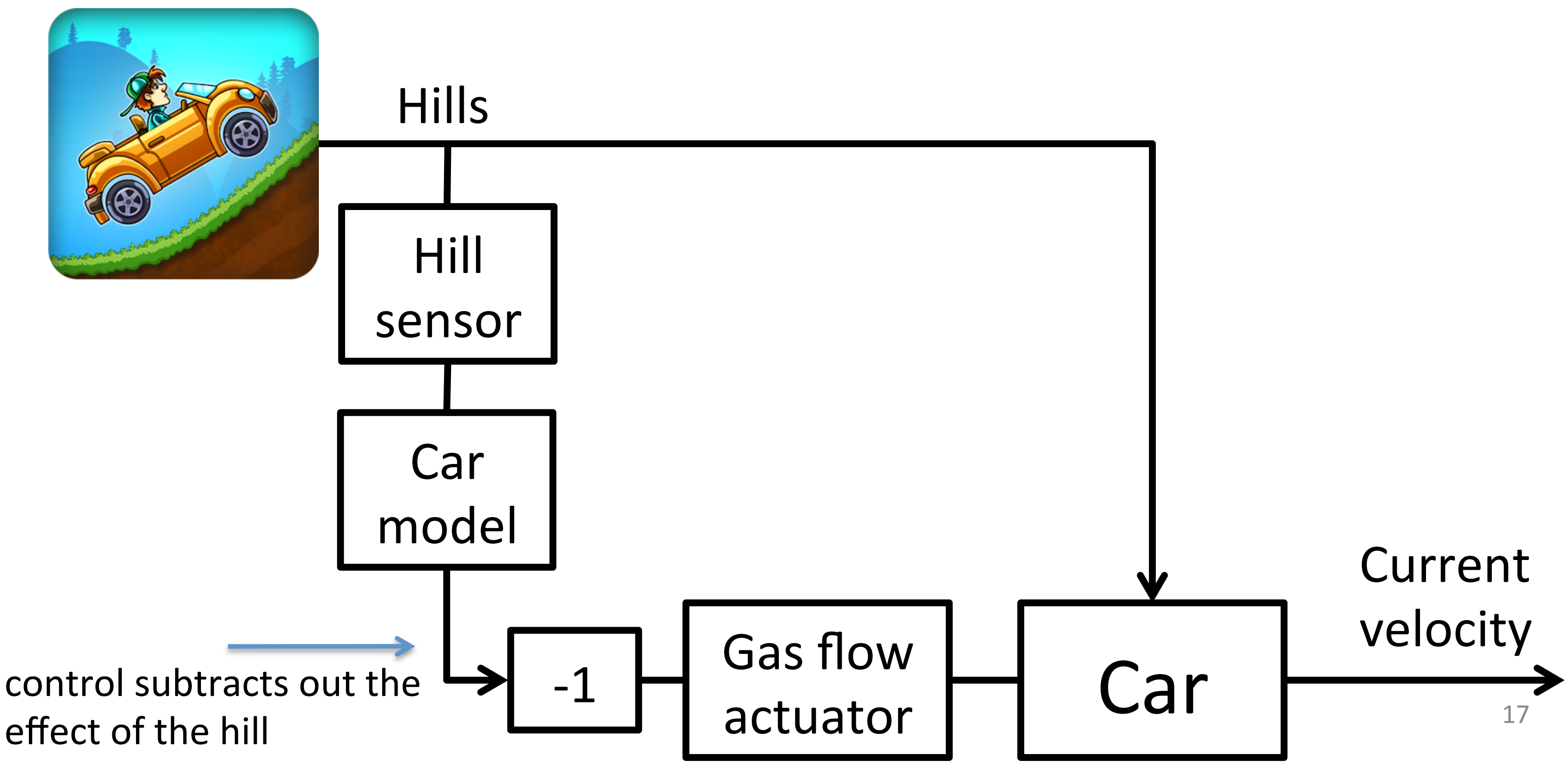
Types of control – feedback

- Feedback – **Reacts** to changes in the error after they occur. Could be unstable.

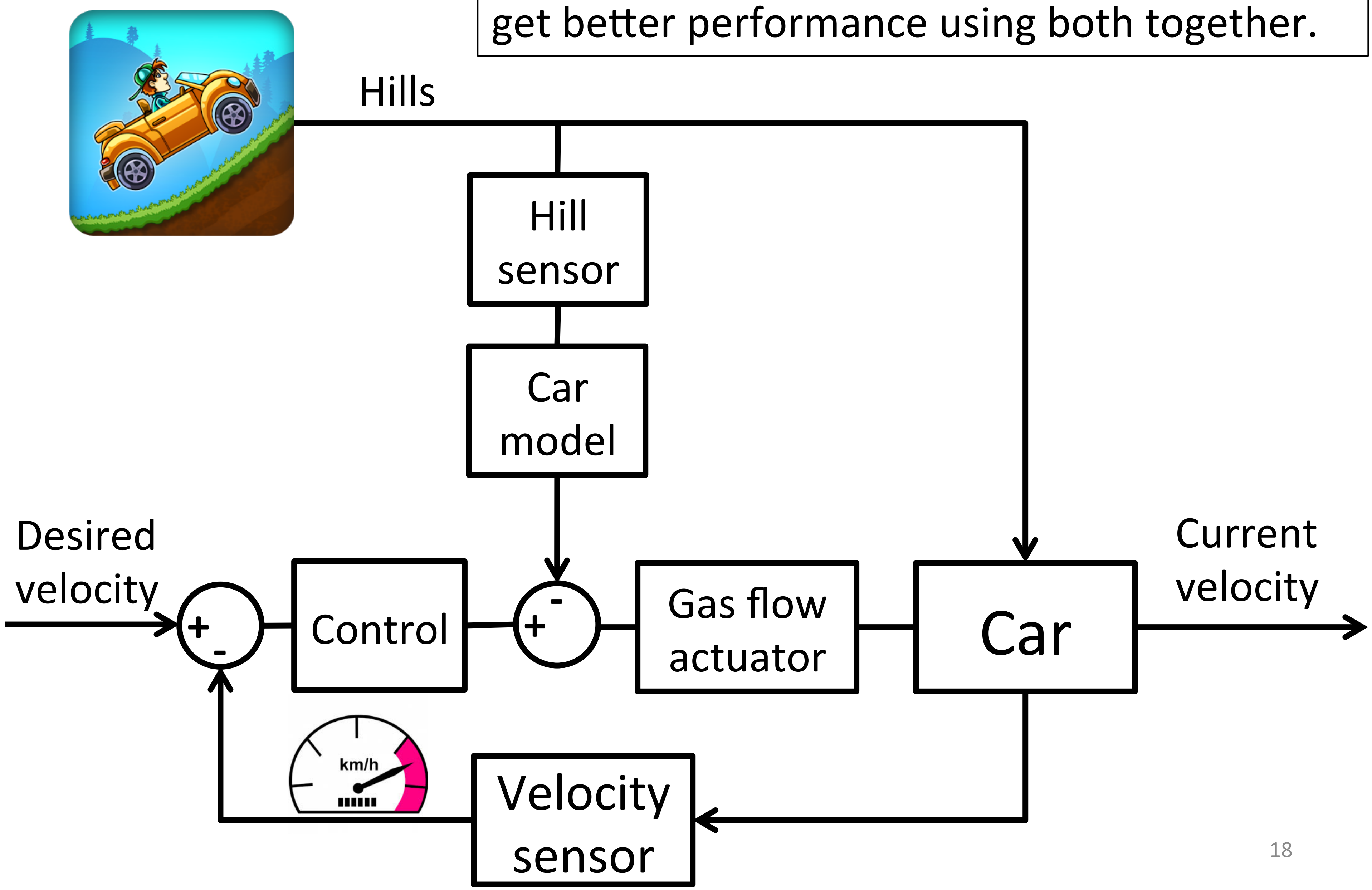


LIGO Types of control – feedforward

- Feedforward – **Predict** the future and correct in advance. Theoretically always stable.

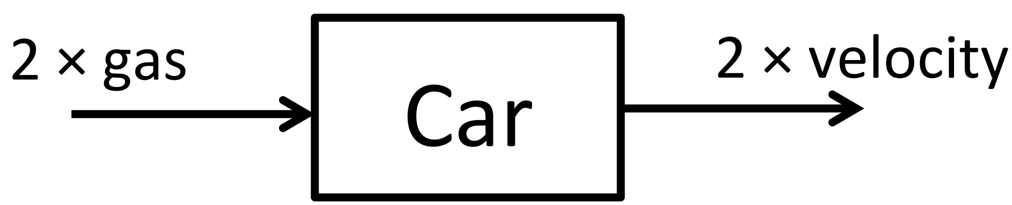


Feedback with feedforward
- Complimentary limitations, so often we can get better performance using both together.



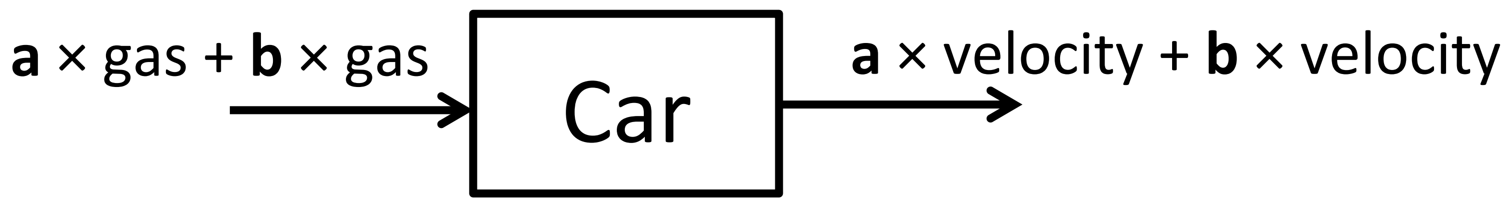
Types of control - linear

- Linear
  - Output is a linear combination of the inputs



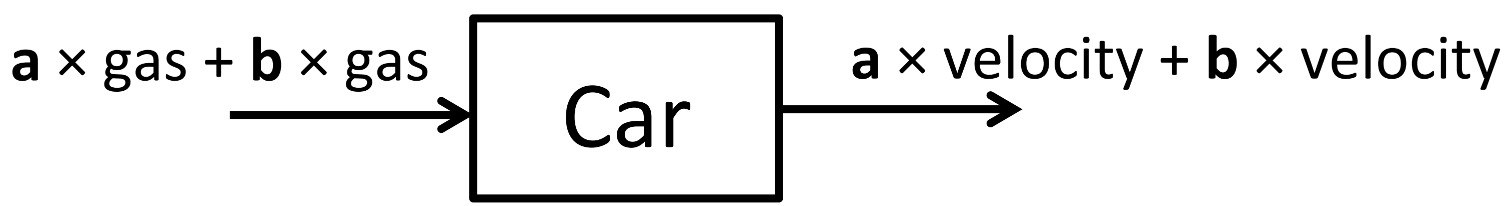
Types of control - linear

- Linear
  - Output is a linear combination of the inputs



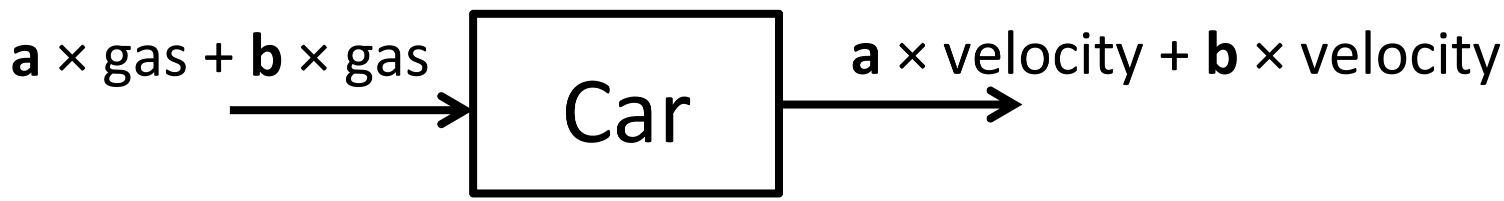
Types of control - linear

- Linear
  - Output is a linear combination of the inputs
  - Linear systems have a very well defined and rigorous control theory



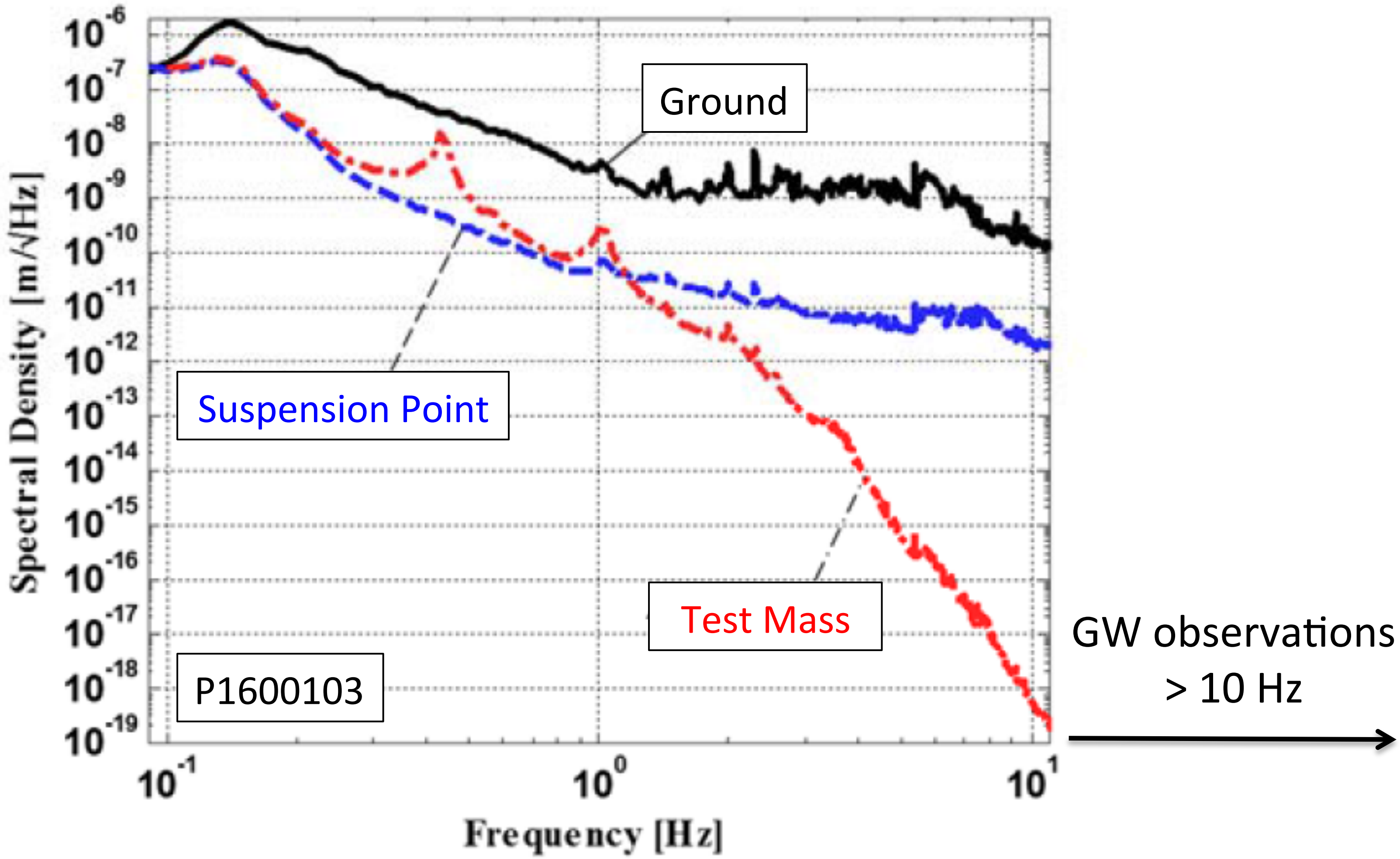
Types of control - linear

- Linear
  - Output is a linear combination of the inputs
  - Linear systems have a very well defined and rigorous control theory



*** Also, the output frequency = the input frequency ***

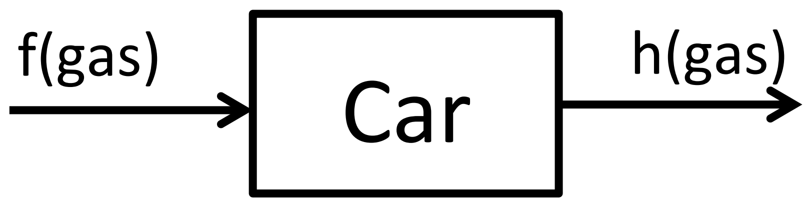
aLIGO Seismic Isolation Performance



Linearity is the reason we need good seismic isolation below 10 Hz, even though we're searching for GWs above 10 Hz. **Large amplitude** low frequency motion will have non-negligible 2nd order influences on the interferometer, causing **upconversion** to frequencies above 10 Hz. Mechanisms include nonlinear behavior of the cavity, scattered light, optic pitch and yaw.

LIGO Types of control - computation

- Nonlinear
  - Output is some general function of the input
  - No single theory for non-linear control



Nonlinear system examples

- Acquiring cavity lock – the sensor signal only exists intermittently
- Rockets – the mass gets smaller as the propellant is consumed
- Robotic arms – the moment of inertia depends on the arm's position

Lecture 1 – Part 2

System Modeling

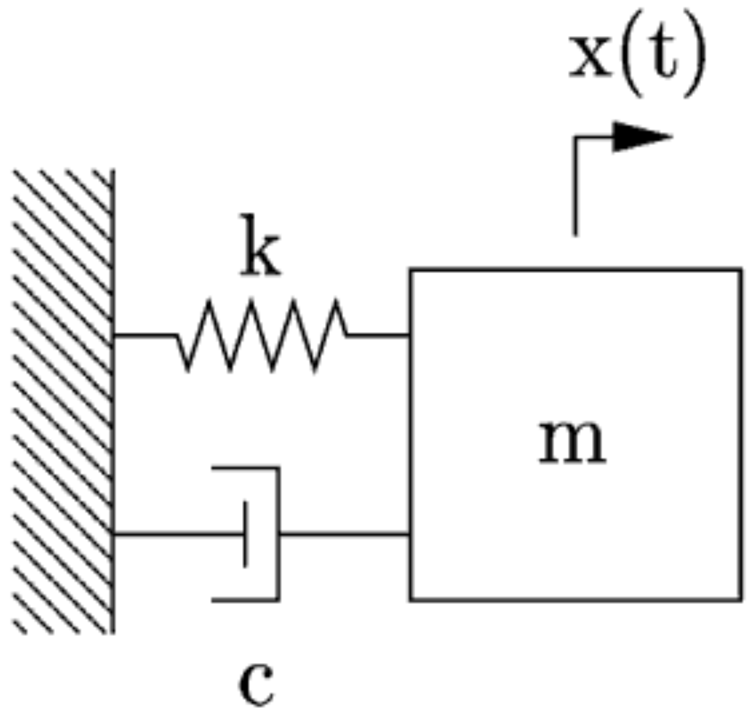
Lecture 1 – Part 2

System Modeling

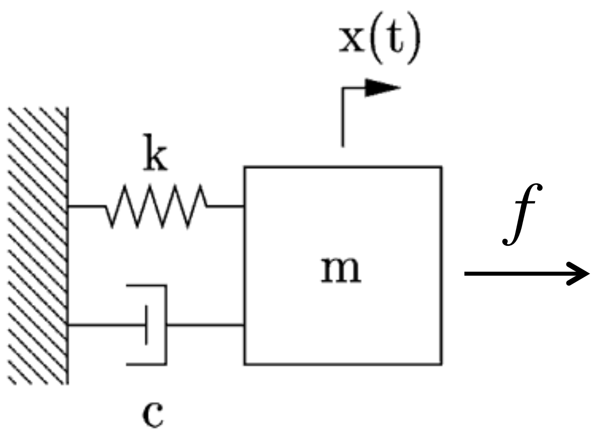
Recurring theme: more than 1 way to look at a system. All are equivalent, but each is useful in its own context.

System Models

Example – mass spring system



System Models

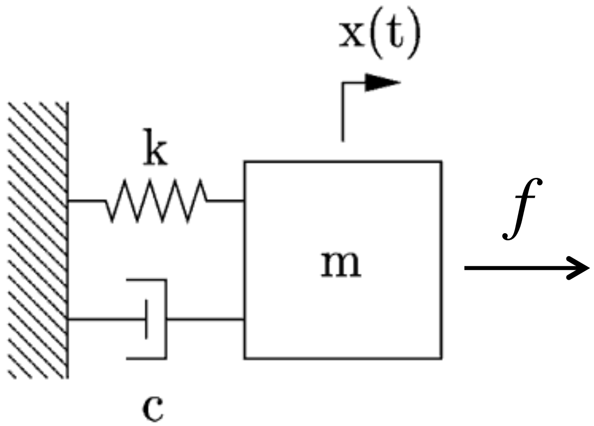


k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

System Models



Equation of motion

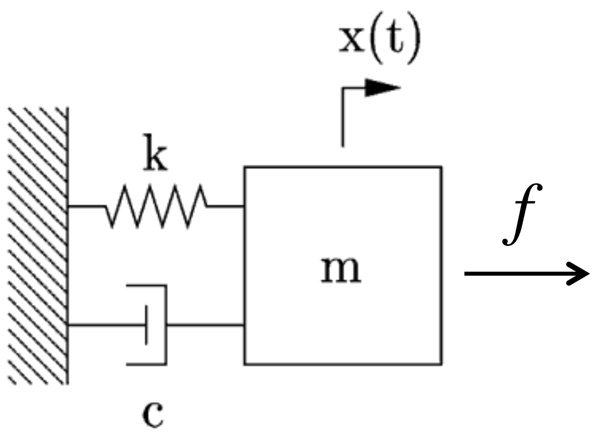
$$m\ddot{x} + c\dot{x} + kx = f$$

General solution

$$x = A_1 e^{(\sigma + i\omega)t} + A_2 e^{(\sigma - i\omega)t} + x_f$$

- A_1 and A_2 are constants whose values depend on the initial conditions
- x_f is a particular solution with the same form as f
- $\sigma \pm i\omega$ are the roots of the characteristic equation
  - σ is the decay time constant
  - ω is the natural frequency

System Models



k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

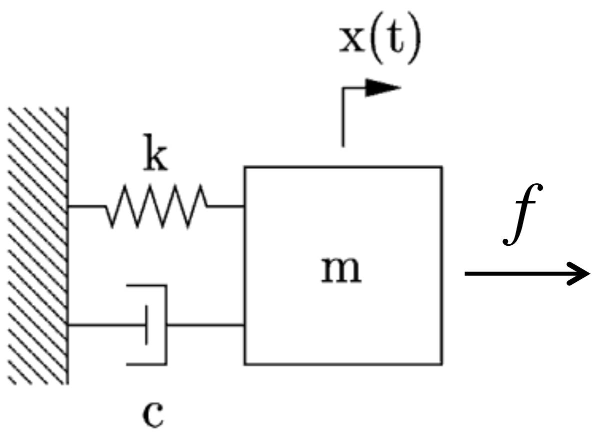
Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

Time domain
State space model

Frequency domain
Transfer functions

System Models



k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

Equation of motion

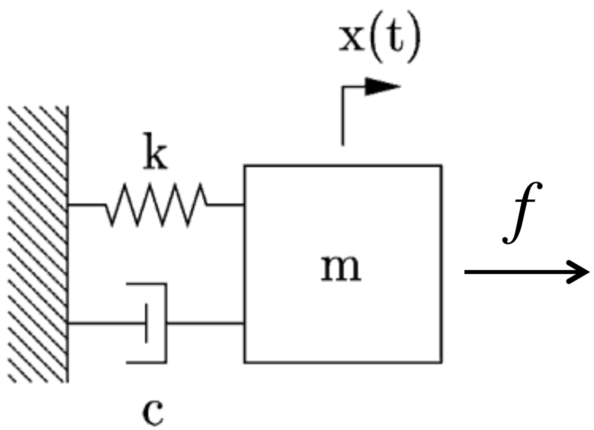
$$m\ddot{x} + c\dot{x} + kx = f$$

Time domain
State space model

Frequency domain
Transfer functions

Both formats are widely used, often at the same time.

Models – time domain



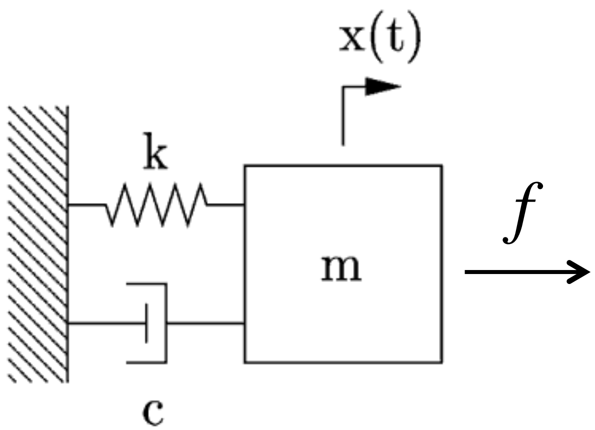
Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

- The **state space** form rewrites an N^{th} order differential equation as a system of N, 1^{st} order differential equations.

k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

Models – time domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

- The **state space** form rewrites an N^{th} order differential equation as a system of N, 1^{st} order differential equations.

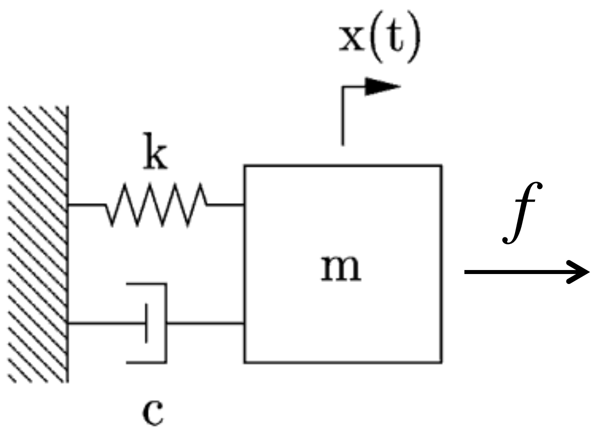
k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

System states: $x_1 = x$ displacement

$x_2 = \dot{x}$ velocity

* Mechanical systems have 2 states for every DOF: typically displacement & velocity

Models – time domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

- The **state space** form rewrites an N^{th} order differential equation as a system of N, 1^{st} order differential equations.

System states: $x_1 = x$

$x_2 = \dot{x}$

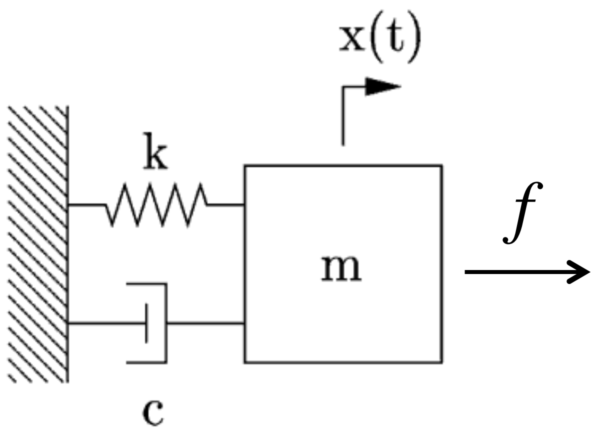
Matrix equation

$$\boxed{\dot{\vec{x}} = A\vec{x} + B\vec{u}}$$

k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

* Mechanical systems have 2 states for every DOF: typically displacement & velocity

Models – time domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

- The **state space** form rewrites an N^{th} order differential equation as a system of N, 1^{st} order differential equations.

k -> stiffness
c -> viscous damping
m -> mass
f -> external force
x -> mass position

System states: $x_1 = x$

$$x_2 = \dot{x}$$

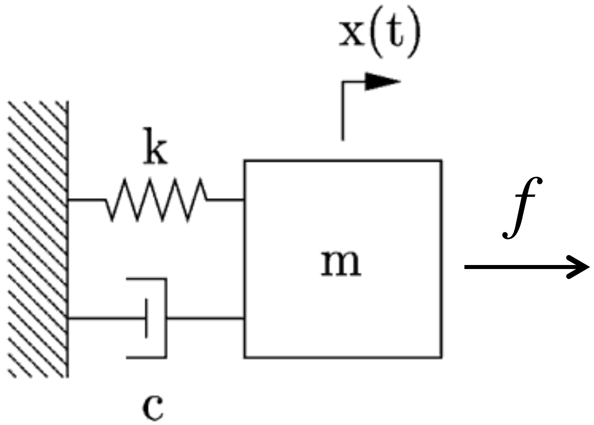
Matrix equation

$$\boxed{\dot{\vec{x}} = A\vec{x} + B\vec{u}}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} f$$

* Mechanical systems have 2 states for every DOF: typically displacement & velocity

Models – time domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

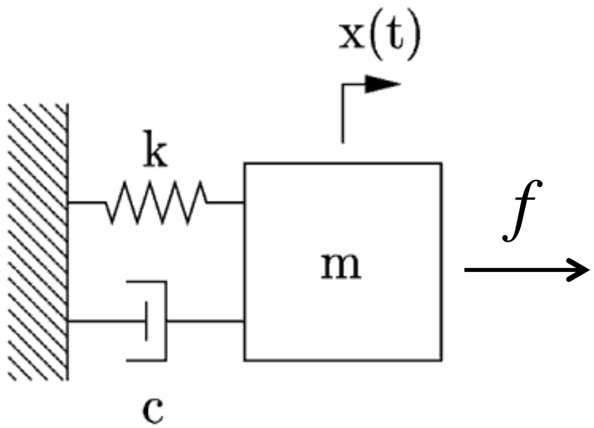
$$\dot{\vec{x}} = A\vec{x} + B\vec{u}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} f$$

A matrix: internal system dynamics

B matrix: external input

Models – time domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

$$\dot{\vec{x}} = A\vec{x} + B\vec{u}$$ ← System dynamics

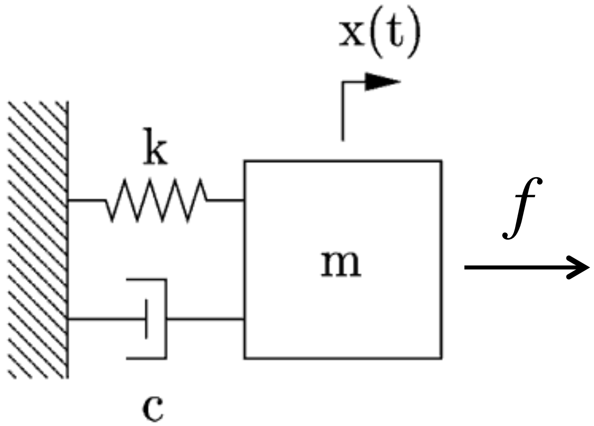
$$\boxed{\vec{y} = C\vec{x} + D\vec{u}}$$ ← Sensing function

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} f$$ Ex. Displacement sensor

The complete state space form is given by the **A**, **B**, **C**, & **D** matrices.

- A is the dynamic behavior matrix
- B is the input matrix.
- C is the output matrix
- D directly connects the input to the output (if such a connection exists)

Models – frequency domain



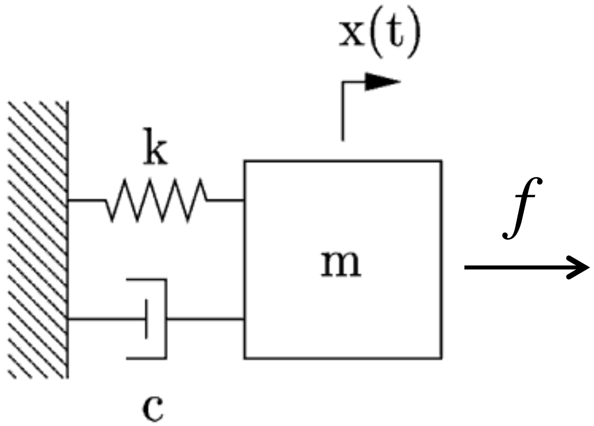
Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

Convert EOM to the frequency domain with the Laplace transform.
Time derivative -> Laplace variable *s*

Laplace transform $$x\left(ms^2 + cs + k\right) = f$$

Models – frequency domain



Equation of motion

$$m\ddot{x} + c\dot{x} + kx = f$$

Convert EOM to the frequency domain with the Laplace transform.

Time derivative -> Laplace variable *s*

Laplace transform $$x\left(ms^2 + cs + k\right) = f$$

Force to displacement transfer function $$\frac{x}{f} = \frac{1}{ms^2 + cs + k}$$

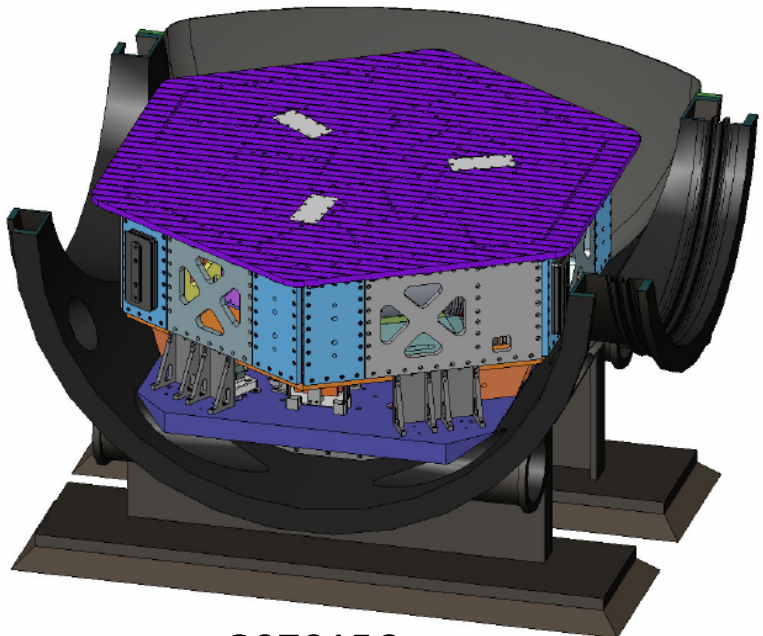
Models – time & frequency domain comparison

	Time domain (SS)	Frequency domain (TF)
Domain	$\dot{\vec{x}} = A\vec{x} + B\vec{u}$ $\vec{y} = C\vec{x} + D\vec{u}$	$\frac{x}{f} = \frac{1}{ms^2 + cs + k}$
Solution exponents $\sigma \pm i\omega$	*eig(A)*	Poles of *x/f* -> roots of $ms^2 + cs + k$
System order	# of rows of *A, or the # of states*	# of poles of *x/f*, or the order of the denominator's polynomial

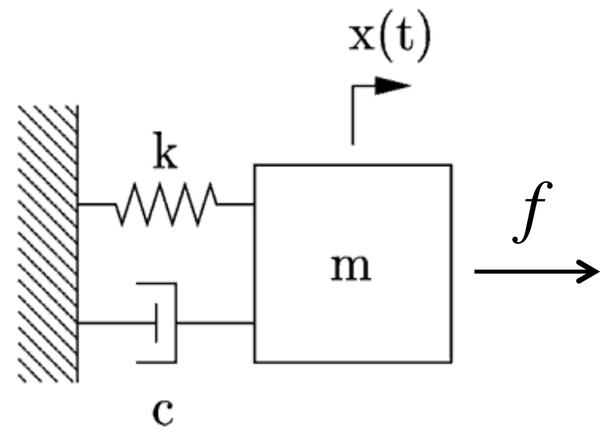
Models – time & frequency domain comparison

	Time domain (SS)	Frequency domain (TF)
Domain	$\dot{\vec{x}} = A\vec{x} + B\vec{u}$ $\vec{y} = C\vec{x} + D\vec{u}$	$\frac{x}{f} = \frac{1}{ms^2 + cs + k}$
When is each more useful?	- MIMO (multi-input, multi-output) - Has many states - Various matrix operations are useful for studying its properties - Some 'modern' controls techniques are defined in SS - Easy to numerically integrate	- Examining the frequency content of the system's behavior - Designing SISO (single-input, single output) control filters
Example system: quadruple pendulum test mass suspension	The model is defined as a SS system, since it is MIMO and has many states.	The various controllers are designed by extracting TFs between individual inputs and outputs from the SS model.

Example system – HAM ISI



G070156



$$m\ddot{x} + c\dot{x} + kx = f$$

Physical parameters from G070156
- k -> 125770 N/m
- m = 1900 kg
- c -> 3000 N/(m/s)   (guess)

State space model

$$\dot{\vec{x}} = A\vec{x} + B\vec{u}$$

$$\vec{y} = C\vec{x} + D\vec{u}$$

Example system – HAM ISI

State space model

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} f \quad \Rightarrow \quad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -66.2 & -1.6 \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 5.3e-4 \end{bmatrix}}_{B} f$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{[0]}_{D} f$$

Roots of the system

$$eig(A) = -0.814 \pm 8.10i$$

Matlab code for state space format:

```
HAMISI_SS = ss(A,B,C,D);
```

3 ways to characterize these models

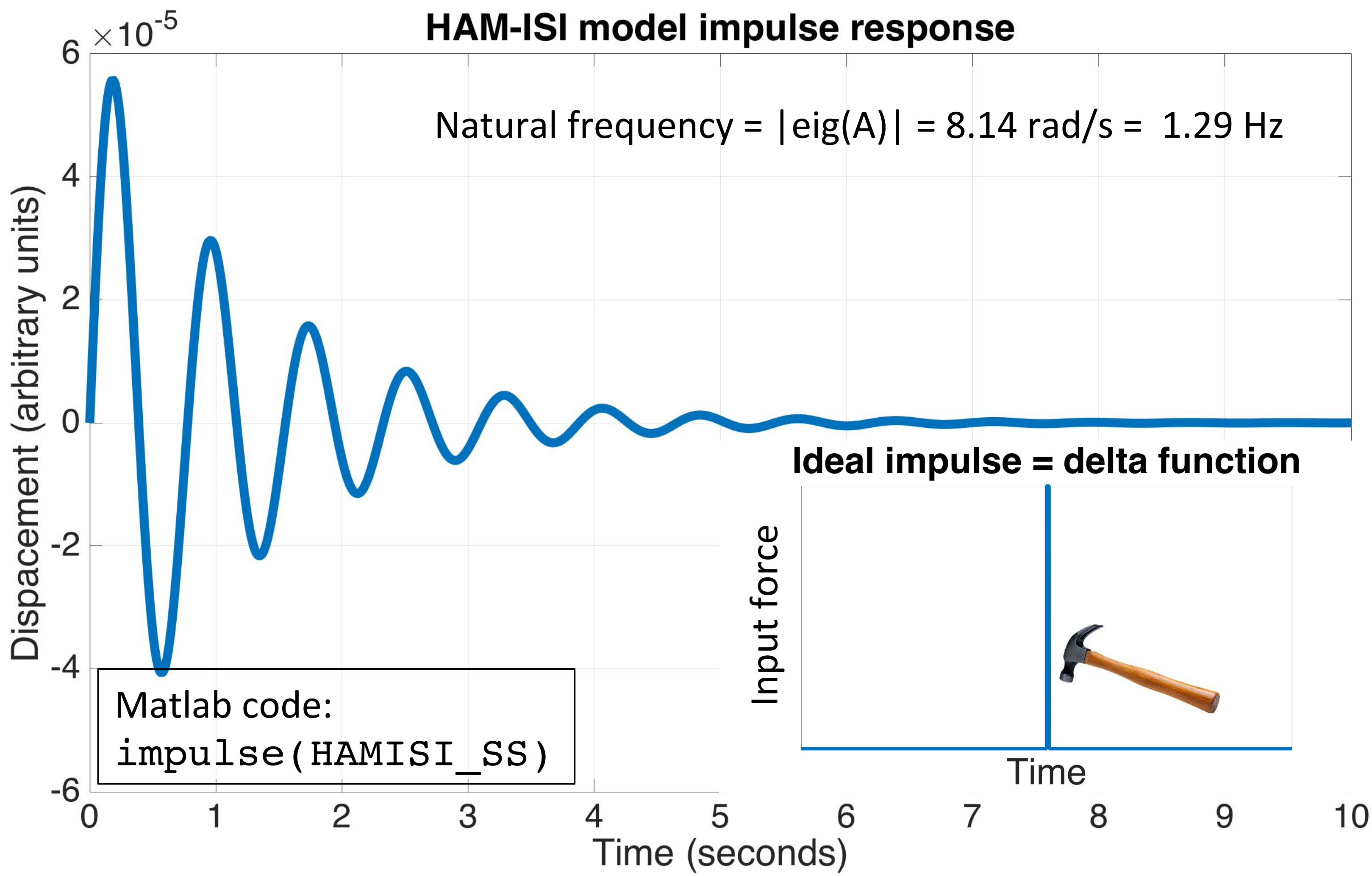
- Impulse response (or the similar step response)
- Complex plane (zero-pole map)
- * Bode plot of the transfer function

These are all equivalent, which we'll see on the following slides.

* Bode plots are used more extensively than anything else, but all are useful

Example system – HAM ISI

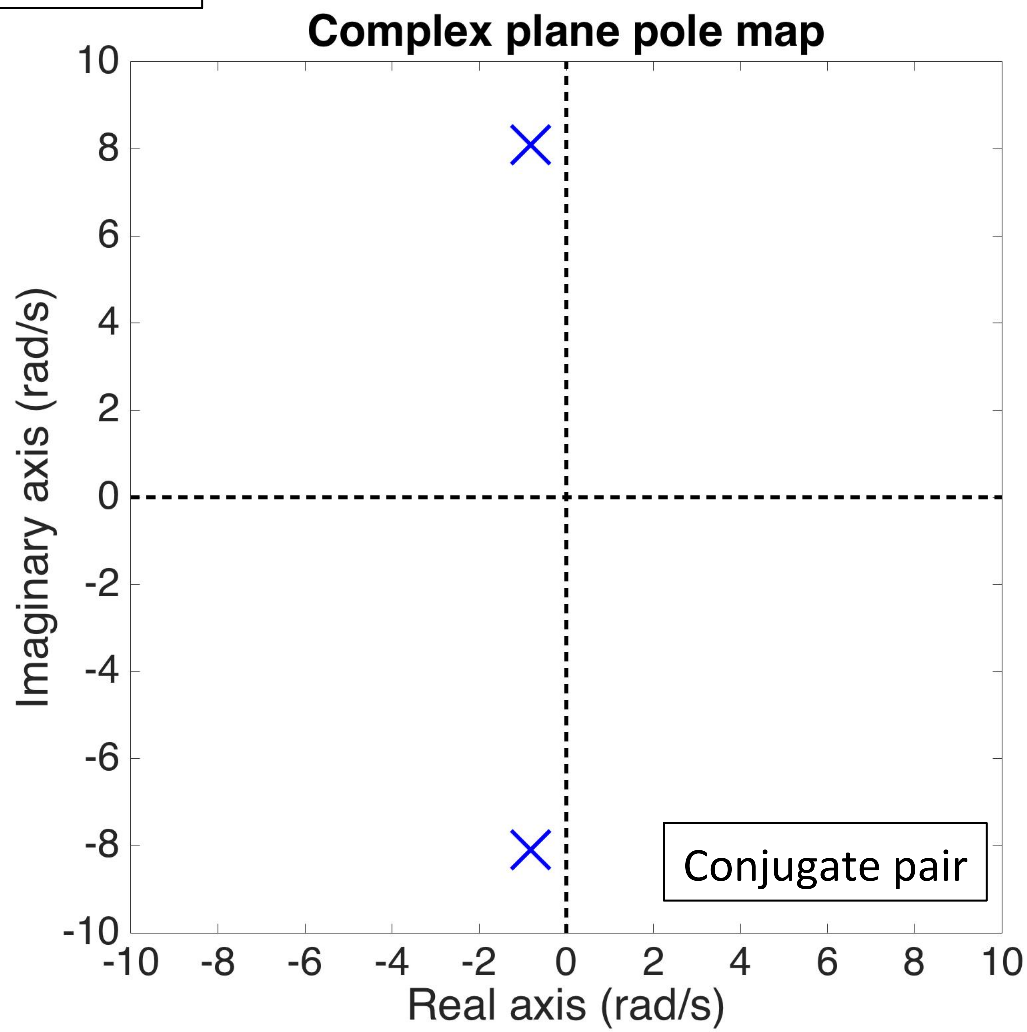
One way to characterize the system is to plot the impulse response



Example system – HAM ISI

Matlab code for a pole-zero map:
```
pzmap(HAMISI_SS)
```

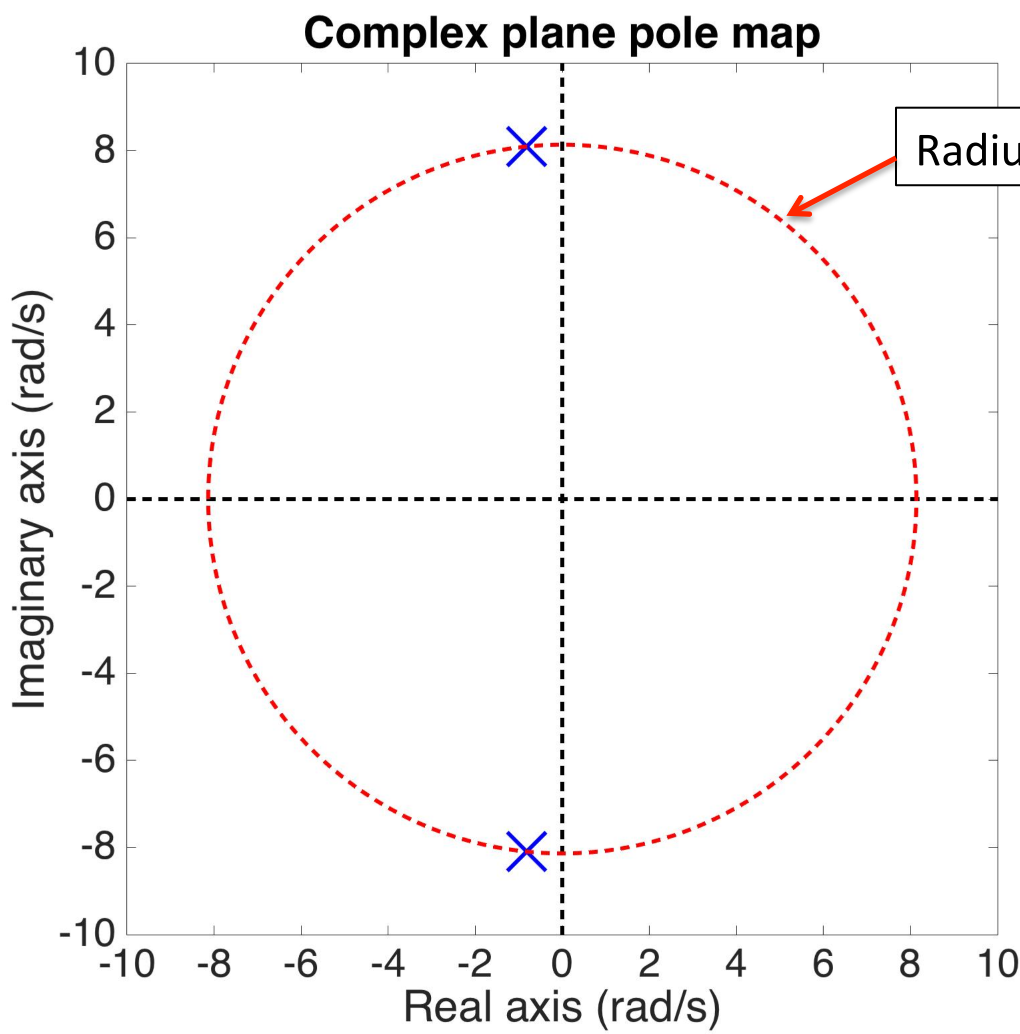
$$eig(A) = -0.814 \pm 8.10i$$



Another way to characterize the system is to plot the eigenvalues (poles) in the complex plane

Example system – HAM ISI

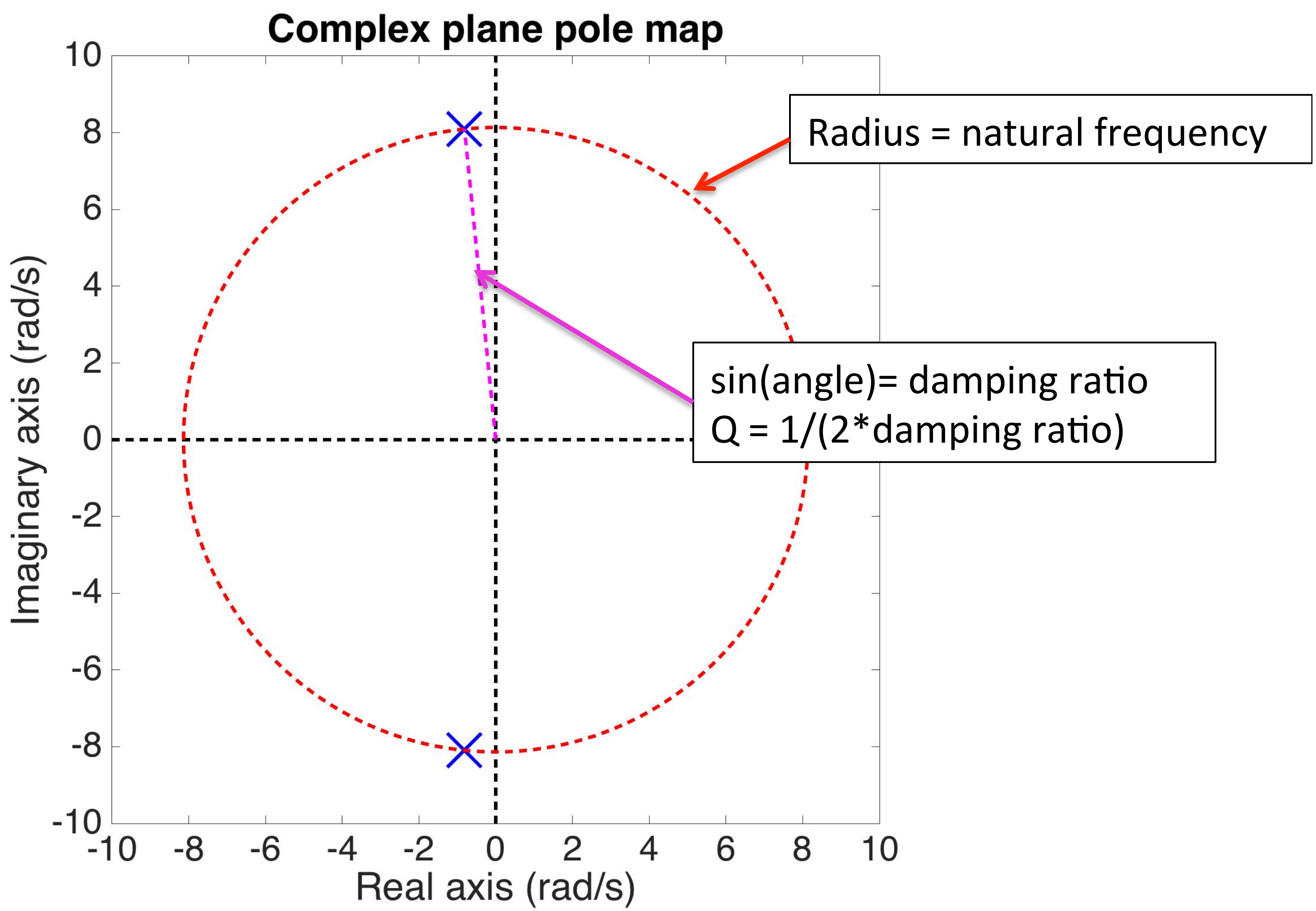
$$eig(A) = -0.814 \pm 8.10i$$



Radius = natural frequency

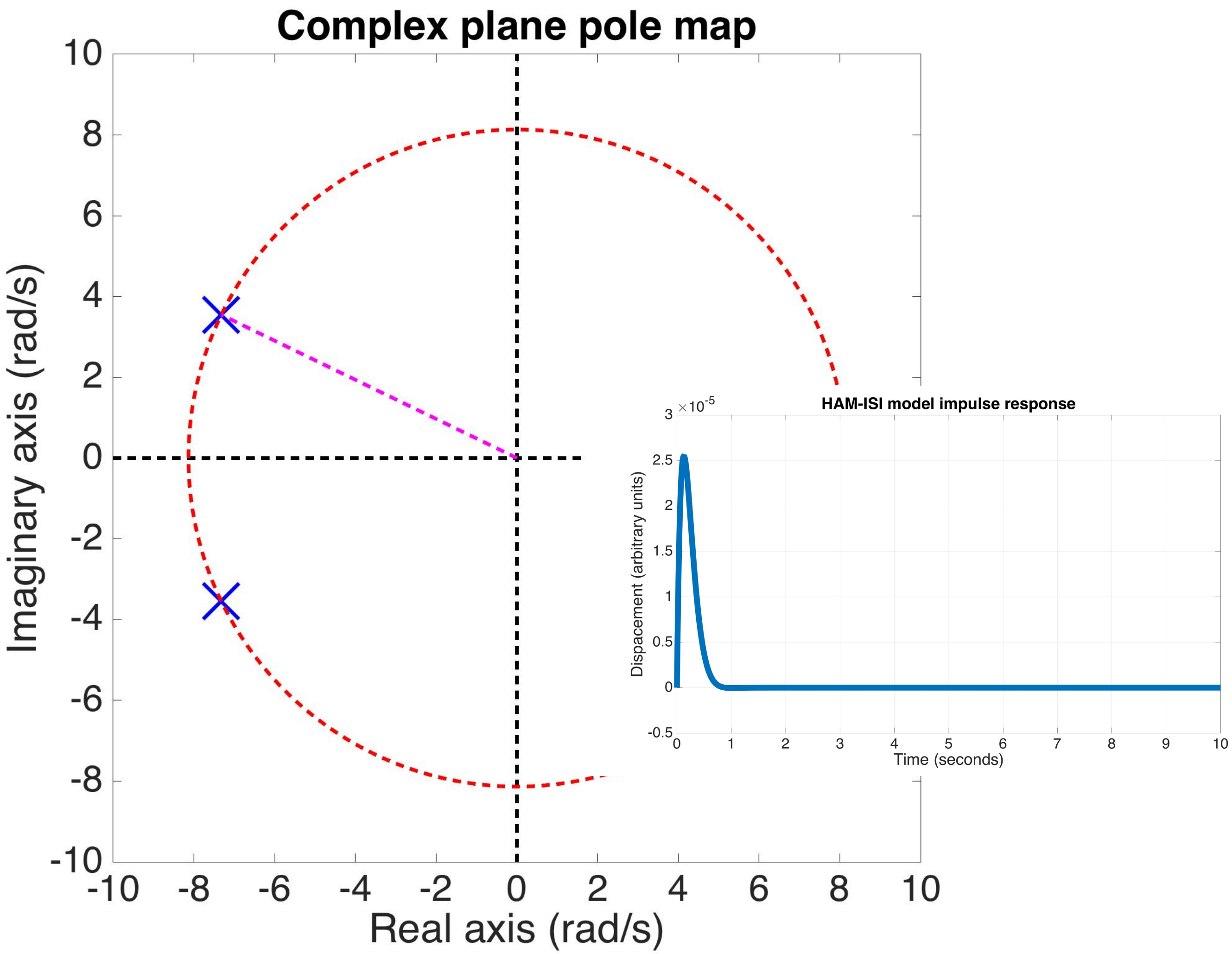
Example system – HAM ISI

$$eig(A) = -0.814 \pm 8.10i$$



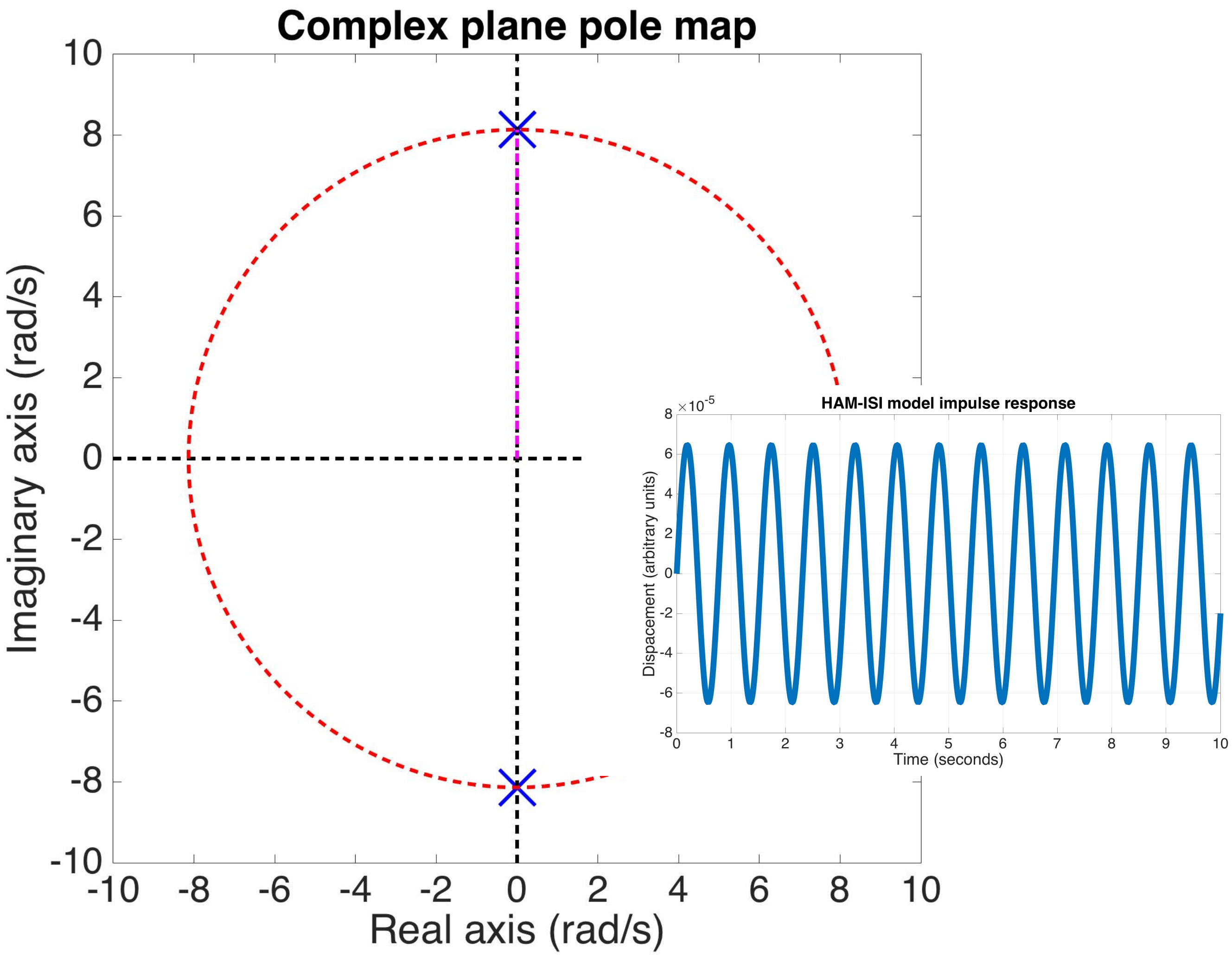
Example system – HAM ISI

$$eig(A) = -7.32 \pm 3.55i$$



Example system – HAM ISI

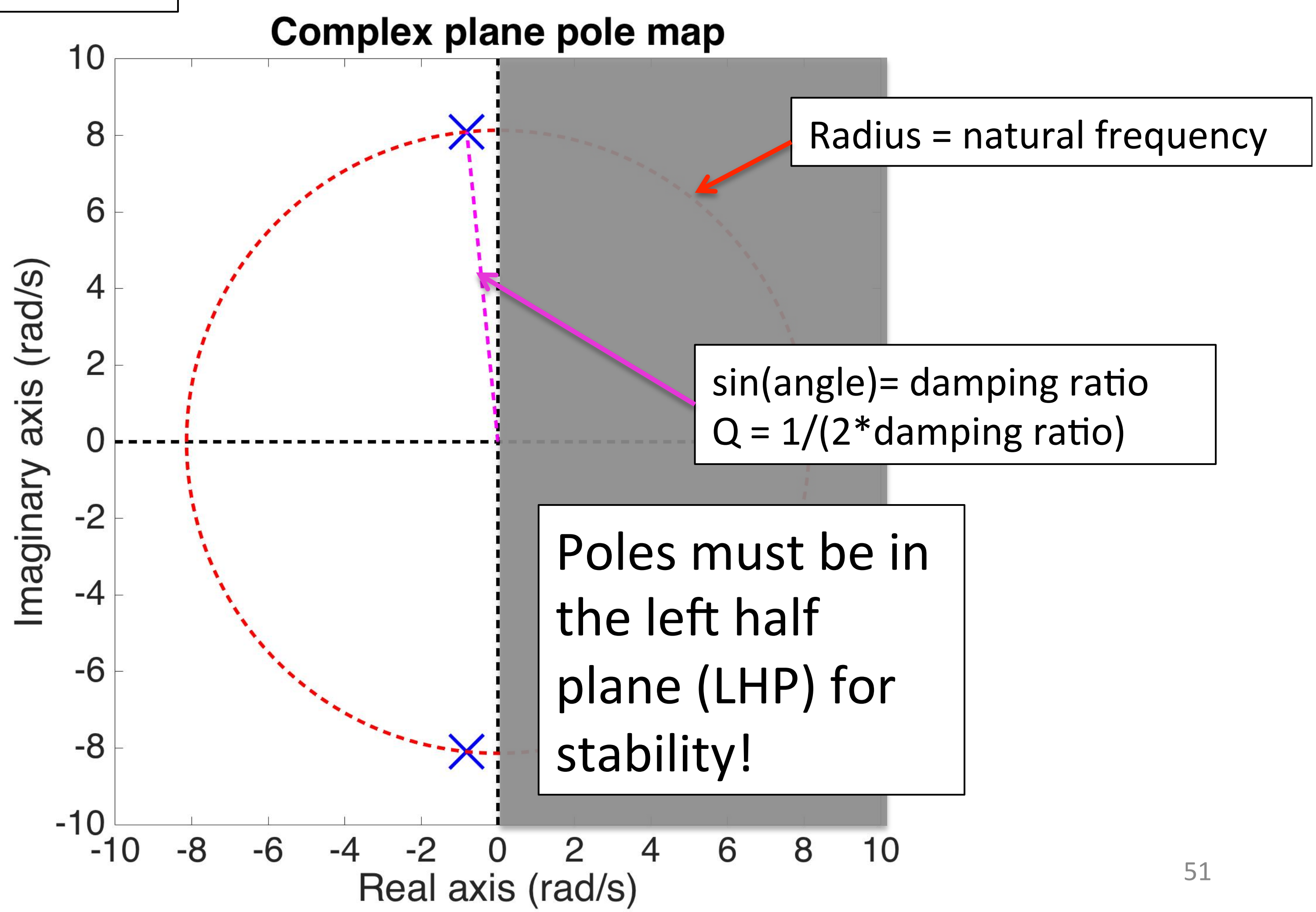
$$eig(A) = 0 \pm 8.14i$$



Example system – HAM ISI

Matlab code for a pole-zero map:
`pzmap(HAMISI_SS)`

$$eig(A) = -0.814 \pm 8.10i$$



Example system – HAM ISI

Yet another way to characterize the system is to plot the transfer function

$$\frac{x}{f} = \frac{1}{ms^2 + cs + k}$$

$$s = 0 + i\omega$$

$$\omega = 2\pi f$$

This substitution converts the Laplace transform to the Fourier transform

$$\frac{x}{f} = \frac{1}{\left[125770 - 1900(2\pi f)^2\right] + 3000(2\pi f)i}$$

Poles of the system

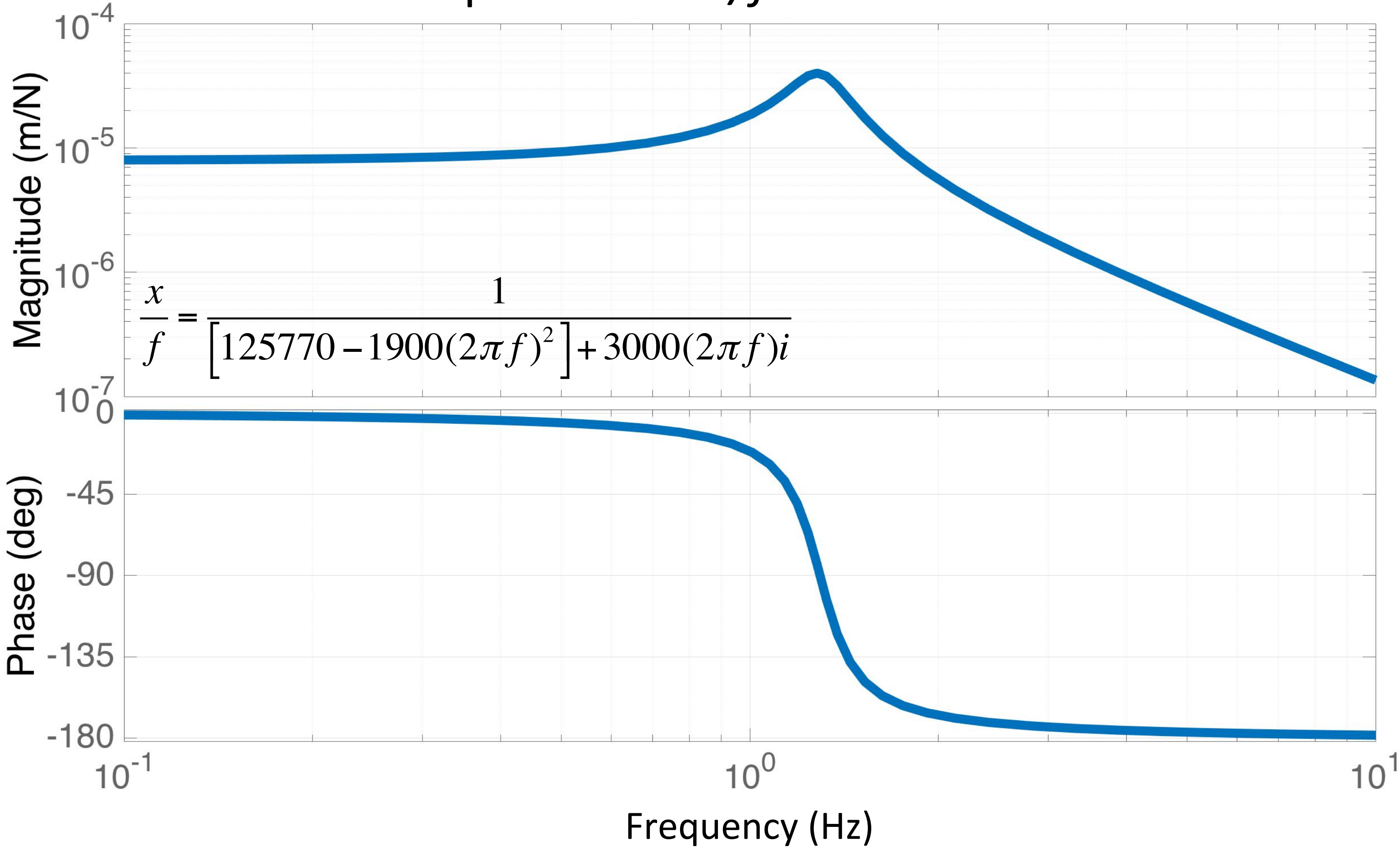
$$roots(ms^2 + cs + k) = eig(A) = -0.814 \pm 8.10i$$

Matlab code for transfer function format:

```
HAMISI_TF = tf(1,[1900,3000,125770]);
```

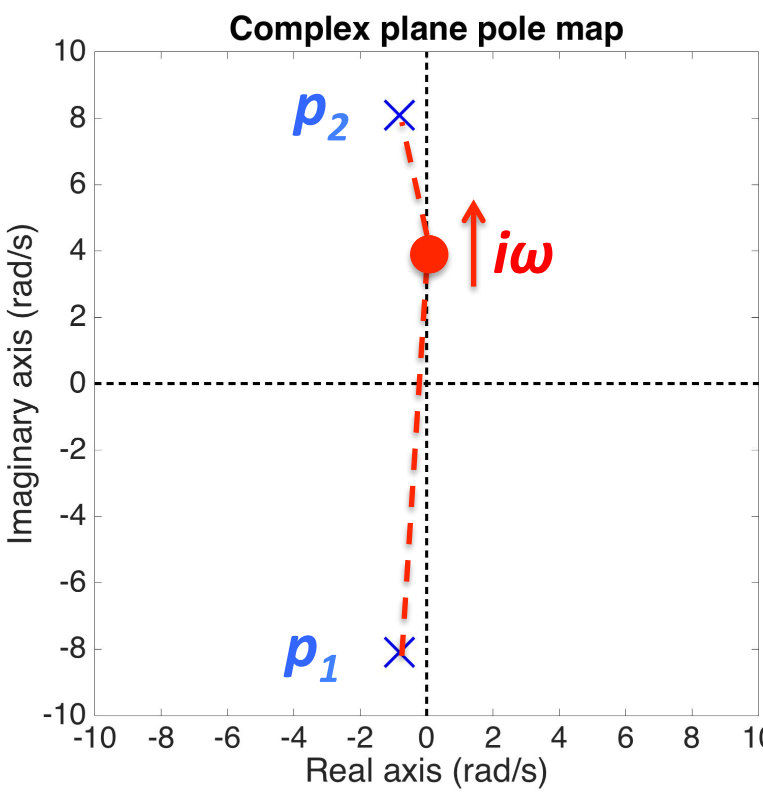
Example system – HAM ISI

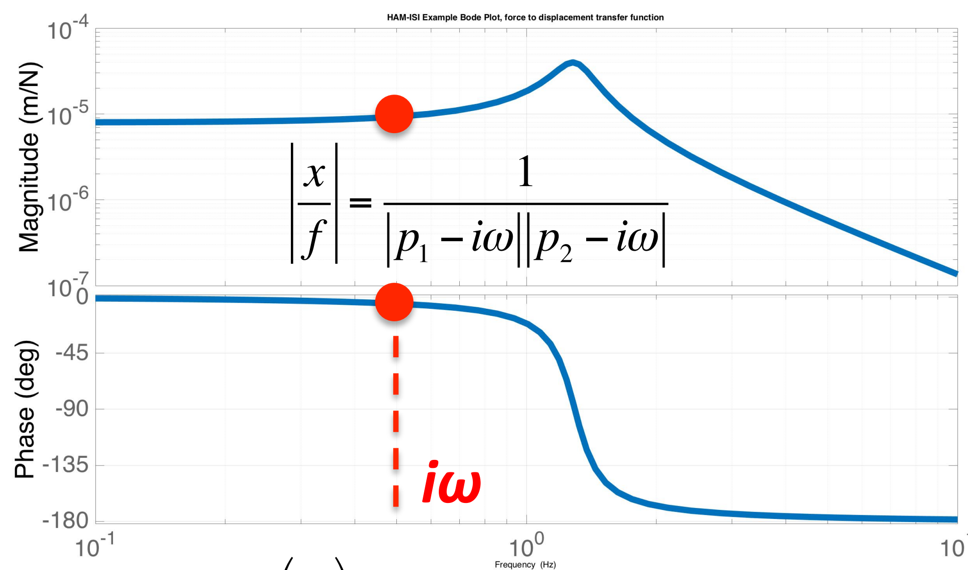
Bode plot of the *x/f* transfer function



Pole map vs Bode plot

$$\frac{x}{f}=\frac{1}{\left[125770-1900(2\pi f)^2\right]+3000(2\pi f)i}$$





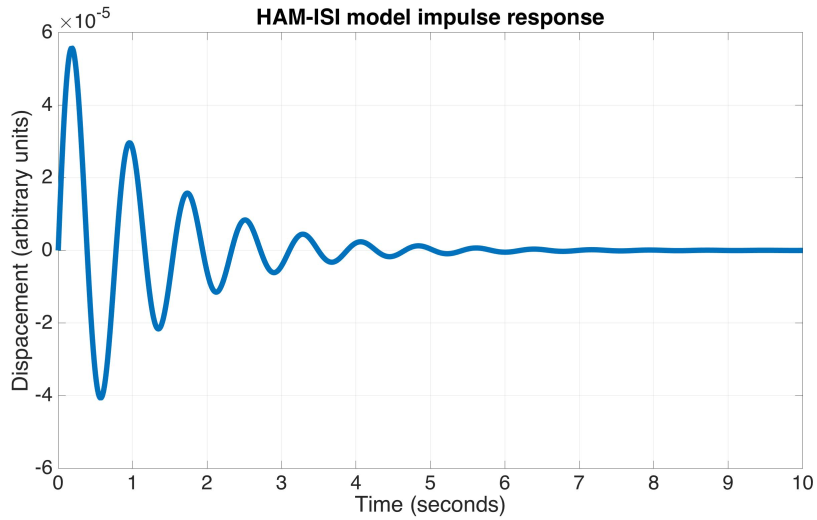
$$\left|\frac{x}{f}\right|=\frac{1}{|p_1-i\omega||p_2-i\omega|}$$

$$angle\left(\frac{x}{f}\right)=angle\left(p_1-i\omega\right)+angle\left(p_2-i\omega\right)$$

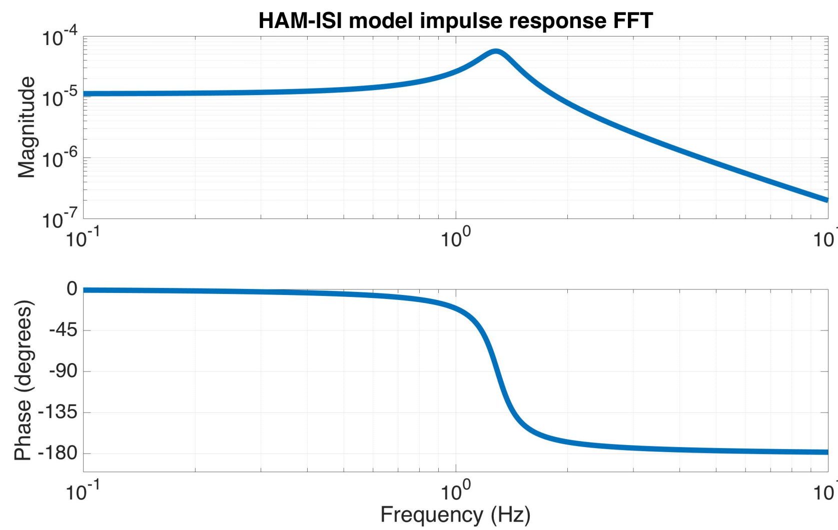
Frequency (Hz)

Impulse vs Bode plot

Impulse response



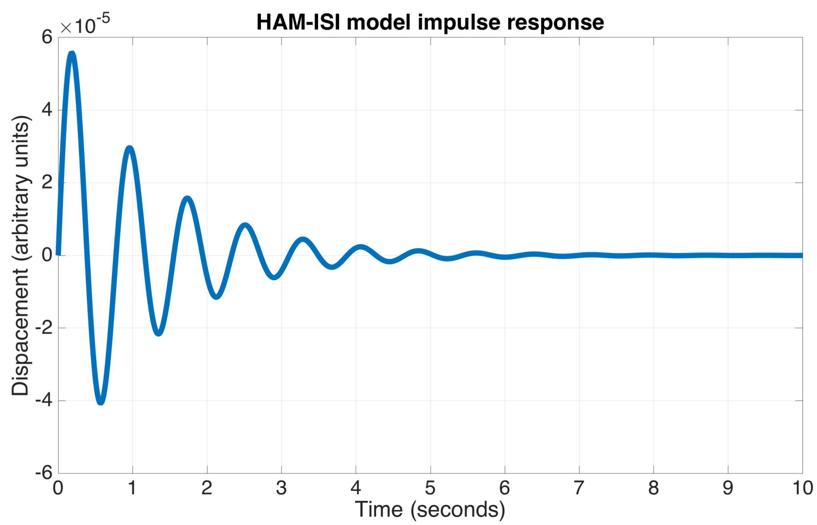
FFT of the
Impulse response



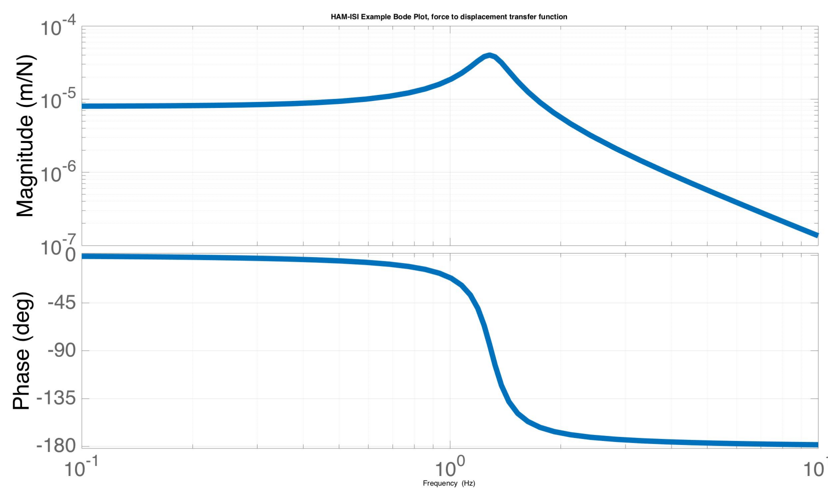
Impulse vs Bode plot

Impulse response

Transfer function



=



Time domain

FFT

IFFT

Frequency domain

Lecture 1 Summary

- Control helps us achieve a desired behavior
  - In general maintain desired setpoints
- Control can be either **feedback** or **feedforward**
- Nearly all our controlled systems are **linear**
- System are modeled in both the **time domain** and **frequency domain**. Both domains are equivalent. Each is useful in its own context.

Lecture 1 – Backups

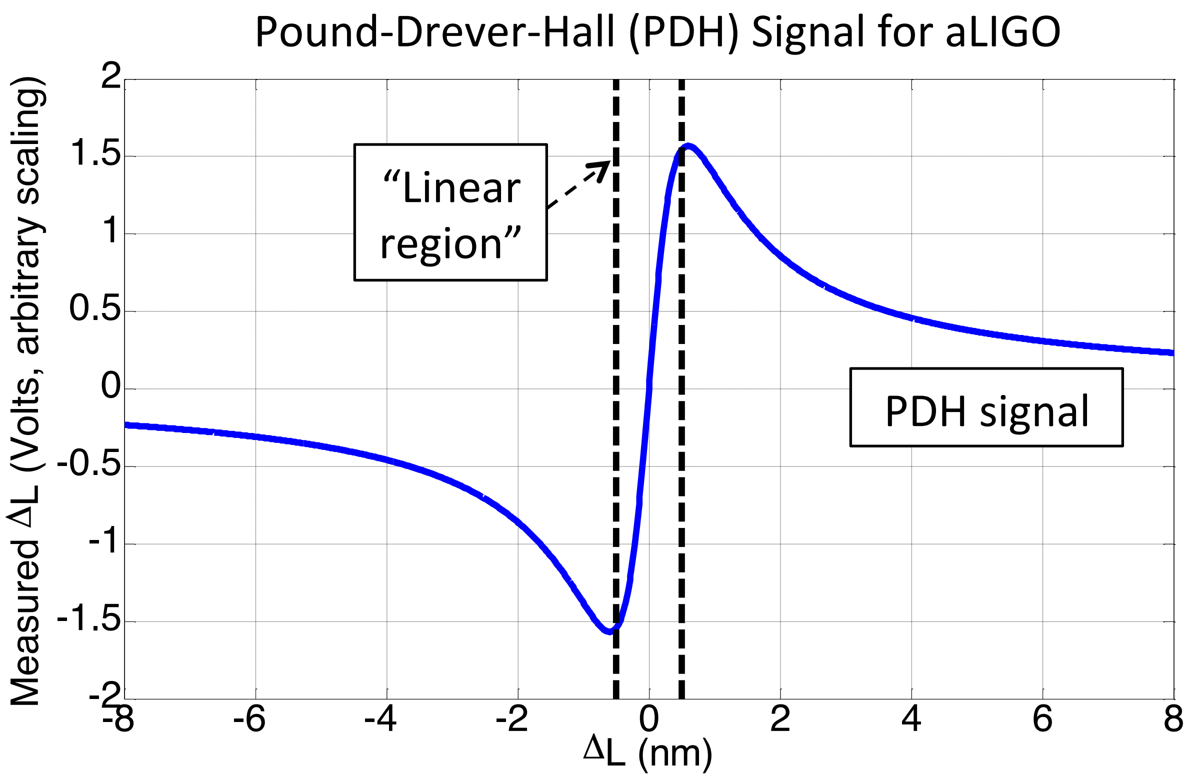
Relationship between Laplace and Fourier

- Laplace becomes Fourier when the real part of *s* goes to zero and the imagery part is frequency in units of rad/s

$$s = 0 + i\omega,\ \omega = 2\pi f$$

- Why do we use Laplace for the transfer functions, but Fourier for the bode plots?
  - As far as I know, this isn't discussed much in controls classes. My hypothesis is that the **Fourier** transform is sufficient for reproducing **time domain signals**, since any time domain signal can be reproduced by an infinite sum of sine waves. However, the Fourier transform does not contain enough information to represent the **dynamics** of the system that produced that signal. Since poles and zeros in general have both real and imaginary parts, the **Laplace** transform, with the addition of the real part of *s*, is sufficiently general.

Cavity Signal

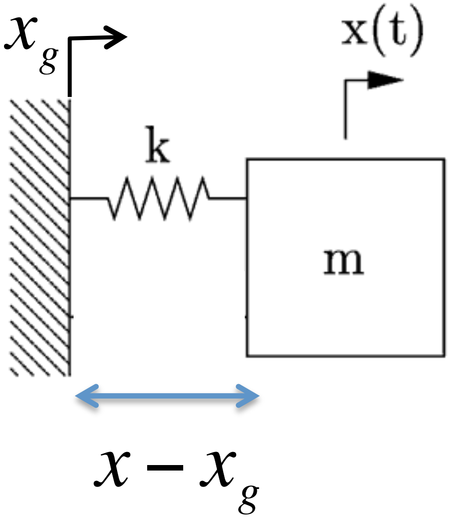


The PDH signal for a 4 km aLIGO Fabry-Perot cavity with mirror power transmissions of 1.4% and 7.5 ppm. The cavity finesse is 445. The linear region between the dashed lines is 1 nm wide.

$$PDH = C\frac{\sin\left(4\pi\frac{\Delta L}{\lambda}\right)}{1+\left[\frac{2F}{\pi}\sin\left(2\pi\frac{\Delta L}{\lambda}\right)\right]^2}$$

F = cavity finesse = 445
λ = laser wavelength = 1064 nm
C = arbitrary electronic scaling

LIGO Models – SS rel. disp. output



Relative displacement sensor

Equation of motion

$$m\ddot{x} + kx = kx_g$$

$$\dot{\vec{x}} = A\vec{x} + B\vec{u}$$ ← System dynamics

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ k/m \end{bmatrix} x_g$$

$$\vec{y} = C\vec{x} + D\vec{u}$$ ← Sensing function

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [-1] x_g$$ Ex. Relative displacement sensor

Note, to include the damping term, a different set of state variables is required. See T1400023.