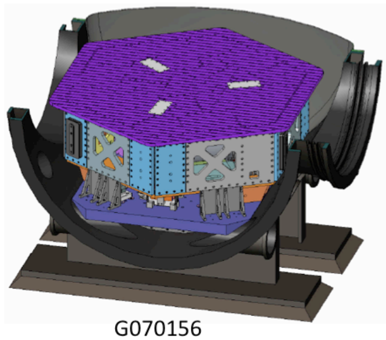
Lecture 2

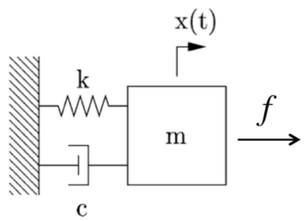
Basic control design
- Part 1: Feedforward
- Part 2: Feedback
- Part 3: Sensor blending

G1600726

LIGO Example system – HAM ISI



G070156

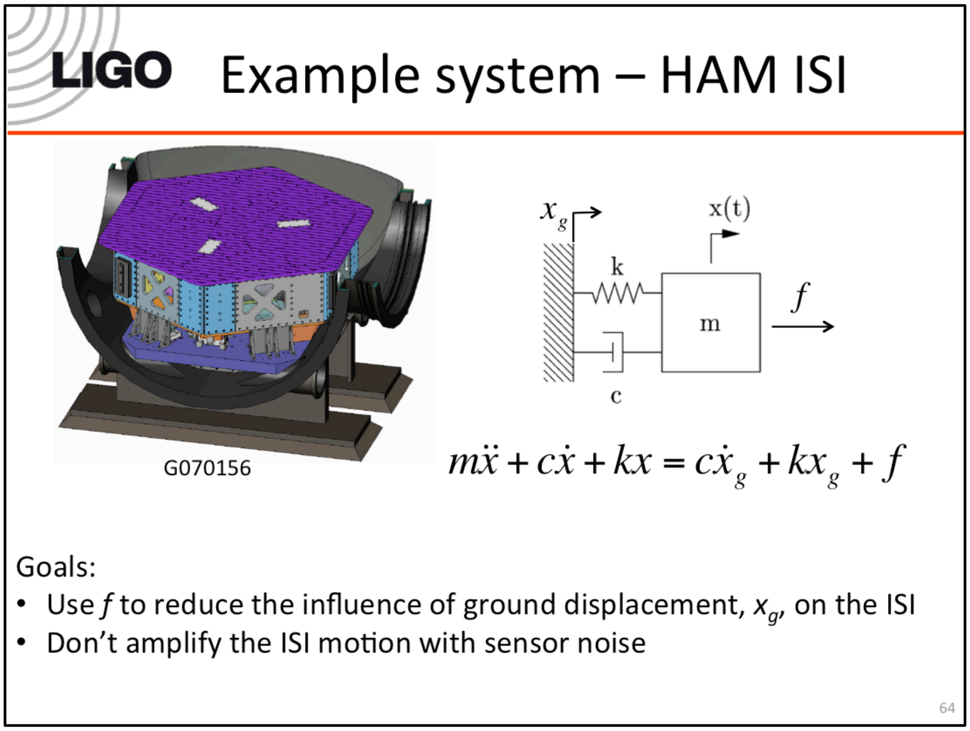


$$m\ddot{x} + c\dot{x} + kx = f$$

63

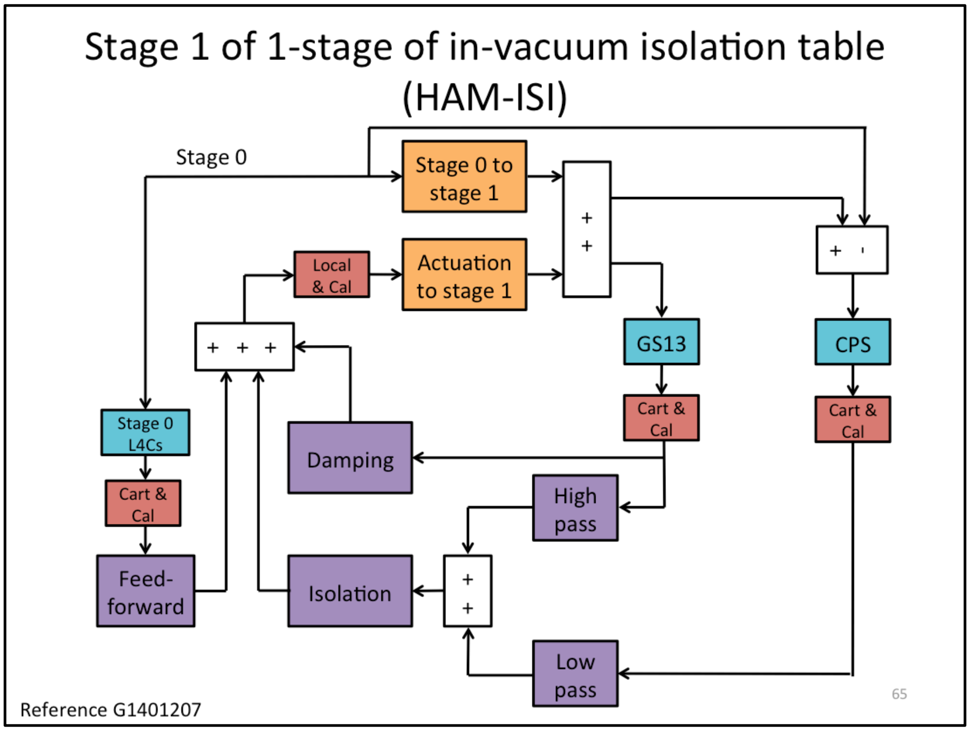
In this lecture, we'll use the HAM-ISI as an illustrative example of control system design. The concepts applied to this system apply to many other systems as well.

We'll model the HAM-ISI as a single degree of freedom (DOF) mass-spring-damper system. Here is the equation of motion we saw in Lecture 1, with the external force input.



More generally, there is also a ground motion input, which adds driving terms to the right side of the equation.

The goals here are to apply external forces, *f*, to minimize the influence of the ground displacements x_g, without amplifying the motion of the ISI with sensor noise.



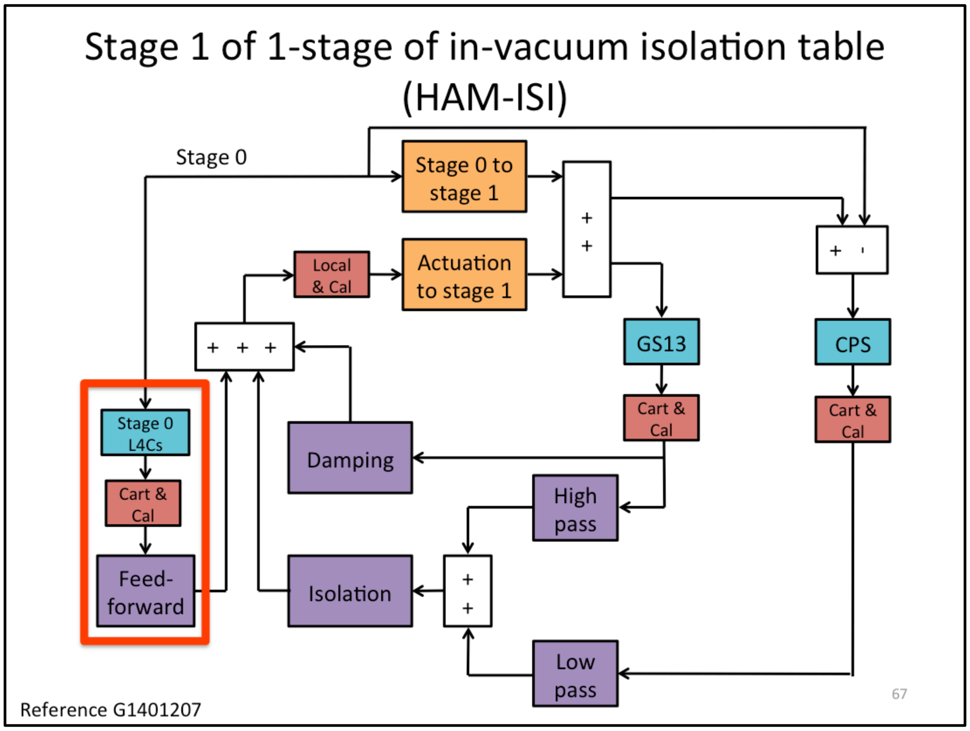
In the end, the control of the HAM-ISI we use to achieve these goals can be represented with a block diagram like this. The hope is that by the end of this lecture, we'll understand each piece of this diagram.

Lecture 2 – Part 1

Feedforward

G1600726

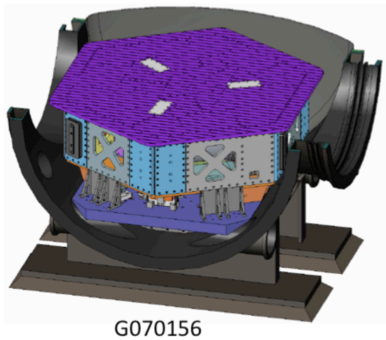
66



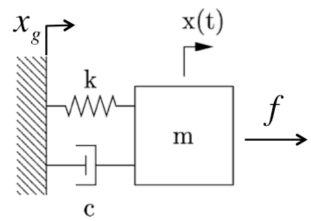
This is the feedforward part of this block diagram. There is a sensor that measures the ground motion, and a feedforward filter that drives the actuators to cancel this ground motion.

We won't focus on the Cart & Cal matrix transformations here. For more information on these matrix transformations, see the Lecture 2 backup slides.

Feedfoward Control



G070156



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

Ideal feedforward controller → $f = -c\dot{x}_g - kx_g$

- An inertial sensor on the ground measures x_g
- The actuator applies the correcting force *f* before the ISI responds
  - it's like the ground never even moved
- Performance is limited by how well the controller is tuned, and how much coherence there is between the ground sensor and the ISI sensor.
- The feedforward controller does not depend on the feedback design

68

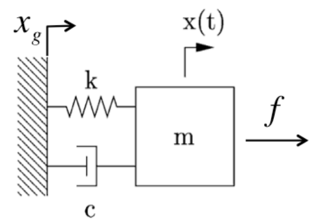
With feedforward control, we simply try measure the ground motion, and cancel out its terms on the right side of the equation directly. This is limited by how good our sensor is, how much coherence it has with the ISI's motion, and how good our model of the ISI is (e.g. how well we know k and c).

The feedforward control does not depend on the feedback design. This is because feedback reacts after the error has already begun to increase, while feedforward is proactive, acting before the error has a chance to change. Thus, feedforward acts before the feedback has a chance to respond. From the point of view of the feedback, the feedforward has done nothing more than reduce the amount of ground motion.

Feedfoward Control

In practice, the feedforward control is achieved with the following 4 steps:

1. Measure the TF between the ground and the ISI
2. Measure the TF between the actuator and the ISI
3. Calculate the ratio of step 1 to step 2
4. Fit a filter to this TF ratio. This is the feedforward control filter.



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

69

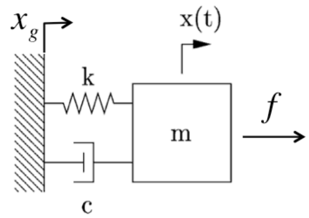
In practice, we don't have perfect models of our systems. For example, we don't know the calibrations of our sensors and actuators perfectly, in addition to not knowing k and c perfectly. So we base the feedforward control off measurements. The first step is to measure a transfer function between the sensor on the ground and the sensors on the ISI.

LIGO

Feedfoward Control

In practice, the feedforward control is achieved with the following 4 steps:

1. Measure the TF between the ground and the ISI
2. Measure the TF between the actuator and the ISI
3. Calculate the ratio of step 1 to step 2
4. Fit a filter to this TF ratio. This is the feedforward control filter.



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

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

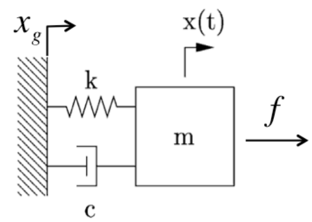
70

The second step is to drive the actuators and measure the transfer function between the actuators and the sensors on the ISI.

LIGO Feedfoward Control

In practice, the feedforward control is achieved with the following 4 steps:

1. Measure the TF between the ground and the ISI
2. Measure the TF between the actuator and the ISI
3. Calculate the ratio of step 1 to step 2
4. Fit a filter to this TF ratio. This is the feedforward control filter.



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

$=$ 3) $$\frac{f}{x_g} = cs+k$$

71

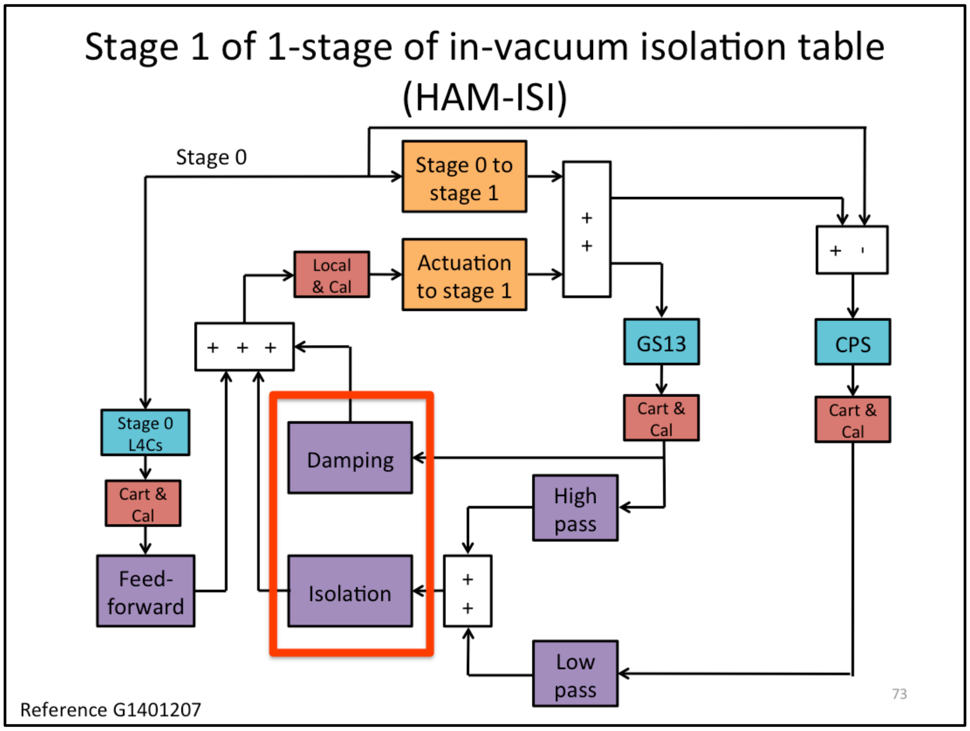
The third step divides the first transfer function by the second. This gives us a transfer function in units of (actuator force) / (ground displacement). This describes the feedforward control we want to apply. Notice that it is the Laplace transform of the ideal feedforward law discussed a few slides ago. The 4th and final step is to fit a filter to this measurement, that matches in magnitude and phase. This filter is the feedforward controller we apply.

Any uncertainties in the sensor and actuator calibrations, and the ISI parameters, will be taken into account with these measurements.

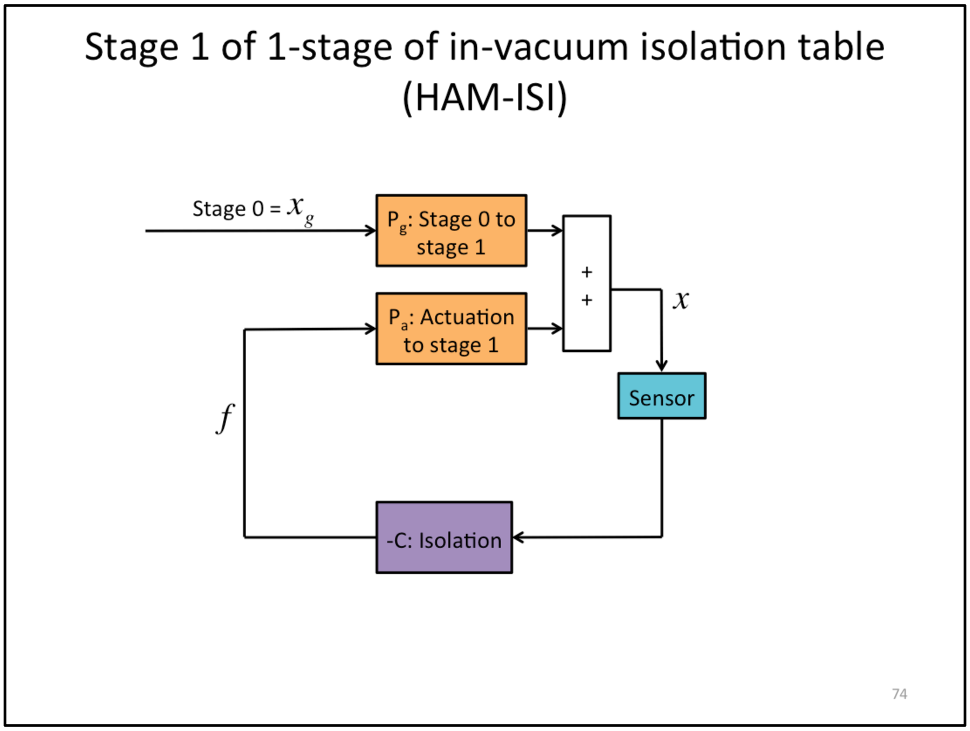
Lecture 2 – Part 2

Feedback

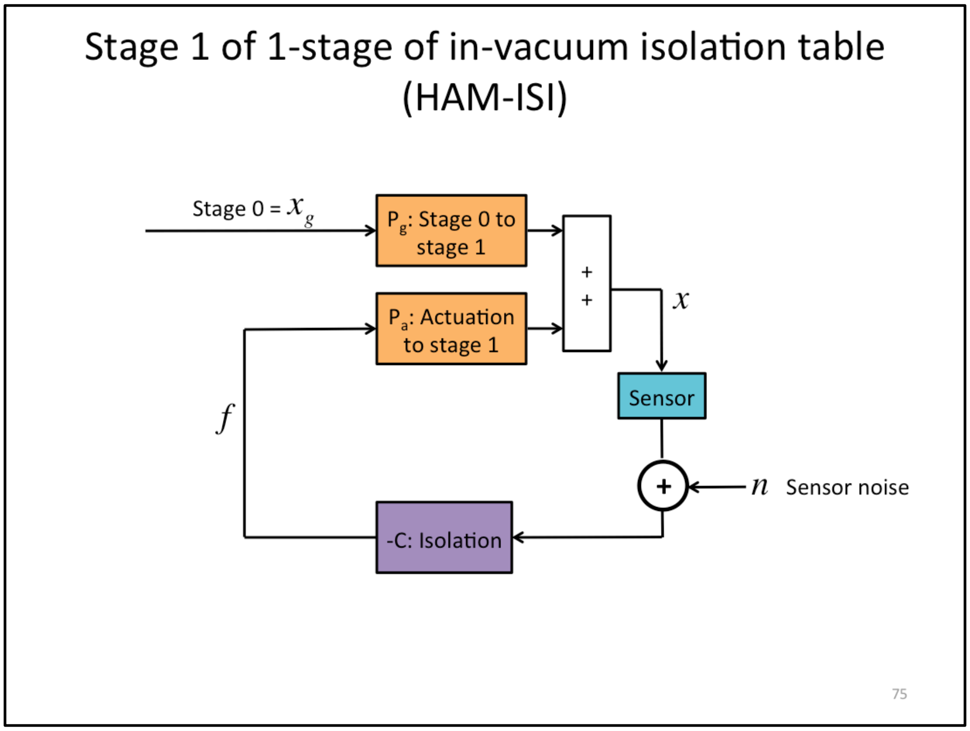
G1600726



These blocks represent the HAM-ISI feedback. Don't be confused by the fact that there are two blocks here, Damping and Isolation. They have the same purpose, but are used in different states of the interferometer. Damping provides a small amount of isolation, but is very robust, and prevents the platform's resonance frequencies from ringing up. Isolation is less robust, but provides very high performance isolation. Typically, when turning the interferometer on, we start with the damping feedback, and then engage the isolation feedback.

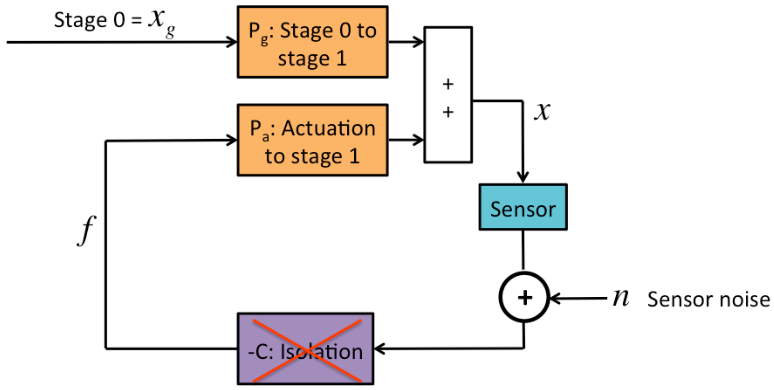


To see how feedback works, let's remove all elements in the bock diagram except for the isolation path (recall, isolation an damping have the same basic roll, so let's pick just one of them).



Let's add some sensor noise to this diagram, since it is important, but wasn't included before.

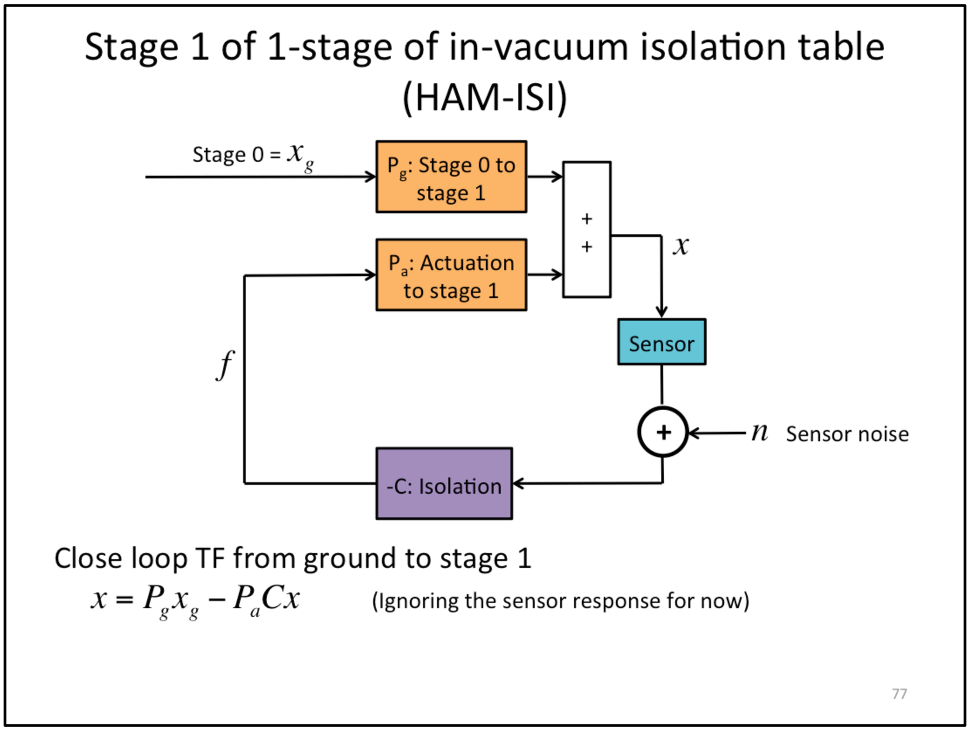
Stage 1 of 1-stage of in-vacuum isolation table (HAM-ISI)



- Uncontrolled TF from ground to stage 1
  $x = P_g x_g$

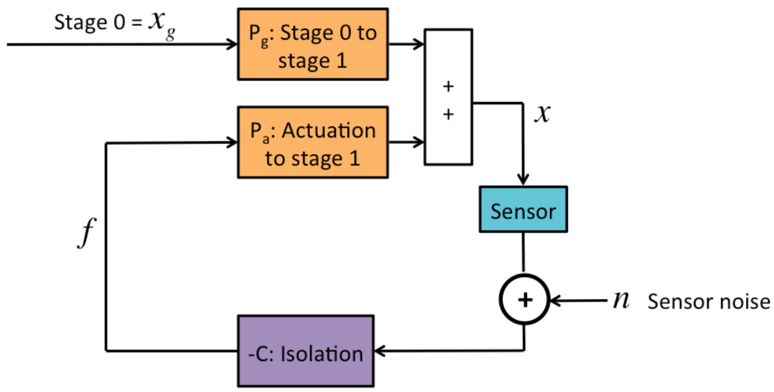
76

Let's now write down the uncontrolled transfer function from ground displacement to platform displacement.



Then, when we turn on the feedback loop, we just add one more term.

Stage 1 of 1-stage of in-vacuum isolation table (HAM-ISI)



Close loop TF from ground to stage 1

$$x = P_g x_g - P_a C x$$

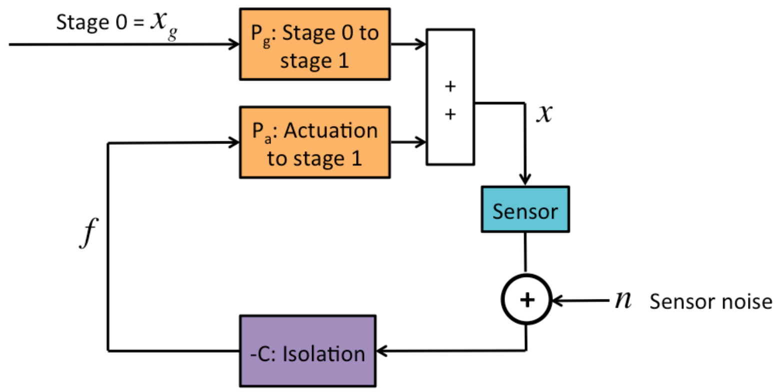
(Ignoring the sensor response for now)

$$x = \frac{P_g}{1 + P_a C} x_g$$

78

We can then solve for platform displacement, x, as a function of ground displacement, x_g. This is the closed loop transfer function from the ground to the platform.

Stage 1 of 1-stage of in-vacuum isolation table (HAM-ISI)



Close loop TF from sensor noise to stage 1

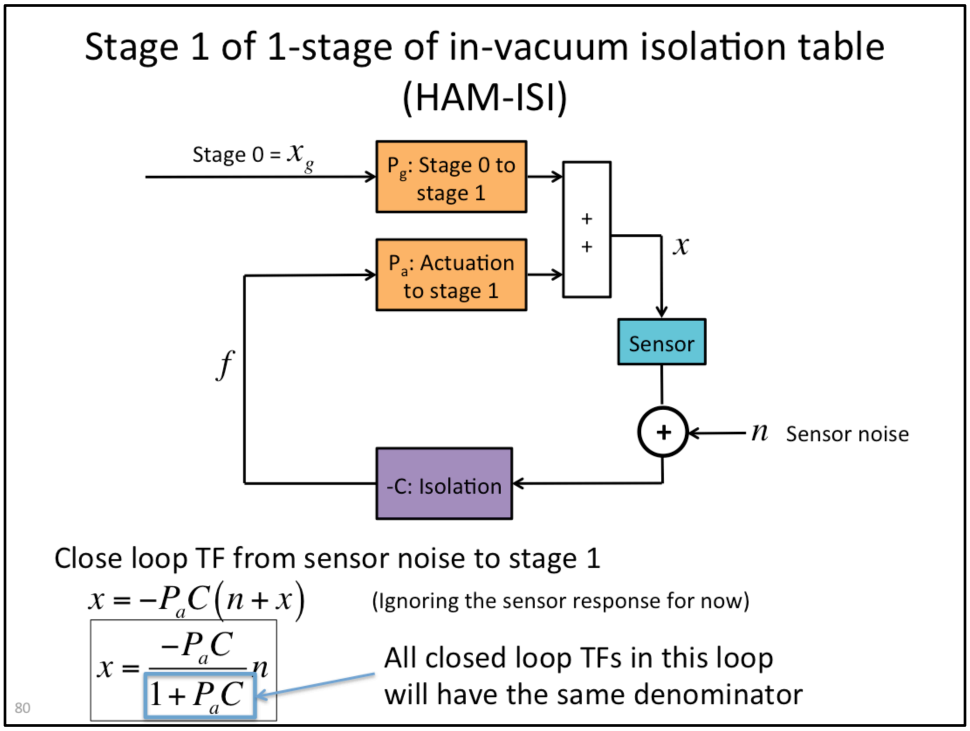
$$x = -P_a C(n + x)$$

(Ignoring the sensor response for now)

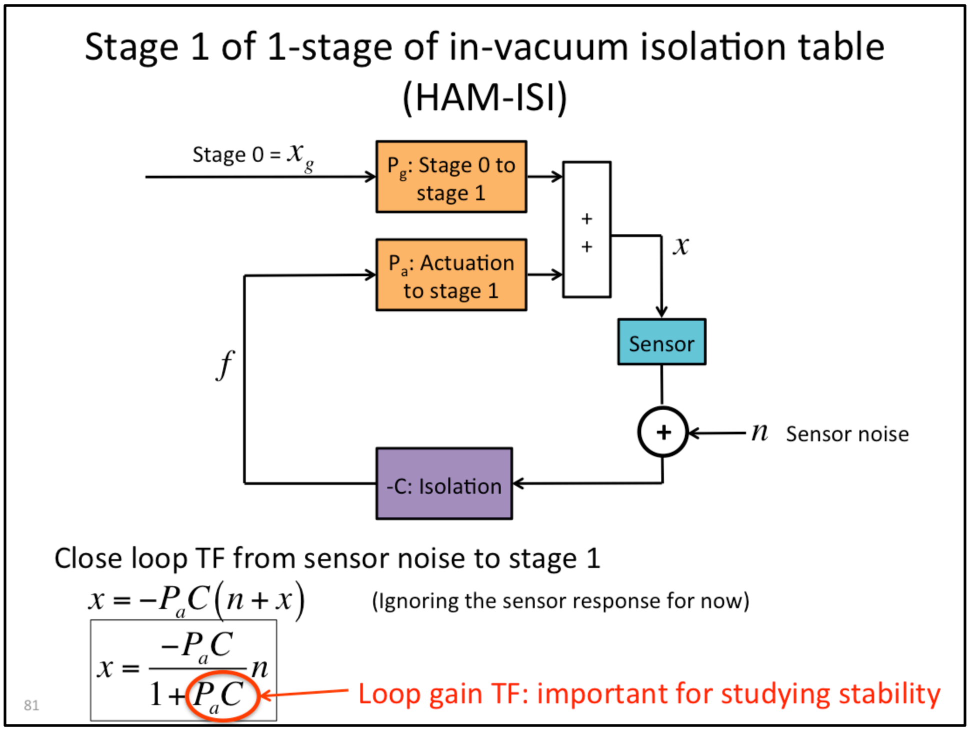
$$x = \frac{-P_a C}{1 + P_a C} n$$

79

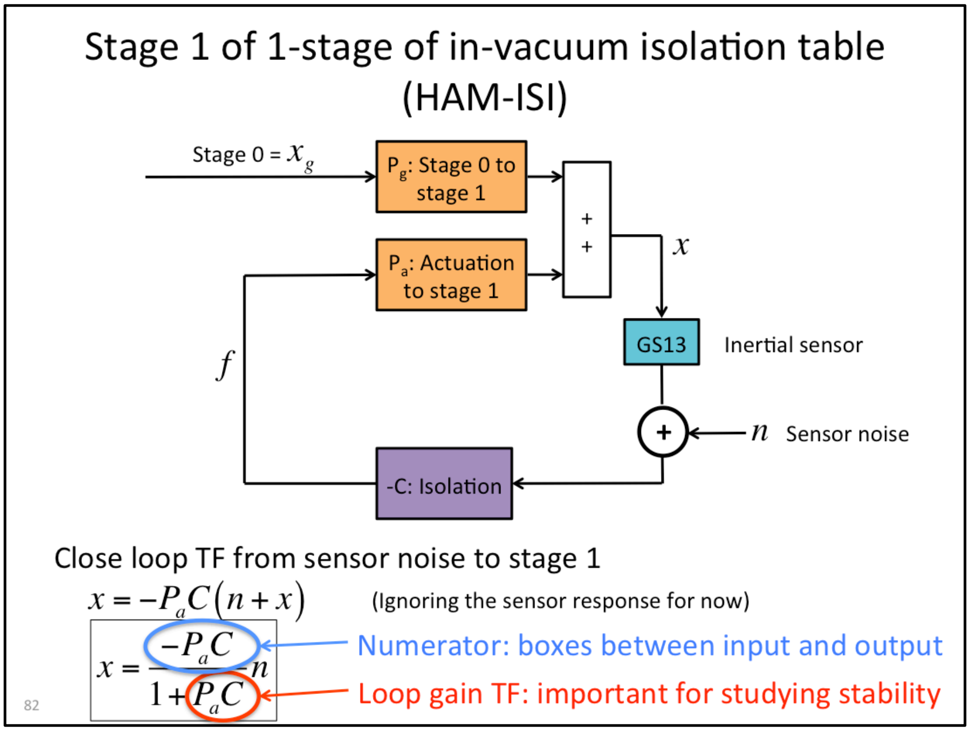
Similarly, we can find the closed loop transfer function from sensor noise, n, to platform displacement, x.



Note, all closed loop transfer functions in this loop will have the same denominator, with the form 1 + something.



That something is called the loop gain transfer function. It is a product of all the boxes in the closed loop. It is this loop gain transfer function that we analyze to study the loop's stability. Just to clarify, this means we are analyzing the loop's open loop characteristics to study its closed loop behavior.



The numerator changes depending on which inputs and outputs you are examining. It will always turn out, that the numerator is just the product of the boxes between the input and output. In this case, we have the isolation block and P_a block between the sensor noise and the plant displacement. Therefore, it is very straightforward to find a closed loop transfer function just by inspecting the block diagram. No need to solve the algebraic equations each time.

Closed Loop TFs

$$x = \frac{P_g}{1 + P_a C} x_g$$

Seismic noise transmission

$$x = \frac{-P_a C}{1 + P_a C} n$$

Sensor noise transmission

83

Let’s now compare the two closed loop transfer functions that we have: the seismic transmission, and sensor noise transmission.

LIGO

Closed Loop TFs

$$x = \frac{P_g}{1 + P_a C} x_g$$

Seismic noise transmission

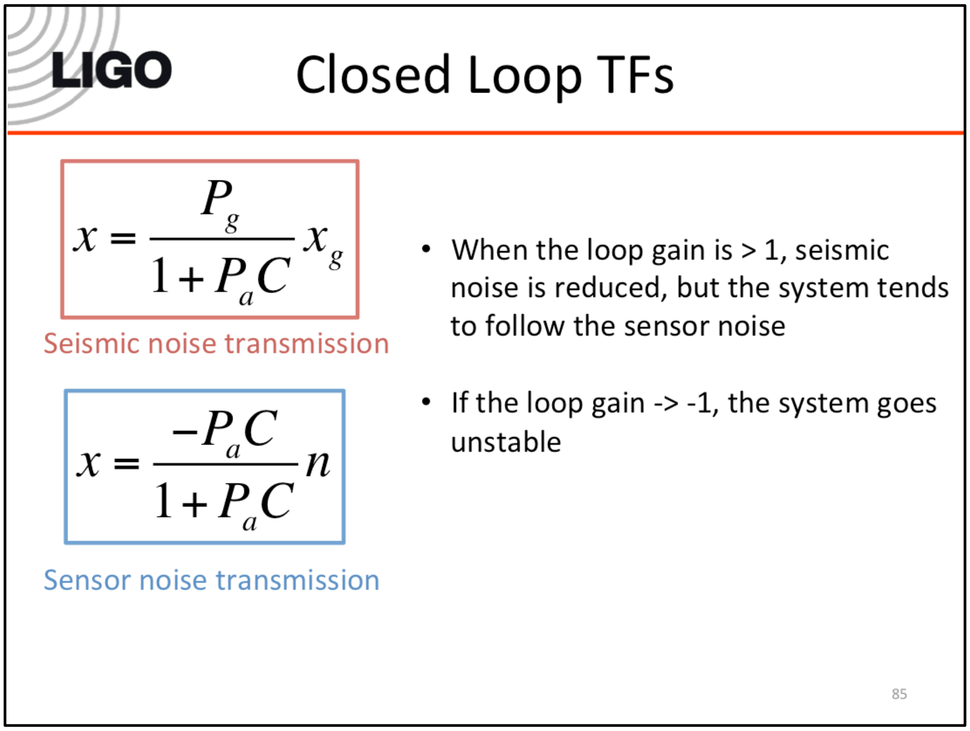
$$x = \frac{-P_a C}{1 + P_a C} n$$

Sensor noise transmission

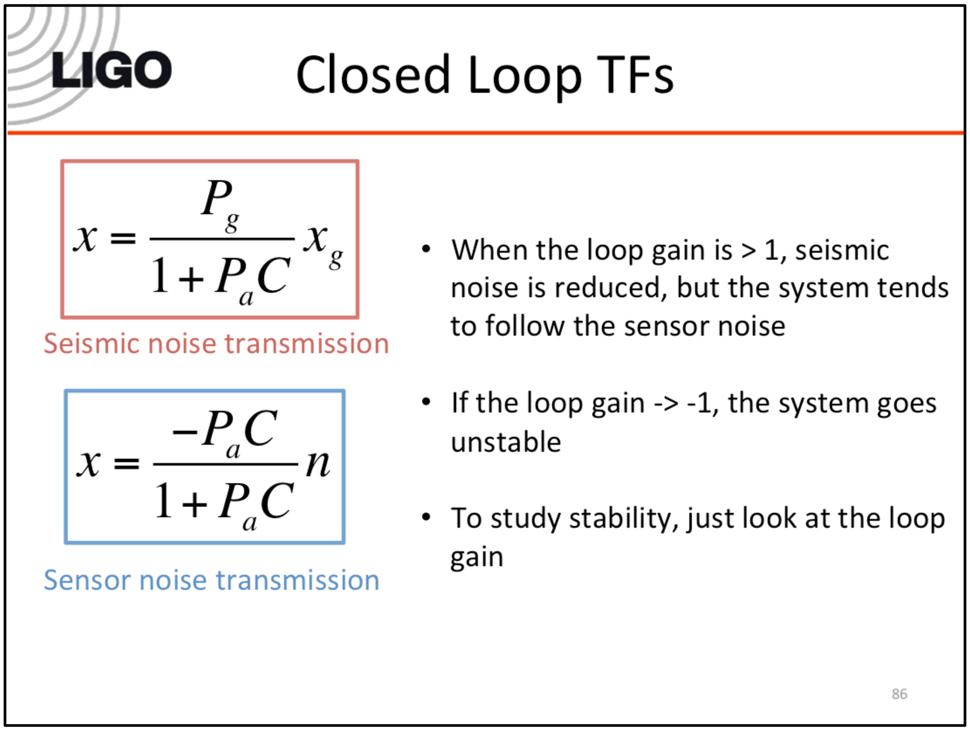
- When the loop gain is > 1, seismic noise is reduced, but the system tends to follow the sensor noise

84

In the seismic case, when the loop gain is large, the seismic noise is reduced. However, in the sensor noise case, the transfer function approaches -1. Thus, the platform is driven to follow the sensor noise. This is fine at frequencies where the sensor noise is small; specifically less than how much the platform would respond to the seismic noise without control. However, at other frequencies where sensor noise is larger, big loop gains would cause the platform to move more than it would without control. So in general, we want to have large loop gains where seismic noise moves the platform a lot, and small loop gains everywhere else to minimize the influence of sensor noise. (sometimes suppressing seismic noise at low frequencies is important enough it is best to tolerate some sensor noise at high frequencies)

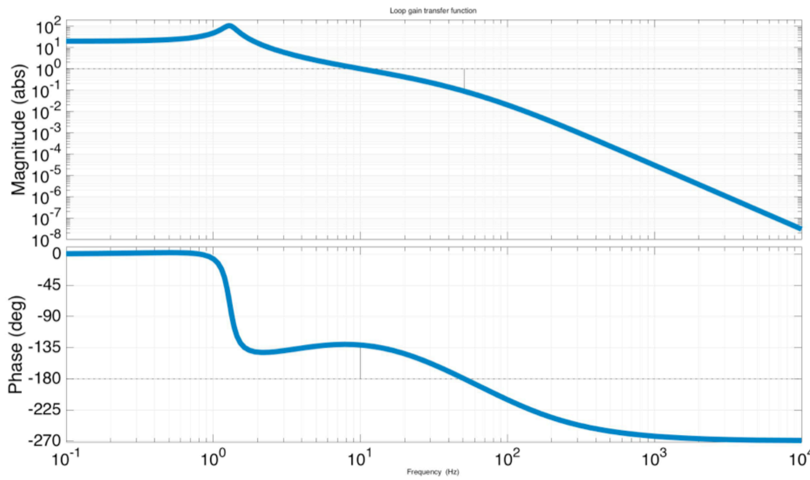


Note, that if the loop gain equals -1, the closed loop transfer function goes to infinity. Clearly, the -1 point must be meaningful for stability.



In practice, it is not just loop gain = -1, but how it approaches -1 that is important for stability. As mentioned before, we can study stability just by examining the properties of the loop gain, in particular its magnitude and phase (Bode plot).

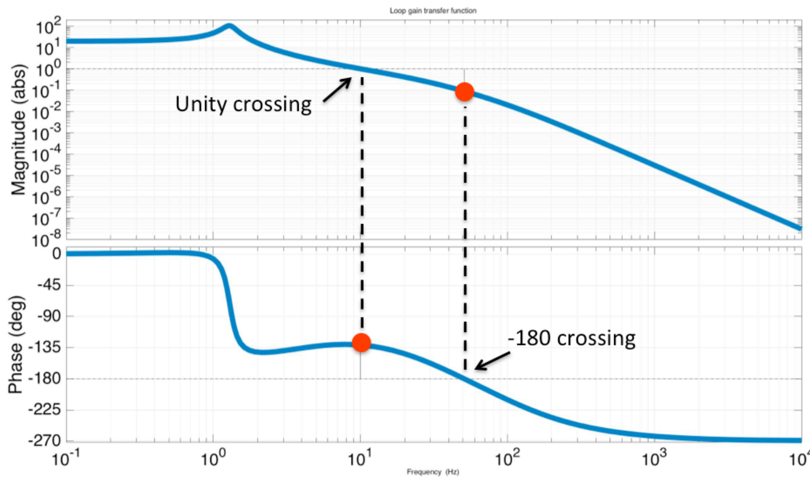
Ex. Loop Gain TF: $P_a C$



Matlab for control filter: `C = zpk(-2*pi*3.33,-2*pi*[30;100],1.4e+10)`

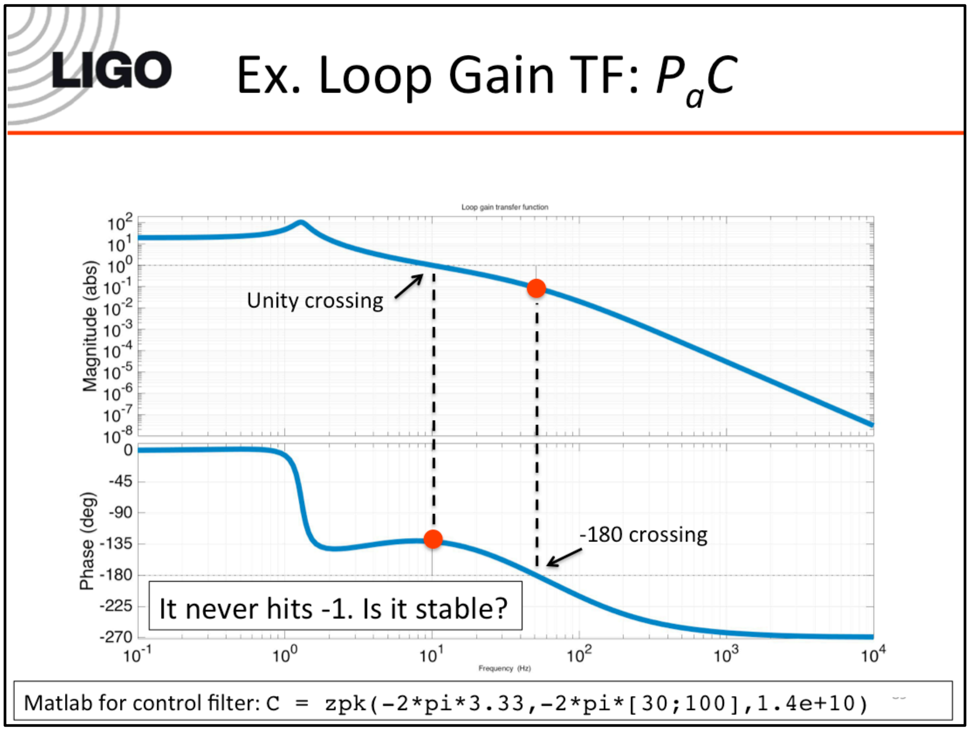
Here is a Bode plot of an example loop gain for the HAM-ISI, using the HAM-ISI model from lecture 1, and the feedback filter C shown here.

Ex. Loop Gain TF: $P_a C$

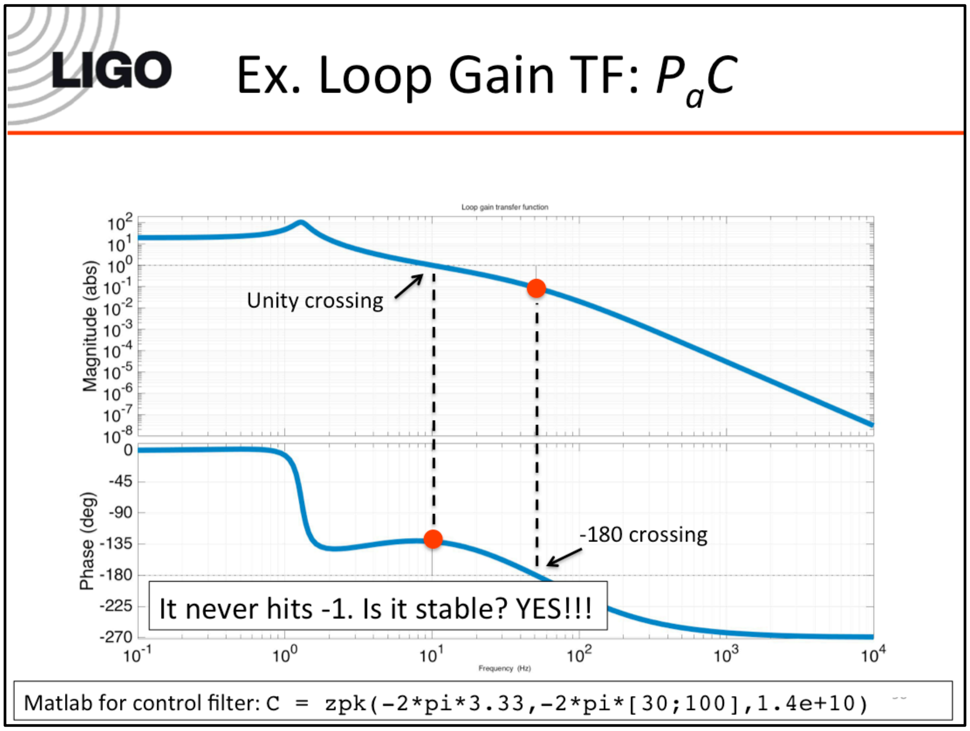


Matlab for control filter: `C = zpk(-2*pi*3.33,-2*pi*[30;100],1.4e+10)`

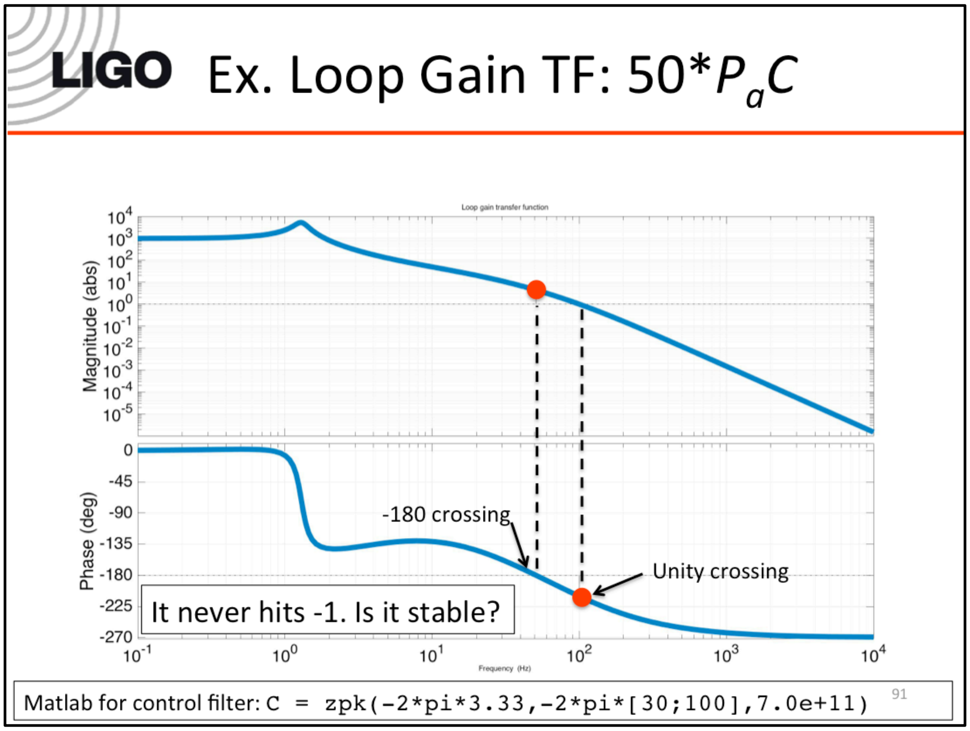
Let's see how close it approaches to the -1 point. Note, -1 is equal to a magnitude of 1 and phase of +-180 degrees. Here, when the magnitude is 1, the phase is about -135. So where have a *phase margin* of 180-135 = 45 degrees. Then, when the phase reaches -180, the gain is about 0.1, so we have a gain margin of 1/0.1 = 10.



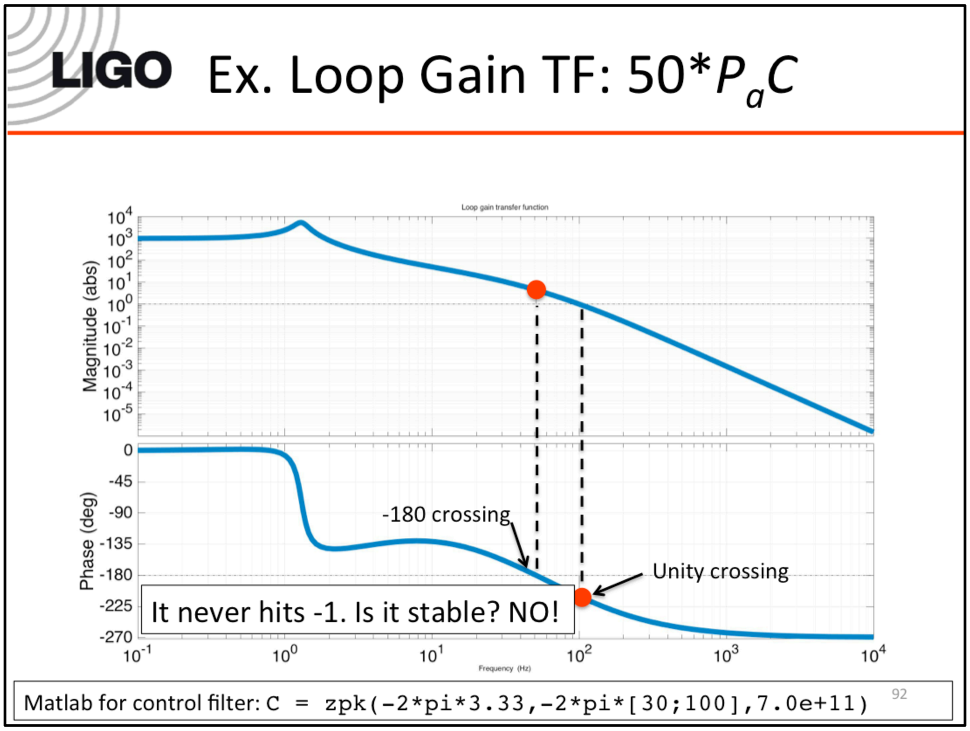
So we’re 45 degrees away from -1 at one point, and an order of magnitude at the other. Is it stable?



Indeed, it is in this case.

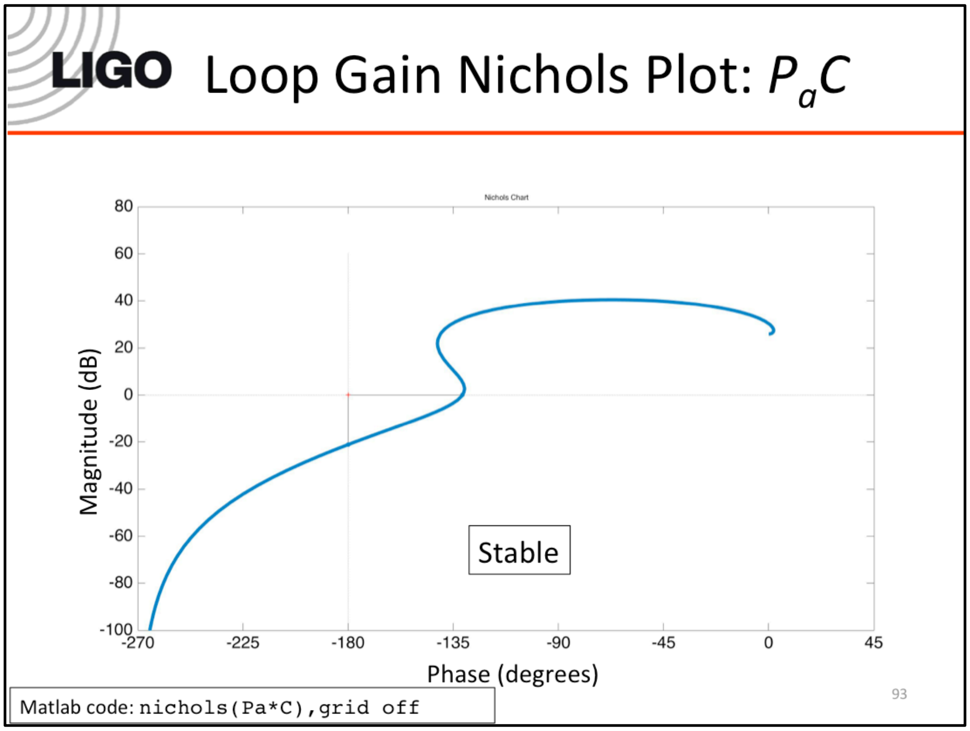


Let's look at another example. Here we have exactly the same loop gain, except the magnitude is 50 times larger. When it crosses unity magnitude, the phase is about -220. So the phase margin is 180-220 = -40. When it crosses -180, the magnitude is about 10. So we have a gain margin of 1/10 = 0.1. As before, we never actually pass through -1. Is this loop stable?



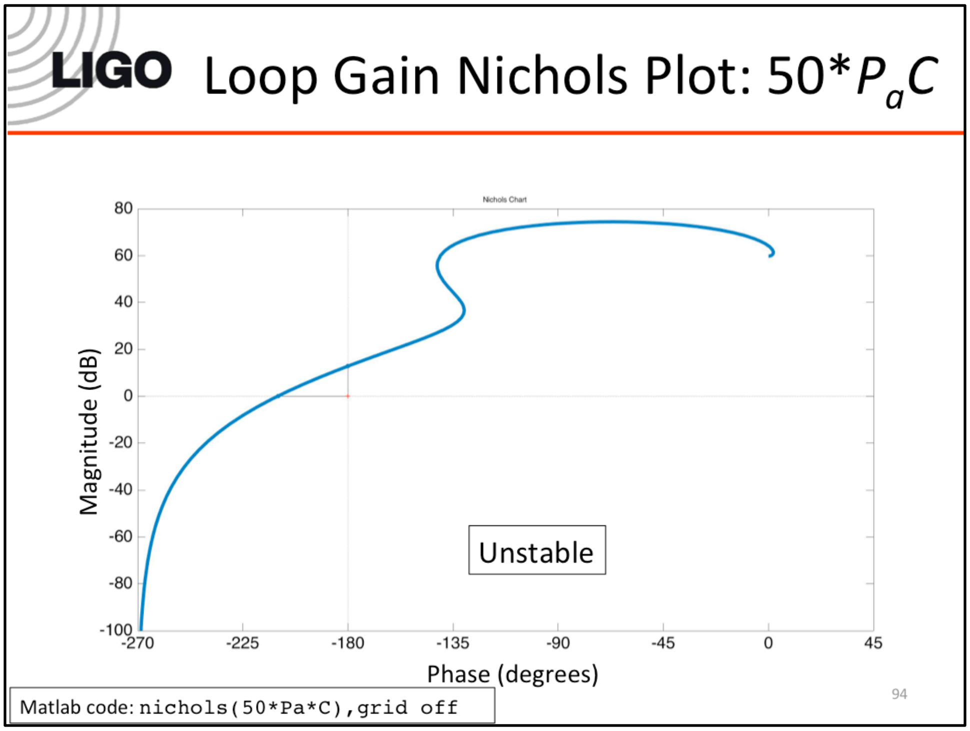
No, it isn't. As mentioned before, how you approach -1 is just as important as avoiding it. In general you need positive phase margins, and gain margins that are greater than 1.

The mathematical proof for what makes a system stable or unstable is very abstract, and has to do with loop gain encirclements of the -1 point. Generally, you'll see this proof in a controls class once, and then never see it again. If you're interested in it, the Ogata text mentioned at the beginning of lecture 1 discusses this. In the end, it suffices to know what the resulting stability rules are.

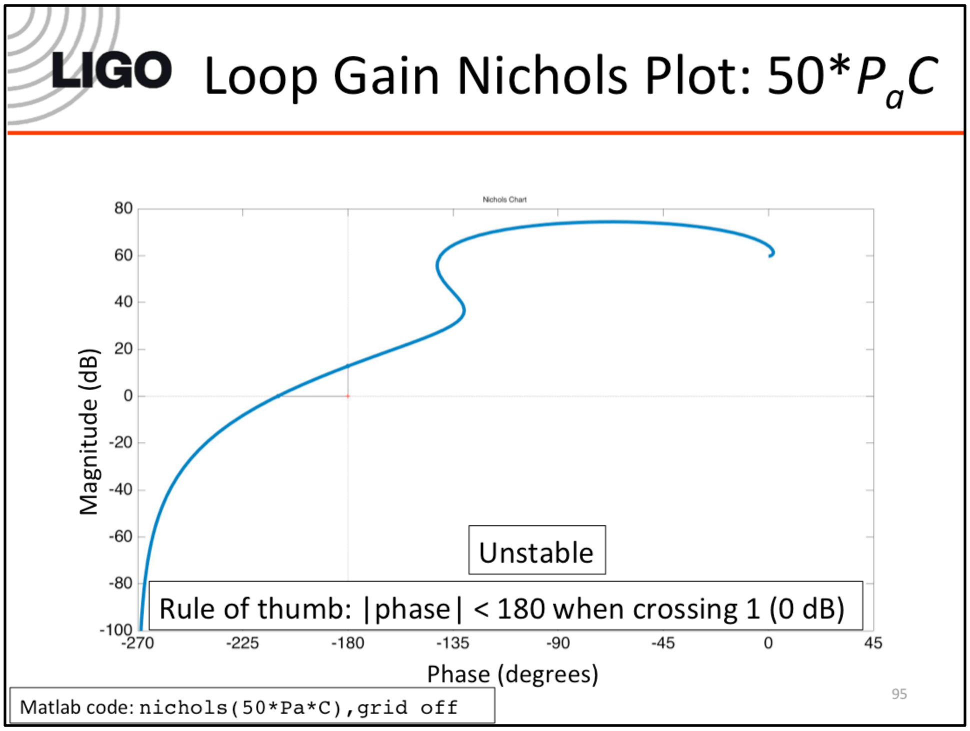


These stability rules are, in my opinion, best represented by the Nichols plot of the loop gain. This Nichols plot is of the stable loop gain we just saw. It shows the same information as the bode plot, except here we have magnitude on the vertical axis (in dB units where 0 dB = 1) and phase on the horizontal. We don't typically look at these plots when designing loops (perhaps we should). Typically we just look at the Bode plot and pick off phase and gain margins. However, stability is much more obvious in these Nichols plots than the Bode plots, so it is very useful to at least have these in mind when looking at the loop gain bode plots. The similar Nyquist plots are good too, and more traditional than these Nichols plots, however, Nichols plots are easier in my opinion because they are log spaced. **In these Nichol's plots a system is unstable if the -1 point is enclosed by the area under the curve.** Here, it is not, so it is stable.

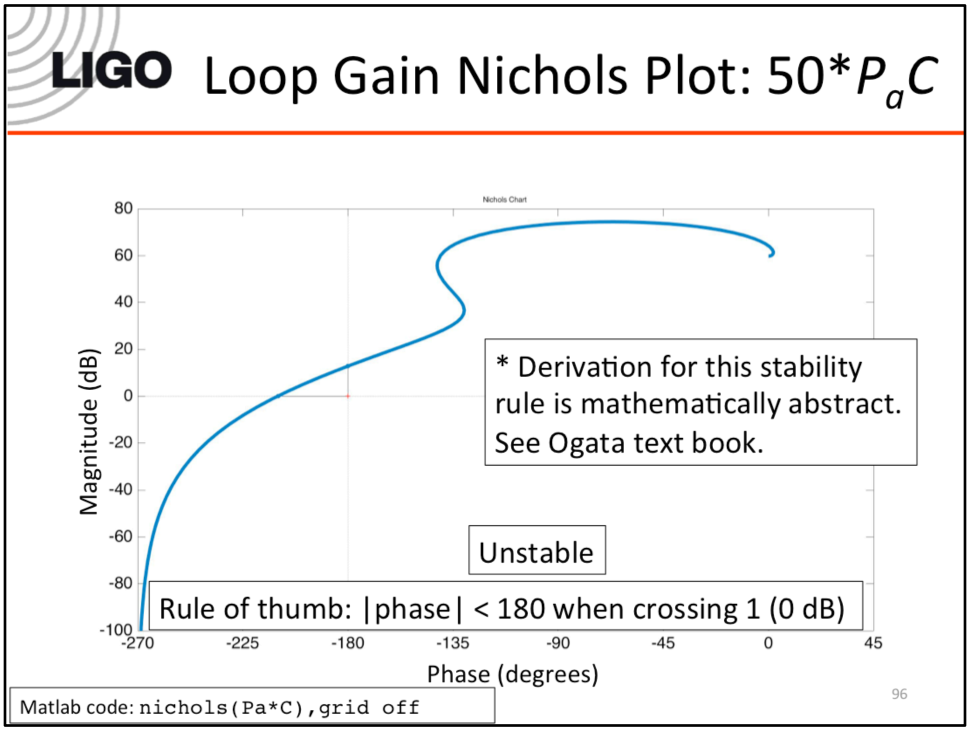
Note, if the uncontrolled plant has any unstable poles, the rules are subtly different. We'll assume all our plants are naturally stable, so the rules discussed here apply.



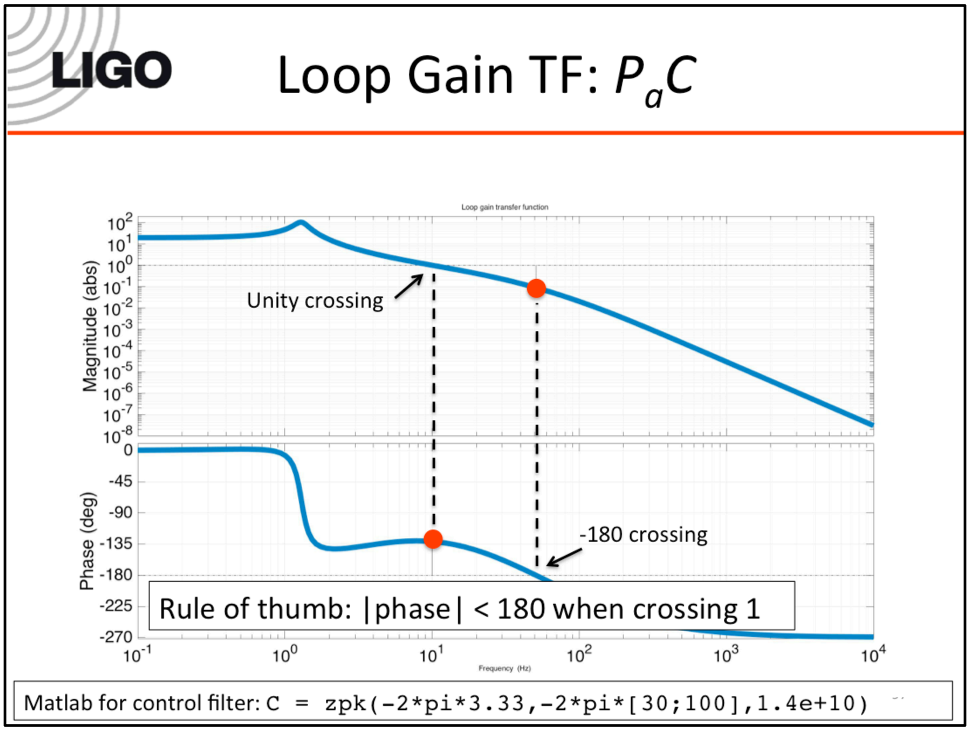
Here is the Nichol's plot with the unstable loop gain. See how the -1 point is now enclosed by the area under the curve.



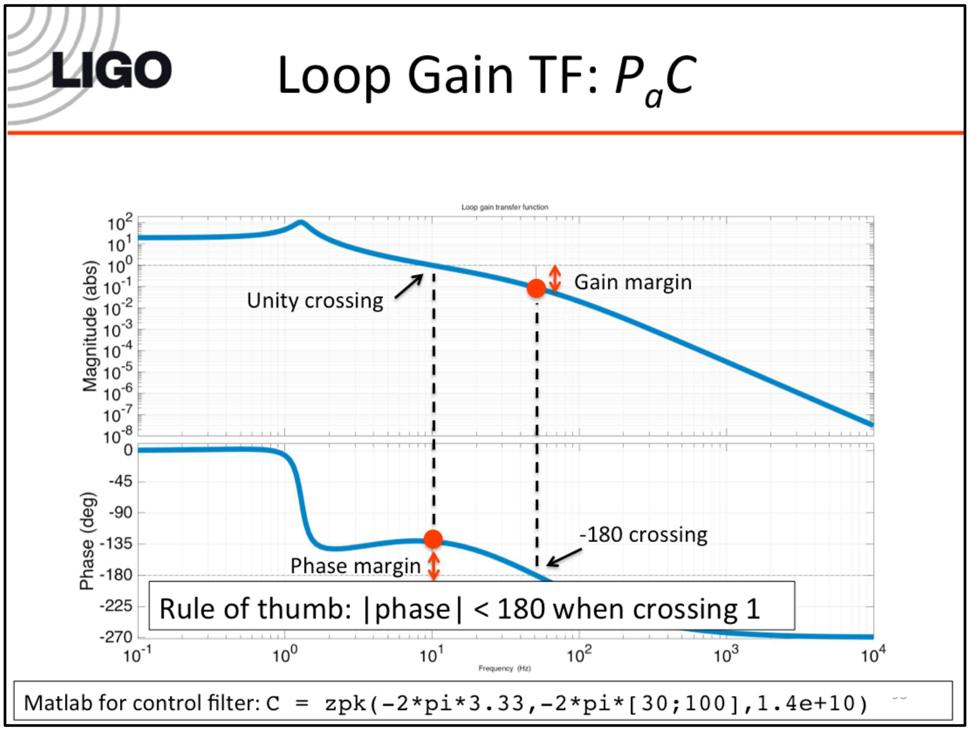
In general, you can keep a system stable simply by ensuring the loop gain phase is within +-180 degrees whenever the loop gain magnitude crosses unity.



As mentioned before, the derivation for these stability rules is mathematically abstract. See the Ogata text for details on the derivation.



We typically measure stability with phase and gain margin. So here is the stable loop gain again.



Stable phase margins are positive, where they are measured as the distance above the -180 degree line (and below the +180 degree line). Here it is about 45 degrees (180-135).

Stable gain margins are measured as the factor below unity whenever +-180 degrees is crossed. Here it is about 10, since the magnitude is 10 times below unity when crossing -180.

LIGO How to do control design

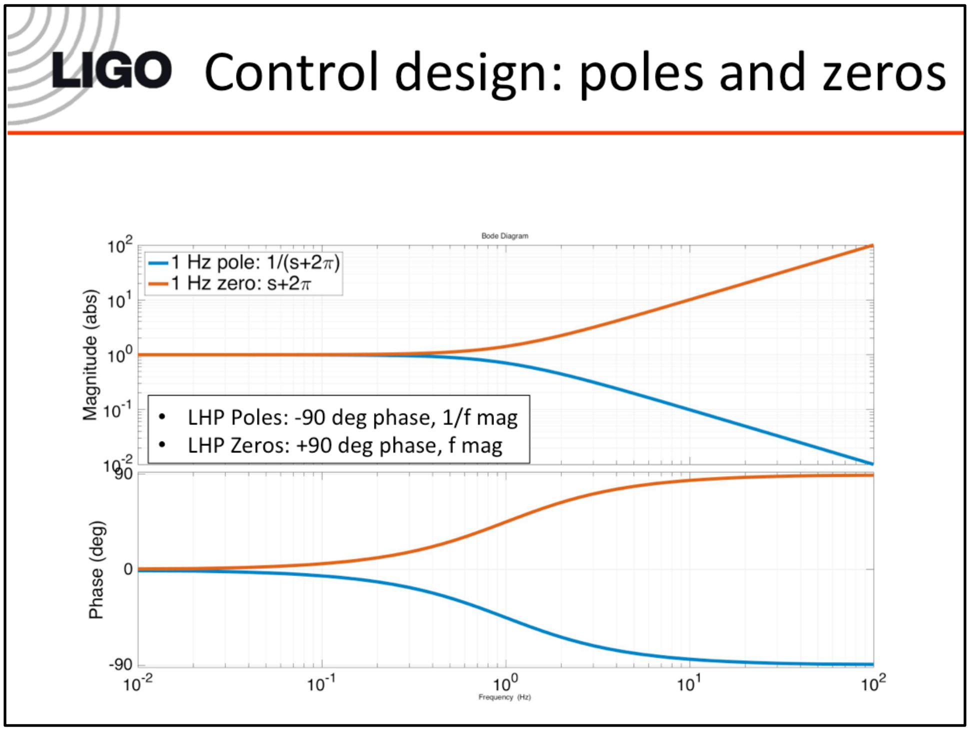
Multiple methods, but the most common is called ‘**loop shaping**’

$$C = \frac{\prod_{j=1}^{m}\left(s + z_j\right)}{\prod_{k=1}^{n}\left(s + p_k\right)}$$

- Place poles and zeros until the loop gain is ‘shaped’ the way you like it
- Causal filters require at least as many poles as zeros: n ≥ m. Non-causal filters respond with infinite magnitude and positive phase at infinite frequency, which they can only do if they have access to future data.

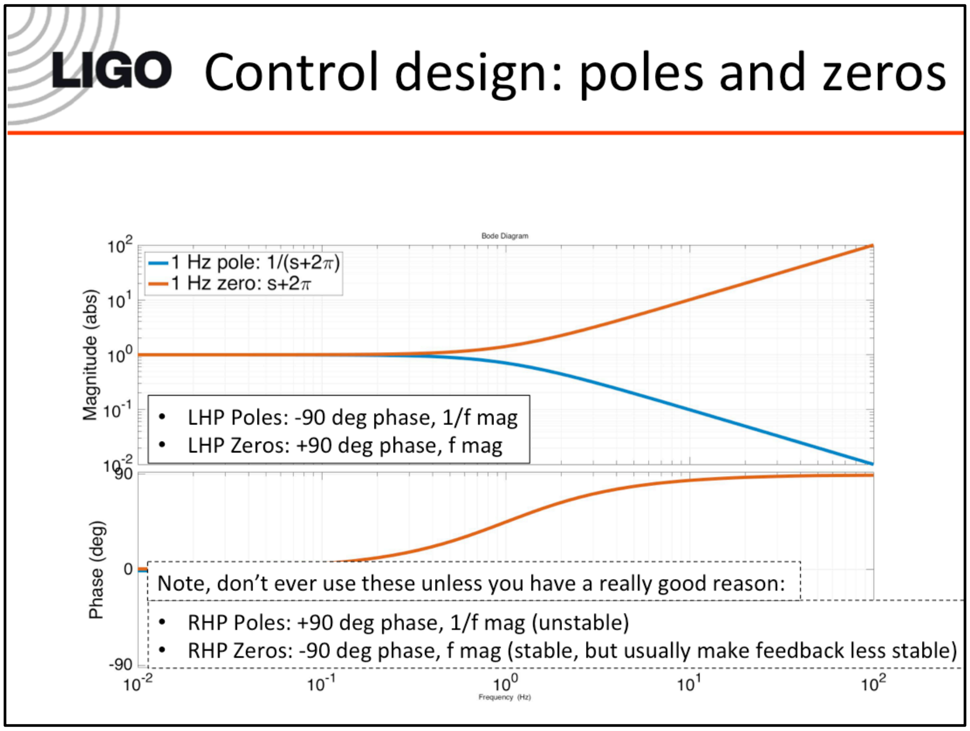
99

In the next few slides, we follow an example of designing a feedback filter using the most common method, called loop shaping. Other design techniques exist, but this is by far the most common.
The technique involves placing poles and zeros in the feedback filter until the loop gain has the characteristics you desire.
Note, causal filters must have at least as many poles as zeros. If it is not causal, the filter can not be used in real-time, because it would require information from future data. Causality is not really in the scope of this lecture, but we’ll touch on it briefly on the next slide. There is also a backup slide in this lecture for it.



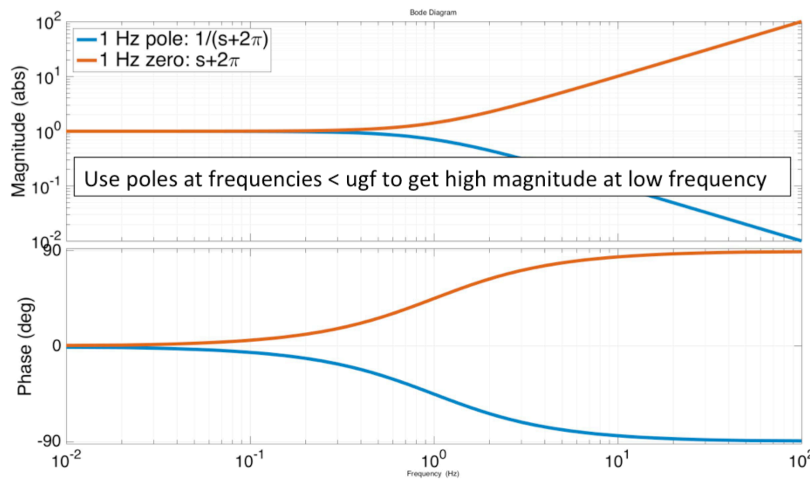
Before beginning the feedback design example, it is useful to look at a bode plot of a single zero, in red, and pole, in blue. Here both are set to 1 Hz. Note that the zero magnitude increases linearly with frequency, and the phase transitions to +90 degrees. The pole is the exact opposite. The magnitude is inversely proportional to frequency, and the phase goes to -90 degrees. (you can kind of think of the zeros and poles'turning on' as you move past them in frequency) For complex airs of poles and zeros, see the backup slides.

For the case of non-causal filters, we would have more zeros than poles, which means to magnitude would increase up to infinity at infinite frequency, with positive phase. This means the filter output would occur before the input. In realtime clearly this can't happen. Non-realtime filters, for offline data processing, can have more zeros than poles.

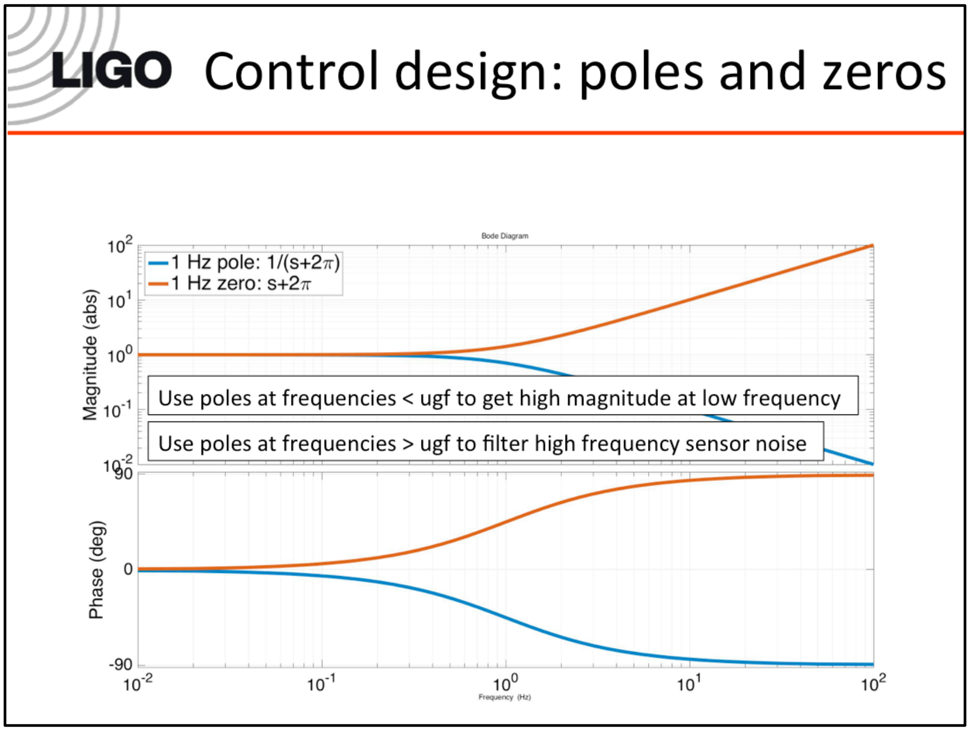


Note, these zero and pole properties are only valid for left half plane (LHP) poles and zeros. Right half plane (RHP) poles and zeros have the reverse phase characteristics (poles -> +90 phase, zeros -90 phase). It is best to avoid using either of these unless you have a really good reason. RHP poles are naturally unstable. RHP zeros are actually stable, however, they tend to make feedback less stable (because of their negative phase). RHP poles are zeros are not completely useless it turns out, but you should have a good reason for using them. For example, sometimes the only way to stabilize an unstable plant is to use RHP poles. This is a bit weird because it means you're using an unstable controller to stabilize an unstable plant. In any case, nearly all our LIGO plants are stable (exceptions being some modes of the optics under high laser power, due to coupling to radiation pressure).

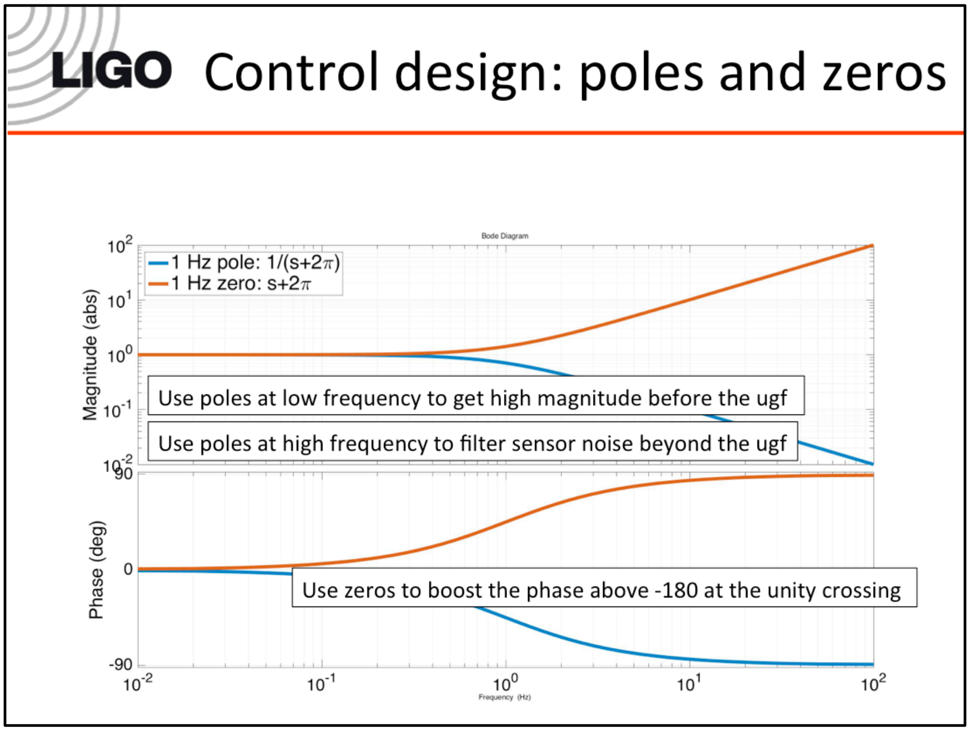
Control design: poles and zeros



In general, the loop shaping design approach is to place poles at frequencies below the unity gain frequency (ugf) to get high low frequency gain.



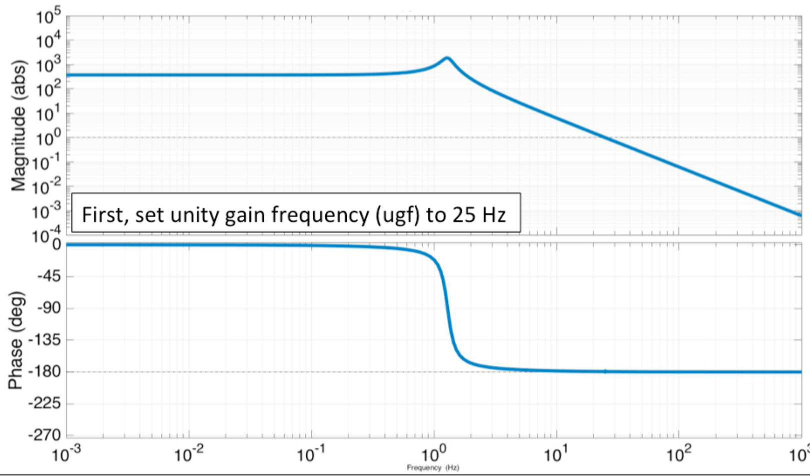
And, to place them at at frequencies greater than the ugf to filter out sensor noise at high frequencies.



Then, the zeros are used primarily to boost the phase above -180 at the UGF so the loop is stable.

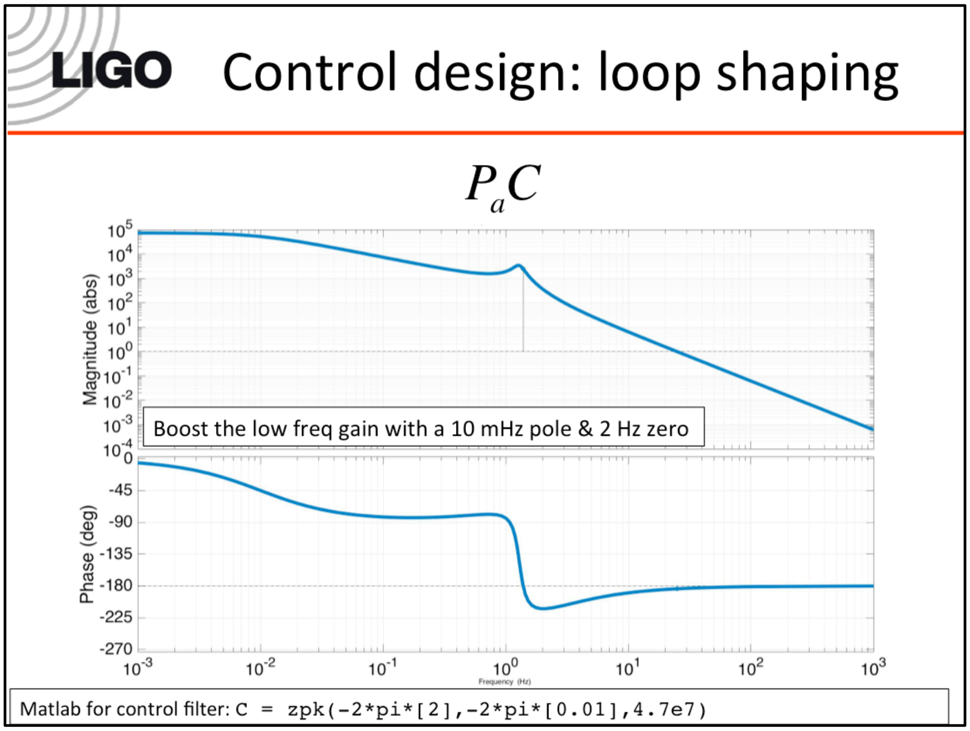
Control design: loop shaping

$$P_a C$$

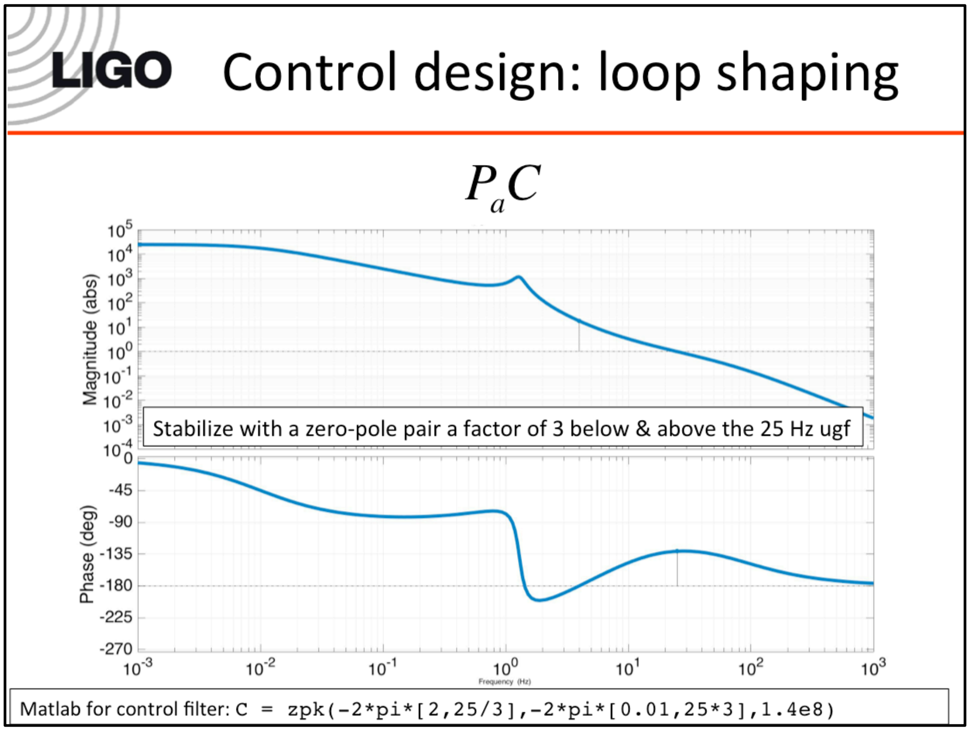


Matlab for control filter: `C = zpk([],[],4.7e7)`

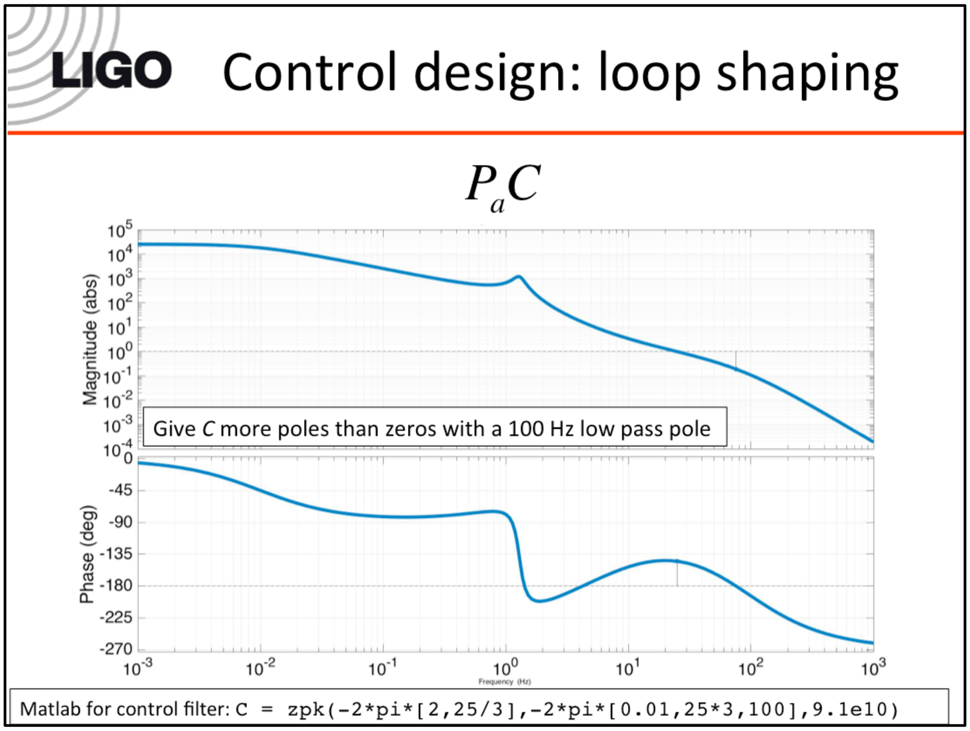
So here is an example of a loop gain bode plot for the HAM-ISI. At this point the filter C is just a static gain value, chosen to set the ugf at 25 Hz. 25 Hz – 30 Hz is where the UGFs end up on the ISIs in practice, due to various limitations with going higher (primarily phase loss from sampling and continuous body vibrational modes).



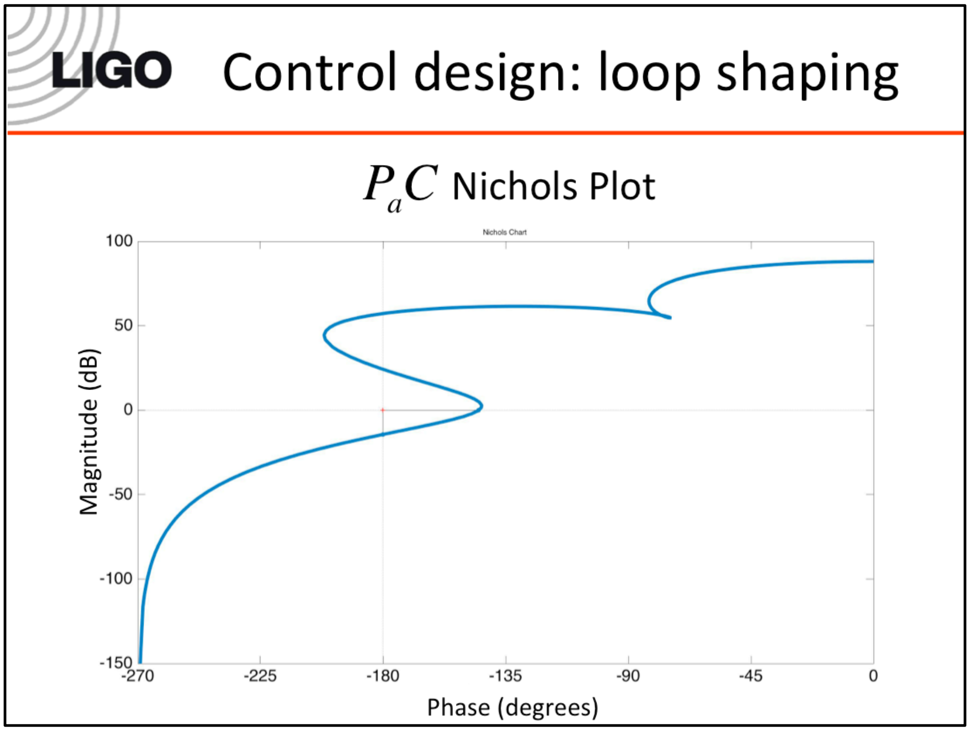
Then, we want to get more gain at low frequencies to reduce the influence of seismic noise where it is greatest. To do this, I have added a pole-zero pair, with the pole at 0.01 Hz and the zero at 2 Hz. We could do this without the zero, but the zero minimizes the phase loss at the UGF. However, the phase is still not quite stable at the UGF.



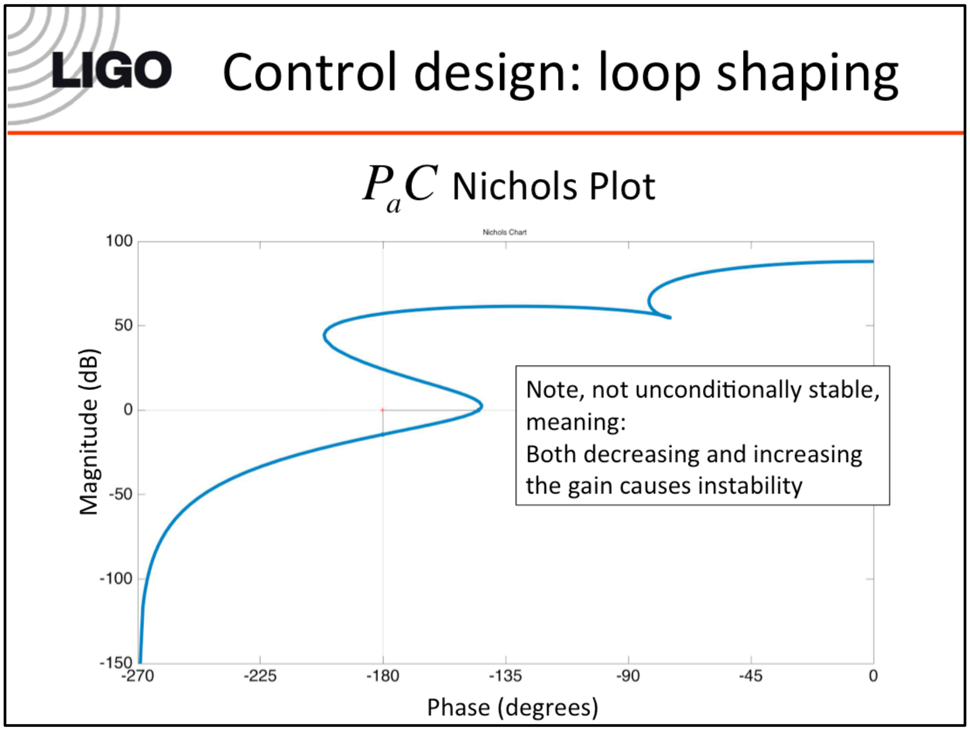
So we add another zero-pole pair, just to boost the phase around the UGF. These are centered around the UGF, with the zero a factor of 3 below, and the pole a factor of 3 above. Note, there is a compromise in doing this. If you flip between this slide and the previous one, you see that we have lost some low frequency gain. However, that gain does us no good if the system is unstable.



Finally, we add a single pole at 100 Hz, just to filter away high frequency noise. In general it is good practice to ensure your filters have more poles than zeros, just so the response approaches zero at infinite frequency. This not only filters noise, but helps ensure the actuators don't consume so much voltage that they saturate.



This is the Nichols plot of the resulting loop gain. The area under the curve does not include the -1 point, so it is stable.



However, the -1 point is surrounded above and below by those lobes. This means that if we either increase or decrease the gain (curve moves up and down respectively) the system will go unstable. In this case we say the system is not ***unconditionally stable.*** An unconditionally stable system is one that is stable for all gains below a certain value. Non-unconditionanly stable systems are fine, we use them all the time, you just have to 1) be sure that the value of the gain is not going to change, and 2) be careful how you turn the feedback filter on.

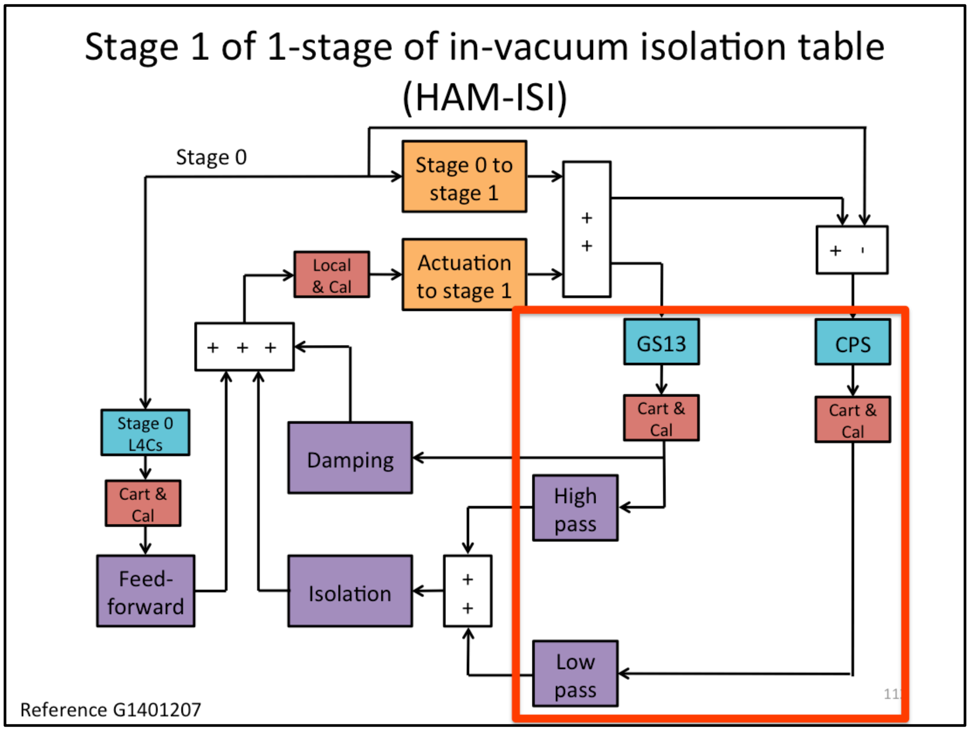
You can't just ramp the filter's gain on like we usually do in this case. So you either have to turn it on all at once, or turn it on in pieces, where you start with an unconditionally stable piece, ramp its gain on, and then turn on the non-unconditionally stable piece.

Lecture 2 – Part 3

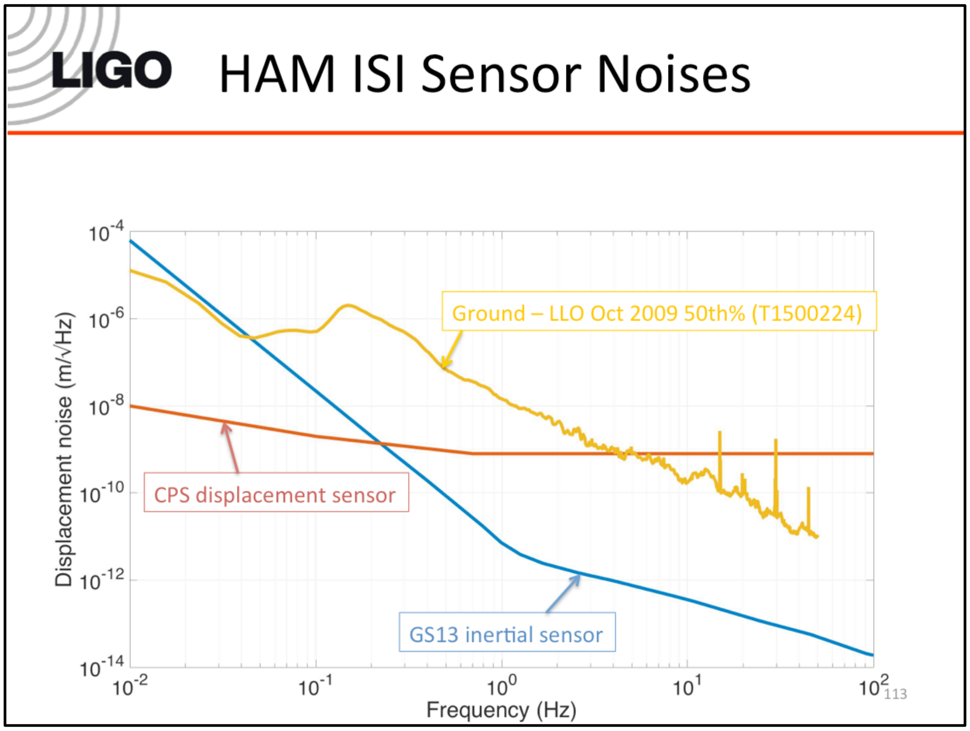
Sensor Blending

G1600726

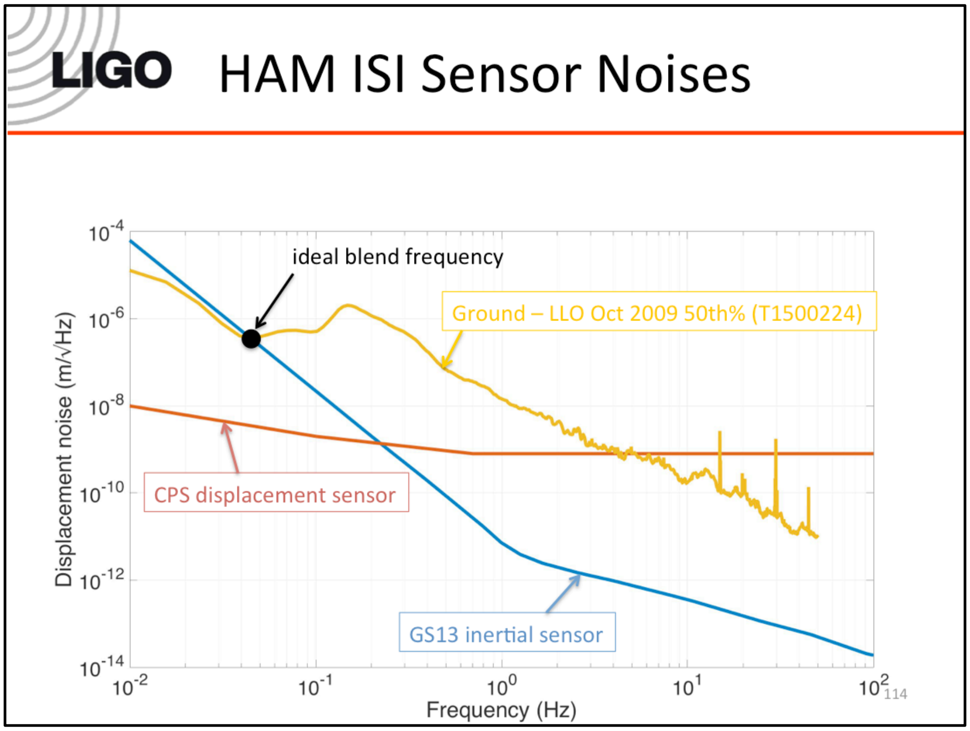
111



The final part of this loop is the sensor blending. The HAM-ISI uses a combination of relative capacitive displacement sensors (CPSs) and inertial sensors (the GS13 is a commercial geophone). These must be blended together into a single 'super-sensor' that we can send to the isolation feedback.



This plot illustrates why we need sensor blending. It shows the displacement noise of the CPSs and the GS13, compared to a typical ground motion spectrum. Note that at low frequencies the CPS noise is better, and at high frequencies the GS13 noise is better.



We’re most interested in inertial isolation though, so the important crossover frequency is not where the sensor noises intersect, but where the inertial sensor noise intersects the ground motion. Here it is about 0.045 Hz. So we should use the displacement sensor below this, and the inertial sensor above. Thus the best ‘blend frequency’ is 0.045 Hz.

Note this ignores the issue of tilt-horizontal coupling all inertial sensors are subject to, which is not within the scope of this lecture. In general, horizontal inertial sensors are sensitive to being tilted. This sensitivity is proportional to g/w^2, where g is gravity and omega is 2*pi*frequency. So at low frequencies, tilt becomes a serious issue. This often limits how low we set the blend frequency (depending on the weather). For more information see P080073.

Blend filter design

- Low pass the inertial sensor, high pass the displacement sensor
- Low pass + high pass = 1

To do the blending, we need to low pass the displacement sensor, and high pass the inertial sensor. Then we sum the two outputs to make a single sensor signal. Thus, to preserve units, the sum of the low pass and high pass must be 1 (assuming the two sensors are already calibrated to the same units). Strictly speaking, we don't need to enforce the sum = 1, but not doing so would make the loop gain (and stability) dependent on the blending. We prefer to have the loop gain be the same for all blend configurations. With this sum = 1 requirement, the filters are said to be 'complementary pairs'.

Blend filter design

- Low pass the inertial sensor, high pass the displacement sensor
- Low pass + high pass = 1

Simple approach

B_{LP} Make some low pass filter

116

The simplest approach to designing blend filters with the sum = 1 constraint is to simply choose a low pass filter for the displacement sensor (or the high pass).

LIGO

Blend filter design

- Low pass the inertial sensor, high pass the displacement sensor
- Low pass + high pass = 1

Simple approach

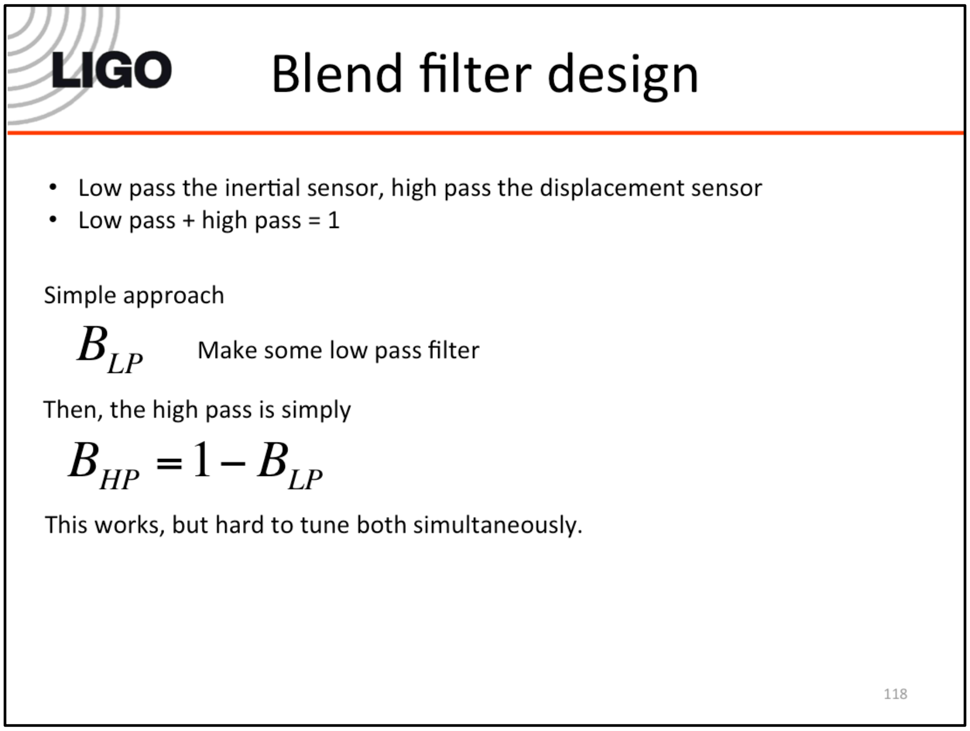
B_{LP} Make some low pass filter

Then, the high pass is simply

$$B_{HP} = 1 - B_{LP}$$

117

Then the high pass for the inertial is just 1 – the low pass. (Or vice-versa).



This is fine, but then the high pass design completely depends on the low pass. We would prefer to have the freedom to design both independently.

LIGO

Blend filter design

- Low pass the inertial sensor, high pass the displacement sensor
- Low pass + high pass = 1

Simple approach

B_{LP} Make some low pass filter

Then, the high pass is simply

$$B_{HP} = 1 - B_{LP}$$

This works, but hard to tune both simultaneously.
Try this instead:

$$B_{LP} = \frac{B_{LP\_prototype}}{B_{LP\_prototype} + B_{HP\_prototype}} \qquad B_{HP} = \frac{B_{HP\_prototype}}{B_{LP\_prototype} + B_{HP\_prototype}}$$

119

So instead, we take a different approach. We design the low pass and high pass we would like to have. These are called the 'prototype' filters. Then we normalize each by the sum of these prototypes. Therefore, the resulting low and high pass filters must be complementary pairs.

Blend filter design

But be careful!

$$B_{LP} = \frac{B_{LP\_prototype}}{B_{LP\_prototype} + B_{HP\_prototype}}$$

120

There is a price we pay for doing this though!

Blend filter design

But be careful!

$$B_{LP} = \frac{B_{LP\_prototype}}{B_{LP\_prototype} + B_{HP\_prototype}}$$

$$B_{LP} = \frac{1}{B_{LP\_prototype}} \frac{B_{LP\_prototype}}{1 + B_{HP\_prototype} / B_{LP\_prototype}}$$

This looks like a closed loop TF,
where the 'loop gain' is the ratio of the prototype filters.

121

If you factor out one of the prototype filters from the denominator, you get a denominator with 1 + something.

Blend filter design

But be careful!

$$B_{LP} = \frac{B_{LP\_prototype}}{B_{LP\_prototype} + B_{HP\_prototype}}$$

$$B_{LP} = \frac{1}{B_{LP\_prototype}} \frac{B_{LP\_prototype}}{1 + B_{HP\_prototype} / B_{LP\_prototype}}$$

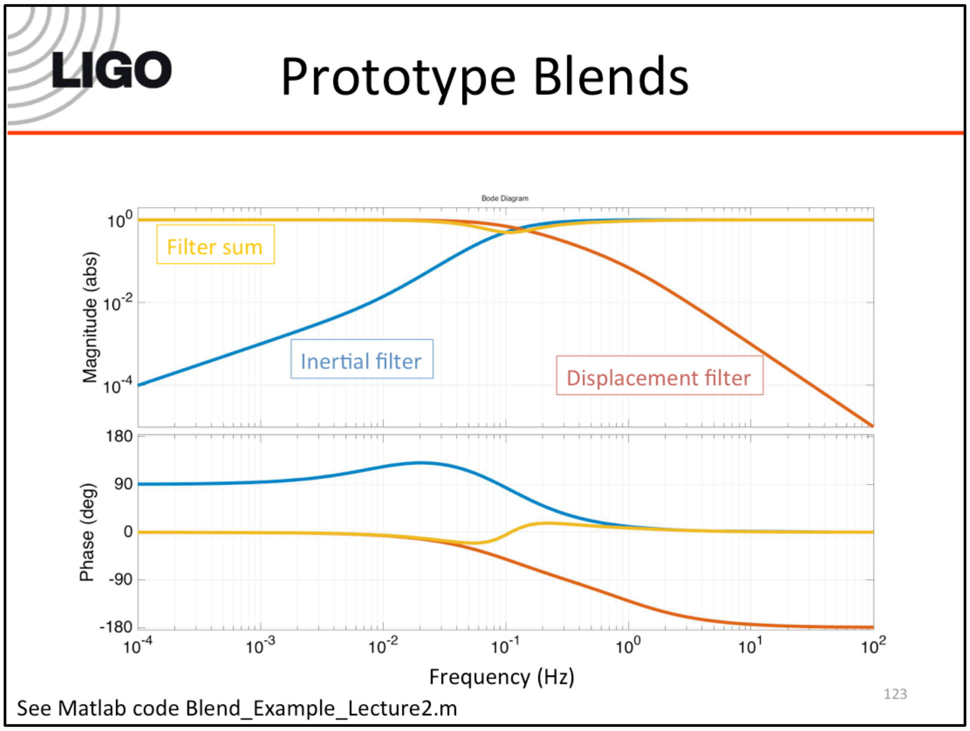
This looks like a closed loop TF,
where the 'loop gain' is the ratio of the prototype filters.

- We must watch out for stability.
- In practice, just keep the filters within 180 degrees of each other when their magnitudes cross.
- With this approach we've traded some stability for more design parameters

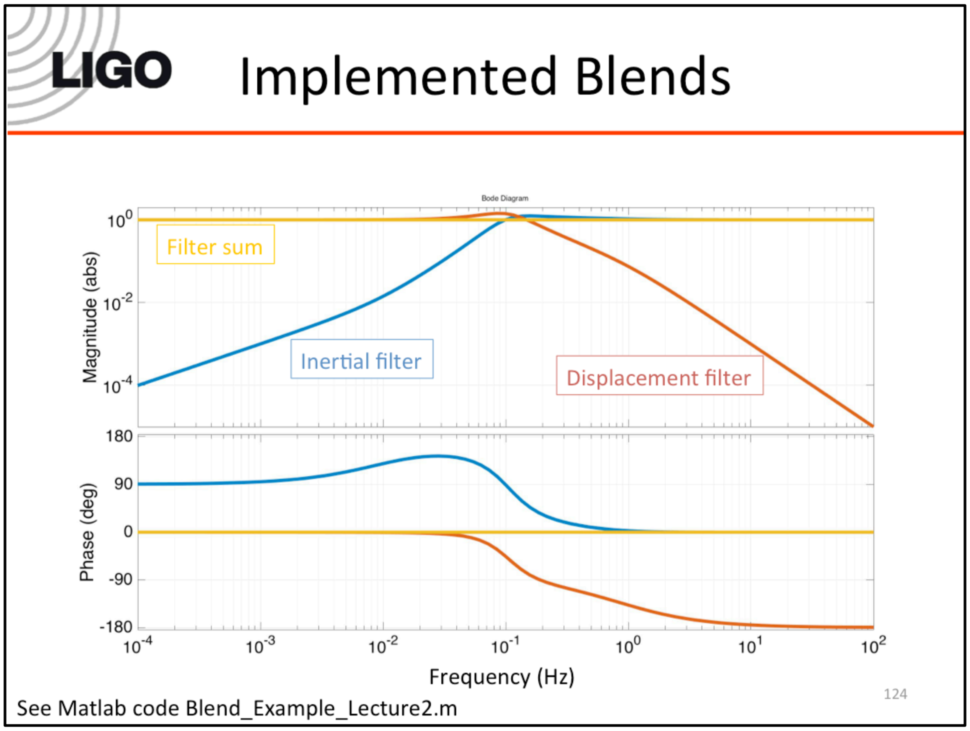
122

This is kind of like when we have 1 + loopgain. Actually, it is exactly like that. And all the same stability rules apply. Thus, we must ensure that the two prototype filters are stable with each other. In practice, you just need to make sure their phases are within 180 degrees of each other when their phases cross.

So we have made a trade where we sacrifice some stability to generate more design parameters.



Here is an example of two prototype blend filters. See the example matlab code with this lecture.

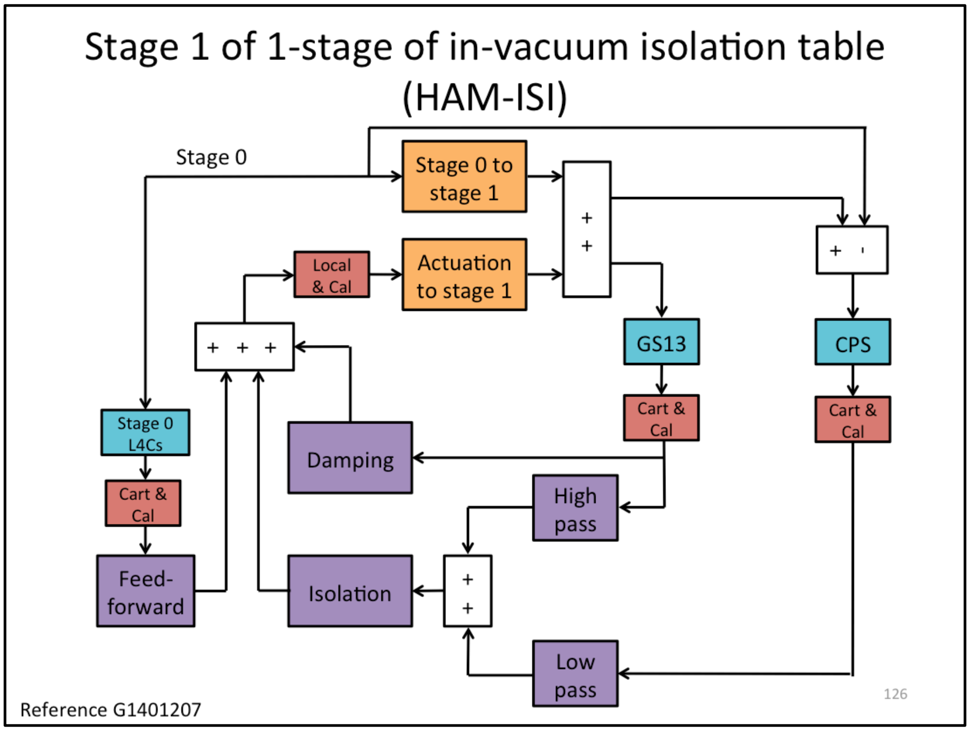


Then, when you do the normalization to make the complimentary pairs we actually implement, you get these filters.

Lecture 2 Summary

- Seismic feedforward control depends only on the system's connection to the ground.
- For feedback stability the abs(phase) < 180 when the magnitude drops below 1
- Sensor blending uses the displace sensor at low frequencies, the inertial sensor at high frequencies. Stability rules apply.

G1600726 125

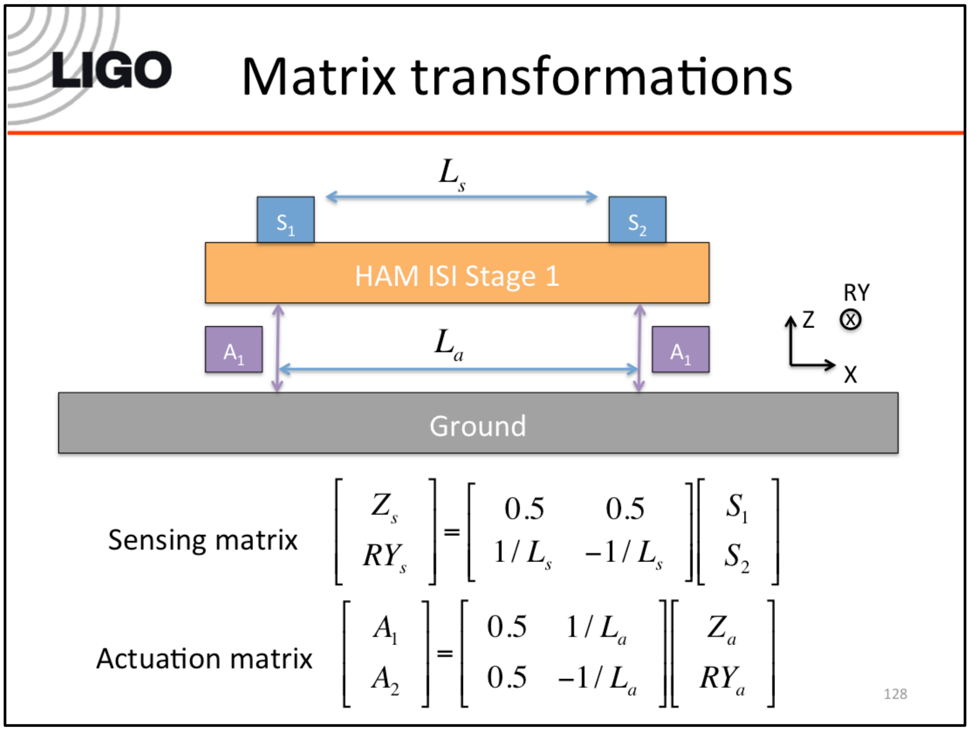


Stage 1 of 1-stage of in-vacuum isolation table
(HAM-ISI)
Stage 0
Stage 0 to stage 1
+
+
+ -
Local & Cal
Actuation to stage 1
+ + +
GS13
CPS
Stage 0 L4Cs
Cart & Cal
Cart & Cal
Cart & Cal
Damping
High pass
Feed-forward
Isolation
+
+
Low pass
126
Reference G1401207

Lecture 2 – Backups

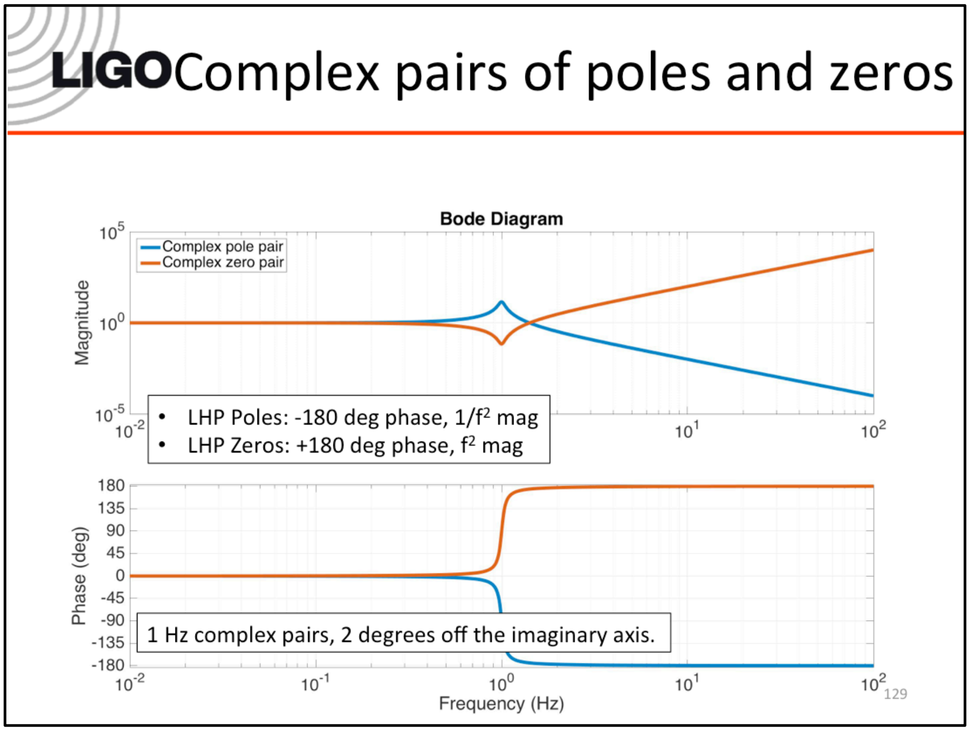
G1600726

127



This backup slide describes the matrix transformations that are shown in the example HAM-ISI control block diagram. Consider that we have 2 vertical sensors and 2 vertical actuators. We can combine the signals of the sensors to generate a Z sensor signal and an RY sensor signal. For Z, we simply take the average of the 2 (S1+S2)/2, which keeps the Z signal in the same units as the individual sensors (e.g. meters). For RY, we take the difference, normalized by the length between the sensors, so that we have units of rotation (e.g. radians). These transformations are grouped into a matrix, as shown here by the sensing matrix.

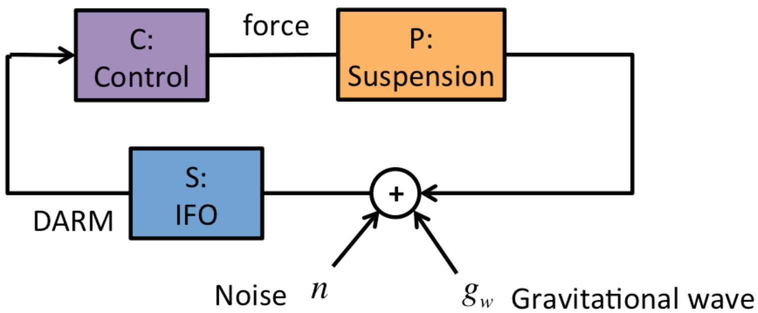
Similarly, we can combine the actuator signals to generate Z and RY actuator signals, as shown by the Actuation matrix. The actuation matrix is the transpose of the sensing matrix if La = Ls (or more generally, if the sensors and actuators are collocated).



Complex pairs of poles and zeros are in many ways just like 2 repeated real poles and zeros. The main difference is that they have a damping term, which depends on the angle they make with the imaginary axis of the complex plane. In this case the angle is only 2 degrees, so the damping is small, where Quality factor = 1 / (2 * sin(angle)). Lightly damped poles have a large resonance peak, and a short phase transition from 0 to -180 degrees. Lightly damped zeros have a large 'notch' feature, and a short phase transition from 0 to +180 degrees. For zero damping, the pole peak goes to infinite magnitude, and the zero notch goes to zero magnitude. The phase transition is instantaneous.

LIGO Extracting the GW signal

More detail in G1600412



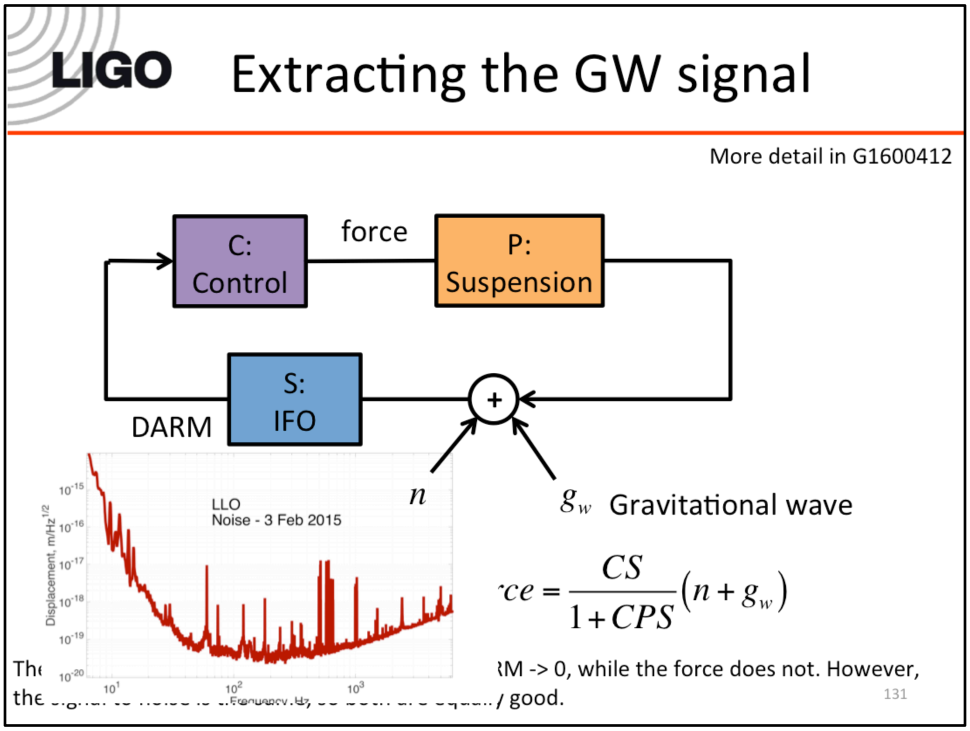
$$DARM = \frac{S}{1+CPS}(n+g_w) \qquad force = \frac{CS}{1+CPS}(n+g_w)$$

The GW exists in both signals. For large gains, DARM -> 0, while the force does not. However, the signal to noise is the same, so both are equally good.

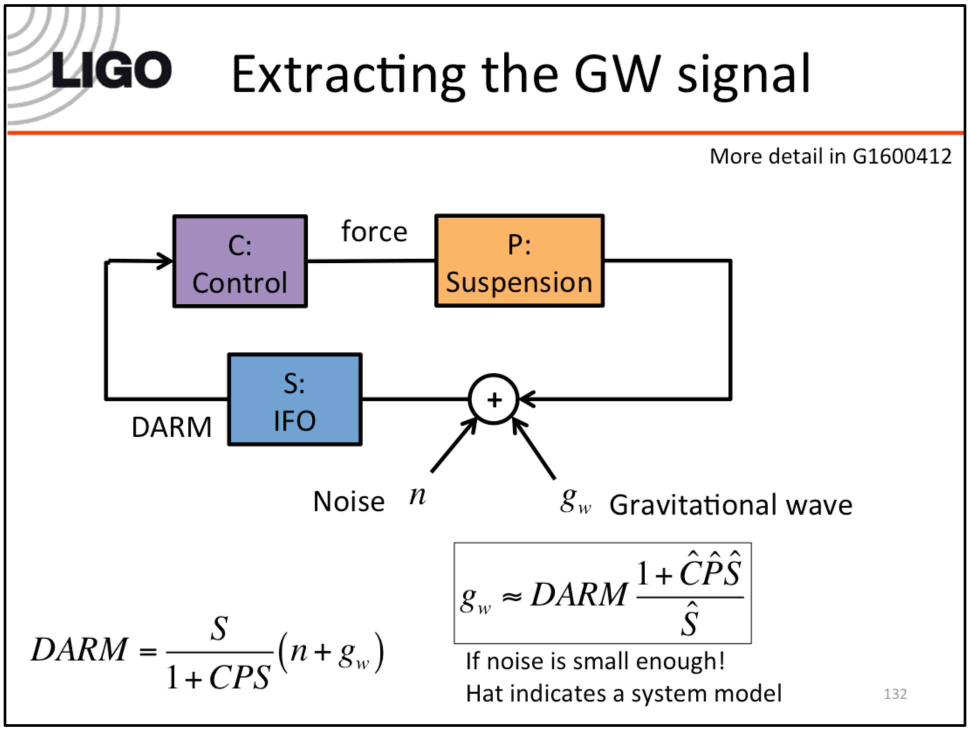
130

The arm cavity feedback control holds the arms at a fixed length. So how do we get the gravitational wave signal from the interferometer if the arm lengths aren't allowed to change?

First, the feedback control doesn't zero out the length fluctuations, it just makes them smaller. The length fluctuations are suppressed by 1+loopgain (if S = 1). So if we have a good model of the loop gain (which can be measured easily enough), we just multiply the measured photodiode signal (DARM) by 1+loopgain. You can think of this more physically as taking into account how much force the control is applying to minimize the length fluctuations. The loop also happens to suppress the cavity noise by the same amount as the gravitational wave, so the signal to noise ratio of the gravitational wave is unchanged by the feedback. We could also get the signal from any other place in the loop. For example, the gravitational wave signal also appears in the feedback force we apply with the actuators.



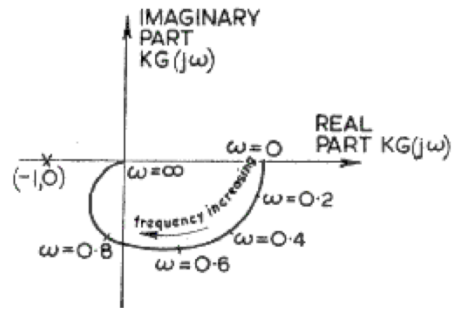
Here, the cavity noise is the same curve we see when we look at the aLIGO sensitivity plots.



If the gravitational wave is bigger than all other noise sources, than our measurement of it is the measured DARM signal times our model for the inverse of the closed loop transfer function. Here, the hats indicate models of the real system. It is the job of the aLIGO calibration group to ensure these models are good representations of what we have.

Nyquist plot

- These plots are traditionally shown over Nichols plots, but are harder to look at since they can't be put in logspace.
- Stability is achieved by not circling the -1 point



- If a plant has unstable poles, then the rules change. See Ogata text.

133

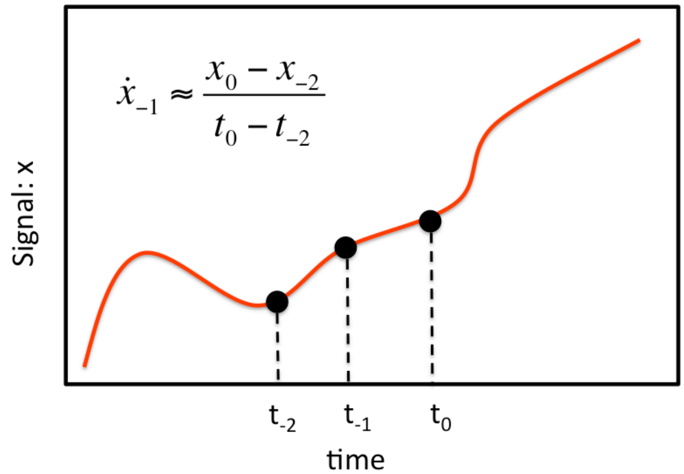
LIGO

Closed Loop TFs

$$x = \underbrace{\boxed{\frac{1}{1 + P_a C}}}_{S} P_g x_g$$

S is called the 'sensitivity' TF and is common to all closed loop TFs in a loop

Another explanation for causality



A zero-pole pair (where the pole is at higher freq) is like a derivative approximation, where the pole determines the effective sampling time. Deleting the pole is the same as setting it to infinite frequency, which makes the time step = 0, which means we'd effectively be seeing the future by knowing the slope instantaneously.