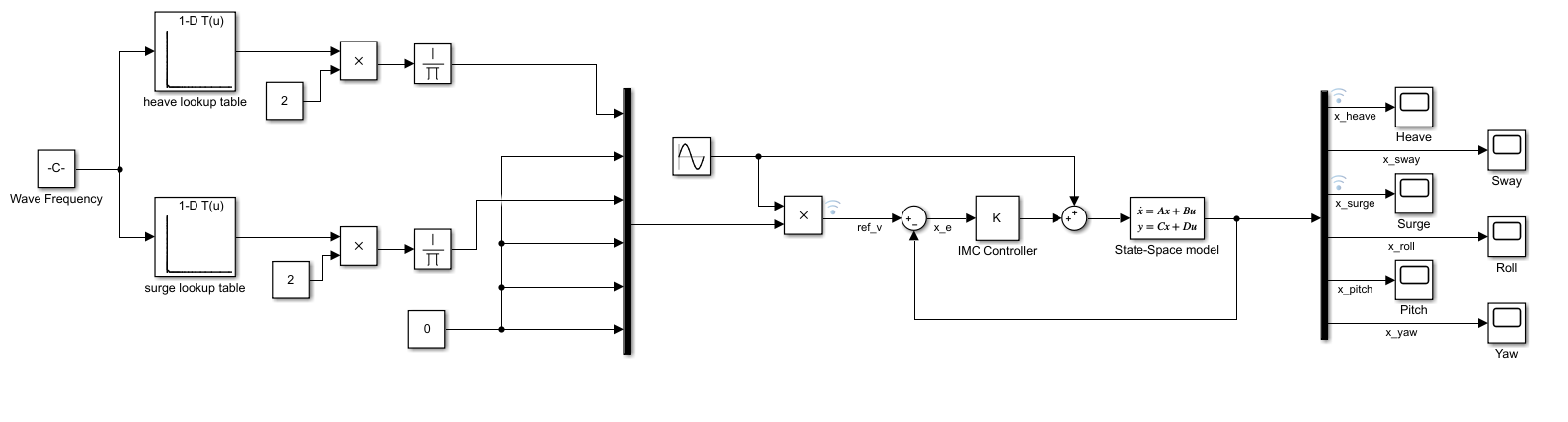
First I implemented the control loop. I didn’t originally understand that the inverse was necessary.

The top level looks like this:



On the left hand side, a simple sinusoid and its frequency are fed into the system. The sinusoid is defined by the script inputfile.m. There are also lookup tables. Values calculated from the transfer functions for radiation force for surge and heave are used in these tables. They map the frequency of the incoming wave (excitation force) to the expected amplitude response of the system. (Still need to check the dimensions of these vis a vis decibels or physical values). Some simple operations perform the operation 1/2B(w) where B(w) is the amplitude response. (See Ringwood).

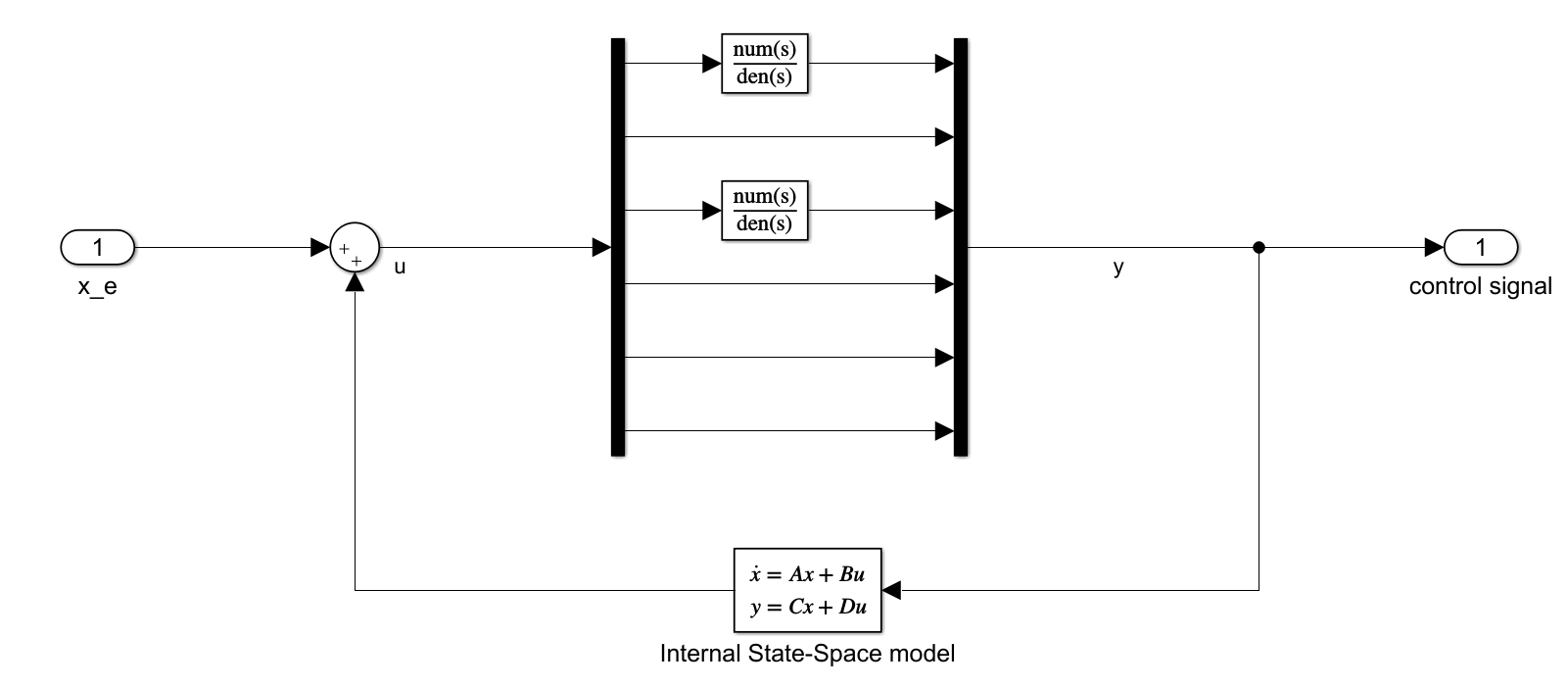
This is Muxed with zeroes for everything else then multiplied by the Excitation force to give the reference velocity. This is all the same as with Ringwood’s paper but simplified by a sinusoidal wave, removing the need for the predictive EKF and adaptive law to select the ‘dominant’ wave frequency.

In this way it can be confirmed whether or not the drifting problem exists outside of these elements.

In the middle is the feedback loop. It’s a negative feedback controller. The controller ‘K’ contains all of the internal model control which relies on a positive feedback loop. A state space model is used to model the plant. Eventually this will be replaced by the full WecSim environment. Currently the internal model and the ‘external’ model are identical so near perfect prediction should be achievable.

On the right is the output and scopes to view the system responses.

One layer down is the IMC Controller. It looks like this:



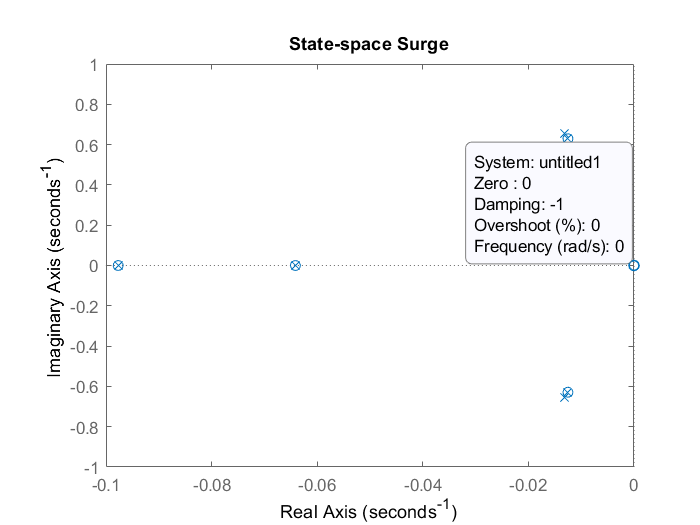
The signal is de-muxed so I can get at the Heave and Surge signals. The transfer functions are the result of turning the SS model into a system of transfer functions, flipping/inverting them, then multiplying them by a filter to make them proper. This is then re-muxed and given as output. The effect is that only the Heave and Surge motions are controlled.

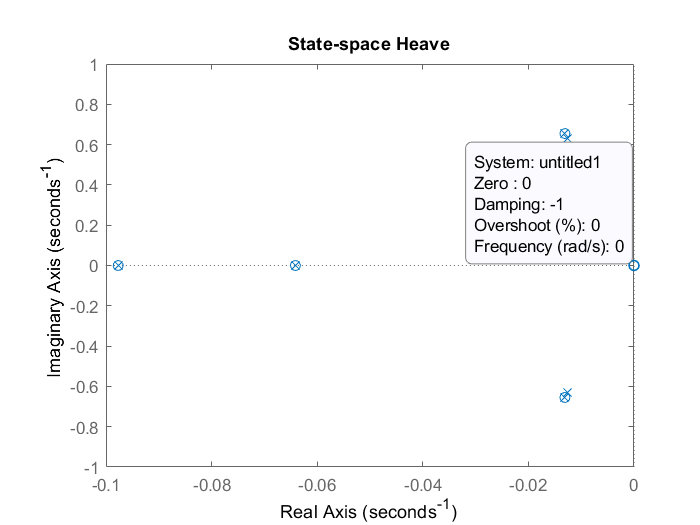
The design of the filter; the stability of the inverse; and how close the inverse-filter combination is to a ‘true’ left-hand inverse are critical to the success of this control scheme. It’s taken a lot of effort to fine-tune these things. I’m also investigating another inverse strategy called the Massey-Sain algorithm, but it seems unlikely to work.

To invert the system the function ‘ss2tf’ is used in Matlab to turn the state space model into a system of transfer functions. The output is very high order, but the step response for the relevant motions is identical to the SS model:

First off, it is necessary to confirm what Ringwood asserts. He claims that in his model of the plant there is a zero at (0,0) in the phase diagram. This will become a pole in the inverted system (because you just flip the TF) which needs to be offset to achieve conditional system stability.

A pzmap of the ss2tf version of the SS model shows that this is the case for us too in both heave and surge:





In both controlled motions this zero exists and will need to be balanced.

Ringwood says that to make the inverse TF physically realisable (aka proper) it must be multiplied by a filter ‘F(s)’. Due to the poles at zero in the inverse TF the numerator of F(s) must have an s (i.e. be at least order 1) to balance it.

In our case the inverse transfer function is order 37 on top and order 36 on the bottom. So the lowest order filter that meets these requirements is order 1 on top and order 2 on the bottom. The bandpass filter offered by Ringwood fits these requirements.

This rules out other attempts of mine such as PID control and PI control. Can’t have D because that would make the numerator and denominator the same, leaving the invTF improper. PI is realisable but it doesn’t cancel the pole at zero.

There may be other filters that meet these requirements, but the bandpass filter has the added benefit of removing the amplitude shoot-offs at high and low frequencies that result from inverting the SS Transfer function.

The heave SS TF and Inv TF look like this on a bode plot:







Similarly, in Surge:







Both heave and surge have a resonant peak at about 0.63rads roughly. Heave has a little peak a bit higher at 1 rads. The shape of the inverse is ideal but as expected it shoots to infinite amplitude at high and low frequencies. The bandpass filter can eliminate this, but high and low limits must be tuned for optimal system response (minimum phase lag). Some lag is unavoidable.

An additional parameter that was added was a gain, K, which was tuned by eye to about 1.4, until it was realised that the gain was just the numerator.

(Might need to be tuned separately for each motion?)

The general form of the bandpass filter is:

Where is the gain, is the high cut-off frequency and is the low cut-off frequency.

I observed by eye that the gain for the system was wrong, possibly due to some scaling issue or possibly just because that’s what the SS model gain does and needs to be corrected for. So needs to be tuned.

I tried with a bunch of cut-off frequencies without much success. Eventually I realised that I was giving wave frequencies in Hz, but the system was doing everything in rads-1, so all my proposed cut-off frequencies were off by a factor of .

From Andy’s knowledge of the sea data and sea wave dynamics apparently waves have a max frequency of 1Hz. Presumably there’s not much of a minimum on this. My first ‘serious’ attempt at the bandpass filter used cutoff frequencies of ‘1’ and ‘0.2’. I thought these were in Hz but actually they were rads. Nonetheless I was able to tune the gain to ‘7’ by eye using the Simulink data inspector. I plan to do a ‘final tune’ using a round or two of newton Raphson.

Checking with various different frequencies it seems that tracking is good at high frequencies but very poor at low frequencies??? Phase lag is also massively worse at low frequencies, but this be a proportional issue.

Bode Plots of TF, orig inverse, and filtered inverse in heave and surge:

It was found that the ss2tf function produced transfer functions of a very high order, it was suggested that the ident toolbox could be used to create a lower-order transfer function that would behave in the same way but be easier to compensate for.

Instead I went off the rails and implemented a discrete time inverse that works very well but with a lag of one time step due to the methodology:

## Massey-Sain algorithm.

The Massey-Sain algorithm offers a method for testing whether a system of linear differential equations (aka a state space model) is invertible. It then allows for the construction of a suitable inverse with minimum delay 1.