Notes 05/06/2020

Requirements:

* Thermal Fluid Outlet Temperatre MUST stay between 680-700K
* Cooling Medium enters jacket @ 600K
* Require steady minimum methanol production of 1 kmol/h after 10s
* Control flow rate, everything else is disturbance
* Reactant inlet flow rate 80 kmol/h
* Cooling medium flow rate 400 kmol/h
* Reactant inlet temp 760K +/- 30 K
* Cooling Medium inlet temperature +/- 30K

Assume network of FOPTD transfer functions g11, g12, g21 and g22 between Reactant Flowrate (RF), Coolant Flowrate (CF), Methanol Flowrate (MF) and Coolant Outlet Temperature (CT).

Notes on Step Tests for Reactant Flow Rate effecting Methanol flow via g11 and coolant outlet temp via g21

* Average time delay for g11 is 4.431s, average time constant is 6.149s.
* Both of these values vary polynomially with the step change. Time delay seems to vary via 3rd order polynomial, time constant via 5th order.
* Average gain is around 1, and this does not vary substantially
* Try first with just average time delay/ time constant, then try adjusting these values based on input
* Average time delay for g21 is 4.430s, average time constant is 5.909s.
* These values also vary according to a 3rd/5th order polynomial respectively.
* Average gain is 0.667 and this varies minimally

Note: At this point I discovered that the methanol output was out by a factor of 100. This means that the gain of g11 should be 0.01.

Notes on Step Tests for Coolant Flow Rate effecting Methanol flow via g12 and coolant outlet temp via g22

* Seems to be no effect of CF on MF, g12 = 0
* Average time delay for g22 is 4.165s, average gain is -0.138 (inverse response as expected)
* Time constant varies, average is 3.045s including one anomalous value. Excluding this value brings the average to 2.949s and the values approximate a 2nd order polynomial trend.

Version 1 does not work, the feedback controllers simply continue to increase the flowrates. (NB controllers tuned using SIMC). Version 2 also does not work, the addition of gc3 gives some weird behaviour. As expected, control more complex than simple feedback loops are required.

One last thing to try with the simple feedback approach will be to account for the varying time delays and time constants, dynamically calculating the SIMC tuning parameters (probably won’t work)

Whilst Programming version 3, realised that the PI controllers were not set to ideal. This was changed, but had little effect.

Also somehow my spreadsheet had reverted to having the wrong gain for g11 (not accounting for the fact it was out by a factor of 100). Adjusting this allowed version 1 to reach the set points, however the transient was inadequate. Version 2 also now reaches the correct steady state but has an (even worse) transient.

Version 3 did not work, avoid using variable parameters. Next avenue should be to build on version 1 (probably the best version so far) with smith predictors.

Smith predictors might do something with the steady-state, but the transient is still inadequate. For version 5 I’ll experiment using Cohen-coon PI tuning and version 6 I’ll use cohen-coon PID.

Version 5 Still takes over 30 seconds to reach steady state. I also suspect the scope is currently in the wrong place.

To begin with version 6, I added a max module to ensure that flowrates could not go negative, as this was something I was seeing. The system still took longer than 30 seconds to reach steady state, and much longer than 10 seconds to reach 1kmol/h methanol.

The implementation of PID made the system unstable. As an experiment I reverted to PI control but kept the PID tuning parameters, the result was much more oscillatory than before.

Two more things I want to try at this point are feedforward control, and cascade control (which on further thought probably wouldn’t work). Feedforward would require step tests in the disturbances to determine their transfer functions, which will be time consuming so I’ll try that tomorrow.

For version 7 I will take version 5 and use SIMC tuning with lambda = 3Td. The result takes much longer to reach steady state, indicating cohen-coon is the better tuning method. Thus I returned to cohen-coon tuning and varied the parameters to see if any improvement could be made. The best parameters I could find still took ~30 seconds for MF to reach 1kmol/h. The inverse response may effect this, as could the fact I’ve neglected g21.

For fun, I implemented a harsh step change of +30K in both temperatures at 60s to see how my tuned controller fared. Although the system went out of range, it quickly returned and I was impressed. However I’d ideally like to avoid going outside at all for pertubations in the range of +/- 30K. Running the same disturbance with the original cohen-coon params, the system took longer to return.

To improve further I could add feedforward or include g21. Version 9 will do the latter, using a cohen-coon tuned PI controller as a basis with a smith predictor accounting for g21, and version 10 will seek to further tune gc3. Version 9 shows a significant improvement in terms of speed, but perhaps a little bit less stable.

Unhappy with this instability, version 10 originally returned all controllers to their cohen-coon parmeters. This was even worse, so a compromise is evidently needed. I further tuned all 3 controllers until I was happy.

Of course, saving of the files went tits up so version 9 doesn’t actually have this controller. I can’t seem to recreate my initial success so have reverted to not accounting for g21, instead only controlling g11 & g22. In version 11 I will try incorporate g21 into my smith predictor. This shows some improvement.

One thing that occurred to me is that the methanol flowrate is to be maximised; nothing specifies that the set point needs to be 1kmol/h. I initially tried changing the set point to 2kmol/h, however this led to a highly oscillatory transient and inadequate response to a strong pertubation. 1.5 was acceptable, so I set the set point to 1.4 kmol/h to be sure.

Version 11 reached 1 kmol/h production rate in 10s and entered the temperature range in 18s. Version 12 makes this controller fit the test.

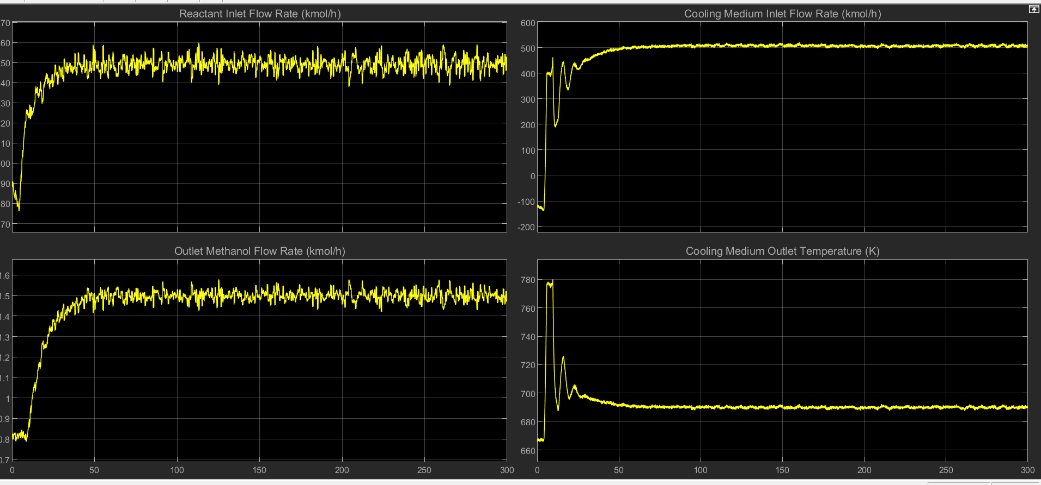
Testing version 12 is ok, but the controller still falls out of range on occasion. I may try when I next have time to run some other versions with the test to see how they fare. Perhaps going back to a less-tuned version and increasing the MF set point is worth a try. I should try ZN tuning, could look at MPC, and could try ff control mayhaps.

Considering that I started at 2:30, I took 9 hours to produce a controller that works as well if not better than our original one. bruh. (excluding 1-2 hours for dinner etc, and 3-4 hours were step tests).

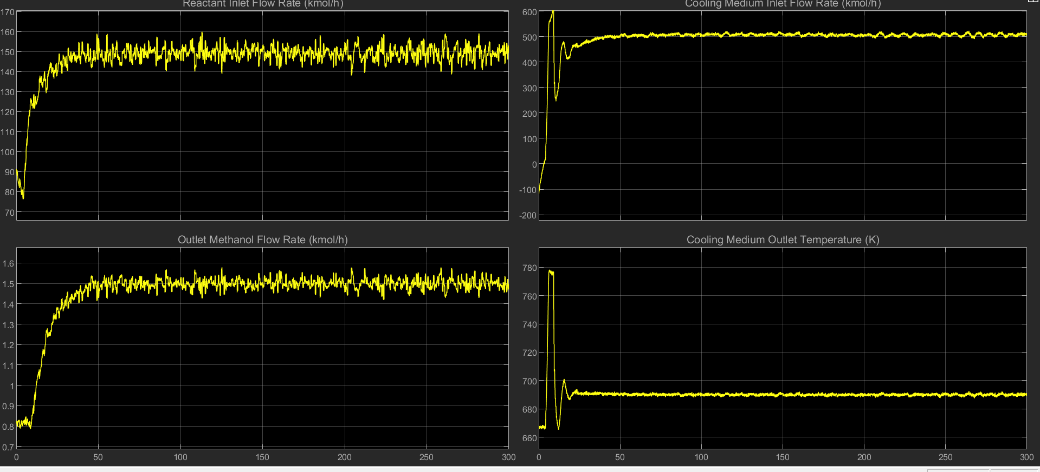
07/06/2020

First thing I’m trying today is to see how well a less tuned (i.e. using straight Cohen-coon params) controller functions if I increase the methanol flow set point. Version 13 built on version 9, and used the Cohen-coon parameters and initially a set pt of 1.5 kmol/h for methanol flowrate. Version 9 might not have been the correct starting point as the CT smith predictor didn’t account for g21, so this was amended.

Version 13 without amended smith predictor.

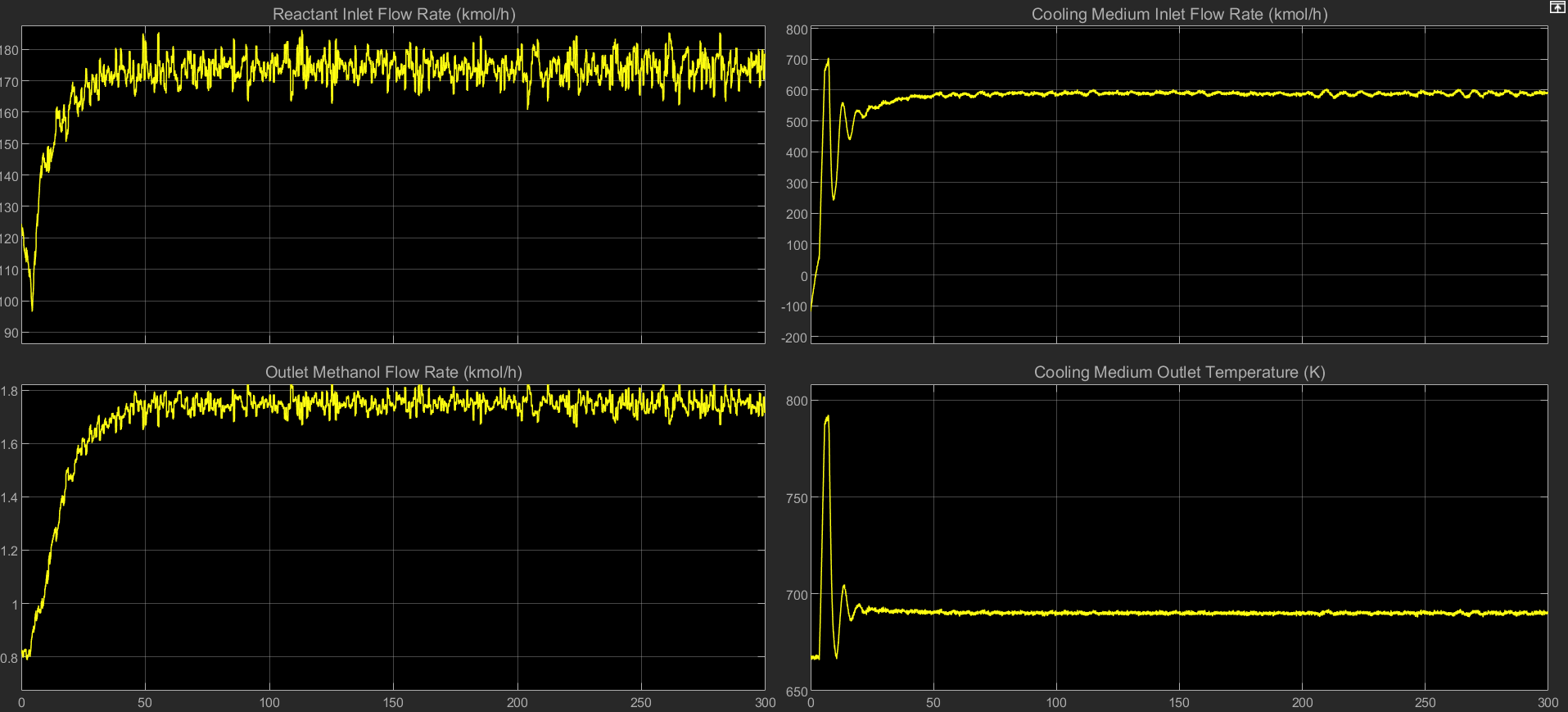


Version 13 with amended smith predictor (some improvement).

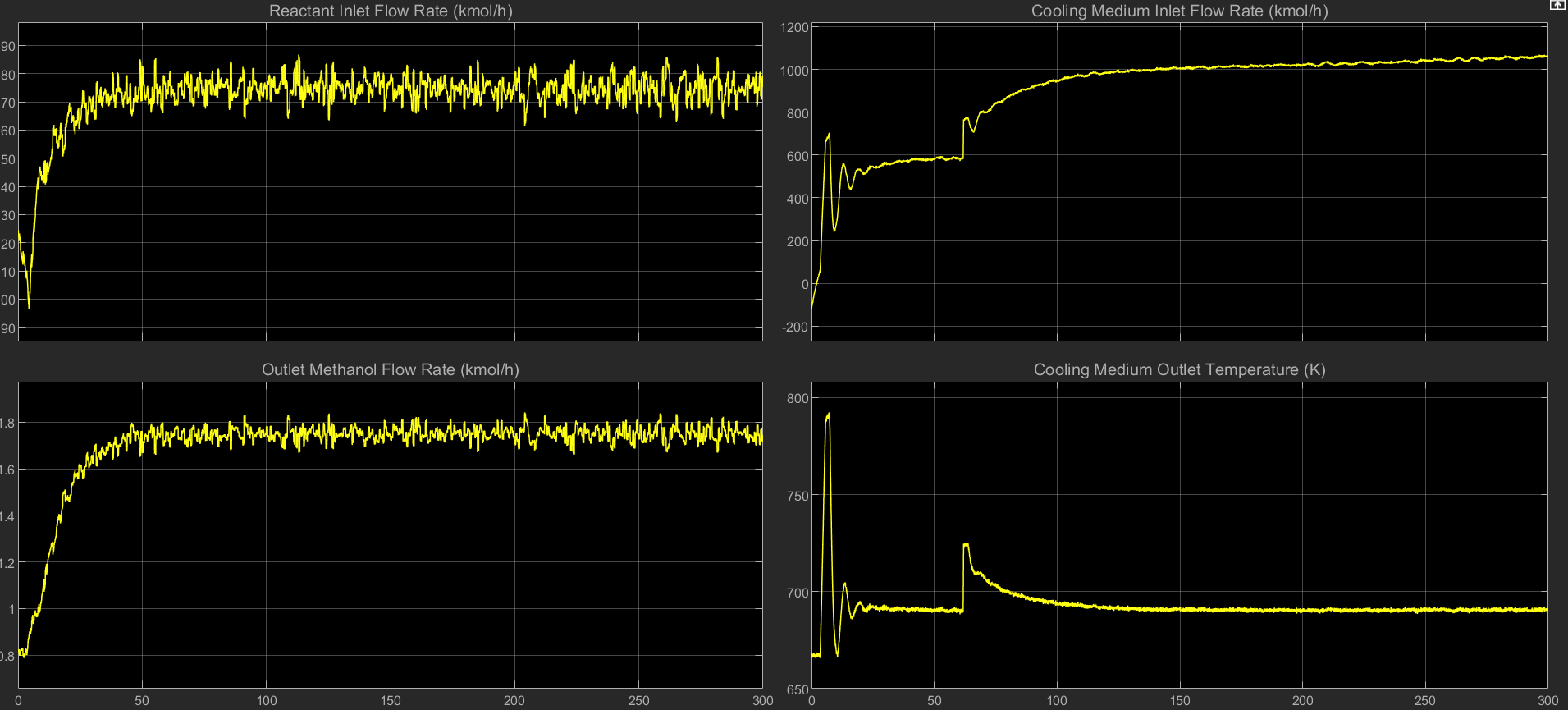


This set point takes 12 seconds to reach 1 kmol/k production. A set point of around 1.7-1.75 kmol/h is required to achieve 1 kmol/h within 10 seconds. A higher set point is of course desirable so long as stability is not affected.

Set pt changed to 1.75 kmol/h MF

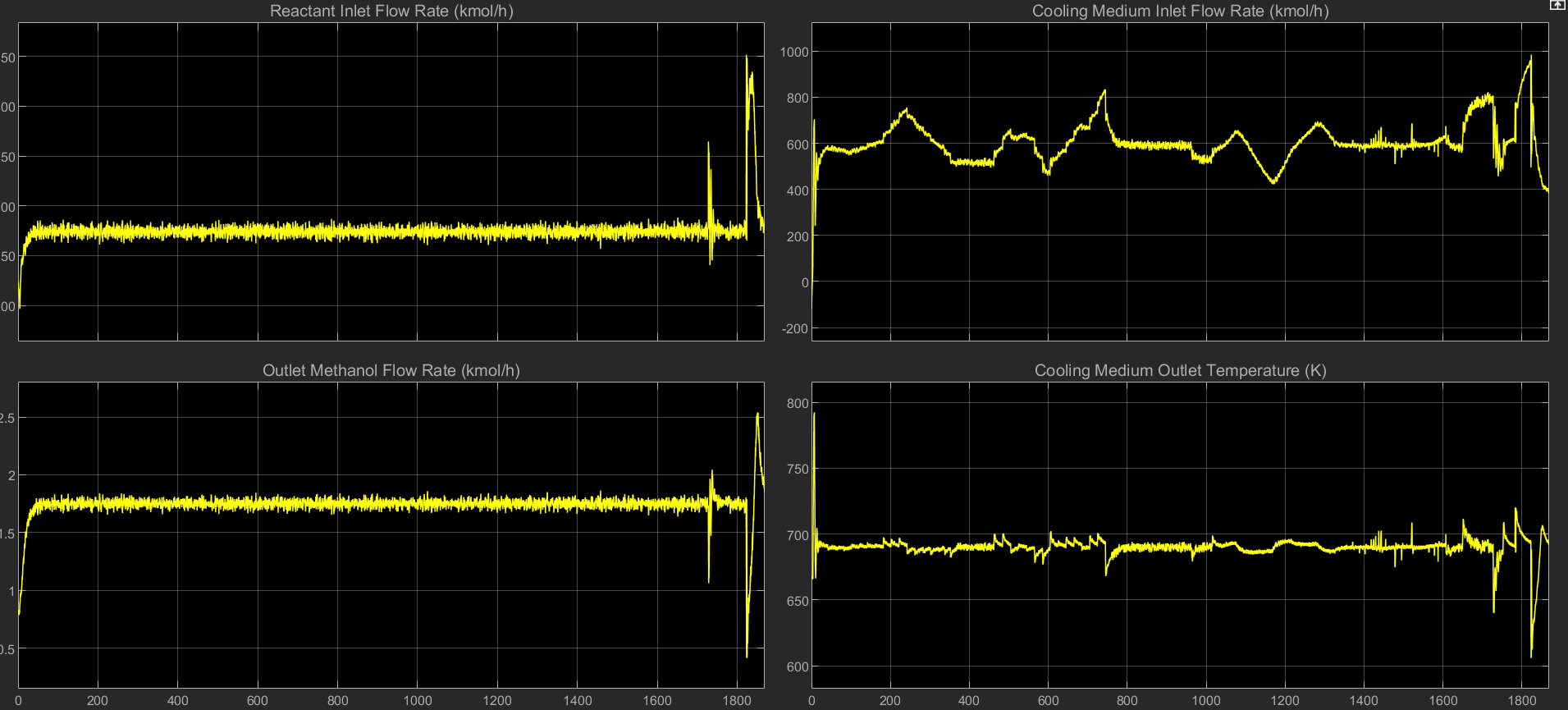


This was subjected to a harsh step change to gauge its response, and the temperature took a very long time to return to an acceptable range. This makes me think that the less tuned version will fare worse in the test than version 12.



Indeed, version 13 is a lot less effective when subjected to the controller test than version 12. A reasonable step forward would be to add some logic to track the amount of time spent outside the appropriate range, and thus quantify how well the controllers perform.

Result of initial version 13 test:



I added another scope with two feeds, one giving the deviation outside the range over time and one giving the integrated absolute error. The final version of the latter will make a good means of assessing a controller. Another good value to track would be the total methanol production.

Version 13’s final integrated error was 1192. It should be noted that version 13 remains stable even at the end. A simple integrator connected to the methanol output tracked the total amount of methanol produced. Version 13 produced 0.906 kmol methanol in 1869s, an average rate of 0.485 mol/s

The part of the controller that calculated these values was copied over to version 12, which was re-run to get an idea of how version 13 compared. Version 12 had an integrated error of 2472, over double that of version 13, owing to the instability after 1800s, and only produced 0.725 kmol methanol.

This makes it clearer to me as to why our original controller failed. It was far too overtuned. Clearly this is a case of K.I.S.S. : Keep it simple stupid.

09/06/2020, 10/06/2020

Reading up on MPC, the lecture notes don’t cover it in detail unlike some other avenues I’m yet to try. I will thus put a pin in this until I’ve exhausted other approaches that require less research (as this is what I would have done had I’ve been doing this under a deadline)

The two other avenues I wish to explore at present are feedforward control and Ziegler-nichols tuning. The former will require tedious analysis of the disturbance step responses and may decrease stability, so I will focus on the latter today.

Smith predictors were of use previously so I will begin with a system that utilises 2 P-only controllers and 2 smith predictors, with the CT predictor accounting for both g22 and g21.

Initially the controllers gc1 and gc2 were set to 1 and -1 respectively (since we know the gain of gc2 must be negative). Of course, gc1 had to approach ~100 before the system began to reach the set point, as this is what the cohen-coon tuning gave (roughly).

I then proceeded to increase the gain of gc1, however there weren’t clear sustained oscillations even with a gain of 10000. (NB this is without any disturbances, which I believe is how ZN tuning is meant to be done?). I tried again whilst bypassing the smith predictors. This then showed the oscilliatory behaviour I expected.

|  |  |  |  |
| --- | --- | --- | --- |
| Gc1 gain | Gc2 gain | Response MF | Response CT |
| 100 | -1 | stable | Stable |
| 200 | -1 | stable | Stable |
| 300 | -1 | Unstable\* | Unstable\* |
| 250 | -1 | Unstable\* | Unstable\* |
| 225 | -1 | Unstable | Unstable |
| 220 | -1 | Stable | Stable |
| 222 | -1 | Stable | Stable |
| 223 | -1 | Stable | Stable |
| **224** | **-1** | **Critical** | **Critical** |
| 1 | -1 | Stable | Stable |
| 1 | -2 | Stable | Stable |
| 1 | -10 | Stable? | Stable? |
| 1 | -100 | Stable? | Stable? |
| 1 | -1000 | Stable?? | Stable?? |

Note that these instabilities weren’t the standard oscillations of increasing amplitude that I expected, but periodic spikes every 30 or so seconds. I called this unstable since it is obviously not what I want my controller to do. This is probably due to the fact that the flowrates could not go below 0, thus truncating the response once it oscillated beyond a certain amplitude.

I found that Ku1 = 224, Pu1 = 16, Ku2 = ??, Pu2 = ??

One frustrating thing was the time taken to simulate the controller when tuning gc2, not sure why it was so slow just to simulate. It may be because the inlet flowrates are negative, and this may be having an effect on the logic within the reactor block. The fact that the flowrate was always negative meant there was vey little difference in the output when varying the gain on gc2.

The difficulty here is that I’m using ZN to tune a MIMO system, and I’m not sure what the other controller should be set to whilst testing. The obvious choice is 1, since a pure gain of 1 does not effect the system. However this is obviously not working for gc2, which I believe is down to the interaction of the inputs.

The solution I will try next is using ZN tuning parameters for gc1 instead of setting it to 1, and then seeing if I can determine the ultimate gain and period of gc2. The ultimate gain of gc1 was 224, so a gain of 112 was used for gc1 and gc2 tested again.

|  |  |  |  |
| --- | --- | --- | --- |
| Gc1 Gain | Gc2 Gain | Response MF | Response CT |
| 112 | -1 | Stable | Stable |
| 112 | -10 | Stable | Critical? |
| 112 | -5 | Stable | Unstable |
| 112 | -3 | Stable | Unstable |
| 112 | -2 | Stable | Stable |
| 112 | -2.5 | Stable | Unstable |
| 112 | -2.2 | Stable | Unstable/Critical? |

Note that for similar reasons I was apprehensive about -10 being the critical value given I’d gone by such a large jump. In reality one is meant to gradually increase the gain (or decrease since it’s negative) of gc2 until the response is critical. My going in steps of 10 is to quickly determine roughly the order of magnitude of the ultimate gain.

The response with the very negative gain of gc2 is periodic with constant amplitude, but not a clean oscillation (i.e. it steps up and down and plateau’s in between). I’m looking for the gain where there are critical, continuous oscillations.

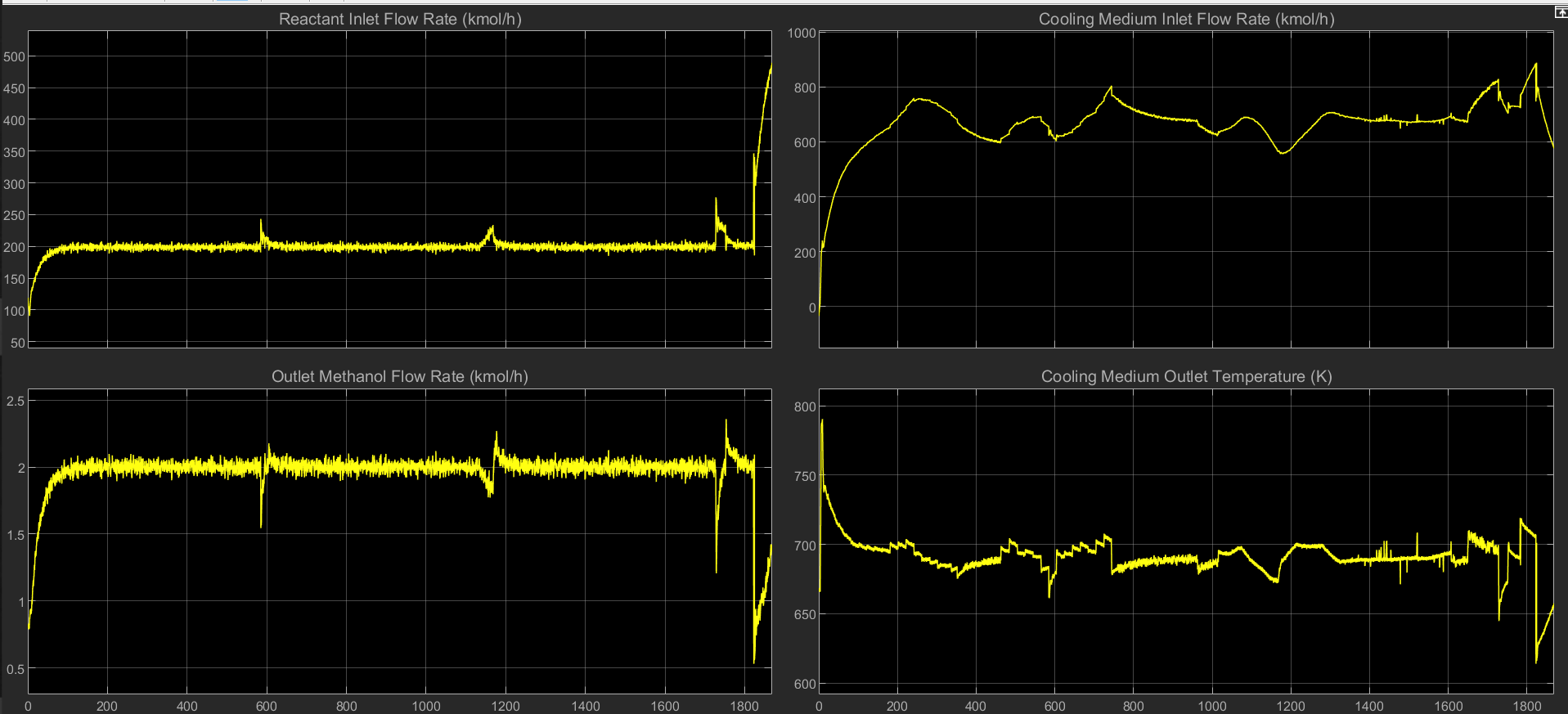
Also note that of course the MF response would be stable; it is unaffected by gc2 and gc1 has been tuned.

I decided I was still getting difficult results, so tuned gc1 fully as a ZN-tuned PI controller and enabled the MF smith predictor. My thinking was that in the final controller this is the level of control I’d have on g21, and thus I can focus primarily on g22 (which is what gc2 mainly regulates).

|  |  |
| --- | --- |
| Gc2 Gain | Response CT |
| -1 | Stable |
| -2 | Stable |
| -3 | Stable |
| -4 | Critical/Unstable |
| -3.9 | Critical/Unstable |
| -3.5 | Critical/Unstable |
| -3.2 | Stable |
| -3.3 | Critical/just a bit stable |
| -3.34 | ~ critical |

The critical gain of gc2 is around -3.34 and the time period of ~9.6s

With the ultimate gain and period of the controllers, a ZN tuned system with 2 PI controllers can be produced (Version 15). These parameters were run with the test, with a methanol flowrate set point of 2 kmol/h (such that the flowrate reached 1 kmol/h in 10s).



The integrated absolute deviation was 3901, and the system produced ~1 kmol methanol. The production is only good because a high set point is required to reach 1 kmol/h in 10s; the deviation beyond the temperature range makes this the worst performing controller tested thus far (the best remains version 13).

By considering version 12, 13 and 15, there is evidently an ‘optimal’ tuning arrangement of the two PI controllers with smith predictors that lies between v12 and v15. The next thing to consider would be feedforward control.

For fun, I figured I’d try find the ‘optimal’ parameters by trial-and-error. In reality I wouldn’t have been able to do this for the coursework (as the test came out after the controllers were submit), however I’m curious as to where the optimal parameters lie and how close cohen-coon is.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Iteration | Kc1 | Tau I 1 | Kc2 | Tau I 2 | Integrated er |
| 0 (v12) | 200 | 3 | -7 | 3 | 2472 |
| 0 (v13) | 133.2 | 6.087 | 1.92 | 5.95 | 1929 |
| 0 (v15) | 100.8 | 13.33 | -1.50 | 8.00 | 3901 |
| 1 | 117 | 9.71 | -3.36 | 5.91 | 2125 |
| 2 | 125.1 | 7.9 | -4.29 | 4.87 | 1923 |
| 3 | 129.15 | 6.99 | -4.76 | 4.35 | 1897 |

Note that the tuning will be slightly off since the MF set point is 2 kmol/h

It turns out that the cohen-coon tuning was pretty close to the best tuning parameters.