Virtual Lab Experience III
Ilana Zane
November 28, 2022

A1) Human Motor Control System

Figure 1 verifies that the human motor control system can track the desired trajectory. Figure 2 shows a zoomed in portion of Figure 1.

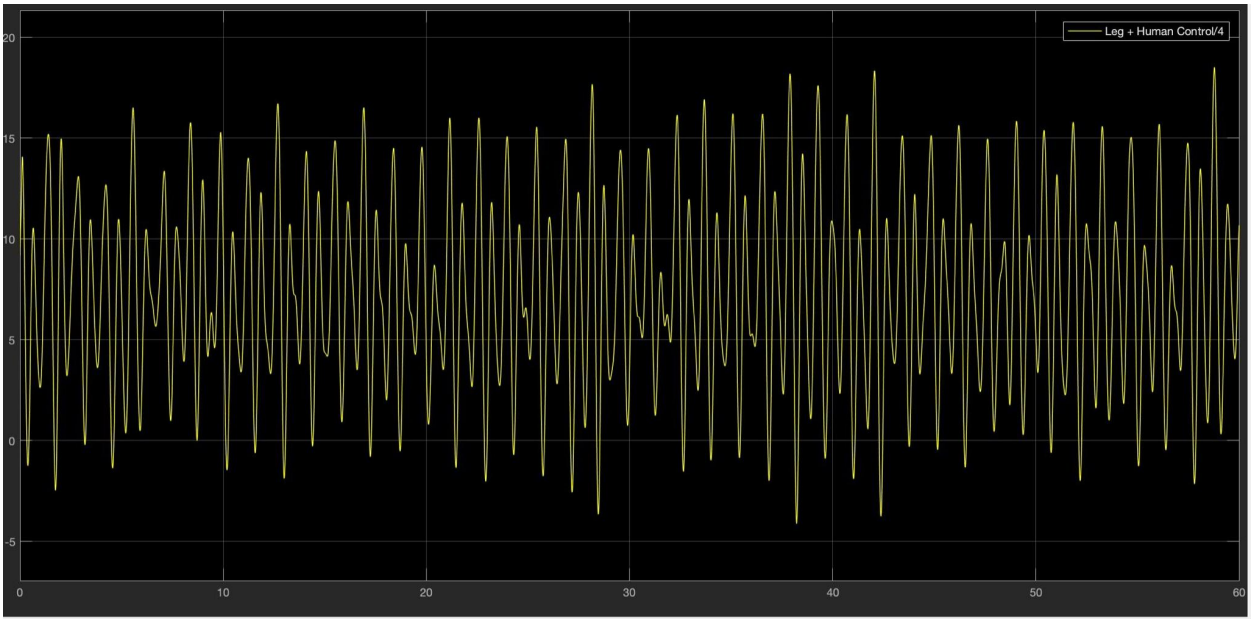

Figure 1: τ_H vs time where the yellow line is τ_H

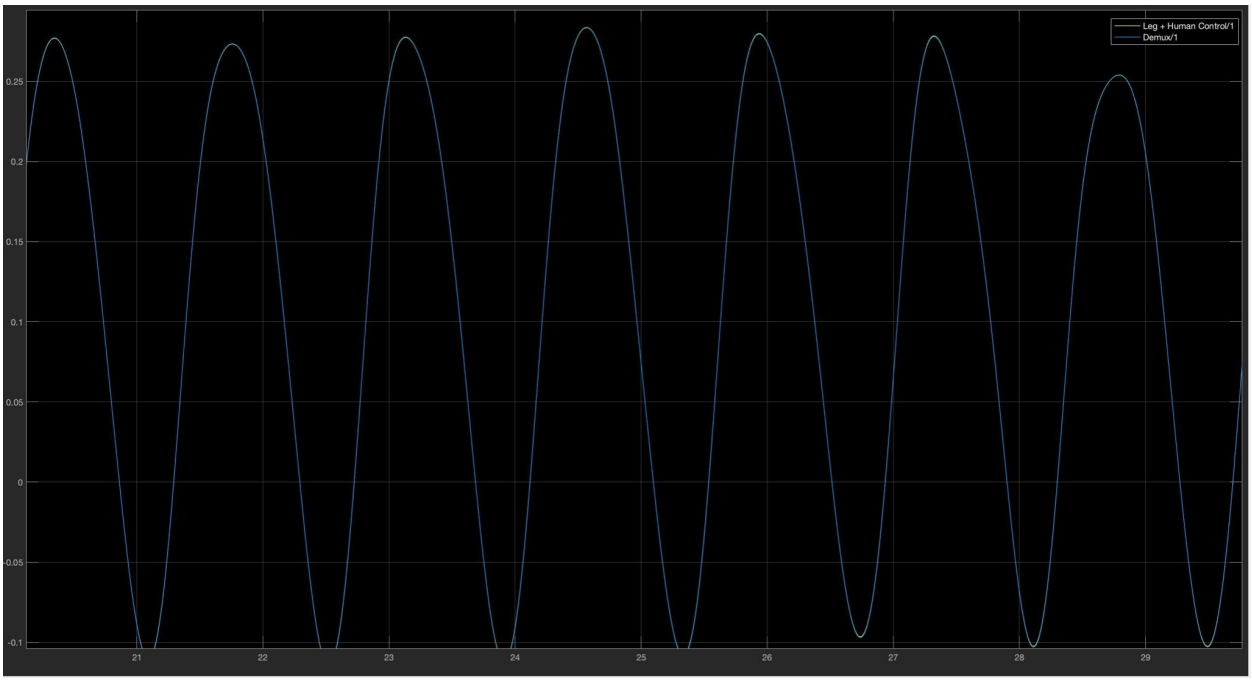

Figure 2: $\theta_{L,des}$, θ_L vs Time where yellow is θ_L and blue is $\theta_{L,des}$

A2) Inverse Dynamics

For Part 2, I created a function block that took in the parameters: *IHC*, *IRC*, *mH*, *LH*, *mR*, *LR*, *g* and output the total torque, τ_{TOT}. The parameters θ_{des} and $\dot{\theta}_{des}$ came from the provided human motor control block.

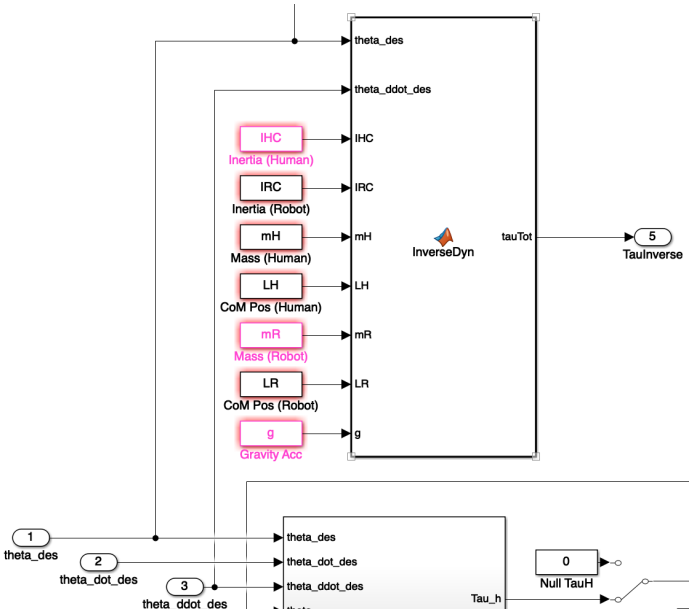

Figure 3: Simulink model for inverse dynamics

In order to solve for the total torque, I used the following formula:

$$\dot{\theta} = 1/ I_{TOT0} [\tau_H - mH * g * LH * sin(\theta) + \tau_R - mR * g * LR * sin(\theta).$$

Rearranging the equation and solving for $\tau_H + \tau_R$, which is equal to the total torque resulted in the equation as seen in Figure 4.

ME655_Lect09_VL3_Simulink ▸ Leg + Human Control ▸ MATLAB Function

```
1  function tauTot = InverseDyn(theta_des,theta_ddot_des,IHC,IRC,mH,LH,mR,LR,g)
2
3  tauTot = (IHC + IRC + mH*LH^2 +mR*LR^2)*theta_ddot_des + (mH*LH+mR*LR)*g*sin(theta_des);
4
5  end
```

Figure 4: Inverse Dynamic Equation used in Simulink Block

Given the assumption that $\tau_R = 0$. Then our total torque from the inverse dynamic equation should be equal to τ_H, the human torque without assistance, because $\tau_{TOT} = \tau_H + \tau_R$. In Figure 5 we see that the blue line is the unassisted human torque and the yellow line is the torque from the inverse dynamic model. The output of the total torque matches the human torque very closely – we can infer from this that the inverse dynamic model works well.

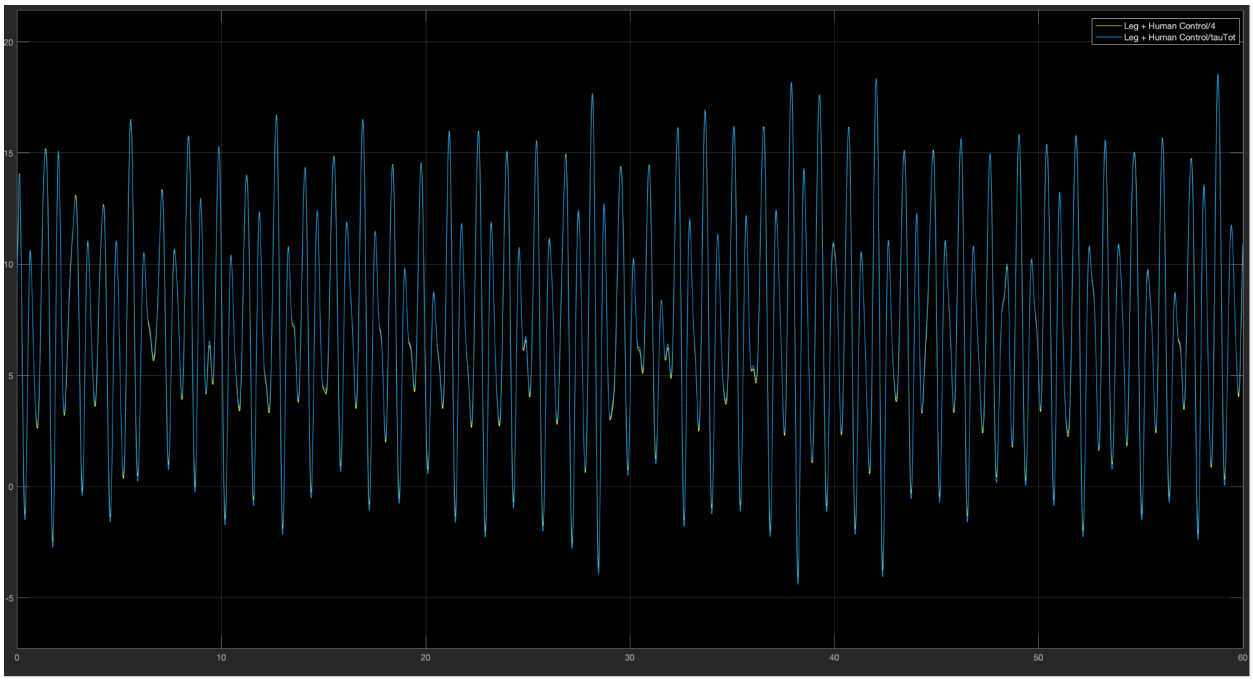

Figure 5: Yellow is $\tau_{INVERSE\text{-}DYNAMIC}$ and blue is $\tau_{NO\text{-}ASSIST}$

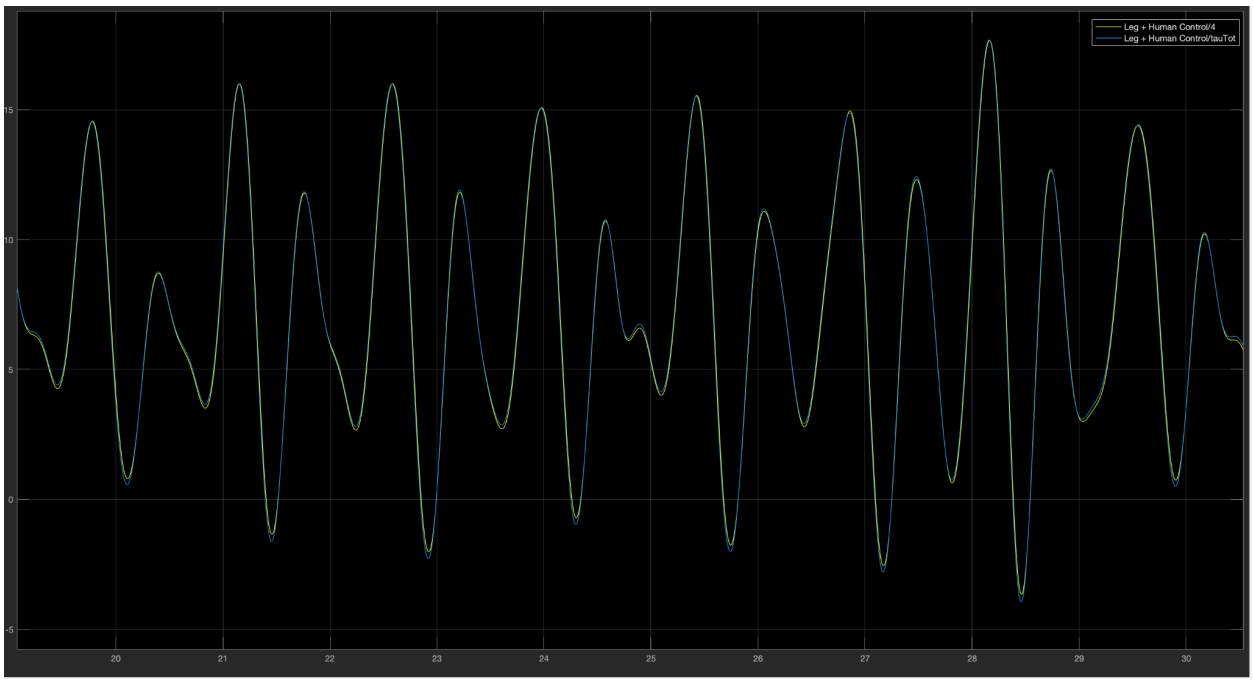

Figure 6: Zoomed in section. In this plot, the yellow represents $_{NO\text{-}ASSIST}$ and blue is $\tau_{INVERSE\text{-}DYNAMIC}$

A3) Trajectory Estimation using AFO's

In Part 3, I created a pool of AFO's to compute the estimates of $\hat{\theta}_{L,AFO}$, $\hat{\theta}_{L,AFO}$ and $\hat{\theta}_{L,AFO}$. The input parameter θ_{des}, came from the human motor control block and given values were *M*, ε, and *v*. The change in time (Δt) was calculated from the provided data to be 0.003.

First, initial conditions are set. If the AFO is not activated (0 to 5 seconds) then the oscillator state is not updated (i.e. φ, ω, a, and θ are all equal to 0). If the AFO is activated after 5 seconds and the parameters φ, ω and θ are still 0 then we assign initial conditions as $\varphi = \pi/2$, $\omega = 3\pi/2$, and $\theta = 0$. From this point on the model computes the error signal, updates the oscillator states, which in turn updates the estimates of $\hat{\theta}$, $\hat{\theta}$, and $\hat{\theta}$.
Figure 7 shows the AFO model in Simulink. Since I couldn't figure out how to recursively feed vectors in matlab (I figured it out later) and there were only six adaptive oscillators, I hard coded the update states for φ, ω, a, and θ.

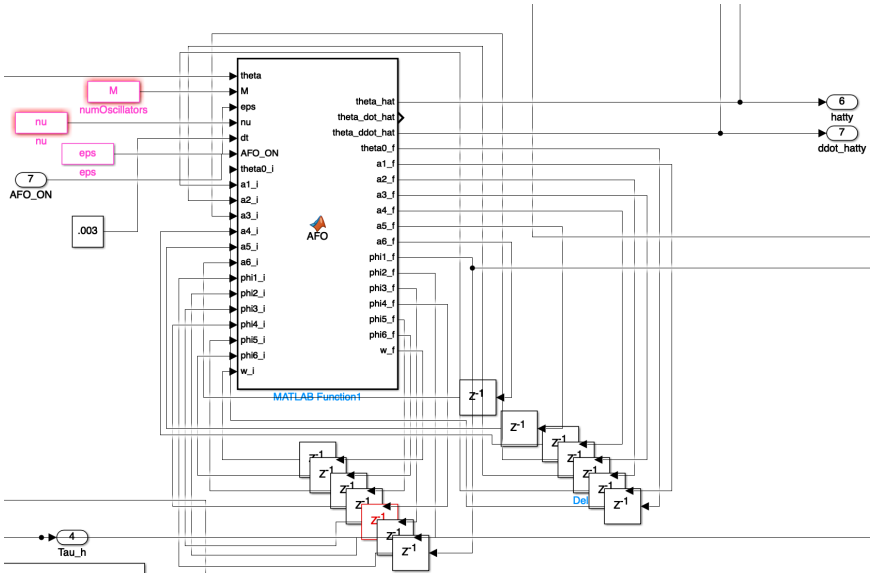

Figure 7: AFO model

Figures 8 and 9 show θ_L and $\hat{\theta}_{AFO}$ vs time. We can see from Figure 8 that before 5 seconds, $\hat{\theta}$ is 0 and as time increases $\hat{\theta}$ begins to converge with θ_L which means that the AFO model is working.

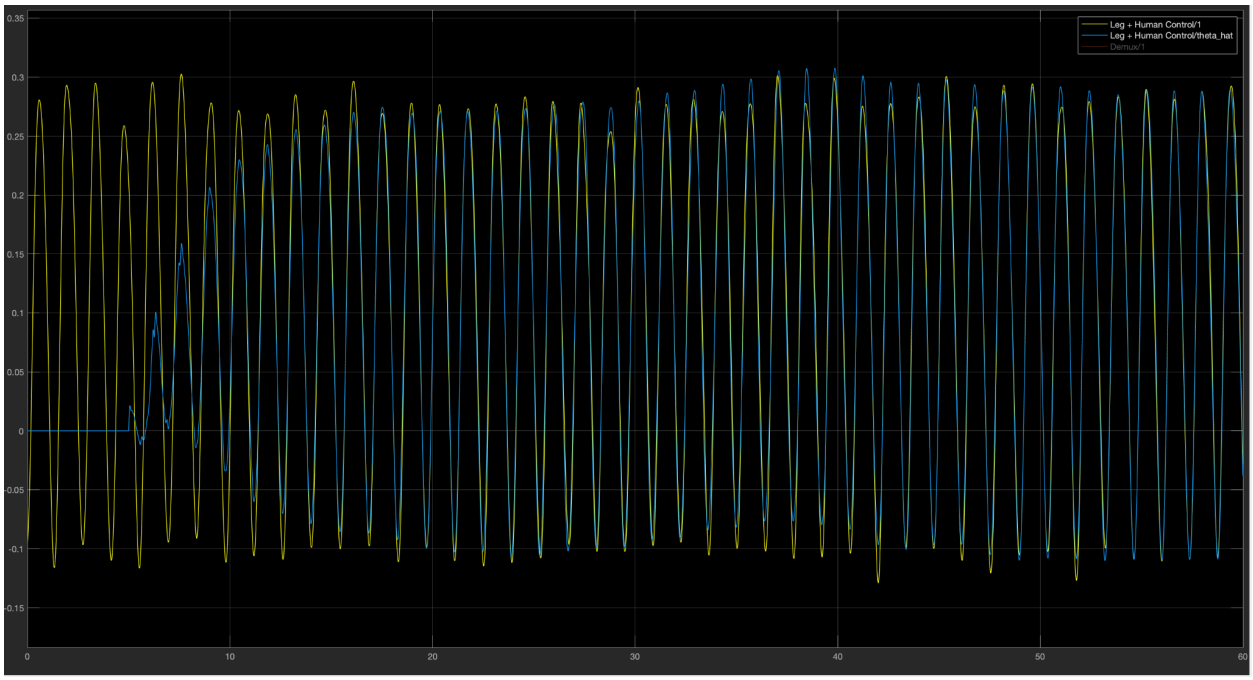

Figure 8: θ_L and $\widehat{\theta}_{L,AFO}$ vs time. The yellow is θ_L and the Blue is $\widehat{\theta}_{L,AFO}$

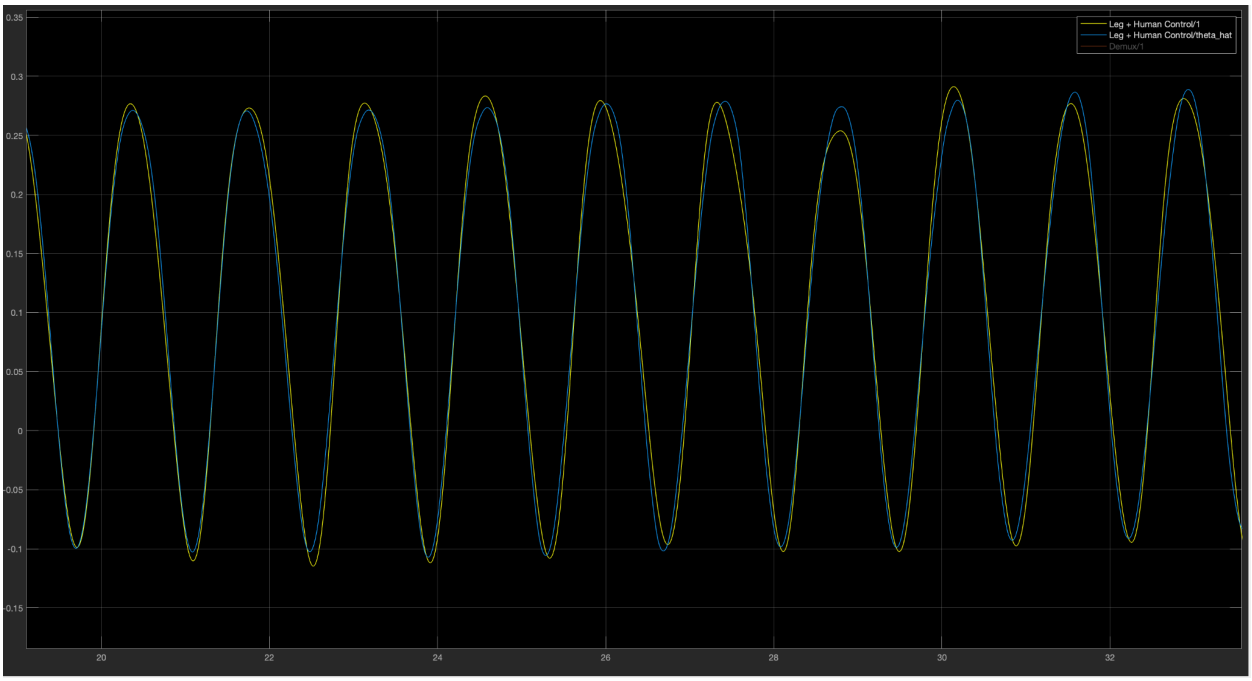

Figure 9: Zoomed in plot. The yellow and blue lines correspond to Figure 8.

A4) Model Based Assistive Controller

For Part 4, in order to make the model-based assistive controller, I created a separate Inverse Dynamics simulink block that was identical to the one created in Part 2. The only difference now is that instead of using θ_{des} and $\dot{\theta}_{des}$ from the provided human motor control model, these values were replaced with $\hat{\theta}$ and $\hat{\dot{\theta}}$ from the previously created AFO model. Also, to ensure that the controller is activated at $t_{START,CTRL} = t_{START,AFO} + 10s$ the parameter *Robot_Ctrl_ON* is fed in from the provided simulink time model. Keeping the other input values the same as before, the model outputs an assistive torque (τ_R) labeled as Assist in the model (Figure 10)

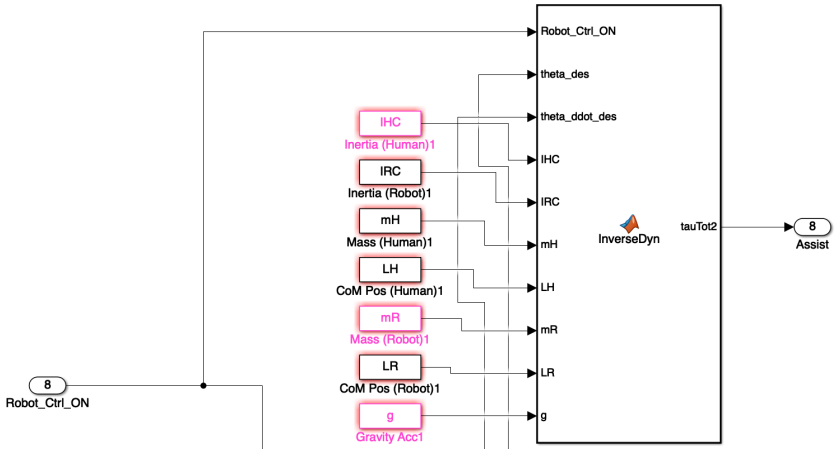

Figure 10: Model Based Assistive Controller

Once the assistive torque was calculated, I created another *Leg+Human Control* model that is identical to the one provided. The assistive torque from the original model is multiplied by 50% and fed into the new *Leg+Human Control Model* as τ_R. The original τ_H from the first model is plotted alongside the new assistive torque. This process is shown in Figure 11 along the blue highlighted paths.

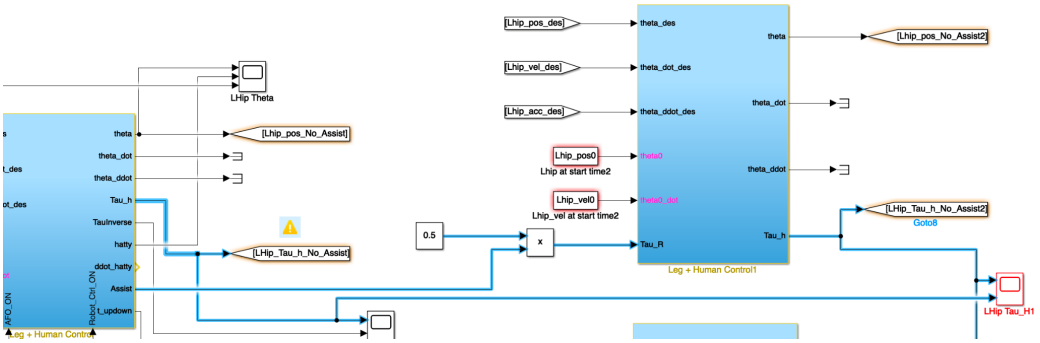

Figure 11: original τ_H is plotted alongside 50% of the assistive torque.

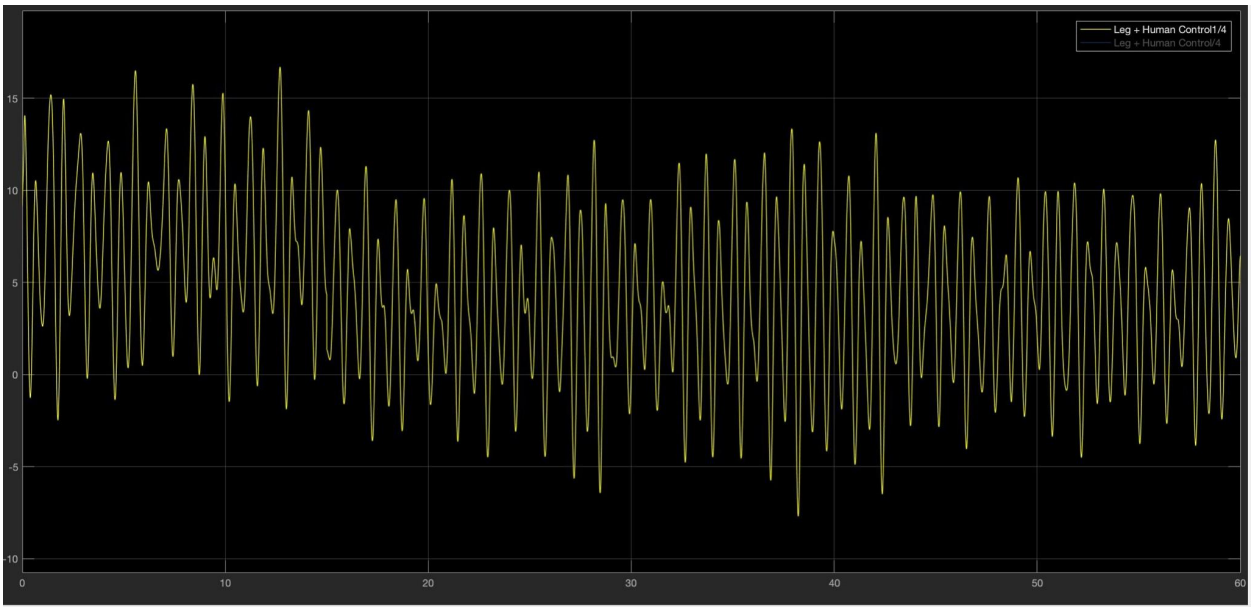

Figure 12: τ_H assistive vs time

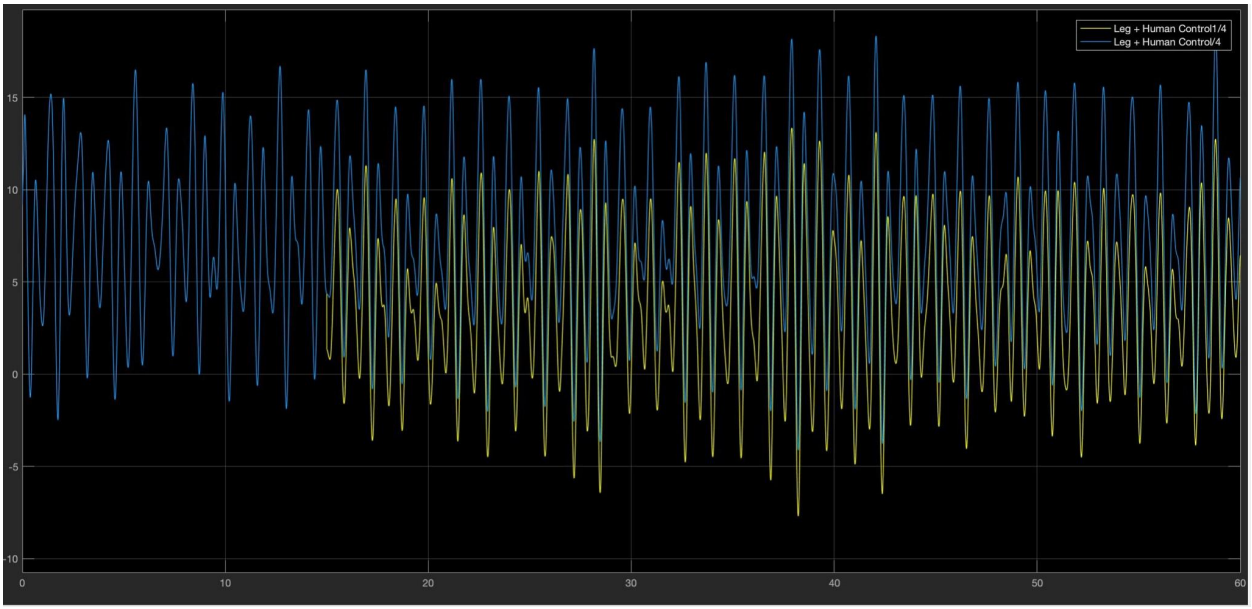

Figure 13: $\tau_{H,NO\text{-}ASSIST}$ vs $\tau_{H,ASSIST}$. The blue line is $\tau_{H,NO\text{-}ASSIST}$ and the yellow line is $\tau_{H,ASSIST}$

In part 1 we can see that the human torque is generally constant and follows a trend. In Figure 12 we see that for the first 10 seconds the plots are the same, but after the 10 second mark, where the original plot still has that general trend, Figure 12 shows that the range of the torque shifts downwards. This shift to a lower torque range can be attributed to the assistive torque becoming present after the 10 second mark.

With the assistive torque present, the human torque can decrease and allow the assistive torque to help power the leg through the gait cycle as seen in Figure 13.

A5) Human Torque Estimation using Nonlinear Filter

In Part 5, I created a matlab function that implements a kernel-based nonlinear filter to obtain $\check{\tau}_H$. In order to do so, the function created takes in parameters *e*, *P*, *w*, φ_1 from the AFO model, τ_H from the human motor control block, then non linear filter parameters N,H, and lambda are taken as input from the provided matlab code, and parameter *AFO_ON*. The output is $\check{\tau}$. If the AFO is not on (before the five second mark) then *P* is initialized to be an array of ones of size *N*, *w* is initialized to be an array of zeros of size *N*, and $\check{\tau}$ is 0. Once the AFO is activated, the error signal and filter states, *P* and *w*, are recursively updated. To calculate $\check{\tau}$, the dot product is taken between Υ and the vector *w*, divided by the sum of the summation of Υ. The function is shown in Figure 14.

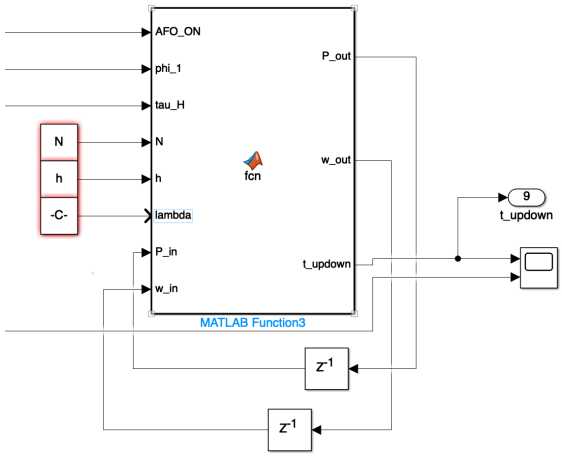

Figure 14: Nonlinear Filter Function with output $\check{\tau}_H$

Figures 15 and 16 show the convergence of τ_H and $\check{\tau}_H$. We can see from Figure 15, that after the AFO is activated (at 5 seconds) a plot of $\check{\tau}_H$ forms and converges with the original τ_H. Since $\check{\tau}_H$ converges with τ_H, this means that $\check{\tau}_H$ is an accurate, real-time estimate of the wearer's torque τ_H as a function of the gait phase, φ, obtained from the previous AFO.

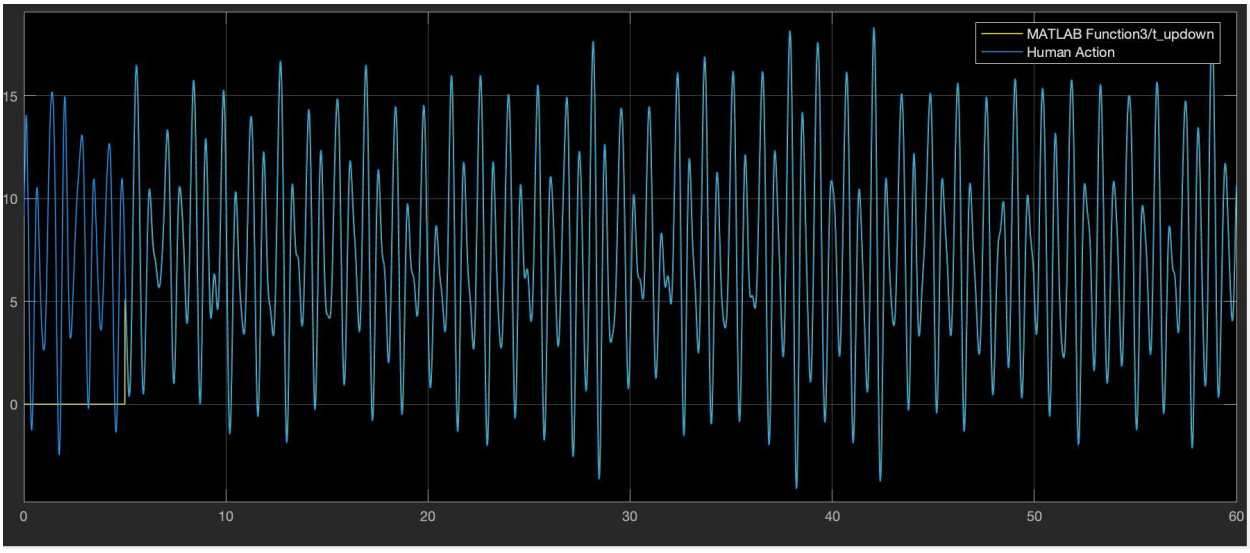

Figure 15: The yellow line is $\check{\tau}_H$ and the blue line is τ_H

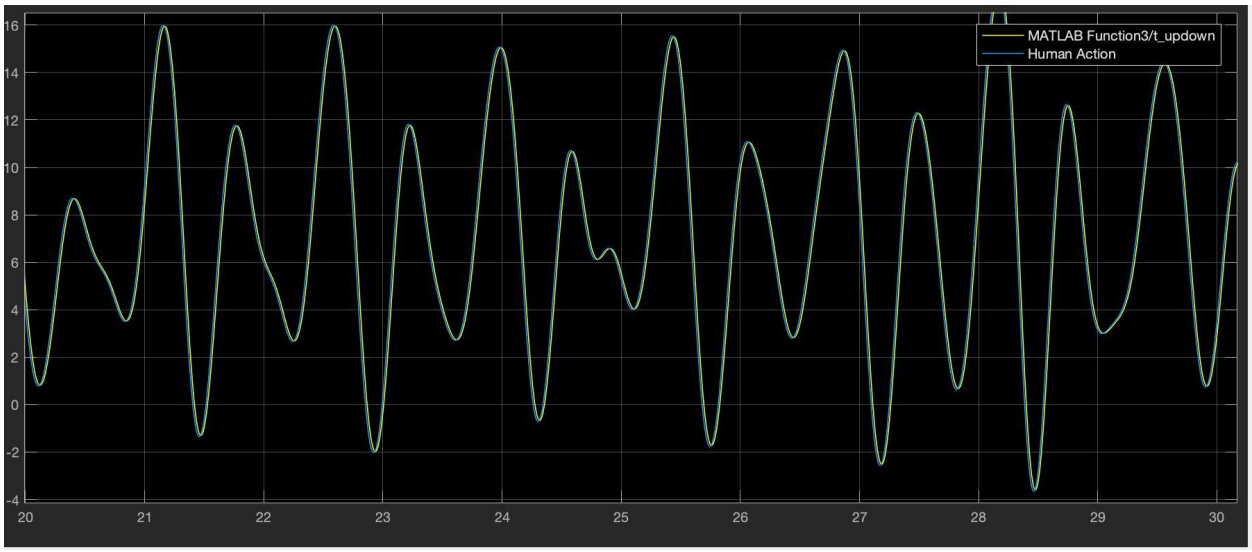

Figure 16: Zoomed in plot.

A6) Model-Free Assistive Controller

For Part 6, a third *Human + Leg Control* block was created that is identical to the original, provided block. However, the input for τ_R becomes 50% of $\check{\tau}_H$, the output torque from the nonlinear filter. This reduced torque is fed into the control block and is plotted alongside the original $\check{\tau}_H$ vs time. Figure 19 shows the torque from the AFO vs the torque from the NLF vs time. Figure 17 shows the model where the blue, highlighted paths indicate where the plotted torques are coming from within the model.

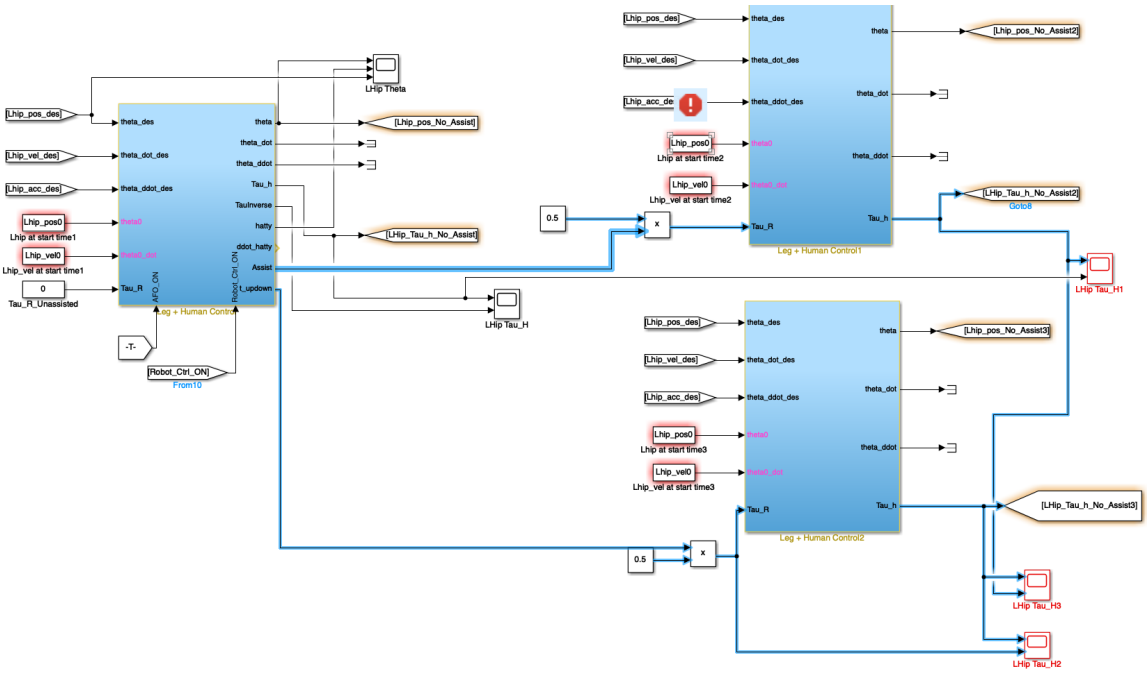

Figure 17: Simulink model showing the different plotted torques and where they originate from within the model.

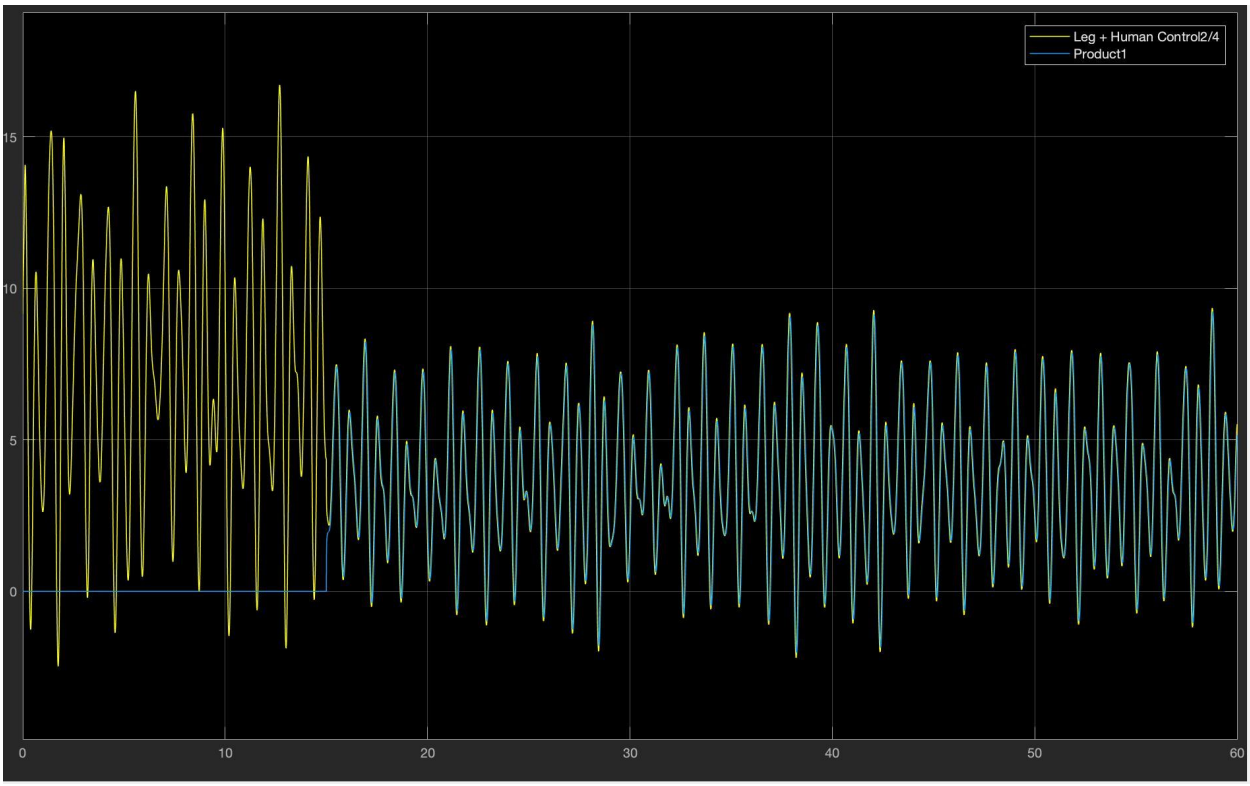

Figure 18: Plot of τ_H and $\check{\tau}_H$ vs time. τ_H is the yellow line and the blue line is $\check{\tau}_H$.

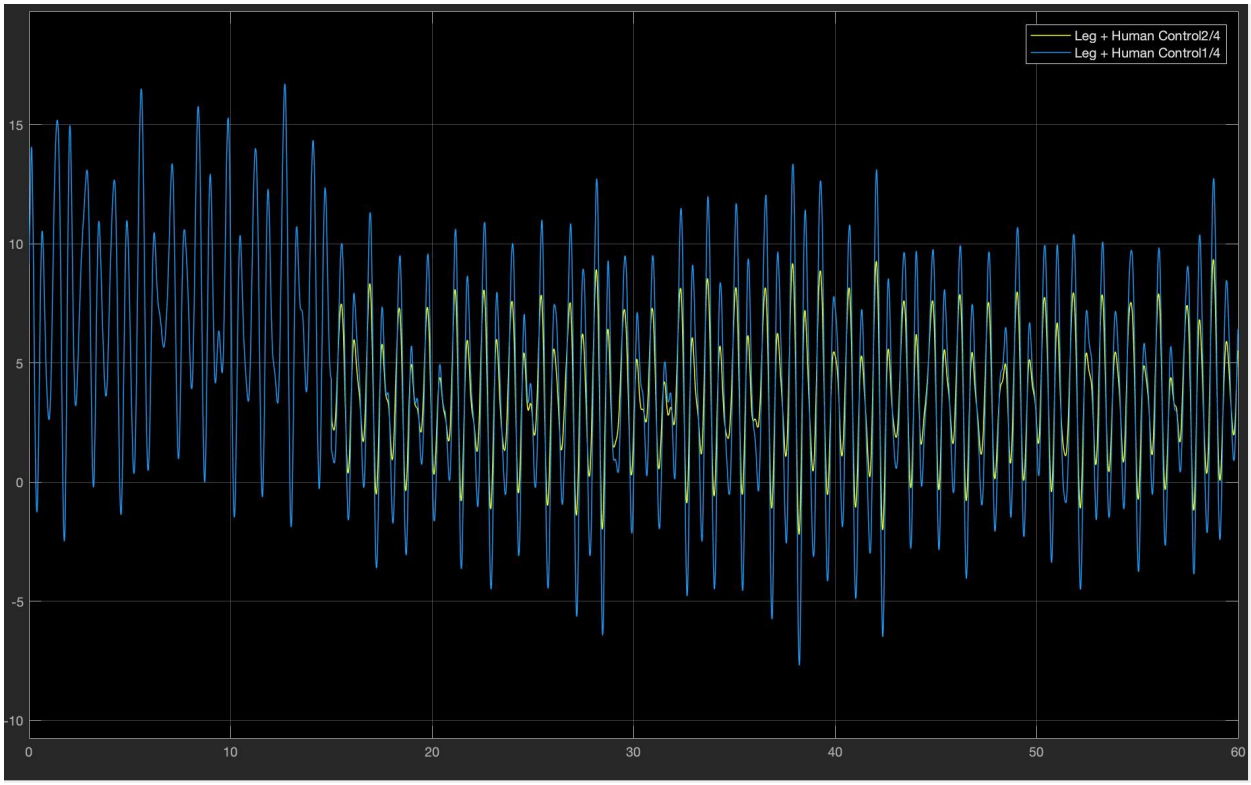

Figure 19: Model-based (AFO+Inverse dynamics) τ_H vs time and model-free (NLF) τ_H vs time. The blue line represents model based and the yellow is model-free

Figure 19 shows the output torque for the AFO model in blue and the output torque for the NLF in yellow. For the first 15 seconds before any assistance is provided the torque is the same. However, after the separate controls are activated, the results show that the models both reduce output human torque, but the NLF produces a smaller range of torques while the AFO produces a greater range. This means that the assistive torque with the AFO can reduce the human torque by most but can also cause the unassisted torque to be greater in comparison to the NLF in some intervals. The NLF model neither significantly increases or decreases the human torque in comparison to the AFO– the NLF model is more stable than the AFO.

Problem 7 is omitted.