Modelica Assignment

By Arno Deceuninck (s0181217) and Dogukan Altay (s0211552)

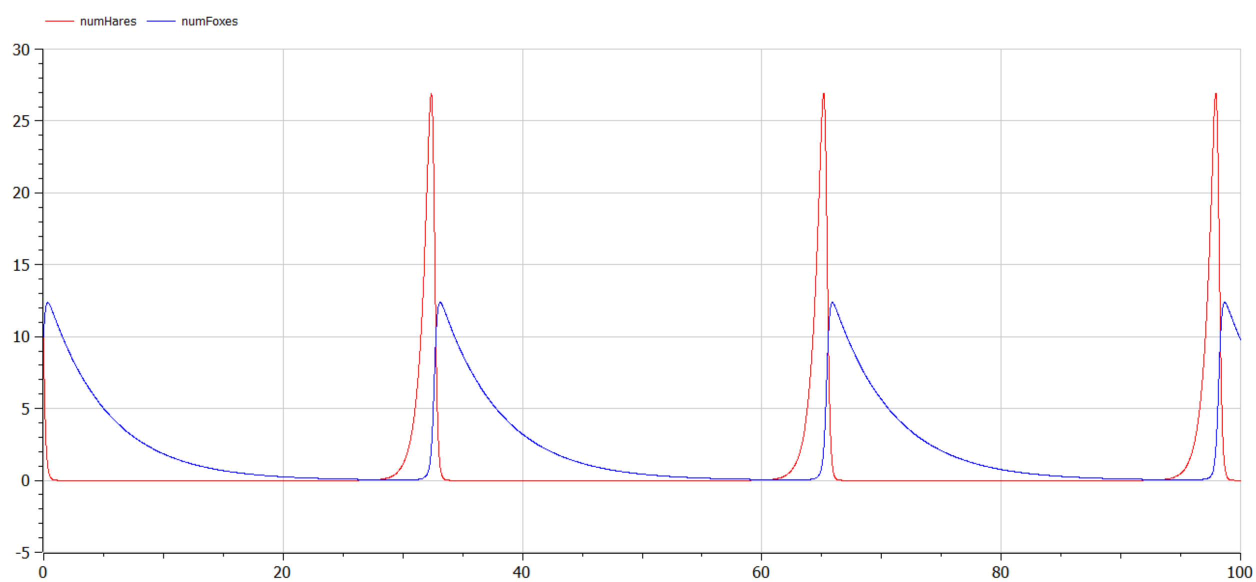
Introductory problem

1. Predator-Prey model

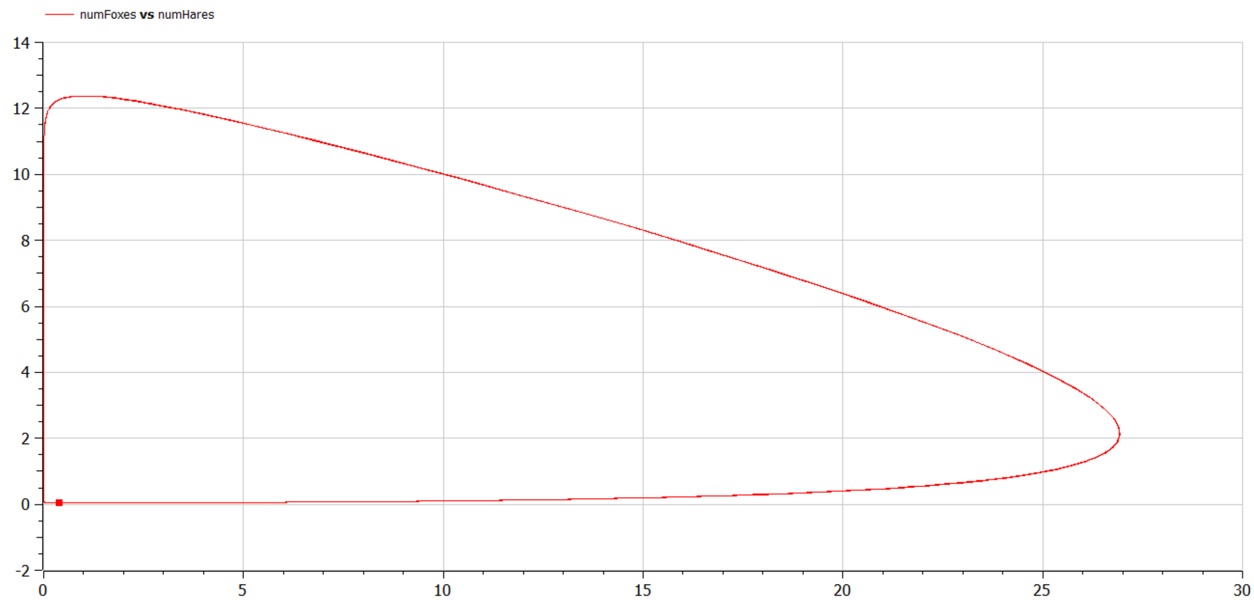
See pray_hares_1.mo.

2. Simulate 100s

2.1 plot of the population dynamics w.r.t. the time



2.2 phase-space plot (predator count vs prey count)



2.3 Does this system stabilize?

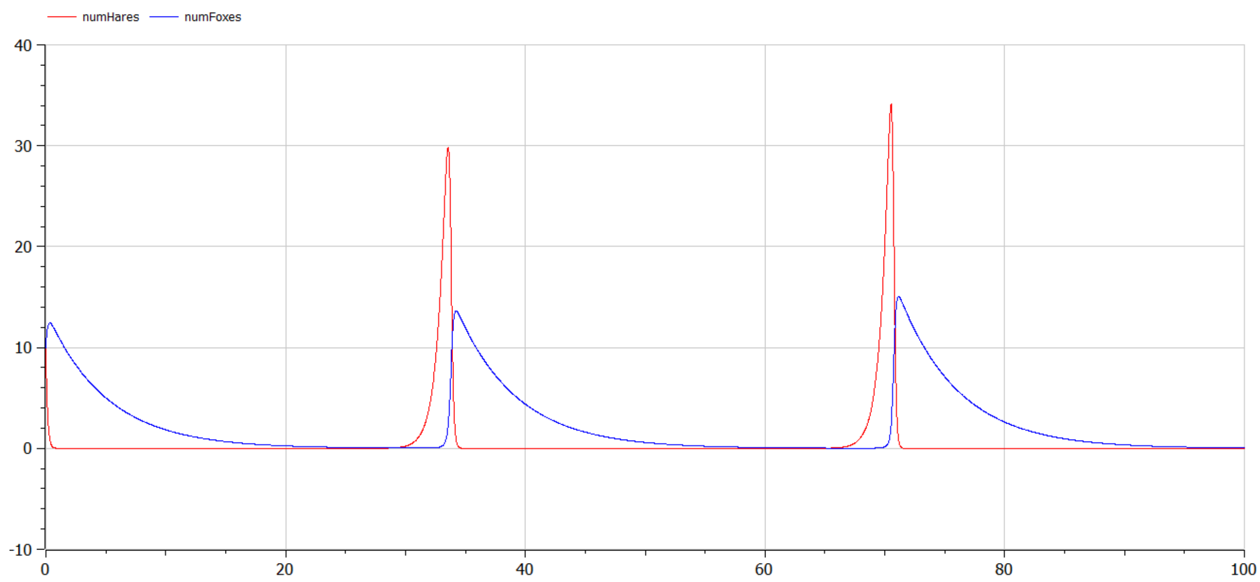
The system will keep the same repeating cycle, because the phase-space plot can be bounded by a rectangle. The lines do not converge though.

2.4 Changing “Integration” parameters

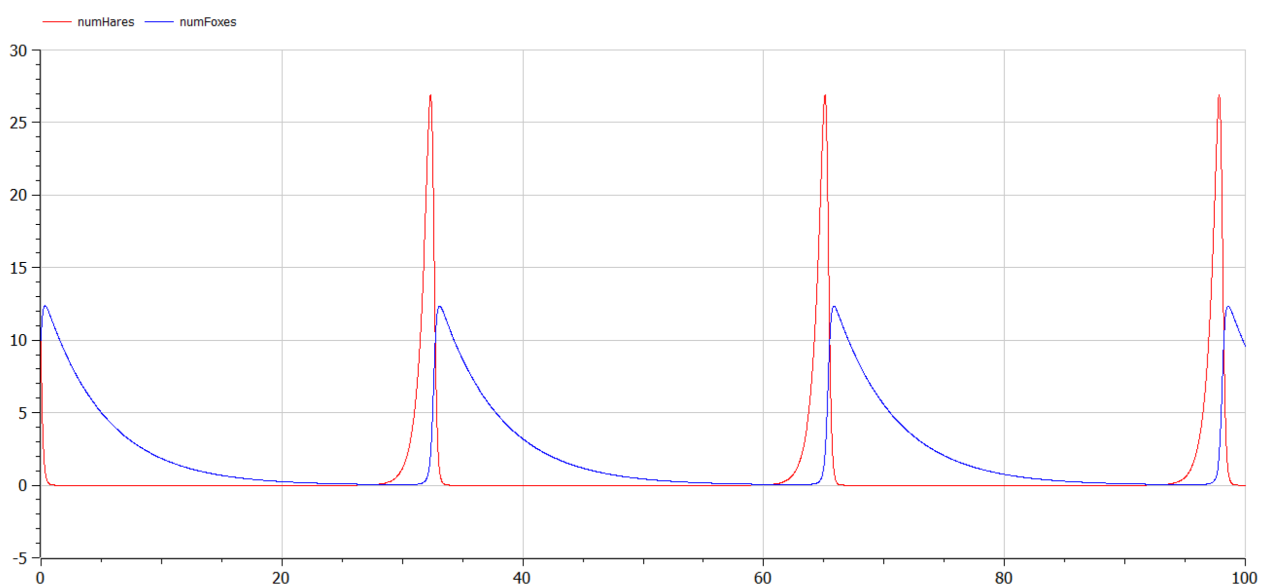
2.4.1 Changing integration method

The length of the repeating cycle differs. Also the number of preys/preditors (ex. The max of each) differs. Changing the tolerance and jacobian doesn’t give a noticeable difference.

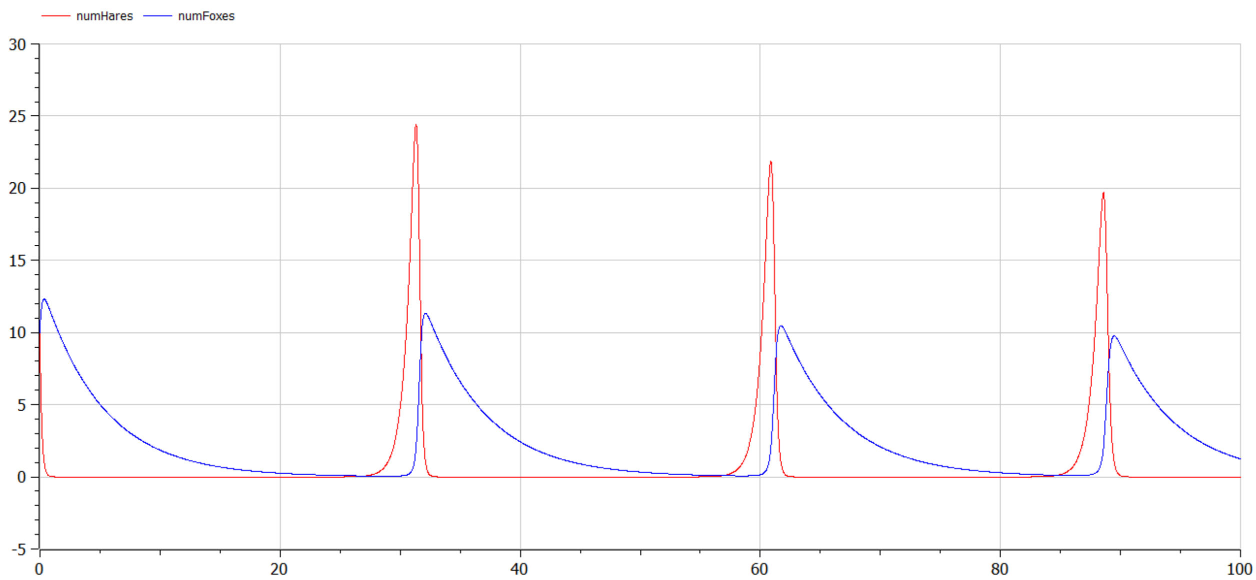
2.4.1.1 Euler



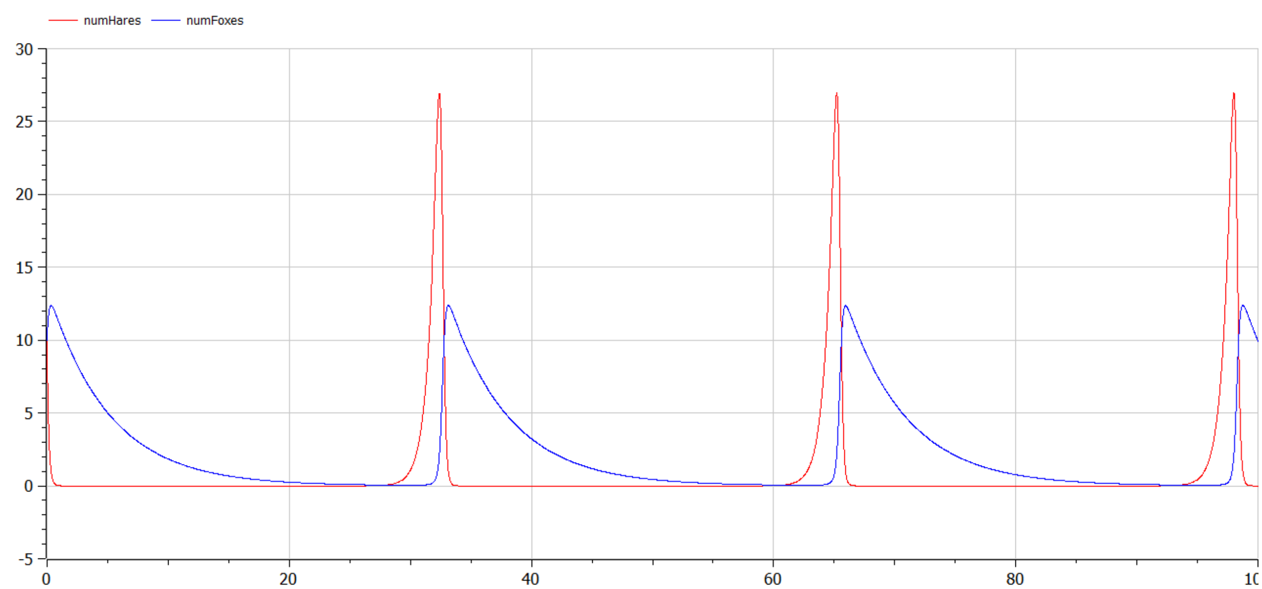
2.4.1.2 Heun



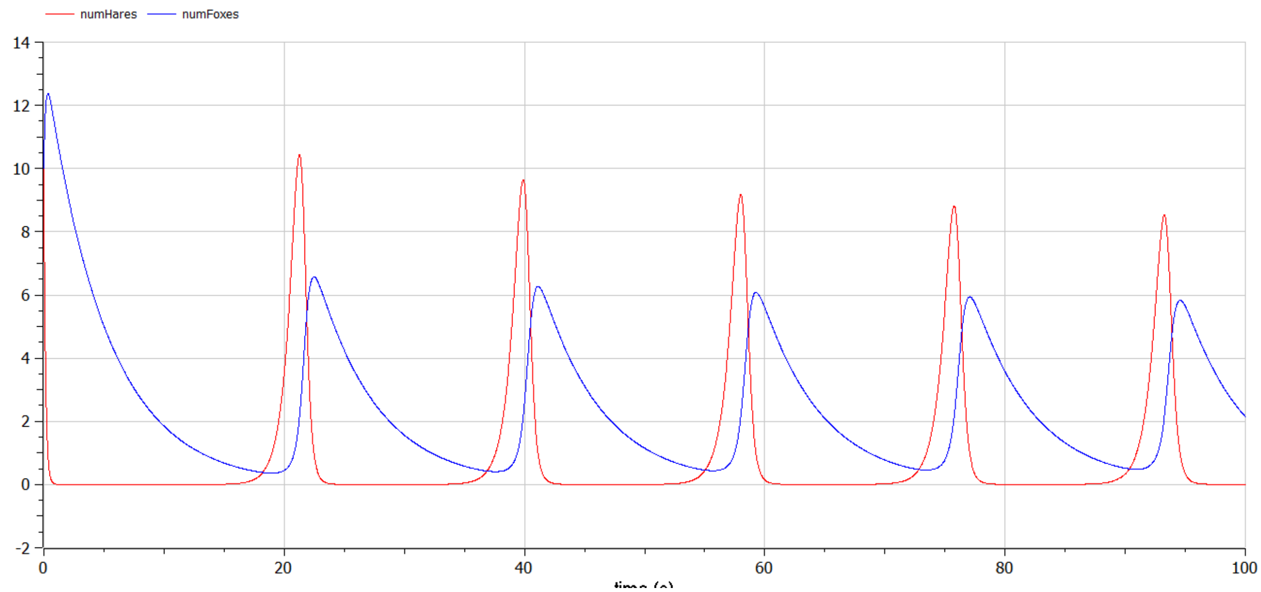
2.4.1.3 Impeuler



2.4.1.4 Trapezoid



2.4.1.5 Irksco

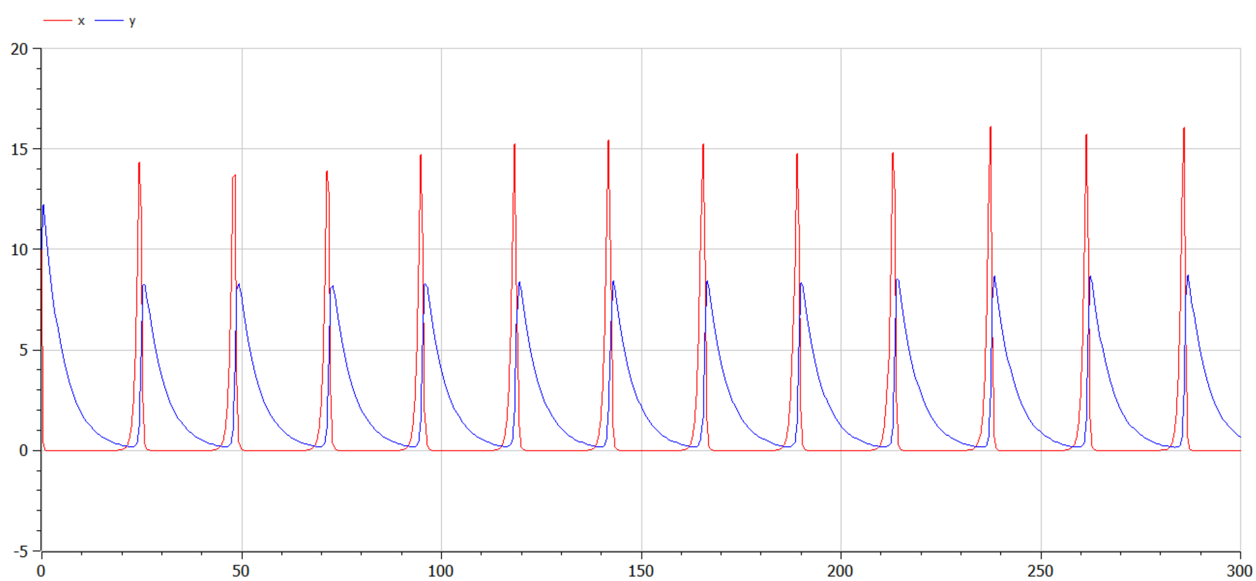


3. Parameters and output

See pray_hares_3_with_parameters.mo.

4. Diagram view

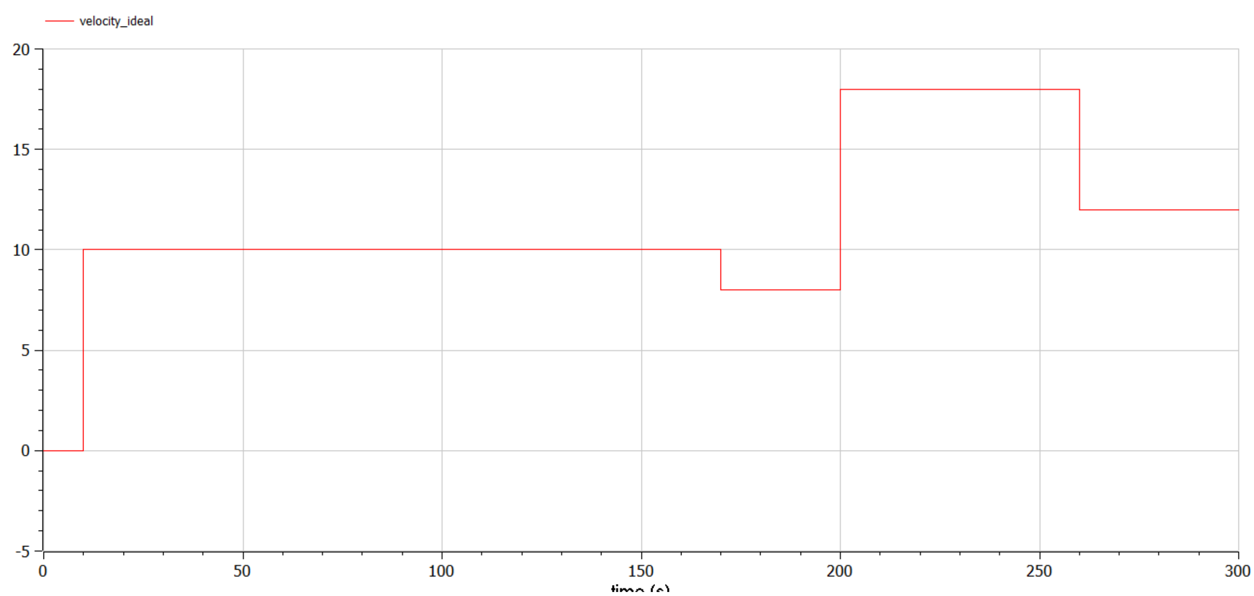
See pray_hares_4_diagram.mo. The shape of the graph is similar, but the period (number of peaks) is a lot higher than the implementations above.



PRT Problem

1. Look-up model

This is the associated graph for a simulation of 300 seconds with an interval size of 0.5:



2. Plant

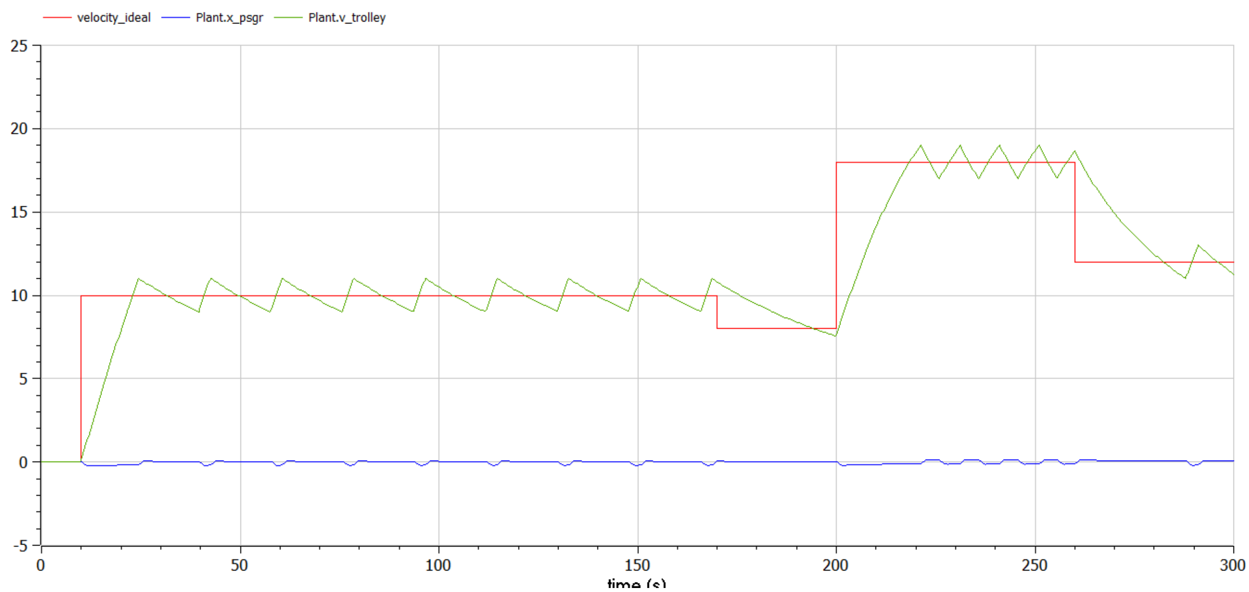
The model can be found in ptr_plant.mo.

3. Bang-Bang controller

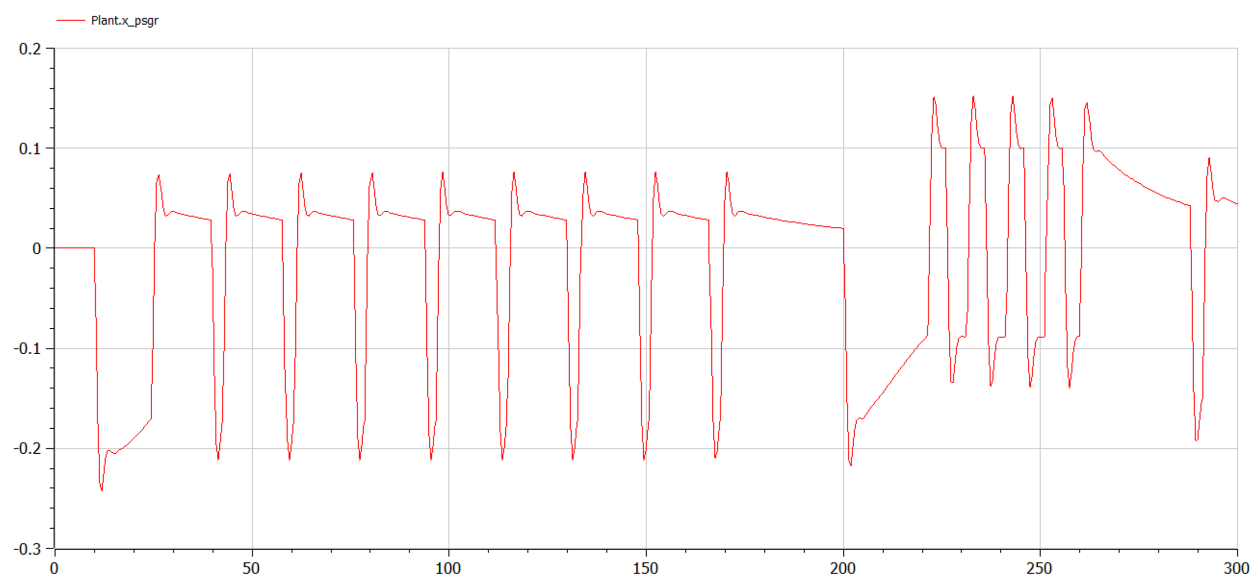
This can be found in bang_bang_controller.mo.

4. Simulation

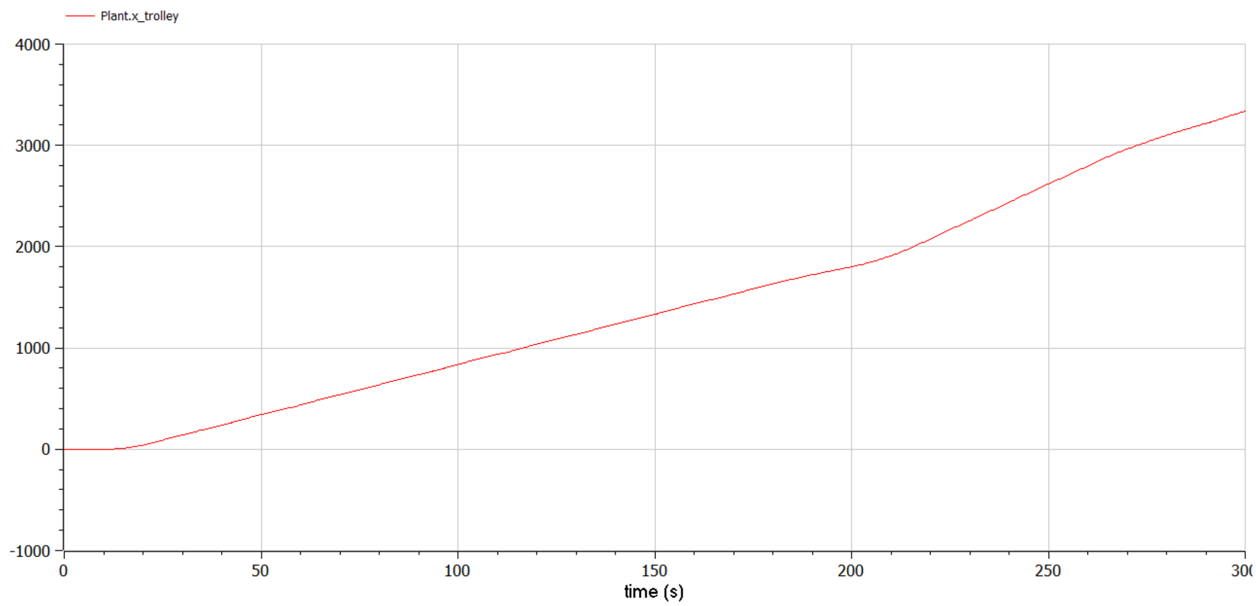
This is a plot of some variables of a simulation of 300s with an interval of 0.5:



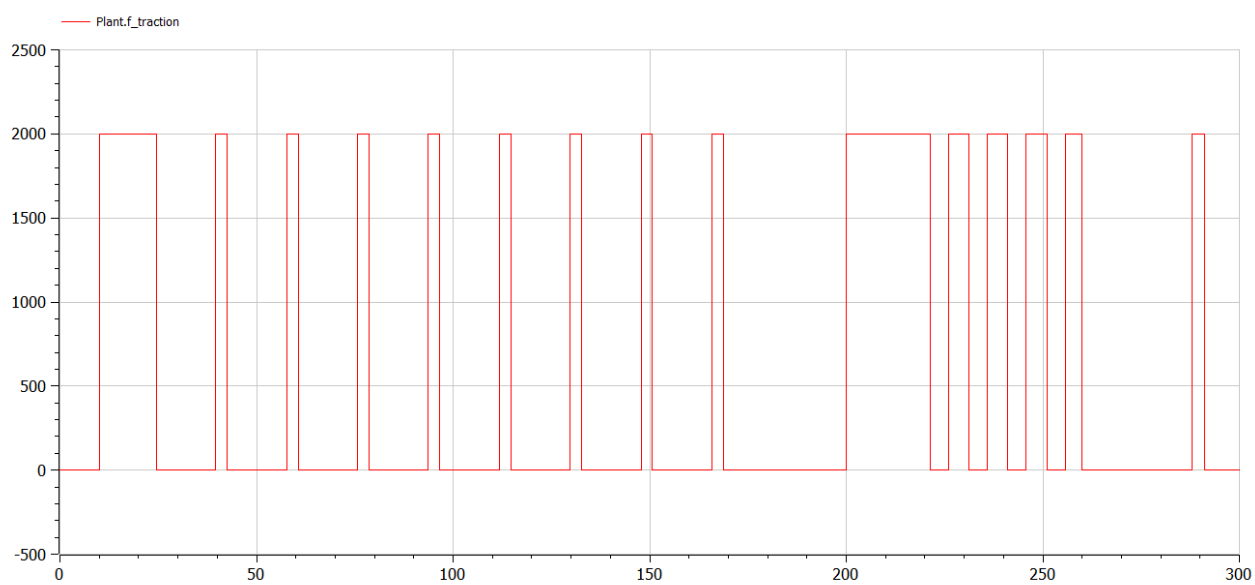
velocity_ideal
Plant.x_psgr
Plant.v_trolley
25
20
15
10
5
0
-5
0
50
100
150
200
250
300
time (s)



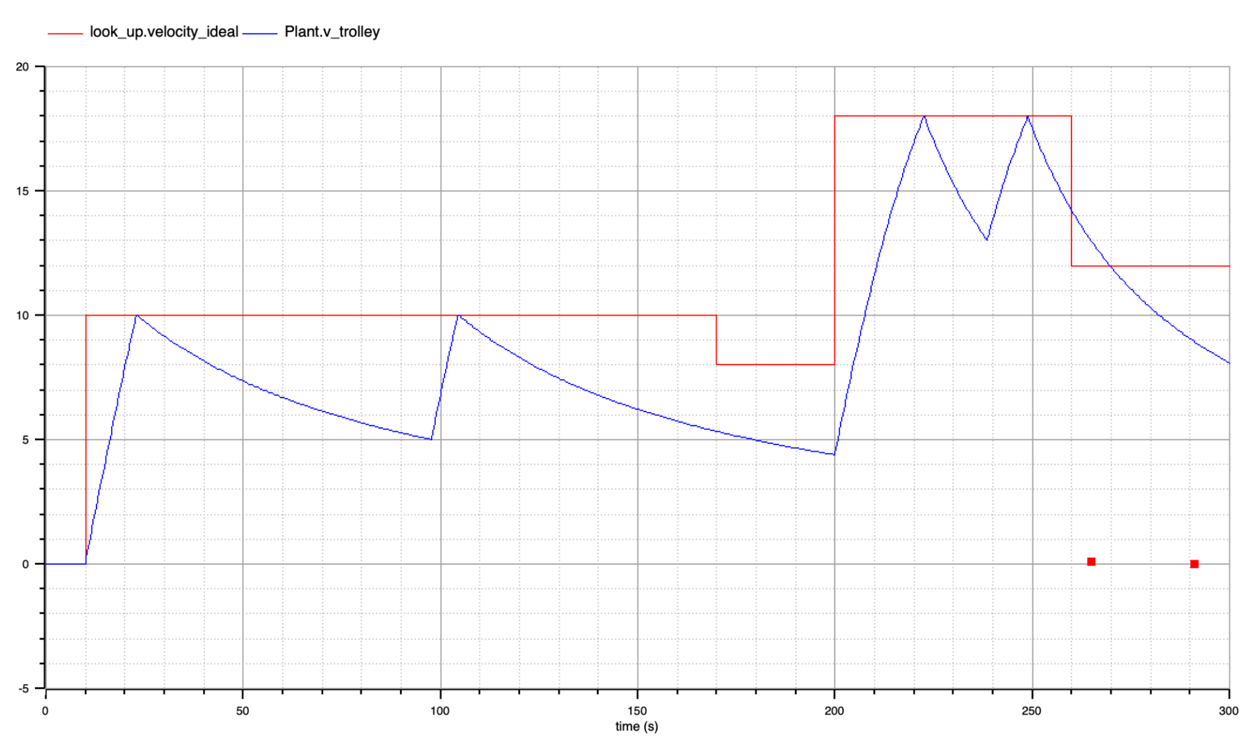
Plant.x_psgr
0.2
0.1
0
-0.1
-0.2
-0.3
0
50
100
150
200
250
300



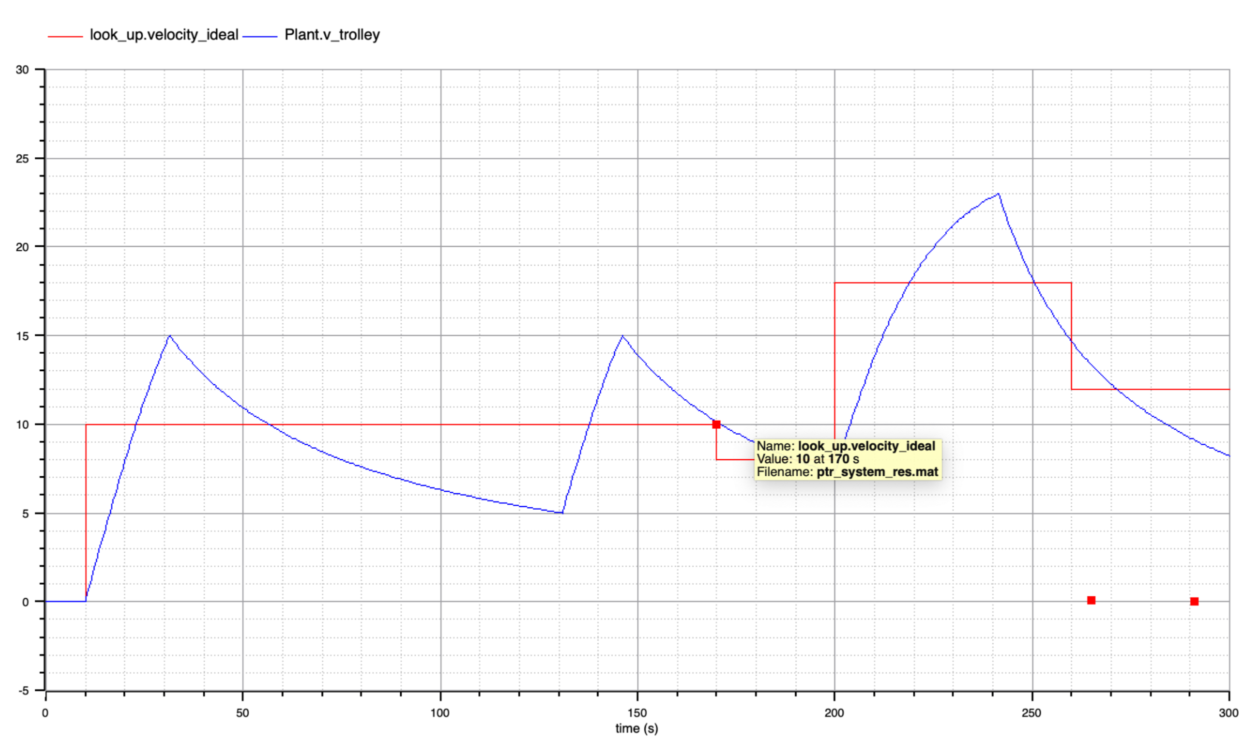
The passenger doesn’t fall. You can see on every change of acceleration of the trolley also a change in the people displacement.



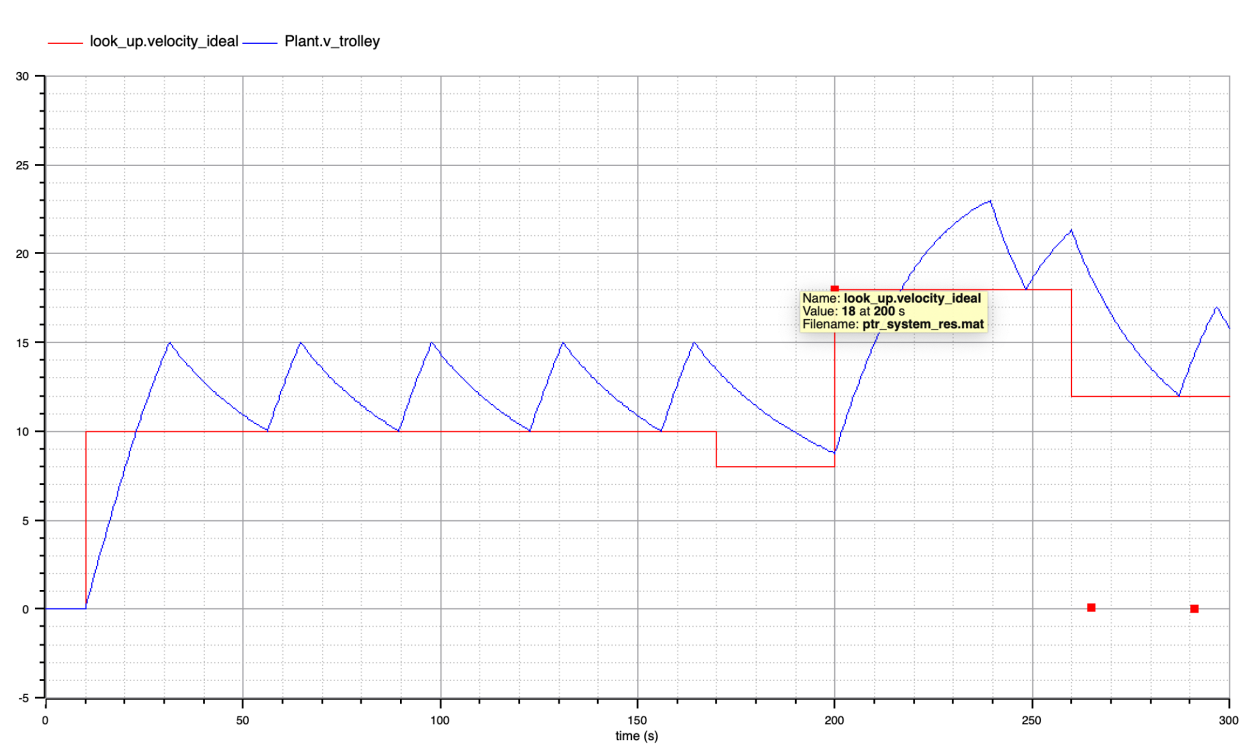
5. TODO



The result of the velocity wrt ideal velocity for d_min = 0, d_max = 5. It is clear to see that the controller undershoots the trolley by 5m/s.



The result of the velocity wrt ideal velocity for d_min = -5, d_max = -5. It is clear to see that the bang-bang effect distributed much longer with much lower acceleration of the trolley.

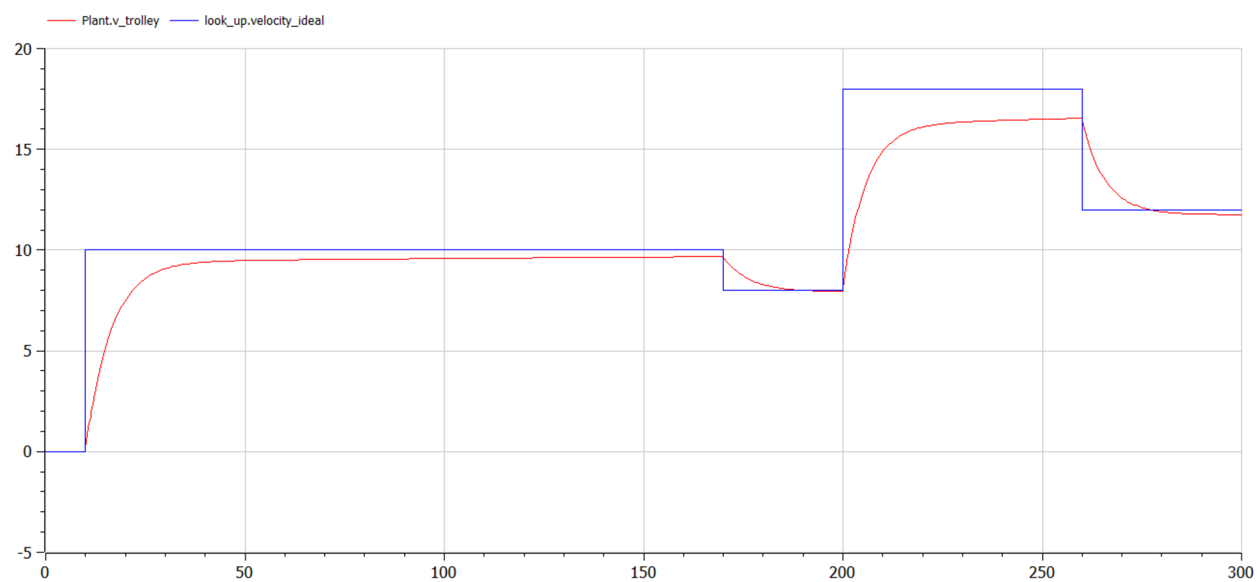


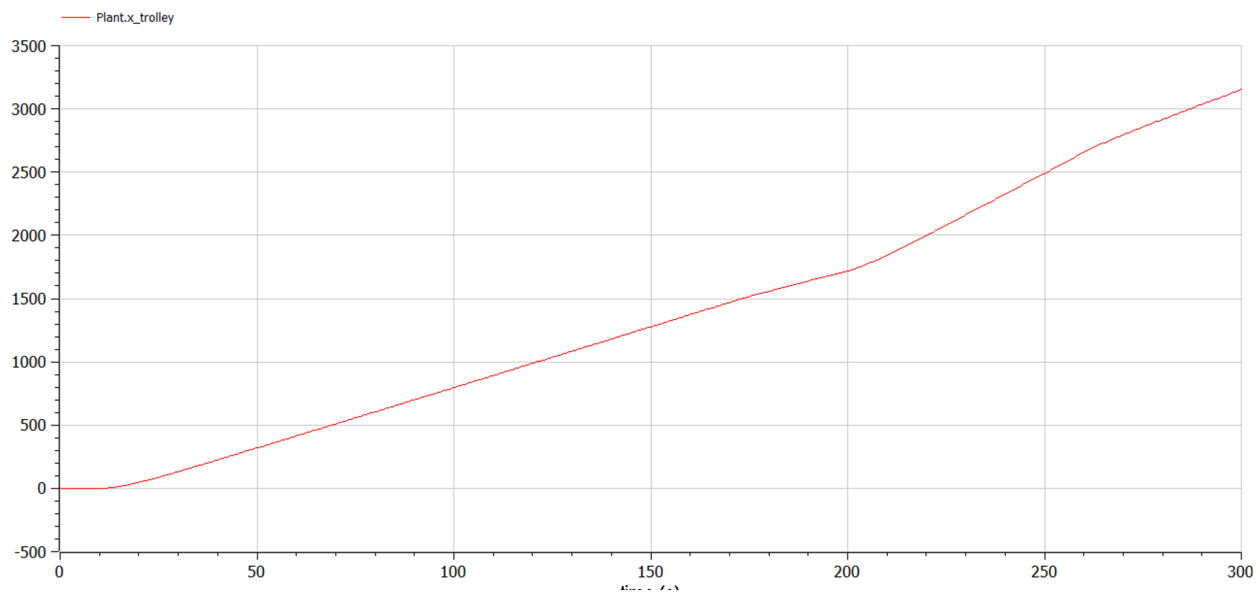
The result of the velocity wrt ideal velocity for d_min = -5, d_max = 0. It is clear to see that the controller overshoots the trolley by 5m/s.

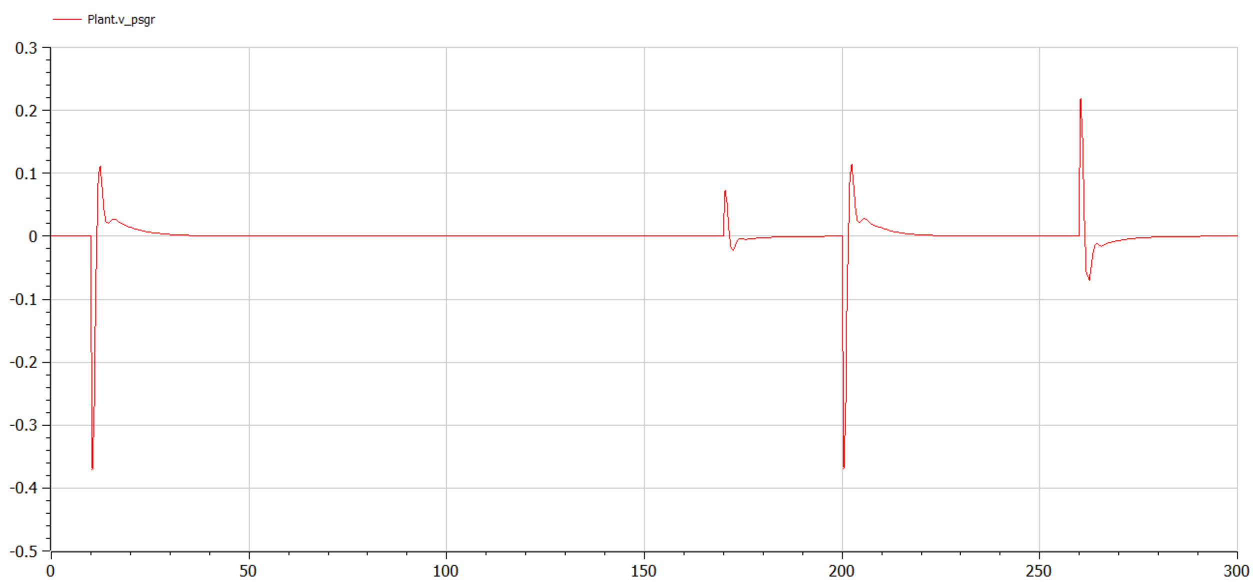
When we changed the when-statements to if statements, unfortunately If d_min < u < d_max; the simulation comes to a halt. Therefore I couldn't observed the results properly. However, the difference between if statements and when statements in Modelica, when statements triggered only if the expression is true and if statements remains active as long as the expression is true.

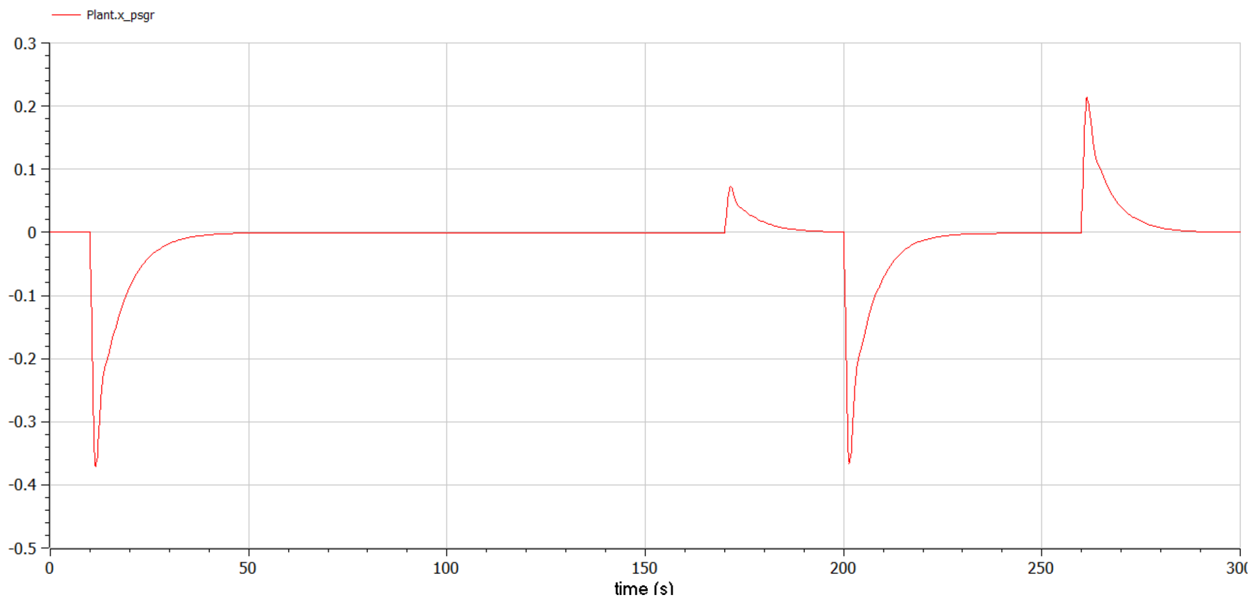
6. PID-controller

You don't see the periodic "Bang-Bang" effect anymore. The trolley speed doesn't cross the ideal velocity as often as with the Bang-Bang-controller. There are only sudden changes in the acceleration of the passenger when the lookup speed changes value. There are though 2 points where the x_psgr becomes lower than 0.35, so he falls with our chosen values for Ki, Kp and Kd.




Plant.x_trolley
3500
3000
2500
2000
1500
1000
500
0
-500
0
50
100
150
200
250
300


Plant.v_psgr
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
50
100
150
200
250
300
time (s)



7. Tuning the PID-controller

In our cost function, we try to minimize the speed difference (v_ideal-v_trolley). Because this might be negative, we squared this. We also added x_psgr and also squared this because this can be negative too. The weight of the x_psgr was 0.5 because it doesn't matter that much, as long as the passenger doesn't fall. If the passenger falls, the cost will be a penalty value of 1000000 to be sure it is high enough to never get selected.
We created a Python script to find the optimal values for Ki, Kp and Kd according to this cost function.
The optimal values we found were ki=1.5, kd=0.0 and kp=320.0.