CONTROL THEORY PROJECT

3 exercises about introduction to control theory

Chloé ALBESA

Winter semester 2018

Summary

Introduction

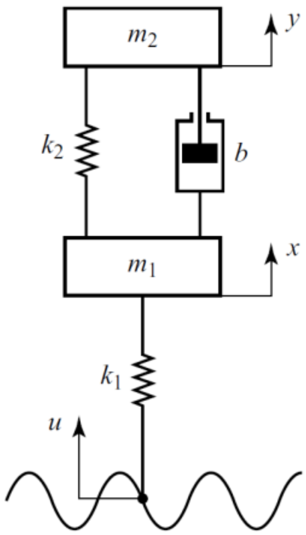
The purpose of this project is to apply notions that have been studied in seminars of control theory. All of the three exercises are based on the same scheme. We will first study the system itself, then its stability, then implement a controller in it and see which controller can be the best solution and how the controller improve the behaviour of our system according to what we are looking for.

The purpose is to apply different controllers to each system, in order to compare the different methods to each other and to find which one suit the best in every case. These exercises are realised with Matlab R2016b.

Exercise 1: Mass spring damper

1. Transfer function

This exercise is a classical mechanical system composed of a mass, a spring and a damper.



m_1	1.35	mass [kg]
m_2	1.45	mass [kg]
k_1	0.95	Spring constant [N/m]
k_2	1.05	Spring constant [N/m]
b	0.25	Damping constant [Ns/m]

You can see the schematic representation of the system and the given data that come along. Thanks to this scheme, we can establish the basic mathematical relationships defining this problem, using the principles of general mechanics.

$$m_1\ddot{x} = k_2(y - x) + b(\dot{y} - \dot{x}) + k_1(u - x)$$

$$m_2\ddot{y} = -k_2(y - x) + b(\dot{y} - \dot{x})$$

The input being u(s) and the output being the vertical position y(s), the transfer function we are looking for is expressed as:

$$H(s) = \frac{Y(s)}{U(s)}$$

The first step to follow in order to find the transfer function is the Laplace transform. The equations that we obtain are:

$$m_1 * s^2 * x(s) = k_2 * \big(y(s) - x(s)\big) + b\big(s * y(s) - s * x(s)\big) + k_1 * (u(s) - x(s))$$

$$m_2 * s^2 * y(s) = -k_2\big(y(s) - x(s)\big) + b(s * y(s) - s * x(s))$$

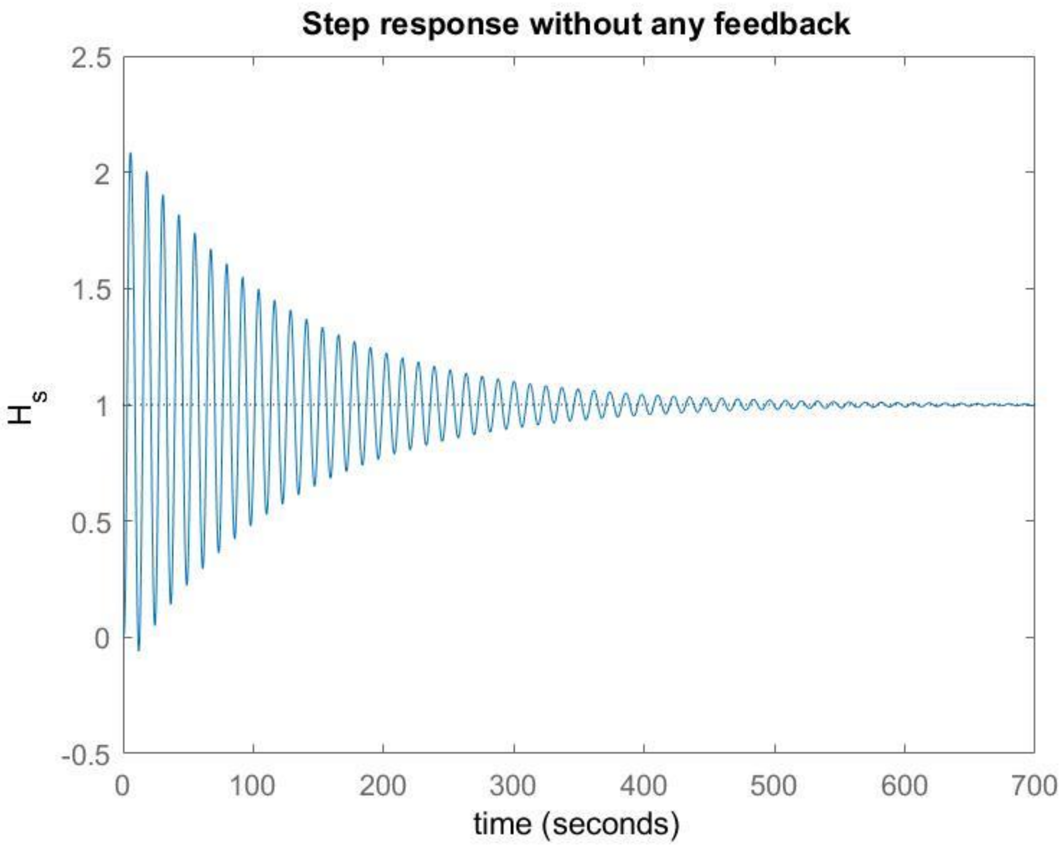
Using both these equations we can find the transfer function:

$$H(s) = \frac{K_1K_2 + K_1bs}{m_1m_2s^4 + b(m_1 + m_2)s^3 + [K_2(m_1 + m_2) + K_1m_2]s^2 + K_1bs + K_1K_2}$$

$$H(s) = \frac{0.1213s + 0.5096}{s^4 + 0.35765s^3 + 2.206s^2 + 0.1213s + 0.5096}$$

2. System analysis

a. Step response



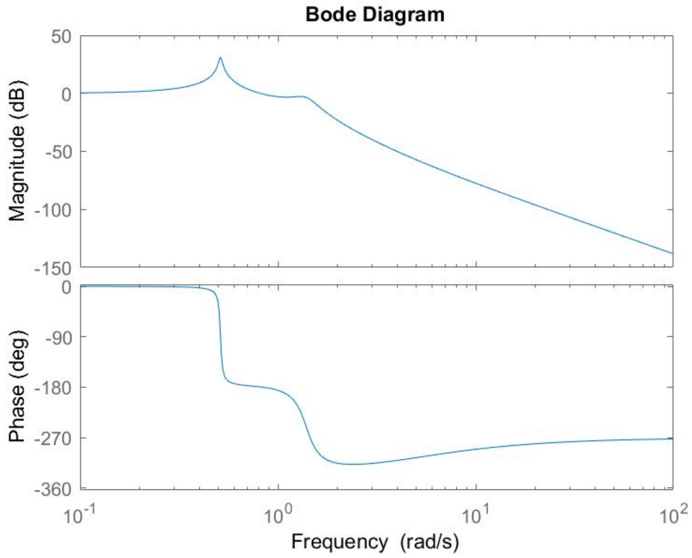
On this step response we can see that the system is stable (it converges to 0) even though there are a lot of oscillation at the beginning of the signal. This step is plotted without any feedback or controller. This is the pure system step response.

Thanks to Matlab, we can get some information about this step response:

```
    RiseTime: 1.6007
SettlingTime: 484.4305
 SettlingMin: -0.0596
 SettlingMax: 2.0859
   Overshoot: 108.5941
  Undershoot: 5.9556
        Peak: 2.0859
    PeakTime: 5.8689
```

Thanks to a short Matlab code, we were able to check that this system is stable even though he is oscillating a lot. Such a property can be both looked for or not according to what you are willing to do with the system. In this case, where we are studying a damper and a spring, it is possible that this property is looked for, however, is it easily explainable. With the controllers, we will try to suppress the oscillations or at least to reduce them in order to make the system reach its looked-for position without this behaviour.

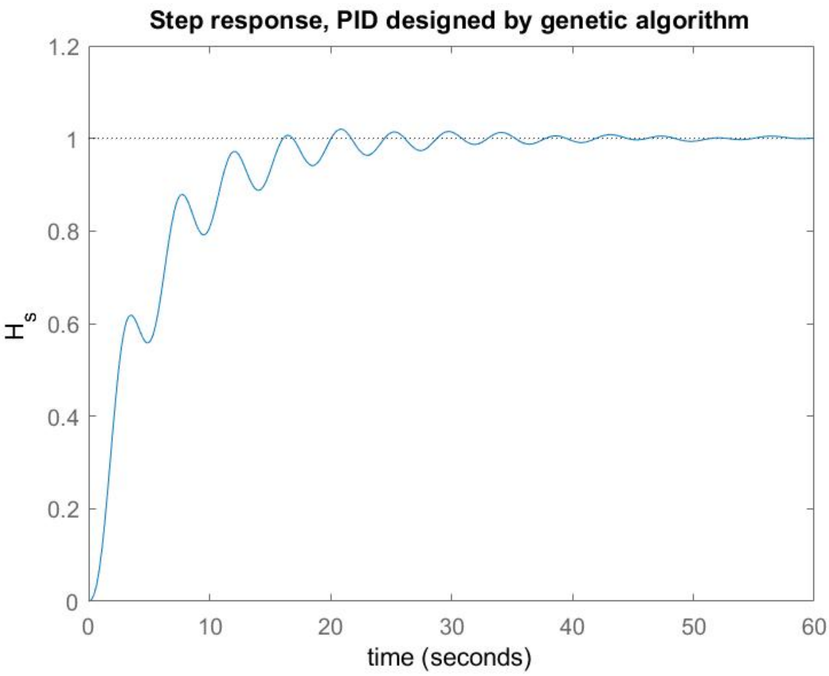
b. Frequency response



Once more, this frequency response is without any feedback or controller. According to the Bode Diagram, we can notice that this system behaves like a high pass filter.

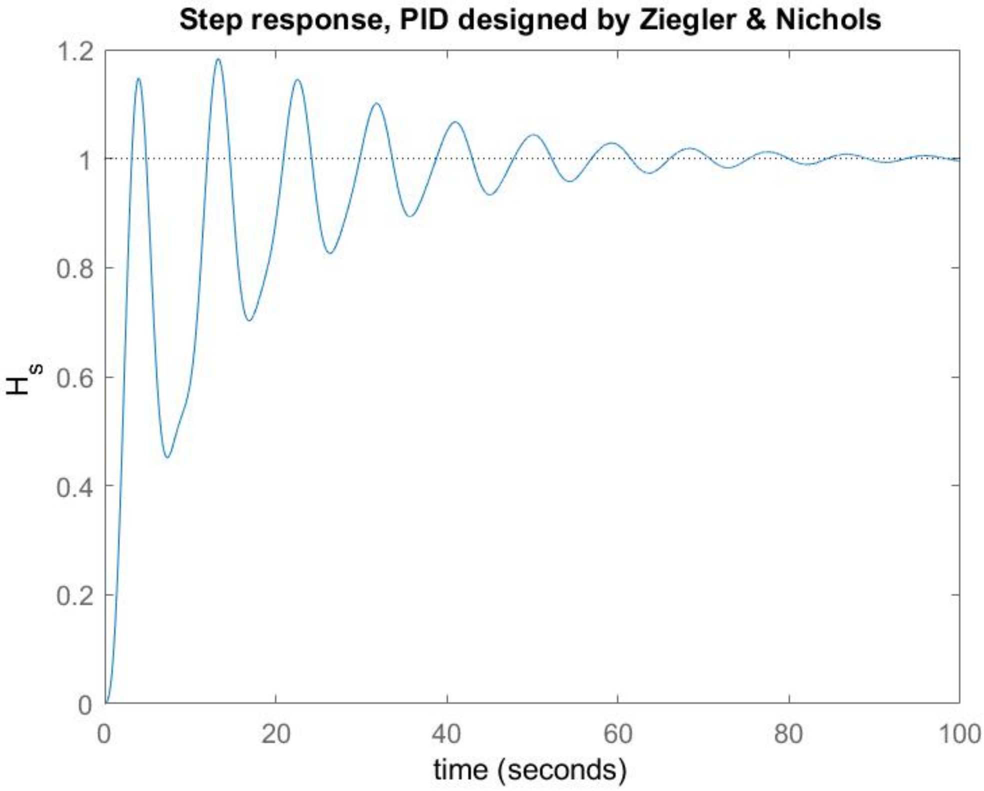
3. Controlled system

Results using Genetic Algorithm



With the use of a PID designed by a genetic algorithm, we can see that the oscillatory behaviour of the system is still present but really attenuated. The interesting fact with this controller is that there is no overshooting so, if this is an important requirement for your system this design can be good. On the other hand, this system is quite slow.

Results using Ziegler and Nichols controller



This controller has not the same effect as the previous one, as we can obviously see. The wanted point is reached for the first time far earlier but, on the other there is a big overshoot and quite a lot of oscillation. So, one more time, it depends on what you are willing to do with this application.

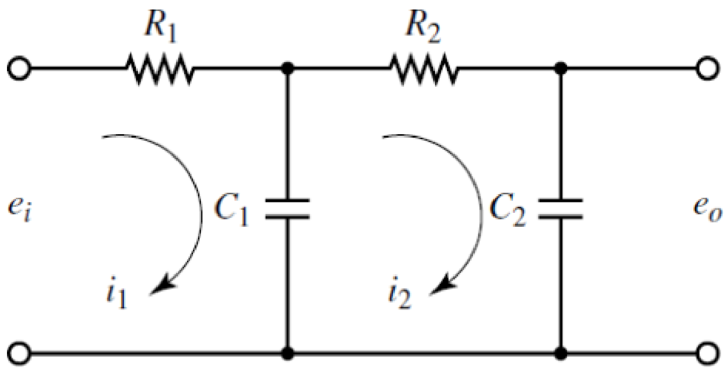
4. Conclusion about exercise 1 system and controller

Without any controller, the most important characteristic of this system is its oscillatory behaviour. This behaviour can be attenuated thanks to controllers but if you want more stability (by limiting the oscillations) you must accept to have a slower system; a compromise has to be done.

Exercise 2: RC Circuit

1. Transfer function

This system is an electronic system composed of a resistor and a capacitor. The scheme of this system is the following:



C_1	1.10^{-6}	capacitor [F]
C_2	$1.15.10^{-6}$	capacitor [F]
R_1	5.10^{5}	resistor [Ω]
R_2	$0.5.10^{4}$	resistor [Ω]

Thanks to this schema and using basic electronic laws we can establish 3 equations for this system:

$$\frac{1}{C_1}\int (i_1 - i_2)dt + R_1 i_1 = e_i$$

$$\frac{1}{C_1}\int (i_2 - i_1)dt + R_2 i_2 + \frac{1}{C_2}\int i_2 dt = 0$$

$$\frac{1}{C_2}\int i_2 dt = e_0$$

The input being $e_i(s)$ and the output being $e_0(s)$ the transfer function we are looking for is expressed as:

$$H(s) = \frac{E_0(s)}{E_i(s)}$$

The first step to follow in order to find the transfer function is the Laplace transform. The equations that we obtain are:

$$\frac{1}{C_1 s} * [I_1(s) - I_2(s)] + R_1 I_1(s) = E_i(s)$$

$$\frac{1}{C_1 s}[I_2(s) - I_1(s)] + R_2 I_2(s) + \frac{1}{C_2 s} I_2(s) = 0$$

$$\frac{1}{C_2 s} I_2(s) = E_0(s)$$

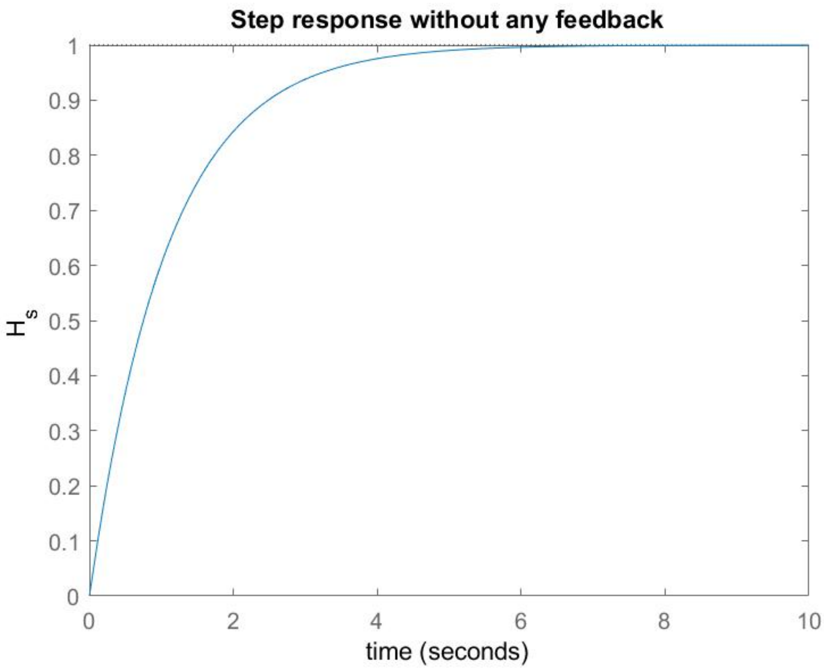
Using both these equations we can find the transfer function:

$$H(s) = \frac{1}{R_1 R_2 C_1 C_2 s^2 + (R_1 C_1 + R_1 C_2 + R_2 C_2)s + 1}$$

$$H(s) = \frac{347.8}{s^2 + 375.9s + 347.8}$$

2. System analysis

a. Step response

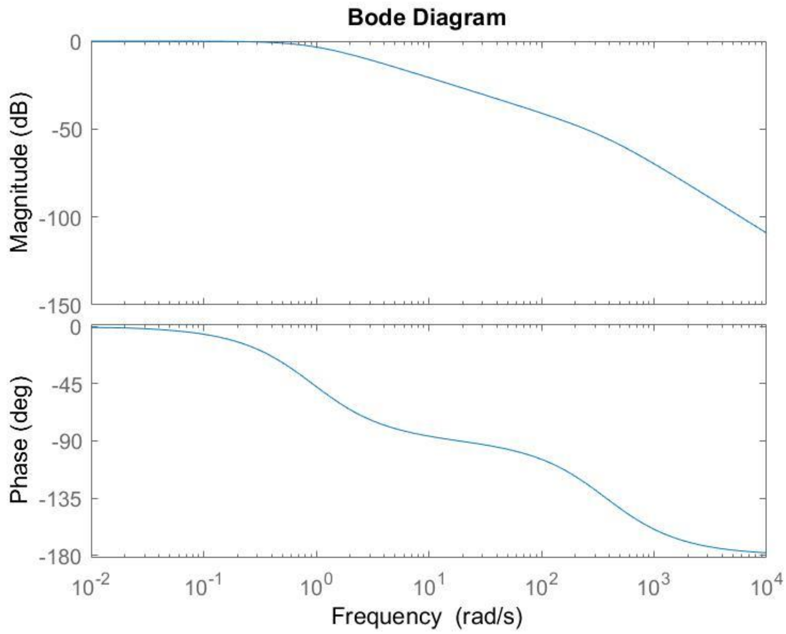


Thanks to Matlab, we can get some information about this step response:

```
    RiseTime: 2.3686
SettlingTime: 4.2202
 SettlingMin: 0.9043
 SettlingMax: 1.0000
   Overshoot: 0
  Undershoot: 0
        Peak: 1.0000
    PeakTime: 11.3693
```

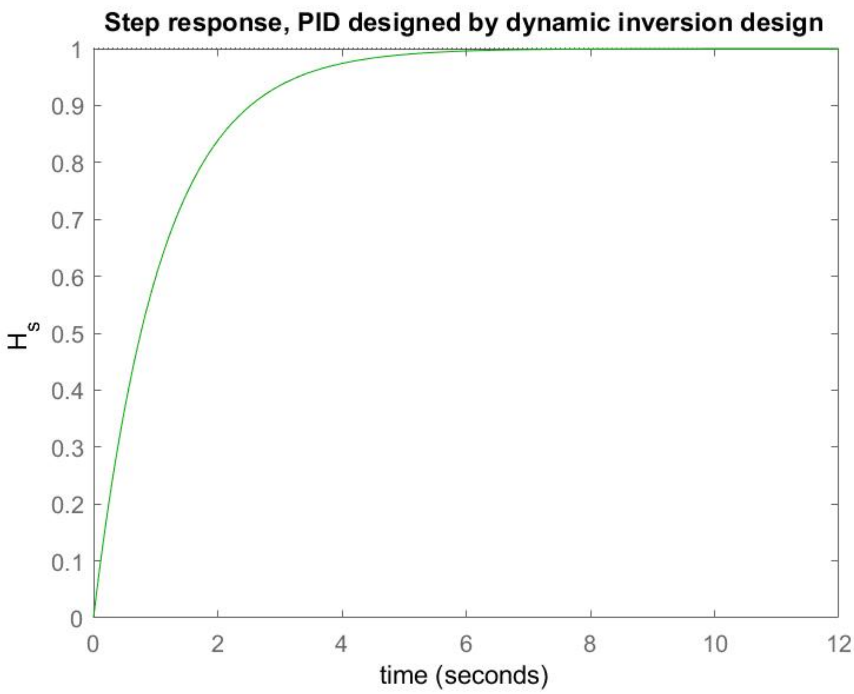
Thanks to Matlab code, we know that this system is stable. This response is naturally stable enough, there is no oscillation or overshoot.

b. Frequency response



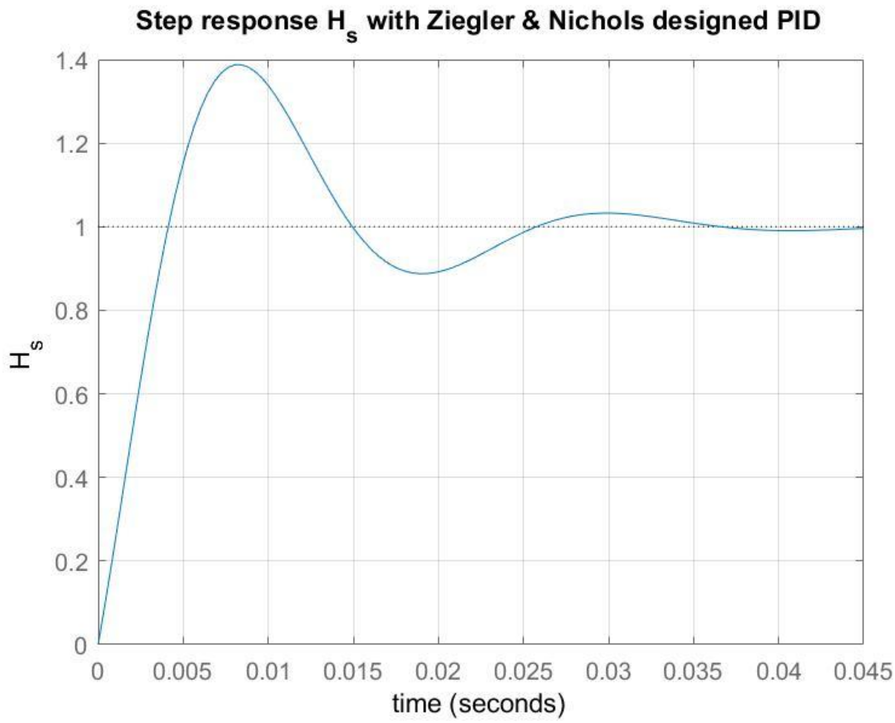
3. Controlled system

Using dynamic inversion method



This method does not show great results. We cannot say it is bad results as the system is still stable, without any oscillation, but it does not bring any improvement.

Using Ziegler and Nichols



With this method, we can see that the system is far faster than without controller, yet the system now shows some oscillation as well as an overshoot. As usual, there must be a compromise between stability and rapidity.

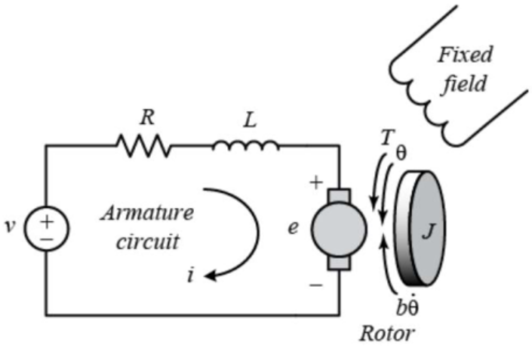
4. Conclusion about exercise 2 system and controller

This system is naturally stable, yet not really fast. The dynamic inversion design did not help to improve our system, whereas the Ziegler and Nichols propose a solution where the system is much faster, but as a compromise, less stable. Once more, it depends on the application for the system.

Exercise 3: DC motors

1. Transfer function

This system is an electro-mechanic system composed of a DC motor.



J	$3.2284.10^{-6}$	moment of inertia of the rotor $[kg.m^2]$
b	$3.5077.10^{-6}$	motor viscous friction constant [$N.m.s$]
K_b	0.0274	electromotive force constant [$V/rad/sec$]
K_t	0.0274	motor torque constant [$N.m/A$]
R	4	electric resistance [Ω]
L	$2.75.10^{-6}$	electric inductance [H]

Thanks to this schema and using basic electro-mechanic laws we can establish 2 equations for this system:

$$J\ddot{\theta} + b\dot{\theta} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta}$$

Two different transfer functions are asked from us. In both case the voltage V is the input but is one case the output will be the angular speed $\dot{\theta}$ and in the other case, it will be the angular position θ. So, the two transfer equations that we want to get are:

$$H(s) = \frac{\dot{\theta}(s)}{V(s)}$$

$$H(s) = \frac{\theta(s)}{V(s)}$$

The first step to follow in order to find the transfer function is the Laplace transform. The equations that we obtain are:

<u>For the angular speed case</u>

$$[Js + b]\dot{\theta}(s) = KI(s)$$

$$[Ls + R]I(s) = V(s) - K\dot{\theta}(s)$$

For the angular position case

$$[Js^2 + bs]\theta(s) = KI(s)$$

$$[Ls + R]I(s) = V(s) - Ks\theta(s)$$

So finally, we get two different transfer function expressions:

For the angular speed case

$$H(s) = \frac{K}{LJs^2 + (Lb + RJ)s + (Rb + K^2)}$$

$$H(s) = \frac{3.086 * 10^9}{s^2 + 1.455 * 10^6 * s + 8.614 * 10^7}$$

For the angular position case

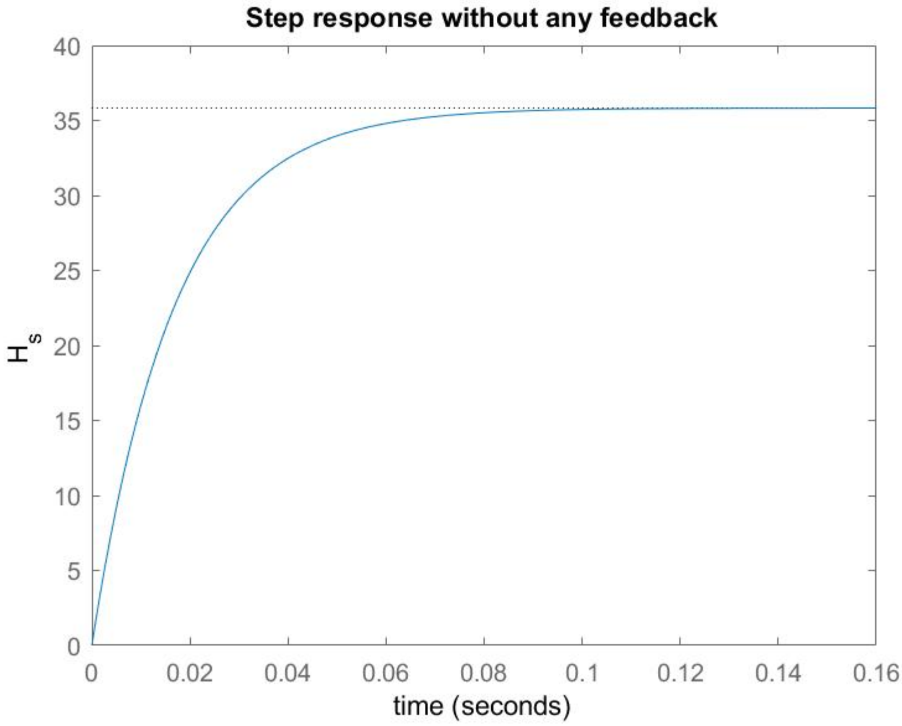
$$H(s) = \frac{K}{LJs^3 + (Lb + RJ)s^2 + (Rb + K^2)s}$$

$$H(s) = \frac{3.086 * 10^9}{s^3 + 1.455 * 10^6 * s^2 + 8.614 * 10^7 * s}$$

2. System analysis

a. Step response

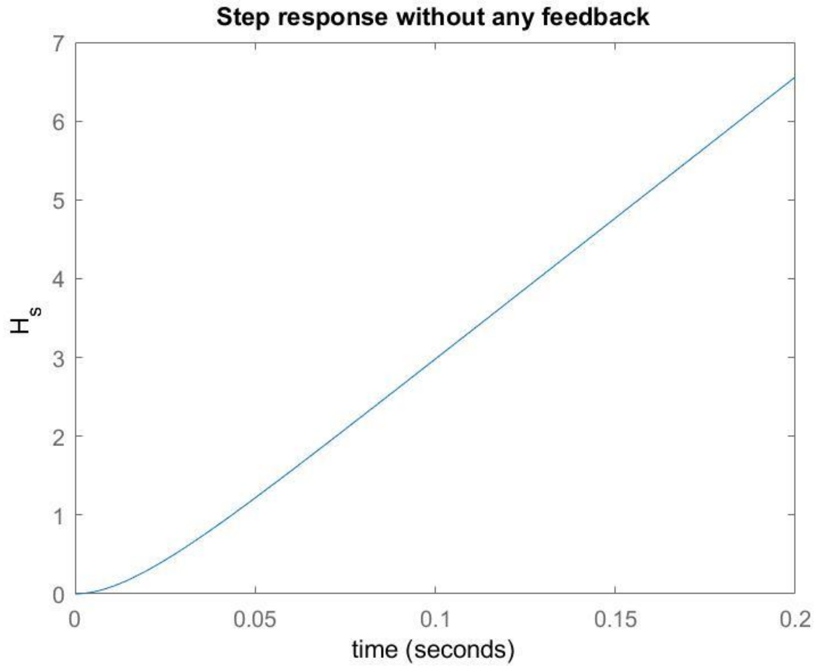
For the angular speed case




```
     RiseTime: 0.0371
 SettlingTime: 0.0661
  SettlingMin: 32.4052
  SettlingMax: 35.8258
    Overshoot: 0
   Undershoot: 0
         Peak: 35.8258
     PeakTime: 0.1781
```

Thanks to Matlab code, we know that this system is stable. The system is naturally without any oscillation or overshoot but is quite slow. Maybe a possible improvement for the controller designing is to make the system faster.

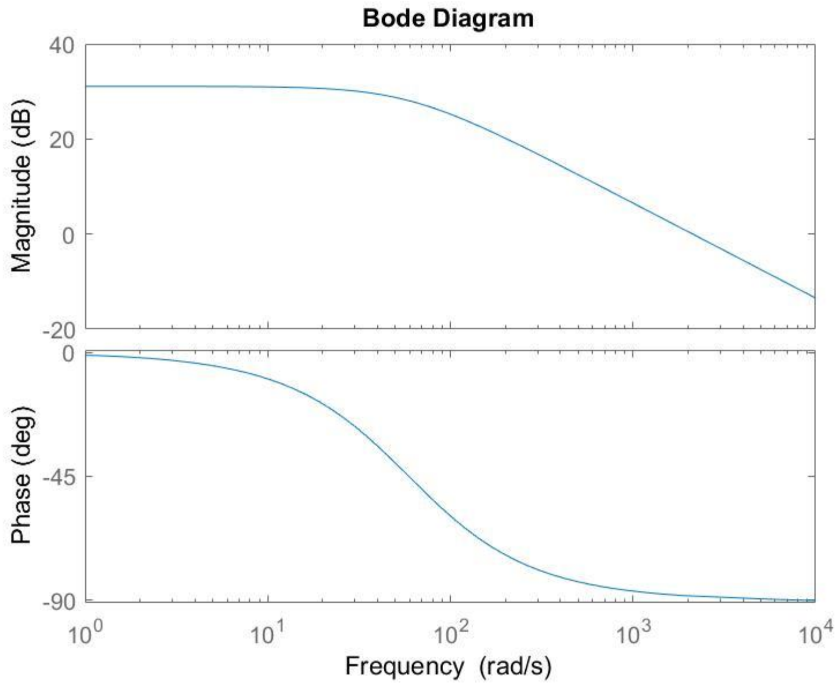
<u>For the angular position case</u>



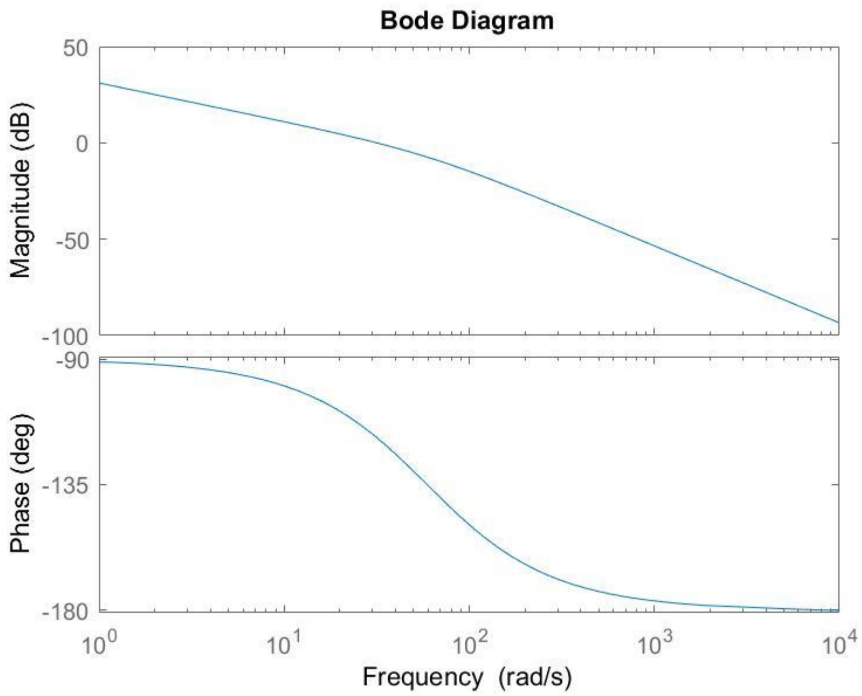
This step response may seem a bit strange but can be understand regarding the situation. We are studying the position of a DC motor which is running. The position we are looking into is the angular position. Even though the real position is between 0 and 2π, the sensor does not seem to take into account that a revolution has been done and goes on counting. That is why we obtain such a strange, or at least not expected plot.

b. Frequency response

For the angular speed case



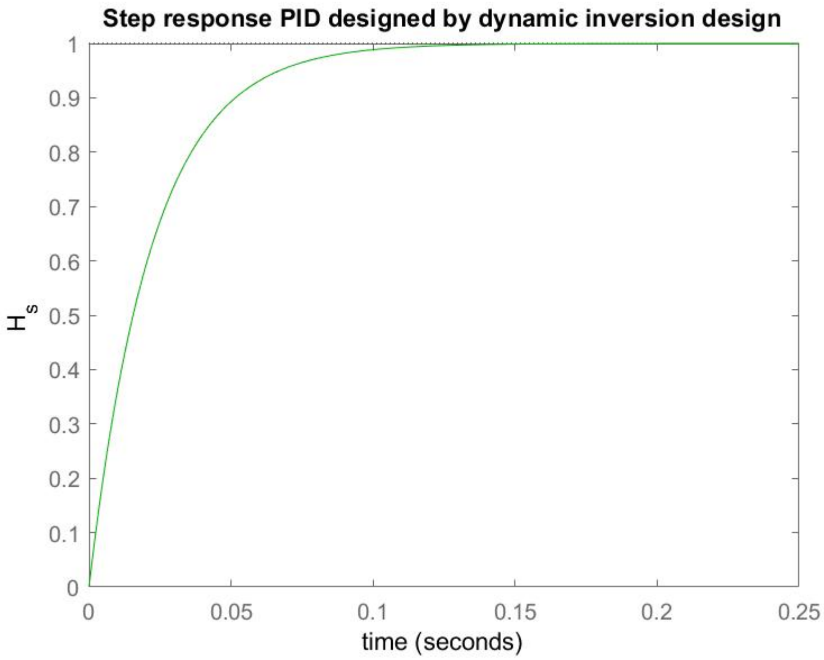
For the angular position case



3. Controlled system

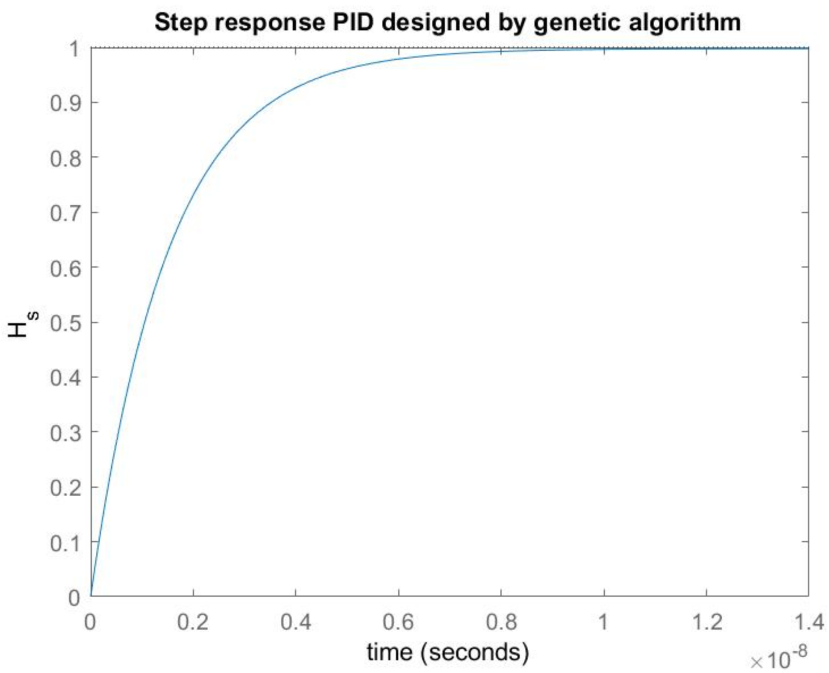
a. Angular speed case

Dynamic inversion design



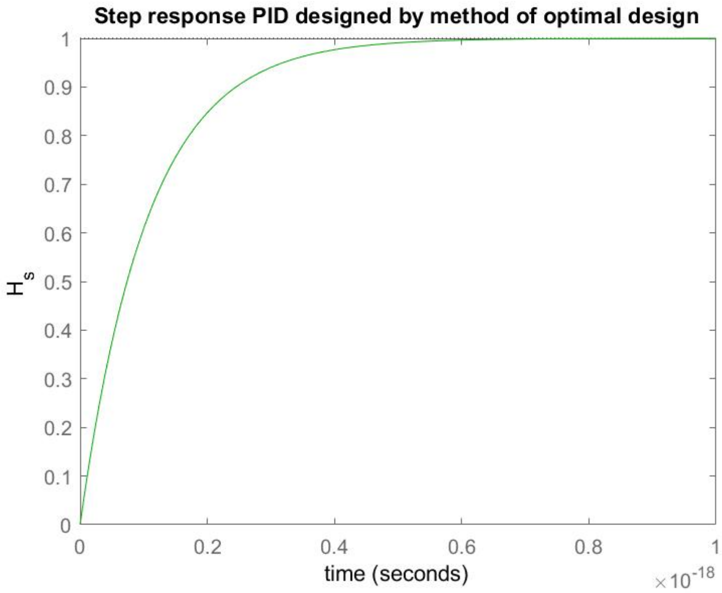
This controller makes the system a bit faster but not on a noticeable way. This controller is not really conclusive, even though it did not make the system unstable or with an overshoot, it does not improve the rapidity as well.

Genetic algorithm



This time, the system is way faster (be careful with the scale, it is multiplied by 10^{-8}). And this improvement had no impact on the stability or the overshoot. It is a really great result if you do not wish any overshooting.

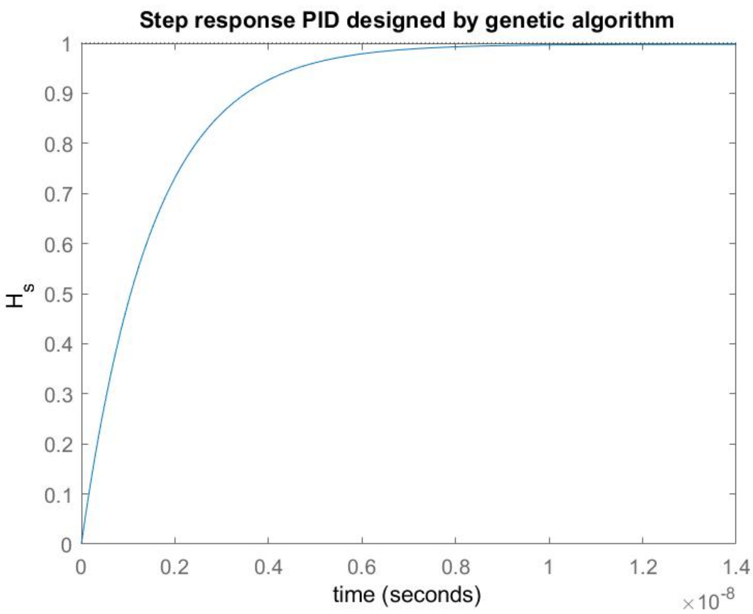
Optimal design



This controller improves the rapidity of the system even more (same remark about the scale). And once more, no overshoot or oscillation are present. This controller is a great controller if you do not wish an oscillatory behaviour.

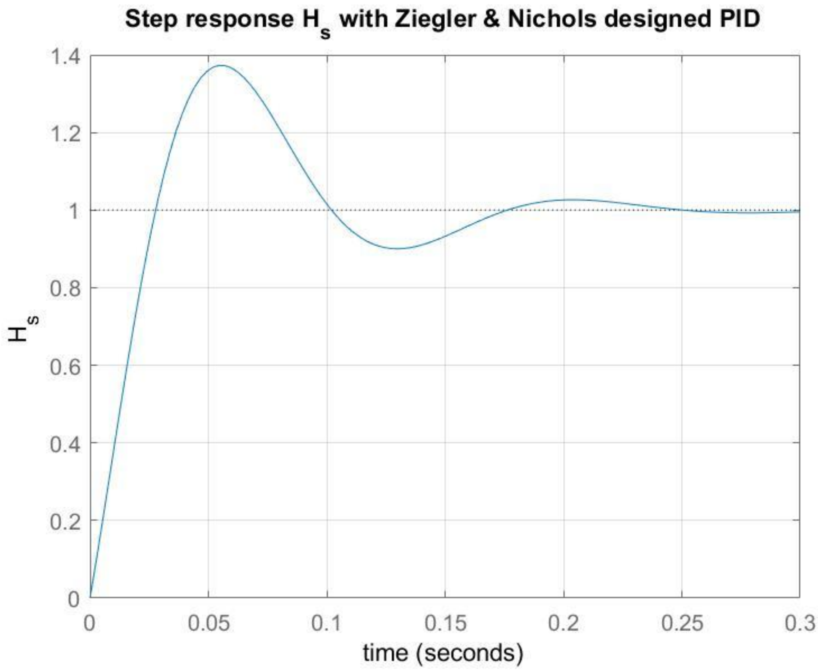
b. Angular position case

Genetic Algorithm



Once more, this type of controller enables a faster system. It does not create oscillation or overshoot. It is an interesting if you do not wish an overshoot.

Ziegler and Nichols



This controller makes the system faster but create a small oscillation and an overshoot. This solution can be interesting if you wish an overshoot. Otherwise, the one designed with the genetic algorithm should be better.

4. Conclusion about exercise 3 system and controller

In this case, we observed that the dynamic inversion design does not seem to be interesting, the results showed no real improvement.

In both case, the genetic algorithm gives good results. But the best results are showed by the method of optimal design, which made the system way faster.

Conclusion about the project

Through this project, we applied different notions that had been seen during seminars. We used differential equation to define systems, turned them into Laplace transform and from that, we get the transfer functions. Thanks to the study of those transfer functions, we were able to learn more about the behaviour of the system and about its stability.

Then we applied various controller to them, to see how they could affect or improve the systems and finally to decide which one was the best for each system. However, to decide which controller really is the best and should be used, we would need more information about the context. Depending on the attempts placed on the system, the best controller may not always be the same. It will depend on your specifications, for example if you need more of a rapid system or of a stable system.