Control Theory (RMS-1-612-s ; RIMA-1-612-s)

Laboratory no. 1 – Classical Feedback Control (updated: 28th January 2021)

Issue date: 1st March 2021
Due date: 24th March 2021

Phong Ba Dao, dr hab. inż., prof. AGH
Department of Robotics and Mechatronics
AGH University of Science and Technology

Keywords: feedback control, linear time-invariant (LTI) systems, transfer functions, step response, Bode plot, Nyquist plot, root locus plot, conditionally stable systems, PID controllers, Ziegler-Nichols method, Matlab/Simulink, Control Systems Toolbox.

Recommended reading: Chapters 3,4,5,6 of [1] and Chapters 4,7,8,9 of [2].

[1] Roland S. Burns, *Advanced Control Engineering*, Butterworth-Heinemann, 2001, ISBN 978-0-7506-5100-4.

[2] Richard C. Dorf and Robert H. Bishop, *Modern Control Systems*, 8th edition, Addison-Welley, Boston, MA, 1997, ISBN: 978-0-2013-0864-8.

Remarks:

(1) Short instructions are provided to support you in solving exercises. You should read the instructions carefully. Though some parts of the exercises are explained and guided in the instructions but they are general solutions and not yet complete. Therefore, based on the instructions provided you should elaborate and discuss in the report your solutions and results obtained in detail.

(2) Before using Matlab commands (or functions) you should first check the syntax, descriptions, input arguments, output arguments, and illustrated examples by typing help "command name".

(3) The report must be typed on a word-processor and submitted as a single PDF file via **UPeL platform**.

Exercise 1

This exercise considers the *effect of dominant poles* on the dynamic characteristics of a control system.

(a) Construct ten poles as follows: The pole number one is placed at +1. Other poles are equally spaced around the unit circle in the complex *s*-plane. Now construct a transfer function with no zeros and the five poles, out of the ten, which are in the left half of the complex *s*-plane.

(b) Compute and plot the step response of the system you created in part (a). What are the time-domain performance specifications (i.e. the time-to-peak, percentage overshoot, rise time, and settling time) obtained for this step response?

(c) Repeat parts (a) and (b) for systems with six and fourteen poles. Compare the results. If n is equal to the number of poles of the system, do you observe any trends with respect to the time-domain performance specifications as n increases? Explain.

Exercise 2

This exercise investigates a *stiff system* that has a pole close to the origin of coordinates (i.e. a very small pole) and other poles very far away from it, all in the left half of the complex s-plane.

Consider a linear system represented by the following transfer function

$$G(s) = \frac{1}{(s+20)(s+0.001)(s+1)}$$

a) Is the (open-loop) system given by the above transfer function stable? Explain.

b) Create a Nyquist plot for the system and then blow up (or magnify) the region near the origin. Display only the region limited by $-0.003 \le \mathrm{Re}\{G(j\omega)\} \le 0$ and $-0.0005 \le \mathrm{Im}\{G(j\omega)\} \le 0.0005$. How

do we know this region would be interesting? Explain. Based on the Nyquist plot, is the closed-loop system stable or unstable? Explain.

Exercise 3

This exercise investigates the system which is *conditionally stable*.

a) Create Bode, Nyquist and root locus plots for the linear transfer function shown in the figure below.

b) What can you say about the frequency response plots of this conditionally stable system? Explain.

c) Based on the Nyquist plot, is the closed-loop system stable or unstable? Explain.

d) Based on the Bode and root locus plots, determine for what approximate ranges of values of the gain K and frequency ω the closed-loop system is stable (or unstable)?

e) Select one value of the gain K that makes the closed-loop system unstable and another one that makes the closed-loop system stable and then plot the corresponding step responses to confirm the results found in part (3d).

It should be noted that you solve this exercise in Matlab by using m-files or scripts, therefore it may not be necessary to create the following block diagram in Simulink. It can be understood that the block "Saturation" in Figure 1 represents the meaning (or functionality) of the gain K.

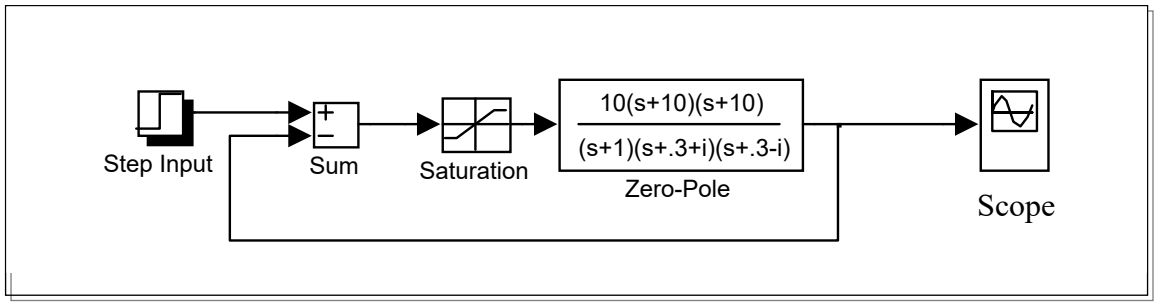


Figure 1. Block diagram of the conditionally stable system.

Exercise 4

This exercise revises two classical methods commonly used for the design of feedback control systems. Design PD or PID controllers for the system $G(s) = \dfrac{8}{(s+0.01)(s+0.2)(s+1)}$ by using:

a) The Root-Locus Design GUI (**sisotool**) tool of Matlab Control System Toolbox such that both the following time-domain specifications are obtained:

- Dominant closed-loop poles damping ratio $\zeta = 0.707$;

- Dominant closed-loop poles time constant $\tau = 5$ seconds.

Notice that: As we have $\xi\omega_n = \dfrac{1}{\tau} \Rightarrow$ the natural frequency $\omega_n = \dfrac{1}{0.707 * 5} = 0.2829$ [rad/s]. The setting time $t_s \cong 4\tau = 20$ seconds. Therefore, one can use (ξ, ω_n) or (ξ, t_s) as the design constraints and then check if the designed system meets all required specifications (ξ, ω_n, t_s).

b) The Ziegler-Nichols method for the same specifications mentioned above.

Discuss and compare the results obtained by using these two methods. Which method you would prefer to use? Explain.

Instructions

Exercise 1:

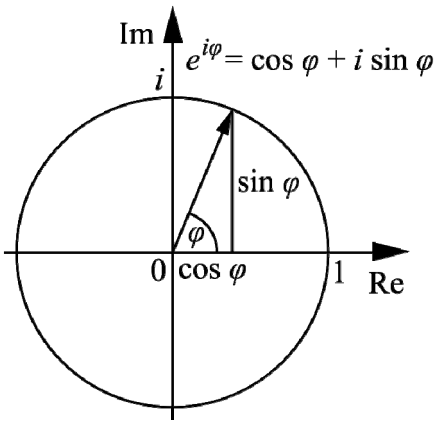


Figure 2.

These formulas can be used to compute the desired poles: $\varphi^{(rad)} = \frac{2\pi\varphi}{360}$; $pole = 1e^{i\varphi^{(rad)}}$

It is advised to investigate the distance between complex pole pairs and the imaginary axis. The complex pole pair – which is closest to the imaginary axis – is called the dominant poles that primarily decide the dynamic behaviour of the given system.

Matlab functions:

pzmap(SYS) computes the poles and (transmission) zeros of the Linear Time-Invariant (LTI) model SYS and plots them in the complex plane. The poles are plotted as x's and the zeros are plotted as o's.

[P,Z] = pzmap(SYS) returns the poles and zeros of the system in two column vectors P and Z.

SYS = tf(NUM,DEN) creates a continuous-time transfer function SYS with numerator(s) NUM and denominator(s) DEN.

zp2tf performs zero-pole to transfer function conversion, e.g. [NUM,DEN] = zp2tf(Z,P,K);

tf2zp performs transfer function to zero-pole conversion, e.g. [Z,P,K] = tf2zp(NUM,DEN);

ltiview opens the LTI Viewer GUI, e.g. ltiview({'pzmap';'step';'nyquist';'bode'}, SYS);

S = stepinfo(SYS) computes the step response characteristics for the LTI model SYS (created with either TF, ZPK, or SS).

The time-domain performance specifications for a step response can also be obtained by analysis of the plot, for example, by right mouse click on the plot and select "Characteristics → Peak Response".

Exercise 2:

A stiff system is a system having some poles close to the origin of coordinates (i.e. they are very small poles) and other poles very far away from them, all in the left half of the complex s-plane. In other words, stiff systems are stable systems in which very slow and very fast modes (or poles) co-exist.

It should be noted that the Nyquist plot is created for the open-loop system.

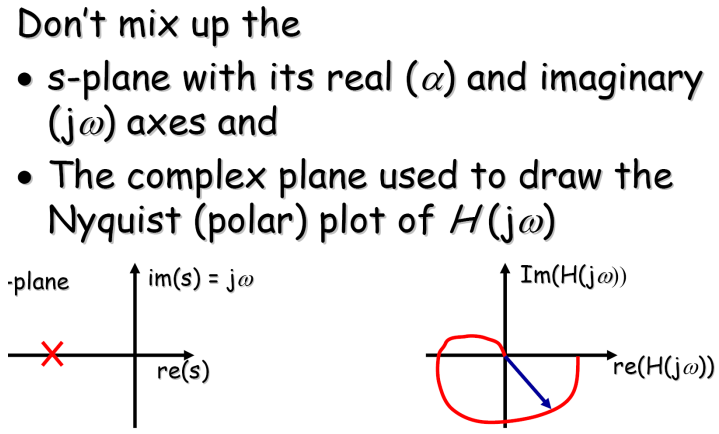
Focus on the fact that at the first glance on the Nyquist plot, it looks like that we have a first-order system. However, the transfer function says that it is a third-order system.

There are two formulations of the Nyquist criterion of stability depending on whether the open-loop system is stable or unstable.

1. If the open-loop system is stable, the closed-loop system would be stable if the Nyquist plot of the open-loop system does not encircle the point with coordinates (-1, j0) at the complex plane.

2. If the open-loop system is unstable, the closed-loop system would be stable if the Nyquist plot of the open-loop system encircles the point with coordinates (-1, j0) at the complex plane k times (where k is

the number of roots in the right-half complex *s*-plane RHP) in the counter-clockwise direction when frequency changes from zero to infinity.



Matlab functions:

nyquist(SYS) draws the Nyquist plot of the LTI model SYS (created with either TF, ZPK, or SS). The frequency range and number of points are chosen automatically.

axis([XMIN XMAX YMIN YMAX]) sets scaling for the x- and y-axes on the current plot.

Exercise 3:

It should be noted that the Bode, Nyquist and root locus plots are created for the open-loop system.

For part (3c) it is advised to blow up the region near the origin and the point (-1, j0) of the Nyquist plot.

The root locus plot is used to analyse the negative feedback loop and show the trajectory of the closed-loop poles when the feedback gain K varies from 0 to ∞.

A conditionally stable system is one in which the phase delay of the loop exceeds -180 degrees while there is still gain in the loop (see Figure 3). Another definition: One system with variable gain is conditionally stable if it is Bounded-Input Bounded-Output (BIBO) stable for certain values of gain, but not BIBO stable for other values of gain. In other words, for some ranges of the gain K and frequency ω the system is stable, but for other ranges the system is unstable. If a system is conditionally stable it is important to determine for what ranges of K and ω the system exhibits stability or instability.

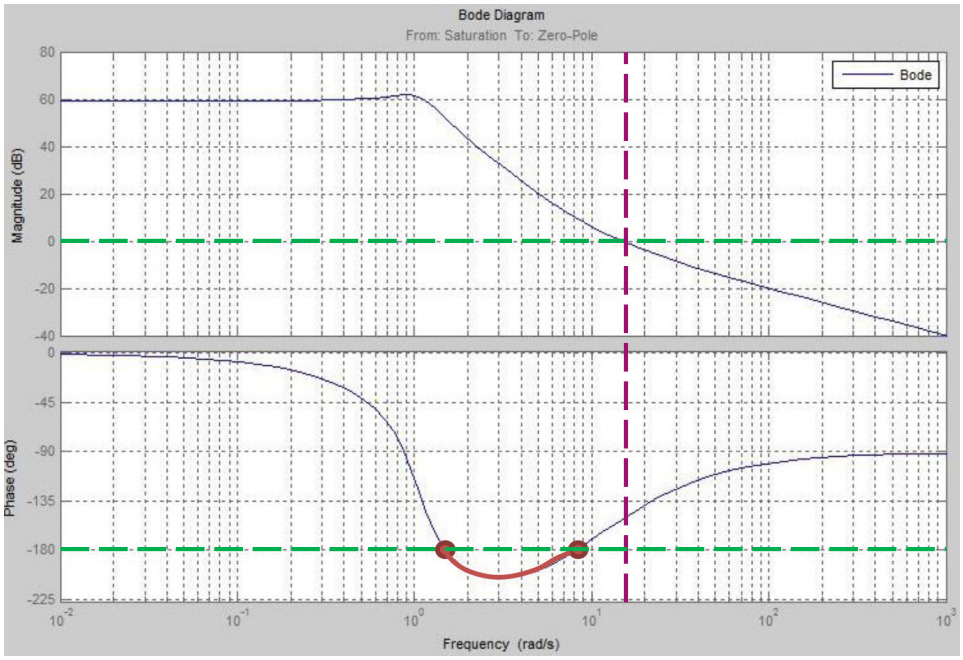


Figure 3. Bode plot of a conditionally stable system.

Matlab functions:

rlocus(SYS) computes and plots the root locus of the single-input, single-output LTI model SYS.

bode(SYS) draws the Bode plot of the LTI model SYS (created with either TF, ZPK, or SS).

SYS = series(SYS1,SYS2) connects two LTI models SYS1 and SYS2 in series.

SYS = parallel(SYS1,SYS2) connects two LTI models SYS1 and SYS2 in parallel.

SYS = feedback(SYS1,SYS2,sign) creates a closed-loop feedback system SYS. The system SYS1 is placed at the direct branch while the system SYS2 is placed at the feedback branch. By default, sign = -1, that is, negative feedback is assumed. To apply positive feedback, we set sign = 1.

Exercise 4:

For part (4a) you are referred to "Example 2" in the reference [4][a] given in the "Textbooks" directory of the "Course materials" section. To understand the theoretical background behind the method used, you are referred to "Section 2. PD Compensator Design" on pages 6-10 in the reference [5][b].

It should be noted that there is a mistake in the reference [4]: On page 7B.8, line 6 from the bottom, it was written that "...the left of the pole with the smallest magnitude...". The correct one should be "...the left of the pole with the SECOND smallest magnitude...".

The following tutorials are also recommended for reading:

Introduction: Root Locus Controller Design

http://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlRootLocus

DC Motor Speed: Root Locus Controller Design

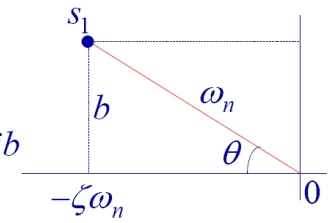
http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=ControlRootLocus

The main idea of a root locus design is to predict the closed-loop response using the root locus plot (created for the open-loop transfer function) which depicts possible closed-loop pole locations. Then by adding zeros and/or poles via the controller, the root locus can be modified in order to achieve a desired closed-loop response.

From the required design specifications (ξ, ω_n, t_s) you should first determine the desired locations of the dominant closed-loop poles by using the guide below.

$$\zeta\omega_n = \frac{1}{\tau} \quad \text{and} \quad \theta = \cos^{-1}\zeta$$

$$b = \zeta\omega_n \tan\theta \quad \text{and} \quad s_1 = -\zeta\omega_n + jb$$



As a result, you find the two dominant closed-loop poles, i.e. $s_{1,2} = -0.2 \pm j0.2$.

Then, with the Root-Locus Design GUI (**sisotool**) tool of Matlab Control System Toolbox, you should obtain the following results.

When G(s) has K = 1 then you should obtain the controller, PD = $0.0639 \times \frac{1 + 3.1s}{1}$.

In this case, one can see in Figures 4 and 5 that the controlled system meets all required design specifications (i.e. $\zeta = 0.707$, $\omega_n \approx 0.2829$, $t_s = 19.4 < 20$ seconds).

[a] Matlab Control System Toolbox - Root Locus Design GUI
[b] Root-Locus Design

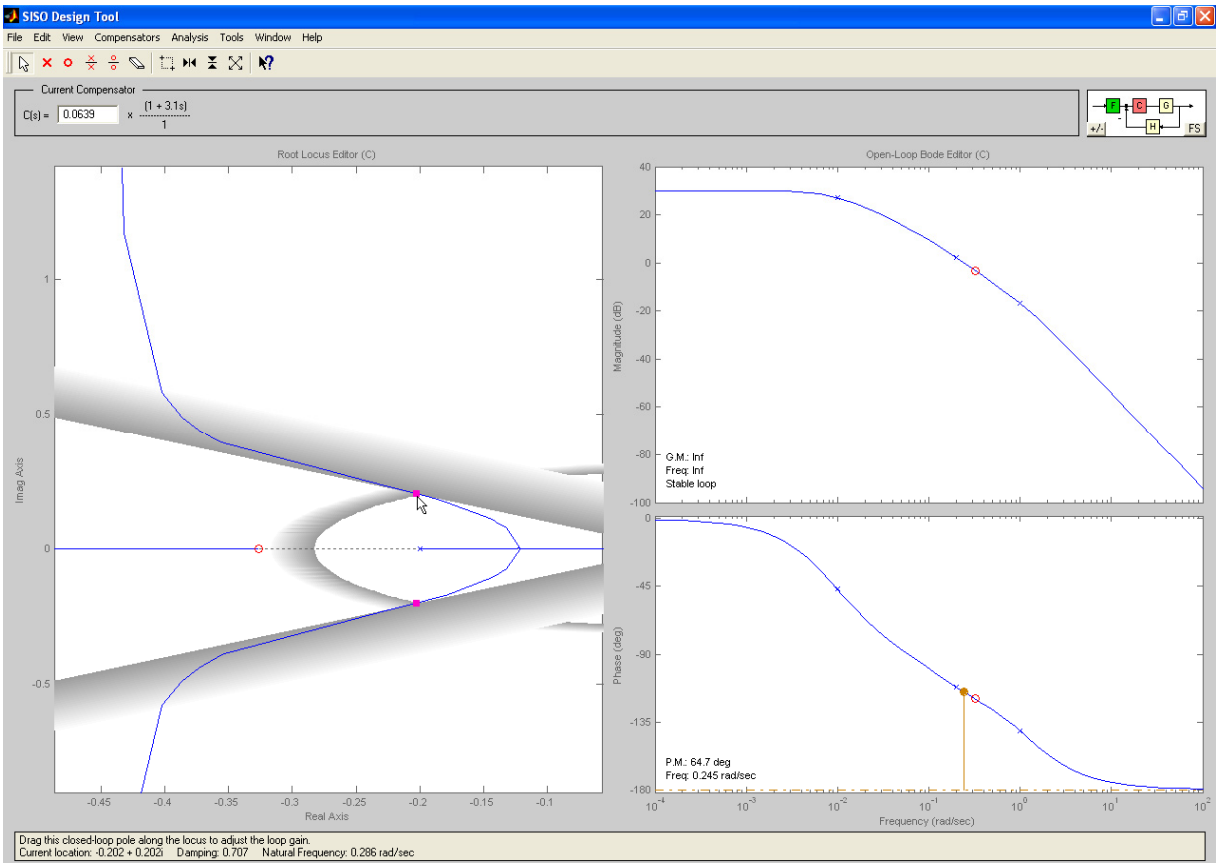


Figure 4.

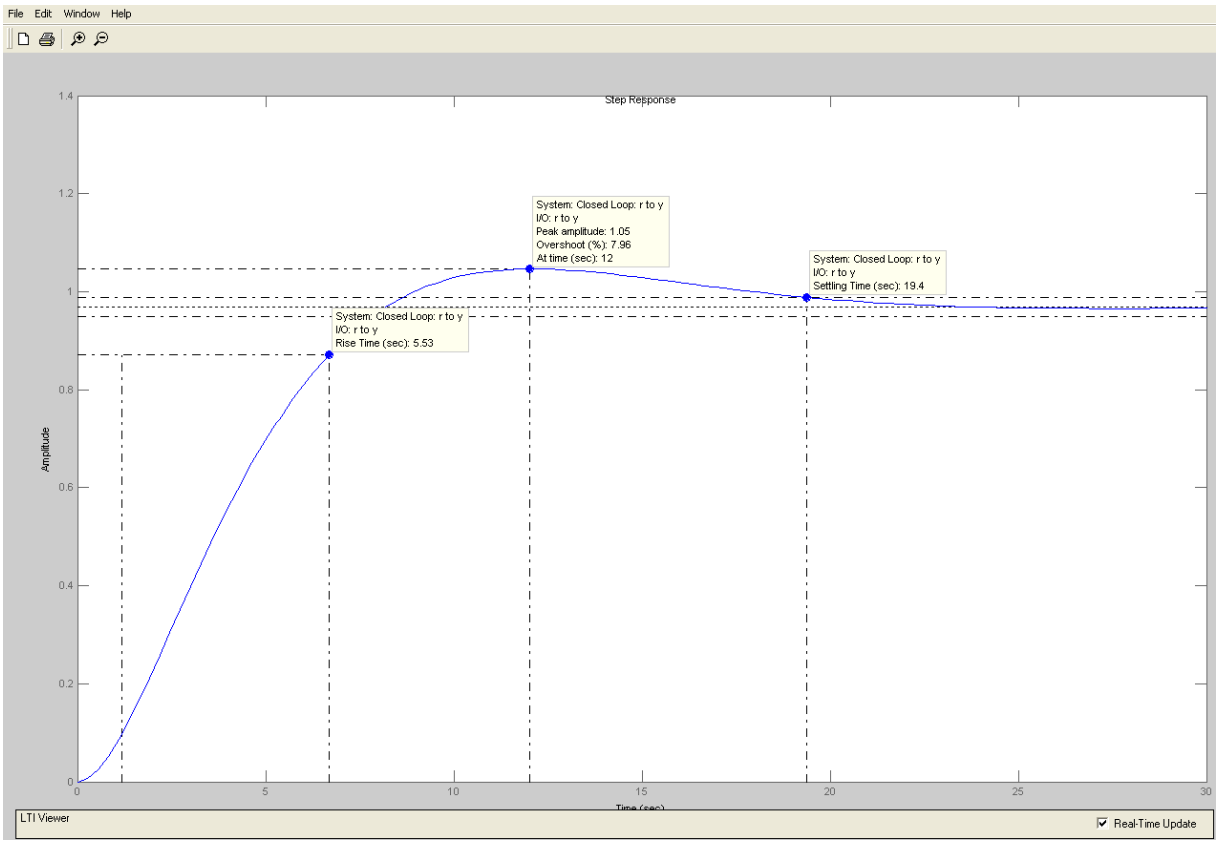


Figure 5.

When G(s) has K = 8 then you should obtain the controller, PD = $0.00811 \times \frac{1+3.1s}{1}$.

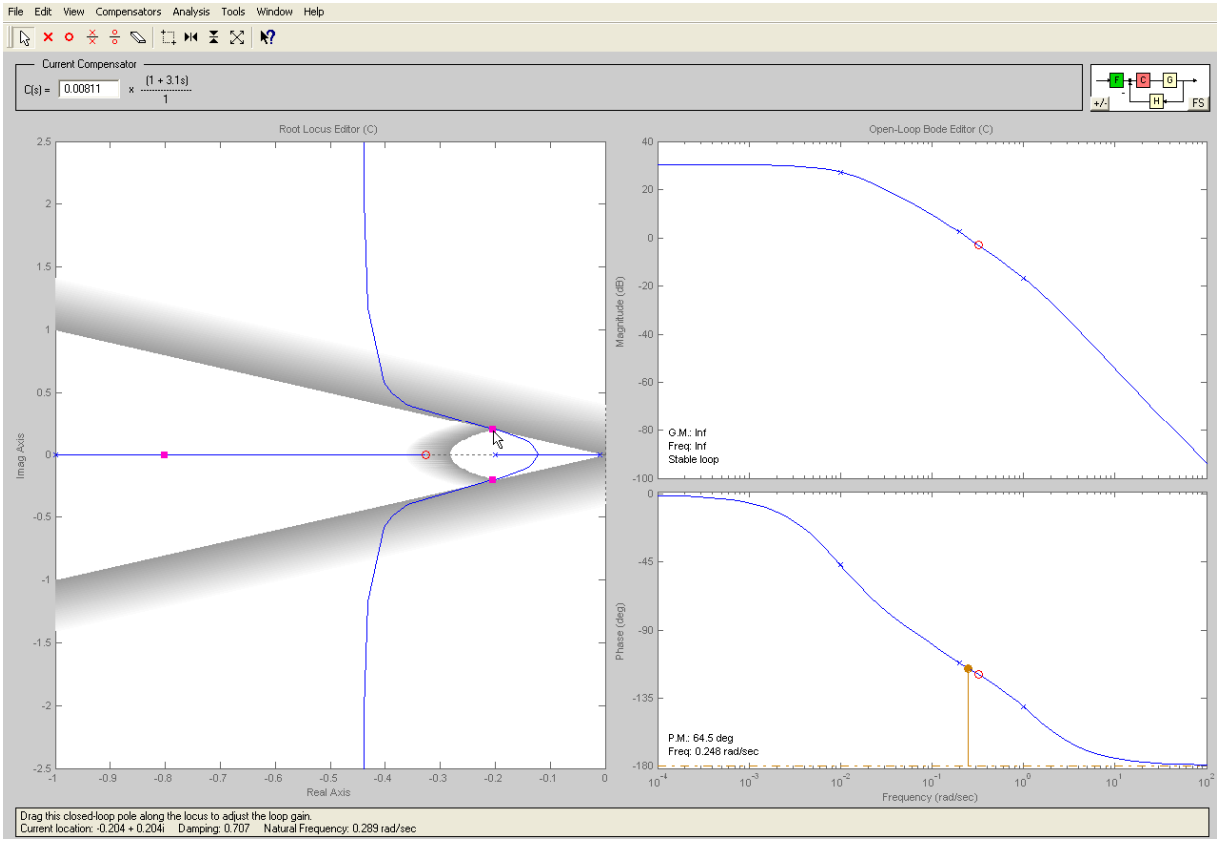


Figure 6.

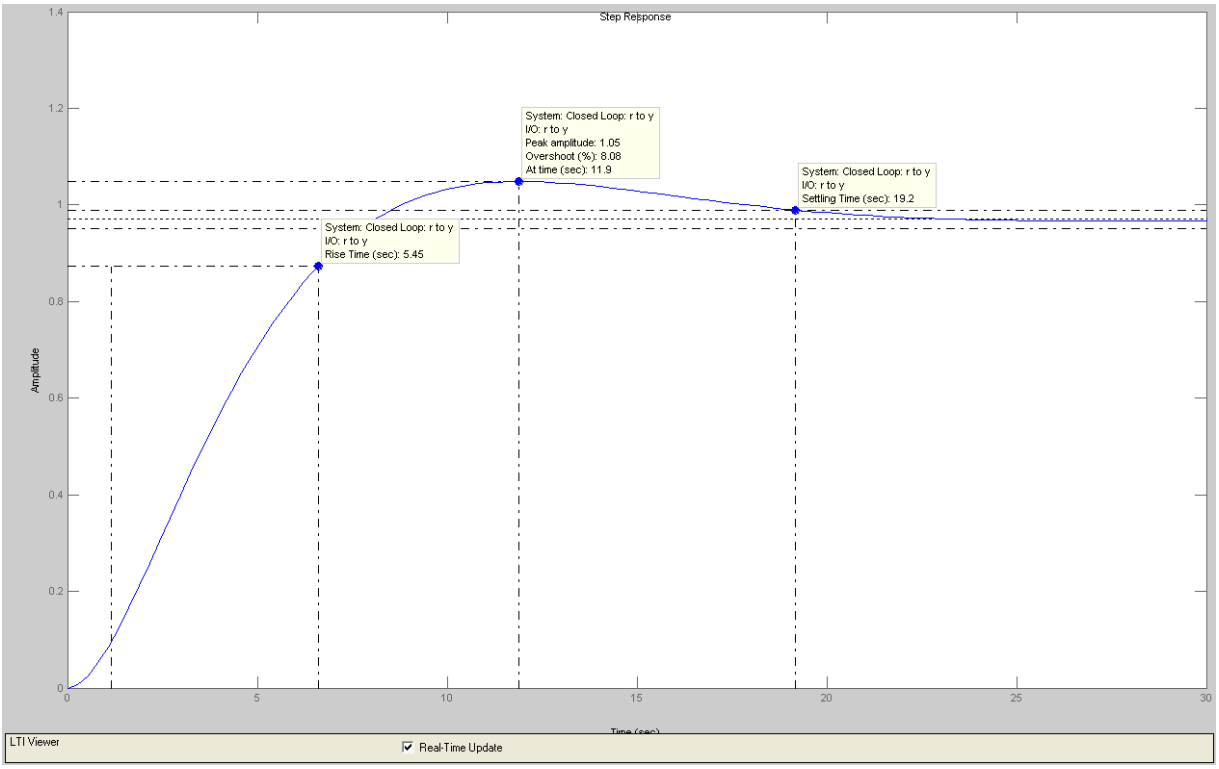


Figure 7.

The results – obtained in Figures 6 and 7 – also meet all design requirements.

In addition, one can see that $\frac{0.0639}{0.00811} \approx 8$. This means that it doesn't matter to put K = 1 or K = 8 in G(s) for the root locus design and analysis.

For part (4b):

Let consider a PID controller in the following form

$$PID(s) = K_1\left(1 + \frac{1}{T_i s} + T_d s\right)$$

The above equation can also be written as

$$PID(s) = \frac{K_1\left(T_i T_d s^2 + T_i s + 1\right)}{T_i s}$$

where K_1 is called the proportional control gain, T_i is called the integral action time and T_d is called the derivative action time.

The selection of PID controller parameters K_1, T_i and T_d can be performed using the classical control system design techniques. Ziegler and Nichols (1942) developed the so-called Continuous Cycling Method for this purpose. This is a closed-loop technique whereby, using only proportional control, the controller gain K_1 is increased until the system controlled output oscillates continually at constant amplitude, like a second-order system with no damping. This condition is referred to as marginal stability. This value of controller gain is called the ultimate (or critical) gain K_u, and the corresponding time period for one oscillation of the system controlled output is called the ultimate (or critical) period T_u. The controller parameters, as a function of K_u and T_u, are given in the following table.

Table 4.3 Ziegler–Nichols PID parameters using the Continuous Cycling Method

Controller type	K_1	T_i	T_d
P	$K_u/2$	–	–
PI	$K_u/2.2$	$T_u/1.2$	–
PID	$K_u/1.7$	$T_u/2$	$T_u/8$