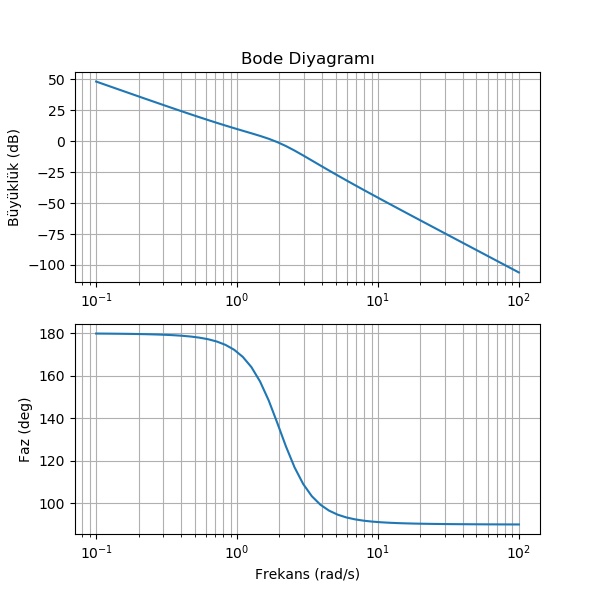
Soru 1:

Aşağıda verilen açık çevrim transfer fonksiyonlarının bode diyagramını çiziniz,

5 s + 10

-----------------------

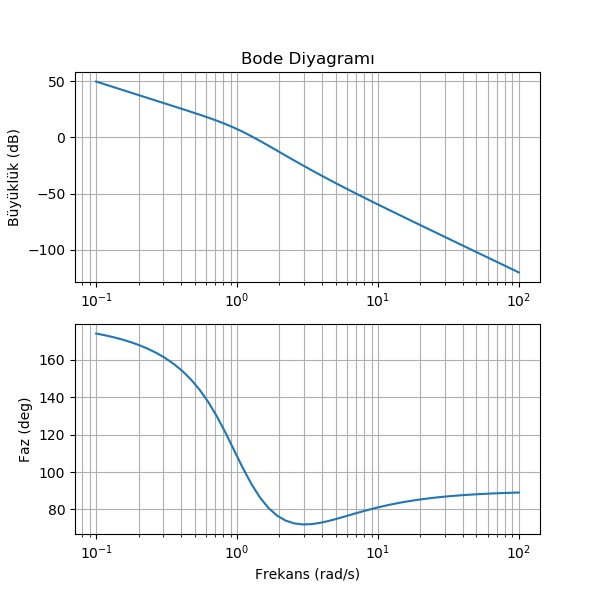
s^4 + 2.129 s^3 + 4 s^2



s + 3

---------------------

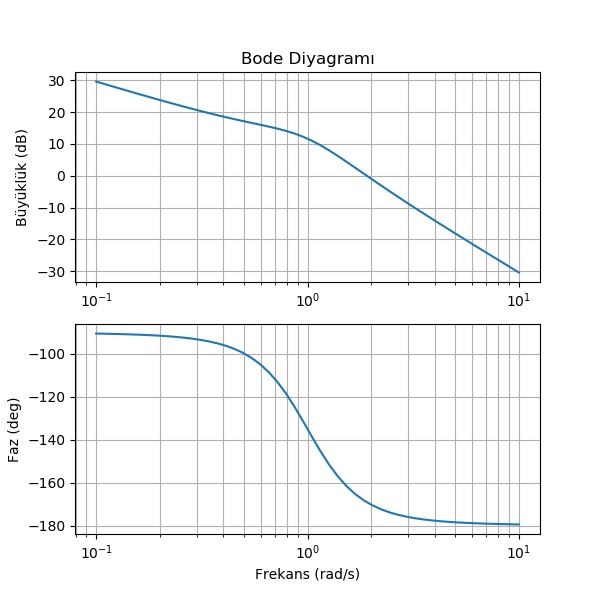
s^4 + 1.357 s^3 + s^2



3 s + 3

-------------------

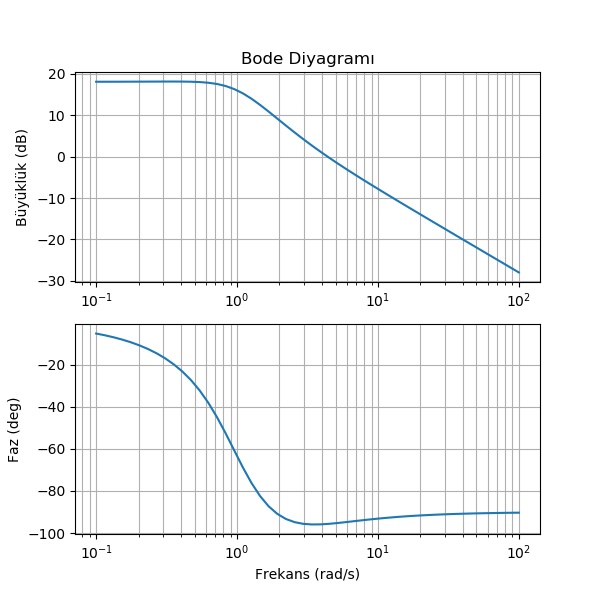
s^3 + 1.107 s^2 + s



4 s + 8

-----------------

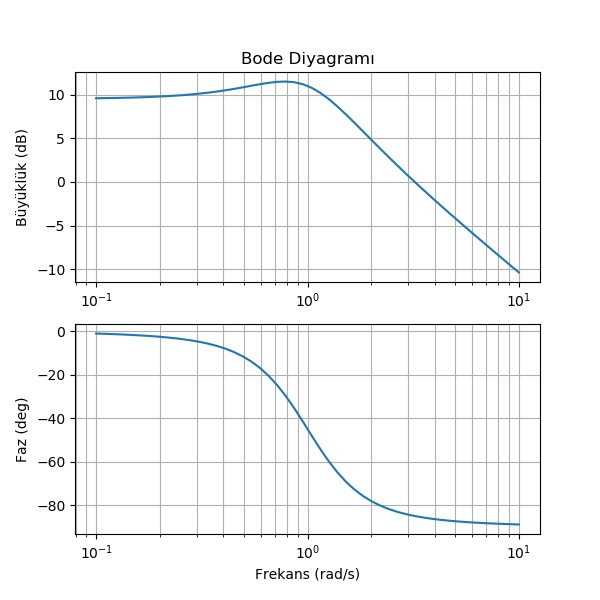
s^2 + 1.418 s + 1



3 s + 3

-------------------

s^3 + 1.192 s^2 + s



1. birim geri bildirimli, sistemin kararlılığı hakkında, kazanç ve faz marjlarını bularak bode diyagramı üzerinden yorum yapınız.
2. Rezonans değeri, rezonans frekansı, bant genişliği frekansını bulunuz

Bode diyagramlarından da görülebilmektedir ki, tüm sistemler için faz-frekans grafikleri hiçbir zaman 180° çizgisini kesmemektedir ya da sonsuzda kesmektedir. Dolayısıyla sistemlerin kazanç marjları yoktur ya da sonsuzdur. Faz marjları ve bode diyagramların çizimleri ise aşağıdaki python kodu ile oluşturulmuştur:

import numpy as np

import matplotlib.pyplot as plt

import scipy

import control

tf1 = control.tf([5, 10],[1, 2.129, 4, 0, 0])

tf2 = control.tf([1, 3],[1, 1.357, 1, 0, 0])

tf3 = control.tf([3, 3],[1, 1.107, 1, 0])

tf4 = control.tf([4, 8],[1, 1.418, 1])

tf5 = control.tf([3, 3],[1, 1.192, 1])

for i, tfx in enumerate([tf1, tf2, tf3, tf4, tf5]):

plt.figure(figsize=(6,6))

mag, phase, w = control.bode(tfx, dB=True)

mag = 20\*np.log10(mag)

phase = np.degrees(phase)

bandwidth = max(w[mag >= -3])

gm, pm, wg, wp = control.margin(tfx)

resonant\_peak = max(mag)

resonant\_freq = scipy.interpolate.interp1d(mag, w)(resonant\_peak)

print(f'tf{i+1} faz marjı={pm:6.2f}°, kazanç marjı={gm:.2f}rad/s', )

print(f'rezonans = {resonant\_peak:5.2f}dB, rezonans frekansı = {resonant\_freq:5.2f}rad/s')

print(f'bant genişliği = {bandwidth:.2f}rad/s', end='\n'+'-'\*55+'\n')

plt.subplot(2, 1, 1)

plt.semilogx(w, mag)

plt.ylabel('Büyüklük (dB)')

plt.title('Bode Diyagramı')

plt.grid(which='both')

plt.subplot(2, 1, 2)

plt.semilogx(w, phase)

plt.ylabel('Faz (deg)')

plt.xlabel('Frekans (rad/s)')

plt.grid(which='both')

plt.savefig(f'tf{i+1}.jpg')

plt.show()

bu kod çalıştırıldığında, yukarıdaki bode diyagramlarını kaydedip, aşağıdaki faz marjlarını konsol ekranında çıktı olarak yazmaktadır.

tf1 faz marjı=-38.88°, kazanç marjı=infrad/s

rezonans = 47.98dB, rezonans frekansı = 0.10rad/s

bant genişliği = 1.93rad/s

-------------------------------------------------------

tf2 faz marjı=-88.28°, kazanç marjı=infrad/s

rezonans = 49.55dB, rezonans frekansı = 0.10rad/s

bant genişliği = 1.26rad/s

-------------------------------------------------------

tf3 faz marjı= 10.98°, kazanç marjı=infrad/s

rezonans = 29.62dB, rezonans frekansı = 0.10rad/s

bant genişliği = 2.02rad/s

-------------------------------------------------------

tf4 faz marjı= 84.32°, kazanç marjı=infrad/s

rezonans = 18.12dB, rezonans frekansı = 0.36rad/s

bant genişliği = 5.18rad/s

-------------------------------------------------------

tf5 faz marjı= 95.03°, kazanç marjı=infrad/s

rezonans = 11.53dB, rezonans frekansı = 0.79rad/s

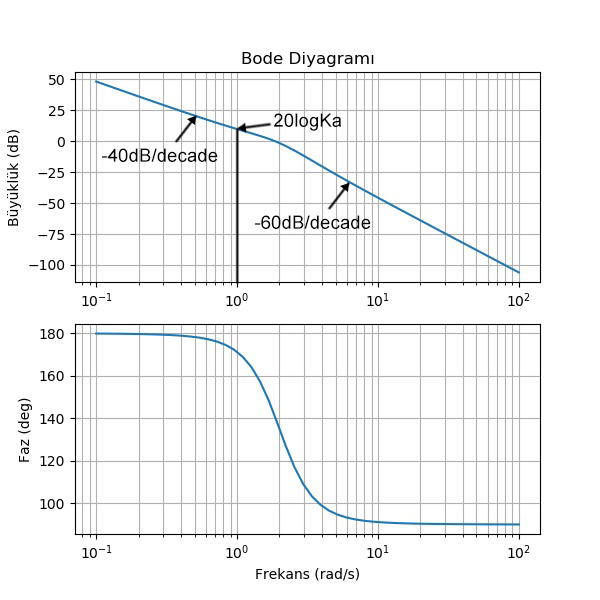
bant genişliği = 4.29rad/s

-------------------------------------------------------

Faz marjı, pozitif olduğu durumlarda kararsızlıktan ne kadar uzak olduğunu gösterir. Negatif olduğunda ise sistemin kararsız olduğunu ve kararlılıktan ne kadar uzak olduğunun göstergesidir. Sistem 1 ve 2 kararsız, 3,4 ve 5 kararlıdır.

1. Sistemlerin kalıcı durum hatalarını bode diyagramı üzerinden adım, rampa ve ivme giriş için belirleyiniz.

Transfer Fonksiyonu 1’de, w=1’de -40dB/decade değişim olduğu için tip numarası 2’dir. Sistemin adım ve rampa giriş kalıcı durum hatası yoktur. İvme girişteki kalıcı durum hatasını hesaplamak için ise önce Ka hesaplanmalıdır.



Bu işlemleri otomatik olarak aşağıdaki kod ile yapabiliriz.

#küçük frekanslardaki davranışı, w=1'e exterpolate etmek için daha önce

#sayısal yöntemlerde yazdığımız interpolasyon

#fonksiyonunu, sınırları umursamadan kullanırsak eksterpolasyon yapabiliriz.

#bu bize asimptotun w=1'i kestiği noktanın desibelini verir.

logkx20 = lagrange\_interpolation(0, np.log10(w[w<0.3]), mag[w<0.3],n=1, ignorerange=True)

kx = 10\*\*(logkx20/20)

print(f'tf{i+1}', '20logkx=',logkx20,'kx =', kx)

tf1 20logkx= 8.431181041477089 kx = 2.6397272280769917

tf2 20logkx= 9.706527763046381 kx = 3.057217860855301

tf3 20logkx= 11.386378284194052 kx = 3.7095302265067587

tf4 20logkx= 18.19215800854758 kx = 8.120969893019081

tf5 20logkx= 11.147108455139232 kx = 3.6087385817330175

|  |  |  |  |
| --- | --- | --- | --- |
| Transfer Fonksiyonu | Tip Numarası |  | Kalıcı durum hatası |
| Tf1 | 2 |  | 0, 0, 0.38 |
| Tf2 | 2 |  | 0, 0, 0.33 |
| Tf3 | 1 |  | 0, 0.27, |
| Tf4 | 0 |  | 0.11, , |
| Tf5 | 0 |  | 0.22, , |

Soru 2:

Aşağıda Ekte grafiklerin deneysel olarak elde edildiğini varsayarak

1. Sistemlerin Tip numarasını belirleyin

Figür 1 -20dB/decade düşüşe sahip olduğundan tip numarası 1.

Figür 2 -20dB/decade düşüşe sahip olduğundan tip numarası 1.

Figür 3 -20dB/decade düşüşe sahip olduğundan tip numarası 1.

Figür 4 düşük frekanslarda düşüşe sahip olmadığı için tip 0.

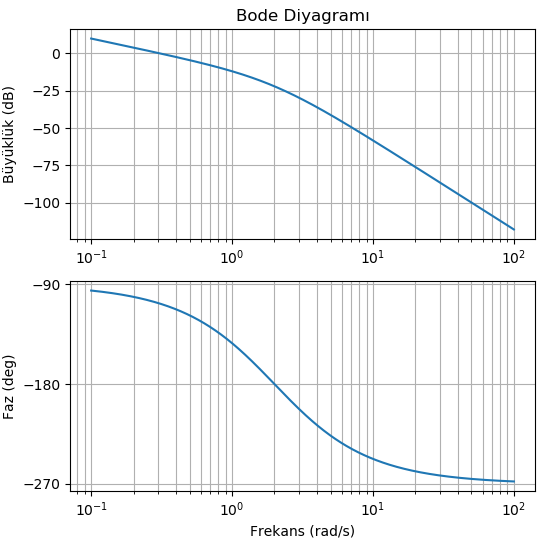
Figür 5 -20dB/decade düşüşe sahip olduğundan tip numarası 1.

1. Olası sistemi tahmin edin, ilgili frekansları ve kazanç değerlerini bulun
2. Bulduğunuz değerlere göre sistemlerin transfer fonksiyonunu bulunuz.

İlk grafik için:

Transfer fonksiyonu

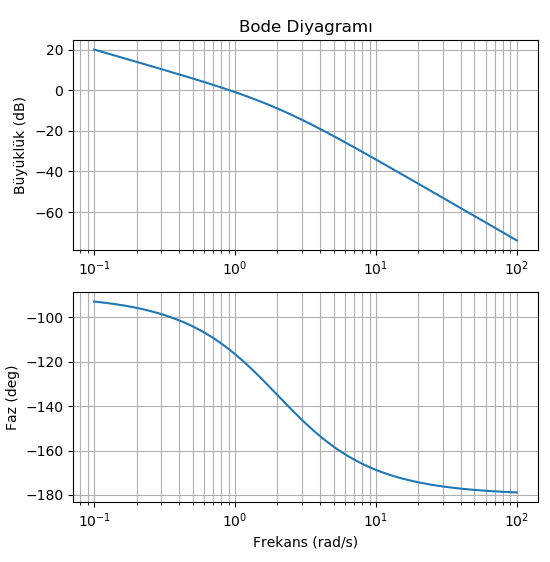
Pythonda çizerek kontrol edildiğinde;



İkinci grafik için:

Transfer fonksiyonu

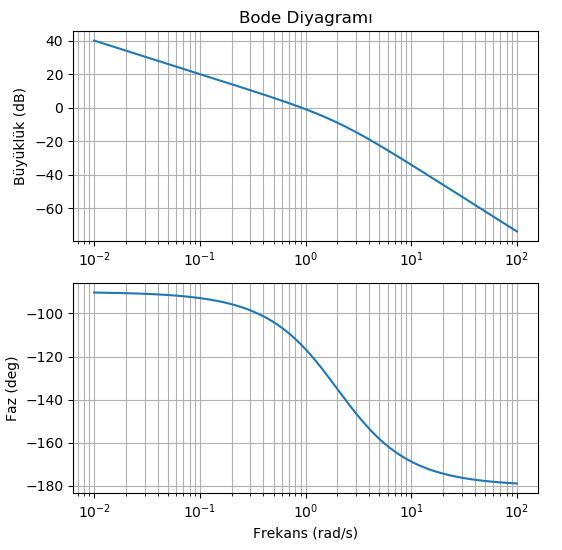
Pythonda çizerek kontrol edildiğinde;



Üçüncü grafik için:

Transfer fonksiyonu

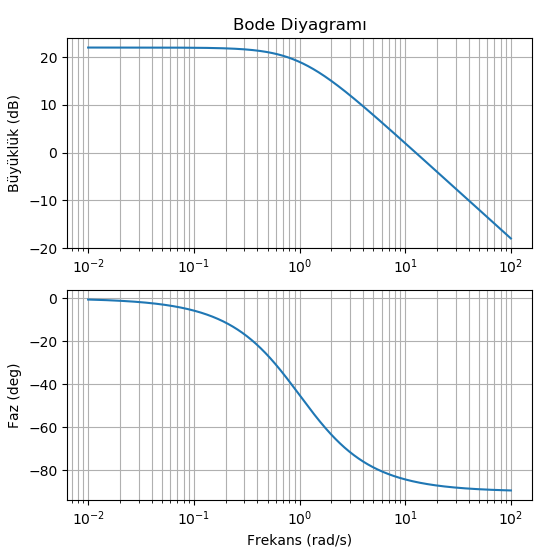
Pythonda çizerek kontrol edildiğinde;



Dördüncü grafik için:

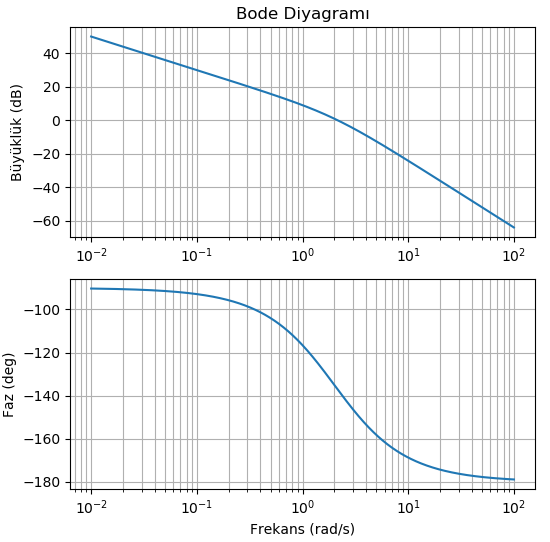
Transfer fonksiyonu

Pythonda çizerek kontrol edildiğinde;



Beşinci grafik için:

Transfer fonksiyonu



Soru 3:

Aşağıda verilen sistemlerin kararlılığını Nyquist diyagramını çizerek, diyagram üzerinden nyquist kriterlerini kullanarak açıklayınız. Kararınızın doğruluğunu bildiğiniz diğer kararlılık analiz yöntemlerinden birini kullanarak doğrulayınız.

Burada

sağ-yarıdüzleminde in sıfırlarının sayısı

noktasının, saat yönünde çevrelenme sayısı

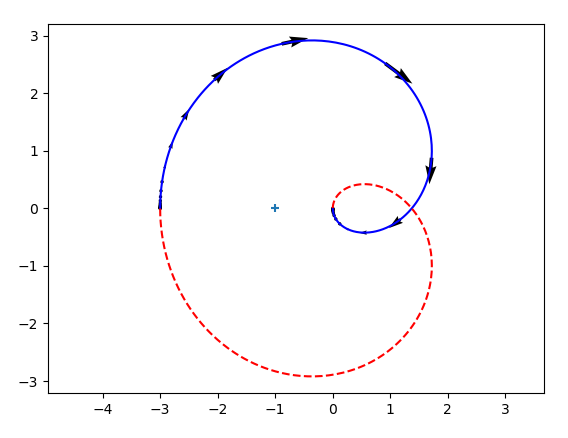
sağ-yarı düzleminde 'in kutuplarının sayısı

Sistemlerin kararlı olduklarını söyleyebilmek için Z’nin sıfır olması gerekmektedir.

4 s - 12

-----------------

s^2 + 2.914 s + 4



Sistemin açık çevrim transfer fonksiyonunun pozitif kutbu yoktur. Ancak -1+0j noktasını sarmalamıştır. Birim geri bildirim durumunda sistem kararsız olacaktır.

Kanıtlamak istersek;

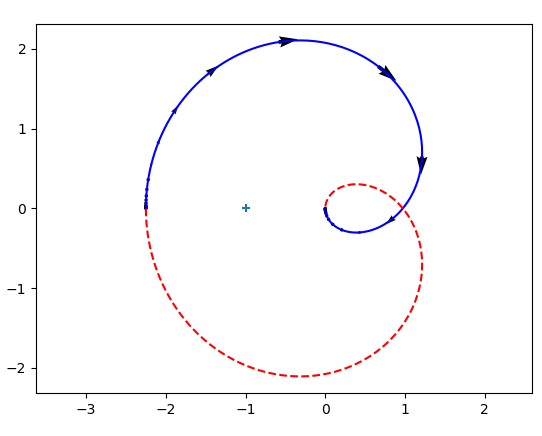
Negatif birim geri bildirim yapılmış kapalı çevrim transfer fonksiyonu:

KÇTF’nin paydası hurwitz kuralına uymadığı için kararsız olduğunu söyleyebiliriz.

3 s - 9

-----------------

s^2 + 3.086 s + 4



Sistemin açık çevrim transfer fonksiyonunun pozitif kutbu yoktur. Ancak -1+0j noktasını sarmalamıştır. Birim geri bildirim durumunda sistem kararsız olacaktır.

Kanıtlamak istersek;

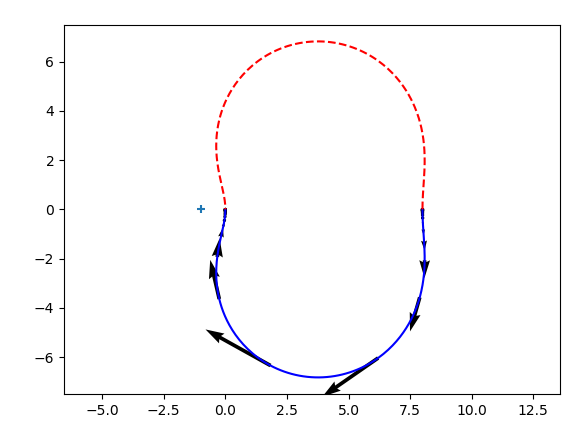
Negatif birim geri bildirim yapılmış kapalı çevrim transfer fonksiyonu:

KÇTF’nin paydası hurwitz kuralına uymadığı için kararsız olduğunu söyleyebiliriz.

4 s + 8

-----------------

s^2 + 1.176 s + 1



Sistemin açık çevrim transfer fonksiyonunun pozitif kutbu yoktur. Ayrıca Nyquist eğrisi -1+0j noktasını çevrelememiştir. Z=P+N şartları sağlanmaktadır ve Z=0 olduğundan sistem kararlıdır.

Kanıtlamak istersek;

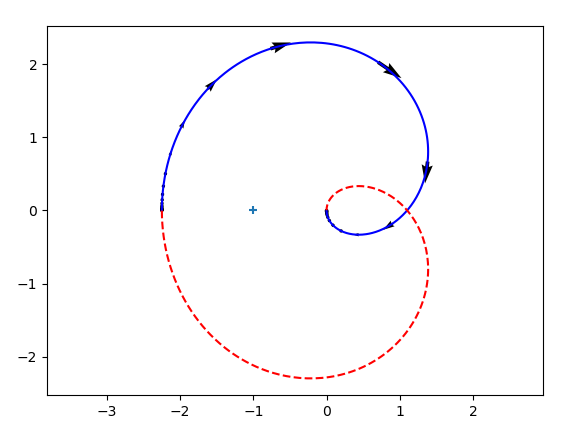
Negatif birim geri bildirim yapılmış kapalı çevrim transfer fonksiyonu:

KÇTF’nin paydası hurwitz kuralına uyduğu ve 2. dereceden bir polinom olduğu için kesinlikle kararlı olduğunu söyleyebiliriz.

3 s - 9

-----------------

s^2 + 2.724 s + 4



Sistemin açık çevrim transfer fonksiyonunun pozitif kutbu yoktur. Ancak -1+0j noktasını sarmalamıştır. Birim geri bildirim durumunda sistem kararsız olacaktır.

Kanıtlamak istersek;

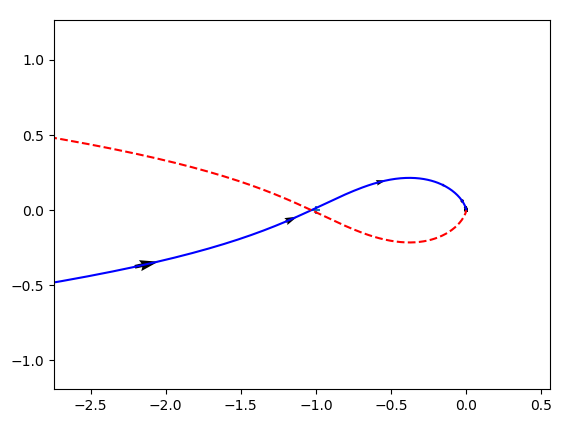
Negatif birim geri bildirim yapılmış kapalı çevrim transfer fonksiyonu:

KÇTF’nin paydası hurwitz kuralına uymadığı için kararsız olduğunu söyleyebiliriz.

4 s + 4

-----------------------

s^4 + 2.346 s^3 + 4 s^2

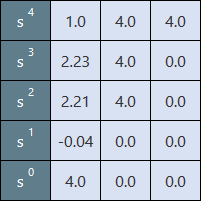


Sistemin açık çevrim transfer fonksiyonunun pozitif kutbu yoktur. Ancak -1+0j noktasını sarmalamıştır. Birim geri bildirim durumunda sistem kararsız olacaktır.

Kanıtlamak istersek;

Negatif birim geri bildirim yapılmış kapalı çevrim transfer fonksiyonu:

KÇTF’nin paydası hurwitz kuralına uymaktadır ancak kararlı olup olmadığını söyleyebilmek için routh tablosunu oluşturmamız gerekir.



İlk sütunda 2 kez işaret değişmiştir. Dolayısıyla sistemin 2 adet kararsız kökü vardır. Sistem kararsızdır.

Soru 4:

Aşağıdaki grafiklerin Nyquist grafiklerini çizerek kararlılk için K’nın kritik değerini belirleyiniz.

Bu transfer fonksiyonlarının nyquist grafiklerini çizmek için özel bir nyquist fonksiyonu pythonda yazılmıştır.

def nyquist(tf, w=None, plot=True, \*args, \*\*kwargs):

"""

Ex:

tf = lambda s: (5\*s+10)/(s\*\*4+2.129\*s\*\*3+4\*s\*\*2+5\*s+10)

"""

if w is None:

w = np.logspace(-3, 3, 1000)

gjw = np.array([tf(1j\*wx, \*args, \*\*kwargs) for wx in w])

if plot:

plt.axis('equal')

plt.quiver(gjw.real[::30], gjw.imag[::30], gjw.real[1::30]-gjw.real[:-1:30],

gjw.imag[1::30]-gjw.imag[:-1:30], scale=2)

plt.plot(gjw.real, gjw.imag, 'b')

plt.plot(gjw.real, -gjw.imag, 'r--')

plt.scatter([-1], [0], marker='+')

return gjw.real, gjw.imag

K exp(-0.55 s)

--------------

s - 3.0

Sistemin sağ yarı düzlemde bir adet kutubu vardır(P=1). Dolayısıyla sistemin eşitliğini sağlaması ve kararlı olması için (Z=0), nyquist eğrisi -1+0j noktasının etrafını 1 kez sarmalamalıdır (N=-1).

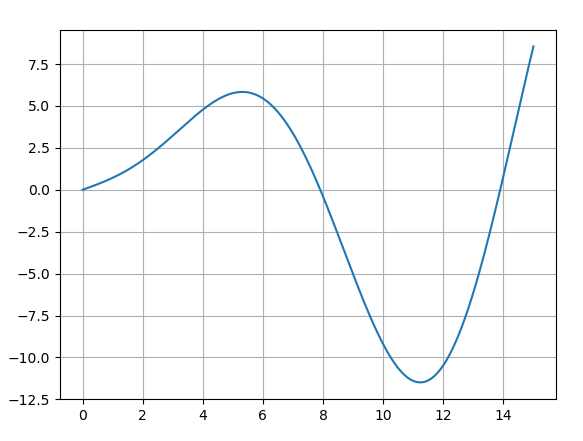
Bu eşitliğin çözümlerinden bir tanesi: ’dır ve bu durumda nyquist eğrisi K=3 için doğrusunun üzerinden geçer.(figür 4.1.a)

Dolayısıyla,

K=3’ten büyük değerlerde nyquist eğrisi -1+0j’yi sarmalar ve sistemi kararlı kılar. Ancak belli bir K değerinden sonra birden fazla kez sarmalamaya da başlayacaktır. Bu da sistemi kararsız kılacaktır. Bu K değerinin hesaplanması için (1) eşitliğinin diğer köklerini bulalım.

Bunu ‘Düzeltilmiş Sekant’ yöntemi ile çözmek için aşağıdaki python scriptini çalıştıralım.

Düzeltilmiş sekant yöntemini kullanmadan önce başlangıç noktası seçmek için grafiği çizdirebiliriz.



Şekil f(w)= -wcos(0.55w)+3 sin(0.55w)

8 civarında bir kök olduğu görüldüğünden 8 etrafında kökü arayalım.

def corrected\_secant(f, x0, d=0.001, e=0.1, max\_iter=10, explain=False): # Düzeltilmiş Sekant

fprime = lambda x0: (f(x0+d)-f(x0))/d

for i in range(max\_iter):

x1 = x0 - f(x0)/fprime(x0)

ey = abs((x1-x0)/x1)\*100 # neden sonraki terim paydada?

x0 = x1

if ey < e:

break

if i == max\_iter-1:

print('maximum iterations reached, maybe choose another x0 or increase max\_iter')

if explain:

return f'{i+1} iterasyonda, εy=%{e}, kök={x1}'

return x1

f = lambda w: -w\*cos(0.55\*w)+3\*sin(0.55\*w)

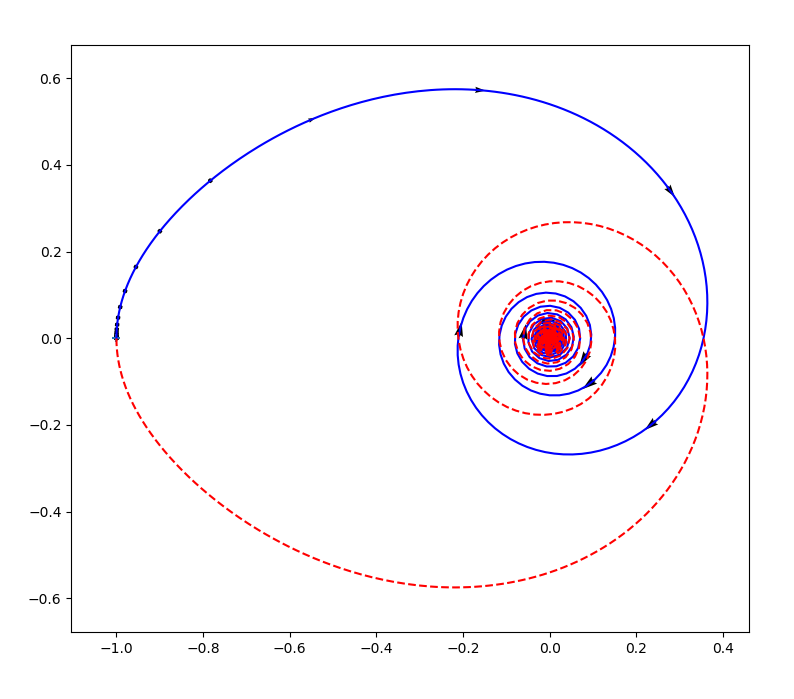
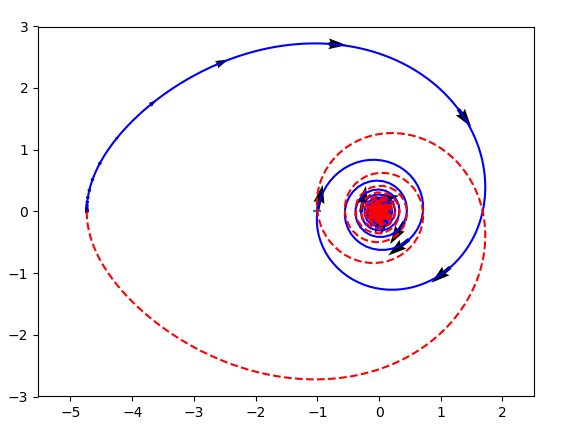
print(corrected\_secant(f, 8))

7.908788147912582

Burada w=7.9 yazıp ’ye eşitlersek, nyquist grafiğinin bu noktadan geçmesi için gereken kritik K değerini elde ederiz.

bu noktadan sonra nyquist eğrisi, -1+0j noktasının etrafından iki kez geçmeye başlar ve eşitliğini bozar. Sistemi kararsız yapar.

Bu da gösterir ki, için sistem kararlıdır.



Figür 4.1.b K=14.21

Figür 4.1.a K=3

K exp(-0.57 s)

--------------

s - 1.0

sistemin reel pozitif kutbu vardır. Nyquist eğrisi -1+0j noktasını 1 kez sarmalamalıdır.

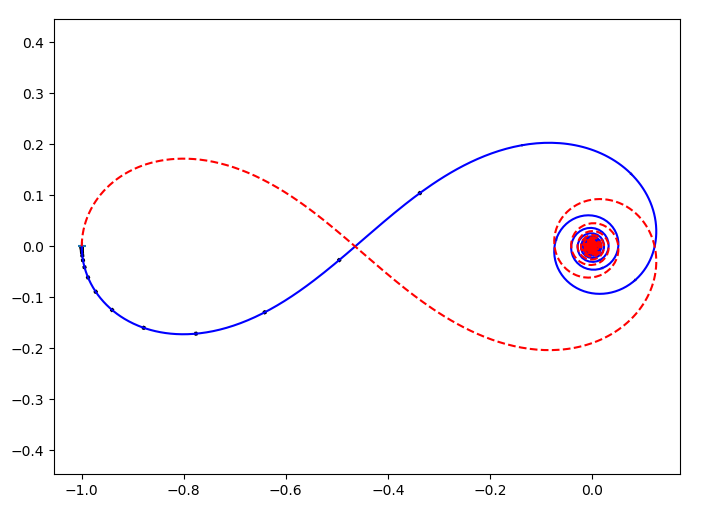
Yukarıdaki örneğe benzer şekilde, w=0 için K=1’de kritik kazanç değeri olduğunu söyleyebiliriz.

K>1 için nyquist eğrisi -1+0j noktasını sarmalar ve sistem kararlı olur.

tf7 = lambda s, K=1: K\*(np.exp(-0.57\*s))/(s-1)

nyquist(tf7, K=1)

plt.show()



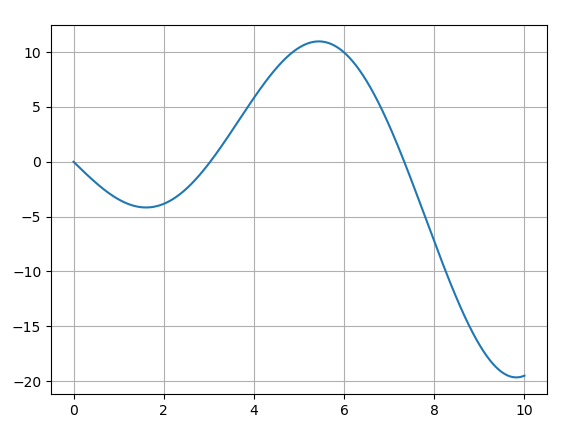
Figür 4.2 K=1

K exp(-0.67 s)

--------------

2.0 s + 3.0

Karmaşık kısmın grafiğini çizersek;



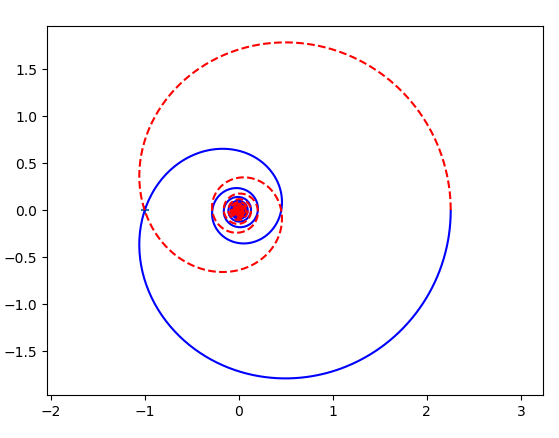
Düzeltilmiş sekant yöntemiyle

>>> f = lambda w:-3\*sin(0.67\*w)-2\*w\*cos(0.67\*w)

>>> corrected\_secant(f, 3)

3.030468976227838

Kritik değerdir.

K>6.76 için nyquist eğrisi

-1+0j noktasını sarmalar,

sistem kararsızlaşır.

**Ekler:**

Soru 2’nin grafikleri aşağıdadır.

