MODELADO, SIMULACION Y CONTROL DE SISTEMAS DINAMICOS

P.F.PULESTON y F.VALENCIAGA

Nota: Este apunte tiene por objetivo principal introducir al modelado, simulación y control de sistemas dinámicos empleando Matlab. Gran parte del material aquí presentado esta basado en los Control Tutorial for Matlab de la ***Universidad de Michigan****, Demos y Manuales de Matlab. Abundante información adicional y soluciones a muchos otros problemas vinculados al control de sistemas, procesamiento de señal y calculo matricial en general, puede encontrarse en dicha bibliografía.*

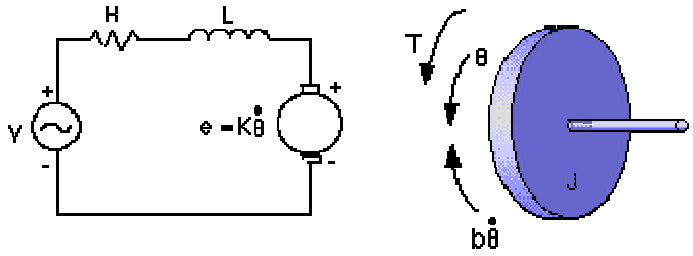
Parte II. Ejemplo de Aplicación Al Control De Sistemas

Sección 1: Modelado de un Motor de CC

Ecuaciones físicas del sistema
Requerimientos de Diseño
Representación en MATLAB y respuesta de lazo cerrado

Ecuaciones físicas del sistema

Un actuador mecánico muy difundido es el motor de CC. Provee directamente movimiento rotacional y, adecuadamente acondicionado, movimiento traslacional. El circuito eléctrico de armadura y el diagrama mecánico rotacional, se muestran en la figura:



Para el ejemplo se consideraron los siguientes parámetros:

* momento de inercia del sistema (J) = 0.01 kg.m^2/s^2

* coeficiente de roce (b) = 0.1 Nms

* constante de fuerza electromotriz (K=Ke=Kt) = 0.01 Nm/Amp

* resistencia de armadura (R) = 1 ohm

* inductancia de armadura (L) = 0.5 H

* entrada (V): Fuente de Tensión

* posición del eje: Θ

* Se supone rotor y eje rígidos.

La cupla (*T*) está relacionada con la corriente de armadura y la fem (*e*) con la velocidad de rotación, según las ecuaciones:

$$T = K_t \cdot i$$

$$e = K_e \cdot \dot{\theta}$$

siendo ambas constantes iguales (K_t=K_e=K)

En base a la ley de Newton y la ley de Kirchoff, resultan las siguientes ecuaciones diferenciales que describen la dinámica del sistema:

$$J \cdot \ddot{\theta} + b \cdot \dot{\theta} = K \cdot i$$

$$L \cdot \frac{di}{dt} + R \cdot i = V - K \cdot \dot{\theta}$$

1. Función de Transferencia

Aplicando la Transformada de Laplace y haciendo cero las condiciones iniciales, las ecuaciones del sistema quedan expresadas en el dominio de s:

$$s \cdot (J \cdot s + b) \cdot \Theta(s) = K \cdot I(s)$$

$$(L \cdot s + R) \cdot I(s) = V - K \cdot s \cdot \Theta(s)$$

Eliminando I(s) se obtiene la transferencia entre la entrada de tensión de armadura V y la velocidad de rotación $\dot{\Theta}$ como salida:

$$\frac{\dot{\Theta}}{V} = \frac{K}{(J \cdot s + b) \cdot (L \cdot s + R) + K^2}$$

2. Espacio de Estados

La descripción del sistema de estados en el dominio temporal puede obtenerse definiendo las variables físicas velocidad de rotación $\dot{\theta}(t)$ y corriente de armadura *i(t)*, como variables de estado, la tensión de armadura *v(t)* como entrada y la velocidad de rotación como salida:

$$\frac{d}{dt}\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K}{J} \\ -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} \cdot v$$

Requerimientos de Diseño

El motor sin compensar puede rotar solamente a 0,1rad/s con una entrada de 1 Volt (ver simulación de la planta a lazo abierto). Uno de los requerimientos es que en estado estacionario presente un error respecto de la velocidad deseada menor que el 1%. Dinámicamente se espera un tiempo de establecimiento de 2 seg y un sobrepaso menor que el 5% para evitar daños en la máquina. Es decir:

- tiempo de establecimiento de 2 seg

- sobrepaso menor que el 5%
- Error de estado estacionario 1%

Representación en Matlab y respuesta de lazo cerrado

1. Función de Transferencia

Para representar la función de transferencia es necesario considerar los polinomios numerador y denominador:

$$num = K$$
$$den = (J \cdot s + b) \cdot (L \cdot s + R) + K^2$$

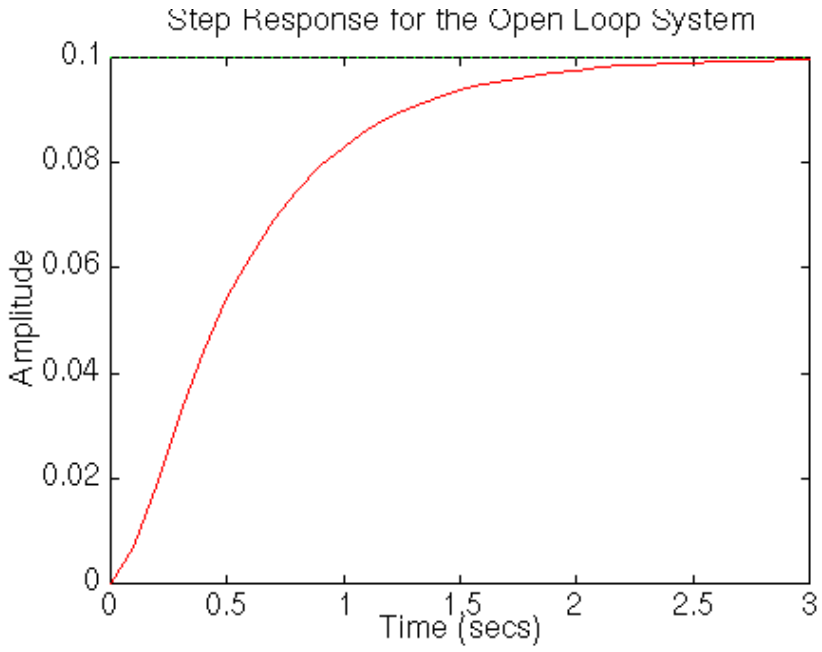
Creando un archivo *.m*:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
```

Para observar la respuesta al escalón del sistema a lazo abierto, basta con agregar al archivo los siguientes comandos y ejecutarlo:

```
Step(num,den,0:0.1:3)
title('Step Response for the Open Loop System')
```

Resultando:



De la figura se observa que a lazo abierto se obtiene una salida 10 veces más chica que la deseada (0,1 rad/s) y 3 seg de establecimiento, sin cumplir las especificaciones.

2. Espacio de Estados

Análogamente se puede crear el siguiente archivo *.m* y ejecutarlo, resultando la misma figura.

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J   K/J
   -K/L   -R/L];
B=[0
   1/L];
C=[1   0];
D=0;

step(A, B, C, D)
```

Nota: dependiendo la versión de Matlab la función step puede requerir de la creación de un sistema SYS*:*

```
SYS=SS(A,B,C,D);
step(SYS)
```

Sección 2: Método de Diseño de un PID para Control de Velocidad de un Motor CC

Control Proporcional
Control PID
Sintonía

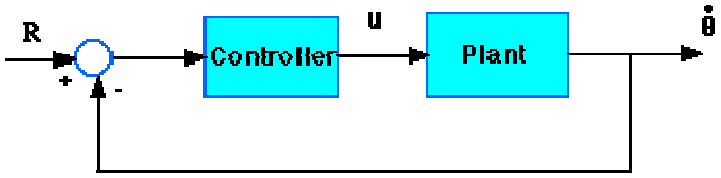
Recordando las ecuaciones dinámicas y transferencia del motor de CC:

$$s \cdot (J \cdot s + b) \cdot \Theta(s) = K \cdot I(s)$$

$$(L \cdot s + R) \cdot I(s) = V - K \cdot s \cdot \Theta(s)$$

$$\frac{\dot{\Theta}}{V} = \frac{K}{(J \cdot s + b) \cdot (L \cdot s + R) + K^2}$$

resulta el siguiente diagrama esquemático del sistema compensado:



El sistema a lazo cerrado debe cumplir las especificaciones establecidas anteriormente:

- tiempo de establecimiento de 2 seg
- sobrepaso menor que el 5%
- Error de estado estacionario 1%

El objetivo de esta sección será diseñar un controlador PID que permita verificar las especificaciones. Como primer paso crearemos el siguiente archivo *.m*:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
```

Correspondiente al sistema a lazo abierto y recordando que la transferencia del PID es de la forma:

$$K_P + \frac{K_I}{s} + K_D \cdot s = \frac{K_D \cdot s^2 + K_P \cdot s + K_I}{s}$$

Control Proporcional

Probemos primero un control proporcional con ganancia 100. Para ello hay que agregar los siguientes comandos al final del archivo *.m*:

```
Kp=100;
numa=Kp*num;
dena=den;
```

Para determinar la transferencia de lazo cerrado se puede utilizar el comando *cloop*:

```
[numac,denac]=cloop(numa,dena);
```

siendo los vectores *numac* y *denac* el numerador y denominador de la transferencia a lazo cerrado

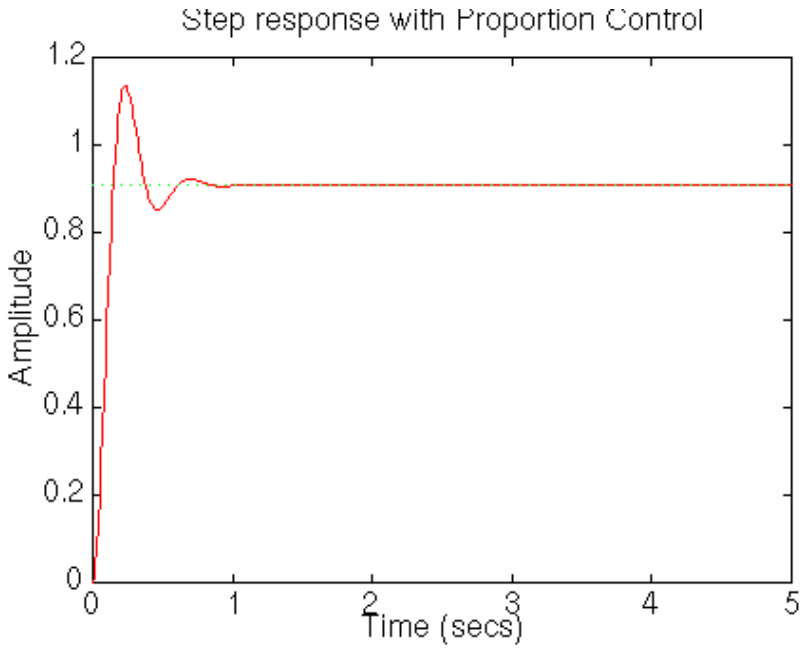
Nota: dependiendo la versión de Matlab la función cloop puede no existir y ser necesario recurrir al comando feedback:

```
SYS1=tf(numa,dema)
SYS2=tf(1,1)
SYSlc=feedback(SYS1,SYS2);
```

Para ver la respuesta de lazo cerrado al escalón de entrada basta con agregar al archivo *.m* los siguientes comandos:

```
t=0:0.01:5;
step(numac,denac,t)
title('Step response with Proportion Control')
```

Resultando:



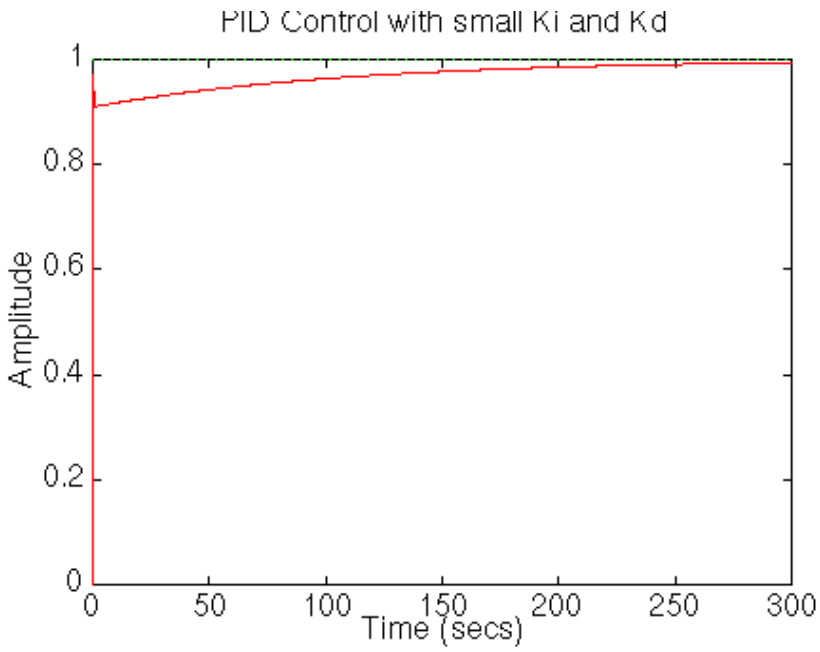
Control PID

Puede observarse que el error de estado estacionario y el sobrepaso son excesivos. Es sabido que agregando un término integral se elimina el error de estado estacionario al escalón, mientras que un término derivativo, adecuadamente sintonizado, puede reducir el sobrepaso. Probemos entonces con un controlador PID con bajas ganancias K_I y K_D. Para ello hay que modificar el archivo *.m* de la siguiente manera:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
```

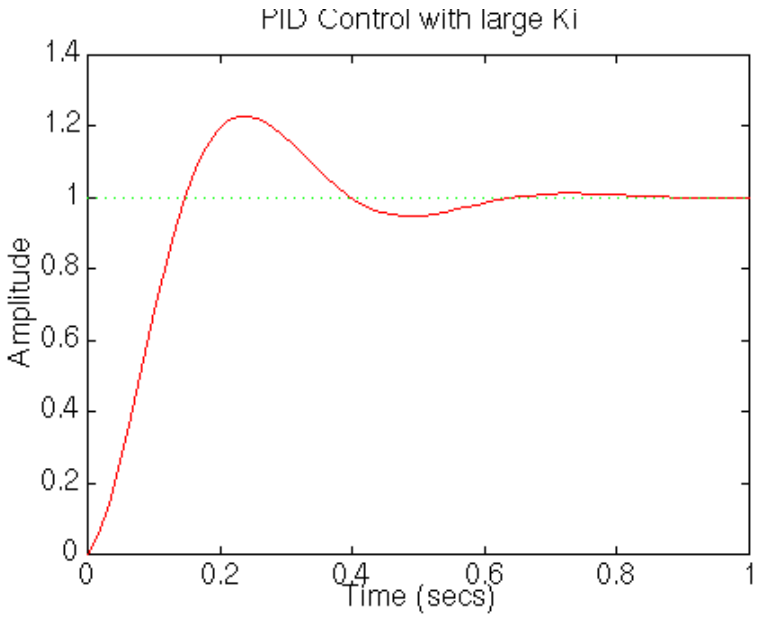
```
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];

Kp=100;
Ki=1;
Kd=1;

numc=[Kd, Kp, Ki];
denc=[1 0];
numa=conv(num,numc);
dena=conv(den,denc);
[numac,denac]=cloop(numa,dena);
step(numac,denac)
title('PID Control with small Ki and Kd')
```

resultando

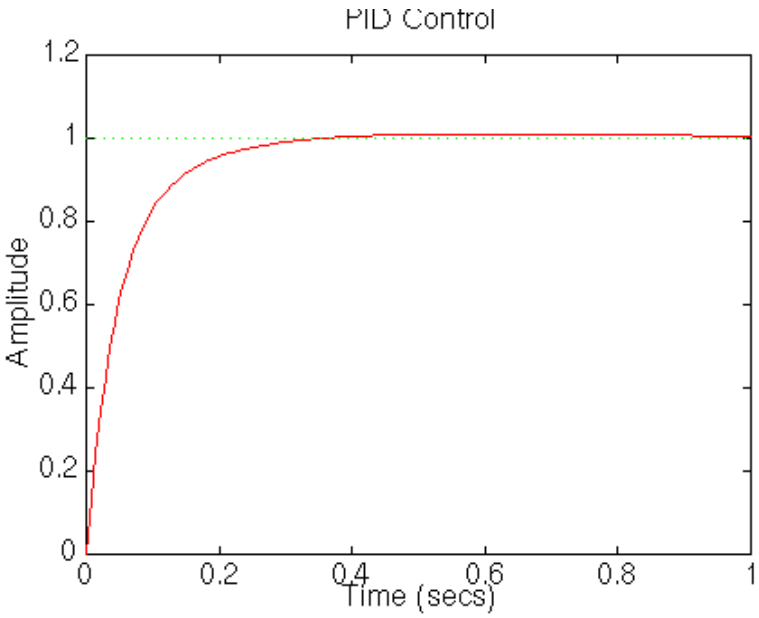


Sintonía

Ahora el tiempo de establecimiento es muy largo, por lo que incrementaremos K_I a 200. Modificando este valor en nuestro archivo y ejecutándolo se obtiene:



siendo una respuesta mucho más rápida que la anterior, pero a costa de aumentar el sobrepaso. En consecuencia, para reducirlo, incrementaremos el valor de K_D a 10 en el archivo *.m*:



Resultando:

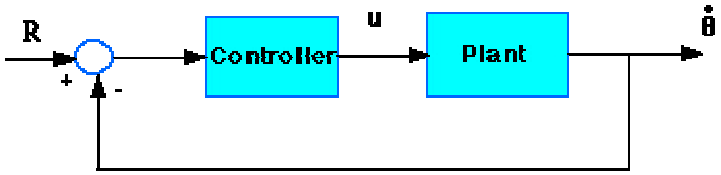
```
Kp=100,
Ki=200,
Kd=10,
```

una sintonía adecuada para cumplir los requerimientos de diseño.

Sección 3: Diseño del Compensador Empleando el Método de Lugar de Raíces

Graficación del Diagrama de Lugar de Raíces
Obtención de la ganancia usando el comando *rlocfind*
Compensador por atraso de fase
Graficación de la respuesta de lazo cerrado

En esta sección trataremos el diseño de un compensador para el motor de CC, basándonos en el método del lugar de raíces. Consideremos las mismas especificaciones de diseño. Como en la sección anterior el primer paso es crear un archivo *.m* que contenga el modelo de la planta a lazo abierto.



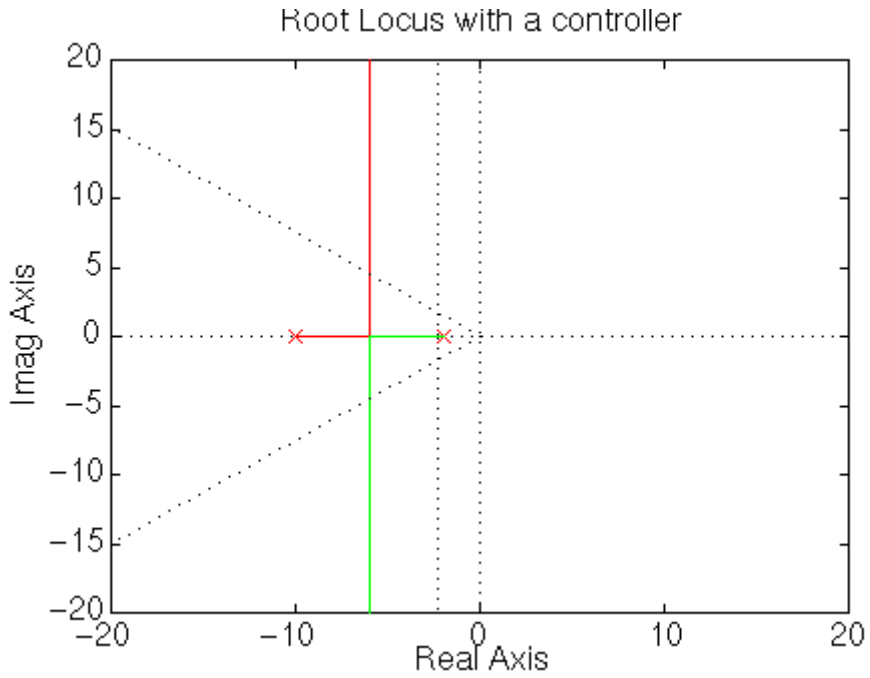

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
num=K;den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
```

Graficación del Diagrama de Lugar de Raíces

La idea principal del método es compensar agregando polos y/o ceros adecuadamente de modo tal que la respuesta a lazo cerrado se corresponda con la deseada. Como primer paso entonces, veremos el lugar de raíces de la planta sin compensar. Para ello basta con agregar las siguientes líneas de comando:

```
rlocus(num,den)
sgrid(.8,0)
title('Root Locus without a controller')
```

La función *sgrid*(ξ, ω_n) genera una grilla en el plano s, indicando los lugares de coeficiente de amortiguamiento ξ constante (por ejemplo, 0.8 corresponde a un sobrepaso de 5%) y los de frecuencia natural ω_n constante. El diagrama de lugar de raíces resultante es:



Obtención de la ganancia usando el comando *rlocfind*

Si se quisiera determinar la ganancia correspondiente a una dada posición de los polos de lazo cerrado del lugar de raíces, se debe recurrir al comando *rlocfind*. Agregándole al archivo *.m* el conjunto de comandos que se detallan a continuación, se obtendrá la ganancia y la respuesta de lazo cerrado al escalón:

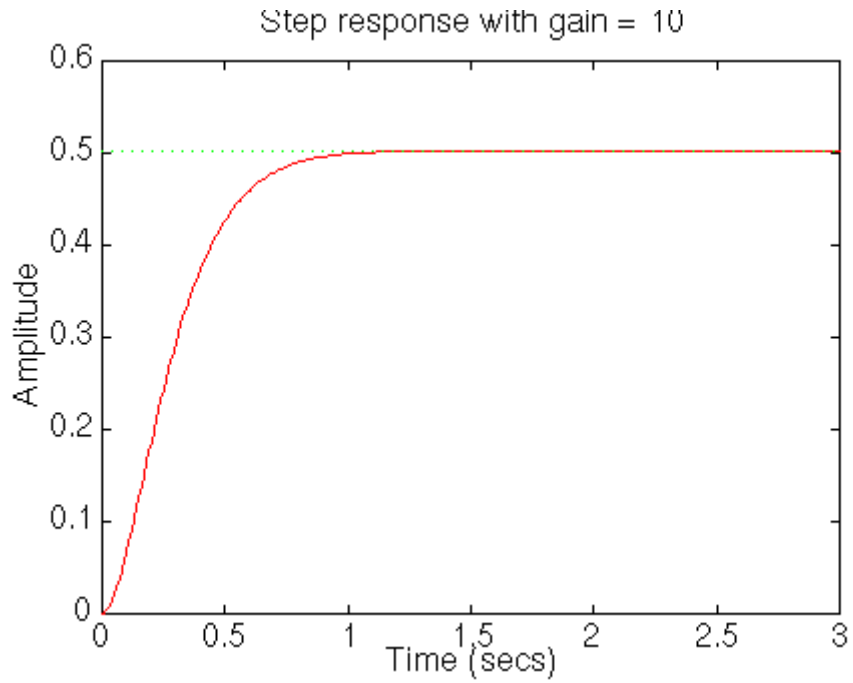
```
[k,poles] = rlocfind(num,den)
[numc,denc]=cloop(k*num,den,-1);
t=0:0.01:3;
step(numc,denc,t)
title('Step response with gain')
```

Seleccionando en la figura un punto, como por ejemplo -6+2.5i, Matlab retornará algo similar a:

```
selected_point =
  -5.9596 + 2.0513i
```

```
k =
    10.0934

poles =
  -6.0000 + 2.0511i
  -6.0000 - 2.0511i
```

y el gráfico:



Puede observarse que se cumple el sobrepaso y tiempo de establecimiento, pero que el error de estado estacionario es del orden de 50%. Si para reducirlo se incrementa la ganancia, el sobrepaso sería excesivo, por lo tanto se recurrirá a un compensador por atraso de fase.

Compensador por atraso de fase

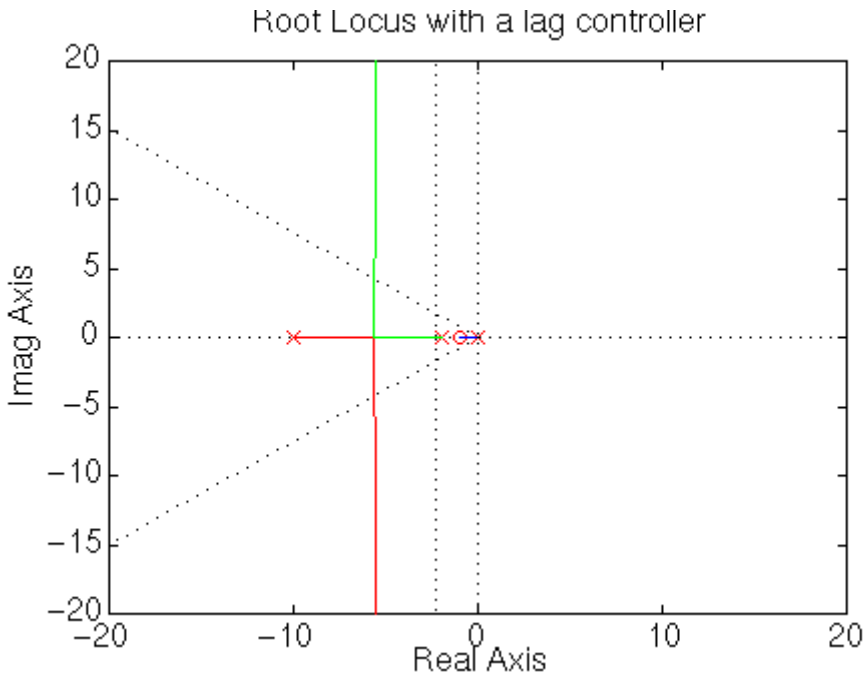
Dado que con ganancia se satisficieron los requerimientos dinámicos, se agregará un compensador por atraso a los efectos de mejorar el error de estado estacionario, tratando de no modificar considerablemente la dinámica. La transferencia del compensador es:

$$\frac{(s+1)}{(s+0.01)}$$

Para incluirlo en el lazo basta con realizar el siguiente *.m*:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;

num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];

z1=1;
p1=0.01;

numa = [1 z1];
dena = [1 p1];
numb=conv(num,numa);
denb=conv(den,dena);

rlocus(numb,denb)

sgrid(.8,0)
sigrid(2.3)
title('Root Locus with a lag controller')
```

siendo *numb* y *denb* el numerador y denominador de la transferencia directa del sistema compensado. Ejecutándolo, se obtiene:

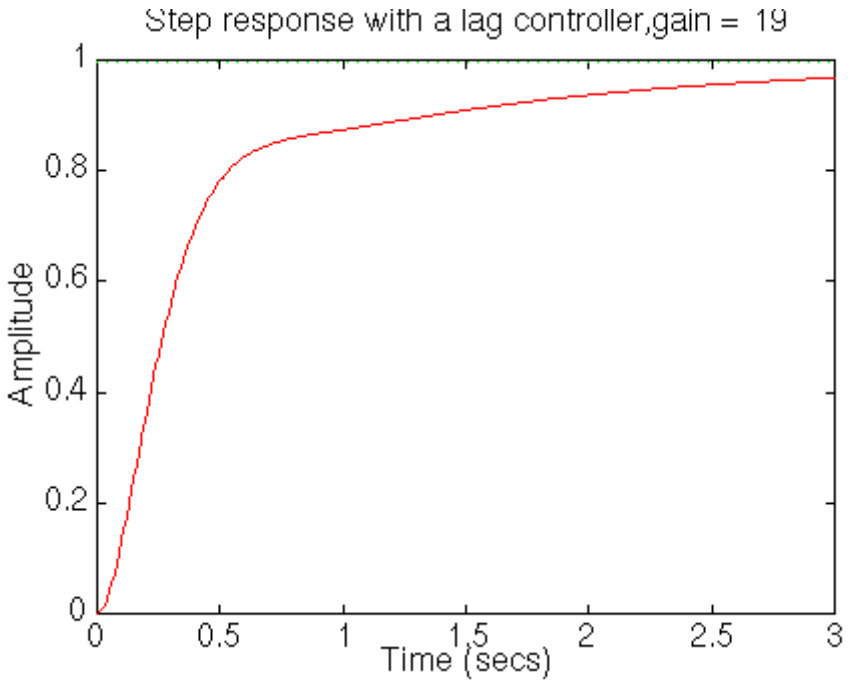


Graficación de la respuesta de lazo cerrado

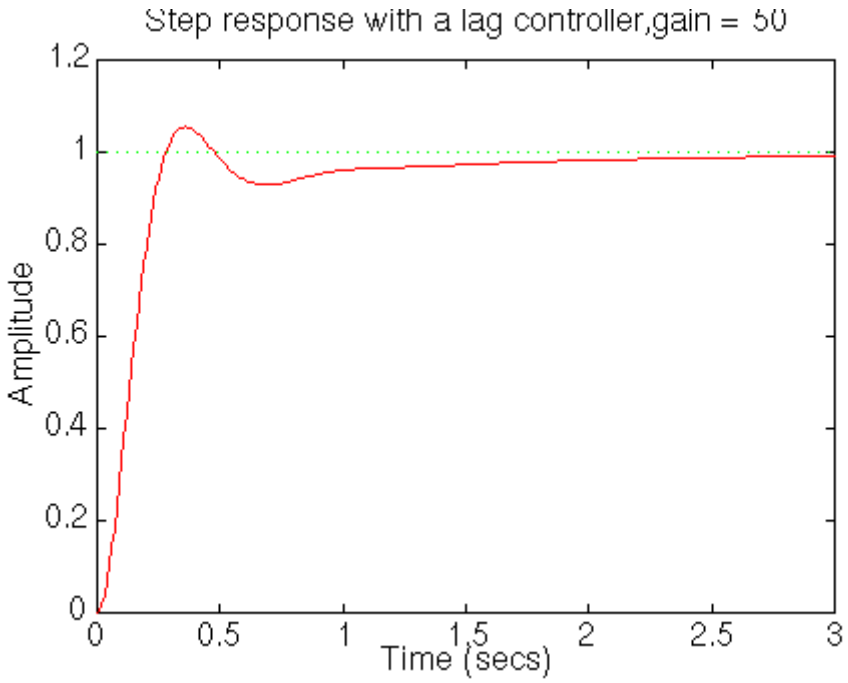
El siguiente archivo *.m* permite ver la respuesta al escalón del sistema a lazo cerrado:

```
[k,poles]=rlocfind(numb,denb)
[numc,denc]=cloop(k*numb,denb,-1);
t=0:0.01:3;
step(numc,denc,t)
title('Step response with a lag controller')
```

Luego de ejecutarlo y, sobre el diagrama de lugar de raíces, ubicar y clickear con el cursor del mouse sobre la línea diagonal punteada que cumplía los requerimientos dinámicos, se obtiene:



La ganancia obtenida será del orden de 20. Obsérvese que la respuesta no es del todo satisfactoria y que, gracias a la presencia del compensador, se podría aumentar la ganancia sin que por ello se comprometa el requerimiento de sobrepaso (que ahora esta muy por debajo de 5%). Por lo tanto se puede rejecutar el programa y elegir un punto por encima de la línea punteada, correspondiente a mayor ganacia y por ende menor error de estado estacionario. Por ensayo y error se puede obtener algo similar a:



correspondiente a una ganancia de aproximadamente 50, y verificando todas las especificaciones.

Sección 4: Método de Diseño Frecuencial para Control de Velocidad de un Motor CC

Diseño de un controlador por realimentación del vector de estados
Agregado de un controlador proporcional
Graficación de la respuesta a lazo cerrado
Agregado de un controlador por atraso de fase

En esta sección trataremos el diseño de un compensador para el motor de CC, basándonos en el método frecuencial. Consideremos las mismas especificaciones de diseño. Como en la sección anterior el primer paso es crear un archivo *.m* que contenga el modelo de la planta a lazo abierto.

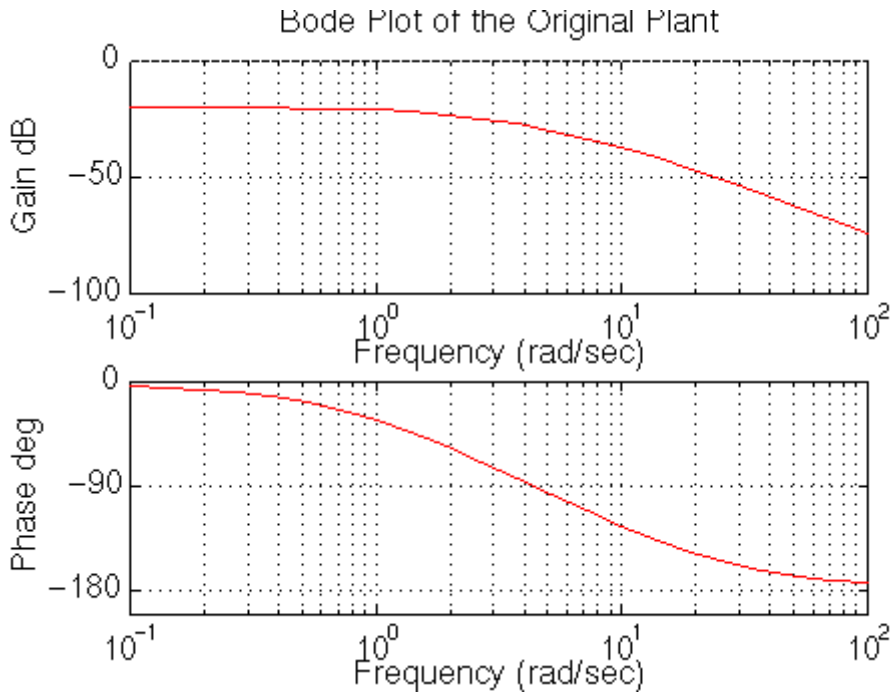
```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
```

Graficación del Diagrama de Bode

La idea principal de este método de diseño es utilizar el diagrama de Bode de la planta a lazo abierto para estimar su comportamiento a lazo cerrado. La idea entonces será agregar un compensador que modifique adecuadamente el Bode de lazo abierto de modo que la respuesta de lazo cerrado del conjunto cumpla con las especificaciones de diseño. Por consiguiente, el primer paso es graficar el Bode de la planta a lazo abierto, para ello hay que incluir el siguiente comando a nuestro archivo:

```
bode(num,den)
```

Ejecutándolo:



Agregado de un controlador proporcional

Del Bode puede observarse que la ganancia en continua es del orden de 0.1 (aproximadamente –29 dB), lo cual resulta en un importante error de EE. Nótese además, que el margen de fase (MF) puede ser mayor a 60 grados si ω es menor que 10 rad/s. En consecuencia, agregando ganancia de forma tal que el ancho de banda sea de 10 rad/s, resulta un MF de aproximadamente 60 grados. También puede observarse que el margen de ganancia (MG) en ω=10 rad/s es levemente mayor a -40 dB, (0.01 en magnitud). El comando *bode* invocado con argumento a izquierda, retorna las magnitudes exactas:

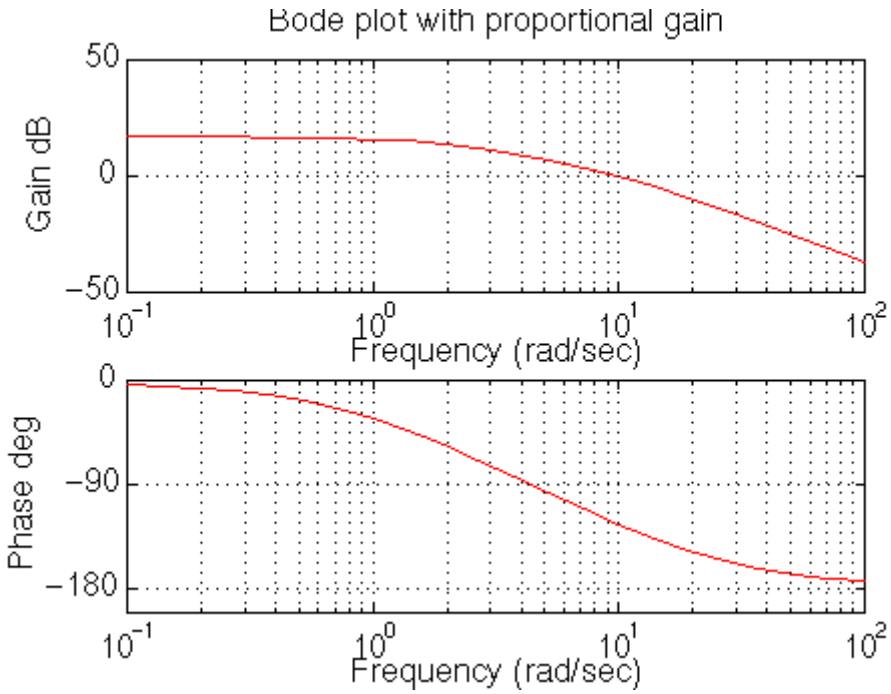
```
[mag,phase,w] = bode(num,den,10);
mag =

    0.0139
```

Nota: es importante recordar que dependiendo de la versión de Matlab, dicho comando puede necesitar como parámetro al sistema definido como SYS y no como num, den.

Por lo tanto, para tener ganancia 1 a ω=10 rad/s, basta con multiplicar el numerador por 1/0.0139 o sea aproximadamente por 70.

```
num = 70*num
```

y reejecutar el archivo *.m*:



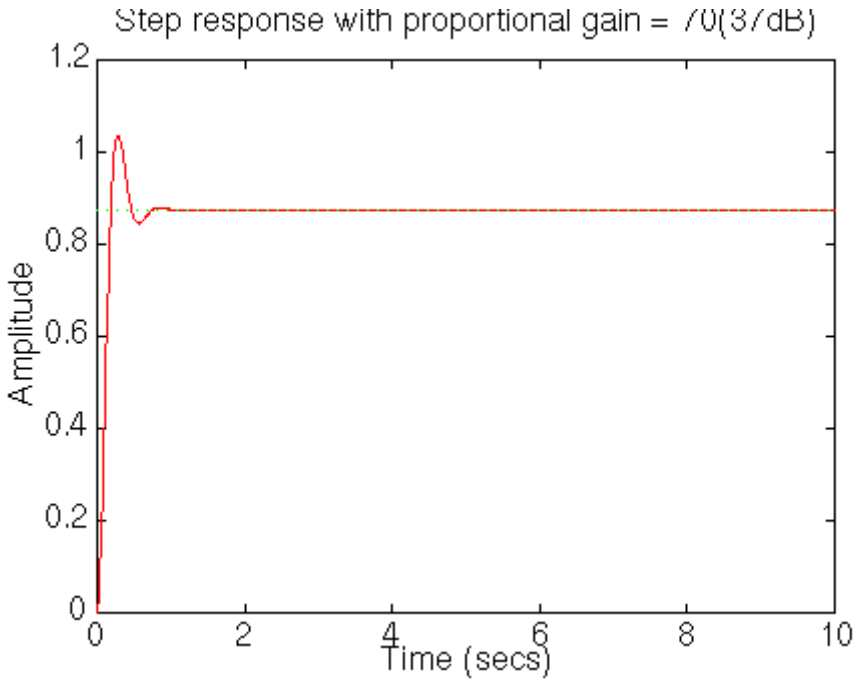
Resultando una ganancia en continua aproximada de 7.

Graficación de la respuesta de lazo cerrado

Para ver el comportamiento a lazo cerrado hay que agregar los siguientes comandos:

```
[numc,denc]=cloop(num, den, -1);
t=0:0.01:10;
step(numc,denc,t)
```

si se desea que no muestre los diagramas de Bode anteriores es suficiente con transformar los comandos *bode* en comentarios, precediéndolos de %. Ejecutando, se obtiene:



El tiempo de establecimiento es corto, pero el sobrepaso y el error de estado estacionario son inadmisibles. El sobrepaso podría reducirse disminuyendo levemente la ganancia (y así aumentar el margen de fase), pero esto tendría el efecto adverso de incrementar el error de estado estacionario. Por consiguiente, un controlador por atraso de fase parecería adecuado.

Agregado de un controlador por atraso de fase

El controlador por atraso permitirá aumentar la ganancia en continua (a los efectos de reducir el error de estado estacionario) y, simultáneamente, disminuir la ganancia en la zona de frecuencias de interés (respecto de la propuesta con K=70), de tal modo que el MF aumente (reduciendo el sobrepaso). Para el diseño entonces, propongamos una ganancia proporcional algo menor a la del caso anterior de unas 50 veces, e incluyamos en cascada un controlador cuya transferencia sea:

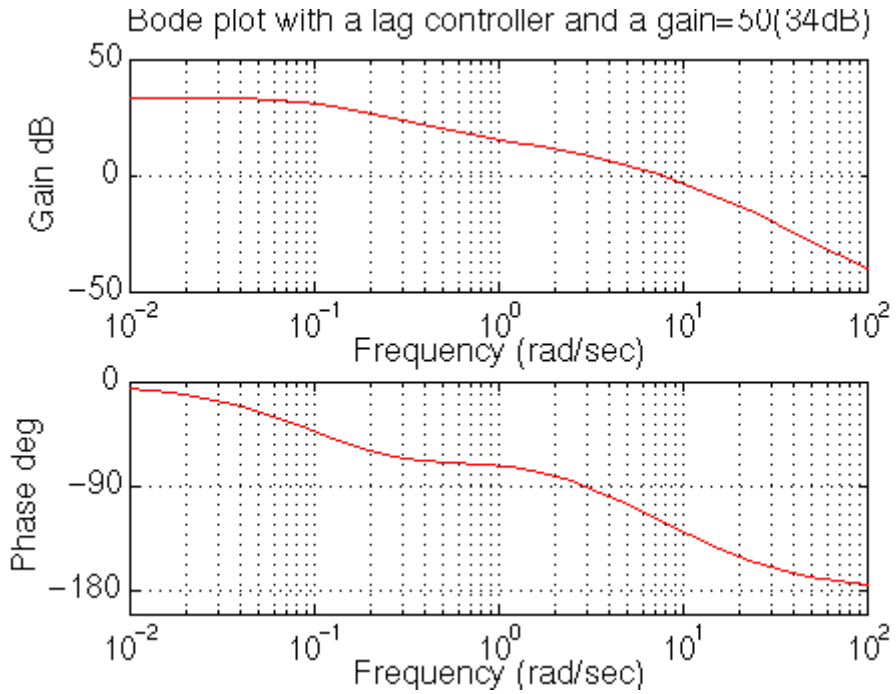
$$\frac{(s+1)}{(s+0.1)}$$

Obsérvese que este control a aporta una ganancia de 10 en continua (que, conjuntamente con los 50, resulta en una ganancia en bajas frecuencias de 500,

reduciendo el error de EE 500/70=7.1 veces respecto del caso proporcional puro). La frecuencia de corte es menor a ω=10 rad/s, incrementándose el MF y reduciéndose el sobrepaso. El único inconveniente podría surgir por el consecuente aumento en el tiempo de establecimiento, aunque el sistema parece no estar muy comprometido en este sentido. Para ver el Bode resultante, incluir:

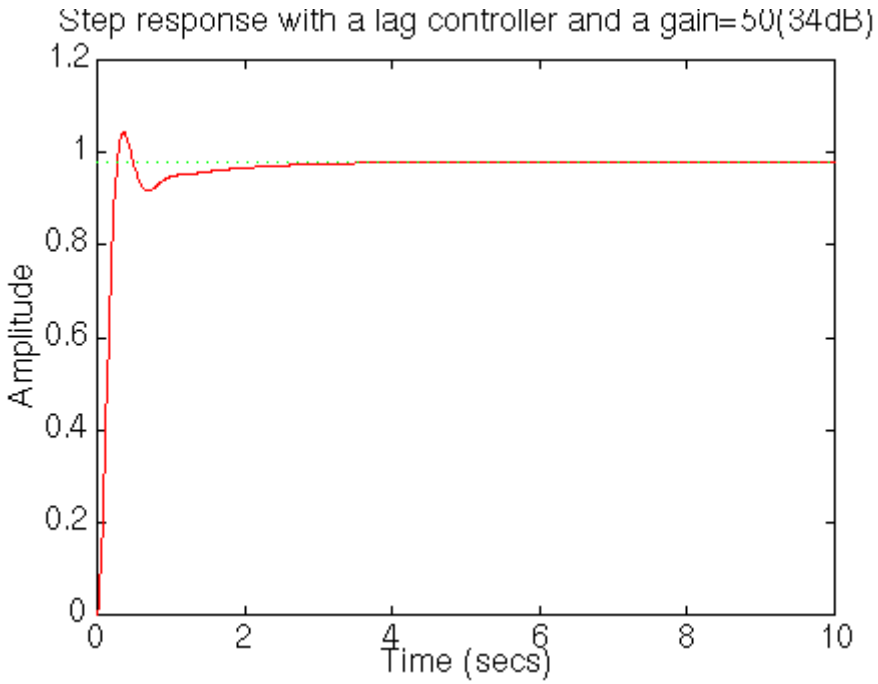
```
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
num=50*K;

z=1;
p=0.1;
numa=[1 z];
dena=[1 p];
numb=conv(num,numa);
denb=conv(den,dena);

bode(numb,denb)
```

y ejecutar:



Ahora, el MF parece aceptable y el error de EE es del orden de 1/40dB o 1%. Para cerrar el lazo y poder observar la respuesta obtenida, se agregan los siguientes comandos:

```
[numc,denc]=cloop(numb, denb, -1);
t=0:0.01:10;
step(numc,denc,t)
```



Finalmente, la respuesta a lazo cerrado cumple con las especificaciones: error de EE menor que 1%, sobrepaso cercano al 5% y tiempo de establecimiento de unos 2 s.

Sección 5: Control por Variables de Estado de la Velocidad de un Motor CC

Diseño de un controlador por realimentación del vector de estados
Incorporación de acción integral en el control
Sistema extendido

Recordando las ecuaciones de estado obtenidas:

$$\frac{d}{dt}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix}=\begin{bmatrix}-\frac{b}{J} & \frac{K}{J}\\ -\frac{K}{L} & -\frac{R}{L}\end{bmatrix}\cdot\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix}+\begin{bmatrix}0\\ \frac{1}{L}\end{bmatrix}\cdot v$$

$$\dot{\theta}=\begin{bmatrix}1 & 0\end{bmatrix}\cdot\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix}$$

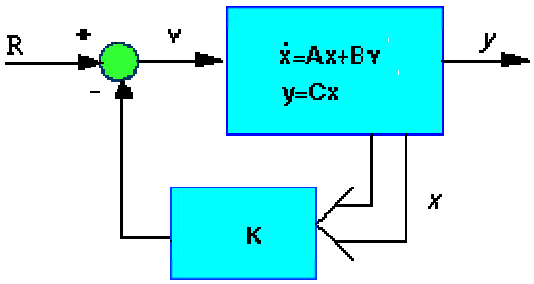
se diseñará un controlador para cumplir las antedichas especificaciones.

Como primer paso hay que crear un archivo *.m* que contenga las matrices del sistema de estados a lazo abierto:

```
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
A=[-b/J   K/J
   -K/L   -R/L];
B=[0
   1/L];
C=[1   0];
D=0;
```

Diseño de un controlador por realimentación del vector estados

Dado que ambos estados son fácilmente medibles (con un amperímetro y un tacómetro), se puede suponer que no es necesario un observador. Un diagrama esquemático del sistema con realimentación de estados es:



Recuérdese que el polinomio característico del sistema a lazo cerrado está determinado por el determinante de (sI-(A-BK)), siendo s la variable de Laplace. En este caso la matriz A-B*K de lazo cerrado es de 2x2, teniendo el sistema dos polos. Dado que el sistema es totalmente controlable, por medio de la realimentación del vector de estados completo se pueden ubicar los polos de lazo cerrado en cualquier posición deseada del espacio de estados. En un primer intento los asignaremos en -5 + i y -5-i (esto es amortiguamiento ξ = 0.98 y σ=5, correspondiente a un sobrepaso del 0.1% y un tiempo de establecimiento de 1 s, respectivamente). El vector de ganancia de realimentación puede obtenerse con el comando *place*. Agregando al archivo *.m*:

```
p1 = -5 + i;
p2 = -5 - i;
K = place(A,B,[p1 p2]);
```

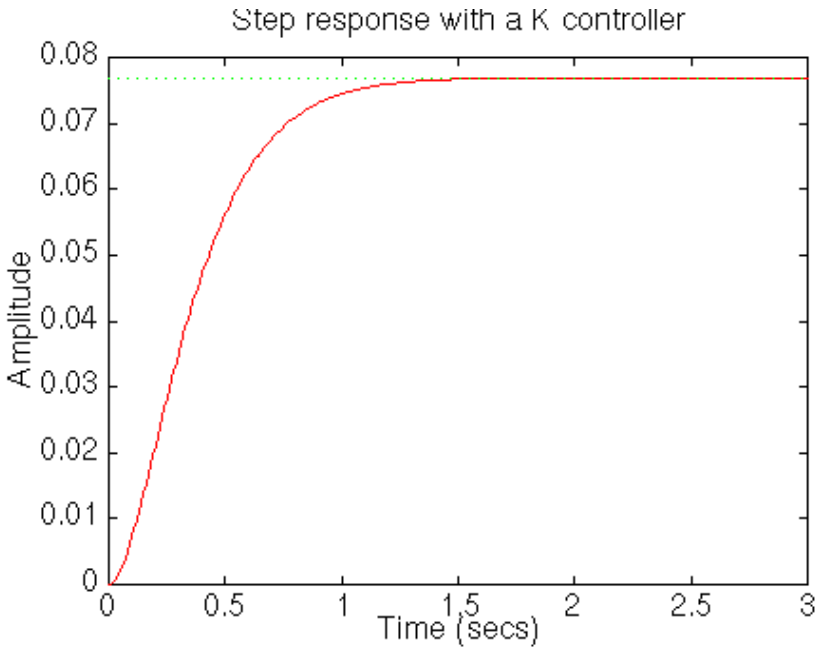
resulta el sistema de estados a lazo cerrado:

$$\dot{X} = (A - B \cdot K) \cdot X + B \cdot R$$
$$Y = C \cdot X$$

Para ver la respuesta a lazo cerrado basta incluir:

```
t=0:0.01:3;
step(A-B*K,B,C,D,1,t)
```

y ejecutar:



Incorporación de acción integral en el control. Sistema extendido

De la gráfica se observa que el error de EE es muy grande. Esto se soluciona incorporando acción integral en el control. Para ello hay que extender el sistema agregando un estado que sea la integral del error de seguimiento Y-R:

$$\dot{X}_i = Y - R$$

resultando el sistema de estados extendido:

$$\dot{X}_e = A_e \cdot X_e + B_e \cdot R = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ X_i \end{bmatrix} + \begin{bmatrix} B \\ -1 \end{bmatrix} \cdot R$$

$$Y = C_e \cdot X_e = \begin{bmatrix} C & 0 \end{bmatrix} \cdot X_e$$

en Matlab:

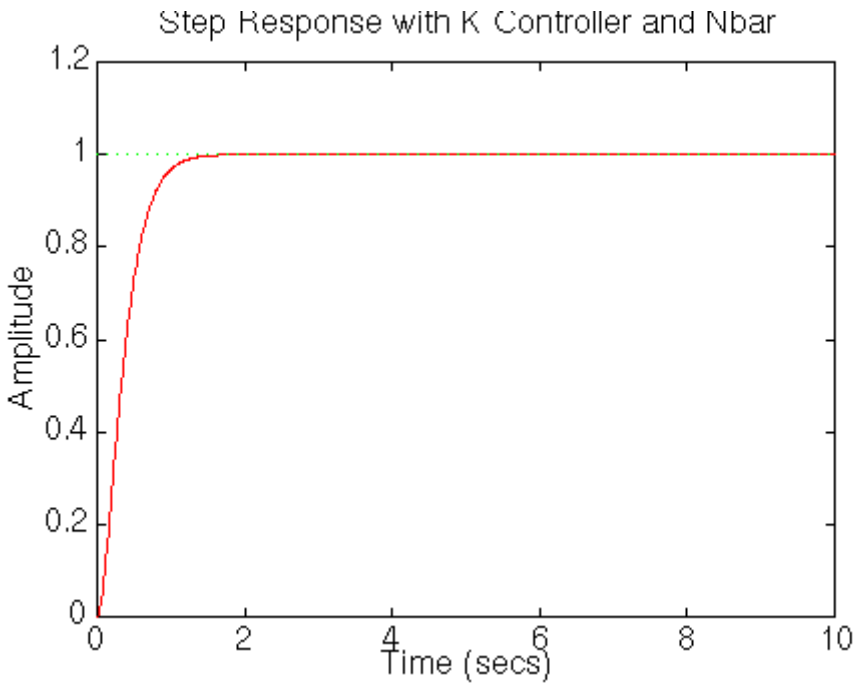
```
Ae = [A [0 0]';C 0];
Be = [B;0];
Ce = [c 0];
De = 0;
```

Calcularemos la ganancia de realimentación extendida K_e para obtener dos polos de lazo cerrado en -5 + i y -5-i y el tercero 100 veces más lejos a los efectos de que p_1 y p_2 sean dominantes:

```
p1 = -5 + i;
p2 = -5 - i;
p3 = 100*real(p1);
Ke = place(Ae,Be,[p1 p2 p3]);
```

Finalmente, para graficar la respuesta a lazo cerrado:

```
t=0:0.01:10;
step(Ae-Be*Ke,Be,Ce,De,1,t)
title('Step Response with Ke')
```



Puede observarse que la respuesta a lazo cerrado cumple con las especificaciones.

Nota: tener presente que dependiendo de la versión de Matlab, algunos comandos pueden necesitar como parámetro al sistema definido como SYS y no como A,B,C,D (Ej: SYSe=SS(Ae,Be,Ce,De)*).*

Sección 6: Control Digital de la Velocidad de un Motor CC

Conversión de continuo a digital
Controlador PID discreto

El primer paso en el diseño será convertir la transferencia de lazo abierto del sistema continuo:

$$\frac{\dot{\Theta}}{V} = \frac{K}{(J \cdot s + b) \cdot (L \cdot s + R) + K^2}$$

a discreto utilizando el comando *c2dm*. Este comando requiere los siguientes argumentos: el polinomio numerador (*num*), el denominador (*den*), el período de muestreo (*Ts*) y el tipo de circuito retenedor deseado. En este ejemplo utilizaremos el retenedor de orden cero('*zoh*')

Tomemos Ts=0.12s, que es aproximadamente un décimo de la constante de tiempo de un sistema con tiempo de establecimiento de 2s. Creando el siguiente archivo *.m*:

```
R=1;
L=0.5;
Kt=0.01;
J=0.01;
b=0.1;
num = Kt;
den = [(J*L) (J*R)+(L*b) (R*b)+(Kt^2)];

Ts = 0.12;
[numz,denz] = c2dm(num,den,Ts,'zoh')
```

y ejecutándolo, se obtiene:

```
numz =
         0    0.0092    0.0057
denz =
    1.0000   -1.0877    0.2369
```

Por lo que la transferencia discreta es:

$$\frac{\dot{\Theta}}{V} = \frac{0.0092 \cdot z + 0.0057}{z^2 - 1.0877 \cdot z - 0.2369}$$

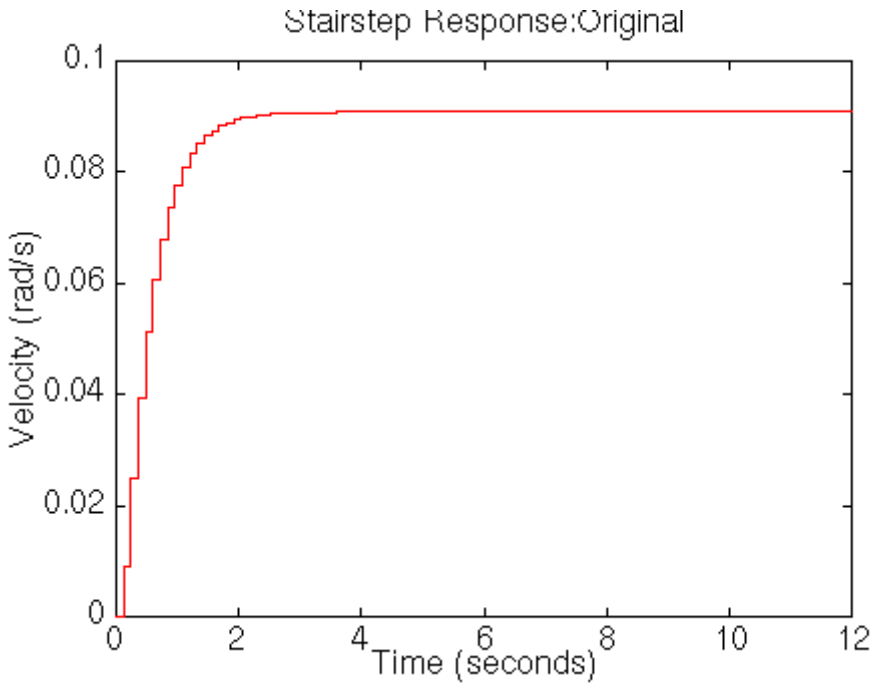
Primero observaremos como es la respuesta de lazo cerrado sin ningún tipo de controlador. Puede verse que la matriz *numz* tiene un cero extra correspondiente a la segunda potencia. Este debe ser eliminado para poder cerrar el lazo con el comando *cloop*. Para ello:

```
numz = [numz(2) numz(3)];
[numz_cl,denz_cl] = cloop(numz,denz);
```

Hecho esto, se aplica un escalón de entrada al sistema discreto con el comando *dstep* (indicando como último parámetro el número de puntos, es este caso será 101). Generando el vector de tiempo en base al Ts y graficando:

```
[x1] = dstep(numz_cl,denz_cl,101);
t=0:0.12:0.12*100;
plot(t,x1)
% opcional en lugar de plot stairs(t,x1)
xlabel('Time (seconds)')
ylabel('Velocity (rad/s)')
title('Stairstep Response:Original')
```

resulta:



Controlador PID discreto

Recordando la transferencia analógica de un PID:

$$PID(s) = K_P + \frac{K_I}{s} + K_D \cdot s$$

Existen diversas formas de mapear del plano s al z. Dado que la transferencia en s del PID tiene más ceros que polos, elegiremos por simplicidad para este ejemplo una transformación bilineal tal como:

$$s = \frac{2}{T \cdot s} \cdot \frac{z-1}{z+1}$$

Esta conversión a z la hará el comando *c2dm* empleando el método tustin (el cual usa aproximación bilineal para pasar de continuo a discreto). La constantes del PID analógico eran Kp = 100, Ki = 200 y Kd = 10, por lo tanto

```
% Discrete PID controller with bilinear approximation
Kp = 100;
Ki = 200;
Kd = 10;

[dencz,numcz]=c2dm([1 0],[Kd Kp Ki],Ts,'tustin')
```

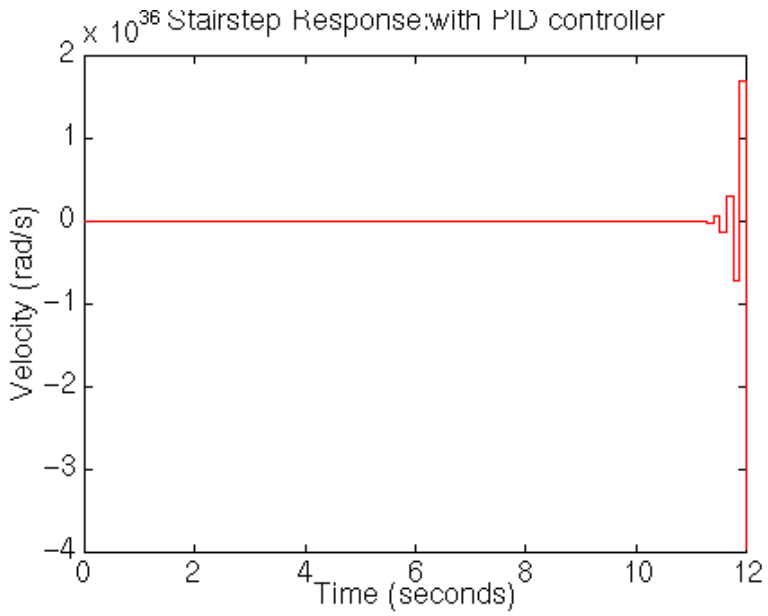
resultando:

```
dencz =

    0.0036         0   -0.0036

numcz =

    1.0000   -1.1100    0.2823
```

Nótese que se debió invertir numerador por denominador. La razón es que la transferencia del PID en s es impropia y el comando no lo aceptaría, sin embargo, con esta inversión (y cambiando los argumentos a derecha de la igualdad) se puede "engañar" a *c2dm* para que calcule la transferencia digital del PID. Poniendo el PID digital en cascada con la planta discreta y cerrando el lazo:

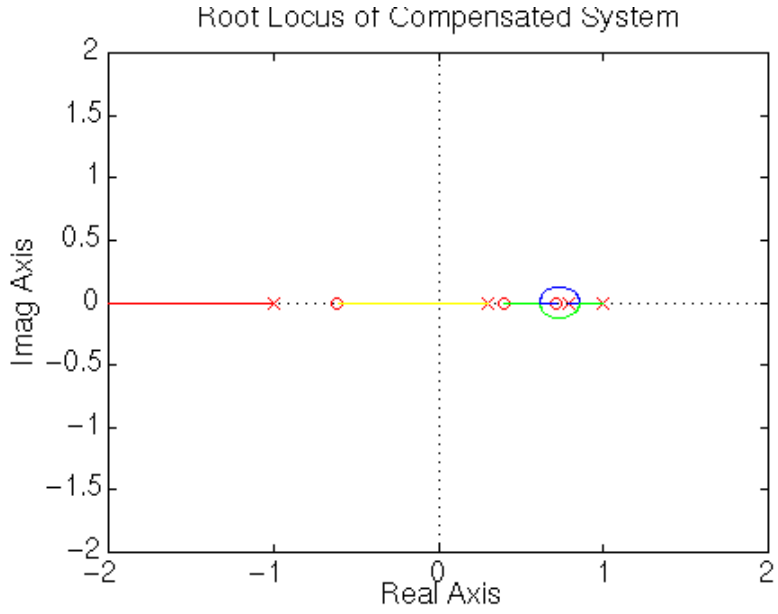
```
numaz = conv(numz,numcz);
denaz = conv(denz,dencz);
```

```
[numaz_cl,denaz_cl] = cloop(numaz,denaz);
[x2] = dstep(numaz_cl,denaz_cl,101);
t=0:0.12:12;
stairs(t,x2)
xlabel('Time (seconds)')
ylabel('Velocity (rad/s)')
title('Stairstep Response:with PID controller')
```



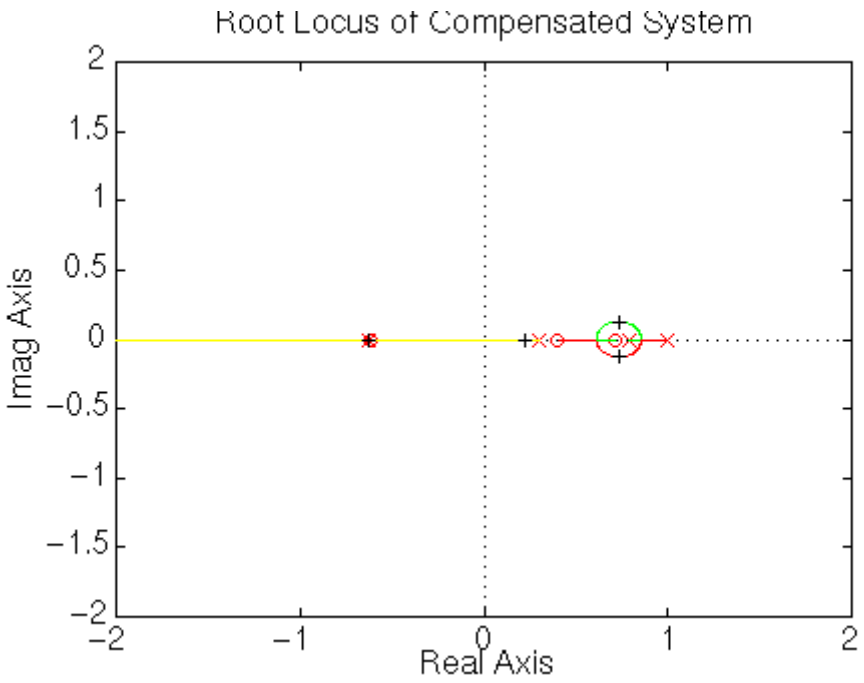
Se puede ver que el sistema a lazo cerrado es inestable. Hay algo mal en el compensador, por lo que es conveniente observar el lugar de raíces en z del sistema compensado:

```
rlocus(numaz,denaz)
title('Root Locus of Compensated System')
```



Se detecta que para esa sintonía el PID pone un polo de lazo abierto en –1, que al cerrar el lazo con cualquier ganancia positiva, sale fuera del circulo unitario inestabilizando el sistema. La solución es cambiar la sintonía del controlador. Una posibilidad es hacer que ese polo cancele aproximadamente el cero en –0.62, haciendo que el sistema sea estable para un dado rango de ganancias. Para ello hay que modificar el denominador del compensador, manteniendo el polo en z=1 (integrador en z) y el otro en-0.625. Posteriormente se coloca el compensador (el numerador no fue modificado) en cascada y se grafica el lugar de raíces:

```
dencz = conv([1 -1],[1/0.625 1])
numaz = conv(numz,numcz);
denaz = conv(denz,dencz);

rlocus(numaz,denaz)
title('Root Locus of Compensated System');
```



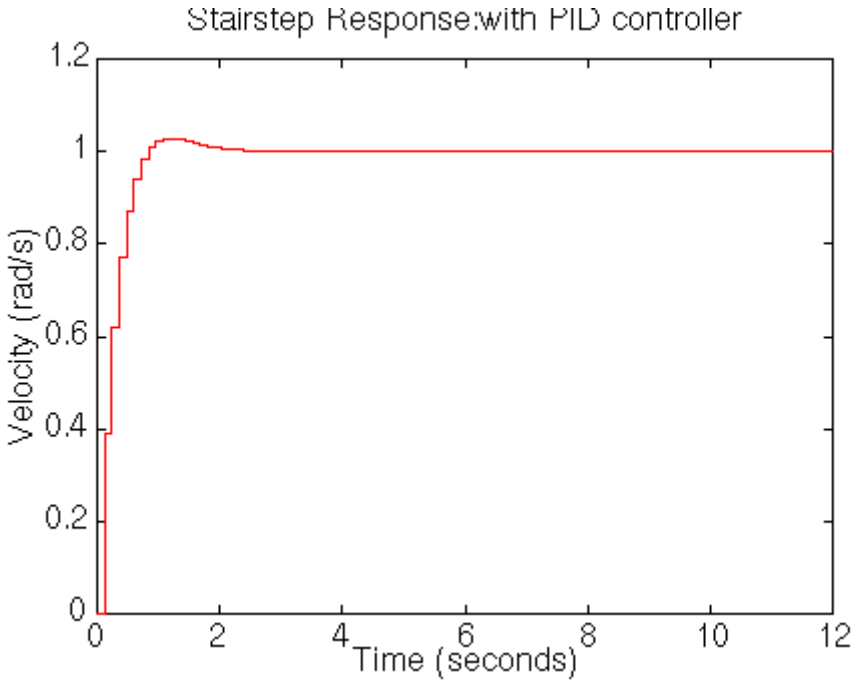
La ganancia necesaria para satisfacer las especificaciones puede obtenerse del lugar de raíces empleando el comando *rlocfind*

```
[K,poles] = rlocfind(numaz,denaz)
```

debiendo seleccionar en el lugar de raíces donde se desea posicionar los polos de

lazo cerrado. Luego se cierra el lazo con esa ganancia:

```
[numaz_cl,denaz_cl] = cloop(K*numaz,denaz);

[x3] = dstep(numaz_cl,denaz_cl,101);
t=0:0.12:12;
stairs(t,x3)
xlabel('Time (seconds)')
ylabel('Velocity (rad/s)')
title('Stairstep Response:with PID controller')
```



Graficando se observa que ahora el sistema cumple las especificaciones de diseño.