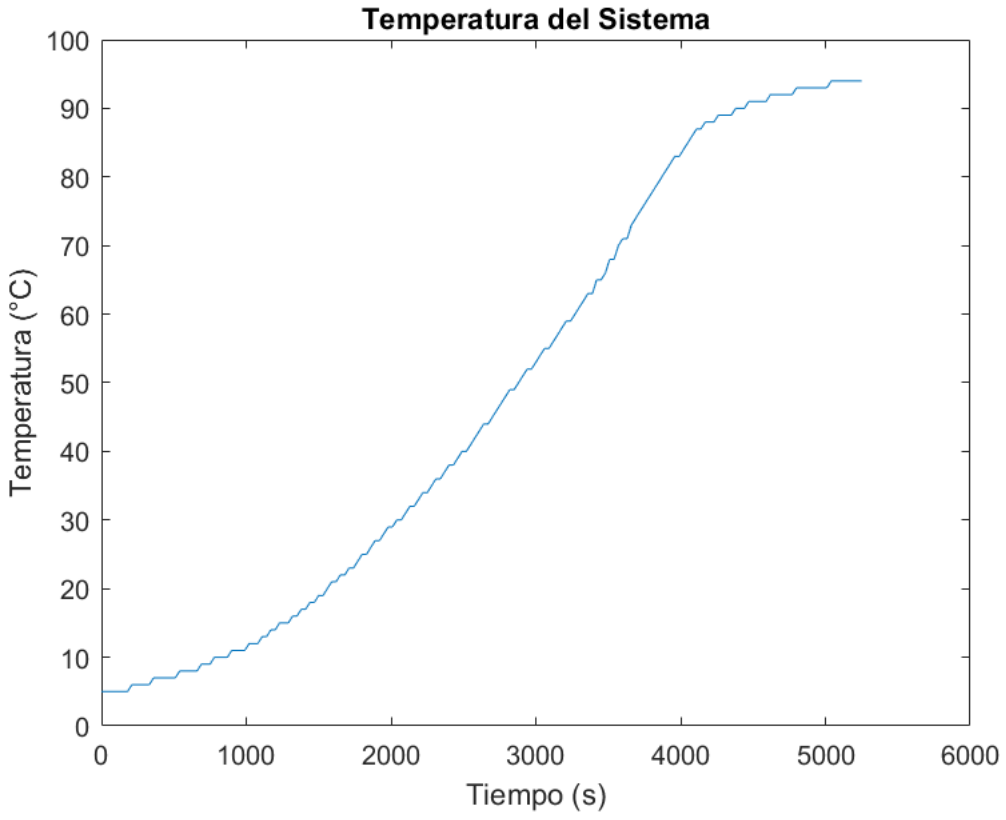
Configuración de documento

```
format shortG
data = readtable('PID.xlsx','Sheet', "Hoja2");
warning('off','curvefit:fit:equationBadlyConditioned');
```

Datos iniciales

Los siguientes datos se obtuvieron midiendo la temperatura de una cubeta con hielos, agua y una resistencia de calentamiento cada treinta segundos, durante un periodo de 1:45 hrs.

```
figure(1);
plot(data.Tiempo,data.Temperatura)
title("Temperatura del Sistema");
ylabel("Temperatura (°C)");
xlabel("Tiempo (s)")
```

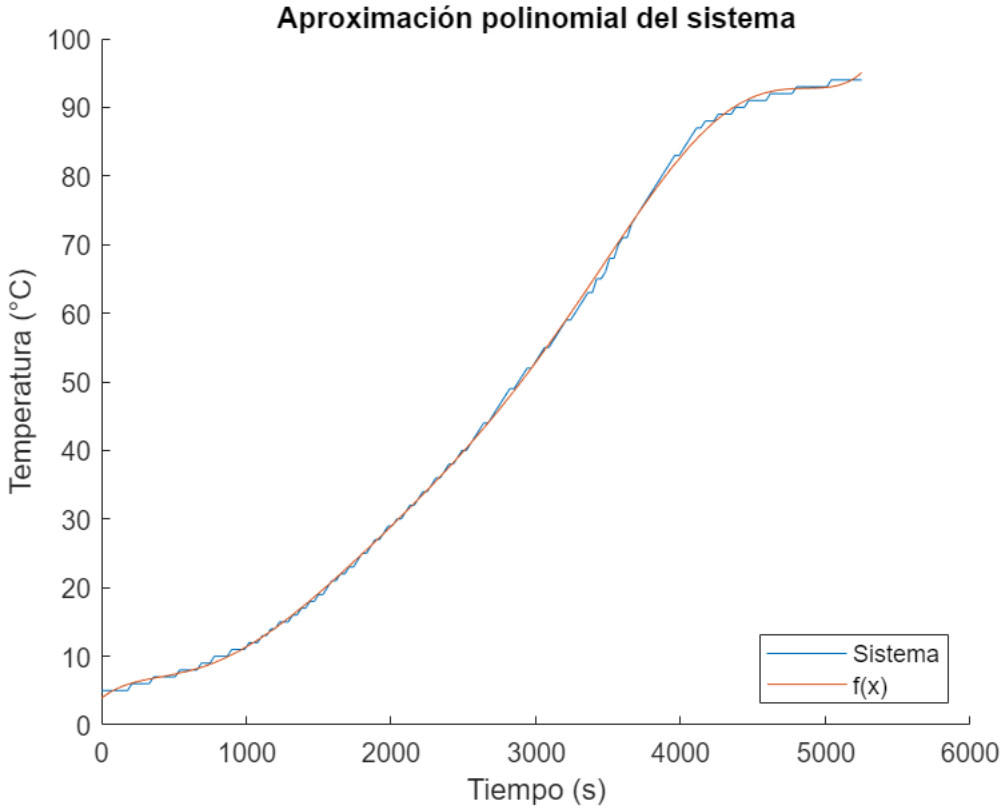


Obtención de la tangente en el punto de inflección

Polinomio representativo

Comenzamos obteniendo un polinomio de grado 7 para representar nuestra curva lo más cercanamente posible.

```
[xData, yData] = prepareCurveData( data.Tiempo, data.Temperatura );
ft = fittype( 'poly7' );
[fitresult, gof] = fit( xData, yData, ft );
syms x
f(x) = poly2sym(coeffvalues(fitresult));

figure(2);
hold on
plot(data.Tiempo,data.Temperatura);
fplot(f,[0,5250]);
title("Aproximación polinomial del sistema");
ylabel("Temperatura (°C)");
xlabel("Tiempo (s)")
legend(["Sistema","f(x)"],"Location","southeast");
```



Obtención del punto de inflección

A través de la segunda derivada, podemos obtener los puntos de inflección de nuestro sistema. Estos puntos se encuentran donde $f''(x) = 0$. Obtendremos varios puntos, así que seleccionaremos el que más se adecúa a nuestras necesidades, que en este caso es el segundo.

```
h(x) = simplify(diff(f,x,2));
inflection_list = vpasolve(h == 0, x, [-inf, inf]);
inflection = double(inflection_list(2))
```

```
inflection =
       3464.9
```

Formulación de la línea tangente

Una vez obtenido el punto de inflección adecuado, podemos utilizar la función original y su primera derivada, evaluadas en ese punto, para obtener la línea tangente. La línea tangente en un punto x_o tiene la forma:

$$y(x) = f'(x_o)(x - x_o) + f(x_o).$$

```
d(x) = diff(f,x,1);
m = d(inflection);
y = f(inflection);
tang(x) = m*(x-inflection)+y;
tang = vpa(tang,4)
```

tang(x) = $0.03153\,x - 42.23$

Obtención de las constantes para la función de transferencia

Una vez obtenida la tangente, podemos obtener las constantes *K, T* y *L*, necesarias para obtener nuestra función de transferencia y los valores del controlador PID. La constante *K* es igual al valor máximo de nuestro sistema, que en este caso son **100°C**. La constante *L* es el tiempo de retardo, obtenido a partir del punto donde nuestra línea de tangente cruza el eje X. Finalmente, la constante *T*, o tau, es el tiempo del sistema, comprendido desde *L* hasta donde la línea tangente cruza nuestro valor de *K*.

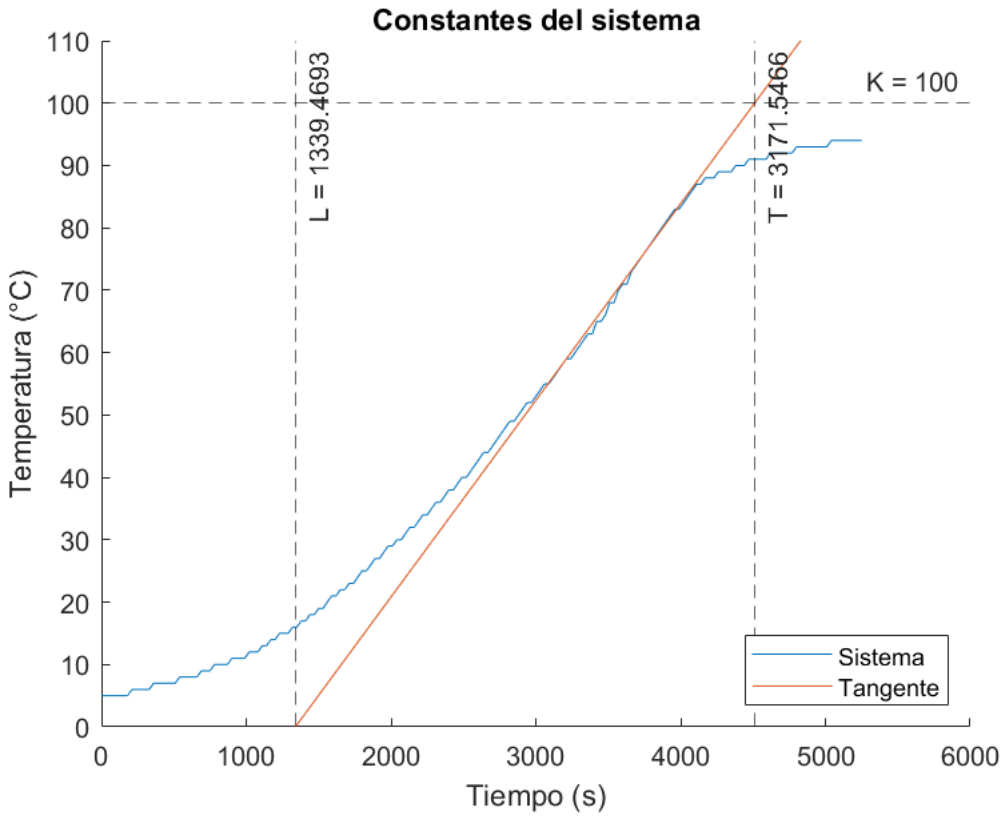
```
K = 100;
L = double(solve(tang == 0,x))
```

```
L =
       1339.5
```

```
T = double(solve(tang == K,x)) - L
```

```
T =
       3171.5
```

```
figure(3);

hold on

plot(data.Tiempo, data.Temperatura);
fplot(tang);

xline(L,"--","L = "+L);
xline(T+L,"--","T = "+T);
yline(K,"--","K = "+K);
title("Constantes del sistema");
ylabel("Temperatura (°C)");
xlabel("Tiempo (s)")
ylim([0 110]);
xlim([0 6000]);
legend(["Sistema","Tangente"],"Location","southeast");
```



Función de transferencia

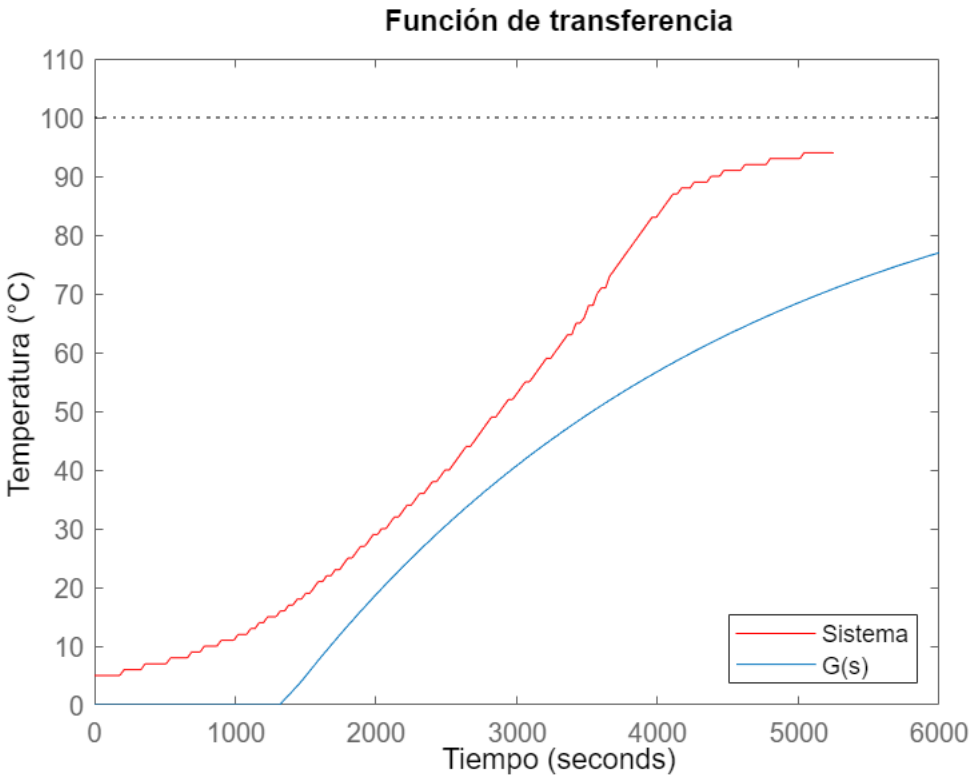
Con las constantes previamente obtenidas, podemos generar una función de transferencia para la aproximación del sistema, la cual tiene la forma:

$$G(s) = e^{-Ls}\frac{K}{T+1}.$$

```
G = tf(K,[T 1],'InputDelay',L)
```

```
G =

                        100
  exp(-1.34e+03*s) * ----------
                     3172 s + 1

Continuous-time transfer function.
```

```
figure(4);
hold on
plot(data.Tiempo, data.Temperatura,"r");
step(G);
title("Función de transferencia");
ylabel("Temperatura (°C)");
xlabel("Tiempo")
legend(["Sistema","G(s)"],"Location","southeast");
ylim([0 110]);
xlim([0 6000]);
```



Como podemos observar, esta primera función de transferencia, a pesar de tener la forma correcta, tiene un error significativo. Para eliminar esto, podemos agregarle un *offset* a *T* y ajustar la curva para que se acerque más a nuestro sistema, y así reducir el error.

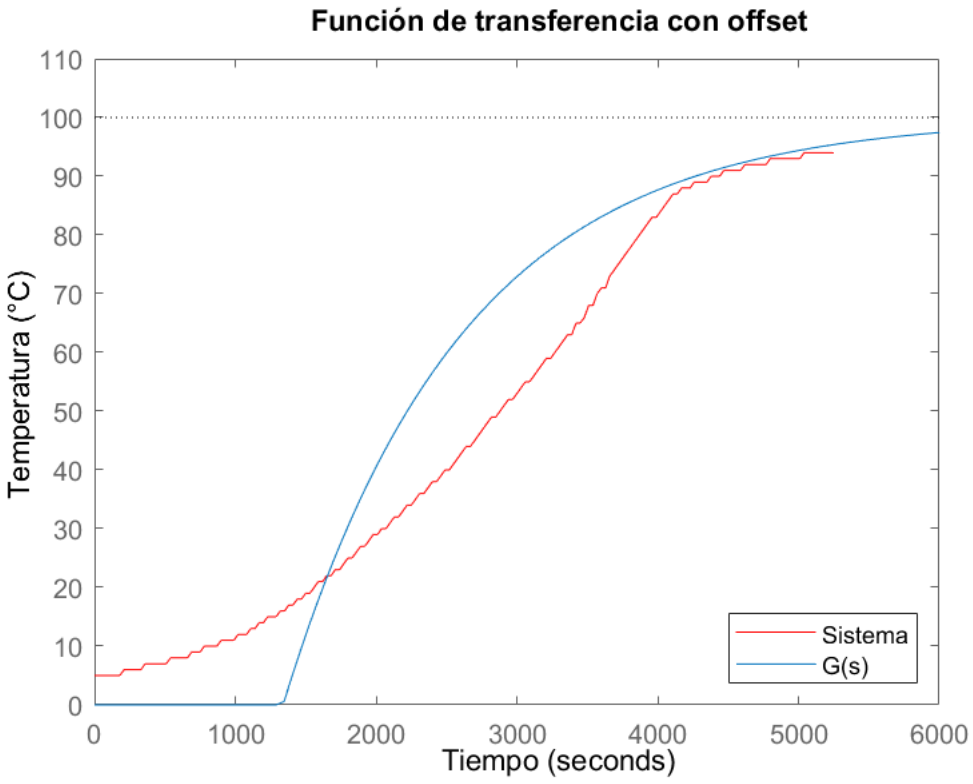
```
offset = 1900;
T = T - offset
```

```
T =
       1271.5
```

```
G = tf(K,[T 1],'InputDelay',L)
```

```
G =

                        100
  exp(-1.34e+03*s) * ----------
                     1272 s + 1

Continuous-time transfer function.
```

```
figure(5);

hold on

plot(data.Tiempo, data.Temperatura,"r");
step(G);
title("Función de transferencia con offset");
ylabel("Temperatura (°C)");
xlabel("Tiempo")
legend(["Sistema","G(s)"],"Location","southeast");
ylim([0 110]);
xlim([0 6000]);
```



Obtención de constantes PID

Finalmente, una vez obtenida la función de transferencia y los valores de *T* y *L,* podemos sintonizar la banda proporcional y los tiempos integrales y derivativos, mismos que alimentaremos al controlador **REX-c100**. Para obtener dichos valores, utilizaremos como guía la siguiente herramienta:

Tipo de controlador	K_p	T_i	T_d
P	$\frac{T}{L}$	∞	0
PI	$0.9\frac{T}{L}$	$\frac{L}{0.3}$	0
PID	$1.2\frac{T}{L}$	$2L$	$0.5L$

En esta ocasión, utilizaremos un controlador PID, por lo que estaremos basándonos en la tercera fila de la tabla anterior.

```
Kp = 1.2*T/L
```

```
Kp =
       1.1391
```

```
Ti = 2*L
```

```
Ti =
       2678.9
```

```
Td = L/2
```

```
Td =
       669.73
```

Es importante destacar que el controlador REX-c100 no utiliza la ganancia proporcional, sino que utiliza la *banda proporcional*, que es simplemente el porcentaje inverso de la ganancia.

```
Pb = 100/Kp
```

```
Pb =
       87.785
```