# HW #4

# Cagatay Duygu - 48369962

# Q 4.1 a)

In[ ]:= `Quit[]`

In[1]:=
```
μo = 0.000001258
Ap = 0.000146
ho = 0.000508
m = 0.3
Nc = 100
g = 9.81
```

Out[1]= $1.258 \times 10^{-6}$

Out[2]= 0.000146

Out[3]= 0.000508

Out[4]= 0.3

Out[5]= 100

Out[6]= 9.81

In[7]:= $\alpha = \mu o\ Nc\ Ap^2$

Out[7]= $2.68155 \times 10^{-12}$

In[8]:= $Io = \sqrt{m\ g\ \frac{ho^2}{\alpha}}$

Out[8]= 532.189

In[9]:= $ki = 2\ \alpha\ \frac{Io}{ho^2}$

Out[9]= 0.01106

In[10]:=

$$kx = 2\,\frac{\alpha\,Io^2}{ho^3}$$

Out[10]=

11 586.6

In[11]:=

$$A = \begin{pmatrix} 0 & 1 \\ \frac{kx}{m} & 0 \end{pmatrix}$$

Out[11]=

```
{{0, 1}, {38 622., 0}}
```

In[12]:=

$$CC = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

Out[12]=

```
{{1, 0}}
```

In[13]:=

$$B = \begin{pmatrix} 0 \\ \frac{ki}{m} \end{pmatrix}$$

Out[13]=

```
{{0}, {0.0368666}}
```

In[14]:=

```
Eigenvalues[A]
```

Out[14]=

```
{196.525, -196.525}
```

In[15]:=

```
Eigenvalues[A]
```

Out[15]=

```
{196.525, -196.525}
```

First element is bigger than zero. Thus, this system is not stable. Let's check observability and controllability.

In[35]:=

$$P = Join[B, A.B, 2];$$
$$P \text{ // MatrixForm}$$
$$Om = Join[CC^{T}, A^{T}.CC^{T}, 2]^{T};$$
$$Om \text{ // MatrixForm // N}$$

Out[36]//MatrixForm=

$$\begin{pmatrix} 0 & 0.0368666 \\ 0.0368666 & 0. \end{pmatrix}$$

Out[38]//MatrixForm=

$$\begin{pmatrix} 1. & 0. \\ 0. & 1. \end{pmatrix}$$

In[20]:=

```
MatrixRank[P]
```

Out[20]=

2

In[39]:=

```
MatrixRank[Om]
```

Out[39]=

2

Both ranks of P, Ob != 0. The system is observable and controllable.

In[40]:=
```
I2 = IdentityMatrix[2];
aa[s] = Det[s I2 - A] // Expand // Rationalize
```

Out[41]=

$$-\frac{4905000}{127} + s^2$$

## Desired characteristic polynomial:

In[42]:=
```
αd[s] = (s + α1) (s + α2) // N // Expand
```

Out[42]=

$$s^2 + s\,\alpha 1 + s\,\alpha 2 + \alpha 1\,\alpha 2$$

In[43]:=
```
Ut = (1 0
      0 1);
αdc = Reverse[Drop[CoefficientList[αd[s], s], -1]];
ac = Reverse[Drop[CoefficientList[aa[s], s], -1]];
Lt = {(αdc - ac).Inverse[OmT.Ut]};
L = LtT // FullSimplify // Rationalize
```

Out[47]=

$$\left\{\{\alpha 1 + \alpha 2\}, \left\{\frac{4905000}{127} + \alpha 1\,\alpha 2\right\}\right\}$$

In[48]:=
```
AA = A - L.CC // Chop ;
MatrixForm[AA]
```

Out[49]//MatrixForm=

$$\begin{pmatrix} -\alpha 1 - \alpha 2 & 1 \\ -\alpha 1\,\alpha 2 & 0 \end{pmatrix}$$

In[50]:=
```
Eigenvalues[AA] // FullSimplify
```

Out[50]=

$$\{-\alpha 1, -\alpha 2\}$$

I could be able to place observer eigenvalues into the desired place.

In[51]:=
```
x[t_] := {x1[t], x2[t]};
y[t_] := CC.x[t]
u[t_] := {u1[t]}
EqOL = Thread[x'[t] == A.x[t] + B.u[t]] // Chop // Flatten;
TableForm[EqOL]

xo[t_] := {xo1[t], xo2[t]};
Eqo = Thread[xo'[t] == AA.xo[t] + L.y[t] + B.u[t]] // Chop // Flatten;
TableForm[Eqo]
```

Out[55]//TableForm=

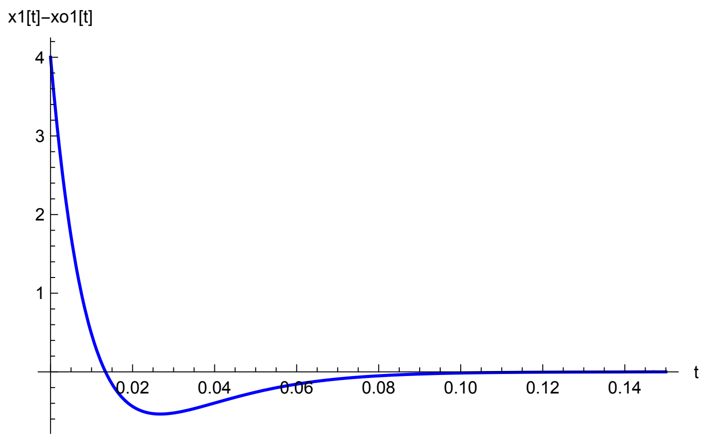
$$x1'[t] == x2[t]$$
$$x2'[t] == 0.0368666\, u1[t] + 38\,622.\, x1[t]$$

Out[57]//TableForm=

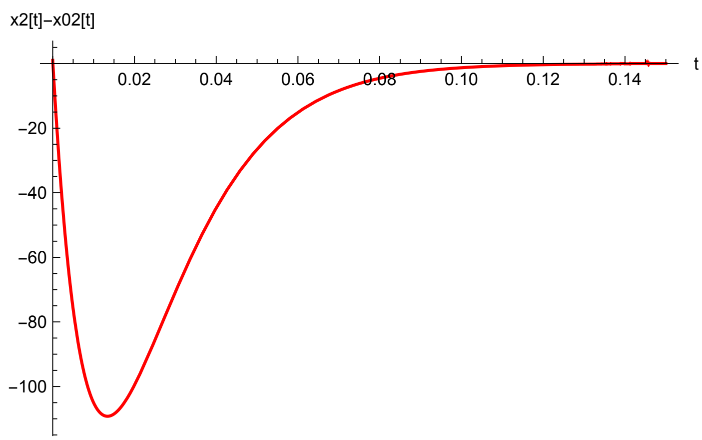
$$xo1'[t] == (\alpha 1 + \alpha 2)\, x1[t] + (-\alpha 1 - \alpha 2)\, xo1[t] + xo2[t]$$
$$xo2'[t] == 0.0368666\, u1[t] + \left(\frac{4\,905\,000}{127} + \alpha 1\, \alpha 2\right) x1[t] - \alpha 1\, \alpha 2\, xo1[t]$$

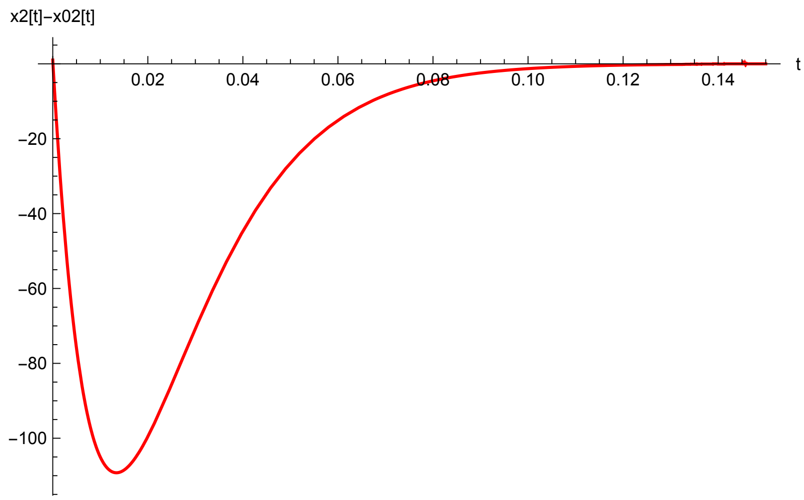
In[58]:=
```
ICob = {xo1[0] == 0, xo2[0] == 0};
IC = {x1[0] == 4, x2[0] == 1};
Inputs = {u1[t] → 1, α1 → 70, α2 → 80}; tmax = .15;
ObResponse = NDSolve[{EqOL /. Inputs, Eqo /. Inputs, IC, ICob},
   {x[t], xo[t]} // Flatten, {t, 0, tmax}];

Plot[Evaluate[{x1[t] - xo1[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x1[t]-xo1[t]"}, PlotRange → All, PlotStyle → {Blue}]
Plot[Evaluate[{x2[t] - xo2[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x2[t]-x02[t]"}, PlotRange → All, PlotStyle → {Red}]
```

Out[62]=




Out[63]=






# Q 4.1 b) Reduced Order Observer

In[ ]:=

```
p = 1;
n = 2;
Print["x = ", MatrixForm[x[t]]]
xv1[t_] := Take[x[t], p];
xv2[t_] := Take[x[t], -(n - p)];
Print["xv1 = ", MatrixForm[xv1[t]], "\t xv2 = ", MatrixForm[xv2[t]]]


A11 = Take[A, p, p];
A12 = Take[A, p, {p + 1, n}];
A21 = Take[A, {p + 1, n}, p];
A22 = Take[A, {p + 1, n}, {p + 1, n}];
B1 = Take[B, p];
B2 = Take[B, -{n - p}];


Print["A11 = ", MatrixForm[A11], "\t A12 = ", MatrixForm[A12],
 "\t A21 = ", MatrixForm[A21], "\t A22 = ", MatrixForm[A22],
 "\t B1=", MatrixForm[B1], "\t B2=", MatrixForm[B2]]
```

x = $\begin{pmatrix} x1[t] \\ x2[t] \end{pmatrix}$

xv1 = ( x1[t] )    xv2 = ( x2[t] )

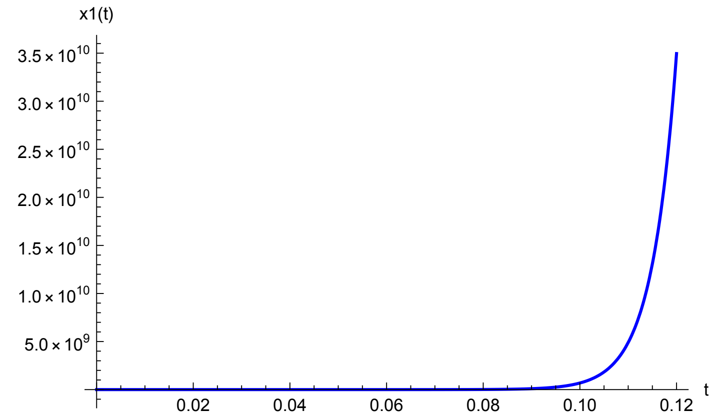
A11 = ( 0 )    A12 = ( 1 )    A21 = ( 38 622. )
A22 = ( 0 )    B1=( 0 )    B2=( 0.0368666 )

In[ ]:=
```
Clear[lred]
I1 = IdentityMatrix[n - p];
aob[s_] := Det[s I1 - A22 + {lred}.A12]
aob[s]
```
Out[ ]=
```
lred + s
```
In[ ]:=
```
SolRo = Solve[aob[s] == s + α1, lred] // Flatten
```
Out[ ]=
```
{lred → α1}
```

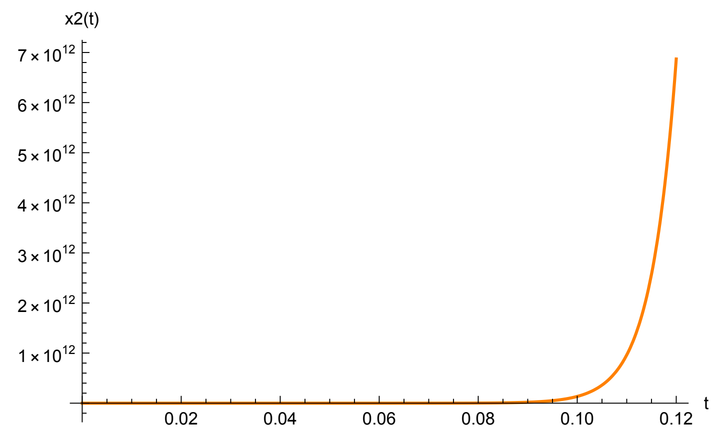
## Reduced Observer

In[ ]:=
```
Lred = {lred} /. SolRo
Ar = A22 - Lred.A12; MatrixForm[Ar];
xro[t_] = {xo2[t]};
xhat[t_] := {xv1[t], xro[t]} // Flatten
xhat[t]
u[t_] := {u1[t]};
yr[t_] := xv1'[t] - A11.xv1[t] - B1.u[t]
zr[t_] := A21.xv1[t] + B2.u[t]
EqRedo = Thread[xro'[t] == Ar.xro[t] + Lred.yr[t] + zr[t]] // Chop;

EqRedo
```
Out[ ]=
```
{α1}
```
Out[ ]=
```
{x1[t], xo2[t]}
```
Out[ ]=
```
{xo2′[t] == 0.0368666 u1[t] + 38 622. x1[t] - α1 xo2[t] + α1 x1′[t]}
```
In[ ]:=
```
ICro = {xo2[0] == 0};
u[t_] := {u1[t]}
BC = {u1[t] → 1, α1 → 70}; tmax = .12;
ObResponse = NDSolve[{EqOL /. BC, EqRedo /. BC, IC, ICro},
   {x[t], xro[t]} // Flatten, {t, 0, tmax}];

Plot[Evaluate[{x1[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x1(t)"}, PlotRange → All, PlotStyle → {Blue}]
Plot[Evaluate[{x2[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x2(t)"}, PlotRange → All, PlotStyle → {Orange}]
Plot[Evaluate[{x2[t] - xo2[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x2(t)-xo2(t)"}, PlotRange → All, PlotStyle → {Red}]
```

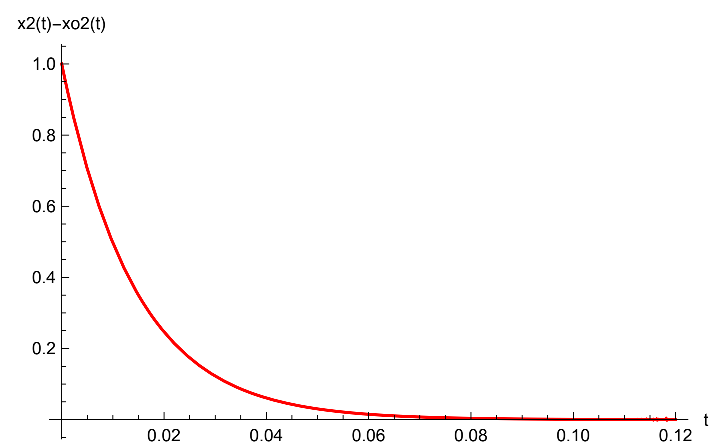
Out[ ]=




Out[ ]=




Out[ ]=




# Q4.1 C)

# Combined observer-controller system with the full-order observer

Let’s call desired poles as pole1 and pole2. Then, desired characteristic polynomial would be:

In[∘]:=
```
αpd[s] = (s + pole1) (s + pole2) // Expand // N
```
Out[∘]=
```
pole1 pole2 + pole1 s + pole2 s + s^2
```
In[∘]:=
```
αc =
 Reverse[Drop[CoefficientList[αpd[s], s], -1]] // Chop // Rationalize // Simplify
ac = ac // Round
KK = {(αc - ac).Inverse[Ut].Inverse[P]} // Simplify // Rationalize // Chop
P
```
Out[∘]=
```
{pole1 + pole2, pole1 pole2}
```
Out[∘]=
```
{0, -38622}
```
Out[∘]=
```
{{1.04761×10^6 + 27.1248 pole1 pole2, 27.1248 (pole1 + pole2)}}
```
Out[∘]=
```
{{0, 0.0368666}, {0.0368666, 0.}}
```
Check Eigenvalues:

In[∘]:=
```
A - B.KK // Simplify
```
Out[∘]=
```
{{0, 1.}, {0.0472441 - 1. pole1 pole2, -1. (pole1 + pole2)}}
```
In[∘]:=
```
Eigenvalues[A - B.KK // Simplify] // Chop // N // Rationalize
```
Out[∘]=

$$\left\{\frac{1}{2}\left(-\text{pole1}-\text{pole2}-\sqrt{0.188976+\text{pole1}^2-2\,\text{pole1 pole2}+\text{pole2}^2}\right),\ \frac{1}{2}\left(-\text{pole1}-\text{pole2}+\sqrt{0.188976+\text{pole1}^2-2\,\text{pole1 pole2}+\text{pole2}^2}\right)\right\}$$

There is a problem. //Chop does not work.

In[∘]:=
```
F = Inverse[-CC.Inverse[A - B.KK].B];
EqObsController = Thread[x'[t] == A.x[t] - B.Ksf.xo[t] + B.F.{v[t]}] // Chop;
u[t_] := F.{v[t]} - Ksf.xo[t]
EqObserver = Thread[xo'[t] == Ac.xo[t] + L.y[t] + B.u[t]] // Chop;
TableForm[AllEqn = {EqObsController, EqObserver} // Flatten]
```
Out[∘]//TableForm=
```
x1′[t] == x2[t]
x2′[t] == 1. (-0.0472441 + 1. pole1 pole2) v[t] + 38622. x1[t] - 1. (38622. + p1 p2) xo1[t] - 1
xo1′[t] == (α1 + α2) x1[t] + (-α1 - α2) xo1[t] + xo2[t]
```
$$\text{xo2}'[t] == \left(\frac{4905000}{127}+\alpha1\,\alpha2\right)\text{x1}[t]-\alpha1\,\alpha2\,\text{xo1}[t]+0.0368666\,(27.1248\,(-0.0472441+1.\text{ pole1 pc}$$

In[ ]:=

```
H[s] = -KK.Inverse[s I2 - (A - B.KK - L.Cm)].L /.
    {pole1 → 90, pole2 → 110, α1 → 300, α2 → 300} // Simplify
```

Out[ ]=

$$\left\{\left\{\frac{-2.95\times10^{11}-1.48746\times10^{9}\,s}{258522.+800.\,s+s^2}\right\}\right\}$$

In[ ]:=

```
DesInput = {v[t] → 1, pole1 → 80, pole2 → 120, α1 → 300, α2 → 300};
tmax = 0.12
ObsContrResponse =
  NDSolve[{AllEqn /. DesInput, IC , ICo}, {x[t], xo[t]} // Flatten, {t, 0, tmax}];
Plot[Evaluate[{x1[t], xo1[t]} /. ObsContrResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x₁(t), x̂₁(t)"}, PlotRange → All]
Plot[Evaluate[{x2[t], xo2[t]} /. ObsContrResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x₂(t), x̂₂(t)"}, PlotRange → All]
```

Out[ ]=

0.12

NDSolve : Encountered non −numerical value for a derivative at t == 0.`.

ReplaceAll :

{NDSolve [{{x1′[t] == x2[t], x2′[t] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[t] == 600 x1[≪1≫] − 600 xo1 [≪1≫] + xo2 [t], xo2′[t] == $\frac{16335000}{127}$ x1[≪1≫] − 90000 xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0] == 4, x2 [0] == 1}, {xo1 [0] == 0, xo2 [0] == 0}}, {x1[t], x2 [t], xo1 [t], xo2 [t]}, {t, 0, 0.12 }]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 2.4514285714285715` *^−6 cannot be used as a variable.

ReplaceAll :

{NDSolve [{{x1′[$2.45143\times10^{-6}$] == x2[$2.45143\times10^{-6}$], x2′[$2.45143\times10^{-6}$] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[$2.45143\times10^{-6}$] == 600 x1[≪1≫] − 600 xo1 [≪1≫] + xo2 [$2.45143\times10^{-6}$], xo2′[$2.45143\times10^{-6}$] == $\frac{16335000}{127}$ x1[≪1≫] − 90000 xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0] == 4, x2 [0] == 1}, {xo1 [0] == 0, xo2 [0] == 0}}, {x1[$2.45143\times10^{-6}$], x2[$2.45143\times10^{-6}$], xo1 [$2.45143\times10^{-6}$], xo2 [$2.45143\times10^{-6}$]}, {$2.45143\times10^{-6}$, 0, 0.12 }]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 2.4514285714285715` *^−6 cannot be used as a variable.

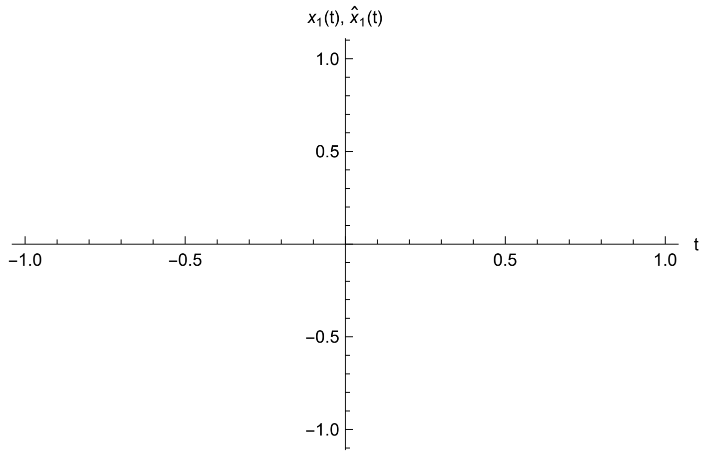
ReplaceAll : {NDSolve [{{x1′[2.45143 ×10^−6] == x2[2.45143 ×10^−6], x2′[2.45143 ×10^−6] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[2.45143 ×10^−6] == 600. x1 [≪1≫] − 600. xo1 [≪1≫] + xo2 [2.45143 ×10^−6], xo2′[2.45143 ×10^−6] == 128622. x1 [≪1≫] − 90000. xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0.] == 4., x2 [0.] == 1.}, {xo1 [0.] == 0., xo2 [0.] == 0.}}, {x1[2.45143 ×10^−6], x2[2.45143 ×10^−6], xo1 [2.45143 ×10^−6], xo2 [2.45143 ×10^−6]}, {2.45143 ×10^−6, 0., 0.12 }]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.

NDSolve : 0.002451431020408163` cannot be used as a variable.

General : Further output of NDSolve::dsvar will be suppressed during this calculation.

*Out[∘ ]=*




NDSolve : Encountered non −numerical value for a derivative at t == 0.`.

ReplaceAll : {NDSolve [{{x1′[t] == x2[t], x2′[t] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[t] == 600 x1 [≪1≫] − 600 xo1 [≪1≫] + xo2 [t], xo2′[t] == $\frac{16335000}{127}$ x1[≪1≫] − 90000 xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0] == 4, x2 [0] == 1}, {xo1 [0] == 0, xo2 [0] == 0}}, {x1[t], x2 [t], xo1 [t], xo2 [t]}, {t, 0, 0.12 }]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 2.4514285714285715` *^−6 cannot be used as a variable.

ReplaceAll : {NDSolve [{{x1′[2.45143 ×10^−6] == x2[2.45143 ×10^−6], x2′[2.45143 ×10^−6] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[2.45143 ×10^−6] == 600 x1 [≪1≫] − 600 xo1 [≪1≫] + xo2 [2.45143 ×10^−6], xo2′[2.45143 ×10^−6] == $\frac{16335000}{127}$ x1[≪1≫] − 90000 xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0] == 4, x2 [0] == 1}, {xo1 [0] == 0, xo2 [0] == 0}}, {x1[2.45143 ×10^−6], x2[2.45143 ×10^−6], xo1 [2.45143 ×10^−6], xo2 [2.45143 ×10^−6]}, {2.45143 ×10^−6, 0, 0.12 }]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 2.4514285714285715\` *^−6 cannot be used as a variable.

ReplaceAll :

{NDSolve [{{x1′[2.45143 ×10^−6] == x2[2.45143 ×10^−6], x2′[2.45143 ×10^−6] == 1. p1 p2 + 38622. x1 [≪1≫] − 1. Plus [≪2≫] xo1 [≪1≫] − 1. Plus [≪2≫] xo2 [≪1≫], xo1′[2.45143 ×10^−6] == 600. x1 [≪1≫] − 600. xo1 [≪1≫] + xo2 [2.45143 ×10^−6], xo2′[2.45143 ×10^−6] == 128622. x1 [≪1≫] − 90000. xo1 [≪1≫] + 0.0368666 Plus [≪3≫]}, {x1[0.] == 4., x2 [0.] == 1.}, {xo1 [0.] == 0., xo2 [0.] == 0.}}, {x1[2.45143 ×10^−6], x2[2.45143 ×10^−6], xo1 [2.45143 ×10^−6], xo2 [2.45143 ×10^−6]}, {2.45143 ×10^−6, 0., 0.12 }]}

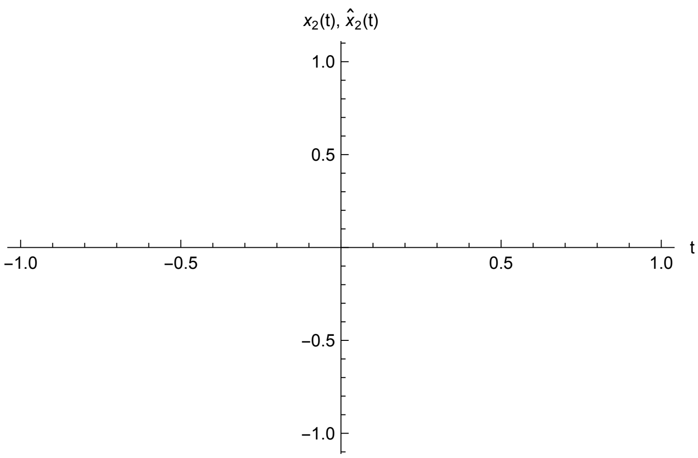
is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.

NDSolve : 0.002451431020408163\` cannot be used as a variable.

General : Further output of NDSolve::dsvar will be suppressed during this calculation.

Out[ ]=




There was a problem in truncating values. I saw it while I was checking the eigenvalues after placing the poles. Chop function did not work. AllEqn function came probably

# wrong. Thus I will define most of the parameters in parametric version to see if I can get the plot:

In[ ]:=
```
Quit[]
```

In[ ]:=

$$A = \begin{pmatrix} 0 & 1 \\ \frac{kx}{m} & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ \frac{ki}{m} \end{pmatrix}$$

$$CC = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

Out[ ]=

$$\left\{\{0, 1\}, \left\{\frac{kx}{m}, 0\right\}\right\}$$

Out[ ]=

$$\left\{\{0\}, \left\{\frac{ki}{m}\right\}\right\}$$

Out[ ]=

$$\{\{1, 0\}\}$$

In[ ]:=
```
P = Join[B, A.B, 2];
```

In[ ]:=
```
αpd[s] = (s + p2) (s + p1) // Expand // N
```

Out[ ]=

$$p1\, p2 + p1\, s + p2\, s + s^2$$

In[ ]:=

$$Ut = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};$$

In[ ]:=
```
αc = Reverse[Drop[CoefficientList[αp[s], s], -1]];
KK = {(αc - ac).Inverse[Ut].Inverse[P]}
```

Out[ ]=

$$\left\{\left\{\frac{m\,(-ac + p1\, p2)}{ki}, \frac{m\,(-ac + p1 + p2)}{ki}\right\}\right\}$$

In[ ]:=
```
I2 = IdentityMatrix[2];
a[s] = Det[s I2 - A] // Expand
ac = Reverse[Drop[CoefficientList[a[s], s], -1]]
```

Out[ ]=

$$-\frac{kx}{m} + s^2$$

Out[ ]=

$$\left\{0, -\frac{kx}{m}\right\}$$

In[ ]:=
```
αc = Reverse[Drop[CoefficientList[αpd[s], s], -1]]
KK = {(αc - ac).Inverse[Ut].Inverse[P]}
```

Out[ ]=

$$\{p1 + p2, p1\, p2\}$$

Out[ ]=

$$\left\{\left\{\frac{m\left(\frac{kx}{m} + p1\, p2\right)}{ki}, \frac{m\,(p1 + p2)}{ki}\right\}\right\}$$

In[ ]:=
```
Eigenvalues[A - B.KK]
```

Out[ ]=

$$\{-p1, -p2\}$$

# Now we placed the eigenvalues correctly.

In[ ]:=
```
μo = 0.000001258
Ap = 0.000146
ho = 0.000508
m = 0.3
Nc = 100
g = 9.81
```

Out[ ]=

$1.258 \times 10^{-6}$

Out[ ]=

0.000146

Out[ ]=

0.000508

Out[ ]=

0.3

Out[ ]=

100

Out[ ]=

9.81

In[ ]:= $\alpha = \mu o \, Nc \, Ap^2$

Out[ ]=

$2.68155 \times 10^{-12}$

In[ ]:= $Io = \sqrt{m \, g \, \frac{ho^2}{\alpha}}$

Out[ ]=

532.189

In[ ]:= $kx = 2 \frac{\alpha \, Io^2}{ho^3}$

$ki = \text{Simplify}\left[2 \alpha \frac{Io}{ho^2}\right]$

Out[ ]=

11586.6

Out[ ]=

0.01106

In[ ]:=
```
F = Inverse[-CC.Inverse[A - B.KK].B];
EqObsController = Thread[x'[t] == A.x[t] - B.Ksf.xo[t] + B.F.{v[t]}] // Chop;
u[t_] := F.{v[t]} - KK.xo[t]
EqObserver = Thread[xo'[t] == Ac.xo[t] + L.y[t] + B.u[t]];
TableForm[AllEqn = {EqObsController, EqObserver} // Flatten // Chop // Simplify]
```

Out[ ]//TableForm=
```
{{0, 0}, {38622. + 1. p1 p2, 1. (p1 + p2)}}.xo[t] + x′[t] == {{0, 1}, {38622., 0}}.x[t]
{{0, 0}, {38622. + 1. p1 p2, 1. (p1 + p2)}}.xo[t] + x′[t] == {{0, 1}, {38622., 0}}.x[t] + 1. p1
Ac.xo[t] + {{1. (α1 + α2)}, {38622. + 1. α1 α2}}.y[t] == xo′[t]
xo′[t] == Ac.xo[t] - 0.0368666 {{1.04762 × 10^6 + 27.1248 p1 p2, 27.1248 (p1 + p2)}}.xo[t] + {{1
```

In[ ]:=
```
Om = Join[Cmᵀ, Aᵀ.Cmᵀ, 2]ᵀ;
αo[s] = (s + α1) (s + α2) // Expand // N
αoc = Reverse[Drop[CoefficientList[αo[s], s], -1]];
ac = Reverse[Drop[CoefficientList[a[s], s], -1]];
Lt = {(αoc - ac).Inverse[Omᵀ.Ut]};
L = Ltᵀ // Chop // Simplify
```

Out[ ]=

$$s^2 + s\,\alpha 1 + s\,\alpha 2 + \alpha 1\,\alpha 2$$

Out[ ]=

$$\left\{\{\alpha 1 + \alpha 2\},\ \left\{\frac{2\,g}{ho} + \alpha 1\,\alpha 2\right\}\right\}$$

In[ ]:=
```
H[s] = -KK.Inverse[s I2 - (A - B.KK - L.CC)].L /.
    {p1 → 110, p2 → 80, α1 → 200, α2 → 200} // Simplify
```

Out[ ]=

$$\left\{\left\{\frac{-1.80751\times 10^{11} - 9.19721\times 10^{8}\,s}{163422. + 590.\,s + s^2}\right\}\right\}$$

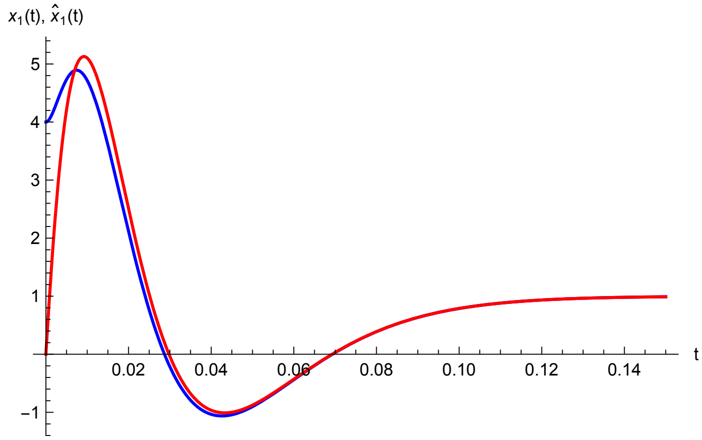
In[ ]:=
```
DesInput = {v[t] → 1, p1 → 80, p2 → 70, α1 → 200, α2 → 100};
tmax = 0.15
ObsContrResponse =
  NDSolve[{AllEqn /. DesInput, IC , ICo}, {x[t], xo[t]} // Flatten, {t, 0, tmax}];
Plot[Evaluate[{x1[t], xo1[t]} /. ObsContrResponse], {t, 0, tmax},
 AxesLabel → {"t", "x₁(t), x̂₁(t)"}, PlotRange → All, PlotStyle → {Blue, Red}]
Plot[Evaluate[{x2[t], xo2[t]} /. ObsContrResponse], {t, 0, tmax},
 AxesLabel → {"t", "x₂(t), x̂₂(t)"}, PlotRange → All, PlotStyle → {Blue, Red}]
```

```
Ksf = KK
```

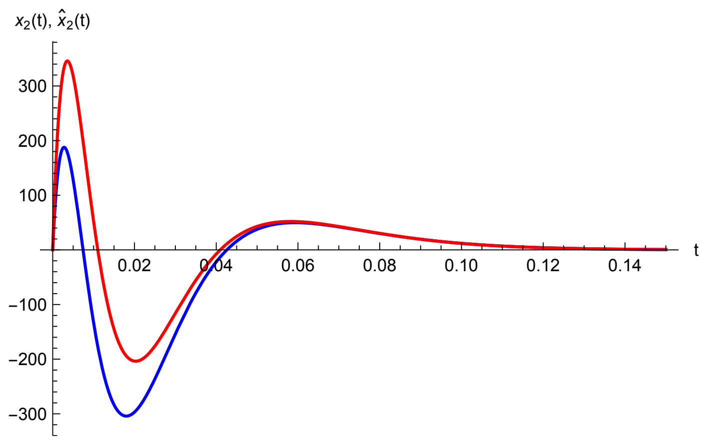
Out[∘]=

```
0.15
```

Out[∘]=



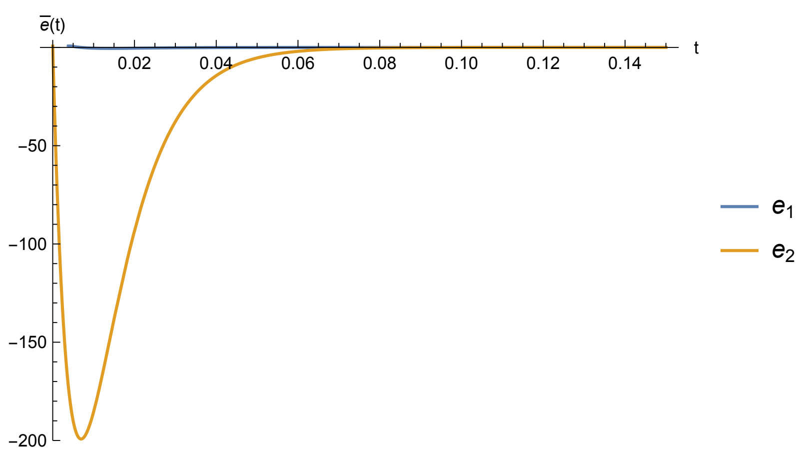

Out[∘]=




In[∘]:=

```
Plot[Evaluate[{x[t] - xo[t]} /. ObsContrResponse], {t, 0, tmax},
 AxesLabel → {"t", "e̅(t)"}, PlotRange → {-200, 1}, PlotLegends → {e1, e2, e3}]
```

Out[∘]=



```
Ksf = KK;
```

In[ ]:=
```
Vals = {g → 9.81, μo → 0.000001258, Ap → 0.000146, ho → 0.000508, m → 0.3, Nc → 100};
Inputs = {v[t] → 1, α1 → 100, p1 → 30, p2 → 30};
EqRedObsFdbk =
  Thread[x'[t] == Flatten[A.x[t] - B.Ksf.xhat[t] + B.F v[t]] /. Vals /. Inputs];
u[t_] := F v[t] - Ksf.xhat[t]
EqRedObserver =
  Thread[xro'[t] == Flatten[Ar.xro[t] + Lr.yr[t] + zr[t]] /. Vals /. Inputs] // Flatten;
ColumnForm[EqRedObsFdbk]
ColumnForm[EqRedObserver]
```

Out[ ]=
```
x1′[t] == 0. + x2[t]
x2′[t] == 900. - 900. x1[t] - 60 xo2[t]
```

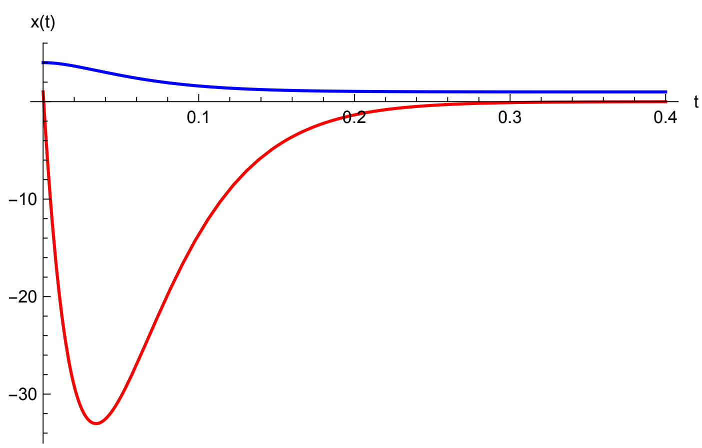
Out[ ]=
```
xo2′[t] == 38622. x1[t] + 0.0368666 (24412.3 - 1.07203×10^6 x1[t] - 1627.49 xo2[t]) - 100 xo2
```

In[ ]:=
```
A.x[t] - B.Ksf.xhat[t] + B.F v[t] /. Vals /. Inputs
```

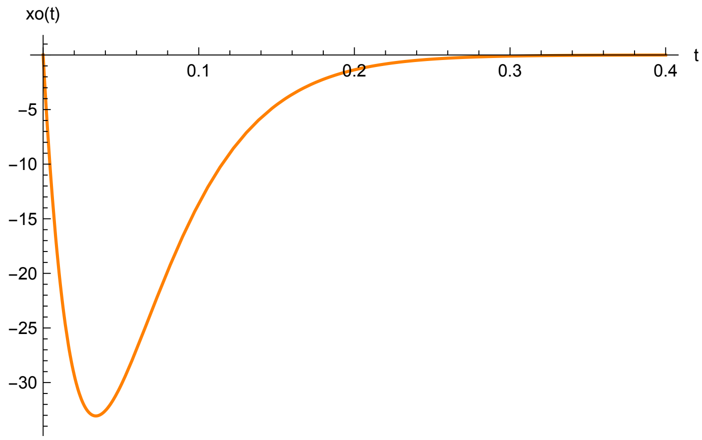
Out[ ]=
```
{{0. + x2[t]}, {900. - 900. x1[t] - 60 xo2[t]}}
```

In[ ]:=
```
tmax = 0.4;
RedObResponse = NDSolve[{EqRedObsFdbk /. Inputs, EqRedObserver /. Inputs, IC, ICo},
   {x[t], xro[t]} // Flatten, {t, 0, tmax}];
Plot[Evaluate[{x[t]} /. RedObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x(t)"}, PlotRange → All, PlotStyle → {Blue, Red}]
Plot[Evaluate[{xo[t]} /. RedObResponse], {t, 0, tmax},
 AxesLabel → {"t", "xo(t)"}, PlotRange → All, PlotStyle → {Blue, Orange}]
Plot[Evaluate[{x1[t] - xo2[t]} /. RedObResponse], {t, 0, tmax},
 AxesLabel → {"t", "x̄1(t)- x̄o1(t)"}, PlotRange → All, PlotStyle → {Blue}]
```

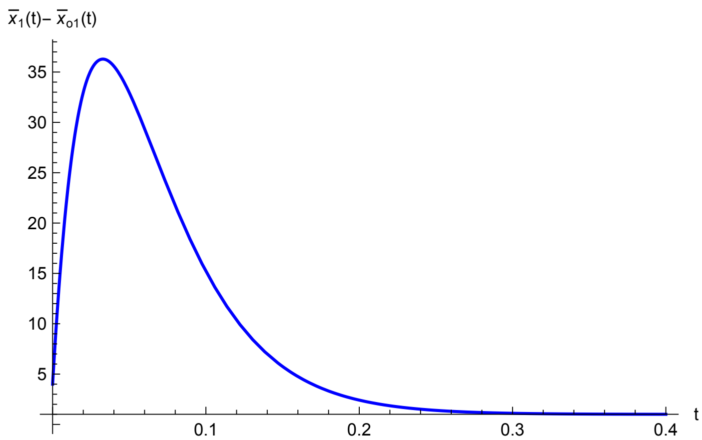
Out[ ]=



Out[ ]=


xo(t)
t
0.1
0.2
0.3
0.4
−5
−10
−15
−20
−25
−30

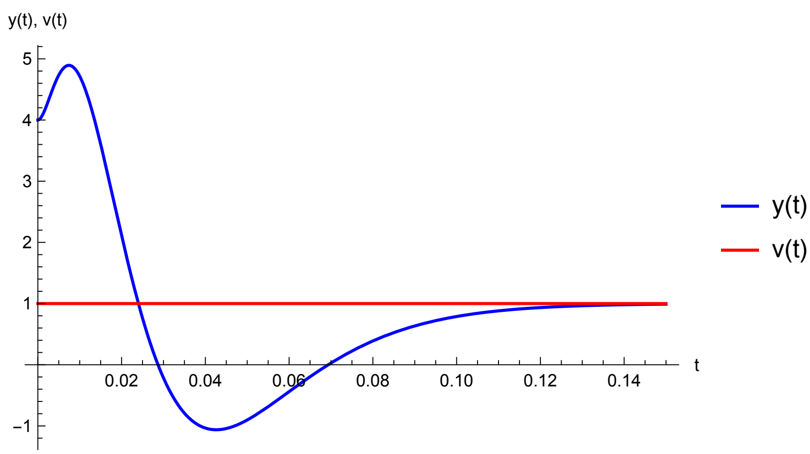
Out[ ]=


$\overline{x}_1$(t)− $\overline{x}_{o1}$(t)
35
30
25
20
15
10
5
t
0.1
0.2
0.3
0.4

# output

In[•]:=
```
Plot[Evaluate[{y[t] /. ObsContrResponse, v[t] /. DesInput}],
 {t, 0, tmax}, AxesLabel → {"t", "y(t), v(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red, {Dashed, Blue}, {Dashed, Red}},
 PlotLegends → {"y(t)", "v(t)"} ]
```

Out[•]=




In[•]:=
```
Inputs = {v[t] → 1, α1 → 100, p1 → 30, p2 → 30};
EqRedObsFdbk =
  Thread[x'[t] == Flatten[A.x[t] - B.Ksf.xhat[t] + B.F v[t]] /. Vals /. Inputs];
u[t_] := F v[t] - Ksf.xhat[t]
EqRedObserver =
  Thread[xro'[t] == Flatten[Ar.xro[t] + Lr.yr[t] + zr[t]] /. Vals /. Inputs] // Flatten;
ColumnForm[EqRedObsFdbk]
ColumnForm[EqRedObserver]
```

Out[•]=

```
x1′[t] == 0. + x2[t]
x2′[t] == 900. - 900. x1[t] - 60 xo2[t]
```

Out[•]=

$$xo2'[t] == 38622.\, x1[t] + 0.0368666\left(24412.3 - 1.07203\times10^{6}\, x1[t] - 1627.49\, xo2[t]\right) - 100\, xo2$$

# Q4 .4

```
Quit[]
```

In[ ]:=

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$Cm = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$W = \sigma \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

```
Θ = {{1}}
```

Out[ ]=
```
{{1, 1}, {0, 1}}
```

Out[ ]=
```
{{0}, {1}}
```

Out[ ]=
```
{{1, 0}}
```

Out[ ]=
```
{{σ, 0}, {0, σ}}
```

Out[ ]=
```
{{1}}
```

## a) Kalman Filter Gains

$\sigma = 2$

In[ ]:=
```
W1 = W /. σ → 2.;
Σ1 = RiccatiSolve[{Aᵀ, Cmᵀ}, {W1, Θ}];
Print["Σ₁ = ", Σ1]
L1 = Σ1 .Cmᵀ.Inverse[Θ];
Print["L₁ = ", L1]
```

```
Σ₁ = {{4.91069, 6.14676}, {6.14676, 17.8913}}
L₁ = {{4.91069}, {6.14676}}
```

$\sigma = 0.2$

In[ ]:=
```
W2 = W /. σ → 0.2;
Σ2 = RiccatiSolve[{Aᵀ, Cmᵀ}, {W2, Θ}];
Print["Σ₂ = ", Σ2]
L2 = Σ2.Cmᵀ.Inverse[Θ];
Print["L₂ = ", L2]
```

```
Σ₂ = {{4.13692, 4.32014}, {4.32014, 9.23179}}
L₂ = {{4.13692}, {4.32014}}
```

# b)

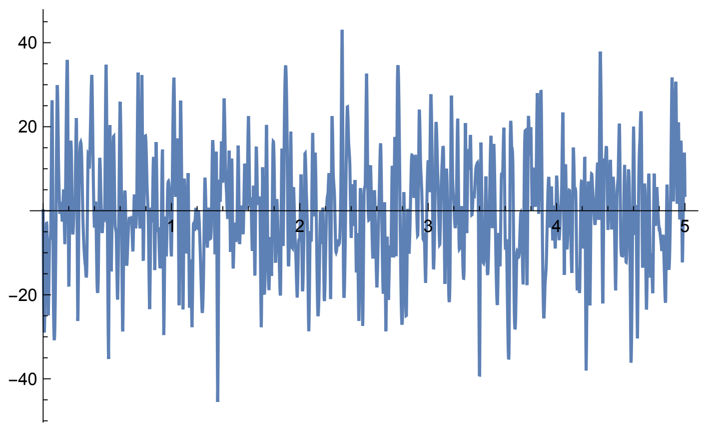
$\sigma = 2$

In[∘]:=
```
tmax = 5; step = 0.01; σg = Sqrt[W1[[1, 1]]/step]
g1 = Interpolation[Thread[{Range[0, tmax, step],
     Join[{0}, RandomReal[NormalDistribution[0, σg], tmax / step]]}], t];
Plot[g1, {t, 0, tmax}]
```

Out[∘]=

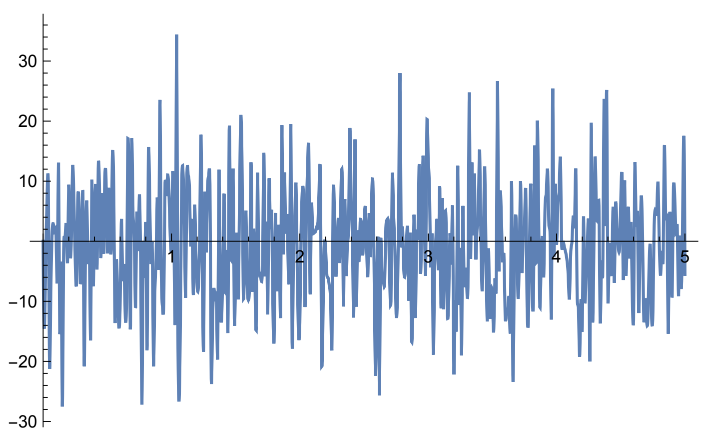
14.1421

Out[∘]=




In[∘]:=
```
σθ = Sqrt[1/step];
θ = Interpolation[Thread[{Range[0, tmax, step],
     Join[{0}, RandomReal[NormalDistribution[0, σθ], tmax / step]]}], t];
Plot[θ, {t, 0, tmax}]
```

Out[∘]=




Open Loop :

System Eqs :

In[ ]:=
```
Clear[x, x1, x2]
x[t_] := {x1[t], x2[t]}; y[t_] := Cm.x[t] + θ;
u[t_] := 0;
OpenLoopEq = Thread[x'[t] == A.x[t] + B.{u[t]} + (1
                                                   1).{g1}];
IC = {x1[0] == 0, x2[0] == 0};
```

Observer eqs :

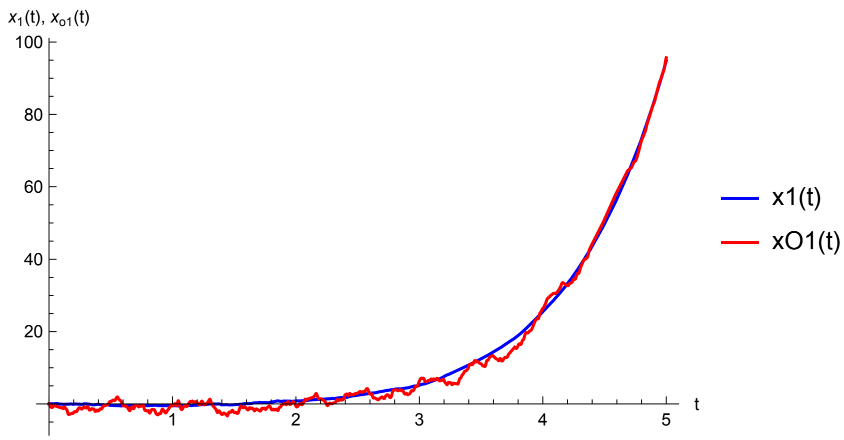
In[ ]:=
```
xo[t_] := {xo1[t], xo2[t]}; ICo = {xo1[0] == 0, xo2[0] == 0};
EqObserver =
  Thread[xo'[t] == (A - L1.Cm).xo[t] + L1.y[t] + B.{u[t]}] // Chop // Flatten;
TableForm[EqObserver]
```

Out[ ]//TableForm=
```
xo1′[t] == -3.91069 xo1[t] + 1. xo2[t] + 4.91069 (x1[t] + InterpolatingFunction[ Domai
                                                                                  Outpu

xo2′[t] == Sin[3.14159 t] - 6.14676 xo1[t] + 1. xo2[t] + 6.14676 (x1[t] + InterpolatingFuncti
```

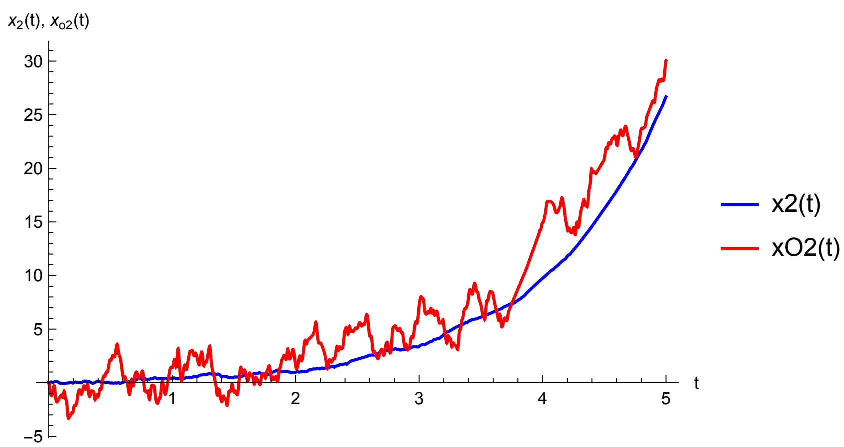
System Response :

In[∘]:=
```
ObResponse = NDSolve[{OpenLoopEq, EqObserver, IC, ICo},
    {x[t], xo[t]} // Flatten, {t, 0, tmax}, MaxSteps → 10^6];
Plot[Evaluate[{x1[t], xo1[t]} /. ObResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x1(t), xo1(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red},  PlotLegends → {"x1(t)", "xO1(t)"}]
Plot[Evaluate[{x2[t], xo2[t]} /. ObResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x2(t), xo2(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red}, PlotLegends → {"x2(t)", "xO2(t)"}]
```

Out[∘]=



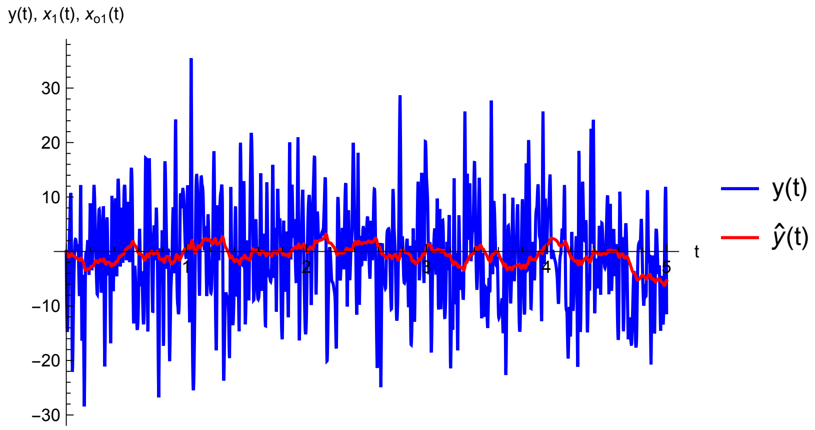

Out[∘]=




Output

In[ ]:=
```
Plot[Evaluate[{y[t], Cm.xo[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "y(t), x1(t), xo1(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red, Green}, PlotLegends → {"y(t)", "ŷ(t)"}]
```

Out[ ]=



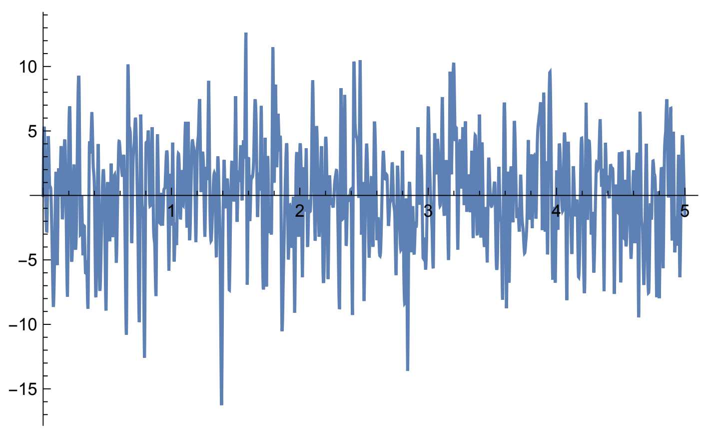

$\sigma = 0.2$

In[ ]:=
```
tmax = 5; step = 0.01; σg = Sqrt[W2[[1, 1]]/step]
g2 = Interpolation[Thread[{Range[0, tmax, step],
      Join[{0}, RandomReal[NormalDistribution[0, σg], tmax / step]]}], t];
Plot[g2, {t, 0, tmax}]
```

Out[ ]=

4.47214

Out[ ]=




Open Loop :

In[∘]:=
```
Clear[x, x1, x2]
x[t_] := {x1[t], x2[t]}; y[t_] := Cm.x[t] + θ;
u[t_] := Sin[2 π 0.5 t];
OLEq = Thread[x'[t] == A.x[t] + B.{u[t]} + (1
                                           1).{g2}];
IC = {x1[0] == 0, x2[0] == 0};
TableForm[OLEq]
```

Out[∘]//TableForm=

```
x1′[t] == x1[t] + x2[t] + InterpolatingFunction[ Domain: {{0., 5. }} Output: scalar ][t]

x2′[t] == Sin[3.14159 t] + x2[t] + InterpolatingFunction[ Domain: {{0., 5. }} Output: scalar ][t]
```

Observer eqns :

In[∘]:=
```
xo[t_] := {xo1[t], xo2[t]}; ICo = {xo1[0] == 0, xo2[0] == 0};
EqObserver =
  Thread[xo'[t] == (A - L1.Cm).xo[t] + L1.y[t] + B.{u[t]}] // Chop // Flatten;
TableForm[EqObserver]
```

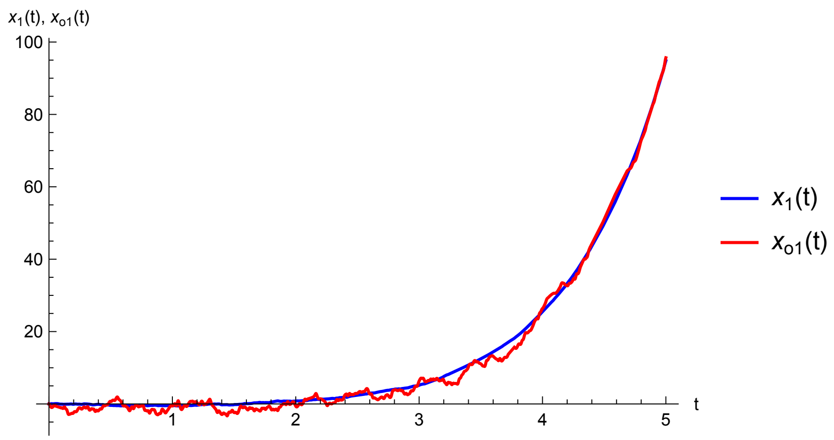
Out[∘]//TableForm=

```
xo1′[t] == -3.91069 xo1[t] + 1. xo2[t] + 4.91069 (x1[t] + InterpolatingFunction[ Domai Outpu

xo2′[t] == Sin[3.14159 t] - 6.14676 xo1[t] + 1. xo2[t] + 6.14676 (x1[t] + InterpolatingFuncti
```

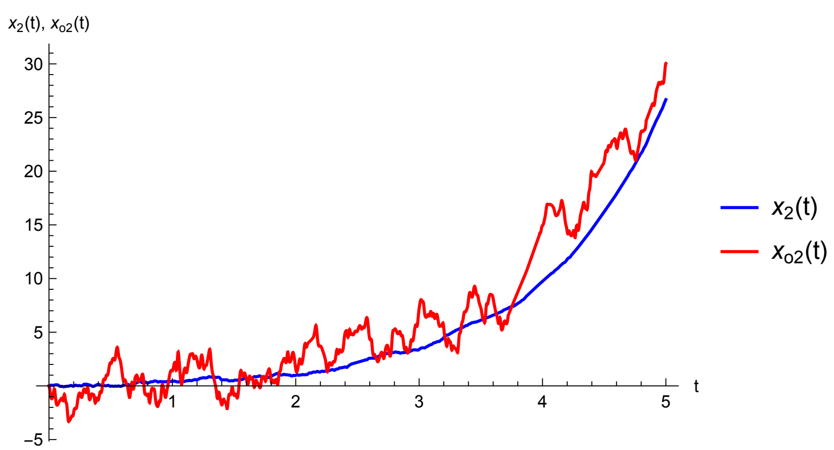
Response :

In[ ]:=
```
ObResponse = NDSolve[{OLEq, EqObserver, IC, ICo},
   {x[t], xo[t]} // Flatten, {t, 0, tmax}, MaxSteps → 10^6];
Plot[Evaluate[{x1[t], xo1[t]} /. ObResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x1(t), xo1(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red}, PlotLegends → {"x1(t)", "xo1(t)"}]
Plot[Evaluate[{x2[t], xo2[t]} /. ObResponse],
 {t, 0, tmax}, AxesLabel → {"t", "x2(t), xo2(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red}, PlotLegends → {"x2(t)", "xo2(t)"}]
```

Out[ ]=



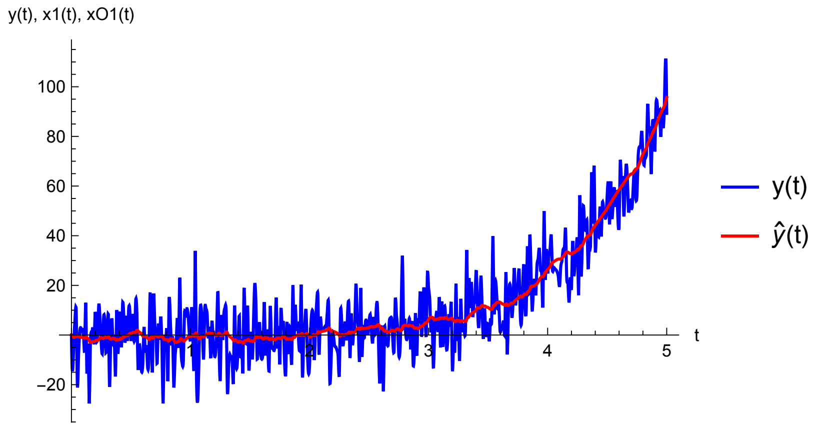

Out[ ]=




Output:

In[ ]:=
```
Plot[Evaluate[{y[t], Cm.xo[t]} /. ObResponse], {t, 0, tmax},
 AxesLabel → {"t", "y(t), x1(t), xO1(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red, Green}, PlotLegends → {"y(t)", "ŷ(t)"}]
```

Out[ ]=




# Q 4.2

In[ ]:=
```
A = (0 1
     -3 4)
B = (0
     1)
Cm = (1 1)
```

Out[ ]=
```
{{0, 1}, {-3, 4}}
```

Out[ ]=
```
{{0}, {1}}
```

Out[ ]=
```
{{1, 1}}
```

In[ ]:=
```
Eigenvalues[A]
```

Out[ ]=
```
{3, 1}
```

Two eigvalues are positive. Unstable. Let’s check controllability and observability.

In[ ]:=
```
P = Join[B, A.B, 2]; MatrixRank[P]
Om = Join[Cmᵀ, Aᵀ.Cmᵀ, 2]ᵀ; MatrixRank[Om]
```

Out[ ]=
```
2
```

Out[ ]=
```
2
```

System is both controllable and observable.

In[ ]:=
```
I2 = IdentityMatrix[2];
a[s] = Det[s I2 - A] // Expand
```

Out[ ]=

$3 - 4 s + s^2$

Desired poles

In[ ]:=
```
αo[s] = (s + 4)^2 // Expand // N
```

Out[ ]=

$16. + 8. s + s^2$

In[ ]:=
```
Ut = (1 -4
      0  1);
αoc = Reverse[Drop[CoefficientList[αo[s], s], -1]];
ac = Reverse[Drop[CoefficientList[a[s], s], -1]];
Lt = {(αoc - ac).Inverse[Om^T.Ut]};
L = Lt^T
```

Out[ ]=

{{-0.125}, {12.125}}

Observer state matrix :

In[ ]:=
```
Ac = A - L.Cm // Simplify
```

Out[ ]=

{{0.125, 1.125}, {-15.125, -8.125}}

Check observer eigenvalues if it is put correctly:

In[ ]:=
```
Eigenvalues[A - L.Cm]
```

Out[ ]=

$\{-4. + 5.6244 \times 10^{-8} i, -4. - 5.6244 \times 10^{-8} i\}$

In[ ]:=
```
Chop[Eigenvalues[A - L.Cm], 10^-7]
```

Out[ ]=

{-4., -4.}

Observer eqs :

In[ ]:=
```
x[t_] := {x1[t], x2[t]};
y[t_] := Cm.x[t]
u[t_] := {u1[t]}
xo[t_] := {xo1[t], xo2[t]};
EqObserver = Thread[xo'[t] == Ac.xo[t] + L.y[t] + B.u[t]] // Chop // Flatten
```

Out[ ]=

```
{xo1'[t] == -0.125 (x1[t] + x2[t]) + 0.125 xo1[t] + 1.125 xo2[t],
 xo2'[t] == u1[t] + 12.125 (x1[t] + x2[t]) - 15.125 xo1[t] - 8.125 xo2[t]}
```

Combined Observer - Controller :

Design a closed - loop controller with poles at -1, and -2:

In[ ]:=
```
αp[s] = (s + 1) (s + 2) // Expand // N
```
Out[ ]=
$2. + 3.\ s + s^2$

In[ ]:=
```
αc = Reverse[Drop[CoefficientList[αp[s], s], -1]];
Ksf = {(αc - ac).Inverse[Ut].Inverse[P]}
```
Out[ ]=
```
{{-1., 7.}}
```

Check the eigenvalues of the closed-loop system:

In[ ]:=
```
Eigenvalues[A - B.Ksf]
```
Out[ ]=
```
{-2., -1.}
```

Combined Observer - Controller :

In[ ]:=
```
EqObsController = Thread[x'[t] == A.x[t] - B.Ksf.xo[t] + B.{v[t]}];
u[t_] := v[t] - Ksf.xo[t]
EqObserver = Thread[xo'[t] == Ac.xo[t] + L.y[t] + B.u[t]];
AllEqn = {EqObsController, EqObserver} // Flatten
```
Out[ ]=
```
{x1′[t] == 0. + x2[t], x2′[t] == v[t] - 3 x1[t] + 4 x2[t] + 1. xo1[t] - 7. xo2[t],
 xo1′[t] == -0.125 (x1[t] + x2[t]) + 0.125 xo1[t] + 1.125 xo2[t],
 xo2′[t] == v[t] + 12.125 (x1[t] + x2[t]) - 14.125 xo1[t] - 15.125 xo2[t]}
```

b) TF

In[ ]:=
```
H[s] = -Ksf.Inverse[s I2 - (A - B.Ksf - L.Cm)].L // Simplify
```
Out[ ]=
$$\left\{\left\{\frac{10. - 85.\ s}{14. + 15.\ s + s^2}\right\}\right\}$$

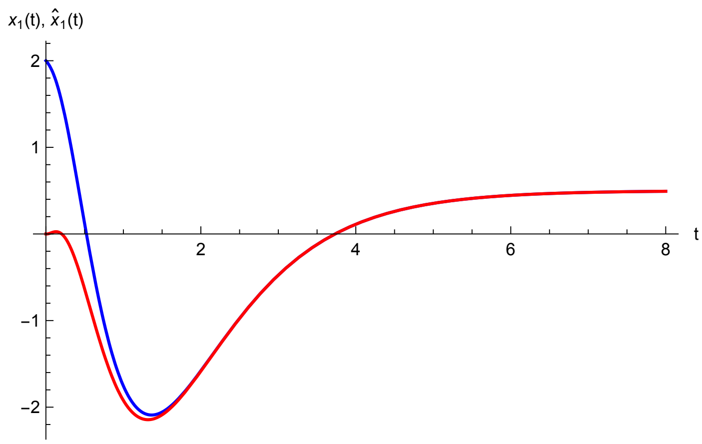
## c) System Response :

In[∘]:=
```
DesInput = {v[t] → 1};
IC = {x1[0] == 2, x2[0] == -1};
ICo = {xo1[0] == 0, xo2[0] == 0};
tmax = 8
ObsResponse =
  NDSolve[{AllEqn /. DesInput, IC , ICo}, {x[t], xo[t]} // Flatten, {t, 0, tmax}];
Plot[Evaluate[{x1[t], xo1[t]} /. ObsResponse], {t, 0, tmax},
 AxesLabel → {"t", "x₁(t), x̂₁(t)"}, PlotRange → All,  PlotStyle → {Blue, Red}]
Plot[Evaluate[{x2[t], xo2[t]} /. ObsResponse], {t, 0, tmax},
 AxesLabel → {"t", "x₂(t), x̂₂(t)"}, PlotRange → All, PlotStyle → {Blue, Red}]
```

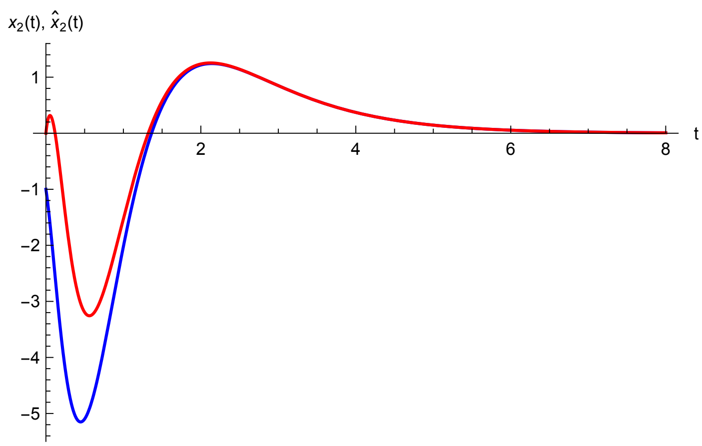
Out[∘]=

8

Out[∘]=




Out[∘]=



# Q 4 .5

In[ ]:=
```
Quit[]
```

In[ ]:=

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$CC = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$W = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$$

```
Θ = {{1}}
Q = CCᵀ.CC
R = {{1}}
```

Out[ ]=
```
{{1, 1}, {0, 1}}
```

Out[ ]=
```
{{0}, {1}}
```

Out[ ]=
```
{{1, 0}}
```

Out[ ]=
```
{{0.5, 0}, {0, 0.5}}
```

Out[ ]=
```
{{1}}
```

Out[ ]=
```
{{1, 0}, {0, 0}}
```

Out[ ]=
```
{{1}}
```

## ARE

In[ ]:=
```
Σ = RiccatiSolve[{Aᵀ, CCᵀ}, {W, Θ}]
```

Out[ ]=
```
{{4.30834, 4.72255}, {4.72255, 10.9012}}
```

In[ ]:=
```
L = Σ .CCᵀ.Inverse[Θ]
```

Out[ ]=
```
{{4.30834}, {4.72255}}
```

In[ ]:=
```
P = RiccatiSolve[{A, B}, {Q, R}] // N // Chop
```

Out[ ]=
```
{{10.1333, 4.61158}, {4.61158, 4.19737}}
```

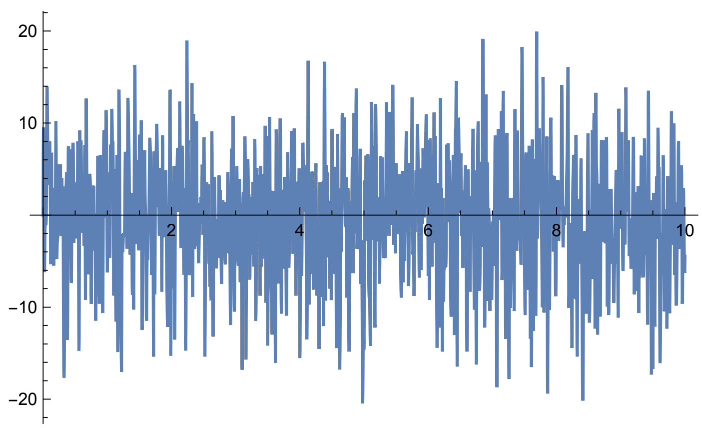
In[ ]:=
```
K = Inverse[R].Bᵀ.P
```
Out[ ]=
```
{{4.61158, 4.19737}}
```

Objective Function :

In[ ]:=
```
J = Tr[P.L.Θ.Lᵀ] + Tr[Σ.Q]
```
Out[ ]=
```
473.671
```

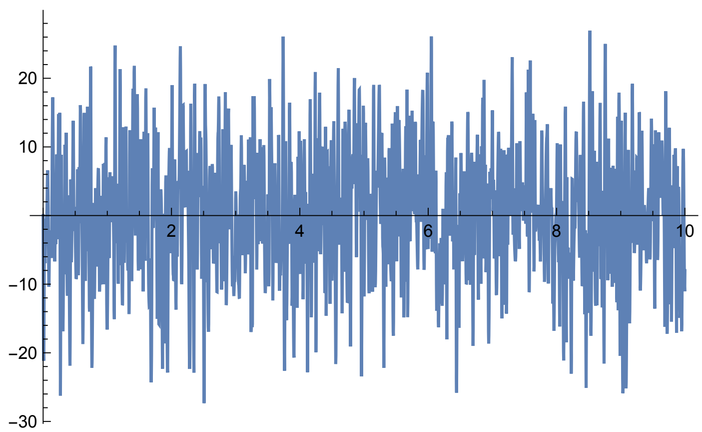
## Closed-loop:

In[ ]:=
```
tmax = 10; step = 0.01; σg = Sqrt[W⟦1, 1⟧ / step];
g = Interpolation[Thread[{Range[0, tmax, step],
     Join[{0}, RandomReal[NormalDistribution[0, σg], tmax / step]]}], t];
Plot[g, {t, 0, tmax}]
```
Out[ ]=



In[∘]:=
```
σθ = Sqrt[1/step];
θ = Interpolation[Thread[{Range[0, tmax, step],
      Join[{0}, RandomReal[NormalDistribution[0, σθ], tmax / step]]}], t];
Plot[θ, {t, 0, tmax}]
```

Out[∘]=



# Combined Observer - Controller :

In[∘]:=
```
Clear[x, x1, x2, u]
x[t_] := {x1[t], x2[t]}; y[t_] := CC.x[t] + θ;
r[t_] := 0;
xo[t_] := {xo1[t], xo2[t]}; ICo = {xo1[0] == 0, xo2[0] == 0};

EqObsController = Thread[x'[t] == A.x[t] - B.K.xo[t] + B.{v[t]} + {{1}, {1}}.{g}];
u[t_] := F.{v[t]} - K.xo[t]
EqObserver = Thread[xo'[t] == (A - L.CC).xo[t] + L.y[t] + B.u[t]] // Chop // Flatten;
TableForm[AllEqn = {EqObsController, EqObserver} // Flatten]
```

Out[∘]//TableForm=

```
x1′[t] == 0. + x1[t] + x2[t] + InterpolatingFunction[ Domain: {{0., 10. }} Output: scalar ][t]

x2′[t] == v[t] + x2[t] - 4.61158 xo1[t] - 4.19737 xo2[t] + InterpolatingFunction[ Domai Outpu

xo1′[t] == -3.30834 xo1[t] + 1. xo2[t] + 4.30834 (x1[t] + InterpolatingFunction[ Domai Outpu

xo2′[t] == F.{v[t]} - 9.33413 xo1[t] - 3.19737 xo2[t] + 4.72255 (x1[t] + InterpolatingFuncti
```

# Closed-Loop :

In[ ]:=
```
DesInput = {v[t] → r[t]}; IC = {x1[0] == 0, x2[0] == 0};
ObsContrResponse = NDSolve[{AllEqn /. DesInput, IC , ICo},
   {x[t], xo[t]} // Flatten, {t, 0, tmax}, StartingStepSize → 1 / 1000,
   Method → {"FixedStep", Method → "ExplicitEuler"}, MaxSteps → 10^6];
Plot[Evaluate[{x1[t], xo1[t]} /. ObsContrResponse], {t, 0, tmax},
 PlotLegends → {"x(t)", "x̂(t)"}, AxesLabel → {"t", "x̄1(t), x̂1(t)"}, PlotRange → All]
Plot[Evaluate[{x2[t], xo2[t]} /. ObsContrResponse], {t, 0, tmax},
 PlotLegends → {"x(t)", "x̂(t)"}, AxesLabel → {"t", "x̄2(t), x̂2(t)"}, PlotRange → All]
```

```
NDSolve : Encountered non −numerical value for a derivative at t == 0.`.

ReplaceAll : {NDSolve[{{x1′[t] == 0.
    + x1[t] + x2[t] + InterpolatingFunction[{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], x2′[t] == x2[t] −
    4.61158 xo1[≪1≫] − 4.19737 xo2[≪1≫] + InterpolatingFunction[{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫},
      {≪1≫}][t], xo1′[t] == −3.30834 xo1[≪1≫] + 1. xo2[≪1≫] + 4.30834 Plus[≪2≫], xo2′[t] == F.{≪
       1≫} − 9.33413 xo1[≪1≫] − 3.19737 xo2[≪1≫] + 4.72255 Plus[≪2≫]}, {x1[0] == 0, x2[0] == 0},
  {xo1[0] == 0, xo2[0] == 0}}, {x1[t], x2[t], xo1[t], xo2[t]}, {t, 0, 10}, StartingStepSize → 1/1000, Method → {
    FixedStep, Method →
   ExplicitEuler}, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch
 table, and so cannot be used for replacing.

NDSolve : 0.0002042857142857143` cannot be used as a variable.

ReplaceAll : {NDSolve[{{x1′[0.000204286] == 0.646598
    + x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598
    + x2[0.000204286] − 4.61158 xo1[≪1≫] − 4.19737 xo2[≪1≫], xo1′[0.000204286] == 4.30834
     Plus[≪2≫] − 3.30834 xo1[≪1≫] + 1. xo2[≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus[
      ≪2≫] − 9.33413 xo1[≪1≫] − 3.19737 xo2[≪1≫]}, {x1[0] == 0, x2[0] == 0}, {xo1[0] == 0, xo2[0] ==
    0}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫, MaxSteps
   → 1000000]} is neither a list of replacement rules nor a valid dispatch table, and so
 cannot be used for replacing.

NDSolve : Value of option MaxSteps −> 1.`*^6 should be a positive integer or Infinity.

ReplaceAll : {NDSolve[{{x1′[0.000204286] == 0.646598
    + x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598
    + x2[0.000204286] − 4.61158 xo1[≪1≫] − 4.19737 xo2[≪1≫], xo1′[0.000204286] == 4.30834
     Plus[≪2≫] − 3.30834 xo1[≪1≫] + 1. xo2[≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus[
      ≪2≫] − 9.33413 xo1[≪1≫] − 3.19737 xo2[≪1≫]}, {x1[0.] == 0., x2[0.] == 0.}, {xo1[0.] == 0., xo2[
    0.] == 0.}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫,
  MaxSteps → 1. × 10^6]} is neither a list of replacement rules nor a valid dispatch table, and so
 cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.
```

NDSolve : 0.20428591836734694` cannot be used as a variable.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

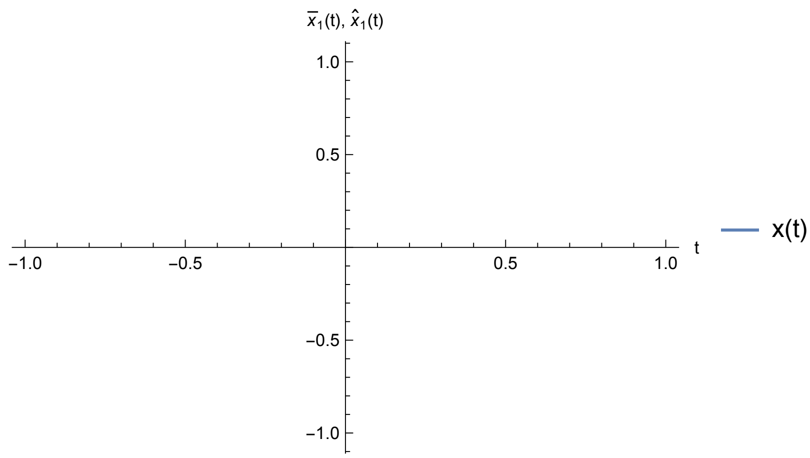
NDSolve : 0.40836755102040817` cannot be used as a variable.

General : Further output of NDSolve::dsvar will be suppressed during this calculation.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

General : Further output of NDSolve::ioppf will be suppressed during this calculation.

Out[∘]=




NDSolve : Encountered non −numerical value for a derivative at t == 0.`.

ReplaceAll : {NDSolve [{{x1′[t] == 0. + x1[t] + x2[t] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], x2′[t] == x2[t] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], xo1′[t] == −3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫] + 4.30834 Plus [≪2≫], xo2′[t] == F.{≪1≫} − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫] + 4.72255 Plus [≪2≫]}, {x1[0] == 0, x2[0] == 0}, {xo1[0] == 0, xo2[0] == 0}}, {x1[t], x2[t], xo1[t], xo2[t]}, {t, 0, 10}, StartingStepSize → $\frac{1}{1000}$, Method → {FixedStep, Method → ExplicitEuler}, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 0.0002042857142857143` cannot be used as a variable.

ReplaceAll : {NDSolve [{{x1′[0.000204286] == 0.646598 + x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598 + x2[0.000204286] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫], xo1′[0.000204286] == 4.30834 Plus [≪2≫] − 3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus [≪2≫] − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫]}, {x1[0] == 0, x2[0] == 0}, {xo1[0] == 0, xo2[0] == 0}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

ReplaceAll : {NDSolve [{{x1′[0.000204286 ] == 0.646598 + x1[0.000204286 ] + x2[0.000204286 ], x2′[0.000204286 ] == 0.646598 + x2[0.000204286 ] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫], xo1′[0.000204286 ] == 4.30834 Plus[≪2≫] − 3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫], xo2′[0.000204286 ] == F.{≪1≫} + 4.72255 Plus [≪2≫] − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫]}, {x1[0.] == 0., x2 [0.] == 0.}, {xo1 [0.] == 0., xo2 [0.] == 0.}}, {x1[0.000204286 ], x2[0.000204286 ], xo1[0.000204286 ], xo2[0.000204286 ]}, ≪3≫, MaxSteps → 1. × $10^6$]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.

NDSolve : 0.20428591836734694\` cannot be used as a variable.

NDSolve : Value of option MaxSteps −> 1.\`*^6 should be a positive integer or Infinity.

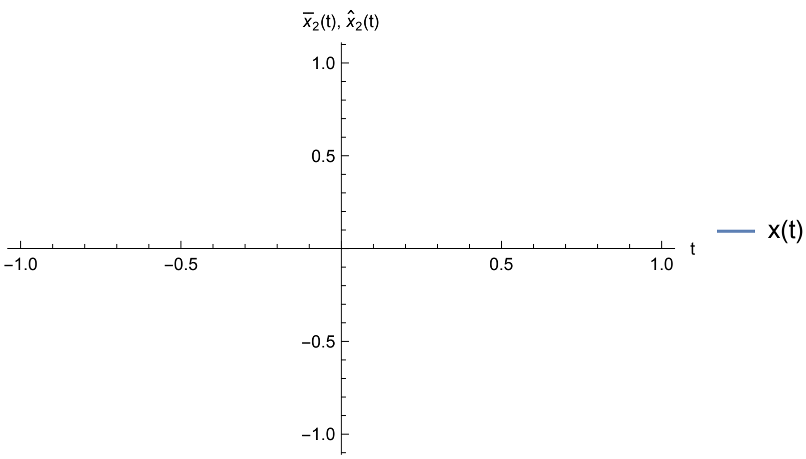
NDSolve : 0.40836755102040817\` cannot be used as a variable.

General : Further output of NDSolve::dsvar will be suppressed during this calculation.

NDSolve : Value of option MaxSteps −> 1.\`*^6 should be a positive integer or Infinity.

General : Further output of NDSolve::ioppf will be suppressed during this calculation.

Out[ ]=




# I had a problem with plotting. I could not figure out why.

NDSolve : Encountered non −numerical value for a derivative at t == 0.\`.

ReplaceAll : {NDSolve[{{x1′[t] == 0.
+ x1[t] + x2[t] + InterpolatingFunction[{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], x2′[t] == x2[t] –
4.61158 xo1[≪1≫] – 4.19737 xo2[≪1≫] + InterpolatingFunction[{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫},
{≪1≫}][t], xo1′[t] == –3.30834 xo1[≪1≫] + 1. xo2[≪1≫] + 4.30834 Plus[≪2≫], xo2′[t] == F.{≪
1≫} – 9.33413 xo1[≪1≫] – 3.19737 xo2[≪1≫] + 4.72255 Plus[≪2≫]}, {x1[0] == 0, x2[0] == 0},
{xo1[0] == 0, xo2[0] == 0}}, {x1[t], x2[t], xo1[t], xo2[t]}, {t, 0, 10}, StartingStepSize → $\frac{1}{1000}$, Method → {
FixedStep, Method →
ExplicitEuler}, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch
table, and so cannot be used for replacing.

NDSolve : 0.0002042857142857143` cannot be used as a variable.

ReplaceAll : {≪1≫} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for
replacing.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

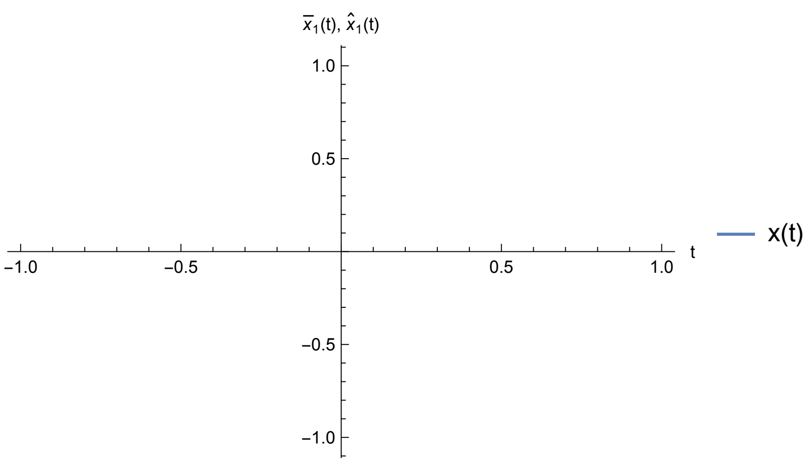
ReplaceAll : {NDSolve[{{x1′[0.000204286] == 0.646598
+ x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598
+ x2[0.000204286] – 4.61158 xo1[≪1≫] – 4.19737 xo2[≪1≫], xo1′[0.000204286] == 4.30834
Plus[≪2≫] – 3.30834 xo1[≪1≫] + 1. xo2[≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus[
≪2≫] – 9.33413 xo1[≪1≫] – 3.19737 xo2[≪1≫]}, {x1[0.] == 0., x2[0.] == 0.}, {xo1[0.] == 0., xo2[
0.] == 0.}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫,
MaxSteps → 1. × $10^6$]} is neither a list of replacement rules nor a valid dispatch table, and so
cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.

NDSolve : 0.20428591836734694` cannot be used as a variable.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

Out[∘]=




NDSolve : Encountered non–numerical value for a derivative at t == 0.`.

••• ReplaceAll : {NDSolve [{{x1′[t] == 0.
+ x1[t] + x2[t] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], x2′[t] == x2[t] −
4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫},
{≪1≫}][t], xo1′[t] == −3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫] + 4.30834 Plus [≪2≫], xo2′[t] == F.{≪
1≫} − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫] + 4.72255 Plus [≪2≫]}, {x1[0] == 0, x2[0] == 0},
{xo1[0] == 0, xo2[0] == 0}}, {x1[t], x2[t], xo1[t], xo2[t]}, {t, 0, 10}, StartingStepSize → $\frac{1}{1000}$, Method → {
FixedStep, Method →
ExplicitEuler }, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch
table, and so cannot be used for replacing.

••• NDSolve : 0.0002042857142857143` cannot be used as a variable.

••• ReplaceAll : {≪1≫} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

••• NDSolve : Value of option MaxSteps −> 1.`∗^6 should be a positive integer or Infinity.

••• ReplaceAll : {NDSolve [{{x1′[0.000204286] == 0.646598
+ x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598
+ x2[0.000204286] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫], xo1′[0.000204286] == 4.30834
Plus [≪2≫] − 3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus [
≪2≫] − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫]}, {x1[0.] == 0., x2[0.] == 0.}, {xo1[0.] == 0., xo2[
0.] == 0.}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫,
MaxSteps → $1. \times 10^6$]} is neither a list of replacement rules nor a valid dispatch table, and so
cannot be used for replacing.

••• General : Further output of ReplaceAll::reps will be suppressed during this calculation.

••• NDSolve : 0.20428591836734694` cannot be used as a variable.

••• NDSolve : Value of option MaxSteps −> 1.`∗^6 should be a positive integer or Infinity.

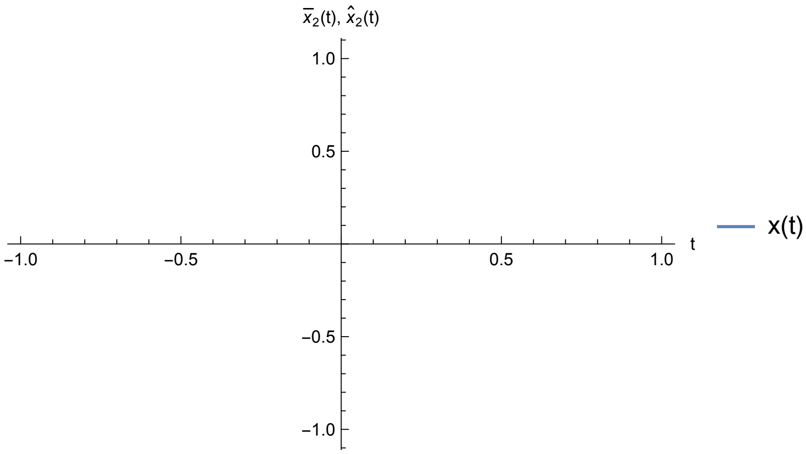
••• NDSolve : 0.40836755102040817` cannot be used as a variable.

••• General : Further output of NDSolve::dsvar will be suppressed during this calculation.

••• NDSolve : Value of option MaxSteps −> 1.`∗^6 should be a positive integer or Infinity.

••• General : Further output of NDSolve::ioppf will be suppressed during this calculation.

Out[ ]=




## Output:

I had a problem with plotting. I could not figure out why.

In[ ]:=

```
Plot[Evaluate[{y[t], CC.xo[t]} /. ObsContrResponse], {t, 0, tmax},
 AxesLabel → {"t", "y(t), x₁(t), xₒ₁(t)"}, PlotRange → All,
 PlotStyle → {Blue, Red, Green}, PlotLegends → {"y(t)", "ŷ(t)"}]
```

NDSolve : Encountered non −numerical value for a derivative at t == 0.`.

ReplaceAll : {NDSolve [{{x1′[t] == 0. + x1[t] + x2[t] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], x2′[t] == x2[t] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫] + InterpolatingFunction [{≪1≫}, {≪13≫}, {≪1≫}, {≪3≫}, {≪1≫}][t], xo1′[t] == −3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫] + 4.30834 Plus [≪2≫], xo2′[t] == F.{≪1≫} − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫] + 4.72255 Plus [≪2≫]}, {x1[0] == 0, x2[0] == 0}, {xo1[0] == 0, xo2[0] == 0}}, {x1[t], x2[t], xo1[t], xo2[t]}, {t, 0, 10}, StartingStepSize → $\frac{1}{100}$, Method → {FixedStep, Method → ExplicitEuler}, MaxSteps → 1000000]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : 0.0002042857142857143` cannot be used as a variable.

ReplaceAll : {≪1≫} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

NDSolve : Value of option MaxSteps −> 1.`*^6 should be a positive integer or Infinity.

ReplaceAll : {NDSolve [{{x1′[0.000204286] == 0.646598 + x1[0.000204286] + x2[0.000204286], x2′[0.000204286] == 0.646598 + x2[0.000204286] − 4.61158 xo1 [≪1≫] − 4.19737 xo2 [≪1≫], xo1′[0.000204286] == 4.30834 Plus [≪2≫] − 3.30834 xo1 [≪1≫] + 1. xo2 [≪1≫], xo2′[0.000204286] == F.{≪1≫} + 4.72255 Plus [≪2≫] − 9.33413 xo1 [≪1≫] − 3.19737 xo2 [≪1≫]}, {x1[0.] == 0., x2[0.] == 0.}, {xo1[0.] == 0., xo2[0.] == 0.}}, {x1[0.000204286], x2[0.000204286], xo1[0.000204286], xo2[0.000204286]}, ≪3≫, MaxSteps → 1. × $10^6$]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

General : Further output of ReplaceAll::reps will be suppressed during this calculation.

NDSolve : 0.20428591836734694` cannot be used as a variable.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

NDSolve : 0.40836755102040817` cannot be used as a variable.

General : Further output of NDSolve::dsvar will be suppressed during this calculation.

NDSolve : Value of option MaxSteps -> 1.`*^6 should be a positive integer or Infinity.

General : Further output of NDSolve::ioppf will be suppressed during this calculation.

Out[ ]=