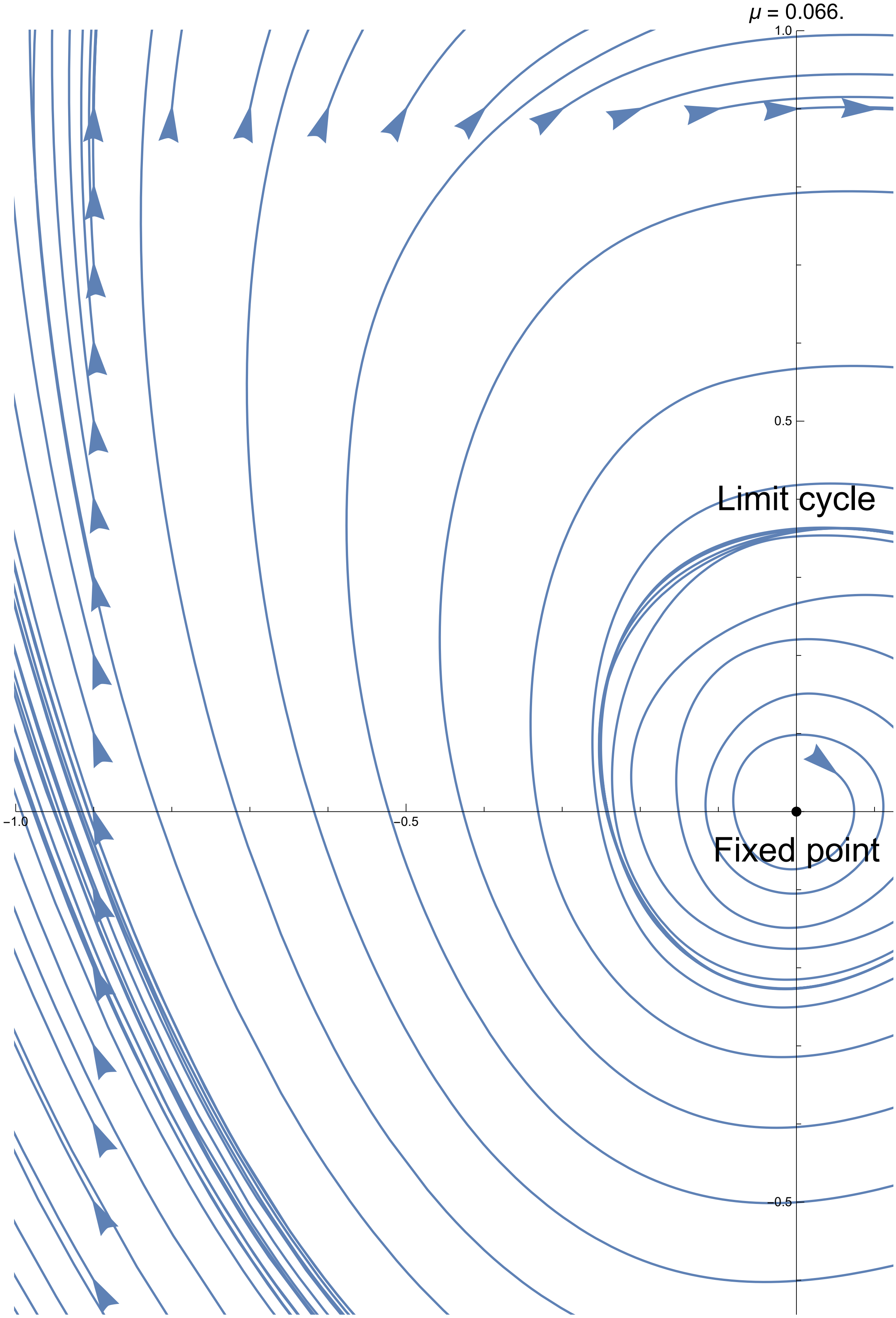
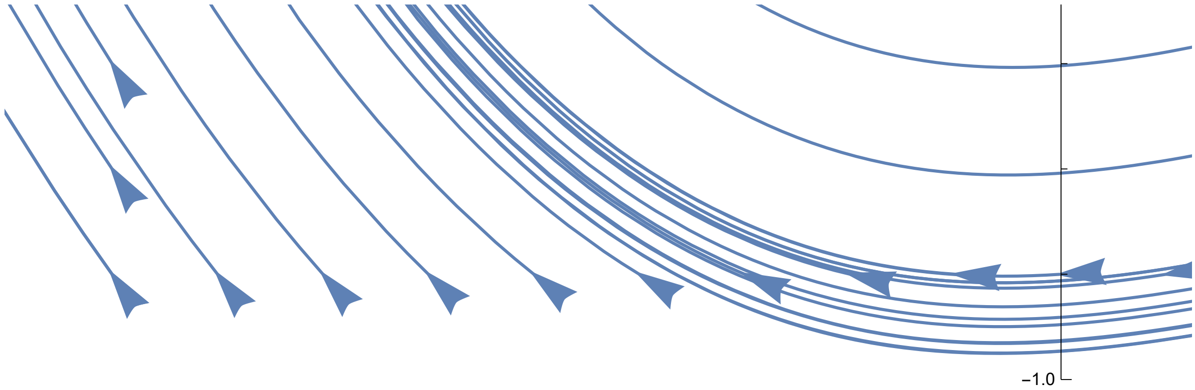
In[1723]:=

```
minxplot = -1;
maxxplot = 1;
minyplot = -1;
maxyplot = 1;

tMax = 50;
μ = 0.066;
title = StringForm["μ = `1`.", μ];

sol[u_, v_] :=
 NDSolve[{x'[t] == μ * x[t] + y[t] - x[t]^2, y'[t] == -x[t] + μ * y[t] + 2 * x[t]^2,
   x[0] == u, y[0] == v}, {x, y}, {t, tMax}, Method → "StiffnessSwitching"]

p1 = ParametricPlot[Evaluate[{x[t], y[t]} /. sol[0.05, 0.05]],
    {t, 0, tMax}, PlotRange → {{minxplot, maxxplot}, {minyplot, maxyplot}},
    PlotLabel → Style[title, FontSize → 15]] /.
   Line[x_] :→ {Arrowheads[{0.02, 0.}], Arrow[x]};

p2 = ParametricPlot[Evaluate[{x[t], y[t]} /. sol[0.1, 0]],
    {t, 0, tMax}, PlotRange → {{minxplot, maxxplot}, {minyplot, maxyplot}},
    PlotLabel → Style[title, FontSize → 15]] /.
   Line[x_] :→ {Arrowheads[{0.02, 0.}], Arrow[x]};
p3 = ParametricPlot[Evaluate[{x[t], y[t]} /. sol[0.01, 0.01]],
    {t, 0, tMax}, PlotRange → {{minxplot, maxxplot}, {minyplot, maxyplot}},
    PlotLabel → Style[title, FontSize → 15]] /.
   Line[x_] :→ {Arrowheads[{0.02, 0.}], Arrow[x]};

miny = -0.9;
maxy = 0.9;
minx = -0.9;
maxx = 0.9;
step = 0.1;

Table11 = Table[{minx, y}, {y, miny, maxy, step}];
Table12 = Table[{x, maxy}, {x, minx, maxx, step}];
Table13 = Table[{maxx, y}, {y, miny, maxy, step}];
Table14 = Table[{x, miny}, {x, minx, maxx, step}];

TableFinal = Join[Table11, Table12, Table13, Table14];
xStar = (μ^2 + 1) / (μ + 2);
yStar = xStar^2 - μ * xStar;


Show[p1,
 Table[ParametricPlot[
    Evaluate[{x[t], y[t]} /. sol[TableFinal[[i, 1]], TableFinal[[i, 2]]]],
    {t, 0, tMax}, PlotRange → {{minxplot, maxxplot}, {minyplot, maxyplot}},
    PlotLabel → Style[title, FontSize → 15]] /.
   Line[x_] :→ {Arrowheads[{0.02, 0.}], Arrow[x]}, {i, Length[TableFinal]}],
 Graphics[{PointSize[Large], Point[{0, 0}]}],
 Graphics[Text[Style["Fixed point", Large], {0, -0.05}]],
 Graphics[{PointSize[Large], Point[{xStar, yStar}]}],
 Graphics[Text[Style["Fixed point", Large], {xStar, yStar - 0.05}]],
 Graphics[Text[Style["Limit cycle", Large], {0, 0.4}]]
```

]

Out[∘]=





In[∘]:= xStar = (μ^2 + 1) / (μ + 2)
yStar = xStar^2 - μ * xStar

Out[∘]= $\frac{1}{2}$

In[∘]:= yStar

Out[∘]= 6

In[920]:=

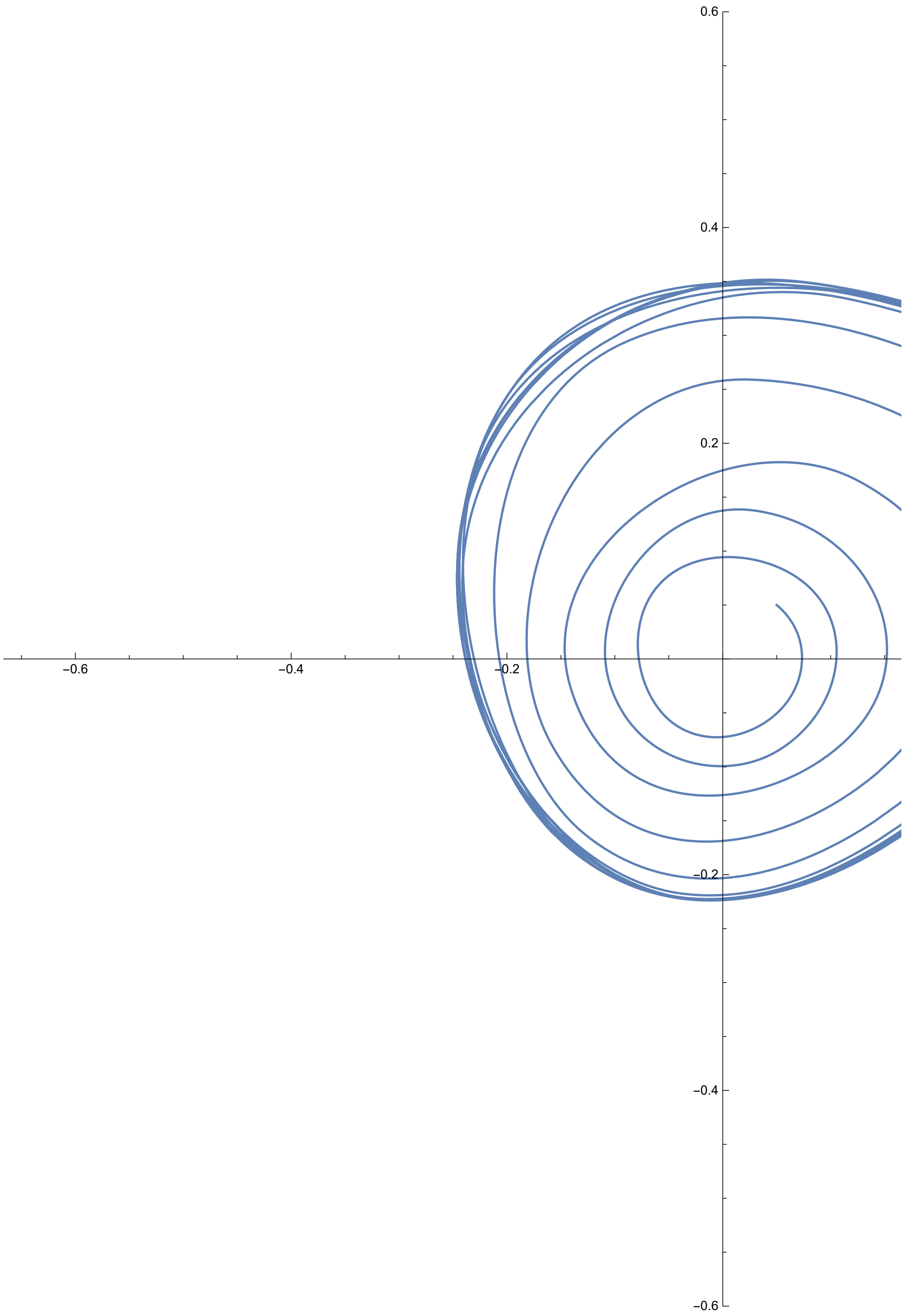
```
Clear["Global`*"]
tMax = 1000;
μ = 0.059;
sol := NDSolve[{x'[t] == μ * x[t] + y[t] - x[t]^2, y'[t] == -x[t] + μ * y[t] + 2 * x[t]^2,
   x[0] == 0.05, y[0] == 0.05}, {x, y}, {t, tMax}, Method → "StiffnessSwitching"]
xStar = (μ^2 + 1) / (μ + 2);
yStar = xStar^2 - μ * xStar;

distance[t_] = (((x[t] /. sol[[1]]) - xStar)^2 + ((y[t] /. sol[[1]]) - yStar)^2);

min = Minimize[
  {(((x[t] /. sol[[1]]) - xStar)^2 + ((y[t] /. sol[[1]]) - yStar)^2), 0 < t < tMax}, t]
tMin = t /. min[[2]]
distance[tMin]

Show[
 ParametricPlot[Evaluate[{x[t], y[t]} /. sol], {t, 0, 100}, PlotRange → {-0.6, 0.6}],
 Graphics[{PointSize[Large], Pink, Point[{xStar, yStar}]}],
 Graphics[
  {PointSize[Large], Red, Point[{x[tMin] /. sol[[1]], y[tMin] /. sol[[1]]}]}]
]
```

Out[927]= {0.00424391, {t → 379.585}}

Out[928]= 379.585

Out[929]= 0.00424391

Out[930]=



In[421]:= `NumberForm[0.00110473, 16]`

Out[421]//NumberForm=

`0.001104725158038263`

In[410]:= `NumberForm[0.000435907, 16]`

Out[410]//NumberForm=

`0.0004359074464754355`

In[422]:= `NumberForm[0.000435907, 16]`

Out[422]//NumberForm=

`0.0004359074464754355`

In[319]:= `distance[t /. min[[2]]]`

Out[319]= `0.0489753`

```
muc = 0.066;
mu = [0.065, 0.064, 0.063, 0.062, 0.061, 0.060, 0.059];
gamma = sqrt([0.000266334, 0.000638389, 0.00128266, 0.00198841,
 0.00258186, 0.0035553,0.00424391]);

mudiff = muc-mu;

loglog(mudiff, gamma)

Const = polyfit(log(mudiff),log(gamma), 1);
m = Const(1);
k = Const(2);
YBL = mudiff.^m.*exp(k);
hold on
loglog(mudiff, YBL)
```

Published with MATLAB® R2020a

In[1464]:=

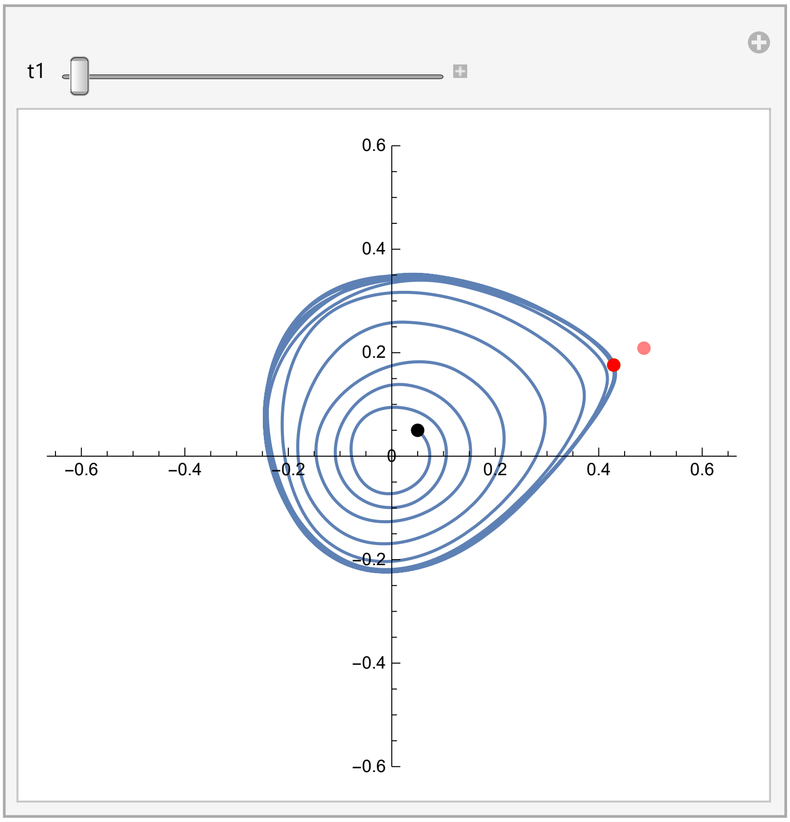
```
Clear["Global`*"]
tMax = 1000;
μ = 0.059;
sol := NDSolve[{x'[t] == μ * x[t] + y[t] - x[t]^2, y'[t] == -x[t] + μ * y[t] + 2 * x[t]^2,
   x[0] == 0.05, y[0] == 0.05}, {x, y}, {t, tMax}, Method → "StiffnessSwitching"]
xStar = (μ^2 + 1) / (μ + 2);
yStar = xStar^2 - μ * xStar;
t0 = 900;


distance[T_] = (((x[t0 - T] /. sol[[1]]) - (x[t0] /. sol[[1]]))^2 +
     ((y[t0 - T] /. sol[[1]]) - (y[t0] /. sol[[1]]))^2);

min = Minimize[{ (((x[t0 - T] /. sol[[1]]) - (x[t0] /. sol[[1]]))^2 +
      ((y[t0 - T] /. sol[[1]]) - (y[t0] /. sol[[1]]))^2), 5 < T < 15}, T]
tMin = t0 - T /. min[[2]];
distance[ T /. min[[2]]]
 T /. min[[2]]

Manipulate[Show[
  ParametricPlot[Evaluate[{x[t], y[t]} /. sol], {t, 0, 100}, PlotRange → {-0.6, 0.6}],
  Graphics[
   {PointSize[Large], Green, Point[{x[t0] /. sol[[1]], y[t0] /. sol[[1]]}]}],
  Graphics[{PointSize[Large], Pink, Point[{xStar, yStar}]}],
  Graphics[
   {PointSize[Large], Red, Point[{x[tMin] /. sol[[1]], y[tMin] /. sol[[1]]}]}],
  Graphics[{PointSize[Large], Black,
    Point[{x[t1] /. sol[[1]], y[t1] /. sol[[1]]}]}],
  Graphics[Text[StringForm["``", t1], {0, 0}]]
 ],
 {t1, 0, tMax}
]
```

Out[1472]= $\{1.01863\times10^{-8}, \{T\to 9.63421\}\}$

Out[1474]= 1.01863×10^{-8}

Out[1475]= 9.63421

Out[1476]=



In[421]:=
```
NumberForm[0.00110473, 16]
```
Out[421]//NumberForm=
```
0.001104725158038263
```

In[410]:=
```
NumberForm[0.000435907, 16]
```
Out[410]//NumberForm=
```
0.0004359074464754355
```

In[422]:=
```
NumberForm[0.000435907, 16]
```
Out[422]//NumberForm=
```
0.0004359074464754355
```

In[319]:=
```
distance[t /. min[[2]]]
```
Out[319]=
```
0.0489753
```

```
mu = [0.065, 0.064, 0.063, 0.062, 0.061, 0.060, 0.059];
T = [12.4699, 11.4586, 10.8724, 10.457, 10.1344, 9.85957, 9.63421];

mulinspace = linspace(0.065, 0.059);

muc = 0.066;
A = 2.4053;
a = 0.7258;
eigen = @(mu) (2*mu-1+sqrt(5+9*mu.^2+4*mu.^3+mu.^4))./(2+mu);
eigen(0.065)

C = mean(-T ./ (log(A*(abs(mu-muc).^a))./eigen(mu)));

estimatedT = -C*(log(A*(abs(mulinspace-muc).^a))./eigen(mulinspace));

% for i = 1:length(mu)
%     fprintf('Mu is %.4f, T is %.3f, estimated T is %.3f.\n', mu(i),
 T(i),  estimatedT(i))
% end

clf
hold on
p1 = plot(muc-mu,T, 'xblack');
plot(muc-mu,T, 'black')
p2 = plot(muc-mulinspace,estimatedT);
xlabel('$\mu_c - \mu$', 'interpreter', 'latex')
ylabel('$T$', 'interpreter', 'latex')
title('Plot of numerically obtained period and estimated
 period', 'interpreter', 'latex')
legend([p1, p2], 'Numerically obtained period', 'Estimated period')
```

Published with MATLAB® R2020a