4.2 Stability exponents for a toy model

We define a simple flow in polar coordinates on which to test that the Lyapunov exponent calculation works.
Define the simple dynamics

$$\dot{r} = \mu r - r^3 = r(\mu - r^2)$$
$$\dot{\theta} = \omega + \nu r^2$$

which has a stable fixed point and a limit cycle if $\mu > 0$.

(a) Calculate the radius r_0 and the period T of the limit cycle for $\mu > 0$. Give your result on the form $[r_0, T]$.

To find the radius of a limit cycle one needs to know that its radius is constant, $\dot{r} = 0$. So one can now solve

$$0 = \mu r - r^3.$$

One receives $r_1 = 0$, $r_2 = \sqrt{\mu}$ and $r_3 = -\sqrt{\mu}$, where off only r_2 is a real solution.

To find the period T of the limit cycle one uses the angular velocity $\dot{\theta}$. Take the circumference of the limit cycle 2π and divide it by the angular velocity and then apply r_2.

$$T = \frac{2\pi}{\omega + \nu\mu}$$

Transform the dynamical system (1) into the Cartesian coordinates X_1 and X_2, where $X_1 = r\cos\theta$ and $X_2 = r\sin\theta$. Compare your result to the dynamical system $\dot{\mathrm{X}} = \mathrm{F}(\mathrm{X})$ with

$$\dot{X}_1 = F_1(X) = \frac{1}{10} X_1 - X_2^3 - X_1 X_2^2 - X_1^2 X_2 - X_2 - X_1^3$$
$$\dot{X}_2 = F_2(X) = X_1 + \frac{1}{10} X_2 + X_1 X_2^2 + X_1^3 - X_2^3 - X_1^2 X_2.$$

With $r = \sqrt{x_1^2 + x_2^2}$ and $\theta = \arctan\left(\frac{x_1}{x_2}\right)$ the system can be rewritten using Mathematica to the following equations:

$$\dot{x}_1 = \mu x_1 - \nu x_1^2 x_2 - x_1 x_2^2 - x_1^3 - \nu x_2^3 - \omega x_2$$

$$\dot{x}_2 = \nu x_1^3 + \nu x_1 x_2^2 - x_1^2 x_2 + \omega x_1 + \mu x_2 - x_2^3$$

In[∘]:=
```
ClearAll["Global`*"]

left1 = D[Sqrt[x1[t]^2 + x2[t]^2], t] // Simplify;
left2 = D[ArcTan[x1[t], x2[t]], t] // Simplify;

right1 = (Sqrt[x1[t]^2 + x2[t]^2]) * ( μ - (x1[t]^2 + x2[t]^2));
right2 = ω + ν*(x1[t]^2 + x2[t]^2);

eq1 = left1 == right1;
eq2 = left2 == right2;

sol1 = Solve[eq1,x1'[t]]//Simplify;
sol2 = Solve[eq2,x2'[t]]//Simplify;

(* Substitution of sol1 in sol2 *)
sol2WithSubstitution = sol2 /. sol1⟦1⟧//Simplify;

                                                       x2[t] (-x1[t]^3+ω x2[t]+ν x2[t]^3+x1
gleichung1 = x2'[t] == ν x1[t]^3+x1[t] (ω+2 ν x2[t]^2)+ ------------------------------------

solF1 = Solve[gleichung1,x2'[t]]//ExpandAll;

sol1WithSubstitution = sol1 /. solF1⟦1⟧//ExpandAll;
```

b) Make a phase portrait of the dynamical system (2), showing a few representative trajectories. In the same figure, plot the limit cycle using a suitable representative trajectory.

Upload your figure as .pdf or .png. Using StreamPlot[] is ***not*** *acceptable.*

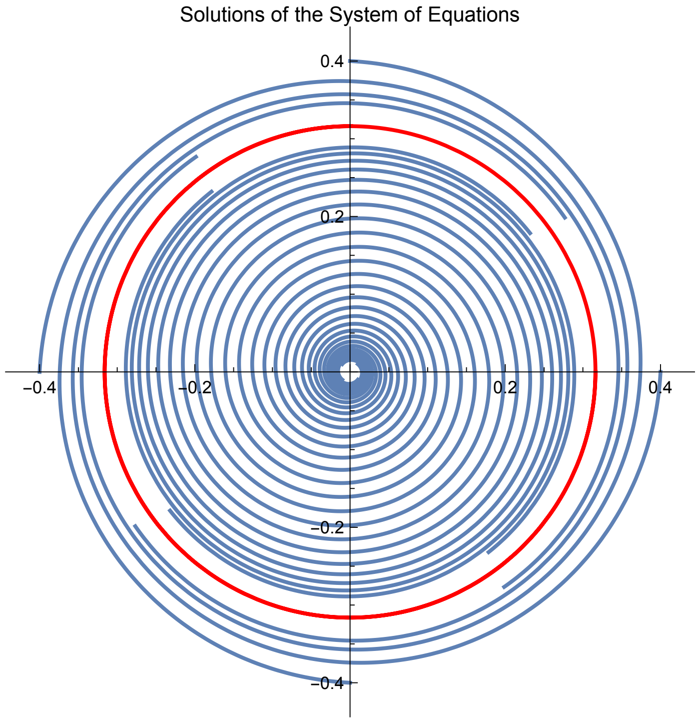
(0.5 points)

In[ ]:=

```
ClearAll["Global`*"]

(* Parameters for equations *)
mu = 1/10;
nu = 1;
omega = 1;

(* Define the system of equations *)
dotX1 = x1'[t] == mu*x1[t] + nu*x1[t]^2*x2[t] - x1[t]*x2[t]^2 - x1[t]^3 + nu*x2[t]^3 +
dotX2 = x2'[t] == -nu*x1[t]^3 - nu*x1[t]*x2[t]^2 - x1[t]^2*x2[t] - omega*x1[t] + mu*x2[

system1 = {dotX1, dotX2};

FixedPoint1 = {0, 0};

t0 = 0;
tMax1 = 40;
tMax2 = 5;
tMax3 = 10;
eta1 = 0.01;
eta2 = 0.4;
eta3 = Sqrt[mu] - 0.0001;

(* Initial starting points with distance radius 0.2 from FixedPoint1 *)
initialConditions1 = {
   {x1[0] == FixedPoint1[[1]] + eta1, x2[0] == FixedPoint1[[2]] + eta1},
   {x1[0] == FixedPoint1[[1]] - eta1, x2[0] == FixedPoint1[[2]] + eta1},
   {x1[0] == FixedPoint1[[1]] + eta1, x2[0] == FixedPoint1[[2]] - eta1},
   {x1[0] == FixedPoint1[[1]] - eta1, x2[0] == FixedPoint1[[2]] - eta1}
};

(* Initial starting points with distance radius 1 from FixedPoint1 *)
initialConditions2 = {
   {x1[0] == FixedPoint1[[1]] + eta2, x2[0] == FixedPoint1[[2]]},
   {x1[0] == FixedPoint1[[1]] - eta2, x2[0] == FixedPoint1[[2]]},
   {x1[0] == FixedPoint1[[1]], x2[0] == FixedPoint1[[2]] + eta2},
   {x1[0] == FixedPoint1[[1]], x2[0] == FixedPoint1[[2]] - eta2}
};

ICLimitCycle = {x1[0] == FixedPoint1[[1]] + eta3, x2[0] == FixedPoint1[[2]]};

(* Use NDSolve for each set of initial conditions *)
sol1 = NDSolve[{system1, #}, {x1, x2}, {t, t0, tMax1}] & /@ initialConditions1;
sol2 = NDSolve[{system1, #}, {x1, x2}, {t, t0, tMax2}] & /@ initialConditions2;
sol3 = NDSolve[{system1, ICLimitCycle}, {x1, x2}, {t, t0, tMax3}];

(* Plot the solutions *)
TrajectoryPlot1 = ParametricPlot[Evaluate[{x1[t], x2[t]} /. #], {t, t0, tMax1},
```

```
     PlotStyle → Automatic] & /@ sol1;
TrajectoryPlot2 = ParametricPlot[Evaluate[{x1[t], x2[t]} /. #], {t, t0, tMax2},
     PlotStyle → Automatic] & /@ sol2;
TrajectoryPlot3 = ParametricPlot[Evaluate[{x1[t], x2[t]} /. sol3], {t, t0, tMax3},
     PlotStyle → Red];  (* Fix: Set PlotStyle to Red *)

Show[TrajectoryPlot1, TrajectoryPlot2, TrajectoryPlot3,
 FrameLabel → {"t", "Solution"},
 PlotLabel → "Solutions of the System of Equations",
 PlotRange → Full
 ]
```

Out[ ]=



c) For which values of μ, ω and v is the system (1) written in Cartesian coordinates identical to (2). Write your result as the vector $[\mu, \omega, v]$.

This can be done using the calculation from right before b). The comparison of the coefficients yields:

$$[\mu, \omega, v] = \left[\frac{1}{10}, 1, 1\right].$$

From now on, we consider only the dynamical system

(2). The deformation matrix M corresponding to (2) satisfies the differential equation

$$\dot{M} = J(t)\,M(t)$$

with M(0)=I (the identity matrix) and $J_{i\,j} = \frac{\partial F_i(X)}{\partial X_j}$.

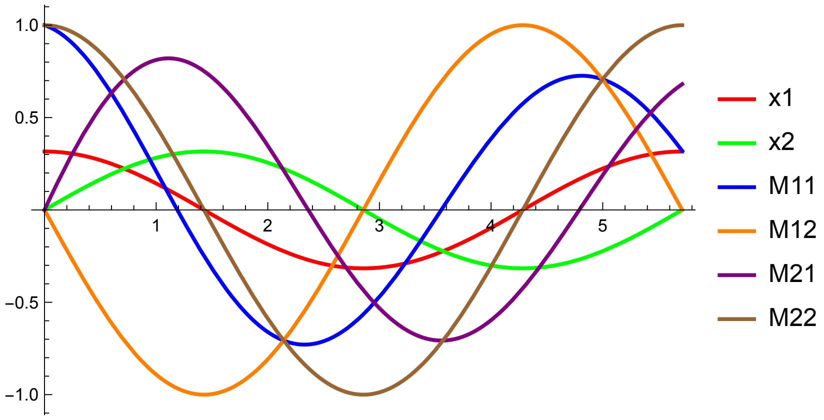
Set up a computer program to numerically solve the differential equation in the six variables X_1, X_2 and M_{11}, M_{12}, M_{21} and M_{22}.

---

d) Starting on the limit cycle with $X_1(0) > 0$ and $X_2(0) = 0$, plot all six quantities as functions of t for one period T of the limit cycle, $t \in [0, T]$. using a different colour for each quantity.

In[ ]:=
```
ClearAll["Global`*"]

μ = 1/10;
ω = 1;
ν = 1;

f1[x1_, x2_] := μ*x1[t] - ν*x1[t]^2*x2[t] - x1[t]*x2[t]^2 - x1[t]^3 - ν*x2[t]^3 - ω*x2
f2[x1_, x2_] := ν*x1[t]^3 + ν*x1[t]*x2[t]^2 - x1[t]^2*x2[t] + ω*x1[t] + μ*x2[t] - x2[t

J11 = D[f1[x1, x2], x1[t]];
J12 = D[f1[x1, x2], x2[t]];
J21 = D[f2[x1, x2], x1[t]];
J22 = D[f2[x1, x2], x2[t]];

dotM11 = M11'[t] == J11*M11[t] + J12*M21[t];
dotM12 = M12'[t] == J11*M12[t] + J12*M22[t];
dotM21 = M21'[t] == J21*M11[t] + J22*M21[t];
dotM22 = M22'[t] == J21*M12[t] + J22*M22[t];

dotx1 = x1'[t] == μ*x1[t] - ν*x1[t]^2*x2[t] - x1[t]*x2[t]^2 - x1[t]^3 - ν*x2[t]^3 - ω*x
dotx2 = x2'[t] == ν*x1[t]^3 + ν*x1[t]*x2[t]^2 - x1[t]^2*x2[t] + ω*x1[t] + μ*x2[t] - x2[

System = {dotM11, dotM12, dotM21, dotM22, dotx1, dotx2};
InitialConditions = {x1[0] == Sqrt[μ], x2[0] == 0, M11[0] == 1, M12[0] == 0, M21[0] == 0,

t0 = 0;
tMax = 2*Pi/(ω+ν*μ);

sol = NDSolve[{System, InitialConditions}, {x1, x2, M11, M12, M21, M22}, {t, t0, tMax}

x1Values = x1[t] /. sol[[1]];
x2Values = x2[t] /. sol[[1]];
M11Values = M11[t] /. sol[[1]];
M12Values = M12[t] /. sol[[1]];
M21Values = M21[t] /. sol[[1]];
M22Values = M22[t] /. sol[[1]];

PlotTrajectories =
  Plot[Evaluate[{x1[t], x2[t], M11[t], M12[t], M21[t], M22[t]} /. sol[[1]]], {t, t0, tMa
    PlotLegends → {"x1", "x2", "M11", "M12", "M21", "M22"},
    PlotStyle → {{Red, Thick}, {Green, Thick}, {Blue, Thick}, {Orange, Thick}, {Purple
    FrameLabel → {{"Values", None}, {"Time", "Trajectories of x1, x2, M11, M12, M21, M
    PlotRange → All
  ];

Show[PlotTrajectories]
```

Out[ ]=



e) Give your numerical result for $M(T)$ obtained in d) to 4 relevant digits accuracy. Write it as a matrix of the form $[[M_{11}(T), M_{12}(T)], [M_{21}(T), M_{22}(T)]]$.

In[ ]:=

```
(* Evaluate values at tMax *)
M11AtTMax = M11Values /. t → tMax;
M12AtTMax = M12Values /. t → tMax;
M21AtTMax = M21Values /. t → tMax;
M22AtTMax = M22Values /. t → tMax;

(* Display results with 4 relevant digits accuracy *)
sol = Round[{{M11AtTMax, M12AtTMax}, {M21AtTMax, M22AtTMax}}, 0.0001]//MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} 0.3191 & 0. \\ 0.6809 & 1. \end{pmatrix}$$

f) Calculate the stability exponents of separations $\tilde{\sigma}_1$ and $\tilde{\sigma}_2$ of the limit cycle from the eigenvalues of $M(T)$ to 4 relevant digits accuracy. Write your results as the ordered vector $[\tilde{\sigma}_1, \tilde{\sigma}_2]$ with $\tilde{\sigma}_1 \leq \tilde{\sigma}_2$.

$$\sigma_i = \frac{1}{T} \log(\text{eigenvalue}(\text{M(T)}))$$

$[\sigma_1, \sigma_2]$ where $\sigma_1 \leq \sigma_2$

In[ ]:=
```
solMat = {{M11AtTMax, M12AtTMax}, {M21AtTMax, M22AtTMax}};

EV = Eigenvalues[solMat];

solTilde = Round[{1/tMax * Log[EV⟦2⟧],1/tMax * Log[EV⟦1⟧]},0.0001]
```

Out[ ]=
```
{-0.2, 0.}
```

---

g) Using what you know from all parts of this problem, calculate the deformation matrix M(T) analytically. Write your exact result (in Cartesian coordinates) in the form [[M_{11}, M_{12}], [M_{21}, M_{22}]]. Write exponentials as exp().

$$\frac{dM}{dt} = J(t)\,M$$

$$\int_{M(0)}^{M(T)} \frac{dM}{M} = \int_0^T J(t')\,dt'$$

$$M(T) = M_0\, exp\left[\int_0^T J(t')\,dt'\right]$$

$J_{polar} = \left[\mu - 2r^2, 0, 2vr, 0\right]$ (matrix)

$\rightarrow M_{polar}(T) = exp[J\,T]$

Now we want to transform them from polar to Cartesian.

$J_G = \dfrac{dPolar}{dCartesian} = \left[\dfrac{dr}{dx_1}, \dfrac{dr}{dx_2}, \dfrac{d\theta}{dx_1}, \dfrac{d\theta}{dx_2}\right]$ (matrix)

$M_{cartesian} = J_G^{-1}\, M_{polar}\, J_G$ here since M is expressed in polar, J_G is it as well. J^{-1}_G

In[35]:=
```
ClearAll["Global`*"]

J11 = D[Sqrt[x1^2+x2^2],x1];
J12 = D[Sqrt[x1^2+x2^2],x2];
J21 = D[ArcTan[x1,x2],x1];
J22 = D[ArcTan[x1,x2],x2];

JG = {{J11,J12},{J21,J22}};
JGInv = Inverse[JG];

JakobiPol = {{μ-3r^2,0},{2*ν*r,0}};
JakExp = MatrixExp[JakobiPol*T];


T = 2*Pi/(ω+ν*μ);
x1 = Sqrt[μ];
x2 = 0;
ω = 1;
ν = 1;
μ = 1/10;
r = Sqrt[x1^2+x2^2];

solM = JGInv.JakExp.JG //Simplify
```

Out[51]=

$$\left\{\left\{e^{-4\pi/11}, 0\right\}, \left\{1 - e^{-4\pi/11}, 1\right\}\right\}$$

h) Compute the stability exponents of separations $\tilde{\sigma}_1$ and $\tilde{\sigma}_2$ of the limit cycle analytically. Write your result on the ordered form $[\tilde{\sigma}_1, \tilde{\sigma}_2]$ with $\tilde{\sigma}_1 \leq \tilde{\sigma}_2$.

Take the Eigenvalues of the $M_{Cartesian}$ and do the analysis on the eigenvalues.

In[57]:=
```
solm = Eigenvalues[solM];

Round[{1/T * Log[solm〚2〛],1/T * Log[solm〚1〛]},0.0001]
```

Out[58]=

$$\{-0.2, 0.\}$$