College of Engineering
School of Aerospace Engineering

AE 6511: Optimal Guidance and Controls

HW5

Professor:
Panagiotis Tsiotras
Gtech AE Professor

Student:
Tomoki Koike
Gtech MS Student

November 8, 2021

Table of Contents

Problem 1

A rocket is launched from the origin (0, 0) with velocity $u(0)$ parallel to the x-axis and $v(0)$ parallel to the y-axis. Assuming a constant thrust, we wish to find the thrust direction $\theta(t)$ for minimum time to the point $(x_f, 0)$. The equations of motion can be written as

$$\begin{aligned}\dot{u}(t) &= \cos\theta(t)\\ \dot{v}(t) &= \sin\theta(t)\\ \dot{x}(t) &= u(t)\\ \dot{y}(t) &= v(t)\end{aligned}$$

(a) Show that the minimum final time is the smallest positive real root of the quartic equation

$$t_f^4 - 4t_f^2 + 8x_f t_f \cos\gamma_0 - 4x_f^2 = 0$$

where $x(t_f) = x_f, u(0) = \cos\gamma_0$, and $v(0) = \sin\gamma_0$.

(b) What is the optimal thrust direction profile $\theta^*(t)$ in terms of t_f, γ_0, and x_f?.

Solution:
(a) If we define the performance index of this problem in a form of a Mayer cost and set it up so that we want to minimize the time taken to travel from point (0, 0) to (x_f, 0) we have

$$\min J = t_f.$$

The Hamiltonian for this problem is then

$$\begin{aligned}H &= L + \lambda^T f\\ &= \lambda_u \cos\theta + \lambda_v \sin\theta + \lambda_x u + \lambda_y v.\end{aligned}$$

The system of adjoint equations are

$$\begin{aligned}\dot{\lambda}_u &= -\frac{\partial H}{\partial u} = -\lambda_x\\ \dot{\lambda}_v &= -\frac{\partial H}{\partial v} = -\lambda_y\\ \dot{\lambda}_x &= -\frac{\partial H}{\partial x} = 0\\ \dot{\lambda}_y &= -\frac{\partial H}{\partial y} = 0\end{aligned}$$

For this equation we know that, $t_0 = 0$, $u(0) = u_0$, $v(0) = v_0$, $x(0) = x_0 = 0$, $y(0) = y_0 = 0$, $x(t_f) = x_f$, and $y(t_f) = y_f = 0$ are fixed and $u(t_f) = u_f$, and $v(t_f) = v_f$ are free, and finally t_f is the objective that we want to minimize.

The transversality condition is

$$\begin{aligned} H(t_f) &= -\phi_t(t_f, x(t_f)) \\ \lambda(t_f) &= \phi_x^T(t_f, x(t_f)). \end{aligned}$$

where $\phi(t, x(t)) = t$. Which implies the following

$$\begin{aligned} H(t_f) &= -1 \\ \lambda_u(t_f) &= 0 \\ \lambda_v(t_f) &= 0 \end{aligned}$$

The optimal control is computed from

$$H_\theta = -\lambda_u \sin\theta + \lambda_v \cos\theta = 0.$$

This gives

$$\tan\theta = \frac{\lambda_v}{\lambda_u}.$$

and by solving the system of adjoint equations, we have

$$\begin{aligned} \lambda_x &= c_1 \\ \lambda_y &= c_2 \\ \lambda_u &= -c_1 t + c_3 \\ \lambda_v &= -c_2 t + c_4 \end{aligned}$$

which gives us

$$\tan\theta^* = \frac{-c_2 t + c_4}{-c_1 t + c_3}.$$

From, $\lambda_u(t_f) = 0$, $\lambda_v(t_f) = 0$ we know that

$$c_3 = c_1 t_f, \qquad c_4 = c_2 t_f,$$

and therefore,

$$\tan\theta^* = \frac{-c_2 t + c_4 t_f}{-c_1 + c_3 t_f} = \frac{c_2}{c_1} = \frac{\lambda_y}{\lambda_x}.$$

This implies that $\tan\theta$ is constant meaning that θ is constant. Which allows us to integrate the states as follows (while applying the initial conditions)

$$\begin{aligned}
u(t) &= t\cos\theta + u_0 \\
v(t) &= t\sin\theta + v_0 \\
x(t) &= \frac{1}{2}t^2\cos\theta + u_0 t \\
y(t) &= \frac{1}{2}t^2\sin\theta + v_0 t.
\end{aligned}$$

Then if we take the equations of $x(t)$ and $y(t)$

$$\begin{aligned}
(x(t) - u_0 t) &= \frac{1}{2}t^2\cos\theta \\
(y(t) - v_0 t) &= \frac{1}{2}t^2\sin\theta
\end{aligned}$$

Square both sides of these equations and add them up for $t = t_f$ then we have ($y_f = 0$)

$$\begin{aligned}
(x_f - u_0 t_f)^2 + (v_0 t_f)^2 &= \frac{t_f^4}{4} \\
x_f^2 - 2x_f u_0 t_f + u_0^2 t_f^2 + v_0^2 t_f^2 &= \frac{t_f^4}{4} \\
t_f^4 - 4(u_0^2 + v_0^2)t_f^2 + 8x_f t_f u_0 - 4x_f^2 &= 0.
\end{aligned}$$

Now let

$$u_0 = \cos\gamma_0, \qquad v_0 = \sin\gamma_0$$

Then this equation becomes

$$t_f^4 - 4(\cos^2\gamma_0 + \sin^2\gamma_0)t_f^2 + 8x_f t_f\cos\gamma_0 - 4x_f^2 = 0.$$

Hence,

$$\boxed{t_f^4 - 4t_f^2 + 8x_f t_f\cos\gamma_0 - 4x_f^2 = 0.}$$

(b) Recall from the derivations of part (a) we had the following equation

$$\begin{aligned}
(x_f - u_0 t_f) &= \frac{1}{2}t_f^2\cos\theta \\
(y_f - v_0 t_f) &= \frac{1}{2}t_f^2\sin\theta
\end{aligned}$$

If we divide the second equation by the first equation we have the following relation

$$\boxed{\tan\theta^* = \frac{y_f - t_f\sin\gamma_0}{x_f - t_f\cos\gamma_0}.}$$

Problem 2

Consider the following system of differential equations

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = -x_2 + u$$

subject to initial conditions $x_1(0) = x_2(0) = 0$. We want to find an optimal control $u(t)$ to minimize the cost

$$J = \frac{1}{2}\int_0^{t_f} u^2(t)dt$$

for the following cases.

(a) When $t_f = 3$ and the final state is given by $x(3) = \begin{bmatrix}1 & 2\end{bmatrix}^T$.

(b) When $t_f = 3$, $x_1(3) = 1$ and $x_2(3)$ is not specified.

(c) When $t_f = 3$ and the final state is not specified explicitly, but we would like it to be as close as possible to $x(3) = \begin{bmatrix}1 & 2\end{bmatrix}^T$. If the optimal trajectory you get is not sufficiently close to the required value, how would you modify the cost to achieve the desired accuracy? How does the overall cost change when the optimal trajectories satisfy the boundary condition at t_f with increasing accuracy? Compare these results with the case (a) above.

(d) When $t_f = 3$ and the final state should be on the line $2x_1 + 5x_2 = 20$.

(e) When t_f is not specified and $x(t_f)$ should be on the (moving) line

$$2x_1 + 5x_2 = 20 + \frac{t^2}{2}.$$

For each of these cases, derive the necessary conditions for the optimal control.

Solve *analytically* the necessary conditions in order to calculate the optimal control and the corresponding optimal trajectories for each cases. Plot the optimal trajectories vs. time and the optimal paths in the $x_1 - x_2$ plane. Verify that the boundary conditions are satisfied. Verify that the Hamiltonian is constant along the optimal trajectories. What is the optimal cost for each case?

Solution:
(a) The Hamiltonian equation for this optimization problem is

$$H = \frac{1}{2}u^2 + \lambda_1 x_2 + \lambda_2(-x_2 + u).$$

Then then adjoint equations for this problem are

$$\begin{aligned}\dot{\lambda}_1 &= -\frac{\partial H}{\partial x_1} = 0\\ \dot{\lambda}_2 &= -\frac{\partial H}{\partial x_2} = -\lambda_1 + \lambda_2.\end{aligned}$$

The optimal control becomes

$$\begin{aligned}\frac{\partial H}{\partial u} &= u + \lambda_2 = 0\\ \therefore u &= -\lambda_2.\end{aligned}$$

By intergrating the adjoint equations we have

$$\begin{aligned}\lambda_1 &= c = \text{const.}\\ \dot{\lambda}_2 &= -c + \lambda_2\end{aligned}$$

which is

$$\begin{aligned}\frac{d\lambda_2}{\lambda_2 - c} &= dt\\ \leftrightarrow \ln|\lambda_2 - c| &= t + c_1\\ \lambda_2 - c &= c_1 e^t\\ \therefore \lambda_2 &= c_1 e^t + c.\end{aligned}$$

Now, since we have $u = -\lambda_2$, we can plug what we have derived into $\dot{x}_2 = -x_2 + u$ and we obtain

$$\dot{x}_2 + x_2 = -c_1 e^t - c.$$

Let $P(t) = 1$ and $Q(t) = c_1 e^t + c$, then we can solve this first order nonhomogeneous ODE in the following manner

$$\begin{aligned}x_2(t) &= e^{-\int P(t)dt}\left[\int Q(t)e^{\int P(t)dt}dt + c_2\right]\\ &= e^{-t}\left[\int (-c_1 e^t - c)e^t dt + c_2\right]\\ &= -\frac{c_1}{2}e^t + c_2 e^{-t} - c.\end{aligned}$$

Then $\dot{x}_1 = x_2$ gives

$$\begin{aligned} x_1(t) &= \int \left(-\frac{c_1}{2}e^t + c_2 e^{-t} - c\right) dt \\ &= -\frac{c_1}{2}e^t - c_2 e^{-t} - ct + c_3. \end{aligned}$$

Now we have 4 equations from the boundary conditions and 4 unknowns, and therefore this problem is solvable. The boundary conditions are

$$x_1(0) = 0, \qquad x_2(0) = 0, \qquad x_1(3) = 1, \qquad x_2(3) = 2$$

and the unknowns are

$$c, \qquad c_1, \qquad c_2, \qquad c_3.$$

We solve this using `MATLAB` (refer to the code in Problem 2: MATLAB Code). The result becomes

$$\begin{cases} c = \dfrac{e^3 - 3}{e^3 + 5} = -0.6811 \\ c_1 = -\dfrac{6\,(e^3+1)}{e^6 + 4\,e^3 - 5} = 0.2642 \\ c_2 = \dfrac{e^3\,(e^3 - 7)}{e^6 + 4\,e^3 - 5} = 0.5490 \\ c_3 = -\dfrac{-e^6 + 10\,e^3 + 3}{e^6 + 4\,e^3 - 5} = 0.4168 \end{cases}$$

Hence, the optimal control for this problem becomes

$$\boxed{u^* = \frac{6\,(e^3+1)}{e^6 + 4\,e^3 - 5}e^t - \frac{e^3 - 3}{e^3 + 5} = 0.2642e^t - 0.6811.}$$

The minimum performance index is

$$\min J = \frac{0.5000\,(-277.0264\,e^9 + 7.9788\text{e+03}\,e^6 + 1.4764\text{e+04}\,e^3 + 6.5086\text{e+03})}{(e^6 + 4\,e^3 - 5)^2} + 1.5000$$

which is simply

$$\boxed{\min J = 4.2859.}$$

If we plot the optimal trajectories and the Hamiltonian in Figure 1 we can see that indeed the trajectories satisfy the boundary conditions and the Hamiltonian is a constant value.

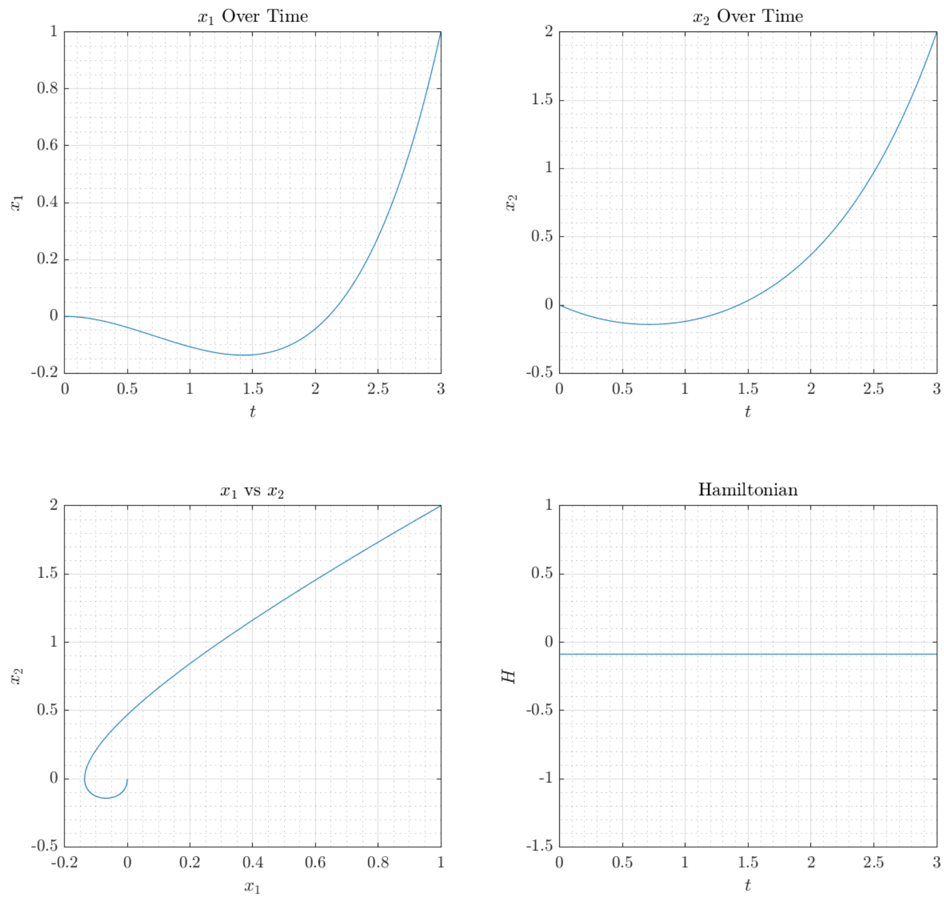

Figure 1: Problem 2(a) optimal trajectories and the Hamiltonian

(b) The Hamiltonian, adjoint equations, and optimal control equation is the same as (a). Although, for this problem $x_2(3)$ is defined to be free which gives rise to a transversality condition of

$$\lambda_2(3) = 0.$$

With this condition we can deduce

$$\lambda_2 = c_1e^t - c_1e^3.$$

Then after that we use the same procedure as problem (a) and we obtain the following expressions

$$x_1(t) = -\frac{c_1}{2}e^t + c_2e^{-t} + c_1e^3$$
$$x_2(t) = -\frac{c_1}{2}e^t - c_2e^{-t} + c_1e^3t + c_3$$

Now we have 3 boundary conditions and 3 unknowns which is solvable.

$$x_1(0) = 0, \qquad x_2(0) = 0, \qquad x_1(3) = 1$$

and the unknowns are

$$c_1, \qquad c_2, \qquad c_3.$$

Using `MATLAB` we obtain (refer to the code in Problem 2: MATLAB Code)

$$\begin{cases} c_1 &= \dfrac{2\,e^3}{3\,e^6 + 4\,e^3 - 1} = 0.0311 \\ c_2 &= -\dfrac{e^3\,(2\,e^3 - 1)}{3\,e^6 + 4\,e^3 - 1} = -0.6101 \\ c_3 &= -\dfrac{2\,e^3\,(e^3 - 1)}{3\,e^6 + 4\,e^3 - 1} = -0.5945 \end{cases}$$

Hence, the optimal control for this problem becomes

$$\boxed{u^* = -\frac{2\,e^3}{3\,e^6 + 4\,e^3 - 1}e^t + \frac{2\,e^3}{3\,e^6 + 4\,e^3 - 1}e^3 = -0.0311e^t + 0.6257.}$$

The minimum performance index is

$$\min J = \frac{1.2896\text{e+}03\,e^6}{\left(3\,e^6 + 4\,e^3 - 1\right)^2}$$

which is simply

$$\boxed{\min J = 0.3128.}$$

If we plot the optimal trajectories and the Hamiltonian in Figure 2 we can see that indeed the trajectories satisfy the boundary conditions and the Hamiltonian is a constant value.

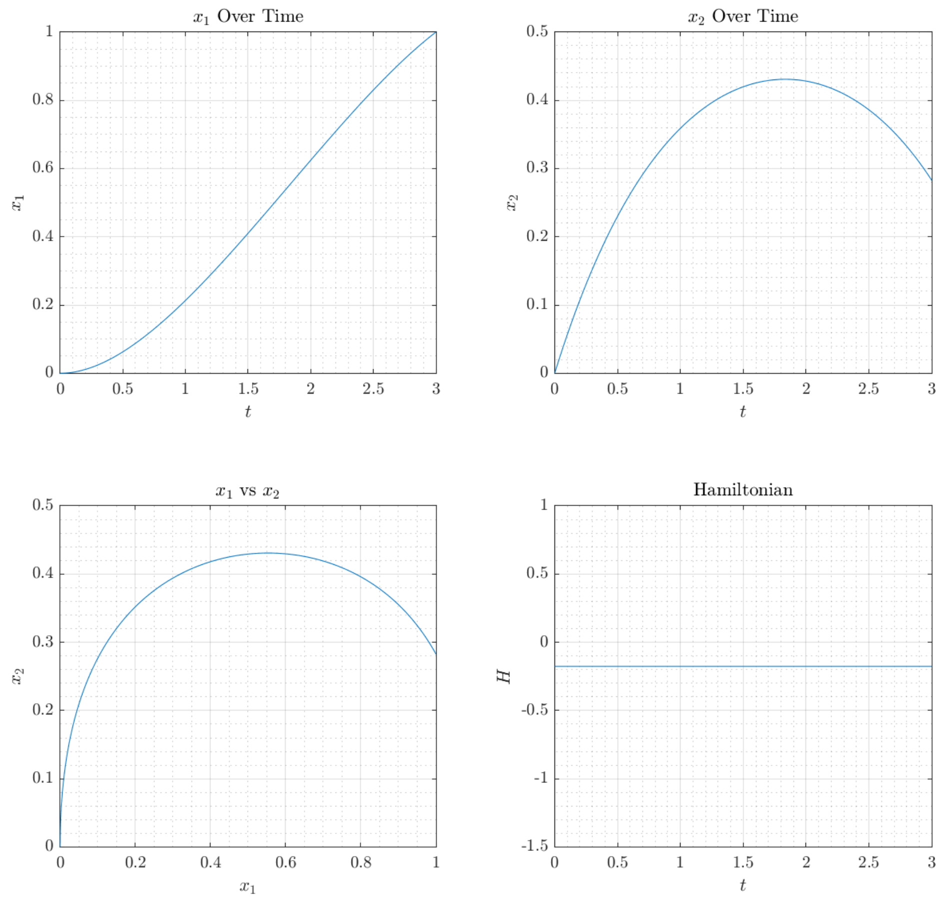

Figure 2: Problem 2(b) optimal trajectories and the Hamiltonian

(c) For this problem, since we want the final states to be close to $[1 \quad 2]^T$ as possible, we modify the performance index by adding a terminal constraint in the form of the following.

$$\boxed{J = (x_1(t_f) - 1)^2 + (x_2(t_f) - 2)^2 + \frac{1}{2}\int_0^{t_f} u^2(t)dt.}$$

Even though we have modified the performance index by adding a terminal penalty term, the Hamiltonian, adjoint equations, and optimal control term are the same. However, we have a different transversality condition compared to problems (a) and (b), which is

$$\lambda(t_f) = \Phi_x^T(x(t_f))$$
$$\text{where } \Phi(x(t_f)) = (x_1(t_f) - 1)^2 + (x_2(t_f) - 2)^2.$$

Thus the transversality condition is

$$\lambda_1(t_f) = 2\left(x_1(t_f) - 1\right)$$
$$\lambda_2(t_f) = 2\left(x_2(t_f) - 2\right).$$

Additionally we have 2 equations from the boundary conditions $x_1(0) = 0$ and $x_2(0) = 0$. We have the exact same setup as problem (a) which is

$$
\begin{aligned}
x_1(t) &= -\frac{c_1}{2}e^t - c_2e^{-t} - ct + c_3 \\
x_2(t) &= -\frac{c_1}{2}e^t + c_2e^{-t} - c
\end{aligned}
$$

and

$$
\begin{aligned}
\lambda_1 &= c \\
\lambda_2 &= c_1e^t + c.
\end{aligned}
$$

Now from the transversality condition and the boundary conditions we can solve for the optimal control and the minimum performance index using `MATLAB` (refer to the code in Problem 2: MATLAB Code).

$$
\begin{cases}
c = -\dfrac{2\,(4\,e^3 - 3)}{7\,e^6 + 12\,e^3 - 12} = -0.0507 \\
c_1 = -\dfrac{2\,(7\,e^3 + 6)}{7\,e^6 + 12\,e^3 - 12} = -0.0960 \\
c_2 = -\dfrac{15\,e^3}{7\,e^6 + 12\,e^3 - 12} = -0.0987 \\
c_3 = -\dfrac{2\,(11\,e^3 + 3)}{7\,e^6 + 12\,e^3 - 12} = -0.1467
\end{cases}
$$

Hence, the optimal control for this problem becomes

$$
\boxed{u^* = \frac{2\,(7\,e^3 + 6)}{7\,e^6 + 12\,e^3 - 12}e^t + \frac{2\,(4\,e^3 - 3)}{7\,e^6 + 12\,e^3 - 12} = 0.0960e^t + 0.0507.}
$$

The minimum performance index is

$$
\min J = \frac{49\,e^{12} + 140\,e^9 + 2.2027\mathrm{e}{+}04\,e^6 + 3.3805\mathrm{e}{+}04\,e^3 + 1.3185\mathrm{e}{+}04}{\left(7\,e^6 + 12\,e^3 - 12\right)^2}
$$

which is simply

$$
\boxed{\min J = 2.0049.}
$$

If we plot the optimal trajectories and the Hamiltonian in Figure 3 we can see that the Hamiltonian is a constant value.

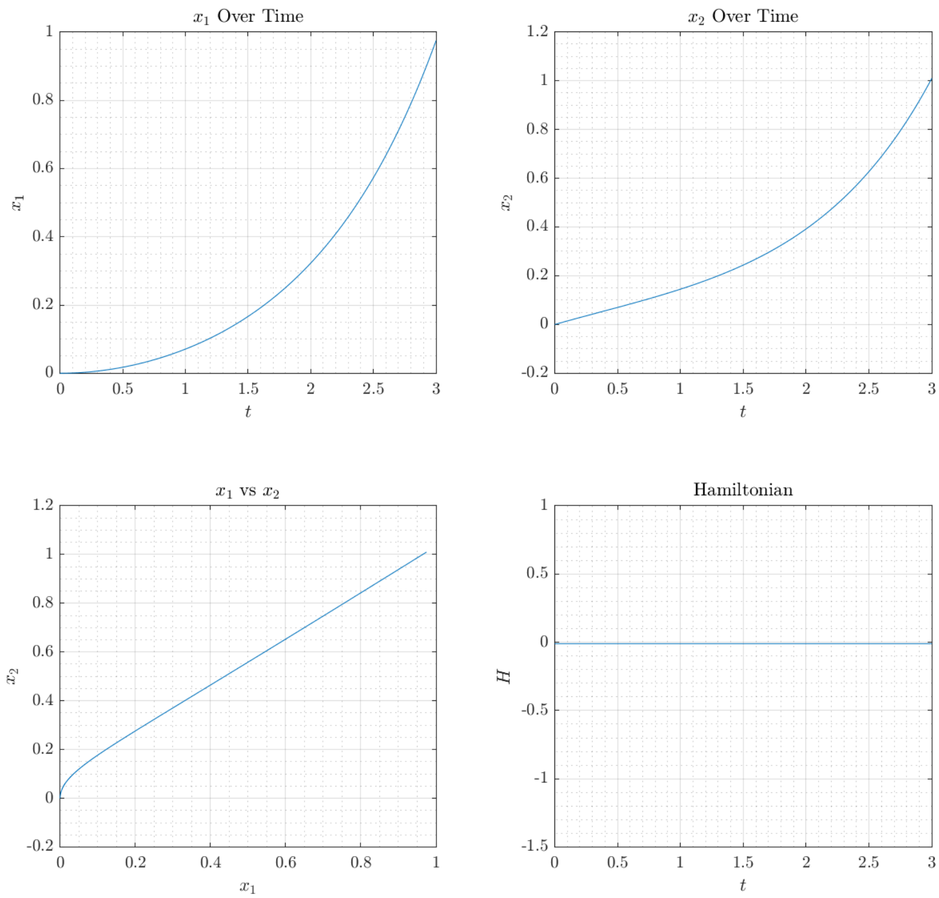

Figure 3: Problem 2(c) optimal trajectories and the Hamiltonian

However, for $x_1(3)$ and $x_2(3)$ the corresponding values are 0.9747 and 1.0102 respectively. The final state for x_1 is relatively close to the desired value of 1, but x_2 is not. However, this is the optimal control for this setup and the overall cost will increase if we try to force the final state to be exactly $[1 \quad 2]^T$. This is because we know that from problem (a) where the final states are exactly 1 and 2 the final cost is larger than what we have for this problem.

(d) For this problem we will apply a terminal constraint Ψ. This terminal cost is in the form of

$$\Psi(x(t_f), t_f) = 2x_1(t_f) + 5x_2(t_f) - 20.$$

Even though, we have added a terminal constraint on the problem the Hamiltonian, adjoint equations, and the optimal control remains the same. However, from the terminal constraint we get a different transversality condition as follows.

$$-\lambda(t_f) = \Psi_x^T \zeta$$

$$\begin{bmatrix} -\lambda_1(t_f) \\ -\lambda_2(t_f) \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix} \zeta$$

This gives us

$$\lambda_2(t_f) = \frac{5}{2}\lambda_1(t_f).$$

And, we have one more equation which is the terminal constraint

$$2x_1(t_f) + 5x_2(t_f) = 20.$$

Additionally we have 2 equations from the boundary conditions $x_1(0) = 0$ and $x_2(0) = 0$. We have the exact same setup as problem (a) which is

$$\begin{aligned} x_1(t) &= -\frac{c_1}{2}e^t - c_2 e^{-t} - ct + c_3 \\ x_2(t) &= -\frac{c_1}{2}e^t + c_2 e^{-t} - c \end{aligned}$$

and

$$\begin{aligned} \lambda_1 &= c \\ \lambda_2 &= c_1 e^t + c. \end{aligned}$$

Now we have 4 unknowns and 4 equations which makes this problem solvable, and therefore, we utilize `MATLAB` to solve this problem. The results we get are as follows.

$$\begin{cases} c = \dfrac{26.6667\, e^6}{-19\, e^6 + 8\, e^3 + 3} = -1.4341 \\ c_1 = \dfrac{40\, e^3}{-19\, e^6 + 8\, e^3 + 3} = -0.1071 \\ c_2 = \dfrac{6.6667\, e^3\, (4\, e^3 + 3)}{-19\, e^6 + 8\, e^3 + 3} = -1.4877 \\ c_3 = \dfrac{13.3333\, e^3\, (2\, e^3 + 3)}{-19\, e^6 + 8\, e^3 + 3} = -1.5412 \end{cases}$$

Hence, the optimal control for this problem becomes

$$\boxed{u^* = -\frac{40\, e^3}{-19\, e^6 + 8\, e^3 + 3}e^t - \frac{26.6667\, e^6}{-19\, e^6 + 8\, e^3 + 3} = 0.1071e^t + 1.4341.}$$

The minimum performance index is

$$\min J = \frac{0.1111\,(36845\,e^{12} + 1.7634\mathrm{e}{+}05\,e^{9} + 1.4703\mathrm{e}{+}06\,e^{6} + 480\,e^{3} + 6525)}{(-19\,e^{6} + 8\,e^{3} + 3)^{2}}$$

which is simply

$$\boxed{\min J = 15.8334.}$$

If we plot the optimal trajectories and the Hamiltonian in Figure 4 we can see that the Hamiltonian is a constant value and that the final states lie on the linear constraint of $2x_1 + 5x_2 = 20$ which is indicated as a red dotted line in the third plot.

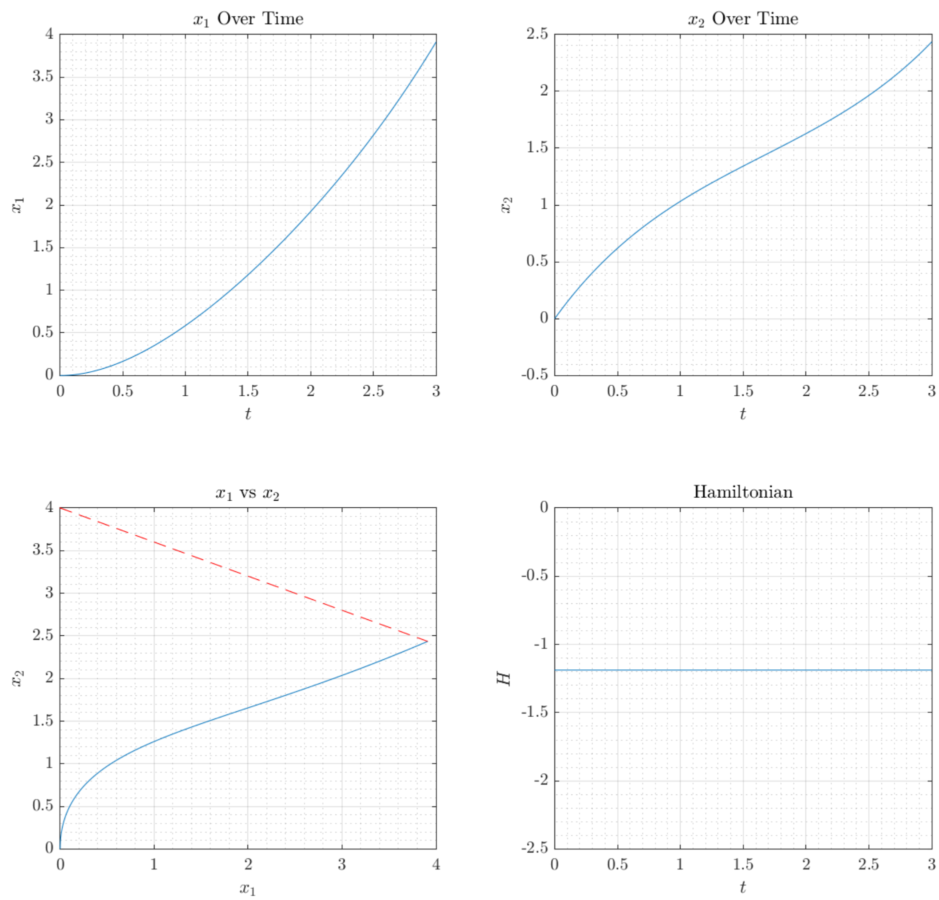

Figure 4: Problem 2(d) optimal trajectories and the Hamiltonian

(e) For this problem the final time is not specified and the terminal constraint will be modified from problem (c).

$$\Psi(x(t_f), t_f) = 2x_1 + 5x - 2 - 20 - \frac{t^2}{2}.$$

Everything will be the same as problem (c) except that the transversality constraint will be in the form of

$$\begin{bmatrix} H(t_f) \\ -\lambda(t_f) \end{bmatrix} = \begin{bmatrix} \Psi_t \\ \Psi_x \end{bmatrix} \zeta$$

$$\begin{bmatrix} H(t_f) \\ -\lambda_1(t_f) \\ -\lambda_2(t_f) \end{bmatrix} = \begin{bmatrix} -t_f \\ 2 \\ 5 \end{bmatrix} \zeta.$$

This gives two equations

$$\lambda_2(t_f) = \frac{5}{2}\lambda_2(t_f)$$
$$H(t_f) = \frac{t_f \lambda_1}{2}.$$

And we have another equation from the terminal constraint itself

$$2x_1(t_f) + 5x_2(t_f) = 20 + \frac{t^2}{2}.$$

Now we know that we have 5 unknowns (additional t_f with the same unknowns as problems (a), (c), and (d)) and 5 equations which makes this problem solvable. Using `MATLAB` we can find the results as follows (refer to the code in the Problem 2: MATLAB Code).

$$\begin{cases} c = -1.8359 \\ c_1 = -0.2547 \\ c_2 = -1.9633 \\ c_3 = -2.0906 \\ t_f = 2.3807 \end{cases}$$

Hence, the optimal control for this problem becomes

$$\boxed{u^* = 0.2547e^t + 1.8359.}$$

The minimum performance index is

$$\boxed{\min J = 10.4803.}$$

If we plot the optimal trajectories and the Hamiltonian in Figure 5 we can see that the Hamiltonian is a constant value and that the final states lie on the linear constraint of $2x_1 + 5x_2 = 20 + \frac{t^2}{2}$ which is indicated as a red dotted line in the third plot.

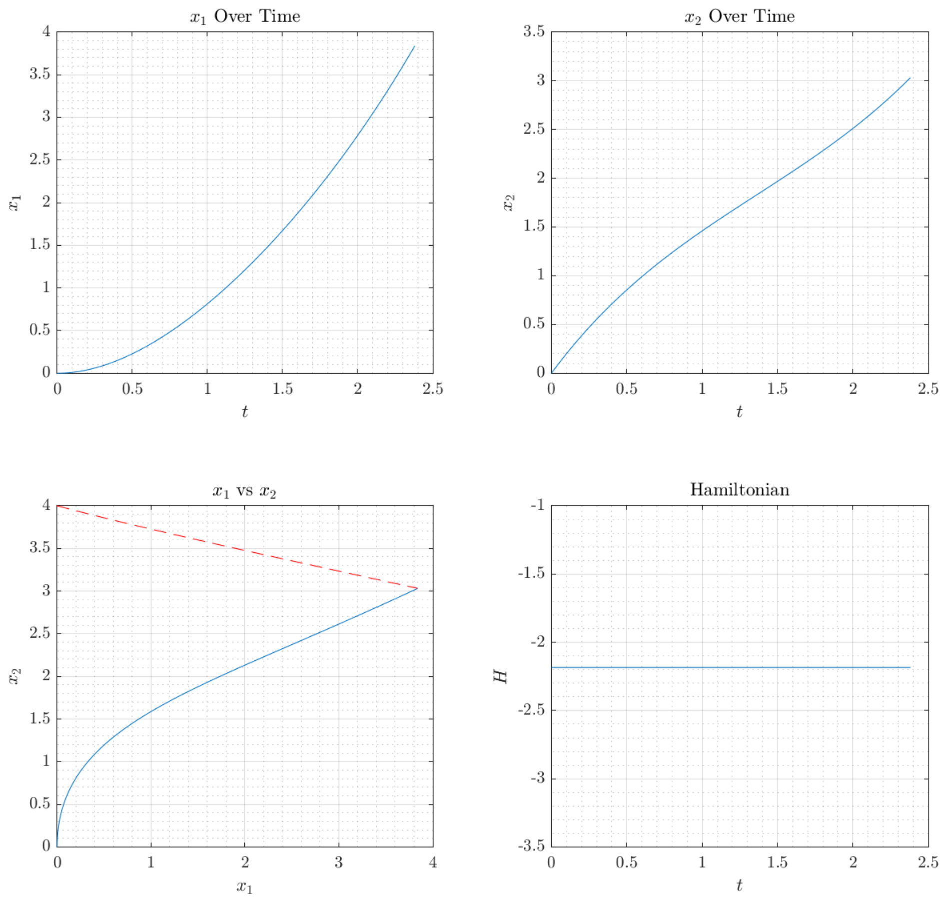

Figure 5: Problem 2(e) optimal trajectories and the Hamiltonian

Problem 3

Consider the system

$$\dot{x} = \cos\theta + u(y)$$
$$\dot{y} = \sin\theta$$

where

$$u(y) = -\alpha(3y - y^3)$$

(a) Compute the minimum-time paths from $x(0) = y(0) = 1$ to the origin. In particular, show that the optimal strategy for $\theta(t)$ satisfies the following differential equations

$$\dot{\theta} = 3\alpha(1 - y^2)\cos^2\theta.$$

What role does α play in the solution?

(b) Find (numerically) the solution for $\alpha = 0.2$. Hint: First show that the initial and final values of θ are related by the expression $\sec\theta(0) - 2\alpha = \sec\theta(t_f)$.

Solution:
For this problem we define the performance index as

$$J = \int_0^{t_f} dt.$$

Then the Hamiltonian becomes

$$H = 1 + \lambda_1\left(\cos\theta - \alpha(3y - y^3)\right) + \lambda_2 \sin\theta.$$

The adjoint equations become

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x} = 0$$
$$\dot{\lambda}_2 = -\frac{\partial H}{\partial y} = 3\alpha\lambda_1 - 3\alpha\lambda_1 y^2$$

and the optimal control becomes

$$\frac{\partial H}{\partial \theta} = -\lambda_1\sin\theta + \lambda_2\cos\theta = 0$$
$$\leftrightarrow \underbrace{\lambda_2\cos\theta = \lambda_1\sin\theta}_{(1)}.$$

and

$$\underbrace{\tan\theta = \frac{\lambda_2}{\lambda_1}.}_{(2)}$$

Now from the adjoint equation we know that

$$\lambda_1 = c = const.$$

And if we take the derivative of the expression (1) we will have

$$\begin{aligned}
\dot{\lambda}_2\cos\theta - \lambda_2\dot{\theta}\sin\theta &= \dot{\lambda}_1\sin\theta + \lambda_1\dot{\theta}\cos\theta \\
\dot{\theta}\left(\lambda_1\cos\theta + \lambda_2\sin\theta\right) &= \lambda_2\cos\theta \\
\dot{\theta}\left(c\cos\theta + \lambda_2\sin\theta\right) &= 3\alpha c(1-y^2)\cos\theta
\end{aligned}$$

from the expression (2) we have

$$\lambda_2 = c\tan\theta$$

and if we plug this into the derivation we have

$$\begin{aligned}
\dot{\theta}c\left(\cos\theta + \tan\theta\sin\theta\right) &= 3\alpha c(1-y^2)\cos\theta \\
\dot{\theta}c\left(\cos\theta + \frac{\sin^2\theta}{\cos\theta}\right) &= 3\alpha c(1-y^2)\cos\theta \\
\frac{\dot{\theta}}{\cos\theta} &= 3\alpha(1-y^2)\cos\theta
\end{aligned}$$

Hence,

$$\boxed{\dot{\theta} = 3\alpha(1-y^2)\cos^2\theta.}$$

In this solution, α plays the role of the constant velocity of this system.

(b) From the answer of problem (a), we can rewrite this as

$$\begin{aligned}
\dot{\theta} &= -\frac{\dot{u}}{\dot{y}}\cos^2\theta \\
\dot{\theta} &= -\dot{u}\frac{\cos^2\theta}{\sin\theta} \\
\int_0^{t_f}\tan\theta\sec\theta d\theta &= -\int_0^{t_f}\dot{u}dt \\
\left[\sec\theta\right]_0^{t_f} &= \left[\alpha\left(3y - y^3\right)\right]_0^{t_f}
\end{aligned}$$

and from the boundary condition of $y(0) = 1$ and $y(t_f) = 0$ we can compute

$$\sec\theta(t_f) - \sec\theta(0) = -2\alpha$$

and hence,

$$\boxed{\sec\theta(t_f) = \sec\theta(0) - 2\alpha.}$$

For this type of free final time optimal control problem we are able to use the `bvp4c` command in `MATLAB` to find the optimal time and control. However, since the final time is free we will implement the treatment of changing the independent variable t to $\tau = t/t_f$ and treating t_f as an auxiliary variable. Then the augmented state and adjoint equations will become $\dot{\tilde{x}} = t_f f(\mathbf{x}, \boldsymbol{\lambda}, \tau)$. Thus, defining the states `y()` for the `bvp4c` as follows we can numerically solve the optimization problem.

$$\begin{cases} \texttt{y(1)} = x, \quad \texttt{y(2)} = y, \quad \texttt{y(3)} = \theta \\ \texttt{y(4)} = \lambda_1, \quad \texttt{y(5)} = \lambda_2, \quad \texttt{y(6)} = t_f \end{cases}$$

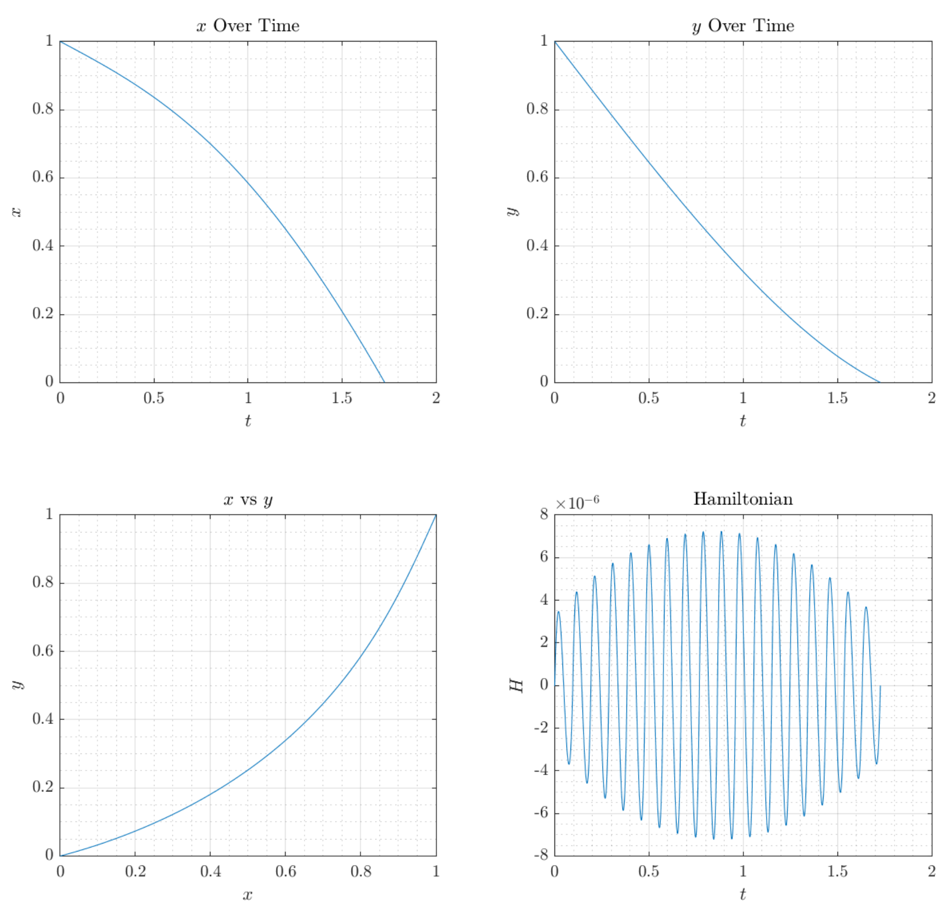

Figure 6: Problem 3 optimal trajectories and the Hamiltonian

From Figure 6 we can observe that the Hamiltonian is not exactly a constant but fluctuating in a very small range. Since the scale of this fluctuation is very small we can treat this as nearly constant.

BVP4C Statistics

```
The solution was obtained on a mesh of 19 points.
The maximum residual is  6.171e-06.
There were 576 calls to the ODE function.
There were 112 calls to the BC function.
```

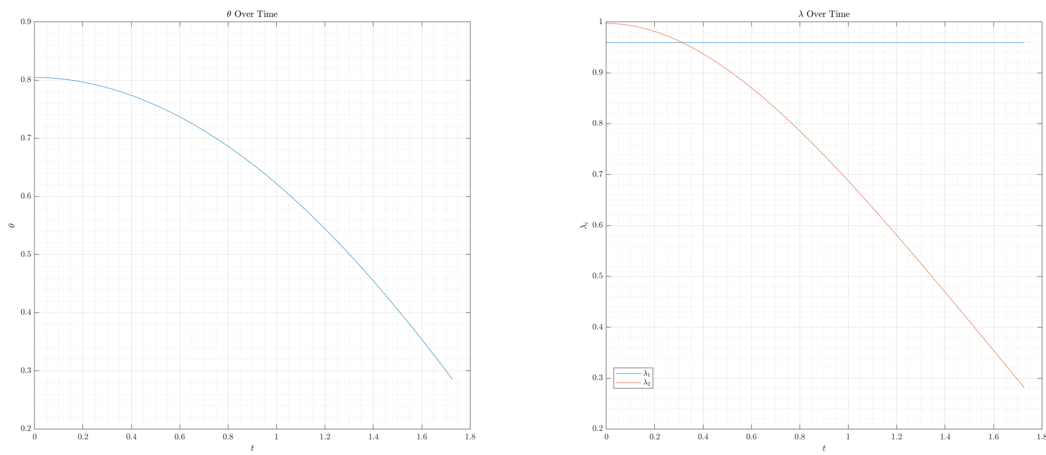

Figure 7: Problem 3 optimal control and costates

and we get the minimum time of

$$\boxed{\min t_f = 1.7270.}$$

Problem 4

Write a `MATLAB` code based on the document uploaded to Canvas to solve the following problem. A ship is located at the point $(x_0, y_0) = (-20, 0)$ml at time $t = 0$ when it encounters a medical emergency and it has to reach the shore as soon as possible. It is known that there is a small city at the location $(x_1, y_1) = (-15, 35.5)$ml with a medical center. As the captain of the ship, you are to determine the fastest possible route to the city. It is assumed that the speed of the ship with respect to the water is constant, $v = 15ml/hr$. You also know the speed and direction of the sea currents in the area, which are given to you from a meteorological satellite as $\vec{v}_c = u(x, y)\hat{\mathbf{i}} + v(x, y)\hat{\mathbf{j}}$.

(a) Derive the necessary conditions for the optimal control strategy, and calculate the optimal path and the time to reach the city, assuming that the currents are constant, given by

$$\vec{v}_c = 2\hat{\mathbf{i}} - 6\hat{\mathbf{j}}$$

Plot the optimal path in the $x - y$ plane along with the vectors showing the direction of the currents.

(b) When you are about to start your dash to the shore, you learn that the doctor in the medical center will be able to fly by helicopter to any point at the shore to pick up the patient. Find the new optimal path and the time to reach the shore, assuming that the contour of the shoreline is known to be

$$\Psi(x, y) = 25 - 0.25x - 0.002x^3 - y = 0.$$

Plot the optimal path in the $x - y$ plane along with the vectors showing the direction of the currents.

(c) An update of the meteorological data from the satellite shows that strong winds have developed in the area and that the currents have changed significantly. The new currents are

$$\vec{v}_c = -(y - 50)\hat{\mathbf{i}} + 2(x - 15)\hat{\mathbf{j}}.$$

Recalculate the optimal control and plot the optimal path in the $x - y$ plane along with the vectors showing the direction of the currents. Plot the optimal steering angle history $\theta^*(t)$.

In call cases, plot the Hamiltonian and verify that it remains zero for all time.

Solution:
The differential equations for this system can be formulated as follows from the given statement.

$$\dot{x} = V\cos\theta + u(x, y)$$
$$\dot{y} = V\sin\theta + v(x, y)$$

Let the performance index and the Hamiltonian for this problem be

$$J = t_f$$
$$H = \lambda_1(V\cos\theta + u) + \lambda_2(V\sin\theta + v).$$

For this problem the costate equations become

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x} = -\lambda_1 u_x - \lambda_2 v_x$$
$$\dot{\lambda}_2 = -\frac{\partial H}{\partial x} = -\lambda_1 u_y - \lambda_2 v_y$$

and the Hamiltonian does not depend on time; therefore,

$$\dot{H} = 0 \to \tan\theta = \frac{\lambda_2}{\lambda_1}$$
$$H(t_f) = -1;$$

From this we can compute the relations

$$\lambda_1 = -\frac{\cos\theta}{V + u\cos\theta + v\sin\theta}$$

$$\lambda_2 = -\frac{\sin\theta}{V + u\cos\theta + v\sin\theta}$$

$$\dot{\theta} = \sin^2\theta u_x + \sin\theta\cos\theta(u_x - v_y) - \cos^2\theta u_y.$$

These are the necessary conditions for this problem. Using the boundary conditions we are able to solve this problem analytically or numerically.

(a) With the constant current of $\vec{v}_c = 2\hat{\mathbf{i}} - 6\hat{\mathbf{j}}$ we have

$$\dot{x} = V\cos\theta + 2$$
$$\dot{y} = V\sin\theta - 6$$

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x} = 0$$
$$\dot{\lambda}_2 = -\frac{\partial H}{\partial x} = 0$$

Referring to the code in Problem 4(a): MATLAB Code we obtain the following results. From Figure 8 we can observe that the Hamiltonian is fluctuating at an infinitesimal value, and therefore can be approximated to a constant 0.

$$\boxed{\min t_f = 1.7578.}$$

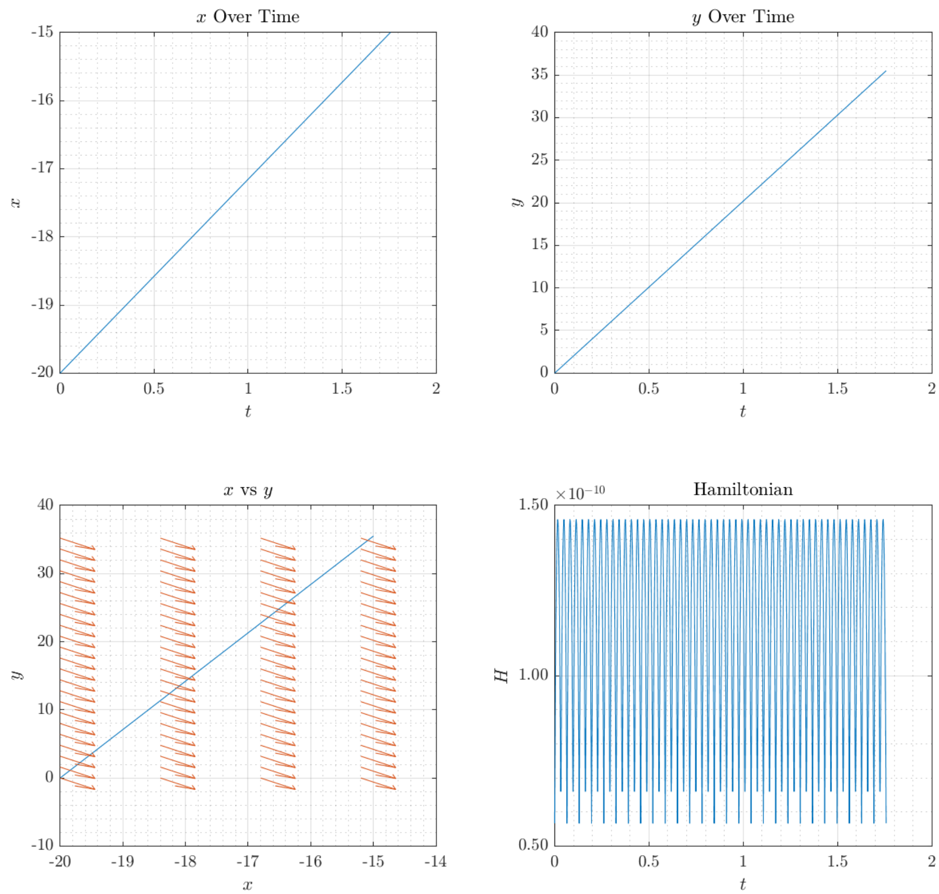

Figure 8: Problem 4(a) optimal trajectories and the Hamiltonian

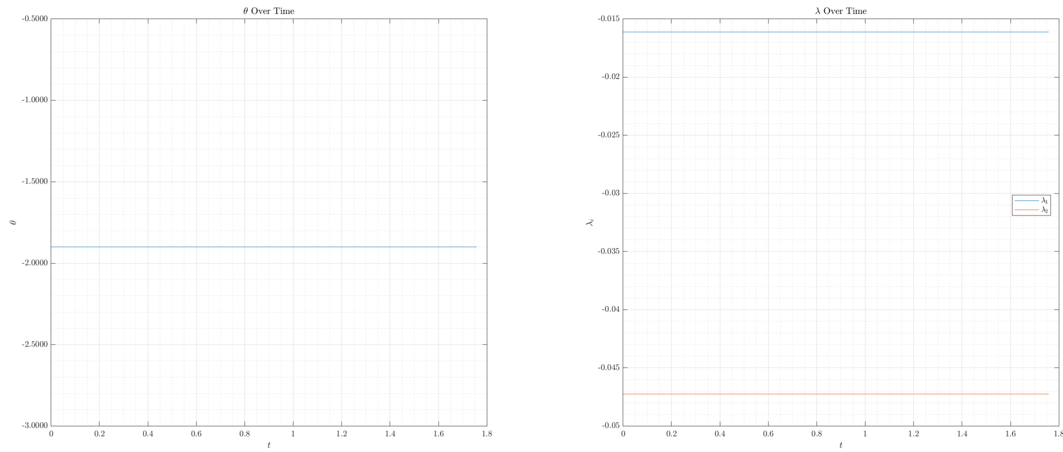

Figure 9: Problem 4(a) optimal control and costate

(b) For this case we have a terminal constraint of

$$\Psi(t_f, x(t_f)) = 25 - 0.25x_f - 0.002x_f^3 - y_f = 0.$$

and a terminal cost of

$$\Phi(t_f, x(t_f)) = (x_f + 15)^2 + (y_f - 35.5)^2$$

Only the transversality condition changes from problem (a), which becomes

$$\begin{bmatrix} H(t_f) + \Phi_t(x(t_f), t_f) \\ -\lambda(t_f) + \Phi_x(x(t_f), t_f) \end{bmatrix} = \begin{bmatrix} \Psi_t^T(x(t_f), t_f) \\ \Psi_x^T(x(t_f), t_f) \end{bmatrix}$$

$$\begin{bmatrix} H(t_f) \\ -\lambda_1(t_f) + 2(x_f + 15) \\ -\lambda_2(t_f) + 2(y_f - 35.5) \end{bmatrix} = \begin{bmatrix} 0 \\ -0.25 - 0.006x_f^2 \\ -1 \end{bmatrix} \zeta$$

$$\text{where } \zeta = \text{ const.}$$

Thus, we have

$$-\lambda_1(t_f) + 2(x_f + 15) = (-0.25 - 0.006x_f^2)(\lambda_2(t_f) - 2(y_f - 35.5)).$$

and we know that

$$25 - 0.25x_f - 0.002x_f^3 - y = 0.$$

Now we can solve the problem numerically. This gave us the result of

$$\boxed{t_f = 3.6181.}$$

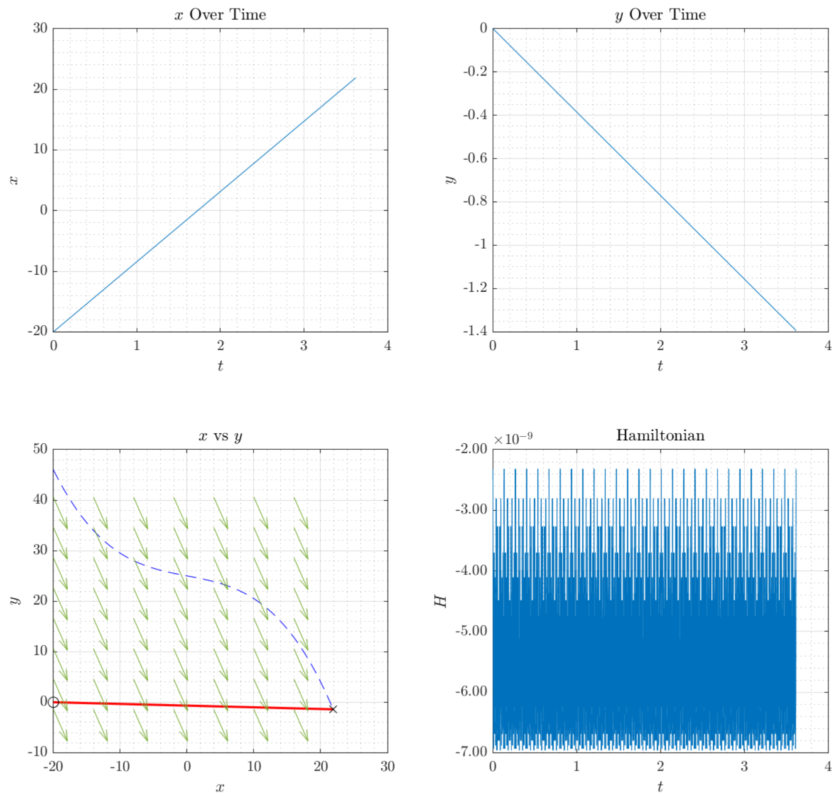

Figure 10: Problem 4(b) optimal trajectories and the Hamiltonian

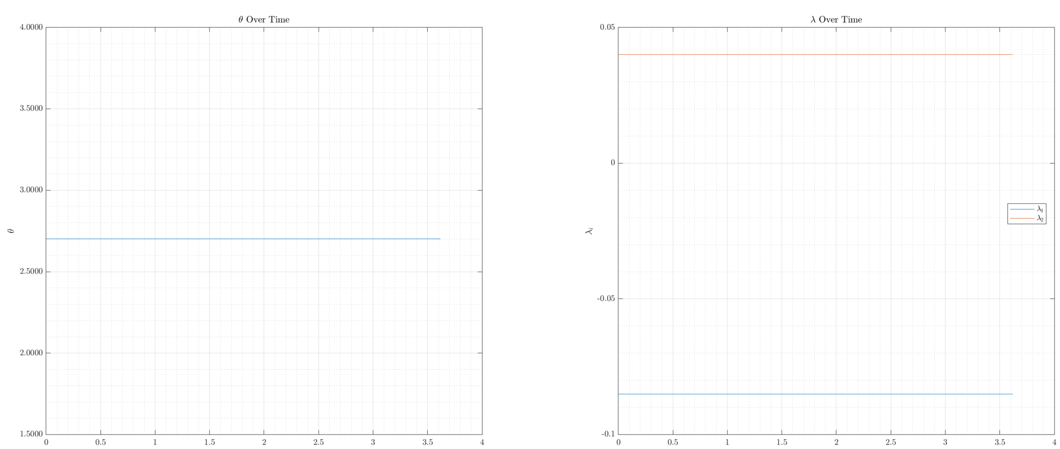

Figure 11: Problem 4(b) optimal control and costate

The red dotted line in Figure 11 is the shoreline and we can see that the final state is on it. Furthermore we can see that the control and the Hamiltonian are both constants as expected. The Hamiltonian is bouncing at a very small range that can be approximated to a constant of 0. The code used to accomplish this problem is in Problem 4(b): MATLAB Code.

(c) For this problem the dynamics constraint is updated to

$$\dot{x} = V\cos\theta - (y - 50)$$
$$\dot{y} = V\sin\theta + 2(x - 15)$$

and the costates are

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x} = -2\lambda_2$$
$$\dot{\lambda}_2 = -\frac{\partial H}{\partial x} = \lambda_1$$

the optimal control remains the same. The transversality conditions and the terminal constraint are the same as problem (b). We can solve this numerically using `MATLAB`. The numerical approach taken is identical to problem 4(b) and refer to the code in Problem 4(c): MATLAB Code.

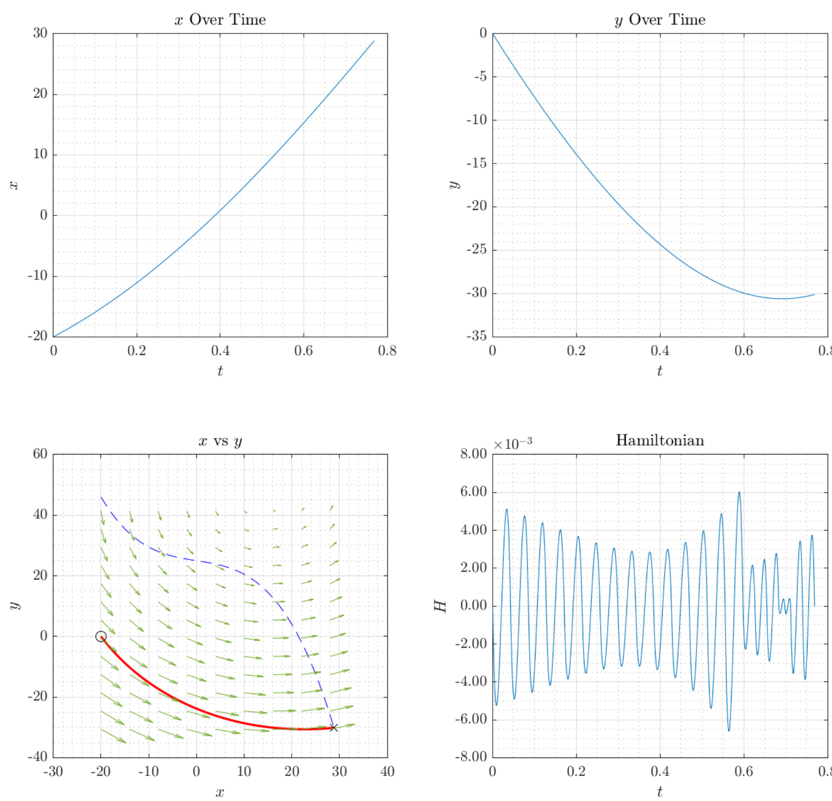

Figure 12: Problem 4(c) optimal trajectories and the Hamiltonian

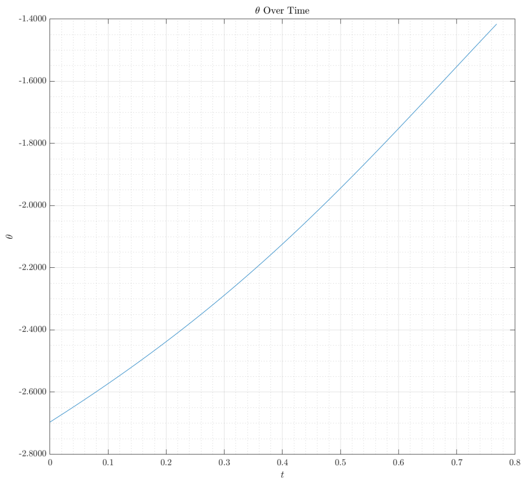

Figure 13: Problem 4(c) optimal control

In Figure 12 we can see that the final state lies on the shoreline and the Hamiltonian fluctuates at a extremely small range which means that it is very close to being a constant 0. The resulting minimal time is

$$\boxed{\min t_f = 0.7687.}$$

Problem 5

A man has a quantity of savings $S > 0$ at a bank. He has no other income and he is trying to find a way to spend all his money of the next time period $[0, t_f]$ in order to maximize his enjoyment. Assume that its instantaneous rate of enjoyment is

$$E = 2\sqrt{r}$$

where r is the spending rate of his fortune. Future enjoyment is counted less today, so he will try to maximize

$$J(r) = \int_0^{t_f} \exp(-\beta t)E(t)dt = \int_0^{t_f} 2\exp(-\beta t)\sqrt{r(t)}dt.$$

In the meantime, the bank gives him some interest proportional to its total capital $x(t)$. THis gives

$$\dot{x}(t) = \alpha x(t) - r(t)$$

with boundary conditions $x(0) = S$ and $x(t_f) = 0$. Assume that $\alpha > \beta > \alpha/2 > 0$.

Find the optimal spending policy $r(t)$ and the optimal capital history $x(t)$.

Solution:
Since this is a maximization problem the Hamiltonian of this problem is

$$H = -2e^{-\beta t}\sqrt{r(t)} + \lambda(\alpha x(t) - r(t)).$$

and the costate equations for this becomes

$$\dot{\lambda} = -\frac{\partial H}{\partial x} = -\lambda\alpha$$

and the optimal control becomes

$$\frac{\partial H}{\partial r} = -2e^{-\beta t}\left(\frac{1}{2}\frac{1}{\sqrt{r}}\right) - \lambda$$
$$\lambda\sqrt{r} + e^{-\beta t} = 0.$$

Since, the final time for this problem is not specified, i.e. free, we have the transversality condition of

$$H(t_f) = 0$$
$$-2e^{-\beta t_f}\sqrt{r(t_f)} + \lambda(\alpha x(t_f) - r(t_f)) = 0.$$

From the costate equation we are able to derive the expression of

$$\dot{\lambda} = -\alpha\lambda$$
$$\lambda(t) = ce^{-\alpha t}.$$

Then from the optimal control and the λ expression we have the following

$$ce^{-\alpha t}\sqrt{r} + e^{-\beta t} = 0$$
$$r(t) = c^{-2}e^{2(\alpha-\beta)t}.$$

Now we can solve the differential equation

$$\dot{x} - \alpha x = c^{-2}e^{2(\alpha-\beta)t}$$

$$x(t) = e^{\alpha t}\left[\int c^{-2}e^{2(\alpha-\beta)t}e^{-\alpha t}dt + c_1\right]$$
$$x(t) = \frac{1}{c^2(\alpha-2\beta)}e^{2(\alpha-\beta)t} + c_1e^{\alpha t}.$$

Then applying the boundary conditions we have

$$c = \frac{e\,(2\,\alpha - 2\,\beta)}{\alpha}$$

$$c_1 = S - \frac{\alpha^2}{e^2\,(\alpha - 2\,\beta)\,(2\,\alpha - 2\,\beta)^2}$$

$$t_f = \frac{\ln\left(-e\,(\alpha-2\,\beta)\left(S - \frac{\alpha^2}{4e^2\,(\alpha-\beta)^2\,(\alpha-2\,\beta)}\right)(2\,\alpha-2\,\beta)\right) - \ln(\alpha)}{\alpha - 2\,\beta}$$

Hence,

$$\boxed{r(t) = \frac{\alpha^2}{2e^2(\alpha-\beta)^2}e^{2(\alpha-\beta)t}.}$$

and

$$\boxed{x(t) = e^{\alpha\,t}\left(S - \frac{\alpha^2}{e^2\,(\alpha-2\,\beta)\,(2\,\alpha-2\,\beta)^2}\right) + \frac{\alpha^2\,e^{t\,(2\,\alpha-2\,\beta)}}{e^2\,(\alpha-2\,\beta)\,(2\,\alpha-2\,\beta)^2}}$$

Appendix

6.1 Problem 2: MATLAB Code

```
% AE6511 Hw5 Problem 2 MATLAB code
% Tomoki Koike
clear all; close all; clc;  % housekeeping commands
set(groot, 'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
%%
%% (a)
syms t c c_1 c_2 c_3 e
assume(c, 'real');
assume(c_1, 'real');
assume(c_2, 'real');
assume(c_3, 'real');
x_1(t) = -c_1/2 * exp(t) - c_2 * exp(-t) - c*t + c_3;
x_2(t) = -c_1/2 * exp(t) + c_2*exp(-t) - c;
% x_1(t) = -c_1/2 * e^(t) - c_2 * e^(-t) - c*t + c_3;
% x_2(t) = -c_1/2 * e^(t) + c_2*e^(-t) - c;
%%
% solve for unknowns with the boundary conditions
eqns = [x_1(0)==0, x_2(0)==0, x_1(3)==1, x_2(3)==2];
sol = solve(eqns, [c, c_1, c_2, c_3]);
sol.c
sol.c_1
sol.c_2
sol.c_3
%%
% Find the minimum performance index
l1 = sol.c;
l2(t) = sol.c_1 * exp(t) + sol.c;
x_1n(t) = subs(x_1, [c, c_1, c_2, c_3], [sol.c sol.c_1 sol.c_2 sol.c_3]);
x_2n(t) = subs(x_2, [c, c_1, c_2], [sol.c sol.c_1 sol.c_2]);
u = -l2;
H(t) = 0.5 * u^2 + l1 * x_2n + l2 * (-x_2n + u);
J_min = 1/2 * int(u^2, 0, 3)
%%
% Plotting the result
% t
tspan = linspace(0, 3, 1000);
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x1 vs t
```

```
    subplot(2,2,1)
    plot(tspan, x_1n(tspan))
    title('$x_1$ Over Time')
    xlabel('$t$')
    ylabel('$x_1$')
    grid on; grid minor; box on;
    % x2 vs t
    subplot(2,2,2)
    plot(tspan, x_2n(tspan))
    title('$x_2$ Over Time')
    xlabel('$t$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % x_1 - x_2
    subplot(2,2,3)
    plot(x_1n(tspan), x_2n(tspan))
    title('$x_1$ vs $x_2$')
    xlabel('$x_1$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H(tspan))
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
% saveas(fig, 'p2a.png')
%%
%% (b)
syms t c_1 c_2 c_3 e
assume(c_1, 'real');
assume(c_2, 'real');
assume(c_3, 'real');
x_1(t) = -c_1/2 * exp(t) - c_2 * exp(-t) + c_1*exp(3)*t + c_3;
x_2(t) = -c_1/2 * exp(t) + c_2*exp(-t) + c_1*exp(3);
% x_1(t) = -c_1/2 * e^(t) - c_2 * e^(-t) + c_1*e^3*t + c_3;
% x_2(t) = -c_1/2 * e^(t) + c_2*e^(-t) + c_1*e^3;
%%
% solve for unknowns with the boundary conditions
eqns = [x_1(0)==0, x_2(0)==0, x_1(3)==1];
sol = solve(eqns, [c_1, c_2, c_3]);
sol.c_1
sol.c_2
sol.c_3
```

```
%%
% Find the minimum performance index
l1 = -sol.c_1 * exp(3);
l2(t) = sol.c_1 * exp(t) + l1;
x_1n(t) = subs(x_1, [c_1, c_2, c_3], [sol.c_1 sol.c_2 sol.c_3]);
x_2n(t) = subs(x_2, [c_1, c_2], [sol.c_1 sol.c_2]);
u = -l2;
H(t) = 0.5 * u^2 + l1 * x_2n + l2 * (-x_2n + u);
J_min = 1/2 * int(u^2, 0, 3)
%%
% Plotting the result
% t
tspan = linspace(0, 3, 1000);
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x1 vs t
    subplot(2,2,1)
    plot(tspan, x_1n(tspan))
    title('$x_1$ Over Time')
    xlabel('$t$')
    ylabel('$x_1$')
    grid on; grid minor; box on;
    % x2 vs t
    subplot(2,2,2)
    plot(tspan, x_2n(tspan))
    title('$x_2$ Over Time')
    xlabel('$t$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % x_1 - x_2
    subplot(2,2,3)
    plot(x_1n(tspan), x_2n(tspan))
    title('$x_1$ vs $x_2$')
    xlabel('$x_1$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H(tspan))
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
saveas(fig, 'p2b.png')
%%
%% (c)
```

```
syms t c c_1 c_2 c_3 e
assume(c, 'real');
assume(c_1, 'real');
assume(c_2, 'real');
assume(c_3, 'real');
x_1(t) = -c_1/2 * exp(t) - c_2 * exp(-t) - c*t + c_3;
x_2(t) = -c_1/2 * exp(t) + c_2*exp(-t) - c;
lambda1 = c;
lambda2(t) = c_1 * exp(t) + c;
% x_1(t) = -c_1/2 * e^(t) - c_2 * e^(-t) - c*t + c_3;
% x_2(t) = -c_1/2 * e^(t) + c_2*e^(-t) - c;
% lambda1 = c;
% lambda2(t) = c_1 * e^(t) + c;
%%
% solve for unknowns with the boundary conditions
TC1 = lambda1 == 2 * (x_1(3) - 1);
TC2 = lambda2(3) == 2 * (x_2(3) - 2);
eqns = [x_1(0)==0, x_2(0)==0, TC1, TC2];
sol = solve(eqns, [c, c_1, c_2, c_3]);
sol.c
sol.c_1
sol.c_2
sol.c_3
%%
% Find the minimum performance index
l1 = sol.c;
l2(t) = sol.c_1 * exp(t) + sol.c;
x_1n(t) = subs(x_1, [c, c_1, c_2, c_3], [sol.c sol.c_1 sol.c_2 sol.c_3]);
x_2n(t) = subs(x_2, [c, c_1, c_2], [sol.c sol.c_1 sol.c_2]);
u = -l2;
H(t) = 0.5 * u^2 + l1 * x_2n + l2 * (-x_2n + u);
Phi = (x_1n - 1)^2 + (x_2n - 2)^2;
J_min = simplify(Phi(3) + 1/2 * int(u^2, 0, 3))
%%
% Plotting the result
% t
tspan = linspace(0, 3, 1000);
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x1 vs t
    subplot(2,2,1)
    plot(tspan, x_1n(tspan))
    title('$x_1$ Over Time')
    xlabel('$t$')
    ylabel('$x_1$')
    grid on; grid minor; box on;
```

```
    % x2 vs t
    subplot(2,2,2)
    plot(tspan, x_2n(tspan))
    title('$x_2$ Over Time')
    xlabel('$t$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % x_1 - x_2
    subplot(2,2,3)
    plot(x_1n(tspan), x_2n(tspan))
    title('$x_1$ vs $x_2$')
    xlabel('$x_1$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H(tspan))
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
saveas(fig, 'p2c.png')
%%
%% (d)
syms t c c_1 c_2 c_3 e
assume(c, 'real');
assume(c_1, 'real');
assume(c_2, 'real');
assume(c_3, 'real');
x_1(t) = -c_1/2 * exp(t) - c_2 * exp(-t) - c*t + c_3;
x_2(t) = -c_1/2 * exp(t) + c_2*exp(-t) - c;
lambda1 = c;
lambda2(t) = c_1 * exp(t) + c;

% x_1(t) = -c_1/2 * e^(t) - c_2 * e^(-t) - c*t + c_3;
% x_2(t) = -c_1/2 * e^(t) + c_2*e^(-t) - c;
% lambda1 = c;
% lambda2(t) = c_1 * e^(t) + c;
%%
% solve for unknowns with the boundary conditions
TC1 = lambda2(3) == 5/2 * lambda1;
TC2 = 2*x_1(3) + 5*x_2(3) == 20;
eqns = [x_1(0)==0, x_2(0)==0, TC1, TC2];
sol = solve(eqns, [c, c_1, c_2, c_3]);
sol.c
```

```
sol.c_1
sol.c_2
sol.c_3
%%
% Find the minimum performance index
l1 = sol.c;
l2(t) = sol.c_1 * exp(t) + sol.c;
x_1n(t) = subs(x_1, [c, c_1, c_2, c_3], [sol.c sol.c_1 sol.c_2 sol.c_3]);
x_2n(t) = subs(x_2, [c, c_1, c_2], [sol.c sol.c_1 sol.c_2]);
u = -l2;
H(t) = 0.5 * u^2 + l1 * x_2n + l2 * (-x_2n + u);
Phi = (x_1n - 1)^2 + (x_2n - 2)^2;
J_min = simplify(Phi(3) + 1/2 * int(u^2, 0, 3))
%%
% Plotting the result
% t
tspan = linspace(0, 3, 1000);
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x1 vs t
    subplot(2,2,1)
    plot(tspan, x_1n(tspan))
    title('$x_1$ Over Time')
    xlabel('$t$')
    ylabel('$x_1$')
    grid on; grid minor; box on;
    % x2 vs t
    subplot(2,2,2)
    plot(tspan, x_2n(tspan))
    title('$x_2$ Over Time')
    xlabel('$t$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % x_1 - x_2
    subplot(2,2,3)
    plot(x_1n(tspan), x_2n(tspan))
    hold on;
    plot(x_1n(tspan), -2/5 * x_1n(tspan) + 4, '--r')
    title('$x_1$ vs $x_2$')
    xlabel('$x_1$')
    ylabel('$x_2$')
    grid on; grid minor; box on; hold off;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H(tspan))
    title('Hamiltonian')
```

```
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
saveas(fig, 'p2d.png')
%%
%% (e)
syms t c c_1 c_2 c_3 e t_f
assume(c, 'real');
assume(c_1, 'real');
assume(c_2, 'real');
assume(c_3, 'real');
x_1(t) = -c_1/2 * exp(t) - c_2 * exp(-t) - c*t + c_3;
x_2(t) = -c_1/2 * exp(t) + c_2*exp(-t) - c;
lambda1 = c;
lambda2(t) = c_1 * exp(t) + c;
U(t) = -lambda2;
HH(t) = 0.5*U^2 + lambda1 * x_2 + lambda2 * (-x_2 + U);
%%
% solve for unknowns with the boundary conditions
TC1 = lambda2(t_f) == 5/2 * lambda1;
TC2 = HH(t_f) == t_f * lambda1 / 2;
TC3 = 2*x_1(t_f) + 5*x_2(t_f) == 20 + t_f^2 / 2;
eqns = [x_1(0)==0, x_2(0)==0, TC1, TC2, TC3];
sol = vpasolve(eqns, [c, c_1, c_2, c_3 t_f]);
sol.c
sol.c_1
sol.c_2
sol.c_3
sol.t_f
%%
% Find the minimum performance index
l1 = sol.c;
l2(t) = sol.c_1 * exp(t) + sol.c;
x_1n(t) = subs(x_1, [c, c_1, c_2, c_3], [sol.c sol.c_1 sol.c_2 sol.c_3]);
x_2n(t) = subs(x_2, [c, c_1, c_2], [sol.c sol.c_1 sol.c_2]);
u = -l2;
H(t) = 0.5 * u^2 + l1 * x_2n + l2 * (-x_2n + u);
Phi = (x_1n - 1)^2 + (x_2n - 2)^2;
J_min = simplify(1/2 * int(u^2, 0, sol.t_f))
%%
% Plotting the result
% t
tspan = linspace(0, sol.t_f, 1000);
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x1 vs t
```

```
    subplot(2,2,1)
    plot(tspan, x_1n(tspan))
    title('$x_1$ Over Time')
    xlabel('$t$')
    ylabel('$x_1$')
    grid on; grid minor; box on;
    % x2 vs t
    subplot(2,2,2)
    plot(tspan, x_2n(tspan))
    title('$x_2$ Over Time')
    xlabel('$t$')
    ylabel('$x_2$')
    grid on; grid minor; box on;
    % x_1 - x_2
    subplot(2,2,3)
    plot(x_1n(tspan), x_2n(tspan))
    hold on;
    plot(x_1n(tspan), -2/5 * x_1n(tspan) + 4 + tspan.^2 / 10, '--r')
    title('$x_1$ vs $x_2$')
    xlabel('$x_1$')
    ylabel('$x_2$')
    grid on; grid minor; box on; hold off;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H(tspan))
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
saveas(fig, 'p2e.png')
```

6.2 Problem 3: MATLAB Code

```
% AE6511 Hw5 Problem 3 MATLAB code
% Tomoki Koike
clear all; close all; clc;  % housekeeping commands
set(groot, 'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
%%
alpha = 0.2;  % global constant for the alpha term
% BVP solution
t_f = 1; % intial guess of final time
```

```matlab
tspan = linspace(0, t_f, 2000);
init_guess = [1, 1, 1, 1, t_f]';
method = 1;

switch method
    case 1  % bvp4c
        solinit = bvpinit(linspace(0, 1, 4), init_guess);
        opts = bvpset('RelTol',1e-5, 'AbsTol',1e-6,'Stats','on','Nmax',500);
        sol = bvp4c(@bvp_ode, @bvp_bc, solinit, opts, alpha);

        % Unpack results
        T = linspace(0,1,1000);
        [xopt,xdopt] = deval(sol,T);
        xopt = xopt.'; xdopt = xdopt.';
        tf_eval = mean(xopt(1,5));
        tf_out = abs(tf_eval);
        tspan = tf_out * T;
        x_sol = xopt(:,1);
        y_sol = xopt(:,2);
        lambda1_sol = xopt(:,3);
        lambda2_sol = xopt(:,4);
        theta_sol = atan2(lambda2_sol,lambda1_sol);
        H = (-1 + lambda1_sol .* xdopt(:,1)./tf_eval ...
            + lambda2_sol .* xdopt(:,2)/tf_eval);

    case 2  % ode45
        optslv = optimset('TolX',1e-7,'TolFun',1e-7, ...
            'Display','off','MaxIter',500);
        xout = ode45(@(t, y) bvp_ode(t, y, alpha), ...
            [0, t_f], init_guess, optslv);
        sol = xout.y'; T = xout.x;
        % Hamiltonian
        xsol = interp1(T,sol(:,1),tspan); ysol = interp1(T,sol(:,2),tspan);
        thetasol = interp1(T,sol(:,3),tspan);
        lambda1 = interp1(T,sol(:,4),tspan); lambda2 = interp1(T,sol(:,5),
           tspan);
        u = -alpha * (3*ysol - ysol.^3);
        H = 1 + lambda1 .* (cos(thetasol) + u) + lambda2 .* sin(thetasol);
end
%%
% Plotting the results

fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x vs t
    subplot(2,2,1)
```

```
    plot(tspan, x_sol)
    title('$x$ Over Time')
    xlabel('$t$')
    ylabel('$x$')
    grid on; grid minor; box on;
    % y vs t
    subplot(2,2,2)
    plot(tspan, y_sol)
    title('$y$ Over Time')
    xlabel('$t$')
    ylabel('$y$')
    grid on; grid minor; box on;
    % x - y
    subplot(2,2,3)
    plot(x_sol, y_sol)
    title('$x$ vs $y$')
    xlabel('$x$')
    ylabel('$y$')
    grid on; grid minor; box on;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H)
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    grid on; grid minor; box on;
saveas(fig,'p3.png');
%%
% Plot control
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, lambda1_sol,'DisplayName','$\lambda_1$')
    hold on;
    plot(tspan, lambda2_sol,'DisplayName','$\lambda_2$')
    title('$\lambda$ Over Time')
    xlabel('$t$')
    ylabel('$\lambda_i$')
    legend('Location','best'); grid on; grid minor; box on; hold off;
saveas(fig,'p3_lambda.png');
%%
% Plot control
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, theta_sol)
    title('$\theta$ Over Time')
    xlabel('$t$')
    ylabel('$\theta$')
```

```
    grid on; grid minor; box on;
saveas(fig,'p3_theta.png');
%% Functions

function dydt = bvp_ode(t, y, alpha)
%{
    Function: bvp_ode
    -------------------
    ODE representation of the optimal control problem to be solved by bvp.

    args:
        t: time
        y: state variables
        alpha: constant (known parameter)
    returns:
        derivative of the state variables
%}

    u = -alpha * (3 * y(2) - y(2)^3);
    theta = atan2(y(4),y(3));
    dydt(1) = cos(theta) + u;
    dydt(2) = sin(theta);
    dydt(3) = 0;
    dydt(4) = 3 * alpha * y(3) - 3 * alpha * y(3) * y(2)^2;
    dydt(5) = 0;
    dydt = dydt' * y(5);
end

function res = bvp_bc(ya, yb, alpha)
%{
    Function: bvp_bc
    ------------------
    Boundary conditions for the optimal control problem solved by bvp.

    args:
        ya: lower boundary conditions
        yb: upper boundary conditions
        alpha: constant (known parameter)
    returns:
        matrix containing all the boundary conditions
%}
    theta_f = atan2(yb(4),yb(3));
    u_f = -alpha*(3*yb(2) - yb(2)^3);
    res(1) = ya(1) - 1;
    res(2) = ya(2) - 1;
```

```
    res(3) = yb(1);
    res(4) = yb(2);
    res(5) = ((yb(3)*(cos(theta_f)+u_f) ...
        +yb(4)*sin(theta_f) - 1)*yb(5));
    res = res';
end
```

6.3 Problem 4(a): MATLAB Code

```
% AE6511 Hw5 Problem 4(a) MATLAB code
% Tomoki Koike
clear all; close all; clc;  % housekeeping commands
set(groot, 'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
%%
% BVP4C
V = 15;
x0 = [0, 0, 0.0055, -0.1121, 1];
mesh = linspace(0, 1, 10);
solinit = bvpinit(mesh, x0);
opts = bvpset('RelTol',1e-5, 'AbsTol',1e-6,'Stats','on','Nmax',10000);
sol = bvp4c(@odefcn,@bcfcn,solinit, opts, V);

% Unpack results
T = linspace(0,1,1000);
[xopt,xdopt] = deval(sol,T);
xopt = xopt.'; xdopt = xdopt.';
tf_eval = mean(xopt(1,5));
tf_out = abs(tf_eval);
tspan = tf_out * T;
x_sol = xopt(:,1);
y_sol = xopt(:,2);
lambda1_sol = xopt(:,3);
lambda2_sol = xopt(:,4);
theta_sol = atan2(lambda2_sol,lambda1_sol);
H = (-1 + lambda1_sol .* xdopt(:,1)./tf_eval ...
    + lambda2_sol .* xdopt(:,2)/tf_eval);

% Phase portrait/Current
X_min = min(x_sol);
X_max = max(x_sol);
Y_min = min(y_sol);
```

```
Y_max = max(y_sol);
[X,Y] = meshgrid(X_min:1.6:X_max,Y_min:1.6:Y_max);
U = 2*ones(size(X));
V = -6*ones(size(Y));

%%
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x vs t
    subplot(2,2,1)
    plot(tspan, x_sol)
    title('$x$ Over Time')
    xlabel('$t$')
    ylabel('$x$')
    grid on; grid minor; box on;
    % y vs t
    subplot(2,2,2)
    plot(tspan, y_sol)
    title('$y$ Over Time')
    xlabel('$t$')
    ylabel('$y$')
    grid on; grid minor; box on;
    % x - y
    subplot(2,2,3)
    plot(x_sol, y_sol)
    hold on;
    quiver(X,Y,U,V)
    title('$x$ vs $y$')
    xlabel('$x$')
    ylabel('$y$')
    grid on; grid minor; box on; hold off;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H)
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    ytickformat('%,.2f')
    grid on; grid minor; box on;
saveas(fig, 'p4a.png');
%%
% Plot costates
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, lambda1_sol,'DisplayName','$\lambda_1$')
    hold on;
    plot(tspan, lambda2_sol,'DisplayName','$\lambda_2$')
```

```
    title('$\lambda$ Over Time')
    xlabel('$t$')
    ylabel('$\lambda_i$')
    legend('Location','best'); grid on; grid minor; box on; hold off;
saveas(fig,'p4a_lambda.png');
%%
% Plot control
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, theta_sol)
    title('$\theta$ Over Time')
    xlabel('$t$')
    ylabel('$\theta$')
    ytickformat('%,.4f')
    grid on; grid minor; box on;
saveas(fig, 'p4a_theta.png');
%% Function

function dxdt = odefcn(t,x,V)
    dxdt = zeros(5,1);
    theta = atan2(x(4),x(3));
    dxdt(1) = V * cos(theta) + 2; % x
    dxdt(2) = V * sin(theta) - 6; % y
    dxdt(3) = 0; % lambda1
    dxdt(4) = 0; % lambda2
    dxdt(5) = 0; % tf
    dxdt = dxdt * x(5);
end

function res = bcfcn(xa,xb,V)
    res = zeros(5,1);
    theta_f = atan2(xb(4),xb(3));
    res(1) = xa(1) + 20; % x(0)
    res(2) = xa(2);  % y(0)
    res(3) = xb(1) + 15; % x(tf)
    res(4) = xb(2) - 35.5; % y(tf)
    res(5) = (-1 + xb(3) * (V*cos(theta_f)+2) ...
        + xb(4)*(V*sin(theta_f)-6))*xb(5); % H(t_f)
end
```

6.4 Problem 4(b): MATLAB Code

```
% AE6511 Hw5 Problem 4(b) MATLAB code
% Tomoki Koike
```

```
clear all; close all; clc;  % housekeeping commands
set(groot, 'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
%%
% BVP4C
V = 15;
x0 = [0, 0, 1, 1, 1];
mesh = linspace(0, 1, 10);
solinit = bvpinit(mesh, x0);
opts = bvpset('RelTol',1e-5, 'AbsTol',1e-6,'Stats','on','Nmax',10000);
sol = bvp4c(@odefcn,@bcfcn,solinit, opts, V);

% Unpack results
T = linspace(0,1,1000);
[xopt,xdopt] = deval(sol,T);
xopt = xopt.'; xdopt = xdopt.';
tf_eval = mean(xopt(1,5));
tf_out = abs(tf_eval);
tspan = tf_out * T;
x_sol = xopt(:,1);
y_sol = xopt(:,2);
lambda1_sol = xopt(:,3);
lambda2_sol = xopt(:,4);
theta_sol = atan2(lambda2_sol,lambda1_sol);
H = (-1 + lambda1_sol .* xdopt(:,1)./tf_eval ...
    + lambda2_sol .* xdopt(:,2)/tf_eval);

% Shoreline
x_shore = linspace(min(x_sol),max(x_sol),1000);
y_shore = -0.002*x_shore.^3 - 0.25*x_shore + 25;

% Phase portrait/Current
X_min = min(min(x_sol),min(x_shore));
X_max = max(max(x_sol),max(x_shore));
Y_min = min(min(y_sol),min(y_shore));
Y_max = max(max(y_sol),max(y_shore));
[X,Y] = meshgrid(X_min:6:X_max,Y_min:6:Y_max);
U = 2*ones(size(X));
V = -6*ones(size(Y));
%%
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x vs t
    subplot(2,2,1)
    plot(tspan, x_sol)
```

```
    title('$x$ Over Time')
    xlabel('$t$')
    ylabel('$x$')
    grid on; grid minor; box on;
    % y vs t
    subplot(2,2,2)
    plot(tspan, y_sol)
    title('$y$ Over Time')
    xlabel('$t$')
    ylabel('$y$')
    grid on; grid minor; box on;
    % x - y with current
    subplot(2,2,3)
    plot(x_sol, y_sol,'-r',LineWidth=1.5)
    hold on;
    plot(x_sol(1),y_sol(1),'ok',MarkerSize=7)
    plot(x_sol(end),y_sol(end),'xk',MarkerSize=7)
    plot(x_shore,y_shore, '--b')
    quiver(X,Y,U,V)
    title('$x$ vs $y$')
    xlabel('$x$')
    ylabel('$y$')
    grid on; grid minor; box on; hold off;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H)
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    ytickformat('%,.2f')
    grid on; grid minor; box on;
saveas(fig, 'p4b.png');
%%
% Plot costates
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, lambda1_sol,'DisplayName','$\lambda_1$')
    hold on;
    plot(tspan, lambda2_sol,'DisplayName','$\lambda_2$')
    title('$\lambda$ Over Time')
    xlabel('$t$')
    ylabel('$\lambda_i$')
    legend('Location','best'); grid on; grid minor; box on; hold off;
saveas(fig,'p4b_lambda.png');
%%
% Plot control
```

```
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, theta_sol)
    title('$\theta$ Over Time')
    xlabel('$t$')
    ylabel('$\theta$')
    ytickformat('%,.4f')
    grid on; grid minor; box on;
saveas(fig, 'p4b_theta.png');
%% Function

function dxdt = odefcn(t,x,V)
    dxdt = zeros(5,1);
    theta = atan2(x(4),x(3));
    dxdt(1) = V * cos(theta) + 2; % x
    dxdt(2) = V * sin(theta) - 6; % y
    dxdt(3) = 0; % lambda1
    dxdt(4) = 0; % lambda2
    dxdt(5) = 0; % tf
    dxdt = dxdt * x(5);
end

function res = bcfcn(xa,xb,V)
    res = zeros(5,1);
    theta_f = atan2(xb(4),xb(3));
    res(1) = xa(1) + 20; % x(0)
    res(2) = xa(2);  % y(0)
    res(3) = ((-0.25 - 0.006 * xb(1)^2)*(xb(4)-2*(xb(2)-35.5) ...
        + xb(3) - 2*(xb(1)+15))); % -l = Psi_x
    res(4) = 25 - 0.25*xb(1) - 0.002*xb(1)^3 - xb(2); % Psi(t_f)
    res(5) = (-1 + xb(3) * (V*cos(theta_f)+2) ...
        + xb(4)*(V*sin(theta_f)-6)) * xb(5); % H(t_f)
end
```

6.5 Problem 4(c): MATLAB Code

```
% AE6511 Hw5 Problem 4(c) MATLAB code
% Tomoki Koike
clear all; close all; clc;  % housekeeping commands
set(groot, 'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
%%
% BVP4C
```

```
V = 15;
x0 = [0, 0, 2, 3, 1];
mesh = linspace(0, 1, 10);
solinit = bvpinit(mesh, x0);
opts = bvpset('RelTol',1e-5, 'AbsTol',1e-6,'Stats','on','Nmax',10000);
sol = bvp4c(@odefcn,@bcfcn,solinit, opts, V);

% Unpack results
T = linspace(0,1,50000);
[xopt,xdopt] = deval(sol,T);
xopt = xopt.'; xdopt = xdopt.';
tf_eval = mean(xopt(1,5));
tf_out = abs(tf_eval);
tspan = tf_out * T;
x_sol = xopt(:,1);
y_sol = xopt(:,2);
lambda1_sol = xopt(:,3);
lambda2_sol = xopt(:,4);
theta_sol = atan2(lambda2_sol,lambda1_sol);
H = (-1 + lambda1_sol .* xdopt(:,1)./tf_eval ...
    + lambda2_sol .* xdopt(:,2)/tf_eval);

% Shoreline
x_shore = linspace(min(x_sol),max(x_sol),length(tspan));
y_shore = -0.002*x_shore.^3 - 0.25*x_shore + 25;

% Phase portrait/Current
X_min = min(min(x_sol),min(x_shore));
X_max = max(max(x_sol),max(x_shore));
Y_min = min(min(y_sol),min(y_shore));
Y_max = max(max(y_sol),max(y_shore));
[X,Y] = meshgrid(X_min:6:X_max,Y_min:6:Y_max);
U = -Y + 50;
V = 2*X - 30;
%%
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    % x vs t
    subplot(2,2,1)
    plot(tspan, x_sol)
    title('$x$ Over Time')
    xlabel('$t$')
    ylabel('$x$')
    grid on; grid minor; box on;
    % y vs t
    subplot(2,2,2)
```

```
    plot(tspan, y_sol)
    title('$y$ Over Time')
    xlabel('$t$')
    ylabel('$y$')
    grid on; grid minor; box on;
    % x - y
    subplot(2,2,3)
    plot(x_sol, y_sol,'-r',LineWidth=1.5)
    hold on;
    plot(x_sol(1),y_sol(1),'ok',MarkerSize=7)
    plot(x_sol(end),y_sol(end),'xk',MarkerSize=7)
    plot(x_shore,y_shore, '--b')
    quiver(X,Y,U,V)
    title('$x$ vs $y$')
    xlabel('$x$')
    ylabel('$y$')
    grid on; grid minor; box on; hold off;
    % Hamiltonian
    subplot(2,2,4)
    plot(tspan, H)
    title('Hamiltonian')
    xlabel('$t$')
    ylabel('$H$')
    ytickformat('%,.2f')
    grid on; grid minor; box on;
saveas(fig, 'p4c.png');
%%
% Plot costates
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, lambda1_sol,'DisplayName','$\lambda_1$')
    hold on;
    plot(tspan, lambda2_sol,'DisplayName','$\lambda_2$')
    title('$\lambda$ Over Time')
    xlabel('$t$')
    ylabel('$\lambda_i$')
    legend('Location','best'); grid on; grid minor; box on; hold off;
saveas(fig,'p4c_lambda.png');
%%
% Plot control
fig = figure("Renderer","painters","Position",[60 60 900 800]);
    plot(tspan, theta_sol)
    title('$\theta$ Over Time')
    xlabel('$t$')
    ylabel('$\theta$')
    ytickformat('%,.4f')
```

```matlab
    grid on; grid minor; box on;
saveas(fig, 'p4c_theta.png');
%% Function

function dxdt = odefcn(t,x,V)
    dxdt = zeros(5,1);
    theta = atan2(x(4),x(3));
    dxdt(1) = V * cos(theta) - x(2) + 50; % x
    dxdt(2) = V * sin(theta) + 2*x(1) - 30; % y
    dxdt(3) = -2*x(4); % lambda1
    dxdt(4) = x(3); % lambda2
    dxdt(5) = 0; % tf
    dxdt = dxdt * x(5);
end

function res = bcfcn(xa,xb,V)
    res = zeros(5,1);
    theta_f = atan2(xb(4),xb(3));
    res(1) = xa(1) + 20; % x(0)
    res(2) = xa(2);  % y(0)
    res(3) = ((-0.25 - 0.006 * xb(1)^2)*(xb(4)-2*(xb(2)-35.5) ...
        + xb(3) - 2*(xb(1)+15))); % -l = Psi_x
    res(4) = (25 - 0.25*xb(1) - 0.002*xb(1)^3 - xb(2)); % Psi(t_f)
    res(5) = (-1 + xb(3) * (V*cos(theta_f)-xb(2)+50) ...
        + xb(4)*(V*sin(theta_f)+2*xb(1)-30)) * xb(5); % H(t_f)
end
```