HW1

Kyle Morgenstein
kjm3887

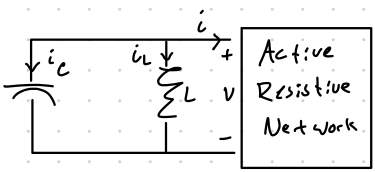
1

$$I\ddot{q}_1 + MgL\sin q_1 + k(q_1 - q_2) = 0$$
$$J\ddot{q}_2 - k(q_1 - q_2) = u$$
$$\Downarrow$$
$$\ddot{q}_1 = -I^{-1}(MgL\sin q_1 + k(q_1 - q_2))$$
$$\ddot{q}_2 = J^{-1}(u - k(q_1 - q_2))$$

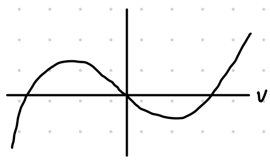
State Variables

$$q = [q_1 \;\; q_2 \;\; \dot{q}_1 \;\; \dot{q}_2]$$
$$\dot{q} = [\dot{q}_1 \;\; \dot{q}_2 \;\; \ddot{q}_1 \;\; \ddot{q}_2]$$
$$\dot{q} = f(q) \;, \quad f(q) = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ -I^{-1}(MgL\sin q_1 + k(q_1 - q_2)) \\ J^{-1}(u - k(q_1 - q_2)) \end{bmatrix}$$

2



$i = h(v)$



$$i_C + i_L + i_{active} = 0$$
$$i_C = C\dot{v} \qquad i_{active} = h(v)$$
$$v = L\dot{i}_L$$

$$C\dot{v} + i_L + h(v) = 0$$

$$x = \begin{bmatrix} v \\ i_L \end{bmatrix} \quad \dot{x} = \begin{bmatrix} \dot{v} \\ \dot{i}_L \end{bmatrix} = \begin{bmatrix} -i_L/C \;\; -h(v)/C \\ v/L \end{bmatrix}$$

③ $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$$\dot{x} = \begin{bmatrix} x_2 \\ -x_1 + \frac{1}{16} x_1^5 - x_2 \end{bmatrix}$$

a) equilibrium points: $\dot{x} = 0 \rightarrow x_2 = 0$, $x_1^5/16 - x_1 = 0$

$x_1^4/16 - 1 = 0$

$(x_1^2/4 - 1)(x_1^2/4 + 1) = 0$

$(x_1/2 - 1)(x_1/2 + 1)(x_1^2/4 + 1) = 0$

$x_1 = \pm 2, \pm 2i, 0$

i) $x_1 = \pm 2$, $x_2 = 0 \rightarrow$ saddle

ii) $x_1 = 0$, $x_2 = 0 \rightarrow$ stable node

iii) $x_1 = \pm 2i$, $x_2 = 0 \rightarrow$ circular

b) phase portrait

see notebook below

④

$$\ddot{y} + ky + c\dot{y} + \eta(y, \dot{y}) = 0$$

$$\eta(y,\dot{y}) = \begin{cases} \mu_k mg \sin(\dot{y}) & |\dot{y}| > 0 \\ -ky & \dot{y} = 0, |y| \leq \mu_s mg/k \\ -\mu_s mg \sin(y) & \dot{y} = 0, |y| > \mu_s mg/k \end{cases}$$

$x = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$ $\dot{x} = \begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix}$

$$\ddot{y} = -ky - c\dot{y} - \eta(y, \dot{y})$$

see notebook below

⑤ $\dot{x} = \begin{bmatrix} x_2 \\ u \end{bmatrix}$, $u \in \{-1, 1\}$

$x_1 = x$ $\quad \dot{x}_1 = v$ $\quad x(t) = x_0 + v_0 t + vt + \frac{1}{2} a t^2$

$x_2 = v$ $\quad \dot{x}_2 = a$

see notebook below

hw1

September 10, 2025

1 Homework 1

due 09/10/2025 23:59

[28]:
```python
# deps
from typing import Callable
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

from utils import phase_portrait
```

1.1 Problem 3

1) Find equilibrium points for

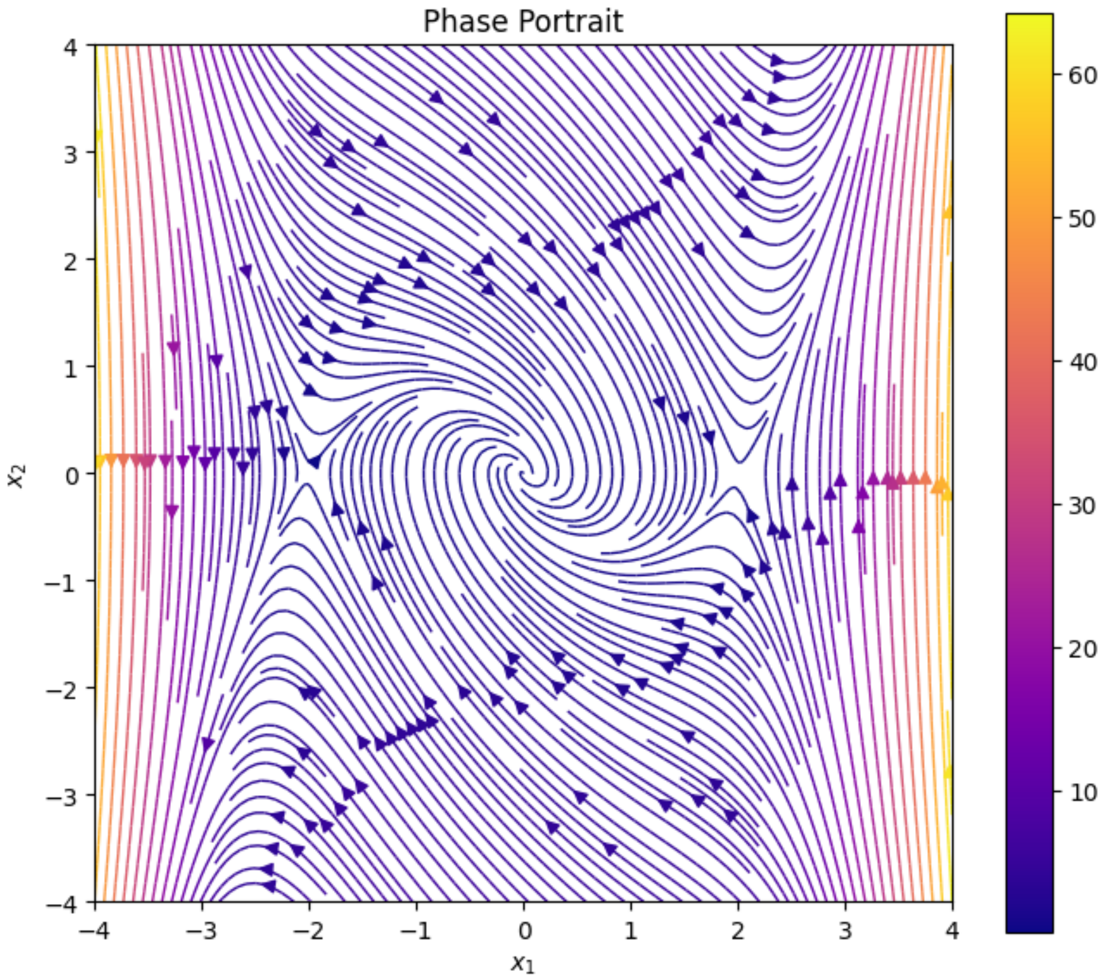
$$\dot{x}_1 = x_2 \tag{1}$$

$$\dot{x}_2 = -x_1 + \frac{1}{16}x_1^5 - x_2 \tag{2}$$

2) Plot the phase portrait

[29]:
```python
def dynamics(
    t: float | None,
    state: list,
) -> list:
    x1, x2 = state
    dx1 = x2
    dx2 = -x1 + x1**5/16 - x2
    return [dx1, dx2]

phase_portrait(x_max=4, n_pts=100, dynamics=dynamics, save=True)
```



1.2 Problem 4

1) Sketch the phase portrait for

$$\dot{x}_1 = x_2 \tag{3}$$
$$\dot{x}_2 = -kx_1 - cx_2 - \eta(x_1, x_2) \tag{4}$$

2) Describe the behavior qualitatively

   The phase portrait has an unstable node at the origin, with circular orbits on either side of the x_1 plane. The behavior is disjoint about x_2 given the sign function in the dynamics.

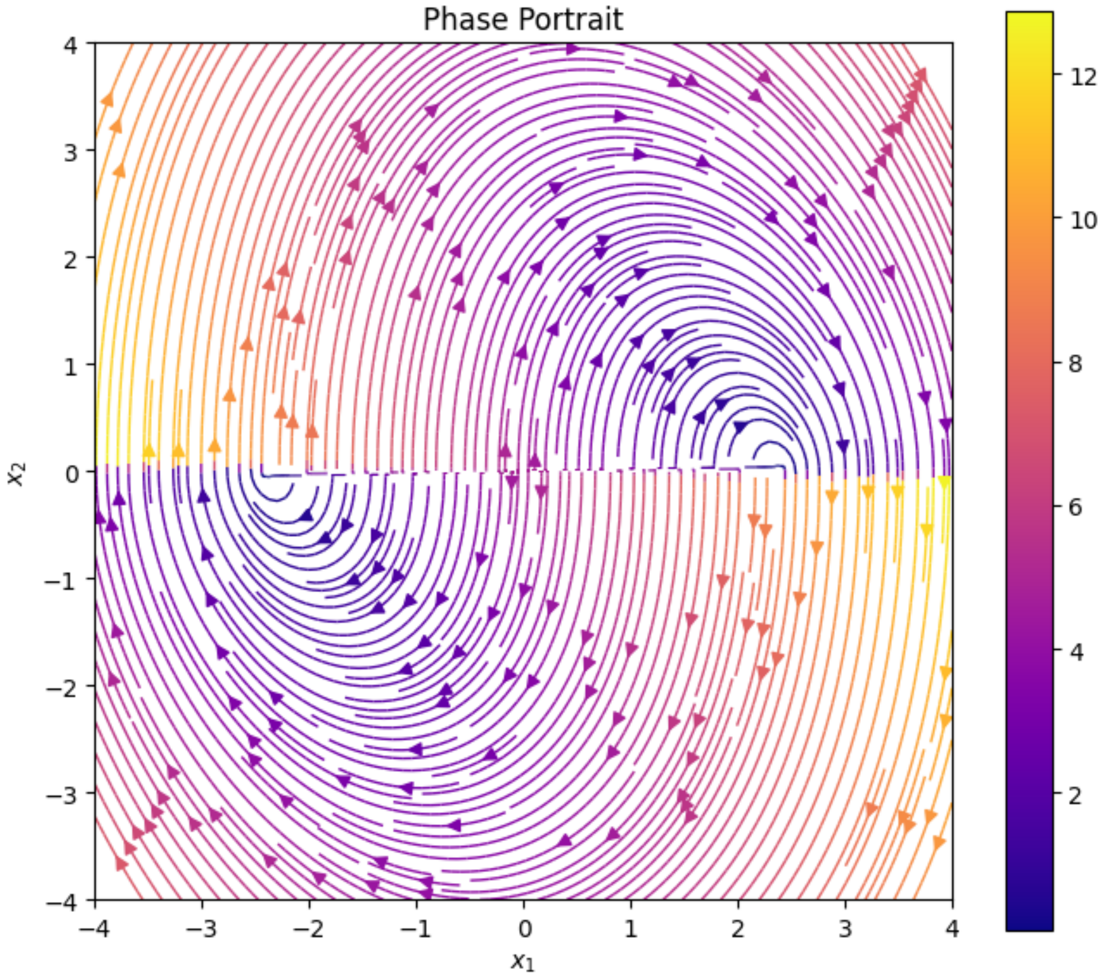
[30]:
```
def sign(n):
    return np.where(
        n > 0,
        1,
        np.where(
```

```python
            n < 0,
            -1,
            0
        )
    )

def eta(
    state: list,
    k: float,
    m: float,
    g: float,
    mu_k: float,
    mu_s: float,
):
    x1, x2 = state
    return np.where(
        np.abs(x2) > 0,
        mu_k * m * g * sign(x2),
        np.where(
            np.abs(x1) <= mu_s * m * g / k,
            -k * x1,
            mu_s * m * g * sign(x1)
        )
    )

def dynamics(
    t: float | None,
    state: list,
    k: float = 2.0,
    c: float = 1.0,
    m: float = 1.0,
    g: float = -9.8,
    mu_k: float = 0.5,
    mu_s: float = 0.5,
) -> list:
    x1, x2 = state
    dx1 = x2
    dx2 = -k * x1 -c * x2 - eta(state, k, m, g, mu_k, mu_s)
    return [dx1, dx2]

phase_portrait(x_max=4, n_pts=100, dynamics=dynamics, save=True)
```



1.3 Problem 5

1) Sketch phase portraits for $u = 1$ and $u = -1$

$$\dot{x}_1 = x_2 \tag{5}$$
$$\dot{x}_2 = u \tag{6}$$

2) Develop graphical control strategy

Define switching curve $x_1 = \pm\frac{x_2^2}{2}$. Then, from anywhere in $x_1 - x_2$ plane, the origin can be obtained by following the corresponding curve. When above the curve, apply negative control input; when below the curve, apply positive control input. This is a "bang bang" controller.

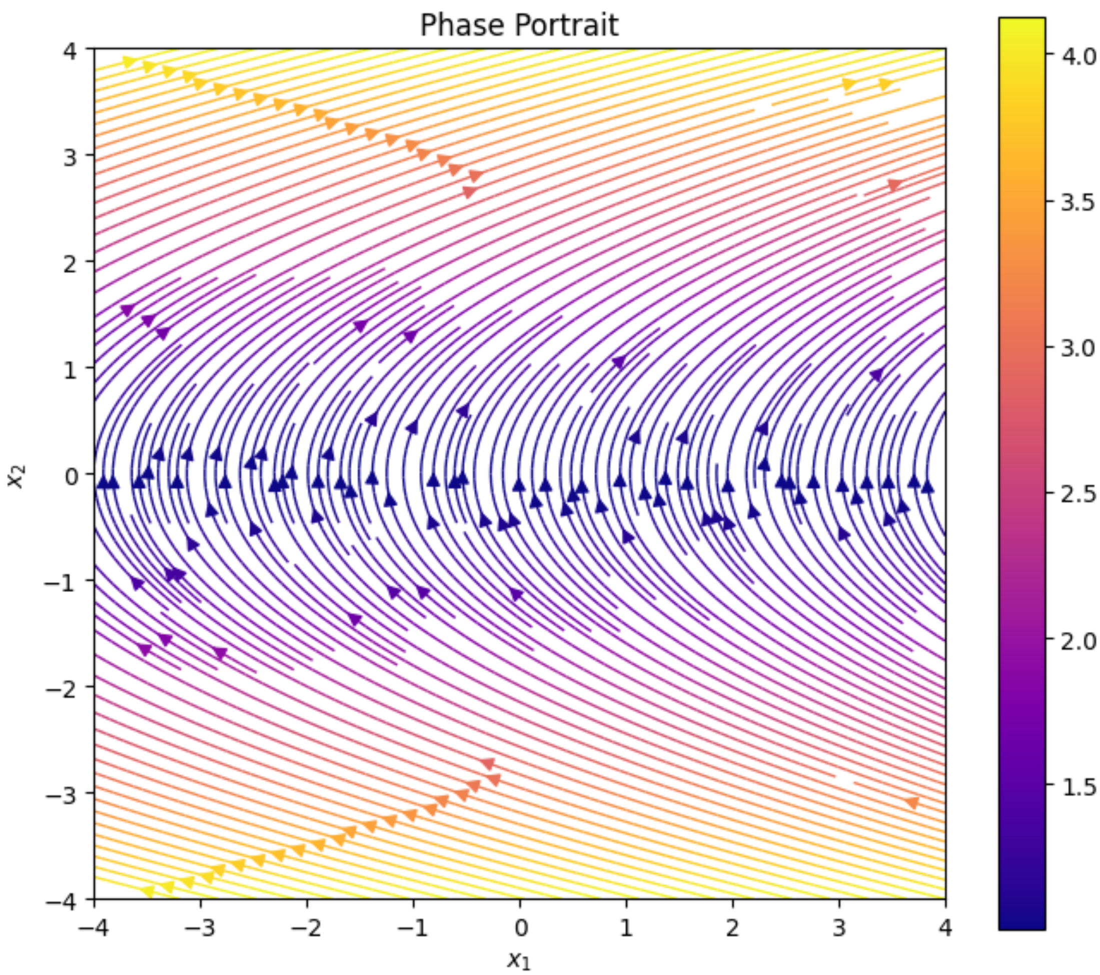
[31]:
```
def dynamics(
        t: float | None,
        state: list,
        u: float,
```

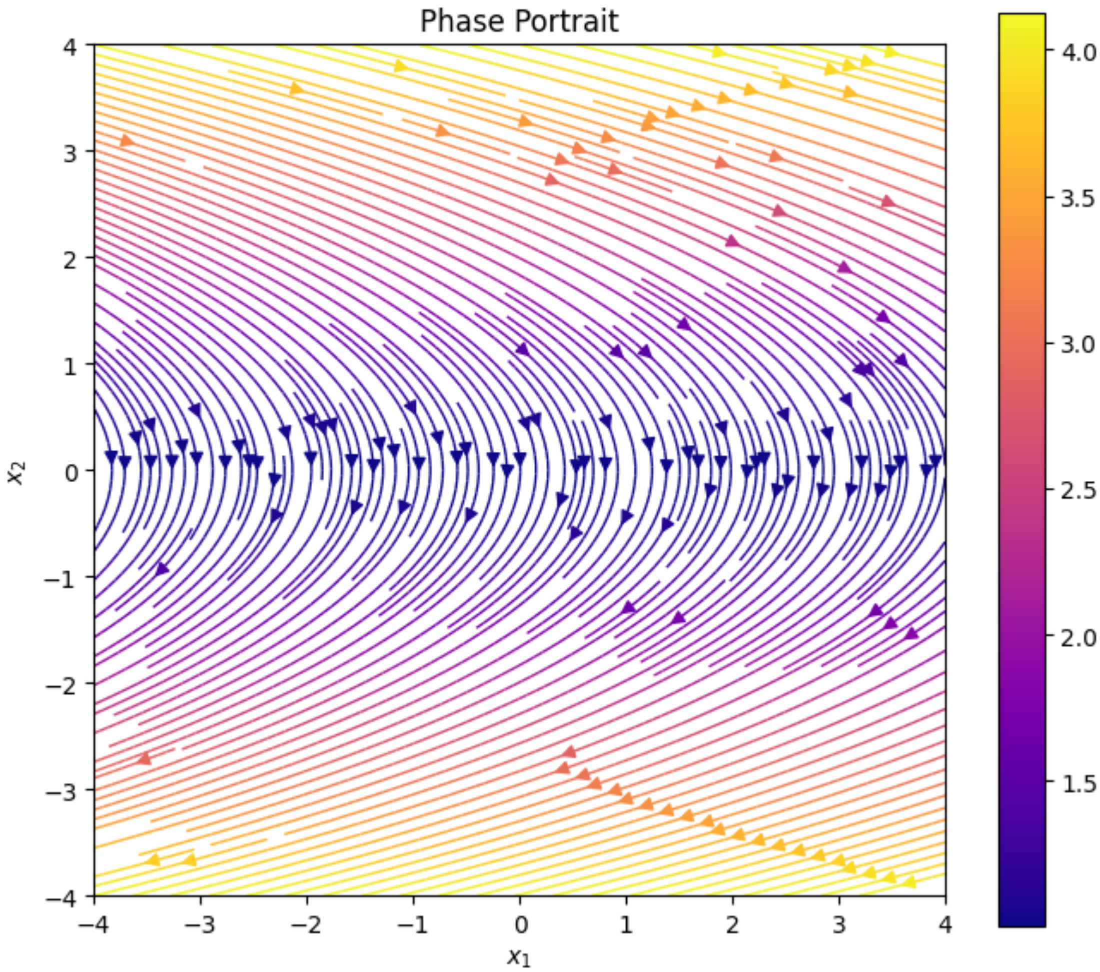
```
) -> list:
    x1, x2 = state
    dx1 = x2
    dx2 = u * np.ones_like(x2)
    return [dx1, dx2]

def set_u(u) -> Callable[..., list]:
    def wrapped_dynamics(t, state):# -> list:
        return dynamics(t, state, u)
    return wrapped_dynamics

dynamics_plus = set_u(1)
dynamics_minus = set_u(-1)

phase_portrait(x_max=4, n_pts=100, dynamics=dynamics_plus, save=True)
phase_portrait(x_max=4, n_pts=100, dynamics=dynamics_minus, save=True)
```




Phase Portrait
4
3
2
1
0
−1
−2
−3
−4
x_2
−4
−3
−2
−1
0
1
2
3
4
x_1
4.0
3.5
3.0
2.5
2.0
1.5