lab1

September 20, 2021

1

1.1 №1.

1.1.1 1.

:

$$\ddot{x} + \omega^2 \cdot x = 0$$

2- :

$$\begin{cases} \dfrac{dx}{dt} = p \\ \dfrac{dp}{dt} = -\omega^2 \cdot x \end{cases}$$

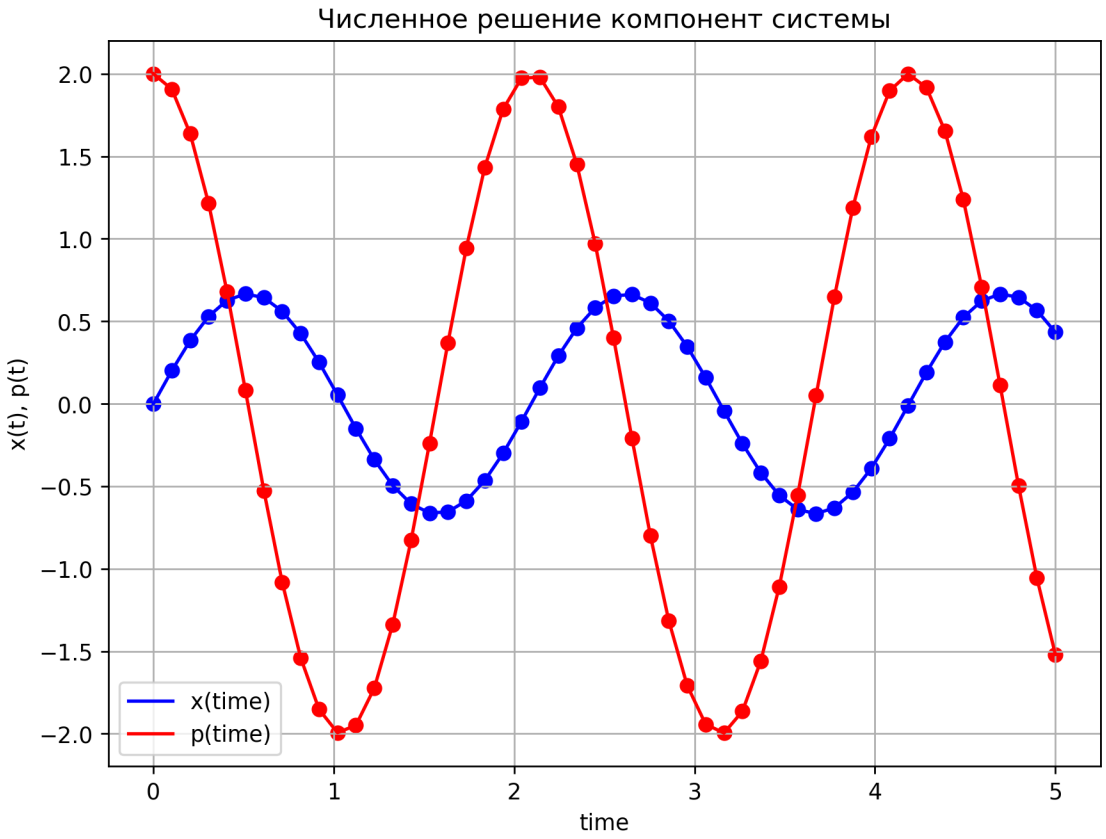
[ ]:
```
import scipy.integrate as integr                    #
import numpy as np                                  #
import matplotlib.pylab as plt                      #       python

colors = ["#00FFFF",                    # Azure
          "#0000FF",                    # Blue
          "#FF0000",                    # Red
          "#00FF00",                    # Green
          "#D2691E",                    #
          "#AAA662",
          "#9A0EEA",
          "#808000",
          "#FFA500",
          "#9ACD32",
          "#ED0DD9",
          "#FC5A50",
          "#929591",
          "#029386",
          "#C79FEF",
          "#FAC205",
          "#6E750E",
          "#06C2AC"]
```

```
#
def abline(slope, intercept):
    axes = plt.gca()
    x_vals = np.array(axes.get_xlim())
    y_vals = intercept + slope * x_vals
    plt.plot(x_vals, y_vals, c = "black")
```

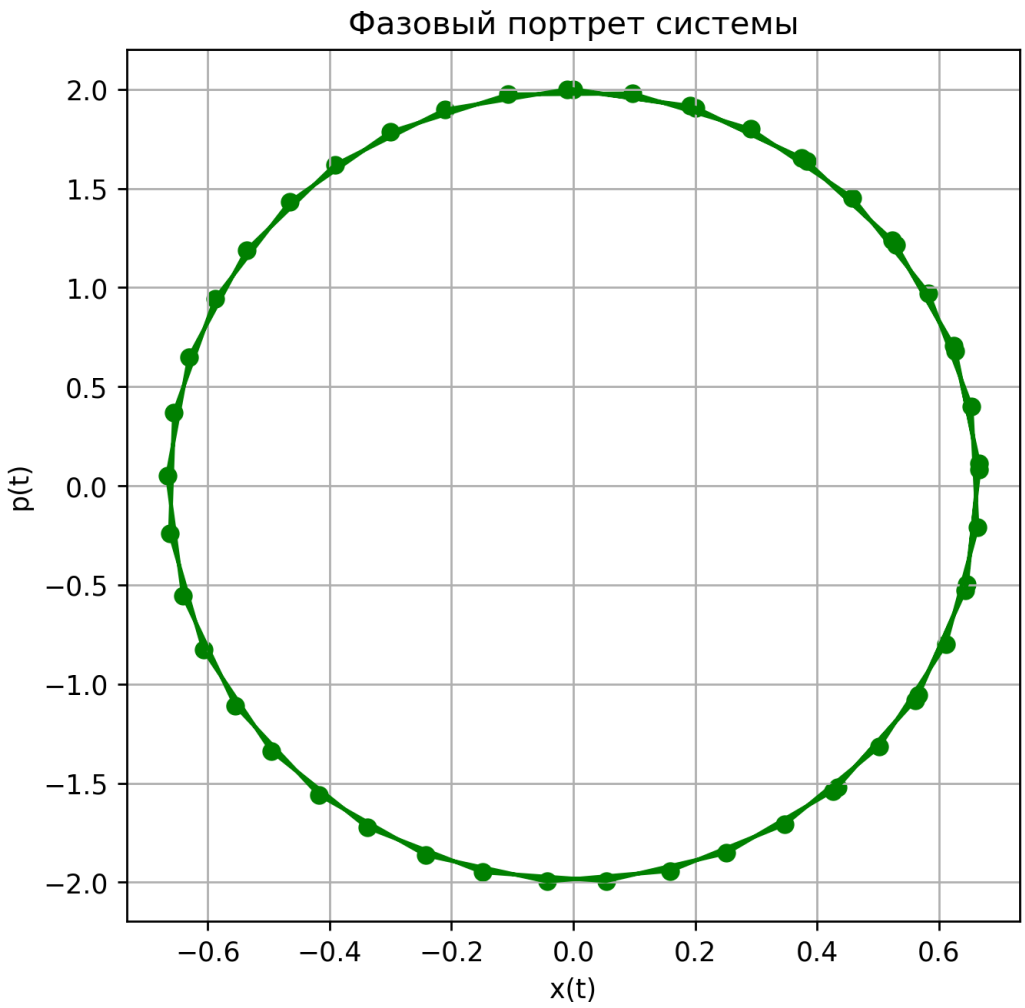
[ ]:
```
#
def f(y, t, omega = 1):
    y1, y2 = y
    return [y2, -1 * omega**2 * y1]

time = np.linspace(0, 5, 50)        #
y0 = [0, 2]                         #

#
w = integr.odeint(f,            #
                  y0,           #
                  time,         #
                  (3, )         #               omega
                  )

y1 = w[:, 0]            #     x
y2 = w[:, 1]            #     p
```

Python matplotlib.pyplot

[ ]:
```
#
fig = plt.figure(facecolor="white", figsize = (8, 6), dpi=250)

#
plt.plot(time, y1, c = "b", label = "x(time)")     #          (t, x(t))
plt.scatter(time, y1, c = "b")                      #          (t, x(t))

#
plt.plot(time, y2, c = "r", label = "p(time)")     #          (t, p(t))
plt.scatter(time, y2, c = "r")                      #          (t, p(t))

#
plt.xlabel("time")          #
plt.ylabel("x(t), p(t)")    #
plt.title("                   ")
plt.legend()

plt.grid(True)              #
plt.show()                  #
```



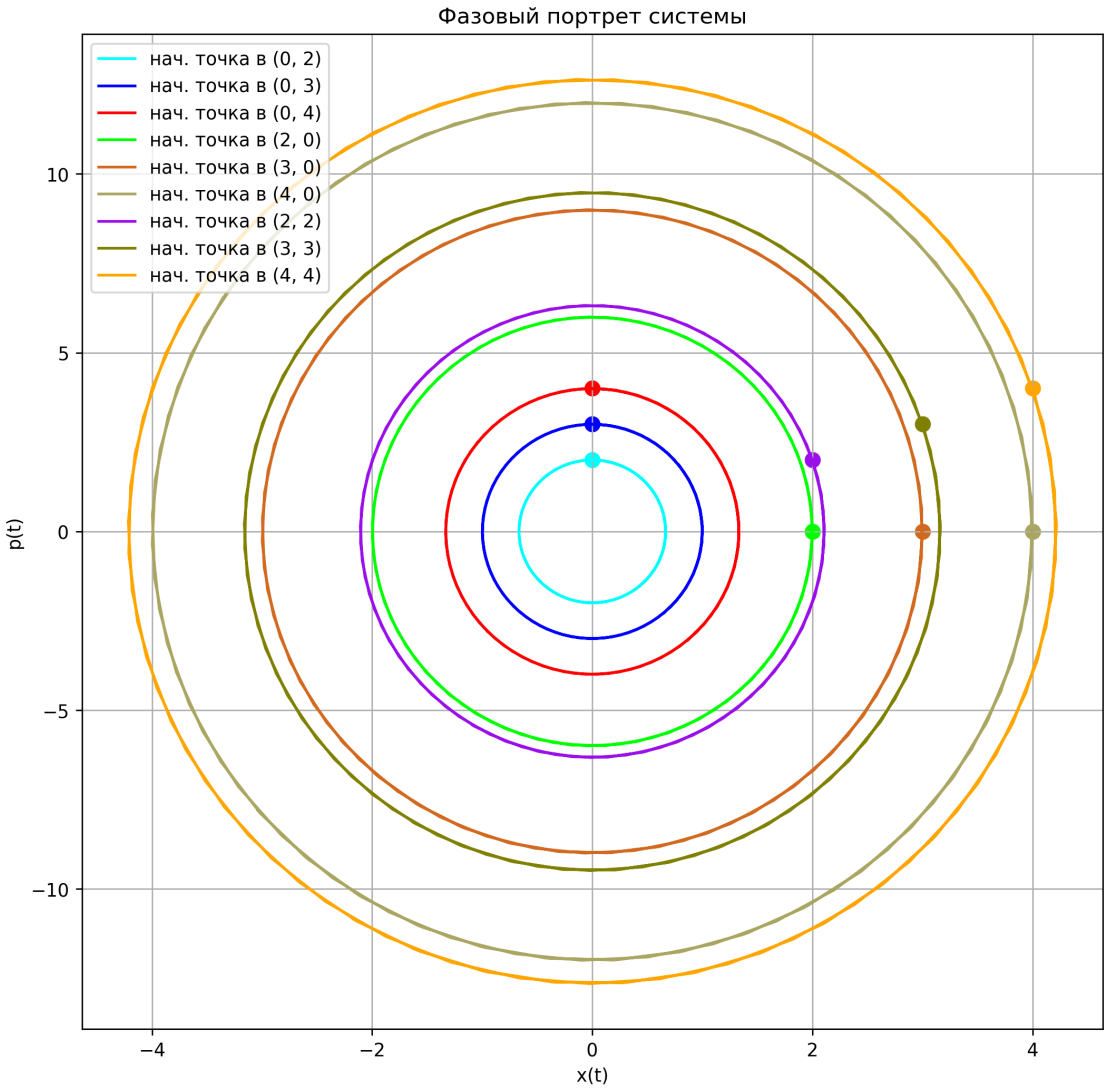
[ ]:
```
fig = plt.figure(facecolor="white", figsize=(6,6), dpi=250)

#
plt.plot(y1, y2, c = "g")
plt.scatter(y1, y2, c = "g")

#
plt.xlabel("x(t)")
plt.ylabel("p(t)")
plt.title("              ")

plt.grid(True)
plt.show()
```



[ ]:
```
time = np.linspace(0, 5, 100)         #
y0_init = [[0, 2], [0, 3], [0, 4],
           [2, 0], [3, 0], [4, 0],
           [2, 2], [3, 3], [4, 4]]    #

W_global = []
#
for y0 in y0_init:
    W_global.append(
        integr.odeint(f,              #
                      y0,             #
                      time,           #
                      (3, )           #                omega
                      )
```

```
    )

y1 = W_global[0][:, 0]            #     x
y2 = W_global[0][:, 1]            #     p
```

[ ]:
```
fig = plt.figure(facecolor="white", figsize=(10, 10), dpi=250)

#
for line_number in range(len(y0_init)):
    #                     -
    y1 = W_global[line_number][:, 0]
    y2 = W_global[line_number][:, 1]
    #
    plt.plot(y1, y2, c = colors[line_number],
                    label = ("  .       (" +
                              str(y0_init[line_number][0]) + ", " +
                              str(y0_init[line_number][1]) + ")"))
    #
    plt.scatter(y1[0], y2[0], c = colors[line_number], s = 60)

#
plt.xlabel("x(t)")
plt.ylabel("p(t)")
plt.title("              ")
plt.legend()

plt.grid(True)
plt.show()
```



1.1.2 2.

:

$$\ddot{x} + \omega^2 sin(x) = 0$$

:

$$\begin{cases} \dfrac{dx}{dt} & = p \\ \dfrac{dp}{dt} & = -\omega^2 \cdot sin(x) \end{cases}$$

k = 1:

$$\begin{cases} \dfrac{dx}{dt} & = p \\ \dfrac{dp}{dt} & = \omega^2 \cdot x \end{cases}$$

:

$$A = \begin{pmatrix} 0 & 1 \\ \omega^2 & 0 \end{pmatrix}$$

:

$$\lambda^2 - \omega^2 * 1 = 0$$

$$\lambda_{1,2} = \pm\omega \in R$$

$$\lambda_1 > 0,\ \lambda_2 < 0$$

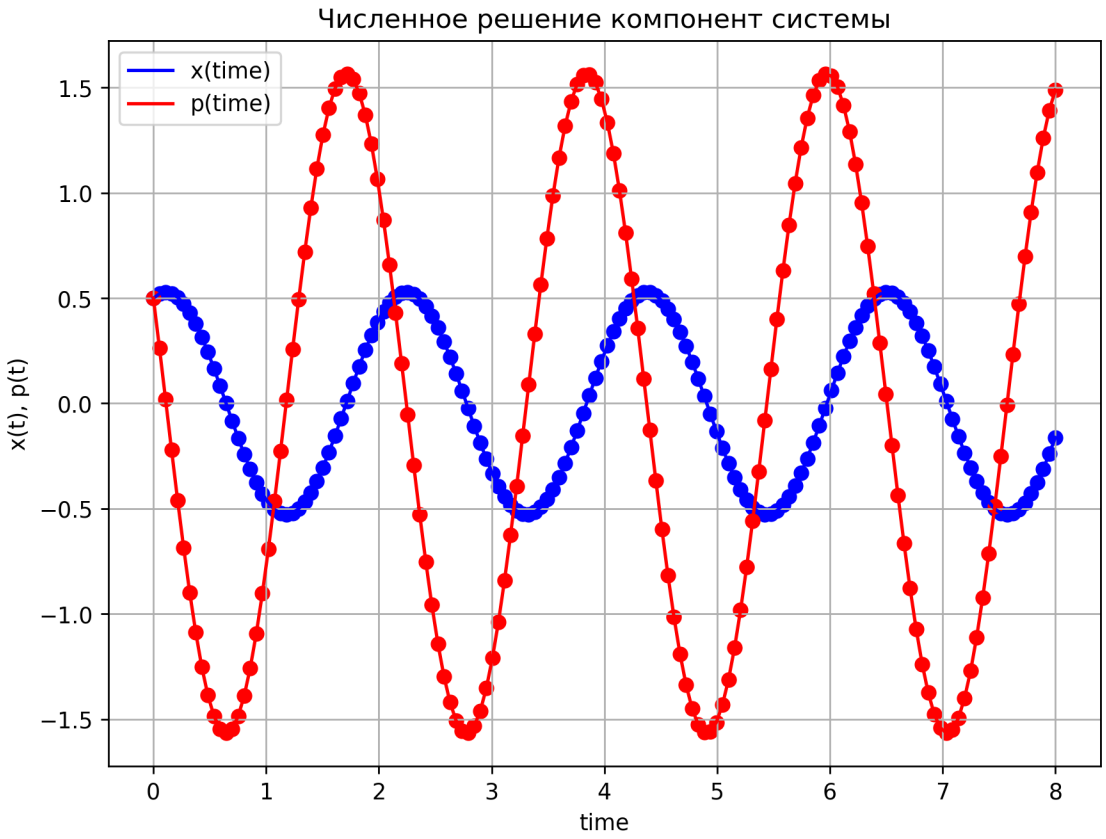
:

$$(a - \lambda_{1,2}) \cdot x + b \cdot y = 0$$
$$c \cdot x + (d - \lambda_{1,2}) \cdot y = 0$$

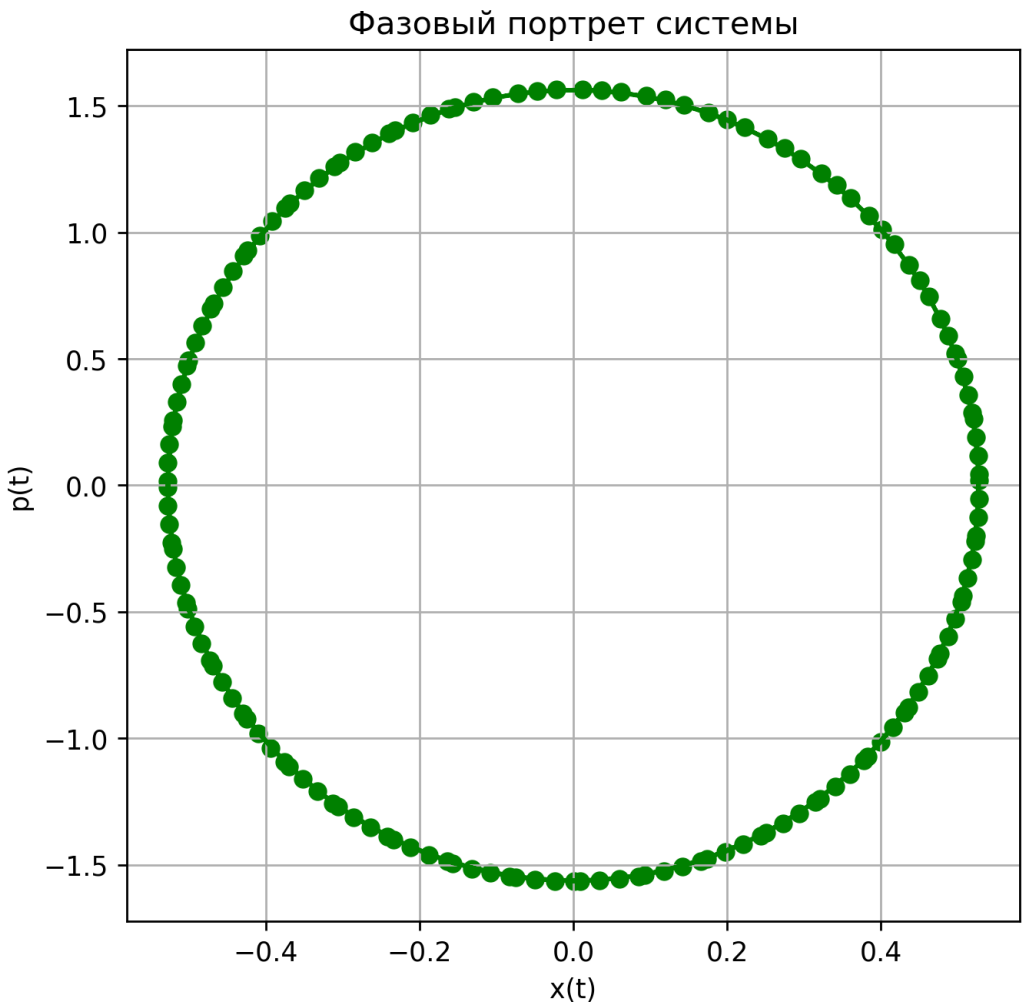
[ ]:
```
#
def f(y, t, omega = 1):
    y1, y2 = y
    return [y2, -1 * omega**2 * np.sin(y1)]

time = np.linspace(0, 8, 150)        #
omega0 = 3
#
#
#
y0_init = [[0.5, 0.5],  #              (0, 0)
           [1, 1],
           [1.5, 1.5],
           [2, 2],
           [2.3, 2.3],
           [0.5 + 2 * np.pi, 0.5],
           [1 + 2 * np.pi, 1],
           [1.5 + 2 * np.pi, 1.5],
           [2 + 2 * np.pi, 2],
           [2.3 + 2 * np.pi, 2.3],
           [0.5 - 2 * np.pi, 0.5],
```

```
            [1 - 2 * np.pi, 1],
            [1.5 - 2 * np.pi, 1.5],
            [2 - 2 * np.pi, 2],
            [2.3 - 2 * np.pi, 2.3],
            [2.4 - 4 * np.pi, 2.4],
            [3 - 6 * np.pi, 3],
            [2.4 + 4 * np.pi, -2.4]]

W_global = []
#
for y0 in y0_init:
    W_global.append(
        integr.odeint(f,            #
                      y0,           #
                      time,         #
                      (omega0, )          #               omega
                      )
    )

y1 = W_global[0][:, 0]            #     x
y2 = W_global[0][:, 1]            #     p
```

[ ]:
```
#
fig = plt.figure(facecolor="white", figsize = (8, 6), dpi=250)

#
plt.plot(time, y1, c = "b", label = "x(time)")     #           (t, x(t))
plt.scatter(time, y1, c = "b")                      #           (t, x(t))

#
plt.plot(time, y2, c = "r", label = "p(time)")     #           (t, p(t))
plt.scatter(time, y2, c = "r")                      #           (t, p(t))

#
plt.xlabel("time")          #
plt.ylabel("x(t), p(t)")    #
plt.title("                    ")
plt.legend()

plt.grid(True)              #
plt.show()                  #
```



[ ]:
```
fig = plt.figure(facecolor="white", figsize=(6,6), dpi=250)

#
plt.plot(y1, y2, c = "g")
plt.scatter(y1, y2, c = "g")

#
plt.xlabel("x(t)")
plt.ylabel("p(t)")
plt.title("              ")

plt.grid(True)
plt.show()
```



[ ]:
```
fig = plt.figure(facecolor="white", figsize=(15, 8), dpi=250)

#
for line_number in range(len(y0_init)):
    #                      -
    y1 = W_global[line_number][:, 0]
    y2 = W_global[line_number][:, 1]
    #
    plt.plot(y1, y2, c = colors[line_number],
                    label = ("  .       (" +
                             str(round(y0_init[line_number][0], 2)) + ", " +
                             str(round(y0_init[line_number][1], 2)) + ")"))
    #
    plt.scatter(y1[0], y2[0], c = colors[line_number], s = 50)
```

```
intercepts = [-5 * np.pi, -3 * np.pi, -np.pi, np.pi, 3 * np.pi, 5 * np.pi]
for interc in intercepts:
    abline(omega0, omega0 * interc)
    abline(-omega0, -omega0 * interc)

#
plt.xlabel("x(t)")
plt.ylabel("p(t)")
plt.title("              ")
plt.legend()
plt.xlim((-17, 17))
plt.ylim((-10, 10))

plt.grid(True)
plt.show()
```