Поповкин Артемий Андреевич
Б9122-02.03.01 СЦТ

Постановка задачи

Требуется для **модели Лоренца** исследовать поведение трёх связанных переменных, описывающих упрощённую модель атмосферной конвекции, и определить их состояние в будущем на основе начальных условий и параметров системы.

Описание модели

Описание

Модель Лоренца — это система трёх нелинейных обыкновенных дифференциальных уравнений, известная своим хаотическим поведением при определённых значениях параметров. Она является упрощённой моделью для изучения конвекционных потоков в атмосфере.

Обозначения

- t — время

Входные данные

- x_0 — начальное значение первой переменной
- y_0 — начальное значение второй переменной
- z_0 — начальное значение третьей переменной

Фазовые переменные

- $x\left(t\right)$ — скорость вращения конвекционных валов (или пропорциональна ей) в момент времени t.
- $y\left(t\right)$ — разность температур между восходящим и нисходящим потоками в момент времени t.
- $z\left(t\right)$ — отклонение вертикального температурного профиля от линейного в момент времени t.

Параметры

- σ — число Прандтля (отношение вязкости к температуропроводности)
- ρ — число Рэлея, нормированное на его критическое значение

- β — безразмерный параметр, связанный с геометрией области конвекции

Уравнение

$$\begin{cases} \dot{x} = \sigma\left(y - x\right) \\ \dot{y} = x\left(\rho - z\right) - y \\ \dot{z} = xy - \beta z \end{cases}$$

Решением уравнения будет называть функцию, удовлетворяющую следующей задаче Коши:

$$\begin{cases} \dot{x} = \sigma\left(y - x\right) \\ \dot{y} = x\left(\rho - z\right) - y \\ \dot{z} = xy - bz \\ x\left(0\right) = x_0 \quad y\left(0\right) = y_0 \quad z\left(0\right) = z_0 \end{cases}$$

Аналитическое исследование системы

Условие устойчивости

Для начала найдём равновесия модели

$$\begin{cases} 0 = \sigma\left(y - x\right) \\ 0 = \backslash\mathrm{rhoz} - y - xz \\ 0 = xy - \beta z \end{cases}$$

Решив систему, получаем следующие равновесия:

Виды равновесий

Тривиальное

$x_t = y_t = 0$

Нетривиальные

$$\begin{cases} x_{1,2} = y_{1,2} = \pm\sqrt{\beta\left(\rho - 1\right)} \\ z_{1,2} = \rho - 1 \end{cases}$$

Исследование на устойчивость

Для проверки устойчивости равновесий найдем собственные значения матрицы Якоби системы в данных точках.

$$J - \lambda E = \begin{pmatrix} -\sigma - \lambda & \sigma & 0 \\ \rho - z & -1 - \lambda & -x \\ y & x & -\beta - \lambda \end{pmatrix}$$

получаем:

$$\lambda^3 + (1+\beta+\sigma)\lambda^2 + \left(\sigma + \sigma\beta + \beta - \sigma\rho + \sigma z + x^2\right)\lambda + \sigma\left(\beta + z\beta + x^2 - \rho\beta\right) = 0$$

Воспользуемся критерием Гурвица:

$$H = \begin{pmatrix} 1+\beta+\sigma & \left(\beta + z\beta + x^2 - \rho\beta\right)\sigma & 0 \\ 1 & (1+\beta-\rho+z)\sigma + \beta + x^2 & 0 \\ 0 & 1+\beta+\sigma & \left(\beta + z\beta + x^2 - \rho\beta\right)\sigma \end{pmatrix}$$

$$\Delta_1 = 1 + \beta + \sigma > 0$$

$$\Delta_2 = \left(\sigma + \sigma\beta + \beta - \sigma\rho + \sigma z + x^2\right)(1+\beta+\sigma) - \left(\beta + z\beta + x^2 - \rho\beta\right)\sigma$$

$$\Delta_3 = \left(\beta + z\beta + x^2 - \rho\beta\right)\sigma\Delta_2$$

Тривиальное равновесие

$$\Delta_2 = (1+\sigma)\left(-\sigma(r-1) + \beta(\sigma+1)\right) + \beta^2(\sigma+1)$$

$$\Delta_2 > 0 \iff r < 1 + \frac{(1+\sigma+\beta)\beta}{\sigma}$$

$$\Delta_3 > 0 \iff r < 1$$

Имеем следующие случаи:

- $r < 1$
  - Устойчивый узел
- $r > 1$
  - Седло

Нетривиальные равновесия

x_1, y_1 вместе с x_2, y_2

$$\Delta_2 = (1+\beta+\sigma)(\sigma+\rho)\beta - (\rho-1)\beta\sigma = \beta\left(\sigma(2+\beta+\sigma) + (1+\beta)\rho\right)$$

$$\Delta_2 > 0 \iff r > -\frac{\sigma(2+\beta+\sigma)}{1+\beta}$$

$$\Delta_3 > 0 \iff r > 1$$

Имеем следующие случаи:

- $r > 1$
  - Устойчивый узел
- $r < 1$
  - Седло

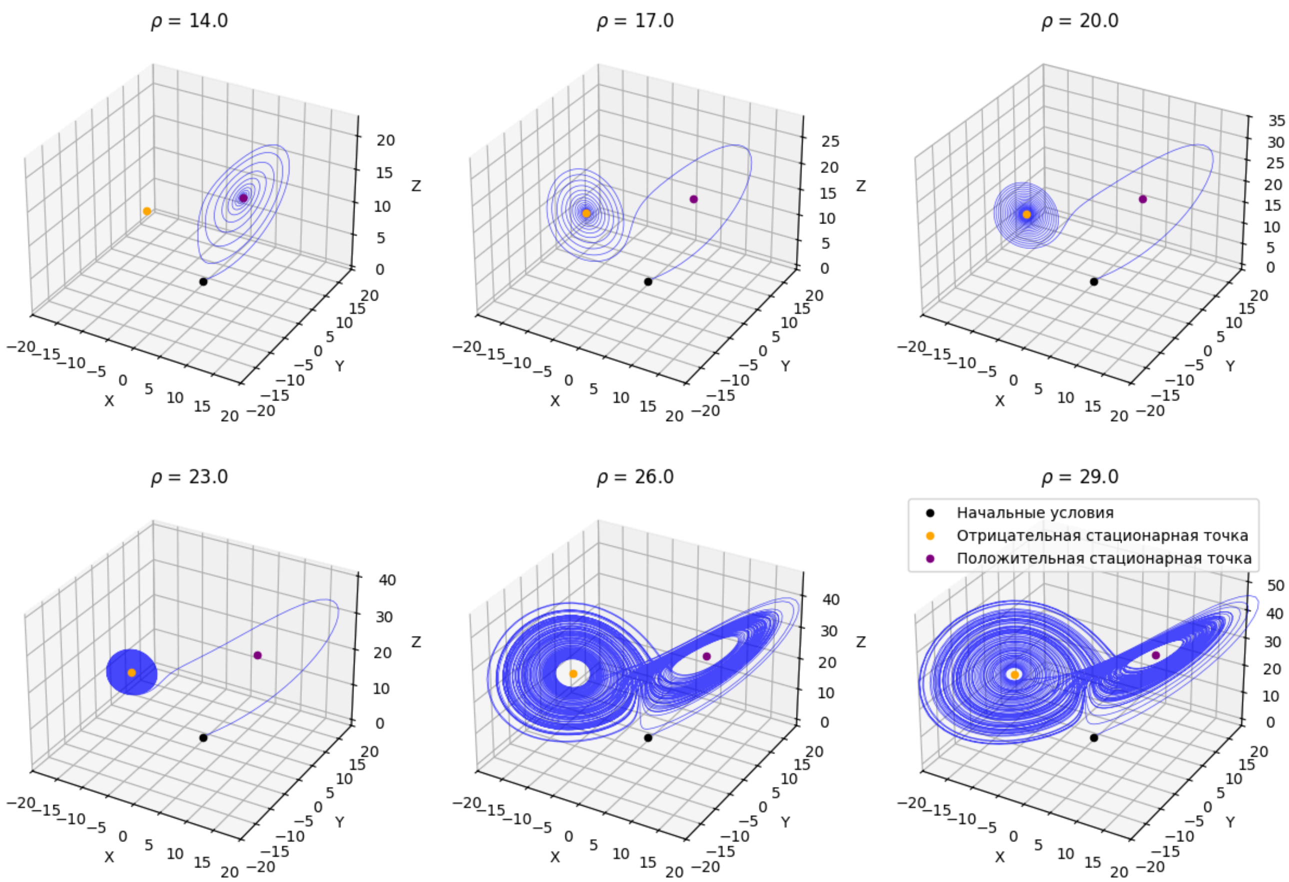
Численные эксперименты

В силу того, что данная модель не описывает практически ни одну реальную систему, поставим следующую задачу: Продемонстрировать хаотический характер системы Лоренца.

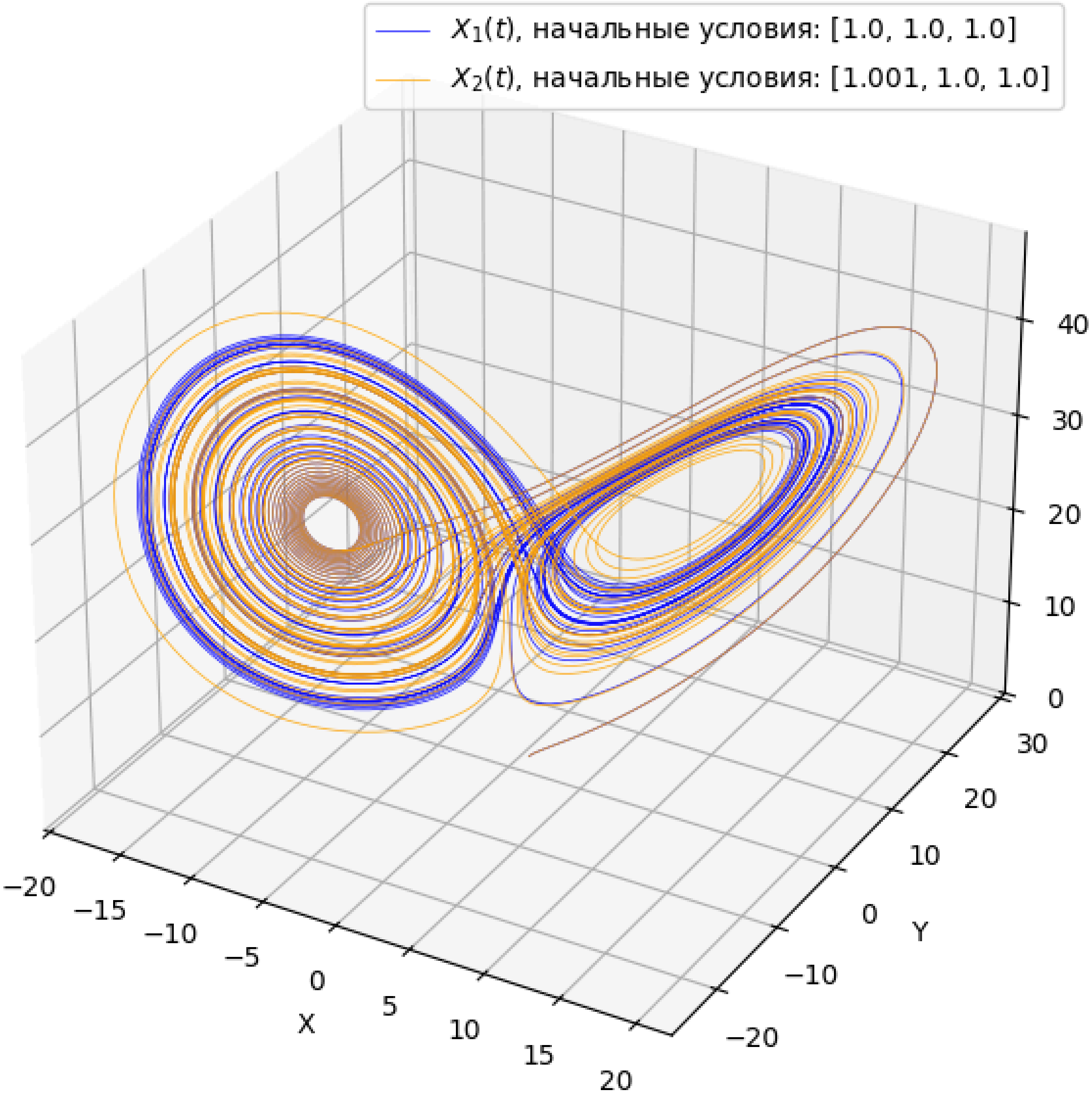
Первым шагом изучим влияние параметра ρ на сценарий развития системы.
Были зафиксированы следующие параметры:

- $\sigma = 10$
- $\beta = \frac{8}{3}$
- $x_0 = y_0 = z_0 = 1$
- $r \in [14, \ldots, 29]$ с шагом $= 3$

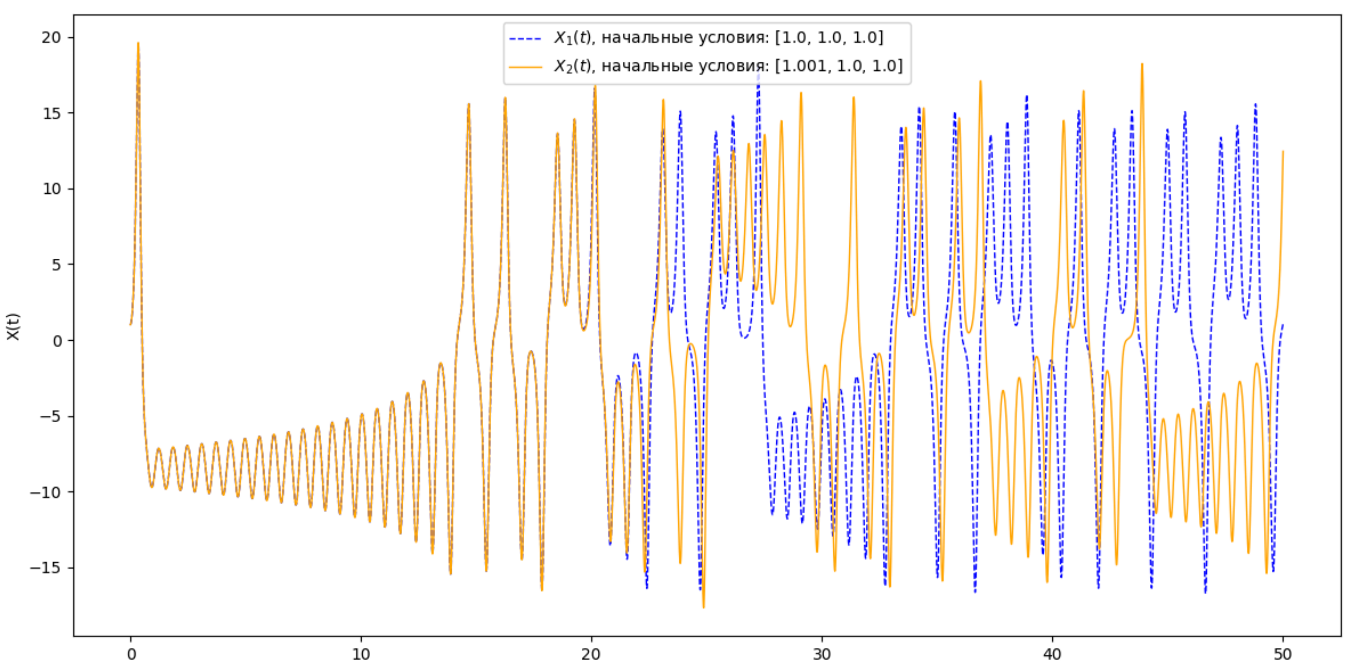
  Построены следующие фазовые траектории модели:



Как мы можем видеть, при достаточно малом ρ, притягивающим оказывается конкретная стационарная точка.
В свою очередь, при достаточно большом его значении, фазовая траектория становится столь чувствительна, что даже малые погрешности численного метода приводят к смене моделью орбиты движения - начинает наблюдаться детерминированный хаос.
Дабы показать хаотичность данной модели, были построены две траектории при одинаковых значениях параметров, но немного отличающихся начальных условиях:



$X_1(t)$, начальные условия: [1.0, 1.0, 1.0]
$X_2(t)$, начальные условия: [1.001, 1.0, 1.0]
X
Y



$X_1(t)$, начальные условия: [1.0, 1.0, 1.0]
$X_2(t)$, начальные условия: [1.001, 1.0, 1.0]
X(t)

Заключение

В результате проведенного исследования была изучена модель Лоренца - классическая система, демонстрирующая хаотическое поведение. Аналитическое исследование системы позволило идентифицировать точки равновесия и проанализировать их устойчивость в зависимости от параметра ρ.
Практическим итогом работы стала численная реализация модели и визуализация её динамики.
С помощью метода Рунге-Кутты 4-го порядка был продемонстрирован возникающий в модели эффект бабочки.
Были получены фазовые траектории системы при различных значениях параметра ρ, что наглядно продемонстрировало переход от регулярного поведения к хаотическому с ростом этого параметра.

Листинг кода

```
# Imports
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
from mpl_toolkits.mplot3d import Axes3D

# Funcs
def stable(r, b):
    sq = np.sqrt(b * (r - 1))
    return ((-sq, -sq, r - 1), (sq, sq, r - 1))


def lorenz_system(t, state, sigma, r, b):
    x, y, z = state
    dx_dt = sigma * (y - x)
    dy_dt = x * (r - z) - y
    dz_dt = x * y - b * z
    return [dx_dt, dy_dt, dz_dt]
# Params
t_span = (0, 100)
t_eval = np.linspace(0, 100, 10000)
# Solution
## R sensitivity
### Params
sigma = 10.0
b = 8.0 / 3.0
r_arange = np.arange(14.0, 14 + 3 * 6, 3)
initial_state = [1.0, 1.0, 1.0]
r_arange
### Testing
```

```
fig = plt.figure(figsize=(15, 10))

for idx, r in enumerate(r_arange):
    solution = solve_ivp(
        lorenz_system,
        t_span,
        initial_state,
        args=(sigma, r, b),
        t_eval=t_eval,
        method="RK45",
    )
    x, y, z = solution.y

    ax = fig.add_subplot(2, 3, idx + 1, projection="3d")
    ax.scatter(*initial_state, color="black", label="Начальные условия")
    u1, u2 = stable(r, b)
    ax.scatter(*u1, color="orange", label="Отрицательная стационарная точка")
    ax.scatter(*u2, color="purple", label="Положительная стационарная точка")
    ax.plot(x, y, z, lw=0.5, color="blue", alpha=0.7)
    ax.set_xlabel("X")
    ax.set_ylabel("Y")
    ax.set_zlabel("Z")
    ax.set_title(f"$\\rho$ = {r}")
    plt.xlim(-20, 20)
    plt.ylim(-20, 20)
plt.show()
## Initial state sensitivity
initial_state1 = [1.0, 1.0, 1.0]
initial_state2 = [1.001, 1.0, 1.0]
r = 28.0
### Calculation

t_eval = np.linspace(0, 50, 10000)
sol1 = solve_ivp(
    lorenz_system,
    t_span,
    initial_state1,
    args=(sigma, r, b),
    t_eval=t_eval,
    method="RK45",
)
sol2 = solve_ivp(
    lorenz_system,
    t_span,
    initial_state2,
    args=(sigma, r, b),
```

```
    t_eval=t_eval,
    method="RK45",
)
x1, y1, z1 = sol1.y
x2, y2, z2 = sol2.y
### Plotting
fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(111, projection="3d")

ax.plot(
    x1,
    y1,
    z1,
    lw=0.5,
    color="blue",
    label=f"$X_1(t)$, начальные условия: {initial_state1}",
    alpha=0.9,
)
ax.plot(
    x2,
    y2,
    z2,
    lw=0.5,
    color="orange",
    label=f"$X_2(t)$, начальные условия: {initial_state2}",
    alpha=0.9,
)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
plt.legend()
plt.tight_layout()
plt.show()

plt.figure(figsize=(12, 6))
plt.plot(
    sol1.t,
    x1,
    "b--",
    label=f"$X_1(t)$, начальные условия: {initial_state1}",
    linewidth=1,
)
plt.plot(
    sol2.t,
    x2,
    "orange",
```

```python
    label=f"$X_2(t)$, начальные условия: {initial_state2}",
    linewidth=1,
)
plt.ylabel("X(t)")
plt.legend()
plt.tight_layout()
```