计算物理第四次作业

第一题：计算矩阵本征值

题目描述

利用 QR 算法、Jacobi 算法、Sturm 序列 + 对分法，求下列矩阵的本征值：

$$T = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix}$$

解答

1. QR 算法

QR 算法通过反复进行 QR 分解 $A_k = Q_k R_k$ 并更新 $A_{k+1} = R_k Q_k$ 来使矩阵收敛到上三角矩阵（对于对称矩阵为对角矩阵）。

核心代码如下：

```python
def qr_algorithm(A, max_iter=50, tol=1e-6):
    n = A.shape[0]
    Ak = A.copy()
    for k in range(1, max_iter + 1):
        Q, R = np.linalg.qr(Ak)
        Ak = R @ Q
        # Check convergence...
    return np.diag(Ak)
```

2. Jacobi 算法

Jacobi 算法通过一系列相似变换（旋转）将对称矩阵对角化。每次选择模最大的非对角元素进行消除。

核心代码如下：

```python
def jacobi_algorithm(A, max_iter=1000, tol=1e-8):
    # ...
    # Calculate rotation angle
    if abs(Ak[p, p] - Ak[q, q]) < 1e-10:
        theta = np.pi / 4
    else:
        theta = 0.5 * np.arctan(2 * Ak[p, q] / (Ak[p, p] - Ak[q, q]))
    # Construct rotation matrix J and update Ak = J.T @ Ak @ J
    # ...
```

3. Sturm 序列 + 对分法

利用 Sturm 序列的性质计算给定区间内本征值的个数，结合对分法精确定位本征值。

核心代码如下：

```
def sturm_sequence_count(d, e, lam):
    # ...
    for k in range(2, n + 1):
        val = (d[k-1] - lam) * seq[-1] - (e[k-2]**2) * seq[-2]
        seq.append(val)
    # Count sign changes
    # ...

def sturm_bisection(A, tol=1e-6):
    # ...
    # Gerschgorin bounds for search interval
    # Bisection search for each eigenvalue
    # ...
```

结果

三种方法计算得到的本征值高度一致：

- **QR 算法**: `[4.74528124, 3.17728292, 1.82271708, 0.25471876]`
- **Jacobi 算法**: `[0.25471876, 3.17728292, 1.82271708, 4.74528124]`
- **Sturm 序列**: `[0.25471866, 1.82271731, 3.17728293, 4.74528158]`

第二题：幂次法求矩阵最大模本征值

题目描述

利用幂次法求解一维原子链振动问题的最大本征频率平方 $\omega^2_{\max}$。

解答

1. 矩阵构建

根据周期性边界条件，矩阵 A 的形式为：

```
def construct_matrix(N):
    A = np.zeros((N, N))
    for i in range(N):
        A[i, i] = 2.0
        A[i, (i + 1) % N] = -1.0
        A[i, (i - 1) % N] = -1.0
    return A
```

2. 幂次法实现

迭代公式：$z^{(k)} = Aq^{(k-1)}, q^{(k)} = z^{(k)}/\|z^{(k)}\|$。

```python
def power_method(A, max_iter=2000, tol=1e-10):
    # ...
    for k in range(1, max_iter + 1):
        z = A @ q
        norm_z = np.linalg.norm(z)
        q_next = z / norm_z
        lambda_next = np.vdot(q_next, A @ q_next).real
        # ...
```

结果

对于 $N = 10$ 的情形：

- **最大本征值 ($\omega^2_{\max}$)**: `4.00000000`
- **理论值**: $4\sin^2(N\pi/2N) = 4$。
- **本征矢**: 对应于相邻原子反向振动的模式。

第三题：孤立子数值解

题目描述

求解 KdV 方程 $u_t + uu_x + \delta^2 u_{xxx} = 0$，展示孤立子现象。

解答

采用 **Zabusky-Kruskal 格式**（Leapfrog 时间步进 + 中心差分空间离散）：

$$u_j^{n+1} = u_j^{n-1} - 2\Delta t \left[\frac{u_{j+1}^n + u_j^n + u_{j-1}^n}{3} \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} + \delta^2 \frac{u_{j+2}^n - 2u_{j+1}^n + 2u_{j-1}^n - u_{j-2}^n}{2(\Delta x)^3} \right]$$

核心代码：

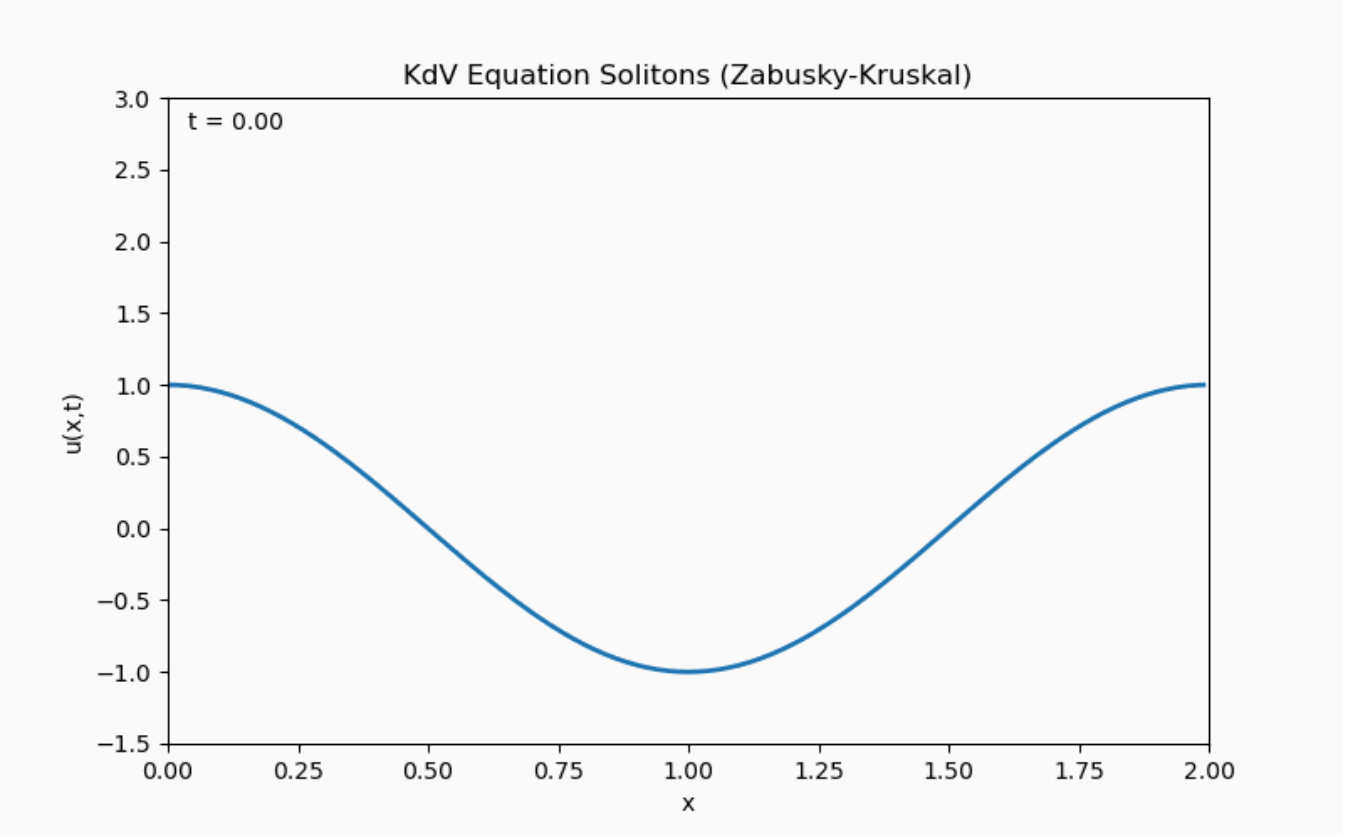
```python
def compute_rhs(u):
    u_jp1 = np.roll(u, -1)
    u_jm1 = np.roll(u, 1)
    u_jp2 = np.roll(u, -2)
    u_jm2 = np.roll(u, 2)

    nonlinear = (u_jp1 + u + u_jm1) / 3.0 * (u_jp1 - u_jm1) / (2 * dx)
    dispersion = delta_sq * (u_jp2 - 2*u_jp1 + 2*u_jm1 - u_jm2) / (2 * dx**3)

    return -(nonlinear + dispersion)
```

结果

初始的余弦波分裂成一列孤立子，振幅大的速度快，振幅小的速度慢。



第四题：二维波动方程

题目描述

求解二维波动方程，对比解析解与数值解，并分析稳定性。

解答

(a) 解析解

分离变量法得到：
$u(x,y,t) = \sin(\pi x)\sin(2\pi y)\cos(\sqrt{5}\pi t)$

(b) 数值解

使用显式差分格式：

```
# u^{n+1} = 2u^n - u^{n-1} + rx^2 * u_xx + ry^2 * u_yy
u_np1[1:-1, 1:-1] = 2*u_n[1:-1, 1:-1] - u_nm1[1:-1, 1:-1] + \
                    rx2 * u_n_xx + ry2 * u_n_yy
```

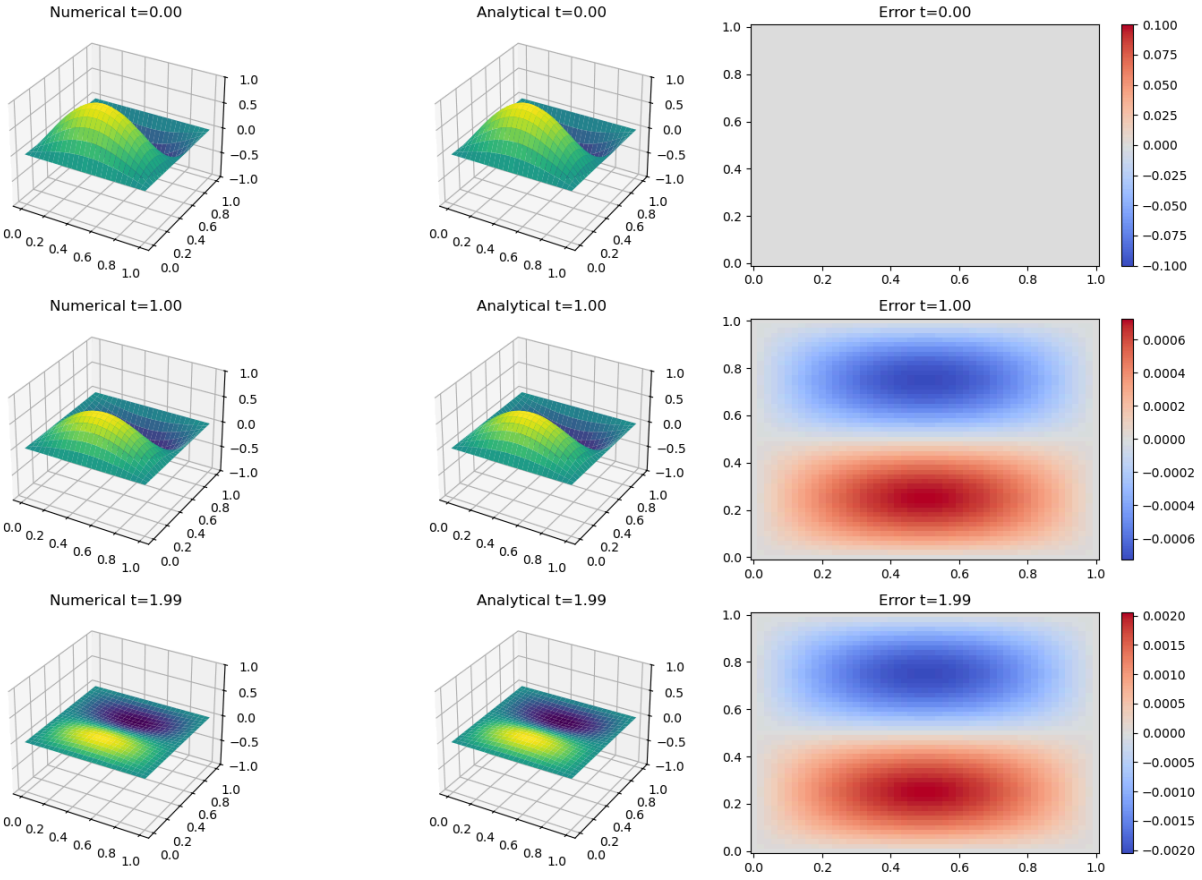
(c) 稳定性分析

稳定性条件：$\Delta t \leq \frac{1}{\sqrt{\lambda}}\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}\right)^{-1/2}$。

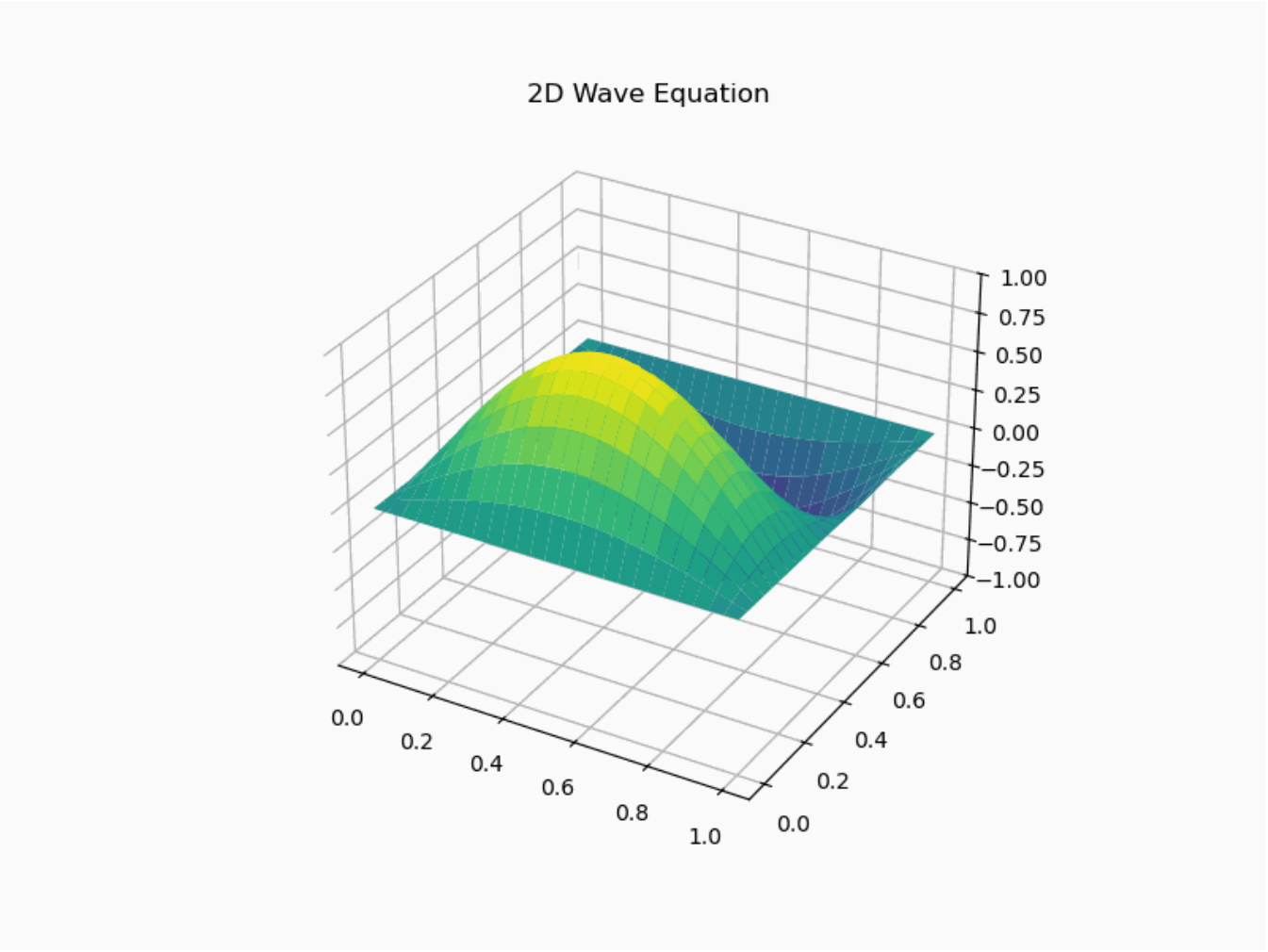
- 当 $\lambda = 2.0$ (满足 CFL) 时，模拟稳定。
- 当 $\lambda = 0.8$ (不满足 CFL) 时，模拟发散。

结果

数值解与解析解对比 ($t = 0, 1, 2$):



动画展示:



第五题：随机数产生器

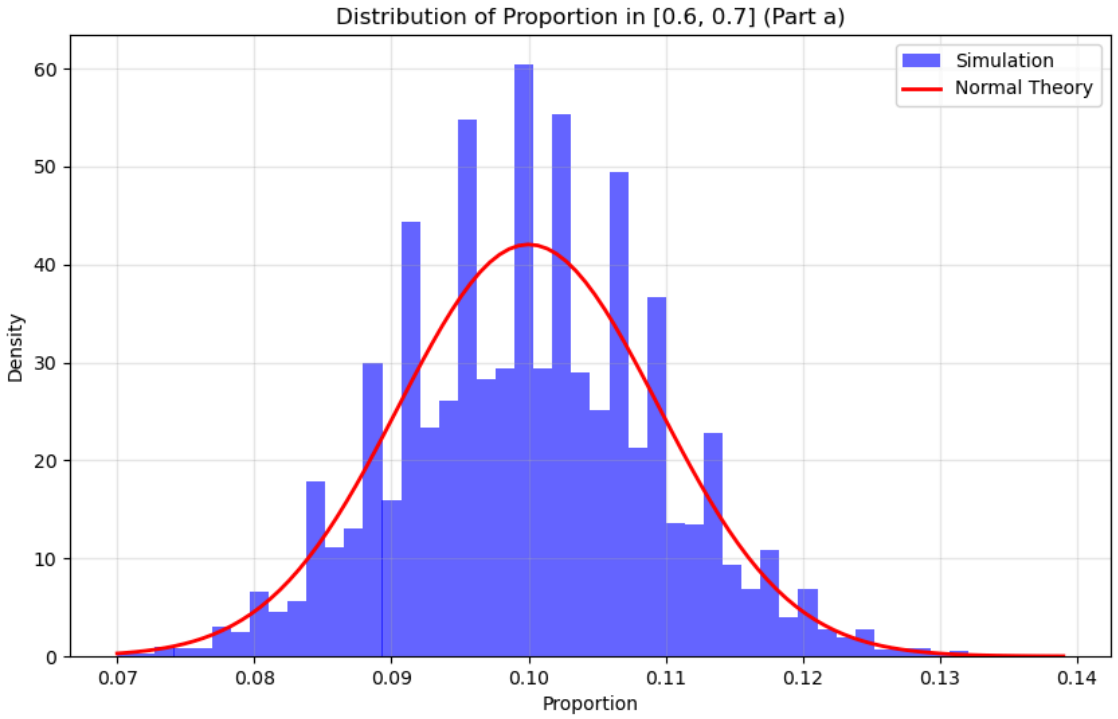
题目描述

验证随机数分布性质，并实现 16807 产生器。

解答

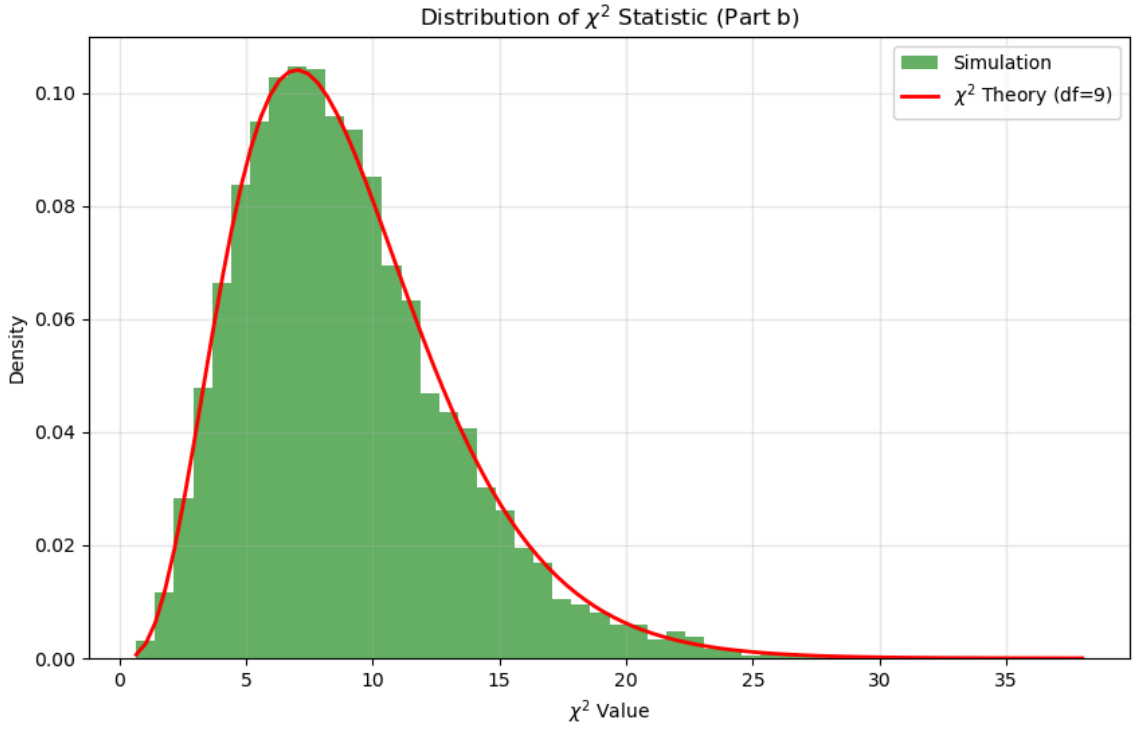
(a) 分布检验

统计落入 $[0.6, 0.7]$ 区间的比例，结果符合正态分布。



(b) χ^2 检验

计算 $\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$，结果符合自由度为 9 的 χ^2 分布。



(c) 16807 产生器

实现线性同余发生器：

```python
class LCG16807:
    def __init__(self, seed=1):
        self.state = seed
        self.a = 16807
        self.m = 2147483647

    def next(self):
        self.state = (self.a * self.state) % self.m
        return self.state
```

验证结果：$x(10000) = 1043618065$，验证成功。