Project 2

In case of any question, contact the below TAs:
Ehsan Forootan
Kiana Hooshanfar
Mehrab Pasha

Instructions: You may use Python, Matlab or Julia to solve these questions. In case you are using notebook formats (Such as Jupiter or CoLab), write your explanations in the same notebook as well. If not, you need to turn in an additional report. Make sure the results are visualized.

Question 1.

Eigenvalue Stability Analysis in Dynamical Systems

Consider a discrete-time linear dynamical system defined by:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k$$

where $A \in \mathbb{R}^{n\times n}$. Use the following matrices:

$$A_1 = \begin{bmatrix} 0.9 & 0.3 \\ -0.1 & 0.8 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1.1 & 0.4 \\ -0.2 & 1.0 \end{bmatrix}$$

(a) Generate at least 100 different initial vectors $\mathbf{x}_0$ by sampling from a normal distribution $\mathcal{N}(0, 5)$.

(b) Simulate the trajectories for 100 time steps for each initial vector for both A_1 and A_2. Report the mean ending points for each matrix.

(c) Determine the eigenvalues of each matrix and classify each system as stable, asymptotically stable, or unstable(negative is stable, zero is asymptotically stable,).

(d) Plot the phase portraits for each matrix using the generated initial vectors.

(e) imagine the system

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + \eta$$

where η is a noise vector with size two, where each element of it is taken from a $\mathcal{N}(0, 1)$ and in each time step, is sampled. Repeat part a till d for both matrices and plot the phase portraits.

Question 2.

Condition Number and Sensitivity

Let B be a nearly singular 3×3 matrix:

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4.0001 & 6 \\ 3 & 6 & 9 \end{bmatrix}$$

(a) Generate a random vector $\mathbf{b}$ from a uniform distribution on $U \sim [0, 1]^3$.

(b) Compute the condition number of B in the 2-norm.

(c) Solve the system $B\mathbf{x} = \mathbf{b}$ and compute the relative error.

(d) Perturb $\mathbf{b}$ slightly (e.g., add Gaussian noise $\mathcal{N}(0, \epsilon)$) for various ϵ values. Plot the solution error as a function of ϵ. Report your conclusion.

Question 3.

Signal Transmission and Error Correction in Communication Systems

In digital communication, messages are transmitted as vectors and may be corrupted by noise. Linear algebra plays a central role in ”encoding”, ”decoding”, and ”error detection”. In this problem, you will simulate a simple communication system.

(a) Design a simple linear code using a generator matrix $G \in \mathbb{R}^{k\times n}$, where $k = 3$ and $n = 5$. Use random binary entries (0 or 1) for G.

(b) Generate 10 random binary message vectors $\mathbf{m}_i \in \mathbb{F}_2^k$ and encode them into codewords $\mathbf{c}_i = \mathbf{m}_i G$ (modulo 2).

(c) Simulate a noisy communication channel by flipping each bit of the codeword with a fixed probability $p = 0.1$ (i.e., simulate bit-flip noise using a Bernoulli distribution).

(d) Describe why this noise model is **non-linear**: the bit flips are not a linear transformation of the codeword and cannot be represented by a matrix multiplication.

(e) Construct a parity-check matrix H for your code and compute the syndrome $\mathbf{s} = H\mathbf{r}^T$ (mod 2) for each received vector $\mathbf{r}$.

(f) Identify which received vectors contain detectable errors. Comment on whether the system can correct them or just detect them.

(g) Explain how linear algebra enables efficient encoding and detection in this communication system.

(h) Discuss which parts of your simulation involve non-linear operations (e.g., bit-flip noise, decoding with thresholds) and why these cannot be represented using matrices or linear transformations.

Note:

Non-linear elements in communication systems include noise models, decoding strategies using thresholds or majority logic, and entropy-based compression. These are not linear because they involve probabilistic or conditional behavior, which does not obey the superposition principle. However, linear algebra provides the backbone for encoding, transmission modeling, and initial error detection.

Question 4.

Markov Chains and Steady-State Analysis

Markov Chains are widely used in areas such as search engines, economics, biology, and communication networks. They model systems where the next state depends only on the current state (memoryless property). In this problem, you will use linear algebra to simulate a Markov process, analyze convergence behavior, and compute steady states.

(a) Generate a random stochastic matrix $P \in \mathbb{R}^{n\times n}$, where $n = 6$. Each row must sum to 1. Use the Dirichlet distribution to generate each row.

(b) Verify that P is a valid transition matrix by confirming non-negativity and that each row sums to 1.

(c) Choose an initial state vector $\mathbf{x}_0 \in \mathbb{R}^n$ with non-negative entries that sum to 1.

(d) Simulate the Markov process for 100 time steps: $\mathbf{x}_{k+1} = \mathbf{x}_k P$.

(e) Plot the evolution of the state vector over time. Discuss whether the system appears to converge to a steady state.

(f) Use linear algebra to find the steady-state vector $\mathbf{x}_\infty$ satisfying $\mathbf{x}_\infty P = \mathbf{x}_\infty$, with $\sum x_i = 1$.

(g) Compare this computed steady-state with the final $\mathbf{x}_k$ from your simulation.

(h) Compute the eigenvalues of P. Discuss why convergence occurs based on the dominant eigenvalue and the Perron-Frobenius theorem.

(i) Add a small perturbation δP (e.g., change a few probabilities slightly) and re-normalize the rows to ensure $P + \delta P$ is still stochastic.

(j) Recompute the steady state and compare it to the original. How sensitive is the steady state to small changes in the transition probabilities?

(k) Comment on the implications of this sensitivity for real-world systems such as internet search rankings or weather prediction.