Government Engineering College, Barton Hill
Third Semester B. Tech. Degree Aug-Dec 2020 (ext. to Feb 2021)
ECL 201: Scientific Computing Lab
Instructors: Birenjith P. S., Aparna S. Thampy, Sruthi B. R.

Model Questions

1) Write a recursive function to generate Hadamard matrix

$$H_{2^n} = \begin{bmatrix} H_{2^{n-1}} & H_{2^{n-1}} \\ H_{2^{n-1}} & -H_{2^{n-1}} \end{bmatrix}$$

with $H_1 = [1]$ for $n = 0$. For example

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

2) Let N be an even integer. Write a program to generate an $(N \times N)$ random matrix A with entries in $\{-1, 0, 1\}$ with rank $\rho(A) \leq N/2$. (Check out the function *numpy.random.randint*.)

3) Let A be an $(N \times N)$ random matrix with entries in $\{-1, 0, 1\}$ such that rank of A $\rho(A) = \lceil N/2 \rceil$. Repeat the following simulation for $N = 10, 100, 500$.
   (a) Carry out singular value decomposition of A so that we can write $A = USV^T$:

$$A = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1N} \\ v_{21} & v_{22} & & u_{2N} \\ \vdots & & \ddots & \vdots \\ v_{N1} & u_{N2} & & u_{NN} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_N \end{bmatrix} \cdot \begin{bmatrix} v_{11} & v_{21} & \cdots & v_{N1} \\ v_{12} & v_{22} & & v_{N2} \\ \vdots & & \ddots & \vdots \\ v_{1N} & v_{2N} & & v_{NN} \end{bmatrix}.$$

   (b) Approximate A as

$$\hat{A}(r) = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1r} \\ v_{21} & v_{22} & & u_{2r} \\ \vdots & & \ddots & \vdots \\ v_{N1} & u_{N2} & & u_{Nr} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_r \end{bmatrix} \cdot \begin{bmatrix} v_{11} & v_{21} & \cdots & v_{N1} \\ v_{12} & v_{22} & & v_{N2} \\ \vdots & & \ddots & \vdots \\ v_{1r} & v_{2r} & & v_{Nr} \end{bmatrix}.$$

   for every $1 \leq r \leq N$.
   (c) Calculate Frobenius norm of the error $e[r] = ||(A - \hat{A}(r))||_F^2$ and plot $e[r]$ versus r. Use *scipy.linalg.norm* to compute the Frobenius norm of a matrix.
   (d) Calculate normalized norm of the error $e_{\text{norm}}[r] = ||(A - \hat{A}(r))||_F^2/||A||_F^2$ and plot $e_{\text{norm}}[r]$ versus r. Find the smallest r such that the normalized norm of the error is less than 0.1.

4) Generate M samples of standard normal Gaussian random variables $X_1, X_2, \ldots X_M$. (Check out the function *numpy.random.randn.*) This simulation tries to estimate mean and variance of a normal Gaussian random variable from large number of observed samples. Repeat the following experiment for $M = 5, 10, 100, 500$.
   (a) Calculate $Y_i = 2X_i + 3, \; i = 1, 2, \ldots, N.$

(a) Calculate the estimate values of mean and variances for X and Y as:

$$\begin{aligned}\hat{\mu}_X &= \frac{1}{M}\sum_{i=1}^{M} X_i\\ \hat{v}_{X,1} &= \frac{1}{M}\sum_{i=1}^{M}(X_i-\hat{\mu}_X)^2\\ \hat{v}_{X,2} &= \frac{1}{M-1}\sum_{i=1}^{M}(X_i-\hat{\mu}_X)^2\\ \hat{\mu}_Y &= \frac{1}{M}\sum_{i=1}^{M} Y_i\\ \hat{v}_{Y,1} &= \frac{1}{M}\sum_{i=1}^{M}(Y_i-\hat{\mu}_Y)^2\\ \hat{v}_{Y,2} &= \frac{1}{M-1}\sum_{i=1}^{M}(Y_i-\hat{\mu}_Y)^2\end{aligned}$$

Note that variance has two different estimators, the first that divides by M and the second that divides by $M-1$.

(c) Calculate the squared error between estimated and actual values:

$$\begin{aligned}e_1[M] &= |\hat{\mu}_X - 0|^2\\ e_2[M] &= |\hat{v}_{X,1} - 1|^2\\ e_3[M] &= |\hat{v}_{X,2} - 1|^2\\ e_4[M] &= |\hat{\mu}_Y - 3|^2\\ e_5[M] &= |\hat{v}_{Y,1} - 4|^2\\ e_6[M] &= |\hat{v}_{Y,2} - 4|^2\end{aligned}$$

(d) Plot $e_j[M]$ versus M for $j = 1, 2, \ldots, 6$

5) (a) The toss of a biased coin is represented by a Bernoulli random variable with $\Pr(1) = p = 0.7$ and $\Pr(0) = 0.3$. Write a python program to simulate 100 tosses.

(b) Let M denote the number of tosses. Let $X_1, X_2, \ldots X_M$ denote the outcomes of tosses. Calculate

$$\begin{aligned}\hat{p}(M) &= \frac{1}{M}\sum_{i=1}^{M} X_i\\ e[M] &= |p - \hat{p}(M)|\end{aligned}$$

Compute $e[M]$ for $M = 1, 2, \ldots 100$ and plot it against M.

6) The throw of a biased dice is represnted by a random variable X with probability distribution $\Pr(1) = \Pr(2) = \Pr(3) = \Pr(4) = 0.2$ and $\Pr(5) = \Pr(6) = 0.1$. Write a python program to simulate 100 throws of the dice.

7) A circuit has three loop equations

$$\begin{aligned} 5i_1 + 2i_2 + i_3 &= 1 \\ 2i_1 + 6i_2 - i_3 &= 0 \\ i_1 - i_2 + 3i_3 &= 0 \end{aligned}$$

The voltage across a 2Ω resistor is given by $V = 2(i_1 + i_2)$. Write a python program to compute V.

8) A $1V$ battery is connected to an RC circuit with $R = 2\Omega$ and $C = 1F$. At $t = 0$, the switch is turned on and the initial current $i(0+) = 1A$. Represent the time-evolution of current $i(t)$ using a differential equation and solve it using a python program.
   (a) Plot $i(t)$ and the voltage $v_C(t) = 1 - Ri(t)$ across the capacitor for $t = 0$ to 5 seconds.
   (b) Find the time t_0 at which $v_c(t_0) = 0.632V$. Verify that $t_0 = RC$

9) A $1V$ battery is connected to an RLC circuit with $R = 3\Omega$ and $L = 1mH$ and $C = 0.5\mu F$. At $t = 0$, the switch is turned on and the initial values are $i(0+) = 0A$ and $\frac{di}{dt}(0+) = 1000As^{-1}$. Represent the time-evolution of current $i(t)$ using a differential equation and solve it using a python program.
   (a) Plot $i(t)$, the voltage $v_L(t)$ across inductor, and $v_C(t) = 1 - Ri(t) - L\frac{di}{dt}$ across capacitor for $t = 0$ to 0.005 seconds.
   (b) Observe if $v_C(t)$ increases or oscillates. What do you infer from this observation?

10) Let x be a vector of length N. We say x crosses 0 at index m if either of the following holds: (i) $x[m-1] > 0$ and $x[m] \leq 0$, (ii) $x[m-1] = 0$ and $x[m] < 0$, (iii) $x[m-1] < 0$ and $x[m] \geq 0$, or (iv) $x[m-1] = 0$ and $x[m] > 0$. Write a python function that outputs the indices at which x crosses zero. (Try to implement it without using for loops, exploiting the power of vectorized computing and making use of standard functions.)

11) We are given with an input polynomial

$$p(t) \quad = \quad t^5 + t^3 + t + 1.$$

Write a python program to check if $p(t)$ has a root t_0 in the range $t \in [-1, \ +1]$. If yes, identify an approximate value of t_0.

12) We are given with an input polynomial

$$p(x) \quad = \quad 2t^4 - t^3 + 3t^2 + 1.$$

Write a python program:
   (a) to plot derivative of $p(t)$ in the range $t \in [-1, \ +1]$.
   (b) to identify a local minimum of $p(t)$ (if it exists) in the range $t \in [-1, \ +1]$ making use of its derivative.

13) The energy of a signal $x(t)$ time-limited in $[-T, T]$ is given by

$$E[x] \quad = \quad \int_{-T}^{T} |x(t)|^2 dt.$$

Write python program to compute energy of

(a)

$$x(t) = \begin{cases} t & 0 \le t \le 1 \\ 2-t & 1 \le t \le 2 \\ 0 & \text{otherwise.} \end{cases}$$

(b)

$$x(t) = \begin{cases} t^2/2 & 0 \le t \le 1 \\ 0 & \text{otherwise.} \end{cases}$$

14) Consider the function

$$y = \sqrt{1-(x-1)^2}$$

(a) Plot y against x in the range $x \in [0\ 2]$.
(a) Compute area under the curve y, and use the result to get an estimate of π.

15) Let $a = 2$ be fixed. Consider the function

$$y = \begin{cases} \sqrt{3}x & 0 \le x \le 0.5a \\ \frac{\sqrt{3}}{2}a & 0.5a \le x \le 1.5a \\ \sqrt{3}(2a-x) & 1.5a \le x \le 2a \end{cases}$$

(a) Plot y against x in the range $x \in [0\ 2a]$.
(a) Compute area under the curve y and use the result to find area of a regular hexagon of side a.

16) A csv file contains data on age and number of students with that age. The data in two such files *file1.csv* and *file2.csv* need to be merged. If data associated to particular age a is present in both *file1.csv* and *file2.csv*, then the corresponding entries s_1 and s_2 must be added. If a particular age is present only in one of the files, then no addition is required. Plot the histogram of merged data.

17) There are n electrons and r energy levels. An electron occupies one of the energy levels uniformly at random. This can be simulated by a random variable that takes value in the range $\{1, 2, \ldots, r\}$. Once all electrons occupy various energy levels, let X_i denote the number of electrons occupying level i, $1 \le i \le r$. An energy level is called dominant if it is occupied by maximum number of electrons. Let M denote the number of electrons in dominant energy level.
(a) Fix $n = 1000$, $r = 50$.
(b) Repeat the experiment of placing n electrons into r energy levels $T = 1000$ times. In each time, determine the $M(t), 1 \le t \le T$.
(c) Compute

$$M_{\text{avg}}(T) = \frac{1}{T}\sum_{t=1}^{T} M(t)$$

Plot $M_{\text{avg}}(T)$ for $n = 100, 500, 1000, 1500, 2000$ for fixed $r = 50$. Plot $M_{\text{avg}}(T)$ for $r = 10, 50, 100, 150, 200$ for fixed $n = 1000$.

18) Work out the simulation for Wigner's semicircle law that was given as assignment.

19) Let x be a vector of length N. Write a python program to determine mean, median and mode of values in x without using functions in *numpy*.