CS215-Assignment2-Report

Anand Narasimhan

October 2022

1 Introduction

2 Question 1

2.1 Generating random points inside an ellipse

First, let's proceed with generating random points inside a circle, and then scale the x and y co-ordinates by the respective factors to produce an ellipse (Since there is a 1 to 1 mapping between each point in a circle and the scaled ellipse)

For a circle, we will use the method that we used last time for a general function : The inverse sampling method, from the inverse of the CDF. We've copied the explanation from the previous assignment over here.

" We observe that the $rand()$ function outputs values from $[0, 1)$ following a uniform distribution. We also observe that the CDF of any distribution is an increasing function with range $[0, 1]$.

The way we want our random generator from the distribution to work is as follows :

We want the probability of an infinitesimally small interval from x to $x + dx$ being chosen to be $P(x)dx$, where $P(x)$ is our PDF for the distribution. Makes sense, right? The probability of a small interval around the point x being selected is proportional to the PDF at that point, and also ensuring that the total probability of something being selected is 1, we arrive at the probability $P(x)dx$ for an infinitesimally small interval x to $x + dx$.

Now, how do we ensure this? We need to link this to what we have, and that is a $rand()$ function which picks out an interval from y to $y + dy$ with a constant probability dy, irrespective of about which y the interval is chosen.

So let's look at the CDF. Let's say the interval x to $x + dx$ is mapped to the interval y to $y+dy$ by the CDF. So since the CDF is increasing, we can also have a reverse mapping from y to x. So let's say that our strategy is to pick a length from y to $y + dy$ with $y \in [0, 1)$ (The number 1, by itself, isn't included so as to make the interval within bounds), and output the corresponding length of x to

$x + dx$ as the output of our generator. We observe that, the lengths dy and dx are related as $dy = P(x)dx$, and this allows our method to work properly, since the probability of selecting a length dy is dy itself (since each point is equally likely for the $rand()$ function, and the total length is 1, the probability of the output being dx is also dy, which when written in terms of dx becomes $P(x)dx$, which is what we wanted. ””

So now, coming to the actual process. We selected an angle θ uniformly over 0 to 2π, and what we needed to do was to select the distance r such that the probability varies linearly with r (Since the probability of a particular value of r selected is proportional to the length of its circumference). Therefore, the CDF varied as r^2, and so the inverse CDF varied as $\sqrt{r}$. Therefore, we sampled the values of r by taking the square root of the number generated using the $rand()$ function, which would produce a number between 0 and 1, with probability varying linearly.

Now that we have both r and θ, we need to scale the unit circle such that it maps to an ellipse with major axis 2 and minor axis 1. We first calculate X and Y by calculating $\cos(\theta)$ and $\sin(\theta)$ respectively, and then scale X by 2 to get the actual random points for the required ellipse.

2.2 Generating random points inside a triangle

The way we went about this was to first construct a uniform distribution of a unit square, using two random numbers for the x and y co-ordinates, scaling it so that it became a rectangle with height e and length π and then performing certain flips to get the distribution that we wanted.

We partitioned the rectangle into two parts by the line $x = \pi/3$, and then proceeded to deal with the parts individually.

For the left part, we partitioned that rectangle into two halves about the diagonal passing through the origin, namely the equation $y - \frac{3e}{\pi}x = 0$. The points with this value greater than 0 were flipped into the other half, by the operations $x = \pi/3 - x$ and $y = e - y$

For the second part, a similar operation was performed, this time about the diagonal connecting the top vertex of the triangle and the bottom right vertex, the equation being

$$\frac{2\pi}{3}y + (x - \pi)e = 0$$

with points that had a value greater than 0 flipped into the other half by the operations $x = 4\pi/3 - x$ and $y = e - y$.

3 Question 2

3.1 Generating random points

The method to generate the random points was as follows :

We are given the matrix C, and we use the function $eig()$ to calculate the eigenvalues and the matrix of eigenvectors. Let the diagonal matrix with diagonal elements being the eigenvalues be D and the orthogonal matrix (Since matrix C is symmetric, it is diagonalised by an orthogonal matrix) be P. Then, we have

$$P^{-1}CP = D \iff PDP^{-1} = C$$

We also know that the multivariate gaussian can be written in terms of a linear combination of univariate gaussians in the following way :

$$X = AW + \mu$$

where X is the multivariate gaussian, W is a column vector of i.i.d univariate normally distributed gaussians, and A is a matrix of coefficients.

We also know that $C = AA^T$ as well. We have another form for C as well, $C = PDP^{-1} = PDP^T$ since $P^{-1} = P^T$ and we can write D as a product of two identical diagonal matrices, each of whose diagonals consist of the square root of the elements of D. Let that new matrix be called E. We also know that since E is diagonal, $E = E^T$.

Therefore, we have $C = PEE^TP = AA^T$. Therefore, a possible value for A is clearly

$$A = PE$$

Our method now is just to find the matrix W by a random normal gaussian generator, multiply it by $A = PE$, where the values of P and E can be found from the $diag()$ function, and then add the mean vector μ to it.

We repeat the above process for different N, by sampling N random values for the W vector in general, by writing W not as a vector, but as a matrix with the number of columns being the value of N.

3.2 MLE of the mean

To do this, we take the log likelihood function, that is, the sum of the logarithms of the PDF of the multivariate gaussian, differentiate it wrt to μ and equate it to 0.

In the slides, there is a formula stated

$$\frac{\partial}{\partial \mu}(\mu - x)^T C^{-1}(\mu - x) = 2C^{-1}(\mu - x)$$

We get the log likelihood function to be

$$\sum_{i=1}^{N} -0.5(y_i - \mu)^T C^{-1}(y_i - \mu) - \log((((2\pi)^D)|C|)^{0.5})$$

where the second term is a constant wrt to μ

The differentiation can be taken into the sum, and it becomes :

$$\sum_{i=1}^{N} \frac{\partial}{\partial \mu} - 0.5(y_i - \mu)^T C^{-1}(y_i - \mu) = \sum_{i=1}^{N} C^{-1}(y_i - \mu)$$

Equating this to 0, we can multiply by C on both sides to make the equation

$$\mu = \frac{1}{N} \sum_{i=1}^{n} y_i$$

which is the result for the MLE of the mean.

3.3 MLE of the covariance

We have the same log likelihood function

$$\sum_{i=1}^{N} -0.5(y_i - \mu)^T C^{-1}(y_i - \mu) - \log((((2\pi)^D)|C|)^{0.5})$$

and we need to differentiate this wrt to C this time and equate it to 0, to get our estimate for the covariance

Again, the formulas given on sir's slides were

$$\frac{\partial}{\partial C}(x - \mu)^T C^{-1}(x - \mu) = -C^{-T}(x - \mu)(x - \mu)^T C^{-T}$$

and

$$\frac{\partial}{\partial C} \log(|C|) = C^{-T}$$

Using this to take the derivative of the likelihood function wrt to C and setting it to 0, we get :

$$\sum_{i=1}^{N} \left(-0.5C^{-T}(y_i - \mu)(y_i - \mu)^T C^{-T} - 0.5C^{-T}\right) = 0$$

$$C^{-T} \left(\sum_{i=1}^{N} (y_i - \mu)(y_i - \mu)^T \right) C^{-T} = NC^{-T}$$

$$\frac{1}{N} \sum_{i=1}^{N} (y_i - \mu)(y_i - \mu)^T = C^T = C \qquad \text{(Since } C \text{ is symmetric)}$$

Therefore, we have our MLE estimate for C, in terms of our MLE estimate for μ, which we calculated previously. Using this, we calculate the errors.

4 Question 3

5 Question 5

For this question, since we had to represent the original 784 co-ordinate space in a space with only 84 vectors, we look for the eigenvectors of the co-variance matrix (which we calculate for a specific digit based on the samples provided for that digit) that have the maximum eigenvalues, and therefore have the maximum variance associated with their marginals.

Therefore, the function to compute the 84 co-ordinates involves changing the basis from the standard basis (which is what the image is represented as) into a basis of those 84 eigenvectors, by projecting each and every point onto the hyperplane made up by these eigenvectors (basis vectors of the subspace)

Our 84 co-ordinates will then be a vector of the coefficients, when that projected point is represented as a linear combination of those 84 eigenvectors.

To get those coefficients, we first sort the eigenvalues and their corresponding eigenvectors in decreasing order, so that the 84 eigenvectors with the largest eigenvalue are the first 84 columns of the orthogonal matrix of eigenvectors.

After that, we transform the basis from the standard basis into the basis of the entire set of eigenvectors (not just the 84, all the 784) by pre-multiplying with the inverse of the eigenvector matrix (which also happens to be the transpose). We then project it onto the plane, which in this basis, is as simple as just selecting the first 84 coefficients and making all the other coefficients 0. The 84 co-ordinates that are required are the first 84 coefficients.

To convert these coefficients into an image again, we add 0s to those 84 co-ordinates to make it a vector of length 784, and then pre-multiply with the eigenvector matrix, which takes us back into the standard basis. We then reshape it into a 28×28 matrix and then make an image out of it.