STAT 580 Homework 2 Yonghyun Kwon

March 9, 2020

1 Problem 1

Let $Z = \sqrt{Y}\frac{X}{||X||}$, $U = X^TX$ then apply change of variable formula to obtain the joint distribution of Z and U.X and Y then can be written as

$$X = \sqrt{U}\frac{Z}{Z^TZ}$$
$$Y = Z^TZ$$

In this problem, however, direct derivation of Jacobian matrix is not easy. Therefore, substitute first and apply chain rule to obtain the determinant of Jacobian matrix.

$$V = Z$$
$$W = \frac{\sqrt{U}}{\sqrt{Z^TZ}}$$

$$X = VW$$
$$Y = V^TV$$

The determinant of Jacobian matrix is

$$\begin{aligned}
\left|\frac{\partial(x,y)}{\partial(z,u)}\right| &= \left|\frac{\partial(v,w)}{\partial(z,u)}\right|\left|\frac{\partial(x,y)}{\partial(v,w)}\right| \\
&= \begin{vmatrix} I_p & 0 \\ -\sqrt{u}|z^Tz|^{-3/2}z & \frac{1}{2\sqrt{u}}\frac{1}{\sqrt{z^Tz}} \end{vmatrix} \times \begin{vmatrix} wI_p & v \\ 2v^T & 0 \end{vmatrix} \\
&= \left|\frac{1}{2\sqrt{u}\sqrt{z^Tz}}\right| \left|wI_p\right| \left|0 - 2v^T(wI)^{-1}v\right| \\
&= \frac{|w|^{p-1}v^Tv}{\sqrt{u}\sqrt{z^Tz}} = \left(\frac{u}{z^Tz}\right)^{\frac{p}{2}-1}
\end{aligned}$$

Since $X \sim N_p(\mathbf{0}, \mathbf{I})$ and $Y \sim \chi^2_p$ are independent, the joint distribution of Z and U is

$$\begin{aligned}
f_{ZU}(z,u) &= f_{XY}(x,y)\left|\frac{\partial(x,y)}{\partial(z,u)}\right| \\
&= \left|2\pi I_p\right|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}x^T x\right) \frac{1}{\Gamma\left(\frac{p}{2}\right)2^{\frac{p}{2}}} y^{\frac{p}{2}-1} \exp\left(-\frac{y}{2}\right)\left(\frac{u}{z^T z}\right)^{\frac{p}{2}-1} \\
&= \left|2\pi I_p\right|^{-\frac{1}{2}} \exp\left(-\frac{u}{2}\right) \frac{1}{\Gamma\left(\frac{p}{2}\right)2^{\frac{p}{2}}} \left(z^T z\right)^{\frac{p}{2}-1} \exp\left(-\frac{z^T z}{2}\right)\left(\frac{u}{z^T z}\right)^{\frac{p}{2}-1} \\
&= \left|2\pi I_p\right|^{-\frac{1}{2}} \exp\left(-\frac{z^T z}{2}\right) \frac{1}{\Gamma\left(\frac{p}{2}\right)2^{\frac{p}{2}}} (u)^{\frac{p}{2}-1} \exp\left(-\frac{u}{2}\right) I\{u>0\} \\
&= f_Z(z) f_U(u)
\end{aligned}$$

Since the joint distribution is splited into two functions, Z and U are independent and by integrating with respect to u, we get

$$f_Z(z) = \left|2\pi I_p\right|^{-\frac{1}{2}} \exp\left(-\frac{z^T z}{2}\right)$$

2 Problem 2

The written code is attached with the homework file.(pass_value.c) The result of the code and its short explanation is as follows:

*1. Integer variable **d** and its pointer **pd** are defined.*

—Outside the function—

address of the integer is &d = 0x7ffff2566d4c

value of the pointer is pd = 0x7ffff2566d4c

address of the pointer to the integer is &pd = 0x7ffff2566d50

*2. Since **pd** points the address of **d**, address of the integer **d** is equal to the value of the pointer **pd**.*

—-Inside the function before increment—-

address of the integer is &i = 0x7ffff2566d2c

value of the pointer is pi = 0x7ffff2566d4c

address of the pointer to the integer is &pi = 0x7ffff2566d20

*3. When a function is called, its arguments are copied and assigned to a memory. In this case, value of **d**, 0 is copied and assigned to new memory(**i**) and the value of **pd** is copied to the memory of **pi**. The value of the pointer **pi** is equal to that of **pd**, but their addresses are different.*

—-Inside the function after increment—-

address of the integer is &i = 0x7ffff2566d2c

value of the pointer is pi = 0x7ffff2566d50

address of the pointer to the integer is &pi = 0x7ffff2566d20

*4. After increment, the address of **i** and **pi** does not change. However, the value of **i** are increased by 1 and the value of **pi** is increased by 4. Since integer type variable has 4 byte, adding 1 to the pointer results in 4 increament. Also, note in this case that value of the pointer **pi** is equal to the address of **pd**.*

—Outside the function—

address of the integer is &d = 0x7ffff2566d4c

value of the pointer is pd = 0x7ffff2566d4c

address of the pointer to the integer is &pd = 0x7ffff2566d50

*5. Since the function passes its arguements by reference, the address of integer **&d**, value of the pointer **pd**, and the address of the pointer to the integer **&pd** all do not change.

3 Problem 3

We can write a function that executes the matrix multiplication algorith using CCS representation that I wrote in the proveious homework. The code is attached with this document.(ccs_product.c)

Function *cssprod* takes *x* array, *y* array, *a* array, *r* array, *c* array, and the length of *a* array: *lena*. This algortithm does the same work as that in the homework2 does.

In main function, an example for performing the matrix multiplication is given. In this example, the matrix and array are given as follows:

$$W \times x = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 4 \\ 4 & 0 & 10 & 0 & 0 & 0 & 5 \\ 0 & 13 & -4 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5.5 \\ 6 \\ 7 \end{pmatrix} = \begin{pmatrix} 30.0 \\ 69.0 \\ 14.0 \\ 18.5 \\ 0.0 \\ 0.0 \end{pmatrix}$$

$$a = (4, 2, 1, 13, 10, -4, 3, 4, 5)$$
$$r = (1, 3, 0, 2, 1, 2, 3, 0, 1)$$
$$c = (0, 2, 4, 6, 6, 7, 7, 9)$$
$$x = (1, 2, 3, 4, 5.5, 6, 7)$$

In main function, W is not stored and x, a, r, c are only stored in memory. Also, y is initialized to 0 array.

We can check the expected output in C as follows

y[0] = 30.000000 y[1] = 69.000000 y[2] = 14.000000 y[3] = 18.500000 y[4] = 0.000000 y[5] = 0.000000

4 Problem 4

4.1 (a)

To find the Maximum Likelihood Estimator for μ and Σ, first, assume that $p_1 \geq p_2 \geq \cdots \geq p_n$, without loss of generality, since we can swap the index of samples to satisfy this condition.

Now write n by p matrix X

$$X = \begin{pmatrix} & * & * & * & \cdots \\ z_1 & & * & * & \cdots \\ & z_2 & z_3 & * & \cdots \\ & & & z_4 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{1}$$

where $*$ refers to missing data.

Also, denote $n_1 =$ length of $z_1, \cdots, n_p =$ length of z_p

The log pdf of $z_1, \cdots, z_p$ can be written as

$$\begin{aligned} \log_{\mu,\Sigma} f(z_1, \cdots, z_p) &= \log_{\mu,\Sigma} f(z_1) + \log_{\mu,\Sigma}(z_2|z_1) + \cdots \log_{\mu,\Sigma}(z_p|z_1, \cdots z_{p-1}) \\ &= \log_{\mu,\Sigma} f(z_1) + \log_{\mu,\Sigma}(z_2|z_{1(z_2)}) + \cdots \log_{\mu,\Sigma}(z_p|z_{1(z_p)}, \cdots z_{p-1(z_p)}) \end{aligned}$$

where $z_{k(z_j)}$ is a length of n_j vector extracted from z_k. For instance, we can write

$$z_1 = \begin{pmatrix} \vdots \\ z_{1(z_2)} \end{pmatrix}$$

Now assume that we have mle of $k-1$ vector μ_{k-1} and $k-1 \times k-1$ vector $\Sigma_{(k-1)(k-1)}$ matrix. Then we try to find a scalar μ_k, $k-1$ vector $\Sigma_{(k-1)k}$, and scalar Σ_{kk} inductively

Note that for $k = 2, \cdots, k = p$, the conditional distribution of z_k can be written as

$$z_k|\mathcal{X}_k \sim N_{n_k}(\mathbb{1}_{n_k}\beta_{k0} + \mathcal{X}_k\beta_{k1}, \sigma_k^2 I_{n_k})$$

where $\mathcal{X}_k$ is $n_k \times (k-1)$ matrix

$$\mathcal{X}_k = \begin{pmatrix} z_{1(z_k)} & z_{2(z_k)} & \cdots & z_{k-1(z_k)} \end{pmatrix}$$

This is because, writting $z_{j(z_k)} = (z_{j(z_k),1}, \cdots, z_{j(z_k),n_k})^T$, $z^*_{k-1,i} = (z_{1(z_k),i}, \cdots, z_{k-1(z_k),i})^T$, for $i = 1, \cdots n_k$,

$$z_{ki}|z^*_{k-1,i} \overset{i.i.d.}{\sim} N_{n_k}\left(\mu_k + \Sigma_{k(k-1)}\Sigma^{-1}_{(k-1)(k-1)}(z^*_{k-1,i} - \mu_{k-1}), \sigma_{kk\cdot(k-1)}\right)$$

where μ_{k-1}, $\Sigma_{(k-1)k}$ and $\Sigma_{(k-1)(k-1)}$ are $k-1$ vector, $k-1$ vector, and $(k-1) \times (k-1)$ matrix each. This implies that

$$z_k|\mathcal{X}_k = \begin{pmatrix} z_{k1}|z^*_{k-1,1} \\ \vdots \\ z_{kn_k}|z^*_{k-1,n_k} \end{pmatrix} \sim N_{n_k}\left(\mathbb{1}_{n_k}\mu_k + [\mathcal{X}_k - \mathbb{1}_{n_k}\mu^T_{k-1}]\Sigma^{-1}_{(k-1)(k-1)}\Sigma_{(k-1)k},\ \sigma_{kk\cdot(k-1)}I_{n_k}\right)$$
$$\stackrel{let}{=} N_{n_k}(\mathbb{1}_{n_k}\beta_{k0} + \mathcal{X}_k\beta_{k1}, \sigma^2_k I_{n_k})$$

Due to invariance property of MLE, we can find the MLE of μ, Σ by obtaining the MLE of β_{k0}'s, β_{k1}'s, and σ^2_k. Moreover, finding the MLE of such β_{k0}'s, β_{k1}'s, and σ^2_k is equivalent to finding the MLE of regression model:

$$z_k = \mathbb{1}_{n_k}\beta_{k0} + \mathcal{X}_k\beta_{k1} + \varepsilon$$

where $\varepsilon \sim N_{n_k}(0, \sigma^2_k I_{n_k})$

Recall that the MLE of regression model is

$$\hat{\beta}_{k1} = \left[\mathcal{X}^T_k\left(I_{n_k} - \frac{1}{n_k}\mathbb{1}_{n_k}\mathbb{1}^T_{n_k}\right)\mathcal{X}_k\right]^{-1}\mathcal{X}^T_k(z_k - \mathbb{1}_{n_k}\bar{z_k})$$
$$\hat{\beta}_{k0} = \bar{z}_k - \frac{1}{n_k}\mathbb{1}^T_{n_k}\mathcal{X}_k\hat{\beta}_{k1}$$
$$\hat{\sigma}^2_k = \frac{1}{n_k}||z_k - \mathbb{1}_{n_k}\hat{\beta}_{k0} - \mathcal{X}_k\hat{\beta}_{k1}||^2$$

Now, from the above parametrazation, it follows that

$$\Sigma_{(k-1)k} = \Sigma_{(k-1)(k-1)}\hat{\beta}_{k1}$$
$$\mu_k = \mu^T_{k-1}\hat{\beta_{k1}} + \hat{\beta_{k0}}$$
$$\sigma^2_{kk} = \hat{\beta}^T_{k1}\Sigma_{(k-1)(k-1)}\hat{\beta}_{k1} + \hat{\sigma}^2_k$$

Hence, we can find the MLE inductively, starting from

$$\hat{\mu}_1 = \frac{1}{n}\sum_i^n x_{i1}$$
$$\hat{\sigma}_{11} = \frac{1}{n}\sum_i^n(x_{i1} - \bar{x}_1)^2$$

4.2 (b)

Suppose that we observe all data and there is no missingness. The complete log-likelihood is

$$l(\mu, \Sigma; X_1, \cdots, X_n) = -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_i(X_i - \mu)^T\Sigma^{-1}(X_i - \mu)$$

4.3 E-step

Now, conditioning the log-likelihood on the observed data X^o,

$$\begin{aligned} E[l(\mu, \Sigma; X_1, \cdots, X_n)|X^o] &= -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_i tr\left(\Sigma^{-1}E\left[(X_i-\mu)(X_i-\mu)^T|x_{i1}, \cdots, x_{ip_i}\right]\right) \\ &= -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_i tr\left(\Sigma^{-1}\left[(X_iX_i^T)^* - X_i^*\mu^T - \mu X_i^{*T} + \mu\mu^T\right]\right) \end{aligned}$$

where

$$E\left[X_i|x_{i1}, \cdots, x_{ip_i}\right] = E\left[X_i|X_i^o\right] = \begin{pmatrix} X_i^o & \mu_{i2} + \Sigma_{i21}\Sigma_{i11}^{-1}(X_i^o - \mu_{i1}) \end{pmatrix} \stackrel{let}{=} X_i^*$$

$$E\left[X_iX_i^T|X_i^o\right] = \begin{pmatrix} X_i^o & X_i^* \\ X_i^{*T} & \Sigma_{i22} - \Sigma_{i21}\Sigma_{i11}^{-1}\Sigma_{i12} + X_i^*X_i^{*T} \end{pmatrix} \stackrel{let}{=} (X_iX_i^T)^*$$

$$\begin{aligned} \mu_{i1} &= E[X_i^o] \\ \Sigma_{i11} &= Var[X_i^o] \\ \Sigma_{i21} &= Cov[X_i^M, X_i^o] \\ \Sigma_{i22} &= Var[X_i^M] \end{aligned}$$

4.4 M-step

Now, maximize

$$\begin{aligned} E[l(\mu, \Sigma; X_1, \cdots, X_n)|X^o] = &-\frac{n}{2}\log|2\pi\Sigma| \\ &- \frac{1}{2}\sum_i tr\left(\Sigma^{-1}\left[(X_iX_i^T)^* - X_i^*X_i^{*T}\right]\right) - \frac{1}{2}\sum_i\left((\mu - X_i^*)^T\Sigma^{-1}(\mu - X_i^*)\right) \end{aligned}$$

Differentiating with respect to μ and solving the estimating equation,

$$\hat{\mu}^{(k+1)} = \frac{1}{n}\sum_i^n X_i^*$$

Then differentiating with respect tot Σ and solving the estimating equation, we get

$$\hat{\Sigma}^{(k+1)} = \frac{1}{n}\left(\sum_i (X_iX_i^T)^* - \left(\sum_i X_i^*\right)\left(\sum_i X_i^*\right)^T\right)$$

4.5 (c)

The number of computations needed in finding MLE depends on β_{k1} and β_{k0}. If we are regressing n vector y to $n \times p$ matrix X the computational complexity would be dominated by

$$\begin{aligned} X^T X &\implies p^2 n \\ X^T y &\implies pn \\ (X^T X)^{-1} X^T y &\implies p^3 \end{aligned}$$

Since pn andp^3 can be regarded as negligible, the computation is determined by $p^2 n$. In this problem, since we are regressing $n_k \times k$ matrix $(\mathbb{1}_{n_k} \mathcal{X}_k)$ to n_k vector z_k, the computational complexity is

$$o\left(\sum_{k=1}^{p} k^2 n_k\right)$$

On the other hand, the number of computations needed in one EM step is dominated by

$$\Sigma_{i11}^{-1} \Sigma_{i12} \implies p_i^2 q_i$$

where p_i is the number of observed variables for i-th data and q_i is the numbrer of missing variables for i-th data.

So the computational complexity for EM algorithm is

$$o\left(\sum_{i=1}^{n} p_i^2 q_i\right)$$

To compare the number of computations, we assume that $n_k = m(p-k+1)$, That is, for k-th column, the number of missing data is $m(k-1)$ each. Then the computational complexity for MLE is

$$o\left(\sum_{k=1}^{p} k^2 n_k\right) = o\left(m \sum_{k=1}^{p} k^2 (p-k)\right) = o\left(\frac{mp^4}{12}\right)$$

For EM algorithm,

$$o\left(\sum_{i=1}^{n} p_i^2 q_i\right) = o\left(m \sum_{k=1}^{p} k(p-k)\right) = o\left(\frac{mp^3}{6}\right)$$

If p is large enough, one EM step would be efficient than direct maximization of the loglikelihood.

4.6 (d)

Suppose p_i observed coordinates are not the first ones. Then it is possible that we cannot apply direct maximiazation of loglikelihood as we did in (a). Since the first column still has mising data, we cannot find mle inductively from μ_1 and $\Sigma_1 1$. Thus, we have to find the maximum likelihood by iterative method.

However, for EM algorithm, by swapping the index of columns appropriately, we can use the same method as above. So the number of computational would be proportional to $o\left(\frac{mp^3}{6}\right)$