MTH 9821 Numerical Methods for Finance – Homework 2

Bledar Kulemani, Avinash Oza, Bingcheng Wang, Xinlu Xiao, Wenli Dong

September 8, 2016

1 Problem 1 (by Avinash Oza)

Nothing required to be turned in.

2 Problem 6.3 (by Xinlu Xiao)

(i)

$$A = \begin{pmatrix} 1 & -0.2 & 0.8 \\ -0.2 & 2 & 0.3 \\ 0.8 & 0.3 & 1.1 \end{pmatrix}$$

$R_1 = \sum_{k=2:3} |A(1,k)|$, so $R_1 = 0.2 + 0.8 = 1$, and $|A(1,1)| = 1$, we know that $|A(1,1)| = R_1$.
$R_2 = \sum_{k=1,3} |A(2,k)|$, so $R_2 = 0.2 + 0.3 = 0.5$, and $|A(2,2)| = 2$, we know that $|A(2,2)| > R_2$.
$R_3 = \sum_{k=1,2} |A(3,k)|$, so $R_3 = 0.8 + 0.3 = 1.1$, and $|A(3,3)| = 1.1$, we know that $|A(3,3)| = R_3$.
So by the definition of weakly diagonal dominant matrix, we have $|A(j,j)| \geq R_j, \forall j = 1:3$, so matrix A is weakly diagonal dominant matrix.
And from the Gershgorin's Theroem, we know that $|A(j,j) - \lambda| \leq R_j$, so we can get:

$$A(j,j) - \lambda \leq |A(j,j) - \lambda| \leq R_j$$

and therefore,

$$\lambda \geq A(j,j) - R_j$$

And since A is a weakly diagonal dominant matrix, so that $A(j,j) \geq R_j$, then we can conclude that $\lambda \geq 0$ for any eigenvalue λ of A, and therefore that the matrix A is symmetric positive definite.

(ii) We can calculate all the leading minors of the matrix A:

$$det\begin{pmatrix}1\end{pmatrix} = 1 > 0$$

$$det\begin{pmatrix} 1 & -0.2 \\ -0.2 & 2 \end{pmatrix} = 1.96 > 0$$

$$det\begin{pmatrix} 1 & -0.2 & 0.8 \\ -0.2 & 2 & 0.3 \\ 0.8 & 0.3 & 1.1 \end{pmatrix} = 0.69 > 0$$

So that all the leading principal minors are positive, according to Sylvester's Criterion, we know that the matrix A is symmetric positive definite.

3 Problem 6.6 (by Xinlu Xiao)

$$B_N = \begin{pmatrix} 2 & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix} = U_N^t U_N$$

$$= \begin{pmatrix} U_N(1,1) & 0 & 0 & \dots & 0 \\ U_N(1,2) & U_N(2,2) & \dots & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ U_N(1,N) & \dots & \dots & \dots & U_N(N,N) \end{pmatrix} \begin{pmatrix} U_N(1,1) & U_N(1,2) & \dots & \dots & U_N(1,N) \\ 0 & U_N(2,2) & \dots & \dots & U_N(2,N) \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & U_N(N,N) \end{pmatrix}$$

From the multiplication of the matrix, we know that

$$U_N(1,1)^2 = B_N(1,1), \quad so \quad U_N(1,1) = \sqrt{2}$$

$$U_N(1,1)U_N(1,1:N) = B_N(1,1:N)$$

And since B_N is a triangular symmetric positive definite matrix, we know that $B_N(1,3:N) = \boldsymbol{0}$, and since $U_N(1,1) \neq 0$, so that $U_N(1,3:N) = \boldsymbol{0}$.
And

$$U_N(1,1)U_N(1,2) = B_N(1,2), so \quad U_N(1,2) = \frac{B_N(1,2)}{U_N(1,1)} = -\frac{1}{\sqrt{2}}$$

then we update the matrix B_N

$$B_N(2:N,2:N) = B_N(2:N,2:N) - (U_N(1,2:N))^t U(1,2:N)$$

$$= \begin{pmatrix} 2 & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix} - \begin{pmatrix} -\frac{1}{\sqrt{2}} \\ 0 \\ \dots \\ 0 \end{pmatrix} \begin{pmatrix} -\frac{1}{\sqrt{2}} & 0 & \dots & 0 \end{pmatrix} = \begin{pmatrix} \frac{3}{2} & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix}$$

And similarly, we can get the

$$U_N(2,2) = \sqrt{\frac{3}{2}}, and \quad U_N(2,3) = \frac{B_N(2,3)}{U_N(2,2)} = -\frac{1}{\sqrt{\frac{3}{2}}} = -\sqrt{\frac{2}{3}}, \quad and \quad U_N(2,4:N) = \boldsymbol{0}$$

And we can use the induction, assume that $U_N(i,i) = \sqrt{\frac{i+1}{i}}, \forall i = 1:N$ and $U_N(i,i+1) = -\sqrt{\frac{i}{i+1}}, \forall i = 1:N-1$

$$B_N((i+1):N,(i+1):N) = B_N((i+1):N,(i+1):N) - (U_N(i,(i+1):N))^t U(i,(i+1):N)$$

$$= \begin{pmatrix} 2 & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix} - \begin{pmatrix} -\sqrt{\frac{i}{i+1}} \\ 0 \\ \dots \\ 0 \end{pmatrix} \begin{pmatrix} -\sqrt{\frac{i}{i+1}} & 0 & \dots & 0 \end{pmatrix} = \begin{pmatrix} \frac{i+2}{i+1} & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix}$$

So that we can get

$$U_N(i+1,i+1) = \sqrt{\frac{i+2}{i+1}}$$

$$U_N(i+1,i+2) = \frac{B_N(i+1,i+2)}{U_N(i+1,i+1)} = -\sqrt{\frac{i+1}{i+2}}$$

So the induction has been proved.
Thus

$$U_N(i,i) = \sqrt{\frac{i+1}{i}}, \forall i = 1:N$$

$$U_N(i,i+1) = -\sqrt{\frac{i}{i+1}}, \forall i = 1:N-1$$

4 Problem 6.7 (by Bledar Kulemani)

Let B_N be the N x N tridiagonal symmetric positive matrix be given by (1).

$$\mathbf{B_N} = \begin{pmatrix} 2 & -1 & \dots & 0 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & -1 & 2 \end{pmatrix} \tag{1}$$

and let U_N be the Cholesky factor of the matrix B_N be given by:

$$U_N(i,i) = \sqrt{\frac{i+1}{i}} \tag{2}$$

$$U_N(i,i+1) = -\sqrt{\frac{i}{i+1}} \tag{3}$$

(i) Show that the solution to a linear system $B_N * x =$ b, where b and x are the N x 1 column vectors can be obtained by using the explicit pseudocode below:

```
Function Call:
x = linear_solve_cholesky_B_N(b)

Input:
b = N x 1 column vector

Output:
x = solution to $B_N * x$ = b

y(1)= $\frac{b(1)}{\sqrt{2}}$
for $i$=2:$N$
    y($i$) = $\frac{b(i)-y(i-1)*\sqrt{\frac{i-1}{i}}}{\sqrt{\frac{i+1}{i}}}$
end

x($N$) = $\frac{y(N)*\sqrt{N}}{\sqrt{N+1}}$
for $i$=2:$N$
    y($i$) = $\frac{y(i)+x(i+1)*\sqrt{\frac{i}{i+1}}}{\sqrt{\frac{i+1}{i}}}$
end
```

(ii) What is the operation count for the pseudocode above, and how does it compare to $8n+O(1)$, the operation count for the optimal linear solver for the tridiagonal symmetric definite matrices?

<u>Proof</u>

(i)

We know that $U_N^T U_N x = b$. If we let $U_N x = y$, then we have $U_N^T y = b$. Note that U_N^T is a lower triangular matrix. Therefore, this can be solved using the forward substitution(L,b) algorithm. Its implementation is as follows:

$y(1) = \frac{b(1)}{U_N^T(1,1)} = \frac{b(1)}{\sqrt{B_N(1,1)}} = \frac{b(1)}{\sqrt{2}}$

Then, equation (2) becomes:

$$U_N(i,i) = \sqrt{\frac{i+1}{i}} \qquad for \;\; every \;\; i = 2:N \tag{4}$$

and equation (3) becomes:

$$U_N(i,i+1) = -\sqrt{\frac{i-1}{i}} \qquad for \;\; every \;\; i = 2:N \tag{5}$$

Note that to move from equation (3) to equation (5) we just let $i = i-1$. We then use the forward_subst_bidiag(L,b) to find the vector y.

```
y(1)= $\frac{b(1)}{\sqrt{2}}$

for $i$=2:$N$
        $y(i) = \frac{b(i)-y(i-1)L(i,i-1)}{L(i,i)} = \frac{b(i)-y(i-1)U(i-1,i)}{L(i,i)} = \frac{b(i)+y(i-1)\sqrt{\frac{i-1}{i}}}{\sqrt{\frac{i+i}{i}}}$        Note that $L(i,j) = U(j,i)$
end
```

Now that we have the vector y, we can use it to find vector x in $U_B x = y$. To solve for x, we use the backward substitution algorithm for bidiagonal matrices. The algorithm is shown in Table 2.4 of the book.

```
$x(N) = \frac{y(N)}{U_N(N,N)} = \frac{y(N)}{\sqrt{\frac{N+1}{N}}} = \frac{y(N)\sqrt{N}}{\sqrt{N+1}}$
for $i$=2:$N$
    $y(i) = \frac{y(i)-x(i+1)U(i,i+1)}{U(i,i)} = \frac{y(i)-x(i+1)*(-\sqrt{\frac{i}{i+1}})}{\sqrt{\frac{i+1}{i}}} = \frac{y(i)+x(i+1)*\sqrt{\frac{i}{i+1}}}{\sqrt{\frac{i+1}{i}}}$
end
```

If we put these two loops together, then we see that the explicit pseudocode solves the system $B_N x = y$.

(ii)
The loop that runs from 2:N has:
a) $(N-2)+1$ iterations
b) 9 operations per iteration.

The loop that runs from N-1:1 has:
a) $(N-1)-1+1$ iterations
b) 9 operations per iterations

Additionally, there are 2 operations needed to calculate $y(1)$ and 5 operations to calculate $x(N)$. Therefore, the total no. of ops. is $18(N-1)+7 = 18n + O(1)$.

5 Problem 6.8 (by Bledar Kulemani)

Let B_N be the NxN tridiagonal symmetric positive definite matrix given in problem 7. Show that the L and U factors of the matrix B_N are the lower triangular bidiagonal matrix and the upper triangular bidiagonal matrix given by:
$L(i,i) = 1$ for every $i = 1 : N$ and $L(i+1,i) = -\frac{i}{i+1}$ for every $i = 1 : (N-1)$
$U(i,i) = \frac{i+1}{i}$ for every $i = 1 : N$ and $U(i,i+1) = -1$ for every $i = 1 : (N-1)$

Proof

It is important to notice that the leading principal minors of the matrix B_N are non-zero. Therefore, we can apply the lu_no_pivoting_tridiag(A) algorithm provided in the Table 2.7. We first initialize the lower triangular matrix L to Identity matrix and the upper triangular matrix U to the Zero matrix, i.e.

```
$L = \mathbf{I}_{NxN}$
$U = \mathbf{0}_{NxN}$

for $i = 1 : (N-1)$
    $L(i,i) = 1$
    $L(i+1,i) = \frac{A(i+1,i)}{A(i,i)}$
    $U(i,i) = A(i,i)$
    $U(i+1,i+1) = A(i,i+1)$
    $A(i+1,i+1) = A(i+1,i+1) - L(i+1,i) * U(i,i+1)$
end
$L(n,n) = 1;$    $U(n,n) = A(n,n)$
```

To prove the claim, we will use induction.

Basis: $i = 1$
Using the algorithm above, we have:

$L(1,1) = 1$
$L(2,1) = \frac{A(2,1)}{A(1,1)} = \frac{-1}{2}$ and

$U(1,1) = A(1,1) = 2$
$U(1,2) = A(1,2) = -1$

As we can see from the basis step, the case when $i = 1$ is true. Let's assume that the claim is true for $i = k$, i.e.
$L(k,k) = 1$
$L(k+1,k) = \frac{-k}{k+1}$ and

$U(k,k) = \frac{k+1}{k}$
$U(k,k+1) = -1$

Then, we need to check whether the claim is true for $i = k+1$, i.e.
$L(k+1,k+1) = 1$
$L(k+2,k+1) = \frac{-(k+1)}{k+2}$ and

$U(k+1,k+1) = \frac{k+2}{k+1}$
$U(k+1,k+2) = -1$

After step k, the LU decomposition without row pivoting updates the block of A that is on the bottom right corner of element with index (k,k), i.e.

$$A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - L(k+1:n, k) * U(k, k+1:n)$$

Particularly,
$A(k+1,k+1) = A(k+1,k+1) - L(k+1,k) * U(k,k+1)$
$A(k+1,k+1) = 2 - \frac{-k}{k+1} * (-1)$ (by induction hypothesis)
$A(k+1,k+1) = \frac{k+2}{k+1}$

Additionally,

$A(k+1,k+2) = A(k+1,k+2) - L(k+2,k) * U(k,k+2)$
$A(k+1,k+1) = -1 - \frac{A(k+2,k)}{U(k,k)} * U(k,k+2)$ (by induction hypothesis)
$A(k+1,k+1) = -1$ (since $A(k+2,k) = 0$)

Therefore we have:

$U(k+1,k+1) = A(k+1,k+1) = \frac{k+2}{k+1}$
$U(k+1,k+2) = A(k+1,k+2) = -1$ and

$L(k+2,k+1) = \frac{A(k+2,k+1)}{A(k+1,k+1)}$
$L(k+2,k+1) = \frac{A(k+1,k+2)}{A(k+1,k+1)}$ (by symmetry)
$L(k+2,k+1) = \frac{-1}{\frac{k+2}{k+1}} = -\frac{k+1}{k+2}$
$L(k+1,k+1) = 1$ (since L is a LU factor)

Then, by induction, the proof follows. Note that the operation count is $3n + O(1)$.

6 Problem 6.9 (by Bledar Kulemani)

Let B_N be the NxN tridiagonal symmetric positive definite matrix given in problem 7, and let L and U be the LU factors of the matrix B_N given in problem 8.
(i) Show that the solution to a linear system $B_N * x = b$ where b and x are the $Nx1$ column vectors can be obtained by using the pseudocode below:

```
Function Call:
x = linear_solve_LU_B_N(b)

Input:
b = N x 1 column vector

Output:
x = solution to $B_N * x =$ b

y(1)= $b(1)$
for $i$=2:$N$
    $y(i) = b(i) + \frac{(i-1)*y(i-1)}{i}$
end

$x(N) = \frac{N_y(N)}{N+1}$
for $i$=2:$N$
    y($i$) = $\frac{i*(y(i)+x(i+1))}{i+1}$
end
```

<u>Proof</u>

$L * U * x = b$
$L * Y = b$

We will first use the forward substitution to find the vector b.

```
$y(1) = \frac{b(1)}{L(1,1)}$
for $i = 2 : N$
    $y(i) = \frac{b(i)-L(i,i-1)*y(i-1)}{L(i,i)}$
    $y(i) = \frac{b(i)-(-\frac{i-1}{i})*y(i-1)}{1}$
    $y(i) = b(i) + \frac{i-1}{i} * y(i-1)$
end
```

Then, we can use backward_subst_bidiag to solve for the vector x.

```
    $x(N) = \frac{y(N)}{U(N,N)} = \frac{y(N)}{\frac{N+1}{N}} = \frac{y(N)*N}{N+1}$

for $i = (n-1) : 1$
    $x(i) = \frac{y(i)-U(i,i+1)*x(i+1)}{U(i,i)} = \frac{y(i)+x(i+1)}{\frac{i+1}{i}} = \frac{i*(y(i)+x(i+1))}{i+1}$
end
```

After putting these two pieces of code together, we see that the algorithm is produced. Operation count:

First for loop: $4 * (N + 1)$
Second for loop: $4 * (N + 1)$
Operation count in calculating $x(N)$ is 3
Operation count in calculating $y(1)$ is 0, since $y(1) = b(1)$

The total operation count for this algorithm is: $8 * (N - 1) + 3 = 8N - 5 = 8n + O(1)$

7 Problem 6.10 (by Bledar Kulemani)

Write an explicit optional pseudocode for solving linear systems corresponding to the same tridiagonal symmetric positive definite matrix. In other words, write a pseudocode for solving p linear systems $A * x_i = b_i$, for $i = 1 : p$ where A is tridiagonal symmetric positive definite matrix.

Solution

An optimal explicit code to solve p linear systems corresponding to the same tridiagonal symmetric positive definite matrix is outlined below:
- Find the LU decomposition of the matrix
- Use a "for" loop to solve the system using forward and backward substitutions for each vector b_i

```
    Input:
A - tridiagonal symmetric positive definite matrix of size n
$b_i$ - column vector of size n

    Output:
$x_i$ - solution corresponding to the vector $b_i$
for $i = 1 : n - 1$
    $L(i,i) = 1$;    $L(i+1,i) = \frac{A(i+1,i)}{A(i,i)}$
    $U(i,i) = A(i,i)$;    $U(i,i+1) = A(i,i+1)$
    $A(i+1,i+1) = A(i+1,i+1) - L(i+1,i) * U(i,i+1)$
end

    $L(n,n) = 1$    $U(n,n) = A(n,n)$
// At this point we have the LU decomposition, and it will be used by the solvers
for $i = 1 : p$
    $y(1) = b_i(1)$
    for $j = 2 : n$
            $y(j) = b_i(j) - L(j,j-1) * y(j-1)$ // forward substitution for $L * y = b_i$
end

    $x_i(n) = \frac{y(n)}{U(n,n)}$
            for $j = (n-1) : i$
            $x_i(j) = \frac{y(j) - U(j,j+1) * x_i(j+1)}{U(j,j)}$
    end      // backward substitution for $U * x_i = y$
end
```

Now we calculate the operation count:

For the LU decomposition, there are $3 * (n-1)$. For the forward substitution, there are $2 * n - 2$. For the backward substitution, there are $3 * n - 2$. The number of operations for the forward and backward substitution is multiplied by p, since the code runs p-times.

Therefore, the total number of the ops is: $p * (3n - 2 + 2n - 2) + 3n - 3 = 5 * n * p + 3 * n - 4 * p - 3$

8 Problem 7.19 (by Xinlu Xiao)

Covariance matrix is symmetric positive semidefinite matrix, and we can calculate all the leading principal minors of the matrix, we can get

$$det \begin{pmatrix} 1 \end{pmatrix} = 1 > 0$$

$$det \begin{pmatrix} 1 & 1 \\ 1 & 4 \end{pmatrix} = 3 > 0$$

$$det \begin{pmatrix} 1 & 1 & 0.5 \\ 1 & 4 & -2 \\ 0.5 & -2 & 9 \end{pmatrix} = 20 > 0$$

So the covariance matrix is symmetric positive definite matrix, it has Cholesky decompositon, we can get

$$U = \begin{pmatrix} 1 & 1 & 0.5 \\ 0 & 1.7321 & -1.4434 \\ 0 & 0 & 2.5820 \end{pmatrix}$$

$$U^t = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1.7321 & 0 \\ 0.5 & -1.4434 & 2.5820 \end{pmatrix}$$

So that the three normal random variables given by

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} = U^t \begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1.7321 & 0 \\ 0.5 & -1.4434 & 2.5820 \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \end{pmatrix} = \begin{pmatrix} Z_1 \\ Z_1 + 1.7321 Z_2 \\ 0.5 Z_1 - 1.4434 Z_2 + 2.582 Z_3 \end{pmatrix}$$

9 Problem 8.3 (by Bingcheng Wang)

9.1 Subquestion i

The mid prices of the options are shown in Table 8.1.

Table 9.1: Mid Prices of Options

Strike	Mid Prices Call	Mid Prices Put
1450	432.8	9.55
1500	385.75	12.35
1550	339.55	16.05
1600	294.5	20.85
1675	229.7	30.55
1700	208.8	34.55
1750	168.8	44.45
1775	150.1	50.45
1800	131.7	57.3
1825	114.25	64.65
1850	97.7	73.05
1875	82.35	82.45
1900	68.05	93.3
1925	55.05	105.05
1975	33.65	133.55
2000	25.2	150.55
2050	13	188.3
2100	6.05	231.2

The OLS formula is

$$y = PVF - disc \times K$$

where $y = C - P$, C is the mid prices of the call options, P is the mid prices of the put options, and K is the strike price. By using OLS, we get

$$y = 1869.4031 - 0.9972K$$

Thus $PVF = 1869.4031$, $disc = 0.9972$.

9.2 Subquestion ii

The original Black-Scholes Formula is

$$C_{BS}(S, K, T, \sigma, r, q) = Se^{-qT}N(d_1) - Ke^{-rT}N(d_2)$$
$$P_{BS}(S, K, T, \sigma, r, q) = -Se^{-qT}N(-d_1) + Ke^{-rT}N(-d_2)$$

where S is the spot price of the index corresponding to these options, q is the dividend rate, r is the risk-free rate. The function $N(\cdot)$ is the standard normal cdf, and

$$d_1 = \frac{\log(S/K) + (r - q + \sigma^2/2)T}{\sigma\sqrt{T}}$$
$$d_2 = \frac{\log(S/K) + (r - q - \sigma^2/2)T}{\sigma\sqrt{T}}$$

We can rewrite Black-Scholes formula using PVF and $disc$, then

$$C_{BS} = PVF \cdot N(d_1) - K \cdot disc \cdot N(d_2)$$
$$P_{BS} = -PVF \cdot N(-d_1) + K \cdot disc \cdot N(-d_2)$$

and

$$d_1 = \frac{\log\left(\frac{PVF}{K \cdot disc}\right)}{\sigma\sqrt{T}} + \frac{\sigma\sqrt{T}}{2}$$
$$d_2 = \frac{\log\left(\frac{PVF}{K \cdot disc}\right)}{\sigma\sqrt{T}} - \frac{\sigma\sqrt{T}}{2}$$

Let C_{BS} and P_{BS} be the mid prices of the call and put options. We can use Newton's method to calculate σ, which will be the implied volatility. The implied volatility is shown in Table 8.2.

From Table 8.2 we can see that under the same strike price, the implied volatilities of calls and puts are very close.

Table 9.2: Implied Volatility

K	$\sigma^2_{\text{imp, c}}$	$\sigma^2_{\text{imp, p}}$
1450	0.0473548237418695	0.047763472746148
1500	0.0431902618976051	0.043488616065135
1550	0.039308491461635	0.0395996294773441
1600	0.0357431718481531	0.0359616052698727
1675	0.0307945166866716	0.0307075462798457
1700	0.0290614122800266	0.0289593032638681
1750	0.0258423104463723	0.0257818527826046
1775	0.0244890865947133	0.0242987325201529
1800	0.0229048005647355	0.0229011137326154
1825	0.0214286123535537	0.0213593275657909
1850	0.0199790343583373	0.0199229868497235
1875	0.0186480463469214	0.018513108797099
1900	0.0173158231345902	0.0172811093183057
1925	0.016049424826671	0.015938192074204
1975	0.0138591431080441	0.0137615241485985
2000	0.0128711121494107	0.0130099516032848
2050	0.0112692419430679	0.0114918512531617
2100	0.0101438878456743	0.0104521034753514

10 Problem 8.7 (by Bingcheng Wang)

10.1 Subquestion i

By using OLS regression, we get

$$T_{3,LR} = 0.0123 + 0.1272T_2 + 0.3340T_5 + 0.5298T_{10}$$

The approximation error

$$\text{error}_{LR} = ||T_3 - T_{3,LR}|| = 0.04301298$$

10.2 Subquestion ii

By using linear interpolation, we get $T_{3,linear_interp} =$ [4.650000, 4.770000, 4.773333, 4.783333, 4.783333, 4.796667, 4.780000, 4.786667, 4.810000 4.790000, 4.803333, 4.800000, 4.783333, 4.760000, 4.763333].

Thus the approximation error

$$\text{error}_{linear_interp} = ||T_3 - T_{3,linear_interp}|| = 0.2066129$$

10.3 Subquestion iii

By using cubic spline interpolation, we get $T_{3,cubic_interp} =$ [4.641333, 4.762000, 4.765889, 4.780889, 4.781222, 4.789111, 4.773667, 4.781778, 4.805667, 4.786000, 4.799556, 4.795333, 4.779556, 4.756333, 4.758222].

Thus the approximation error

$$\text{error}_{cubic_interp} = ||T_3 - T_{3,cubic_interp}|| = 0.1864353$$

10.4 Subquestion iv

The OLS regression has the least approximation error, and the linear interpolation has the approximation error. That is because OLS estimator is BLUE. It has very little residual, while linear interpolation is very rough.

11 Problem 8.8 (by Bingcheng Wang)

11.1 Subquestion i

The weekly percentage returns of these stocks are listed in Table 10.1.

Table 11.1: Weekly returns of some stocks

date	JPM	GS	MS	BAC	RBS	CS	UBS	RY	BCS
2012-01-23	0.0359	0.13	0.119	0.103	0.183	0.183	0.178	0.0391	0.139
2012-01-30	0.0289	0.0515	0.0944	0.0758	0.0436	0.0441	0.0438	0.0248	0.0718
2012-02-06	-0.0176	-0.029	-0.0317	0.0294	-0.0352	-0.0768	-0.0523	-0.00343	-0.019
2012-02-13	0.0231	0.0157	-0.0256	-0.00622	0.00912	0.0422	0.0305	-0.00134	0.0704
2012-02-20	0	0	0	0	0	0	0	0	0
2012-02-27	0.0561	0.0381	-0.0152	0.015	-0.00226	0.017	-0.0324	0.0689	0.0316
2012-03-05	0.00981	-0.0223	-0.0261	-0.00986	-0.0668	-0.0303	-0.0284	0.00501	-0.0625
2012-03-12	0.0862	0.0481	0.063	0.217	0.091	0.116	0.0719	0.0214	0.0667
2012-03-19	0.0133	0.0264	0.0407	0.00512	-0.00222	-0.014	-0.014	-0.00872	-0.0212
2012-03-26	0.0181	-0.0143	-0.0337	-0.0275	-0.0145	-0.0238	-0.0149	-0.000352	-0.0421
2012-04-02	-0.0294	-0.0513	-0.0636	-0.0356	-0.0939	-0.0655	-0.0655	-0.0137	-0.0827
2012-04-09	-0.0254	-0.0246	-0.0608	-0.0597	-0.0187	-0.0342	-0.0477	-0.0212	-0.0218
2012-04-16	-0.0113	-0.023	0.0117	-0.037	-0.0267	0.0133	0.00162	0.0283	0.00446
2012-04-23	0.0145	0.0175	-0.0276	-0.0132	0.0392	-0.0549	0.0161	0.0186	0.0636
2012-04-30	-0.0368	-0.0474	-0.0557	-0.062	-0.00881	-0.0883	-0.0214	-0.0506	-0.0709
2012-05-07	-0.115	-0.063	-0.0659	-0.0246	-0.0698	-0.0277	-0.0114	-0.0229	-0.0404
2012-05-14	-0.094	-0.065	-0.107	-0.0704	-0.145	-0.0707	-0.0755	-0.0606	-0.134
2012-05-21	0.000303	0.0127	-0.00752	0.0186	0.0399	0.00715	0.0266	-0.0356	0.0243
2012-05-28	0	0	0	0	0	0	0	0	0
2012-06-04	0.00546	-0.0176	0.0348	0.0589	0.0706	0.0264	0.013	0.00352	0.0431
2012-06-11	0.0401	0.0118	0.0425	0.045	0.122	-0.0697	0.0239	0.0243	0.0649
2012-06-18	0.0273	-0.0212	-0.0105	0.00507	-0.0268	-0.00425	-0.00334	0.0101	-0.00713
2012-06-25	-0.00734	0.0238	0.0312	0.0303	-0.108	-0.0245	-0.0201	0.0102	-0.183
2012-07-02	-0.0432	-0.00409	-0.0303	-0.0636	-0.0779	-0.0262	-0.0589	0.0124	-0.00293
2012-07-09	0.0642	0.0205	-0.00639	0.0209	0.0319	-0.0213	-0.0299	-0.00234	-0.00294
2012-07-16	-0.0603	-0.0335	-0.0907	-0.096	-0.0124	-0.0287	-0.0505	0	-0.0324
2012-07-23	0.0883	0.0795	0.0597	0.034	0.0814	0.0543	0.0798	0.00665	0.066
2012-07-30	-0.0216	-0.00652	0.0215	0.0164	-0.0145	-0.0364	-0.0173	-0.000194	0.0105
2012-08-06	0.0243	0.0202	0.0602	0.0418	0.0338	0.0221	0.0121	0.00369	0.0858
2012-08-13	0.000272	0.00565	-0.00137	0.0336	0.0355	0.0381	0.0128	0.0557	0.0477
2012-08-20	0.00518	0.0131	-0.00206	0.02	-0.0274	0.0526	0.0145	-0.00623	-0.0174
2012-08-27	-0.000813	0.0117	0.0302	-0.0209	0.0141	0.0026	-0.00446	0.033	-0.0194
2012-09-03	0	0	0	0	0	0	0	0	0
2012-09-10	0.119	0.148	0.216	0.197	0.253	0.2	0.209	0.0318	0.273
2012-09-17	-0.0167	-0.0382	-0.0636	-0.0461	-0.0111	-0.00995	-0.0415	-0.00658	-0.0263
2012-09-24	-0.00986	-0.026	-0.0199	-0.0307	-0.0662	-0.0756	-0.0573	0	-0.0381
2012-10-01	0.0381	0.0495	0.0454	0.0555	0.0168	0.0671	0.0509	0.0223	0.0454
2012-10-08	-0.00216	0.00746	-0.0109	-0.0215	0.0213	-0.0062	-0.0148	-0.0123	0.0193
2012-10-15	0.0168	0.0285	0.0127	0.0351	0.0347	0.0473	0.0349	0.0162	0.0088

11.2 Subquestion ii

By using OLS regression, we get

$$\begin{aligned} JPM = -0.003222 + 0.755635\ GS - 0.064294\ MS + 0.302755\ BAC + 0.227142\ RBS \\ - 0.094029\ CS - 0.398494\ UBS + 0.147026\ RY + 0.010758\ BCS \end{aligned}$$

where those tickers represent the weekly returns of the corresponding tickers.

The approximation error is 0.1319426.

11.3 Subquestion iii

By using OLS regression, we get

$$JPM = -0.001105 + 0.614013\ GS - 0.067123\ MS + 0.251915\ BAC$$

where those tickers represent the weekly returns of the corresponding tickers.

The approximation error is 0.1514875.

11.4 Subquestion iv

$$\begin{aligned} JPM = 8.83982 + 0.05480\ GS - 0.06069\ MS + 1.46197\ BAC + 0.93938\ RBS \\ 0.85709\ CS - 2.24728\ UBS + 0.28110\ RY - 0.07965\ BCS \end{aligned}$$

where those tickers represent the prices of the corresponding tickers.

The approximation error is 6.443954. Compared to subquestion ii, the approximation error is much larger.