ECE 133A HW 4

Lawrence Liu

November 1, 2022

Exercise A5.6

(a)

Let $DX+XD=A$, we have that $A_{ij}=(D_{ii}+D_{jj})X_{ij}$, Since $DX+XD=B$ we have that

$$A_{ij}=B_{ij}$$
$$B_{ij}=(D_{ii}+D_{jj})X_{ij}$$

Therefore we get that

$$X_{ij}=\frac{B_{ij}}{D_{ii}+D_{jj}}$$

for any i,j, this will exist since $D_{ii}+D_{jj}\neq 0$ for all i and j this computation will cost us 2 flops, 1 for addition and one for division so in total solving for all X_{ij} will cost us $2n^2$ flops.

(b)

Let

$$L = \begin{bmatrix} L_{11} & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{bmatrix}$$

$$X = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{21} & X_{22} & \cdots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{nn} \end{bmatrix}$$

Then we have that

$$LX = \begin{bmatrix} L_{11}X_{11} & L_{11}X_{12} & \cdots & L_{11}X_{1n} \\ L_{21}X_{11} + L_{22}X_{21} & L_{21}X_{12} + L_{22}X_{22} & \cdots & L_{21}X_{1n} + L_{22}X_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ L_{n1}X_{11} + \cdots + L_{nn}X_{n1} & L_{n1}X_{12} + \cdots + L_{nn}X_{n2} & \cdots & L_{n1}X_{1n} + \cdots + L_{nn}X_{nn} \end{bmatrix}$$

And

$$XL^T = \begin{bmatrix} L_{11}X_{11} & L21X_{11} + L22X_{12} & \cdots & L_{n1}X_{11} + \cdots + L_{nn}X_{1n} \\ L_{11}X_{21} & L21X_{21} + L22X_{22} & \cdots & L_{n1}X_{21} + \cdots + L_{nn}X_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ L_{11}X_{n1} & L21X_{n1} + L22X_{n2} & \cdots & L_{n1}X_{n1} + \cdots + L_{nn}X_{nn} \end{bmatrix}$$

therefore we have that

$$B_{ij} = \sum_{k=1}^{i} L_{ik}X_{kj} + \sum_{k=1}^{j} L_{jk}X_{ik}$$

Therefore if we know X_{lm} for all $0 \leq l \leq i$ and $0 \leq m \leq j$ Except for X_{ij} we can express

$$X_{ij} = \frac{B_{ij} - \sum_{k=1}^{i-1} L_{ik}X_{kj} - \sum_{k=1}^{j-1} L_{jk}X_{ik}}{L_{ii} + L_{jj}}$$

Since $L_{ii} + L_{jj} \neq 0$ for all i, j this will exist. The two summations will cost us $2(i-1+j-1)-2$ flops, and the subtractions will cost us 2 flops, and the division will cost us 2 flops since we need to first compute the sum $L_{ii} + L_{jj}$ so in total this computation will cost us $2(i+j)$ flops. Therefore to solve for all X_{ij} it will cost us

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{n} 2(i+j) &= 2\sum_{i=1}^{n}\sum_{j=1}^{n}(i+j) \\
&= 2\sum_{i=1}^{n} ni + \sum_{j=1}^{n} nj \\
&= 2\left(\frac{n^2(n+1)}{2} + \frac{n^2(n+1)}{2}\right) \\
&= \boxed{2n^2(n+1)}
\end{aligned}$$

Exercise A6.3

(a)

Since $S^T = -S$ we have that

$$S = \begin{bmatrix} 0 & c_{12} & \cdots & c_{1n} \\ -c_{12} & 0 & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -c_{1n} & -c_{2n} & \cdots & 0 \end{bmatrix}$$

therefore we have that for any $X = [x_1, x_2, \cdots, x_n]^T$

$$Sx = \begin{bmatrix} 0 + c_{12}x_2 + \cdots + c_{1n}x_n \\ -c_{12}x_1 + 0 + \cdots + c_{2n}x_n \\ \vdots \\ -c_{1n}x_1 - c_{2n}x_2 + \cdots + 0 \end{bmatrix}$$

Therefore we have:

$$x^T S x = \sum_{i=1}^{n} \sum_{j=i+1}^{n} c_{ij} x_i x_j - \sum_{i=1}^{n} \sum_{j=i+1}^{n} c_{ij} x_i x_j = 0$$

In order for $(I - S)x = 0$ will only happen if $x = 0$,

$$\begin{aligned}
(I - S)x &= 0 \\
x^T(I - S)x &= 0 \\
x^T I x - x^T S x &= 0 \\
x^T I x &= 0 x^T x && = 0 x \cdot x = 0
\end{aligned}$$

Therefore $(I - S)x = 0$ will only happen if $x = 0$ so we have that $I - S$ is nonsingular.

(b)

Similarly as how $I - S$ is nonsingular we can show that $I + S$ is nonsingular. Since

$$\begin{aligned}
(I + S)x &= 0 \\
x^T(I + S)x &= 0 \\
x^T I x + x^T S x &= 0 \\
x^T I x &= 0 x^T x && = 0 x \cdot x = 0
\end{aligned}$$

Which once again leads to the result that $(I + S)x = 0$ only when $x = 0$, and thus $I + S$ is nonsingular. Therefore we have that $(I - S)^{-1}$ and $(I + S)^{-1}$ exist. Thus we get

$$\begin{aligned}
(I + S)(I - S)^{-1} &= (2I - (I - S))(I - S)^{-1} \\
&= 2(I - S)^{-1} - I \\
&= (I - S)^{-1}(2I - (I - S)) \\
&= (I - S)^{-1}(I + S)
\end{aligned}$$

(c)

In order for a matrix A to be symmetric we must have that $A^T A = I$ we have

$$\begin{aligned}
\left((I+S)(I-S)^{-1}\right)^T \left((I+S)(I-S)^{-1}\right) &= \left((I-S)^{-1}\right)^T (I-S)(I+S)(I-S)^{-1} \\
&= \left((I-S)^{-1}\right)^T (I-S)(I-S)^{-1}(I+S) \\
&= \left((I-S)^{-1}\right)^T (I+S) \\
&= \left((I-S)^T\right)^{-1} (I-S)^{-1}(I+S) \\
&= \left((I+S)\right)^{-1} (I+S) \\
&= I
\end{aligned}$$

Therefore we have that $A = (I+S)(I-S)^{-1}$ is orthogonal.

Exercise A6.9

(a)

We have that

$$S^{k-1} = \begin{bmatrix} 0 & I_{k-1} \\ I_{n-k+1} & 0 \end{bmatrix}$$

Therefore S^{k-1} will circularly shift W when we multiple W by it, in other words, the ith row of WS^{k-1}, $\left(WS^{k-1}\right)_i$ will be

$$\left(WS^{k-1}\right)_i = \begin{bmatrix}\omega^{-i(k-1)} & \omega^{-ik} & \cdots & \omega^{-i(n-1)} & \omega^0 & \omega^i & \cdots & \omega^{i(k-2)}\end{bmatrix}$$

Likewise for $\mathbf{diag}(We_k)W$, we have that the ith row of $\mathbf{diag}(We_k)W$, $(\mathbf{diag}(W_{e_k})W)_i$ will be

$$\begin{aligned}
(\mathbf{diag}(We_k)W)_i &= \begin{bmatrix}\omega^{-i(k-1)} & \omega^{-i}\omega^{-i(k-1)} & \cdots & \omega^{i}n-1\omega^{-i(k-1)}\end{bmatrix} \\
&= \begin{bmatrix}\omega^{-i(k-1)} & \omega^{-ik} & \cdots & \omega^{-i(n+k-2)}\end{bmatrix}
\end{aligned}$$

Therefore for $k > 1$, (since it is obvious for $k = 1$ the equality holds)

$$(\mathbf{diag}(We_k)W)_i = \begin{bmatrix} \omega^{-i(k-1)} & \omega^{-ik} & \cdots & \omega^{-i(n-1)} & \omega^0 & \omega^i & \cdots & \omega^{i(k-2)} \end{bmatrix}$$

Since $\omega = e^{2\pi j/n}$.
Therefore we have that

$$WS^{k-1} = \mathbf{diag}(We_k)W$$

(b)

Let A_i be the matrix with the ith column being a and the rest of the matrix being 0, then we have that

$$\begin{aligned} T(a) &= \sum_{i=1}^{n} s^{i-1} A_i \\ &= \sum_{i=1}^{n} \frac{1}{n} W^H \mathbf{diag}(We_i) W A_i \\ &= \frac{1}{n} W^H \sum_{i=1}^{n} \mathbf{diag}(We_i) W A_i \end{aligned}$$

We have that WA_i is a matrix with the ith column being

$$\left[\sum_{j=1}^{n} a_j, \sum_{j=1}^{n} a_j \omega^{-(j-1}, \cdots \sum_{j=1}^{n} a_j \omega^{-(j-1)(n-1)} \right]$$

and the rest of the matrix being 0, therefore we have that, $\mathbf{diag}(We_i)WA_i$ is a matrix with the ith column being

$$\left[\sum_{j=1}^{n} a_j, \omega^{-i} \sum_{j=1}^{n} a_j \omega^{-(j-1}, \cdots \omega^{-i(n-1)} \sum_{j=1}^{n} a_j \omega^{-(j-1)(n-1)} \right]$$

Therefore we have that

$$\sum_{i=1}^{n} \mathbf{diag}(We_i) W A_i = \mathbf{diag}(Wa) W$$

And thus we have that

$$T(a) = \frac{1}{n} W^H \mathbf{diag}(Wa) W$$

(c)

Therefore we have that

$$T(a)x = \frac{1}{n} W^H \mathbf{diag}(Wa) Wx$$

Wx and Wa are the fourier transform of x and a respectively, therefore we can calculate them in $n\log(n)$ time each. Then we have that $\mathbf{diag}(Wa)Wx$ is simply just multiplying the ith value of Wa by the ith value of Wx, which can be done in n time. Then $\frac{1}{n}W^H\mathbf{diag}(Wa)Wx$ is just taking the inverse fourier transform of our calculated value, which will cost us $n\log(n)$ time. Therefore the total time of our algorithm is $\boxed{3n\log(n) + n}$ time

(d)

The inverse of $T(a)$ is

$$T(a)^{-1} = \frac{1}{n} W \mathbf{diag}(\frac{1}{Wa}) W^H$$

therefore we can calculate b more efficently, similarly to hwo we did it with part c. It is implemented below in Julia:

```
using FFTW
using PyPlot

function naive_calculation(a,b,n)
    A=hcat( [circshift(a,k) for k=0:n-1]...)
    x=A\b
    return x
end

function fast_method(a,b,n)
    a_w=fft(a)
    b_w=fft(b)
    x_w=a_w./b_w
    x=ifft(x_w)
    return x
end
```

```
function main()
    n=100
    a=rand(n)
    b=rand(n)
    n_runs=20

    sizes=LinRange(10,2500,n_runs)
    times_naive=zeros(length(sizes))
    times_fast=zeros(length(sizes))
    for i in 1:n_runs
        n=sizes[i]
        println("n=$n")
        time_naive=0
        time_fast=0
        n=round(Int,n)
        for j in 1:100
            a=rand(n)
            b=rand(n)
            time_naive+=@elapsed naive_calculation(a,b,n)
            time_fast+=@elapsed fast_method(a,b,n)
        end
        times_naive[i]=time_naive/100
        times_fast[i]=time_fast/100
    end
    plot(sizes,times_naive,label="naive")
    plot(sizes,times_fast,label="fast")
    legend()
    xlabel("n")
    ylabel("time")
    title("Time to solve Ax=b")
    savefig("HW4/69d.png")
    close("all")

end
main()
```

Which results in the following graphs for the times of the two methods:

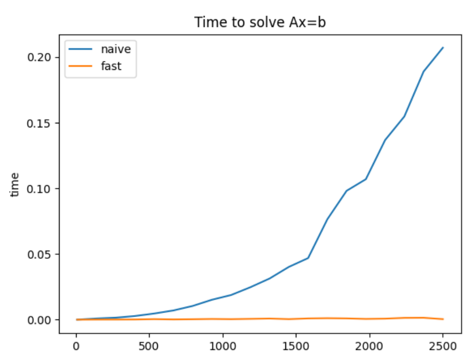

Figure 1: Time of the two methods

Exercise A6.15

(a)

We have

$$
\begin{aligned}
Q \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} Q^T &= \frac{1}{2} \begin{bmatrix} I_n & I_n \\ J_n & -J_n \end{bmatrix} \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} \begin{bmatrix} I_n & J_n \\ I_n & -J_n \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} I_n & I_n \\ J_n & -J_n \end{bmatrix} \begin{bmatrix} I_n & J_n \\ -I_n & J_n \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 0 & 2J_n \\ 2J_n & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & J_n \\ J_n & 0 \end{bmatrix}
\end{aligned}
$$

If a matrix is orthogonal then we have that $Q^T Q = I_{2n}$, therefore we have that

$$
\begin{aligned}
Q^T Q &= \frac{1}{2} \begin{bmatrix} I_n & J_n \\ I_n & -J_n \end{bmatrix} \begin{bmatrix} I_n & I_n \\ J_n & -J_n \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 2I_n & 0 \\ 0 & 2I_n \end{bmatrix} \\
&= I_{2n}
\end{aligned}
$$

So Q is orthogonal.

(b)

We have

$$
\begin{aligned}
J_{2n}A &= AJ_{2n} \\
Q^T J_{2n}A &= Q^T AJ_{2n} \\
Q^T J_{2n}AQ &= Q^T AJ_{2n}Q \\
Q^T J_{2n}AQ &= Q^T AQ \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} Q^T Q \\
Q^T J_{2n}AQ \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} &= Q^T AQ \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} \\
Q^T J_{2n}AQ \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} &= Q^T AQ \\
Q^T AQ &= \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} Q^T AQ \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} \\
Q^T AQ &= \frac{1}{2} \begin{bmatrix} I_n & J_n \\ -I_n & J_n \end{bmatrix} A \begin{bmatrix} I_n & -I_n \\ J_n & J_n \end{bmatrix}
\end{aligned}
$$

Let us express A in terms of n x n submatricies, B, C, D, and E. We have $A = \begin{bmatrix} B & D \\ E & C \end{bmatrix}$, and

$$
\begin{aligned}
J_{2n}A &= AJ_{2n} \\
\begin{bmatrix} 0 & J_n \\ J_n & 0 \end{bmatrix} \begin{bmatrix} B & D \\ E & C \end{bmatrix} &= \begin{bmatrix} B & D \\ E & C \end{bmatrix} \begin{bmatrix} 0 & J_n \\ J_n & 0 \end{bmatrix} \\
\begin{bmatrix} J_nE & J_nC \\ J_nB & J_nD \end{bmatrix} &= \begin{bmatrix} DJ_n & BJ_n \\ CJ_n & EJ_n \end{bmatrix}
\end{aligned}
$$

Therefore we have:

$$
\begin{aligned}
Q^T A Q &= \frac{1}{2}\begin{bmatrix} I_n & J_n \\ -I_n & J_n \end{bmatrix} A \begin{bmatrix} I_n & -I_n \\ J_n & J_n \end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix} I_n & J_n \\ -I_n & J_n \end{bmatrix} \begin{bmatrix} B & D \\ E & C \end{bmatrix} \begin{bmatrix} I_n & -I_n \\ J_n & J_n \end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix} I_n & J_n \\ -I_n & J_n \end{bmatrix} \begin{bmatrix} B + DJ_n & -B + DJ_n \\ E + CJ_n & -E + CJ_n \end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix} B + DJ_n + J_nE + J_nCJ_n & -B + DJ_n - J_nE + J_nCJ_n \\ -B - DJ_n + J_nE + J_nCJ_n & B - DJ_n - J_nE + J_nCJ_n \end{bmatrix}
\end{aligned}
$$

From $J_{2n}A = AJ_{2n}$ we get that $DJ_n - J_nE = 0$ and $CJ_n - J_nB = 0$. Therefore we have that $-B + DJ_n - J_nE + J_nCJ_n = 0$ and $-B - DJ_n + J_nE + J_nCJ_n = 0$ and thus we get:

$$
Q^T A Q = \frac{1}{2}\begin{bmatrix} B + DJ_n + J_nE + J_nCJ_n & 0 \\ 0 & B - DJ_n - J_nE + J_nCJ_n \end{bmatrix}
$$

(c)

Multiplying both sides of $Ax = b$ by Q^T we get:

$$Q^T AQx = Q^T b$$

Then letting $x = Qy$ we get

$$Q^T AQy = Q^T b$$

Since $Q^T AQ$ is block diagonal, this is just solving two n x n systems of equations. which will take $2 \cdot \frac{2}{3}n^3$ operations. Since multiplying b by Q^T takes $2n^2$ operations, and multiplying y by Q takes $2n^2$ operations, we have that the leading term of our imporoved solution is $\frac{4}{3}n^3$ So we have been able to reduce the number of flops by a factor if 4.

Exercise A6.16

We have that

$$A_{ij} = \sum_{k=1}^{n} Q_{ik} R_{kj}$$

Since R is upper triangular we have that

$$A_{ij} = \sum_{k=i}^{j} Q_{ik} R_{kj}$$

Exercise A7.5

We first LU factorize A, which will cost us $\frac{2}{3}n^3$ flops. Then we solve for $x' = A^{-1}b$, which will cost us $2n^2$ flops, then with this we can solve for $x'' = A^{-2}b = A^{-1}x'$, which will cost us $2n^2$ flops. Then once again, solving for $x''' = A^{-3}b = A^{-1}x''$, which will cost us $2n^2$ flops. then sum up b and x', x'', x''' to get x, which will cost us $3n$ flops. So in total we will take $\boxed{\dfrac{2}{3}n^3 + 6n^2 + 3n}$ flops.

Exercise A7.10

First we LU factorize A, which will cost us $\frac{2}{3}n^3$ flops. Then we solve for $x = A^{-1}b$, which will cost us $2n^2$ flops, and we solve for $x_u = A^{-1}u$, which will cost us $2n^2$ flops, then we have

$$y = (A + uv^T)^{-1}b = x - \frac{1}{1 + v^T x_u} x_u v^T x$$

Then we calculate $v^T x_u$, which will cost us $2n - 1$ flops and $v^T x$ which will likewise cost $2n - 1$ flops. Then we have

$$y = (A + uv^T)^{-1}b = x - \frac{v^T x}{1 + v^T x_u} x_u$$

Solving for x consists of first of all finding $\frac{x_u v^T}{1+v^T x_u}$ given $v^T x_u$ and $v^T x$, which will cost 2 flops, and then we multiply that value to every value in x_u, which will cost us n flops, and then subtracting the resulting vector from x will once again cost us n flops, thus we have that the total flops needed would be:

$$\frac{2}{3}n^3 + 4n^2 + 4n + 2n = \boxed{\frac{2}{3}n^3 + 4n^2 + 6n}$$