數值線性代數 Final

b05502087 王竑睿

1 A function $f : R^{m\times n} \rightarrow R$ is unitarily invariant if

$$f(A) = f(UAV)$$

for any A $\in R^{m\times n}$ **and for any orthogonal U** $\in R^{m\times n}$ **and V** $\in R^{m\times m}$

(a) Use the fact that the Frobenius norm is unitarily invariant to show that

$$\|A\|_F = \sum_{i=1}^{r} \sigma_i^2$$

where $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq \sigma_r > 0$ **are singular values of A,** $r \leq \min(m, n)$

[solution:]

$$\begin{aligned} &\|A\|_F \\ =&\left\|U \sum V\right\|_F \quad \text{(SVD分解)} \\ =&\left\|\sum\right\|_F \quad \text{(Frobenius norm 是 unitarily invariant)} \\ =&\sqrt{\sum_{i=1}^{r} \sigma_i^2} \quad \text{(Frobenius norm：square root of the sum of squares of its elements)} \end{aligned}$$

(b) Show that the truncated SVD $A_k = U_k \sum_k V_k^T$**,** $k \leq r$**, provides the optimal rank-k approximation of A in the sense defined by**

$$\|A - A_k\|_F = \min_{B, rank(B)=k} \|A - B\|_F$$

where $\|A - A_k\|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2}$

[solution:]

- 令B是所有rank-k的matrices，令V是B的列向量所張出的空間，因此V的維度為k
- 若要minimize $\|A - B\|_F^2$, B的各列向量應該要是A投影到V空間上的結果
  否則只要把B的row換成A的對應列在V上的投影即可得到更小的結果
- 因為$A_k = U_k \sum V_k^T$能夠最小化A的列向量到任何k維子空間的平方距離
  因此A_k即為所求

推導$\|A-A_k\|_F=\sqrt{\sum_{i=k+1}^{r}\sigma_i^2}$

令u_k為U的各行向量,v_k為V的各行向量

$$\begin{aligned}
&A-A_k\\
=&\sum_{i=1}^{r}\sigma_i u_i v_i'-\sum_{i=1}^{k}\sigma_i u_i v_i'\\
=&\sum_{i=k+1}^{r}\sigma_i u_i v_i'\\
\Rightarrow&\|A-A_k\|_F\\
=&\left\|\sum_{i=k+1}^{r}\sigma_i u_i v_i'\right\|_F\\
=&\left\|U'\sum_{i=k+1}^{r}\sigma_i u_i v_i' V\right\|_F \quad \text{(Frobenius norm 是 unitarily invariant)}\\
=&\left\|\sum_{i=k+1}^{r}\sigma_i e_i e_i'\right\|_F \quad \text{(Unitary Matrix 乘入變成單位向量)}\\
=&\sqrt{\sum_{i=k+1}^{r}\sigma_i^2}
\end{aligned}$$

(c) Following part (b), what is $\|A-A_k\|_2$ and what is $\|A-A_k\|_*$?

[**solution:** $\|A-A_k\|_2$]

$$\begin{aligned}
&\left\|(A-A_k)v\right\|_2 \quad \text{(v是}A-A_k\text{的top singular vector)}\\
=&\left\|\sum_{i=k+1}^{r}\sigma_i u_i v_i' \sum_{j=k+1}^{r}\alpha_j v_j\right\|_2 \quad \text{將}v_j\text{利用}(v_1,v_2,...,v_r)\text{分解}\\
=&\left\|\sum_{i=k+1}^{r}\alpha_i\sigma_i u_i v_i' v_i\right\|\\
=&\left\|\sum_{i=k+1}^{r}\alpha_i\sigma_i u_i\right\| \quad \text{(v是單位向量)}\\
=&\sqrt{\sum_{i=k+1}^{r}\alpha_i^2\sigma_i^2} \quad \text{(向量的2-norm)}
\end{aligned}$$

⇒依據norm的定義，要找一個v使這個2-norm最大

因為$|v|^2=\sum_{i=1}^{r}\alpha_i^2=1$

又因為σ_{k+1}是σ_{k+1}到σ_r中最大的。

所以,讓$\alpha_{k+1}=1$,其餘為0,可得到最大值

$$\sigma_{k+1}$$

[solution: $\|A - A_k\|_*$]

$$\|A - A_k\|_*$$
$$= \left\| \sum_{i=k+1}^{r} \sigma_i u_i v_i' \right\|_*$$
$$= \sum_{i=k+1}^{r} \sigma_i$$

Claim that:

$$\|A - A_k\|_* = \sum_{i=k+1}^{r} \sigma_i \leq \|A - B_k\|_* \quad \text{(if } B_k = XY' \text{ has k columns)}$$

By triangle inequality, if $A = A' + A''$:

then $\sigma_1(A) \leq \sigma_1(A') + \sigma_1(A'')$

$$\Rightarrow \sigma_i(A') + \sigma_j(A'')$$
$$= \sigma_1(A' - A'_{i-1}) + \sigma_1(A' - A'_{j-1})$$
$$\geq \sigma_1(A' - A'_{i-1} - A''_{j-1})$$
$$\geq \sigma_1(A - A_{i+j-2}) \quad (\text{since } rank(A'_{i-1} + A''_{j-1}) \leq rank(A_{i+j-2}))$$
$$= \sigma_{i+j-1}(A)$$

[since $\sigma_{k+1}(B_k) = 0$, when $A' = A - B_k$ and $A'' = B_k$, we conclude that for $i \geq 1$, $j = k+1$]

$\sigma_i(A - B_k) \geq \sigma_{k+i}(A)$

Therefore,

$$\|A - B_k\|_* = \sum_{i=1}^{n} \sigma_i(A - B_k) \geq \sum_{i=k+1}^{n} \sigma_i(A) = \|A - A_k\|_*$$

2 Given the values of the Laplace transform at points $s_j, 0 < s_1 < ... < s_n < \infty$, we want to estimate the function f.

- **本題目中使用到的Gauss Quadrature參考:**

  `$https://github.com/sfstoolbox/sfs-matlab/blob/master/SFS_general/legpts.m$`

- 利用以下code，取得矩陣A, y以及true signal下的xtrue

```
function Xtrue = getTrueX(T)
    N = size(T);
    Xtrue = zeros(N);
    for i = 1:N
        t = T(i);
        if(t<=1)
            Xtrue(i) = t;
        elseif(1<=t && t<3)
            Xtrue(i) = 3/2-t/2;
        elseif(3<=t)
            Xtrue(i) = 0;
```

```
            end
        end
    end
    function A = getA(W,S,T)
        J = size(S,1);
        K = size(T,1);
        A = zeros(J,K);
        for j = 1:J
            for k = 1:K
                A(j,k) = W(k)*exp((-1)*S(j)*T(k));
            end
        end
    end
    function Y = getY(S)
        N = size(S);
        Y = zeros(N);
        for i = 1:N
            Y(i) = getLf(S(i));
        end
    end
    function S = log_dis(N)
        S = zeros(N,1);
        for j = 1:N
            temp = (-1 + (j-1)/20)*log(10);
            S(j) = exp(temp);
        end
    end
    function Lf = getLf(s)
        Lf = (2-3*exp((-1)*s)+exp((-3)*s))/(2*s*s);
    end
```

(a) To appreciate the ill-posedness of this problem, try to estimate the values $x_j = f(t_j)$ by direct solution of the system Ax = y, using the backslash command in Matlab, using analytically known data with no artificial error added to it

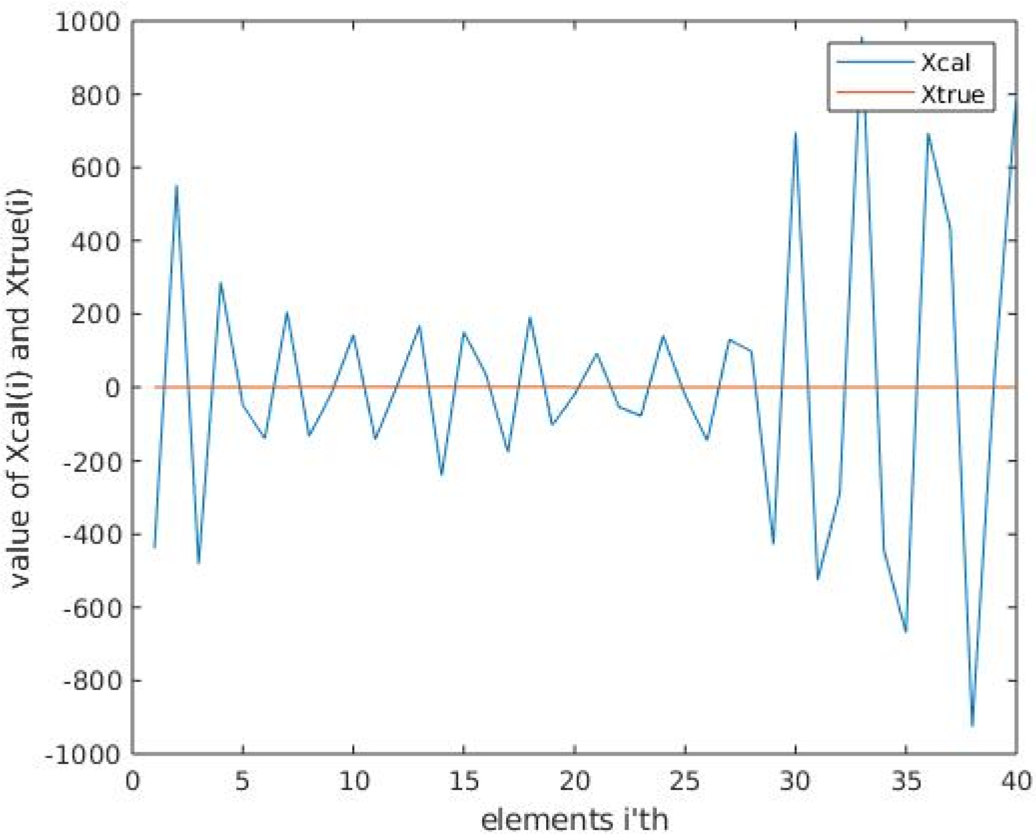
[Solution:]

利用以下code，進行matlab的左除運算，並且作圖

```
    function solve()
        N = 40;
        S = log_dis(N); %do logarithmically distributed to get sj
        Y = getY(S);
        [T W] = legpts(N,[0,5],'GW'); %do quad to get wk, tk
        A = getA(W,S,T);%use s t to get A
        Xcal = A\Y;%Ax=y
        Xtrue = getTrueX(T); %true f(t)
        plot([1:N],Xcal);
        hold on
        plot([1:N],Xtrue);
        xlabel("elements i'th");
```

```
        ylabel("value of Xcal(i) and Xtrue(i)");
        legend('Xcal','Xtrue')
        norm(Xcal-Xtrue,2)
    end
```

norm(Xcal-Xtrue,2)的結果為2.441640785552484e+03 $\approx$ 2441.64



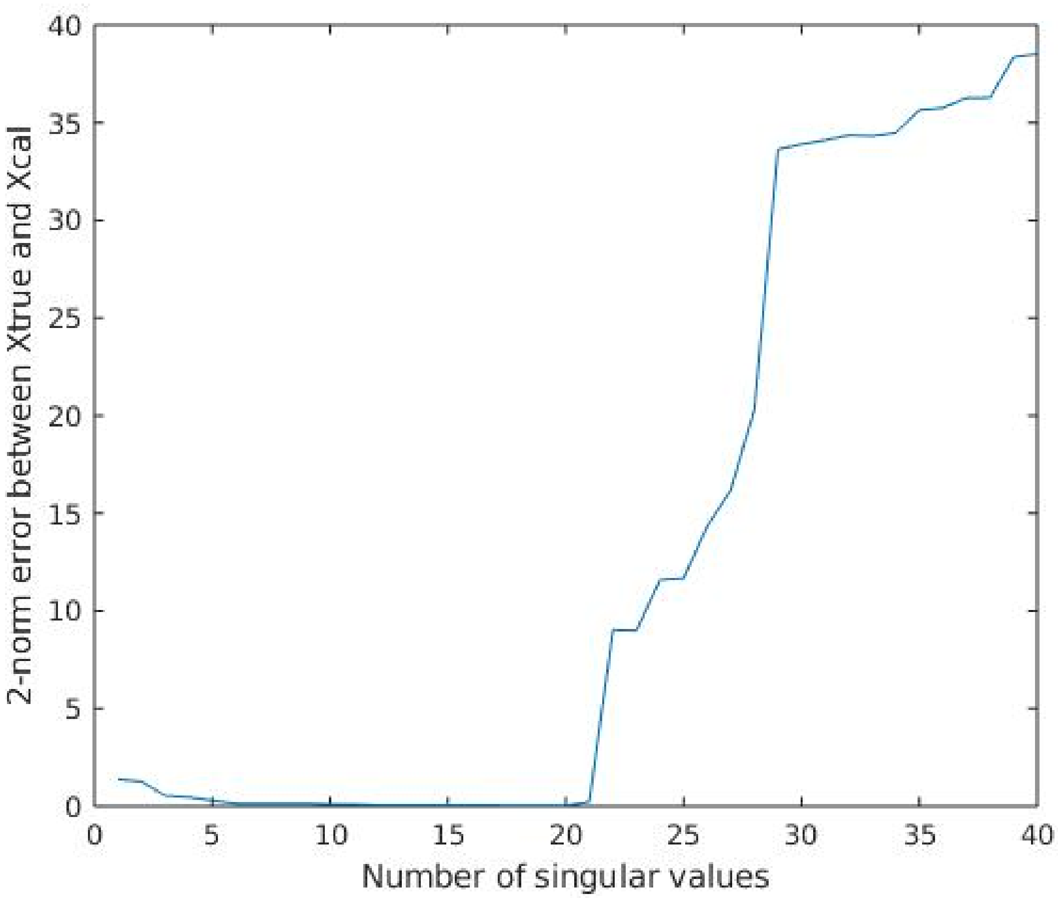
(b) Calculate the SVD of A and use TSVD to approximate the solution of this problem. How many singular values you need to get an error of the solution to $O(10^{-d})$, for some $d > 0$

[Solution:]

- 利用以下code，進行Truncated SVD並作圖
- matlab svd會對singular value進行排序，大的靠左上，因此我們從右下開始truncate

```
    function TSVD()
        %do quad to get wk, tk
        %do logarithmically distributed to get sj
        N = 40;
        S = log_dis(N);
        Y = getY(S);
        [T W] = legpts(N,[0,5],'FAST');
        %use s t to get A
        A = getA(W,S,T);
        [U,S,V] = svd(A);
        PLOTX = [];
```

```
    PLOTY = [];
    for singuNum = 1:N
        Snew = S;
        for i=1:N
            if(i>singuNum)
                Snew(i,i) = 0;
            else
                Snew(i,i) = 1./Snew(i,i);
            end
        end
        Strun = zeros(N,N);
        for i=1:N
            if(Snew(i,i)~=0)
                Strun(i,i) = Snew(i,i);
            end
        end
        Xcal = V*Strun*U'*Y;
        Xtrue = getTrueX(T);
        PLOTX = [PLOTX, size(nonzeros(diag(Snew)),1)];
        PLOTY = [PLOTY, norm(Xcal - Xtrue,2)];
    end
    plot(PLOTX,PLOTY);
    xlabel('Number of singular values')
    ylabel('2-norm error between Xtrue and Xcal')
end
```



- 由圖可知，在使用21個singular value (truncate掉19個)時，就能夠讓error降到10^{-1}以下
- 2-norm error約為0.226246062159252 ≈ 0.2262

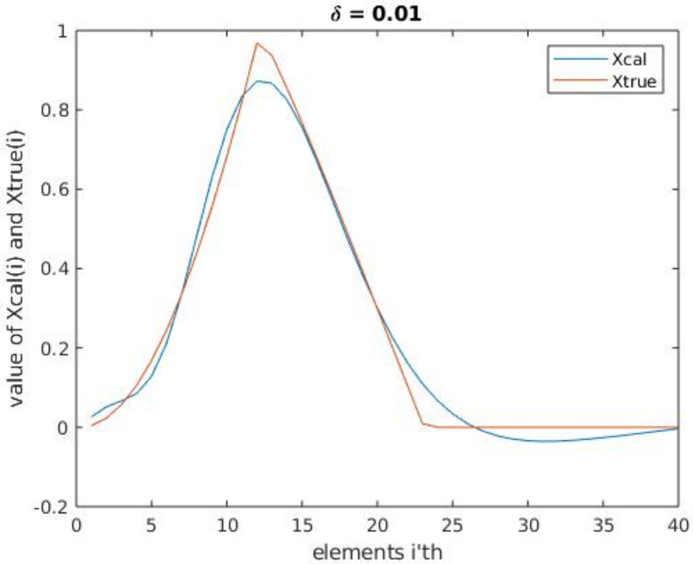
(c) Add a tiny random error to the data, and estimate x from the noisy data using the Tikhonov regularization

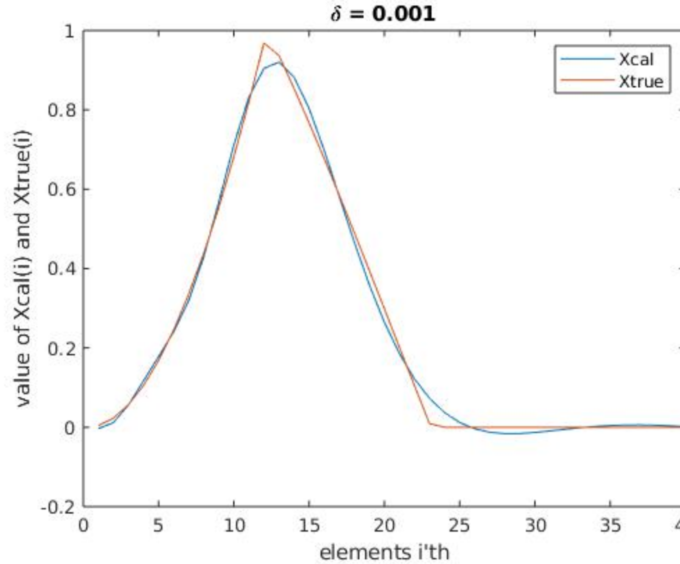
$$x_\delta = argmin_x(\|Ax - y\|_2^2 + \delta^2\|x\|_2^2)$$

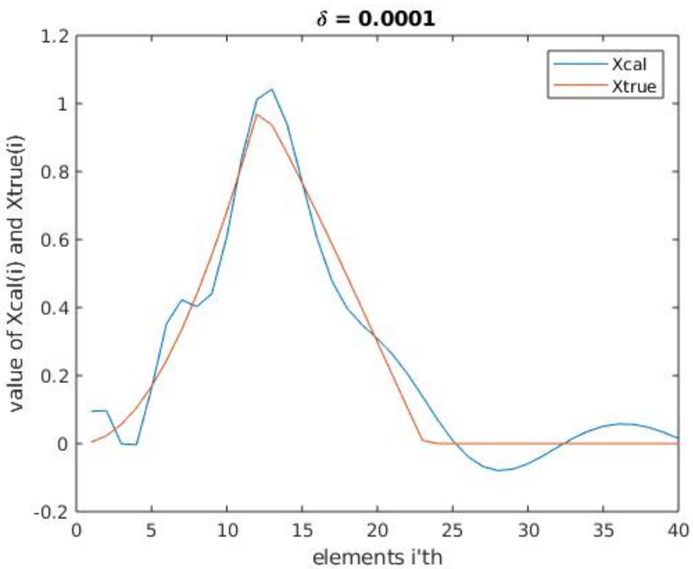
Try different values of the regularization parameter δ, and show the estimates of the solutions. In all the cases considered here, be sure to make suitable plots of solutions or erros, and give comments of your findings.

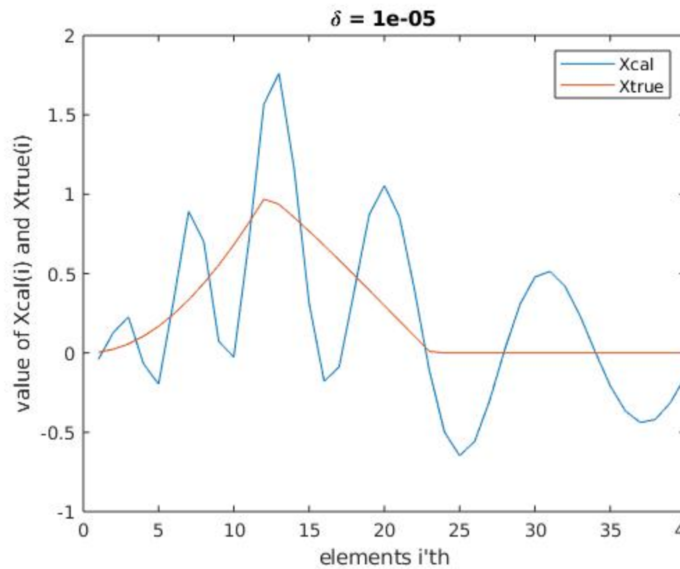
- 利用以下code，對y加入random noise。其中noise是落在[-1e-4,1e-4]間的均勻分佈
- 進行不同δ值的Tikhonov regularization。並進行作圖

```
function Tik()
    N = 40;
    S = log_dis(N);
    Y = getY(S);
    %add random noise
    rad = 1e-4
    noise = (-1)*rad + 2*rad*rand(N,1)
    Ynoise = Y+noise;

    [T W] = legpts(N,[0,5],'GW');
    %use s t to get A
    A = getA(W,S,T);
    %use normal equation to get Xcal
    for pw = 2:5
        figure(pw);
        delta = 10^(-pw);
        I = eye(N);
        Aplus = (A'*A+delta*delta*I);
        B = A'*Ynoise;
        %Aplus*X = B
        Xcal = Aplus \ B;
        Xtrue = getTrueX(T);

        plot([1:N],Xcal);
        hold on
        plot([1:N],Xtrue);
        %title('\delta')
        title(['\delta = ',num2str(delta)])
        xlabel("elements i'th");
        ylabel("value of Xcal(i) and Xtrue(i)");
        legend('Xcal','Xtrue');
        norm(Xcal-Xtrue,2)
    end
end
```









- 上四圖為40維solution Xcal 與 true signal Xtrue 各個維度的值
- δ為1e-2, 1e-3, 1e-4, 1e-5的2-norm error 分別為
  $0.246113880621484 \approx 0.2461$
  $0.138483848796576 \approx 0.1385$
  $0.435398463354816 \approx 0.4354$
  $2.785405979120068 \approx 2.7854$
- δ為1e-3時，有最低的2-norm error