به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

BSS

گزارش تمرین ۵

سالار صفردوست
۸۱۰۱۹۹۴۵۰
۱۴۰۲/۰۲/۱۳

الف

```
%% Question 1


N0 = 3;
s = SubSel(D,x,N0);

function s = SubSel(D,x,N0)

    tic
    [M,N] = size(D);
    subsets_index = nchoosek(1:N,N0);

    D_3d = zeros(M,N0,length(subsets_index));
    D_3d_pseudo = zeros(N0,M,length(subsets_index));
    for i = 1:length(subsets_index)
        D_3d(:,:,i) = D(:,subsets_index(i,:));
        D_3d_pseudo(:,:,i) = pinv(D(:,subsets_index(i,:)));
    end

    S = pagemtimes(D_3d_pseudo,x);
    E = vecnorm(squeeze(pagemtimes(D_3d,S) - x));

    s = zeros(N,1);
    [~ , i] = min(E);
    s(subsets_index(i,:)) = S(:,1,i);
    toc

end
```

```
Elapsed time is 0.835078 seconds.
```

بله، در این قسمت جوابی که به دست می‌آوریم مطابق N_0 می‌باشد که به عنوان ورودی به تابع داده‌ایم، درکلی‌ترین حالت در صورت ندانستن N_0 نیاز است که تمامی حالات انتخاب ۱ از N، انتخاب ۲ از N، ... و انتخاب M از N را در نظر گرفته، میزان خطای هر کدام را به دست آورده و در انتها کمترین آن‌ها را انتخاب کنیم که بسیار طولانی می‌باشد.

دیده می‌شود به ازای $N_0 = 3$ نیز حتی مدت زمان صرف شده برای محاسبات نزدیک به ۱ ثانیه است.

```
>> [s(1:15),s(16:30),s(31:45),s(46:60)]

ans =

         0         0         0         0
         0         0         0         0
    5.0000         0         0         0
         0         0         0         0
         0         0         0         0
    7.0000         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0   -3.0000
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
```

ب

در واقع با اینکار فرض اسپارسیتی عملاً بی‌استفاده می‌شود و از N_0 هم استفاده نمی‌شود و s به دست آمده نیز لزوماً اسپارس نخواهد شد.

```
%% Question 2

s = pinv(D)*x;
```

```
>> [s(1:15),s(16:30),s(31:45),s(46:60)]

ans =

   -0.8779    1.0944    0.3394   -0.5883
   -0.2642   -0.1561   -0.8016   -0.0615
    1.5206   -0.2717   -0.4127    0.6042
    0.5541    0.7519    0.3028   -1.0018
   -0.0881    0.1938    1.0453   -0.3153
    1.7056    0.0893   -0.7418   -0.3747
   -0.4224    0.6346   -0.4494    0.1485
   -0.6219   -1.1332    0.5399    0.1670
   -0.1926    0.2557    0.2173   -1.0062
   -0.5498   -0.1637    0.2408    0.7273
   -0.9494    0.8393   -0.2003   -0.1761
    0.2295   -0.1203    0.6208   -0.4949
    0.7217   -0.4728    0.1933    0.1770
   -0.5231    0.3206    0.0180    0.4521
    0.8819    0.1317   -0.0023   -0.4678
```

ج

```
%% Question 3

N0 = 3;
s1 = MP(D,x,N0);

T=0.1;
s2 = MP(D,x,T);
```

```
function s = MP(D,x,N0_or_T)

    tic
    [~,N] = size(D);
    s = zeros(N,1);

    if N0_or_T>1
        N0 = N0_or_T;
        for n = 1:N0
            inner_product = D.' * x;
            [~ , i] = max(abs(inner_product));
            s(i) = inner_product(i);
            x = x-s(i)*(D(:,i));
        end
    else
        T = N0_or_T;
        E = 1;
        norm_x1 = norm(x);
        while (T < E)
            inner_product = D.' * x;
            [~ , i] = max(abs(inner_product));
            s(i) = inner_product(i);
            x = x-s(i)*(D(:,i));
            E = norm(x)/norm_x1;
        end
    end
    toc

end
```

```
Elapsed time is 0.000998 seconds.
Elapsed time is 0.001812 seconds.
```

```
>> [s1(1:15),s1(16:30),s1(31:45),s1(46:60)]

ans =

         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
   11.1791         0         0         0
         0         0         0         0
         0         0         0         0
   -2.0532         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0   -3.7558
```

```
>> [s2(1:15),s2(16:30),s2(31:45),s2(46:60)]

ans =

         0    1.2263         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
   11.1791         0         0         0
         0         0         0         0
         0         0         0         0
   -2.0532         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0   -1.8857         0   -3.7558
```

به دو شیوه می‌توان این روش را به کار برد، در روش اوّل N_0 مشخص است و تعداد تکرار داخل حلقه توسط آن تعیین می‌شود، در روش دوم N_0 مشخص نیست و حلقه تا جایی ادامه پیدا می‌کند که مقدار خطای x به دست آمده از زیرمجموعه‌ی D نسبت به x اصلی از مقدار $Threshold$ کمتر شود.

د

```
%% Question 4

N0 = 3;
s1 = OMP(D,x,N0);

T = 0.1;
s2 = OMP(D,x,T);

function s = OMP(D,x,N0_or_T)

    tic
    [~,N] = size(D);
    s = zeros(N,1);

    if N0_or_T>1
        N0 = N0_or_T;
        for n = 1:N0

            if(mod(n,2)==1)
                inner_product = D.' * x;
                [~ , i] = max(abs(inner_product));
                s(i) = inner_product(i);
                xr = x-s(i)*(D(:,i));
                j = i;
            else
                inner_product = D.' * xr;
                [~ , i] = max(abs(inner_product));
                s([i j]) = pinv(D(:,[i j]))*x;
                x = x-D(:,[i j])*s([i j]);
            end

        end
    else
        T = N0_or_T;
        E = 1;
        n = 1;
        norm_x1 = norm(x);
        while (T < E)
            if(mod(n,2)==1)
                inner_product = D.' * x;

                [~ , i] = max(abs(inner_product));
                s(i) = inner_product(i);
                xr = x-s(i)*(D(:,i));
                j = i;
                E = norm(xr)/norm_x1;
            else
                inner_product = D.' * xr;
                [~ , i] = max(abs(inner_product));
                s([i j]) = pinv(D(:,[i j]))*x;
                x = x-D(:,[i j])*s([i j]);
                E = norm(x)/norm_x1;
            end
            n = n+1;
        end
    end
    toc

end
```

```
  Elapsed time is 0.000575 seconds.
  Elapsed time is 0.000124 seconds.

>> [s1(1:15),s1(16:30),s1(31:45),s1(46:60)]

ans =

         0    2.1553         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
   10.8544         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0   -3.7837

>> [s2(1:15),s2(16:30),s2(31:45),s2(46:60)]

ans =

         0    2.2376         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
   -1.1657         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
    1.7141         0         0   -3.7837
```

به دو شیوه می‌توان این روش را به کار برد، در روش اوّل N_0 مشخص است و تعداد تکرار داخل حلقه توسط آن تعیین می‌شود، در روش دوم N_0 مشخص نیست و حلقه تا جایی ادامه پیدا می‌کند که مقدار خطای x به دست آمده از زیرمجموعه‌ی D نسبت به x اصلی از مقدار $Threshold$ کمتر شود.

```
%% Question 5

tic
D_star = [D,-D];
s_star = linprog( ones(2*size(D,2),1) , [] , [] , D_star ...
                , x , zeros(2*size(D,2),1) , []);
s = s_star(1:size(D,2))-s_star(size(D,2)+1:end);
toc
```

```
Elapsed time is 0.014966 seconds.
>> [s(1:15),s(16:30),s(31:45),s(46:60)]

ans =

         0         0         0         0
         0         0         0         0
    5.0000         0         0         0
         0         0         0         0
         0         0         0         0
    7.0000         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0   -3.0000
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
```

خیر، در این روش نیاز به دانستن N_0 وجود ندارد، چرا که به طور کلی روش بر اساس مینیمم کردن نرم یک بردار s عمل می‌کند و در صورت مسئله محدودیتی برای آنکه تعدادی از مقادیر s صفر باشد وجود ندارد.

۹

```
%% Question 7

iterations = 50;
s = IRLS(D,x,iterations);
```

```
function s = IRLS(D,x,iterations)

    tic
    [~,N] = size(D);
    s = zeros(N,1);

    t = 1e-6;
    w = (rand(1,N));
    for i = 1:iterations
        y = pinv(D*(diag(w.^-0.5)))*x;
        s = (diag(w.^-0.5))*y;
        cond = abs(s)>t;
        w(cond) = 1./abs(s(cond));
        w(~cond) = 1/t;
    end
    s(~cond) = 0;
    toc

end
```

```
Elapsed time is 0.006207 seconds.
```

```
ans =

         0         0         0         0
         0         0         0         0
    5.0000         0         0         0
         0         0         0         0
         0         0         0         0
    7.0000         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0   -3.0000
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
```

خیر، در این روش نیاز به دانستن N_0 وجود ندارد، چرا که به طور کلی روش بر اساس مینیمم کردن نرم یک بردار s عمل می‌کند و در صورت مسئله محدودیتی برای آنکه تعدادی از مقادیر s صفر باشد وجود ندارد.

ی

روش $subset\ selection$ دقیق ولی بسیار زمانبر است.

روش‌های MP و OMP سریع ولی با دقت پایین هستند(دقت روش OMP بیشتر از MP می‌باشد.)، دقت این دو روش را می‌توان از روی مقدار $\frac{|x-Ds|}{|x|}$ مقایسه کرد.

روش‌های BP و $IRLS$ روش‌هایی با دقت بالا و سرعت معمولی هستند که هر دوی آن‌ها مناسب برای استفاده در استخراج منابع می‌باشند و روش $IRLS$ البته سریعتر از BP به نتیجه رسیده است.

در نتیجه روش آخر بهترین روش برای تشخیص منابع است.