به نام خدا

Blind Source Separation (BSS)

تکلیف شماره

5

محمدرضا آرانی

810100511

دانشگاه تهران

1402/02/13

جدول محتویات

بخش اول:

تمرین سری پنجم (موعد تحویل ۴ شنبه ۱۳ اردیبشهت ساعت ۵ بعد از ظهر)

در این تمرین می خواهیم روش های مختلف بازیابی سیگنال تُنک (sparse) را پیاده سازی کنیم. در فایل hw5.mat ماتریس دیکشنری D با ابعاد 60×10 و بردار مشاهدات x با ابعاد 1×60 قرار داده شده است. در واقع بردار مشاهدات از رابطه ی $x = D\,s$ بدون وجود هیچ گونه نویزی تولید شده است.

قسمت-1

الف) با فرض این که بدانیم sparsity level برابر 3 است ($N_0 = 3$) و جواب مساله یکتاست، با استفاده از روش subset selection بردار اسپارس s را بیابید. درایه های غیر صفر s و مدت زمانی که طول کشید تا s را پیدا کنید گزارش کنید. آیا دانستن N_0 کمکی به حل این قسمت می کند؟

```
% Subset Selection:
N0 =3;


[Row_D,Col_D] = size(D);


S_hat = cell(Col_D,Col_D) ;
Err   = 1e5*ones(Col_D,Col_D,Col_D)  ;
minErr = Err(1,1,1);
tic;
% Sapesity Level Determines the depth of for loops! --> in this
for i=1:Col_D-N0+1
    for j=i+1:Col_D-N0+2
        for k = j+1:Col_D-N0+3


        s_hat = pinv([D(:,i),D(:,j),D(:,k)])*x  ;
        S_hat{i,j} = s_hat;
        Err(i,j,k) = norm(x-[D(:,i),D(:,j),D(:,k)]*s_hat,2) ; % Calculate the error using
norm2
```

```
        if(Err(i,j,k)<minErr)% Choose Best Columns:
            best_i = i;
            best_j = j;
            best_k = k;
            best_s_hat = s_hat;
            best_D_sub = [D(:,i),D(:,j),D(:,k)];
            minErr = Err(i,j,k);
        end


        end
    end
end
% Choose Best Columns:
delta_t_subset = toc;
disp("Consumed Time: "+ delta_t_subset+"(s)")
```

دانستن N کمک می‌کند به دنبال N ساب ست هایی باشیم که طول برابر N دارند و در غیر اینصورت با شروع از طول 1 و ختم به طول برابر طول بردار s به بهترین جواب برسیم.

با داشتن این تعداد N به میزان خطای

```
disp("Min Err: "+minErr)
```

```
Min Err: 6.7056e-15
```

مقادیر مرتبط با اندیس‌های انتخاب شده‌ی در s برابر:

```
disp([best_i , best_j , best_k]')
```

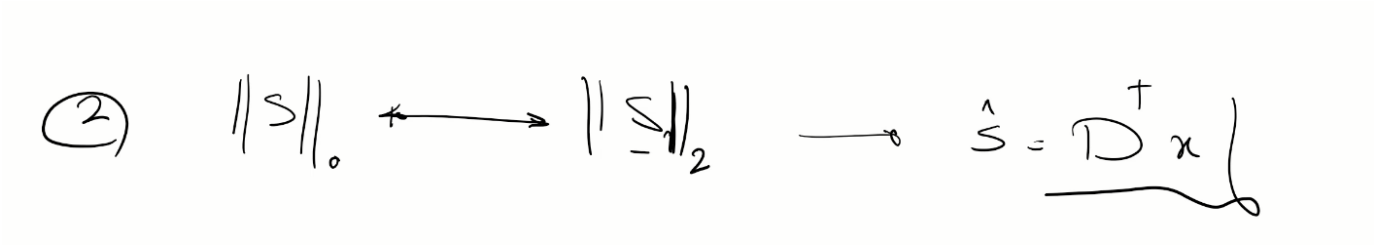
```
     3
     6
    54
```

قسمت-2

ب) قسمت الف را با فرض این که به جای نرم صفر، نرم دو را کمینه کنیم تکرار کنید. آیا دانستن N_0 کمکی به حل این قسمت می کند؟

با تغییر رویکرد از روی نرم 0 به نرم 2، پاسخ یکتا خواهیم داشت که از طریق زیر به دست می‌آید:

4



شکل 1

با استفاده از این راه به جواب‌های زیر می‌رسیم:

sorted_indices =

6

3

23

16

دانستن N در این روش کمکی به حل مسئله نمی‌کند و تغییری در زمان اجرای الگوریتم ندارد.

قسمت-3

ج) قسمت الف را با استفاده از روش Matching Pursuit (MP) تکرار کنید. آیا دانستن N_0 کمکی به حل این قسمت می کند؟ یک بار دیگر بدون این که مقدار N_0 را دانسته در نظر بگیرید، این قسمت را تکرار کنید.

```
disp("Elapsed Time (MP): "+ delta_t_MP+"(s)");
Elapsed Time (MP): 0.0037923(s)
disp(" Best Indices Are ")
 Best Indices Are
disp(Chosen_IDX)
     6     3    55
```

در این روش زمان مصرف شده، و اندیس‌های انتخاب شده به طور تقریبی با دیگر روش‌ها برابراند. دانستن مقدار N0 به صورت شایانی در این روش کمک خواهد کرد چراکه الگوریتم از اوردر O(N0) خواهد بود ولی بدون دانستن آن از اوردر O(Cold_D) که تعداد ستون‌های D است خواهد بود.

قسمت 4-

در این قسمت، به صورت مشابهی با MP عمل شده، تنها تفاوت در انتخاب ضرایب و به‌روزرسانی‌آنها در هنگام انتخاب ستون بعدی می‌باشد.

```
xr = x;
Chosen_IDX_3 = inf+zeros(1,Col_D);
thresh = 1e-6;
tic;
B = D;
for  i=1:N0

    B = D;
    if(i>1)
    B(:,Chosen_IDX_2(1:i-1)) = [];
    end
    Corr_matrix = repmat(xr,1,Col_D-i+1).*B;
    Corr_sum = sum(Corr_matrix,1); % Summation for each Column -->> a Row Vector
    [Value,Idx] = max(Corr_sum); % Choosing Best Fitted
    if(sum(Idx>Chosen_IDX_2)>0)
        Idx  = Idx + sum(Idx>Chosen_IDX_3) ;
    end
    Chosen_IDX_3(i) =  Idx;
    xr = x - Value*D(:,Idx); % xr = x  - <x,di>di
    % Update Coefficients:
    if(i>1)
        D_sub_omp = D(:,Chosen_IDX_3(1:i)) ;
        xr = xr - D(:,Chosen_IDX_3(1:i))*(pinv(D_sub_omp)*xr);
    end
    % Stop Criteria:
%     if(norm(x - D(:,Chosen_IDX_3(1:i)) ,2)<thresh )
%         break;
```

```
%     end


end
delta_t_OMP = toc;
```

نتایج به دست آمده از این روش به صورت زیر است:

```
disp("Elapsed Time (OMP): "+ delta_t_OMP+"(s)");
Elapsed Time (OMP): 0.0056359(s)
disp(" Best Indices Are ")
 Best Indices Are
disp(Chosen_IDX_3(Chosen_IDX_3<Col_D+5))
     6     3    22
```

در این الگوریتم هم دانستن N0 مانند MP، تاثیر گذار است.

قسمت-5

ه) قسمت الف را با استفاده از روش Basis Pursuit (BP) تکرار کنید. آیا دانستن N_0 برای حل این قسمت ضروری است؟

```
% - Solution of Hw5- part e
load('hw5.mat')
[M,N]=size(D);
N0=3;
% Linear Programming
f=ones(2*N,1);
Aeq=[D -D];
beq=x;
lb=zeros(2*N,1);
tic;
yhat = linprog(f,[],[],Aeq,beq,lb,[]); % Linear-Programming: given Cost function: (f) +
Equality Constraints with respect to Variables
Optimal solution found.
% in Matrix Form


splus=   yhat(1:N);
sminus=  yhat(N+1:end);
```

```
sBP=     splus-sminus;



posBP=find(abs(sBP)>0.01)'; % Choose Nonzero Elements
delta_t_LinP = toc;
disp("Elapsed Time (LP): "+ delta_t_LinP+"(s)");
Elapsed Time (LP): 0.074427(s)
disp('BP:')
BP:
[posBP;sBP(posBP)']

ans = 2×3
    3.0000    6.0000   54.0000
    5.0000    7.0000   -3.0000
```

قسمت-6

و) قسمت الف را با استفاده از روش Iteratively Reweighted Least Square (IRLS) تکرار کنید. آیا دانستن N_0 برای حل این قسمت ضروری است؟

```
load("hw5.mat");
w=ones(N,1);
MAXITER=100;
threshold_IRLS = 1e-3;
tic;
for i = 1:MAXITER
    W=diag(w);
    y=pinv(D*(W^-1))*x;
    S_irls=y./sqrt(w);
    High_Val = 1e+10;
    Low_Val  = 1e-10;
    for n=1:N
        if abs(S_irls(n))<threshold_IRLS
            w(n)=High_Val;   % In case of Low Value of s
        elseif abs(S_irls(n))>1e6
            w(n)=Low_Val;   % In case of High Vlaue of s
        else
            w(n)=1./abs(S_irls(n));  % Regularly
        end
```

```
    end
end
delta_t_IRLS = toc;


disp("Elapsed Time (IRLS): "+ delta_t_IRLS+"(s)");
Elapsed Time (IRLS): 0.092848(s)

IRLS_Index=find(abs(S_irls)>0.1)';
disp('IRLS Solution:')
IRLS Solution:
disp(IRLS_Index)
     3     6    54
% The results are the same as the BP and the MP!
disp("S value according to chosen Indexes: "  + S_irls(IRLS_Index) )
    "S value according to chosen Indexes: 2.924"
    "S value according to chosen Indexes: 3.6593"
    "S value according to chosen Indexes: -2.0801"
```

در این روش نیز از دانستن N0 بهره می‌بریم به این صورت که اوردر این الگوریتم با داشتن یک حلقه‌ی for درونی به تعداد N0 است پس O(cN0) داریم که این عدد ثابت c در اوردر لحاظ نمی‌شود.

نتایج این روش نیز مانند روش BP و روش MP و روش Subset است.

قسمت-7

ی) به نظر شما کدام یک از روش های بالا بهترین است؟ چه معیارهایی را در نظر گرفتید؟ توضیح دهید.

معیارهای مورد نظر برای مقایسه شامل:

- آسانی پیاده‌سازی
- دقت بالا
- زمان کم

- میزان دانش اولیه مورد نیاز

می‌باشد. به این ترتیب نمی‌توان هیچ یک را برتر از دیگری دانست ولی در هر مورد می‌توان مقایسه انجام داد. برای مثال در مقایسه‌ی زمان انجام،

```
Elapsed Time (IRLS): 0.092848(s)
Elapsed Time (MP): 0.0037923(s)
Elapsed Time (OMP): 0.0056359(s)
Elapsed Time (LP): 0.074427(s)
```

می‌توان دید که MP در این مورد به خوبی عمل کرده و پس از آن OMP در رده‌ی دوم است. پیاده‌سازی MP و OMP نیز به نسبت ساده است. در این بین IRLS بیشترین زمان اجرا را دارد ولی برای N های بزرگ قطعا بهتر عمل می‌کند. مقایسه برای تعداد N داده شده انجام شده است.

پایان