CO 602/CM 740: Fundamentals of Optimization
Problem Set 7

Instructor: Henry Wolkowicz

Fall 2016.
Handed out: Sunday 2016-Nov-20.
Due: Friday 2016-Nov-25 by 3PM.*

Contents

1 Derivatives **1**

2 Matrix Rounding **3**

1 Derivatives

1. Let $A, B \in \mathbb{S}^n$ be real symmetric matrices of order n. Let $C \in \mathcal{M}^n$ be a real square matrix order n. Consider $f : \mathcal{M}^n \to \mathbb{R}$ a real valued function on square matrices order n defined by

$$f(X) = \operatorname{trace}(AXBX^T + CX^T).^1$$

   (a) Derive the gradient of f, i.e., the linear mapping that provides the best linear model in the Taylor expansion

$$f(X + \Delta X) = f(X) + \langle \nabla f(X), \Delta X \rangle + o(\|\Delta X\|).$$

   Solution 1.1. *The solution to Item 1a can be found by expansion, i.e.*

$$\begin{aligned} f(X + \Delta X) &= \operatorname{trace}(A(X + \Delta X)B(X + \Delta X)^T + C(X + \Delta X)^T) \\ &= f(X) + \operatorname{trace}\left(A\Delta XBX^T + AXB\Delta X^T + \operatorname{trace} C\Delta X^T\right) + \operatorname{trace} A\Delta XB\Delta X^T \\ &= f(X) + \langle \nabla f(X), \Delta X \rangle + o(\|\Delta X\|). \end{aligned}$$

   Using the commutativity of trace and symmetry of A, B, we see that

$$\langle \nabla f(X), \Delta X \rangle = \operatorname{trace} 2AXB\Delta X^T = \langle 2AXB + C, \Delta X \rangle.$$

   (b) Derive the Hessian $\nabla^2 f(X) : \mathbb{R}^{m \times n} \to \mathbb{S}^{mn}$, where $\mathbb{S}^{mn}$ is the space of real symmetric $mn \times mn$ matrices, i.e., find the the best quadratic model in the Taylor expansion

$$f(X + \Delta X) = f(X) + \langle \nabla f(X), \Delta X \rangle + \frac{1}{2}\langle \operatorname{vec}(\Delta X), \nabla^2 f(X) \operatorname{vec}(\Delta X) \rangle + o(\|\Delta X\|^2),$$

   where the vec function changes X to a vector using its rows. Therefore $\|X\|^2 = \|\operatorname{vec}(X)\|^2$.[2]

*The late penalty of 10% is postponed till Monday 10AM.

[1] Recall that the Kronecker product is the block matrix $B \otimes A = (B_{ij}A)$. We let $\operatorname{vec}(X)$ denote the vector formed columnwise from the matrix X. Then $\operatorname{vec}(AXB) = (B \otimes A)\operatorname{vec}(X)$ and $\operatorname{trace}(AXBX^T) = \operatorname{vec}(X)^T(B \otimes A)\operatorname{vec}(X)$.

[2] This is called the *Frobenius norm* of X

Solution 1.2. *The solution to Item 1b can be found similarly.*

$$\begin{aligned} f(X+\Delta X) &= \operatorname{trace}(A(X+\Delta X)B(X+\Delta X)^T + C(X+\Delta X)^T) \\ &= f(X) + \operatorname{trace}\left(A\Delta X B X^T + AXB\Delta X^T + \operatorname{trace} C\Delta X^T\right) + \operatorname{trace} A\Delta X B\Delta X^T \\ &= f(X) + \langle \nabla f(X), \Delta X\rangle + \operatorname{trace} A\Delta X B\Delta X^T. \end{aligned}$$

Therefore

$$\frac{1}{2}\langle \Delta X, \nabla^2 f(X)\Delta X\rangle = \operatorname{trace} A\Delta X B\Delta X^T = \frac{1}{2}\operatorname{vec}(\Delta X)^T 2(B\otimes A)\operatorname{vec}(\Delta X),$$

where recall that $B \otimes A$ denotes the Kronecker product.

2. Consider the instance with the matrices (also in file *assign7F16ABC.mat*):

$$A = \begin{bmatrix} 19.0000 & -7.2200 & 7.3800 & 4.3600 & 4.5400 \\ -7.2200 & 14.8800 & 8.6600 & -6.9400 & 0.6600 \\ 7.3800 & 8.6600 & 24.2800 & -4.8000 & 3.5200 \\ 4.3600 & -6.9400 & -4.8000 & 12.3400 & 2.3800 \\ 4.5400 & 0.6600 & 3.5200 & 2.3800 & 7.2400 \end{bmatrix},$$

$$B = \begin{bmatrix} 14.9200 & -5.3800 & -1.8200 & -0.7600 & 3.5800 \\ -5.3800 & 11.4000 & -3.1000 & 1.7400 & -2.1000 \\ -1.8200 & -3.1000 & 31.3200 & -11.7800 & -10.1200 \\ -0.7600 & 1.7400 & -11.7800 & 20.6600 & 5.7200 \\ 3.5800 & -2.1000 & -10.1200 & 5.7200 & 15.2200 \end{bmatrix},$$

$$C = \begin{bmatrix} 1.9600 & 1.2100 & 1.2700 & 1.5500 & -0.4400 \\ 1.2100 & -0.9400 & 1.7200 & 1.4200 & -0.5800 \\ 1.2700 & 1.7200 & -3.0600 & -0.3000 & 1.1200 \\ 1.5500 & 1.4200 & -0.3000 & 1.1000 & -2.8500 \\ -0.4400 & -0.5800 & 1.1200 & -2.8500 & 0.2200 \end{bmatrix}.$$

   (a) Use the MATLAB quadprog command to solve the minimization problem to find the optimal value and the optimal X. Verify optimality using first and second order optimality conditions. Is it a global minimum? Why? How many iterations did the algorithm take? Can you explain why?

   (b) Add the constraint that $0 \le X_{ij} \le 1$ for all ij.

Solution 1.3. *A sample MATLAB code follows. Minor changes generate a new problem and/or solve with the interval constraints.*

```
> clear
> warning off
> n=5;
> n2=n^2;
> A=randn(n);
> A=A*A'+eye(n);
> A=round(1e2*A);
> A=(1e-2*A);
> A=A+A';
> B=randn(n);
> B=B*B'+eye(n);
> B=B+B';
> B=round(1e2*B);
```

```
>  B=(1e-2*B);
>  B=B+B';
>  C=randn(n);
>  C=round(1e2*C);
>  C=(1e-2*C);
>  C=C+C';
>  load assign7F16ABC   % for actual assign problem data
>  c=C(:);
>  D=kron(B,A);
>  %[x,fval,flag,output]=quadprog(2*D,c);
>  [x,fval,flag,output]=quadprog(2*D,c,[],[],[],[],zeros(n2),ones(n2));
>  X=reshape(x,n,n);
>  fprintf('optimality value %g \n',trace(A*X*B*X'+C*X'))
>  fprintf('optimality/stationarity value %g \n',norm(2*A*X*B+C))
>  fprintf('min eigvalue of Hessian %g \n',min(eig(D)))
```

BONUS Confirm the optimal value, solution, and optimal gradient value using the MATLAB function fminunc. But use the options that allow for you to provide the gradient and the Hessian at each iteration.

3. Consider the function $f : \mathbb{S}^n_{++} \to \mathbb{R}$ given by $f(X) = \log\det(X)$, where $\mathbb{S}^n_{++}$ denotes the cone of positive definite matrices (symmetric and all eigenvalues are positive). As above in Item 1, find the gradient and Hessian. (Hint: For the Hessian, recall that $XX^{-1} = I$.)

2 Matrix Rounding

Consider the matrix

$$A = \begin{bmatrix} 5.3000 & 9.5600 & 6.8500 & 1.3300 & 1.5600 \\ 8.6000 & 0.6700 & 2.0800 & 1.0200 & 0.9000 \\ 6.7800 & 5.4200 & 6.0800 & 9.5900 & 4.5400 \\ 8.0600 & 2.8200 & 3.2600 & 1.5300 & 6.6900 \\ 5.3100 & 4.8100 & 8.8100 & 1.5300 & 8.3100 \end{bmatrix}.$$

(Available as in file *Aforrounding.mat*.) The *consistent matrix rounded* problem rounds the elements of A to the next largest or smallest integer so that the rounded row and column sums stay consistent with the row and column sums of the rounded matrix.

Formulate this problem as a *max flow network* problem. Solve the problem to find a consistent flow.

Solution 2.1. *A possible MATLAB file for the solution follows:*

```
%% Generate a random matrix with two decimals accuracy
%% Model the problem 'consistently' so row and column sums are consistent.
>  warning off
>  m=5;  % rows
>  n=6;  % cols
>  R=100*rand(m,n);
>  R=R*10^3;
>  R=round(R);
>  R=R*10^(-3);
>  rows=sum(R,2);
>  cols=sum(R,1)';
>  A=zeros(1+m+n+1,1+m+m*n+n);  % node-arc incidence matrix
>  nf=size(A,2);
```

```
>  u=zeros(nf,1);  % upper bounds
>  l=zeros(nf,1);  % lower bounds
>  %u(1)=sum(sum(R))+1e4;
>  u(1)=sum(sum(R))+1;
>  b=zeros(1+m+n+1,1);  % RHS
>  A(1,1)=-1;    % arc for max.
>  A(end,1)=1;
>  arc=1;
>  feas=zeros(nf,1);
>  feas(arc)=sum(sum(R));
>  %% start with node 1 --- left hand node - rows
>  for ii=1:m,
>   arc=arc+1;
>          feas(arc)=rows(ii);
>   A(1,arc)=1;   % leaving
>   A(1+ii,arc)=-1;   % arriving
>   u(arc)=ceil(rows(ii));
>   l(arc)=floor(rows(ii));
>  end
>  %% continue with intermediate nodes
>  arcfeas=arc;
>  for ii=1:m,
>   for jj=1:n,
>   arc=arc+1;
>                  feas(arc)=R(ii,jj);
>   A(1+ii,arc)=1;   % leaving node ii
>   A(1+m+jj,arc)=-1;   % arriving node m+jj
>           u(arc)=ceil(R(ii,jj));
>           l(arc)=floor(R(ii,jj));
>   end
>  end
>  %% finish with last node m+n+2 --- right hand node
>  for jj=1:n,
>   arc=arc+1;
>          feas(arc)=cols(jj);
>   A(1+m+jj,arc)=1;   % leaving
>   A(end,arc)=-1;   % arriving
>   u(arc)=ceil(cols(jj));
>   l(arc)=floor(cols(jj));
>  end
>  Abar=A(1:end-1,:);
>  bbar=b(1:end-1);
>  e0=zeros(nf,1);
>  e0(1)=1;
>  %%cvx_clear
>  %%cvx_begin
>  %%variable f(nf,1)    % flows
>  %%variable bt
>  %%maximize f(1)
>  %%subject to
>  %%      Abar*f == bbar;
```

```
>  %%      f <= u;
>  %%      %f >= l;
>  %%cvx_end
>  %u=u+1;
>  %l=0*l+1
>  options=optimoptions(@linprog,'Algorithm','dual-simplex')
>  [f,fval,exitflag,output,lambda] = linprog(-e0,[],[],Abar,bbar,l,u,options);
>  %% recover optimal f
>  arc=arcfeas;
>  Ropt=zeros(m,n);
>  for ii=1:m,
>   for jj=1:n,
>   arc=arc+1;
>                  Ropt(ii,jj)=f(arc);
>   end
>  end
```