Mikhail Gaerlan
7 March 2018
MAT 226B Freund

Problem 1

(a)

$$\begin{aligned} Ab &= \begin{bmatrix} 0 & 1 \\ I & a \end{bmatrix} e_1 = e_2 \\ A^2 b &= Ae_2 = e_3 \\ &\vdots \\ A^{k-1} b &= e_k \end{aligned}$$

Thus $K_k(A,b) = \operatorname{span}\left\{b, Ab, A^2 b, \ldots, A^{k-1} b\right\} = \operatorname{span}\{e_1, \ldots, e_k\} = \mathbb{R}^k$. If $d(A,b) < n$, then $A^k b \in K_k(A,b)$ for $k > n$. However, $A^k b = e_k \notin \operatorname{span}\{e_1, \ldots, e_{k-1}\} = K_d(A,b)$. Thus $d(A,b) = n$ since $d(A,b) \leq n$.

(b)

The number of iterations the MR methods needs is $d(A, r_0) = d(A, b - Ax_0) = d(A,b) = n$.

(c)

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 \\ I & a \end{bmatrix} \qquad A^T = \begin{bmatrix} 0 & I \\ 1 & a^T \end{bmatrix} \\ A^T A &= \begin{bmatrix} 0 & I \\ 1 & a^T \end{bmatrix} \begin{bmatrix} 0 & 1 \\ I & a \end{bmatrix} \\ &= \begin{bmatrix} I & a \\ a^T & 1 + a^T a \end{bmatrix} \\ &= I + \begin{bmatrix} 0 & a \\ a^T & a^T a \end{bmatrix} \\ &= I + A' \quad \text{where } A' = \begin{bmatrix} 0 & a \\ a^T & a^T a \end{bmatrix} \end{aligned}$$

Since $\operatorname{rank}(A') = 2$, then A' has at most 2 distinct nonzero eigenvalues and 0 as an eigenvalue with multiplicity $n-2$. Thus A will have at most 3 distinct eigenvalues by the following lemma.

Lemma 0.1. *If A' has an eigenvalue λ, then $\lambda + 1$ is an eigenvalue of A.*

Proof. Let λ be an eigenvalue of A' with eigenvector v then

$$\begin{aligned} A'v &= \lambda v \\ A'v + Iv &= \lambda v + v \\ \left(A' + I\right) v &= (\lambda + 1)\, v \\ Av &= (\lambda + 1)\, v \end{aligned}$$

which means $\lambda + 1$ is an eigenvalue of A. $\square$

(d)

The CGNE method will converge in at most 3 iterations since there are 3 distinct eigenvalues of A.

Problem 2

(a)

$$\begin{aligned} A' &= M_1^{-1} A M_2^{-2} \\ &= M_1^{-1} (M - E)\, M_2^{-2} \\ &= M_1^{-1} M_1 M_2 M_2^{-2} - M_1^{-1} E M_2^{-2} \\ &= I - M_1^{-1} E M_2^{-2} \end{aligned}$$

Since $\operatorname{rank}(E) = k$ then $\operatorname{rank}\left(M_1^{-1} E M_2^{-1}\right) \leq k$, so there will be at most $k+1$ distinct eigenvalues by the same argument from 1(c).

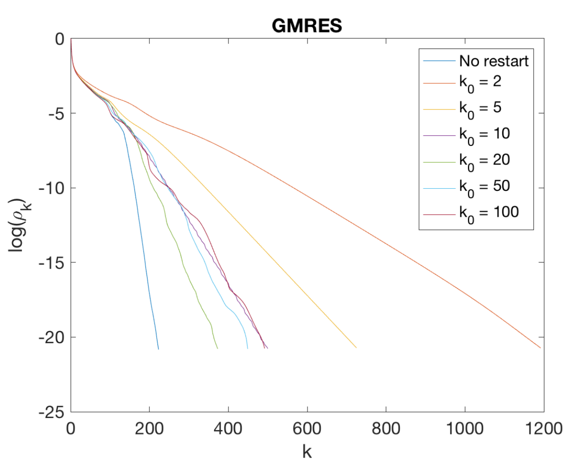
(b)

The CGNE method will converge in at most $r+1$ iterations by the same argument from 1(d).

Problem 3

```
function [x,m,relres] = gmres_default(A,b,k0,tol,x0)
%Solve Ax=b using GMRES
%   Input  - A        matrix
%          - b        right-hand side
%          - k0       restart parameter
%          - tol      tolerance
%          - x0       initial solution
%   Output - x        approximate solution
%          - m        number of matrix-vector products
%          - relres   history of relative residual norms
[x,~,~,iter,relres] = gmres(A,b,k0,tol,length(b),[],[],x0);
m = (iter(1)-1)*k0+iter(2)+iter(1);relres = relres/norm(b-A*x0);
end
```

```
clearvars; clf; load('HW4_Problem3.mat');
n = length(b); k = [n,2,5,10,20,50,100]; m = []; x0 = zeros(n,1);
for k0 = k
    [~,n,relres] = gmres_default(A,b,k0,1e-9,x0);
    m = [m,n]; plot(log(relres)); hold on;
end
xlabel("k"); ylabel("log(\rho_k)"); set(gca,'FontSize',16);
title("GMRES"); legend(["No restart",strcat("k_0 = ",string(k(2:end)))]);
saveas(gcf,"../Figures/homework4_3.png");
matrix2latex(m',"../Tables/homework4_3.tex",'alignment','r')
```



Matrix-vector products	
No restart	223
$k_0 = 2$	1787
$k_0 = 5$	869
$k_0 = 10$	549
$k_0 = 20$	391
$k_0 = 50$	457
$k_0 = 100$	496

Problem 4

(a)

$$
\begin{aligned}
A' &= M_1^{-1} A M_2^{-1} \\
&= D\,(D-F)^{-1}\,(D_0-F-G)\,(D-G)^{-1} \\
&= D\left[(D-F)^{-1}\,((Do-2D)+(D-F)+(D-G))\,(D-G)^{-1}\right] \\
&= D\left[(D-F)^{-1}\,(Do-2D)\,(D-G)^{-1}+(D-G)^{-1}+(D-F)^{-1}\right] \\
&= D\left((D-G)^{-1}+(D-F)^{-1}\left(I+D_1\,(D-G)^{-1}\right)\right)
\end{aligned}
$$

(b)

To multiply a vector $q = Av$, we can follow the steps:

1. Solve $u = (D-G)^{-1}\,v$.
2. Multiply $w = D_1 u$.
3. Add $x = v + w$.
4. Solve $z = (D-F)^{-1}\,x$.
5. Add $p = u + z$.
6. Multiply $q = Dp$.

(c)

The two SAXPYs and the two diagonal matrix multiplications each involve n flops. The upper and lower triangular solves require m multiplications and m substractions for each non-diagonal entry and n divisions for a total of $n + 2m$. So the total number of flops required is $6n + 4m$ flops.

Problem 5

(a)

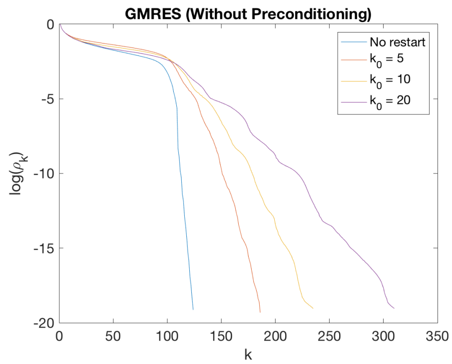
```
function [x,m,relres] = gmres_diag(A,b,k0,tol,x0)
%Solve Ax=b using GMRES with a right diagonal preconditioner
%   Input  - A       matrix
%          - b       right-hand side
%          - k0      restart parameter
%          - tol     tolerance
%          - x0      initial solution
%   Output - x       approximate solution
%          - m       number of matrix-vector products
%          - relres  history of relative residual norms
[x,~,~,iter,relres] = gmres(A,b,k0,tol,length(b),@idens,@diags,x0);
m = (iter(1)-1)*k0+iter(2)+iter(1);relres = relres/norm(b-A*x0);
    function x = idens(b); x = b; end
    function x = diags(b); x = b./diag(A); end
end
```

```
function [x,m,relres] = gmres_ssor(A,b,k0,tol,D,x0)
%Solve A'x'=b' using GMRES with an SSOR-type preconditioner
%   Input  - A       matrix
%          - b       right-hand side
%          - tol     tolerance
%          - k0      restart parameter
%          - D       diagonal matrix
%          - x0      initial solution
%   Output - x       approximate solution
%          - m       number of matrix-vector products
%          - relres  history of relative residual norms
D0 = diag(diag(A)); D1 = D0 - 2*D;
F = -tril(A,-1); G = -triu(A,1);M1 = (D-F)*(D^(-1)); M2 = D-G;
[x,~,~,iter,relres] = gmres(@mult_Ap,M1\b,k0,tol,length(b),[],[],M2*x0);
m = (iter(1)-1)*k0+iter(2)+iter(1);relres = relres/norm(b-A*x0);
    function b = mult_Ap(x)
        u = (D-G)\x;
        l = (D-F)\(x+diag(D1).*u);
        b = diag(D).*(u+l);
    end
end
```

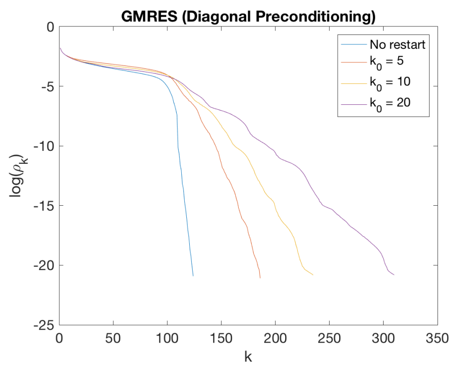
(b)

```
for k = 1:2
    clearvars -except k; load(strcat("HW4_Problem5b_",int2str(k),".mat"));
    n = length(b); x0 = ones(n,1); tol = 1e-8; k0s = [n,5,10,20];
    clf; m = []; for k0 = k0s
        [~,mm,relres] = gmres_default(A,b,k0,tol,x0);
        m = [m,mm]; plot(log(relres)); hold on;
    end
    xlabel("k"); ylabel("log(\rho_k)"); set(gca,'FontSize',16);
    legend(["No restart",strcat("k_0 = ",string(k0s(2:end)))]);
    title("GMRES (Without Preconditioning)");
    saveas(gcf,strcat("../Figures/homework4_5_",int2str(k),"_default.png"));
    matrix2latex(m',strcat("../Tables/homework4_5_",int2str(k),"_default.tex"),'alignment','r')
    clf; m = []; for k0 = k0s
        [~,mm,relres] = gmres_diag(A,b,k0,tol,x0);
        m = [m,mm]; plot(log(relres)); hold on
    end
    xlabel("k"); ylabel("log(\rho_k)"); set(gca,'FontSize',16);
    legend(["No restart",strcat("k_0 = ",string(k0s(2:end)))]);
    title("GMRES (Diagonal Preconditioning)")
    saveas(gcf,strcat("../Figures/homework4_5_",int2str(k),"_diag.png"));
    matrix2latex(m',strcat("../Tables/homework4_5_",int2str(k),"_diag.tex"),'alignment','r')
    clf; m = []; D = diag(diag(A)); for k0 = k0s
        [~,mm,relres] = gmres_ssor(A,b,k0,tol,D,x0);
        m = [m,mm]; plot(log(relres)); hold on
    end
    xlabel("k"); ylabel("log(\rho_k)"); set(gca,'FontSize',16);
    legend(["No restart",strcat("k_0 = ",string(k0s(2:end)))]);
    title("GMRES (SSOR D = D_0)");
    saveas(gcf,strcat("../Figures/homework4_5_",int2str(k),"_ssordiag.png"));
    matrix2latex(m',strcat("../Tables/homework4_5_",int2str(k),"_ssordiag.tex"),'alignment','r')
    clf; m = []; D = 10*speye(n); for k0 = k0s
        [~,mm,relres] = gmres_ssor(A,b,k0,tol,D,x0);
        m = [m,mm]; plot(log(relres)); hold on
    end
    xlabel("k"); ylabel("log(\rho_k)"); set(gca,'FontSize',16);
    legend(["No restart",strcat("k_0 = ",string(k0s(2:end)))]);
    title("GMRES (SSOR D = 10I)");
    saveas(gcf,strcat("../Figures/homework4_5_",int2str(k),"_ssoriden.png"));
    matrix2latex(m',strcat("../Tables/homework4_5_",int2str(k),"_ssoriden.tex"),'alignment','r')
end
```

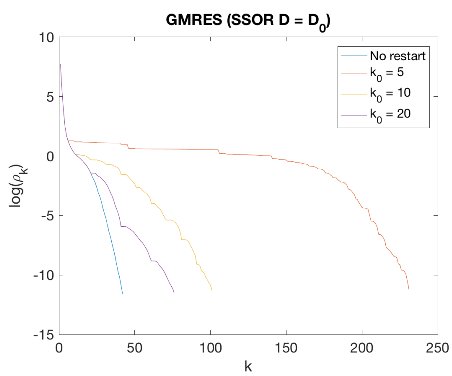
For the first matrix with $n = 50653$, the code produces the following output:



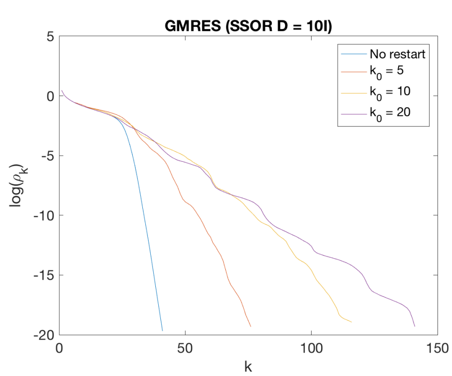
Matrix-vector products	
No restart	124
$k_0 = 2$	222
$k_0 = 5$	258
$k_0 = 10$	325



Matrix-vector products	
No restart	124
$k_0 = 2$	222
$k_0 = 5$	258
$k_0 = 10$	325

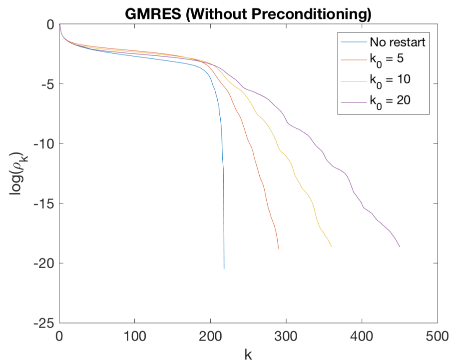


Matrix-vector products	
No restart	42
$k_0 = 2$	276
$k_0 = 5$	110
$k_0 = 10$	79

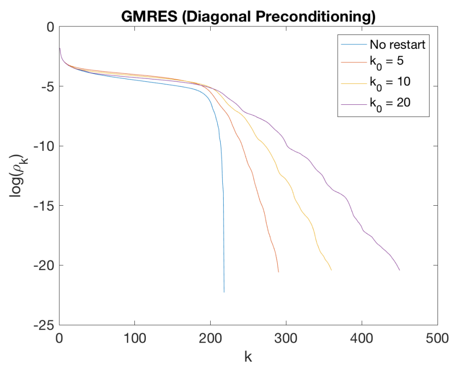


Matrix-vector products	
No restart	41
$k_0 = 2$	90
$k_0 = 5$	127
$k_0 = 10$	147

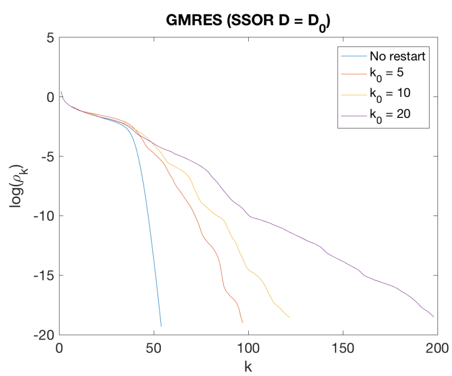
For the first matrix with $n = 389017$, the code produces the following output:



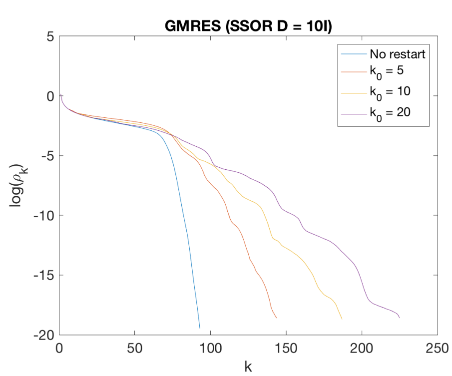
Matrix-vector products	
No restart	218
$k_0 = 2$	347
$k_0 = 5$	395
$k_0 = 10$	472



Matrix-vector products	
No restart	218
$k_0 = 2$	347
$k_0 = 5$	395
$k_0 = 10$	472



Matrix-vector products	
No restart	54
$k_0 = 2$	116
$k_0 = 5$	134
$k_0 = 10$	207



Matrix-vector products	
No restart	93
$k_0 = 2$	172
$k_0 = 5$	205
$k_0 = 10$	236