MATH693a
Dr. Peter Blomgren
HW04
Conjugate Gradient Method

Matteo Polimeno

November 17^{th}, 2018

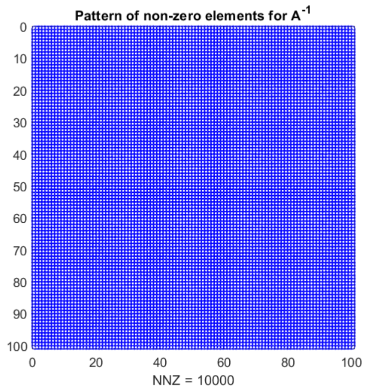
Problem 1

We summarize the most important results from problem 1 in the table below. More information will be extracted from the plots on display.
We run the Conjugate Gradient Algorithm as long as the Euclidean norm of the residual is greater than a tolerance set at 10^{-9}.
For the one,two and three-dimensional problem we run the algorithm until the laptop could handle it.
For the one-dimensional problem $A\mathbf{x} = \mathbf{b}$, we have the following results:

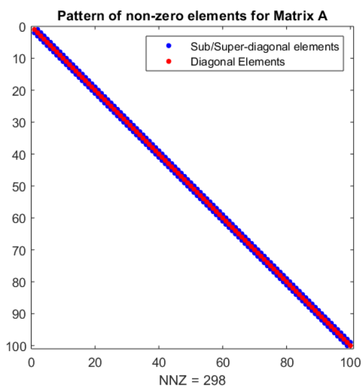
n	Non-zero elements	Elapsed Time	Iterations (sec)	Condition Number
10^2	298	0.0074	50	2.4×10^{-4}
10^3	2998	0.0092	500	2.5×10^{-6}
10^4	29998	0.5970	5000	2.5×10^{-8}
10^5	299998	51.9987	50000	2.5×10^{-10}

We can immediately appreciate that we have convergence in a number of iterations that is exactly $n/2$ for a given value of n and that the condition number decreases by a factor of about 10^{-2} as n increases by a factor of 10.

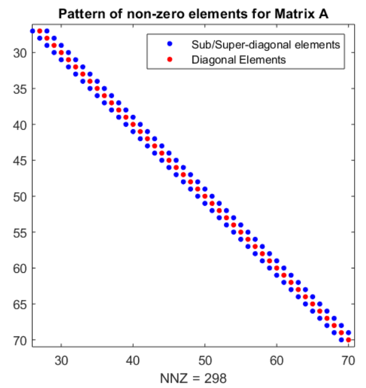
We displayed the plots for the norm of the residual for $10^2 \leq n \leq 10^4$, as well as a plot of the sparse matrix A and its dense inverse for $n = 10^2$.



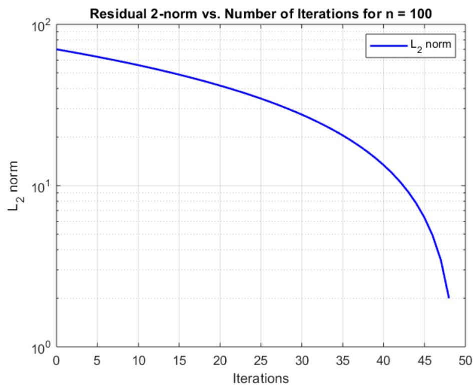
(a)



(b)

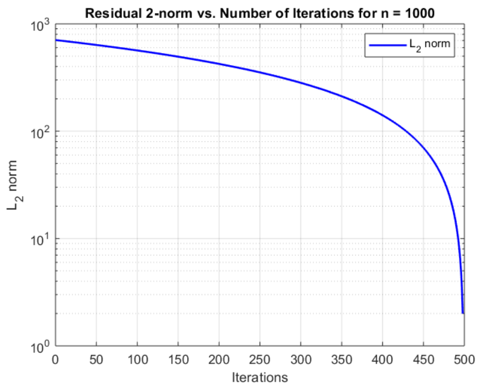


c)

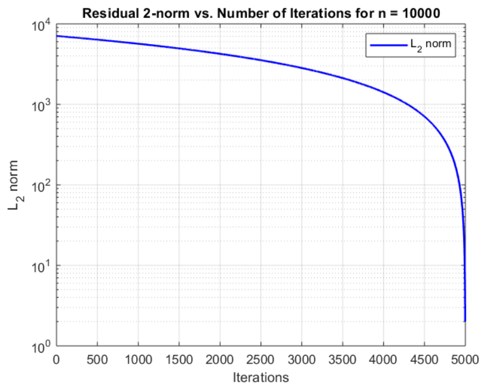


d)

Figure 1: A^{-1} a), A b) and its zoomed-in version c) and the Euclidean Norm of the residual as a function of the iterations d)



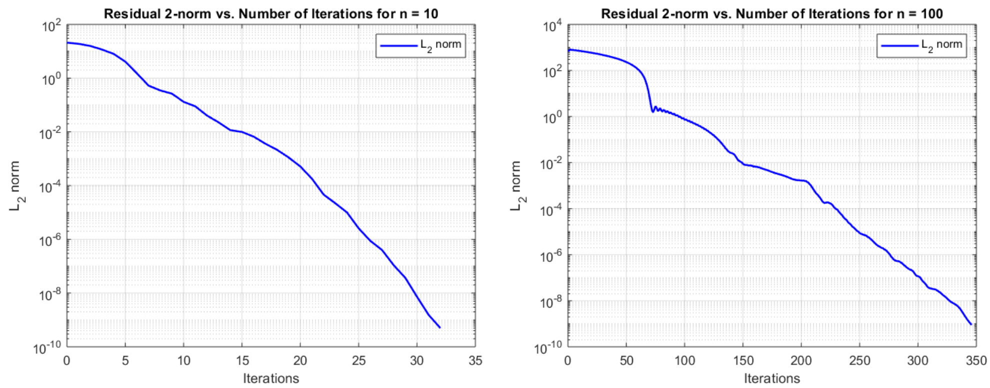
(a)



(b)

Figure 2: Euclidean norm for the given values of n

For the 2-dimensional problem, we have



(a) (b)

Figure 3: Euclidean norm for the given values of n

For the 3-dimensional problem, we have:

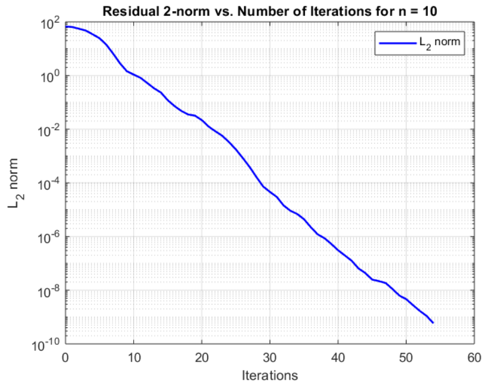


Figure 4: Euclidean norm for the given values of n

Problem 2

For this problem we use the Conjugate Gradient Algorithm to solve the system $A\mathbf{x} = \mathbf{b}$, where A is the Hilbert Matrix. We summarize our result below.

n	Elapsed Time	Iterations (sec)	Condition Number
5	0.0087	6	4.8×10^5
8	0.0051	19	1.5×10^{10}
12	0.0117	35	1.9×10^{16}
20	0.0201	66	2.5×10^{16}

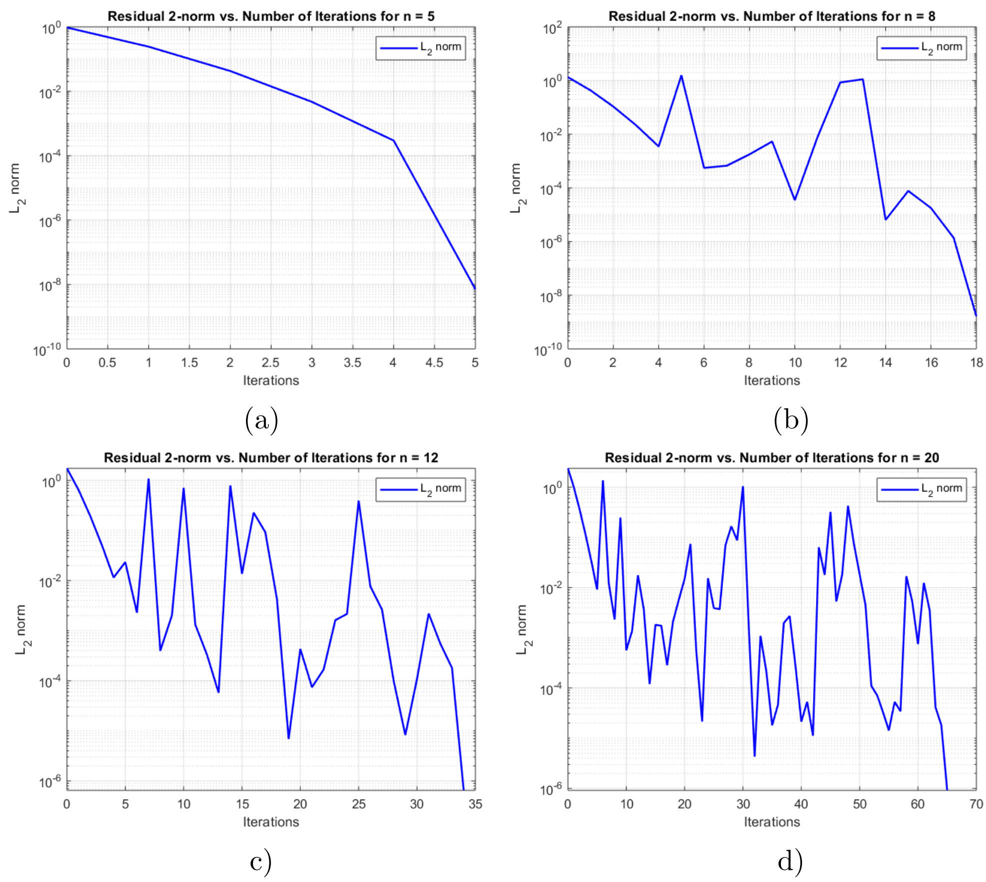


Figure 5: Residual Euclidean Norm as a function of the iterations for the given values of n

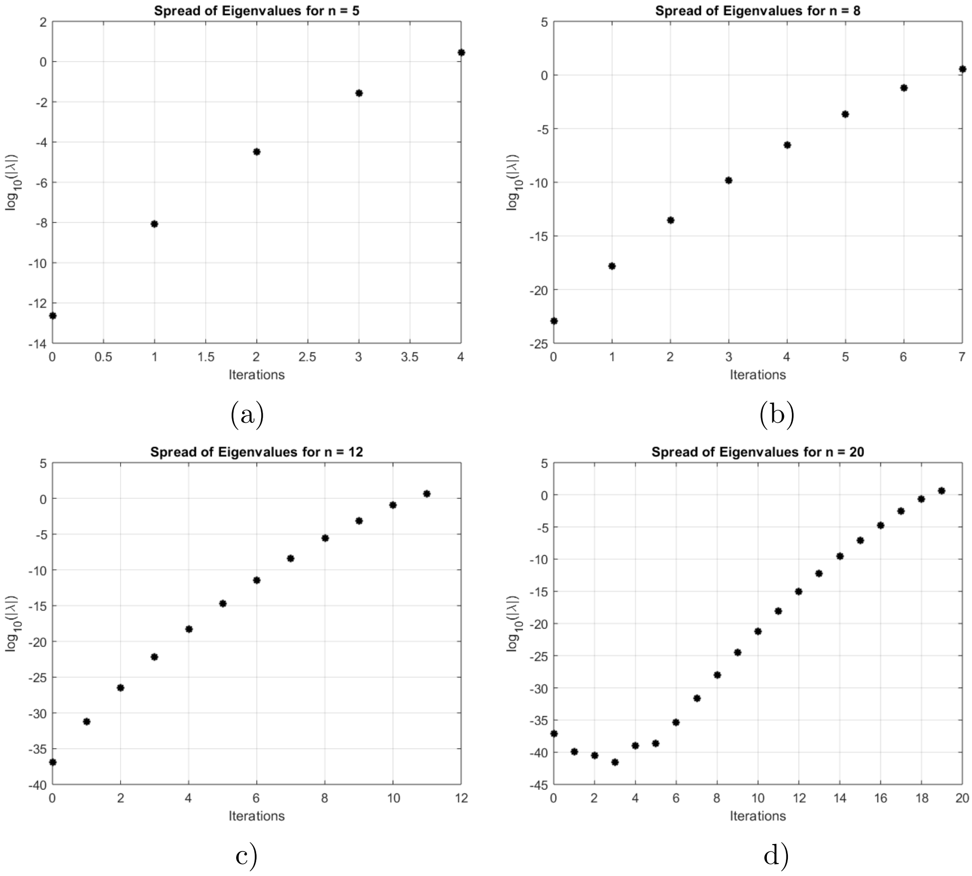


Figure 6: Spread of the Eigenvalues of the Hilbert Matrix for dimensions $n = 5$ a), $n = 8$ b), $n = 12$ c) and $n = 20$ d)

Appendix - Matlab Code

Codes for Problem 1

One-dimensional

```
m = 3;
n = 10^m;
d = ones(n,1);
A = spdiags([d -2*d d],[-1 0 1],n,n);
xk = zeros(size(d));
b = ones(size(d));
rk = A*xk - b;
pk = -rk;
kk = 0;
tol = 1e-9;

tic
while norm(rk) > tol
    rt = rk';
    pt = pk';
    Apk = A*pk; %vector
    pAp = pt*Apk; %scalar
    rtr = rt*rk;
    ak = rtr/pAp; %scalar
    xkp1 = xk + ak*pk;
    rkp1 = rk + ak*Apk;
    rkpt = rkp1';
    num = rkpt*rkp1; %scalar
    bkp1 = num/rtr; %scalar
    pkp1 = -rkp1 + bkp1*pk;
    xk = xkp1;
    rk = rkp1;
    pk = pkp1;
    kk = kk+1;
    nrk(kk) = norm(rk);
end
toc
endt = toc;
mu = eig(A);
mu = sort(mu); %sorting from smallest to biggest
kappa = mu(n)/mu(1); %condition number for A
B = nnz(A); %number of nonzero elements in A
nvec = 0:length(nrk)-1;
```

```
C = A^-1;
D = nnz(C);

figure(1)
semilogy(nvec,nrk,'b-','linewidth',1.5);
legend('L_{2} norm')
xlabel('Iterations')
ylabel('L_{2} norm')
grid on
title(['Residual 2-norm vs. Number of Iterations for n = ' num2str(n)])

figure(2)
spy(A,'b')
hold on
spy(A==-2*d,'r');
hold off
xlabel(['NNZ = ' num2str(B)])
legend('Sub/Super-diagonal elements','Diagonal Elements')
title('Pattern of non-zero elements for Matrix A')

figure(3)
spy(C,'bo')
xlabel(['NNZ = ' num2str(D)])
title('Pattern of non-zero elements for A^{-1}')
```

Two-dimensional

```
m = 0;
n = 10^m;
dim = 2;
d = ones(n^dim,1);
A = spdiags([d d -4*d d d],[-n -1 0 1 n],n^dim,n^dim);
xk = zeros(size(d));
b = ones(size(d));
rk = A*xk - b;
pk = -rk;
kk = 0;
tol = 1e-9;
tic
while norm(rk) > tol
    rt = rk';
    pt = pk';
    Apk = A*pk; %vector
    pAp = pt*Apk; %scalar
    rtr = rt*rk;
    ak = rtr/pAp; %scalar
    xkp1 = xk + ak*pk;
    rkp1 = rk + ak*Apk;
    rkpt = rkp1';
    num = rkpt*rkp1; %scalar
    bkp1 = num/rtr; %scalar
    pkp1 = -rkp1 + bkp1*pk;
    xk = xkp1;
    rk = rkp1;
    pk = pkp1;
    kk = kk+1;
    nrk(kk) = norm(rk);
end
toc
endt = toc;

mu = eig(A);
kappa = mu(n)/mu(1); %condition number for A

B = nnz(A); %number of nonzero elements in A
nvec = 0:length(nrk)-1;

figure(1)
semilogy(nvec,nrk,'b-','linewidth',1.5);
```

```
legend('L_{2} norm')
xlabel('Iterations')
ylabel('L_{2} norm')
grid on
title(['Residual 2-norm vs. Number of Iterations for n
    = ' num2str(n)])
```

Three-dimensional

```
m = 2;
n = 10^m;
dim = 3;
d = ones(n^dim,1);
A = spdiags([d d d -6*d d d d],[-n^2 -n -1 0 1 n n^2],
    n^dim,n^dim);
xk = zeros(size(d));
b = ones(size(d));
rk = A*xk - b;
pk = -rk;
kk = 0;
tol = 1e-9;
tic
while norm(rk) > tol
    rt = rk';
    pt = pk';
    Apk = A*pk; %vector
    pAp = pt*Apk; %scalar
    rtr = rt*rk;
    ak = rtr/pAp; %scalar
    xkp1 = xk + ak*pk;
    rkp1 = rk + ak*Apk;
    rkpt = rkp1';
    num = rkpt*rkp1; %scalar
    bkp1 = num/rtr; %scalar
    pkp1 = -rkp1 + bkp1*pk;
    xk = xkp1;
    rk = rkp1;
    pk = pkp1;
    kk = kk+1;
    nrk(kk) = norm(rk);
end
toc
endt = toc;

% mu = eig(A);
% kappa = mu(n)/mu(1); %condition number for A

B = nnz(A); %number of nonzero elements in A
nvec = 0:length(nrk)-1;

C = A^-1;
```

```
D = nnz(C);

figure(1)
semilogy(nvec,nrk,'b-','linewidth',1.5);
legend('L_{2} norm')
xlabel('Iterations')
ylabel('L_{2} norm')
grid on
title(['Residual 2-norm vs. Number of Iterations for n = ' num2str(n)])
```

Code for Problem 2

```
n = 20;
A = zeros(n,n);
for ii=1:n
    for jj=1:n
        A(ii,jj) = 1/(ii+jj-1);
    end
end
xk = zeros(n,1);
b = ones(n,1);
rk = A*xk - b;
pk = -rk;
kk = 0;
tol = 1e-6;

tic
while norm(rk) > tol
    rt = rk';
    pt = pk';
    Apk = A*pk; %vector
    pAp = pt*Apk; %scalar
    rtr = rt*rk;
    ak = rtr/pAp; %scalar
    xkp1 = xk + ak*pk;
    rkp1 = rk + ak*Apk;
    rkpt = rkp1';
    num = rkpt*rkp1; %scalar
    bkp1 = num/rtr; %scalar
    pkp1 = -rkp1 + bkp1*pk;
    xk = xkp1;
    rk = rkp1;
    pk = pkp1;
    kk = kk+1;
    nrk(kk) = norm(rk);
end
toc
endt = toc;


mu = eig(A);
mu = sort(mu); %sorting from smallest to biggest
kappa = mu(n)/mu(1); %condition number for A

```

```
nvec = 0:length(nrk)-1;
muvec = 0:n-1;

figure(1)
semilogy(nvec,nrk,'b-','linewidth',1.5);
legend('L_{2} norm')
xlabel('Iterations')
ylabel('L_{2} norm')
grid on
title(['Residual 2-norm vs. Number of Iterations for n
    = ' num2str(n)])

mu = log(abs(mu));
figure(2)
plot(muvec,mu,'k*','linewidth',1.5)
title(['Spread of Eigenvalues for n = ' num2str(n)])
xlabel('Iterations')
ylabel('log_{10}(|\lambda|)')
grid on
```