

MATH693a  
Dr. Peter Blomgren  
HW04  
Conjugate Gradient Method

Matteo Polimeno

November 17<sup>th</sup>, 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  $10^{-9}$ .

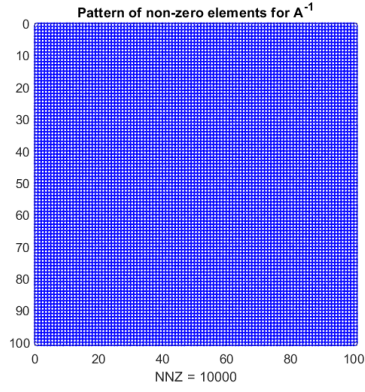
For the one,two and three-dimensional problem we run the algorithm until the laptop could handle it.

For the one-dimensional problem  $A\mathbf{x} = \mathbf{b}$ , we have the following results:

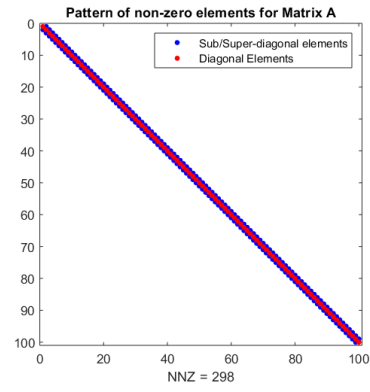
n	Non-zero elements	Elapsed Time	Iterations (sec)	Condition Number
$10^2$	298	0.0074	50	$2.4 \times 10^{-4}$
$10^3$	2998	0.0092	500	$2.5 \times 10^{-6}$
$10^4$	29998	0.5970	5000	$2.5 \times 10^{-8}$
$10^5$	299998	51.9987	50000	$2.5 \times 10^{-10}$

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.

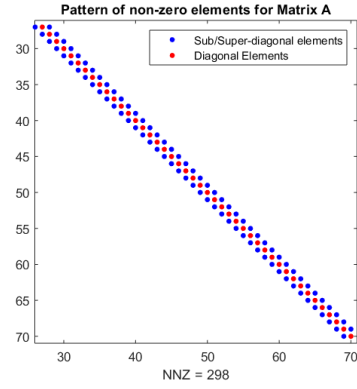
We displayed the plots for the norm of the residual for  $10^2 \leq n \leq 10^4$ , as well as a plot of the sparse matrix  $A$  and its dense inverse for  $n = 10^2$ .



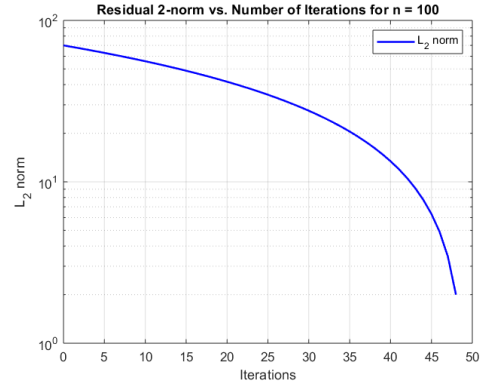
(a)



(b)

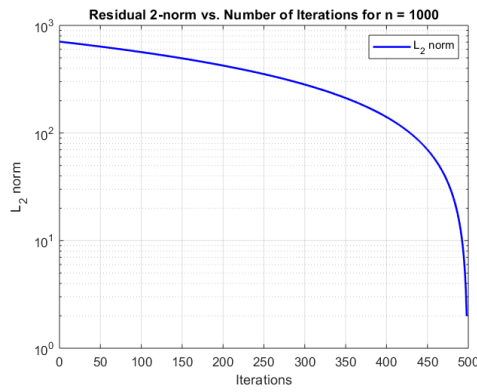


c)

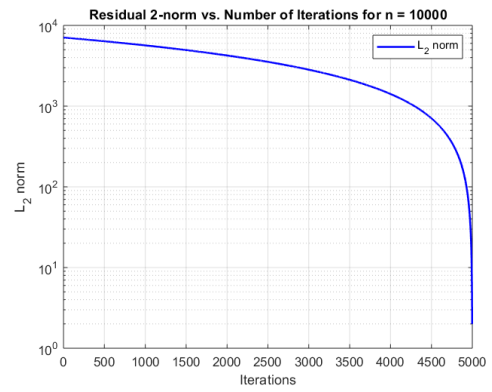


d)

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

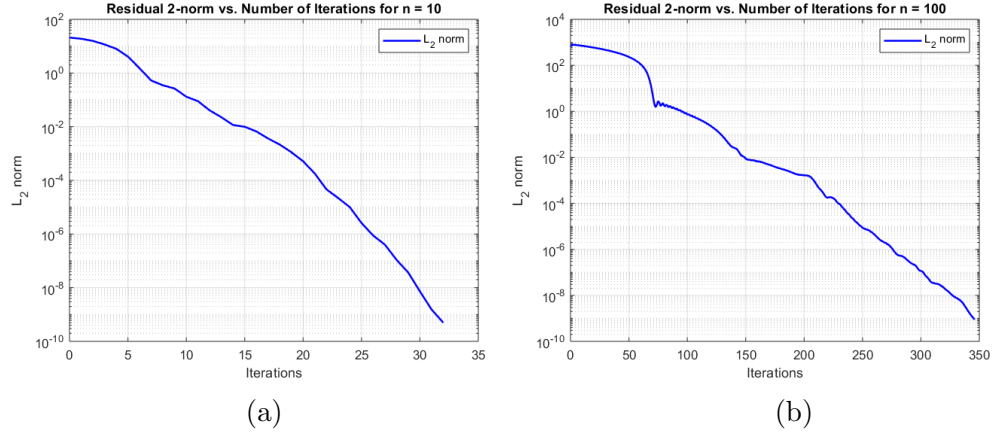


Figure 3: Euclidean norm for the given values of  $n$

For the 3-dimensional problem, we have:

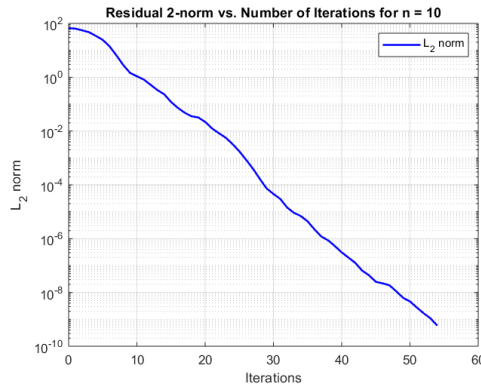
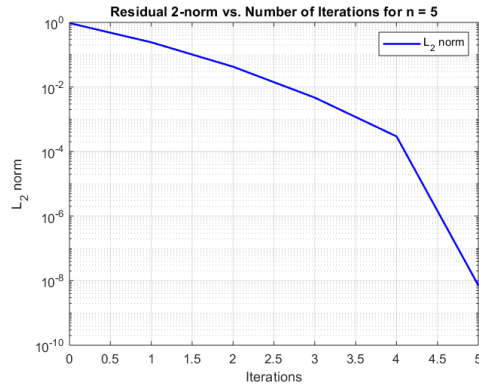


Figure 4: Euclidean norm for the given values of  $n$

## 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.

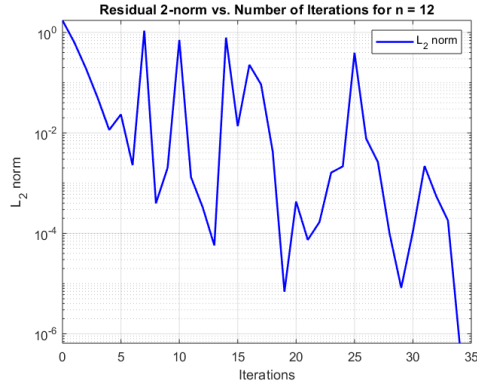
n	Elapsed Time	Iterations (sec)	Condition Number
5	0.0087	6	$4.8 \times 10^5$
8	0.0051	19	$1.5 \times 10^{10}$
12	0.0117	35	$1.9 \times 10^{16}$
20	0.0201	66	$2.5 \times 10^{16}$



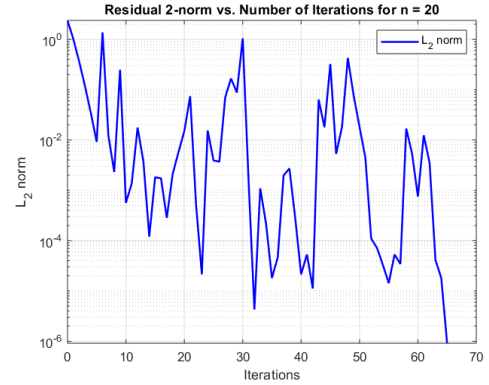
(a)



(b)

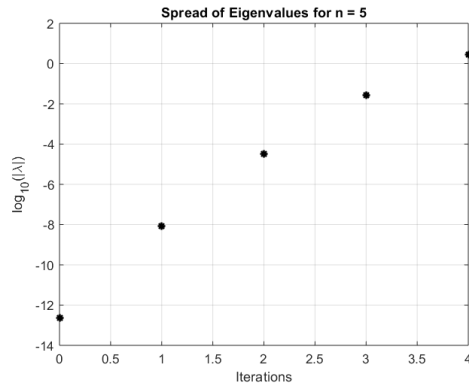


c)

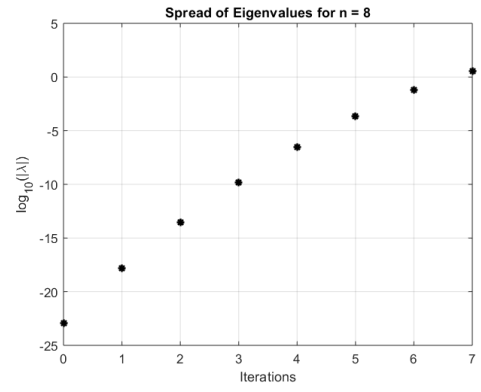


d)

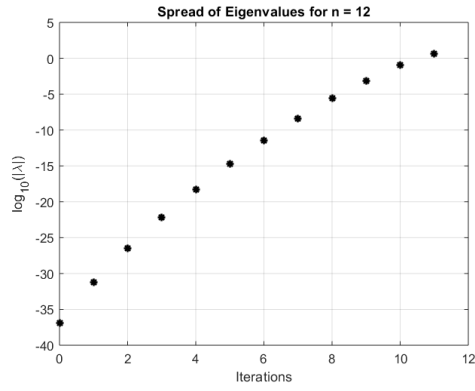
Figure 5: Residual Euclidean Norm as a function of the iterations for the given values of n



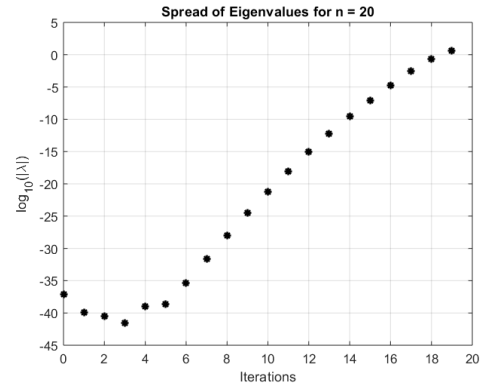
(a)



(b)



(c)



(d)

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

```
1 m = 3;
2 n = 10^m;
3 d = ones(n,1);
4 A = spdiags([d -2*d d],[-1 0 1],n,n);
5 xk = zeros(size(d));
6 b = ones(size(d));
7 rk = A*xk - b;
8 pk = -rk;
9 kk = 0;
10 tol = 1e-9;
11
12 tic
13 while norm(rk) > tol
14     rt = rk';
15     pt = pk';
16     Apk = A*pk; %vector
17     pAp = pt*Apk; %scalar
18     rtr = rt*rk;
19     ak = rtr/pAp; %scalar
20     xkp1 = xk + ak*pk;
21     rkp1 = rk + ak*Apk;
22     rkpt = rkp1';
23     num = rkpt*rkp1; %scalar
24     bkp1 = num/rtr; %scalar
25     pkp1 = -rkp1 + bkp1*pk;
26     xk = xkp1;
27     rk = rkp1;
28     pk = pkp1;
29     kk = kk+1;
30     nrk(kk) = norm(rk);
31 end
32 toc
33 endt = toc;
34 mu = eig(A);
35 mu = sort(mu); %sorting from smallest to biggest
36 kappa = mu(n)/mu(1); %condition number for A
37 B = nnz(A); %number of nonzero elements in A
38 nvec = 0:length(nrk)-1;
```

```

39
40 C = A^-1;
41 D = nnz(C);
42
43 figure(1)
44 semilogy(nvec,nrk,'b-','linewidth',1.5);
45 legend('L_{2} norm')
46 xlabel('Iterations')
47 ylabel('L_{2} norm')
48 grid on
49 title(['Residual 2-norm vs. Number of Iterations for n
        = ' num2str(n)])
50
51 figure(2)
52 spy(A,'b')
53 hold on
54 spy(A==2*d,'r');
55 hold off
56 xlabel(['NNZ = ' num2str(B)])
57 legend('Sub/Super-diagonal elements','Diagonal
        Elements')
58 title('Pattern of non-zero elements for Matrix A')
59
60 figure(3)
61 spy(C,'bo')
62 xlabel(['NNZ = ' num2str(D)])
63 title('Pattern of non-zero elements for A^{-1}')

```

## Two-dimensional

```
1 m = 0;
2 n = 10^m;
3 dim = 2;
4 d = ones(n^dim,1);
5 A = spdiags([d d -4*d d d],[-n -1 0 1 n],n^dim,n^dim);
6 xk = zeros(size(d));
7 b = ones(size(d));
8 rk = A*xk - b;
9 pk = -rk;
10 kk = 0;
11 tol = 1e-9;
12 tic
13 while norm(rk) > tol
14     rt = rk';
15     pt = pk';
16     Apk = A*pk; %vector
17     pAp = pt*Apk; %scalar
18     rtr = rt*rk;
19     ak = rtr/pAp; %scalar
20     xkp1 = xk + ak*pk;
21     rkp1 = rk + ak*Apk;
22     rkpt = rkp1';
23     num = rkpt*rkp1; %scalar
24     bkp1 = num/rtr; %scalar
25     pkp1 = -rkp1 + bkp1*pk;
26     xk = xkp1;
27     rk = rkp1;
28     pk = pkp1;
29     kk = kk+1;
30     nrk(kk) = norm(rk);
31 end
32 toc
33 endt = toc;
34
35 mu = eig(A);
36 kappa = mu(n)/mu(1); %condition number for A
37
38 B = nnz(A); %number of nonzero elements in A
39 nvec = 0:length(nrk)-1;
40
41 figure(1)
42 semilogy(nvec,nrk,'b-','linewidth',1.5);
```



```

43 legend( 'L-{2} norm ' )
44 xlabel( 'Iterations ' )
45 ylabel( 'L-{2} norm ' )
46 grid on
47 title( [ 'Residual 2-norm vs. Number of Iterations for n
          = ' num2str(n) ] )

```

### Three-dimensional

```
1 m = 2;
2 n = 10^m;
3 dim = 3;
4 d = ones(n^dim,1);
5 A = spdiags([d d d -6*d d d d],[-n^2 -n -1 0 1 n n^2],
             n^dim,n^dim);
6 xk = zeros(size(d));
7 b = ones(size(d));
8 rk = A*xk - b;
9 pk = -rk;
10 kk = 0;
11 tol = 1e-9;
12 tic
13 while norm(rk) > tol
14     rt = rk';
15     pt = pk';
16     Apk = A*pk; %vector
17     pAp = pt*Apk; %scalar
18     rtr = rt*rk;
19     ak = rtr/pAp; %scalar
20     xkp1 = xk + ak*pk;
21     rkp1 = rk + ak*Apk;
22     rkpt = rkp1';
23     num = rkpt*rkp1; %scalar
24     bkp1 = num/rtr; %scalar
25     pkp1 = -rkp1 + bkp1*pk;
26     xk = xkp1;
27     rk = rkp1;
28     pk = pkp1;
29     kk = kk+1;
30     nrk(kk) = norm(rk);
31 end
32 toc
33 endt = toc;
34
35 % mu = eig(A);
36 % kappa = mu(n)/mu(1); %condition number for A
37
38 B = nnz(A); %number of nonzero elements in A
39 nvec = 0:length(nrk)-1;
40
41 C = A^-1;
```

```

42 D = nnz(C);
43
44 figure(1)
45 semilogy(nvec,nrk,'b-','linewidth',1.5);
46 legend('L_{2} norm')
47 xlabel('Iterations')
48 ylabel('L_{2} norm')
49 grid on
50 title(['Residual 2-norm vs. Number of Iterations for n'
        = ' num2str(n)'])

```

## Code for Problem 2

```
1  n = 20;
2  A = zeros(n,n);
3  for ii=1:n
4      for jj=1:n
5          A(ii,jj) = 1/(ii+jj-1);
6      end
7  end
8  xk = zeros(n,1);
9  b = ones(n,1);
10 rk = A*xk - b;
11 pk = -rk;
12 kk = 0;
13 tol = 1e-6;
14
15 tic
16 while norm(rk) > tol
17     rt = rk';
18     pt = pk';
19     Apk = A*pk; %vector
20     pAp = pt*Apk; %scalar
21     rtr = rt*rk;
22     ak = rtr/pAp; %scalar
23     xkp1 = xk + ak*pk;
24     rkp1 = rk + ak*Apk;
25     rkpt = rkp1';
26     num = rkpt*rkp1; %scalar
27     bkp1 = num/rtr; %scalar
28     pkp1 = -rkp1 + bkp1*pk;
29     xk = xkp1;
30     rk = rkp1;
31     pk = pkp1;
32     kk = kk+1;
33     nrk(kk) = norm(rk);
34 end
35 toc
36 endt = toc;
37
38
39 mu = eig(A);
40 mu = sort(mu); %sorting from smallest to biggest
41 kappa = mu(n)/mu(1); %condition number for A
42
```

```

43 nvec = 0:length(nrk)-1;
44 muvec = 0:n-1;
45
46 figure(1)
47 semilogy(nvec,nrk,'b-','linewidth',1.5);
48 legend('L_{2} norm')
49 xlabel('Iterations')
50 ylabel('L_{2} norm')
51 grid on
52 title(['Residual 2-norm vs. Number of Iterations for n'
        = ' num2str(n)'])
53
54 mu = log(abs(mu));
55 figure(2)
56 plot(muvec,mu,'k*','linewidth',1.5)
57 title(['Spread of Eigenvalues for n = ' num2str(n)])
58 xlabel('Iterations')
59 ylabel('log_{10}(|\lambda|)')
60 grid on

```