

HW 9

Math 501  
Numerical analysis

Spring, 2007  
California State University, Fullerton

Nasser M. Abbasi

August 13, 2021

Compiled on August 13, 2021 at 10:04pm

# Contents

<b>1</b>	<b>Source code</b>	<b>8</b>
1.1	nma_inverse_power.m . . . . .	8
1.2	nma_inverse_shifted_power.m . . . . .	8
1.3	nma_power.m . . . . .	9
1.4	nma_shifted_power.m . . . . .	10
1.5	nma_test.m . . . . .	10

20/20

HW # 9

Math 501

Nasser Abbasi

April 11, 2007.

Problem # 9, Section 4.7

Computer Assignment 4/4/2007

Iterative Eigen values

Power

Inverse Power

Shifted Power

Shifted Inverse Power

Problem # 9, section 4.7

let  $A$  be  $n \times n$ , and  $A$ -orthonormal system exist, show that  $A$  is SPD.

Solution outline express  $\bar{x}$  in terms of basis  $u^{(i)}$ , start with definition of  $\bar{x}^T A \bar{x}$ , show this must be  $> 0$  utilize property of  $A$ -orthonormal to cancel terms:

solution

given  $A$   $n \times n$

given  $A$ -orthonormal system  $[u^1 | u^2 | \dots | u^n]$

$$\text{i.e. } \langle u^i, Au^j \rangle = \delta_{ij}$$

$$(1) \left\{ \begin{array}{l} \text{i.e. } (u^1)^T Au^1 = 1 \quad (u^2)^T Au^1 = 0 \quad \dots \quad (u^n)^T Au^1 = 0 \\ (u^1)^T Au^2 = 0 \quad (u^3)^T Au^2 = 1 \quad \dots \quad (u^n)^T Au^2 = 0 \\ (u^1)^T Au^3 = 0 \quad \vdots \quad \vdots \\ \vdots \\ (u^1)^T Au^n = 0 \quad (u^2)^T Au^n = 0 \quad \dots \quad (u^n)^T Au^n = 1 \end{array} \right.$$

now  $A$  is SPD if  $\bar{x}^T A \bar{x} > 0$  ( $\bar{x} \neq 0$ )

since  $u^i$  are Basis in  $\mathbb{R}^n$ , we can write vector  $\bar{x}$  in terms of these basis. so

$$\bar{x} = \langle \bar{x}, u^1 \rangle u^1 + \langle \bar{x}, u^2 \rangle u^2 + \dots + \langle \bar{x}, u^n \rangle u^n$$

hence using this, we now write  $\bar{x}^T A \bar{x}$  and compare to (1)

$$\bar{x}^T A \bar{x} = (\langle \bar{x}, u^1 \rangle u^1 + \langle \bar{x}, u^2 \rangle u^2 + \dots + \langle \bar{x}, u^n \rangle u^n)^T A (\langle \bar{x}, u^1 \rangle u^1 + \langle \bar{x}, u^2 \rangle u^2 + \dots + \langle \bar{x}, u^n \rangle u^n)$$

let  $\langle \bar{x}, u^1 \rangle = a_1$ ,  $\langle \bar{x}, u^2 \rangle = a_2$  etc... these are the coordinates of  $\bar{x}$  in this subspace. so we write

$$\bar{x}^T A \bar{x} = (a_1 u^1 + a_2 u^2 + \dots + a_n u^n)^T A (a_1 u^1 + a_2 u^2 + \dots + a_n u^n)$$

$$= (a_1 u^1 + a_2 u^2 + \dots + a_n u^n)^T (A a_1 u^1 + A a_2 u^2 + \dots + A a_n u^n)$$

$$\begin{aligned} &= a_1^2 (u^1)^T A u^1 + a_1 a_2 u^1{}^T A u^2 + a_1 a_3 u^1{}^T A u^3 + \dots + a_1 a_n (u^1)^T A u^n \\ &+ a_2 a_1 (u^2)^T A u^1 + a_2^2 (u^2)^T A u^2 + a_2 a_3 (u^2)^T A u^3 + \dots + a_2 a_n (u^2)^T A u^n \\ &\dots + a_n a_1 (u^n)^T A u^1 + a_n a_2 (u^n)^T A u^2 + \dots + a_n^2 (u^n)^T A u^n \end{aligned}$$

So we see the pattern for  $\bar{x}^T A \bar{x}$  as

$$\begin{aligned}
 &= a_1^2 (\bar{u}^1)^T A \bar{u}^1 + \dots + \dots + \dots \\
 &\quad + a_2^2 (\bar{u}^2)^T A \bar{u}^2 + \dots \\
 &\quad \dots \dots \dots + a_3^2 (\bar{u}^3)^T A \bar{u}^3 + \dots
 \end{aligned} \tag{2}$$

But since from (1) we see that  $(\bar{u}^1)^T A \bar{u}^1 = 1$ ,  
 $(\bar{u}^2)^T A \bar{u}^2 = 1, \dots, (\bar{u}^n)^T A \bar{u}^n = 1$  and  
 everything else is zero, then (2) can

be written as

$$\bar{x}^T A \bar{x} = a_1^2 + a_2^2 + a_3^2 + \dots + a_n^2$$

but the  $a_i$  are the coordinates of  $\bar{x}$  in  $\mathbb{R}^n$ .  
 and for  $\bar{x} \neq \bar{0}$ , then at least one coordinate  
 must not vanish. Hence  $\bar{x}^T A \bar{x} \neq 0$ .

in addition, since the coordinates are all squared  
 hence it is positive sum. 10/10

$$\implies \boxed{\bar{x}^T A \bar{x} > 0} \quad \text{ie } A \text{ is SPD} \quad \text{QED}$$

Name: Nasser Abbasi  
 Math 501 – Numerical Analysis & Computation – Dr. Lee - Spring 2007

Computer Assignment 04/04/2007

1) Implement in MATLAB the following iterative Eigen methods:

- a) Power
- b) Inverse Power
- c) Shifted Power
- d) Shifted Inverse Power

2) Suppose  $A = \begin{bmatrix} 4 & 2 & 1 & 0 & 0 \\ 1 & 4 & 1 & 1 & 0 \\ 2 & 1 & 4 & 1 & 2 \\ 0 & 1 & 1 & 4 & 1 \\ 0 & 0 & 1 & 2 & 4 \end{bmatrix}$  and start with  $\vec{w}^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$ , find with 10 iterations:

- a) The dominant eigenpair of A
- b) The least dominant eigenpair of A
- + c) The eigenpair of A farthest to 2
- d) The eigenpair of A closest to 6.5

```

% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% test script to run problem 2 on the data given

A=[4 2 1 0 0;
   1 4 1 1 0;
   2 1 4 1 2;
   0 1 1 4 1;
   0 0 1 2 4];
initialEigenvectorGuess=[1 1 1 1 1];

%use Matlab to see the values to verify against

[v,1]=eig(A)

%Set parameters
maxIter=10;
delErr=0.0001; %not specified, try these
delEps=0.0001; %not specified, try these

[lambda]=nma_power(A,initialEigenvectorGuess,maxIter,delErr,delEps);
fprintf('----- power result. Eigenvalue=%f\n',lambda);

[lambda]=nma_inverse_power(A,initialEigenvectorGuess,maxIter,delErr,delEps);
fprintf('----- power inverse result. Eigenvalue=%f\n',lambda);

[lambda,k]=nma_shifted_power(A,initialEigenvectorGuess,maxIter,delErr,delEps,2);
lambda=2+lambda;
fprintf('----- power shifted result. Eigenvalue=%f\n',lambda);

[lambda,k]=nma_inverse_shifted_power(A,initialEigenvectorGuess,maxIter,delErr,delEps,6.5);
lambda=6.5+lambda;
fprintf('----- power inverse shifted result. Eigenvalue=%f\n',lambda);

```

## TEST RESULT

v =

-0.3861	-0.6325	0.0000	0.2405	-0.5000
-0.3861	-0.3162	0.4082	0.2405	0.5000
-0.6354	0.0000	-0.8165	-0.8767	-0.0000
-0.3861	0.3162	0.4082	0.2405	-0.5000
-0.3861	0.6325	0.0000	0.2405	0.5000

l =

7.6458	0	0	0	0
0	5.0000	0	0	0
0	0	3.0000	0	0
0	0	0	2.3542	0
0	0	0	0	2.0000

```

----- power result. Eigenvalue=7.645769
----- power inverse result. Eigenvalue=2.354139
----- power shifted result. Eigenvalue=7.645755
----- power inverse shifted result. Eigenvalue=7.645724

```

```

function
[lambda_new,k]=nma_inverse_shifted_power(A,initialEigenvectorGuess,maxIter,delE
rr,delEps,mu)
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative Inverse shifted power method

w_old=initialEigenvectorGuess(:);
lambda_old=rand;
In=eye(size(A,1));
A=inv(A-mu*In);

for k=1:maxIter
    w_old=w_old/norm(w_old);
    w_new=A*w_old;
    lambda_new=dot(w_old,w_new);

    if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
        break;
    else
        w_old=w_new;
        lambda_old=lambda_new;
    end
end

lambda_new=1/lambda_new;

```

```

=====

function [lambda_new,k]=nma_shifted_power(A,initialEigenvectorGuess,...
maxIter,delErr,delEps,mu)
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative shifted power method

w_old=initialEigenvectorGuess(:);
lambda_old=mu;
In=eye(size(A,1));
A=A-mu*In;

for k=1:maxIter
    w_old=w_old/norm(w_old);
    w_new=A*w_old;
    lambda_new=dot(w_old,w_new);

    if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
        break;
    else
        w_old=w_new;
        lambda_old=lambda_new;
    end
end

```



# 1 Source code

---

## 1.1 nma\_inverse\_power.m

```
function lambda_new=nma_inverse_power(A,initialEigenvectorGuess,maxIter,delErr,delEps)
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative Inverse power method

A=inv(A);

w_old=initialEigenvectorGuess(:);
lambda_old=rand;

for k=1:maxIter
    w_old=w_old/norm(w_old);
    w_new=A*w_old;
    lambda_new=dot(w_old,w_new);

    if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
        break;
    else
        w_old=w_new;
        lambda_old=lambda_new;
    end
end

lambda_new=1/lambda_new;
```

## 1.2 nma\_inverse\_shifted\_power.m

```
function [lambda_new,k]=nma_inverse_shifted_power(A,initialEigenvectorGuess,maxIter,delErr,de
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative Inverse shifted power method

w_old=initialEigenvectorGuess(:);
lambda_old=rand;
In=eye(size(A,1));
A=inv(A-mu*In);

for k=1:maxIter
    w_old=w_old/norm(w_old);
```

```

w_new=A*w_old;
lambda_new=dot(w_old,w_new);

if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
    break;
else
    w_old=w_new;
    lambda_old=lambda_new;
end
end

lambda_new=1/lambda_new;

```

### 1.3 nma\_power.m

```

function lambda_new=nma_power(A,initialEigenvectorGuess,maxIter,delErr,delEps)
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative power method

w_old=initialEigenvectorGuess(:);

lambda_old=rand;

for k=1:maxIter
    w_old=w_old/norm(w_old);
    w_new=A*w_old;
    lambda_new=dot(w_old,w_new);
    if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
        break;
    else
        w_old=w_new;
        lambda_old=lambda_new;
    end
end
end

```

## 1.4 nma\_shifted\_power.m

```

function [lambda_new,k]=nma_shifted_power(A,initialEigenvectorGuess, ...
    maxIter,delErr,delEps,mu)
%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% IMPLEMENT iterative shifted power method

w_old=initialEigenvectorGuess(:);
lambda_old=mu;
In=eye(size(A,1));
A=A-mu*In;

for k=1:maxIter
    w_old=w_old/norm(w_old);
    w_new=A*w_old;
    lambda_new=dot(w_old,w_new);

    if norm(w_new-w_old)<delErr || abs(lambda_old-lambda_new)<delEps
        break;
    else
        w_old=w_new;
        lambda_old=lambda_new;
    end
end
end

```

## 1.5 nma\_test.m

```

%
% MATH 501, Computer Assignment 04/04/2007
% by Nasser Abbasi
%
% test script to run problem 2 on the data given

A=[4 2 1 0 0;
    1 4 1 1 0;
    2 1 4 1 2;
    0 1 1 4 1;
    0 0 1 2 4];
initialEigenvectorGuess=[1 1 1 1 1];

%use Matlab to see the values to verify against

[v,1]=eig(A)

```

```
%Set parameters
maxIter=10;
delErr=0.0001; %not specified, try these
delEps=0.0001; %not specified, try these

[lambda]=nma_power(A,initialEigenVectorGuess,maxIter,delErr,delEps);
fprintf('----- power result. Eigenvalue=%f\n',lambda);

[lambda]=nma_inverse_power(A,initialEigenVectorGuess,maxIter,delErr,delEps);
fprintf('----- power inverse result. Eigenvalue=%f\n',lambda);

[lambda,k]=nma_shifted_power(A,initialEigenVectorGuess,maxIter,delErr,delEps,2);
lambda=2+lambda;
fprintf('----- power shifted result. Eigenvalue=%f\n',lambda);

[lambda,k]=nma_inverse_shifted_power(A,initialEigenVectorGuess,maxIter,delErr,delEps,6.5);
lambda=6.5+lambda;
fprintf('----- power inverse shifted result. Eigenvalue=%f\n',lambda);
```