

Solutions : Problem Set # 4

1) In this problem we examine the second kind integral equation:

$$\lambda\sigma(x) - \int_0^1 (x+x')\sigma(x')dx' = c_1x + c_2 \quad (1)$$

We know from theory that the first kind integral equations have rich null spaces. We look at this *second kind* integral equation to demonstrate that it too, has a null space. In order to examine the null space we follow a procedure similar to that outlined in the May 8th lecture, slide 19 onwards.

a) We would first like to find the unknown density $\sigma(x)$, given $\lambda = 0.5, c_1 = 1, c_2 = 0$. The hint for this problem is to try $\sigma(x) = \text{constant}$.

First we start with the integral equation:

$$\lambda\sigma(x) - \int_0^1 (x+x')\sigma(x')dx' = c_1x + c_2$$

By setting the density, $\sigma(x) = \text{constant}$, we get, and filling in the various given quantities:

$$0.5 \cdot \sigma - \sigma \int_0^1 (x+x')dx' = x$$

We can simplify this as:

$$0.5 \cdot \sigma - \sigma \left| xx' + \frac{x'^2}{2} \right|_0^1 = x$$

Further:

$$0.5 \cdot \sigma - \sigma x - \sigma \frac{1}{2} = x$$

From this we see the appropriate value of the density which satisfies the equation for the given RHS forcing:

$$\sigma = -1$$

b) In this section we examine the null space of the integral equation. Here we are asked to find values of λ , for which the integral equation does not have a unique solution. We look at a similar approach as was examined in class, where a density distribution is assumed and then verified to be in the null space.

We start by trying several other functions for the density. By looking at the integral equation closely, and splitting it into two parts for clarity:

$$\lambda\sigma(x) - \int_0^1 (x+x')\sigma(x')dx' = 0$$

$$\lambda\sigma(x) - \left[\int_0^1 (x\sigma(x'))dx' + \int_0^1 \sigma(x')dx' \right] = 0$$

We see from this that the first integral will always give a linear function in x , while the second integral will always give a constant. Therefore in order to have this satisfy the $RHS = 0$, we see that it is likely that the density function, $\sigma(x)$, is a first order polynomial. We therefore try, $\sigma(x) = ax + b$, in the equation:

$$(\lambda)(ax + b) - \left[\int_0^1 ((x)(ax'+b))dx' + \int_0^1 (x')(ax'+b)dx' \right] = 0$$

This gives us:

$$\lambda ax + \lambda b - \left[\int_0^1 (xax'+bx)dx' + \int_0^1 (x'ax'+x'b)dx' \right] = 0$$

Performing the integration we get:

$$\lambda ax + \lambda b - \left[\left(xa \frac{x'^2}{2} + bx x' \right) + \left(a \frac{x'^3}{3} + \frac{x'^2}{2} b \right) \right]_0^1 = 0$$

Simplifying:

$$\lambda ax + \lambda b - \left[\left(\frac{a}{2} x + bx \right) + \left(\frac{a}{3} + \frac{b}{2} \right) \right] = 0$$

By grouping the terms of similar order in x^1 , and x^0 we get:

$$\lambda ax - \left(\frac{a}{2} x + bx \right) = 0$$

$$\lambda b - \left(\frac{a}{3} + \frac{b}{2} \right) = 0$$

We can see from this that the following eigenvalue problem is produced:

$$\begin{bmatrix} \frac{1}{2} & 1 \\ \frac{1}{3} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \lambda \begin{bmatrix} a \\ b \end{bmatrix}$$

From here we see that the problem is one of solving the eigenvalues to get the λ values:

$$\lambda_1 = \frac{1}{2} + \frac{\sqrt{3}}{3} \approx 1.077350$$

$$\lambda_2 = \frac{1}{2} - \frac{\sqrt{3}}{3} \approx -0.077350$$

So given these values we can determine a , and, b , the eigenvector values:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 \\ 1/2 \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} -\sqrt{3}/2 \\ 1/2 \end{bmatrix}$$

So, for,

$$\lambda_1 = \frac{1}{2} + \frac{\sqrt{3}}{3} \approx 1.077350 \quad \text{we see that the corresponding } \sigma(x) = \sqrt{3}/2 x + 1/2$$

$$\lambda_2 = \frac{1}{2} - \frac{\sqrt{3}}{3} \approx -0.077350 \quad \text{we see that the corresponding } \sigma(x) = -\sqrt{3}/2 x + 1/2$$

c) For part c we look for two solutions which satisfy the integral equation for the problem. A simple way to do this is choose $\sigma(x) = 1$.

$$\lambda - \int_0^1 (x + x') dx' = c_1 x + c_2$$

$$\lambda - (x + \frac{1}{2}) = c_1 x + c_2$$

$$\left. \begin{array}{l} \frac{1}{2} + \frac{\sqrt{3}}{3} \\ \frac{1}{2} - \frac{\sqrt{3}}{3} \end{array} \right\} - (x + \frac{1}{2}) = c_1 x + c_2$$

$$\lambda_1 \text{ \& } \lambda_2 : c_1 = 1$$

$$\lambda_1 : c_2 = \frac{\sqrt{3}}{3}$$

$$\lambda_2 : c_2 = -\frac{\sqrt{3}}{3}$$

i. Solution # 1:

$$\sigma(x) = 1, \text{ for } \lambda_1 = \frac{1}{2} + \frac{\sqrt{3}}{3}, \text{ and } \lambda_2 = \frac{1}{2} - \frac{\sqrt{3}}{3}$$

ii. Solution # 2:

$$\sigma(x) = 1 + \frac{\sqrt{3}}{2}x + \frac{1}{2}, \text{ for } \lambda_1 = \frac{1}{2} + \frac{\sqrt{3}}{3}$$

iii. Solution # 3:

$$\sigma(x) = 1 - \frac{\sqrt{3}}{2}x + \frac{1}{2}, \text{ for } \lambda_2 = \frac{1}{2} - \frac{\sqrt{3}}{3}$$

2) In this problem we examine the general idea of Nystrom Methods. We go through several steps to solve the following integral equation:

$$\sigma(x) + \int_0^1 (x - x')^2 \sigma(x') dx' = \cos\left(\frac{\pi}{2}x\right) \quad (2)$$

a) In the Legendre.m code we are asked why we can set the RHS vector to be [2 0 0 0 ... 0 0]'. We can look at it in two ways:

- i) The Legendre polynomials, when integrated over the interval [-1,1] give a non-zero solution for $P_0=1$ only. The integral is a value of 2, hence we see a value of '2'. The rest are equally weighted above and below the axis by construction, and therefore result in zero integrals. This is actually a consequence of the method used to derive and determine Legendre polynomials (see **note**).
- ii) The Legendre Polynomials, by construction are an orthogonal set of functions. This means that the inner product of any of the higher order polynomials with the first one, $P_0=1$, will give zero. So the only non-zero entry in the RHS is the first one which has an integrated area of 2, in the interval [-1,1].

Note: For interests sake, the Legendre polynomials are constructed using a process which guarantees that they will be an orthogonal basis in the interval [-1,1]. The method used to derive the polynomials is the **Gram-Schmidt** procedure (see Strang, G., *An Introduction To Applied Mathematics*, M.I.T.,1986 ~ pp. 278-282). The resulting polynomials are:

$$\begin{aligned}
P_0 &= 1 \\
P_1 &= x \\
P_2 &= x^2 - 1/3
\end{aligned}$$

Etc...

This ***Gram-Schmidt*** idea is an important one in Numerical Methods, as it incorporates the idea of orthogonality, which is often very useful in engineering numerical analysis.

- b) In order to rescale the quadrature points we can take either a simple or, slightly more complex approach. The simple approach is analogous to a mapping to a new coordinate system. The complex approach would be to re-derive the weights and evaluation points for each new interval encountered. Rather than re-deriving the quadrature scheme for each new interval of integration we show that it is simpler to scale and shift the scheme:

- i) For the weights, we scale them by $1/2$, since the test interval is reduced by $1/2$.

$$W_i^{new} = \frac{W_i^{old}}{2}$$

- ii) For the evaluation points we have to scale and shift. Here we scale by $1/2$ and shift to the right $1/2$. This will transform from $[-1,1]$, to $[0,1]$.

$$x_i^{old} = \frac{x_i^{new}}{2} + \frac{1}{2}$$

As a test example we look at computing the integral:

$$\int_0^1 x^{12} dx = \frac{1}{13} \approx 0.076923$$

We see that the integral is computed exactly for those cases where the number of quadrature points is 7 or greater, ie. $(2n-1) =$ order of polynomial. This shows that, indeed, the quadrature scheme uses a polynomial base, and when the quadrature scheme uses a similar base to the polynomial it is approximating we can get very good results.

n	Integrated Values	Error
1	0.000244141	0.076678936
2	0.028956619	0.047966458
3	0.066259063	0.010664014
4	0.076009163	0.000913914
5	0.076898481	2.45963E-05
6	0.076922987	9.00975E-08
7	0.076923077	0
8	0.076923077	0
9	0.076923077	0
10	0.076923077	0
11	0.076923077	0
12	0.076923077	0
13	0.076923077	0
14	0.076923077	0
15	0.076923077	0
16	0.076923077	0
17	0.076923077	0
18	0.076923077	0
19	0.076923077	0
20	0.076923077	0
21	0.076923077	0
22	0.076923077	0
23	0.076923077	0
24	0.076923077	0
25	0.076923077	0

Integration of a twelfth order polynomial.

c) We use the Nystrom method as described in class to solve the integral equation.

$$\sigma(x) + \int_0^1 (x-x')^2 \sigma(x') dx' = \cos\left(\frac{\pi}{2}x\right)$$

The idea is to use the quadrature evaluation points as the collocation points for solving the integral equation numerically. When we do this we get a linear system of equations:

$$\begin{bmatrix} 1 + w_1 G(x_1, x_1) & K & K & w_n G(x_1, x_n) \\ M & O & & M \\ M & & O & M \\ w_1 G(x_n, x_1) & K & K & 1 + w_n G(x_n, x_n) \end{bmatrix} \begin{bmatrix} \sigma_{n1} \\ M \\ M \\ \sigma_{nm} \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\pi}{2}x_1\right) \\ M \\ M \\ \cos\left(\frac{\pi}{2}x_n\right) \end{bmatrix}$$

To see the implementation of the Nystrom Method in Matlab[®] please refer to the code inserted in the appendix.

- i) When we examine the convergence of the solution, we need to be a little more careful. We know that the quadrature scheme uses increasing order of polynomials in the approximation of the integral. We can, from this say that the Nystrom method uses a polynomial basis in the approximation of the solution. The polynomial basis we use has full support. Once we have a solution at the given points we can use a polynomial interpolation through the solution points given to compare points from one polynomial to the next.

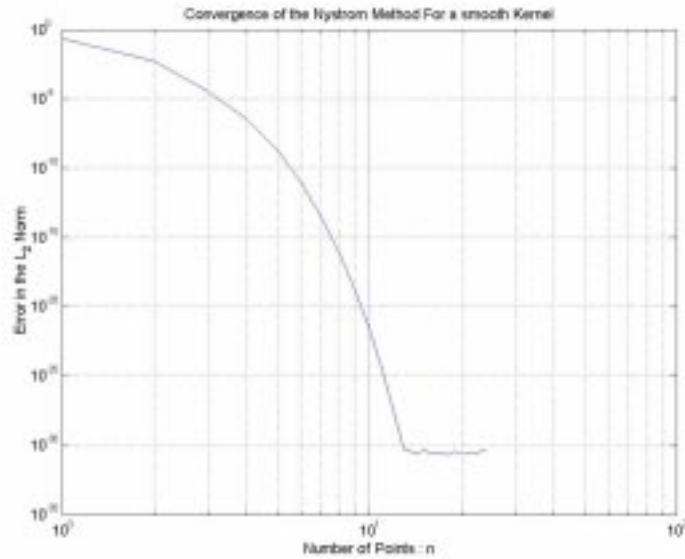
Another way to compare one solution to the next would be to use the odd order Nystrom calculations, and compare the common midpoint in the solution. This would eliminate the need for a poly fit and would at the same time give a decent idea on how the errors decrease with Nystrom/Quadrature order.

In the following example we look at the convergence of the Nystrom implementation by comparing the solution approximated with an nth order quadrature scheme with the 25 point quadrature scheme. We note that the error can be measured in an $\|L\|_2$ norm for comparison:

$$e|_n = \|u_{25} - u_n\|_2 = \left(\int_0^1 |\hat{u}_{25} - \hat{u}_n|^2 dx \right)^{1/2}$$

Where \hat{u} , represents the polynomial fit of the solution.

The resulting error vs. quadrature order plot we see is:



- ii) The code for this portion of the assignment is outlined after question # 3.

-
- 3) In this question we examine a very similar integral equation to what was looked at in part 2 of this problem set. The integral equation we are now looking at is:

$$\sigma(x) + \int_0^1 |x - x'| \sigma(x') dx' = \cos\left(\frac{\pi}{2} x\right)$$

- a) We examine the convergence of the Nystrom method for this case, compared with the convergence of the integral equation in part # 2. Note that we see exponential convergence as expected.

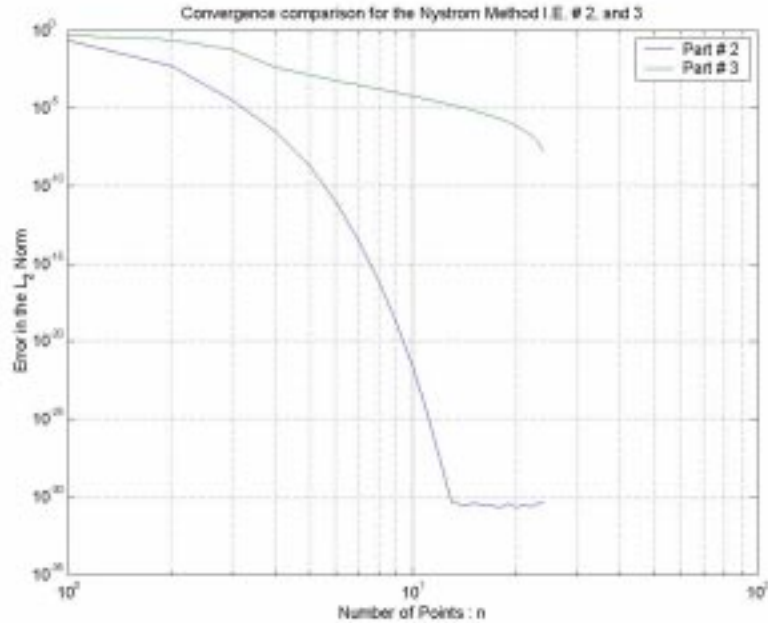
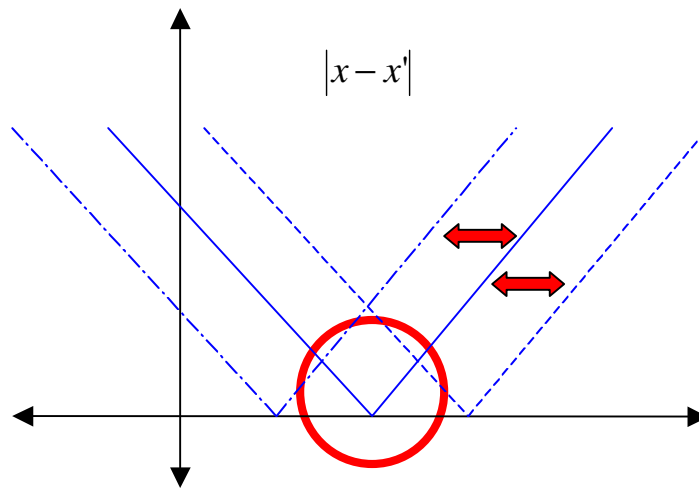


Figure: The error plotted against the number of quadrature points.

- b) When we examine the convergence of the two cases we see that the convergence for part 3 is a lot slower than the convergence shown in part 2. This is a consequence of the kernel in the integral operator. The kernel in the first equation is smooth. The kernel in the second equation is only piecewise continuous, due to the absolute value operator. We try to fit this with a set of smooth constant polynomials. The first derivative discontinuity is not modeled well with polynomials, and hence, we see very slow convergence. We also should note due to the $|x-x'|$, the first derivative discontinuity moves around the domain with x , and as such, the solution to this problem is not simply to divide the integral around the $x=0$ point, but rather around the point where $x=x'$.



We know from the convergence analysis for the Nystrom method that we will only see exponential convergence if the integrand is smooth, due to the implicit use of a smooth polynomial basis.

-
- c) We could try to regain a fast convergence rate for the Nystrom method in this case in several different ways:
- i. We could try to analytically integrate the equation for points near the first derivative discontinuity.
 - ii. We could try to evaluate the integral in the region close to the singularity with a different quadrature scheme for more accuracy.
 - iii. We could try to split the integral into two parts and apply our quadrature scheme very carefully to the two individual parts in order to capture the integral without trying to integrate over the first derivative discontinuity (so split the integral about x , so integrate from $[-1, x]$, and then add to $[x, 1]$). Care should be taken to carefully perform this integration.

iv. Nystrom Method Code (part # 2 & 3)

```
function [x,w,xm,wm,sigma, sigma1] = legendrequad(n)
% Computes the quadrature points and weights based on the legendre polynomials,
% which are orthogonal on the interval from -1 to 1.
% This routine relies on evallegendrepoly, a routine which evaluates
% a legendre polynomial of a given order at an x value between
% -1 and 1
% n - order of the quadrature
% x - quadrature points
% w - quadrature weights

if n > 25
disp('Can not use legendre quad for greater than n=25');
break;
end

% Below we compute the roots of the nth order legendre polynomial
% using the evallegendrepoly function. This routine uses polyfit (okay
% I was lazy) and polyfit becomes too ill-conditioned for n > 25
m = n+1;
even_pts = [1:m]/(m/2) - 1 - 1/m;
pts = sin(0.5 * pi * even_pts); % better spacing for polynomial points
vals = evallegendrepoly(pts,m-1);
coeffs = polyfit(pts,vals,m-1); % Polyfit fits a polynomial to a set of points
rzeros = roots(coeffs);
[rzeros,index] = sort(rzeros);

% Now form the matrix to compute the weights
for i = 1:n
    for j = 1:n
        A(i,j) = evallegendrepoly(rzeros(j),i-1);
    end
end

% The right hand side should be made up from integrals of the
% legendre polynomials over the interval -1 to 1.
% Why does the below work?
rhs = zeros(n,1);
rhs(1) = 2;

w = A \ rhs; % Solve the system for the weights.
x = rzeros; % Points are zeros of high order chebychev poly.

%=====
% Modified Code
%=====
% the following portion of code was mod-
% ified by Dave Willis, 11/05/2002
%
% SMA5215, 16.910/2.097
% © MIT 2002. All rights reserved.
%=====

%=====
% Rescaling the weights and the evaluation
% points.
%=====
wm=w/2;
xm=x/2+.5;

%=====
% Solution for the first integral equation
%=====
% Since this code is simple we tack it on
% to the end of the legendre code. We mod-
% ify the I/O of the code appropriately
%=====

rhs_gq = cos(pi/2*xm);
AA = zeros(n,n);
for i=1:n
    for j=1:n
        AA(i,j) = wm(j)*(xm(i)-xm(j))^2;
```

```

        end
        AA(i,i) = AA(i,i)+1.;
    end
    sigma = AA\rhs_gq;

%=====
% Solution for the second integral equation
%=====
% Since this code is simple we tack it on
% to the end of the legendre code. We mod-
% ify the I/O of the code appropriately
%=====

rhs_gq1 = cos(pi/2*x);
AA1 = zeros(n,n);
for i=1:n
    for j=1:n
        AA1(i,j) = w(j)*abs(x(i)-x(j));
    end
    AA1(i,i) = AA1(i,i)+1.;
end
sigma1 = AA1\rhs_gq1;

```

Error Calculations (Matlab® Code)

```

%=====
% Error.m
%=====
% The flowing code is a routine used to
% compute the convergence of the Nystrom
% method with increasing order of the
% quadrature scheme.
%=====
%
% Written : 11 May, 2002
% By: Dave Willis
% For: SMA 5212/16.910/2.097
%
% ©MIT 2002. All rights reserved.
%
%=====

% 25th order polynomial "exact solution"

[x1,w,xm1,wm,sigma25,s125] = legendrequad(n);

% Poly fit of this function
[P1]=polyfit(xm1,sigma25,24);
[P2]=polyfit(x1,s125,24);

% eval points for polyval
X1=0:.001:1;
X2=-1:.002:1;

% looping through other orders of polynomials
for n=2:25
    [x,w,xm,wm,sigma,s1] = legendrequad(n);

    [Pe1 ]=polyfit(xm,sigma,n-1);
    [Pe2 ]=polyfit(x,s1,n-1);
    error(n) =( sum( ((polyval(Pe1,X1)-polyval(P1,X1))).^2)*.001).^1/2;
    error1(n)=( sum( ((polyval(Pe2,X2)-polyval(P2,X2))).^2)*.002).^1/2;
end

```