**ASSIGNMENT B: APPROXIMATION OF FUNCTION**

**Author: Minh Hieu Do (ID: 288414)**

**Course: Numerical Methods (ENUME)**

**Advisor: Andrzej Miekina, Ph.D., Assistant Professor**

**Warsaw, 22/05/2018**

**Contents**

[1. The concise description of numerical algorithms 3](#_Toc514535833)

[a. Least-squares approximation of a function 3](#_Toc514535834)

[b. Legendre polynomials 3](#_Toc514535835)

[c. Accuracy indictors 3](#_Toc514535836)

[2. The methodology for testing numerical algorithms 4](#_Toc514535837)

[3. The results of testing numerical algorithms 5](#_Toc514535838)

[a. Problem 1 5](#_Toc514535839)

[b. Problem 2 6](#_Toc514535840)

[c. Problem 3 8](#_Toc514535841)

[d. Problem 4 9](#_Toc514535842)

[4. Conclusion 12](#_Toc514535843)

[List of references 13](#_Toc514535844)

[MATLAB Code 14](#_Toc514535845)

[Figure 1. Exact data for N = 5 5](#_Toc514535959)

[Figure 2. Exact data for N = 10 5](#_Toc514535960)

[Figure 3. Exact data for N = 15 6](#_Toc514535961)

[Figure 4. The approximation of f(x) for N = 10 and K = 5 6](#_Toc514535962)

[Figure 5. The approximation of f(x) for N = 20 and K = 15 7](#_Toc514535963)

[Figure 6. The approximation of f(x) for N = 40 and K = 35 7](#_Toc514535964)

[Figure 7. The dependence of on N and K 8](#_Toc514535965)

[Figure 8. The dependence of on N and K 8](#_Toc514535966)

[Figure 9. The dependence of on with the level of disturbance is 0.01 9](#_Toc514535967)

[Figure 10. The dependence of on with the level of disturbance is 0.01 9](#_Toc514535968)

[Figure 12. The dependence of on with the level of disturbance is 0.001 10](#_Toc514535969)

[Figure 13. The dependence of on with the level of disturbance is 0.0001 11](#_Toc514535970)

[Figure 14. The dependence of on with the level of disturbance is 0.0001 11](#_Toc514535971)

# The concise description of numerical algorithms

## Least-squares approximation of a function

A set of *n* discrete data points is given.

Consider the function

where is a linearly independent function (base function)

The coefficients are given by the solution to the equation

where

and

## Legendre polynomials

The set of *Legendre polynomials* is orthogonal on [-1, 1] with *w(x) = 1*

## Accuracy indictors

# The methodology for testing numerical algorithms

* Make the graphs of the function
* Design a MATLAB procedure using the method of least square and the *Legendre polynomials* as the base function to approximate the function on the basis of the data .
* Plot the graphs to compare the approximation to the exact data for several pairs of the values of the parameters N and K.
* Make the matrixes of and for K = 4, …, 40; N = k+2, …, 42.
* Plot 3D graphs of the dependence of and on K and N.
* Repeat the above study with the pseudorandom additive errors following the normal distribution with the zero mean and variance :

by using the MATLAB function ***randn*** to generate the errors.

# The results of testing numerical algorithms

## Problem 1

A close up of a person

Description generated with high confidence

Figure 1. Exact data for N = 5

A picture containing text

Description generated with very high confidence

Figure 2. Exact data for N = 10

A picture containing text, map

Description generated with very high confidence

Figure 3. Exact data for N = 15

## *Problem 2*

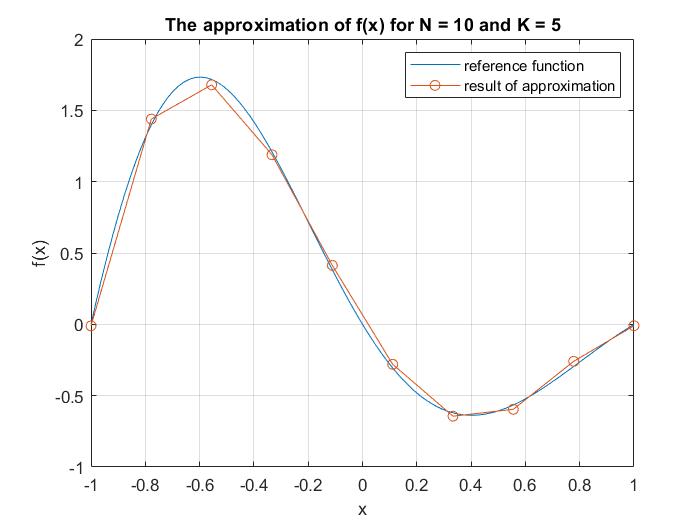


Figure 4. The approximation of f(x) for N = 10 and K = 5

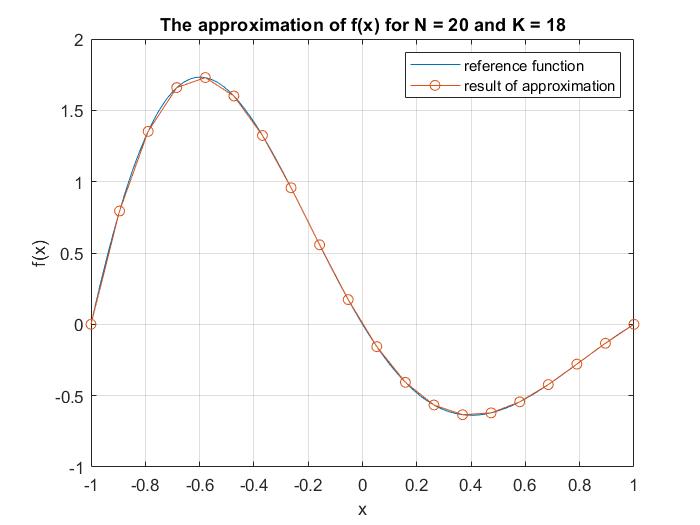


Figure 5. The approximation of f(x) for N = 20 and K = 15

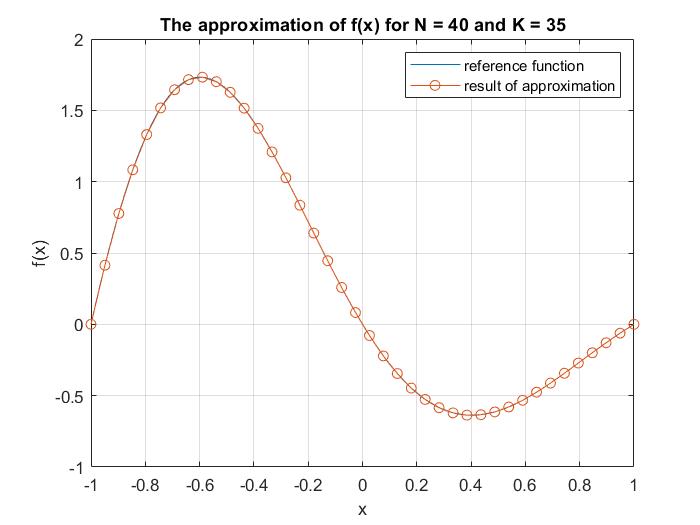


Figure 6. The approximation of f(x) for N = 40 and K = 35

## Problem 3

A close up of a map

Description generated with high confidence

Figure 7. The dependence of on N and KA close up of a map

Description generated with high confidence

Figure 8. The dependence of on N and K

## Problem 4

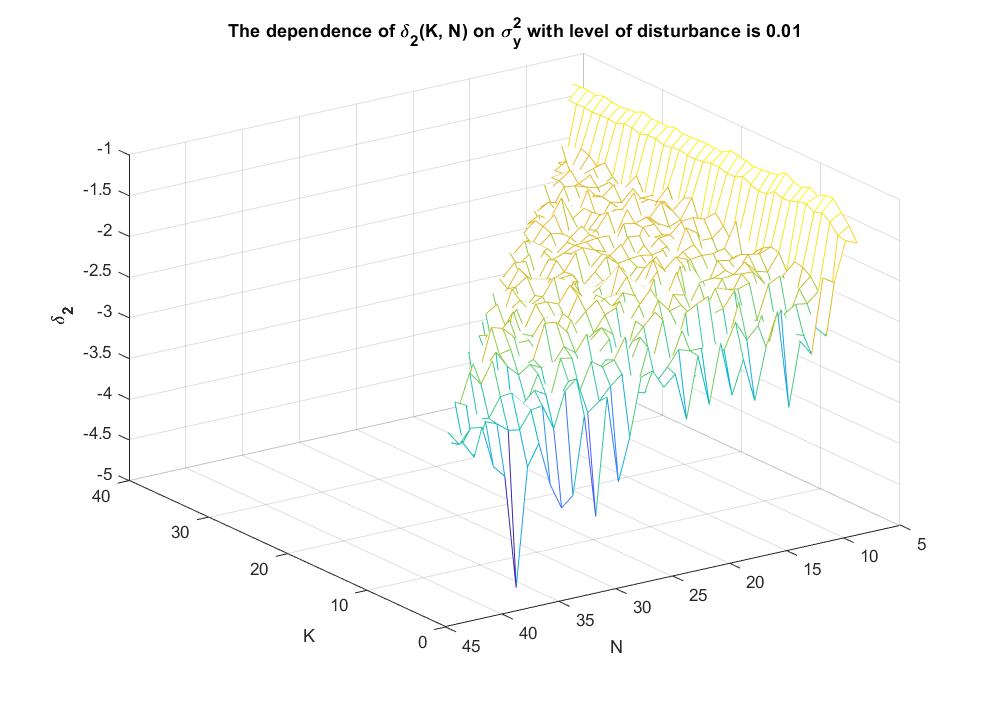


Figure 9. The dependence of on with the level of disturbance is 0.01

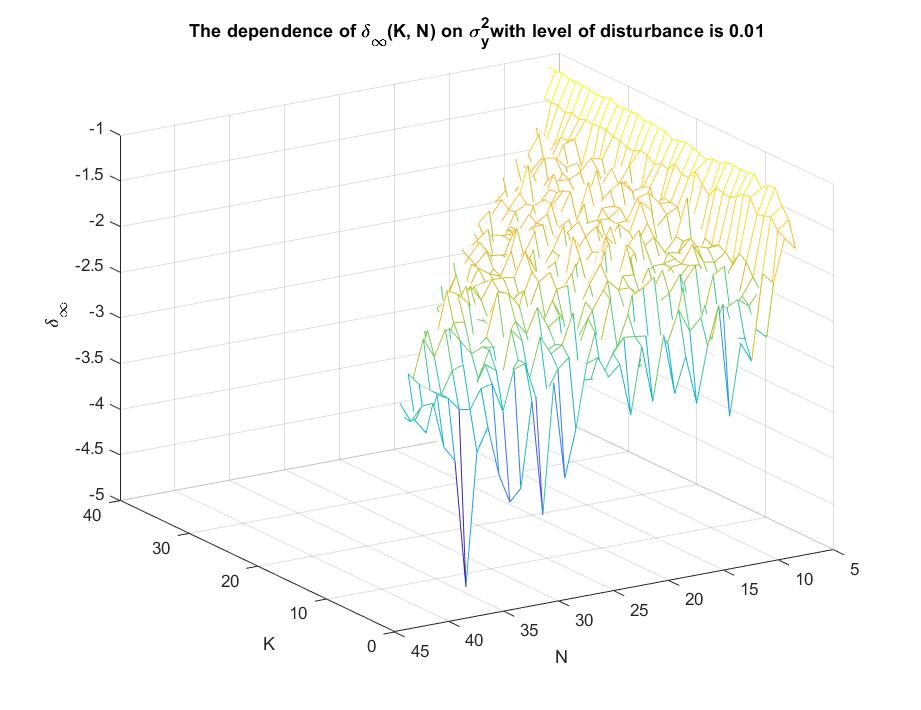


Figure 10. The dependence of on with the level of disturbance is 0.01

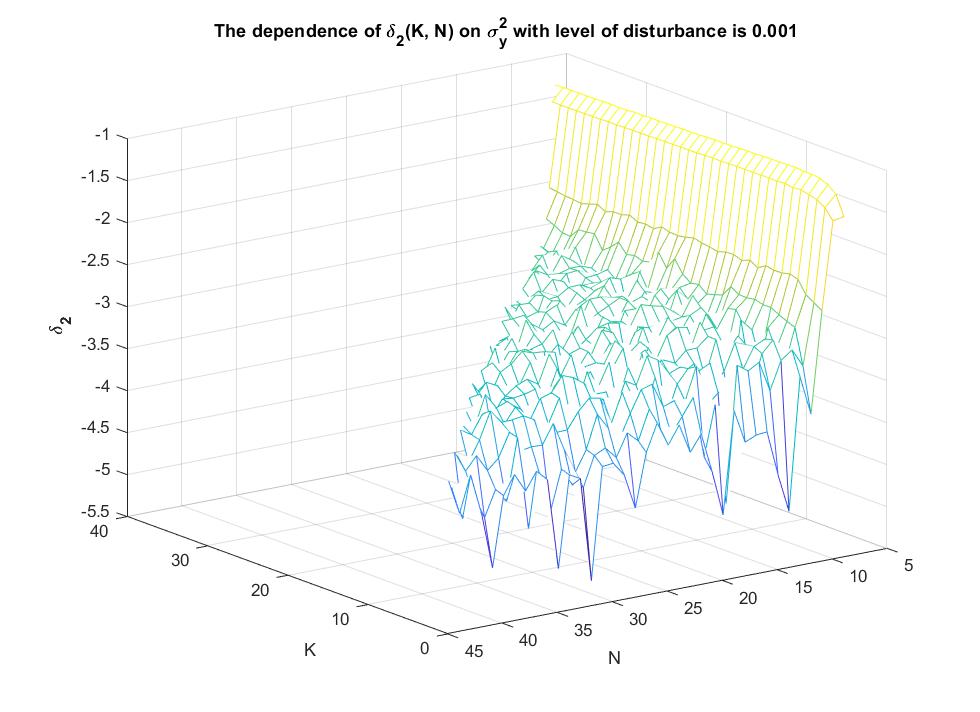


Figure 11. The dependence of on with the level of disturbance is 0.001

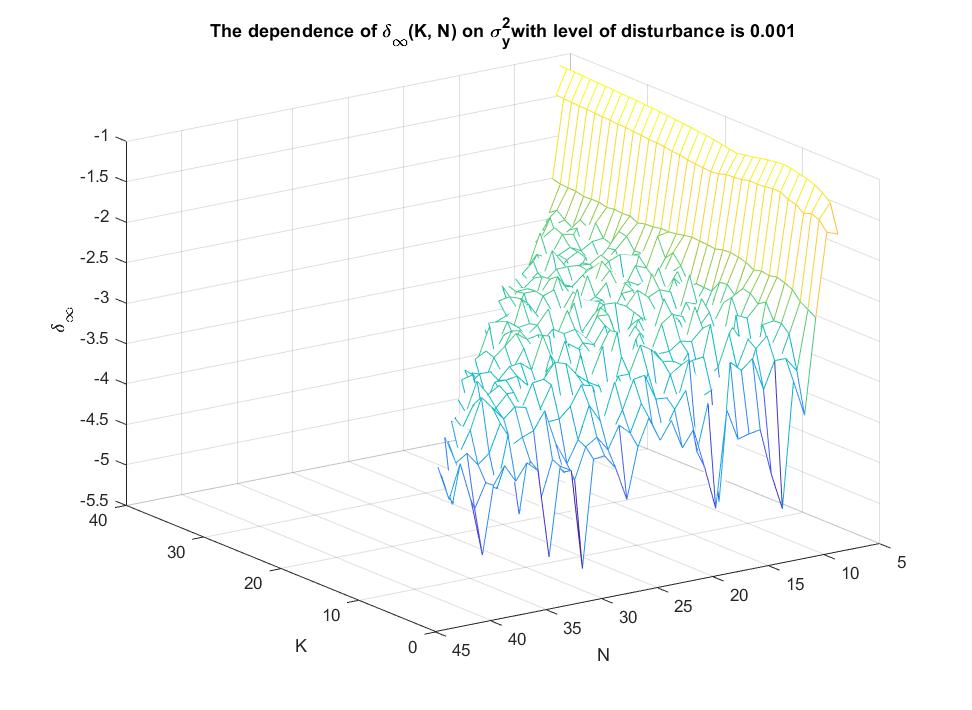
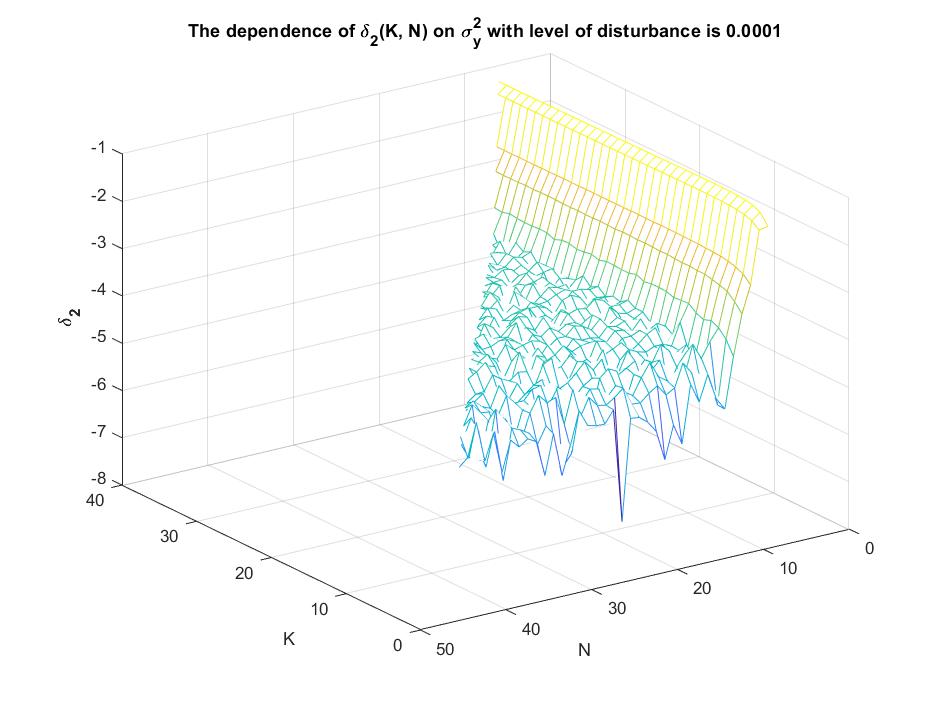


Figure 12. The dependence of on with the level of disturbance is 0.001



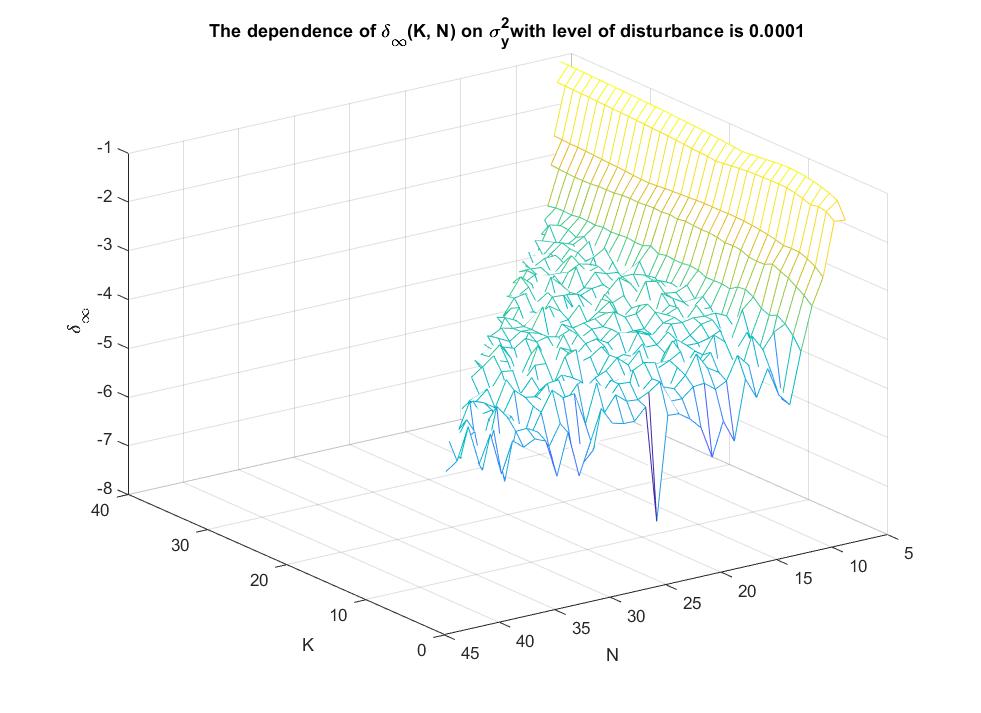
Figure 13. The dependence of on with the level of disturbance is 0.0001

Figure 14. The dependence of  on with the level of disturbance is 0.0001

# Conclusion

After using the method of least squares to approximate the function based on the discrete data, two numerical methods of solving the system of normal equations: Cholesky-Banachiewicz and MATLAB/Simulink built-in method, were used to solve and analyze the solutions of approximation. The obtained results are very close to each other. From three pairs of the values of the parameters N and K, it can be concluded that the higher N and K are, the more accurate the approximation is.

For the given function, with , both accuracy indicators and bottom at N around 25.

By analyzing the graphs of the dependence of and on , it can be seen that the lower the level of disturbance is, the more precise the approximation is.

# List of references

R. Z. Morawski, Lecture notes for ENUME students

A. Miękina, ENUME MatLab Intro 2018

MathWorks, MATLAB Documentation, <https://www.mathworks.com/help/index.html>

# MATLAB Code

clear all

close all

clc

f = @(x) -sin(pi\*x).\*exp(-x);

Ns = [5, 10, 15];

Ks = [2, 5, 8];

x1 = linspace(-1, 1, 100);

%%Problem 1

for i = 1 : 3

N = Ns(i);

y = f(x1);

figure (i)

plot(x1, y);

hold on

grid on

[x, y] = createXY(N);

plot(x, y, '\*');

xlabel('x');

ylabel('y');

title(['N = ', num2str(N)]);

end

Ns = [10, 20, 40];

Ks = [5, 18, 35];

%%Problem 2

for i = 1 : 3

N = Ns(i);

K = Ks(i);

[x, y, fxLS, fxLSChol] = LSsolve(K, N);

figure(i+3)

plot(x1, f(x1), x, fxLS, '-o')

legend('reference function', 'result of approximation')

grid on

xlabel('x')

ylabel('f(x)')

title(['The approximation of f(x) for N = ', num2str(N), ' and K = ', num2str(K)]);

end

%Problem 3

%Dependence of indicator on N and K

K = [4:40];

N = [6:42];

accuracy = zeros(length(K), length(N));

accuracyInf = zeros(length(K), length(N));

for k = 4 : 40

for n = k+2 : 42

[x, y, fxLS, ] = LSsolve(k, n);

accuracy(k-3, n-k-1) = norm(fxLS - y) / norm(y);

accuracyInf(k-3, n-k-1) = norm(fxLS - y, Inf) / norm(y, Inf);

end

end

figure(7)

mesh(K, N, log10(accuracy))

xlabel('K');

ylabel('N');

zlabel('\bf \delta\_{2}');

title('The dependence of \delta\_{2}(K, N) on N and K');

grid on

figure(8)

mesh(K, N, log10(accuracyInf))

xlabel('K');

ylabel('N');

zlabel('\bf \delta\_{\infty}');

title('The dependence of \delta\_{\infty}(K, N) on N and K');

grid on

%Problem 4

err = [0.01, 0.001, 0.0001];

accuracy = zeros(length(K), length(N));

accuracyInf = zeros(length(K), length(N));

variance = zeros(length(K), length(N));

for i = 1 : 3

lvDisturbance = err(i);

for k = 4 : 40

for n = k+2 : 42

[x, y, fxLS, ] = LSsolveErr(k, n, lvDisturbance);

accuracy(k-3, n-k-1) = norm(fxLS - y) / norm(y);

accuracyInf(k-3, n-k-1) = norm(fxLS - y, Inf) / norm(y, Inf);

variance(k-3, n-k-1) = var(y);

end

end

figure(2\*i + 7)

mesh(K, N, log10(accuracy))

xlabel('K');

ylabel('N');

zlabel('\bf \delta\_{2}');

title(['The dependence of \delta\_{2}(K, N) on \sigma\_{y}^{2} with level of disturbance is ', num2str(lvDisturbance)]);

grid on

figure(2\*i + 8)

mesh(K, N, log10(accuracyInf))

xlabel('K');

ylabel('N');

zlabel('\bf \delta\_{\infty}');

title(['The dependence of \delta\_{\infty}(K, N) on \sigma\_{y}^{2}with level of disturbance is ', num2str(lvDisturbance)]);

grid on

end

%%Function to solve approximation

function[x, y, fxLS, fxLSChol] = LSsolve(K, N)

[x, y] = createXY(N);

y = y';

P = createBase(K, N, x);

res = (P' \* P) \ (P' \* y);

resCB = solveCB(P' \* P, P' \* y);

fxLS = P \* res;

fxLSChol = P \* resCB;

end

%%Function to solve approximation with error

function[x, y, fxLS, fxLSChol] = LSsolveErr(K, N, err)

[x, y] = createXY(N);

y = y';

yErr = randn(N,1) \* err;

y = y .\* (1 + yErr);

P = createBase(K, N, x);

res = (P' \* P) \ (P' \* y);

resCB = solveCB(P' \* P, P' \* y);

fxLS = P \* res;

fxLSChol = P \* resCB;

end

%%Function to create data {(xn, yn)|n = 1, ..., N}-------------

function [X, Y] = createXY(N)

f = @(x) -sin(pi\*x).\*exp(-x);

for n = 1 : N

X(n) = -1 + 2\*(n-1)/(N-1);

end

Y = f(X);

end

%%Function to creaatebase---------------------------------------

function P = createBase(K, N, x)

for n = 1 : N

P(n, 1) = 1;

P(n, 2) = x(n);

for j = 3 : K+1

k = j-1;

P(n, j) = (2\*k-1)/k \* x(n) \* P(n, j-1) - (k-1)/k \* P(n, j-2);

end

end

end

%%Cholesky function----------------------------------------------

function [L] = Cholesky(A)

N = length(A);

L = A-A;

for i = 1 : N

L(i, i) = sqrt( A(i, i) - L(i, :)\*L(i, :)' );

for j = (i + 1) : N

L(j, i) = ( A(j, i) - L(i, :)\*L(j, :)' )/L(i, i);

end

end

end

%%Function to sovle linear system---------------------------------------

function [X] = solveCB(A, b)

L = Cholesky(A);

Lt = L';

[n , ~] = size(A);

y = zeros(n, 1);

X = zeros(n, 1);

y(1) = b(1)/L(1, 1);

for i = 2 : n

y(i) = (b(i) - L(i, :)\*y)/L(i, i);

end

X(n) = y(n)/Lt(n, n);

for i = n-1 : -1 : 1

X(i) = (y(i) - Lt(i, :)\*X)/L(i, i);

end

end