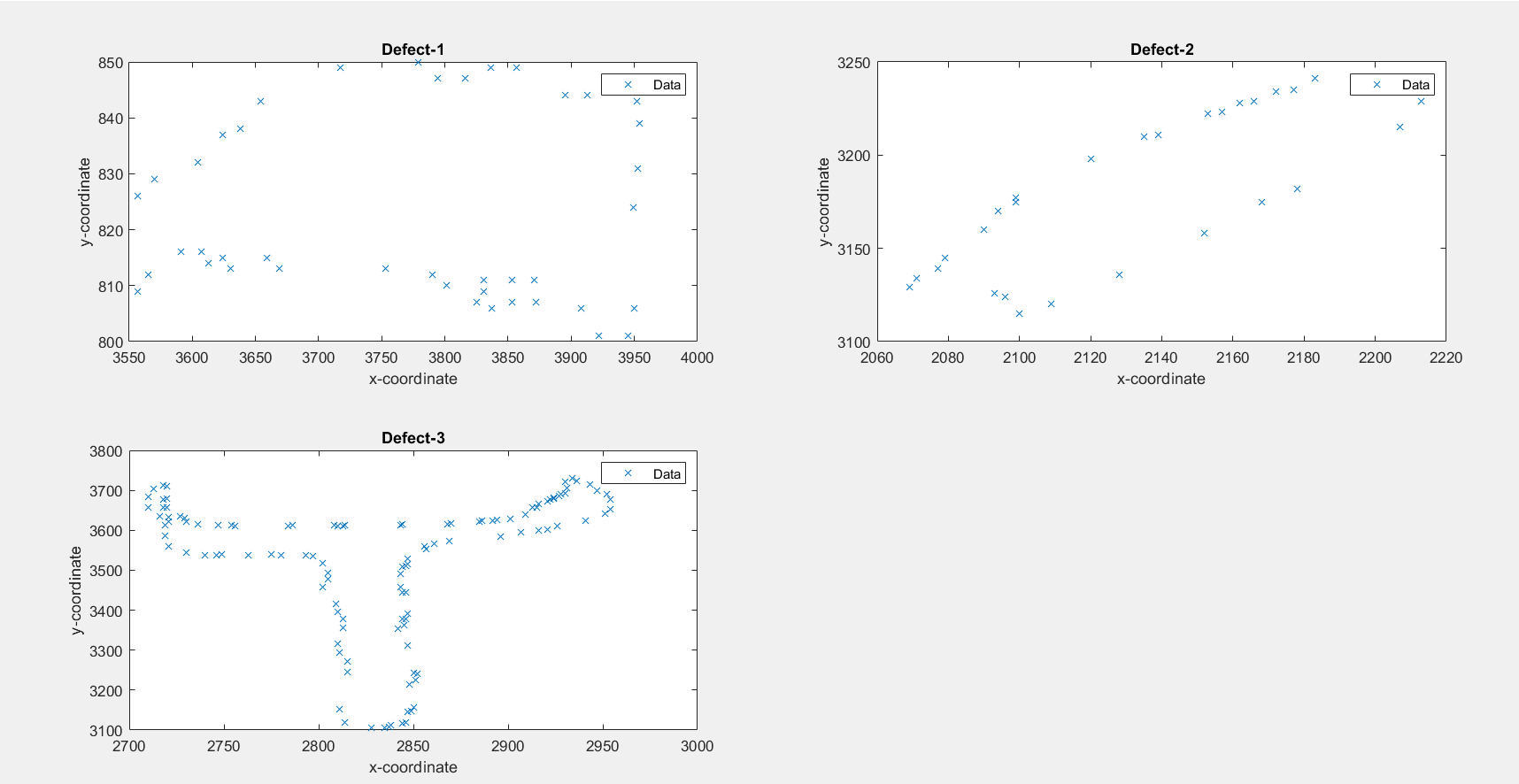
Plots

# Question-4 plots



The above plots agree with the inferences concluded using OLS/TLS.

Defect-1: along horizontal axis

Defect-2: along y = x

Defect-3: along vertical axis

Code

# OLS.m (function to perform OLS)

function [alpha, beta, uhat, yhat, s] = OLS(u,y)

N = length(u);

ybar = mean(y);

ubar = mean(u);

syy = var(y,1);

suu = var(u,1);

syu = 1/N\*sum((y-ybar).\*(u-ubar));

uhat = u;

alpha = syu/suu;

beta = ybar - alpha\*ubar;

yhat = alpha\*uhat + beta;

s = struct('alpha',alpha,'beta',beta,'ybar',ybar,'ubar',ubar,'syy',syy,'syu',syu,'suu',suu);

end

# IOLS.m (function to perform IOLS)

function [alpha, beta, uhat, yhat, s] = IOLS(u,y)

N = length(u);

ybar = mean(y);

ubar = mean(u);

syy = var(y,1);

suu = var(u,1);

syu = 1/N\*sum((y-ybar).\*(u-ubar));

yhat = y;

alpha = syy/syu;

beta = ybar - alpha\*ubar;

uhat = (yhat - beta)/alpha;

s = struct('alpha',alpha,'beta',beta,'ybar',ybar,'ubar',ubar,'syy',syy,'syu',syu,'suu',suu);

end

# TLS.m (function to perform TLS)

function [alpha, beta, uhat, yhat, s] = TLS(u,y)

N = length(u);

ybar = mean(y);

ubar = mean(u);

syy = var(y,1);

suu = var(u,1);

syu = 1/N\*sum((y-ybar).\*(u-ubar));

alpha = (syy - suu + sqrt((syy-suu)^2 + 4\*syu^2))/2/syu;

beta = ybar - alpha\*ubar;

uhat = (alpha\*(y-beta)+u)/(alpha^2+1);

yhat = alpha\*uhat + beta;

s = struct('alpha',alpha,'beta',beta,'ybar',ybar,'ubar',ubar,'syy',syy,'syu',syu,'suu',suu);

end

# Question-2

clear; close all;

%% Open data

A = readmatrix('CO2.csv');

time = A(:,1);

CO2 = A(:,2);

temp = A(:,3);

temp\_cut = 3.6;

%% Predict maximum permissible level of CO2 - OLS

% y -> temperature deviation

% u -> CO2

u = CO2;

y = temp;

[alpha, beta, uhat, yhat, s] = OLS(u,y);

disp(s);

CO2\_cut\_OLS = (temp\_cut-beta)/alpha;

%% Predict maximum permissible level of CO2 - TLS

[alpha\_TLS, beta\_TLS, uhat\_TLS, yhat\_TLS,s\_TLS] = TLS(u,y);

CO2\_cut\_TLS = (temp\_cut-beta\_TLS)/alpha\_TLS;

%% Predict year

ut = time;

yt = CO2;

[alpha\_t, beta\_t, uhat\_t, yhat\_t, s\_t] = OLS(ut,yt);

t\_OLS = (CO2\_cut\_OLS-beta\_t)/alpha\_t;

tpred\_OLS = ceil(t\_OLS);

t\_TLS = (CO2\_cut\_TLS-beta\_t)/alpha\_t;

tpred\_TLS = ceil(t\_TLS);

# Question-3

close all; clear;

%% Data

EP = [1.98,2.31,3.29,3.56,1.23,1.57,2.05,0.66,0.31,2.82,0.13,3.15,2.72,2.31,1.92,1.56,0.94,2.27,3.17,2.36]';

CF = [1.87,2.2,3.15,3.42,1.1,1.41,1.84,0.68,0.27,2.8,0.14,3.2,2.7,2.43,1.78,1.53,0.84,2.21,3.10,2.34]';

% u - EP, y - CF

u = EP;

y = CF;

N = length(u);

%% Part a)

% OLS

[alpha\_OLS, beta\_OLS, uhat\_OLS, yhat\_OLS,s\_OLS] = OLS(u,y);

sigma\_e\_OLS = 1/(N-2)\*sum((y-alpha\_OLS\*u-beta\_OLS).^2);

CI\_OLS = [alpha\_OLS-2.16\*sigma\_e\_OLS,alpha\_OLS+2.16\*sigma\_e\_OLS];

% IOLS

[alpha\_IOLS, beta\_IOLS, uhat\_IOLS, yhat\_IOLS,s\_IOLS] = IOLS(u,y);

sigma\_e\_IOLS = 1/(N-2)\*sum((y-alpha\_IOLS\*u-beta\_IOLS).^2);

CI\_IOLS = [alpha\_IOLS-2.16\*sigma\_e\_IOLS,alpha\_OLS+2.16\*sigma\_e\_IOLS];

% TLS

[alpha\_TLS, beta\_TLS, uhat\_TLS, yhat\_TLS,s\_TLS] = TLS(u,y);

sigma\_e\_TLS = 1/(N-2)\*sum((y-alpha\_TLS\*u-beta\_TLS).^2);

CI\_TLS = [alpha\_TLS-2.16\*sigma\_e\_TLS,alpha\_TLS+2.16\*sigma\_e\_TLS];

%% Part b)

ui = 2.31;

yi = 2.20;

% OLS: u is perfect

OLS\_pred = ui;

% IOLS: y is perfect

IOLS\_pred = yi;

% TLS: both imperfect. doing a perpendicular projection will give us an

% estimate for EP and CF. Since we need a single estimate we will assume

% alpha = 1 and beta = 0. So yhat = uhat = (u+y)/2

TLS\_pred = (ui+yi)/2;

# Question-4

clear; close all;

%% Open data

A = readmatrix('defects\_annotation\_data.csv');

x1 = rem\_NaN(A(:,1)); y1 = rem\_NaN(A(:,2));

x2 = rem\_NaN(A(:,4)); y2 = rem\_NaN(A(:,5));

x3 = rem\_NaN(A(:,7)); y3 = rem\_NaN(A(:,8));

%% Defect-1

N = length(x1);

% TLS

[alpha\_TLS, beta\_TLS, uhat\_TLS, yhat\_TLS,s\_TLS] = TLS(x1,y1);

sigma\_e\_TLS = 1/(N-2)\*sum((y1-alpha\_TLS\*x1-beta\_TLS).^2);

CI\_TLS = [alpha\_TLS-2.16\*sigma\_e\_TLS,alpha\_TLS+2.16\*sigma\_e\_TLS];

% TLS - inverted

[alpha\_TLS2, beta\_TLS2, uhat\_TLS2, yhat\_TLS2,s\_TLS2] = TLS(y1,x1);

sigma\_e\_TLS2 = 1/(N-2)\*sum((y1-alpha\_TLS2\*x1-beta\_TLS2).^2);

CI\_TLS2 = [alpha\_TLS2-2.16\*sigma\_e\_TLS2,alpha\_TLS2+2.16\*sigma\_e\_TLS2];

% OLS

[alpha\_OLS, beta\_OLS, uhat\_OLS, yhat\_OLS,s\_OLS] = OLS(x1,y1);

sigma\_e\_OLS = 1/(N-2)\*sum((y1-alpha\_OLS\*x1-beta\_OLS).^2);

CI\_OLS = [alpha\_OLS-2.16\*sigma\_e\_OLS,alpha\_OLS+2.16\*sigma\_e\_OLS];

% OLS - inverted

[alpha\_OLS2, beta\_OLS2, uhat\_OLS2, yhat\_OLS2,s\_OLS2] = OLS(y1,x1);

sigma\_e\_OLS2 = 1/(N-2)\*sum((y1-alpha\_OLS2\*x1-beta\_OLS2).^2);

CI\_OLS2 = [alpha\_OLS2-2.16\*sigma\_e\_OLS2,alpha\_OLS2+2.16\*sigma\_e\_OLS2];

% Plot

subplot(2,2,1);

plot(x1,y1,'x');

title('Defect-1'); xlabel('x-coordinate'); ylabel('y-coordinate');

legend('Data');

%% Defect-2

% TLS

[alpha\_TLS\_def2, beta\_TLS\_def2, uhat\_TLS\_def2, yhat\_TLS\_def2,s\_TLS\_def2] = TLS(x2,y2);

sigma\_e\_TLS\_def2 = 1/(N-2)\*sum((y2-alpha\_TLS\_def2\*x2-beta\_TLS\_def2).^2);

CI\_TLS\_def2 = [alpha\_TLS\_def2-2.16\*sigma\_e\_TLS\_def2,alpha\_TLS\_def2+2.16\*sigma\_e\_TLS\_def2];

% TLS - inverted

[alpha\_TLS2\_def2, beta\_TLS2\_def2, uhat\_TLS2\_def2, yhat\_TLS2\_def2,s\_TLS2\_def2] = TLS(y2,x2);

sigma\_e\_TLS2\_def2 = 1/(N-2)\*sum((y2-alpha\_TLS2\_def2\*x2-beta\_TLS2\_def2).^2);

CI\_TLS2\_def2 = [alpha\_TLS2\_def2-2.16\*sigma\_e\_TLS2\_def2,alpha\_TLS2\_def2+2.16\*sigma\_e\_TLS2\_def2];

% OLS

[alpha\_OLS\_def2, beta\_OLS\_def2, uhat\_OLS\_def2, yhat\_OLS\_def2,s\_OLS\_def2] = OLS(x2,y2);

sigma\_e\_OLS\_def2 = 1/(N-2)\*sum((y2-alpha\_OLS\_def2\*x2-beta\_OLS\_def2).^2);

CI\_OLS\_def2 = [alpha\_OLS\_def2-2.16\*sigma\_e\_OLS\_def2,alpha\_OLS\_def2+2.16\*sigma\_e\_OLS\_def2];

% OLS - inverted

[alpha\_OLS2\_def2, beta\_OLS2\_def2, uhat\_OLS2\_def2, yhat\_OLS2\_def2,s\_OLS2\_def2] = OLS(y2,x2);

sigma\_e\_OLS2\_def2 = 1/(N-2)\*sum((y2-alpha\_OLS2\_def2\*x2-beta\_OLS2\_def2).^2);

CI\_OLS2\_def2 = [alpha\_OLS2\_def2-2.16\*sigma\_e\_OLS2\_def2,alpha\_OLS2\_def2+2.16\*sigma\_e\_OLS2\_def2];

% Plot

subplot(2,2,2);

plot(x2,y2,'x');

title('Defect-2'); xlabel('x-coordinate'); ylabel('y-coordinate');

legend('Data');

%% Defect-3

% TLS

[alpha\_TLS\_def3, beta\_TLS\_def3, uhat\_TLS\_def3, yhat\_TLS\_def3,s\_TLS\_def3] = TLS(x3,y3);

sigma\_e\_TLS\_def3 = 1/(N-2)\*sum((y3-alpha\_TLS\_def3\*x3-beta\_TLS\_def3).^2);

CI\_TLS\_def3 = [alpha\_TLS\_def3-2.16\*sigma\_e\_TLS\_def3,alpha\_TLS\_def3+2.16\*sigma\_e\_TLS\_def3];

% TLS - inverted

[alpha\_TLS2\_def3, beta\_TLS2\_def3, uhat\_TLS2\_def3, yhat\_TLS2\_def3,s\_TLS2\_def3] = TLS(y3,x3);

sigma\_e\_TLS2\_def3 = 1/(N-2)\*sum((y3-alpha\_TLS2\_def3\*x3-beta\_TLS2\_def3).^2);

CI\_TLS2\_def3 = [alpha\_TLS2\_def3-2.16\*sigma\_e\_TLS2\_def3,alpha\_TLS2\_def3+2.16\*sigma\_e\_TLS2\_def3];

% OLS

[alpha\_OLS\_def3, beta\_OLS\_def3, uhat\_OLS\_def3, yhat\_OLS\_def3,s\_OLS\_def3] = OLS(x3,y3);

sigma\_e\_OLS\_def3 = 1/(N-2)\*sum((y3-alpha\_OLS\_def3\*x3-beta\_OLS\_def3).^2);

CI\_OLS\_def3 = [alpha\_OLS\_def3-2.16\*sigma\_e\_OLS\_def3,alpha\_OLS\_def3+2.16\*sigma\_e\_OLS\_def3];

% OLS - inverted

[alpha\_OLS2\_def3, beta\_OLS2\_def3, uhat\_OLS2\_def3, yhat\_OLS2\_def3,s\_OLS2\_def3] = OLS(y3,x3);

sigma\_e\_OLS2\_def3 = 1/(N-2)\*sum((y2-alpha\_OLS2\_def3\*x2-beta\_OLS2\_def3).^2);

CI\_OLS2\_def3 = [alpha\_OLS2\_def3-2.16\*sigma\_e\_OLS2\_def3,alpha\_OLS2\_def3+2.16\*sigma\_e\_OLS2\_def3];

% Plot

subplot(2,2,3);

plot(x3,y3,'x');

title('Defect-3'); xlabel('x-coordinate'); ylabel('y-coordinate');

legend('Data');

%% function to remove NaN values

function vnew = rem\_NaN(v)

vnew = v(~isnan(v));

end