HW1

Ruijie Song

Feb.4.2021

Exercise 1: Histogram and Cross-Validation

a)

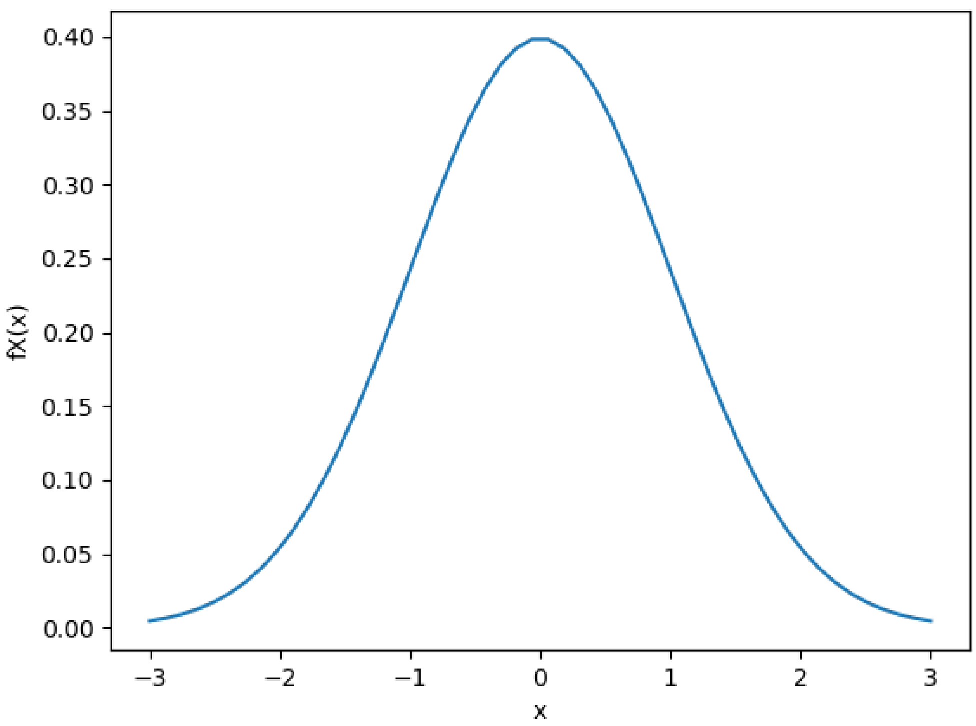


Figure 1. fX

b)
ii)

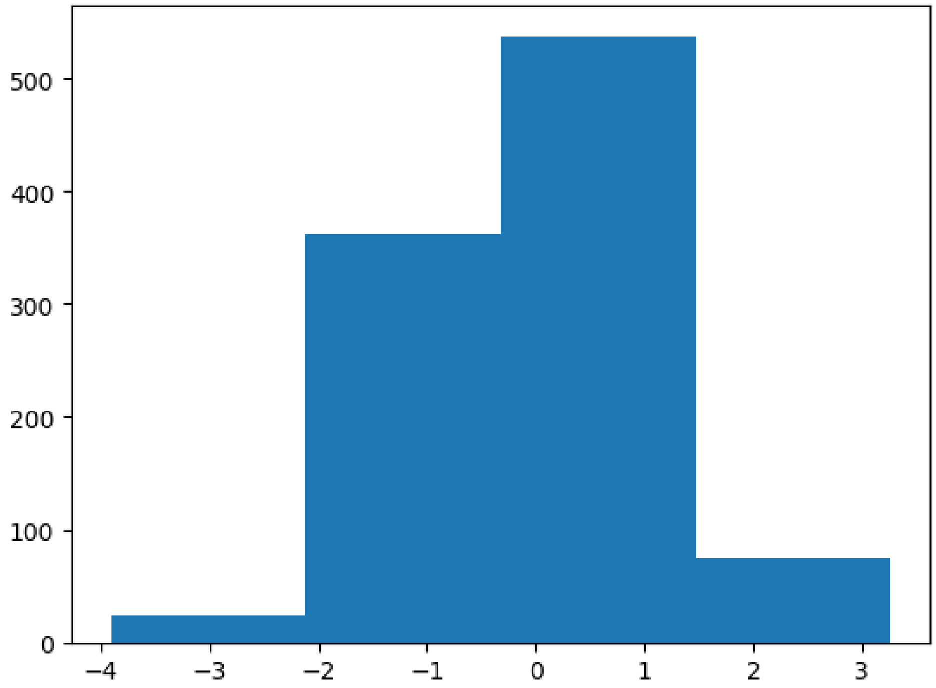


Figure 2. m=4

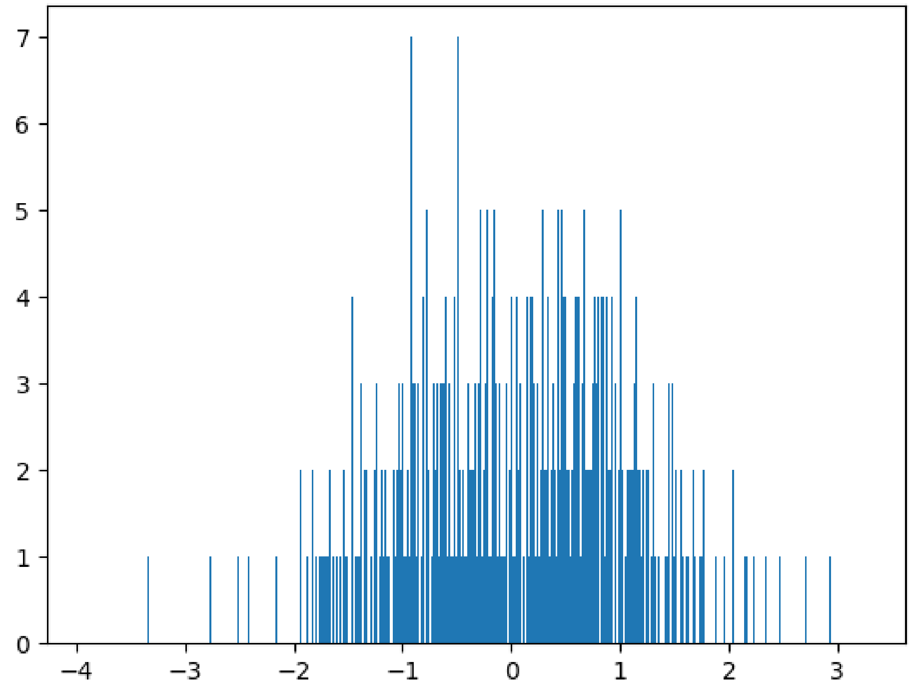


Figure 3. m=1000

iii)
mean = -0.021858244152l9112
standard deviation = 1.0469359084378456

iv)

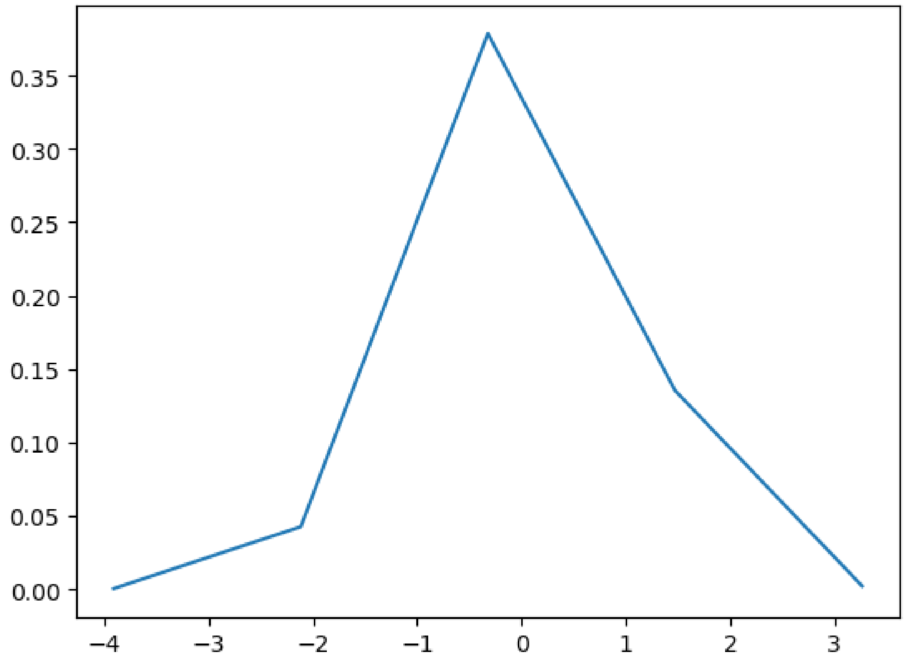


Figure 4. m=4

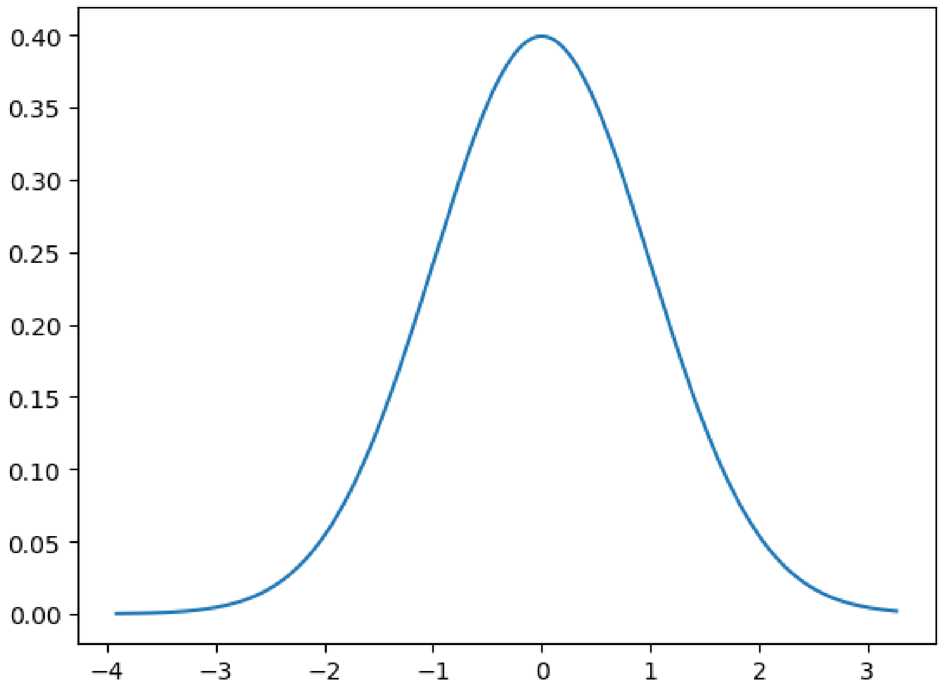


Figure 5. m=1000

c)
i)

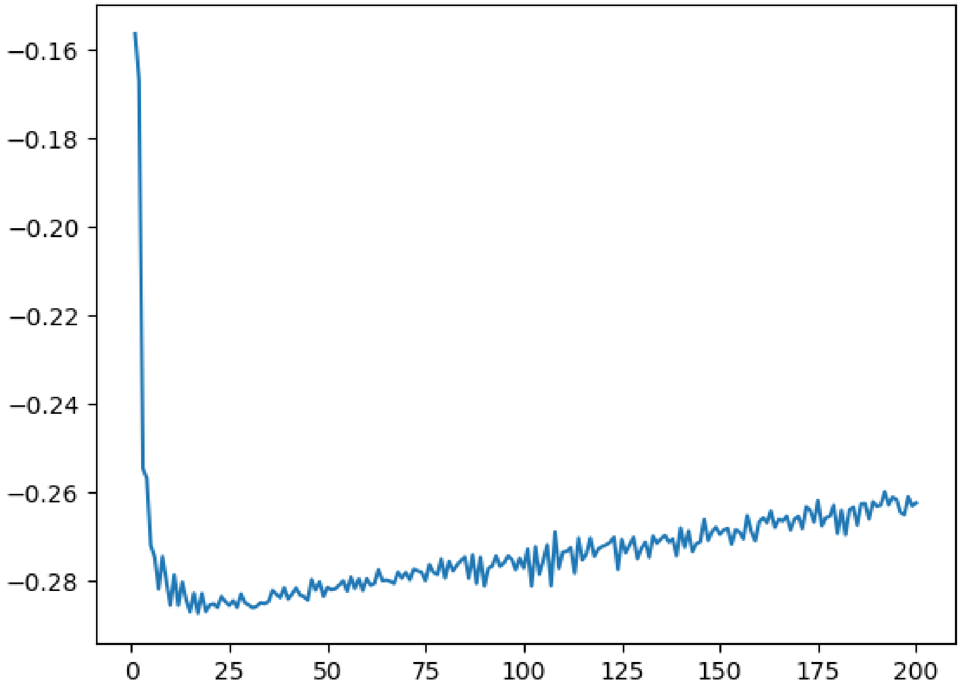


Figure 6. J(h)

ii)

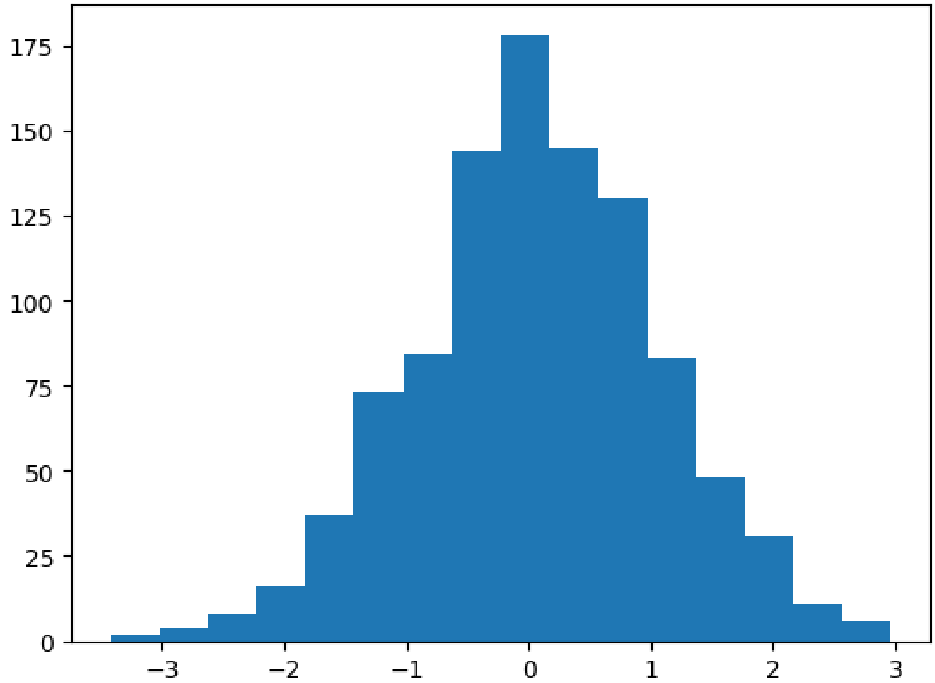


Figure 7. histogram of your data with that m∗

m* = 16
iii)

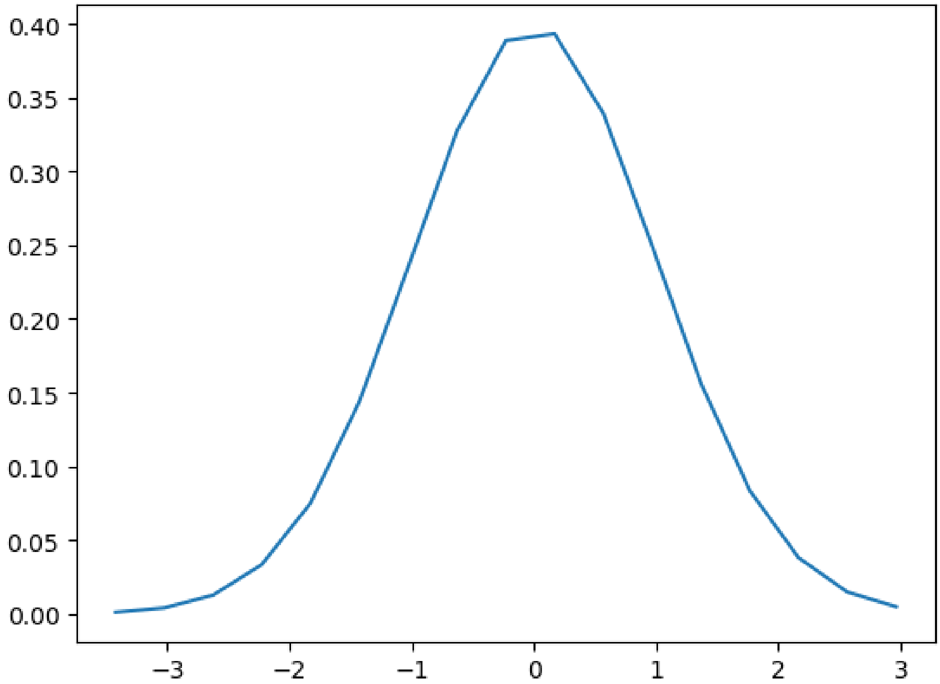


Figure 8. Gaussian curve

Exercise 2: Gaussian Whitening

a)

i)

2.

a) $X \sim N(\mu, \Sigma)$ $\quad f_X(x) = \frac{1}{\sqrt{(2\pi)^2 |\Sigma|}} \exp\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\}$

$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \mu = \begin{bmatrix} 2 \\ 6 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

i) $|\Sigma| = 4 - 1 = 3 \quad x - \mu = \begin{bmatrix} x_1 - 2 \\ x_2 - 6 \end{bmatrix}$

$\Sigma^{-1} = \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix}$

$\therefore f_X(x) = \frac{1}{\sqrt{(2\pi)^2 3}} \exp\{-\frac{1}{2} [x_1 - 2, x_2 - 6] \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix} \begin{bmatrix} x_1 - 2 \\ x_2 - 6 \end{bmatrix}\}$

$= \frac{1}{\sqrt{(2\pi)^2 3}} \exp\{\frac{2x_1^2 + 4x_1 - 2x_1 x_2 + 2x_2^2 + 56 - 20x_2}{-6}\}$

ii)

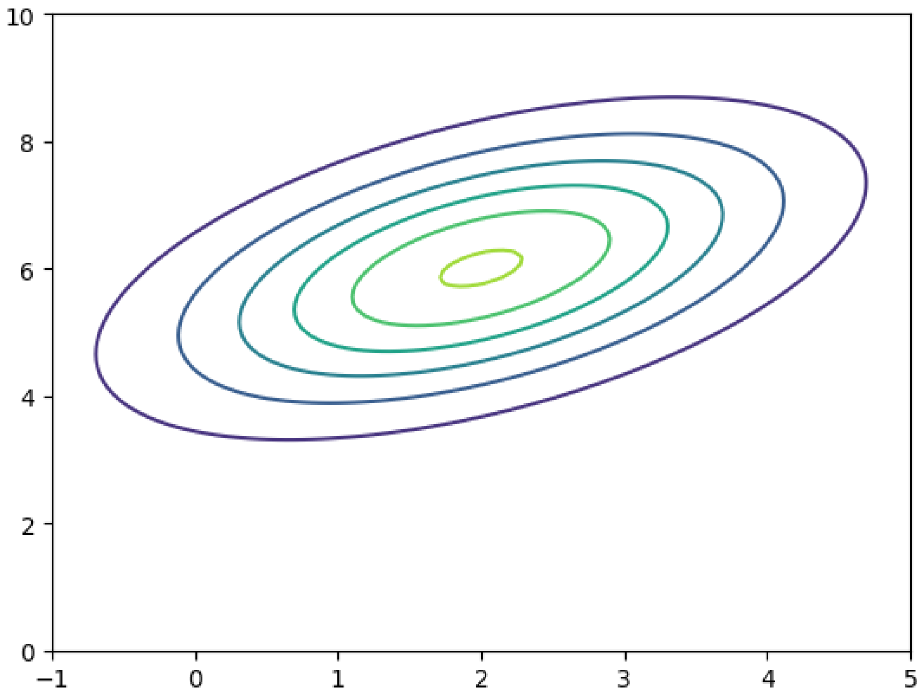


Figure 9. The contour of fX

b)

b) $A \in \mathbb{R}^{d\times d}$ $\quad b \in \mathbb{R}^{d}$ $\quad Y = AX + b$ $\quad X$: r.v $\sim N(0, I)$

$\mu_Y \overset{def}{=} E[Y]$ $\quad \Sigma_Y \overset{def}{=} E[(Y-\mu_Y)(Y-\mu_Y)^T]$

i) $E[Y] = E[AX + b] = A\cdot \underbrace{E[X]}_{0} + b = b = \mu_Y$

$\Sigma_Y = E[(AX + \cancel{b} - \cancel{b})(AX + \cancel{b} - \cancel{b})^T] = E[(AX)(AX)^T]$

$= E(AXX^TA^T) = A\,E[XX^T]\,A^T = AA^T$

ii) symmetric if $\Sigma = \Sigma^T$, positive semi-definite if $x^T\Sigma x \geq 0$ for any $x \in \mathbb{R}^n$

$\cancel{AA^T}$ $\Sigma_Y = AA^T$ must be symmetric since $A \in \mathbb{R}^{d\times d}$

Let x be a non-zero vector

$x^T\Sigma x = x^TA^TAx = (Ax)^T(Ax) = \|Ax\|^2 \geq 0.$

$\therefore$ Σ is symmetric positive semi-definite when $A \in \mathbb{R}^{d\times d}$

iii) $x^T\Sigma x > 0$ for $x \in \mathbb{R}^n$ when $\|Ax\| \neq 0$

iv) $Y \sim N(\mu_Y, \Sigma_Y)$ $\quad \mu_Y = \begin{bmatrix}2\\6\end{bmatrix}$, $\Sigma_Y = \begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}$

$b = \mu_Y = \begin{bmatrix}2\\6\end{bmatrix}$ $\quad AA^T = \begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}$

$A = \begin{bmatrix}a & b\\c & d\end{bmatrix}$ $\quad \begin{bmatrix}a & b\\c & d\end{bmatrix}^{-1}\begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}\begin{bmatrix}a & b\\c & d\end{bmatrix}^T = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$

$\begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}\begin{bmatrix}a & b\\c & d\end{bmatrix} = \begin{bmatrix}a & b\\c & d\end{bmatrix}\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} = \begin{bmatrix}a & b\\c & d\end{bmatrix}$

$\begin{bmatrix}2a+c & 2b+d\\a+2c & b+2d\end{bmatrix} = \begin{bmatrix}a & b\\c & d\end{bmatrix}$ $\quad \begin{cases}\cancel{2}a+c = \cancel{a}\ 0\\ \cancel{2}b+d = \cancel{b}\\ a+\cancel{2}c = \cancel{c}\\ b+\cancel{2}d = \cancel{d}\ 0\end{cases}$

$a+c=0$ $\quad a=-c$

$b+d=0$ $\quad b=-d$ $\quad \therefore A = \begin{bmatrix}a & b\\-a & -b\end{bmatrix}$

$U = \begin{bmatrix}-\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2}\\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2}\end{bmatrix}$ $\quad \Sigma = \begin{bmatrix}1 & 0\\0 & 3\end{bmatrix}$

$\Sigma^{\frac{1}{2}} = A = U\Lambda^{\frac{1}{2}}U^T = \begin{bmatrix}1 & \frac{1}{2}\\ \frac{1}{2} & 1\end{bmatrix}$

c)
i)

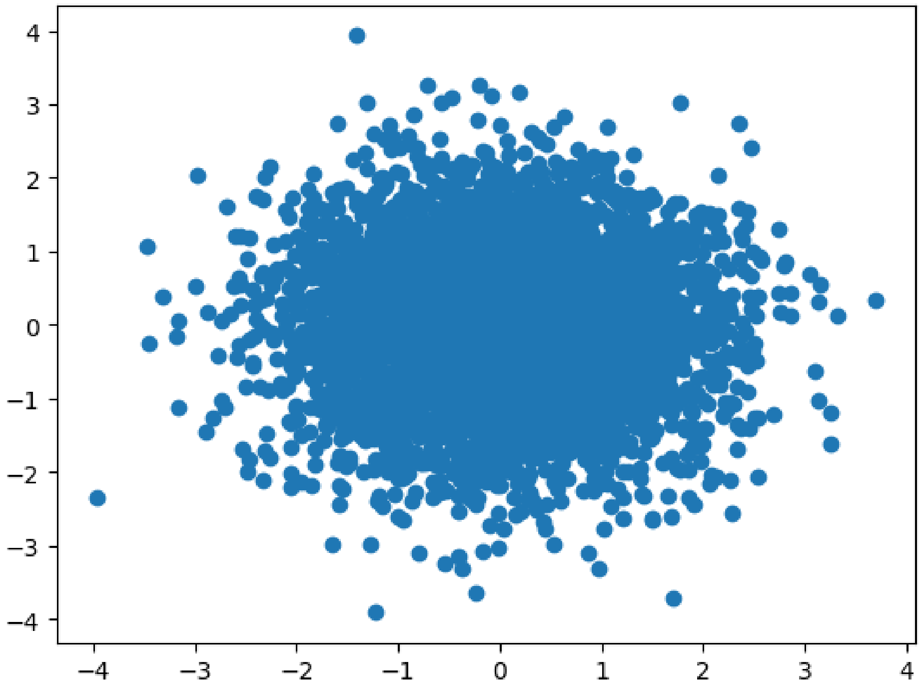


Figure 10. scatter plot

ii)

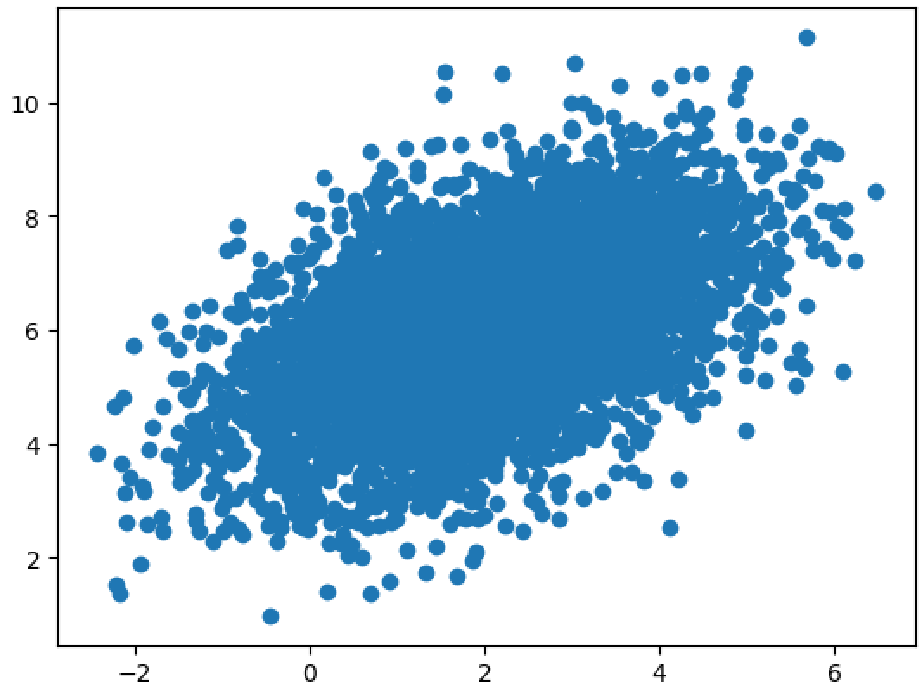


Figure 11. data calculated by numpy.random.multivariate_normal

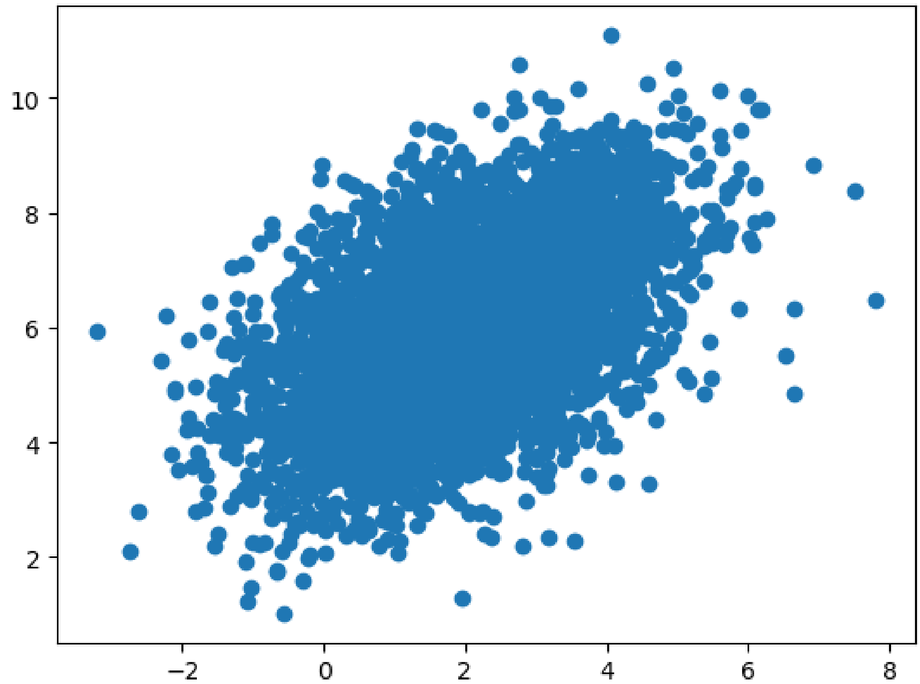
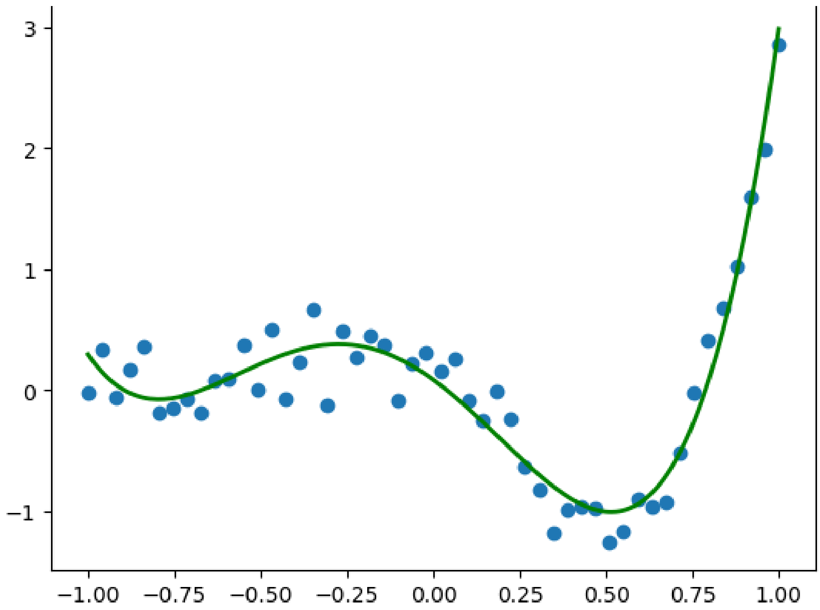


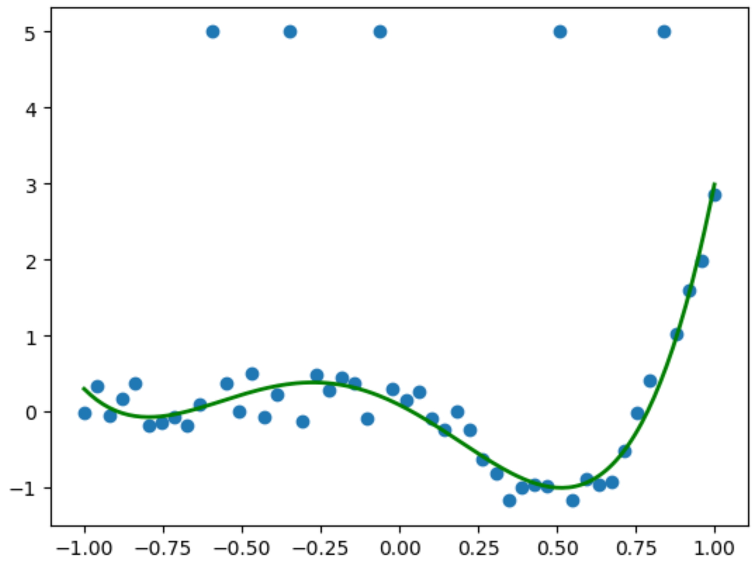
Figure 12. data calculated by affine transformation

Exercise 3: Linear Regression

a & c.



d.



There are a few differences between those 2 plot. However, the outliers do not affect much.

b & e.

3. b) $\hat{\beta} = \underset{\beta}{\arg\min} \, \|y - X\beta\|^2$

$$\beta = (X^T X)^{-1} X^T \underline{y}$$

X: data

$$y = g_{\beta}(X) = \beta^T X$$

e) $\hat{\beta} = \underset{\beta}{\arg\min} \, \|y - X\beta\|_1$

$$\Downarrow$$

$$\min_{\beta, \cancel{\{u_n\}}} \sum_{n=1}^{N} |y - X\beta|$$

~~s.t. $u_n \geq -(y - X\beta)$~~

~~$u_n \geq y - X\beta$~~

$$\Downarrow$$

$$\min_{\beta, \{u_n\}} \sum_{n=1}^{N} u_n \quad \text{s.t.} \quad u_n = |y - X\beta|$$

$$\Downarrow$$

$$\min_{\beta, \{u_n\}} \sum_{n=1}^{N} u_n \quad \text{s.t.} \quad \begin{aligned} u_n &\geq -(y - X\beta) \\ u_n &\geq (y - X\beta) \end{aligned}$$

$$\Downarrow$$

$$\min_{\beta, u} \left(\underbrace{[0 \cdots 0, 1 \cdots 1]}_{c} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \\ u_1 \\ \vdots \\ u_N \end{bmatrix} \right)$$

$$\underset{x}{\text{minimize}} \quad c^T x$$

$$\text{s.t.} \quad Ax \leq b$$

$$\underbrace{\begin{bmatrix} X_1 & -1 & & \\ \vdots & & \ddots & \\ X_N & & & -1 \\ -X_1 & -1 & & \\ \vdots & & \ddots & \\ -X_N & & & -1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \\ u_1 \\ \vdots \\ u_N \end{bmatrix}}_{x} \leq \underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_N \\ -y_1 \\ \vdots \\ -y_N \end{bmatrix}}_{b}$$

f.

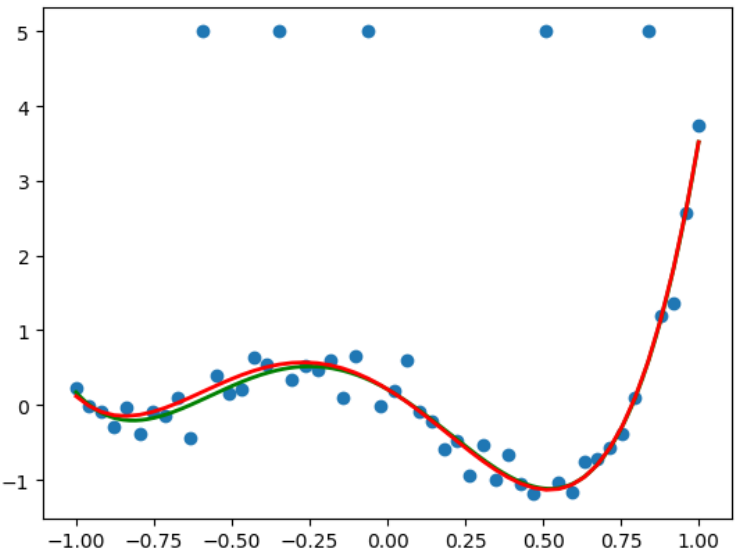


Figure 13. The green curve is for d, The red curve is for f

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Feb  3 01:40:44 2021

@author: 11327
"""

import numpy as  np
import matplotlib.pyplot as plt
from scipy.stats import norm
from scipy.special import eval_legendre
from scipy.optimize import linprog


# 1a

x = np.linspace(-3,3)
fX = 1/np.sqrt(2*np.pi) * np.exp(-(x*x) / 2)

plt.plot(x,fX)
plt.xlabel('x')
plt.ylabel('fX(x)')
plt.savefig('D:/phd2/ece595/1a.png')
plt.show()

# 1b
x = np.random.normal(0, 1, 1000)

plt.figure(1)
hp1 = plt.hist(x,4)

plt.figure(2)
hp2 = plt.hist(x,1000)

m,sd = norm.fit(x)
plt.figure(3)
plt.plot(hp1[1],norm.pdf(hp1[1]))
plt.figure(4)
plt.plot(hp2[1],norm.pdf(hp2[1]))

# 1c
x = np.random.normal(0, 1, 1000)
n = 1000
maxx = np.max(x)
minx = np.min(x)
diff = maxx - minx
# pj = x[0:200:1]
# sumpj = np.sum(pj**2)
J = []
M = np.linspace(1,200,200)
for m in M:
    m = int(m)
    pj = (np.histogram(x,m))[0] / np.sum((np.histogram(x,m))[0])
    sumpj = np.sum(pj**2)
    Jh = 2/((n-1)*(diff/m)) - (n+1)/((diff/m)*(n-1))*sumpj
    J = np.append(J,Jh)

plt.figure(1)
plt.plot(M,J)

Jmin = np.min(J)
```

```
mmin = np.where(J==Jmin)

plt.figure(2)
hp1 = plt.hist(x,int(mmin[0]))
plt.figure(3)
plt.plot(hp1[1],norm.pdf(hp1[1]))

# 2a
x1 = np.linspace(-1,5,100)
x2 = np.linspace(0,10,100)
X1,X2 = np.meshgrid(x1,x2)
fX = (1/np.sqrt(3*(2*np.pi)**2))*np.exp((2*X1**2 + 4*X1 - 2*X1*X2 + 2*X2**2+56-20*X2)/(-6))

plt.figure(1)
plt.contour(X1,X2,fX)

# 2c
x = np.random.multivariate_normal([0, 0], [[1, 0], [0, 1]], 5000)
plt.figure(1)
plt.scatter(x[:, 0], x[:, 1])

X = np.random.multivariate_normal([2, 6], [[2, 1], [1, 2]], 5000)
plt.figure(2)
plt.scatter(X[:, 0], X[:, 1])

S = [[2, 1], [1, 2]]
b = [2,6]
L,U = np.linalg.eig(S)
A = np.dot(np.dot(U,np.diag(L**(0.5))),U.T)
y = np.dot(A,x.T)
y1 = y[0,:] + 2
y2 = y[1,:] + 6
plt.figure(3)
plt.scatter(y1, y2)
y = [y1,y2]

# 3a
b0 = -0.001
b1 = 0.01
b2 = 0.55
b3 = 1.5
b4 = 1.2
x = np.linspace(-1,1,50) # 50 points in the interval [-1,1]
L1 = eval_legendre(1,x)
L2 = eval_legendre(2,x)
L3 = eval_legendre(3,x)
L4 = eval_legendre(4,x)
error = np.random.normal(0,0.2,50)
y = b0 + b1*L1 + b2*L2 + b3*L3 + b4*L4 + error


X = np.column_stack((eval_legendre(0,x), eval_legendre(1,x), \
                     eval_legendre(2,x), eval_legendre(3,x), \
                     eval_legendre(4,x)))
theta = np.linalg.lstsq(X, y, rcond=None)[0]
t     = np.linspace(-1, 1, 200);
yhat  = theta[0]*eval_legendre(0,t) + theta[1]*eval_legendre(1,t) + \
        theta[2]*eval_legendre(2,t) + theta[3]*eval_legendre(3,t) + \
        theta[4]*eval_legendre(4,t)

plt.figure(1)
```

```
plt.scatter(x, y)
plt.plot(t,yhat,'g', linewidth=2)
plt.show()

idx = [10,16,23,37,45] # these are the locations of the outliers
y[idx] = 5 # set the outliers to have a value 5
plt.figure(2)
plt.scatter(x, y)
plt.plot(t,yhat,'g', linewidth=2)
plt.show()


N = 50
for p in range(5):
  X[:,p] = eval_legendre(p,x)

Au = np.hstack((X,-np.eye(N)))
Al = np.hstack((-X,-np.eye(N)))
A = np.vstack((Au,Al))
b = np.hstack((y,-y))
c = np.hstack((np.zeros(5),np.ones(N)))

sol = linprog(c,A,b,bounds=(None,None))
beta = sol.x[0:5]

t2     = np.linspace(-1, 1, 50);
yhat2  = beta[0]*eval_legendre(0,t2) + beta[1]*eval_legendre(1,t2) + \
        beta[2]*eval_legendre(2,t2) + beta[3]*eval_legendre(3,t2) + \
        beta[4]*eval_legendre(4,t2)
plt.plot(t2,yhat2,'r', linewidth=2)
plt.show()
```

Implementing Noise2Noise for Dynamic Scenes

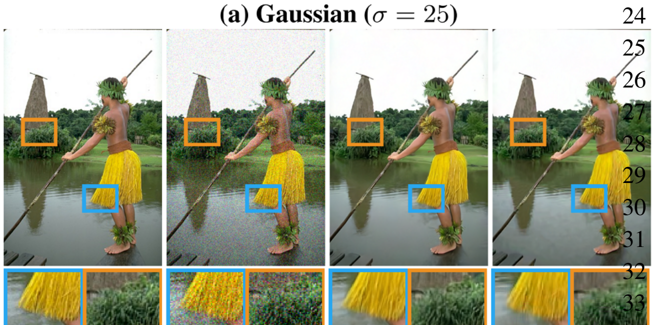
Ruijie Song M.S. [*1]

Abstract

Implementing Noise2Noise for Dynamic Scenes

1. Dataset

(a) Gaussian ($\sigma = 25$)



Ground truth Input Our Comparison BM3D

There's the dataset.

1.1. Codes

```
# Copyright (c) 2018, NVIDIA CORPORATION
#
# This work is licensed under the Creative
# 4.0 International License. To view a cop
# http://creativecommons.org/licenses/by-n
# Creative Commons, PO Box 1866, Mountain

import tensorflow as tf

def parse_tfrecord_tf(record):
    features = tf.parse_single_example(rec
        'shape': tf.FixedLenFeature([3], t
        'data': tf.FixedLenFeature([], tf.
    data = tf.decode_raw(features['data'],
    return tf.reshape(data, features['shape'

# [c,h,w] -> [h,w,c]
def chw_to_hwc(x):
    return tf.transpose(x, perm=[1, 2, 0])

# [h,w,c] -> [c,h,w]
def hwc_to_chw(x):
    return tf.transpose(x, perm=[2, 0, 1])

def resize_small_image(x):
    shape = tf.shape(x)
    return tf.cond(
        tf.logical_or(
            tf.less(shape[2], 256),
            tf.less(shape[1], 256)
        ),
        true_fn=lambda: hwc_to_chw(tf.image.resize_i
        false_fn=lambda: tf.cast(x, tf.float32)
    )

def random_crop_noised_clean(x, add_noise):
    cropped = tf.random_crop(resize_small_image(x),
    return (add_noise(cropped), add_noise(cropped),

def create_dataset(train_tfrecords, minibatch_size,
    print ('Setting up dataset source from', train_
    buffer_mb   = 256
    num_threads = 2
    dset = tf.data.TFRecordDataset(train_tfrecords,
    dset = dset.repeat()
    buf_size = 1000
    dset = dset.prefetch(buf_size)
    dset = dset.map(parse_tfrecord_tf, num_paralle
    dset = dset.shuffle(buffer_size=buf_size)
    dset = dset.map(lambda x: random_crop_noised_cl
    dset = dset.batch(minibatch_size)
    it = dset.make_one_shot_iterator()
    return it
```

A. Do *not* have an appendix here

Do not put content after the references. Put anything that you might normally include after the references in a separate supplementary file.

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut

[*]Equal contribution [1]Department of Electrical and Computer Engineering, Purdue University, West Lafayette, USA. Correspondence to: Cieua Vvvvv <c.vvvvv@googol.com>, Eee Pppp <ep@eden.co.uk>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021.

it up, please be careful to use a tool that doesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine.

Please do not use Apple's preview to cut off supplementary material. In previous years it has altered margins, and created headaches at the camera-ready stage.