Problem Set 1 Solutions

MAS.622J/1.126J: Pattern Recognition and Analysis

Originally Due Monday, 18 September 2006

Problem 1: Why?

a. Describe an application of pattern recognition related to your research. What are the features? What is the decision to be made? Speculate on how one might solve the problem. Limit your answer to a page.

b. In the same way, describe an application of pattern recognition you would be interested in pursuing for fun in your life outside of work.

Solution: Refer to examples discussed in lecture.

Problem 2: Probability Warm-Up

Let X and Y be random variables. Let $\mu_X \equiv \mathrm{E}[X]$ denote the expected value of X and $\sigma_X^2 \equiv \mathrm{E}[(X-\mu_X)^2]$ denote the variance of X. Use excruciating detail to answer the following:

a. Show $\mathrm{E}[X+Y] = \mathrm{E}[X] + \mathrm{E}[Y]$.

b. Show $\sigma_X^2 = \mathrm{E}[X^2] - \mu_X^2$.

c. Show that independent implies uncorrelated.

d. Show that uncorrelated does not imply independent.

e. Let $Z = X + Y$. Show that if X and Y are uncorrelated, then $\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2$.

f. Let X_1 and X_2 be independent and identically distributed continuous random variables. Can $\Pr[X_1 \le X_2]$ be calculated? If so, find its value. If not, explain.

g. Let X_1 and X_2 be independent and identically distributed discrete random variables. Can $\Pr[X_1 \le X_2]$ be calculated? If so, find its value. If not, explain.

Solution:

a. The following is for continuous random variables. A similar argument holds for discrete random variables.

$$
\begin{aligned}
\mathrm{E}[X+Y] &= \int\!\!\int (x+y)\, p(x,y)\, dx\, dy \\
&= \int\!\!\int x\, p(x,y)\, dx\, dy + \int\!\!\int y\, p(x,y)\, dx\, dy \\
&= \int x\, p(x)\, dx + \int y\, p(y)\, dy \\
&= \mathrm{E}[X] + \mathrm{E}[Y]
\end{aligned}
$$

b. Making use of the definition of variance and the previous part, we have:

$$
\begin{aligned}
\sigma_X^2 &= \mathrm{E}[(X-\mu_X)^2] \\
&= \mathrm{E}[X^2 - 2\mu_X X + \mu_X^2] \\
&= \mathrm{E}[X^2] - \mathrm{E}[2\mu_X X] + \mathrm{E}[\mu_X^2] \\
&= \mathrm{E}[X^2] - 2\mu_X \mathrm{E}[X] + \mu_X^2 \\
&= \mathrm{E}[X^2] - 2\mu_X \mu_X + \mu_X^2 \\
&= \mathrm{E}[X^2] - 2\mu_X^2 + \mu_X^2 \\
&= \mathrm{E}[X^2] - \mu_X^2
\end{aligned}
$$

c. Let X and Y be independent continuous random variables (a similar argument holds for discrete random variables). Then,

$$
\begin{aligned}
\mathrm{E}[XY] &= \int\!\!\int xy\, p(x,y)\, dx\, dy \\
&= \int\!\!\int xy\, p(x)\, p(y)\, dx\, dy \\
&= \int x\, p(x)\, dx \int y\, p(y)\, dy \\
&= \mathrm{E}[X]\, \mathrm{E}[Y]
\end{aligned}
$$

d. Let X and Y be discrete random variables such that X takes on values from $\{0,1\}$ and Y takes on values from $\{-1,0,1\}$. Let the probability mass function of X be

$$
\begin{aligned}
p_x[x=0] &= 0.5 \\
p_x[x=1] &= 0.5
\end{aligned}
$$

and the probability mass function of Y conditioned on X be

$$
p_{y|x}[y=-1|x=0] = 0.5
$$

$$
\begin{aligned}
p_{y|x}[y=0|x=0] &= 0 \\
p_{y|x}[y=1|x=0] &= 0.5 \\
p_{y|x}[y=-1|x=1] &= 0 \\
p_{y|x}[y=0|x=1] &= 1 \\
p_{y|x}[y=1|x=1] &= 0.
\end{aligned}
$$

Given the above, and the fact that $p_{x,y}[x,y] = p_{y|x}[y|x]\,p_x[x]$, we get

$$
\begin{aligned}
p_{x,y}[x=0,y=-1] &= 0.25 \\
p_{x,y}[x=0,y=0] &= 0 \\
p_{x,y}[x=0,y=1] &= 0.25 \\
p_{x,y}[x=1,y=-1] &= 0 \\
p_{x,y}[x=1,y=0] &= 0.5 \\
p_{x,y}[x=1,y=1] &= 0.
\end{aligned}
$$

However, the product of the marginals is given by

$$
\begin{aligned}
p_x[x=0]\,p_y[y=-1] &= 0.125 \\
p_x[x=0]\,p_y[y=0] &= 0.25 \\
p_x[x=0]\,p_y[y=1] &= 0.125 \\
p_x[x=1]\,p_y[y=-1] &= 0.125 \\
p_x[x=1]\,p_y[y=0] &= 0.25 \\
p_x[x=1]\,p_y[y=1] &= 0.125.
\end{aligned}
$$

Thus, we see that $p_{x,y}[x,y] \neq p_x[x]\,p_y[y]$ and X and Y are not independent. independent. However, since XY is identically zero, we also get

$$
\begin{aligned}
\mathrm{cov}(X,Y) = \sigma^2_{XY} &= \mathrm{E}[(X-\mu_X)(Y-\mu_Y)] \\
&= \mathrm{E}[XY] - \mu_X\mu_Y \\
&= \mathrm{E}[0] - (0.5)(0) \\
&= 0 - 0 \\
&= 0.
\end{aligned}
$$

Therefore, X and Y are uncorrelated but not independent.

e. Given that $Z = X + Y$ and that X and Y are uncorrelated, we have

$$
\begin{aligned}
\sigma^2_Z &= \mathrm{E}[(Z-\mu_Z)^2] \\
&= \mathrm{E}[Z^2] - \mu_Z^2 \\
&= \mathrm{E}[(X+Y)^2] - (\mu_X + \mu_Y)^2 \\
&= \mathrm{E}[X^2 + 2XY + Y^2] - (\mu_X^2 + 2\mu_X\mu_Y + \mu_Y^2) \\
&= \mathrm{E}[X^2] + 2\mathrm{E}[XY] + \mathrm{E}[Y^2] - \mu_X^2 - 2\mu_X\mu_Y - \mu_Y^2
\end{aligned}
$$

$$
\begin{aligned}
&= (\mathrm{E}[X^2]-\mu_X^2)+2(\mathrm{E}[XY]-\mu_X\mu_Y)+(\mathrm{E}[Y^2]-\mu_Y^2) \\
&= \sigma_X^2+2\sigma_{XY}^2+\sigma_Y^2 \\
&= \sigma_X^2+\sigma_Y^2,
\end{aligned}
$$

where only the last equality depends on X and Y being uncorrelated.

f. Given that X_1 and X_2 are continuous random variables, we know that $\Pr[X_1 = x] = 0$ and $\Pr[X_2 = x] = 0$ for any value of x. Thus,

$$\Pr[X_1 \le X_2] = \Pr[X_1 < X_2].$$

Given that X_1 and X_2 are i.i.d., we know that replacing X_1 with X_2 and X_2 with X_1 will have no effect on the world. In particular, we know that

$$\Pr[X_1 < X_2] = \Pr[X_2 < X_1].$$

However, since probabilities must sum to one, we have

$$\Pr[X_1 < X_2] + \Pr[X_2 < X_1] = 1.$$

Thus,

$$\Pr[X_1 \le X_2] = \frac{1}{2}.$$

g. For discrete random variables, unlike the continuous case above, we need to know the distributions of X_1 and X_2 in order to find $\Pr[X_1 = x]$ and $\Pr[X_2 = x]$. Thus, the argument we used above fails. In general, it is not possible to find $\Pr[X_1 \le X_2]$ without knowledge of the distributions of both X_1 and X_2.

Problem 3: High-Dimensional Probability

Let $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ be a random vector, where the $\{X_i\}$ are independent and identically distributed (i.i.d.) continuous random variables with a uniform probability density function between 0 and 1:

$$p(x_i) = \begin{cases} 1, \text{for } 0 \le x_i \le 1 \\ 0, \text{otherwise} \end{cases}$$

Each value $\mathbf{x}$ of the random vector $\mathbf{X}$ can be considered as a point in a n-dimensional hypercube. Since the probability density function of $\mathbf{X}$ is uniform, volume in this n-dimensional space corresponds directly to probability. Find an expression for the percentage of a n-dimensional hypercube's volume located within ϵ of the hypercube's surface. Plot this percentage as a function of n for $1 \le n \le 100000$ and $\epsilon = 0.0001$. What do your findings about high-dimensional hypercubes tell you about random variables?

Solution: The volume, $V_n(L)$, of a n-dimensional hypercube with sides of length L is

$$V_n(L) = L^n.$$

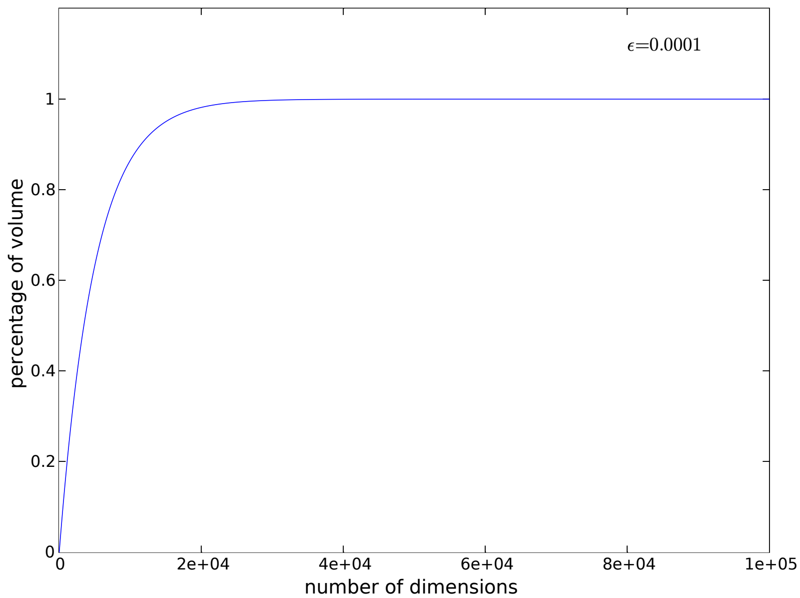

Figure 1: The percentage of a n-dimensional unit hypercube's volume within ϵ of the surface as a function of n.

Thus, the ratio, $R_n(\epsilon)$, of the volumes of a n-dimensional hypercube with sides of length $1-2\epsilon$ and an n-dimensional hypercube with sides of length 1 is

$$R_n(\epsilon) = \frac{V_n(1-2\epsilon)}{V_n(1)} = \frac{(1-2\epsilon)^n}{1^n} = (1-2\epsilon)^n.$$

Therefore, the percentage, $P_n(\epsilon)$, of a n-dimensional unit hypercube's volume located within ϵ of the hypercube's surface is

$$P_n(\epsilon) = 1 - R_n(\epsilon) = 1 - (1-2\epsilon)^n.$$

See Figure 1.

This result shows that randomly choosing a high-dimensional vector in which each element is i.i.d. and uniform will with high probability result in a point near the edge of the 'box' containing all the points being considered. It also hints at the power of considering long strings of i.i.d. variables. See, for example, the asymptotic equipartition theorem (AEP), which we may touch on later in class.

The Python code used to create Figure 1 is

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='warn'
```

```
import LinearAlgebra as la

e = 0.0001
n = arange(1,100001)
p = 1 - (1-2*e)**n
plot(n,p)
axis([0,n[-1],0,1.2])
xlabel('number of dimensions')
ylabel('percentage of volume')
text(0.8*n[-1], 1.1, r'$\epsilon=0.0001$')
show()
```

The Matlab code used to create Figure 1 is

```
e = 0.0001;
n = 1:100001;
p = 1 - (1-2*e).^n;
plot(n,p);
ns = size(n);
axis([0,n(ns(2)),0,1.2]);
xlabel('number of dimensions');
ylabel('percentage of volume');
text(0.8*n(ns(2)), 1.1, '\epsilon=0.0001');
```

Problem 4: Teatime with Gauss and Bayes

Let $p(x,y) = \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{(y-\mu)^2}{2\alpha^2} + \frac{(x-y)^2}{2\beta^2}\right)}$.

a. Find $p(x)$, $p(y)$, $p(x|y)$, and $p(y|x)$. In addition, give a brief description of each of these distributions.

b. Let $\mu = 0$, $\alpha = 40$, and $\beta = 3$. Plot $p(y)$ and $p(y|x = 13.7)$ for a reasonable range of y. What is the difference between these two distributions?

Solution:

a. To find $p(y)$, simply factor $p(x,y)$ and then integrate over x:

$$
\begin{aligned}
p(y) &= \int_{-\infty}^{\infty} p(x,y)\,dx \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{(y-\mu)^2}{2\alpha^2} + \frac{(x-y)^2}{2\beta^2}\right)}\,dx \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\frac{(y-\mu)^2}{2\alpha^2}}\, e^{-\frac{(x-y)^2}{2\beta^2}}\,dx
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{\sqrt{2\pi\alpha^2}} e^{-\frac{(y-\mu)^2}{2\alpha^2}} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(x-y)^2}{2\beta^2}}\, dx \\
&= \frac{1}{\sqrt{2\pi\alpha^2}} e^{-\frac{(y-\mu)^2}{2\alpha^2}} \\
&= \mathcal{N}(\mu, \alpha^2)
\end{aligned}
$$

The integral goes to 1 because it is of the form of a probability distribution integrated over the entire domain. To find $p(x|y)$, divide $p(x,y)$ by $p(y)$:

$$
\begin{aligned}
p(x|y) &= \frac{p(x,y)}{p(y)} \\
&= \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(x-y)^2}{2\beta^2}} \\
&= \mathcal{N}(y, \beta^2)
\end{aligned}
$$

Finding $p(x)$ and $p(y|x)$ follows essentially the same procedure, but the algebra is more involved and requires completing the square in the exponent.

$$
\begin{aligned}
p(x) &= \int_{-\infty}^{\infty} p(x,y)\, dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{(y-\mu)^2}{2\alpha^2} + \frac{(x-y)^2}{2\beta^2}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{\beta^2(y-\mu)^2 + \alpha^2(x-y)^2}{2\alpha^2\beta^2}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{\beta^2 y^2 - 2\beta^2\mu y + \beta^2\mu^2 + \alpha^2 x^2 - 2\alpha^2 xy + \alpha^2 y^2}{2\alpha^2\beta^2}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{(\alpha^2+\beta^2)y^2 - 2(\alpha^2 x + \beta^2\mu)y + (\beta^2\mu^2 + \alpha^2 x^2)}{2\alpha^2\beta^2}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{y^2 - 2\frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2} y + \frac{\beta^2\mu^2 + \alpha^2 x^2}{\alpha^2+\beta^2}}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{y^2 - 2\frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2} y + \left(\frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2}\right)^2 - \left(\frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2}\right)^2 + \frac{\beta^2\mu^2 + \alpha^2 x^2}{\alpha^2+\beta^2}}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{\left(y - \frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2}\right)^2 - \left(\frac{\alpha^2 x + \beta^2\mu}{\alpha^2+\beta^2}\right)^2 + \frac{\beta^2\mu^2 + \alpha^2 x^2}{\alpha^2+\beta^2}}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} dy
\end{aligned}
$$

$$
\begin{aligned}
&= \int_{-\infty}^{\infty} \frac{1}{2\pi\alpha\beta} e^{-\left(\frac{\left(y-\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} e^{-\left(\frac{\frac{\beta^2\mu^2+\alpha^2x^2}{\alpha^2+\beta^2}-\left(\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} dy \\
&= \frac{1}{2\pi\alpha\beta}\sqrt{2\pi\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}} e^{-\left(\frac{\frac{\beta^2\mu^2+\alpha^2x^2}{\alpha^2+\beta^2}-\left(\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}} e^{-\left(\frac{\left(y-\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} dy \\
&= \frac{1}{\sqrt{2\pi(\alpha^2+\beta^2)}} e^{-\left(\frac{\frac{\beta^2\mu^2+\alpha^2x^2}{\alpha^2+\beta^2}-\left(\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} \\
&= \frac{1}{\sqrt{2\pi(\alpha^2+\beta^2)}} e^{-\left(\frac{(\alpha^2+\beta^2)(\beta^2\mu^2+\alpha^2x^2)-(\alpha^2x+\beta^2\mu)^2}{2\alpha^2\beta^2(\alpha^2+\beta^2)}\right)} \\
&= \frac{1}{\sqrt{2\pi(\alpha^2+\beta^2)}} e^{-\left(\frac{\alpha^2\beta^2\mu^2+\alpha^4x^2+\beta^4\mu^2+\alpha^2\beta^2x^2-\alpha^4x^2-2\alpha^2\beta^2\mu x-\beta^4\mu^2}{2\alpha^2\beta^2(\alpha^2+\beta^2)}\right)} \\
&= \frac{1}{\sqrt{2\pi(\alpha^2+\beta^2)}} e^{-\left(\frac{\alpha^2\beta^2x^2-2\alpha^2\beta^2\mu x+\alpha^2\beta^2\mu^2}{2\alpha^2\beta^2(\alpha^2+\beta^2)}\right)} \\
&= \frac{1}{\sqrt{2\pi(\alpha^2+\beta^2)}} e^{-\left(\frac{(x-\mu)^2}{2(\alpha^2+\beta^2)}\right)} \\
&= \mathcal{N}(\mu, \alpha^2+\beta^2)
\end{aligned}
$$

To find $p(y|x)$ we simply divide $p(x,y)$ by $p(x)$. In finding $p(x)$, we already know the form of $p(y|x)$ (see the longest line in the derivation of $p(x)$ above):

$$
\begin{aligned}
p(y|x) &= \frac{p(x,y)}{p(x)} \\
&= \frac{1}{\sqrt{2\pi\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}} e^{-\left(\frac{\left(y-\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}\right)^2}{2\frac{\alpha^2\beta^2}{\alpha^2+\beta^2}}\right)} \\
&= \mathcal{N}\left(\frac{\alpha^2 x+\beta^2\mu}{\alpha^2+\beta^2}, \frac{\alpha^2\beta^2}{\alpha^2+\beta^2}\right)
\end{aligned}
$$

Note that all the above distibutions are Gaussian.

b. The following Python code produced Figure 2:

```
from matplotlib.numerix import *
```

```
from numarray import *
from pylab import plot, legend, axis, xlabel, text, show
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w

def normal(x, mean, var) :
    return (1.0/sqrt(2*pi*var))*e**(-((x-mean)**2)/(2*var))

m = 0.0
a = 40.0
b = 3.0
x = 13.7

y = arange(-150, 150, 1)
mean = ((a**2)*x + (b**2)*m)/(a**2 + b**2)
var = ((a*b)**2)/(a**2 + b**2)
p_y_given_x = normal(y, mean, var)
p_y = normal(y,m,a**2 + b**2)

plot(y,p_y_given_x)
plot(y,p_y)
legend(('p(y|x)', 'p(y)'))
axis([y[0],y[-1],0,0.15])
xlabel('y')
text(-100,0.12, r'$\mu=0$')
text(-100,0.11, r'$\alpha=40$')
text(-100,0.1, r'$\beta=3$')
show()
```

c. The following Matlab code produced Figure 2:

```
m = 0.0
a = 40.0
b = 3.0
x = 13.7

y = -150:1:150
mean = ((a^2)*x + (b^2)*m)/(a^2 + b^2)
var = ((a*b)^2)/(a^2 + b^2)
p_y_given_x = (1.0/sqrt(2*pi*var))*exp(-((y-mean).^2)/(2*var))
var2 = a^2 + b^2
p_y = (1.0/sqrt(2*pi*var2))*exp(-((y-m).^2)/(2*var2))

hold off
plot(y,p_y_given_x,'b')
hold on
plot(y,p_y,'r')
```

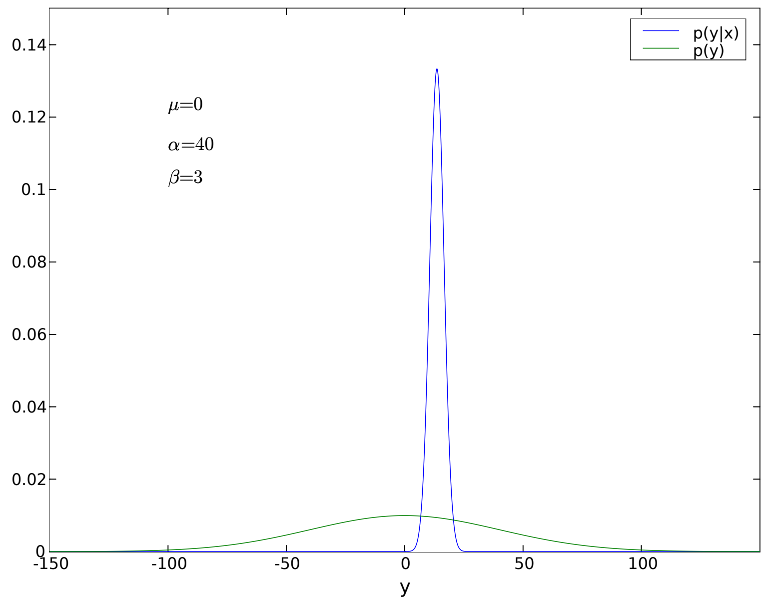

Figure 2: The marginal p.d.f. of y and the p.d.f. of y given x for a specific value of x. Notice how knowing x makes your knowledge of y more certain.

```
legend('p(y|x)', 'p(y)')
sy = size(y)
axis([y(1),y(sy(2)),0,0.15])
xlabel('y')
text(-100,0.12, '\mu=0')
text(-100,0.11, '\alpha=40')
text(-100,0.1, '\beta=3')
```

Problem 5: Covariance Matrix

Let $\Lambda_X = \begin{bmatrix} 64 & -25 \\ -25 & 64 \end{bmatrix}$.

a. Verify that Λ_X is a valid covariance matrix.

b. Find the eigenvalues and eigenvectors of Λ_X by hand. Show all your work.

c. Write a program to find and verify the eigenvalues and eigenvectors of Λ_X.

d. We provide 200 data points sampled from the distribution $\mathcal{N}(0, \Lambda_X)$. Download the dataset from the course website and plot the data points.

Project the data onto the covariance matrix eigenvectors and plot the transformed data. What is the difference between the two plots?

Solution:

a. The matrix Λ_X is a valid covariance matrix if it is symmetric and positive semi-definite. Clearly, it is symmetric, since $\Lambda_X^T = \Lambda_X$. One way to prove it is positive semi-definite is to show that all its eigenvalues are non-negative. This is indeed the case, as shown in the next part of the problem.

b. We can find the eigenvectors and eigenvalues of Λ_X by starting with the definition of an eigenvector. Namely, an vector $\mathbf{e}$ is an eigenvector of Λ_X if it satisfies

$$\Lambda_X \mathbf{e} = \lambda \mathbf{e}$$

for some constant scalar λ, which is called the eigenvalue corresponding to $\mathbf{e}$. This can be rewritten as

$$(\Lambda_X - \lambda I)\mathbf{e} = \mathbf{0}.$$

This is equivalent to

$$\det(\Lambda_X - \lambda I) = 0.$$

Thus, we require that

$$(64 - \lambda)^2 - 25^2 = 0$$

By inspection, this is true when $\lambda = 89$ and $\lambda = 39$, both of which are non-negative, thus confirming that Λ_X is indeed a positive semi-definite matrix.

To find the eigenvectors, we plug the eigenvalues back into the equation above to get

$$(\Lambda_X - 89I)\mathbf{e} = \begin{bmatrix} 64-89 & -25 \\ -25 & 64-89 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} -25 & -25 \\ -25 & -25 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

which gives $a = -b$. Normalized, this results in the eigenvector

$$\mathbf{e}_1 = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}.$$

Similarly, $\lambda = 39$ gives

$$(\Lambda_X - 39I)\mathbf{e} = \begin{bmatrix} 64-39 & -25 \\ -25 & 64-39 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 25 & -25 \\ 25 & -25 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

which gives $a = b$. Normalized, this results in the eigenvector

$$\mathbf{e}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}.$$

c. The following Python program prints out the eigenvectors and eigenvalues of Λ_X:

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show, t
from LinearAlgebra import *
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w

L = array([[64, -25],[-25, 64]])
print eigenvectors(L)
```

d. The following Matlab program prints out the eigenvectors and eigenvalues of Λ_X:

```
L = [[64, -25]; [-25, 64]];
[u, v] = eig(L)
```

e. The following Python program generated Figure 3:

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show, t
from LinearAlgebra import *
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w

C = array([[64,-25],[-25,64]])
(u,v) = eigenvectors(C)

f = file("ps1.dat", "r")
p = f.readlines()
for i in range(len(p)) :
    temp = p[i].split(' ')
    p[i] = [float(temp[0]), float(temp[1])]

q = zeros(p.shape, 'f')
for i in range(len(p)) :
    q[i][0] = dot(p[i],v[0])
    q[i][1] = dot(p[i],v[1])

q = transpose(q)
p = transpose(p)

subplot(211)
scatter(p[0],p[1])
axis((-40,40,-40,40))
```

```
title('Original Data')
xlabel('x')
ylabel('y')
subplot(212)
scatter(q[0],q[1])
axis((-40,40,-40,40))
title('Transformed Data')
xlabel('first eigenvector')
ylabel('second eigenvector')
show()
```

f. The following Matlab program generated Figure 3:

```
C = [[64,-25]; [-25,64]];
[u,v] = eig(C);

p = load('ps1.dat');

sp = size(p);
q = zeros(sp);
for i = 1:sp(1)
    q(i,1) = p(i,:)*u(:,1);
    q(i,2) = p(i,:)*u(:,2);
end

q = q';
p = p';

subplot(211)
scatter(p(1,:),p(2,:))
axis([-40,40,-40,40])
title('Original Data')
xlabel('x')
ylabel('y')
subplot(212)
scatter(q(1,:),q(2,:))
axis([-40,40,-40,40])
title('Transformed Data')
xlabel('first eigenvector')
ylabel('second eigenvector')
```

The second plot in Figure 3 shows the data rotated to align with the eigenvectors of the data's covariance matrix.

Problem 6: Distribution Linearity

Let X_1 and X_2 be i.i.d. according to

$$p(x_i) = \begin{cases} 1, \text{for } 0 \leq x_i \leq 1 \\ 0, \text{otherwise} \end{cases} \quad \text{for } i = 1, 2$$

Let $Y = X_1 + X_2$.

a. Find an expression for $p(y)$. Plot $p(y)$ for some reasonable range of y.

b. Find an expression for $p(x_1|y)$. Plot $p(x_1|y)$ as a function of x_1 with y treated as a known parameter for some reasonable value of y and some reasonable range of x_1.

c. Repeat the parts above, this time letting X_1 and X_2 be i.i.d. according to $\mathcal{N}(0, 1)$.

d. What was the point of this problem? Hint: check out the title.

Solution:

a. From basic probability theory, we know that the probability density function of the sum of two independent random variables is the convolution of the two probability density functions. So,

$$\begin{aligned}
p_y(y) &= (p_{x_1} * p_{x_2})(y) \\
&= \int_{-\infty}^{\infty} p_{x_1}(x)\, p_{x_2}(y-x)\, dx \\
&= \int_0^1 1\, p_{x_2}(y-x)\, dx \\
&= \int_0^1 p_{x_2}(y-x)\, dx \\
&= \int_0^1 \left\{ \begin{array}{ll} 1 & \text{for } 0 \leq y - x \leq 1 \\ 0 & \text{otherwise} \end{array} \right\} dx \\
&= \int_0^1 \left\{ \begin{array}{ll} 1 & \text{for } y - 1 \leq x \leq y \\ 0 & \text{otherwise} \end{array} \right\} dx \\
&= \int_{\max(0,y-1)}^{\min(1,y)} 1\, dx \\
&= \max\{0, \min(1, y) - \max(0, y-1)\} \\
&= \begin{cases} 0 & \text{for } y \leq 0 \\ y & \text{for } 0 \leq y \leq 1 \\ 2 - y & \text{for } 1 \leq y \leq 2 \\ 0 & \text{for } y \geq 2 \end{cases}
\end{aligned}$$

This p.d.f. is shown in Figure 4, which was produced using the following Python program:

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w

plot([-1,0,1,2,3],[0,0,1,0,0])
axis([-1,3,-0.5,2])
xlabel('y')
ylabel('p(y)')
show()
```

This p.d.f. is shown in Figure 4, which was produced using the following Matlab program:

```
hold off
subplot(111)
plot([-1,0,1,2,3],[0,0,1,0,0])
hold on
axis([-1,3,-0.5,2])
xlabel('y')
ylabel('p(y)')
```

b. Using Bayes' Rule, we have

$$\begin{aligned} p_{x_1|y}(x_1|y) &= \frac{p_{x_1,y}(x_1,y)}{p_y(y)} \\ &= \frac{p_{y|x_1}(y|x_1)\,p_{x_1}(x_1)}{p_y(y)} \end{aligned}$$

We already know $p_y(y)$ and $p_{x_1}(x_1)$. Finding $p_{y|x_1}(y|x_1)$ is a matter of realizing that $y = x_1 + x_2$ implies that, given x_1, y is simply x_2 offset by a constant. Thus,

$$p_{y|x_1}(y|x_1) = p_{x_2}(y - x_1)$$

and

$$\begin{aligned} p_{x_1|y}(x_1|y) &= \frac{p_{x_2}(y-x_1)\,p_{x_1}(x_1)}{p_y(y)} \\ &= \begin{cases} \frac{1}{y} & \text{for } 0 \le y \le 1 \text{ and } 0 \le x_1 \le y \\ \frac{1}{2-y} & \text{for } 1 \le y \le 2 \text{ and } y-1 \le x_1 \le 1 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

See Figure 5, which was produced by the following Python program:

```
from matplotlib.numerix import *
from numarray import *
```

```
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w

subplot(211)
for y in arange(0.1, 1.1, stride=0.1) :
    x = array([0,0,y,y,1,1])
    Px_given_y = array([0,1.0/y,1.0/y,0,0,0])
    plot(x,Px_given_y)

xlabel(r'$x_1$')
ylabel(r'$p(x_1 \ given \ 0<y<1)$')
axis([-0.1,1.1,0,12])

subplot(212)
for y in arange(1.0, 2.0, stride=0.1) :
    x = array([0,0,y-1,y-1,1,1])
    Px_given_y = array([0,0,0,1.0/(2-y),1.0/(2-y),0])
    plot(x,Px_given_y)

xlabel('$x_1$')
ylabel('$p(x_1 \ given \ 1<y<2)$')
axis([-0.1,1.1,0,12])

show()
```

See Figure 5, which was produced by the following Matlab program:

```
hold off
subplot(211)
hold on
for y = 0.1:0.1:1
    x = [0,0,y,y,1]
    Px_given_y = [0,1.0/y,1.0/y,0,0]
    plot(x,Px_given_y,'b')
end

xlabel('x_1')
ylabel('p(x_1 | 0<y<1)')
axis([-0.1,1.1,0,12])

subplot(212)
hold on
for y = 1.0:0.1:1.9
    x = [0,y-1,y-1,1,1]
    Px_given_y = [0,0,1.0/(2-y),1.0/(2-y),0]
    plot(x,Px_given_y,'r')
```

```
end

xlabel('x_1')
ylabel('p(x_1 | 1<y<2)')
axis([-0.1,1.1,0,12])
```

c. Repeating the above using normal distributions, we get

$$
\begin{aligned}
p_y(y) &= (p_{x_1} * p_{x_2})(y) \\
&= \int_{-\infty}^{\infty} p_{x_1}(x)\, p_{x_2}(y-x)\, dx \\
&= \int_{-\infty}^{\infty} \left(\frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}\right) \left(\frac{1}{\sqrt{2\pi}} e^{\frac{-(y-x)^2}{2}}\right) dx \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi} e^{\frac{-(2x^2-2xy+y^2)}{2}}\, dx \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi} e^{\frac{-(x^2-xy+\frac{y^2}{4}-\frac{y^2}{4}+\frac{y^2}{2})}{2\frac{1}{2}}}\, dx \\
&= \int_{-\infty}^{\infty} \frac{1}{2\pi} e^{-\left((x-\frac{y}{2})^2-\frac{y^2}{4}+\frac{y^2}{2}\right)}\, dx \\
&= \frac{1}{\sqrt{4\pi}} e^{-\frac{y^2}{4}} \int_{-\infty}^{\infty} \frac{1}{\sqrt{\pi}} e^{-(x-\frac{y}{2})^2}\, dx \\
&= \frac{1}{\sqrt{4\pi}} e^{-\frac{y^2}{4}} \\
&= \mathcal{N}(0,2)
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
p_{x_1|y}(x_1|y) &= \frac{p_{x_2}(y-x_1)\, p_{x_1}(x_1)}{p_y(y)} \\
&= \frac{\left(\frac{1}{\sqrt{2\pi}} e^{\frac{-(y-x_1)^2}{2}}\right)\left(\frac{1}{\sqrt{2\pi}} e^{\frac{-x_1^2}{2}}\right)}{\left(\frac{1}{\sqrt{4\pi}} e^{\frac{-y^2}{4}}\right)} \\
&= \frac{1}{\sqrt{\pi}} e^{-\frac{y^2-4x_1y+4x_1^2}{4}} \\
&= \frac{1}{\sqrt{\pi}} e^{-(x_1-\frac{y}{2})^2} \\
&= \mathcal{N}\left(\frac{y}{2}, \frac{1}{2}\right)
\end{aligned}
$$

See Figure 6, which was produced by the following Python program:

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w
import LinearAlgebra as la

subplot(211)
y = arange(-5,5,0.01)
p = (1.0/sqrt(4*pi))*(e**(-(y**2)/4))
plot(y,p)
xlabel(r'$y$')
ylabel(r'$p(y)$')
axis([-5,5,-0.2,1.0])

subplot(212)
y = 1.6
x = arange(-5,5,0.01)
p = (1.0/sqrt(pi))*(e**(-((x-y/2)**2)))
plot(x,p)
xlabel(r'$x_1$')
ylabel(r'$p(x_1 \ given \ y=1.6)$')
axis([-5,5,-0.2,1.0])

show()
```

See Figure 6, which was produced by the following Matlab program:

```
subplot(211);
y = -5:0.01:5;
p = (1.0/sqrt(4*pi))*(exp(-(y.^2)/4));
plot(y,p);
xlabel('y');
ylabel('p(y)');
axis([-5,5,-0.2,1.0]);

subplot(212);
y = 1.6;
x = -5:0.01:5;
p = (1.0/sqrt(pi))*(exp(-((x-y/2).^2)));
plot(x,p);
xlabel('x_1');
ylabel('p(x_1 | y=1.6)');
axis([-5,5,-0.2,1.0]);
```

d. The point of this problem is to show that probability density functions are in general not closed under linear combinations of i.i.d. random variables.

That is, given two i.i.d. random variables x_1 and x_2 with distribution of type A, the random variable $y = x_1 + x_2$ does not in general have a distribution of type A. Gaussian (a.k.a normal) distributions are an exception. In fact, Gaussians are the only non-trivial family of functions that are both closed and linear under convolution (and therefore under addition of i.i.d. random variables):

$$\mathcal{N}(\mu_1, \sigma_1^2) * \mathcal{N}(\mu_2, \sigma_2^2) = \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$$

Problem 7: Monty Hall

To get credit for this problem, you must not only write your own correct solution, but also write a computer simulation (in either Matlab or Python) of the process of playing this game:

Suppose I hide the ring of power in one of three identical boxes while you weren't looking. The other two boxes remain empty. After hiding the ring of power, I ask you to guess which box it's in. I know which box it's in and, after you've made your guess, I deliberately open the lid of an empty box, which is one of the two boxes you did not choose. Thus, the ring of power is either in the box you chose or the remaining closed box you did not choose. Once you have made your initial choice and I've revealed to you an empty box, I then give you the opportunity to change your mind – you can either stick with your original choice, or choose the unopened box. You get to keep the contents of whichever box you finally decide upon.

- What choice should you make in order to maximize your chances of receiving the ring of power? Explain your answer.
- Write a simulation. There are two choices in this game for the contestant in this game: (1) choice of box, (2) choice of whether or not to switch. In your simulation, first let the host choose a random box to place the ring of power. Show a trace of your program's output for a single game play, as well as a cumulative probability of winning for 1000 rounds of the two policies (1) to choose a random box and then switch and (2) to choose a random box and not switch.

Solution:

- Always switch your answer to the box you didn't choose the first time. This reason is as follows. You have a 1/3 chance of initially picking the correct box. That is, there is a 2/3 chance the correct answer is one of the other two boxes. Learning which of the two other boxes is empty does not change these probabilities; your initial choice still has a 1/3 chance of being correct. That is, there is a 2/3 chance the remaining box is the correct answer. Therefore you should change your choice.

Another way to understand the problem is to extend it to 100 boxes, only one of which has the ring of power. After you make your initial choice, I then open 98 of the 99 remaining boxes and show you that they are empty. Clearly, with very high probability the ring of power resides in the one remaining box you did not initially choose.

- Here is a sample simulation output for the Monty Hall problem:

```
actual:   1
guess1:   2
reveal:   3
swap   :  0
guess2:   2

actual:   3
guess1:   3
reveal:   1
swap   :  0
guess2:   3

actual:   2
guess1:   3
reveal:   1
swap   :  0
guess2:   3

swap             :  0
win              :  292
lose             :  708
win/(win+lose):  0.292

actual:   3
guess1:   1
reveal:   2
swap   :  1
guess2:   3

actual:   1
guess1:   1
reveal:   2
swap   :  1
guess2:   3

actual:   3
guess1:   2
reveal:   1
```

```
swap   :   1
guess2 :   3

swap            : 1
win             : 686
lose            : 314
win/(win+lose): 0.686
```

Here is a Python program that generates the Monty Hall simulation output above:

```
from matplotlib.numerix import *
from numarray import *
from pylab import plot, subplot, legend, axis, xlabel, ylabel, text, show, r
Error.setMode(all=None, overflow='warn', underflow='ignore', dividebyzero='w
from LinearAlgebra import *


for swap in range(2) :
    win  = 0
    lose = 0
    for i in range(1000) :
        actual = int(rand()*3)+1;
        guess1 = int(rand()*3)+1;
        if guess1 == actual :
            reveal = int(rand()*2)+1;
            if reveal == actual :
                reveal = reveal + 1;
        else:
            if guess1 == 1 and actual == 2 :
                reveal = 3;
            elif guess1 == 1 and actual == 3 :
                reveal = 2;
            elif guess1 == 2 and actual == 1 :
                reveal = 3;
            elif guess1 == 2 and actual == 3 :
                reveal = 1;
            elif guess1 == 3 and actual == 1 :
                reveal = 2;
            elif guess1 == 3 and actual == 2 :
                reveal = 1;
        if swap == 1 :
            if guess1 == 1 and reveal == 2 :
                guess2 = 3;
            elif guess1 == 1 and reveal == 3 :
                guess2 = 2;
            elif guess1 == 2 and reveal == 1 :
```

```
            guess2 = 3;
        elif guess1 == 2 and reveal == 3 :
            guess2 = 1;
        elif guess1 == 3 and reveal == 1 :
            guess2 = 2;
        elif guess1 == 3 and reveal == 2 :
            guess2 = 1;
    else:
        guess2 = guess1;

    if guess2 == actual :
        win = win + 1;
    else:
        lose = lose + 1;

    # only print trace for first 3 games
    if i < 3 :
        print 'actual: ', actual
        print 'guess1: ', guess1
        print 'reveal: ', reveal
        print 'swap  : ', swap
        print 'guess2: ', guess2

# print results for each game play policy
print 'swap           :', swap
print 'win            :', win
print 'lose           :', lose
print 'win/(win+lose):', float(win) / float(win + lose)
```

Here is a Matlab program that simulates the Monty Hall simulation output above:

```
for swap = 0:1
  win  = 0;
  lose = 0;
  for i = 1:1000
    actual = floor(rand()*3)+1;
    guess1 = floor(rand()*3)+1;
    if guess1 == actual
      reveal = floor(rand()*2)+1;
      if reveal == actual
        reveal = reveal + 1;
      end
    else
      if guess1 == 1 && actual == 2
        reveal = 3;
      elseif guess1 == 1 && actual == 3
```

```
          reveal = 2;
        elseif guess1 == 2 && actual == 1
          reveal = 3;
        elseif guess1 == 2 && actual == 3
          reveal = 1;
        elseif guess1 == 3 && actual == 1
          reveal = 2;
        elseif guess1 == 3 && actual == 2
          reveal = 1;
        end
      end
      if swap == 1
        if guess1 == 1 && reveal == 2
          guess2 = 3;
        elseif guess1 == 1 && reveal == 3
          guess2 = 2;
        elseif guess1 == 2 && reveal == 1
          guess2 = 3;
        elseif guess1 == 2 && reveal == 3
          guess2 = 1;
        elseif guess1 == 3 && reveal == 1
          guess2 = 2;
        elseif guess1 == 3 && reveal == 2
          guess2 = 1;
        end
      else
        guess2 = guess1;
      end
      if guess2 == actual
        win = win + 1;
      else
        lose = lose + 1;
      end
      %% only print trace for first 3 games
      if i <= 3
        actual
        guess1
        reveal
        swap
        guess2
      end
    end
    %% print results for each game play policy
    swap
    win / (win + lose)
end
```

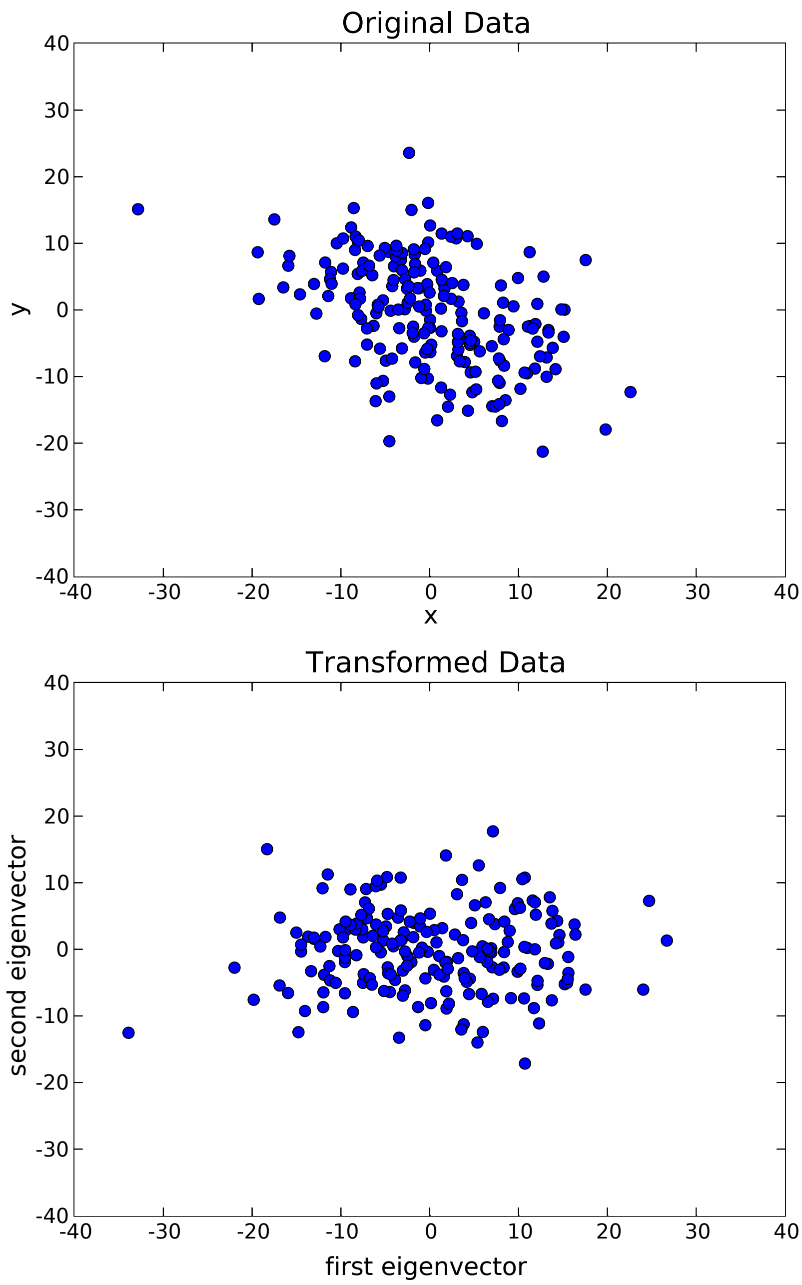

Figure 3: The original data and the data transformed into the coordinate system defined by the eigenvectors of their covariance matrix.

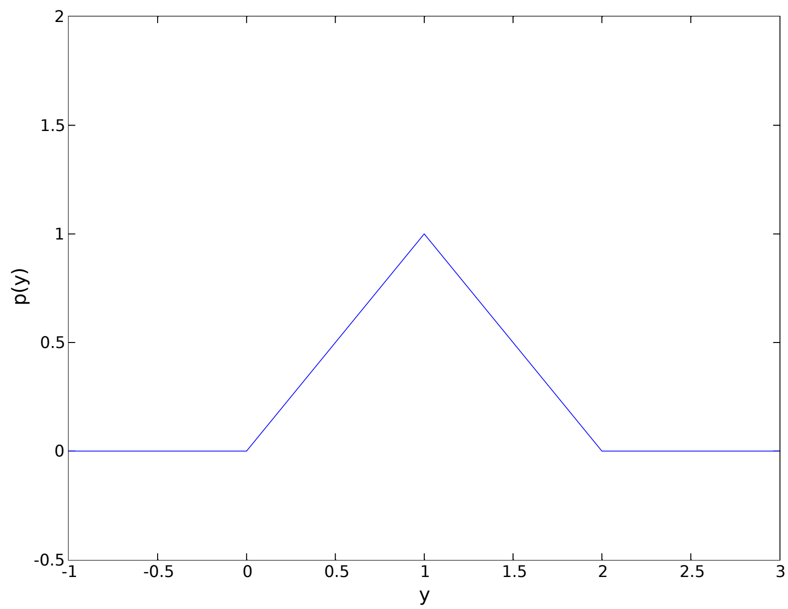

Figure 4: The probability density function of the sum of two independent uniform random variables.

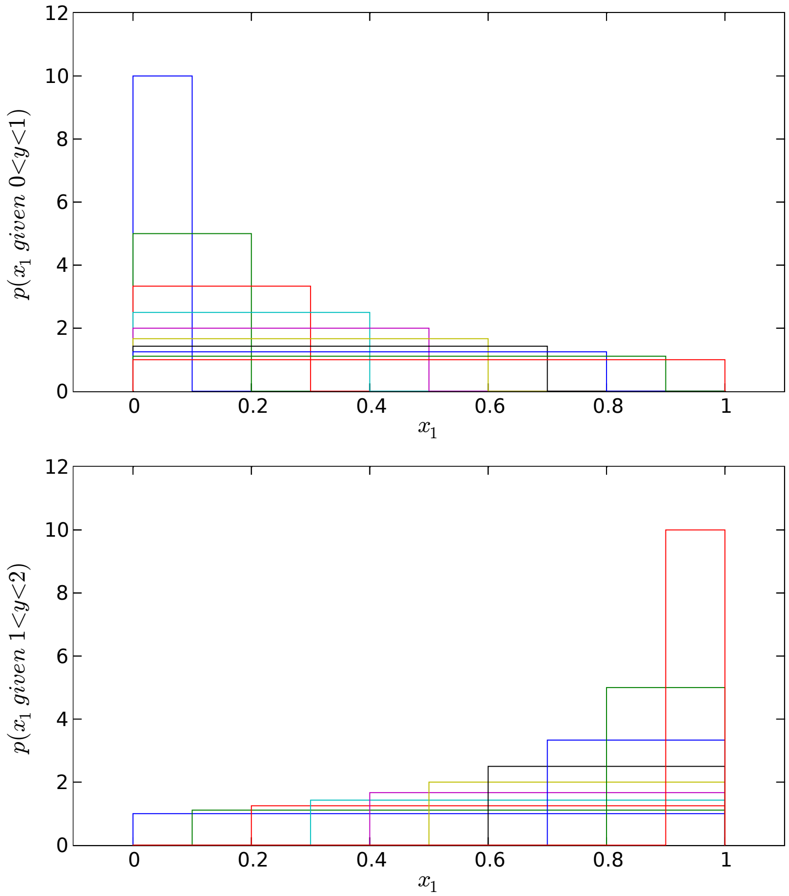

Figure 5: The probability density function of x_1 given certain values of y, where $y = x_1 + x_2$ and x_1 and x_2 are i.i.d. uniform random variables.

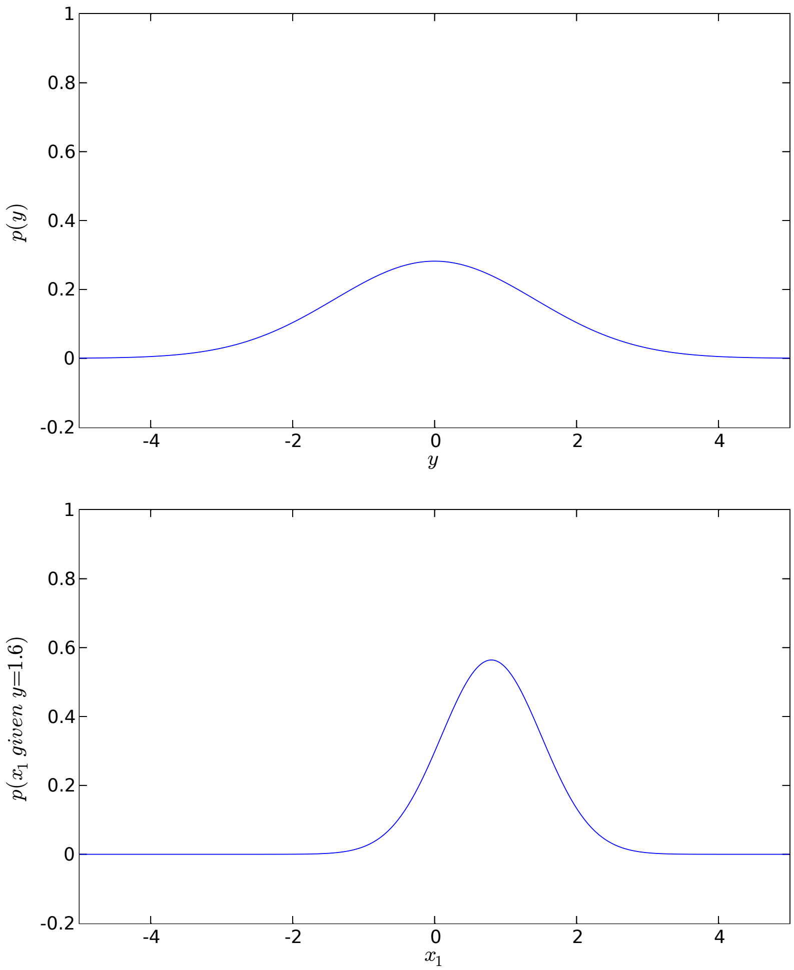

Figure 6: The probability density function of y and the probability density function for x_1 given $y = 1.6$, where $y = x_1 + x_2$ and x_1 and x_2 are i.i.d. $\mathcal{N}(0, 1)$ random variables.