Data Driven Sampling (Lecture 3): Sample random variable on computer

Hongwei YUAN

University of Macau

Definition (Uniform distribution)

If $\theta_1 < \theta_2$, a random variable X is said to have a continuous uniform probability distribution on the interval (θ_1, θ_2) if and only if the density function of Y is

$$f(x) = \begin{cases} \frac{1}{\theta_2 - \theta_1}, & \theta_1 \leq x \leq \theta_2, \\ 0, & elsewhere. \end{cases}$$

We denote this distribution by $\mathcal{U}(\theta_1, \theta_2)$.

- In this lecture, we will often use the uniform distribution $\mathcal{U}(0, 1)$.

Let X be a random variable, its cumulative distribution function (abbreviated as cdf) is

$$F(x) = \mathbb{P}(X \leq x), \qquad x \in \mathbb{R}.$$

Theorem

If cdf F is a strictly increasing and continuous function, then the random variable $F(X)$ is a $\mathcal{U}(0,1)$ distributed random variable.

- By the definition $F(x) = \mathbb{P}(X \leq x)$, we know that F is a nondecreasing function, i.e. for any $x_1 < x_2$, we have $F(x_1) \leq F(x_2)$.
- In the above theorem, we assume that F is strictly increasing, i.e. for any $x_1 < x_2$, we have $F(x_1) < F(x_2)$. Hence, we know F has its inverse function F^{-1}.

$0 \leq F(x) \leq 1$

Proof.

Since F is strictly increasing and continuous, it has its inverse function F^{-1}, for any $u \in (0,1)$, there exists a unique $x \in \mathbb{R}$ so that $u = F(x)$. Thus,

$$\mathbb{P}(F(X) \leq u) = \mathbb{P}(F(X) \leq F(x)) = \mathbb{P}(F^{-1}(F(X)) \leq F^{-1}(F(x)))$$

$P(Y \leq u)$, $Y = F(X)$

$$= \mathbb{P}(X \leq x) = F(x) = u.$$

So we got $P(Y \leq u) = u \Rightarrow Y \sim U(0,1)$

$\Rightarrow F(X) \sim U(0,1)$

□

uniform

$P(U \le y) = y$

Theorem

If cdf F is strictly increasing and continuous, and U is a $\mathcal{U}(0,1)$-distributed random variable, then the random variable $F^{-1}(U)$ has a cdf F.

$X = F^{-1}(U)$

$U = F(X)$

Proof.

For any $x \in \mathbb{R}$, we have

$$\mathbb{P}(X \le x) = \mathbb{P}(F^{-1}(U) \le x) = \mathbb{P}(U \le F(x)) = F(x).$$

□

- This theorem is actually true for any cdf.

For an arbitrary cdf F, define its general inverse: non-decreasing function: F

② write down.

$$F^{-}(u) = \inf\{x : F(x) \geq u\}, \qquad u \in [0, 1].$$

Theorem

③ check

Given a cdf F, then the random variable X defined by $X = F^{-}(U)$ has a cdf F.

The proof is not required! ① F of Binomial dist.

Assignment: Verify that the theorem is true for the Binomial distribution $Bin(4, 1/2)$.

Assignment 3-1:

Solution: $F \sim Bin(4, \frac{1}{2})$ $\because p(X=0) = \binom{4}{0}(\frac{1}{2})^0(1-\frac{1}{2})^4 = 1 \cdot 1 \cdot (\frac{1}{2})^4 = \frac{1}{16}$ $\therefore F(0) = \frac{1}{16}$;

$\because p(X=1) = \binom{4}{1}(\frac{1}{2})^1(1-\frac{1}{2})^3 = 4 \cdot \frac{1}{2} \cdot \frac{1}{2^3} = \frac{1}{4}$, $\therefore F(1) = \frac{1}{16} + \frac{1}{4} = \frac{5}{16}$;

$\because p(X=2) = \binom{4}{2}(\frac{1}{2})^2(1-\frac{1}{2})^2 = \frac{4!}{2! \cdot 2!} \cdot \frac{1}{4} \cdot \frac{1}{4} = \frac{3 \times 4}{1 \times 2} \times \frac{1}{16} = \frac{3}{8}$ $\therefore F(2) = F(1) + p(X=2) = \frac{5}{16} + \frac{6}{16} = \frac{11}{16}$;

$\because p(X=3) = \binom{4}{3}(\frac{1}{2})^3(1-\frac{1}{2})^1 = 4 \cdot \frac{1}{16} = \frac{1}{4}$ $\therefore F(3) = F(2) + p(X=3) = \frac{11}{16} + \frac{1}{4} = \frac{15}{16}$;

$\because p(X=4) = \binom{4}{4}(\frac{1}{2})^0(1-\frac{1}{2})^4 = \frac{1}{16}$ $\therefore F(4) = F(3) + p(X=4) = \frac{15}{16} + \frac{1}{16} = 1$

Hence $\therefore$ when $0 \le u \le \frac{1}{16}$, $F^{-1}(u) = 0$

when $\frac{1}{16} < u \le \frac{5}{16}$, $F^{-1}(u) = 1$

when $\frac{5}{16} < u \le \frac{11}{16}$, $F^{-1}(u) = 2$

when $\frac{11}{16} < u \le \frac{15}{16}$, $F^{-1}(u) = 3$

when $\frac{15}{16} < u \le 1$, $F^{-1}(u) = 4$

$$\Rightarrow X = \begin{cases} 0, & 0 \le u \le \frac{1}{16} \\ 1, & \frac{1}{16} < u \le \frac{5}{16} \\ 2, & \frac{5}{16} < u \le \frac{11}{16} \\ 3, & \frac{11}{16} < u \le \frac{15}{16} \\ 4, & \frac{15}{16} < u \le 1 \end{cases}$$

So to verify X has a cdf F, we verify $p(X \le k) = F(k)$

① $\because p(X \le 0) = p(0 \le U \le \frac{1}{16}) = \frac{1}{16} - 0 = \frac{1}{16}$

Same time $F(0) = \frac{1}{16}$

$\therefore p(X \le 0) = F(0)$ verified

② $\because p(X \le 1) = p(X=0) + p(X=1) = p(0 \le U \le \frac{5}{16}) = \frac{5}{16} - 0 = \frac{5}{16}$

Same time $F(1) = \frac{5}{16}$

$\therefore p(X \le 1) = F(1)$ verified

③ $\because p(X \le 2) = p(X=0) + p(X=1) + p(X=2) = p(0 \le U \le \frac{11}{16})$
$= \frac{11}{16} - 0 = \frac{11}{16}$

Same time $F(2) = \frac{11}{16}$

$\therefore p(X \le 2) = F(2)$ verified

④ $p(X \le 3) = \sum_{i=0}^{3} p(X=i) = p(0 \le U \le \frac{15}{16}) = \frac{15}{16} - 0 = \frac{15}{16}$

Same time $F(3) = \frac{15}{16}$

$\therefore p(X \le 3) = F(3)$ verified

⑤ $p(X \le 4) = \sum_{i=0}^{4} p(X=i) = p(0 \le U \le 1) = 1 - 0 = 1$

Same time $F(4) = 1$

$\therefore p(X \le 4) = F(4)$ verified.

Finally $p(X \le k) = F(k)$ verified

$\therefore p(F^{-1}(U) \le k) = F(k)$ verified

$\therefore$ $F^{-1}(U)$ has distribution of F. for cdf.

$\square$

Example

Given $Exp(1)$ distribution's cdf $F(x) = 1 - e^{-x}$. The inverse function of F is $F^{-1}(u) = -\log(1-u)$ for all $u \in [0,1]$. By Theorem 4, if $U \sim \mathcal{U}(0,1)$, then we know

$$F^{-1}(U) \sim Exp(1).$$

Since $1 - U \sim \mathcal{U}(0,1)$, then

$$F^{-1}(1-U) = -\log(U) \sim Exp(1).$$

$h(u) = \begin{cases} 1, & u \in [0,1] \\ 0, & \text{elsewhere.} \end{cases}$ Uniform dist

$U \sim \mathcal{U}(0,1)$

$1-U \sim \mathcal{U}(0,1)$ ☆

p.d.f. of Exp(1):

$f(y) = \begin{cases} e^{-y}, & y > 0 \\ 0, & \text{elsewhere} \end{cases}$

Theorem

$Y := F^{-1}(U) \sim Exp(1)$.

sample u_0 from $\mathcal{U}(0,1)$, evaluate $y_0 = -\ln u_0$ method

$F(y) = \int_{-\infty}^{y} f(t)\,dt$

$= \begin{cases} 1 - e^{-y}, & y > 0 \\ 0, & \text{elsewhere.} \end{cases}$

$0 = F(-\infty) \le F(y) \le F(\infty) = 1$

$u = F(y) = 1 - e^{-y} \iff e^{-y} = 1 - u$

$\iff -y = \ln(1-u)$

$y = -\ln(1-u)$

$y = F^{-1}(u)$

$= -\ln(1-u)$

Sampling a random variable on computer

- From the python program, we see that **a random variable on computer** is actually **a random number**.
- In the practice, we often need to generate many random numbers (e.g. 1000 random numbers).
- Create a histogram of these random numbers to visualize these random variables.

Sampling a random variable on computer

The following is a Python program (red sentences) for generating one random variable $U \sim \mathcal{U}(0,1)$

- import numpy as np #import the python library 'numpy'
- left,right,size=0,1,1 # three parameters of uniform distribution
- X=np.random.uniform(left,right,size) #generate a $\mathcal{U}(0,1)$ random variables
- print(X) #print the random variable X

Assignment: Try this code by yourself and choose different parameters:left, right, size

Assignment_3.2.ipynb

Assignment_3.2.ipynb > np.random.seed(13)

\+ Code + Markdown | Run All Restart Clear All Outputs | View data Jupyter Variables Outline ... Python 3.11.9

```python
### Import Library
import numpy as np
```

[1] ✓ 0.0s Python

```python
left,right,size=1,2,15 # three parameters of uniform distribution
np.random.seed(39)
X=np.random.uniform(left,right,size) #generate a U(0, 1) random variables
print(X) #print the random variable X
```

[2] ✓ 0.0s Open 'X' in Data Wrangler Python

```
[1.54688916 1.79789902 1.82040188 1.12204987 1.60200201 1.52551458
 1.46390841 1.47144574 1.63271284 1.92566388 1.81550292 1.94444175
 1.91958514 1.4148308  1.82581196]
```

```python
np.random.seed(13)
X=np.random.uniform(left,right,size) #generate a U(0, 1) random variables
print(X) #print the random variable X
```

[3] ✓ 0.0s Open 'X' in Data Wrangler Python

```
[1.77770241 1.23754122 1.82427853 1.9657492  1.97260111 1.45344925
 1.60904246 1.77552651 1.64161334 1.72201823 1.03503652 1.29844947
 1.05851249 1.85706094 1.37285403]
```

Frequency histogram and probability histogram of 10000 standard normal distribution $N(0,1)$ random variables

```
import numpy as np 数学函数.
import pandas as pd 数据
import matplotlib.pyplot as plt plot.
# Generate 10000 standard normal random variables and store them in X
mu,sigma,size=0,1,10000
X = pd.Series(np.random.normal(mu,sigma,size))
# Create frequency histogram of these 10000 random variables.
bins=50
n,bins,patches  = plt.hist(X,bins, facecolor='blue', edgecolor='black')
plt.title('Frequency histogram of X')
plt.xlabel('X')
plt.ylabel('Frequency')
plt.show() ⇒ plt.show(Block=True) stay for a while.
# Create probability histogram of these 10000 random variables.
bins=50
n,bins,patches  = plt.hist(X,bins, facecolor='blue', edgecolor='black',density='true')
plt.title('Probability histogram of X')
plt.xlabel('X')
plt.ylabel('Probability')
plt.show()
```

一组向量

等分成50组

横纵坐标颜色.

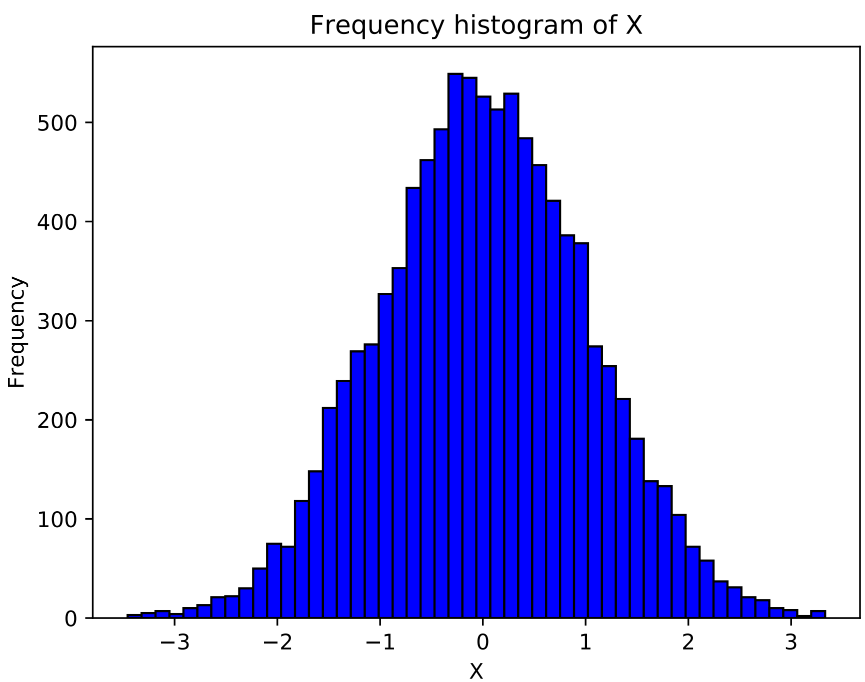
50组

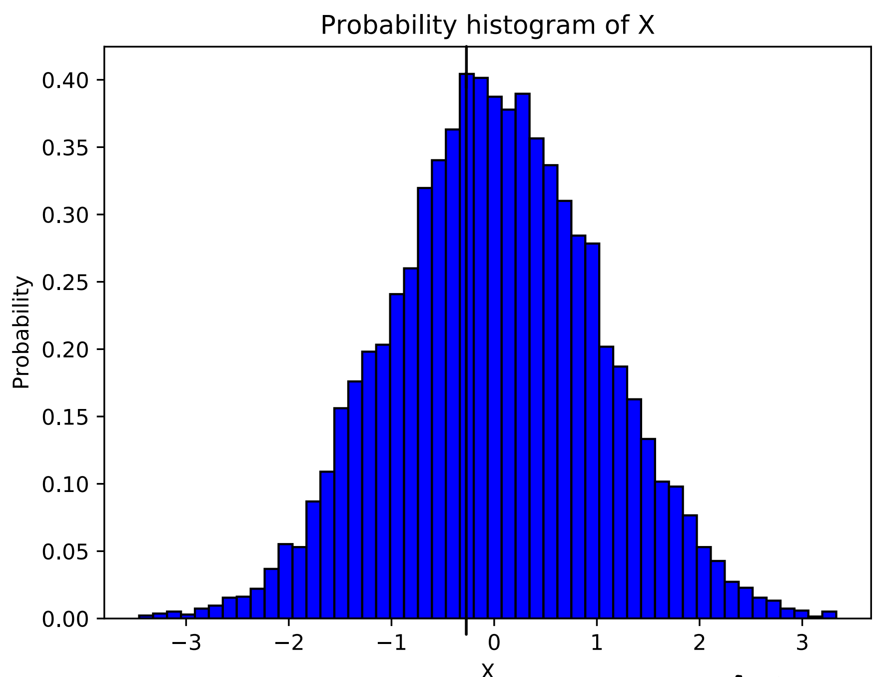
-4 4

$\frac{640}{10000} = 0.064$ maxed for one bar

$8/50 = 0.016$ for one bar

Frequency histogram and probability histogram of 10000 standard normal distribution $N(0,1)$ random variables





highest centralized around 0
as Standard normal sampled

Python Program for Example 5

generate Exp(1) by generating uniform random variables.

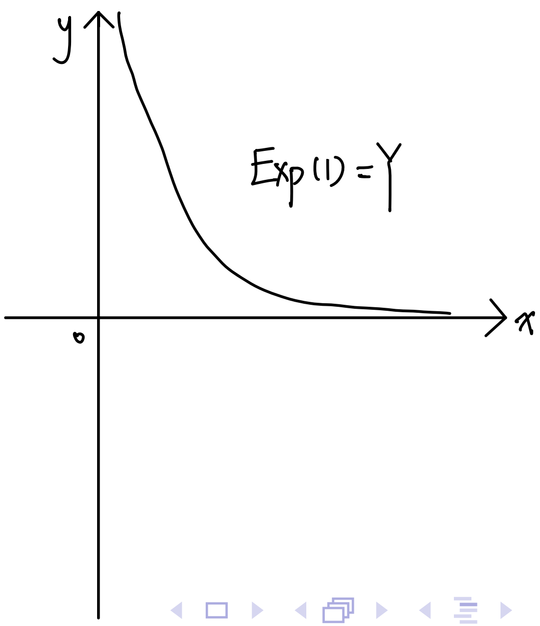
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# Generate 10000 standard normal random variables and store them in X
left,right,size=0,1,10000
U = pd.Series(np.random.uniform(left,right,size))
X=-np.log(U)
# Create probability histogram of Exp(1)
bins=50
n,bins,patches  = plt.hist(X,bins, facecolor='blue', edgecolor='black',density='true')
plt.title('The histogram Exp(1) by X=-log(U)')
plt.xlabel('X')
plt.ylabel('Probability')
plt.savefig("EPHist.pdf")
plt.show()
```

uniform

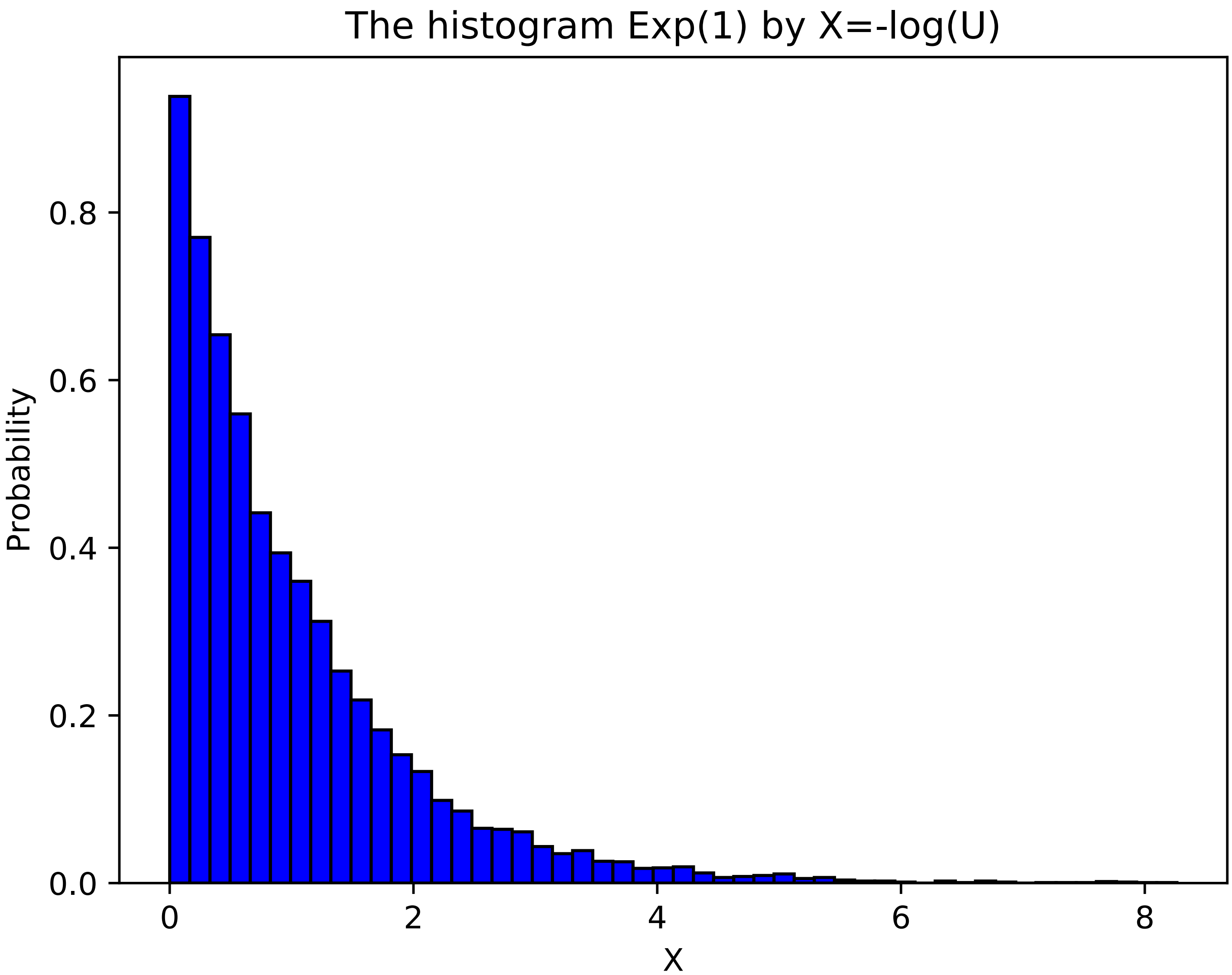
$\sum P = 1$

$-\log U \sim Exp(1)$

where $U \sim \mathcal{U}([0,1])$



Probability histogram of Exp(1) in Example 5



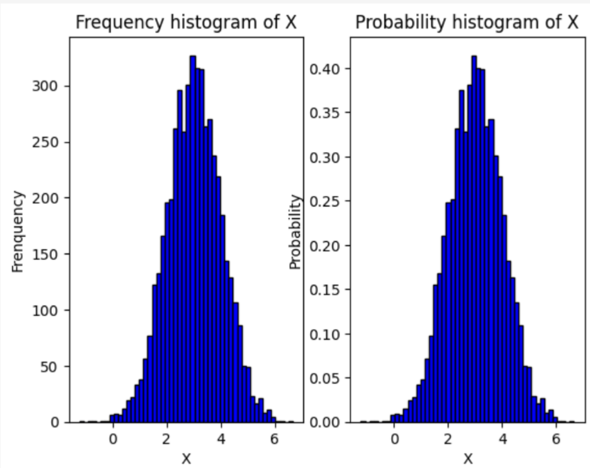
Assignment: Run two codes on your own computer and change the parameters therein.

Assignment_3.3.ipynb > ...

+ Code + Markdown | Run All Restart Clear All Outputs | View data Jupyter Variables Outline ... Python 3.11.9

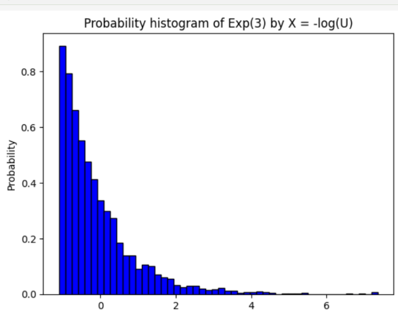
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# Generate 5000 standard normal random variables and store them in X
mu,sigma,size = 3,1,5000
X = pd.Series(np.random.normal(mu,sigma,size))
# Create frequency histogram of these 10000 random variables.
bins = 50
plt.subplot(1,2,1)
n,bins,patches = plt.hist(X,bins,facecolor="blue",edgecolor="black")
plt.title('Frequency histogram of X')
plt.xlabel('X')
plt.ylabel('Frenquency')
# Create probability histogram of these 10000 random variables.
plt.subplot(1,2,2)
n,bins,patches = plt.hist(X,bins,facecolor="blue",edgecolor="black",density="true")
plt.title('Probability histogram of X')
plt.xlabel('X')
plt.ylabel('Probability')
plt.subplots_adjust()
plt.show()
```

[1] ✓ 0.6s Python




```python
# Generate 3000 uniform random variables and store them in U
left,right,size = 0,3,3000
U = pd.Series(np.random.uniform(left,right,size))
X = -np.log(U)
# Create probability histogram of Exp(3)
bins = 50
n,bins,patches = plt.hist(X,bins,facecolor="blue",edgecolor="black",density="true")
plt.title('Probability histogram of Exp(3) by X = -log(U)')
plt.xlabel('X')
plt.ylabel('Probability')
plt.savefig("EPHist.pdf")
plt.show()
```

[2] ✓ 0.1s Python



We have learned how to sample a **one** dimensional random variable for a given cdf F, in the following way:

$$X = F^{-}(U)$$

where $U \sim \mathcal{U}(0,1)$ and

$$F^{-}(u) = \inf\{x : F(x) \geq u\}, \qquad u \in [0,1].$$

Problem: This method only works for **one** dimensional random variable!

Question: How to sample a **multi-dimensional** random variable?

Accept-reject method

- We know a (possibly multi-dimensional) distribution whose probability density function (abbreviated as pdf) is known as f. We shall use accept-reject method to generate a random variable X (by computer) with this distribution.
- The basic idea of accept-reject method is as the following:
  - first generate a random variable Y which has a $\boxed{\text{pdf } g}$ and can be easily sampled;
  - then compare U and $f(Y)/g(Y)$.

Accept-reject method

Assumption: Let the pdfs f and g satisfy the following property: there exists a constant $M > 0$ such that

$f(x) \neq 0$

$$\boxed{\frac{f(x)}{g(x)} \leq M, \quad \forall\, x.}$$

$f(x) \lesssim g(x)$, $\forall x$

$$\boxed{\frac{f(x)}{Mg(x)} \leq 1 \quad \text{for } \forall x}$$

Reason: If M is very large. $\frac{f(x)}{Mg(x)} \to 0$ $\therefore$ higher probability of rejection.

M more exact efficiency higher.

Algorithm: Accept-reject method

1 Sample a random variable $Y \sim g$, $U \sim \mathcal{U}(0, 1)$;

2 If $U \leq \frac{f(Y)}{Mg(Y)}$, **accept**, i.e., take $X = Y$ and stop;

3 If $U > \frac{f(Y)}{Mg(Y)}$, **reject**, i.e., do nothing but return to step 1.

Accept-reject method

Theorem

Let f and g satisfy the assumption in the previous slide, then the random variable produced by the algorithm in the previous slide has a distribution with the density f.

Proof.

We only show the theorem for one dimensional case. It suffices to show that

$$\mathbb{P}\left(Y \leq x \middle| U \leq \frac{f(Y)}{Mg(Y)}\right) = \mathbb{P}(X \leq x), \quad \forall \quad x \in \mathbb{R}.$$

Let us compute the conditional probability on the left hand. $\square$

Accept-reject method

Conditional Probability.

$$\mathbb{P}\left(Y \le x \middle| U \le \frac{f(Y)}{Mg(Y)}\right) \overset{\downarrow}{=} \frac{\mathbb{P}(Y \le x, U \le \frac{f(Y)}{Mg(Y)})}{\mathbb{P}(U \le \frac{f(Y)}{Mg(Y)})}$$

Since Y and U are independent

$\therefore f(y,u) = f_1(y) \cdot f_2(u)$

$f_1(y) \to f(y)$, $f_2(u) \to \begin{cases} 1, & u \in [0,1] \\ 0 & \text{anywhere} \end{cases}$

$$= \frac{\int_{-\infty}^{x} \left(\int_0^{\frac{f(y)}{Mg(y)}} \mathrm{d}u\right) g(y)\mathrm{d}y}{\int_{-\infty}^{\infty} \left(\int_0^{\frac{f(y)}{Mg(y)}} \mathrm{d}u\right) g(y)\mathrm{d}y} = \frac{\int_{-\infty}^{x} \frac{f(y)}{M\cancel{g(y)}} \cancel{g(y)}\mathrm{d}y}{\int_{-\infty}^{\infty} \frac{f(y)}{M\cancel{g(y)}} \cancel{g(y)}\mathrm{d}y}$$

$$= \frac{\int_{-\infty}^{x} f(y)\mathrm{d}y}{\int_{-\infty}^{\infty} f(y)\mathrm{d}y} = \int_{-\infty}^{x} f(y)\mathrm{d}y = \mathbb{P}(X \le x).$$

Example: Generate Gamma distribution

$u = F(y) = \int_{-\infty}^{y} f(t)\,dt$

$y = F^{-1}(u)$

- Gamma(α, β) distribution density: $f(x) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$ for $x \geq 0$;
- When $\alpha = 1$, it is an Exp$(1/\beta)$ distribution, whose random variable can be generated by $X = -\beta \log U$;
- When $\alpha \neq 1$, we cannot use the general inverse method because the inverse function often does not have an explicit form.
- When $\alpha \neq 1$, we can use an accept-reject method.

Example: Generate Gamma distribution

Let $\alpha = 2.5$ and $\beta = 2$, then

$$f(x) = \frac{2^{2.5} x^{1.5} e^{-2x}}{\Gamma(2.5)}.$$

We use Exp(1) distribution as a reference, i.e.,

$$g(x) = e^{-x}.$$

It is easily see that

$$\frac{f(x)}{g(x)} = \frac{2^{2.5} x^{1.5} e^{-x}}{\Gamma(2.5)} \leq 4$$

Example: Generate Gamma distribution

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
X = [ ]
size=10000
Y = np.random.exponential(1,size)
U = np.random.uniform(0,1,size)
a = 2**(2.5)/math.gamma(2.5)
f = a*(Y**1.5)*np.exp(-2*Y)
g = np.exp(-Y)
M = 4
for i in range(size):
    if U[i]<=f[i]/(M*g[i]):
        X.append(Y[i])
# Create histogram of these 10000 random variables.
# Create probability histogram of these 10000 random variables.
bins=100
n,bins,patches  = plt.hist(X,bins, facecolor='blue', edgecolor='black',density='true')
plt.title('Probability histogram of X')
plt.xlabel('X')
plt.ylabel('Probability')
plt.savefig("NPHist.pdf")
plt.show()
```

Exp(1) ~ x

g(x)

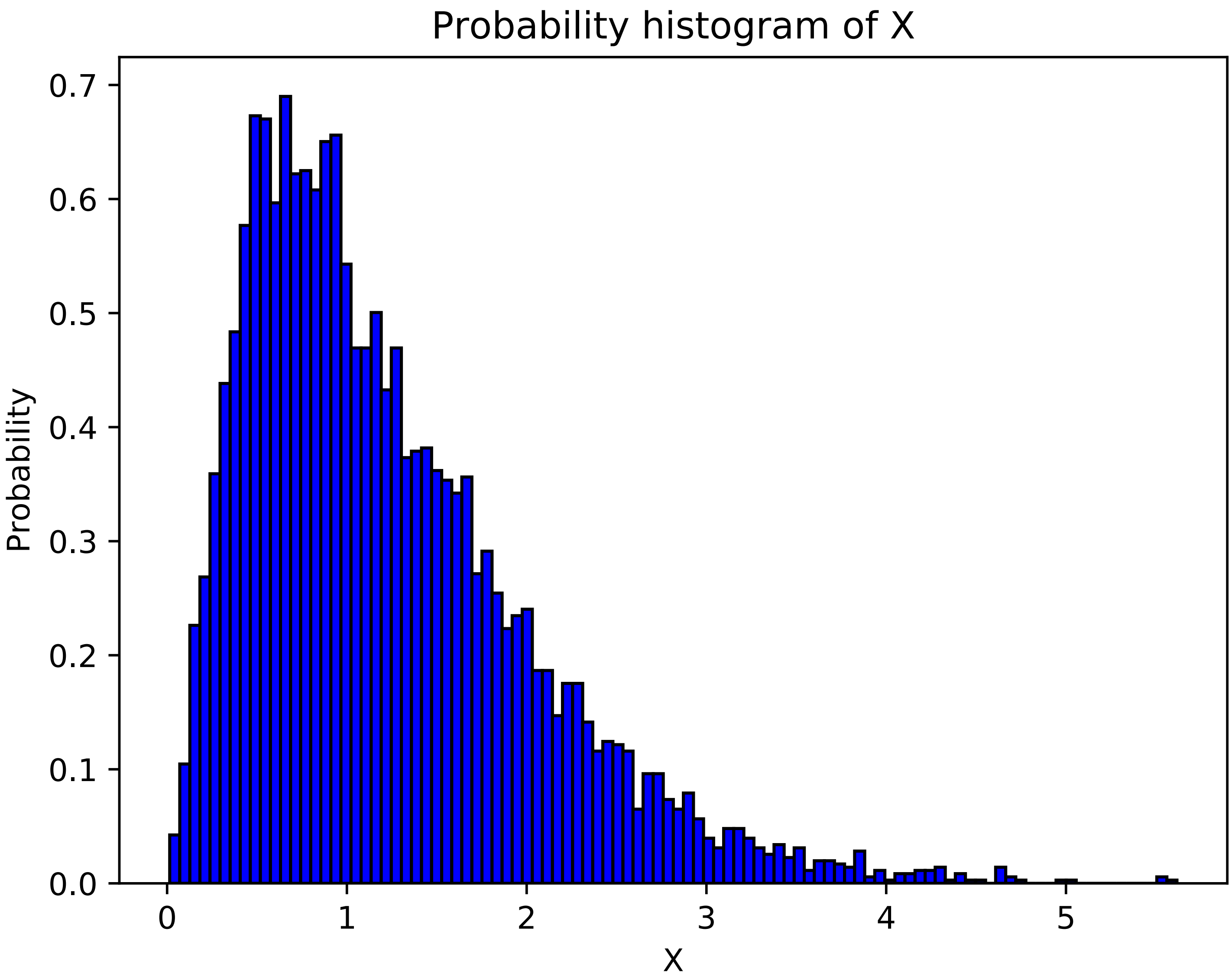
$\frac{f(x)}{g(x)} \le M$

$\frac{[\max f(x)]}{U[0,?]} \le M$

while $Y_i \le 10000$:

$f(x) \le B$

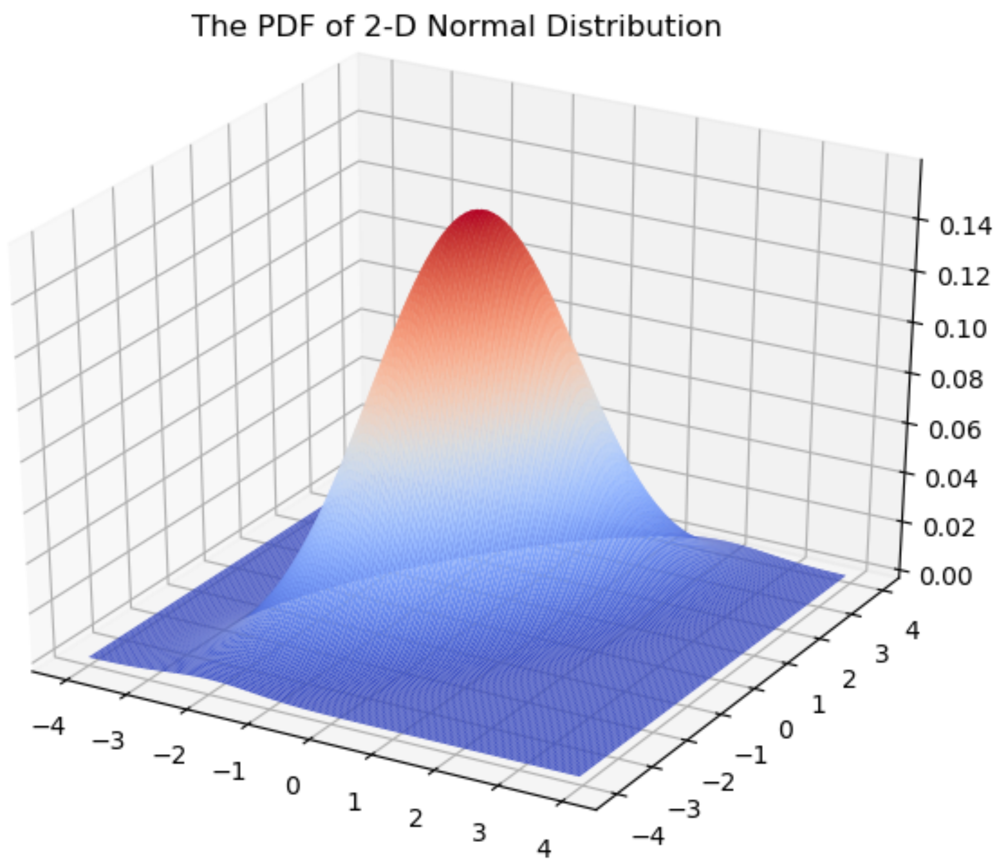
Example: Generate Gamma distribution



Example: 2D normal distribution

```
import matplotlib.pyplot as plt
import numpy
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm  #### get color map
import numpy as np
from scipy.stats import multivariate_normal as mvnorm

#### define the range of axises ####
x, y = np.mgrid[-4:4:.05, -4:4:.05]
pos = np.dstack((x, y))

#### define 2-D normal rv ####
mean = np.array([0,0])
cov  = np.array([[1,1],[1,2]])

rv = mvnorm(mean,cov)
Y = rv.pdf(pos)

#### define 3D figure ####
fig = plt.figure()
ax = Axes3D(fig)
ax.plot_surface(pos[:,:,0], pos[:,:,1], Y, rstride = 1, cstride = 1, cmap = cm.coolwarm)
fig.suptitle("The PDF of 2-D Normal Distribution")
plt.savefig('2D Norm')
plt.show()
```

Example: 2D normal distribution



Assignment: Use the accept-reject method to make a python program which will draw a probability histogram of this 2D normal distribution.

```python
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
# Sample data: List of tuples (x, y)
# Target Bivariate Normal Distribution Parameters
mu_x, mu_y = 0, 0  # Means
sigma_x, sigma_y = 1, 1  # Standard deviations
rho = 0.8  # Correlation coefficient

# Proposal Distribution (Independent Normal)
def proposal():
    return np.random.normal(0, 1), np.random.normal(0, 1)

# Bivariate Normal PDF
def bivariate_normal(x, y):
    z = ((x - mu_x)**2 / sigma_x**2 + (y - mu_y)**2 / sigma_y**2 -
         2 * rho * (x - mu_x) * (y - mu_y) / (sigma_x * sigma_y))
    norm_factor = 2 * np.pi * sigma_x * sigma_y * np.sqrt(1 - rho**2)
    return np.exp(-z / (2 * (1 - rho**2))) / norm_factor

# Accept-Reject Sampling
M = 1.5  # Chosen upper bound for f(x, y) / g(x, y)
n_samples = 10000
samples = []

while len(samples) < n_samples:
    x, y = proposal()
    u = np.random.uniform(0, 1)
    if u < bivariate_normal(x, y) / (M * (1 / (2 * np.pi))):
        samples.append((x, y))

samples = np.array(samples)

x_vals, y_vals = samples[:, 0], samples[:, 1]  # Unpack x and y values

# Define histogram bins
bins = 50  # Number of bins in each dimension
hist, x_edges, y_edges = np.histogram2d(x_vals, y_vals, bins=bins, density=True)  # Normalize to probability

# Prepare 3D bar positions
x_pos, y_pos = np.meshgrid(x_edges[:-1], y_edges[:-1], indexing="ij")
x_pos = x_pos.ravel()
y_pos = y_pos.ravel()
z_pos = np.zeros_like(x_pos)

# Bar dimensions
dx = dy = (x_edges[1] - x_edges[0]) * np.ones_like(z_pos)
dz = hist.ravel()  # Heights (probability)

# Create 3D figure
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')

# Plot 3D probability histogram
ax.bar3d(x_pos, y_pos, z_pos, dx, dy, dz, color='cyan', alpha=0.7, edgecolor='black')

# Labels and title
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Probability')
ax.set_title('3D Probability Histogram by accept-reject method of 2D normal distribution.')

# Adjusting view angle for better visualization
ax.view_init(elev=30, azim=45)

plt.show()
```

✓ 0.9s Python

3D Probability Histogram by accept-reject method of 2D normal distribution.