

ENGG1003 - Thursday Week 9

Random numbers from normal distributions
—aka random numbers from Gaussian distributions

Steve Weller

University of Newcastle

6 May 2021

Last compiled: May 6, 2021 3:36pm +10:00

Lecture overview

1 normal distribution

- ▶ also known as *Gaussian distribution* or “bell curve”
- ▶ today: draw random samples from “standard” normal distribution

2 compute probabilities using normal distribution

- ▶ uses numerical integration

1) Normal distribution

- introduced *uniformly* distributed random numbers in week 4
- today: introduce “standard” normal distribution
 - ▶ extend next week to general form
 - ▶ also known as random numbers from *Gaussian distribution*
 - ▶ occur *very* widely in all branches of Engineering
- two goals today using standard normal distribution:
 - 1 use Python to draw random samples
 - 2 use Python to compute probability of random number falling in specified range

Standard normal distribution

- generate 100,000 random numbers using `normal()` function in numpy's random library
 - ▶ standard normal: first two parameters in call to `normal()` are 0.0 and 1.0

```
x = np.random.normal(0.0, 1.0, size=100000)
```

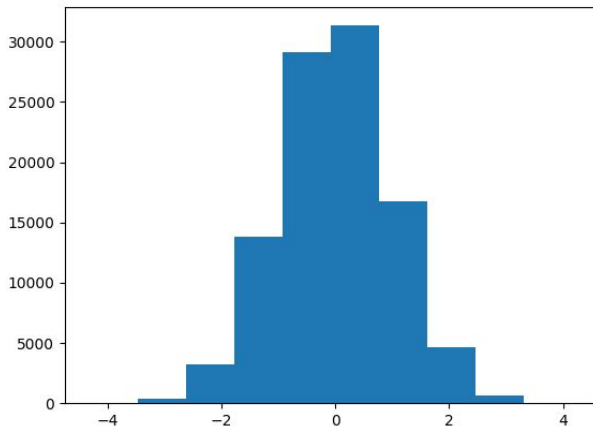
- use `hist()` function in matplotlib library to generate histograms of observed data
 - ▶ 10 bins
 - ▶ 100 bins

Python code: generate histograms

histdemo.py

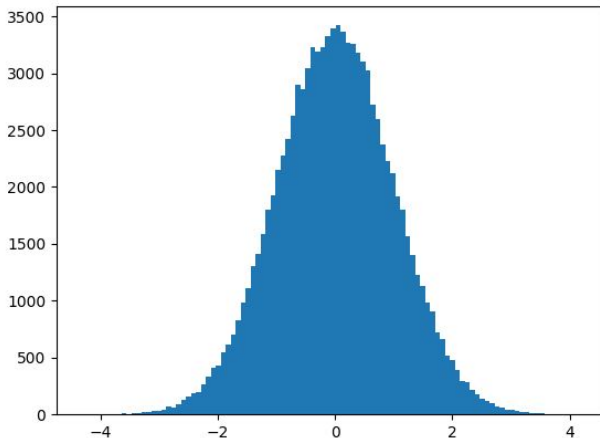
```
1 # histdemo
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 np.random.seed(1)
6 d = np.random.normal(0.0, 1.0, size=100000)
7
8 x = np.linspace(-5,5,num=1000)
9 f = 1/(np.sqrt(2 * np.pi)) * np.exp(-x**2 / 2)
10
11 plt.hist(d, 100, density=True)
12 plt.plot(x, f, color='r', linewidth=3)
13 #plt.hist(d, 100)
14
15 #plt.plot(d, 'o')
16 plt.show()
```

Histogram: 10 bins



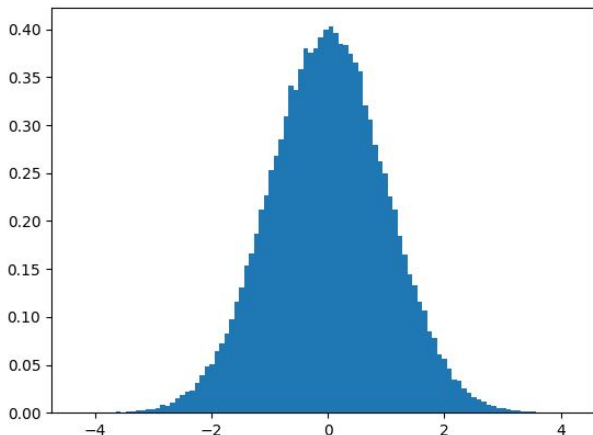
- height of each rectangle reflects number of samples in each “bin”

Histogram: 100 bins



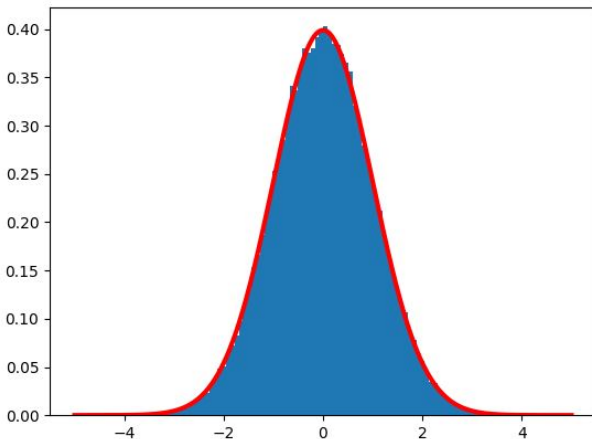
- identical data set as for 10 bins

Normalized histogram (area 1), 100 bins



- same histogram, except total area of rectangles is normalized to be 1

Normalized histogram with PDF



red curve is *probability density function (PDF)*

Standard normal distribution

Standard normal probability density function:

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

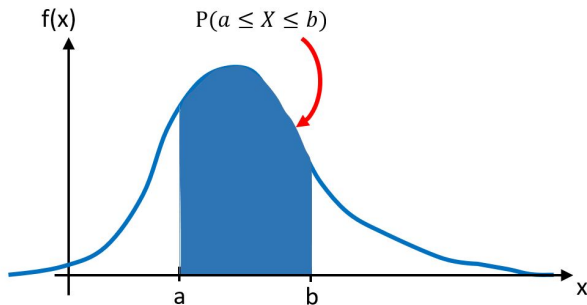
- *standard* normal distribution is a special case of distribution we'll see next week
- corresponds to parameters $\mu = 0$ and $\sigma = 1$

```
x = np.random.normal(0.0, 1.0, size=100000)
```

Probability density functions

If X is a random number drawn from a distribution with PDF $f(x)$, probability X takes a value in interval $[a, b]$ is

$$P(a \leq X \leq b) = \int_a^b f(x)dx$$



Properties of PDFs

Two key properties of any probability density function:

1 non-negativity

$$f(x) \geq 0 \text{ for all } x$$

▶ since probability can't be negative

2 normalization

entire area under $f(x)$ must be equal to 1, since

$$P(-\infty \leq X \leq \infty) = \int_{-\infty}^{\infty} f(x)dx = 1$$

▶ reason for the $1/\sqrt{2\pi}$ factor in standard normal PDF

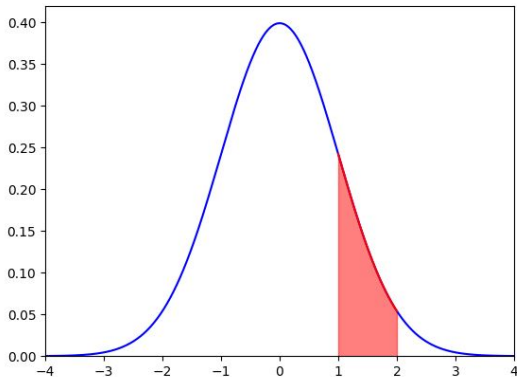
2) Computing probabilities using standard normal distribution

- probability of random number X drawn from standard normal distribution taking value in $[a, b]$:

$$P(a \leq X \leq b) = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-x^2/2} dx$$

- no exact expression exists for $\int_a^b e^{-x^2/2} dx$
 - ▶ need to use numerical integration!
- **Example:** use trapezoidal method to approximate $P(1 \leq X \leq 2)$ when X is drawn from standard normal distribution

Example: fraction of standard normal numbers in $[1, 2]$



$$\text{Red shaded area} = \frac{1}{\sqrt{2\pi}} \int_1^2 e^{-x^2/2} dx \approx 0.1359$$

Python code: fraction of numbers in $[1, 2]$

standardnormal.py

```
1 # standardnormal
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 def f(x):
6     return 1/(np.sqrt(2 * np.pi)) * np.exp(-x**2 / 2)
7
8 def trapezoidal(f, a, b, n):
9     h = (b-a)/n
10    f_sum = 0
11    for i in range(1, n, 1):
12        x = a + i*h
13        f_sum = f_sum + f(x)
14    return h*(0.5*f(a) + f_sum + 0.5*f(b))
```

- lines 5–6: PDF of standard normal distribution

Python code

standardnormal.py—continued

```
1 a = 1
2 b = 2
3 probab_ab = trapezoidal(f, a, b, 100)
4 print('Probability X in range [{},{}] is: {:.4f}'.format(a, b,
    probab_ab))
5
6 x = np.linspace(-4, 4, 1000)
7 xab = np.linspace(a, b, 100)
8
9 plt.plot(x, f(x), 'b')                                # standard normal pdf
10 plt.plot(xab, f(xab), 'r')
11 plt.fill_between(xab, f(xab), color='r', alpha=0.5) #alpha=
    transparency
12 plt.axis([-4, 4, 0, 0.42])
13 plt.show()
```

- line 3: approximate $\frac{1}{\sqrt{2\pi}} \int_1^2 e^{-x^2/2} dx$, 100 panels
- line 7: $1 \leq x \leq 2$ for red shaded area plot

Live demo: standard normal generation

- draw $N = 10^6$ random numbers from standard normal distribution
- for large N , expect fraction of random numbers X in range $[1, 2]$ to be close to

$$P(1 \leq X \leq 2) = \frac{1}{\sqrt{2\pi}} \int_1^2 e^{-x^2/2} dx \approx 0.1359$$

- live demo of `standardnormaldemo.py`
- observed fraction ≈ 0.136

Python code

standardnormaldemo.py

```
1 # standardnormaldemo
2 import numpy as np
3
4 N = 1000000
5 x = np.random.normal(0.0, 1.0, size=N)
6 a = 1
7 b = 2
8 num_ab = 0
9 for k in range(0, len(x)):
10     if a <= x[k] <= b:
11         num_ab += 1
12
13 print('{} standard normal random numbers'.format(N))
14 print('Fraction of random numbers in range [{},{}] = {:.4f}'.
       format(a, b, num_ab/N))
```

- lines 8–12: `for` loop counts number of random numbers x satisfying $1 \leq x \leq 2$

Lecture summary

1 standard normal distribution

- ▶ drawing N random samples using
`numpy.random.normal(0.0, 1.0, size=N)`

2 compute probability $P(a \leq X \leq b)$ for standard normal distribution

- ▶ needs numerical integration using PDF

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-x^2/2} dx$$

3 generalise standard normal distribution, and applications