

2 - Utility and investor types

Espen Sirnes

2025-08-10

Table of contents

1	Expected Utility	1
1.1	Expectation of the normal distribution	2
1.2	A discrete example	4
1.3	A continuous example	7
1.4	The objective function	10
2	Optimal Portfolio with One Asset	11
2.1	Investor Types	11
2.2	Assumptions in a Financial Market	12
3	The Risk Premium	13
3.1	Absolute Risk Aversion (ARA)	13
3.1.1	Constant Absolute Risk Aversion (CARA)	14
3.2	Relative Risk Aversion (RRA)	14
4	The Normal Distribution and risk	15
5	Literature	20

1 Expected Utility

In finance, the utility function is typically defined as the utility derived from a certain amount x . It is generally assumed that greater wealth yields higher utility, hence the utility function should be increasing, meaning $U'(x) > 0$. In situations involving risk, such as holding shares with uncertain future

values, it becomes useful calculate the expected utility of a set of possible outcomes x_i with probabilities $P(x_i)$ and utilities $u(x_i)$ (Neumann and Morgenstern 1947).

This builds on the general notion of expectations in statistics. For a set of outcomes x_i and probabilities $P(x_i)$ for example, the expected value is $X = \{x_0, x_1, \dots, x_n\}$ is

$$\mathbb{E}[X] = \sum_{i=0}^N P(x_i)x_i \quad (1)$$

For example

```
import numpy as np

p = np.array([0.5, 0.2, 0.3])
x = np.array([3, 10, 20])

np.sum(p*x)
```

```
np.float64(9.5)
```

In the same way, the discrete expected utility for a set of outcomes x_i with probabilities $P(x_i)$ is defined as follows:

$$U(X) = \mathbb{E}[u(X)] = \sum_{i=0}^N P(x_i)u(x_i) \quad (2)$$

where $P(x_i)$ is the probability that X assumes the value x_i , and $u(x_i)$ represents the utility derived if that event occurs.

For example, say utilities 2,-4 and 6 occur with probabilities 0.5,0.2 and 0.3, then the expected utility is

```
import numpy as np

p = np.array([0.5, 0.2, 0.3])
u = np.array([2, -4, 6])

U = np.sum(p*u)
U
```

```
np.float64(1.9999999999999998)
```

For a continuous distribution, the idea is the same. We multiply each utility state $u(x_i)$ with the probability it occurs $f(x)dx$. The integral sign $\int_{-\infty}^{\infty} \dots$ (Leibniz 1969), is the equivalent of a sum for continuous variables. The expected utility is therefore given by:

$$U(X) = \mathbb{E}[u(X)] = \int_{-\infty}^{\infty} u(x)f(x) dx, \quad (3)$$

where $f(x)$ is the density function of X , such as the normal distribution.

1.1 Expectation of the normal distribution

To make this a bit more tangible, let us assume $f(x)$ represents the standard normal distribution (variance=1 and mean=0), and assume we collect the outcomes x into five intervals. According to the normal distribution (Gauss 1809), each outcome will then have the probabilities shown in the following figure (the code to is too long to have in this document, but it can be found in the local module `utility.py` in this repository):

Figure 1:

```
import utility
import numpy as np
dx = 2
maxmin = 4
fig, ax, p, x = utility.normaldist(dx, maxmin)

print(f"Probabilities: {np.round(dx*p,3)}")
print(f"Outcomes: {np.round(x,2)}")
```

```
Probabilities: [0.    0.108 0.798 0.108 0.    ]
Outcomes: [-4 -2  0  2  4]
```

Normal Distribution and corresponding discrete probabilities ($dx = 2$)

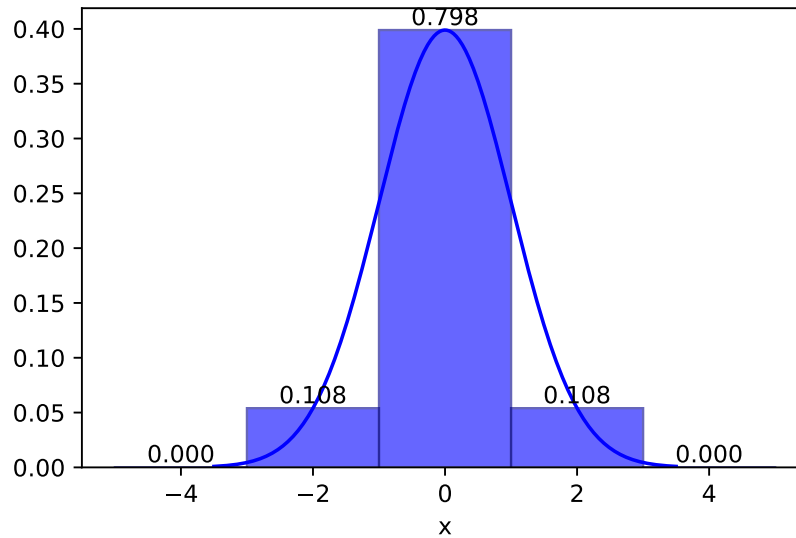


Figure 1: Discrete version of the normal distribution

The argument dx is the width of the bars in the figure, and $maxmin$ determines which range of values should be plotted. $maxmin = 4$ means that values from -4 to 4 are plotted.

The probability of each outcome (bar) and the associated outcomes, are returned as p and x , respectively.

The probabilities do not sum exactly to one, as expected for a probability distribution. This discrepancy arises due to the coarse nature of the discrete categories.

Coding Challenges:

- Challenge 1: Reduce the interval size dx and sum the probabilities to observe if the total approaches 1.
- Challenge 2: Now, assume the utility function is $f(x) = -e(-\rho x)$ (the natural logarithm). What is the expected utility for different values of dx ?

As demonstrated above, decreasing the interval size dx brings the discrete distribution closer to the continuous one. The continuous distribution serves

as an approximation when outcomes are real numbers, which is often the case in investment scenarios.

The primary advantage of continuous variables over discrete ones is that we can apply powerful mathematical tools, such as derivatives and integrals, from calculus.

1.2 A discrete example

A key outcome of this model is that with a concave utility function, an individual will always prefer a guaranteed cash flow over a speculative investment with the same expected return. This behavior is illustrated by the following example.

Consider an individual with an exponential (Euler 1748) utility function:

$$u(x) = -e^{-\rho \cdot x} \quad (4)$$

which is increasing and concave, as shown by its first and second derivatives:

$$u'(x) = -\rho \cdot u(x) > 0 \quad (5)$$

$$u''(x) = \rho^2 \cdot u(x) < 0 \quad (6)$$

Suppose there is a 50% chance of $x = 0.5$ and a 50% chance of $x = 1.5$. That is, the possible outcomes are given by $X = \{0.5, 1.5\}$, with associated probabilities $P(X) = \{0.5, 0.5\}$. The expected value is then $\frac{1}{2}(-e^{-\rho \cdot 0.5} - e^{-\rho \cdot 1.5})$.

Below is the utility function defined as the function `ufunc(x, rho)`, and both the expected utility and the utility of the expected value are calculated, assuming $\rho = 1.0$:

```
import numpy as np

# Values and probabilities
x_gamble = np.array([0.5, 1.5]) # Outcomes of the
                                # gamble
p_gamble = np.array([0.5, 0.5]) # Probabilities
RHO = 5.0
```

```

#utility function
def u_func(x, rho):
    return -np.exp(-rho*x)

# Expected utility
expected_utility = np.sum(p_gamble *
                          u_func(x_gamble, RHO))
expected_value = np.sum(p_gamble *
                        x_gamble)
utility_exp_value = u_func(expected_value, RHO)

print(f"Expected utility: {expected_utility}")
print(f"Expected value: {expected_value}")
print(f"Utility of expected value: {utility_exp_value}")

```

Expected utility: -0.04131904149702332

Expected value: 1.0

Utility of expected value: -0.006737946999085467

By comparing the utility of the expected value of 1.0 with the expected utility, you can verify that the utility of expected value is greater than the expected utility:

$$u(EX) = e^{\rho \cdot 1} > \frac{1}{2}e^{\rho \cdot 0.5} + \frac{1}{2}e^{\rho \cdot 1.5} = \mathbb{E}u(X), \quad (7)$$

This example demonstrates that the utility of a certain outcome ($-e^{\rho}$) is preferred over the expected utility of a gamble. This preference underscores risk aversion. The situation is illustrated in the following plot:

```

import utility
fig,ax = utility.plot(RHO, x_gamble, p_gamble)

```

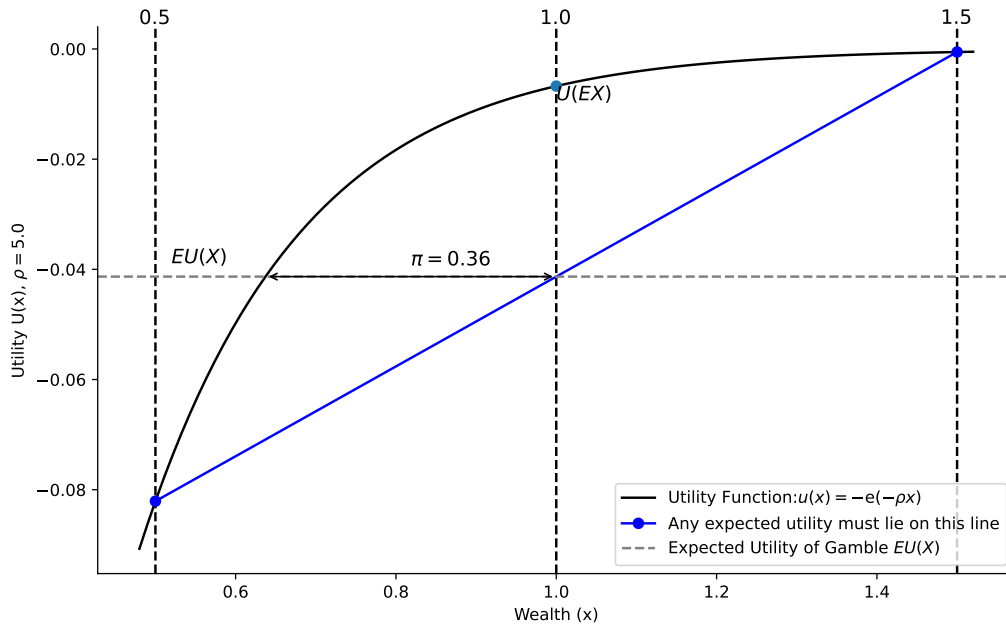


Figure 2: Graphical illustration of the risk premium

The code is a bit too long to be included here, but can be found in the local module `utility.py` in this repository.

As we can see, the utility at wealth 1 is 0, while the mean utility is lower. If you draw a line between the utility of 0.5 and 1.5, you will see that the indicated expected utility, is where that line intersects 1.

Hence, with a concave utility function, a person will always prefer a sure amount rather than a bet with the same expected payoff. In this case, the risk-averse investor will invariably prefer the certain payment of 1 over the gamble.

The difference indicated by ρ in the figure, is the certainty equivalence. It is the amount that you would need to compensate the investor, in order to take the gamble. In a real market, this is the premium that investors demand to hold risky assets to safe bills. It is usually about 2-4% on average for the whole market.

1.3 A continous example

Let us now calculate the expected utility with more than two outcomes. We remember from the calculation of Figure 1 and calculation of discrete expectations in Equation 2 that a discrete version of the continuous normal distribution can be calculated as

```
import utility
import numpy as np

dx = 2
minmax = 4
RHO = 5
fig, ax, p, x = utility.normaldist(dx, minmax)
expected_utility = np.sum(p*dx*utility.u_func(x, RHO))

print(f'Exact expected utility:{-np.exp(-RHO*(0.0-0.5*RHO*1.0))}')
print(f'Approximate expected utility:{expected_utility}')
```

Exact expected utility:-268337.2865208745

Approximate expected utility:-132238.79351363966

Normal Distribution and corresponding discrete probabilities (dx = 2)

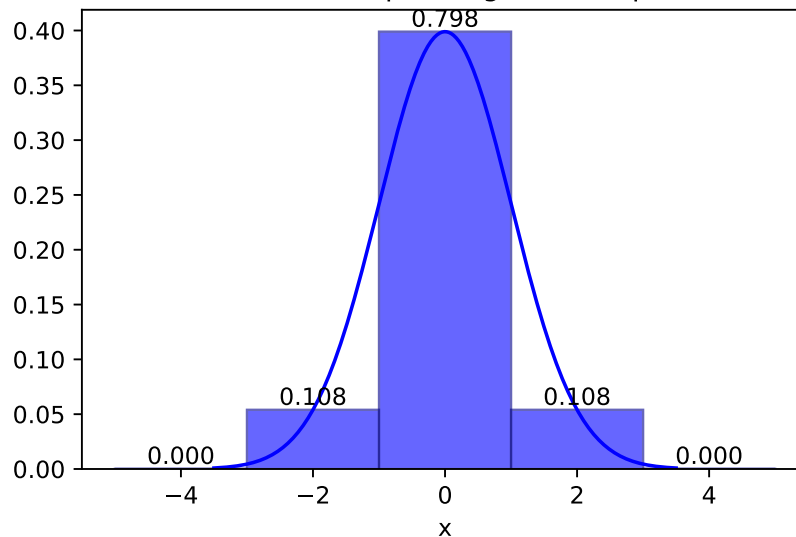


Figure 3: Graphical illustration of the risk premium

Given these discrete probabilities and outcomes, we can calculate the expected utility of $u(x) = -e(-\rho x)$ by multiplying probabilities by outcomes in terms of utility:

```
import utility

expected_utility = np.sum(p*dx*utility.u_func(x, RHO))

print(f'Approximate expected utility:{expected_utility}')
```

Approximate expected utility:-132238.79351363966

Now, the normal distribution is defined as

$$\frac{1}{\sigma\sqrt{2\rho}}e^{-\frac{(x-\mu)^2}{2\sigma}} \quad (8)$$

If we calculate the expected utility by taking the integral, it turns out that

$$\mathbb{E}[u(X)] = \int_{-\infty}^{\infty} u(x)f(x) dx = e^{-\rho(\mu-\frac{1}{2}\rho\sigma^2)} \quad (9)$$

Let us now try to compare this with our approximated expected utility above. If we have a standard normal distribution, so $\sigma = 1$ and $\mu = 0$. The risk aversion coefficient is $\rho = 5$. The exact and approximate expected utility is

```
print(f'Exact expected utility:{-np.exp(-RHO*(0-0.5*RHO*1.0))}')
```

```
expected_utility = np.sum(p*dx*utility.u_func(x, RHO))
print(f'Approximate expected utility:{expected_utility}')
```

Exact expected utility:-268337.2865208745

Approximate expected utility:-132238.79351363966

As we can clearly see from above, the approximation is not good. The problem is that the discrete normal probability function that we have used, is too coarse. We therefore need to try smaller intervals for the bars. Also, we want to calculate a wider range than +/-4. Let us try with bar width of 0.25 and a range +/- 8:

```

import utility
import numpy as np

dx = 0.25
minmax = 8
RHO = 5
fig, ax, p, x = utility.normaldist(dx, minmax)
expected_utility = np.sum(p*dx*utility.u_func(x, RHO))

print(f'Exact expected utility:{-np.exp(-RHO*(0.0-0.5*RHO*1.0))}')
print(f'Approximate expected utility:{expected_utility}')

```

Exact expected utility:-268337.2865208745
Approximate expected utility:-268105.24655357056

Normal Distribution and corresponding discrete probabilities (dx = 0.25)

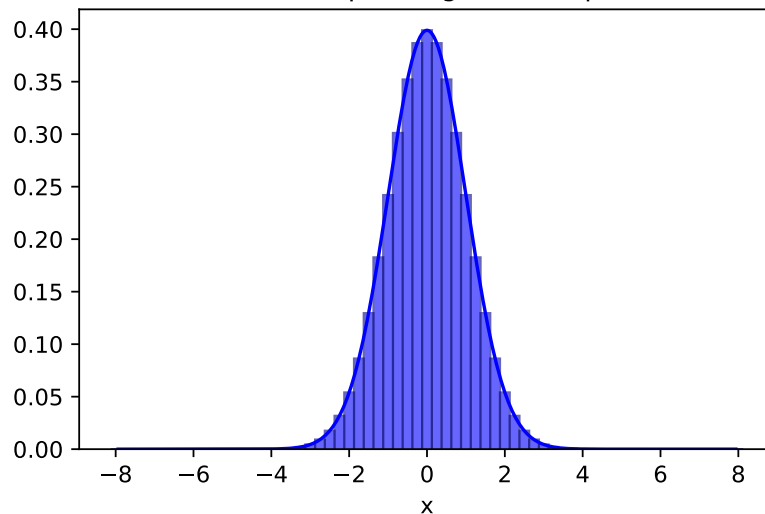


Figure 4: Graphical illustration of the risk premium

We see now in Figure 4 that the discrete distribution is quite similar to the continuous one, and the approximation is also very close to the analytical exact expected utility.

1.4 The objective function

The most important lesson here, is what is inside the brackets in equation (Equation 9). That expression determines in effect the utility, so let us rewrite it here:

$$\mu - \frac{1}{2}\rho\sigma^2 \quad (10)$$

This is therefore usually assumed to be the objective function of any investor.

Hence, if returns are normally distributed, then investors would want to maximize the difference between the mean and the variance. This is true for any utility function if returns are normally distributed, but it is not in general true if returns are distributed differently.

Coding Challenges:

- Challenge 1: Calculate the approximate expected utility with some other parameters and a non standard normal distribution ($\sigma <> 1$ and $\mu <> 1$, and see if the approximate is still a good approximation.

2 Optimal Portfolio with One Asset

With the knowledge from Equation 10, we can actually calculate the optimal level of investment for one risky asset and a risk free allocation (Tobin 1958). We will later see that the method for doing that, is very similar to how we calculate optimal portfolios with many different assets. For now, we will assume we only consider one asset, for example the market index. Say you buy shares for w NOK. Then, we can use the method in the previous section to find that the expected utility depends on

$$w\mu - \frac{w^2}{2}\rho\sigma^2 \quad (11)$$

Taking the derivative of Equation 11, which we assume is a concave and increasing function, we get

$$\frac{d(w\mu - \frac{w^2}{2}\rho\sigma^2)}{dw} = \mu - w\rho\sigma^2 \quad (12)$$

Setting Equation 12 equal to zero and solving for w gives

$$w = \frac{\mu}{\rho\sigma^2} \quad (13)$$

From this, we conclude: 1. Higher risk aversion leads to lesser investment. 2. Greater expected returns encourage more investment. 3. Increased risk (σ^2) discourages investment.

In the next lecture, we will extend these principles to portfolios with multiple assets using matrix algebra.

2.1 Investor Types

The shape of a utility function critically influences how individuals respond to risk. This subsection discusses three primary types of investors based on their risk preferences and the corresponding shapes of their utility functions.

- **Risk-Averse Investors:** These individuals have concave utility functions, as in the example above, indicating a preference for certain outcomes over uncertain ones with the same expected value. Commonly modeled in financial theory, risk-averse investors prioritize minimizing risk over maximizing returns. They tend to diversify their portfolios across various asset classes to reduce volatility. Even with the option of unlimited borrowing, they typically opt to limit their investment exposure.
- **Risk-Neutral Investors:** For risk-neutral individuals, volatility is inconsequential. Their utility functions are linear, reflecting indifference to the level of risk associated with any investment. They focus solely on maximizing expected returns and are likely to invest in the asset with the highest expected payoff, irrespective of the associated risks. This type of investor is willing to allocate as much capital as possible to maximize potential gains.
- **Risk-Loving (Risk-Seeking) Investors:** Risk lovers have convex utility functions and engage in behaviors akin to gambling, where the expected return is typically negative. They derive satisfaction from the risk itself and often pursue investments that offer the highest possible returns, irrespective of the high levels of risk involved. Such behavior is commonly seen in speculative ventures and high-stakes gambling.

Interestingly, it is not uncommon for individuals to display traits of both risk-averse and risk-seeking behaviors, a phenomenon that may seem paradoxical. For instance, the same person might purchase insurance (a risk-averse action) while also indulging in lottery gambling (a risk-seeking behavior). This can be explained by the utility function's varying shape at different levels of wealth or stakes: a person might be risk-seeking with small, disposable amounts of money but risk-averse with larger, life-impacting sums. This dual nature influences how individuals choose to allocate their investments across different risk levels.

2.2 Assumptions in a Financial Market

In financial theory, the assumption typically made about market participants is that they are predominantly risk averse. This assumption is crucial as portfolio optimization and related strategies largely rely on this characteristic. Risk-neutral or risk-loving investors, who either disregard risk or actively seek it, are considered exceptions rather than the norm in these models. This foundational assumption allows for the development of investment strategies that aim to maximize returns while minimizing risk, aligning with the preferences of risk-averse individuals.

3 The Risk Premium

We understand that a risk-averse individual has a concave utility function, indicated by $u''(x) < 0$. But how do we quantify the degree of risk aversion?

A direct approach might be to use the second derivative of the utility function, $u''(x)$, as a measure of risk aversion. However, this method has limitations, particularly at higher wealth levels. The issue arises because the curvature of the utility function tends to flatten as wealth increases, implying a decrease in relative risk aversion. If we solely relied on $u''(x)$ for measuring risk aversion, it might inaccurately suggest that wealthier individuals become nearly risk-neutral, due to the diminishing curvature in their utility functions. This phenomenon can be explained by the principle of decreasing marginal utility, which asserts that the incremental value or utility derived from each additional unit of wealth diminishes at higher wealth levels.

3.1 Absolute Risk Aversion (ARA)

A more nuanced measure of risk aversion is defined through the Absolute Risk Aversion (ARA) (Pratt 1964; Arrow 1971) index, given by:

$$\rho_{ARA}(x) = -\frac{u''(x)}{u'(x)} \quad (14)$$

This metric, also known as the Arrow-Pratt measure of risk aversion, remains robust against the non-linear scaling of wealth. As wealth increases, both the second derivative $u''(x)$ and the first derivative $u'(x)$ typically decrease, but their ratio, representing the ARA, adjusts proportionally. Thus, the measure of risk aversion does not necessarily approach zero as wealth grows, providing a more reliable indicator across different wealth levels.

This measure is intrinsically linked to the concept of the risk premium.

3.1.1 Constant Absolute Risk Aversion (CARA)

If ARA is constant, we refer to it as Constant Absolute Risk Aversion (CARA). The utility function aligning with CARA can be derived by solving the differential equation $-\rho_{CARA} = -u''/u'$, which gives:

$$u(x) = -e^{-x \cdot \rho_{CARA}} \quad (15)$$

This is the exact same utility function as in the previous examples. Although the CARA utility function is negative, it should not be interpreted as the individual deriving negative utility from wealth. Instead, it serves as a ranking mechanism among different wealth levels, ensuring $u(x) > u(y)$ whenever $x > y$, and $u(x) > u(y)$ whenever $\text{var}(x) > \text{var}(y)$ and $x = y$.

This utility specification simplifies calculations and is extensively used due to its mathematical tractability. It also implies that for a CARA utility function, the risk premium can be calculated exactly as $\frac{1}{2}\pi_{ARA}\sigma^2$.

3.2 Relative Risk Aversion (RRA)

Relative Risk Aversion (RRA) is more applicable when the risky decision involves a proportion of an individual's wealth, rather than a fixed amount.

This measure is particularly relevant when returns are expressed as a percentage of wealth, aligning more closely with practical financial scenarios. RRA is mathematically defined as:

$$\pi_{RRA}(x) = -x \frac{u''(x)}{u'(x)} \quad (16)$$

The choice between using Absolute Risk Aversion (ARA) and RRA depends on the nature of the risk involved. RRA is preferred when analyzing bets that are proportional to wealth, while ARA is more suitable for fixed-level bets.

The constant RRA utility, is a Cobb-Douglas-type function:

$$u_{CRA}(x) = x^{1-\pi_{CRA}} \quad (17)$$

4 The Normal Distribution and risk

As mentioned, if the distribution of the returns is normal, the investor should always maximize the difference between mean and variance. However, if the returns are distributed differently, that is not the case.

It is well known that returns in financial markets are not normally distributed. For example, under the assumption that returns on the Oslo Stock Exchange follow a normal distribution, statistically extreme changes exceeding 10% would be exceedingly rare—estimated to occur once every 17,000 years. Yet, during the 2008 financial crisis, such anomalies were observed twice.

This is called “fat tails”. It means that extreme events are much more likely in actual financial markets, than in the normal distribution world.

However, in most cases, we can represent the empirical distribution with a “mixed normal distribution”. A mixed normal distribution, is a linear combination of normal distributions with different mean and variances.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
```

```

#Set range
MAX_MIN = 0.1
x_vals = np.linspace(-MAX_MIN, MAX_MIN, 1000)

# Create the plot using fig and ax
fig, ax = plt.subplots(figsize=(8, 5))

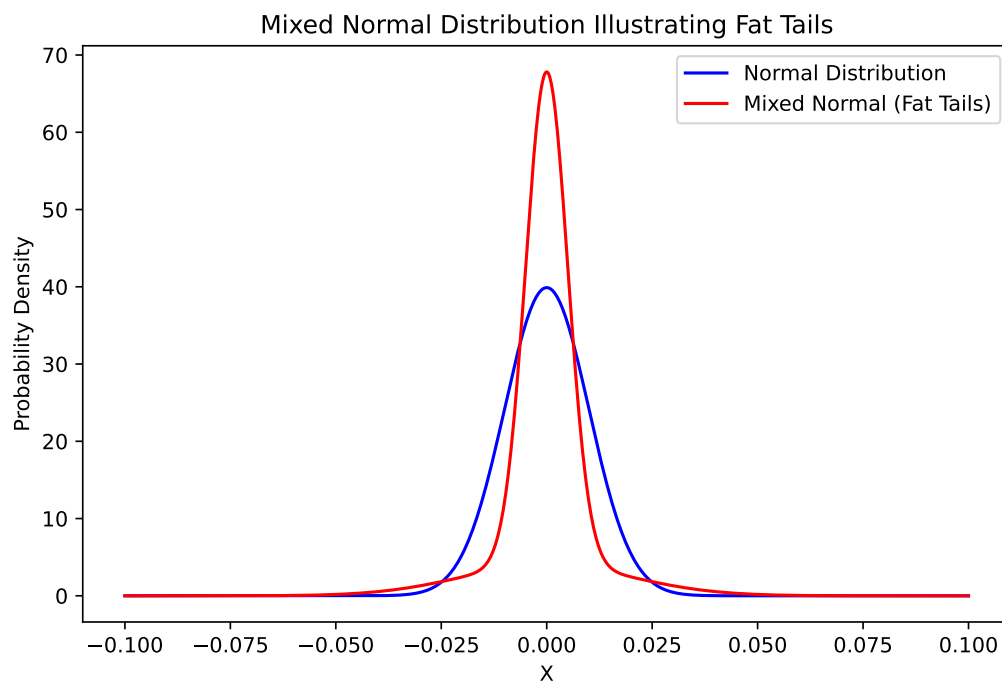
#Calculate mean and std:
rmean = 0
rstd = 0.01

# Normal distribution and mixed normal distribution:
norm_dist = norm.pdf(x_vals, loc=rmean, scale=rstd)
fat_tails = (0.8 * norm.pdf(x_vals, loc=rmean, scale=rstd*0.5)
            + 0.2 * norm.pdf(x_vals, loc=rmean, scale=rstd*2))

# Plotting
ax.plot(x_vals, norm_dist, label="Normal Distribution", color='blue')
ax.plot(x_vals, fat_tails, label="Mixed Normal (Fat Tails)", color='red')

# Set title, labels, and grid and legend
ax.set_title('Mixed Normal Distribution Illustrating Fat Tails')
ax.set_xlabel('X')
ax.set_ylabel('Probability Density')
ax.legend()

```

Coding Challenges:

- Challenge 1: Use data from <https://titlon.uit.no> to create a histogram, and overlay the normal distribution to see if the empirical distribution has fat tails.
- Challenge 2: Try to fit a mixed normal distribution to the empirical distribution.

Solution:

We start by fetching data for the OSEBX index from titlon.uit.no

```
import pandas as pd
#Query script for MySQL client
import pymysql
con = pymysql.connect(host='titlon.uit.no',
                      user      = "user@uit.no",
                      password   = "pawswordfromtitlon",
                      database='OSE')

crsr=con.cursor()
crsr.execute("""
SELECT
```

```

        `Date`, `OSEBXLinked`, `OBXLinked`, `OSEFXLinked`, `ID`
        FROM `OSE`.`equityindex_linked`
        WHERE year(`Date`) >= 1980

    """)
r=crsr.fetchall()
df=pd.DataFrame(list(r),
                 columns=[i[0] for i in crsr.description])
df
#YOU NEED TO BE CONNECTED TO YOUR INSTITUTION VIA VPN,
#OR BE AT THE INSTITUTION, FOR THIS CODE TO WORK

pd.to_pickle(df,'data/index.df')
df = None #in order to avoid memory problems

```

We can then calculate the return series and creating a histogram. The period mean and standard deviation is also calculated.

```

import pandas as pd
# loading the data
df = pd.read_pickle('data/index.df')

# Creating the plot using fig and ax
fig, ax = plt.subplots(figsize=(8, 5))

#fixing Date, setting index and calculate returns
df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
df['Returns'] = df['OSEBXLinked'].pct_change()

#Calculate mean and std:
rmean = np.mean(df['Returns'])
rstd = np.std(df['Returns'])

# Define the histogram intervals (dx = 0.01)
dx = 0.002
bins = int((0.2 - (-0.2)) / dx)

# Create the plot

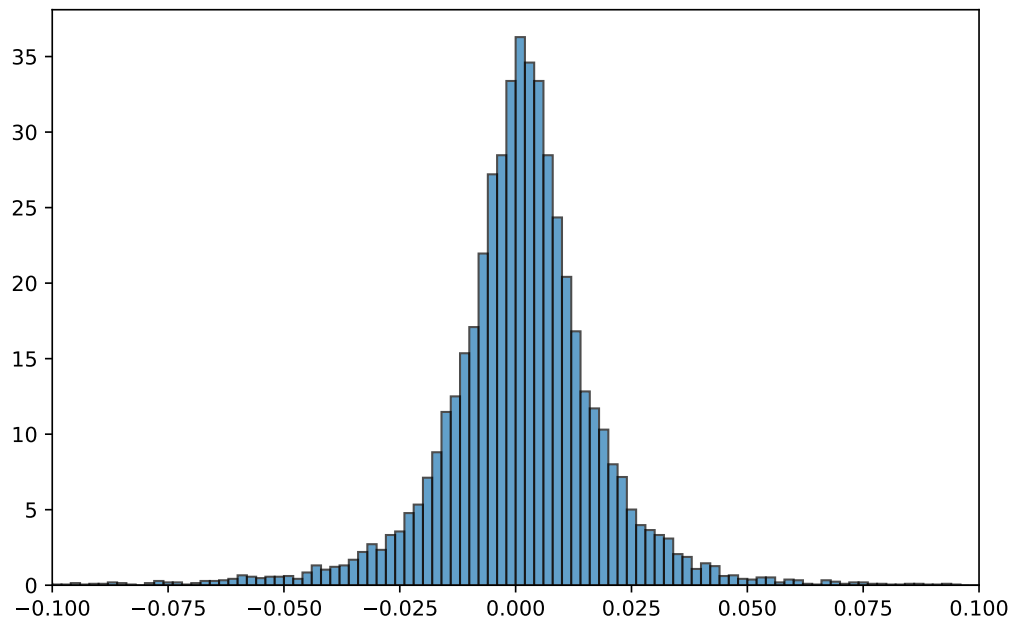
```

```

ax.hist(df['Returns'].dropna(), bins=bins,
        range=(-0.2, 0.2), density=True,
        edgecolor='k', alpha=0.7)
df = None

# Set x-axis limits
ax.set_xlim(-MAX_MIN, MAX_MIN)

```



The normal distribution and the mixed normal is now calculated, based on the estimated mean and standard deviation, and some assumptions on how the mixed normal distributions deviate from the single normal. In this example, it is just done by trial and error, which lead to a weight of 80% on a normal distribution with 40% of the estimated volatility, and 20% on a normal distribution with 200% of the estimated volatility.

This is then added to the existing chart:

```

import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

#Defining the range of x-values:

```

```

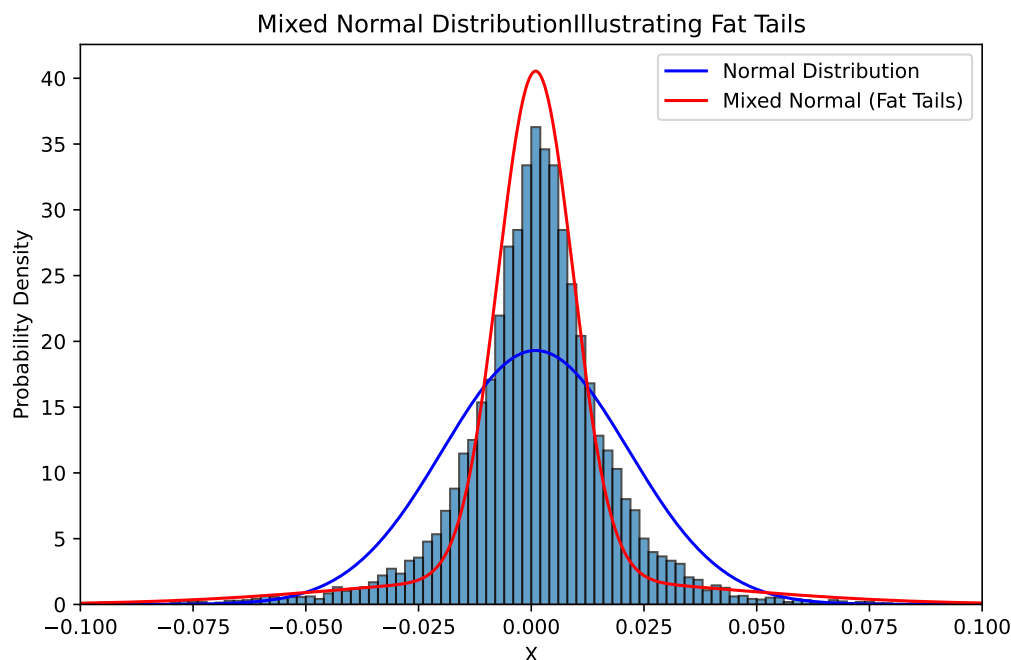
MAX_MIN = 0.1
x_vals = np.linspace(-MAX_MIN, MAX_MIN, 1000)

# Calculate normal dist and the mixed normal dist:
norm_dist = norm.pdf(x_vals, loc=rmean, scale=rstd)
fat_tails = (0.8 * norm.pdf(x_vals, loc=rmean, scale=rstd*0.4) +
             0.2 * norm.pdf(x_vals, loc=rmean, scale=rstd*2))

# Plotting normal dist and mixed normal dist
ax.plot(x_vals, norm_dist,
        label="Normal Distribution", color='blue')
ax.plot(x_vals, fat_tails,
        label="Mixed Normal (Fat Tails)", color='red')

# Set title, labels and legend
ax.set_title('Mixed Normal Distribution'
            'Illustrating Fat Tails')
ax.set_xlabel('X')
ax.set_ylabel('Probability Density')
ax.legend()
fig

```



5 Literature

- Arrow, Kenneth J. 1971. *Essays in the Theory of Risk-Bearing*. Chicago: Markham Publishing Company.
- Euler, Leonhard. 1748. *Introductio in Analysin Infinitorum*. Lausanne: Marc Michel Bousquet.
- Gauss, Carl Friedrich. 1809. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium*. Hamburg: Perthes; Besser.
- Leibniz, Gottfried Wilhelm. 1969. *Leibniz: The Early Mathematical Manuscripts*. Edited by Joseph E. Hofmann. New York: Dover Publications.
- Neumann, John von, and Oskar Morgenstern. 1947. *Theory of Games and Economic Behavior*. 2nd ed. Princeton, NJ: Princeton University Press.
- Pratt, John W. 1964. "Risk Aversion in the Small and in the Large." *Econometrica* 32 (1/2): 122–36. <https://doi.org/10.2307/1913738>.
- Tobin, James. 1958. "Liquidity Preference as Behavior Towards Risk." *The Review of Economic Studies* 25 (2): 65–86. <https://doi.org/10.2307/2296205>.