Published in Towards Data Science

Paras Varshney Follow

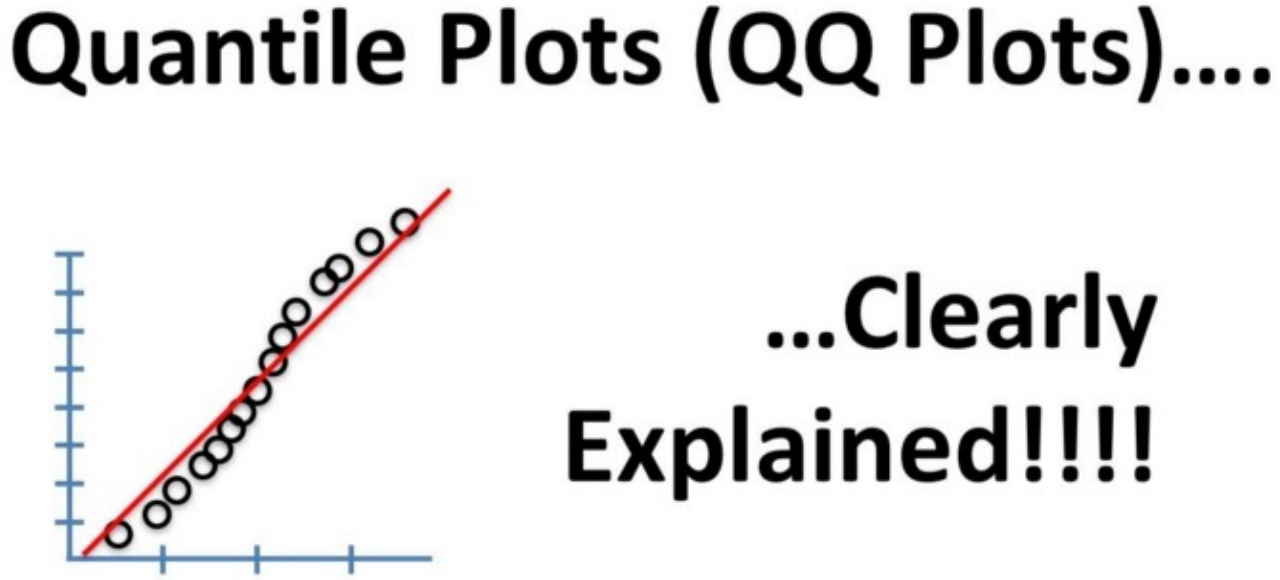
Apr 15, 2020 · 5 min read ★ · Listen

Save

STATISTICS

Q-Q Plots Explained

"Understanding the concept of Q-Q plots"

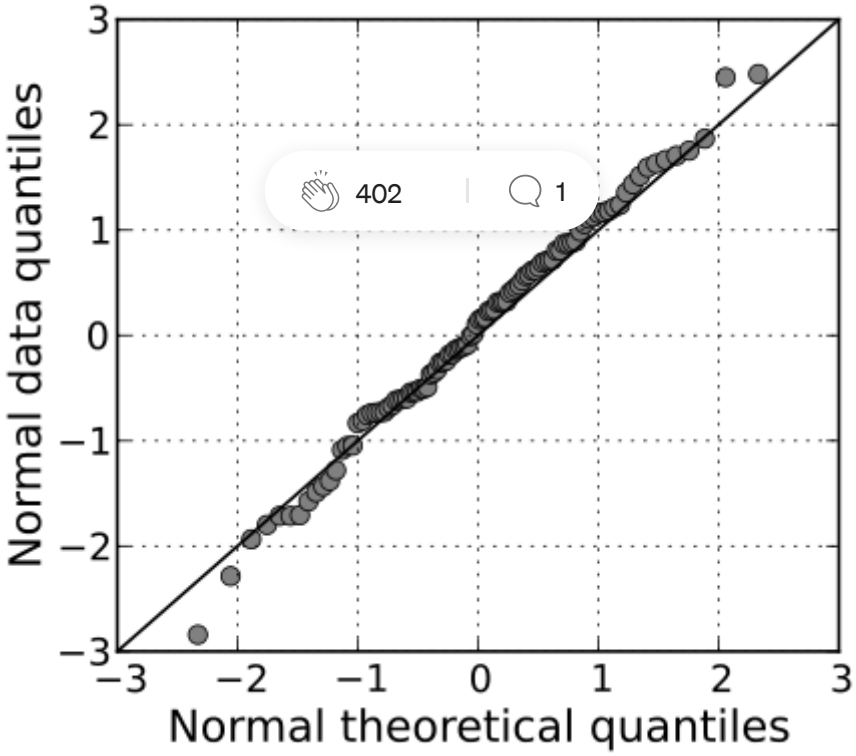


Source: Image Link

In Statistics, Q-Q(quantile-quantile) plots play a very vital role to graphically analyze and compare two probability distributions by plotting their quantiles

Yes, it's just that simple. ıtistician, it's very important for you to know whether the distribution is normal or not so as to apply various statistical measures on the data and interpret it in much more human-understandable visualization and there Q-Q plot comes into the picture. The most fundamental question answered by Q-Q plot is:

> *Is this curve Normally Distributed?*



Source: Wikipedia Q-Q plot for Normal Distribution

Normally distributed, but why?

Q-Q plots are used to find the type of distribution for a random variable whether it be a Gaussian Distribution, Uniform Distribution, Exponential Distribution or even Pareto Distribution, etc. You can tell the type of distribution using the power of the Q-Q plot just by looking at the plot. In general, we are talking about **Normal distributions** only because we have a very beautiful concept of **68–95–99.7 rule** which perfectly fits into the normal distribution So we know how much of the data lies in the range of first

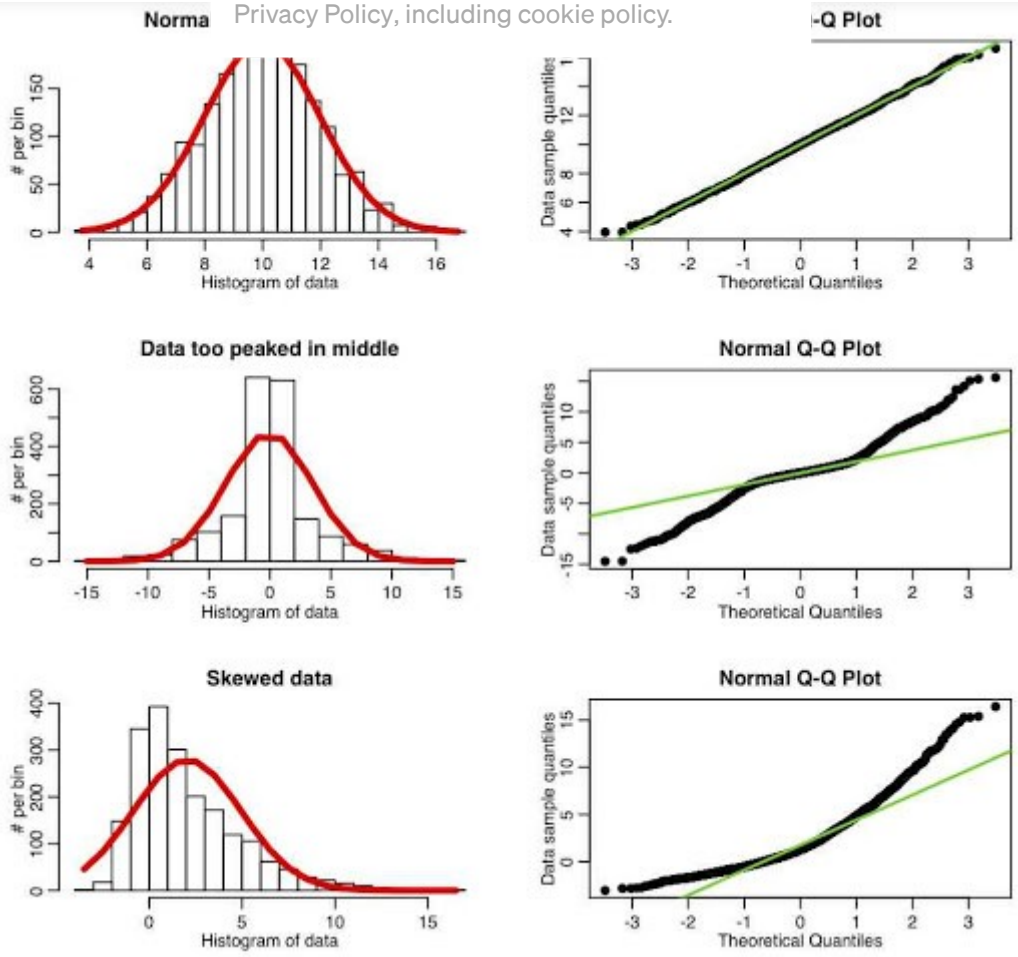
with the data easily. Sec frequently in most of the natural events which have a vast scope.

How does it work?

We plot the theoretical quantiles or basically known as the standard normal variate (a normal distribution with mean=0 and standard deviation=1)on the x-axis and the ordered values for the random variable which we want to find whether it is Gaussian distributed or not, on the y-axis. Which gives a very beautiful and a smooth straight line like structure from each point plotted on the graph.

Now we have to focus on the ends of the straight line. If the points at the ends of the curve formed from the points are not falling on a straight line but indeed are scattered significantly from the positions then we cannot conclude a relationship between the x and y axes which clearly signifies that our ordered values which we wanted to calculate are not Normally distributed.

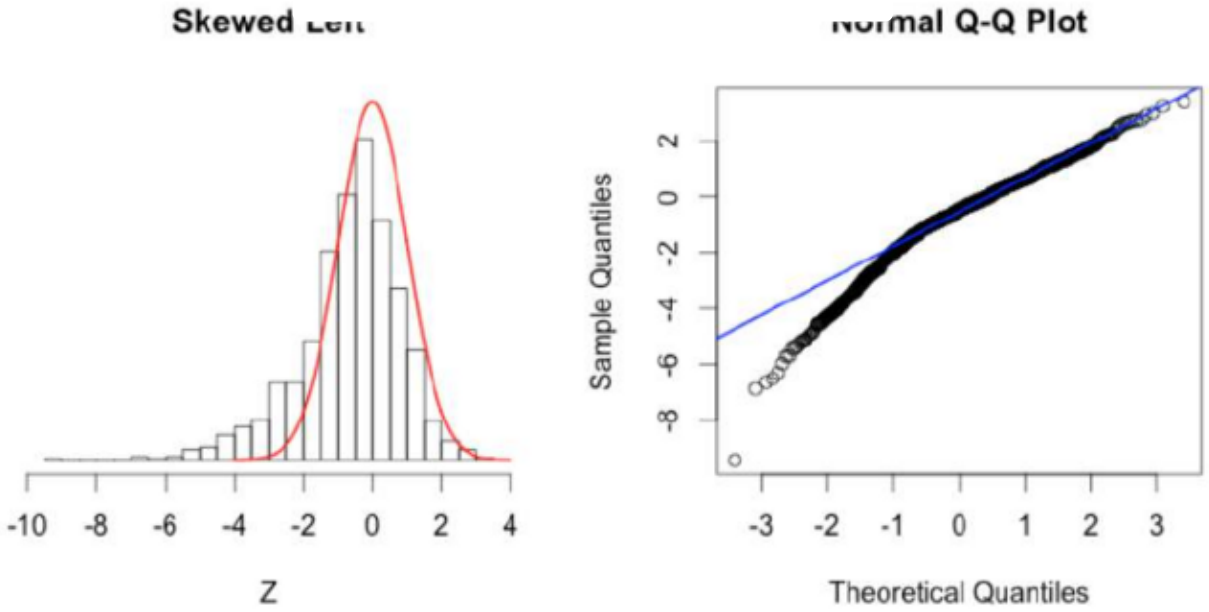
If all the points plotted on the graph perfectly lies on a straight line then we can clearly say that this distribution is Normally distribution because it is evenly aligned with the standard normal variate which is the simple concept of Q-Q plot.



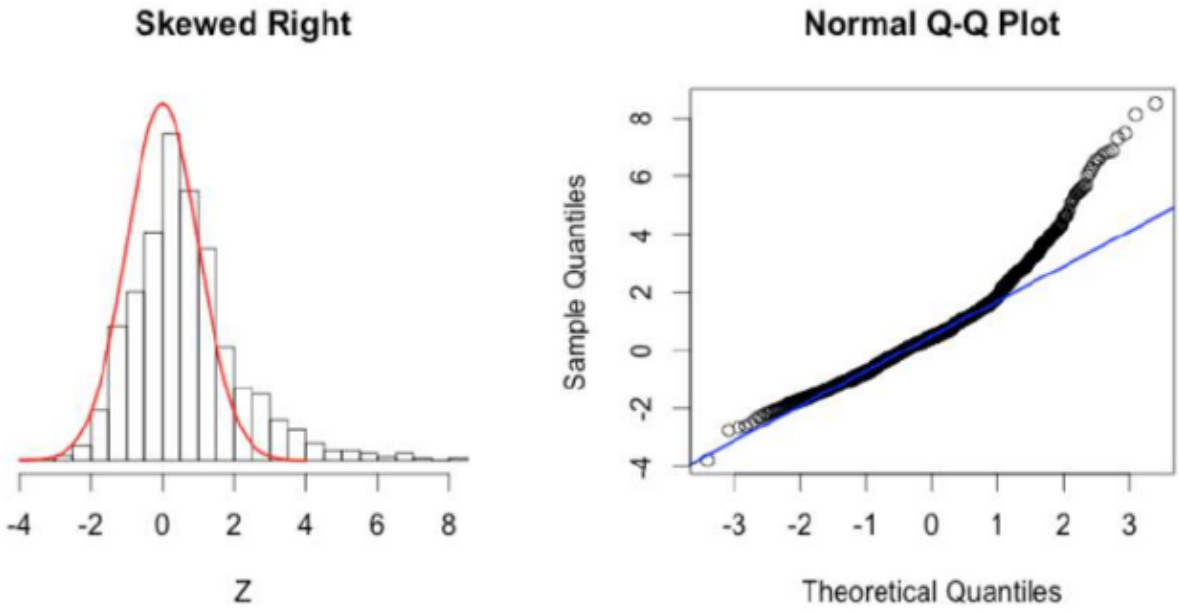
Source: Sherrytowers Q-Q plot examples

Skewed Q-Q plots

Q-Q plots are also used to find the **Skewness** (a measure of "***asymmetry***") of a distribution. When we plot theoretical quantiles on the x-axis and the sample quantiles whose distribution we want to know on the y-axis then we see a very peculiar shape of a Normally distributed Q-Q plot for skewness. If the bottom end of the Q-Q plot deviates from the straight line but the upper end is not, then we can clearly say that the distribution has a longer tail to its left or simply it is **left-skewed** (or ***negatively skewed***) but when we see the upper end of the Q-Q plot to deviate from the straight line and the lower and follows a straight line then the curve has a longer till to its right and it is **right-skewed** (or ***positively skewed***).



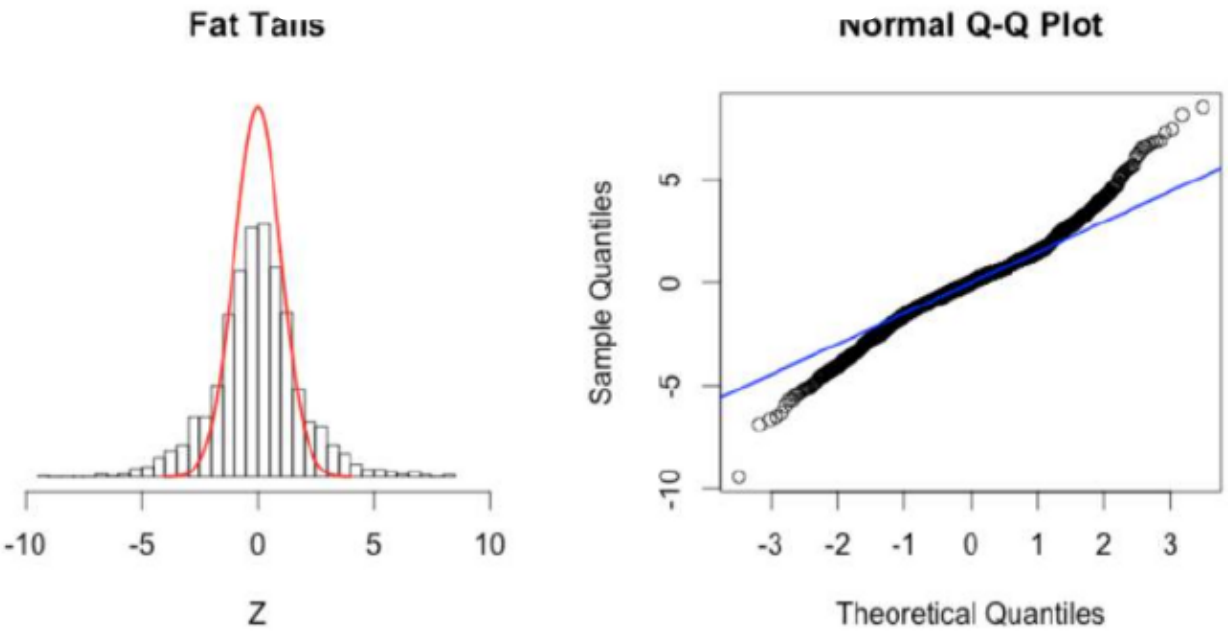
Left Skewed Q-Q plot for Normal DIstribution



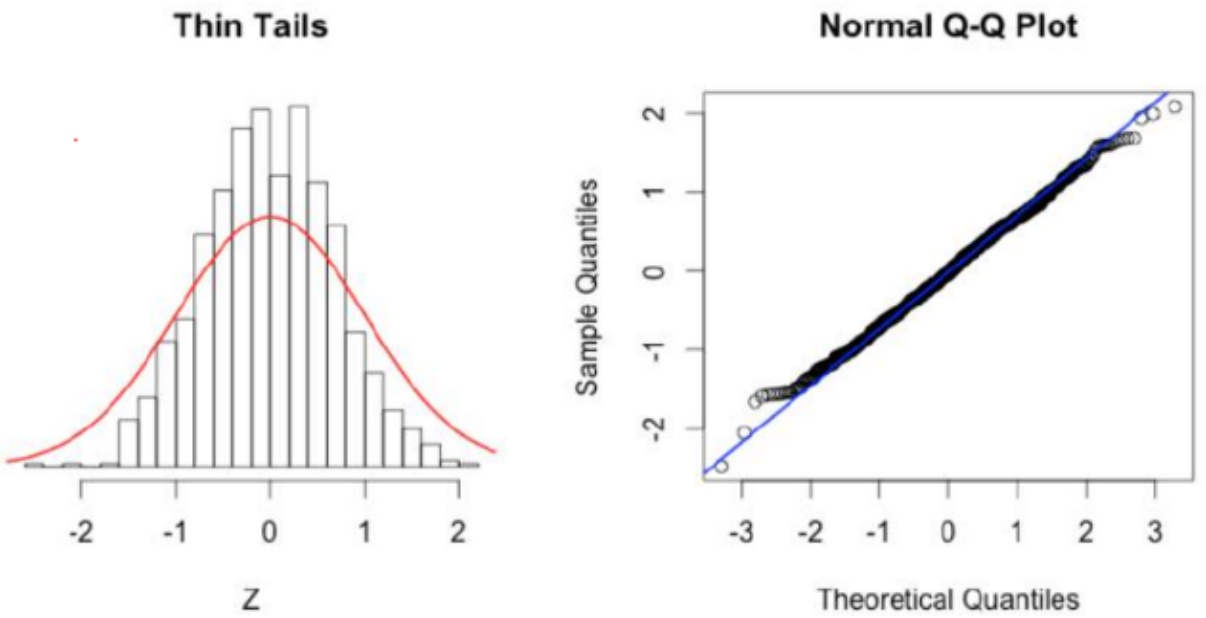
Right Skewed Q-Q plot for Normal DIstribution

Tailed Q-Q plots

Similarly, we can talk about the **Kurtosis** (a measure of "***Tailedness***") of the distribution by simply looking at its Q-Q plot. The distribution with a fat tail will have



Fat-Tailed Q-Q plot for Normal DIstribution



Thin-Tailed Q-Q plot for Normal DIstribution

How much data should do we need?

Note that when the data points are pretty less the Q-Q plot does not perform very precisely and it fails to give a conclusive answer but when we have ample amount of

Q-Q Plots Implementation in Python

Here is a simple implementation of plotting a Q-Q plot in python.

```
import numpy as np
import statsmodels.api as sm
import pylab as py

# np.random generates different random numbers everytime the code is executed.
data_points = np.random.normal(0, 1, 100)

sm.qqplot(data_points, line ='45')
py.show()
```

qqplot.py hosted with ❤ by GitHub view raw

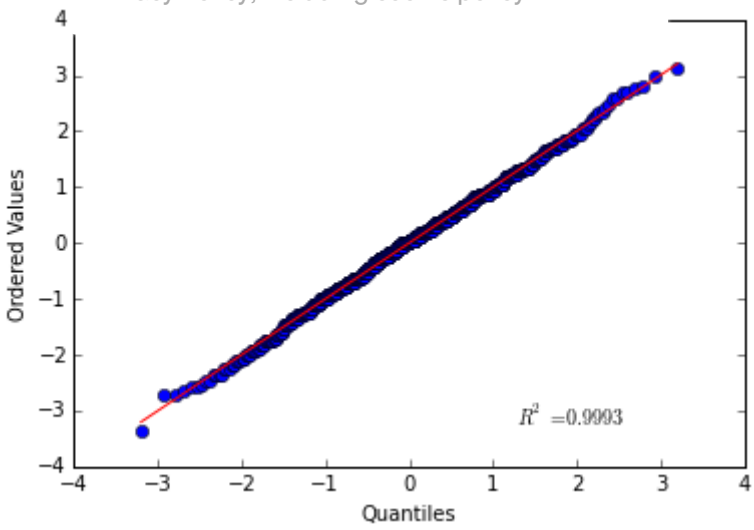
QQ plot implementation using statsmodels api

Another Implementation of the Q-Q plot using the Scipy library.

```
import scipy.stats as stats
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

n = 2000
observation = np.random.binomial(n, 0.53, size = 1000)/n

# standardize the observation
z = (observation-np.mean(observation))/np.std(observation)

stats.probplot(z, dist="norm", plot=plt)
plt.title("Normal Q-Q plot")
plt.show()
```

qqplot2.py hosted with ❤ by GitHub view raw

Q-Q plot implementation using script.stats



Source: StackExchange Output Q-Q Plot

Explore more about Q-Q Plots

I definitely recommend you go and check out the Wikipedia page of the Q-Q plot which has a very beautiful explanation about the complete concept of the mathematics working behind it which would be quite overwhelming in this introductory article. Also, check out the youtube video by Josh Starmer which demonstrates the concept in a good visualizing manner.

More articles about Data Science by Paras Varshney:

How to Evaluate Machine Learning Model Performance in Python?

A Practical Approach to Compute the Model's Performance and Implementation in Python covering all Mathematical...

medium.com

K-Nearest Neighbour Explained-Part 1

The science behind the KNN Algorithm Explained!

medium.com

and many more at Paras Varshney.

I hope you learned something new from this read!

Download the Jupyter Notebook of Q-Q plot implementation.

I write blogs about **Data Science** and **Machine Learning**. Interested to have a coffee with me, follow me on **Medium** and connect me on **LinkedIn**.

Thank you!