Table of Contents

[1. Formulation of the problem 1](#_Toc80024575)

[2. Shared derivations 2](#_Toc80024576)

[2.1 1-day returns 2](#_Toc80024577)

[2.2 n-day returns 2](#_Toc80024578)

[3. Problem A 3](#_Toc80024579)

[3.1 Base outline of the algorithm to solve the problem 3](#_Toc80024580)

[3.2 Results and discussion 4](#_Toc80024581)

[4. Problem B 5](#_Toc80024582)

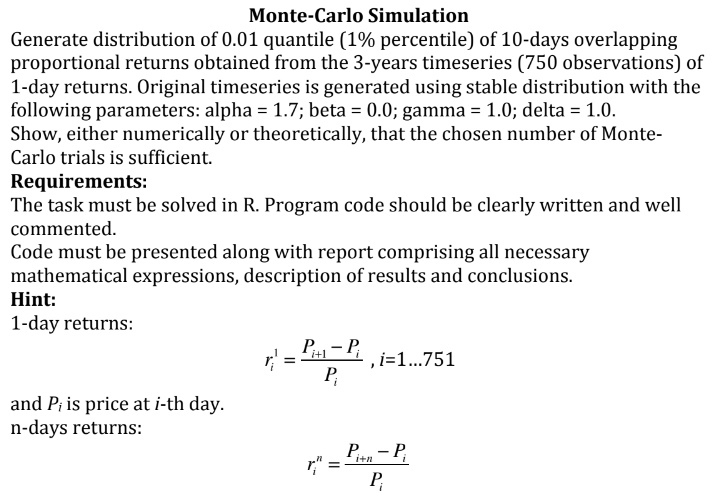
[4.1 Using Two-sample Kolmogorov–Smirnov test checks if the sample is representative 5](#_Toc80024583)

[4.2 Base outline of the algorithm to solve the problem 7](#_Toc80024584)

[4.3 Results and discussion 8](#_Toc80024585)

# 1. Formulation of the problem

This report summarises results obtained from solving the problem given on this picture:



I interpreted this problem two times. First I assumed that the problem was 1) (see below). But after I implemented 1) I revisited the original problem description and understood that the problem presumably was 2) (see below). Hence I implemented 2) too and this report contains two corresponding parts.

1. **Problem A**: To calculate 0.01-quantile of a particular random variable via Monte Carlo simulation (MCS). Realizations of a random variable are given by samples from a particular distribution. On each step of the MCS the sample from this distribution is generated (that is a sample of 10-day returns obtained from a 750-sized sample of 1-day returns). Each MC trial generates approximation to the 0.01-quantile. MCS stop criteria is given by 0.95 confidence interval of 0.01-quantile estimation.
2. **Problem B**: To generate a representative sample from a particular distribution via Monte Carlo simulation. To get a sample element from the distribution one should a) sample 750 1-day returns, b) convert them to 10-day returns, c) get 0.01-data-quantile of 10-day returns. This data-quantile is the sample element (aka the realization of a random variable). MCS stop criteria is until the accumulated sample is representative. Data-quantile is a formal mapping in this case.

See mode details in the corressponding sections.

# 2. Shared derivations

## 2.1 1-day returns

1-day returns are given by random variable. . Where . are realizations of a random variable . .

See [Stable distribution on Wikipedia](https://en.wikipedia.org/wiki/Stable_distribution). Used implementation of [Stable distribution in scipy](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.levy_stable.html).

## 2.2 n-day returns

n-day returns are given by formula:

That is derived from the following:

1-day returns:

where is a price at -th day.  
n-day returns:

Derivation:

# 3. Problem A

To calculate 0.01-quantile of a particular random variable via Monte Carlo simulation (MCS). Realizations of a random variable are given by sintetic samples () from a particular distribution. On each step of the MCS the sample from this distribution is generated (that is a sample of 10-day returns obtained from a sample of 1-day returns ). Each MC trial generates approximation to the 0.01-quantile ( are realizations of a random variable ). MCS stop criteria is given by 0.95 confidence interval of 0.01-quantile estimation. The described method is valid assiming .

## 3.1 Base outline of the algorithm to solve the problem

is a list to store 0.01-quntile approximations of the from MC trials. Stores realizations of a random variable .  
 is a -quantile for confidence interval estimation. .  
where is a [quantile function](https://en.wikipedia.org/wiki/Quantile_function) of the standard normal distribution computed via [scipy](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html) (ppf method).

In a loop do:

1) on -th iteration of the loop,  
2) sample 1-day returns:

3) convert to 10-day returns:

4) compute 0.01-quantile of the sample – via [numpy](https://numpy.org/doc/stable/reference/generated/numpy.quantile.html) and append it to the list,  
5) compute expected value estimate and variance estimate of the using equations from this [stackexchange answer](https://quant.stackexchange.com/a/21769).  
6) if and condition is satisfied then exit loop and Monte Carlo simulation should stop. The condition:

where – relative accuracy.

Let’s assume that , then:

and are computed via online implementation:

## 3.2 Results and discussion

See results of the calculation of the on the Fig. 1.

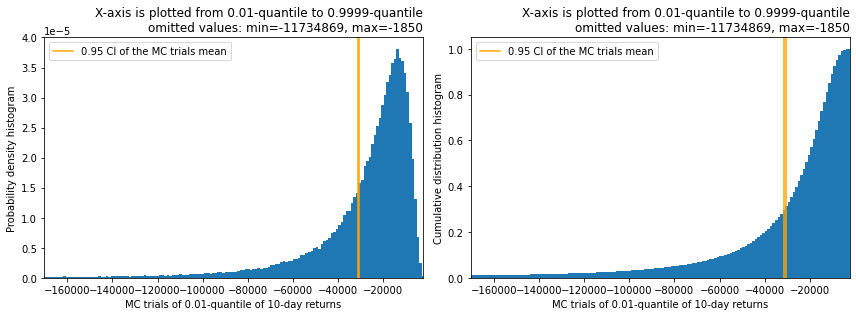


Fig. 1. Probability density histogram (left) and Cumulative dustribution histogram (right) obtained from plotted from 0.01-quantile to 0.9999-quantile. Orange line shows confidence interval for estimated .

Assuming the algorithm above should give reasonable estimate of the desired . The assumption seems intuitive but I guess it needs proof…

# 4. Problem B

To generate a representative sample from a particular distribution via Monte Carlo simulation. To get a sample element from the distribution one should a) sample 1-day returns , b) convert them to 10-day returns , c) get 0.01-data-quantile of 10-day returns. This data-quantile is the sample element (aka the realization of a random variable ). MCS stop criteria is until the accumulated sample is representative. Data-quantile is a formal mapping in this case.

## 4.1 Using Two-sample Kolmogorov–Smirnov test checks if the sample is representative

*This section mostly describes why Two-sample Kolmogorov–Smirnov was useless for hypothesis testing. For the implemented algorithm see the the very end of this section.*

[Kolmogorov–Smirnov test](https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test) is the only test I found in the context of checks if the sample is representative. See this [stackexchange answer](https://stats.stackexchange.com/a/97552). Unfortunately it was applied to a different situation.

First idea was that if we times randomly partition into two subsamples of approximately equal sizes and each time two subsamples pass Kolmogorov–Smirnov test then we decide that is representative. I assumed that this way when the sample size is not big enough, among random partitions would always be a partition with halves that too different (failing the Kolmogorov–Smirnov test).

I desided to use [scipy implementation](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ks_2samp.html) of Two-sample Kolmogorov–Smirnov test that is mainly an implementation of the [doi:10.1007/bf02589501](https://doi.org/10.1007/BF02589501). From the paper:

where samples are from distribution and samples are from distribution. and correspond to one-sided and two-sided cases respectively. Scipy implementation returns tuple of and pvalue that is .

where - cumulative dustribution function for the two-sided Kolmogorov-Smirnov statistic.

And it turned out that applying the first idea is not this straightforward. If the two samples are independent and from the same distribution then the limit of random value distribution (when both tend to infinity) would be Kolmogorov distribution, where is Kolmogorov–Smirnov statistic:

where – cumulative distribution function of the Kolmogorov distribution.

The single-evaluation of test can only be helpful in proving that two samples are from different distributions and only when choosing low enough p-value like 0.0001. Hence I tried to make a test from 100-1000 evaluations of (using random partitions of ) and compare it to some quantile of Kolmogorov distribution. When we do partition checks the is satisfied automatically and if we select significance level as then approximately of partition checks would be below -quantile of Kolmogorov distribution.

It’s a far from ideal measure of closeness to Kolmogorov distribution but even this didn’t work as scipy implementation returns pvalue in respect to instead of . And even more. It turned out that even for small sized samples (like 50 vs. 50 partitions) -quantile matches portion of rejected pvalues ( statistic is big in this case). Hence in practice only value threshold matters as pvalues returned from scipy never help to check if the sample is representative.

Though the criteria using pvalues helps to tell apart different distributions. When running the following code if the second argument is 1. then destributions match:

def check(n: int, sigma2: float):  
 from scipy.stats import ks\_2samp  
 import numpy as np\_  
  
 d\_pval = []  
 for \_ in range(100):  
 d, pvalue = ks\_2samp(np\_.random.normal(0, 1., n), np\_.random.normal(0, sigma2, n))  
 d\_pval.append([d, pvalue])  
 pvals = np\_.array(d\_pval)[:, 1]  
 n\_reject = len(pvals[pvals <= 0.1])  
 print(n\_reject)  
  
  
check(2000, 1.05)  
check(2000, 1.)  
check(20000, 1.05)  
check(20000, 1.)

The typical output was:

27  
15  
97  
8

Running this code never gave <10 for check(2000, 1.05).  
check(2000, 1.) gave ~10.  
Typical for check(20000, 1.05) is >90.  
check(20000, 1.) gave ~10.

Hence:

**The main criteria was setting threshold for Two-sample Kolmogorov–Smirnov test’s statistic (when doing 100-1000 partition tests). Setting it you can select desired smoothness of the generated distribution. That’s not a representativness criteria but intuitively if empirical distribution functions of random partitions never diverge above the threshold then doesn’t have “holes”.**

## 4.2 Base outline of the algorithm to solve the problem

is a list to store 0.01-data-quntiles from MC trials. Stores realizations of a random variable .

In a loop do:

1) on -th iteration of the loop,  
2) sample 1-day returns:

3) convert to 10-day returns:

4) compute 0.01-data-quantile of the sample – via [numpy](https://numpy.org/doc/stable/reference/generated/numpy.quantile.html) and append it to the list,  
5) times randomly partition sample into two subsamples and compute Two-sample Kolmogorov–Smirnov statistic .  
6) if some condition is satisfied then exit loop and Monte Carlo simulation should stop. The condition:

where – the threshold for statictic.

## 4.3 Results and discussion

See results of the generating of representative sample on the Fig. 2.

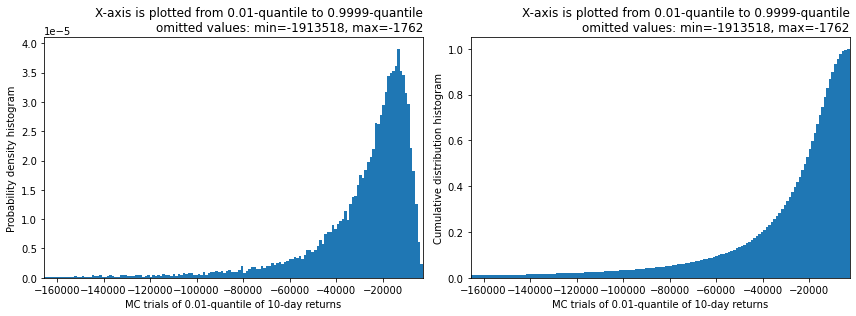


Fig. 2. Probability density histogram (left) and Cumulative dustribution histogram (right) obtained from plotted from 0.01-quantile to 0.9999-quantile.

The main discussion of the method used is given in the section 4.1.