Reducing cost of volatile risk estimates on financial instruments in a cloud environment

Ludvig Boström
Performing high accuracy risk calculations on financial instruments is a computationally heavy task. Since a day can contain a lot of different events that change the stock prices, it is important that these risk calculations are done in real time. Deploying a system to calculate real time risk in a cloud environment would result in high costs because of the heavy calculations that needs to be performed. This thesis is a practical continuation of a previous master thesis written by Sara Ekman [1]. Ekman proposed different strategies to reduce the amount of recalculations while still having real time precision on the risk values. In this thesis the strategies covering European options and Futures are implemented and tested over several days, with a goal of reducing the cost of a daily run by 5%, compared to using constant intervals. Some strategies upheld the goal in simulations but did however loose accuracy. The strategies that performed the best, under tested conditions, was the strategies that did not consider any scenarios when evaluating if recalculations had to be made.
Contents

1 Introduction
   1.1 Background
   1.2 Goal
   1.3 Limitations
      1.3.1 Asian options

2 Theory
   2.1 Instruments
      2.1.1 Futures
      2.1.2 European options
   2.2 Calculations
      2.2.1 Volatility
      2.2.2 Risk measures
   2.3 Cloud
      2.3.1 Delivery models
      2.3.2 Elastic
      2.3.3 Scaling
      2.3.4 Measurements
      2.3.5 Pricing models

3 Related work
   3.1 Instruments
      3.1.1 Fast pricing instruments
   3.2 Methods
      3.2.1 Collecting data
      3.2.2 Set up instruments
      3.2.3 Solution generation
      3.2.4 Optimization strategies
1 Introduction

This section will give an introduction to the problem, why it is to be explored, and the specific aspects of this issue. It will also go into the limitations and why these limits have been set.

1.1 Background

Clearing houses are financial institutions that act as a third party for financial transactions by being the opposing side to each trade. When both a seller and a buyer are clearing members and agree on quantity and price, the clearing house clears that trade. The members do not need to communicate with each other, they both communicate with the clearing house that clear the transactions between the members [2]. Clearing houses are interested with risk calculations on both sides of a trade since they act as a counterpart for both sides. The risk calculations are used to take out collateral [3] from their clients depending on the risks of the positions. This risk is used to measured how much they can lose at a certain probability level. To do this future prices are estimated and the outcomes evaluated. If the price estimations don’t match the real price, the risk estimates may not be able to maintain high accuracy. To perform risk calculations it is needed to look at different price vectors containing multiple scenarios of the price of the instrument at the same point in time [4]. The large set of scenarios make this a very time consuming and computationally heavy task.

A clearing system usually runs in large server rooms with an enormous amount of processing power, to keep all systems in real time and achieve the fastest and most accurate calculations. The processing power is also needed to clear the transactions as fast as possible. The system perform a lot of tasks with one of them being risk calculations on portfolios. A portfolio is a collection of positions on different financial instruments, and the risk calculations will calculate the risk over all of the positions combined. If a clearing system instead would be offered as a service in the cloud it would be a very expensive service. Therefore it is important to find cheaper ways of executing the different parts of the system. Calculating the risk of a portfolio is a heavy operation with regard to processing power and will therefore be expensive to perform for each price change to an asset in a portfolio. Different methods that control the frequency of performing these calculations, while keeping a high accuracy, may be used to reduce the cost of these operations.

Executing services in the cloud is usually made through microservices. This means that different parts of a system will be built into different modules that are scalable by starting duplicate instances of a service. These microservices are lightweight and perform very few tasks each. They are also independent of other microservices [5]. Different cloud services are priced using different pricing models. Some of these models prioritize hardware usage,
some prioritize time usage and some models uses a mix of them.

1.2 Goal

The goal of this thesis is to implement strategies for controlling the frequency of heavy calculations to reduce the cost of performing them in a cloud environment. The primary goal of this thesis is to find a strategy that, within set error margins, can reduce the costs by 5% compared to recalculations at a constant rate.

1.3 Limitations

This thesis will only cover the implementation of strategies concerning Futures and European call options. This decision was based on the limitation of resources, primarily the time available to perform simulations. The work is also limited to a small set of scenarios. The amount of instruments per stock is limited to 12 options and 4 futures.

1.3.1 Asian options

The strategies concerning Asian options, mentioned by Ekman [1], are supposed to reduce the amount of recalculations needed by approximating the price of the Asian option using different methods. The amount of recalculations needed is set to the number of times the approximated price is too far away from the solution price. When performing the calculations to see how many recalculations are needed, the scenario prices of the current time is used. This means that the current scenario prices are needed to calculate themselves. This would not be possible to implement in a way that is usable for a real world scenario. As also mentioned by Ekman [1], solving this requires another level of optimization problem that needs to be solved.

Because of the sheer difficulty of the optimization and focus shifting more towards being mathematical and finding or adjusting the strategies, than computational, it was decided together with Cinnober to omit this instrument type in this thesis.

The Asian options would however have been really interesting to look at since they are a bit more time consuming to price and therefore give a bigger impact on the total durations for the different methods.
2 Theory

This section covers the theory behind the practical work that is done. The focus of this section is on finance and risk calculations, but also covers some information about cloud computing.

2.1 Instruments

This section will cover the two different financial instruments that is discussed in the report. Futures, European options.

2.1.1 Futures

A Futures contract is an agreement between two parties to buy or sell an asset at a certain time $T$ in the future at a predetermined price. The pricing of the futures contract, regarding an underlying asset without dividends, is done by the no arbitrage principle which yields the expression

$$F_0 = S_0e^{r(T-t)} \quad (2.1)$$

where $S_0$ is the price of the underlying asset when entering the contract, $t$ is the time of entering the contract, $r$ is the risk free interest rate and $T$ is the time of maturity [6, Chapter 5].

2.1.2 European options

A European option similarly has a time to maturity $T$ and a preset strike price $K$. The difference is however that with a European option the holder of the contract has the right, but not the obligation, to exercise the option at the time of maturity. If it is a call option, the holder can choose to buy the underlying asset, at price $K$, or not. With a put option, the holder can choose to sell the underlying asset, at price $K$, or not. The pricing of an option can be done in multiple ways, but in this case the most used method will be use. That is the Black–Scholes–Merton pricing formulas. The call option is priced through

$$c = S_0N(d_1) - Ke^{-r(T-t)}N(d_2) \quad (2.2)$$

and the put option

$$p = Ke^{-r(T-t)}N(-d_2) - S_0N(-d_1) \quad (2.3)$$

where

$$d_1 = \frac{\ln\left(\frac{S_0}{K}\right) + (r + \sigma_\tau^2)(T-t)}{\sigma \sqrt{T-t}} \quad (2.4)$$

and

$$d_2 = \frac{\ln\left(\frac{S_0}{K}\right) + (r - \sigma_\tau^2)(T-t)}{\sigma \sqrt{T-t}} = d_1 - \sigma \sqrt{T}. \quad (2.5)$$
Here $N$ is the cumulative probability function for the standard normal distribution, $\sigma$ is the volatility of the underlying asset, $S_0$ is the price of the underlying asset, $t$ is the current point in time, and $r$ is the risk free interest rate [6, Chapter 14].

2.2 Calculations

This section will cover more of the calculations described in the Master thesis by Sara Ekman [1] and confirmed with Options, Futures, and Other Derivatives by John C. Hull [6].

2.2.1 Volatility

Volatility can be estimated through different methods. In this thesis it is estimated by using the exponentially weighted moving average model (EWMA).

EWMA can be used to estimate the volatility through the standard deviation of logarithmic returns. By using a decay factor $0 < \lambda < 1$, the most recent returns are weighted higher than the older ones. This way the most recent returns will have a greater impact on the volatility. The estimated volatility can be described with

$$\sigma_n^2 = \lambda \sigma_{n-1}^2 + (1 - \lambda) \ln\left(\frac{S_n}{S_{n-1}}\right)^2.$$

(2.6)

The factor $\lambda$ is usually set to 0.94 for daily returns [6, Chapter 22].

2.2.2 Risk measures

This section cover two different risk measures, Value at Risk and Expected shortfall. There exists other risk measures for financial instruments but Value at Risk and Expected shortfall where the selected ones in the Master thesis by Ekman [1] and will continue to be so here. This section is also based on the information in that thesis.

Value at Risk

Value at Risk (VaR) measures the least amount of money to loose, at a specific probability level and time period. Usually it is denoted $\text{VaR}_{\alpha 100\%}(L)$ where $\alpha$ describes the probability level and $L$ the sorted possible losses. It is measured as a quantile of the profit and loss distribution. To calculate $\text{VaR}$ the first thing is to set $\alpha$. This is usually set to 0.95 or 0.99. Then the time period has to be specified. Finally the Profit and Loss distribution is decided. Usually set by checking historical observations to estimate the probability distribution. To get scenario prices at time $t$ the historical data can be used to create $N$ one-day real-world scenarios which in turn are applied to the instrument price which then gives $N$ scenario prices at a specific point in time $t$. A Profit and Loss (PnL) vector can then be created by subtracting the scenario price $P_i(t)$ from the instrument price $P(t)$ for each scenario.

$$PnL_i(t) = P(t) - P_i(t).$$

(2.7)

If then sorted in decreasing order the VaR value is the $\alpha N$ element in the PnL-vector. This implies that $\text{VaR}_{\alpha 100\%}(L) = PnL(\alpha N)$. 
Expected shortfall

Expected shortfall is somewhat an extension to Value at Risk. VaR gives the maximum loss given a probability. Expected shortfall on the other hand gives the amount that is expected to be lost given that the VaR value has been breached.

2.3 Cloud

This section will cover the basics of cloud computing to get a better understanding of how things are used to scale, measure performance and how much they will cost.

2.3.1 Delivery models

The descriptions of the delivery models in this section are based on the descriptions made in Cloud Services for Dummies, IBM Limited edition [5].

IaaS — Infrastructure as a Service

When hardware, software, storage, networking, operating systems and various other utility software components are delivered on request. Usually offered through virtualization that gives the option of creating multiple virtual systems on a single physical system. The key characteristics of IaaS that is covered in this report are dynamic scaling and metering [5, Chapter 2].

PaaS — Platform as a Service

Used for combining IaaS with other middleware services, development and deployment tools to give a consistent method to create and deploy applications into the cloud. This way it’s not required that the developers know all the lower lever details about the environment. This is mainly about the process of creating and deploying systems [5, Chapter 2].

SaaS — Software as a Service

Created on top of IaaS and PaaS. Can be implemented directly on a IaaS platform and usually don’t expose the underlying layers to the end-user [5, Chapter 1].

2.3.2 Elastic

In cloud computing elasticity means that resources can be used when needed and therefore only paid for at a per unit basis. When not needed the resource is released and offered to be used by someone else. This way it keeps the cost of idle resources down. [5, Chapter 1]
2.3.3 Scaling

As one of the main characteristics of IaaS stated dynamic scaling is an important part of cloud computing. Being able to automatically and instantly expand resource usage when needed from a provider to manage the workloads is to dynamically scale. In the case of the problem this report regards this will be used in order to perform the risk calculations as quickly and cost efficiently as possible [5, Chapter 2].

2.3.4 Measurements

Another of the key components of IaaS was metering. Being able to measure the resource usage is a must to be able to charge for the exact usage. In the most cases some of the measurements include storage, data transfer and CPU power [5, Chapter 2].

2.3.5 Pricing models

There are a wide range of different pricing models that different cloud service providers offer. This thesis uses the pricing model offered by Amazon Lambda [9]. With this model the price increases based on time usage rounded up to 100ms. The time considered in this model is the duration between calling a function within the cloud and returning from that function. The model also has a higher cost depending on the memory usage of the program.
3 Related work

This project relies heavily on the previous master thesis performed by Ekman [1]. In that work she explored different methods of controlling the frequency of when to recalculate the risk estimates while still having high accuracy. This section will summarize the important parts and finds from the thesis, without going into too much detail, and cover what will be needed to know to be able to get a better understanding of the following information in the report.

3.1 Instruments

The instruments used in the thesis by Ekman [1] and that also will be covered here are mentioned as fast pricing instruments.

3.1.1 Fast pricing instruments

These kinds of instruments are those that can be priced using an analytical expression. Examples can be future contracts and European stock options.

3.2 Methods

The methods used to evaluate when to perform these calculations were for the fast pricing instruments called optimization strategies. The optimization strategies take the set up instruments and generated minute prices for a day and calculate a profit and loss vector. They are later compared to a know it all optimal strategy.

3.2.1 Collecting data

The thesis used 10 stocks from the NASDAQ exchange. They were all chosen from the same exchange since they have the same opening hours meaning that they all use the same time intervals.

The collected data was then used to create a scenario file that would consist of the daily shifts in stock price, volatility and interest rate. Stock price data for the different assets were collected from YAHOO! finance [7] and gave the daily changes in the stock price. Different scenarios were then set up using the fractions between consecutive days. That gave daily shifts in stock price scenarios. The volatilities for the same time interval were evaluated using the GARCH(1,1)-model [6, Chapter 22]. The daily shifts of the volatility were then calculated in the same way as the daily shifts in price. The daily interest rates were collected using the daily treasury yield curves [8]. The daily shifts of the interest rates were
also calculated in the same way as the price shifts.

3.2.2 Set up instruments

The instruments are then set up by creating European options around four different times to maturities and three different strike prices. The futures also gets four different times to maturities. This will give 12 options and four different futures for each underlying stock.

The volatility was estimated using EWMA as described in Section 2.2.1. The parameters used was the estimation window $W_E = 100$ days and $\lambda = 0.94$.

3.2.3 Solution generation

To evaluate each method, solutions for the entire days was created for each instrument. This means that all scenarios for every instrument in every minute was calculated and saved. Each day the exchange is open for 391 minutes.

3.2.4 Optimization strategies

When working with the fast pricing instruments it is first needed to generate the different solutions to be able to find the error of the strategies. This is done by using minute price data together with forecasted volatility and interest rate from the previous day, and together with the scenario file as well. All different outcomes will be calculated for each stock. From the different outcomes PnL-vectors (Profit and Loss vectors) were constructed. These solution generated results was used to compare how accurate the different methods are.

The optimization now starts by setting target limits on the relative errors to $\varepsilon_{\text{target}} = 0.4\%$ for futures and $\varepsilon_{\text{target}} = 8\%$ for European stock options.

Each strategy had to hold for the expression

$$
\frac{1}{N} \sum_{i=0}^{N} \frac{|P_i(S_t) - P_i(S_0)|}{P_i(S_t)} \leq \varepsilon_{\text{target}} \tag{3.1}
$$

where $P_i(S_0)$ is the instrument scenario price for the previously recalculated price for scenario $i$ and $P_i(S_t)$ is the instrument scenario price at the current point in time $t$ for scenario $i$, and $N$ is the total number of scenarios. This means that the maximum mean relative error that is accepted for an instrument is set to 8% and 0.4% respectively.

Constant recalculation

This strategy has two sub-strategies. One uses a constant time step of 60 minutes. If the maximum mean error constraint is not reached it will perform the calculations again in half of the time step. The second one divides the day into three parts since usually stocks are most volatile at the beginning and the end of a day. This gives the first part a time step of 50 minutes, the second part two hours and the third part 60 minutes. This strategy also halves the time step if the constraint is not reached.
Relative change in stock price — Strategy 1

This strategy uses the relative change in stock price, between the current time $t$ and the stock price used in previous recalculation, to determine when the recalculations have to be performed. By using a subset of scenarios and error terms from Taylor expansions used during the calculations an expression was built to describe the relative changes in the price. The final expression for the strategy is

$$\frac{|S_t - S_0|}{S_0} \leq \frac{k\hat{\epsilon}_{\Delta S, k} \prod_{i \in n} P_i(S_0)}{S_0 \left[ \sum_{i \in n} |\Delta_i| \prod_{j \in n \setminus \{i\}} P_j(S_0) \right]} = \hat{\epsilon}_{\Delta S, k}, \quad (3.2)$$

where

$$\hat{\epsilon}_{\Delta S, k} = \epsilon_{\text{target}} - \omega_t - \omega_k - \omega_{\Delta S_1} \quad (3.3)$$

describes the modified target value, $\Delta_i$ is the corresponding change in instrument prices related to the change in the price of the underlying stock for scenario $i$, $k$ is the number of scenarios in the subset, and $n$ is the subset of scenarios. The instrument price for scenario $i$ at the previous recalculation is denoted by $P_i(S_0)$ and at for scenario $j$ it is $P_j(S_0)$. The stock price at the previous recalculation an the current point in time $t$ is denoted by $S_0$ and $S_t$ respectively.

When not considering any scenarios the strategy was described by the expression

$$\frac{|S_t - S_0|}{S_0} \leq \frac{k\hat{\epsilon}_{\Delta S} P(S_0)}{S_0|\Delta|} = \hat{\epsilon}_{\Delta S}, \quad (3.4)$$

where

$$\hat{\epsilon}_{\Delta S} = \epsilon_{\text{target}} - \omega_k - \omega_0 - \omega_{\Delta S_1} \quad (3.5)$$

described the modified target value and $\omega_{\Delta S_1}$ was an error term originating from a Taylor approximation.

Relative error of stock price — Strategy 2

This strategy concerns the stock price as well but will focus on the relative error instead of the relative change. The final equations are derived in a similar matter and gives and expression regarding the how the price changes effect the errors. The strategy is described using this expression

$$\frac{|S_t - S_0|}{S_t} \leq \frac{k\hat{\epsilon}_{E(\Delta S, k)} \prod_{i \in n} \hat{P}_i(S_t)}{S_t \left[ \sum_{i \in n} |\Delta_i| \prod_{j \in n \setminus \{i\}} \hat{P}_j(S_t) \right]} = \hat{\epsilon}_{E(\Delta S, k)}, \quad (3.6)$$

where

$$\hat{P}_i(S_t) = |\Delta_i||S_t - S_0| + P_i(S_0) \quad (3.7)$$

and

$$\hat{\epsilon}_{E(\Delta S, k)} = \epsilon_{\text{target}} - \omega_k - \omega_{\Delta S_2} \quad (3.8)$$
was described as the modified target value.

When not considering any scenarios the strategy was described as

\[
\frac{|S_t - S_0|}{S_t} \leq \frac{\hat{\varepsilon}_{E(\Delta_S)} \hat{P}(S_t)}{S_t |\Delta|} = \tilde{\varepsilon}_{E(\Delta_S)},
\]

(3.9)

where

\[
\hat{\varepsilon}_{E(\Delta_S)} = \varepsilon_{\text{target}} - \omega_0 - \omega \Delta^2
\]

(3.10)

was described as the modified target value regarding the strategy. In the no scenario case \( \hat{P}(S_t) = |\Delta| |S_t - S_0| + P(S_0) \) and \( \omega \Delta \) was an error term that originated from a Taylor approximation.

Relative error of instrument price — Strategy 3

This strategy instead looks directly on the relative error of the instrument price. By picking out a subset of scenarios \( k \) where recalculation will be done it yielded the following expression for strategy

\[
\frac{1}{k} \sum_{i \in n} \left| \frac{P_i(S_t) - P_i(S_0)}{P_i(S_t)} \right| \leq \hat{\varepsilon}_{E(\Delta p)},
\]

(3.11)

where

\[
\hat{\varepsilon}_{E(\Delta p)} = \varepsilon_{\text{target}} - \omega_k
\]

(3.12)

was described as the modified target value regarding this strategy.

When no scenarios are considered the strategy \( E(\Delta p) \) is described by

\[
\frac{|P(S_t) - P(S_0)|}{P(S_t)} \leq \hat{\varepsilon}_{E(\Delta p)},
\]

(3.13)

where

\[
\hat{\varepsilon}_{E(\Delta p)} = \varepsilon_{\text{target}} - \omega_0
\]

(3.14)

was described as the modified target value for the strategy.

Evaluation

When evaluating the results of the different strategies, Ekman constructed an optimal strategy. This strategy looked at the solution generated prices to see when a calculation should have been performed. The other strategies results where then compared to the optimal strategy’s results to compare how well they performed in regard of how many calculations that was needed. How well a strategy performs is measured by calculating the relative difference between the results of that strategy and the result of the optimal strategy.

3.3 Results

The thesis successfully received results that managed to reduce the cost. The different sections will cover how successful it was for the different instruments.
3.3.1 Futures

For the futures the results showed that the no scenario relative change in stock price strategy and the no scenario relative error in stock price strategy performed better than the optimal solution. Except from those the relative error in stock price strategy was only 1.3% worse than the optimal strategy, in regard of number of recalculations. This means that in her results these were the best performing strategies.

3.3.2 European Options

For the European stock option the best strategy for both put and call options was the no scenario relative error of stock price strategy. This gave about 14% worse compared to the optimal solution for the call options and about 17% worse than the optimal solution for the put options, regarding number of recalculations.
4 Method

This section will cover the methods used to implement the different strategies and how they were set-up, run and evaluated. The important algorithms can be seen in Appendix A. The implementations were done in a way that would work in an IaaS delivery model. The strategies were implemented as microservices and run in a local cluster environment using Minikube [12].

4.1 European options and futures

This section will cover how European options and futures contracts were handled. Starting with the gathering of information to how the result was evaluated. The methods used under the following sections are based on the work done by Sara Ekman [1], which is also described in Chapter 3. As mentioned in Section 1.3 this will only cover options. The same methods are however applicable for put options as well.

4.1.1 Information gathering

The stocks that are used for testing these strategies can be seen in Table 1.

<table>
<thead>
<tr>
<th>Stock</th>
<th>Ticker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco Systems, Inc.</td>
<td>CSCO</td>
</tr>
<tr>
<td>Micron Technology, Inc.</td>
<td>MU</td>
</tr>
<tr>
<td>Intel Corporation</td>
<td>INTC</td>
</tr>
<tr>
<td>Microsoft Corporation</td>
<td>MSFT</td>
</tr>
<tr>
<td>Applied Materials, Inc.</td>
<td>AMAT</td>
</tr>
<tr>
<td>Sirius XM Holdings Inc.</td>
<td>SIRI</td>
</tr>
<tr>
<td>NVIDIA Corporation</td>
<td>NVDA</td>
</tr>
</tbody>
</table>

Daily closing prices for the stocks were gathered from Yahoo Finance by using the third party Quotes API for Yahoo Finance library [10]. The date range used for these prices was between 2013-12-30 and 2017-12-31 which gives 1009 entries. Daily interest rates was gathered from the U.S. Department of the Treasury [8] between 2013-12-30 and 2018-01-09 to get equally many entries. The collected values was then used to estimate the volatility corresponding to each day by using EWMA which is covered in Section 2.2.1. EWMA used a calibration period between 2005-01-01 and 2013-12-30 to be able to estimate the volatility as good as possible.

Three intra-day minute closing prices were then gathered through Google Finance [11].
This was done for three separate days 2018-03-06, 2018-03-21 and 2018-04-13. The volatility and interest rates was held constant during the day. The interest rate was assumed to be the same as the previous day and the volatility was estimated for the day by using a two year period as calibration up to that day.

### 4.1.2 Instrument set-up

To be able to use the strategies, instruments needed to be generated. Four different futures contracts was generated for each instrument. The differing factor being the expiration time $T$. The different values used was $T = 1, 3, 6$ and 12 months expressed in years. The interest rate used for each instrument was the one corresponding to each $T$.

European options used the same $T = 1, 3, 6$ and 12 months expressed in years and corresponding interest rates. The options was however set up with three different strike prices $K = 0.9S_0, S_0$ and $1.1S_0$. Thus generating 12 different call options.

### 4.1.3 Scenario set-up

By using the gathered daily values scenarios was set up in triples describing the percentage change in each value between each day. This means that for each new day the shifts in price, risk free interest rate and volatility was saved into a scenario. This set-up generated 1008 different scenarios.

As Algorithm 1 in Appendix A describes, the gathered data is used to for each stock form scenarios containing the daily shifts concerning that stock. This is done by iterating the stocks, calculating the shifts and saving them.

### 4.1.4 Solution generation

To be able to measure the effectiveness of each method a solution file was generated containing all scenarios for every minute for every instrument.

Algorithm 2 in Appendix A describes the method of generating solutions. This way a map structure containing instrument id as key and a map, with each minute as keys, as value. This to keep the data ordered for each instrument each minute.

### 4.1.5 Strategies

This section will cover the parts regarding running the simulations. Algorithms describing the simulations of a day and how calibrated values was acquired will be covered.

Each strategy uses a value $\epsilon_{\text{target}}$ that is the maximum allowed mean relative error for any instrument at any point during a day. For European options this value is chosen to 0.15 and for futures it is chosen to 0.01. This selection was done through discussions considering the maximum possible error during a day and the value needed to make it possible for the constant method to be able to complete.
Constant

To be able to perform a simulation for this method constant time steps needed to be found. This was done through running simulations until every instrument managed to hold for Equation 3.1. Each individual instrument gets its own frequency of recalculation. This way, the scenario recalculations are not performed if not necessary. To find this frequency, a day is simulated with 60 minute intervals and the maximum relative error of each instrument checked. Any instrument not holding for Equation 3.1 gets its interval halved.

By using Algorithm 3 and 4 listed in Appendix A, it is possible to simulate the a day using the data and scenarios obtained in previous sections. The simulation results in the minutes of when to recalculate, the total number of recalculations for each instrument, and the total duration accumulated during the simulated day.

This strategy will in this thesis be denoted as \textit{const}.

Simulation of the relative strategies

The rest of the strategies, Strategy 1, 2 and 3, are simulated in a similar fashion. By first iterating the day, minute by minute, and for each minute iterating over each instrument to find which ones that needs to be recalculated.

As Algorithms 5 and 6 listed in Appendix A show, these methods require a subset selection of scenarios. This selection was set to cover the scenarios that are most extreme. That means the scenarios with the biggest and smallest shifts in price, rate and volatility respectively. Thus the subset contained 6 scenarios. These scenarios would however not necessarily give the biggest changes in price but should in many cases be close to that. Also worth mentioning is that only one strategy requires recalculation of the subset every minute. The needsRecalculation method used in Algorithm 6 is an abstract method to be implemented by each strategy.

When not considering any scenarios for evaluating when to perform recalculations, the algorithm for simulating a day can be found in Algorithm 7 in Appendix A and for each minute it uses Algorithm 8 also listed in Appendix A. These algorithm uses the current prices instead of looking at the scenarios to decide when to perform the recalculations.

Modified target value

Since the following methods introduced additional error terms it was required to find what was called a modified target value to make up for the extra error that can occur when performing the following strategies. This was done by starting at the set target values which was 0.15 for European options and 0.01 for Futures. If a strategy did not hold for Equation 3.1 during a day the limit was halved. If it did hold and it was possible to use a higher limit without exceeding the the real target value it set it to be in the middle of the previous not holding limit and the last holding limit. This way it closed down the modified target value from both sides to find an optimal value. This algorithm can be seen in Algorithm 9 listed in Appendix A.
Relative change in stock price — Strategy 1

This method is best described by Equation 3.2. This method does not require subset recalculations of current subset.

When not considering any scenarios it is instead described by Equation 3.4.

Relative error in stock price — Strategy 2

This method is similar to the previous one but instead of looking at the relative change it looks at the relative error of the stock price. The expression that describes this method can be found in Equation 3.6. This method estimates the current scenario price and does therefore not need to calculate the current subset scenario prices.

When not considering any scenarios it is instead described by Equation 3.9.

Relative error in instrument price — Strategy 3

This method looks at the relative error of the instrument price. The expression that describes this method can be found in Equation 3.11. This method that calculates the current subset of scenario prices.

When not considering any scenarios it is instead described by Equation 3.13.

Evaluation

To best evaluate the results of each method the first thing to do is to make sure that the maximum error of each method is within the bounds of Equation 3.1. For at least the calibration day which is 2018-03-06. The other two days are used get a feel for how these methods perform with these values when not knowing the end result. The mean relative error for each method is also noted to see if any method is more prone to making big mistakes. And since this thesis uses Amazon Lambda [9], described in Section 2.3.5, to evaluate the price. The accumulated time used during the day for all price updates is measured. Assuming that this system will use 1GB of memory and that the free tier is not applicable, it will cost $0.000001667/100ms. This way it is possible to set a comparable dollar value to each method to be able to see the cost reduction of each strategy compared to the strategy with constant intervals. The average number of recalculations needed is also interesting mostly because this uses a lot of resources. This will however with this pricing model not effect the cost.
5 Results

This section will cover the results from simulations over all three days. One of them is used for calibrating modified target values and time intervals. The durations varies between runs depending on the available resources on the hardware. To get a better result when it comes to durations, the average over five simulations is used. In this section all strategies except for the constant one is referenced by number and prefixed by ns if it is without considering any scenarios.

5.1 European options

When considering the European options the constant intervals mostly consisted of recalculating every hour. Though some options with $T = 1$ month needed recalculations up to every third minute. The modified target values received for the calibration day can be seen in Table 2. The values show that Strategy ns3, which is looking at the relative error in instrument price when not considering any scenarios, gets the least additional error terms.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mod. target value</td>
<td>0.0706</td>
<td>0.0732</td>
<td>0.0732</td>
<td>0.0985</td>
<td>0.0896</td>
<td>0.1037</td>
</tr>
</tbody>
</table>

The results of using these values for the calibration day can be seen in Table 3. As the table shows, the durations and number of recalculations needed is far less on the strategies not considering any scenarios. As for the maximum error the strategies are similar apart from the strategy with a constant time step.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>Const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.1415</td>
<td>0.1424</td>
<td>0.1347</td>
<td>0.1416</td>
<td>0.1416</td>
<td>0.1427</td>
<td>0.1488</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>11738</td>
<td>11480</td>
<td>9522</td>
<td>2927</td>
<td>3330</td>
<td>2488</td>
<td>5960</td>
</tr>
</tbody>
</table>

As for the simulations on 2018-03-21 the results can be seen in Table 4. This table shows that the strategies not considering any scenarios are by far the fastest. It is also noticeable that none of the strategies hold for the requirement of a maximum relative error of 0.15. Strategy ns3, the strategy of relative error in instrument price when not considering any scenarios, has however the lowest maximum relative error and is the fastest as well. The constant strategy is the one with the highest maximum relative error, which is close to double the target of 0.15.
Table 4: Results from simulations on 2018-03-21.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.2199</td>
<td>0.1824</td>
<td>0.2199</td>
<td>0.2199</td>
<td>0.2199</td>
<td>0.2199</td>
<td>0.1811</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>10692</td>
<td>10110</td>
<td>7766</td>
<td>2607</td>
<td>2897</td>
<td>1901</td>
<td>7137</td>
</tr>
</tbody>
</table>

In Table 5 the results of the simulations on 2018-04-13 is displayed. This day the constant strategy has a very high maximum relative error. This day the strategies ns1 and ns2, relative change and relative error in instrument price without considering any scenarios, both has maximum relative errors below the target of 0.15. They also have short accumulated durations. Other than the constant strategy the maximum relative errors are considerably lower than the previous day as well.

Table 5: Results from simulations on 2018-04-13.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.1531</td>
<td>0.1802</td>
<td>0.1531</td>
<td>0.1498</td>
<td>0.1498</td>
<td>0.1801</td>
<td>0.3362</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>10521</td>
<td>10251</td>
<td>8151</td>
<td>2820</td>
<td>3125</td>
<td>2405</td>
<td>5695</td>
</tr>
</tbody>
</table>

The total cost in USD of the simulations per day and daily average can be seen in Table 6. This table shows that strategy ns3, relative error in instrument price not considering scenarios, was the cheapest over all days and on average.

Table 6: Cost in USD for each strategy.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 1</td>
<td>0.000197</td>
<td>0.000192</td>
<td>0.000160</td>
<td>0.000050</td>
<td>0.000057</td>
<td>0.000042</td>
<td>0.000010</td>
</tr>
<tr>
<td>Day 2</td>
<td>0.000178</td>
<td>0.000170</td>
<td>0.000130</td>
<td>0.000045</td>
<td>0.000048</td>
<td>0.000033</td>
<td>0.000012</td>
</tr>
<tr>
<td>Day 3</td>
<td>0.000177</td>
<td>0.000172</td>
<td>0.000137</td>
<td>0.000048</td>
<td>0.000053</td>
<td>0.000042</td>
<td>0.000095</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.000184</td>
<td>0.000178</td>
<td>0.000142</td>
<td>0.000048</td>
<td>0.000053</td>
<td>0.000039</td>
<td>0.000101</td>
</tr>
</tbody>
</table>

5.2 Futures

When the strategies were calibrated for futures the target maximum relative error that was used was 0.01. The time steps for the constant method varied a bit more than the European options time steps. The majority was however every hour or half hour. The modified target values retrieved for futures can be seen in Table 7. The are very similar for all strategies with additional error terms of about 0.003.

Table 7: The modified target values for each strategy.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mod. target value</td>
<td>0.00727</td>
<td>0.00723</td>
<td>0.00723</td>
<td>0.00727</td>
<td>0.00720</td>
<td>0.00723</td>
</tr>
</tbody>
</table>
During the calibration day all strategies hold for the target maximum relative error. There is also a notable difference in durations between the strategies. The ones not considering any scenarios is about 45 times faster than the ones that are considering scenarios and about 20 times faster than the constant strategy. These results can be seen in Table 8.

Table 8: Results from simulations on the calibration day 2018-03-06

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.00958</td>
<td>0.00958</td>
<td>0.00958</td>
<td>0.00958</td>
<td>0.00958</td>
<td>0.00958</td>
<td>0.00954</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>1233</td>
<td>1265</td>
<td>1291</td>
<td>26</td>
<td>28</td>
<td>28</td>
<td>643</td>
</tr>
</tbody>
</table>

When using the calibrated values on 2018-03-21 the acquired results can be seen in Table 9. These values show that every strategy except the constant one holds for the target maximum relative error. The constant strategy has a similar duration as the strategies using scenario subsets to calculate when to perform recalculations. The ones not considering any scenarios are however faster by about 45 times on this day as well.

Table 9: Results from simulations on 2018-03-21

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.00916</td>
<td>0.00916</td>
<td>0.00916</td>
<td>0.00916</td>
<td>0.00916</td>
<td>0.00916</td>
<td>0.0123</td>
</tr>
<tr>
<td>Avg. Nr recalc</td>
<td>25.857</td>
<td>27.571</td>
<td>27.571</td>
<td>6.0</td>
<td>6.286</td>
<td>6.286</td>
<td>68.143</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>1265</td>
<td>1195</td>
<td>1312</td>
<td>20</td>
<td>29</td>
<td>28</td>
<td>1143</td>
</tr>
</tbody>
</table>

The last day of simulations show results very similar as the previous day. Except for the duration of the constant strategy that indicates that the strategy was about twice as fast on this day. The fastest strategies was by far the ones not considering any scenarios. This can be seen in Table 10.

Table 10: Results from simulations on 2018-04-13

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. rel. error</td>
<td>0.00992</td>
<td>0.00900</td>
<td>0.00900</td>
<td>0.00992</td>
<td>0.00992</td>
<td>0.00900</td>
<td>0.0171</td>
</tr>
<tr>
<td>Avg. Nr recalc</td>
<td>22.0</td>
<td>24.571</td>
<td>24.571</td>
<td>5.429</td>
<td>5.714</td>
<td>6.286</td>
<td>68.143</td>
</tr>
<tr>
<td>Duration(ms)</td>
<td>1073</td>
<td>1188</td>
<td>1151</td>
<td>21</td>
<td>20</td>
<td>34</td>
<td>612</td>
</tr>
</tbody>
</table>

The cost in USD of each strategy per day and daily average can be seen in Table 11. This shows that all strategies that are not considering any scenarios are the cheapest with all of them rounding up to 100ms per day.

Table 11: Cost in USD for each strategy

<table>
<thead>
<tr>
<th>Strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
<th>const</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 1</td>
<td>0.000022</td>
<td>0.000022</td>
<td>0.000022</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.000012</td>
</tr>
<tr>
<td>Day 2</td>
<td>0.000022</td>
<td>0.000020</td>
<td>0.000023</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.000020</td>
</tr>
<tr>
<td>Day 3</td>
<td>0.000018</td>
<td>0.000020</td>
<td>0.000020</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.000012</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.000021</td>
<td>0.000021</td>
<td>0.000022</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.0000017</td>
<td>0.000014</td>
</tr>
</tbody>
</table>
6 Discussion

This section discusses the results. It also goes into more detail and discusses theories about why the results turned out the way they did.

6.1 European options

The main thing is that the maximum relative errors on 2018-03-21 is over the target. This is however not that strange since the calibrated values are calibrated for a specific day and if another day is more volatile and have more and larger price changes during the day the results may not match. And the maximum relative errors describes more how the strategy matches the changes of the days changes.

The calibration days result can’t really be discussed that much since the values used are adjusted according to that day.

As for the duration of each strategy it is dependant on the amount of recalculations needed for each strategy and the strategies not considering any scenarios had over all days the lowest amount of recalculations and the shortest durations. Thus giving them the lowest costs.

To get the best results of the calculations they will need to be performed as often as possible. But if one of the strategies was to be used and knowingly reduce accuracy, the results show that the best options for price reduction would be to go with the strategy looking at the relative error in instrument price without considering any scenarios.

With a performance focused cost model the same strategies would likely be the cheapest as well. This because they don’t require as many recalculations when determining when to perform the recalculations and also have the fewest recalculations.

6.2 Futures

All of the strategies except for the constant one seemed to perform very well considering the kept the maximum relative errors below the target. And the strategies notably were very similar in maximum relative error as well. This may be because the pricing of futures don’t include volatility which may differ a lot between days thus giving the scenarios big shifts in volatility. Since it is not considered here it may have kept the results closer to each other and seemingly more stable in regards of maximum relative error than European options.

This result clearly favored the strategies not considering any scenarios. The number of recalculations needed were the fewest and the duration the shorted resulting in them being
by far the cheapest strategies. It may be caused by the selection of scenario subset. Different subsets were tested but did not perform better than the selected. There may however be untested subsets that are better. This also applies for European options, as their results also showed a big difference between using a subset or not.

This would probably be the case in a more performance based cost metric as well, since these calculations are a lot easier than those considering subsets of scenarios and have far less recalculations. Thus less prone to scale up using replicas of the microservice.

The result showed that on a daily basis the cost of the strategies not concerning any scenarios were equally cheap regardless of difference in durations. This is because they all were under 100ms which was the limit of the cost model for Amazon Lambda [9]. If it instead would have been calculated as a monthly cost on the daily average duration it would have resulted in showing that ns1, relative change in stock price without considering any scenarios, would have been the cheapest.
7 Conclusion

In this thesis numerous strategies has been tested to try to reduce the cost of running risk calculations in a cloud environment compared to the strategy of constant recalculation intervals. The goal of the thesis was set to reduce the cost by 5%. The different strategies was calibrated from one day and tested over two more days. This way it is possible to get a sense of how the strategies can perform in a more general setting.

The work performed showed that within the specific cases that was used in this thesis it is possible to use the strategies to reduce the cost by more than 5%. The strategies costs was compared to the constant strategy reduced up to about 82% for futures and about 61% for European options. This being said it should also be mentioned that the strategies maximum relative errors differed over the different days and did not all hold for the target relative errors.

The primary conclusion that can be drawn from this thesis is that at least some of the strategies, proposed by Sara Ekman in [1], can be used to reduce the cost of running risk calculations in a cloud environment, compared to using a constant time step. This is with the same limitation of having to uphold a certain maximum relative error on the calibration day. Other days may however not be able to hold for that limit but will most likely still perform better that the constant strategy and reduce the cost.
8 Future work

This work could definitively be continued to work on and further improve. A few examples of what can be done will be covered here.

Using multiple days as calibration days and testing over longer periods. This to test if it would result in better performance of the methods. As this thesis only used one calibration day and two days as tests this should be an interesting continuation.

Another interesting more mathematical approach would be to take on the Asian options and find strategies that are applicable for a more realistic case. This would however demand a lot of theoretical work to figure out how such methods may work and what they want to achieve.

If one would instead look into the possibility of how to calculate the number of machines needed to perform this application in a cloud environment it may further be possible to reduce cost and find other optimal strategies. This makes this another way of continuing this work.

By expanding the amount of stocks and also including put options the time to perform these simulations would increase rapidly. It would however give more general results and would make results more general than those acquired in this thesis.
References

[1] Price Vector Recalculation Optimization
Sara Ekman
Department of Physics, Umeå University, 2017

https://www.investopedia.com/terms/c/clearinghouse.asp
visited 2018-04-19

[3] Collateral, central clearing counterparties and regulation European Central Bank
visited 2018-05-15

https://www.finpipe.com/risk-assessment/
visited 2018-05-15

Judith Hurwitz, Marcia Kaufman, Dr. Fern Halper
John Wiley & Sons, Inc. 2012

[6] Options, Futures, and Other Derivates
John C. Hull
Pearson Education

Google
https://finance.yahoo.com
Visited 2017-02-23

[8] Resource center: Daily treasury yield curve rate
Visited 2017-02-23

[9] AWS Lambda Pricing
Amazon
https://aws.amazon.com/lambda/pricing/
Visited 2018-02-26
[10] *Quotes API for Yahoo Finance*
   Stijn Strickx
   https://financequotes-api.com/
   Visited 2018-04-16

   Google
   https://www.google.com/finance/
   Visited 2018-04-16

[12] *Running Kubernetes Locally via Minikube*
   Kubernetes
   https://kubernetes.io/docs/getting-started-guides/minikube/
   Visited 2018-05-16
A Appendix: Algorithms

This appendix is for the more technical readers and contain the algorithms for simulations and finding calibrated values.

Algorithm 1: Scenario generation

**Input parameters:** tickers, dailyStockPricesMap, calibrationStockPricesMap, interestRates

initialize scenarioMap;
rateShiftList = calcDailyShifts(interestRates);
foreach ticker in tickers do
  dailyStockPrices = dailyStockPricesMap.get(ticker);
  priceShiftList = calcDailyShifts(dailyStockPrices);
  stockCalibrationPrices = calibrationStockPricesMap.get(ticker);
  volatilities = EWMA.estimateDailyVolatilities(dailyStockPrices, stockCalibrationPrices);
  volatilityShiftList = calcDailyShifts(volatilities);
  scenarioMap.put(ticker, new StockScenarios(priceShiftList, volatilityShiftList, rateShiftList));
end

return scenarioMap;
Algorithm 2: Solution generation

**Input parameters:** instruments, scenariosMap, stockMinutePricesMap, interestRate, stockVolatilityMap

initialize solutionMap;

**foreach** instrument **in** instruments **do**

    ticker = instrument.getUnderlyingAsset();
    minutePriceList = stockMinutePricesMap.get(ticker);
    stockScenarios = scenariosMap.get(ticker);
    stockVolatility = stockVolatilityMap.get(ticker);
    initialize minuteSolutionMap;
    **for** minute = 0 ; minute < minutePriceList.size() ; minute++ **do**
        minutePrice = minutePriceList.get(minute);
        inst = instrument.copy();
        inst.setStockPrice(minutePrice);
        inst.updatePrice();
        instrumentScenarios = calculateScenarioPrices(instrument, stockScenarios, minutePrice, stockVolatility, interestRate);
        minuteSolutionMap.put(minute, instrumentScenarios);
    end
    solutionMap.put(instrument.getId(), minuteSolutionMap);
end
return solutionMap;

Algorithm 3: Constant recalculation strategy day simulation

**Input parameters:** instruments, scenariosMap, stockMinutePricesMap, interestRate, stockVolatilityMap, timesteps

initialize nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap;
totalDuration = 0;
**for** minute = 0 ; minute < minutePriceList.size() ; minute++ **do**
    duration = simulateMinute(intitutions, stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, timesteps, nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap, minute);
    totalDuration += duration;
end
return new SimulationResult(nrRecalculationsMap, recalcMinutesMap, totalDuration);
Algorithm 4: Constant recalculation strategy minute simulation

**Input parameters:** instruments, stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, timesteps, nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap, minute

start = currentTimeMillis();
currentInstruments = getCurrentInstruments(instruments, timesteps, minute);
if currentInstruments.size() == 0 then
    return currentTimeMillis()-start;
end

instrumentScenarioPrices = executeRecalculations(stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, minute, currentInstruments, nrRecalculationsMap);
prevScenarioPricesMap.updateAll(instrumentScenarioPrices);
updateRecalculationMinutes(currentInstruments);
return currentTimeMillis()-start;

Algorithm 5: Relative recalculation strategy day simulation

**Input parameters:** tickers, instrumentsMap, scenariosMap, stockMinutePricesMap, interestRate, stockVolatilityMap, limit

initialize nrRecalculationsMap, prevScenarioPricesMap, prevScenariosSubset, recalcMinutesMap;
totalDuration = 0;
subset = getSubset(scenariosMap);
for minute = 0 ; minute < minutePriceList.size() ; minute++ do
    duration = simulateMinute(tickers, instrumentsMap, stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap, minute, subset, prevScenariosSubset, strategy, limit);
totalDuration += duration;
end

return new SimulationResult(nrRecalculationsMap, recalcMinutesMap, totalDuration);
Algorithm 6: Relative recalculation strategy minute simulation

**Input parameters:** tickers, instrumentsMap, stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap, minute, subset, prevScenariosSubset, strategy, limit

start = currentTimeMillis();
initialize toRecalculateList;
**foreach** ticker in tickers **do**
    stockInstruments = instrumentsMap.get(ticker);
    initialize currentScenariosSubset;
    **if** strategy is RelativeErrorInstrumentPrice **then**
        currentScenarioSubset = executeRecalculations(stockMinutePricesMap, subset, interestRate, stockVolatilityMap, minute, stockInstruments);
    **end**
    **for** instrument in stockInstruments **do**
        **if** instrument not in prevScenarioPricesMap **then**
            toRecalculate.add(instrument);
            addRecalcMinute(instrument, minute);
            initialize nrRecalculations value for instrument;
        **end**
        **else if** strategy is RelativeErrorInstrumentPrice AND needsRecalculation(prevScenariosSubset.get(instrument), currentScenariosSubset.get(instrument), limit) **then**
            toRecalculate.add(instrument);
            addRecalcMinute(instrument, minute);
        **end**
        **else if** needsRecalculation(prevScenariosSubset.get(instrument), null, limit) **then**
            toRecalculate.add(instrument);
            addRecalcMinute(instrument, minute);
        **end**
    **end**
    **else if** toRecalculate.size() > 0 **then**
        currentScenario = executeRecalculations(stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, minute, toRecalculate, nrRecalculations);
        prevScenarioPricesMap.updateAll(currentScenario);
    **end**
**end**
return currentTimeMillis()-start;
Algorithm 7: No scenario relative recalculation strategy day simulation

Input parameters: tickers, instrumentsMap, scenariosMap, stockMinutePricesMap, interestRate, stockVolatilityMap, limit

initialize nrRecalculationsMap, prevScenarioPricesMap, prevCalculatedPrices, recalcMinutesMap;
totalDuration = 0;
for minute = 0 : minute < minutePriceList.size() ; minute++ do
  duration = simulateMinute(tickers, instrumentsMap, stockMinutePricesMap, scenariosMap, interestRate, stockVolatilityMap, nrRecalculationsMap, prevScenarioPricesMap, recalcMinutesMap, minute, prevCalculatedPrices, strategy, limit);
  totalDuration += duration;
end
return new SimulationResult(nrRecalculationsMap, recalcMinutesMap, totalDuration);
Algorithm 8: No scenario relative recalculation strategy minute simulation

**Input parameters:** tickers, instrumentsMap, stockMinutePricesMap, scenariosMap,
interestRate, stockVolatilityMap, nrRecalculationsMap,
prevScenarioPricesMap, recalcMinutesMap, minute,
prevCalculatedPrices, strategy, limit

start = currentTimeMillis();
initialize toRecalculateList;

**foreach ticker in tickers do**
  stockMinutePrices = stockMinutePricesMap.get(ticker);
  stockInstruments = instrumentsMap.get(ticker);
  **for instrument in stockInstruments do**
    minutePrice = stockMinutePrices.get(minute);
    if instrument not in prevScenarioPricesMap then
      toRecalculate.add(instrument);
      addRecalcMinute(instrument, minute);
      initialize nrRecalculations value for instrument;
      prevCalculatedPrices.put(instrument, minutePrice);
    else
      lastMinutePrice = prevCalculatedPrices.get(instrument);
      last = instrument.copy();
      last.setStockPrice(lastMinutePrice);
      last.setElapsedTime(minute / (391 * 252));
      curr = instrument.copy();
      curr.setStockPrice(minutePrice);
      curr.setElapsedTime(minute / (391 * 252));
      last.updatePrice();
      curr.updatePrice();
      if needsRecalculation(last, curr, lastMinutePrice, minutePrice, limit) then
        toRecalculate.add(instrument);
        addRecalcMinute(instrument, minute);
        prevCalculatedPrices.put(instrument, minutePrice);
      end
    end
  end
  if toRecalculate.size() > 0 then
    currentScenario = executeRecalculations(stockMinutePricesMap, scenariosMap,
                                           interestRate, stockVolatilityMap, minute, toRecalculate, nrRecalculations);
    prevScenarioPricesMap.updateAll(currentScenario);
  end
end
return currentTimeMillis()-start;
Algorithm 9: Modified target value finder

limit = futures ? 0.01 : 0.15;
realLimit = futures ? 0.01 : 0.15;
delta = futures ? 0.0001 : 0.0005;
max = limit;
double min = 0;
while true do
  holding = maxErrorOfSimulation(run strategy simulations....);
  if not holding then
    if limit < max then
      max = limit;
    end
    limit -= Math.abs(limit-min)/2;
  end
  else
    if limit > min then
      min = limit;
    end
    if (max-limit)/2 < delta OR (limit + max-limit)/2 > realLimit then
      break;
    end
    limit += (max-limit)/2;
  end
end
return limit;