Fast Calculation of Value-at-Risk Using Monte Carlo Simulations and Distributed Computing

Peter Verhoog
Verhoog Consultancy

Marko Koskinen
Techila Technologies Ltd

28 June 2017

1 Introduction

One of the most common risk measures in the finance industry is Value-at-Risk (VaR). Value-at-Risk measures the amount of potential loss that could happen in a portfolio of investments over a given time period with a certain confidence interval. Historically, banks have been using VaR in trading environments. Although VaR has been somewhat controversial as a risk measure, it is currently widely used within the finance industry. Also Solvency II, the regulatory framework for the insurance industry, uses VaR as a measure of risk.

It is possible to calculate VaR in many different ways, each with their own pros and cons. Monte Carlo simulation is a popular method. Monte Carlo simulation can be used also for complex portfolios. The downside of Monte Carlo simulations is the fact that the calculation is computationally intensive. This can lead to a speed-accuracy trade-offs, where the timeframe within which the calculations need to be performed, is not sufficient for creating accurate VaR figures. It is also possible that the time that it takes to perform the calculation are an operational challenge.

The following section explains a simplified implementation of a VaR model, based on Monte Carlo simulation. The available MATLAB and R code examples enable performance comparison of the model in these two popular programming environments. The final chapter demonstrates that the calculation time for Monte Carlo simulations can be effectively decreased by using a scalable distributed computing solution.

2 The VaR model

In our simplified VaR model, the value of a portfolio of financial instruments is simulated under a set of economic scenarios. The scenarios are based on risk factors. The risk factors are inputs to the valuation models of the financial instruments in the portfolio. The scenarios are created randomly, and all have even probability to occur. The economic scenarios are generated based on the estimated future volatility of the risk factors. The correlations between the risk drivers are not taken into account in this simplified example model.

The set of financial instruments in this example is limited to a set of fixed coupon bonds and equity options. To simplify the calculations, the fixed coupon bonds in the example do not have credit risk. The risk drivers, the inputs to the pricing models of the instruments, are changed in the economic scenarios. As a result, the value of the portfolio of instruments will change for each scenario. The probability distribution of these portfolio values gives insight to the risk profile of the portfolio. In this example, the 0.5% percentile of the distribution is used to estimate the 99.5% VaR of the portfolio.

The VaR model consists of three sections:

- run_VaR_cloudfor
- parameters
- bs_function

MATLAB and R versions of the code are included in the latest Techila SDK. Download links to the code material can also be found on the Techila Technologies website at: http://www.techilatechnologies.com

The role of each section is explained in more detail below. Underlined text refers to a source code file with the specified name. For example, run_VaR_cloudfor refers to source code files *run_VaR_cloudfor.m* and *run_VaR_cloudfor.R*.

The run_VaR_cloudfor file contains the main function that triggers the other sections of the code.

The parameters section defines the parameters that are used in the model. It starts by the defining the number of scenarios and the number of instruments. The market data is defined in the second section. A yield curve and a volatility surface are defined. In the third section of the code, the portfolio of instruments is defined. The option instruments are created by randomly creating the inputs to the black-scholes model. The bond instruments are created by randomly selecting a maturity date and a coupon rate. The cash flows of fixed coupon bonds are not scenario-dependent, and are calculated once. The cash flows will be discounted with different discount rates in a later section of the code. In the fourth section of the parameters function, the position vector is created. The position matrix defines the weight of an instrument in the total portfolio. The scenarios are defined in the fifth section of the code. The last part of the parameters section is used to define the Principal Component Analysis (PCA) factors, and volatility shock surface. These parameters define how the changes in risk factors are applied to the yield curve, or the volatility surface.

The bs_function of the code is an implementation of the black-scholes model.

When we run the run_VaR_cloudfor function, it triggers first the parameters code. Once the market data, instruments, and positions are defined, the code values the portfolio of instruments under the current conditions. This is called the basePortfolioValue.

The next section of the code will be the computationally intensive part of the code. This section uses a cloudfor loop to make use of distributed computing in performing the calculations. Cloudfor is a function that is available in the MATLAB toolbox and R package of the Techila Distributed Computing Engine (TDCE). In this section of the code, all the instruments are valued under all scenarios. When multiplied with the position matrix, this results in the portfolio value under the economic scenarios. This vector of portfolio values can be used for risk analysis, and to estimate the VaR the portfolio.

3 Results

When using Techila Distributed Computing Engine (TDCE) to calculate the VaR measures, the performance gains will depend on following:

- Available computing resources
- Total number of scenarios computed
- Amount of data transferred
- How many scenarios are computed in each Job

For this paper, we run the MATLAB and R programming language versions of the VaR code. In our test run, we computed **600.000** scenarios for **12.000** instruments. We used a test environment with **400** Techila Worker CPU cores. Each Job computed **500** scenarios. The performance metrics for the test run are described below.

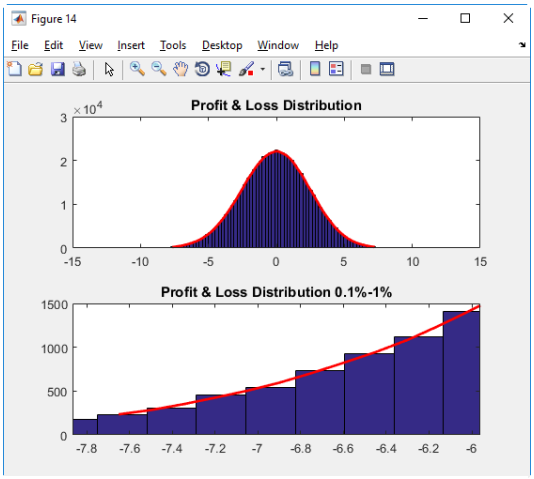

Figure 1. MATLAB computation results

MATLAB vs R

CPU time gives us a rough approximation on how long it would have taken to run a simulation on a single CPU core machine with similar hardware specifications.

When we run the **MATLAB** code, it used **8 h 57 m 37 s** CPU time.

When we run the R version of the same code, the CPU time used was **1 d 12 h 10 m 29 s.**

This shows that the performance of R programming language is not as good as the performance of MATLAB.

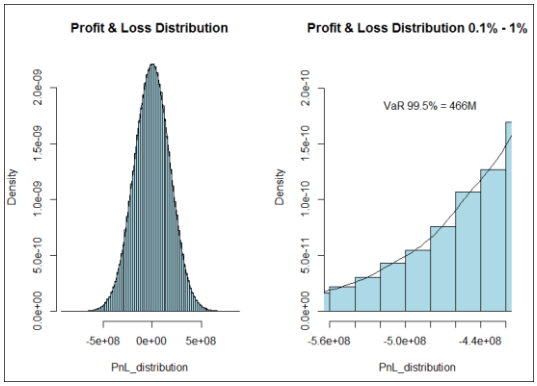

Figure 2. R computation results

MATLAB vs R with scalable distributed computing

Techila Distributed Computing Engine makes it easy to scale out to a distributed computing engine. The MATLAB toolbox and the R package of TDCE enable scalable computing directly from MATLAB and R Studio.

When we run of the MATLAB code using 400 Techila Workers, the computations were completed in **1 m 39 s.**

When we run of the R version of the same code in the same TDCE system, these R computations were completed in **6 m 7 s**.

4 Conclusion

A VaR calculation that use Monte Carlo simulation can be time consuming. The operational processes around VaR calculations can benefit, if there is an easy way to accelerate the Monte Carlo simulations, and cut down the time **from hours to minutes**. Techila Distributed Computing Engine is a next generation grid that enables fast simulation without the complexity of traditional high-performance computing.

In addition to speed, TDCE offers also other benefits, such as real-time access to intermediate results. This can help the user to spot potential data or model issues early, which supports productivity and usability.

The reduction of the waiting time of the VaR calculations and enablement of interactive use of the simulation model can also more extensive risk analysis.

Fast computing also enables new ways of working. Fast VaR calculations using TDCE can, for example, be used to calculate standalone VaR for a smaller amount of risk factors (for example, calculate an interest rate VaR per currency, instead of one interest rate VaR for the portfolio). The increased calculation speed can also enable (or simplify) the calculation of VaR sensitivities, the change of the VaR, when the underlying risk factors move. Especially VaR sensitivities are very useful for risk management purposes, because the VaR sensitivities can predict what will happen with your risk profile during stressed market circumstances. A reduced calculation time can also be used for portfolio optimization, because it will be easier to calculate the portfolio VaR with different instrument weightings.

As we saw in our tests, there is a substantial difference in the processor time usage of MATLAB and R programming language in these VaR examples. The run takes a significant time in both languages, but MATLAB's 9 hours is much faster than R's 1 day 12 hours. If we calculate the VaR measures sequentially on a single computer, this difference could render the use of the slower programming language not feasible.

However, when we use TDCE and distributed computing to calculate the models, we can adjust the degree of parallelism that we would like to use. In our tests, TDCE distributed the work to a computing infrastructure with 400 CPU cores. This reduced the wall-clock times and the time difference between the MATLAB and R runs. It can be said that the VaR calculation was completed quickly in both languages. **MATLAB: 1m 39s. R programming language: 6m 7s.**

From a performance point of view, we can say that the use of TDCE and distributed computing can reduce the effect of the performance differences of the programming languages. The underlying technical performance differences still do remain, but they will have a less significant impact to the efficiency of the financial engineer's work. This will also allow the user to choose the language that works best for the task that he has on his hands.

The study of different computing infrastructures is not included in the scope of this paper. Information on the price-performance aspects can be found in the Techila Technologies report "Cloud HPC In Finance".