C++ in QF team project

Radost Waszkiewicz

2023–12–03

1 The brief

An algo-trader asked to help her implement a risk management component to her trading scheme. As a first step she would like you to compute payout variance of a portfolio of options. (This is an inadequate risk management policy, but it is definitely better than nothing).

The trader intends to use your program to check whether the next purchase will fit within a variance constraint, so speed of the operation is very important to her. Fortunately there is only a small number of options that are traded on this market and approximate values of variances are ok for her needs, so you can run some slow computations in the morning before the stock exchange opens and use the outcomes throughout the day.

All of her code is written in c++ so you are required to work in c++ as well. Before each purchase she would like to make a call to a method of the class that you are implementing which should return an object describing variance before and after purchase.

2 Technical specifications

When writing your code take into consideration following information

1. The typical value of the underlying is approximately 5000. (You can imagine an index such as the S&P 500)
2. The standard deviation of the underlying is around 20% year over year.
3. A typical return for an index-tracking portfolio is +5% annually.
4. The class needs to handle only two types of options – european put and european call.
5. Both long and short contracts should be handled within the class.
6. Options considered have a time to expiry ranging from at least a day to no more than a year.
7. The available option strikes are multiples of 100, extending up to 10,000.
8. Your class should track the sequence of purchases, allowing for a reset to a blank portfolio when necessary.

3 Presentation Guidelines and Grading

In the upcoming presentation, you will be graded on the following criteria:

Computational Task: Begin by executing a computational task before the audience. Load a sequence of purchases from a file (example file will be provided at least a week before the presentation) and compute the variance before each transaction. If, at any point, the variance exceeds a specified cutoff, your program should stop. Present the computed before-and-after variances for each instance where the cutoff is surpassed.

API Presentation: Move on to elucidate the API of your implemented class. Clearly articulate how users can use the class, how to add options, resetting the portfolio, and retrieving variance information.

Code Testing Procedure: Subsequently, provide a overview of your code testing procedures. Discuss the strategies and methodologies you employed to ascertain the correctness of your program.

Code Speed and Caching: Finally discuss the speed of your code. Elaborate on how the speed of operation is influenced by the number of options and highlight any optimizations implemented to enhance overall performance. If applicable, detail the caching mechanisms employed, particularly regarding information cached in the morning before the stock exchange opens.

Good understanding of limitations of your code is more important than the code performance. Give reasoning about precision and performance of your approach, justify choice of any free parameters in your code.

Keep your presentation brief – you'll have a few minutes to complete the computational task and then **a total of 5 minutes** to present your API, testing and talk about speed.

Good luck!