Group AB:

Design Decisions:

Classes and Inheritance Hierarchy

Refer to files: Option.hpp&.cpp; EuroOpt.hpp&.cpp; EuroStockOpt.hpp&.cpp; EuroFutureOpt.hpp&.cpp;

EuroCurrencyOpt.hpp&.cpp; PerpAmOpt.hpp&.cpp

For any specific option, its parameters will be a subset of the following parameters: S, K, T, sig, r, b, q, R. Different styles of options could have very different combinations of parameters. However, the common ones for any Option are S, K, r and sig. Thus, these parameters are encapsulated into the base Option class with another Boolean data member, isCall, to indicate Call / Put status (1 means Call, 0 means Put).

After declaring the data members for the Option class, member functions need to be included. This depends on what behaviors are expected from an Option in general. First of all, Constructors (Default, Parameter, Copy), Destructors and assignment operator = are needed. Second, Setter and Getter functions are needed for its data members, plus IsCall() function to return the Call / Put status and Toggle() function to switch status between Call and Put. In this specific application, the data members could have been set as "public". In this way, Setter and Getter would not be needed. However, setting data members as "private" is better practice for the purpose of data hiding and modification control. That's why the data members are set as "private".

Besides the above functions, the Option class should also include functions to calculate its price, approximate Greeks using divided differences, exact Greeks (if available), and a function named ToString() to show all its parameters to the screen. These are all common behaviors we should expect from any Option.

In the implementation of the member functions, Price() function is set to be a PVMF and the implementation is left to the derived classes, because the calculation can be different among option styles and polymorphism of this function will be useful for MatrixPricer later. It is also useful to overload Price() function so that it take an underlying asset price as input. There are two reasons for this function to be overloaded. First, since the underlying price usually keeps changing, it can be useful for the user to calculate option price based different underlying asset prices. Second, it will make the implementation of approximate Delta and Gamma easier, because it needs to calculate the option price based on a small change on the underlying asset price.

The approximate Delta and Gamma methods are implemented in the base class and the Template Method Pattern is used here. The reason is that for any derived classes, the approximation schemes will be the same, except the calculation of option price, which is delegated to a polymorphic function Price().

The functions for exact Delta and Gamma and ToString() are all PVMF, because the implementations can be different among option styles.

The Option class is an ABC.

The Euro Option class is derived from the Option class. One data member, T, is added. All the calculations based on Black-Scholes formula are implemented in this class and can be inherited to the derived more specific Euro option classes, so that the calculations do not need to be implemented again in the derived class. The Template Method Pattern is again used. Cost of carry is involved in many calculations, but the formula for cost of carry is different. Thus, a polymorphic function, CoC() is introduced and set as PVMF. The implementation of CoC() is left to derived classes. Euro Option class is also an abstract class.

Then, the Euro Stock Option, Euro Future Option and Euro Currency Option classes are all derived from the Euro Option class. In terms of data members, q is added to the Euro Stock Option class, R is added to the Euro Currency Option class. Most of their calculations are inherited from the Euro Option class and only CoC(), ToString() and Setter / Getter for newly added data members need to be implemented. At this stage, all of these derived classes are concrete classes. In the future, as long as the new option style can be calculated by Black-Scholes formula, it can be easily added and implemented, because almost all the calculations are already implemented.

On the other hand, the Perpetual American Option is derived from the Option base class and only b is added as data member. This enables the user to explicitly give cost of carry when inputting option parameters for Perpetual American options. Price() functions are implemented as per the exact formulas. Approximate Delta and Gamma functions are inherited from the Option class. Exact formula of Delta and Gamma are not given. In my implementation, Delta() and Gamma() functions just call the approximate Delta and Gamma instead, so that the Perpetual American Option class does not become an abstract class. In this way, Delta and Gamma functions can still be called later in MatrixPricer(), although only approximate values are returned for Perpetual American Option.

Utilities

Refer to files: Utilities.hpp&.cpp; OptionData.hpp

OptionData is a struct which contains all the possible parameters for any option and all the data members are public. This struct serves as the vehicle for the user to input option parameters. The user can create one OptionData object, or many of them as a vector of OptionData (“matrix” of options).

Functions in Utilities file will facilitate the calculations on options, especially the calculations on many different options at the same time.

Mesher() function can generate a vector of double (a mesh) with user defined start, end and step size. This is useful when the user wants to change certain option parameter monotonically to study how the option’s behavior reacts to the parameter change.

StyleSelect() is used when the user sets the option style in OptionData. A string input representing the option style is needed as the function argument. It is implemented through a map structure, which connects the string input with an int to represent a specific option style. In the future, if new option is added, we only need to update the map structure to include the command for new option.

MatrixFactory() is an overloaded function, whose job is to create a vector of smart pointers pointing to option objects. This is the final step to prepare for calculations on matrix of options. This vector will be passed to MatrixPricer() global function, where the price, Delta and Gamma will be calculated through polymorphism mechanism.

MatrixFactory() function has two versions to meet different needs.

Version#1 takes three arguments: one OptionData object, one mesh, and one string as inputs. This is used when the user wants to experiment option price or Greeks with certain monotonically changing parameter. The user can put the option parameters into an OptionData object, generate a mesh using Mesher() and then choose any parameter by typing its corresponding string. A vector of smart pointers is returned and ready to be used by MatrixPricer() global function.

Version#2 takes a vector of OptionData, which can be from the user input, as the argument. The user can follow the agreed order to type in a matrix of option parameters and pass it to MatrixFactory(). A vector of smart pointers to options is returned and ready to be used by MatrixPricer() global function.

MatrixPricer()

Refer to the global function in Test.cpp

MatrixPricer() is a global function, which takes the vector generated by MatrixFactory() and a command of string as inputs. The string will dictate what calculation to be performed. The calculation is performed through polymorphism, by which the pointers to base class can call functions of derived class. This way is the most flexible design and it will work on matrix containing any options derived from the Option base class.

Exception Handling

Refer to file: Exceptions.hpp&.cpp

In this program, several functions rely on string input from the user to perform the correct job. Although the user will be told the right commands to use, wrong input can happen easily and lead to problems. The following scenarios are possible.

1. When typing option parameters to OptionData, the user types an option style which is not implemented.
2. When using MatrixFactory(), the user might try to mesh a parameter which does not exist.
3. When using MatrixPricer(), the user might request a calculation which is not included.

Thus, an exception hierarchy is designed in this program. The InvalidInput base class has What() member function, which is a PVMF. This enables polymorphic catching nets for exceptions thrown. What() function will be implemented in derived classes to show corresponding error messages on the screen.

The InvalidStyle, InvalidParameter and InvalidOperaton classes are derived from InvalidInput class. A string data member is added to store the wrong string input from the user, which will be a part of the error message generated by What() function when the exception is caught

Answers to Questions:

Exact Solutions of One-Factor Plain Options

(a) See below screenshots for results of Batch 1~4. Refer Test.cpp output, Part 1, section 1.1.
Batch 1:

Microsoft Visual Studio Debug Console

```
Part 1: Test EuroStockOpt class. Batch 1~4. ********************************************

1.1 Test calculations for Batch 1~4.

Test Batch 1 Call Option:
Batch 1 Call Price: 2.13337
Batch 1 BS Delta: 0.372483
Batch 1 BS Gamma: 0.0420428
Batch 1 Approximate Delta (h=0.01): 0.372483
Batch 1 Approximate Gamma (h=0.01): 0.0420428
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0

Test Batch 1 Put Option:
Batch 1 Put Price: 5.84628
Batch 1 BS Delta: -0.627517
Batch 1 BS Gamma: 0.0420428
Batch 1 Approximate Delta (h=0.01): -0.627517
Batch 1 Approximate Gamma (h=0.01): 0.0420428
EuroStockOpt Put: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0
```

Batch 2:

```
Test Batch 2 Call Option:
Batch 2 Call Price: 7.96557
Batch 2 BS Delta: 0.539828
Batch 2 BS Gamma: 0.0198476
Batch 2 Approximate Delta (h=0.01): 0.539828
Batch 2 Approximate Gamma (h=0.01): 0.0198476
EuroStockOpt Call: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0

Test Batch 2 Put Option:
Batch 2 Put Price: 7.96557
Batch 2 BS Delta: -0.460172
Batch 2 BS Gamma: 0.0198476
Batch 2 Approximate Delta (h=0.01): -0.460172
Batch 2 Approximate Gamma (h=0.01): 0.0198476
EuroStockOpt Put: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0
```

Batch 3:

```
Test Batch 3 Call Option:
EuroStockOpt Call: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0
Batch 3 Call Price: 0.204058
Batch 3 BS Delta: 0.185048
Batch 3 BS Gamma: 0.106789
Batch 3 Approximate Delta (h=0.01): 0.185048
Batch 3 Approximate Gamma (h=0.01): 0.106789
EuroStockOpt Call: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0

Test Batch 3 Put Option:
Batch 3 Put Price: 4.07326
Batch 3 BS Delta: -0.814952
Batch 3 BS Gamma: 0.106789
Batch 3 Approximate Delta (h=0.01): -0.814952
Batch 3 Approximate Gamma (h=0.01): 0.106789
EuroStockOpt Put: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0
```

Batch 4:

```
Test Batch 4 Call Option:
EuroStockOpt Call: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0
Batch 4 Call Price: 92.1757
Batch 4 BS Delta: 0.988761
Batch 4 BS Gamma: 0.000179578
Batch 4 Approximate Delta (h=0.01): 0.988761
Batch 4 Approximate Gamma (h=0.01): 0.000179578
EuroStockOpt Call: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0

Test Batch 4 Put Option:
Batch 4 Put Price: 1.2475
Batch 4 BS Delta: -0.0112394
Batch 4 BS Gamma: 0.000179578
Batch 4 Approximate Delta (h=0.01): -0.0112394
Batch 4 Approximate Gamma (h=0.01): 0.000179578
EuroStockOpt Put: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0
```

(b) The put and call prices from part (a) are used to test Put-Call Parity. See the screenshots below. Refer to Test.cpp output, Part 1, section 1.2

```
1.2 Test Put-Call Parity related functions on Batch 1~4.

Batch 1 Parity Call price: 2.13337
Batch 1 Parity Put price: 5.84628
Put-Call Parity Match!

Batch 2 Parity Call price: 7.96557
Batch 2 Parity Put price: 7.96557
Put-Call Parity Match!

Batch 3 Parity Call price: 0.204058
Batch 3 Parity Put price: 4.07326
Put-Call Parity Match!

Batch 4 Parity Call price: 92.1757
Batch 4 Parity Put price: 1.2475
Put-Call Parity Match!
```

(c) Batch 1 Call and Put options are used in this question. S is ranging from 30 to 60. See the screenshots below. Refer to Test.cpp output, Part 2, section 2.1

Call:

Microsoft Visual Studio Debug Console

```
2.1 Test option price calculation using Batch_1 call and put option, let S change from 30 to 60:

EuroStockOpt Call: S = 30; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 3.11926e-07
EuroStockOpt Call: S = 31; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 1.00545e-06
EuroStockOpt Call: S = 32; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.99241e-06
EuroStockOpt Call: S = 33; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 8.27565e-06
EuroStockOpt Call: S = 34; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.13896e-05
EuroStockOpt Call: S = 35; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 5.19359e-05
EuroStockOpt Call: S = 36; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.000119024
EuroStockOpt Call: S = 37; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.000258548
EuroStockOpt Call: S = 38; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.000534403
EuroStockOpt Call: S = 39; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.00105474
EuroStockOpt Call: S = 40; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.00199417
EuroStockOpt Call: S = 41; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.00362244
EuroStockOpt Call: S = 42; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.00633924
EuroStockOpt Call: S = 43; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.010714
EuroStockOpt Call: S = 44; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.0175282
EuroStockOpt Call: S = 45; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.0278174
EuroStockOpt Call: S = 46; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.0429083
EuroStockOpt Call: S = 47; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.0644467
EuroStockOpt Call: S = 48; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.0944125
EuroStockOpt Call: S = 49; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.135117
EuroStockOpt Call: S = 50; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.189181
EuroStockOpt Call: S = 51; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.259494
EuroStockOpt Call: S = 52; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.349151
EuroStockOpt Call: S = 53; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.461376
EuroStockOpt Call: S = 54; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.599432
EuroStockOpt Call: S = 55; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.76652
EuroStockOpt Call: S = 56; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 0.965684
EuroStockOpt Call: S = 57; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 1.19971
EuroStockOpt Call: S = 58; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 1.47106
EuroStockOpt Call: S = 59; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 1.78175
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.13337
```

Put:

```
EuroStockOpt Put: S = 30; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 33.7129
EuroStockOpt Put: S = 31; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 32.7129
EuroStockOpt Put: S = 32; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 31.7129
EuroStockOpt Put: S = 33; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 30.7129
EuroStockOpt Put: S = 34; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 29.7129
EuroStockOpt Put: S = 35; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 28.713
EuroStockOpt Put: S = 36; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 27.713
EuroStockOpt Put: S = 37; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 26.7132
EuroStockOpt Put: S = 38; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 25.7134
EuroStockOpt Put: S = 39; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 24.714
EuroStockOpt Put: S = 40; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 23.7149
EuroStockOpt Put: S = 41; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 22.7165
EuroStockOpt Put: S = 42; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 21.7193
EuroStockOpt Put: S = 43; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 20.7236
EuroStockOpt Put: S = 44; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 19.7304
EuroStockOpt Put: S = 45; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 18.7407
EuroStockOpt Put: S = 46; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 17.7558
EuroStockOpt Put: S = 47; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 16.7774
EuroStockOpt Put: S = 48; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 15.8073
EuroStockOpt Put: S = 49; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 14.848
EuroStockOpt Put: S = 50; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 13.9021
EuroStockOpt Put: S = 51; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 12.9724
EuroStockOpt Put: S = 52; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 12.0621
EuroStockOpt Put: S = 53; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 11.1743
EuroStockOpt Put: S = 54; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 10.3123
EuroStockOpt Put: S = 55; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 9.47943
EuroStockOpt Put: S = 56; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 8.6786
EuroStockOpt Put: S = 57; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 7.91263
EuroStockOpt Put: S = 58; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 7.18397
EuroStockOpt Put: S = 59; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 6.49466
EuroStockOpt Put: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 5.84628
```

(d) Still use Batch 1 Call parameters and let K change from 40 to 65. See the screenshot below. Refer to Test.cpp output, Part 2, section 2.4

```
2.4 Test MatrixPricer() function. Let any parameter change. Let K of Batch_1 change from 40 to 65:

EuroStockOpt Call: S = 60; K = 40; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 20.7969
EuroStockOpt Call: S = 60; K = 41; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 19.8204
EuroStockOpt Call: S = 60; K = 42; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 18.8461
EuroStockOpt Call: S = 60; K = 43; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 17.8749
EuroStockOpt Call: S = 60; K = 44; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 16.9084
EuroStockOpt Call: S = 60; K = 45; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 15.948
EuroStockOpt Call: S = 60; K = 46; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 14.9959
EuroStockOpt Call: S = 60; K = 47; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 14.0544
EuroStockOpt Call: S = 60; K = 48; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 13.1264
EuroStockOpt Call: S = 60; K = 49; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 12.2148
EuroStockOpt Call: S = 60; K = 50; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 11.3229
EuroStockOpt Call: S = 60; K = 51; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 10.4543
EuroStockOpt Call: S = 60; K = 52; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 9.6122
EuroStockOpt Call: S = 60; K = 53; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 8.80017
EuroStockOpt Call: S = 60; K = 54; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 8.02135
EuroStockOpt Call: S = 60; K = 55; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 7.27865
EuroStockOpt Call: S = 60; K = 56; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 6.57456
EuroStockOpt Call: S = 60; K = 57; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 5.91114
EuroStockOpt Call: S = 60; K = 58; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 5.28988
EuroStockOpt Call: S = 60; K = 59; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 4.71174
EuroStockOpt Call: S = 60; K = 60; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 4.1771
EuroStockOpt Call: S = 60; K = 61; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 3.6858
EuroStockOpt Call: S = 60; K = 62; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 3.23714
EuroStockOpt Call: S = 60; K = 63; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.82995
EuroStockOpt Call: S = 60; K = 64; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.46266
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.13337
```

Option Sensitivities, aka the Greeks

(a) See the screenshot below for the calculation of this future option. Refer to Test.cpp output, Part 3, section 3.1 and 3.2

```
Part 3: Test Euro Future Option class.*********************************************************************

3.1 Test Future Option Call Option:
Future Option Call Price: 12.4328
Future Call Option BS Delta: 0.594629
Future Call Option BS Gamma: 0.0134936
Future Call Approximate Delta (h=0.01): 0.594629
Future Call Approximate Gamma (h=0.01): 0.0134936
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36

3.2 Test Future Option Put Option:
Future Option Put Price: 7.6767
Future Put Option BS Delta: -0.356601
Future Put Option BS Gamma: 0.0134936
Future Put Approximate Delta (h=0.01): -0.356601
Future Put Approximate Gamma (h=0.01): 0.0134936
EuroFutureOpt Put: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36
```

(b) The future call option is used. Change the underlying price, S, from 80 to 105. Both Delta and Gamma are calculated. Refer to Test.cpp output, Part 3, section 3.3

Delta:

```
3.3 Test MatrixPricer(). Compute Call Delta and Gamma for range of S from 80 to 105 for the above Euro Future Option:
EuroFutureOpt Call: S = 80; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.215772
EuroFutureOpt Call: S = 81; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.230011
EuroFutureOpt Call: S = 82; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.244562
EuroFutureOpt Call: S = 83; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.259392
EuroFutureOpt Call: S = 84; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.274471
EuroFutureOpt Call: S = 85; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.289765
EuroFutureOpt Call: S = 86; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.305241
EuroFutureOpt Call: S = 87; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.320867
EuroFutureOpt Call: S = 88; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.33661
EuroFutureOpt Call: S = 89; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.352438
EuroFutureOpt Call: S = 90; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.368319
EuroFutureOpt Call: S = 91; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.384223
EuroFutureOpt Call: S = 92; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.400118
EuroFutureOpt Call: S = 93; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.415977
EuroFutureOpt Call: S = 94; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.431772
EuroFutureOpt Call: S = 95; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.447475
EuroFutureOpt Call: S = 96; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.463062
EuroFutureOpt Call: S = 97; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.478508
EuroFutureOpt Call: S = 98; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.493791
EuroFutureOpt Call: S = 99; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.50889
EuroFutureOpt Call: S = 100; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.523785
EuroFutureOpt Call: S = 101; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.538459
EuroFutureOpt Call: S = 102; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.552894
EuroFutureOpt Call: S = 103; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.567076
EuroFutureOpt Call: S = 104; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.580992
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.594629
```

Gamma:

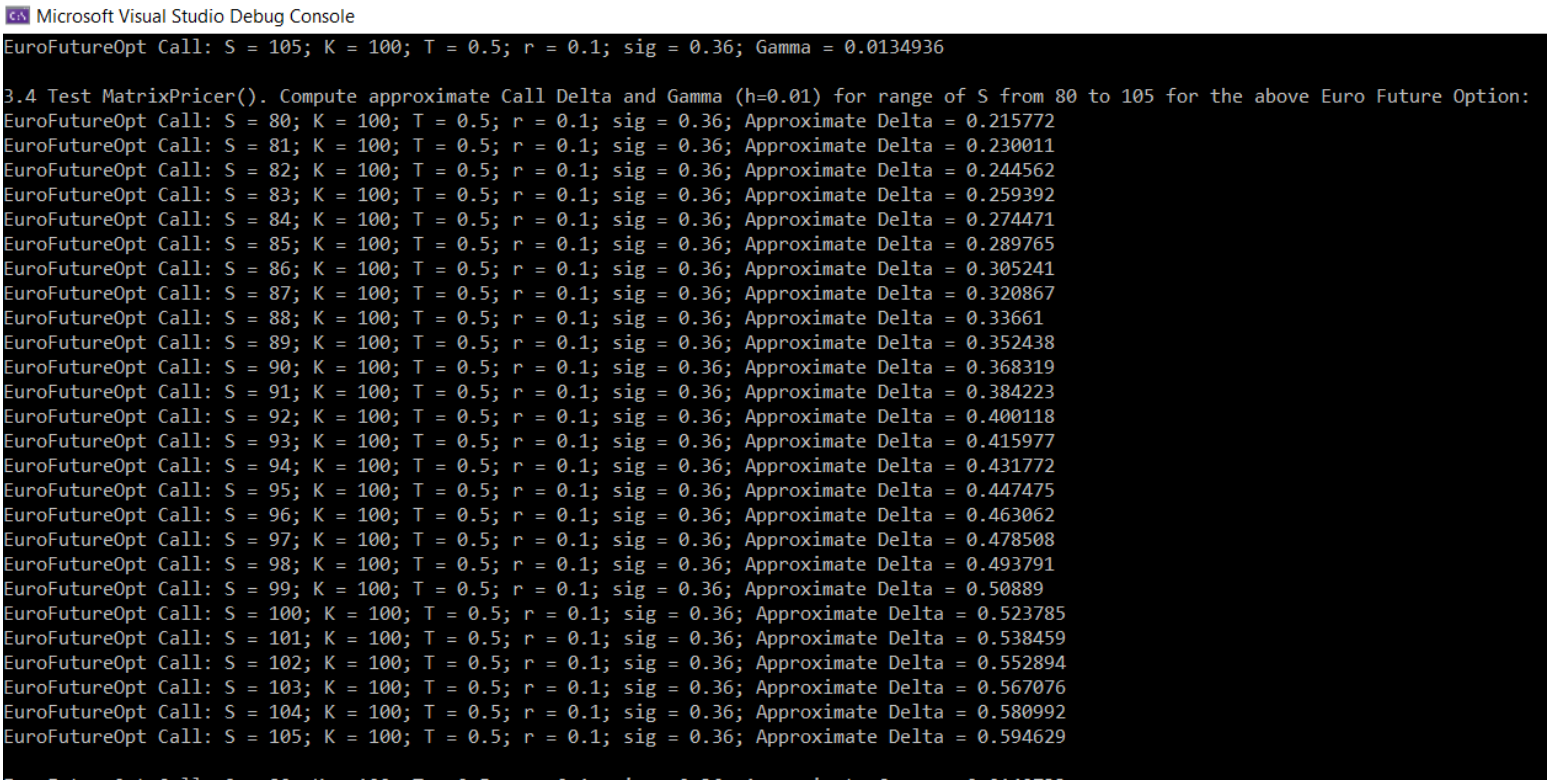
```
EuroFutureOpt Call: S = 80; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0140733
EuroFutureOpt Call: S = 81; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0144001
EuroFutureOpt Call: S = 82; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0146959
EuroFutureOpt Call: S = 83; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0149599
EuroFutureOpt Call: S = 84; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0151916
EuroFutureOpt Call: S = 85; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0153905
EuroFutureOpt Call: S = 86; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0155566
EuroFutureOpt Call: S = 87; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0156899
EuroFutureOpt Call: S = 88; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0157908
EuroFutureOpt Call: S = 89; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0158598
EuroFutureOpt Call: S = 90; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0158974
EuroFutureOpt Call: S = 91; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0159046
EuroFutureOpt Call: S = 92; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0158822
EuroFutureOpt Call: S = 93; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0158313
EuroFutureOpt Call: S = 94; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0157532
EuroFutureOpt Call: S = 95; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0156491
EuroFutureOpt Call: S = 96; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0155203
EuroFutureOpt Call: S = 97; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0153682
EuroFutureOpt Call: S = 98; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0151944
EuroFutureOpt Call: S = 99; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0150003
EuroFutureOpt Call: S = 100; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0147873
EuroFutureOpt Call: S = 101; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0145571
EuroFutureOpt Call: S = 102; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0143112
EuroFutureOpt Call: S = 103; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.014051
EuroFutureOpt Call: S = 104; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.013778
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0134936
```

(c) See the screenshots from (b).

(d) For calculating the approximate Delta and Gamma for this future option, please refer to the screenshot in (a).

The future call option is used again. Change the underlying price, S, from 80 to 105. Both approximate Delta and Gamma are calculated using h = 0.01. Refer to Test.cpp output, Part 3, section 3.4

Approximate Delta:

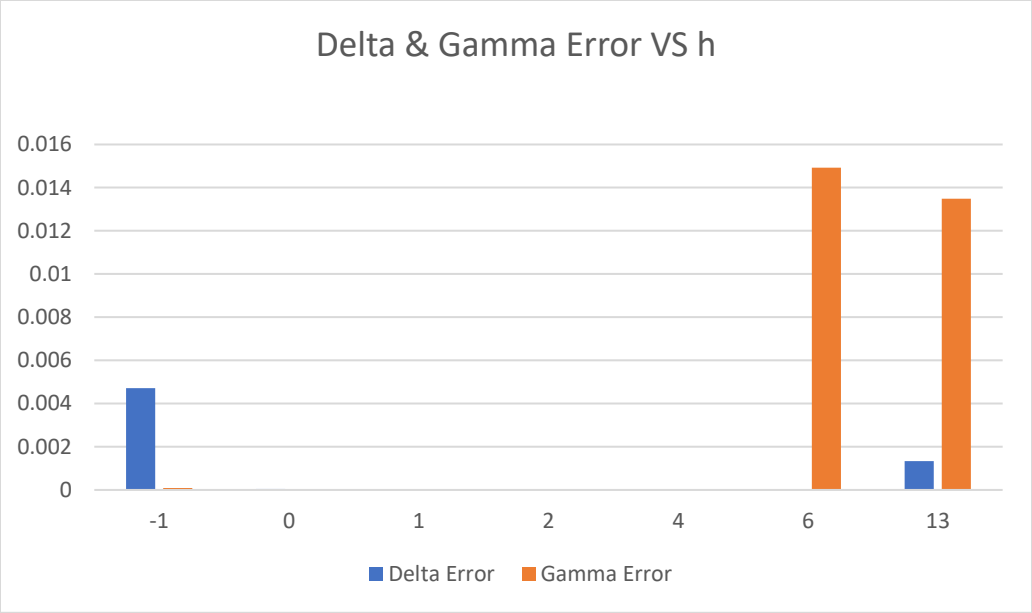


Approximate Gamma:

```
EuroFutureOpt Call: S = 80; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0140733
EuroFutureOpt Call: S = 81; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0144001
EuroFutureOpt Call: S = 82; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0146959
EuroFutureOpt Call: S = 83; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0149599
EuroFutureOpt Call: S = 84; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0151916
EuroFutureOpt Call: S = 85; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0153905
EuroFutureOpt Call: S = 86; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0155566
EuroFutureOpt Call: S = 87; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0156899
EuroFutureOpt Call: S = 88; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0157908
EuroFutureOpt Call: S = 89; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0158598
EuroFutureOpt Call: S = 90; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0158974
EuroFutureOpt Call: S = 91; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0159046
EuroFutureOpt Call: S = 92; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0158822
EuroFutureOpt Call: S = 93; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0158313
EuroFutureOpt Call: S = 94; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0157532
EuroFutureOpt Call: S = 95; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0156491
EuroFutureOpt Call: S = 96; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0155203
EuroFutureOpt Call: S = 97; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0153682
EuroFutureOpt Call: S = 98; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0151944
EuroFutureOpt Call: S = 99; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0150003
EuroFutureOpt Call: S = 100; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0147873
EuroFutureOpt Call: S = 101; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0145571
EuroFutureOpt Call: S = 102; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0143112
EuroFutureOpt Call: S = 103; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.014051
EuroFutureOpt Call: S = 104; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.013778
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Approximate Gamma = 0.0134936
```

In order to find the proper h value to get the desired accuracy, this future put option is used to experiment with different value of h. The error is calculated as the absolute value of the difference between the approximate Delta & Gamma and the exact Delta & Gamma. The result is recorded in the worksheet below. The error of approximate Delta and Gamma are plotted against the -log 10 of h. Refer to Test.cpp output, Part 5.

h	"- log 10 h"	Appx Delta	Exact Delta	Delta Error	Appx Gamma	Exact Gamma	Gamma Error
1.E+01	-1	-0.361311	-0.356601	0.00471	0.0134098	0.0134936	8.38E-05
1.E+00	0	-0.356649	-0.356601	4.8E-05	0.0134928	0.0134936	8E-07
1.E-01	1	-0.356601	-0.356601	0	0.0134936	0.0134936	0
1.E-02	2	-0.356601	-0.356601	0	0.0134936	0.0134936	0
1.E-04	4	-0.356601	-0.356601	0	0.0134932	0.0134936	4E-07
1.E-06	6	-0.356601	-0.356601	0	0.0284217	0.0134936	0.0149281
1.E-13	13	-0.355271	-0.356601	0.00133	0	0.0134936	0.0134936



It is observed that Delta error is close to 0 when h is from 0.1 to 0.000001 and then starts increasing when h is smaller. Due to the fact that Gamma is second order approximation, it is more sensitive to h and when h = 0.0001, the error starts increasing. Thus, h = 0.01 is considered a proper scale for approximate Delta & Gamma to get decent accuracy compared to the exact solutions. Therefore, h = 0.01 is hard-coded into this program when approximate Delta & Gamma are needed.

Perpetual American Options

(a) The formula is implemented as member functions of the Perpetual American Option class.

(b) Refer to Test.cpp output, Part 4, section 4.1 and 4.2

```
Part 4: Test Perpetual American Option class.*********************************************

4.1 Test Perpetual American Option Call Option:
Perpetual American Call Price: 18.5035
Perpetual American Call Approximate Delta (h=0.01): 0.541142
Perpetual American Call Approximate Gamma (h=0.01): 0.0109064
PerpAmOpt Call: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02

4.2 Test Perpetual American Option Put Option:
Perpetual American Option Put Price: 3.03106
Perpetual American Put Approximate Delta (h=0.01): -0.17131
Perpetual American Put Approximate Gamma (h=0.01): 0.0112395
PerpAmOpt Put: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02
```

(c) The Perpetual American Option in HW is used, in which the underlying price, S, is changing from 90 to 110. Both Call and Put prices are calculated. Refer to Test.cpp output, Part 4, section 4.3

Call and Put:

```
4.3 Test MatrixPricer(). Compute Call and Put price for range of S from 90 to 110 for the above Perp American Option:
PerpAmOpt Call: S = 90; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 9.7027
PerpAmOpt Call: S = 91; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 10.0538
PerpAmOpt Call: S = 92; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 10.4136
PerpAmOpt Call: S = 93; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 10.7821
PerpAmOpt Call: S = 94; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 11.1595
PerpAmOpt Call: S = 95; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 11.546
PerpAmOpt Call: S = 96; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 11.9416
PerpAmOpt Call: S = 97; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 12.3464
PerpAmOpt Call: S = 98; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 12.7605
PerpAmOpt Call: S = 99; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 13.1842
PerpAmOpt Call: S = 100; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 13.6174
PerpAmOpt Call: S = 101; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 14.0603
PerpAmOpt Call: S = 102; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 14.5131
PerpAmOpt Call: S = 103; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 14.9758
PerpAmOpt Call: S = 104; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 15.4486
PerpAmOpt Call: S = 105; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 15.9316
PerpAmOpt Call: S = 106; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 16.4249
PerpAmOpt Call: S = 107; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 16.9286
PerpAmOpt Call: S = 108; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 17.4429
PerpAmOpt Call: S = 109; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 17.9678
PerpAmOpt Call: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 18.5035

PerpAmOpt Put: S = 90; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 10.5537
PerpAmOpt Put: S = 91; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 9.85308
PerpAmOpt Put: S = 92; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 9.20584
PerpAmOpt Put: S = 93; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 8.60743
PerpAmOpt Put: S = 94; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 8.05372
PerpAmOpt Put: S = 95; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 7.54093
PerpAmOpt Put: S = 96; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 7.06565
PerpAmOpt Put: S = 97; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 6.6248
PerpAmOpt Put: S = 98; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 6.21556
PerpAmOpt Put: S = 99; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 5.83537
PerpAmOpt Put: S = 100; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 5.48192
PerpAmOpt Put: S = 101; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 5.15308
PerpAmOpt Put: S = 102; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 4.84691
PerpAmOpt Put: S = 103; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 4.56167
PerpAmOpt Put: S = 104; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 4.29572
PerpAmOpt Put: S = 105; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 4.04761
PerpAmOpt Put: S = 106; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.81598
PerpAmOpt Put: S = 107; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.5996
PerpAmOpt Put: S = 108; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.39733
PerpAmOpt Put: S = 109; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.20813
PerpAmOpt Put: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.03106
```

(d) All options in this HW are typed into one matrix. This matrix is passed to MatrixPricer() function to calculate Price, Delta and Gamma for all of them. Refer to Test.cpp output, Part 6.

```
Calculate Price:
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 2.13337
EuroStockOpt Put: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Price = 5.84628
EuroStockOpt Call: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Price = 7.96557
EuroStockOpt Put: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Price = 7.96557
EuroStockOpt Call: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Price = 0.204058
EuroStockOpt Put: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Price = 4.07326
EuroStockOpt Call: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Price = 92.1757
EuroStockOpt Put: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Price = 1.2475
PerpAmOpt Call: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 18.5035
PerpAmOpt Put: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Price = 3.03106
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Price = 12.4328
EuroFutureOpt Put: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Price = 7.6767
```

```
Calculate Delta:
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Delta = 0.372483
EuroStockOpt Put: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Delta = -0.627517
EuroStockOpt Call: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Delta = 0.539828
EuroStockOpt Put: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Delta = -0.460172
EuroStockOpt Call: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Delta = 0.185048
EuroStockOpt Put: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Delta = -0.814952
EuroStockOpt Call: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Delta = 0.988761
EuroStockOpt Put: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Delta = -0.0112394
PerpAmOpt Call: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Delta = 0.541142
PerpAmOpt Put: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Delta = -0.17131
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = 0.594629
EuroFutureOpt Put: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Delta = -0.356601
```

```
Calculate Gamma:
EuroStockOpt Call: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Gamma = 0.0420428
EuroStockOpt Put: S = 60; K = 65; T = 0.25; r = 0.08; sig = 0.3; q = 0; Gamma = 0.0420428
EuroStockOpt Call: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Gamma = 0.0198476
EuroStockOpt Put: S = 100; K = 100; T = 1; r = 0; sig = 0.2; q = 0; Gamma = 0.0198476
EuroStockOpt Call: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Gamma = 0.106789
EuroStockOpt Put: S = 5; K = 10; T = 1; r = 0.12; sig = 0.5; q = 0; Gamma = 0.106789
EuroStockOpt Call: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Gamma = 0.000179578
EuroStockOpt Put: S = 100; K = 100; T = 30; r = 0.08; sig = 0.3; q = 0; Gamma = 0.000179578
PerpAmOpt Call: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Gamma = 0.0109064
PerpAmOpt Put: S = 110; K = 100; r = 0.1; sig = 0.1; b = 0.02; Gamma = 0.0112395
EuroFutureOpt Call: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0134936
EuroFutureOpt Put: S = 105; K = 100; T = 0.5; r = 0.1; sig = 0.36; Gamma = 0.0134936
```