Contents

1 Introduction to the Book . . . 7

1.1 Introduction to QuantStart . . . 7

1.2 What is this book? . . . 7

1.3 Who is this book for? . . . 7

1.4 What are the prerequisites? . . . 8

1.5 Software Requirements . . . 8

1.6 Book Structure . . . 9

1.7 What the book does not cover . . . 10

1.8 Where to get help . . . 10

2 Introduction to C++ . . . 11

2.1 A Brief History Of C++ . . . 11

2.2 The Advantages Of C++ . . . 11

2.3 The Disadvantages Of C++ . . . 12

2.4 The Different Components Of C++ . . . 12

2.5 C-Style Procedural Programming . . . 13

2.6 Object-Oriented Programming . . . 13

2.7 Generic Programming . . . 14

2.8 The Standard Template Library (STL) . . . 14

2.9 The Boost Libraries . . . 15

2.10 C++ In Quantitative Analysis . . . 15

2.11 C++ In This Course . . . 16

3 Your First Quantitative Finance C++ Program . . . 17

3.1 Components Of An Object-Oriented C++ Program . . . 17

3.2 What Are Classes? . . . 18

3.3 Constructors And Destructors . . . 18

3.4 Selectors And Modifiers . . . 19

3.5 European Vanilla Option Specification 20
3.6 VanillaOption Header File 21
3.7 VanillaOption Source File 24
3.8 Passing By Reference Or Value 29

4 Option Pay-Off Hierarchies and Inheritance 31
4.1 Inheritance In Quantitative Finance 31
4.2 Option Pay-off Hierarchies 32
4.3 Implenting The PayOff Base Class 33
4.4 PayOff Header File 33
4.5 PayOffCall Header 34
4.6 PayOffDoubleDigital Header 35
4.7 PayOffCall Source File 36
4.8 PayOffDoubleDigital Source File 36
4.9 Virtual Destructors 37

5 Generic Programming and Template Classes 41
5.1 Why Generic Programming? 41
5.2 Template Classes 41
5.3 A Matrix Template Class 42
5.4 Template Syntax and Declaration 42
5.5 Default Types in Templates 43
5.6 SimpleMatrix Declaration 44
5.7 SimpleMatrix Implementation 47
5.8 Generic Programming vs OOP 51

6 Introduction to the Standard Template Library 53
6.1 Components of the STL 53
6.2 Containers 54
6.2.1 Sequence Containers 54
6.2.2 Associative Containers 55
6.2.3 Container Adaptors 56
6.3 Iterators 56
6.3.1 Iterator Categories 56
6.3.2 Iterator Adaptors 57

6.3.3 Const Iterators 58
6.3.4 Iterator Traits 59
6.4 Algorithms 59
6.4.1 Algorithm Categories 59
6.4.2 Nonmodifying Algorithms 60
6.4.3 Modifying Algorithms 61
6.4.4 Removal Algorithms 62
6.4.5 Mutating Algorithms 63
6.4.6 Sorting Algorithms 63
6.4.7 Sorted Range Algorithms 64
6.4.8 Numeric Algorithms 65
6.5 C++11 STL 66
6.5.1 Containers 66
6.5.2 Algorithms 66

7 Function Objects 69
7.1 Function Pointers 70
7.2 C++ Function Objects (Functors) 72

8 Matrix Classes for Quantitative Finance 75
8.1 Custom Matrix Class Library 76
8.1.1 C++ STL Storage Mechanisms 76
8.1.2 Matrix Mathematical Operations 77
8.1.3 Full Declaration 79
8.1.4 The Source File 81
8.1.5 Allocation and Deallocation 81
8.1.6 Mathematical Operators Implementation 84
8.1.7 Full Source Implementation 89
8.1.8 Using the Matrix Class 96
8.2 External Matrix Libraries 97
8.2.1 Installation 98
8.2.2 Basic Usage 98
8.2.3 Basic Linear Algebra 99
8.2.4 Expression Templates 99
8.2.5 Matrix and Scalar Arithmetic 100

8.2.6 Matrix Transposition . . . 102
8.2.7 Matrix/Matrix and Matrix/Vector Multiplication . . . 103
8.2.8 Vector Operations . . . 104
8.2.9 Reduction . . . 105
8.2.10 Useful Features . . . 106
8.2.11 Next Steps . . . 106

9 Numerical Linear Algebra . . . 107
9.1 Overview . . . 107
9.2 LU Decomposition . . . 108
9.2.1 Algorithm . . . 108
9.2.2 Eigen C++ Implementation . . . 109
9.3 Thomas Tridiagonal Matrix Algorithm . . . 111
9.3.1 C++ Implementation . . . 112
9.4 Cholesky Decomposition . . . 116
9.4.1 Algorithm . . . 117
9.4.2 Eigen Implementation . . . 117
9.5 QR Decomposition . . . 119
9.5.1 Algorithm . . . 119
9.5.2 Eigen Implementation . . . 121

10 European Options with Monte Carlo . . . 123
10.1 Black-Scholes Analytic Pricing Formula . . . 123
10.2 C++ Implementation . . . 124
10.3 Risk Neutral Pricing of a European Vanilla Option . . . 130
10.4 C++ Implementation . . . 132

11 Calculating the "Greeks" . . . 139
11.1 Analytic Formulae . . . 140
11.2 Finite Difference Method . . . 146
11.3 Monte Carlo Method . . . 149

12 Asian/Path-Dependent Options with Monte Carlo . . . 155
12.1 Overview of Asian Options . . . 155
12.2 Asian Options - An Object Oriented Approach . . . 156
12.3 Pay-Off Classes . . . 158

12.4 Path Generation Header 161

12.5 Asian Option Classes 164

12.6 The Main Program 168

13 Implied Volatility 173

13.1 Motivation 173

13.2 Root-Finding Algorithms 174

13.3 Interval Bisection Method 174

13.3.1 Interval Bisection with Function Templates 175

13.3.2 C++ Implementation 175

13.4 Implied Volatility via Newton-Raphson 181

13.4.1 Newton-Raphson Method 181

13.4.2 Pointer to a Member Function 182

13.4.3 Implementing Newton-Raphson 183

14 Random Number Generation and Statistical Distributions 189

14.1 Overview 189

14.2 Random Number Generator Class Hierarchy 190

14.2.1 Linear Congruential Generators 193

14.2.2 Implementing a Linear Congruential Generator 194

14.2.3 Implementation of the Main Program 198

14.3 Statistical Distributions 199

14.3.1 Statistical Distribution Inheritance Hierarchy 200

14.3.2 Standard Normal Distribution Implementation 201

14.3.3 The Main Listing 206

15 Jump-Diffusion Models 209

15.1 Modelling Jump-Diffusion Processes 209

15.2 Prices of European Options under Jump-Diffusions 210

15.3 C++ Implementation 211

16 Stochastic Volatility 215

16.1 Motivation 215

16.2 Process for Correlated Path Generation 216

16.3 Cholesky Decomposition 217

16.4 C++ Implementation 217

16.5 Mathematical Model 223
16.6 Euler Discretisation 224
16.6.1 Correlated Asset Paths 226
16.6.2 Monte Carlo Algorithm 226
16.7 C++ Implementation 227
16.7.1 PayOff Class 227
16.7.2 Option Class 227
16.7.3 Statistics and CorrelatedSND Classes 229
16.7.4 HestonEuler Class 229
16.7.5 Main Program 231

17 Single Factor Black-Scholes with Finite Difference Methods 237
17.1 Black-Scholes PDE for a European Call Option 238
17.2 Finite Difference Discretisation 239
17.3 Implementation 240
17.3.1 PayOff Class 241
17.3.2 VanillaOption Class 243
17.3.3 PDE Classes 245
17.3.4 FDM Class 248
17.3.5 Main Implementation 256
17.4 Execution and Output 258