Options Lookback for Monte Carlo method

1.0

Generated by Doxygen 1.15.0

Chapter 1

Lookback Options Pricing Library

1.1 Abstract

This project implements a C++ dynamic-link library (DLL) for the pricing of lookback options using Monte Carlo simulation. Both call and put lookback options are supported, together with the computation of the main Greeks (Delta, Gamma, Theta, Rho, Vega) via finite difference methods.

The library is designed to be callable from external environments such as VBA, allowing flexible integration in financial applications. Particular attention is given to numerical stability, reproducibility through random seeds, and clear separation between pricing logic and interface code.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 GreeksParams Struct Reference

`#include <LookbackOption.h>`

Public Attributes

- double hS
- double hg
- double dt
- double hR
- double hV

4.1.1 Detailed Description

Definition at line 27 of file LookbackOption.h.

4.1.2 Member Data Documentation

4.1.2.1 dt

`double GreeksParams::dt`

Definition at line 30 of file LookbackOption.h.

4.1.2.2 hg

`double GreeksParams::hg`

Definition at line 29 of file LookbackOption.h.

4.1.2.3 hR

```
double GreeksParams::hR
```

Definition at line 31 of file LookbackOption.h.

4.1.2.4 hS

```
double GreeksParams::hS
```

Definition at line 28 of file LookbackOption.h.

4.1.2.5 hV

```
double GreeksParams::hV
```

Definition at line 32 of file LookbackOption.h.

The documentation for this struct was generated from the following file:

- LookbackOption.h

4.2 LookbackOption Class Reference

```
#include <LookbackOption.h>
```

Public Member Functions

- double price_lookback (const LookbackParams &p) const
- double delta_lookback (const LookbackParams &p, const GreeksParams &g) const
- double gamma_lookback (const LookbackParams &p, const GreeksParams &g) const
- double theta_lookback (const LookbackParams &p, const GreeksParams &g) const
- double rho_lookback (const LookbackParams &p, const GreeksParams &g) const
- double vega_lookback (const LookbackParams &p, const GreeksParams &g) const

4.2.1 Detailed Description

Definition at line 35 of file LookbackOption.h.

4.2.2 Member Function Documentation

4.2.2.1 delta_lookback()

```
double LookbackOption::delta_lookback (
            const LookbackParams & p,
            const GreeksParams & g) const
```

Definition at line 41 of file LookbackOption.cpp.

4.2.2.2 gamma_lookback()

```
double LookbackOption::gamma_lookback (
            const LookbackParams & p,
            const GreeksParams & g) const
```

Definition at line 55 of file LookbackOption.cpp.

4.2.2.3 price_lookback()

```
double LookbackOption::price_lookback (
            const LookbackParams & p) const
```

Definition at line 26 of file LookbackOption.cpp.

4.2.2.4 rho_lookback()

```
double LookbackOption::rho_lookback (
            const LookbackParams & p,
            const GreeksParams & g) const
```

Definition at line 83 of file LookbackOption.cpp.

4.2.2.5 theta_lookback()

```
double LookbackOption::theta_lookback (
            const LookbackParams & p,
            const GreeksParams & g) const
```

Definition at line 70 of file LookbackOption.cpp.

4.2.2.6 vega_lookback()

```
double LookbackOption::vega_lookback (
            const LookbackParams & p,
            const GreeksParams & g) const
```

Definition at line 97 of file LookbackOption.cpp.

The documentation for this class was generated from the following files:

- LookbackOption.h
- LookbackOption.cpp

4.3 LookbackParams Struct Reference

```
#include <LookbackOption.h>
```

Public Attributes

- Option type
- double S0
- double r
- double sigma
- double T
- int steps
- int paths
- unsigned int seed

4.3.1 Detailed Description

Definition at line 14 of file LookbackOption.h.

4.3.2 Member Data Documentation

4.3.2.1 paths

```
int LookbackParams::paths
```

Definition at line 23 of file LookbackOption.h.

4.3.2.2 r

```
double LookbackParams::r
```

Definition at line 17 of file LookbackOption.h.

4.3.2.3 S0

```
double LookbackParams::S0
```

Definition at line 16 of file LookbackOption.h.

4.3.2.4 seed

```
unsigned int LookbackParams::seed
```

Definition at line 24 of file LookbackOption.h.

4.3.2.5 sigma

```
double LookbackParams::sigma
```

Definition at line 18 of file LookbackOption.h.

4.3.2.6 steps

```
int LookbackParams::steps
```

Definition at line 22 of file LookbackOption.h.

4.3.2.7 T

```
double LookbackParams::T
```

Definition at line 19 of file LookbackOption.h.

4.3.2.8 type

```
Option LookbackParams::type
```

Definition at line 15 of file LookbackOption.h.

The documentation for this struct was generated from the following file:

- LookbackOption.h

Chapter 5

File Documentation

5.1 LookbackOption.cpp File Reference

```
#include "LookbackOption.h"
```

5.2 LookbackOption.cpp

Go to the documentation of this file.

```
00001 #include "LookbackOption.h"
00002
00003
00004 //Function to simulate a single path of the option lookback
00005 double LookbackOption::simulate_lookback(Option type, double S0, double r, double sigma, double T, int
      steps, std::mt19937& gen, std::normal_distribution<>& Z) const{
00006
00007     double dt = T / steps;
00008     double S = S0;
00009     double minS = S0;
00010     double maxS = S0;
00011
00012     for (int i = 0; i < steps; i++) {
00013         double z = Z(gen);
00014         S *= std::exp((r - 0.5 * sigma * sigma) * dt + sigma * std::sqrt(dt) * z);
00015         minS = std::min(minS, S);
00016         maxS = std::max(maxS, S);
00017     }
00018     if (type == Option::Call)
00019         return S - minS;  //payoff call lookback
00020     else
00021         return maxS - S;  //payoff put lookback
00022 }
00023
00024
00025 //Price simulation with Monte Carlo
00026 double LookbackOption::price_lookback(const LookbackParams& p) const{
00027
00028     std::mt19937 gen(p.seed);
00029     std::normal_distribution<> Z(0.0, 1.0);
00030
00031     double sum = 0.0;
00032     for (int i = 0; i < p.paths; i++)
00033         sum += simulate_lookback(p.type, p.S0, p.r, p.sigma, p.T, p.steps, gen, Z);
00034
00035     return std::exp(-p.r * p.T) * (sum / p.paths);
00036 }
00037
00038 //Greeks
00039
00040 //Delta of the option
00041 double LookbackOption::delta_lookback(const LookbackParams& p, const GreeksParams& g) const{
00042
```

```
00043     LookbackParams p_plus = p;
00044     LookbackParams p_minus = p;
00045     p_plus.S0 += g.hS;
00046     p_minus.S0 -= g.hS;
00047 
00048     const double P_plus = price_lookback(p_plus);
00049     const double P_minus = price_lookback(p_minus);
00050 
00051     return (P_plus - P_minus) / (2.0 * g.hS);
00052 }
00053 
00054 //Gamma of the option
00055 double LookbackOption::gamma_lookback(const LookbackParams& p, const GreeksParams& g) const{
00056 
00057     LookbackParams p_plus = p;
00058     LookbackParams p_minus = p;
00059     p_plus.S0 += g.hg;
00060     p_minus.S0 -= g.hg;
00061 
00062     const double P_plus = price_lookback(p_plus);
00063     const double P = price_lookback(p);
00064     const double P_minus = price_lookback(p_minus);
00065 
00066     return (P_plus - 2.0 * P + P_minus) / (g.hg * g.hg);
00067 }
00068 
00069 //Theta of the option
00070 double  LookbackOption::theta_lookback(const LookbackParams& p, const GreeksParams& g) const{
00071 
00072     LookbackParams p_today = p;
00073     LookbackParams p_future = p;
00074     p_future.T = p.T - g.dt;
00075 
00076     const double P_today = price_lookback(p_today);
00077     const double P_future = price_lookback(p_future);
00078 
00079     return (P_future - P_today) / g.dt; //negative if the price decreses wrt time
00080 }
00081 
00082 //Rho of the option
00083 double  LookbackOption::rho_lookback(const LookbackParams& p, const GreeksParams& g) const{
00084 
00085     LookbackParams p_plus = p;
00086     LookbackParams p_minus = p;
00087     p_plus.r += g.hR;
00088     p_minus.r -= g.hR;
00089 
00090     double P_plus = price_lookback(p_plus);
00091     double P_minus = price_lookback(p_minus);
00092 
00093     return (P_plus - P_minus) / (2.0 * g.hR);
00094 }
00095 
00096 //Vega of the option
00097 double  LookbackOption::vega_lookback(const LookbackParams& p, const GreeksParams& g) const{
00098 
00099     LookbackParams p_plus = p;
00100     LookbackParams p_minus = p;
00101     p_plus.sigma += g.hV;
00102     p_minus.sigma -= g.hV;
00103 
00104     double P_plus = price_lookback(p_plus);
00105     double P_minus = price_lookback(p_minus);
00106 
00107     return (P_plus - P_minus) / (2.0 * g.hV);
00108 }
```

5.3 LookbackOption.h File Reference

```
#include <cmath>
#include <algorithm>
#include <random>
#include <iostream>
```

Classes

- struct LookbackParams

- struct GreeksParams
- class LookbackOption

Macros

- #define LOOKBACKOPTION_H

Enumerations

- enum class Option { Call , Put }

5.3.1 Macro Definition Documentation

5.3.1.1 LOOKBACKOPTION_H

```
#define LOOKBACKOPTION_H
```

Definition at line 3 of file LookbackOption.h.

5.3.2 Enumeration Type Documentation

5.3.2.1 Option

```
enum class Option  [strong]
```

Enumerator

Call	
Put	

Definition at line 9 of file LookbackOption.h.

5.4 LookbackOption.h

Go to the documentation of this file.

```
00001 #pragma once
00002 #ifndef LOOKBACKOPTION_H
00003 #define LOOKBACKOPTION_H
00004 #include <cmath>
00005 #include <algorithm>
00006 #include <random>
00007 #include <iostream>
00008
00009 enum class Option {
00010     Call,
00011     Put
00012 };
00013
00014 struct LookbackParams {
00015     Option type;
```

```
00016     double S0;
00017     double r;
00018     double sigma;
00019     double T;
00020
00021     //Monte Carlo
00022     int steps;
00023     int paths;
00024     unsigned int seed;
00025 };
00026
00027 struct GreeksParams {
00028     double hS; //delta
00029     double hg; //gamma
00030     double dt; //theta
00031     double hR; //rho
00032     double hV; //vega
00033 };
00034
00035 class LookbackOption {
00036 private:
00037
00038     //Function to simulate a single path of the option lookback
00039     double simulate_lookback(Option type, double S0, double r, double sigma, double T, int steps,
      std::mt19937& gen, std::normal_distribution<>& Z) const;
00040
00041 public:
00042     //Price simulation with Monte Carlo
00043     double price_lookback(const LookbackParams& p) const;
00044
00045     //Greeks
00046
00047     //Delta of the option
00048     double delta_lookback(const LookbackParams& p, const GreeksParams& g) const;
00049
00050     //Gamma of the option
00051     double gamma_lookback(const LookbackParams& p, const GreeksParams& g) const;
00052
00053     //Theta of the option
00054     double theta_lookback(const LookbackParams& p, const GreeksParams& g) const;
00055
00056     //Rho of the option
00057     double rho_lookback(const LookbackParams& p, const GreeksParams& g) const;
00058
00059     //Vega of the option
00060     double vega_lookback(const LookbackParams& p, const GreeksParams& g) const;
00061 };
00062
00063 #endif // !LOOKBACKOPRION_H
00064
```

5.5 pricer_api.cpp File Reference

```
#include "pricer_api.h"
#include "LookbackOption.h"
#include <exception>
#include <string>
#include <cstring>
```

Functions

- int DLL_CALL LookbackPriceGreeks (double S0, double r, double sigma, double T, int isCall, int steps, int paths, double hS, double hg, double dt, double hR, double hV, unsigned int seed, double ∗price, double ∗delta, double ∗gamma, double ∗theta, double ∗rho, double ∗vega, char ∗errMsg, int errLen)
- int DLL_CALL LookbackCountPoints (double Smin, double Smax, double dS)
- int DLL_CALL LookbackPriceDeltaCurve (double Smin, double Smax, double dS, double r, double sigma, double T, int isCall, int steps, int paths, double hS, unsigned int seed, double ∗outS, double ∗outPrice, double ∗outDelta, int nPoints, char ∗errMsg, int errLen)

5.5.1 Function Documentation

5.5.1.1 LookbackCountPoints()

```
int DLL_CALL LookbackCountPoints (
            double Smin,
            double Smax,
            double dS)
```

Definition at line 107 of file pricer_api.cpp.

5.5.1.2 LookbackPriceDeltaCurve()

```
int DLL_CALL LookbackPriceDeltaCurve (
            double Smin,
            double Smax,
            double dS,
            double r,
            double sigma,
            double T,
            int isCall,
            int steps,
            int paths,
            double hS,
            unsigned int seed,
            double * outS,
            double * outPrice,
            double * outDelta,
            int nPoints,
            char * errMsg,
            int errLen)
```

Definition at line 127 of file pricer_api.cpp.

5.5.1.3 LookbackPriceGreeks()

```
int DLL_CALL LookbackPriceGreeks (
            double S0,
            double r,
            double sigma,
            double T,
            int isCall,
            int steps,
            int paths,
            double hS,
            double hg,
            double dt,
            double hR,
            double hV,
            unsigned int seed,
            double * price,
            double * delta,
            double * gamma,
            double * theta,
```

```
            double * rho,
            double * vega,
            char * errMsg,
            int errLen)
```

Definition at line 43 of file pricer_api.cpp.

5.6 pricer_api.cpp

Go to the documentation of this file.

```
00001 #include "pricer_api.h"
00002 #include "LookbackOption.h"
00003 #include <exception>
00004 #include <string>
00005 #include <cstring>
00006
00007 // Helper that writes error message
00008 static void write_err(char* err, int errLen, const std::string& msg)
00009 {
00010     if (!err || errLen <= 0) return;
00011
00012     // strncpy_s wants size_t for sizes
00013     strncpy_s(
00014         err,
00015         static_cast<size_t>(errLen),
00016         msg.c_str(),
00017         _TRUNCATE
00018     );
00019 }
00020
00021 // Convert an integer flag (from VBA) into the internal C++ enum
00022 static Option toOption(int isCall)
00023 {
00024     if (isCall == 1)
00025         return Option::Call;
00026     if (isCall==0 )
00027         return Option::Put;
00028     else
00029         throw std::invalid_argument("isCall must be 0 (Put) or 1 (Call)");
00030 }
00031
00032
00033 // LookbackPriceGreeks
00034 // Public DLL entry point: computes Monte Carlo price and Greeks for a lookback option.
00035 // Results are returned via output pointers so that multiple values can be returned to callers like
      VBA/Excel.
00036
00037 // Return codes:
00038 //  0  = success
00039 // -1  = null output pointers
00040 // -2  = invalid model inputs (S0, sigma, T)
00041 // -3  = invalid Monte Carlo settings (steps, paths)
00042
00043 int DLL_CALL LookbackPriceGreeks(
00044     double S0, double r, double sigma, double T,
00045     int isCall,
00046     int steps, int paths,
00047     double hS, double hg, double dt, double hR, double hV,
00048     unsigned int seed,
00049     double* price, double* delta, double* gamma,
00050     double* theta, double* rho, double* vega, char* errMsg, int errLen) {
00051     // clear message
00052     if (errMsg && errLen > 0) errMsg[0] = '\0';
00053
00054     //checks
00055     if (!price || !delta || !gamma || !theta || !rho || !vega) {
00056         write_err(errMsg, errLen, "Null output pointer(s) passed from VBA.");
00057         return -1;
00058     }
00059
00060     if (S0 <= 0.0 || sigma <= 0.0 || T <= 0.0) {
00061         write_err(errMsg, errLen, "Invalid model inputs: require S0>0, sigma>0, T>0.");
00062         return -3;
00063     }
00064     if (steps <= 0 || paths <= 0) {
00065         write_err(errMsg, errLen, "Invalid Monte Carlo settings: require steps>0, paths>0.");
00066         return -3;
00067     }
```

```
00068     if (dt <= 0.0 || dt >= T) {
00069         write_err(errMsg, errLen, "Invalid theta bump: require 0 < dt < T.");
00070         return -4;
00071     }
00072 
00073     try{
00074     const Option type = toOption(isCall);
00075     LookbackParams p{ type, S0, r, sigma, T, steps, paths, seed };
00076     GreeksParams g{ hS, hg, dt, hR, hV };
00077     LookbackOption pricer;
00078     // Price
00079     *price = pricer.price_lookback(p);
00080 
00081     // Greeks
00082     *delta = pricer.delta_lookback(p,g);
00083     *gamma = pricer.gamma_lookback(p,g);
00084     *theta = pricer.theta_lookback(p,g);
00085     *rho = pricer.rho_lookback(p,g);
00086     *vega = pricer.vega_lookback(p,g);
00087 
00088     return 0;
00089 }
00090     catch (const std::exception& e) {
00091         write_err(errMsg, errLen, std::string("C++ exception: ") + e.what());
00092         return -100;
00093     }
00094     catch(...) {
00095         write_err(errMsg, errLen, "Unknown C++ exception.");
00096         return -101;
00097     }
00098 }
00099 
00100 // LookbackCountPoints
00101 // Compute how many points are needed for a grid Smin..Smax with step dS.
00102 
00103 // Return codes:
00104 //  n>=0 = number of points
00105 //  -1   = invalid step (dS <= 0)
00106 //  -2   = invalid range (Smax < Smin)
00107 int DLL_CALL LookbackCountPoints(double Smin, double Smax, double dS) {
00108     if (dS <= 0.0) return -1;
00109     if (Smax < Smin) return -2;
00110 
00111     // includiamo entrambi gli estremi come fai nel for (S0 <= Smax)
00112     const double span = Smax - Smin;
00113     const int n = (int)std::floor(span / dS + 1.0 + 1e-12);
00114     return std::max(n, 0);
00115 }
00116 
00117 // LookbackPriceDeltaCurve
00118 // Public DLL entry point for plotting: fills pre-allocated arrays with spot values, option prices,
      and deltas over a spot grid.
00119 
00120 // Return codes:
00121 //  0  = success
00122 // -1  = null output pointers
00123 // -2  = invalid nPoints
00124 // -3  = invalid grid definition (or computed expected points <= 0)
00125 // -4  = nPoints too small for the requested grid
00126 
00127 int DLL_CALL LookbackPriceDeltaCurve(
00128     double Smin, double Smax, double dS,
00129     double r, double sigma, double T,
00130     int isCall,
00131     int steps, int paths,
00132     double hS,
00133     unsigned int seed,
00134     double* outS, double* outPrice, double* outDelta,
00135     int nPoints, char* errMsg, int errLen
00136 ) {
00137     // clear message
00138     if (errMsg && errLen > 0) errMsg[0] = '\0';
00139 
00140     if (!outS || !outPrice || !outDelta) {
00141         write_err(errMsg, errLen, "Null output array pointer(s) passed from VBA.");
00142         return -1;
00143     }
00144     if (nPoints <= 0) {
00145         write_err(errMsg, errLen, "Invalid nPoints: must be > 0.");
00146         return -2;
00147     }
00148 
00149     int expected = LookbackCountPoints(Smin, Smax, dS);
00150     if (expected <= 0) {
00151         write_err(errMsg, errLen, "Invalid grid: check Smin/Smax/dS.");
00152         return -3;
00153     }
```

```
00154
00155     if (nPoints < expected) {
00156         write_err(errMsg, errLen, "nPoints too small for the requested grid.");
00157         return -4;
00158     }
00159     try{
00160     Option type = toOption(isCall);
00161     LookbackOption pricer;
00162     GreeksParams g{ hS, hS, 1e-4, 1e-4, 1e-4 };
00163
00164
00165     int k = 0;
00166     for (double S0 = Smin; S0 <= Smax + 1e-12; S0 += dS) {
00167         if (k >= nPoints) break;
00168
00169         outS[k] = S0;
00170         LookbackParams p{ type, S0, r, sigma, T, steps, paths, seed };
00171
00172         // price
00173         outPrice[k] = pricer.price_lookback(p);
00174
00175         // delta
00176         outDelta[k] =pricer.delta_lookback(p,g);
00177
00178         k++;
00179     }
00180
00181     return 0;
00182 }
00183     catch (const std::exception& e) {
00184         write_err(errMsg, errLen, std::string("C++ exception: ") + e.what());
00185         return -100;
00186     }
00187     catch (...) {
00188         write_err(errMsg, errLen, "Unknown C++ exception.");
00189         return -101;
00190     }
00191 }
```

5.7 pricer_api.h File Reference

```
#include "LookbackOption.h"
```

Macros

- #define DLL_EXPORT extern "C"
- #define DLL_CALL

Functions

- DLL_EXPORT int DLL_CALL LookbackPriceGreeks (double S0, double r, double sigma, double T, int isCall, int steps, int paths, double hS, double hg, double dt, double hR, double hV, unsigned int seed, double ∗price, double ∗delta, double ∗gamma, double ∗theta, double ∗rho, double ∗vega, char ∗errMsg, int errLen)
- DLL_EXPORT int DLL_CALL LookbackPriceDeltaCurve (double Smin, double Smax, double dS, double r, double sigma, double T, int isCall, int steps, int paths, double hS, unsigned int seed, double ∗outS, double ∗outPrice, double ∗outDelta, int nPoints, char ∗errMsg, int errLen)
- DLL_EXPORT int DLL_CALL LookbackCountPoints (double Smin, double Smax, double dS)

5.7.1 Macro Definition Documentation

5.7.1.1 DLL_CALL

```
#define DLL_CALL
```

Definition at line 15 of file pricer_api.h.

5.7.1.2 DLL_EXPORT

```
#define DLL_EXPORT extern "C"
```

Definition at line 14 of file pricer_api.h.

5.7.2 Function Documentation

5.7.2.1 LookbackCountPoints()

```
DLL_EXPORT int DLL_CALL LookbackCountPoints (
            double Smin,
            double Smax,
            double dS)
```

Definition at line 107 of file pricer_api.cpp.

5.7.2.2 LookbackPriceDeltaCurve()

```
DLL_EXPORT int DLL_CALL LookbackPriceDeltaCurve (
            double Smin,
            double Smax,
            double dS,
            double r,
            double sigma,
            double T,
            int isCall,
            int steps,
            int paths,
            double hS,
            unsigned int seed,
            double * outS,
            double * outPrice,
            double * outDelta,
            int nPoints,
            char * errMsg,
            int errLen)
```

Definition at line 127 of file pricer_api.cpp.

5.7.2.3 LookbackPriceGreeks()

```
DLL_EXPORT int DLL_CALL LookbackPriceGreeks (
            double S0,
            double r,
            double sigma,
            double T,
            int isCall,
            int steps,
            int paths,
            double hS,
            double hg,
```

```
            double dt,
            double hR,
            double hV,
            unsigned int seed,
            double * price,
            double * delta,
            double * gamma,
            double * theta,
            double * rho,
            double * vega,
            char * errMsg,
            int errLen)
```

Definition at line 43 of file pricer_api.cpp.

5.8 pricer_api.h

Go to the documentation of this file.

```
00001 #pragma once
00002 #include "LookbackOption.h"
00003
00004 // Cross-platform export/calling-convention macros.
00005 // On Windows we must:
00006 //  - export symbols from the DLL (__declspec(dllexport))
00007 //  - use C linkage (extern "C") to avoid C++ name mangling (important for VBA/C/Python bindings)
00008 //  - specify a calling convention (__stdcall) commonly expected by Windows/Excel/VBA
00009
00010 #ifdef _WIN32
00011 #define DLL_EXPORT extern "C" __declspec(dllexport)
00012 #define DLL_CALL __stdcall
00013 #else
00014 #define DLL_EXPORT extern "C"
00015 #define DLL_CALL
00016 #endif
00017
00018 // Computes the Monte Carlo price and Greeks for a lookback option
00019
00020 // Inputs: S0, r, sigma, T, isCall, steps, paths, h_delta, h_gamma, h_theta, h_rho, h_vega, seed
00021 // isCall: option type selector (1=Call, 0=Put)
00022
00023 // Outputs: price, delta, gamma, theta, rho, vega
00024
00025 // Return value: integer status code (0 = success, non-zero = error)
00026 DLL_EXPORT int DLL_CALL LookbackPriceGreeks(
00027     double S0, double r, double sigma, double T,
00028     int isCall,
00029     int steps, int paths,
00030     double hS, double hg, double dt, double hR, double hV,
00031     unsigned int seed,
00032     double* price, double* delta, double* gamma,
00033     double* theta, double* rho, double* vega, char* errMsg, int errLen
00034 );
00035
00036 // Utility function for plotting (fills arrays)
00037 // Computes price(S) and delta(S) over a grid of underlying prices
00038
00039 // Inputs: Smin, Smax, ds, r, sigma, T, isCall, steps, paths,  h_delta, h_gamma, h_theta, h_rho,
      h_vega, seed, n_poins
00040 // nPoints = number of points allocated in output arrays
00041
00042 // Outputs: S[i], Price[i], Delta[i]
00043
00044 // Return value: integer status code (0 = success, non-zero = error)
00045 DLL_EXPORT int DLL_CALL LookbackPriceDeltaCurve(
00046     double Smin, double Smax, double dS,
00047     double r, double sigma, double T,
00048     int isCall,
00049     int steps, int paths,
00050     double hS,
00051     unsigned int seed,
00052     double* outS, double* outPrice, double* outDelta,
00053     int nPoints, char* errMsg, int errLen
00054 );
00055
```

```
00056 // Helper to compute how many grid points are needed for Smin..Smax with step dS.
00057
00058 // Return value: number of points required (or a negative value on invalid inputs)
00059 DLL_EXPORT int DLL_CALL LookbackCountPoints(double Smin, double Smax, double dS);
```

5.9 README.md File Reference

Index