

## European Vanilla Call and Put Options

Generated by Doxygen 1.9.7



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Namespace Documentation</b>	<b>7</b>
4.1 std_norm Namespace Reference	7
4.1.1 Function Documentation	7
4.1.1.1 cdf()	7
4.1.1.2 pdf()	8
4.1.2 Variable Documentation	8
4.1.2.1 norm_coeff	8
<b>5 Class Documentation</b>	<b>9</b>
5.1 EuropeanVanillaOption Class Reference	9
5.1.1 Constructor & Destructor Documentation	10
5.1.1.1 EuropeanVanillaOption() [1/2]	10
5.1.1.2 EuropeanVanillaOption() [2/2]	10
5.1.2 Member Function Documentation	10
5.1.2.1 calc_d_()	10
5.1.2.2 callPrice()	11
5.1.2.3 putPrice()	11
5.1.3 Member Data Documentation	11
5.1.3.1 d_1_	11
5.1.3.2 d_2_	11
5.1.3.3 K_	11
5.1.3.4 r_	12
5.1.3.5 S_	12
5.1.3.6 sigma_	12
5.1.3.7 T_	12
<b>6 File Documentation</b>	<b>13</b>
6.1 src/eur_van_opt.hpp File Reference	13
6.1.1 Detailed Description	13
6.2 src/std_norm.hpp File Reference	14
6.2.1 Detailed Description	14
<b>Index</b>	<b>15</b>



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">std_norm</a>	.....	7
--------------------------	-------	---



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">EuropeanVanillaOption</a> . . . . .	9
-------------------------------------------------	---





## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

src/ <a href="#">eur_van_opt.hpp</a>	
European Vanilla Call and Put Options implementation . . . . .	<a href="#">13</a>
src/ <a href="#">std_norm.hpp</a>	
Standard normal distribution basic functions implementation . . . . .	<a href="#">14</a>



## Chapter 4

# Namespace Documentation

### 4.1 std\_norm Namespace Reference

#### Functions

- double [pdf](#) (double x)  
*Probability density function (PDF) for  $\mathcal{N}(0, 1)$ :*
- double [cdf](#) (double x)  
*Cumulative distribution function (CDF) for  $\mathcal{N}(0, 1)$ :*

#### Variables

- const auto [norm\\_coeff](#) = 1.0 / std::pow(2.0 \* M\_PI, 0.5)

#### 4.1.1 Function Documentation

##### 4.1.1.1 cdf()

```
double std_norm::cdf (  
    double x )
```

Cumulative distribution function (CDF) for  $\mathcal{N}(0, 1)$ :

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-t^2/2} dt.$$

The definite integral calculation is adapted from [Michael Halls-Moore. C++ for Quantitative Finance, 2010](#) which in turn is an adaptation from [Mark S. Joshi. C++ Design Patterns and Derivatives Pricing, 2nd Ed. Cambridge University Press, 2008.](#)

#### Parameters

x	
---	--

**Returns**

double

References [pdf\(\)](#).Referenced by [EuropeanVanillaOption::callPrice\(\)](#), and [EuropeanVanillaOption::putPrice\(\)](#).**4.1.1.2 pdf()**

```
double std_norm::pdf (
    double x )
```

Probability density function (PDF) for  $\mathcal{N}(0, 1)$ :

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

**Parameters**

x	
---	--

**Returns**

double

References [norm\\_coeff](#).Referenced by [cdf\(\)](#).**4.1.2 Variable Documentation****4.1.2.1 norm\_coeff**

```
const auto std_norm::norm_coeff = 1.0 / std::pow(2.0 * M_PI, 0.5)
```

Referenced by [pdf\(\)](#).

# Chapter 5

## Class Documentation

### 5.1 EuropeanVanillaOption Class Reference

```
#include <eur_van_opt.hpp>
```

#### Public Member Functions

- [EuropeanVanillaOption](#) ()  
*Default constructor a new European Vanilla Option object.*
- [EuropeanVanillaOption](#) (const double &K, const double &r, const double &T, const double &S, const double &sigma)  
*Parametric constructor of a new European Vanilla Option object.*
- double [callPrice](#) () const  
*Calculate Call option price:*
- double [putPrice](#) () const  
*Calculate Put option price:*

#### Private Member Functions

- void [calc\\_d\\_](#) ()  
*Calculate intermediate variables:*

#### Private Attributes

- double [K\\_](#)
- double [r\\_](#)
- double [T\\_](#)
- double [S\\_](#)
- double [sigma\\_](#)
- double [d\\_1\\_](#)
- double [d\\_2\\_](#)

## 5.1.1 Constructor & Destructor Documentation

### 5.1.1.1 EuropeanVanillaOption() [1/2]

```
EuropeanVanillaOption::EuropeanVanillaOption ( ) [inline]
```

Default constructor a new European Vanilla Option object.

References [calc\\_d\\_\(\)](#).

### 5.1.1.2 EuropeanVanillaOption() [2/2]

```
EuropeanVanillaOption::EuropeanVanillaOption (
    const double & K,
    const double & r,
    const double & T,
    const double & S,
    const double & sigma ) [inline]
```

Parametric constructor of a new European Vanilla Option object.

#### Parameters

$K$	Strike price of the option
$r$	Risk-free interest rate
$T$	Time to maturity (in years)
$S$	Current price of the underlying asset
$\sigma$	Volatility of the underlying asset's returns

References [calc\\_d\\_\(\)](#).

## 5.1.2 Member Function Documentation

### 5.1.2.1 calc\_d\_()

```
void EuropeanVanillaOption::calc_d_ ( ) [inline], [private]
```

Calculate intermediate variables:

$d_1 = \frac{\ln(S/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$ , represents a standardized measure of how far the current price  $S$  is from the strike price  $K$  after accounting for the time to maturity, risk-free rate, and volatility.

$d_2 = d_1 - \sigma\sqrt{T}$ , reflects the uncertainty (volatility) over the time to maturity.

References [d\\_1\\_](#), [d\\_2\\_](#), [K\\_](#), [r\\_](#), [S\\_](#), [sigma\\_](#), and [T\\_](#).

Referenced by [EuropeanVanillaOption\(\)](#), and [EuropeanVanillaOption\(\)](#).

### 5.1.2.2 callPrice()

```
double EuropeanVanillaOption::callPrice ( ) const [inline]
```

Calculate Call option price:

$$C(S) = SN(d_1) - Ke^{-rT}N(d_2), \text{ where } N = CDF_{\mathcal{N}(0,1)}.$$

#### Returns

double

References [std\\_norm::cdf\(\)](#), [d\\_1\\_](#), [d\\_2\\_](#), [K\\_](#), [r\\_](#), [S\\_](#), and [T\\_](#).

### 5.1.2.3 putPrice()

```
double EuropeanVanillaOption::putPrice ( ) const [inline]
```

Calculate Put option price:

$$P(S) = Ke^{-rT}N(-d_2) - SN(-d_1), \text{ where } N = CDF_{\mathcal{N}(0,1)}.$$

#### Returns

double

References [std\\_norm::cdf\(\)](#), [d\\_1\\_](#), [d\\_2\\_](#), [K\\_](#), [r\\_](#), [S\\_](#), and [T\\_](#).

## 5.1.3 Member Data Documentation

### 5.1.3.1 d\_1\_

```
double EuropeanVanillaOption::d_1_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

### 5.1.3.2 d\_2\_

```
double EuropeanVanillaOption::d_2_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

### 5.1.3.3 K\_

```
double EuropeanVanillaOption::K_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

#### 5.1.3.4 `r_`

```
double EuropeanVanillaOption::r_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

#### 5.1.3.5 `S_`

```
double EuropeanVanillaOption::S_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

#### 5.1.3.6 `sigma_`

```
double EuropeanVanillaOption::sigma_ [private]
```

Referenced by [calc\\_d\\_\(\)](#).

#### 5.1.3.7 `T_`

```
double EuropeanVanillaOption::T_ [private]
```

Referenced by [calc\\_d\\_\(\)](#), [callPrice\(\)](#), and [putPrice\(\)](#).

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

- [src/eur\\_van\\_opt.hpp](#)



# Chapter 6

## File Documentation

### 6.1 src/eur\_van\_opt.hpp File Reference

European Vanilla Call and Put Options implementation.

```
#include "std_norm.hpp"
```

#### Classes

- class [EuropeanVanillaOption](#)

#### 6.1.1 Detailed Description

European Vanilla Call and Put Options implementation.

#### Author

Andrei Batyrov

#### Version

0.1

#### Date

2024-12-01

#### Copyright

Copyright (c) 2024

## 6.2 src/std\_norm.hpp File Reference

Standard normal distribution basic functions implementation.

```
#include <cmath>
```

### Namespaces

- namespace `std_norm`

### Functions

- double `std_norm::pdf` (double x)  
*Probability density function (PDF) for  $\mathcal{N}(0, 1)$ :*
- double `std_norm::cdf` (double x)  
*Cumulative distribution function (CDF) for  $\mathcal{N}(0, 1)$ :*

### Variables

- const auto `std_norm::norm_coeff` = 1.0 / std::pow(2.0 \* M\_PI, 0.5)

### 6.2.1 Detailed Description

Standard normal distribution basic functions implementation.

#### Author

Andrei Batyrov

#### Version

0.1

#### Date

2024-12-01

#### Copyright

Copyright (c) 2024

# Index

- calc\_d\_
  - EuropeanVanillaOption, [10](#)
- callPrice
  - EuropeanVanillaOption, [10](#)
- cdf
  - std\_norm, [7](#)
- d\_1\_
  - EuropeanVanillaOption, [11](#)
- d\_2\_
  - EuropeanVanillaOption, [11](#)
- EuropeanVanillaOption, [9](#)
  - calc\_d\_, [10](#)
  - callPrice, [10](#)
  - d\_1\_, [11](#)
  - d\_2\_, [11](#)
  - EuropeanVanillaOption, [10](#)
  - K\_, [11](#)
  - putPrice, [11](#)
  - r\_, [11](#)
  - S\_, [12](#)
  - sigma\_, [12](#)
  - T\_, [12](#)
- K\_
  - EuropeanVanillaOption, [11](#)
- norm\_coeff
  - std\_norm, [8](#)
- pdf
  - std\_norm, [8](#)
- putPrice
  - EuropeanVanillaOption, [11](#)
- r\_
  - EuropeanVanillaOption, [11](#)
- S\_
  - EuropeanVanillaOption, [12](#)
- sigma\_
  - EuropeanVanillaOption, [12](#)
- src/eur\_van\_opt.hpp, [13](#)
- src/std\_norm.hpp, [14](#)
- std\_norm, [7](#)
  - cdf, [7](#)
  - norm\_coeff, [8](#)
  - pdf, [8](#)
- T\_
  - EuropeanVanillaOption, [12](#)