
FBComplex

Complex number library for FreeBasic

Jean Debord

January 18, 2012

Welcome to **FBComplex**. This library allows to handle complex numbers and functions in FreeBASIC. It is based on the Delphi library Complex-Math by E. F. Glynn (<http://www.efg2.com/Lab/Mathematics/Complex/index.html>).

1 Installation and compilation

1.1 Windows installation

The archive contains a precompiled library `libfbcomplex.a` for Windows. This file should be placed in the `lib\win32` subdirectory of the FreeBASIC directory. Accordingly, the header file `fbcomplex.bi` should be placed in the `inc` subdirectory.

1.2 Linux installation

The library can be created with the following commands:

```
fbcc -c fbcomplex.bas
ar r libfbcomplex.a fbcomplex.o
```

Then the library and header files can be installed in their respective subdirectories, as for Windows.

2 Using the library

In order to use the library, place the following statement at the beginning of the program:

```
#INCLUDE "fbcomplex.bi"
```

FBComplex can be used with FBMath (<http://sourceforge.net/projects/fbmath/>) but fbmath.bi must be included *before* fbcomplex.bi:

```
#INCLUDE "fbmath.bi"
#include "fbcomplex.bi"
```

3 Introduction to complex numbers

3.1 Definitions

A complex number Z is a pair of real numbers:

$$Z = (X, Y)$$

It is related to the coordinates of a point M in the plane (called the *complex plane* in this case).

Complex addition and subtraction are defined as for vectors:

$$Z_1 = (X_1, Y_1)$$

$$Z_2 = (X_2, Y_2)$$

$$Z_1 + Z_2 = (X_1 + X_2, Y_1 + Y_2)$$

$$Z_1 - Z_2 = (X_1 - X_2, Y_1 - Y_2)$$

A complex number may also be written as $Z = X + iY$ where i is the number $(0,1)$. This notation defines the *rectangular form* of the complex. The coordinates X and Y are often called the *real part* and the *imaginary part*, noted respectively $\Re(Z)$ and $\Im(Z)$.

The complex product is defined as:

$$Z_1 \cdot Z_2 = (X_1X_2 - Y_1Y_2, X_1Y_2 + Y_1X_2)$$

It is related to the rotation of vectors in the plane.

The complex product is commutative, i. e. $Z_1Z_2 = Z_2Z_1$

The number i is such that $i^2 = (0, 1) \cdot (0, 1) = (-1, 0) = -1$

Every complex $Z \neq 0$ has an inverse Z^{-1} for the multiplication.

3.2 Polar form

A complex number $Z = X + iY$ can also be written as $R(\cos \theta + i \sin \theta)$, where:

- R is the norm of the vector \overrightarrow{OM} , also called the *modulus* or the absolute value of the complex number and noted $|Z|$
- θ is the angle between the Ox axis and the vector \overrightarrow{OM} , counted in the interval $] - \pi, \pi]$. It is often called the *argument* or the *phase* of the complex number.

This notation defines the *polar form* of the complex number.

We have the following relationships:

$$R = \sqrt{X^2 + Y^2} \quad \theta = \arctan \frac{Y}{X} \quad X = R \cos \theta \quad Y = R \sin \theta$$

3.3 Exponential form

It has been shown that the exponential function can be extended to complex numbers, and that:

$$\exp(i\theta) = \cos \theta + i \sin \theta$$

So, the notation $R(\cos \theta + i \sin \theta)$ is equivalent to $R \exp(i\theta)$, where all the classical properties of the exponential apply, for instance:

$$Z_1 = R_1 \exp(i\theta_1) \quad , \quad Z_2 = R_2 \exp(i\theta_2) \quad \Rightarrow \quad Z_1 Z_2 = R_1 R_2 \exp[i(\theta_1 + \theta_2)]$$

4 Type definition

Type `Complex` is defined as:

```
TYPE Complex
  X AS DOUBLE
  Y AS DOUBLE
END TYPE
```

So, a complex variable `Z` is declared as follows:

```
DIM AS Complex Z
```

Its real and imaginary parts are then `Z.X` and `Z.Y`

The complex variables can be initialized, for instance:

```
DIM AS Complex A = (1, 1), B = (SQR(3), -1)
```

sets $A = 1 + i$ and $B = \sqrt{3} - i$

You can also declare arrays of complexes and initialize them, for instance:

```
DIM AS Complex Mat(1 TO 2, 1 TO 2) = {{(0,0), (1,0)}, _
                                         {(0,1), (1,1)}}
```

5 Error handling

The function `CMathErr()` returns the error code from the last function evaluation. It must be checked immediately after a function call:

```
ExpZ = CExp(Z)
if CMathErr() = FOk then ...
```

If an error occurs, a default value is attributed to the function. The possible error codes are the following:

Error code	Value	Meaning
FOk	0	No error
FDomain	-1	Argument domain error
FSing	-2	Function singularity
FOverflow	-3	Overflow range error
FUnderflow	-4	Underflow range error

6 Number construction

The following functions create a complex number from either its rectangular or polar coordinates:

- Function `Cmplx(X, Y)` returns the complex number $X + iY$
- Function `Polar(R, Theta)` returns the complex number $R(\cos \theta + i \sin \theta)$

The functions `CReal(Z)` and `CImag(Z)` return the real part and the imaginary part of their complex argument Z .

7 Sign and exchange

- Function `CSgn(Z)` returns the sign of the complex Z , such that:

$$\text{CSgn}(Z) = \begin{cases} 1 & \text{if } \Re(Z) > 0 \text{ or } \Re(Z) = 0 \text{ and } \Im(Z) > 0 \\ -1 & \text{if } \Re(Z) < 0 \text{ or } \Re(Z) = 0 \text{ and } \Im(Z) < 0 \end{cases}$$

This function is used to determine in which half-plane ('left' or 'right') of the complex plane the number Z lies.

- Procedure `CSwap(W, Z)` exchanges the two complex numbers W and Z .

8 Modulus and argument

The functions `CAbs(Z)` and `CArg(Z)` give, respectively, the modulus and the argument of the complex Z , i. e. the numbers R and θ such that $Z = R \exp(i\theta)$ with $\theta \in]-\pi, \pi]$.

9 Complex conjugate

The function `CConj(Z)` returns the conjugate of Z i.e. $\bar{Z} = X - iY$

10 Arithmetic operators

The operators `=` and `<>` allow to test the equality of two complex numbers.

The operators `+`, `-`, `*` and `/` accept operands of real (double precision) or complex type.

The exponentiation operator `^` computes Z^a . The algorithms used are the following:

- Z complex, a integer : repeated multiplications with Legendre's algorithm to minimize the number of operations. For instance, Z^8 is computed as $Z^2 = Z \cdot Z$, $Z^4 = Z^2 \cdot Z^2$ and $Z^8 = Z^4 \cdot Z^4$, hence 3 multiplications instead of 7.
- Z complex, a real : DeMoivre's theorem :

$$Z^a = [R \exp(i\theta)]^a = R^a \exp(ai\theta) = R^a (\cos a\theta + i \sin a\theta)$$

- Z real or complex, a complex : $Z^a = \exp(a \ln Z)$

11 Polynomials and rational fractions

Function `CPoly(Z, Coef(), Deg)` evaluates the polynomial:

$$P(Z) = \text{Coef}(0) + \text{Coef}(1) \cdot Z + \text{Coef}(2) \cdot Z^2 + \cdots + \text{Coef}(\text{Deg}) \cdot Z^{\text{Deg}}$$

where Z is complex and the coefficients may be real or complex.

Function `CFrac(Z, Coef(), Deg1, Deg2)` evaluates the rational fraction:

$$F(Z) = \frac{\text{Coef}(0) + \text{Coef}(1) \cdot Z + \cdots + \text{Coef}(\text{Deg1}) \cdot Z^{\text{Deg1}}}{\text{Coef}(\text{Deg1} + 1) + \text{Coef}(\text{Deg1} + 2) \cdot Z + \cdots + \text{Coef}(\text{Deg1} + \text{Deg2} + 1) \cdot Z^{\text{Deg2}}}$$

where Z is complex and the coefficients may be real or complex.

12 Logarithm and exponential

It is obvious from the relationship below that the complex logarithm is a multi-valued function:

$$Z = R \cdot \exp(i\theta) = R \cdot \exp[i(\theta + 2k\pi)] \Rightarrow \ln Z = \ln R + i(\theta + 2k\pi)$$

The function `CLog(Z)` returns the *principal part* of the logarithm, i. e. the value corresponding to $k = 0$ and $\theta \in]-\pi, \pi]$.

The function `CExp(Z)` returns the exponential of Z , according to:

$$\exp(X + iY) = e^X (\cos Y + i \sin Y)$$

13 Complex roots

According to the following relationship:

$$Z = R \cdot \exp(i\theta) = R \cdot \exp[i(\theta + 2k\pi)] \Rightarrow Z^{1/n} = R^{1/n} \cdot \exp\left[i\left(\frac{\theta}{n} + \frac{2k\pi}{n}\right)\right]$$

a complex number has n distinct n -th roots, corresponding to $k = 0 \cdots (n-1)$

The function `CRoot(Z, K, N)` returns the K -th N -th root of the complex number Z (K and N are integers).

The function `CSqrt(Z)` returns the first square root of the complex number Z . It is therefore equivalent to `CRoot(Z, 0, 2)`.

In any case, only the principal part of the function is returned. For instance, if N is integer, $Z^{(1/N)}$ is equivalent to `CRoot(Z, 0, N)`.

14 Trigonometric functions

The following functions are available (where $Z = X + iY$):

Function	Formula
CSin(Z)	$\sin X \cosh Y + i \cos X \sinh Y$
CCos(Z)	$\cos X \cosh Y - i \sin X \sinh Y$
CTan(Z)	$\frac{\sin X \cos X + i \sinh Y \cosh Y}{\cos^2 X + \sinh^2 Y} \quad Z \neq \frac{\pi}{2} + k\pi$
CASin(Z)	$\arcsin(P - Q) + i \operatorname{csgn}(Y - iX) \ln(P + Q + \sqrt{(P + Q)^2 - 1})$ $P = \frac{1}{2}\sqrt{X^2 + 2X + 1 + Y^2} \quad Q = \frac{1}{2}\sqrt{X^2 - 2X + 1 + Y^2}$
CACos(Z)	$\frac{\pi}{2} - \arcsin(Z)$
CATan(Z)	$\frac{1}{2}[\arctan(X, 1 - Y) - \arctan(-X, 1 + Y)] + \frac{1}{4}i \ln \frac{X^2 + (Y+1)^2}{X^2 + (Y-1)^2} \quad Z \neq \pm i$

In addition, subroutine **CSinCos(Z, SinZ, CosZ)** allows to calculate the sine and cosine simultaneously, saving some computations if both functions are needed.

15 Hyperbolic functions

The following functions are available (where $Z = X + iY$):

Function	Formula
CSinh(Z)	$\sinh X \cos Y + i \cosh X \sin Y$
CCosh(Z)	$\cosh X \cos Y + i \sinh X \sin Y$
CTanh(Z)	$\frac{\sinh X \cosh X + i \sin Y \cos Y}{\sinh^2 X + \cos^2 Y} \quad Z \neq i\left(\frac{\pi}{2} + k\pi\right)$
CASinh(Z)	$-i \arcsin(iZ)$
CACosh(Z)	$\operatorname{csgn}[Y + i(1 - X)] \cdot i \arccos(Z)$
CATanh(Z)	$-i \arctan(iZ) \quad Z \neq \pm 1$

In addition, subroutine **CSinhCosh(Z, SinhZ, CoshZ)** allows to calculate the hyperbolic sine and cosine simultaneously, saving some computations if both functions are needed.

16 Gamma function

Function `CLnGamma(Z)` returns the natural logarithm of the Gamma function for the complex argument `Z`.

17 Printing functions

- Subroutine `CPrint(Z, Mask)` prints the complex `Z` in rectangular form, without line feed at the end. `Mask` is a format string, as defined for `PRINT USING`. This parameter is optional, with default value `"####.####"`. For instance :

```
DIM AS Complex Z = (SQR(3), -1)
CPrint Z
PRINT
CPrint Z, "##.#####"

```

will print:

```
1.7321 - 1.0000 * i
1.732051 - 1.000000 * i

```

- Subroutine `CPrintPolar(Z, Mask)` works like the previous one, but uses the exponential notation. For instance :

```
DIM AS Complex Z = (SQR(3), -1)
CPrintPolar Z

```

will print: `2.0000 * exp(-0.5236 * i)`

18 Demo programs

These programs are located in the `demo` subdirectory:

- Program `testcomp` checks the accuracy of the complex functions. It is a slight modification of a Pascal program written by E. Glynn.

The program defines an array of 20 complex numbers. The tests consist mostly of applying a function to each number, then applying the reciprocal function to the result, in order to retrieve the original number.

- Programs `polderiv` and `polderiv1` show how to compute a polynomial and its derivatives by Horner's method.
- Program `mandel` plots the Mandelbrot or Julia sets for the iteration formula $z' = z^p + c$ where p is a positive real exponent.

19 Contributed programs

These programs, located in the `contrib` subdirectory, have been written by 'dodicat' from the FreeBASIC forum.

- Program `matrix.bas` computes the determinant and inverse of a complex matrix.
- Program `gauss_jordan` solves a system of linear equations with complex coefficients by the Gauss-Jordan method.
- Program `newton` plots a fractal figure by applying Newton's method to a polynomial with complex coefficients. Each point is colored according to the number of iterations required for convergence, starting with the point coordinates. The program prints the number of points for which the method did not converge (according to the given maximal number of iterations), and the total elapsed time.