NAG C Library Function Document

nag_complex (a02bac)

1 Purpose

nag_complex (a02bac) returns a complex number from real and imaginary parts.

2 Specification

```
#include <nag.h>
#include <naga02.h>
```

```
Complex nag_complex (double x, double y)
```

3 Description

None.

4 References

None.

5 Arguments

1: **x** – double *Input*

On entry: real part of complex number.

2: **y** – double *Input*

On entry: imaginary part of complex number.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example illustrates the calls to all the complex functions in Chapter a02.

9.1 Program Text

```
/* nag_complex (a02bac) Example Program.
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 */
```

```
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <naga02.h>

int main(void)
{
  Complex  v, w, z;
  double r, theta, x, y;
  Nag_Boolean equal, not_equal;

  Vprintf("nag_complex (a02bac) Example Program Results\n");

  x = 2.0;
  y = -3.0;
  /* nag_complex (a02bac).
   * Complex number from real and imaginary parts
   */
  z = nag_complex(x, y);

  Vprintf("x = %7.4f, y = %7.4f\n", x, y);
  Vprintf("z = complex(x,y) = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_real (a02bbc).
   * Real part of a complex number
   */
  Vprintf("real(z) = %7.4f\n", nag_complex_real(z));
  /* nag_complex_imag (a02bcc).
   * Imaginary part of a complex number
   */
  Vprintf("imag(z) = %7.4f\n\n", nag_complex_imag(z));

  /* nag_complex (a02bac), see above. */
  v = nag_complex(3.0, 1.25);
  /* nag_complex (a02bac), see above. */
  w = nag_complex(2.5, -1.75);
  Vprintf("v = (%7.4f, %7.4f)\n", v.re, v.im);
  Vprintf("w = (%7.4f, %7.4f)\n", w.re, w.im);
  /* nag_complex_add (a02cac).
   * Addition of two complex numbers
   */
  z = nag_complex_add(v, w);
  Vprintf("z = v+w = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_subtract (a02cbc).
   * Subtraction of two complex numbers
   */
  z = nag_complex_subtract(v, w);
  Vprintf("z = v-w = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_multiply (a02ccc).
   * Multiplication of two complex numbers
   */
  z = nag_complex_multiply(v, w);
  Vprintf("z = v*w = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_divide (a02cdc).
   * Quotient of two complex numbers
   */
  z = nag_complex_divide(v, w);
  Vprintf("z = v/w = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_negate (a02cec).
   * Negation of a complex number
   */
  z = nag_complex_negate(w);
  Vprintf("z = -w = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_conjg (a02cfc).
   * Conjugate of a complex number
   */
  z = nag_complex_conjg(w);
  Vprintf("z = conjugate(w) = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_equal (a02cgc).
   * Equality of two complex numbers
   */
  equal = nag_complex_equal(v, w);
  if (equal)
    Vprintf("v == w\n");
  else
    Vprintf("v != w\n");
  /* nag_complex_not_equal (a02chc).
   * Inequality of two complex numbers
   */
  not_equal = nag_complex_not_equal(w, z);
  if (not_equal)
    Vprintf("w != z\n\n");
  else
    Vprintf("w == z\n\n");

  /* nag_complex_arg (a02dac).
   * Argument of a complex number
   */
  theta = nag_complex_arg(z);
  Vprintf("theta = arg(z) = %7.4f\n", theta);
  /* nag_complex_abs (a02dbc).
   * Modulus of a complex number
   */
  r = nag_complex_abs(z);
  Vprintf("r = abs(z) = %7.4f\n", r);
  /* nag_complex_sqrt (a02dcc).
   * Square root of a complex number
   */
  v = nag_complex_sqrt(z);
  Vprintf("v = sqrt(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_i_power (a02ddc).
   * Complex number raised to integer power
   */
  v = nag_complex_i_power(z, (Integer)3);
  Vprintf("v = z**3 = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_r_power (a02dec).
   * Complex number raised to real power
   */
  v = nag_complex_r_power(z, 2.5);
  Vprintf("v = z**2.5 = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_c_power (a02dfc).
   * Complex number raised to complex power
   */
  v = nag_complex_c_power(z, w);
  Vprintf("v = z**w = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_log (a02dgc).
   * Complex logarithm
   */
  v = nag_complex_log(z);
  Vprintf("v = log(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_exp (a02dhc).
   * Complex exponential
   */
  z = nag_complex_exp(v);
  Vprintf("z = exp(v) = (%7.4f, %7.4f)\n", z.re, z.im);
  /* nag_complex_sin (a02djc).
   * Complex sine
   */
  v = nag_complex_sin(z);
  Vprintf("v = sin(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_cos (a02dkc).
   * Complex cosine
   */
  v = nag_complex_cos(z);
  Vprintf("v = cos(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_tan (a02dlc).
   * Complex tangent
   */
  v = nag_complex_tan(z);
  Vprintf("v = tan(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  /* nag_complex_divide (a02cdc), see above. */
  v = nag_complex_divide (a02djc(z),a02dkc(z));
  Vprintf("sin(z)/cos(z) = (%7.4f, %7.4f)\n", v.re, v.im);
  return EXIT_SUCCESS;
}
```

9.2 Program Data

None.

9.3 Program Results

```
nag_complex (a02bac) Example Program Results
x =  2.0000, y = -3.0000
z = complex(x,y) = ( 2.0000, -3.0000)
real(z) =  2.0000
imag(z) = -3.0000

v = ( 3.0000,  1.2500)
w = ( 2.5000, -1.7500)
z = v+w = ( 5.5000, -0.5000)
z = v-w = ( 0.5000,  3.0000)
z = v*w = ( 9.6875, -2.1250)
z = v/w = ( 0.5705,  0.8993)
z = -w = (-2.5000,  1.7500)
z = conjugate(w) = ( 2.5000,  1.7500)
v != w
w != z

theta = arg(z) =  0.6107
r = abs(z) =  3.0516
v = sqrt(z) = ( 1.6661,  0.5252)
v = z**3 = (-7.3438, 27.4531)
v = z**2.5 = ( 0.7153, 16.2522)
v = z**w = (43.1428, -19.5581)
v = log(z) = ( 1.1157,  0.6107)
z = exp(v) = ( 2.5000,  1.7500)
v = sin(z) = ( 1.7740, -2.2355)
v = cos(z) = (-2.3747, -1.6700)
v = tan(z) = (-0.0569,  0.9814)
sin(z)/cos(z) = (-0.0569,  0.9814)
```