NAG Library Function Document

nag_lambertW (c05bac)

1 Purpose

nag_lambertW (c05bac) returns the real values of Lambert's W function $W(x)$.

2 Specification

```
#include <nag.h>
#include <nagc05.h>
double nag_lambertW (double x, Integer branch, Nag_Boolean offset,
       NagError *fail)
```

3 Description

nag_lambertW (c05bac) calculates an approximate value for the real branches of Lambert's W function (sometimes known as the 'product log' or 'Omega' function), which is the inverse function of

$$f(w) = we^w \quad \text{for} \quad w \in C.$$

The function f is many-to-one, and so, except at 0, W is multivalued. nag_lambertW (c05bac) restricts W and its argument x to be real, resulting in a function defined for $x \geq -\exp(-1)$ and which is double valued on the interval $(-\exp(-1), 0)$. This double-valued function is split into two real-valued branches according to the sign of $W(x) + 1$. We denote by W_0 the branch satisfying $W_0(x) \geq -1$ for all real x, and by W_{-1} the branch satisfying $W_{-1}(x) \leq -1$ for all real x. You may select your branch of interest using the argument **branch**.

The precise method used to approximate W is described fully in Barry *et al.* (1995). For x close to $-\exp(-1)$ greater accuracy comes from evaluating $W(-\exp(-1) + \Delta x)$ rather than $W(x)$: by setting **offset** = Nag_TRUE on entry you inform nag_lambertW (c05bac) that you are providing Δx, not x, in **x**.

4 References

Barry D J, Culligan–Hensley P J, and Barry S J (1995) Real values of the W-function *ACM Trans. Math. Software* **21(2)** 161–171

5 Arguments

1: **x** – double *Input*

On entry: if **offset** = Nag_TRUE, **x** is the offset Δx from $-\exp(-1)$ of the intended argument to W; that is, $W(\beta)$ is computed, where $\beta = -\exp(-1) + \Delta x$.

If **offset** = Nag_FALSE, **x** is the argument x of the function; that is, $W(\beta)$ is computed, where $\beta = x$.

Constraints:

if **branch** $= 0$, $-\exp(-1) \leq \beta$;
if **branch** $= -1$, $-\exp(-1) \leq \beta < 0.0$.

2: **branch** – Integer *Input*

On entry: the real branch required.

branch $= 0$
The branch W_0 is selected.

branch $= -1$
The branch W_{-1} is selected.

Constraint: **branch** $= 0$ or -1.

3: **offset** – Nag_Boolean *Input*

On entry: controls whether or not **x** is being specified as an offset from $-\exp(-1)$.

4: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_INT

On entry, **branch** $= \langle value \rangle$.
Constraint: **branch** $= 0$ or -1.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in How to Use the NAG Library and its Documentation for further information.

NE_REAL

On entry, **branch** $= -1$, **offset** $=$ Nag_FALSE and **x** $= \langle value \rangle$.
Constraint: if **branch** $= -1$ and **offset** $=$ Nag_FALSE then **x** < 0.0.

On entry, **branch** $= -1$, **offset** $=$ Nag_TRUE and **x** $= \langle value \rangle$.
Constraint: if **branch** $= -1$ and **offset** $=$ Nag_TRUE then **x** $< \exp(-1.0)$.

On entry, **offset** $=$ Nag_TRUE and **x** $= \langle value \rangle$.
Constraint: if **offset** $=$ Nag_TRUE then **x** ≥ 0.0.

On entry, **offset** $=$ Nag_FALSE and **x** $= \langle value \rangle$.
Constraint: if **offset** $=$ Nag_FALSE then **x** $\geq -\exp(-1.0)$.

NW_REAL

For the given offset **x**, W is negligibly different from -1: **x** $= \langle value \rangle$.

x is close to $-\exp(-1)$. Enter **x** as an offset to $-\exp(-1)$ for greater accuracy: **x** $= \langle value \rangle$.

7 Accuracy

For a high percentage of legal **x** on input, nag_lambertW (c05bac) is accurate to the number of decimal digits of precision on the host machine (see nag_decimal_digits (X02BEC)). An extra digit may be lost on some implementations and for a small proportion of such **x**. This depends on the accuracy of the base-10 logarithm on your system.

8 Parallelism and Performance

nag_lambertW (c05bac) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads from a file the values of the required branch, whether or not the arguments to W are to be considered as offsets to $-\exp(-1)$, and the arguments **x** themselves. It then evaluates the function for these sets of input data **x** and prints the results.

10.1 Program Text

```
/* nag_lambertW (c05bac) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagc05.h>

int main(void)
{
  /* Scalars */
  double w, x;
  Integer branch;
  Integer exit_status = 0;
  char offset[10];
  Nag_Boolean offsetenum;
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_lambertW (c05bac) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%*[^\n] ", &branch);
#else
  scanf("%" NAG_IFMT "%*[^\n] ", &branch);
#endif
#ifdef _WIN32
  scanf_s("%9s%*[^\n] ", offset, (unsigned)_countof(offset));
#else
```

```
  scanf("%9s%*[^\n] ", offset);
#endif
  /*
   * nag_enum_name_to_value (x04nac).
   * Converts NAG enum member name to value
   */
  offsetenum = (Nag_Boolean) nag_enum_name_to_value(offset);
  printf("\n");
  printf("branch = %" NAG_IFMT "\n", branch);
  printf("offset = %s\n", offset);
  printf("\n      x           w(x)\n\n");
#ifdef _WIN32
  while (scanf_s("%lf%*[^\n] ", &x) != EOF)
#else
  while (scanf("%lf%*[^\n] ", &x) != EOF)
#endif
  {
    /*
     * nag_lambertW (c05bac)
     * Real values of Lambert's W function, W(x)
     */
    w = nag_lambertW(x, branch, offsetenum, &fail);
    if (fail.code == NE_NOERROR) {
      printf("%14.5e%14.5e\n", x, w);
    }
    else {
      printf("Error from nag_lambertW (c05bac).\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
  }

END:
  return exit_status;
}
```

10.2 Program Data

```
nag_lambertW (c05bac) Example Program Data
  0                                                       : branch
  Nag_FALSE                                               : offset
  0.5
  1.0
  4.5
  6.0
  70000000.0
```

10.3 Program Results

```
nag_lambertW (c05bac) Example Program Results

branch = 0
offset = Nag_FALSE

      x           w(x)

   5.00000e-01   3.51734e-01
   1.00000e+00   5.67143e-01
   4.50000e+00   1.26724e+00
   6.00000e+00   1.43240e+00
   7.00000e+07   1.53339e+01
```