NAG Library Function Document

nag_mills_ratio (g01mbc)

1 Purpose

nag_mills_ratio (g01mbc) returns the reciprocal of Mills' Ratio.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
double nag_mills_ratio (double x)
```

3 Description

nag_mills_ratio (g01mbc) calculates the reciprocal of Mills' Ratio, the hazard rate, $\lambda(x)$, for the standard Normal distribution. It is defined as the ratio of the ordinate to the upper tail area of the standard Normal distribution, that is,

$$\lambda(x) = \frac{Z(x)}{Q(x)} = \frac{\frac{1}{\sqrt{2\pi}}e^{-\left(x^2/2\right)}}{\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-(t^2/2)}\,dt}.$$

The calculation is based on a Chebyshev expansion as described in nag_erfcx (s15agc).

4 References

Gross A J and Clark V A (1975) *Survival Distributions: Reliability Applications in the Biomedical Sciences* Wiley

5 Arguments

1: **x** – double *Input*

On entry: x, the argument of the reciprocal of Mills' Ratio.

6 Error Indicators and Warnings

None.

7 Accuracy

In the left-hand tail, $x < 0.0$, if $\frac{1}{2}e^{-(1/2)x^2} \leq$ the safe range argument (nag_real_safe_small_number (X02AMC)), then 0.0 is returned, which is close to the true value.

The relative accuracy is bounded by the effective ***machine precision***. See nag_erfcx (s15agc) for further discussion.

8 Parallelism and Performance

nag_mills_ratio (g01mbc) is not threaded in any implementation.

9 Further Comments

If, before entry, x is not a standard Normal variable, it has to be standardized, and on exit, nag_mills_ratio (g01mbc) has to be divided by the standard deviation. That is, if the Normal distribution has mean μ and variance σ^2, then its hazard rate, $\lambda\left(x;\mu,\sigma^2\right)$, is given by

$$\lambda\left(x;\mu,\sigma^2\right) = \lambda((x-\mu)/\sigma)/\sigma.$$

10 Example

The hazard rate is evaluated at different values of x for Normal distributions with different means and variances. The results are then printed.

10.1 Program Text

```
/* nag_mills_ratio (g01mbc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg01.h>

int main(void)
{
  /* Scalars */
  Integer exit_status = 0;
  double rm, x, xmu, xsig, z;
  Integer i;

  printf("nag_mills_ratio (g01mbc) Example Program Results\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  printf("\n%2sMean%5sSigma%4sX%8sReciprocal", "", "", "", "");
  printf("\n                                Mills Ratio\n\n");
  for (i = 1; i <= 3; ++i) {
#ifdef _WIN32
    scanf_s("%lf%lf%lf%*[^\n] ", &x, &xmu, &xsig);
#else
    scanf("%lf%lf%lf%*[^\n] ", &x, &xmu, &xsig);
#endif
    z = (x - xmu) / xsig;
    /* nag_mills_ratio (g01mbc).
     * Computes reciprocal of Mills' Ratio
     */
    rm = nag_mills_ratio(z) / xsig;
    printf("%7.4f%2s%7.4f%2s%7.4f%2s%7.4f", xmu, "", xsig, "", x, "", rm);
    printf("\n");
  }

  return exit_status;
}
```

10.2 Program Data

```
nag_mills_ratio (g01mbc) Example Program Data
0.0 0.0 1.0
-2.0 1.0 2.5
10.3 9.0 1.6
```

10.3 Program Results

```
nag_mills_ratio (g01mbc) Example Program Results

  Mean     Sigma    X        Reciprocal
                             Mills Ratio

 0.0000   1.0000   0.0000   0.7979
 1.0000   2.5000  -2.0000   0.0878
 9.0000   1.6000  10.3000   0.8607
```