NAG Library Function Document

nag_quad_md_numth_coeff_2prime (d01gzc)

1 Purpose

nag_quad_md_numth_coeff_2prime (d01gzc) calculates the optimal coefficients for use by
nag_quad_md_numth_vec (d01gdc), when the number of points is the product of two primes.

2 Specification

```c
#include <nag.h>
#include <nagd01.h>

void nag_quad_md_numth_coeff_2prime (Integer ndim, Integer np1, Integer np2,
        double vk[], NagError *fail)
```

3 Description

Korobov (1963) gives a procedure for calculating optimal coefficients for
\( p \)-point integration over the
\( n \)-cube \([0,1]^n\), when the number of points is
\[ p = p_1 p_2 \]

where \( p_1 \) and \( p_2 \) are distinct prime numbers.

The advantage of this procedure is that if \( p_1 \) is chosen to be the nearest prime integer to \( p_2^2 \), then the
number of elementary operations required to compute the rule is of the order of \( p^{8/3} \) which grows less
rapidly than the number of operations required by nag_quad_md_numth_coeff_prime (d01gyc). The
associated error is likely to be larger although it may be the only practical alternative for high values of
\( p \).

4 References

Korobov N M (1963) Number Theoretic Methods in Approximate Analysis Fizmatgiz, Moscow

5 Arguments

1: ndim – Integer  
   **Input**  
   *On entry:* \( n \), the number of dimensions of the integral.  
   *Constraint:* \( ndim \geq 1 \).

2: np1 – Integer  
   **Input**  
   *On entry:* the larger prime factor \( p_1 \) of the number of points in the integration rule.  
   *Constraint:* \( np1 \) must be a prime number \( \geq 5 \).

3: np2 – Integer  
   **Input**  
   *On entry:* the smaller prime factor \( p_2 \) of the number of points in the integration rule. For maximum
efficiency, \( p_2^2 \) should be close to \( p_1 \).  
   *Constraint:* \( np2 \) must be a prime number such that \( np1 > np2 \geq 2 \).

4: vk[ndim] – double  
   **Output**  
   *On exit:* the \( n \) optimal coefficients.
5: \hspace{1em} \texttt{fail} – NagError *

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 \hspace{1em} \textbf{Error Indicators and Warnings}

**NE\_ACCURACY**

The \textit{machine precision} is insufficient to perform the computation exactly. Try reducing \texttt{np1} or \texttt{np2}: \texttt{np1} = \langle value \rangle and \texttt{np2} = \langle value \rangle.

**NE\_ALLOC\_FAIL**

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

**NE\_BAD\_PARAM**

On entry, argument \langle value \rangle had an illegal value.

**NE\_INT**

On entry, \texttt{ndim} = \langle value \rangle.
Constraint: \texttt{ndim} \geq 1.

On entry, \texttt{np1} = \langle value \rangle.
Constraint: \texttt{np1} must be a prime number.

On entry, \texttt{np2} = \langle value \rangle.
Constraint: \texttt{np2} must be a prime number.

On entry, \texttt{np1} \geq 5.
Constraint: \texttt{np1} \geq 5.

On entry, \texttt{np2} = \langle value \rangle.
Constraint: \texttt{np2} \geq 2.

**NE\_INT\_2**

On entry, \texttt{np1} \times \texttt{np2} exceeds largest machine integer. \texttt{np1} = \langle value \rangle and \texttt{np2} = \langle value \rangle.
On entry, \texttt{np1} = \langle value \rangle and \texttt{np2} = \langle value \rangle.
Constraint: \texttt{np1} > \texttt{np2}.

**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 the Essential Introduction 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 the Essential Introduction for further information.

7 \hspace{1em} \textbf{Accuracy}

The optimal coefficients are returned as exact integers (though stored in a double array).

8 \hspace{1em} \textbf{Parallelism and Performance}

Not applicable.
9  Further Comments

The time taken by nag_quad_md_numth_coeff_2prime (d01gzc) grows at least as fast as \( (p_1p_2)^{4/3} \). (See Section 3.)

10  Example

This example calculates the Korobov optimal coefficients where the number of dimensions is 4 and the number of points is the product of the two prime numbers, 89 and 11.

10.1 Program Text

/* nag_quad_md_numth_coeff_2prime (d01gzc) Example Program.  *
* Copyright 2014 Numerical Algorithms Group.  *
* Mark 23, 2011.  */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagd01.h>

int main(void)
{
  Integer exit_status = 0;
  Integer i, ndim, np1, np2;
  double *vk = 0;
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_quad_md_numth_coeff_2prime (d01gzc) Example Program Results\n");
  /* Skip heading in data file */
  #ifdef _WIN32
    scanf_s("%*[\n ]");
  #else
    scanf("%*[\n ]");
  #endif
  #ifdef _WIN32
    scanf_s("%"NAG_IFMT" %"NAG_IFMT"%*[\n ]", &ndim);
  #else
    scanf("%"NAG_IFMT"%*[\n ]", &ndim);
  #endif
  #ifdef _WIN32
    scanf_s("%"NAG_IFMT" %NAG_IFMT"%*[\n ]", &np1, &np2);
  #else
    scanf("%NAG_IFMT"%*[\n ]", &np1, &np2);
  #endif

  if (!(vk = NAG_ALLOC(ndim, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* nag_quad_md_numth_coeff_2prime (d01gzc).  *
  * Korobov optimal coefficients for use in d01gdc,  *
  * when number of points is product of two primes.  *
  */
  nag_quad_md_numth_coeff_2prime(ndim, np1, np2, vk, &fail);
  if (fail.code != NE_NOERROR)
  {
    printf("Error from nag_quad_md_numth_coeff_2prime (d01gzc).\n%s\n", fail.message);
    exit_status = 1;
  }

  return exit_status;
}

Mark 25
10.2 Program Data
None.

10.3 Program Results
nag_quad_md_numth_coeff_2prime (d01gzc) Example Program Results

\[
\begin{align*}
\text{ndim} &= 4 \\
\text{np1} &= 89 \\
\text{np2} &= 11 \\
\text{Coefficients} &= 1 \ 102 \ 614 \ 951
\end{align*}
\]