1 Purpose

nag_quad_1d_gauss_wgen (d01tcc) returns the weights (normal or adjusted) and abscissae for a
Gaussian integration rule with a specified number of abscissae. Six different types of Gauss rule are
allowed.

2 Specification

```c
#include <nag.h>
#include <nagd01.h>
void nag_quad_1d_gauss_wgen (Nag_QuadType quad_type, double a, double b,
     double c, double d, Integer n, double weight[], double abscis[],
     NagError *fail)
```

3 Description

nag_quad_1d_gauss_wgen (d01tcc) returns the weights $w_i$ and abscissae $x_i$ for use in the summation

$$S = \sum_{i=1}^{n} w_i f(x_i)$$

which approximates a definite integral (see Davis and Rabinowitz (1975) or Stroud and Secrest (1966)).

The following types are provided:

(a) Gauss–Legendre

$$S \approx \int_{a}^{b} f(x) \, dx, \quad \text{exact for } f(x) = P_{2n-1}(x).$$

*Constraint:* $b > a$.

(b) Gauss–Jacobi

normal weights:

$$S \approx \int_{a}^{b} (b-x)^c (x-a)^d f(x) \, dx, \quad \text{exact for } f(x) = P_{2n-1}(x),$$

adjusted weights:

$$S \approx \int_{a}^{b} f(x) \, dx, \quad \text{exact for } f(x) = (b-x)^c (x-a)^d P_{2n-1}(x).$$

*Constraint:* $c > -1$, $d > -1$, $b > a$.

(c) Exponential Gauss

normal weights:

$$S \approx \int_{a}^{b} \left| x - \frac{a + b}{2} \right|^e f(x) \, dx, \quad \text{exact for } f(x) = P_{2n-1}(x),$$

adjusted weights:
\[ S \simeq \int_{a}^{b} f(x) \, dx, \quad \text{exact for } f(x) = x - \frac{a + b}{2} P_{2n-1}(x). \]

Constraint: \( c > -1, \ b > a. \)

(d) Gauss–Laguerre

normal weights:

\[ S \simeq \int_{0}^{\infty} |x - a|^e e^{-bx} f(x) \, dx \quad (b > 0), \]

\[ \simeq \int_{-\infty}^{0} |x - a|^e e^{-bx} f(x) \, dx \quad (b < 0), \quad \text{exact for } f(x) = P_{2n-1}(x), \]

adjusted weights:

\[ S \simeq \int_{0}^{\infty} f(x) \, dx \quad (b > 0), \]

\[ \simeq \int_{-\infty}^{0} f(x) \, dx \quad (b < 0), \quad \text{exact for } f(x) = |x - a|^e e^{-bx} P_{2n-1}(x). \]

Constraint: \( c > -1, \ b \neq 0. \)

(e) Gauss–Hermite

normal weights:

\[ S \simeq \int_{-\infty}^{+\infty} |x - a|^e e^{-b(x-a)^2} f(x) \, dx, \quad \text{exact for } f(x) = P_{2n-1}(x), \]

adjusted weights:

\[ S \simeq \int_{-\infty}^{+\infty} f(x) \, dx, \quad \text{exact for } f(x) = |x - a|^e e^{-b(x-a)^2} P_{2n-1}(x). \]

Constraint: \( c > -1, \ b > 0. \)

(f) Rational Gauss

normal weights:

\[ S \simeq \int_{0}^{\infty} \frac{|x-a|^e}{|x+b|^d} f(x) \, dx \quad (a + b > 0), \]

\[ \simeq \int_{-\infty}^{0} \frac{|x-a|^e}{|x+b|^d} f(x) \, dx \quad (a + b < 0), \quad \text{exact for } f(x) = P_{2n-1}\left(\frac{1}{x+b}\right), \]

adjusted weights:

\[ S \simeq \int_{0}^{\infty} f(x) \, dx \quad (a + b > 0), \]

\[ \simeq \int_{-\infty}^{0} f(x) \, dx \quad (a + b < 0), \quad \text{exact for } f(x) = |x-a|^e \frac{|x-a|^e}{|x+b|^d} P_{2n-1}\left(\frac{1}{x+b}\right). \]

Constraint: \( c > -1, \ d > c + 1, \ a + b \neq 0. \)

In the above formulae, \( P_{2n-1}(x) \) stands for any polynomial of degree \( 2n - 1 \) or less in \( x. \)

The method used to calculate the abscissae involves finding the eigenvalues of the appropriate tridiagonal matrix (see Golub and Welsch (1969)). The weights are then determined by the formula
\[ w_i = \left\{ \frac{1}{\sum_{j=0}^{n-1} P_j^*(x_i)^2} \right\}^{-1} \]

where \( P_j^*(x) \) is the \( j \)th orthogonal polynomial with respect to the weight function over the appropriate interval.

The weights and abscissae produced by \texttt{nag_quad_1d_gauss_wgen} (\texttt{d01tcc}) may be passed to \texttt{nag_quad_md_gauss} (\texttt{d01fbc}), which will evaluate the summations in one or more dimensions.

4 References


Stroud A H and Secrest D (1966) \textit{Gaussian Quadrature Formulas} Prentice–Hall

5 Arguments

1: \hspace{0.5em} \textbf{quad_type} – Nag_QuadType \hspace{5em} Input

\textit{On entry}: indicates the type of quadrature rule.

- \texttt{quad_type = Nag_Quad_Gauss_Legendre} (Gauss–Legendre, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Jacobi} (Gauss–Jacobi, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Jacobi_Adjusted} (Gauss–Jacobi, with adjusted weights).
- \texttt{quad_type = Nag_Quad_Gauss_Exponential} (Exponential Gauss, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Exponential_Adjusted} (Exponential Gauss, with adjusted weights).
- \texttt{quad_type = Nag_Quad_Gauss_Laguerre} (Gauss–Laguerre, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Laguerre_Adjusted} (Gauss–Laguerre, with adjusted weights).
- \texttt{quad_type = Nag_Quad_Gauss_Hermite} (Gauss–Hermite, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Hermite_Adjusted} (Gauss–Hermite, with adjusted weights).
- \texttt{quad_type = Nag_Quad_Gauss_Rational} (Rational Gauss, with normal weights).
- \texttt{quad_type = Nag_Quad_Gauss_Rational_Adjusted} (Rational Gauss, with adjusted weights).

\textit{Constraint}: \texttt{quad_type} is one of the above options.

\textit{Mark 25}
2:  a – double  
3:  b – double  
4:  c – double  
5:  d – double  

On entry: the parameters \(a\), \(b\), \(c\) and \(d\) which occur in the quadrature formulae. \(c\) is not used if quad\_type = Nag\_Quad\_Gauss\_Legendre; \(d\) is not used unless quad\_type = Nag\_Quad\_Gauss\_Jacobi, Nag\_Quad\_Gauss\_Jacobi\_Adjusted, Nag\_Quad\_Gauss\_Rational or Nag\_Quad\_Gauss\_Rational\_Adjusted. For some rules \(c\) and \(d\) must not be too large (see Section 6).

Constraints:

if quad\_type = Nag\_Quad\_Gauss\_Legendre, \(a < b\);  
if quad\_type = Nag\_Quad\_Gauss\_Jacobi or Nag\_Quad\_Gauss\_Jacobi\_Adjusted,  
\(a < b\) and \(c > -1.0\) and \(d > -1.0\);  
if quad\_type = Nag\_Quad\_Gauss\_Exponential or Nag\_Quad\_Gauss\_Exponential\_Adjusted,  
\(a < b\) and \(c > -1.0\);  
if quad\_type = Nag\_Quad\_Gauss\_Laguerre or Nag\_Quad\_Gauss\_Laguerre\_Adjusted,  
\(b \neq 0.0\) and \(c > -1.0\);  
if quad\_type = Nag\_Quad\_Gauss\_Hermite or Nag\_Quad\_Gauss\_Hermite\_Adjusted,  
\(b > 0.0\) and \(c > -1.0\);  
if quad\_type = Nag\_Quad\_Gauss\_Rational or Nag\_Quad\_Gauss\_Rational\_Adjusted,  
\(a + b \neq 0.0\) and \(c > -1.0\) and \(d > c + 1.0\).

6:  n – Integer  

On entry: \(n\), the number of weights and abscissae to be returned. If quad\_type = Nag\_Quad\_Gauss\_Exponential\_Adjusted or Nag\_Quad\_Gauss\_Hermite\_Adjusted and \(c \neq 0.0\), an odd value of \(n\) may raise problems (see fail\_code = NE_INDETERMINATE).

Constraint: \(n > 0\).

7:  weight[n] – double  

On exit: the \(n\) weights.

8:  abscis[n] – double  

On exit: the \(n\) abscissae.

9:  fail – NagError *  

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

6  Error Indicators and Warnings

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 \(<value>\) had an illegal value.

NE_CONSTRAINT

On entry, \(a\), \(b\), \(c\), or \(d\) is not in the allowed range: \(a = \langle value\rangle\), \(b = \langle value\rangle\) \(c = \langle value\rangle\), \(d = \langle value\rangle\) and quad\_type = \langle value\rangle.
**NE_CONVERGENCE**

The algorithm for computing eigenvalues of a tridiagonal matrix has failed to converge.

**NE_INDETERMINATE**

Exponential Gauss or Gauss–Hermite adjusted weights with \( n \) odd and \( c \neq 0.0 \).

Theoretically, in these cases:

- for \( c > 0.0 \), the central adjusted weight is infinite, and the exact function \( f(x) \) is zero at the central abscissa;
- for \( c < 0.0 \), the central adjusted weight is zero, and the exact function \( f(x) \) is infinite at the central abscissa.

In either case, the contribution of the central abscissa to the summation is indeterminate.

In practice, the central weight may not have overflowed or underflowed, if there is sufficient rounding error in the value of the central abscissa.

The weights and abscissa returned may be usable; you must be particularly careful not to ‘round’ the central abscissa to its true value without simultaneously ‘rounding’ the central weight to zero or \( \infty \) as appropriate, or the summation will suffer. It would be preferable to use normal weights, if possible.

**Note:** remember that, when switching from normal weights to adjusted weights or vice versa, redefinition of \( f(x) \) is involved.

**NE_INT**

On entry, \( n = \langle \text{value} \rangle \).

Constraint: \( n > 0 \).

**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.

**NE_TOO_BIG**

One or more of the weights are larger than \( r_{\text{max}} \), the largest floating point number on this computer (see nag_real_largest_number (X02ALC)): \( r_{\text{max}} = \langle \text{value} \rangle \).

Possible solutions are to use a smaller value of \( n \); or, if using adjusted weights to change to normal weights.

**NE_TOO_SMALL**

One or more of the weights are too small to be distinguished from zero on this machine.

The underflowing weights are returned as zero, which may be a usable approximation.

Possible solutions are to use a smaller value of \( n \); or, if using normal weights, to change to adjusted weights.

**7 Accuracy**

The accuracy depends mainly on \( n \), with increasing loss of accuracy for larger values of \( n \). Typically, one or two decimal digits may be lost from machine accuracy with \( n \approx 20 \), and three or four decimal digits may be lost for \( n \approx 100 \).
8 Parallelism and Performance

nag_quad_1d_gauss_wgen (d01tcc) is not threaded by NAG in any implementation.

nag_quad_1d_gauss_wgen (d01tcc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.

9 Further Comments

The major portion of the time is taken up during the calculation of the eigenvalues of the appropriate tridiagonal matrix, where the time is roughly proportional to $n^3$.

10 Example

This example returns the abscissae and (adjusted) weights for the seven-point Gauss–Laguerre formula.

10.1 Program Text

```c
/* nag_quad_1d_gauss_wgen (d01tcc) 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, n;
    double a, b, c, d;
    Nag_QuadType quadtype;
    NagError fail;
    double *abscis = 0, *weight = 0;
    INIT_FAIL(fail);
    printf("nag_quad_1d_gauss_wgen (d01tcc) Example Program Results\n");
    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*[\n "");
    #else
    scanf("%*[\n "");
    #endif
    /* Input a, b, c, d and n */
    #ifdef _WIN32
    scanf_s("%lf %lf %lf %lf", &a, &b, &c, &d);
    #else
    scanf("%lf %lf %lf %lf", &a, &b, &c, &d);
    #endif
    #ifdef _WIN32
    scanf_s("%NAG_IFMT%*[\n "", &n);
    #else
    scanf("%NAG_IFMT%*[\n "", &n);
    #endif
    quadtype = Nag_Quad_Gauss_Laguerre_Adjusted;
    if (!(abscis = NAG_ALLOC(n, double)) ||
```
(! (weight = NAG_ALLOC(n, double)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

/* nag_quad_1d_gauss_wgen (d01tcc).
   * Calculation of weights and abscissae for
   * Gaussian quadrature rules, general choice of rule.
   */

nag_quad_1d_gauss_wgen(quadtype, a, b, c, d, n, weight, abscis, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_quad_1d_gauss_wgen (d01tcc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

printf("\nLaguerre formula, %3" NAG_IFMT " points\n" "
Abscissae Weights\n", n);
for (i = 0; i < n; i++)
{
    printf("%15.5e", abscis[i]);
    printf("%15.5e\n", weight[i]);
}

printf("\n");

END:
NAG_FREE(abscis);
NAG_FREE(weight);

return exit_status;

10.2 Program Data

nag_quad_1d_gauss_wgen (d01tcc) Example Program Data
0.0 1.0 0.0 0.0

10.3 Program Results

nag_quad_1d_gauss_wgen (d01tcc) Example Program Results

Laguerre formula, 7 points

Abscissae Weights
1.93044e-01 4.96478e-01
1.02666e+00 1.17764e+00
2.56788e+00 1.91825e+00
4.90035e+00 2.77185e+00
8.18215e+00 3.84125e+00
1.27342e+01 5.38068e+00
1.93957e+01 8.40543e+00

Mark 25 d01tcc.7
Example Program

Abscissae and Weights for the 7-point Gauss-Laguerre Formula ($a=0, b=1$)