NAG C Library Function Document

nag_legendre_p (s22aac)

1 Purpose

nag_legendre_p (s22aac) returns a sequence of values for either the unnormalized or normalized Legendre functions of the first kind \( P_n^m(x) \) or \( \overline{P}_n^m(x) \) for real \( x \) of a given order \( m \) and degree \( n = 0, 1, \ldots, N \).

2 Specification

```c
void nag_legendre_p (Integer mode, double x, Integer m, Integer nl, double p[],
                 NagError *fail)
```

3 Description

This routine evaluates a sequence of values for either the unnormalized or normalized Legendre \((m = 0)\) or associated Legendre \((m \neq 0)\) functions of the first kind \( P_n^m(x) \) or \( \overline{P}_n^m(x) \), where \( x \) is real with \(-1 \leq x \leq 1\), of order \( m \) and degree \( n = 0, 1, \ldots, N \) defined by

\[
P_n^m(x) = (1 - x^2)^{m/2} \frac{d^m}{dx^m} P_n(x) \text{ if } m \geq 0,
\]

\[
P_n^m(x) = \frac{(n + m)!}{(n - m)!} P_n^m(x) \text{ if } m < 0 \text{ and}
\]

\[
\overline{P}_n^m(x) = \sqrt{\frac{(2n + 1)(n - m)!}{2(n + m)!}} P_n^m(x)
\]

respectively; \( P_n(x) \) is the (unassociated) Legendre polynomial of degree \( n \) given by

\[
P_n(x) \equiv P_n^0(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n}(x^2 - 1)^n
\]

(the Rodrigues formula). Note that some authors (e.g., Abramowitz and Stegun (1972)) include an additional factor of \((-1)^m\) (the Condon-Shortley Phase) in the definitions of \( P_n^m(x) \) and \( \overline{P}_n^m(x) \). They use the notation \( P_m^m(x) \equiv (-1)^m P_n^m(x) \) in order to distinguish between the two cases.

nag_legendre_p is based on a standard recurrence relation given by Abramowitz and Stegun (Abramowitz and Stegun (1972), 8.5.3). Constraints are placed on the values of \( m \) and \( n \) in order to avoid the possibility of machine overflow. It also sets the appropriate elements of the array \( p \) (see Section 4) to zero whenever the required function is not defined for certain values of \( m \) and \( n \) (e.g., \( m = -5 \) and \( n = 3 \)).

4 Parameters

1: \textbf{mode} – Integer \textit{Input}

\textit{On entry:} indicates whether the sequence of function values is to be returned unnormalized or normalized as follows:

- if \textbf{mode} = 1, then the sequence of function values is returned unnormalized;
- if \textbf{mode} = 2, then the sequence of function values is returned normalized.

\textit{Constraint:} \textbf{mode} = 1 or 2.

2: \( x \) – double \textit{Input}

\textit{On entry:} the argument \( x \) of the function.

\textit{Constraint:} abs(\( x \)) \leq 1.0.
3: \( m \) – Integer

\( m \) is the order of the function.

**Input**

\( \text{Constraint: } \left| m \right| \leq 27. \)

4: \( n \) – Integer

\( n \) is the degree \( N \) of the last function required in the sequence.

**Input**

\( \text{Constraints:} \)

\( n \geq 0, \)

\( n \leq 100 \text{ when } m = 0, \)

\( n \leq 55 - \left| m \right| \text{ when } m \neq 0. \)

5: \( p[nl+1] \) – double

\( p[nl+1] \) is the required sequence of function values as follows:

- if \( \text{mode} = 1 \), \( p(n) \) contains \( P_n^m(x) \) for \( n = 0, 1, \ldots, N; \)
- if \( \text{mode} = 2 \), \( p(n) \) contains \( P_n^m(x) \) for \( n = 0, 1, \ldots, N. \)

**Output**

6: \( \text{fail} \) – NagError *

\( \text{fail} \) is the NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

**NE_REAL**

\( \text{On entry, } x = <value>. \)

\( \text{Constraint: } \left| x \right| \leq 1.0. \)

**NE_INT**

\( \text{On entry, } \text{mode} = <value>. \)

\( \text{Constraint: } \text{mode} \leq 1 \text{ or } 2. \)

\( \text{On entry, } n = <value>. \)

\( \text{Constraint: } n \geq 0. \)

\( \text{On entry, } m = <value>. \)

\( \text{Constraint: } \left| m \right| \leq 27. \)

**NE_INT_2**

\( \text{On entry, } n = <value>, m = <value>. \)

\( \text{Constraint: } n \leq 100 \text{ when } m = 0. \)

\( \text{On entry, } n = <value>, m = <value>. \)

\( \text{Constraint: } n \leq 55 - \left| m \right| \text{ when } m \neq 0. \)

6 Further Comments

6.1 Accuracy

The computed function values should be accurate to within a small multiple of the machine precision except when underflow (or overflow) occurs, in which case the true function values are within a small multiple of the underflow (or overflow) threshold of the machine.
6.2 References


7 See Also

None.

8 Example

The following program reads the values of the arguments $x$, $m$ and $N$ from a file, calculates the sequence of unnormalized associated Legendre function values $P_n^m(x)$, $P_{n+1}^m(x)$, ..., $P_{n+N}^m(x)$, and prints the results.

8.1 Program Text

```c
/* nag_legendre_p (s22aac) Example Program. 
 * Copyright 2000 Numerical Algorithms Group. 
 * NAG C Library 
 * Mark 6, 2000. */

#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
    const char fmt_99998[] = "%2d %12.4e\n";
    const char fmt_99999[] = "%3d %5.1f%6d%6d\n\n";
    char str[80];
    double p[101];
    double x;
    Integer exit_status=0;
    NagError fail;
    Integer m, mode, n, nl;

    INIT_FAIL(fail);
    Vprintf("s22aac Example Program Results\n\n");

    /* Skip heading in data file */
    Vscanf("%*[\n ]");
    Vscanf("%ld %lf %ld %ld", &mode, &x, &m, &nl);
    if (mode == 1)
    {
        if (m == 0)
        {
            Vstrcpy(str, "Unnormalized Legendre function values\n");
        }
        else
        {
            Vstrcpy(str, "Unnormalized associated Legendre function values\n");
        }
        else if (mode == 2)
        {
```
if (m == 0)
{
    Vstrncpy(str, "Normalized Legendre function values\n");
}
else
{
    Vstrncpy(str, "Normalized associated Legendre function values\n");
}

s22aac (mode, x, m, nl, p, &fail);
Vprintf("mode x m nl\n\n";
Vprintf(fmt_99999, mode, x, m, nl);

if (fail.code == NE_NOERROR)
{
    Vprintf(str);
    Vprintf("\n"); 
    Vprintf(" n P(n)\n");
    for (n = 0; n <= nl; ++n)
    {
        Vprintf(fmt_99998,n,p[n]);
    }
}
else
{
    Vprintf("Error from s22aac.\n\n", fail.message);
    exit_status = 1;
    goto END;
}
END:
    return exit_status;

8.2 Program Data
s22aac Example Program Data
1 0.5 2 3 : Values of mode, x, m and nl

8.3 Program Results
s22aac Example Program Results

mode x m nl
1 0.5 2 3

Unnormalized associated Legendre function values

n P(n)
0 0.0000e+00
1 0.0000e+00
2 2.2500e+00
3 5.6250e+00