NAG C Library Function Document

nag_jacobian_elliptic (s21cbc)

1 Purpose

nag_jacobian_elliptic (s21cbc) evaluates the Jacobian elliptic functions \( sn, cn \) and \( dn \) for a complex argument \( z \).

2 Specification

void nag_jacobian_elliptic (Complex z, double ak2, Complex *sn, Complex *cn,
                  Complex *dn, NagError *fail)

3 Description

This routine evaluates the Jacobian elliptic functions \( sn(z|k) \), \( cn(z|k) \) and \( dn(z|k) \) given by

\[
\begin{align*}
    sn(z|k) & = \sin \phi \\
    cn(z|k) & = \cos \phi \\
    dn(z|k) & = \sqrt{1 - k^2 \sin^2 \phi},
\end{align*}
\]

where \( z \) is a complex argument, \( k \) is a real parameter (the modulus) with \( k^2 \leq 1 \) and \( \phi \) (the amplitude of \( z \)) is defined by the integral

\[
z = \int_0^\phi \frac{d\theta}{\sqrt{1 - k^2 \sin^2 \theta}}.
\]

The above definitions can be extended for values of \( k^2 > 1 \) (see Salzer (1962)) by means of the formulae

\[
\begin{align*}
    sn(z|k) &= k_1 sn(kz|k_1) \\
    cn(z|k) &= dn(kz|k_1) \\
    dn(z|k) &= cn(kz|k_1)
\end{align*}
\]

where \( k_1 = 1/k \).

Special values include

\[
\begin{align*}
    sn(z|0) &= \sin z \\
    cn(z|0) &= \cos z \\
    dn(z|0) &= 1 \\
    sn(z|1) &= \tanh z \\
    cn(z|1) &= \text{sech} z \\
    dn(z|1) &= \text{sech} z.
\end{align*}
\]

These functions are often simply written as \( sn, cn \) and \( dn \), thereby avoiding explicit reference to the parameter \( k \). They can also be expressed in terms of Jacobian theta functions (see nag_jacobian_theta (s21ccc)).

Another nine elliptic functions may be computed via the formulae

\[
\begin{align*}
    cdz &= cnz/dnz \\
    sdz &= snz/dnz \\
    ndz &= 1/dnz
\end{align*}
\]
dcz = dnz/cnz
ncz = 1/cnz
scz = snz/cnz
nsz = 1/snz
dsz = dnz/snz
csz = cnz/snz

(see Abramowitz and Stegun (1972)).

4 Parameters

1: \textbf{z} – Complex\hfill\textit{Input}

\textit{On entry:} the argument \(z\) of the functions.

\textit{Constraints:}

\begin{align*}
\text{abs}(z, \text{re}) & \leq \sqrt{\lambda}, \\
\text{abs}(z, \text{im}) & \leq \sqrt{\lambda}, \text{ where } \lambda = 1/X02AMC.
\end{align*}

2: \textbf{ak2} – double\hfill\textit{Input}

\textit{On entry:} the value of \(k^2\).

\textit{Constraint:} \(0.0 \leq \text{ak2} \leq 1.0\).

3: \textbf{sn} – Complex *\hfill\textit{Output}

4: \textbf{cn} – Complex *\hfill\textit{Output}

5: \textbf{dn} – Complex *\hfill\textit{Output}

\textit{On exit:} the values of the functions \(snz, cnz\) and \(dnz\), respectively.

6: \textbf{fail} – NagError *\hfill\textit{Input/Output}

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

\textbf{NE_REAL}

\textit{On entry, ak2 = \textit{value}.}

\textit{Constraint:} \(0.0 \leq \text{ak2} \leq 1.0\).

\textbf{NE_COMPLEX}

\textit{On entry, z = (\textit{value}, \textit{value}).}

\textit{Constraint: abs(z.re) \leq \lambda \text{ and abs(z.im) \leq \lambda, where } \lambda = 1/\sqrt{X02AMC}.}

\textbf{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 consult NAG for assistance.
6 Further Comments

The values of $sn_z$, $cn_z$ and $dn_z$ are computed via the formulae

$$sn_z = \frac{sn(u, k)dn(v, k')}{1 - dn^2(u, k)sn^2(v, k')} + i \frac{cn(u, k)dn(u, k)sn(v, k')cn(v, k')}{1 - dn^2(u, k)sn^2(v, k')}$$

$$cn_z = \frac{cn(u, k)cn(v, k')}{1 - dn^2(u, k)sn^2(v, k')} - i \frac{sn(u, k)dn(u, k)sn(v, k')dn(v, k')}{1 - dn^2(u, k)sn^2(v, k')}$$

$$dn_z = \frac{dn(u, k)cn(v, k')dn(v, k')}{1 - dn^2(u, k)sn^2(v, k')} - i \frac{k^2 sn(u, k)cn(u, k)sn(v, k')}{1 - dn^2(u, k)sn^2(v, k')}$$

where $z = u + iv$ and $k' = \sqrt{1 - k^2}$ (the complementary modulus).

6.1 Accuracy

In principle the routine is capable of achieving full relative precision in the computed values. However, the accuracy obtainable in practice depends on the accuracy of the C standard library elementary functions such as sin and cos.

6.2 References


Salzer H E (1962) Quick calculation of Jacobian elliptic functions Comm. ACM 5 399

7 See Also

nag_jacobian_theta (s21ccc)

8 Example

The example program evaluates $sn_z$, $cn_z$ and $dn_z$ at $z = -2.0 + 3.0i$ when $k = 0.5$, and prints the results.

8.1 Program Text

/ * nag_jacobian_elliptic (s21cb) Example Program.  
*  
* Copyright 2000 Numerical Algorithms Group.  
*  
* NAG C Library  
*  
* Mark 6, 2000.  
*  
*/

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

int main(void)
{
    Complex cn, dn, sn, z;
    double ak2;
    Integer exit_status=0;
    NagError fail;

    [NP3491/6]
INIT_FAIL(fail);
Vprintf("s21cbc Example Program Results\n\n");

/* Skip heading in data file */
Vscanf("%*[\n"]");
while (scanf(" (%lf,%lf) %lf%*[\n] ", &z.re, &z.im, &ak2) != EOF)
{
    Vprintf("\n");
    s21cbc (z, ak2, &sn, &cn, &dn, &fail);
    Vprintf(" %8.4f,%8.4f %10.2f\n", z.re, z.im, ak2);
    if (fail.code == NE_NOERROR)
    {
        Vprintf(" sn 
        " cn 
        " dn\n");
        Vprintf(" %8.4f,%8.4f ", sn.re, sn.im);
        Vprintf(" %8.4f,%8.4f ", cn.re, cn.im);
        Vprintf(" %8.4f,%8.4f\"", dn.re, dn.im);
        Vprintf("\n");
    }
    else
    {
        Vprintf("Error from s21cbc.\n\n", fail.message);
        exit_status = 1;
        goto END;
    }
}
END:
return exit_status;
}

8.2 Program Data
s21cbc Example Program Data
(-2.0, 3.0)  0.25 : Values of z and ak2

8.3 Program Results
s21cbc Example Program Results

<table>
<thead>
<tr>
<th>z</th>
<th>ak2</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2.0000</td>
<td>0.25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sn</th>
<th>cn</th>
<th>dn</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.5865</td>
<td>0.2456</td>
<td>( 0.3125, 1.2468)</td>
</tr>
</tbody>
</table>