NAG C Library Chapter Introduction

x02 – Machine Constants

Contents

1 Scope of the Chapter

This chapter provides machine constants required by other functions within the Library. Most of these constants are *not* functions, but they are defined in the header file `<nagx02.h>`. Defined constant names are specified in upper case characters, and functions in lower case. Those machine constants which are defined as functions have also been given upper case names using #define in `<nagx02.h>` and users are recommended to use the upper case form if calling these functions.

This will enhance the portability of calling programs as some computer systems may have a NAG library implementation where a #define of the machine constant has replaced the function definition. There are no individual function documents in this chapter.

2 Background

The chapter is concerned with *parameters* which characterise certain aspects of the computing environment in which the NAG C Library is implemented. They relate primarily to floating-point arithmetic, but also to integer arithmetic, the elementary functions, exception handling and paging. The values of the parameters vary from one implementation to another, but within the context of a single implementation they are constants.

3 A Model of Floating-Point Arithmetic

In order to characterise the important properties of floating-point arithmetic by means of a small number of parameters, NAG uses a simple *model* of floating-point arithmetic. The parameters of the model can be chosen to provide a sufficiently close description of the behaviour of actual implementations of floating-point arithmetic, but not, in general, an exact description; actual implementations vary too much in the details of how numbers are represented or arithmetic operations are performed.

The model is characterised by four integer parameters and one logical parameter. The four integer parameters are:

b: the base
p: the precision (i.e., the number of significant base b digits)
$e_{\min}$: the minimum exponent
$e_{\max}$: the maximum exponent.

These parameters define a set of numerical values of the form

$$f \times b^e$$

where the exponent e must lie in the range $[e_{\min}, e_{\max}]$, and the fraction f (sometimes called the *mantissa* or *significand*) lies in the range $[1/b, 1)$, and may be written

$$f = 0.f_1 f_2 \cdots f_p.$$

Thus f is a p digit fraction to the base b. The f_i are the base b digits of the fraction; they are integers in the range 0 to $b-1$, and the leading digit f_1 must not be zero.

The set of values so defined (together with zero) are called *model numbers*. The model must obey certain rules for the computed results of the following basic arithmetic operations: addition, subtraction, multiplication, negation, absolute value, and comparisons. The rules depend on the value of the logical parameter *rounds*.

If *rounds* is *true*, then the computed result must be the nearest model number to the exact result (assuming that overflow or underflow does not occur); if the exact result is mid-way between two model numbers, then it may be rounded either way.

If *rounds* is *false*, then: if the exact result is a model number, the computed result must be equal to the exact result; otherwise the computed result may be either of the adjacent model numbers on either side of the exact result.

For division and square root, this latter rule is further relaxed (regardless of the value of *rounds*): the computed result may also be one of the next adjacent model numbers on either side of the permitted values just stated.

On some machines, the full set of representable floating-point numbers conforms to the rules of the model with appropriate values of b, p, $e_{\min}$, $e_{\max}$ and *rounds*. For other machines, values of the parameters must be chosen which define a large subset of the representable numbers.

4 Derived Parameters of Floating-Point Arithmetic

Most numerical algorithms require access not to the basic parameters described above, but to certain derived values, of which the most important are:

the ***machine precision***, $\epsilon = \frac{1}{2}b^{1-p}$ if *rounds* is *true*, or $\epsilon = b^{1-p}$ otherwise
(note that this value may be increased very slightly to ensure that $1 + \epsilon > 1$);

the smallest positive model number $= b^{e_{\min}-1}$;

the largest positive model number $= (1 - b^{-p})b^{e_{\max}}$.

One additional derived value is used in the Library. Its definition depends not only on the properties of the basic arithmetic operations just considered, but also on properties of some of the elementary functions. We define the *safe range* parameter to be the smallest positive number z such that for any x in the range $[z, 1/z]$ the following can be computed without undue loss of accuracy, overflow, underflow or other error:

$-x$;

$1/x$;

$-1/x$;

$\sqrt{x}$;

$\log x$;

$\exp(\log x)$;

$y^{\log x/\log y}$ for any y.

In a similar fashion we define the safe range parameter for complex arithmetic as the smallest positive model number z such that for any x in the range $[z,\frac{1}{2}]$ the following can be computed without any loss of accuracy, overflow, underflow or other error:

$-\omega$;

$1/\omega$;

$-1/\omega$;

$\sqrt{\omega}$;

$\log \omega$;

$\exp(\log \omega)$;

$y^{\log \omega/\log y}$ for any y;

$|\omega|$ where ω is any of x, ix, $x + ix$, $1/x$ i/x, $1/x + i/x$ and i is the square root of -1.

This parameter was introduced to take into account the quality of the complex arithmetic on the machine (currently using NAG routines, see Chapter a00).

5 Available Functions

X02AJC Machine precision

X02AKC Smallest positive model number

X02ALC Largest positive model number

X02AMC Safe range of floating-point arithmetic

X02ANC Safe range of NAG complex floating-point arithmetic

X02AHC Largest permissible argument for `sin` and `cos` functions

X02BBC Largest representable integer

X02BEC Maximum number of decimal digits that can be represented

X02BHC Parameter b of model of floating-point arithmetic

X02BJC Parameter p of model of floating-point arithmetic

X02BKC Parameter $e_{\min}$ of model of floating-point arithmetic

X02BLC Parameter $e_{\max}$ of model of floating-point arithmetic

X02DAC Switch for taking precautions to avoid underflow

X02DJC Parameter ROUNDS of model of floating-point arithmetic