# NAG Library Function Document

# nag_bessel_k0_scaled_vector (s18cqc)

## 1 Purpose

nag_bessel_k0_scaled_vector (s18cqc) returns an array of values of the scaled modified Bessel function $e^x K_0(x)$.

## 2 Specification

```
#include <nag.h>
#include <nags.h>
void nag_bessel_k0_scaled_vector (Integer n, const double x[], double f[],
     Integer ivalid[], NagError *fail)
```

## 3 Description

nag_bessel_k0_scaled_vector (s18cqc) evaluates an approximation to $e^{x_i} K_0(x_i)$, where $K_0$ is a modified Bessel function of the second kind for an array of arguments $x_i$, for $i = 1, 2, \ldots, n$. The scaling factor $e^x$ removes most of the variation in $K_0(x)$.

The function uses the same Chebyshev expansions as nag_bessel_k0_vector (s18aqc), which returns an array of the unscaled values of $K_0(x)$.

## 4 References

Abramowitz M and Stegun I A (1972) *Handbook of Mathematical Functions* (3rd Edition) Dover Publications

## 5 Arguments

1: **n** – Integer *Input*

*On entry*: $n$, the number of points.

*Constraint*: $\mathbf{n} \geq 0$.

2: **x**[**n**] – const double *Input*

*On entry*: the argument $x_i$ of the function, for $i = 1, 2, \ldots, \mathbf{n}$.

*Constraint*: $\mathbf{x}[i-1] > 0.0$, for $i = 1, 2, \ldots, \mathbf{n}$.

3: **f**[**n**] – double *Output*

*On exit*: $e^{x_i} K_0(x_i)$, the function values.

4: **ivalid**[**n**] – Integer *Output*

*On exit*: **ivalid**$[i-1]$ contains the error code for $x_i$, for $i = 1, 2, \ldots, \mathbf{n}$.

**ivalid**$[i-1] = 0$
No error.

**ivalid**$[i-1] = 1$
On entry, $x_i \leq 0.0$, $K_0(x_i)$ is undefined. $\mathbf{f}[i-1]$ contains 0.0.

5: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

**NE_BAD_PARAM**

On entry, argument ⟨*value*⟩ had an illegal value.

**NE_INT**

On entry, **n** = ⟨*value*⟩.
Constraint: **n** $\geq 0$.

**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 2.7.6 in How to Use the NAG Library and its Documentation for further information.

**NE_NO_LICENCE**

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

**NW_IVALID**

On entry, at least one value of **x** was invalid.
Check **ivalid** for more information.

## 7 Accuracy

Relative errors in the argument are attenuated when propagated into the function value. When the accuracy of the argument is essentially limited by the ***machine precision***, the accuracy of the function value will be similarly limited by at most a small multiple of the ***machine precision***.

## 8 Parallelism and Performance

nag_bessel_k0_scaled_vector (s18cqc) is not threaded in any implementation.

## 9 Further Comments

None.

## 10 Example

This example reads values of **x** from a file, evaluates the function at each value of $x_i$ and prints the results.

## 10.1 Program Text

```
/* nag_bessel_k0_scaled_vector (s18cqc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
  Integer exit_status = 0;
  Integer i, n;
  double *f = 0, *x = 0;
  Integer *ivalid = 0;
  NagError fail;

  INIT_FAIL(fail);

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  printf("nag_bessel_k0_scaled_vector (s18cqc) Example Program Results\n");
  printf("\n");
  printf("     x           f          ivalid\n");
  printf("\n");
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "", &n);
#else
  scanf("%" NAG_IFMT "", &n);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  /* Allocate memory */
  if (!(x = NAG_ALLOC(n, double)) ||
      !(f = NAG_ALLOC(n, double)) || !(ivalid = NAG_ALLOC(n, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  for (i = 0; i < n; i++)
#ifdef _WIN32
    scanf_s("%lf", &x[i]);
#else
    scanf("%lf", &x[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  /* nag_bessel_k0_scaled_vector (s18cqc).
   * Scaled Bessel function K0(x)
   */
```

```
  nag_bessel_k0_scaled_vector(n, x, f, ivalid, &fail);
  if (fail.code != NE_NOERROR && fail.code != NW_IVALID) {
    printf("Error from nag_bessel_k0_scaled_vector (s18cqc).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
  }

  for (i = 0; i < n; i++)
    printf(" %11.3e %11.3e %4" NAG_IFMT "\n", x[i], f[i], ivalid[i]);

END:
  NAG_FREE(f);
  NAG_FREE(x);
  NAG_FREE(ivalid);

  return exit_status;
}
```

## 10.2 Program Data

```
nag_bessel_k0_scaled_vector (s18cqc) Example Program Data

6

0.4 0.6 1.4 2.5 10.0 1000.0
```

## 10.3 Program Results

```
nag_bessel_k0_scaled_vector (s18cqc) Example Program Results

     x           f          ivalid

   4.000e-01   1.663e+00    0
   6.000e-01   1.417e+00    0
   1.400e+00   9.881e-01    0
   2.500e+00   7.595e-01    0
   1.000e+01   3.916e-01    0
   1.000e+03   3.963e-02    0
```