Chapter 21

Errata – *Numerical Computing with Modern Fortran*

Changes in the text are noted by chapter. Most of these were found by W. Van Snyder. We thank him!

Chapter 1

Page 5, line *last* − 7 **::** change word “type” to “object or structure”

Chapter 2

Page 13, Section 2.1, line 1 **::** change “three compilation units” to “four compilation units”

Page 13, Section 2.1, line 3 **::** change “Subroutines and functions” to “Subroutines, functions, block data and”

Page 20, line *last* − 8 **::** change “mandatory extended precision” to “mandatory double precision”

Page 22, line *last* − 5 **::** change “invoke the module” to “access the module”

Page 23, line 2 **::** change “data and program units” to “data, type definitions and procedures”

Page 23, line 24 **::** change “type definition within” to “type definition and literal constants within”

Chapter 4

Page 40, lines 15 **and** 19 **::** replace the word “parameters” by the word “arguments”

Page 42, line 4 **::** change “standard Fortran names” to “letters and numbers”

Page 43, lines 1, 2 **and** 4 **::** replace “parameter” and “parameters” by “argument” and “arguments” respectively in four places.

Chapter 5

Page 48, lines 20 **and** 31 **::** replace “TYPE (date)” by “TYPE date”

Page 50, line *last* − 4 **::** replace “TYPE (date)” by “TYPE date”

Page 51, offset equation ::

$$\mathbf{y}' = \frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{x})$$

should read

$$\mathbf{y}' = \frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y})$$

Page 51, lines 14, 16 **and** 19 **::** change “x” to “y”

Chapter 6

Page 58, line *last* − 3 **::** change “DOUBLE PRECISION :: t” to “REAL(wp) :: t”

Page 59, line *last* − 4 **::** insert the line “! Recursive Quicksort routine follows” before “...”

Page 65, line 15 **::** change “*compareVal routine*” to “*compareValue routine*” in the caption to Listing 6.7

Page 66, lines 6 **and** 15 **::** replace “Descending” by “Ascending” in two places

Page 67, line 9 **::** change “definitions” to “declarations”

Page 67, line 10/11 **::** replace “somewhere else in the code” by “in an extension of the type.”

Page 67, line 12 **::** delete the word “data” from the caption to Listing 6.9

Chapter 7

Page 71, line *last* − 3 **::** remove “,” after “77”

Page 82, line 18 **::** replace “CASE (0)” by “CASE (inner)”

Page 82, line 21 **::** replace “CASE (1)” by “CASE (outer)”

Page 82, line 23 **::** delete “extended”

Page 83, line *last* − 17 **::** replace “CASE (0)” in the comment by “CASE (inner)”

Chapter 8

Page 90, line *last* − 11 **::** delete “now”, “parameter” and “been”

Page 90, line *last* − 10 **::** delete “considered to be” and “defaults”
replace “preassigned” by “default initial”

Page 95, line 11 **::** replace “parameters” by “arguments”

Page 95, Listing 8.3, line 8 **::** delete “,wp” (The symbol “wp” is defined two lines earlier.)

Page 95, Listing 8.3, line 19 **::** replace the comment “! This improves” by “! An ASSOCIATE construct improves”

Page 96, lines 15 **and** *last* − 5 **::** replace “[mu-1,m+1]” by “[mu-1,mu+1]” in two places.

Chapter 9

Page 102, line 25 **::** replace “elemental functions” by “intrinsic functions”

Page 103, line *last* − 4 **::** replace “t = dmrm2” by “t = dnrm2”

Page 111, line 1 **::** replace “EXIT block” by “EXIT block ! This is the label of the encompassing loop.”

Page 112, last paragraph :: Note for gfortran users: The IEEE modules are now available for some environments. This eliminates the need for use of these modules found in the distributed codes.
See `https://gcc.gnu.org/onlinedocs/gfortran/IEEE-modules.html`.

Page 113, line *last* − 6 **::** replace “gradual underflow,” by “gradual underflow because it is slow and difficult,”

Chapter 10

Page 118, line 12 **::** replace “letter” by “letters”

Page 118, line 15 **::** replace “calling a” by “the calling”

Page 118, line 12 **::** replace “name used” by “name”

Page 120, lines 12 **and** *last* − 14 **::** replace “parameters” by “arguments” in two places

Page 120, lines 22/23 **::** replace the sentence “The argument definitions ... definition.” by “The argument definitions in the communication routine correspond to the following C prototype definition.”

Page 121, line *last* − 10 **::** replace "CHARACTER (c_char)" by "CHARACTER (kind=c_char)"

Page 122, line 6 **::** replace "parameters" by "arguments"

Page 122, line *last* − 14 **::** replace "CHARACTER (c_char)" by "CHARACTER (kind=c_char)"

Page 123, line 1 **::** delete the line "! The DO WHILE construct has been deprecated"

Page 123, lines 2/3 **::** replace the **DO** and **IF** statements by "DO WHILE (cstring(clen+1) /= c_null_char)"

Page 123, line *last* − 13 **::** replace "i.e" by "e.g."

Page 124, line 24 **::** replace "TYPE, BIND(C ,Name='Point') :: point" by "TYPE, BIND(C) :: point"

Page 58, line 31 **and** 32 **::** delete "*" before "x", "y" and "vertexNo"

Page 124, lines *last* − 7 **to** *last* − 4 **::** delete the complete sentence "The normal rules ... into lowercase."

Page 124, line *last* − 3 **::** replace "Other global data" by "Global data"

Page 125, lines 9 **and** 10 **::** remove the '*' before "i" and "r" in the C struct "cblock"

Page 125, line 18 **::** replace "with a variable" by "with an *extern* C variable"

Page 125, line *last* − 15 **::** replace "parameter" by "argument"

Chapter 11

Page 131, line 8 **::** change "*An ascended type for*" to "*Derived types for*"

Page 133, line 12 **::** change "are mostly defined as enumerated types in" to "are primarily named enumerated constants in"

Chapter 12

Page 145, line 19 **::** replace "! and ascends B to" by "! and assigns b to"

Chapter 13

Page 155, lines 13, *last* − 6 **and** *last* − 5 **::** change "parameter" to "argument"

Page 156, line *last* − 11 **::** change "complete array" to "complete matrix or array"

Page 159, line 1 **::** change "Parameter" to "Argument"

Page 159, line 13 **::** change "parameters" to "arguments"

Page 159, line 15 **::** change "rank" to "ranks"

Page 160, line 9 **::** change "parameter" to "argument"

Chapter 14

Page 165, line 3 **::** change "a()[2,3]" to "a(:)[2,3]"

Page 166, line 15 **::** change "via a coarray assignment." to "via all the other images fetching it."

Page 168, lines 4 **and** 9 **::** change "nodes" to "images"

Chapter 15

Page 178, line *last* − 4 **::** delete the word "addressing"

Page 179, First code extract :: The lines of code marked with the comment "!>Corrected<" need changing. The result of these change is that each *work(:)* array will now be used in separate memory space.

```
!! This code extract is part of the accompanying software
...
      INTERFACE
        SUBROUTINE subz(n,nmax,allwork) !>Corrected<
        IMPORT wp
          INTEGER, INTENT (IN) :: n, nmax !>Corrected<
          REAL(wp),TARGET,INTENT(INOUT):: allwork(:,:) !>Corrected<
...
! Illustrate passing work space to multiple threads.
      REAL (wp), ALLOCATABLE, TARGET :: allwork(:,:) !>Corrected<
! Here nmax is the largest value for n:
      INTEGER :: n, nmax = 10001
! Allocate one block of working storage.
      ALLOCATE(allwork(nmax,0:omp_get_max_threads()-1)) !>Corrected<
!$OMP PARALLEL
      DO n = 1, nmax
! The array allwork(:,:) is re-used as work space !>Corrected<
! with each call to the routine subz().
        CALL subz(n,nmax,allwork)  !>Corrected<
      END DO
!$OMP END PARALLEL
```

Pages 179/180, Second code extract on page 179 :: The lines of code marked with the comment "!>Corrected<" need changing.

```
!! This code extract is part of the accompanying software
...
    SUBROUTINE subz(n,nmax,allwork)  !>Corrected<
```

```
      USE omp_lib
...
      IMPLICIT NONE
      INTEGER, INTENT (IN) :: n, nmax   !>Corrected<
      REAL (wp),TARGET,INTENT(INOUT):: allwork(:,:)  !>Corrected<
      INTEGER :: i
...
! Designate an n - segment of the working array
! allwork(:,:) passed to subz as the local      !>Corrected<
! array, work(:).
!$OMP THREADPRIVATE(work)
      work => allwork(:,omp_get_thread_num()+1:) !>Corrected<  $

! Assign work(:) some values.
      DO i = 1, min(n,nmax)  !>Corrected<
        work(i) = REAL(i, wp)
      END DO
! Make sure this assignment is thread-safe:
      DO i = 1, min(n,nmax)  !>Corrected<
        IF (work(i) /= i) THEN
...
```

Chapter 16

Page 181, line 8 **::** replace "*Substituting deleted features*" by "*Substituting deleted and non-standard features*"

Page 185, line *last* − 13 **::** replace "parameter" by "argument"

Page 188, line *last* − 13 **::** add the sentence "The resulting code is for illustration and is not intended to be thread-safe." before "Thus"

Page 190, line *last* − 2 **::** replace "INTEGER, INTENT(OUT) :: callerstep" by "INTEGER, INTENT(INOUT) :: callerstep".

Chapter 17

The remarks below are aimed at enhancing the discussions about testing numerical software and applications. An attempt will be made to integrate them into a future printing.

Page 193, line 6 **::** replace "*Other testing strategies*" by "*Testing strategies and result comparisons*"

NB: There are additional references we could consider here. But the work of Brian Smith, James Lyness and Van Snyder provide examples for numerical software.

page 195, line *last* − 9 **::** add the sentence "A test harness for output comparison of numerical application code results is described by Smith".

"A Test Harness TH for Numerical Applications and Libraries", Brian T. Smith, in Eds. Patrick W. Gaffney, James C. T. Pool: Grid-Based Problem Solving Environments - IFIP TC2/ WG 2.5 Working Conference on Grid-Based Problem Solving Environments: Implications for Development and Deployment of Numerical Software July 17-21, 2006, Prescott, Arizona, USA. Eds. Patrick W. Gaffney and James C. T. Pool, Springer, IFIP 239, ISBN: 978-0-387-73658-7.

page 197, line $last - 14$ **::** at the end of the line add the sentence "A specific example for testing the stability of numerical quadrature codes, *performance profiling*, was proposed by James Lyness."

Summary found in "Elements of Statistical Computing: Numerical Computation, Volume 1", R. A. Thisted, p. 292-293, Chapman and Hall/CRC, 1996.

Also of interest might be an article by Van Snyder about testing functions of one and two variables in:

"Testing functions of one and two arguments", W. Van Snyder, in Proceedings of the IFIP TC2/WG2.5 Working Conference on Quality of Numerical Software: Assessment and Enhancement, Ed. Ron Boisvert. Pages: 155-166, Chapman & Hall, Ltd. London, (1997).

Chapter 18

Page 205, line $last - 5$ **::** delete "compiler"

Page 207, line 2 **::** add the sentence "A similar situation may also arise when a module is use-associated and the "only" clause imports names that are subsequently not used anywhere within the scope of the use association. Some compilers may also be induced to report this clutter problem."

Page 213, line $last - 3$ **::** replace the whole line by "the **DOUBLE COMPLEX** data type and byte size declarations such as **REAL*8** and **COMPLEX*16**"

Chapter 19

Page 220, line $last - 14$ **::** replace "places" by "place"

Page 221, line $last - 1$ **::** replace "then used" by "then be used"

Page 225, line $last - 7$ **::** replace "parameter" by "argument"

Chapter 20

Page 228, line $last - 16$ **::** change "Data are" to "Data is"

Page 228, line $last - 12$ **::** change "modules are illustrated" to "modules is illustrated".

Bibliography

On page 236, references [67] and [68] are specifically given web addresses.

Page 236, line *last* − 5 **::** replace "www.nag.com" with "www.nag.co.uk/nagware/np.asp" or "www.nag.com/nagware/NP/NP_desc.asp"

Page 236, line *last* − 3 **::** replace "www.nag.com" with "www.nag.co.uk/numeric/fl/FLdescription.asp" or "www.nag.com/numeric/fl/FLdescription.asp"