Index

accessor-mutator routine
   example in a module, 7
PUBLIC or PRIVATE
   components, 6
ACM, ix
   editors of CALGO, ix
Adams, Brainerd et al., see books,
   Fortran reference
Airy’s equation
   boundary value problem, 145–149
   connection formulas, 30
   w'' - zw = 0, 30
   Wronskian testing, 33
assignment overloading
   accumulating triplet list, 39
   accumulation
      data fitting, 142
      finite elements, 148
   clearing storage
      Harwell–Boeing (HB) format, 137
      one-time LU factoring, 132
      saving LU factors, 132
      sparse matrix add, +, 44
ASSOCIATE symbol, =>
   example, in module, 6
Association for Computing Machinery,
   see ACM

benchmarking, 197
BLAS, 14,103,113,143,227
   called from C, 119–121, 129–130
   interface for, 25–27
   library building, 17–18
Blue’s algorithm
   constants required for, 106
   l2 norm, 105
book code on SIAM web site
   Airy functions, 29, 227
   BlasModule, 26
   C and Fortran
      interlanguage use, 117, 229
   c_dgemm, 121
   c_saxpy, 120
   c_dgemm, 121
   classes, 47, 228
   coarray examples, 163, 168
   copyString, 122
   cSam, 125
   csaxpy, 119
   csaxpyval, 120
   csyscall, 127
   dgetrf, 14
   dgetrs, 14
   dgemm, 121
   dgemv, 26
   dnrm2, 26
   dnrm2_support, 106
   errnorm, 93
   exampleLapackInterface, 27
   exceptions, 101, 228
   funptrs, 125
   LAPACK example, 14, 227
   LapackInterface, 25
   MPI interface, 151, 229
   nonintrinsic modules
      IEEE_ARITHMETIC.f90, 228
      IEEE_EXCEPTIONS.f90, 228
      IEEE_FEATURES.f90, 228
normalPdf, 94
OpenMP examples, 169, 229
PtrArray, 123
qag, 51, 90
qag2003, 77, 89
qag2003s, 77, 90
qag2003v, 77, 90
qkgsens, 90
qkgenv, 90
QUADPACK, 71, 228
QUADPACK documentation, 89, 228
quadpack2003, 91
quicksort, 57, 228
saxpy, 119, 120
set_precision, 21
sparse defined operation, 35, 228
sparse matrix applications, 141, 229
sparse system solves, 131, 229
syscall, 127
zeroin, 53, 55

books
Fortran reference, x
Adams, Brainerd et al., x
Fortran 77 standard, x
Fortran 90 standard, x
Metcalf, Reid, and Cohen, x

C
interoperability, also see Fortran, modern features
matching array indexing, 123
matching data types, 119
matching Fortran COMMON, C global data, 125
matching pointers to C functions, 125
matching structs and derived types, 124
name mangling, 120
rationale, 117
saxpy, C calls
by reference, by value, 120
string conversion, 122
interoperability for MPI, 151

CALGO, see ACM
cooray, see also MPI
allocateable arrays, 165
codimensions, 164
coidex, 164
corank, 164
critical {...} end critical, 168
data communication,
Fortran 2008 example, 164
example of computing π
with quadrature, 166
example of matrix-vector products,
167
image_index, 166
images, 163
new convention for subscripts, 163
num_images, 165, 166
rationale, 163
specifying data objects, 164
sync all, 166
this_image, 165, 166
code conversion
regression testing, 182
Collected Algorithms from ACM (CALGO), see ACM
compiler options
checking array bounds, 206
checking deleted and obsolete features, 207, 213
checking unassigned variables, 206
checking unused variables, 206
deleted and obsolescent features, 181
optimization level, 209
performance and debugging level, 208
performance and optimization level, 212
data to user routines
COMMON blocks, 51
texample, classes as arguments, 51
texample, zeroin, 55
module variables, 51
thread-safe, 51
defined operation
  .ip., .pi., superLU, solves, 132
  .p., .ip., assignment
  Airy’s equation,
    boundary value problem, 145
  least-squares,
    piecewise linear, 142
  clearing storage
    interior of SuperLU , 137
  INTENT(IN), operands, 132
  parsing precedence, 42
  use of parentheses, 45
  sparse matrix add, +, 42, 43
  sparse matrix product, *, .p.,
    42, 45
  sparse matrix solve, .ip., .pi., 131
  sparse matrix transpose, .t.,
    42, 45
deleted features
  ASSIGN label,
    GO TO assigned label, 187
  END IF with label, 187
  nH or hollerith
    edit descriptors, 187
  PAUSE, 187
  REAL loop variables, 187
derived types
  class procedures, 63–64
  example, in module, 6
  Dijkstra quote on testing, 193
documentation
  arguments
    input type, kind, and rank, 92
    output, 93
  example, CALL qag2003(), 95
  modules, derived types
  PUBLIC objects, 90
  QUADPACK use,
    example, 90
  routines called, 90
  user function
    interface, 91
    recommended packaging, 94
  variable dictionary, 89
  writing style, 89
equivalence partitioning, 198
Fortran
  statement functions, 211
Fortran development
  rationale, ix
Fortran optimization
  algorithm choice, 212
  memory access, 211
  MPI and coarrays, 212
  OpenMP, 212
Fortran standards, see books, Fortran reference
Fortran, modern features,
  see assignment overloading
  see data to user routines
  see defined operation
  see deleted features
  see derived types, classes
  see generic names, subprograms
  see IEEE standard
  see object-oriented features
  see obsolescent features
  see procedures, type-bound
  see QUADPACK
  see recursion
allocating with problem size
  example, 19
ASSOCIATE symbol, 4
avoiding name clashes, 27
CASE statement
  computed-GO TO, 2, 184–185
comments, inline, 2
  example, 3
compilation units
  functions, 13
  main programs, 13
  modules, 13
  subroutines, 13
compiler analysis, 210
continuation lines
  use of symbol &, 3
derived type, xi
  component separator, %, 5
  often a C structure, also a
  record, 5
PUBLIC or PRIVATE components,  
see modules, packaging

dusty decks, 212
early optimization, 209
gfortran, compiler, 14
INCLUDE
  file, 9–10, 32–33
  templates, 31–32
interfaces, 24
  defining a module procedure, 31
interoperability with C, ix, 117
intrinsic modules, 102, 119, 172
library, building, 18
modules, xi, see also modules,
  packaging
multiple statements per line
  avoid, 2, 4
name transformation
  dnrm2, locally called 2norm, 27
numeric inquiry functions, 102
optional arguments, 8
  example, 8, 11
PRESENT intrinsic function, 8
pointer, 4
relational symbols, 5
source formatting, 2
timing routines, 10
  cpu_time, 10
  system_clock, 11
top 10 items, xi
use of symbol =>, 4
VOLATILE attribute, 173
generic names
  Airy functions
    example, 30
packaging Airy functions
    example, 31
subprograms
  advantages, 29, 30

IEEE standard
  floating-point exceptions, 102
  example, d2norm, 103
Fortran floating-point history, 102

Fortran intrinsic modules
  IEEE_ARITHMETIC, 102, 111,
  113
  IEEE_EXCEPTIONS, 102, 111
  IEEE_FEATURES, 102, 111–112
  ISO_C_BINDING, 119–128, 136
Kahan, William, 102, 105
overflow and underflow
  setting flags, 109
setting IEEE halting mode, 107
underflow and denormalized results,
  113
intrinsic subprograms
  c_loc, 160
  cpu_time, 10
  huge, 104
  move_alloc, 41
  present, 8
  random_number, 141
  selected_real_kind, 21
  system_clock, 11
  tiny, 113
  transfer, 9

Kahan’s algorithm
  constants required, 107
  l2 norm, 108

LAPACK
  dgetrf, dgetrs, 14
library building, 17
  ar, ranlib, 17
legacy code, restructuring, 188
legacy interfacing, 14
  BLAS, 14
  code bloat, 17
  compilation, linking, 17
  file suffix .f, 14
LAPACK, 14
  timing, dgetrf, dgetrs, 15
library building, 17
library, precompiled
  calls to module subprograms, 26
calls, interfaces, 25
library, usage
  order, linker, 18
Index

Maple version 11.02
  use of, 148
matrix, sparse, see also defined operation
  representation
    Harwell–Boeing (HB), 36
    triplet, 38
Metcalf, Reid, and Cohen, see books,
  Fortran reference
mistake, purposeful
  quicksort, correct coding, 60
modules, packaging
  derived types
    interfaces, subprograms, 6
    PUBLIC or PRIVATE
    components, 6
KIND parameters, 20
  precision definition
    example, 21
    SELECTED_REAL_KIND, 21
USE-association, 6
  example, 7, 8
MPI, see coarray
  _loc, target attribute, 160
  core suite, 158
  module, Fortran-to-MPI
    interfaces, core suite, 160
  parameters, defining
    C provides values, 159
  routines, number of, 158
object-oriented (OO)
  accessor, mutator functions, 50
  class, 48
  inheritance, polymorphism, 49
  method, 48
  object, 48
obsolescent features
  alternate return, 183
  arithmetic-if test, 182
  character function length, 186
  CHARACTER* declaration, 186
  computed-GO TO, 184
DATA statements, scattered, 185
DO termination, 183
  fixed source format, 182
future obsolescence, 20
  one line functions, 185
OpenMP
  array update
    example, 171
  atomic, 175
  CHUNKSIZE, 180
  cons, 170
  critical, 175
  fragmentation, memory
    example, 178
  general models, pthreads, 180
  IEEE exceptions, 172
  input, output
    example, 173
  mutex, 180
  output only
    example, 172
  private variables, personal, 177
  pros, 170
  pthreads, 180
  race, deadlock, 170
  REDUCTION, example, 172
  SCHEDULE, 180
  subprograms, calling
    not thread-safe, 176
    thread-safe, 170
    thread-safe, tips, 176
  Windows threads, 180
operator overloading
  sparse matrix add, +, 42
optimization
  in-line calls, 62, 211, 212
packaging with modules
  features, 18, 19
precision, range
  IEEE floating-point, 21
procedures, type-bound
  CLASS versus TYPE declaration, 64
  example, 49
  FINAL, 54, 55
  NOPASS, PASS, example, 64
purposeful mistake
  C interoperability
    system calls, 127
QUADPACK,
   see also documentation
derived TYPE (quadpackbase), 75
eenhancements, 73
   CLASS objects,
      abstract interfaces, 73
generic names, 74
procedure pointers, 74
source conversion, 74
vectorization, 74
integrals
   complex, 72
      example, 85
multiple, 72
one-dimensional
      Bessel $J_0$, example, 77
data passing, 77
two-dimensional, 81
      compute recursively, 81
vector, 72
interface
   Fortran 77, qag, 75
generic name, qag2003, 77
scalar integrals, qag2003s, 77
vector integrals, qag2003v, 77
restrictions, Fortran 77, 73
   access to data, 73
efficiency, 73
   integrals, multiple, 73
vectorization, 73
quicksort, xiii, xv, 47, 57–70, 113
algorithms, recursive definition, 58
comparisons, 59, 60
functions, in-lining of, 62
in-situ sorting, 62
operations, 59, 60
random numbers
   complex pairs, 8
   example, 8
recursion
   efficiency, reluctance, 57
   quicksort, 57
reverse communication
   example, Newton method, 188
SIAM, see book code on SIAM web site
solving least squares
   linear and sparse system, 142
   ill-conditioned or singular, 142
standard
   Fortran 2003, x
   ISO Fortran 90, x
SuperLU, C code
clear_suplu, clearing storage, 137
SuperLU solving using C code
   SOLVE with SuperLU, 137
testing
   block coverage, 195, 202
   boundary values, 199
decision coverage, 202
equivalence partitioning, 198
error logging, 196
exhaustive, 194
Lapack coverage, 195
logging coverage, 195
makefiles, 196
metric, 195, 196
oracle, 197
regression, 196
smoke, 201, 203
soak, 196, 197, 203
statement coverage, 195
unit, 196, 197
white box, 196
timing subprograms
cpu_time(), 10
system_clock, 10, 11
tools, software
code coverage, 221
code debuggers, 220
code documentation,
   Doxygen, 219
code profilers, 221
editors, language aware, 215, 225
error processors, 222
make, 216
Makefile, example, 217
source formatters, 223
text control, SVN, 219
x87 assembler, 104