ADAS Subroutine h9qd3b

```
      subroutine h9qd3b( ntdim , nedim   , nfdim   ,
     &                   teff  , tenum   ,
     &                   ne    , nt      , wupper  , wlower  ,
     &                   xa    , dist    , kap_val , dru_val ,
     &                   zeta  , evt     , omega   ,
     &                   n_en  , en      , f       , nte     ,
     &                   nform1, param1  , nform2  , param2  ,
     &                   alpha , q
     &                 )

C-----------------------------------------------------------------------
C
C  ****************** fortran77 subroutine: h9qd3b *********************
C
C  purpose:  To execute quadratures over ionisation collision strengths
C            to determine the ionisation and 3-body recombination
C            coefficients. Free electron distribution function may be
C            Maxwellian, Kappa, Druyvesteyn, or numeric from adf37 file.
C
C  calling program: adas809
C
C  input : (i*4)  ntdim    = max no of temperatures that can be read in
C  input : (i*4)  nedim    = max no of energy points that can be read in
C  input : (r*8)  teff()   = input effective temperatures (eV)
C  input : (r*8)  tenum()  = effective temperatures from adf37 file (eV)
C  input : (i*4)  ne       = no of x parameter values in adf04 type 1
C  input : (i*4)  nt       = no of input temperatures
C  input : (r*8)  wupper   = statistical weight of upper level
C  input : (r*8)  wlower   = statistical weight of lower level
C  input : (r*8)  xa()     = x parameter from adf04 type1
C  input : (i*4)  dist     = distribution type
C                              0 => Maxwellian
C                              1 => kappa
C                              2 => numeric
C                              3 => Druyvesteyn
C  input : (r*8)  kap_val  = value of kappa parameter
C  input : (r*8)  dru_val  = value of x parameter in Druyvesteyn dist
C  input : (r*8)  evt      = ionisation potential
C  input : (r*8)  omega()  = collision strength from adf04 type1
C  input : (i*4)  n_en     = no of energy points in adf37
C  input : (r*8)  en(,)    = energy points of distribution tabulation
C  input : (r*8)  f(,)     = distribution function tabulation
C  input : (i*4)  nte      = no of temperatures in adf37
C  input : (i*4)  nform1   = type of threshold behaviour
C                              1 => cutoff
C                              2 => energy^param1
C  input : (r*8)  param1() = parameter of threshold form
C  input : (i*4)  nform2   = type of high-energy behaviour
C                              1 => cutoff
C                              2 => energy^-param2(1)
C                              3 => exp(-param2(1)*energy)
C  input : (r*8)  param2   = parameter of high-energy form
```

```
C
C  output: (r*8)  alpha    = 3-body recombination coefficient
C  output: (r*8)  q        = ionisation coefficient
C
C  routines:
C          routine    source    brief description
C          ------------------------------------------------------------
C          eei        ADAS      evaluates  exp(x)*E1(x)
C          lngama     ADAS evaluates ln(gamma(x))
C
C  author: Paul Bryans, University of Strathclyde
C
C  date:   22/11/04
C
c  modification history:
c
c  date:   22/11/04 version: 1.1
c  modified: Paul Bryans
c - first release
c
c  date:   26/11/04
c  modified: Allan Whiteford
c - made some arrays locally dimensioned to prevent
c                g77 compiled code failing when it tried to make
c                large automatic arrays.
c
c  date:   26/11/04
c  modified: Paul Bryans
c - Moved final exponential coefficient of Maxwellian
c                and Druyvesteyn ionisation rate inside integrals to
c                eliminate overflow problem when this becomes large.
c
c  date:   18/08/05
c  modified: Paul Bryans
c - Set energy difference equal to zero at threshold.
c                Numerical precision was causing this to be small and
c                negative, giving NaN when raised to non-integer power.
c
C-----------------------------------------------------------------------
      INTEGER             DIST,         NE,           NEDIM,        NFDIM
      INTEGER             NFORM1,       NFORM2,       NT,           NTDIM
      INTEGER             NTE,          N_EN
      REAL*8              ALPHA(NTDIM),               DRU_VAL
      REAL*8              EN(NTDIM,NFDIM),            EVT
      REAL*8              F(NTDIM,NFDIM),             KAP_VAL
      REAL*8              OMEGA(NEDIM),               PARAM1
      REAL*8              PARAM2(2),    Q(NTDIM),     TEFF(NTDIM)
      REAL*8              TENUM(QDNT), WLOWER,        WUPPER
      REAL*8              XA(NEDIM),    ZETA
```