```
int eraCal2jd(int iy, int im, int id, double *djm0, double *djm)
/*
**  - - - - - - - - - -
**   e r a C a l 2 j d
**  - - - - - - - - - -
**
**  Gregorian Calendar to Julian Date.
**
**  Given:
**     iy,im,id  int     year, month, day in Gregorian calendar (Note 1)
**
**  Returned:
**     djm0      double  MJD zero-point: always 2400000.5
**     djm       double  Modified Julian Date for 0 hrs
**
**  Returned (function value):
**               int     status:
**                           0 = OK
**                          -1 = bad year   (Note 3: JD not computed)
**                          -2 = bad month  (JD not computed)
**                          -3 = bad day    (JD computed)
**
**  Notes:
**
**  1) The algorithm used is valid from -4800 March 1, but this
**     implementation rejects dates before -4799 January 1.
**
**  2) The Julian Date is returned in two pieces, in the usual ERFA
**     manner, which is designed to preserve time resolution.  The
**     Julian Date is available as a single number by adding djm0 and
**     djm.
**
**  3) In early eras the conversion is from the "Proleptic Gregorian
**     Calendar";  no account is taken of the date(s) of adoption of
**     the Gregorian Calendar, nor is the AD/BC numbering convention
**     observed.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 12.92 (p604).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEpb(double dj1, double dj2)
/*
**  - - - - - - -
**   e r a E p b
**  - - - - - - -
**
**  Julian Date to Besselian Epoch.
**
**  Given:
**     dj1,dj2    double     Julian Date (see note)
**
**  Returned (function value):
**                double     Besselian Epoch.
**
**  Note:
**
**     The Julian Date is supplied in two pieces, in the usual ERFA
**     manner, which is designed to preserve time resolution.  The
**     Julian Date is available as a single number by adding dj1 and
**     dj2.  The maximum resolution is achieved if dj1 is 2451545.0
**     (J2000.0).
**
**  Reference:
**
**     Lieske, J.H., 1979. Astron.Astrophys., 73, 282.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraEpb2jd(double epb, double *djm0, double *djm)
/*
**  - - - - - - - - - -
**   e r a E p b 2 j d
**  - - - - - - - - - -
**
**  Besselian Epoch to Julian Date.
**
**  Given:
**     epb      double    Besselian Epoch (e.g. 1957.3)
**
**  Returned:
**     djm0     double    MJD zero-point: always 2400000.5
**     djm      double    Modified Julian Date
**
**  Note:
**
**     The Julian Date is returned in two pieces, in the usual ERFA
**     manner, which is designed to preserve time resolution.  The
**     Julian Date is available as a single number by adding djm0 and
**     djm.
**
**  Reference:
**
**     Lieske, J.H., 1979, Astron.Astrophys. 73, 282.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEpj(double dj1, double dj2)
/*
**  - - - - - - -
**   e r a E p j
**  - - - - - - -
**
**  Julian Date to Julian Epoch.
**
**  Given:
**     dj1,dj2    double     Julian Date (see note)
**
**  Returned (function value):
**                double     Julian Epoch
**
**  Note:
**
**     The Julian Date is supplied in two pieces, in the usual ERFA
**     manner, which is designed to preserve time resolution.  The
**     Julian Date is available as a single number by adding dj1 and
**     dj2.  The maximum resolution is achieved if dj1 is 2451545.0
**     (J2000.0).
**
**  Reference:
**
**     Lieske, J.H., 1979, Astron.Astrophys. 73, 282.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraEpj2jd(double epj, double *djm0, double *djm)
/*
**  - - - - - - - - - -
**   e r a E p j 2 j d
**  - - - - - - - - - -
**
**  Julian Epoch to Julian Date.
**
**  Given:
**     epj      double    Julian Epoch (e.g. 1996.8)
**
**  Returned:
**     djm0     double    MJD zero-point: always 2400000.5
**     djm      double    Modified Julian Date
**
**  Note:
**
**     The Julian Date is returned in two pieces, in the usual ERFA
**     manner, which is designed to preserve time resolution.  The
**     Julian Date is available as a single number by adding djm0 and
**     djm.
**
**  Reference:
**
**     Lieske, J.H., 1979, Astron.Astrophys. 73, 282.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraJd2cal(double dj1, double dj2,
              int *iy, int *im, int *id, double *fd)
/*
**  - - - - - - - - - -
**   e r a J d 2 c a l
**  - - - - - - - - - -
**
**  Julian Date to Gregorian year, month, day, and fraction of a day.
**
**  Given:
**     dj1,dj2   double   Julian Date (Notes 1, 2)
**
**  Returned (arguments):
**     iy        int      year
**     im        int      month
**     id        int      day
**     fd        double   fraction of day
**
**  Returned (function value):
**               int      status:
**                           0 = OK
**                          -1 = unacceptable date (Note 1)
**
**  Notes:
**
**  1) The earliest valid date is -68569.5 (-4900 March 1).  The
**     largest value accepted is 1e9.
**
**  2) The Julian Date is apportioned in any convenient way between
**     the arguments dj1 and dj2.  For example, JD=2450123.7 could
**     be expressed in any of these ways, among others:
**
**            dj1             dj2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**  3) In early eras the conversion is from the "proleptic Gregorian
**     calendar";  no account is taken of the date(s) of adoption of
**     the Gregorian calendar, nor is the AD/BC numbering convention
**     observed.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 12.92 (p604).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraJdcalf(int ndp, double dj1, double dj2, int iymdf[4])
/*
**  - - - - - - - - - -
**   e r a J d c a l f
**  - - - - - - - - - -
**
**  Julian Date to Gregorian Calendar, expressed in a form convenient
**  for formatting messages:  rounded to a specified precision.
**
**  Given:
**     ndp       int      number of decimal places of days in fraction
**     dj1,dj2   double   dj1+dj2 = Julian Date (Note 1)
**
**  Returned:
**     iymdf     int[4]   year, month, day, fraction in Gregorian
**                        calendar
**
**  Returned (function value):
**               int      status:
**                          -1 = date out of range
**                           0 = OK
**                          +1 = NDP not 0-9 (interpreted as 0)
**
**  Notes:
**
**  1) The Julian Date is apportioned in any convenient way between
**     the arguments dj1 and dj2.  For example, JD=2450123.7 could
**     be expressed in any of these ways, among others:
**
**             dj1            dj2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**  2) In early eras the conversion is from the "Proleptic Gregorian
**     Calendar";  no account is taken of the date(s) of adoption of
**     the Gregorian Calendar, nor is the AD/BC numbering convention
**     observed.
**
**  3) Refer to the function eraJd2cal.
**
**  4) NDP should be 4 or less if internal overflows are to be
**     avoided on machines which use 16-bit integers.
**
**  Called:
**     eraJd2cal    JD to Gregorian calendar
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 12.92 (p604).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAb(double pnat[3], double v[3], double s, double bm1,
           double ppr[3])
/*
**  - - - - - -
**   e r a A b
**  - - - - - -
**
**  Apply aberration to transform natural direction into proper
**  direction.
**
**  Given:
**    pnat    double[3]   natural direction to the source (unit vector)
**    v       double[3]   observer barycentric velocity in units of c
**    s       double      distance between the Sun and the observer (au)
**    bm1     double      sqrt(1-|v|^2): reciprocal of Lorenz factor
**
**  Returned:
**    ppr     double[3]   proper direction to source (unit vector)
**
**  Notes:
**
**  1) The algorithm is based on Expr. (7.40) in the Explanatory
**     Supplement (Urban & Seidelmann 2013), but with the following
**     changes:
**
**     o  Rigorous rather than approximate normalization is applied.
**
**     o  The gravitational potential term from Expr. (7) in
**        Klioner (2003) is added, taking into account only the Sun’s
**        contribution.  This has a maximum effect of about
**        0.4 microarcsecond.
**
**  2) In almost all cases, the maximum accuracy will be limited by the
**     supplied velocity.  For example, if the ERFA eraEpv00 function is
**     used, errors of up to 5 microarcseconds could occur.
**
**  References:
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013).
**
**     Klioner, Sergei A., "A practical relativistic model for micro-
**     arcsecond astrometry in space", Astr. J. 125, 1580-1597 (2003).
**
**  Called:
**     eraPdp       scalar product of two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApcg(double date1, double date2,
             double ebpv[2][3], double ehp[3],
             eraASTROM *astrom)
/*
**  - - - - - - - -
**   e r a A p c g
**  - - - - - - - -
**
**  For a geocentric observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and GCRS coordinates.
**  The Earth ephemeris is supplied by the caller.
**
**  The parameters produced by this function are required in the
**  parallax, light deflection and aberration parts of the astrometric
**  transformation chain.
**
**  Given:
**     date1  double       TDB as a 2-part...
**     date2  double       ...Julian Date (Note 1)
**     ebpv   double[2][3] Earth barycentric pos/vel (au, au/day)
**     ehp    double[3]    Earth heliocentric position (au)
**
**  Returned:
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) All the vectors are with respect to BCRS axes.
**
**  3) This is one of several functions that inserts into the astrom
```

```
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  4) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraApcs      astrometry parameters, ICRS-GCRS, space observer
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApcg13(double date1, double date2, eraASTROM *astrom)
/*
**  - - - - - - - - - - -
**   e r a A p c g 1 3
**  - - - - - - - - - - -
**
**  For a geocentric observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and GCRS coordinates.
**  The caller supplies the date, and ERFA models are used to predict
**  the Earth ephemeris.
**
**  The parameters produced by this function are required in the
**  parallax, light deflection and aberration parts of the astrometric
**  transformation chain.
**
**  Given:
**     date1  double     TDB as a 2-part...
**     date2  double     ...Julian Date (Note 1)
**
**  Returned:
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) All the vectors are with respect to BCRS axes.
**
**  3) In cases where the caller wishes to supply his own Earth
**     ephemeris, the function eraApcg can be used instead of the present
**     function.
**
```

```
**  4) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  5) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraEpv00     Earth position and velocity
**     eraApcg      astrometry parameters, ICRS-GCRS, geocenter
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApci(double date1, double date2,
             double ebpv[2][3], double ehp[3],
             double x, double y, double s,
             eraASTROM *astrom)
/*
**  - - - - - - - - -
**   e r a A p c i
**  - - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and geocentric CIRS
**  coordinates.  The Earth ephemeris and CIP/CIO are supplied by the
**  caller.
**
**  The parameters produced by this function are required in the
**  parallax, light deflection, aberration, and bias-precession-nutation
**  parts of the astrometric transformation chain.
**
**  Given:
**     date1  double       TDB as a 2-part...
**     date2  double       ...Julian Date (Note 1)
**     ebpv   double[2][3] Earth barycentric position/velocity (au, au/day)
**     ehp    double[3]    Earth heliocentric position (au)
**     x,y    double       CIP X,Y (components of unit vector)
**     s      double       the CIO locator s (radians)
**
**  Returned:
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
```

```
**
**  2) All the vectors are with respect to BCRS axes.
**
**  3) In cases where the caller does not wish to provide the Earth
**     ephemeris and CIP/CIO, the function eraApci13 can be used instead
**     of the present function.  This computes the required quantities
**     using other ERFA functions.
**
**  4) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  5) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraApcg      astrometry parameters, ICRS-GCRS, geocenter
**     eraC2ixys    celestial-to-intermediate matrix, given X,Y and s
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApci13(double date1, double date2,
               eraASTROM *astrom, double *eo)
/*
**  - - - - - - - - - -
**   e r a A p c i 1 3
**  - - - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and geocentric CIRS
**  coordinates.  The caller supplies the date, and ERFA models are used
**  to predict the Earth ephemeris and CIP/CIO.
**
**  The parameters produced by this function are required in the
**  parallax, light deflection, aberration, and bias-precession-nutation
**  parts of the astrometric transformation chain.
**
**  Given:
**     date1  double      TDB as a 2-part...
**     date2  double      ...Julian Date (Note 1)
**
**  Returned:
**     astrom eraASTROM*  star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**     eo     double*     equation of the origins (ERA-GST)
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) All the vectors are with respect to BCRS axes.
**
**  3) In cases where the caller wishes to supply his own Earth
**     ephemeris and CIP/CIO, the function eraApci can be used instead
```

```
**     of the present function.
**
**  4) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  5) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraEpv00     Earth position and velocity
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**     eraApci      astrometry parameters, ICRS-CIRS
**     eraEors      equation of the origins, given NPB matrix and s
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApco(double date1, double date2,
             double ebpv[2][3], double ehp[3],
             double x, double y, double s, double theta,
             double elong, double phi, double hm,
             double xp, double yp, double sp,
             double refa, double refb,
             eraASTROM *astrom)
/*
**  - - - - - - - -
**   e r a A p c o
**  - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and observed
**  coordinates.  The caller supplies the Earth ephemeris, the Earth
**  rotation information and the refraction constants as well as the
**  site coordinates.
**
**  Given:
**     date1  double       TDB as a 2-part...
**     date2  double       ...Julian Date (Note 1)
**     ebpv   double[2][3] Earth barycentric PV (au, au/day, Note 2)
**     ehp    double[3]    Earth heliocentric P (au, Note 2)
**     x,y    double       CIP X,Y (components of unit vector)
**     s      double       the CIO locator s (radians)
**     theta  double       Earth rotation angle (radians)
**     elong  double       longitude (radians, east +ve, Note 3)
**     phi    double       latitude (geodetic, radians, Note 3)
**     hm     double       height above ellipsoid (m, geodetic, Note 3)
**     xp,yp  double       polar motion coordinates (radians, Note 4)
**     sp     double       the TIO locator s’ (radians, Note 4)
**     refa   double       refraction constant A (radians, Note 5)
**     refb   double       refraction constant B (radians, Note 5)
**
**  Returned:
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
```

```
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) The vectors eb, eh, and all the astrom vectors, are with respect
**     to BCRS axes.
**
**  3) The geographical coordinates are with respect to the ERFA_WGS84
**     reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN
**     CONVENTION:  the longitude required by the present function is
**     right-handed, i.e. east-positive, in accordance with geographical
**     convention.
**
**  4) xp and yp are the coordinates (in radians) of the Celestial
**     Intermediate Pole with respect to the International Terrestrial
**     Reference System (see IERS Conventions), measured along the
**     meridians 0 and 90 deg west respectively.  sp is the TIO locator
**     s', in radians, which positions the Terrestrial Intermediate
**     Origin on the equator.  For many applications, xp, yp and
**     (especially) sp can be set to zero.
**
**     Internally, the polar motion is stored in a form rotated onto the
**     local meridian.
**
**  5) The refraction constants refa and refb are for use in a
**     dZ = A*tan(Z)+B*tan^3(Z) model, where Z is the observed
**     (i.e. refracted) zenith distance and dZ is the amount of
**     refraction.
**
**  6) It is advisable to take great care with units, as even unlikely
**     values of the input parameters are accepted and processed in
**     accordance with the models used.
**
**  7) In cases where the caller does not wish to provide the Earth
**     Ephemeris, the Earth rotation information and refraction
**     constants, the function eraApco13 can be used instead of the
**     present function.  This starts from UTC and weather readings etc.
**     and computes suitable values using other ERFA functions.
**
**  8) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
```

```
**
**   9) The context structure astrom produced by this function is used by
**      eraAtioq, eraAtoiq, eraAtciq* and eraAticq*.
**
**  Called:
**     eraAper      astrometry parameters: update ERA
**     eraC2ixys    celestial-to-intermediate matrix, given X,Y and s
**     eraPvtob     position/velocity of terrestrial station
**     eraTrxpv     product of transpose of r-matrix and pv-vector
**     eraApcs      astrometry parameters, ICRS-GCRS, space observer
**     eraCr        copy r-matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraApco13(double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              eraASTROM *astrom, double *eo)
/*
**  - - - - - - - - - -
**   e r a A p c o 1 3
**  - - - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between ICRS and observed
**  coordinates.  The caller supplies UTC, site coordinates, ambient air
**  conditions and observing wavelength, and ERFA models are used to
**  obtain the Earth ephemeris, CIP/CIO and refraction constants.
**
**  The parameters produced by this function are required in the
**  parallax, light deflection, aberration, and bias-precession-nutation
**  parts of the ICRS/CIRS transformations.
**
**  Given:
**     utc1   double     UTC as a 2-part...
**     utc2   double     ...quasi Julian Date (Notes 1,2)
**     dut1   double     UT1-UTC (seconds, Note 3)
**     elong  double     longitude (radians, east +ve, Note 4)
**     phi    double     latitude (geodetic, radians, Note 4)
**     hm     double     height above ellipsoid (m, geodetic, Notes 4,6)
**     xp,yp  double     polar motion coordinates (radians, Note 5)
**     phpa   double     pressure at the observer (hPa = mB, Note 6)
**     tc     double     ambient temperature at the observer (deg C)
**     rh     double     relative humidity at the observer (range 0-1)
**     wl     double     wavelength (micrometers, Note 7)
**
**  Returned:
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**     eo     double*    equation of the origins (ERA-GST)
**
**  Returned (function value):
**            int        status: +1 = dubious year (Note 2)
**                                0 = OK
**                               -1 = unacceptable date
**
**  Notes:
**
**  1)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
```

```
**      it implements the leap-second-ambiguity convention just
**      described.
**
**  2)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the
**      future to be trusted.  See eraDat for further details.
**
**  3)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  4)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
**      (i.e. right-handed), in accordance with geographical convention.
**
**  5)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**      values are the coordinates (in radians) of the Celestial
**      Intermediate Pole with respect to the International Terrestrial
**      Reference System (see IERS Conventions 2003), measured along the
**      meridians 0 and 90 deg west respectively.  For many
**      applications, xp and yp can be set to zero.
**
**      Internally, the polar motion is stored in a form rotated onto
**      the local meridian.
**
**  6)  If hm, the height above the ellipsoid of the observing station
**      in meters, is not known but phpa, the pressure in hPa (=mB), is
**      available, an adequate estimate of hm can be obtained from the
**      expression
**
**            hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**      where tsl is the approximate sea-level air temperature in K
**      (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**      52).  Similarly, if the pressure phpa is not known, it can be
**      estimated from the height of the observing station, hm, as
**      follows:
**
**            phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**      Note, however, that the refraction is nearly proportional to
**      the pressure and that an accurate phpa value is important for
**      precise work.
**
**  7)  The argument wl specifies the observing wavelength in
**      micrometers.  The transition from optical to radio is assumed to
**      occur at 100 micrometers (about 3000 GHz).
**
**  8)  It is advisable to take great care with units, as even unlikely
**      values of the input parameters are accepted and processed in
**      accordance with the models used.
**
**  9)  In cases where the caller wishes to supply his own Earth
**      ephemeris, Earth rotation information and refraction constants,
**      the function eraApco can be used instead of the present function.
**
**  10) This is one of several functions that inserts into the astrom
**      structure star-independent parameters needed for the chain of
**      astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**      The various functions support different classes of observer and
**      portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
```

```
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**      Those with names ending in "13" use contemporary ERFA models to
**      compute the various ephemerides.  The others accept ephemerides
**      supplied by the caller.
**
**      The transformation from ICRS to GCRS covers space motion,
**      parallax, light deflection, and aberration.  From GCRS to CIRS
**      comprises frame bias and precession-nutation.  From CIRS to
**      observed takes account of Earth rotation, polar motion, diurnal
**      aberration and parallax (unless subsumed into the ICRS <-> GCRS
**      transformation), and atmospheric refraction.
**
**  11) The context structure astrom produced by this function is used
**      by eraAtioq, eraAtoiq, eraAtciq* and eraAticq*.
**
**  Called:
**     eraUtctai    UTC to TAI
**     eraTaitt     TAI to TT
**     eraUtcut1    UTC to UT1
**     eraEpv00     Earth position and velocity
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**     eraEra00     Earth rotation angle, IAU 2000
**     eraSp00      the TIO locator s', IERS 2000
**     eraRefco     refraction constants for given ambient conditions
**     eraApco      astrometry parameters, ICRS-observed
**     eraEors      equation of the origins, given NPB matrix and s
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApcs(double date1, double date2, double pv[2][3],
             double ebpv[2][3], double ehp[3],
             eraASTROM *astrom)
/*
**  - - - - - - - -
**   e r a A p c s
**  - - - - - - - -
**
**  For an observer whose geocentric position and velocity are known,
**  prepare star-independent astrometry parameters for transformations
**  between ICRS and GCRS.  The Earth ephemeris is supplied by the
**  caller.
**
**  The parameters produced by this function are required in the space
**  motion, parallax, light deflection and aberration parts of the
**  astrometric transformation chain.
**
**  Given:
**     date1  double       TDB as a 2-part...
**     date2  double       ...Julian Date (Note 1)
**     pv     double[2][3] observer's geocentric pos/vel (m, m/s)
**     ebpv   double[2][3] Earth barycentric PV (au, au/day)
**     ehp    double[3]    Earth heliocentric P (au)
**
**  Returned:
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) All the vectors are with respect to BCRS axes.
```

```
**
**  3) Providing separate arguments for (i) the observer's geocentric
**     position and velocity and (ii) the Earth ephemeris is done for
**     convenience in the geocentric, terrestrial and Earth orbit cases.
**     For deep space applications it maybe more convenient to specify
**     zero geocentric position and velocity and to supply the
**     observer's position and velocity information directly instead of
**     with respect to the Earth.  However, note the different units:
**     m and m/s for the geocentric vectors, au and au/day for the
**     heliocentric and barycentric vectors.
**
**  4) In cases where the caller does not wish to provide the Earth
**     ephemeris, the function eraApcs13 can be used instead of the
**     present function.  This computes the Earth ephemeris using the
**     ERFA function eraEpv00.
**
**  5) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  6) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraCp        copy p-vector
**     eraPm        modulus of p-vector
**     eraPn        decompose p-vector into modulus and direction
**     eraIr        initialize r-matrix to identity
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApcs13(double date1, double date2, double pv[2][3],
               eraASTROM *astrom)
/*
**  - - - - - - - - - - -
**   e r a A p c s 1 3
**  - - - - - - - - - - -
**
**  For an observer whose geocentric position and velocity are known,
**  prepare star-independent astrometry parameters for transformations
**  between ICRS and GCRS.  The Earth ephemeris is from ERFA models.
**
**  The parameters produced by this function are required in the space
**  motion, parallax, light deflection and aberration parts of the
**  astrometric transformation chain.
**
**  Given:
**     date1  double       TDB as a 2-part...
**     date2  double       ...Julian Date (Note 1)
**     pv     double[2][3] observer's geocentric pos/vel (Note 3)
**
**  Returned:
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       unchanged
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) All the vectors are with respect to BCRS axes.
**
**  3) The observer's position and velocity pv are geocentric but with
**     respect to BCRS axes, and in units of m and m/s.  No assumptions
**     are made about proximity to the Earth, and the function can be
```

```
**     used for deep space applications as well as Earth orbit and
**     terrestrial.
**
**  4) In cases where the caller wishes to supply his own Earth
**     ephemeris, the function eraApcs can be used instead of the present
**     function.
**
**  5) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  6) The context structure astrom produced by this function is used by
**     eraAtciq* and eraAticq*.
**
**  Called:
**     eraEpv00     Earth position and velocity
**     eraApcs      astrometry parameters, ICRS-GCRS, space observer
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAper(double theta, eraASTROM *astrom)
/*
**  - - - - - - - -
**   e r a A p e r
**  - - - - - - - -
**
**  In the star-independent astrometry parameters, update only the
**  Earth rotation angle, supplied by the caller explicitly.
**
**  Given:
**     theta   double      Earth rotation angle (radians, Note 2)
**     astrom  eraASTROM*  star-independent astrometry parameters:
**      pmt    double       not used
**      eb     double[3]    not used
**      eh     double[3]    not used
**      em     double       not used
**      v      double[3]    not used
**      bm1    double       not used
**      bpn    double[3][3] not used
**      along  double       longitude + s' (radians)
**      xpl    double       not used
**      ypl    double       not used
**      sphi   double       not used
**      cphi   double       not used
**      diurab double       not used
**      eral   double       not used
**      refa   double       not used
**      refb   double       not used
**
**  Returned:
**     astrom  eraASTROM*  star-independent astrometry parameters:
**      pmt    double       unchanged
**      eb     double[3]    unchanged
**      eh     double[3]    unchanged
**      em     double       unchanged
**      v      double[3]    unchanged
**      bm1    double       unchanged
**      bpn    double[3][3] unchanged
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) This function exists to enable sidereal-tracking applications to
**     avoid wasteful recomputation of the bulk of the astrometry
**     parameters:  only the Earth rotation is updated.
**
**  2) For targets expressed as equinox based positions, such as
**     classical geocentric apparent (RA,Dec), the supplied theta can be
**     Greenwich apparent sidereal time rather than Earth rotation
**     angle.
**
**  3) The function eraAper13 can be used instead of the present
**     function, and starts from UT1 rather than ERA itself.
**
**  4) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
```

```
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAper13(double ut11, double ut12, eraASTROM *astrom)
/*
**  - - - - - - - - - - -
**   e r a A p e r 1 3
**  - - - - - - - - - - -
**
**  In the star-independent astrometry parameters, update only the
**  Earth rotation angle.  The caller provides UT1, (n.b. not UTC).
**
**  Given:
**     ut11    double      UT1 as a 2-part...
**     ut12    double      ...Julian Date (Note 1)
**     astrom  eraASTROM*  star-independent astrometry parameters:
**      pmt    double       not used
**      eb     double[3]    not used
**      eh     double[3]    not used
**      em     double       not used
**      v      double[3]    not used
**      bm1    double       not used
**      bpn    double[3][3] not used
**      along  double       longitude + s' (radians)
**      xpl    double       not used
**      ypl    double       not used
**      sphi   double       not used
**      cphi   double       not used
**      diurab double       not used
**      eral   double       not used
**      refa   double       not used
**      refb   double       not used
**
**  Returned:
**     astrom  eraASTROM*  star-independent astrometry parameters:
**      pmt    double       unchanged
**      eb     double[3]    unchanged
**      eh     double[3]    unchanged
**      em     double       unchanged
**      v      double[3]    unchanged
**      bm1    double       unchanged
**      bpn    double[3][3] unchanged
**      along  double       unchanged
**      xpl    double       unchanged
**      ypl    double       unchanged
**      sphi   double       unchanged
**      cphi   double       unchanged
**      diurab double       unchanged
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       unchanged
**      refb   double       unchanged
**
**  Notes:
**
**  1) The UT1 date (n.b. not UTC) ut11+ut12 is a Julian Date,
**     apportioned in any convenient way between the arguments ut11 and
**     ut12.  For example, JD(UT1)=2450123.7 could be expressed in any
**     of these ways, among others:
**
**            ut11           ut12
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  The date & time method is
**     best matched to the algorithm used:  maximum precision is
**     delivered when the ut11 argument is for 0hrs UT1 on the day in
**     question and the ut12 argument lies in the range 0 to 1, or vice
```

```
**     versa.
**
**  2) If the caller wishes to provide the Earth rotation angle itself,
**     the function eraAper can be used instead.  One use of this
**     technique is to substitute Greenwich apparent sidereal time and
**     thereby to support equinox based transformations directly.
**
**  3) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  Called:
**     eraAper      astrometry parameters: update ERA
**     eraEra00     Earth rotation angle, IAU 2000
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraApio(double sp, double theta,
             double elong, double phi, double hm, double xp, double yp,
             double refa, double refb,
             eraASTROM *astrom)
/*
**  - - - - - - - -
**   e r a A p i o
**  - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between CIRS and observed
**  coordinates.  The caller supplies the Earth orientation information
**  and the refraction constants as well as the site coordinates.
**
**  Given:
**     sp     double      the TIO locator s' (radians, Note 1)
**     theta  double      Earth rotation angle (radians)
**     elong  double      longitude (radians, east +ve, Note 2)
**     phi    double      geodetic latitude (radians, Note 2)
**     hm     double      height above ellipsoid (m, geodetic Note 2)
**     xp,yp  double      polar motion coordinates (radians, Note 3)
**     refa   double      refraction constant A (radians, Note 4)
**     refb   double      refraction constant B (radians, Note 4)
**
**  Returned:
**     astrom eraASTROM*  star-independent astrometry parameters:
**      pmt    double       unchanged
**      eb     double[3]    unchanged
**      eh     double[3]    unchanged
**      em     double       unchanged
**      v      double[3]    unchanged
**      bm1    double       unchanged
**      bpn    double[3][3] unchanged
**      along  double       longitude + s' (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Notes:
**
**  1) sp, the TIO locator s', is a tiny quantity needed only by the
**     most precise applications.  It can either be set to zero or
**     predicted using the ERFA function eraSp00.
**
**  2) The geographical coordinates are with respect to the ERFA_WGS84
**     reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**     longitude required by the present function is east-positive
**     (i.e. right-handed), in accordance with geographical convention.
**
**  3) The polar motion xp,yp can be obtained from IERS bulletins.  The
**     values are the coordinates (in radians) of the Celestial
**     Intermediate Pole with respect to the International Terrestrial
**     Reference System (see IERS Conventions 2003), measured along the
**     meridians 0 and 90 deg west respectively.  For many applications,
**     xp and yp can be set to zero.
**
**     Internally, the polar motion is stored in a form rotated onto the
**     local meridian.
**
**  4) The refraction constants refa and refb are for use in a
**     dZ = A*tan(Z)+B*tan^3(Z) model, where Z is the observed
**     (i.e. refracted) zenith distance and dZ is the amount of
**     refraction.
**
**  5) It is advisable to take great care with units, as even unlikely
```

```
**     values of the input parameters are accepted and processed in
**     accordance with the models used.
**
**  6) In cases where the caller does not wish to provide the Earth
**     rotation information and refraction constants, the function
**     eraApio13 can be used instead of the present function.  This
**     starts from UTC and weather readings etc. and computes suitable
**     values using other ERFA functions.
**
**  7) This is one of several functions that inserts into the astrom
**     structure star-independent parameters needed for the chain of
**     astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**     The various functions support different classes of observer and
**     portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**     Those with names ending in "13" use contemporary ERFA models to
**     compute the various ephemerides.  The others accept ephemerides
**     supplied by the caller.
**
**     The transformation from ICRS to GCRS covers space motion,
**     parallax, light deflection, and aberration.  From GCRS to CIRS
**     comprises frame bias and precession-nutation.  From CIRS to
**     observed takes account of Earth rotation, polar motion, diurnal
**     aberration and parallax (unless subsumed into the ICRS <-> GCRS
**     transformation), and atmospheric refraction.
**
**  8) The context structure astrom produced by this function is used by
**     eraAtioq and eraAtoiq.
**
**  Called:
**     eraPvtob     position/velocity of terrestrial station
**     eraAper      astrometry parameters: update ERA
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraApio13(double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              eraASTROM *astrom)
/*
**  - - - - - - - - - -
**   e r a A p i o 1 3
**  - - - - - - - - - -
**
**  For a terrestrial observer, prepare star-independent astrometry
**  parameters for transformations between CIRS and observed
**  coordinates.  The caller supplies UTC, site coordinates, ambient air
**  conditions and observing wavelength.
**
**  Given:
**     utc1   double      UTC as a 2-part...
**     utc2   double      ...quasi Julian Date (Notes 1,2)
**     dut1   double      UT1-UTC (seconds)
**     elong  double      longitude (radians, east +ve, Note 3)
**     phi    double      geodetic latitude (radians, Note 3)
**     hm     double      height above ellipsoid (m, geodetic Notes 4,6)
**     xp,yp  double      polar motion coordinates (radians, Note 5)
**     phpa   double      pressure at the observer (hPa = mB, Note 6)
**     tc     double      ambient temperature at the observer (deg C)
**     rh     double      relative humidity at the observer (range 0-1)
**     wl     double      wavelength (micrometers, Note 7)
**
**  Returned:
**     astrom eraASTROM*  star-independent astrometry parameters:
**      pmt    double       unchanged
**      eb     double[3]    unchanged
**      eh     double[3]    unchanged
**      em     double       unchanged
**      v      double[3]    unchanged
**      bm1    double       unchanged
**      bpn    double[3][3] unchanged
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned (function value):
**            int         status: +1 = dubious year (Note 2)
**                                 0 = OK
**                                -1 = unacceptable date
**
**  Notes:
**
**  1)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
**      it implements the leap-second-ambiguity convention just
**      described.
**
**  2)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the future
**      to be trusted.  See eraDat for further details.
```

```
**
**  3)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  4)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
**      (i.e. right-handed), in accordance with geographical convention.
**
**  5)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**      values are the coordinates (in radians) of the Celestial
**      Intermediate Pole with respect to the International Terrestrial
**      Reference System (see IERS Conventions 2003), measured along the
**      meridians 0 and 90 deg west respectively.  For many applications,
**      xp and yp can be set to zero.
**
**      Internally, the polar motion is stored in a form rotated onto
**      the local meridian.
**
**  6)  If hm, the height above the ellipsoid of the observing station
**      in meters, is not known but phpa, the pressure in hPa (=mB), is
**      available, an adequate estimate of hm can be obtained from the
**      expression
**
**            hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**      where tsl is the approximate sea-level air temperature in K
**      (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**      52).  Similarly, if the pressure phpa is not known, it can be
**      estimated from the height of the observing station, hm, as
**      follows:
**
**            phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**      Note, however, that the refraction is nearly proportional to the
**      pressure and that an accurate phpa value is important for
**      precise work.
**
**  7)  The argument wl specifies the observing wavelength in
**      micrometers.  The transition from optical to radio is assumed to
**      occur at 100 micrometers (about 3000 GHz).
**
**  8)  It is advisable to take great care with units, as even unlikely
**      values of the input parameters are accepted and processed in
**      accordance with the models used.
**
**  9)  In cases where the caller wishes to supply his own Earth
**      rotation information and refraction constants, the function
**      eraApc can be used instead of the present function.
**
**  10) This is one of several functions that inserts into the astrom
**      structure star-independent parameters needed for the chain of
**      astrometric transformations ICRS <-> GCRS <-> CIRS <-> observed.
**
**      The various functions support different classes of observer and
**      portions of the transformation chain:
**
**          functions         observer        transformation
**
**       eraApcg eraApcg13    geocentric      ICRS <-> GCRS
**       eraApci eraApci13    terrestrial     ICRS <-> CIRS
**       eraApco eraApco13    terrestrial     ICRS <-> observed
**       eraApcs eraApcs13    space           ICRS <-> GCRS
**       eraAper eraAper13    terrestrial     update Earth rotation
**       eraApio eraApio13    terrestrial     CIRS <-> observed
**
**      Those with names ending in "13" use contemporary ERFA models to
```

```
**      compute the various ephemerides.  The others accept ephemerides
**      supplied by the caller.
**
**      The transformation from ICRS to GCRS covers space motion,
**      parallax, light deflection, and aberration.  From GCRS to CIRS
**      comprises frame bias and precession-nutation.  From CIRS to
**      observed takes account of Earth rotation, polar motion, diurnal
**      aberration and parallax (unless subsumed into the ICRS <-> GCRS
**      transformation), and atmospheric refraction.
**
**  11) The context structure astrom produced by this function is used
**      by eraAtioq and eraAtoiq.
**
**  Called:
**     eraUtctai    UTC to TAI
**     eraTaitt     TAI to TT
**     eraUtcut1    UTC to UT1
**     eraSp00      the TIO locator s’, IERS 2000
**     eraEra00     Earth rotation angle, IAU 2000
**     eraRefco     refraction constants for given ambient conditions
**     eraApio      astrometry parameters, CIRS-observed
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtci13(double rc, double dc,
               double pr, double pd, double px, double rv,
               double date1, double date2,
               double *ri, double *di, double *eo)
/*
**  - - - - - - - - - -
**   e r a A t c i 1 3
**  - - - - - - - - - -
**
**  Transform ICRS star data, epoch J2000.0, to CIRS.
**
**  Given:
**     rc     double   ICRS right ascension at J2000.0 (radians, Note 1)
**     dc     double   ICRS declination at J2000.0 (radians, Note 1)
**     pr     double   RA proper motion (radians/year; Note 2)
**     pd     double   Dec proper motion (radians/year)
**     px     double   parallax (arcsec)
**     rv     double   radial velocity (km/s, +ve if receding)
**     date1  double   TDB as a 2-part...
**     date2  double   ...Julian Date (Note 3)
**
**  Returned:
**     ri,di  double*  CIRS geocentric RA,Dec (radians)
**     eo     double*  equation of the origins (ERA-GST, Note 5)
**
**  Notes:
**
**  1) Star data for an epoch other than J2000.0 (for example from the
**     Hipparcos catalog, which has an epoch of J1991.25) will require a
**     preliminary call to eraPmsafe before use.
**
**  2) The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  3) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  4) The available accuracy is better than 1 milliarcsecond, limited
**     mainly by the precession-nutation model that is used, namely
**     IAU 2000A/2006.  Very close to solar system bodies, additional
**     errors of up to several milliarcseconds can occur because of
**     unmodeled light deflection;  however, the Sun’s contribution is
**     taken into account, to first order.  The accuracy limitations of
**     the ERFA function eraEpv00 (used to compute Earth position and
**     velocity) can contribute aberration errors of up to
**     5 microarcseconds.  Light deflection at the Sun’s limb is
**     uncertain at the 0.4 mas level.
**
**  5) Should the transformation to (equinox based) apparent place be
**     required rather than (CIO based) intermediate place, subtract the
```

```
**     equation of the origins from the returned right ascension:
**     RA = RI - EO. (The eraAnp function can then be applied, as
**     required, to keep the result in the conventional 0-2pi range.)
**
**  Called:
**     eraApci13    astrometry parameters, ICRS-CIRS, 2013
**     eraAtciq     quick ICRS to CIRS
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtciq(double rc, double dc,
              double pr, double pd, double px, double rv,
              eraASTROM *astrom, double *ri, double *di)
/*
**  - - - - - - - - -
**   e r a A t c i q
**  - - - - - - - - -
**
**  Quick ICRS, epoch J2000.0, to CIRS transformation, given precomputed
**  star-independent astrometry parameters.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are to be transformed for one date.  The
**  star-independent parameters can be obtained by calling one of the
**  functions eraApci[13], eraApcg[13], eraApco[13] or eraApcs[13].
**
**  If the parallax and proper motions are zero the eraAtciqz function
**  can be used instead.
**
**  Given:
**     rc,dc  double     ICRS RA,Dec at J2000.0 (radians)
**     pr     double     RA proper motion (radians/year; Note 3)
**     pd     double     Dec proper motion (radians/year)
**     px     double     parallax (arcsec)
**     rv     double     radial velocity (km/s, +ve if receding)
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned:
**     ri,di   double    CIRS RA,Dec (radians)
**
**  Notes:
**
**  1) All the vectors are with respect to BCRS axes.
**
**  2) Star data for an epoch other than J2000.0 (for example from the
**     Hipparcos catalog, which has an epoch of J1991.25) will require a
**     preliminary call to eraPmsafe before use.
**
**  3) The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  Called:
**     eraPmpx      proper motion and parallax
**     eraLdsun     light deflection by the Sun
**     eraAb        stellar aberration
**     eraRxp       product of r-matrix and pv-vector
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtciqn(double rc, double dc, double pr, double pd,
               double px, double rv, eraASTROM *astrom,
               int n, eraLDBODY b[], double *ri, double *di)
/*
**  - - - - - - - - - -
**   e r a A t c i q n
**  - - - - - - - - - -
**
**  Quick ICRS, epoch J2000.0, to CIRS transformation, given precomputed
**  star-independent astrometry parameters plus a list of light-
**  deflecting bodies.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are to be transformed for one date.  The
**  star-independent parameters can be obtained by calling one of the
**  functions eraApci[13], eraApcg[13], eraApco[13] or eraApcs[13].
**
**
**  If the only light-deflecting body to be taken into account is the
**  Sun, the eraAtciq function can be used instead.  If in addition the
**  parallax and proper motions are zero, the eraAtciqz function can be
**  used.
**
**  Given:
**     rc,dc  double       ICRS RA,Dec at J2000.0 (radians)
**     pr     double       RA proper motion (radians/year; Note 3)
**     pd     double       Dec proper motion (radians/year)
**     px     double       parallax (arcsec)
**     rv     double       radial velocity (km/s, +ve if receding)
**     astrom eraASTROM*   star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s' (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**      n     int           number of bodies (Note 3)
**      b     eraLDBODY[n]  data for each of the n bodies (Notes 3,4):
**       bm    double        mass of the body (solar masses, Note 5)
**       dl    double        deflection limiter (Note 6)
**       pv    [2][3]        barycentric PV of the body (au, au/day)
**
**  Returned:
**     ri,di   double    CIRS RA,Dec (radians)
**
**  Notes:
**
**  1) Star data for an epoch other than J2000.0 (for example from the
**     Hipparcos catalog, which has an epoch of J1991.25) will require a
**     preliminary call to eraPmsafe before use.
**
**  2) The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  3) The struct b contains n entries, one for each body to be
**     considered.  If n = 0, no gravitational light deflection will be
**     applied, not even for the Sun.
**
**  4) The struct b should include an entry for the Sun as well as for
**     any planet or other body to be taken into account.  The entries
**     should be in the order in which the light passes the body.
```

```
**
**  5) In the entry in the b struct for body i, the mass parameter
**     b[i].bm can, as required, be adjusted in order to allow for such
**     effects as quadrupole field.
**
**  6) The deflection limiter parameter b[i].dl is phi^2/2, where phi is
**     the angular separation (in radians) between star and body at
**     which limiting is applied.  As phi shrinks below the chosen
**     threshold, the deflection is artificially reduced, reaching zero
**     for phi = 0.   Example values suitable for a terrestrial
**     observer, together with masses, are as follows:
**
**        body i     b[i].bm        b[i].dl
**
**        Sun        1.0            6e-6
**        Jupiter    0.00095435     3e-9
**        Saturn     0.00028574     3e-10
**
**  7) For efficiency, validation of the contents of the b array is
**     omitted.  The supplied masses must be greater than zero, the
**     position and velocity vectors must be right, and the deflection
**     limiter greater than zero.
**
**  Called:
**     eraPmpx      proper motion and parallax
**     eraLdn       light deflection by n bodies
**     eraAb        stellar aberration
**     eraRxp       product of r-matrix and pv-vector
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtciqz(double rc, double dc, eraASTROM *astrom,
               double *ri, double *di)
/*
**  - - - - - - - - - -
**   e r a A t c i q z
**  - - - - - - - - - -
**
**  Quick ICRS to CIRS transformation, given precomputed star-
**  independent astrometry parameters, and assuming zero parallax and
**  proper motion.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are to be transformed for one date.  The
**  star-independent parameters can be obtained by calling one of the
**  functions eraApci[13], eraApcg[13], eraApco[13] or eraApcs[13].
**
**  The corresponding function for the case of non-zero parallax and
**  proper motion is eraAtciq.
**
**  Given:
**     rc,dc  double     ICRS astrometric RA,Dec (radians)
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned:
**     ri,di  double     CIRS RA,Dec (radians)
**
**  Note:
**
**     All the vectors are with respect to BCRS axes.
**
**  References:
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013).
**
**     Klioner, Sergei A., "A practical relativistic model for micro-
**     arcsecond astrometry in space", Astr. J. 125, 1580-1597 (2003).
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraLdsun     light deflection due to Sun
**     eraAb        stellar aberration
**     eraRxp       product of r-matrix and p-vector
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range +/- pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraAtco13(double rc, double dc,
              double pr, double pd, double px, double rv,
              double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              double *aob, double *zob, double *hob,
              double *dob, double *rob, double *eo)
/*
**  - - - - - - - - - -
**   e r a A t c o 1 3
**  - - - - - - - - - -
**
**  ICRS RA,Dec to observed place.  The caller supplies UTC, site
**  coordinates, ambient air conditions and observing wavelength.
**
**  ERFA models are used for the Earth ephemeris, bias-precession-
**  nutation, Earth orientation and refraction.
**
**  Given:
**     rc,dc  double   ICRS right ascension at J2000.0 (radians, Note 1)
**     pr     double   RA proper motion (radians/year; Note 2)
**     pd     double   Dec proper motion (radians/year)
**     px     double   parallax (arcsec)
**     rv     double   radial velocity (km/s, +ve if receding)
**     utc1   double   UTC as a 2-part...
**     utc2   double   ...quasi Julian Date (Notes 3-4)
**     dut1   double   UT1-UTC (seconds, Note 5)
**     elong  double   longitude (radians, east +ve, Note 6)
**     phi    double   latitude (geodetic, radians, Note 6)
**     hm     double   height above ellipsoid (m, geodetic, Notes 6,8)
**     xp,yp  double   polar motion coordinates (radians, Note 7)
**     phpa   double   pressure at the observer (hPa = mB, Note 8)
**     tc     double   ambient temperature at the observer (deg C)
**     rh     double   relative humidity at the observer (range 0-1)
**     wl     double   wavelength (micrometers, Note 9)
**
**  Returned:
**     aob    double*  observed azimuth (radians: N=0,E=90)
**     zob    double*  observed zenith distance (radians)
**     hob    double*  observed hour angle (radians)
**     dob    double*  observed declination (radians)
**     rob    double*  observed right ascension (CIO-based, radians)
**     eo     double*  equation of the origins (ERA-GST)
**
**  Returned (function value):
**            int      status: +1 = dubious year (Note 4)
**                              0 = OK
**                             -1 = unacceptable date
**
**  Notes:
**
**  1)  Star data for an epoch other than J2000.0 (for example from the
**      Hipparcos catalog, which has an epoch of J1991.25) will require
**      a preliminary call to eraPmsafe before use.
**
**  2)  The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  3)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
**      it implements the leap-second-ambiguity convention just
**      described.
```

```
**
**  4)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the
**      future to be trusted.  See eraDat for further details.
**
**  5)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  6)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
**      (i.e. right-handed), in accordance with geographical convention.
**
**  7)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**      values are the coordinates (in radians) of the Celestial
**      Intermediate Pole with respect to the International Terrestrial
**      Reference System (see IERS Conventions 2003), measured along the
**      meridians 0 and 90 deg west respectively.  For many
**      applications, xp and yp can be set to zero.
**
**  8)  If hm, the height above the ellipsoid of the observing station
**      in meters, is not known but phpa, the pressure in hPa (=mB),
**      is available, an adequate estimate of hm can be obtained from
**      the expression
**
**            hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**      where tsl is the approximate sea-level air temperature in K
**      (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**      52).  Similarly, if the pressure phpa is not known, it can be
**      estimated from the height of the observing station, hm, as
**      follows:
**
**            phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**      Note, however, that the refraction is nearly proportional to
**      the pressure and that an accurate phpa value is important for
**      precise work.
**
**  9)  The argument wl specifies the observing wavelength in
**      micrometers.  The transition from optical to radio is assumed to
**      occur at 100 micrometers (about 3000 GHz).
**
**  10) The accuracy of the result is limited by the corrections for
**      refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**      Providing the meteorological parameters are known accurately and
**      there are no gross local effects, the predicted observed
**      coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**      (radio) for a zenith distance of less than 70 degrees, better
**      than 30 arcsec (optical or radio) at 85 degrees and better
**      than 20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**      Without refraction, the complementary functions eraAtco13 and
**      eraAtoc13 are self-consistent to better than 1 microarcsecond
**      all over the celestial sphere.  With refraction included,
**      consistency falls off at high zenith distances, but is still
**      better than 0.05 arcsec at 85 degrees.
**
**  11) "Observed" Az,ZD means the position that would be seen by a
**      perfect geodetically aligned theodolite.  (Zenith distance is
**      used rather than altitude in order to reflect the fact that no
**      allowance is made for depression of the horizon.)  This is
**      related to the observed HA,Dec via the standard rotation, using
**      the geodetic latitude (corrected for polar motion), while the
**      observed HA and RA are related simply through the Earth rotation
**      angle and the site longitude.  "Observed" RA,Dec or HA,Dec thus
**      means the position that would be seen by a perfect equatorial
```

```
**      with its polar axis aligned to the Earth's axis of rotation.
**
**  12) It is advisable to take great care with units, as even unlikely
**      values of the input parameters are accepted and processed in
**      accordance with the models used.
**
**  Called:
**     eraApco13    astrometry parameters, ICRS-observed, 2013
**     eraAtciq     quick ICRS to CIRS
**     eraAtioq     quick CIRS to observed
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtic13(double ri, double di, double date1, double date2,
               double *rc, double *dc, double *eo)
/*
**  - - - - - - - - - -
**   e r a A t i c 1 3
**  - - - - - - - - - -
**
**  Transform star RA,Dec from geocentric CIRS to ICRS astrometric.
**
**  Given:
**     ri,di  double  CIRS geocentric RA,Dec (radians)
**     date1  double  TDB as a 2-part...
**     date2  double  ...Julian Date (Note 1)
**
**  Returned:
**     rc,dc  double  ICRS astrometric RA,Dec (radians)
**     eo     double  equation of the origins (ERA-GST, Note 4)
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  For most
**     applications of this function the choice will not be at all
**     critical.
**
**     TT can be used instead of TDB without any significant impact on
**     accuracy.
**
**  2) Iterative techniques are used for the aberration and light
**     deflection corrections so that the functions eraAtic13 (or
**     eraAticq) and eraAtci13 (or eraAtciq) are accurate inverses;
**     even at the edge of the Sun’s disk the discrepancy is only about
**     1 nanoarcsecond.
**
**  3) The available accuracy is better than 1 milliarcsecond, limited
**     mainly by the precession-nutation model that is used, namely
**     IAU 2000A/2006.  Very close to solar system bodies, additional
**     errors of up to several milliarcseconds can occur because of
**     unmodeled light deflection;  however, the Sun’s contribution is
**     taken into account, to first order.  The accuracy limitations of
**     the ERFA function eraEpv00 (used to compute Earth position and
**     velocity) can contribute aberration errors of up to
**     5 microarcseconds.  Light deflection at the Sun’s limb is
**     uncertain at the 0.4 mas level.
**
**  4) Should the transformation to (equinox based) J2000.0 mean place
**     be required rather than (CIO based) ICRS coordinates, subtract the
**     equation of the origins from the returned right ascension:
**     RA = RI - EO.  (The eraAnp function can then be applied, as
**     required, to keep the result in the conventional 0-2pi range.)
**
**  Called:
**     eraApci13    astrometry parameters, ICRS-CIRS, 2013
**     eraAticq     quick CIRS to ICRS astrometric
```

```
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAticq(double ri, double di, eraASTROM *astrom,
              double *rc, double *dc)
/*
**  - - - - - - - - -
**   e r a A t i c q
**  - - - - - - - - -
**
**  Quick CIRS RA,Dec to ICRS astrometric place, given the star-
**  independent astrometry parameters.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are all to be transformed for one date.
**  The star-independent astrometry parameters can be obtained by
**  calling one of the functions eraApci[13], eraApcg[13], eraApco[13]
**  or eraApcs[13].
**
**  Given:
**     ri,di  double     CIRS RA,Dec (radians)
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s' (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned:
**     rc,dc  double     ICRS astrometric RA,Dec (radians)
**
**  Notes:
**
**  1) Only the Sun is taken into account in the light deflection
**     correction.
**
**  2) Iterative techniques are used for the aberration and light
**     deflection corrections so that the functions eraAtic13 (or
**     eraAticq) and eraAtci13 (or eraAtciq) are accurate inverses;
**     even at the edge of the Sun's disk the discrepancy is only about
**     1 nanoarcsecond.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraZp        zero p-vector
**     eraAb        stellar aberration
**     eraLdsun     light deflection by the Sun
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range +/- pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAticqn(double ri, double di, eraASTROM *astrom,
               int n, eraLDBODY b[], double *rc, double *dc)
/*
**  - - - - - - - - - -
**   e r a A t i c q n
**  - - - - - - - - - -
**
**  Quick CIRS to ICRS astrometric place transformation, given the star-
**  independent astrometry parameters plus a list of light-deflecting
**  bodies.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are all to be transformed for one date.
**  The star-independent astrometry parameters can be obtained by
**  calling one of the functions eraApci[13], eraApcg[13], eraApco[13]
**  or eraApcs[13].
```

```
int eraAtio13(double ri, double di,
              double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              double *aob, double *zob, double *hob,
              double *dob, double *rob)
/*
**  - - - - - - - - - -
**   e r a A t i o 1 3
**  - - - - - - - - - -
**
**  CIRS RA,Dec to observed place.  The caller supplies UTC, site
**  coordinates, ambient air conditions and observing wavelength.
**
**  Given:
**     ri     double   CIRS right ascension (CIO-based, radians)
**     di     double   CIRS declination (radians)
**     utc1   double   UTC as a 2-part...
**     utc2   double   ...quasi Julian Date (Notes 1,2)
**     dut1   double   UT1-UTC (seconds, Note 3)
**     elong  double   longitude (radians, east +ve, Note 4)
**     phi    double   geodetic latitude (radians, Note 4)
**     hm     double   height above ellipsoid (m, geodetic Notes 4,6)
**     xp,yp  double   polar motion coordinates (radians, Note 5)
**     phpa   double   pressure at the observer (hPa = mB, Note 6)
**     tc     double   ambient temperature at the observer (deg C)
**     rh     double   relative humidity at the observer (range 0-1)
**     wl     double   wavelength (micrometers, Note 7)
**
**  Returned:
**     aob    double*  observed azimuth (radians: N=0,E=90)
**     zob    double*  observed zenith distance (radians)
**     hob    double*  observed hour angle (radians)
**     dob    double*  observed declination (radians)
**     rob    double*  observed right ascension (CIO-based, radians)
**
**  Returned (function value):
**            int      status: +1 = dubious year (Note 2)
**                              0 = OK
**                             -1 = unacceptable date
**
**  Notes:
**
**  1)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
**      it implements the leap-second-ambiguity convention just
**      described.
**
**  2)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the
**      future to be trusted.  See eraDat for further details.
**
**  3)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  4)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
```

```
**      (i.e. right-handed), in accordance with geographical convention.
**
**   5)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**       values are the coordinates (in radians) of the Celestial
**       Intermediate Pole with respect to the International Terrestrial
**       Reference System (see IERS Conventions 2003), measured along the
**       meridians 0 and 90 deg west respectively.  For many
**       applications, xp and yp can be set to zero.
**
**   6)  If hm, the height above the ellipsoid of the observing station
**       in meters, is not known but phpa, the pressure in hPa (=mB), is
**       available, an adequate estimate of hm can be obtained from the
**       expression
**
**             hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**       where tsl is the approximate sea-level air temperature in K
**       (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**       52).  Similarly, if the pressure phpa is not known, it can be
**       estimated from the height of the observing station, hm, as
**       follows:
**
**             phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**       Note, however, that the refraction is nearly proportional to
**       the pressure and that an accurate phpa value is important for
**       precise work.
**
**   7)  The argument wl specifies the observing wavelength in
**       micrometers.  The transition from optical to radio is assumed to
**       occur at 100 micrometers (about 3000 GHz).
**
**   8)  "Observed" Az,ZD means the position that would be seen by a
**       perfect geodetically aligned theodolite.  (Zenith distance is
**       used rather than altitude in order to reflect the fact that no
**       allowance is made for depression of the horizon.)  This is
**       related to the observed HA,Dec via the standard rotation, using
**       the geodetic latitude (corrected for polar motion), while the
**       observed HA and RA are related simply through the Earth rotation
**       angle and the site longitude.  "Observed" RA,Dec or HA,Dec thus
**       means the position that would be seen by a perfect equatorial
**       with its polar axis aligned to the Earth's axis of rotation.
**
**   9)  The accuracy of the result is limited by the corrections for
**       refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**       Providing the meteorological parameters are known accurately and
**       there are no gross local effects, the predicted astrometric
**       coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**       (radio) for a zenith distance of less than 70 degrees, better
**       than 30 arcsec (optical or radio) at 85 degrees and better
**       than 20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**   10) The complementary functions eraAtio13 and eraAtoi13 are self-
**       consistent to better than 1 microarcsecond all over the
**       celestial sphere.
**
**   11) It is advisable to take great care with units, as even unlikely
**       values of the input parameters are accepted and processed in
**       accordance with the models used.
**
**   Called:
**      eraApio13    astrometry parameters, CIRS-observed, 2013
**      eraAtioq     quick CIRS to observed
**
**   Copyright (C) 2013-2018, NumFOCUS Foundation.
**   Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtioq(double ri, double di, eraASTROM *astrom,
              double *aob, double *zob,
              double *hob, double *dob, double *rob)
/*
**  - - - - - - - - -
**   e r a A t i o q
**  - - - - - - - - -
**
**  Quick CIRS to observed place transformation.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are all to be transformed for one date.
**  The star-independent astrometry parameters can be obtained by
**  calling eraApio[13] or eraApco[13].
**
**  Given:
**     ri     double     CIRS right ascension
**     di     double     CIRS declination
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned:
**     aob    double*    observed azimuth (radians: N=0,E=90)
**     zob    double*    observed zenith distance (radians)
**     hob    double*    observed hour angle (radians)
**     dob    double*    observed declination (radians)
**     rob    double*    observed right ascension (CIO-based, radians)
**
**  Notes:
**
**  1) This function returns zenith distance rather than altitude in
**     order to reflect the fact that no allowance is made for
**     depression of the horizon.
**
**  2) The accuracy of the result is limited by the corrections for
**     refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**     Providing the meteorological parameters are known accurately and
**     there are no gross local effects, the predicted observed
**     coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**     (radio) for a zenith distance of less than 70 degrees, better
**     than 30 arcsec (optical or radio) at 85 degrees and better
**     than 20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**     Without refraction, the complementary functions eraAtioq and
**     eraAtoiq are self-consistent to better than 1 microarcsecond all
**     over the celestial sphere.  With refraction included, consistency
**     falls off at high zenith distances, but is still better than
**     0.05 arcsec at 85 degrees.
**
**  3) It is advisable to take great care with units, as even unlikely
**     values of the input parameters are accepted and processed in
**     accordance with the models used.
**
**  4) The CIRS RA,Dec is obtained from a star catalog mean place by
**     allowing for space motion, parallax, the Sun’s gravitational lens
```

```
**     effect, annual aberration and precession-nutation.  For star
**     positions in the ICRS, these effects can be applied by means of
**     the eraAtci13 (etc.) functions.  Starting from classical "mean
**     place" systems, additional transformations will be needed first.
**
**  5) "Observed" Az,El means the position that would be seen by a
**     perfect geodetically aligned theodolite.  This is obtained from
**     the CIRS RA,Dec by allowing for Earth orientation and diurnal
**     aberration, rotating from equator to horizon coordinates, and
**     then adjusting for refraction.  The HA,Dec is obtained by
**     rotating back into equatorial coordinates, and is the position
**     that would be seen by a perfect equatorial with its polar axis
**     aligned to the Earth's axis of rotation.  Finally, the RA is
**     obtained by subtracting the HA from the local ERA.
**
**  6) The star-independent CIRS-to-observed-place parameters in ASTROM
**     may be computed with eraApio[13] or eraApco[13].  If nothing has
**     changed significantly except the time, eraAper[13] may be used to
**     perform the requisite adjustment to the astrom structure.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraAtoc13(const char *type, double ob1, double ob2,
              double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              double *rc, double *dc)
/*
**  - - - - - - - - - -
**   e r a A t o c 1 3
**  - - - - - - - - - -
**
**  Observed place at a groundbased site to to ICRS astrometric RA,Dec.
**  The caller supplies UTC, site coordinates, ambient air conditions
**  and observing wavelength.
**
**  Given:
**     type   char[]   type of coordinates - "R", "H" or "A" (Notes 1,2)
**     ob1    double   observed Az, HA or RA (radians; Az is N=0,E=90)
**     ob2    double   observed ZD or Dec (radians)
**     utc1   double   UTC as a 2-part...
**     utc2   double   ...quasi Julian Date (Notes 3,4)
**     dut1   double   UT1-UTC (seconds, Note 5)
**     elong  double   longitude (radians, east +ve, Note 6)
**     phi    double   geodetic latitude (radians, Note 6)
**     hm     double   height above ellipsoid (m, geodetic Notes 6,8)
**     xp,yp  double   polar motion coordinates (radians, Note 7)
**     phpa   double   pressure at the observer (hPa = mB, Note 8)
**     tc     double   ambient temperature at the observer (deg C)
**     rh     double   relative humidity at the observer (range 0-1)
**     wl     double   wavelength (micrometers, Note 9)
**
**  Returned:
**     rc,dc  double   ICRS astrometric RA,Dec (radians)
**
**  Returned (function value):
**            int      status: +1 = dubious year (Note 4)
**                              0 = OK
**                             -1 = unacceptable date
**
**  Notes:
**
**  1)  "Observed" Az,ZD means the position that would be seen by a
**      perfect geodetically aligned theodolite.  (Zenith distance is
**      used rather than altitude in order to reflect the fact that no
**      allowance is made for depression of the horizon.)  This is
**      related to the observed HA,Dec via the standard rotation, using
**      the geodetic latitude (corrected for polar motion), while the
**      observed HA and RA are related simply through the Earth rotation
**      angle and the site longitude.  "Observed" RA,Dec or HA,Dec thus
**      means the position that would be seen by a perfect equatorial
**      with its polar axis aligned to the Earth’s axis of rotation.
**
**  2)  Only the first character of the type argument is significant.
**      "R" or "r" indicates that ob1 and ob2 are the observed right
**      ascension and declination;  "H" or "h" indicates that they are
**      hour angle (west +ve) and declination;  anything else ("A" or
**      "a" is recommended) indicates that ob1 and ob2 are azimuth
**      (north zero, east 90 deg) and zenith distance.
**
**  3)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
**      it implements the leap-second-ambiguity convention just
```

```
**      described.
**
**  4)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the
**      future to be trusted.  See eraDat for further details.
**
**  5)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  6)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
**      (i.e. right-handed), in accordance with geographical convention.
**
**  7)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**      values are the coordinates (in radians) of the Celestial
**      Intermediate Pole with respect to the International Terrestrial
**      Reference System (see IERS Conventions 2003), measured along the
**      meridians 0 and 90 deg west respectively.  For many
**      applications, xp and yp can be set to zero.
**
**  8)  If hm, the height above the ellipsoid of the observing station
**      in meters, is not known but phpa, the pressure in hPa (=mB), is
**      available, an adequate estimate of hm can be obtained from the
**      expression
**
**            hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**      where tsl is the approximate sea-level air temperature in K
**      (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**      52).  Similarly, if the pressure phpa is not known, it can be
**      estimated from the height of the observing station, hm, as
**      follows:
**
**            phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**      Note, however, that the refraction is nearly proportional to
**      the pressure and that an accurate phpa value is important for
**      precise work.
**
**  9)  The argument wl specifies the observing wavelength in
**      micrometers.  The transition from optical to radio is assumed to
**      occur at 100 micrometers (about 3000 GHz).
**
**  10) The accuracy of the result is limited by the corrections for
**      refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**      Providing the meteorological parameters are known accurately and
**      there are no gross local effects, the predicted astrometric
**      coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**      (radio) for a zenith distance of less than 70 degrees, better
**      than 30 arcsec (optical or radio) at 85 degrees and better
**      than 20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**      Without refraction, the complementary functions eraAtco13 and
**      eraAtoc13 are self-consistent to better than 1 microarcsecond
**      all over the celestial sphere.  With refraction included,
**      consistency falls off at high zenith distances, but is still
**      better than 0.05 arcsec at 85 degrees.
**
**  11) It is advisable to take great care with units, as even unlikely
**      values of the input parameters are accepted and processed in
**      accordance with the models used.
**
**  Called:
**     eraApco13    astrometry parameters, ICRS-observed
**     eraAtoiq     quick observed to CIRS
**     eraAticq     quick CIRS to ICRS
```

```
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraAtoi13(const char *type, double ob1, double ob2,
              double utc1, double utc2, double dut1,
              double elong, double phi, double hm, double xp, double yp,
              double phpa, double tc, double rh, double wl,
              double *ri, double *di)
/*
**  - - - - - - - - - - -
**   e r a A t o i 1 3
**  - - - - - - - - - - -
**
**  Observed place to CIRS.  The caller supplies UTC, site coordinates,
**  ambient air conditions and observing wavelength.
**
**  Given:
**     type   char[]   type of coordinates - "R", "H" or "A" (Notes 1,2)
**     ob1    double   observed Az, HA or RA (radians; Az is N=0,E=90)
**     ob2    double   observed ZD or Dec (radians)
**     utc1   double   UTC as a 2-part...
**     utc2   double   ...quasi Julian Date (Notes 3,4)
**     dut1   double   UT1-UTC (seconds, Note 5)
**     elong  double   longitude (radians, east +ve, Note 6)
**     phi    double   geodetic latitude (radians, Note 6)
**     hm     double   height above the ellipsoid (meters, Notes 6,8)
**     xp,yp  double   polar motion coordinates (radians, Note 7)
**     phpa   double   pressure at the observer (hPa = mB, Note 8)
**     tc     double   ambient temperature at the observer (deg C)
**     rh     double   relative humidity at the observer (range 0-1)
**     wl     double   wavelength (micrometers, Note 9)
**
**  Returned:
**     ri     double*  CIRS right ascension (CIO-based, radians)
**     di     double*  CIRS declination (radians)
**
**  Returned (function value):
**            int      status: +1 = dubious year (Note 2)
**                              0 = OK
**                             -1 = unacceptable date
**
**  Notes:
**
**  1)  "Observed" Az,ZD means the position that would be seen by a
**      perfect geodetically aligned theodolite.  (Zenith distance is
**      used rather than altitude in order to reflect the fact that no
**      allowance is made for depression of the horizon.)  This is
**      related to the observed HA,Dec via the standard rotation, using
**      the geodetic latitude (corrected for polar motion), while the
**      observed HA and RA are related simply through the Earth rotation
**      angle and the site longitude.  "Observed" RA,Dec or HA,Dec thus
**      means the position that would be seen by a perfect equatorial
**      with its polar axis aligned to the Earth’s axis of rotation.
**
**  2)  Only the first character of the type argument is significant.
**      "R" or "r" indicates that ob1 and ob2 are the observed right
**      ascension and declination;  "H" or "h" indicates that they are
**      hour angle (west +ve) and declination;  anything else ("A" or
**      "a" is recommended) indicates that ob1 and ob2 are azimuth
**      (north zero, east 90 deg) and zenith distance.
**
**  3)  utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**      convenient way between the two arguments, for example where utc1
**      is the Julian Day Number and utc2 is the fraction of a day.
**
**      However, JD cannot unambiguously represent UTC during a leap
**      second unless special measures are taken.  The convention in the
**      present function is that the JD day represents UTC days whether
**      the length is 86399, 86400 or 86401 SI seconds.
**
**      Applications should use the function eraDtf2d to convert from
**      calendar date and time of day into 2-part quasi Julian Date, as
**      it implements the leap-second-ambiguity convention just
```

```
**      described.
**
**  4)  The warning status "dubious year" flags UTCs that predate the
**      introduction of the time scale or that are too far in the
**      future to be trusted.  See eraDat for further details.
**
**  5)  UT1-UTC is tabulated in IERS bulletins.  It increases by exactly
**      one second at the end of each positive UTC leap second,
**      introduced in order to keep UT1-UTC within +/- 0.9s.  n.b. This
**      practice is under review, and in the future UT1-UTC may grow
**      essentially without limit.
**
**  6)  The geographical coordinates are with respect to the ERFA_WGS84
**      reference ellipsoid.  TAKE CARE WITH THE LONGITUDE SIGN:  the
**      longitude required by the present function is east-positive
**      (i.e. right-handed), in accordance with geographical convention.
**
**  7)  The polar motion xp,yp can be obtained from IERS bulletins.  The
**      values are the coordinates (in radians) of the Celestial
**      Intermediate Pole with respect to the International Terrestrial
**      Reference System (see IERS Conventions 2003), measured along the
**      meridians 0 and 90 deg west respectively.  For many
**      applications, xp and yp can be set to zero.
**
**  8)  If hm, the height above the ellipsoid of the observing station
**      in meters, is not known but phpa, the pressure in hPa (=mB), is
**      available, an adequate estimate of hm can be obtained from the
**      expression
**
**            hm = -29.3 * tsl * log ( phpa / 1013.25 );
**
**      where tsl is the approximate sea-level air temperature in K
**      (See Astrophysical Quantities, C.W.Allen, 3rd edition, section
**      52).  Similarly, if the pressure phpa is not known, it can be
**      estimated from the height of the observing station, hm, as
**      follows:
**
**            phpa = 1013.25 * exp ( -hm / ( 29.3 * tsl ) );
**
**      Note, however, that the refraction is nearly proportional to
**      the pressure and that an accurate phpa value is important for
**      precise work.
**
**  9)  The argument wl specifies the observing wavelength in
**      micrometers.  The transition from optical to radio is assumed to
**      occur at 100 micrometers (about 3000 GHz).
**
**  10) The accuracy of the result is limited by the corrections for
**      refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**      Providing the meteorological parameters are known accurately and
**      there are no gross local effects, the predicted astrometric
**      coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**      (radio) for a zenith distance of less than 70 degrees, better
**      than 30 arcsec (optical or radio) at 85 degrees and better
**      than 20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**      Without refraction, the complementary functions eraAtio13 and
**      eraAtoi13 are self-consistent to better than 1 microarcsecond
**      all over the celestial sphere.  With refraction included,
**      consistency falls off at high zenith distances, but is still
**      better than 0.05 arcsec at 85 degrees.
**
**  12) It is advisable to take great care with units, as even unlikely
**      values of the input parameters are accepted and processed in
**      accordance with the models used.
**
**  Called:
**     eraApio13    astrometry parameters, CIRS-observed, 2013
**     eraAtoiq     quick observed to CIRS
**
```

```
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraAtoiq(const char *type,
              double ob1, double ob2, eraASTROM *astrom,
              double *ri, double *di)
/*
**  - - - - - - - - -
**   e r a A t o i q
**  - - - - - - - - -
**
**  Quick observed place to CIRS, given the star-independent astrometry
**  parameters.
**
**  Use of this function is appropriate when efficiency is important and
**  where many star positions are all to be transformed for one date.
**  The star-independent astrometry parameters can be obtained by
**  calling eraApio[13] or eraApco[13].
**
**  Given:
**     type   char[]     type of coordinates: "R", "H" or "A" (Note 1)
**     ob1    double     observed Az, HA or RA (radians; Az is N=0,E=90)
**     ob2    double     observed ZD or Dec (radians)
**     astrom eraASTROM* star-independent astrometry parameters:
**      pmt    double       PM time interval (SSB, Julian years)
**      eb     double[3]    SSB to observer (vector, au)
**      eh     double[3]    Sun to observer (unit vector)
**      em     double       distance from Sun to observer (au)
**      v      double[3]    barycentric observer velocity (vector, c)
**      bm1    double       sqrt(1-|v|^2): reciprocal of Lorenz factor
**      bpn    double[3][3] bias-precession-nutation matrix
**      along  double       longitude + s’ (radians)
**      xpl    double       polar motion xp wrt local meridian (radians)
**      ypl    double       polar motion yp wrt local meridian (radians)
**      sphi   double       sine of geodetic latitude
**      cphi   double       cosine of geodetic latitude
**      diurab double       magnitude of diurnal aberration vector
**      eral   double       "local" Earth rotation angle (radians)
**      refa   double       refraction constant A (radians)
**      refb   double       refraction constant B (radians)
**
**  Returned:
**     ri     double*    CIRS right ascension (CIO-based, radians)
**     di     double*    CIRS declination (radians)
**
**  Notes:
**
**  1) "Observed" Az,El means the position that would be seen by a
**     perfect geodetically aligned theodolite.  This is related to
**     the observed HA,Dec via the standard rotation, using the geodetic
**     latitude (corrected for polar motion), while the observed HA and
**     RA are related simply through the Earth rotation angle and the
**     site longitude.  "Observed" RA,Dec or HA,Dec thus means the
**     position that would be seen by a perfect equatorial with its
**     polar axis aligned to the Earth’s axis of rotation.  By removing
**     from the observed place the effects of atmospheric refraction and
**     diurnal aberration, the CIRS RA,Dec is obtained.
**
**  2) Only the first character of the type argument is significant.
**     "R" or "r" indicates that ob1 and ob2 are the observed right
**     ascension and declination;  "H" or "h" indicates that they are
**     hour angle (west +ve) and declination;  anything else ("A" or
**     "a" is recommended) indicates that ob1 and ob2 are azimuth (north
**     zero, east 90 deg) and zenith distance.  (Zenith distance is used
**     rather than altitude in order to reflect the fact that no
**     allowance is made for depression of the horizon.)
**
**  3) The accuracy of the result is limited by the corrections for
**     refraction, which use a simple A*tan(z) + B*tan^3(z) model.
**     Providing the meteorological parameters are known accurately and
**     there are no gross local effects, the predicted observed
**     coordinates should be within 0.05 arcsec (optical) or 1 arcsec
**     (radio) for a zenith distance of less than 70 degrees, better
```

```
**     than 30 arcsec (optical or radio) at 85 degrees and better than
**     20 arcmin (optical) or 30 arcmin (radio) at the horizon.
**
**     Without refraction, the complementary functions eraAtioq and
**     eraAtoiq are self-consistent to better than 1 microarcsecond all
**     over the celestial sphere.  With refraction included, consistency
**     falls off at high zenith distances, but is still better than
**     0.05 arcsec at 85 degrees.
**
**  4) It is advisable to take great care with units, as even unlikely
**     values of the input parameters are accepted and processed in
**     accordance with the models used.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLd(double bm, double p[3], double q[3], double e[3],
           double em, double dlim, double p1[3])
/*
**  - - - - - -
**   e r a L d
**  - - - - - -
**
**  Apply light deflection by a solar-system body, as part of
**  transforming coordinate direction into natural direction.
**
**  Given:
**     bm     double     mass of the gravitating body (solar masses)
**     p      double[3]  direction from observer to source (unit vector)
**     q      double[3]  direction from body to source (unit vector)
**     e      double[3]  direction from body to observer (unit vector)
**     em     double     distance from body to observer (au)
**     dlim   double     deflection limiter (Note 4)
**
**  Returned:
**     p1     double[3]  observer to deflected source (unit vector)
**
**  Notes:
**
**  1) The algorithm is based on Expr. (70) in Klioner (2003) and
**     Expr. (7.63) in the Explanatory Supplement (Urban & Seidelmann
**     2013), with some rearrangement to minimize the effects of machine
**     precision.
**
**  2) The mass parameter bm can, as required, be adjusted in order to
**     allow for such effects as quadrupole field.
**
**  3) The barycentric position of the deflecting body should ideally
**     correspond to the time of closest approach of the light ray to
**     the body.
**
**  4) The deflection limiter parameter dlim is phi^2/2, where phi is
**     the angular separation (in radians) between source and body at
**     which limiting is applied.  As phi shrinks below the chosen
**     threshold, the deflection is artificially reduced, reaching zero
**     for phi = 0.
**
**  5) The returned vector p1 is not normalized, but the consequential
**     departure from unit magnitude is always negligible.
**
**  6) The arguments p and p1 can be the same array.
**
**  7) To accumulate total light deflection taking into account the
**     contributions from several bodies, call the present function for
**     each body in succession, in decreasing order of distance from the
**     observer.
**
**  8) For efficiency, validation is omitted.  The supplied vectors must
**     be of unit magnitude, and the deflection limiter non-zero and
**     positive.
**
**  References:
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013).
**
**     Klioner, Sergei A., "A practical relativistic model for micro-
**     arcsecond astrometry in space", Astr. J. 125, 1580-1597 (2003).
**
**  Called:
**     eraPdp       scalar product of two p-vectors
**     eraPxp       vector product of two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
```

```
*/
```

```
void eraLdn(int n, eraLDBODY b[], double ob[3], double sc[3],
            double sn[3])
/*+
**  - - - - - - -
**   e r a L d n
**  - - - - - - -
**
**  For a star, apply light deflection by multiple solar-system bodies,
**  as part of transforming coordinate direction into natural direction.
**
**  Given:
**     n    int           number of bodies (note 1)
**     b    eraLDBODY[n]  data for each of the n bodies (Notes 1,2):
**      bm   double         mass of the body (solar masses, Note 3)
**      dl   double         deflection limiter (Note 4)
**      pv   [2][3]         barycentric PV of the body (au, au/day)
**     ob   double[3]     barycentric position of the observer (au)
**     sc   double[3]     observer to star coord direction (unit vector)
**
**  Returned:
**     sn    double[3]      observer to deflected star (unit vector)
**
**  1) The array b contains n entries, one for each body to be
**     considered.  If n = 0, no gravitational light deflection will be
**     applied, not even for the Sun.
**
**  2) The array b should include an entry for the Sun as well as for
**     any planet or other body to be taken into account.  The entries
**     should be in the order in which the light passes the body.
**
**  3) In the entry in the b array for body i, the mass parameter
**     b[i].bm can, as required, be adjusted in order to allow for such
**     effects as quadrupole field.
**
**  4) The deflection limiter parameter b[i].dl is phi^2/2, where phi is
**     the angular separation (in radians) between star and body at
**     which limiting is applied.  As phi shrinks below the chosen
**     threshold, the deflection is artificially reduced, reaching zero
**     for phi = 0.   Example values suitable for a terrestrial
**     observer, together with masses, are as follows:
**
**        body i     b[i].bm        b[i].dl
**
**        Sun        1.0            6e-6
**        Jupiter    0.00095435     3e-9
**        Saturn     0.00028574     3e-10
**
**  5) For cases where the starlight passes the body before reaching the
**     observer, the body is placed back along its barycentric track by
**     the light time from that point to the observer.  For cases where
**     the body is "behind" the observer no such shift is applied.  If
**     a different treatment is preferred, the user has the option of
**     instead using the eraLd function.  Similarly, eraLd can be used
**     for cases where the source is nearby, not a star.
**
**  6) The returned vector sn is not normalized, but the consequential
**     departure from unit magnitude is always negligible.
**
**  7) The arguments sc and sn can be the same array.
**
**  8) For efficiency, validation is omitted.  The supplied masses must
**     be greater than zero, the position and velocity vectors must be
**     right, and the deflection limiter greater than zero.
**
**  Reference:
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013), Section 7.2.4.
**
```

```
**  Called:
**     eraCp        copy p-vector
**     eraPdp       scalar product of two p-vectors
**     eraPmp       p-vector minus p-vector
**     eraPpsp      p-vector plus scaled p-vector
**     eraPn        decompose p-vector into modulus and direction
**     eraLd        light deflection by a solar-system body
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLdsun(double p[3], double e[3], double em, double p1[3])
/*
**  - - - - - - - - -
**   e r a L d s u n
**  - - - - - - - - -
**
**  Deflection of starlight by the Sun.
**
**  Given:
**     p      double[3]  direction from observer to star (unit vector)
**     e      double[3]  direction from Sun to observer (unit vector)
**     em     double     distance from Sun to observer (au)
**
**  Returned:
**     p1     double[3]  observer to deflected star (unit vector)
**
**  Notes:
**
**  1) The source is presumed to be sufficiently distant that its
**     directions seen from the Sun and the observer are essentially
**     the same.
**
**  2) The deflection is restrained when the angle between the star and
**     the center of the Sun is less than a threshold value, falling to
**     zero deflection for zero separation.  The chosen threshold value
**     is within the solar limb for all solar-system applications, and
**     is about 5 arcminutes for the case of a terrestrial observer.
**
**  3) The arguments p and p1 can be the same array.
**
**  Called:
**     eraLd        light deflection by a solar-system body
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPmpx(double rc, double dc, double pr, double pd,
             double px, double rv, double pmt, double pob[3],
             double pco[3])
/*
**  - - - - - - - - -
**   e r a P m p x
**  - - - - - - - - -
**
**  Proper motion and parallax.
**
**  Given:
**     rc,dc  double     ICRS RA,Dec at catalog epoch (radians)
**     pr     double     RA proper motion (radians/year; Note 1)
**     pd     double     Dec proper motion (radians/year)
**     px     double     parallax (arcsec)
**     rv     double     radial velocity (km/s, +ve if receding)
**     pmt    double     proper motion time interval (SSB, Julian years)
**     pob    double[3]  SSB to observer vector (au)
**
**  Returned:
**     pco    double[3]  coordinate direction (BCRS unit vector)
**
**  Notes:
**
**  1) The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  2) The proper motion time interval is for when the starlight
**     reaches the solar system barycenter.
**
**  3) To avoid the need for iteration, the Roemer effect (i.e. the
**     small annual modulation of the proper motion coming from the
**     changing light time) is applied approximately, using the
**     direction of the star at the catalog epoch.
**
**  References:
**
**     1984 Astronomical Almanac, pp B39-B41.
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013), Section 7.2.
**
**  Called:
**     eraPdp       scalar product of two p-vectors
**     eraPn        decompose p-vector into modulus and direction
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraPmsafe(double ra1, double dec1, double pmr1, double pmd1,
              double px1, double rv1,
              double ep1a, double ep1b, double ep2a, double ep2b,
              double *ra2, double *dec2, double *pmr2, double *pmd2,
              double *px2, double *rv2)
/*
**  - - - - - - - - - -
**   e r a P m s a f e
**  - - - - - - - - - -
**
**  Star proper motion:  update star catalog data for space motion, with
**  special handling to handle the zero parallax case.
**
**  Given:
**     ra1    double      right ascension (radians), before
**     dec1   double      declination (radians), before
**     pmr1   double      RA proper motion (radians/year), before
**     pmd1   double      Dec proper motion (radians/year), before
**     px1    double      parallax (arcseconds), before
**     rv1    double      radial velocity (km/s, +ve = receding), before
**     ep1a   double      "before" epoch, part A (Note 1)
**     ep1b   double      "before" epoch, part B (Note 1)
**     ep2a   double      "after" epoch, part A (Note 1)
**     ep2b   double      "after" epoch, part B (Note 1)
**
**  Returned:
**     ra2    double      right ascension (radians), after
**     dec2   double      declination (radians), after
**     pmr2   double      RA proper motion (radians/year), after
**     pmd2   double      Dec proper motion (radians/year), after
**     px2    double      parallax (arcseconds), after
**     rv2    double      radial velocity (km/s, +ve = receding), after
**
**  Returned (function value):
**            int         status:
**                         -1 = system error (should not occur)
**                          0 = no warnings or errors
**                          1 = distance overridden (Note 6)
**                          2 = excessive velocity (Note 7)
**                          4 = solution didn't converge (Note 8)
**                       else = binary logical OR of the above warnings
**
**  Notes:
**
**  1) The starting and ending TDB epochs ep1a+ep1b and ep2a+ep2b are
**     Julian Dates, apportioned in any convenient way between the two
**     parts (A and B).  For example, JD(TDB)=2450123.7 could be
**     expressed in any of these ways, among others:
**
**            epNa            epNb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.
**
**  2) In accordance with normal star-catalog conventions, the object's
**     right ascension and declination are freed from the effects of
**     secular aberration.  The frame, which is aligned to the catalog
**     equator and equinox, is Lorentzian and centered on the SSB.
**
**     The proper motions are the rate of change of the right ascension
**     and declination at the catalog epoch and are in radians per TDB
```

```
**     Julian year.
**
**     The parallax and radial velocity are in the same frame.
**
**  3) Care is needed with units.  The star coordinates are in radians
**     and the proper motions in radians per Julian year, but the
**     parallax is in arcseconds.
**
**  4) The RA proper motion is in terms of coordinate angle, not true
**     angle.  If the catalog uses arcseconds for both RA and Dec proper
**     motions, the RA proper motion will need to be divided by cos(Dec)
**     before use.
**
**  5) Straight-line motion at constant speed, in the inertial frame, is
**     assumed.
**
**  6) An extremely small (or zero or negative) parallax is overridden
**     to ensure that the object is at a finite but very large distance,
**     but not so large that the proper motion is equivalent to a large
**     but safe speed (about 0.1c using the chosen constant).  A warning
**     status of 1 is added to the status if this action has been taken.
**
**  7) If the space velocity is a significant fraction of c (see the
**     constant VMAX in the function eraStarpv), it is arbitrarily set
**     to zero.  When this action occurs, 2 is added to the status.
**
**  8) The relativistic adjustment carried out in the eraStarpv function
**     involves an iterative calculation.  If the process fails to
**     converge within a set number of iterations, 4 is added to the
**     status.
**
**  Called:
**     eraSeps      angle between two points
**     eraStarpm    update star catalog data for space motion
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvtob(double elong, double phi, double hm,
              double xp, double yp, double sp, double theta,
              double pv[2][3])
/*
**  - - - - - - - - -
**   e r a P v t o b
**  - - - - - - - - -
**
**  Position and velocity of a terrestrial observing station.
**
**  Given:
**     elong   double       longitude (radians, east +ve, Note 1)
**     phi     double       latitude (geodetic, radians, Note 1)
**     hm      double       height above ref. ellipsoid (geodetic, m)
**     xp,yp   double       coordinates of the pole (radians, Note 2)
**     sp      double       the TIO locator s’ (radians, Note 2)
**     theta   double       Earth rotation angle (radians, Note 3)
**
**  Returned:
**     pv      double[2][3] position/velocity vector (m, m/s, CIRS)
**
**  Notes:
**
**  1) The terrestrial coordinates are with respect to the ERFA_WGS84
**     reference ellipsoid.
**
**  2) xp and yp are the coordinates (in radians) of the Celestial
**     Intermediate Pole with respect to the International Terrestrial
**     Reference System (see IERS Conventions), measured along the
**     meridians 0 and 90 deg west respectively.  sp is the TIO locator
**     s’, in radians, which positions the Terrestrial Intermediate
**     Origin on the equator.  For many applications, xp, yp and
**     (especially) sp can be set to zero.
**
**  3) If theta is Greenwich apparent sidereal time instead of Earth
**     rotation angle, the result is with respect to the true equator
**     and equinox of date, i.e. with the x-axis at the equinox rather
**     than the celestial intermediate origin.
**
**  4) The velocity units are meters per UT1 second, not per SI second.
**     This is unlikely to have any practical consequences in the modern
**     era.
**
**  5) No validation is performed on the arguments.  Error cases that
**     could lead to arithmetic exceptions are trapped by the eraGd2gc
**     function, and the result set to zeros.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Urban, S. & Seidelmann, P. K. (eds), Explanatory Supplement to
**     the Astronomical Almanac, 3rd ed., University Science Books
**     (2013), Section 7.4.3.3.
**
**  Called:
**     eraGd2gc     geodetic to geocentric transformation
**     eraPom00     polar motion matrix
**     eraTrxp      product of transpose of r-matrix and p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRefco(double phpa, double tc, double rh, double wl,
              double *refa, double *refb)
/*
**  - - - - - - - - -
**   e r a R e f c o
**  - - - - - - - - -
**
**  Determine the constants A and B in the atmospheric refraction model
**  dZ = A tan Z + B tan^3 Z.
**
**  Z is the "observed" zenith distance (i.e. affected by refraction)
**  and dZ is what to add to Z to give the "topocentric" (i.e. in vacuo)
**  zenith distance.
**
**  Given:
**    phpa   double    pressure at the observer (hPa = millibar)
**    tc     double    ambient temperature at the observer (deg C)
**    rh     double    relative humidity at the observer (range 0-1)
**    wl     double    wavelength (micrometers)
**
**  Returned:
**    refa   double*   tan Z coefficient (radians)
**    refb   double*   tan^3 Z coefficient (radians)
**
**  Notes:
**
**  1) The model balances speed and accuracy to give good results in
**     applications where performance at low altitudes is not paramount.
**     Performance is maintained across a range of conditions, and
**     applies to both optical/IR and radio.
**
**  2) The model omits the effects of (i) height above sea level (apart
**     from the reduced pressure itself), (ii) latitude (i.e. the
**     flattening of the Earth), (iii) variations in tropospheric lapse
**     rate and (iv) dispersive effects in the radio.
**
**     The model was tested using the following range of conditions:
**
**       lapse rates 0.0055, 0.0065, 0.0075 deg/meter
**       latitudes 0, 25, 50, 75 degrees
**       heights 0, 2500, 5000 meters ASL
**       pressures mean for height -10% to +5% in steps of 5%
**       temperatures -10 deg to +20 deg with respect to 280 deg at SL
**       relative humidity 0, 0.5, 1
**       wavelengths 0.4, 0.6, ... 2 micron, + radio
**       zenith distances 15, 45, 75 degrees
**
**     The accuracy with respect to raytracing through a model
**     atmosphere was as follows:
**
**                              worst         RMS
**
**       optical/IR           62 mas       8 mas
**       radio               319 mas      49 mas
**
**     For this particular set of conditions:
**
**       lapse rate 0.0065 K/meter
**       latitude 50 degrees
**       sea level
**       pressure 1005 mb
**       temperature 280.15 K
**       humidity 80%
**       wavelength 5740 Angstroms
**
**     the results were as follows:
**
**       ZD       raytrace     eraRefco   Saastamoinen
**
**       10         10.27        10.27        10.27
```

```
**       20         21.19        21.20        21.19
**       30         33.61        33.61        33.60
**       40         48.82        48.83        48.81
**       45         58.16        58.18        58.16
**       50         69.28        69.30        69.27
**       55         82.97        82.99        82.95
**       60        100.51       100.54       100.50
**       65        124.23       124.26       124.20
**       70        158.63       158.68       158.61
**       72        177.32       177.37       177.31
**       74        200.35       200.38       200.32
**       76        229.45       229.43       229.42
**       78        267.44       267.29       267.41
**       80        319.13       318.55       319.10
**
**      deg        arcsec       arcsec       arcsec
**
**     The values for Saastamoinen's formula (which includes terms
**     up to tan^5) are taken from Hohenkerk and Sinclair (1985).
**
**  3) A wl value in the range 0-100 selects the optical/IR case and is
**     wavelength in micrometers.  Any value outside this range selects
**     the radio case.
**
**  4) Outlandish input parameters are silently limited to
**     mathematically safe values.  Zero pressure is permissible, and
**     causes zeroes to be returned.
**
**  5) The algorithm draws on several sources, as follows:
**
**     a) The formula for the saturation vapour pressure of water as
**        a function of temperature and temperature is taken from
**        Equations (A4.5-A4.7) of Gill (1982).
**
**     b) The formula for the water vapour pressure, given the
**        saturation pressure and the relative humidity, is from
**        Crane (1976), Equation (2.5.5).
**
**     c) The refractivity of air is a function of temperature,
**        total pressure, water-vapour pressure and, in the case
**        of optical/IR, wavelength.  The formulae for the two cases are
**        developed from Hohenkerk & Sinclair (1985) and Rueger (2002).
**
**     d) The formula for beta, the ratio of the scale height of the
**        atmosphere to the geocentric distance of the observer, is
**        an adaption of Equation (9) from Stone (1996).  The
**        adaptations, arrived at empirically, consist of (i) a small
**        adjustment to the coefficient and (ii) a humidity term for the
**        radio case only.
**
**     e) The formulae for the refraction constants as a function of
**        n-1 and beta are from Green (1987), Equation (4.31).
**
**  References:
**
**     Crane, R.K., Meeks, M.L. (ed), "Refraction Effects in the Neutral
**     Atmosphere", Methods of Experimental Physics: Astrophysics 12B,
**     Academic Press, 1976.
**
**     Gill, Adrian E., "Atmosphere-Ocean Dynamics", Academic Press,
**     1982.
**
**     Green, R.M., "Spherical Astronomy", Cambridge University Press,
**     1987.
**
**     Hohenkerk, C.Y., & Sinclair, A.T., NAO Technical Note No. 63,
**     1985.
**
**     Rueger, J.M., "Refractive Index Formulae for Electronic Distance
**     Measurement with Radio and Millimetre Waves", in Unisurv Report
```

```
**     S-68, School of Surveying and Spatial Information Systems,
**     University of New South Wales, Sydney, Australia, 2002.
**
**     Stone, Ronald C., P.A.S.P. 108, 1051-1058, 1996.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraEpv00(double date1, double date2,
             double pvh[2][3], double pvb[2][3])
/*
**  - - - - - - - - -
**   e r a E p v 0 0
**  - - - - - - - - -
**
**  Earth position and velocity, heliocentric and barycentric, with
**  respect to the Barycentric Celestial Reference System.
**
**  Given:
**     date1,date2  double        TDB date (Note 1)
**
**  Returned:
**     pvh          double[2][3]  heliocentric Earth position/velocity
**     pvb          double[2][3]  barycentric Earth position/velocity
**
**  Returned (function value):
**                  int           status: 0 = OK
**                                       +1 = warning: date outside
**                                            the range 1900-2100 AD
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways, among
**     others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  However,
**     the accuracy of the result is more likely to be limited by the
**     algorithm itself than the way the date has been expressed.
**
**     n.b. TT can be used instead of TDB in most applications.
**
**  2) On return, the arrays pvh and pvb contain the following:
**
**        pvh[0][0]  x       }
**        pvh[0][1]  y       } heliocentric position, au
**        pvh[0][2]  z       }
**
**        pvh[1][0]  xdot    }
**        pvh[1][1]  ydot    } heliocentric velocity, au/d
**        pvh[1][2]  zdot    }
**
**        pvb[0][0]  x       }
**        pvb[0][1]  y       } barycentric position, au
**        pvb[0][2]  z       }
**
**        pvb[1][0]  xdot    }
**        pvb[1][1]  ydot    } barycentric velocity, au/d
**        pvb[1][2]  zdot    }
**
**     The vectors are with respect to the Barycentric Celestial
**     Reference System.  The time unit is one day in TDB.
**
**  3) The function is a SIMPLIFIED SOLUTION from the planetary theory
**     VSOP2000 (X. Moisson, P. Bretagnon, 2001, Celes. Mechanics &
```

```
**     Dyn. Astron., 80, 3/4, 205-213) and is an adaptation of original
**     Fortran code supplied by P. Bretagnon (private comm., 2000).
**
**  4) Comparisons over the time span 1900-2100 with this simplified
**     solution and the JPL DE405 ephemeris give the following results:
**
**                                RMS    max
**           Heliocentric:
**              position error    3.7   11.2   km
**              velocity error    1.4    5.0   mm/s
**
**           Barycentric:
**              position error    4.6   13.4   km
**              velocity error    1.4    4.9   mm/s
**
**     Comparisons with the JPL DE406 ephemeris show that by 1800 and
**     2200 the position errors are approximately double their 1900-2100
**     size.  By 1500 and 2500 the deterioration is a factor of 10 and
**     by 1000 and 3000 a factor of 60.  The velocity accuracy falls off
**     at about half that rate.
**
**  5) It is permissible to use the same array for pvh and pvb, which
**     will receive the barycentric values.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraPlan94(double date1, double date2, int np, double pv[2][3])
/*
**  - - - - - - - - - -
**   e r a P l a n 9 4
**  - - - - - - - - - -
**
**  Approximate heliocentric position and velocity of a nominated major
**  planet:  Mercury, Venus, EMB, Mars, Jupiter, Saturn, Uranus or
**  Neptune (but not the Earth itself).
**
**  Given:
**     date1  double       TDB date part A (Note 1)
**     date2  double       TDB date part B (Note 1)
**     np     int          planet (1=Mercury, 2=Venus, 3=EMB, 4=Mars,
**                             5=Jupiter, 6=Saturn, 7=Uranus, 8=Neptune)
**
**  Returned (argument):
**     pv     double[2][3] planet p,v (heliocentric, J2000.0, au,au/d)
**
**  Returned (function value):
**            int          status: -1 = illegal NP (outside 1-8)
**                                  0 = OK
**                                 +1 = warning: year outside 1000-3000
**                                 +2 = warning: failed to converge
**
**  Notes:
**
**  1) The date date1+date2 is in the TDB time scale (in practice TT can
**     be used) and is a Julian Date, apportioned in any convenient way
**     between the two arguments.  For example, JD(TDB)=2450123.7 could
**     be expressed in any of these ways, among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     resolution.  The MJD method and the date & time methods are both
**     good compromises between resolution and convenience.  The limited
**     accuracy of the present algorithm is such that any of the methods
**     is satisfactory.
**
**  2) If an np value outside the range 1-8 is supplied, an error status
**     (function value -1) is returned and the pv vector set to zeroes.
**
**  3) For np=3 the result is for the Earth-Moon Barycenter.  To obtain
**     the heliocentric position and velocity of the Earth, use instead
**     the ERFA function eraEpv00.
**
**  4) On successful return, the array pv contains the following:
**
**        pv[0][0]   x      }
**        pv[0][1]   y      } heliocentric position, au
**        pv[0][2]   z      }
**
**        pv[1][0]   xdot   }
**        pv[1][1]   ydot   } heliocentric velocity, au/d
**        pv[1][2]   zdot   }
**
**     The reference frame is equatorial and is with respect to the
**     mean equator and equinox of epoch J2000.0.
**
**  5) The algorithm is due to J.L. Simon, P. Bretagnon, J. Chapront,
**     M. Chapront-Touze, G. Francou and J. Laskar (Bureau des
```

```
**     Longitudes, Paris, France).  From comparisons with JPL
**     ephemeris DE102, they quote the following maximum errors
**     over the interval 1800-2050:
**
**                     L (arcsec)    B (arcsec)      R (km)
**
**        Mercury          4             1             300
**        Venus            5             1             800
**        EMB              6             1            1000
**        Mars            17             1            7700
**        Jupiter         71             5           76000
**        Saturn          81            13          267000
**        Uranus          86             7          712000
**        Neptune         11             1          253000
**
**     Over the interval 1000-3000, they report that the accuracy is no
**     worse than 1.5 times that over 1800-2050.  Outside 1000-3000 the
**     accuracy declines.
**
**     Comparisons of the present function with the JPL DE200 ephemeris
**     give the following RMS errors over the interval 1960-2025:
**
**                      position (km)     velocity (m/s)
**
**        Mercury            334               0.437
**        Venus             1060               0.855
**        EMB               2010               0.815
**        Mars              7690               1.98
**        Jupiter          71700               7.70
**        Saturn          199000              19.4
**        Uranus          564000              16.4
**        Neptune         158000              14.4
**
**     Comparisons against DE200 over the interval 1800-2100 gave the
**     following maximum absolute differences.  (The results using
**     DE406 were essentially the same.)
**
**                   L (arcsec)   B (arcsec)     R (km)   Rdot (m/s)
**
**        Mercury        7            1            500       0.7
**        Venus          7            1           1100       0.9
**        EMB            9            1           1300       1.0
**        Mars          26            1           9000       2.5
**        Jupiter       78            6          82000       8.2
**        Saturn        87           14         263000      24.6
**        Uranus        86            7         661000      27.4
**        Neptune       11            2         248000      21.4
**
**  6) The present ERFA re-implementation of the original Simon et al.
**     Fortran code differs from the original in the following respects:
**
**       *  C instead of Fortran.
**
**       *  The date is supplied in two parts.
**
**       *  The result is returned only in equatorial Cartesian form;
**          the ecliptic longitude, latitude and radius vector are not
**          returned.
**
**       *  The result is in the J2000.0 equatorial frame, not ecliptic.
**
**       *  More is done in-line: there are fewer calls to subroutines.
**
**       *  Different error/warning status values are used.
**
**       *  A different Kepler's-equation-solver is used (avoiding
**          use of double precision complex).
**
**       *  Polynomials in t are nested to minimize rounding errors.
**
```

```
**        *  Explicit double constants are used to avoid mixed-mode
**           expressions.
**
**     None of the above changes affects the result significantly.
**
**  7) The returned status indicates the most serious condition
**     encountered during execution of the function.  Illegal np is
**     considered the most serious, overriding failure to converge,
**     which in turn takes precedence over the remote date warning.
**
**  Called:
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Reference:  Simon, J.L, Bretagnon, P., Chapront, J.,
**              Chapront-Touze, M., Francou, G., and Laskar, J.,
**              Astron.Astrophys., 282, 663 (1994).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFad03(double t)
/*
**  - - - - - - - - - -
**   e r a F a d 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean elongation of the Moon from the Sun.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    D, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFae03(double t)
/*
**  - - - - - - - - - -
**   e r a F a e 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Earth.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Earth, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFaf03(double t)
/*
**  - - - - - - - - -
**   e r a F a f 0 3
**  - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of the Moon minus mean longitude of the ascending
**  node.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    F, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFaju03(double t)
/*
**  - - - - - - - - - -
**   e r a F a j u 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Jupiter.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Jupiter, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFal03(double t)
/*
**  - - - - - - - - -
**   e r a F a l 0 3
**  - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean anomaly of the Moon.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    l, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFalp03(double t)
/*
**  - - - - - - - - - - -
**   e r a F a l p 0 3
**  - - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean anomaly of the Sun.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    l’, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFama03(double t)
/*
**  - - - - - - - - - -
**   e r a F a m a 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Mars.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Mars, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFame03(double t)
/*
**  - - - - - - - - - -
**   e r a F a m e 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Mercury.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Mercury, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFane03(double t)
/*
**  - - - - - - - - - -
**   e r a F a n e 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Neptune.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Neptune, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is adapted from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFaom03(double t)
/*
**  - - - - - - - - - -
**   e r a F a o m 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of the Moon's ascending node.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    Omega, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFapa03(double t)
/*
**  - - - - - - - - - -
**   e r a F a p a 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  general accumulated precession in longitude.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    general precession in longitude, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003).  It
**     is taken from Kinoshita & Souchay (1990) and comes originally
**     from Lieske et al. (1977).
**
**  References:
**
**     Kinoshita, H. and Souchay J. 1990, Celest.Mech. and Dyn.Astron.
**     48, 187
**
**     Lieske, J.H., Lederle, T., Fricke, W. & Morando, B. 1977,
**     Astron.Astrophys. 58, 1-16
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFasa03(double t)
/*
**  - - - - - - - - - -
**   e r a F a s a 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Saturn.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Saturn, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFaur03(double t)
/*
**  - - - - - - - - - -
**   e r a F a u r 0 3
**  - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Uranus.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned  (function value):
**           double    mean longitude of Uranus, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     is adapted from Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraFave03(double t)
/*
**  - - - - - - - - - - -
**   e r a F a v e 0 3
**  - - - - - - - - - - -
**
**  Fundamental argument, IERS Conventions (2003):
**  mean longitude of Venus.
**
**  Given:
**     t     double    TDB, Julian centuries since J2000.0 (Note 1)
**
**  Returned (function value):
**           double    mean longitude of Venus, radians (Note 2)
**
**  Notes:
**
**  1) Though t is strictly TDB, it is usually more convenient to use
**     TT, which makes no significant difference.
**
**  2) The expression used is as adopted in IERS Conventions (2003) and
**     comes from Souchay et al. (1999) after Simon et al. (1994).
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraBi00(double *dpsibi, double *depsbi, double *dra)
/*
**  - - - - - - - -
**   e r a B i 0 0
**  - - - - - - - -
**
**  Frame bias components of IAU 2000 precession-nutation models (part
**  of MHB2000 with additions).
**
**  Returned:
**     dpsibi,depsbi  double  longitude and obliquity corrections
**     dra            double  the ICRS RA of the J2000.0 mean equinox
**
**  Notes:
**
**  1) The frame bias corrections in longitude and obliquity (radians)
**     are required in order to correct for the offset between the GCRS
**     pole and the mean J2000.0 pole.  They define, with respect to the
**     GCRS frame, a J2000.0 mean pole that is consistent with the rest
**     of the IAU 2000A precession-nutation model.
**
**  2) In addition to the displacement of the pole, the complete
**     description of the frame bias requires also an offset in right
**     ascension.  This is not part of the IAU 2000A model, and is from
**     Chapront et al. (2002).  It is returned in radians.
**
**  3) This is a supplemented implementation of one aspect of the IAU
**     2000A nutation model, formally adopted by the IAU General
**     Assembly in 2000, namely MHB2000 (Mathews et al. 2002).
**
**  References:
**
**     Chapront, J., Chapront-Touze, M. & Francou, G., Astron.
**     Astrophys., 387, 700, 2002.
**
**     Mathews, P.M., Herring, T.A., Buffet, B.A., "Modeling of nutation
**     and precession   New nutation series for nonrigid Earth and
**     insights into the Earth’s interior", J.Geophys.Res., 107, B4,
**     2002.  The MHB2000 code itself was obtained on 9th September 2002
**     from ftp://maia.usno.navy.mil/conv2000/chapter5/IAU2000A.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraBp00(double date1, double date2,
             double rb[3][3], double rp[3][3], double rbp[3][3])
/*
**  - - - - - - - - -
**   e r a B p 0 0
**  - - - - - - - - -
**
**  Frame bias and precession, IAU 2000.
**
**  Given:
**     date1,date2  double         TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rb           double[3][3]   frame bias matrix (Note 2)
**     rp           double[3][3]   precession matrix (Note 3)
**     rbp          double[3][3]   bias-precession matrix (Note 4)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**             date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rb transforms vectors from GCRS to mean J2000.0 by
**     applying frame bias.
**
**  3) The matrix rp transforms vectors from J2000.0 mean equator and
**     equinox to mean equator and equinox of date by applying
**     precession.
**
**  4) The matrix rbp transforms vectors from GCRS to mean equator and
**     equinox of date by applying frame bias then precession.  It is
**     the product rp x rb.
**
**  5) It is permissible to re-use the same array in the returned
**     arguments.  The arrays are filled in the order given.
**
**  Called:
**     eraBi00      frame bias components, IAU 2000
**     eraPr00      IAU 2000 precession adjustments
**     eraIr        initialize r-matrix to identity
**     eraRx        rotate around X-axis
**     eraRy        rotate around Y-axis
**     eraRz        rotate around Z-axis
**     eraCr        copy r-matrix
**     eraRxr       product of two r-matrices
**
**  Reference:
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
```

```
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraBp06(double date1, double date2,
             double rb[3][3], double rp[3][3], double rbp[3][3])
/*
**  - - - - - - - -
**   e r a B p 0 6
**  - - - - - - - -
**
**  Frame bias and precession, IAU 2006.
**
**  Given:
**     date1,date2  double         TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rb           double[3][3]   frame bias matrix (Note 2)
**     rp           double[3][3]   precession matrix (Note 3)
**     rbp          double[3][3]   bias-precession matrix (Note 4)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**             date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rb transforms vectors from GCRS to mean J2000.0 by
**     applying frame bias.
**
**  3) The matrix rp transforms vectors from mean J2000.0 to mean of
**     date by applying precession.
**
**  4) The matrix rbp transforms vectors from GCRS to mean of date by
**     applying frame bias then precession.  It is the product rp x rb.
**
**  5) It is permissible to re-use the same array in the returned
**     arguments.  The arrays are filled in the order given.
**
**  Called:
**     eraPfw06     bias-precession F-W angles, IAU 2006
**     eraFw2m      F-W angles to r-matrix
**     eraPmat06    PB matrix, IAU 2006
**     eraTr        transpose r-matrix
**     eraRxr       product of two r-matrices
**     eraCr        copy r-matrix
**
**  References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraBpn2xy(double rbpn[3][3], double *x, double *y)
/*
**  - - - - - - - - - -
**   e r a B p n 2 x y
**  - - - - - - - - - -
**
**  Extract from the bias-precession-nutation matrix the X,Y coordinates
**  of the Celestial Intermediate Pole.
**
**  Given:
**     rbpn      double[3][3]  celestial-to-true matrix (Note 1)
**
**  Returned:
**     x,y       double        Celestial Intermediate Pole (Note 2)
**
**  Notes:
**
**  1) The matrix rbpn transforms vectors from GCRS to true equator (and
**     CIO or equinox) of date, and therefore the Celestial Intermediate
**     Pole unit vector is the bottom row of the matrix.
**
**  2) The arguments x,y are components of the Celestial Intermediate
**     Pole unit vector in the Geocentric Celestial Reference System.
**
**  Reference:
**
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154
**     (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2i00a(double date1, double date2, double rc2i[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 i 0 0 a
**  - - - - - - - - - -
**
**  Form the celestial-to-intermediate matrix for a given date using the
**  IAU 2000A precession-nutation model.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rc2i        double[3][3] celestial-to-intermediate matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS]  =  RPOM * R_3(ERA) * rc2i * [CRS]
**
**               =  rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  3) A faster, but slightly less accurate result (about 1 mas), can be
**     obtained by using instead the eraC2i00b function.
**
**  Called:
**     eraPnm00a    classical NPB matrix, IAU 2000A
**     eraC2ibpn    celestial-to-intermediate matrix, given NPB matrix
**
**  References:
**
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154
**     (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
```

```
*/
```

```
void eraC2i00b(double date1, double date2, double rc2i[3][3])
/*
**  - - - - - - - - - - -
**   e r a C 2 i 0 0 b
**  - - - - - - - - - - -
**
**  Form the celestial-to-intermediate matrix for a given date using the
**  IAU 2000B precession-nutation model.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rc2i        double[3][3] celestial-to-intermediate matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS]  =  RPOM * R_3(ERA) * rc2i * [CRS]
**
**               =  rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  3) The present function is faster, but slightly less accurate (about
**     1 mas), than the eraC2i00a function.
**
**  Called:
**     eraPnm00b    classical NPB matrix, IAU 2000B
**     eraC2ibpn    celestial-to-intermediate matrix, given NPB matrix
**
**  References:
**
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154
**     (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
```

```
*/
```

```
void eraC2i06a(double date1, double date2, double rc2i[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 i 0 6 a
**  - - - - - - - - - -
**
**  Form the celestial-to-intermediate matrix for a given date using the
**  IAU 2006 precession and IAU 2000A nutation models.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rc2i        double[3][3] celestial-to-intermediate matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS]  =  RPOM * R_3(ERA) * rc2i * [CRS]
**
**               =  RC2T * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  Called:
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**     eraC2ixys    celestial-to-intermediate matrix, given X,Y and s
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2ibpn(double date1, double date2, double rbpn[3][3],
               double rc2i[3][3])
/*
**  - - - - - - - - - - -
**   e r a C 2 i b p n
**  - - - - - - - - - - -
**
**  Form the celestial-to-intermediate matrix for a given date given
**  the bias-precession-nutation matrix.  IAU 2000.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**     rbpn        double[3][3] celestial-to-true matrix (Note 2)
**
**  Returned:
**     rc2i        double[3][3] celestial-to-intermediate matrix (Note 3)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix rbpn transforms vectors from GCRS to true equator (and
**     CIO or equinox) of date.  Only the CIP (bottom row) is used.
**
**  3) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * rc2i * [CRS]
**
**              = RC2T * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  4) Although its name does not include "00", This function is in fact
**     specific to the IAU 2000 models.
**
**  Called:
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraC2ixy     celestial-to-intermediate matrix, given X,Y
**
**  References:
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
```

```
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2ixy(double date1, double date2, double x, double y,
              double rc2i[3][3])
/*
**  - - - - - - - - -
**   e r a C 2 i x y
**  - - - - - - - - -
**
**  Form the celestial to intermediate-frame-of-date matrix for a given
**  date when the CIP X,Y coordinates are known.  IAU 2000.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**     x,y         double       Celestial Intermediate Pole (Note 2)
**
**  Returned:
**     rc2i        double[3][3] celestial-to-intermediate matrix (Note 3)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The Celestial Intermediate Pole coordinates are the x,y components
**     of the unit vector in the Geocentric Celestial Reference System.
**
**  3) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * rc2i * [CRS]
**
**              = RC2T * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  4) Although its name does not include "00", This function is in fact
**     specific to the IAU 2000 models.
**
**  Called:
**     eraC2ixys    celestial-to-intermediate matrix, given X,Y and s
**     eraS00       the CIO locator s, given X,Y, IAU 2000A
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2ixys(double x, double y, double s, double rc2i[3][3])
/*
**  - - - - - - - - - - -
**   e r a C 2 i x y s
**  - - - - - - - - - - -
**
**  Form the celestial to intermediate-frame-of-date matrix given the CIP
**  X,Y and the CIO locator s.
**
**  Given:
**     x,y      double         Celestial Intermediate Pole (Note 1)
**     s        double         the CIO locator s (Note 2)
**
**  Returned:
**     rc2i     double[3][3]   celestial-to-intermediate matrix (Note 3)
**
**  Notes:
**
**  1) The Celestial Intermediate Pole coordinates are the x,y
**     components of the unit vector in the Geocentric Celestial
**     Reference System.
**
**  2) The CIO locator s (in radians) positions the Celestial
**     Intermediate Origin on the equator of the CIP.
**
**  3) The matrix rc2i is the first stage in the transformation from
**     celestial to terrestrial coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * rc2i * [CRS]
**
**              = RC2T * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  Called:
**     eraIr        initialize r-matrix to identity
**     eraRz        rotate around Z-axis
**     eraRy        rotate around Y-axis
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2t00a(double tta, double ttb, double uta, double utb,
               double xp, double yp, double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t 0 0 a
**  - - - - - - - - - -
**
**  Form the celestial to terrestrial matrix given the date, the UT1 and
**  the polar motion, using the IAU 2000A nutation model.
**
**  Given:
**     tta,ttb  double         TT as a 2-part Julian Date (Note 1)
**     uta,utb  double         UT1 as a 2-part Julian Date (Note 1)
**     xp,yp    double         coordinates of the pole (radians, Note 2)
**
**  Returned:
**     rc2t     double[3][3]   celestial-to-terrestrial matrix (Note 3)
**
**  Notes:
**
**  1) The TT and UT1 dates tta+ttb and uta+utb are Julian Dates,
**     apportioned in any convenient way between the arguments uta and
**     utb.  For example, JD(UT1)=2450123.7 could be expressed in any of
**     these ways, among others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  In the case of uta,utb, the
**     date & time method is best matched to the Earth rotation angle
**     algorithm used:  maximum precision is delivered when the uta
**     argument is for 0hrs UT1 on the day in question and the utb
**     argument lies in the range 0 to 1, or vice versa.
**
**  2) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  3) The matrix rc2t transforms from celestial to terrestrial
**     coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * RC2I * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), RC2I is the
**     celestial-to-intermediate matrix, ERA is the Earth rotation
**     angle and RPOM is the polar motion matrix.
**
**  4) A faster, but slightly less accurate result (about 1 mas), can
**     be obtained by using instead the eraC2t00b function.
**
**  Called:
**     eraC2i00a    celestial-to-intermediate matrix, IAU 2000A
**     eraEra00     Earth rotation angle, IAU 2000
**     eraSp00      the TIO locator s', IERS 2000
**     eraPom00     polar motion matrix
**     eraC2tcio    form CIO-based celestial-to-terrestrial matrix
**
**  Reference:
```

```
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2t00b(double tta, double ttb, double uta, double utb,
               double xp, double yp, double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t 0 0 b
**  - - - - - - - - - -
**
**  Form the celestial to terrestrial matrix given the date, the UT1 and
**  the polar motion, using the IAU 2000B nutation model.
**
**  Given:
**     tta,ttb  double         TT as a 2-part Julian Date (Note 1)
**     uta,utb  double         UT1 as a 2-part Julian Date (Note 1)
**     xp,yp    double         coordinates of the pole (radians, Note 2)
**
**  Returned:
**     rc2t     double[3][3]   celestial-to-terrestrial matrix (Note 3)
**
**  Notes:
**
**  1) The TT and UT1 dates tta+ttb and uta+utb are Julian Dates,
**     apportioned in any convenient way between the arguments uta and
**     utb.  For example, JD(UT1)=2450123.7 could be expressed in any of
**     these ways, among others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  In the case of uta,utb, the
**     date & time method is best matched to the Earth rotation angle
**     algorithm used:  maximum precision is delivered when the uta
**     argument is for 0hrs UT1 on the day in question and the utb
**     argument lies in the range 0 to 1, or vice versa.
**
**  2) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  3) The matrix rc2t transforms from celestial to terrestrial
**     coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * RC2I * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), RC2I is the
**     celestial-to-intermediate matrix, ERA is the Earth rotation
**     angle and RPOM is the polar motion matrix.
**
**  4) The present function is faster, but slightly less accurate (about
**     1 mas), than the eraC2t00a function.
**
**  Called:
**     eraC2i00b    celestial-to-intermediate matrix, IAU 2000B
**     eraEra00     Earth rotation angle, IAU 2000
**     eraPom00     polar motion matrix
**     eraC2tcio    form CIO-based celestial-to-terrestrial matrix
**
**  Reference:
**
```

```
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2t06a(double tta, double ttb, double uta, double utb,
               double xp, double yp, double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t 0 6 a
**  - - - - - - - - - -
**
**  Form the celestial to terrestrial matrix given the date, the UT1 and
**  the polar motion, using the IAU 2006 precession and IAU 2000A
**  nutation models.
**
**  Given:
**     tta,ttb  double         TT as a 2-part Julian Date (Note 1)
**     uta,utb  double         UT1 as a 2-part Julian Date (Note 1)
**     xp,yp    double         coordinates of the pole (radians, Note 2)
**
**  Returned:
**     rc2t     double[3][3]   celestial-to-terrestrial matrix (Note 3)
**
**  Notes:
**
**  1) The TT and UT1 dates tta+ttb and uta+utb are Julian Dates,
**     apportioned in any convenient way between the arguments uta and
**     utb.  For example, JD(UT1)=2450123.7 could be expressed in any of
**     these ways, among others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  In the case of uta,utb, the
**     date & time method is best matched to the Earth rotation angle
**     algorithm used:  maximum precision is delivered when the uta
**     argument is for 0hrs UT1 on the day in question and the utb
**     argument lies in the range 0 to 1, or vice versa.
**
**  2) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  3) The matrix rc2t transforms from celestial to terrestrial
**     coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * RC2I * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), RC2I is the
**     celestial-to-intermediate matrix, ERA is the Earth rotation
**     angle and RPOM is the polar motion matrix.
**
**  Called:
**     eraC2i06a    celestial-to-intermediate matrix, IAU 2006/2000A
**     eraEra00     Earth rotation angle, IAU 2000
**     eraSp00      the TIO locator s', IERS 2000
**     eraPom00     polar motion matrix
**     eraC2tcio    form CIO-based celestial-to-terrestrial matrix
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
```

```
**     IERS Technical Note No. 32, BKG
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2tcio(double rc2i[3][3], double era, double rpom[3][3],
               double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t c i o
**  - - - - - - - - - -
**
**  Assemble the celestial to terrestrial matrix from CIO-based
**  components (the celestial-to-intermediate matrix, the Earth Rotation
**  Angle and the polar motion matrix).
**
**  Given:
**     rc2i     double[3][3]    celestial-to-intermediate matrix
**     era      double          Earth rotation angle (radians)
**     rpom     double[3][3]    polar-motion matrix
**
**  Returned:
**     rc2t     double[3][3]    celestial-to-terrestrial matrix
**
**  Notes:
**
**  1) This function constructs the rotation matrix that transforms
**     vectors in the celestial system into vectors in the terrestrial
**     system.  It does so starting from precomputed components, namely
**     the matrix which rotates from celestial coordinates to the
**     intermediate frame, the Earth rotation angle and the polar motion
**     matrix.  One use of the present function is when generating a
**     series of celestial-to-terrestrial matrices where only the Earth
**     Rotation Angle changes, avoiding the considerable overhead of
**     recomputing the precession-nutation more often than necessary to
**     achieve given accuracy objectives.
**
**  2) The relationship between the arguments is as follows:
**
**        [TRS] = RPOM * R_3(ERA) * rc2i * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003).
**
**  Called:
**     eraCr        copy r-matrix
**     eraRz        rotate around Z-axis
**     eraRxr       product of two r-matrices
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2teqx(double rbpn[3][3], double gst, double rpom[3][3],
               double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t e q x
**  - - - - - - - - - -
**
**  Assemble the celestial to terrestrial matrix from equinox-based
**  components (the celestial-to-true matrix, the Greenwich Apparent
**  Sidereal Time and the polar motion matrix).
**
**  Given:
**     rbpn   double[3][3]  celestial-to-true matrix
**     gst    double        Greenwich (apparent) Sidereal Time (radians)
**     rpom   double[3][3]  polar-motion matrix
**
**  Returned:
**     rc2t   double[3][3]  celestial-to-terrestrial matrix (Note 2)
**
**  Notes:
**
**  1) This function constructs the rotation matrix that transforms
**     vectors in the celestial system into vectors in the terrestrial
**     system.  It does so starting from precomputed components, namely
**     the matrix which rotates from celestial coordinates to the
**     true equator and equinox of date, the Greenwich Apparent Sidereal
**     Time and the polar motion matrix.  One use of the present function
**     is when generating a series of celestial-to-terrestrial matrices
**     where only the Sidereal Time changes, avoiding the considerable
**     overhead of recomputing the precession-nutation more often than
**     necessary to achieve given accuracy objectives.
**
**  2) The relationship between the arguments is as follows:
**
**        [TRS] = rpom * R_3(gst) * rbpn * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003).
**
**  Called:
**     eraCr        copy r-matrix
**     eraRz        rotate around Z-axis
**     eraRxr       product of two r-matrices
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2tpe(double tta, double ttb, double uta, double utb,
              double dpsi, double deps, double xp, double yp,
              double rc2t[3][3])
/*
**  - - - - - - - - -
**   e r a C 2 t p e
**  - - - - - - - - -
**
**  Form the celestial to terrestrial matrix given the date, the UT1,
**  the nutation and the polar motion.  IAU 2000.
**
**  Given:
**     tta,ttb    double        TT as a 2-part Julian Date (Note 1)
**     uta,utb    double        UT1 as a 2-part Julian Date (Note 1)
**     dpsi,deps  double        nutation (Note 2)
**     xp,yp      double        coordinates of the pole (radians, Note 3)
**
**  Returned:
**     rc2t       double[3][3]  celestial-to-terrestrial matrix (Note 4)
**
**  Notes:
**
**  1) The TT and UT1 dates tta+ttb and uta+utb are Julian Dates,
**     apportioned in any convenient way between the arguments uta and
**     utb.  For example, JD(UT1)=2450123.7 could be expressed in any of
**     these ways, among others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  In the case of uta,utb, the
**     date & time method is best matched to the Earth rotation angle
**     algorithm used:  maximum precision is delivered when the uta
**     argument is for 0hrs UT1 on the day in question and the utb
**     argument lies in the range 0 to 1, or vice versa.
**
**  2) The caller is responsible for providing the nutation components;
**     they are in longitude and obliquity, in radians and are with
**     respect to the equinox and ecliptic of date.  For high-accuracy
**     applications, free core nutation should be included as well as
**     any other relevant corrections to the position of the CIP.
**
**  3) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  4) The matrix rc2t transforms from celestial to terrestrial
**     coordinates:
**
**        [TRS] = RPOM * R_3(GST) * RBPN * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), RBPN is the
**     bias-precession-nutation matrix, GST is the Greenwich (apparent)
**     Sidereal Time and RPOM is the polar motion matrix.
**
**  5) Although its name does not include "00", This function is in fact
**     specific to the IAU 2000 models.
**
```

```
**  Called:
**     eraPn00      bias/precession/nutation results, IAU 2000
**     eraGmst00    Greenwich mean sidereal time, IAU 2000
**     eraSp00      the TIO locator s', IERS 2000
**     eraEe00      equation of the equinoxes, IAU 2000
**     eraPom00     polar motion matrix
**     eraC2teqx    form equinox-based celestial-to-terrestrial matrix
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2txy(double tta, double ttb, double uta, double utb,
              double x, double y, double xp, double yp,
              double rc2t[3][3])
/*
**  - - - - - - - - - -
**   e r a C 2 t x y
**  - - - - - - - - - -
**
**  Form the celestial to terrestrial matrix given the date, the UT1,
**  the CIP coordinates and the polar motion.  IAU 2000.
**
**  Given:
**     tta,ttb  double         TT as a 2-part Julian Date (Note 1)
**     uta,utb  double         UT1 as a 2-part Julian Date (Note 1)
**     x,y      double         Celestial Intermediate Pole (Note 2)
**     xp,yp    double         coordinates of the pole (radians, Note 3)
**
**  Returned:
**     rc2t     double[3][3]   celestial-to-terrestrial matrix (Note 4)
**
**  Notes:
**
**  1) The TT and UT1 dates tta+ttb and uta+utb are Julian Dates,
**     apportioned in any convenient way between the arguments uta and
**     utb.  For example, JD(UT1)=2450123.7 could be expressed in any o
**     these ways, among others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  In the case of uta,utb, the
**     date & time method is best matched to the Earth rotation angle
**     algorithm used:  maximum precision is delivered when the uta
**     argument is for 0hrs UT1 on the day in question and the utb
**     argument lies in the range 0 to 1, or vice versa.
**
**  2) The Celestial Intermediate Pole coordinates are the x,y
**     components of the unit vector in the Geocentric Celestial
**     Reference System.
**
**  3) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  4) The matrix rc2t transforms from celestial to terrestrial
**     coordinates:
**
**        [TRS] = RPOM * R_3(ERA) * RC2I * [CRS]
**
**              = rc2t * [CRS]
**
**     where [CRS] is a vector in the Geocentric Celestial Reference
**     System and [TRS] is a vector in the International Terrestrial
**     Reference System (see IERS Conventions 2003), ERA is the Earth
**     Rotation Angle and RPOM is the polar motion matrix.
**
**  5) Although its name does not include "00", This function is in fact
**     specific to the IAU 2000 models.
**
**  Called:
**     eraC2ixy     celestial-to-intermediate matrix, given X,Y
**     eraEra00     Earth rotation angle, IAU 2000
```

```
**     eraSp00      the TIO locator s', IERS 2000
**     eraPom00     polar motion matrix
**     eraC2tcio    form CIO-based celestial-to-terrestrial matrix
**
** Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEo06a(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a E o 0 6 a
**  - - - - - - - - -
**
**  Equation of the origins, IAU 2006 precession and IAU 2000A nutation.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    equation of the origins in radians
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The equation of the origins is the distance between the true
**     equinox and the celestial intermediate origin and, equivalently,
**     the difference between Earth rotation angle and Greenwich
**     apparent sidereal time (ERA-GST).  It comprises the precession
**     (since J2000.0) in right ascension plus the equation of the
**     equinoxes (including the small correction terms).
**
**  Called:
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**     eraEors      equation of the origins, given NPB matrix and s
**
**  References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEors(double rnpb[3][3], double s)
/*
**  - - - - - - - -
**   e r a E o r s
**  - - - - - - - -
**
**  Equation of the origins, given the classical NPB matrix and the
**  quantity s.
**
**  Given:
**     rnpb  double[3][3]  classical nutation x precession x bias matrix
**     s     double        the quantity s (the CIO locator)
**
**  Returned (function value):
**           double        the equation of the origins in radians.
**
**  Notes:
**
**  1)  The equation of the origins is the distance between the true
**      equinox and the celestial intermediate origin and, equivalently,
**      the difference between Earth rotation angle and Greenwich
**      apparent sidereal time (ERA-GST).  It comprises the precession
**      (since J2000.0) in right ascension plus the equation of the
**      equinoxes (including the small correction terms).
**
**  2)  The algorithm is from Wallace & Capitaine (2006).
**
** References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraFw2m(double gamb, double phib, double psi, double eps,
             double r[3][3])
/*
**  - - - - - - - -
**   e r a F w 2 m
**  - - - - - - - -
**
**  Form rotation matrix given the Fukushima-Williams angles.
**
**  Given:
**     gamb     double         F-W angle gamma_bar (radians)
**     phib     double         F-W angle phi_bar (radians)
**     psi      double         F-W angle psi (radians)
**     eps      double         F-W angle epsilon (radians)
**
**  Returned:
**     r        double[3][3]   rotation matrix
**
**  Notes:
**
**  1) Naming the following points:
**
**           e = J2000.0 ecliptic pole,
**           p = GCRS pole,
**           E = ecliptic pole of date,
**     and   P = CIP,
**
**     the four Fukushima-Williams angles are as follows:
**
**        gamb = gamma = epE
**        phib = phi = pE
**        psi = psi = pEP
**        eps = epsilon = EP
**
**  2) The matrix representing the combined effects of frame bias,
**     precession and nutation is:
**
**        NxPxB = R_1(-eps).R_3(-psi).R_1(phib).R_3(gamb)
**
**  3) Three different matrices can be constructed, depending on the
**     supplied angles:
**
**     o  To obtain the nutation x precession x frame bias matrix,
**        generate the four precession angles, generate the nutation
**        components and add them to the psi_bar and epsilon_A angles,
**        and call the present function.
**
**     o  To obtain the precession x frame bias matrix, generate the
**        four precession angles and call the present function.
**
**     o  To obtain the frame bias matrix, generate the four precession
**        angles for date J2000.0 and call the present function.
**
**     The nutation-only and precession-only matrices can if necessary
**     be obtained by combining these three appropriately.
**
**  Called:
**     eraIr        initialize r-matrix to identity
**     eraRz        rotate around Z-axis
**     eraRx        rotate around X-axis
**
**  Reference:
**
**     Hilton, J. et al., 2006, Celest.Mech.Dyn.Astron. 94, 351
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraFw2xy(double gamb, double phib, double psi, double eps,
              double *x, double *y)
/*
**  - - - - - - - - -
**   e r a F w 2 x y
**  - - - - - - - - -
**
**  CIP X,Y given Fukushima-Williams bias-precession-nutation angles.
**
**  Given:
**     gamb     double    F-W angle gamma_bar (radians)
**     phib     double    F-W angle phi_bar (radians)
**     psi      double    F-W angle psi (radians)
**     eps      double    F-W angle epsilon (radians)
**
**  Returned:
**     x,y      double    CIP unit vector X,Y
**
**  Notes:
**
**  1) Naming the following points:
**
**           e = J2000.0 ecliptic pole,
**           p = GCRS pole
**           E = ecliptic pole of date,
**     and   P = CIP,
**
**     the four Fukushima-Williams angles are as follows:
**
**        gamb = gamma = epE
**        phib = phi = pE
**        psi = psi = pEP
**        eps = epsilon = EP
**
**  2) The matrix representing the combined effects of frame bias,
**     precession and nutation is:
**
**        NxPxB = R_1(-epsA).R_3(-psi).R_1(phib).R_3(gamb)
**
**     The returned values x,y are elements [2][0] and [2][1] of the
**     matrix.  Near J2000.0, they are essentially angles in radians.
**
**  Called:
**     eraFw2m      F-W angles to r-matrix
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**
**  Reference:
**
**     Hilton, J. et al., 2006, Celest.Mech.Dyn.Astron. 94, 351
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLtp(double epj, double rp[3][3])
/*
**  - - - - - - -
**   e r a L t p
**  - - - - - - -
**
**  Long-term precession matrix.
**
**  Given:
**     epj     double         Julian epoch (TT)
**
**  Returned:
**     rp      double[3][3]   precession matrix, J2000.0 to date
**
**  Notes:
**
**  1) The matrix is in the sense
**
**        P_date = rp x P_J2000,
**
**     where P_J2000 is a vector with respect to the J2000.0 mean
**     equator and equinox and P_date is the same vector with respect to
**     the equator and equinox of epoch epj.
**
**  2) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  Called:
**     eraLtpequ    equator pole, long term
**     eraLtpecl    ecliptic pole, long term
**     eraPxp       vector product
**     eraPn        normalize vector
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLtpb(double epj, double rpb[3][3])
/*
**  - - - - - - - -
**   e r a L t p b
**  - - - - - - - -
**
**  Long-term precession matrix, including ICRS frame bias.
**
**  Given:
**     epj     double         Julian epoch (TT)
**
**  Returned:
**     rpb     double[3][3]   precession-bias matrix, J2000.0 to date
**
**  Notes:
**
**  1) The matrix is in the sense
**
**        P_date = rpb x P_ICRS,
**
**     where P_ICRS is a vector in the Geocentric Celestial Reference
**     System, and P_date is the vector with respect to the Celestial
**     Intermediate Reference System at that date but with nutation
**     neglected.
**
**  2) A first order frame bias formulation is used, of sub-
**     microarcsecond accuracy compared with a full 3D rotation.
**
**  3) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLtpecl(double epj, double vec[3])
/*
**  - - - - - - - - - -
**   e r a L t p e c l
**  - - - - - - - - - -
**
**  Long-term precession of the ecliptic.
**
**  Given:
**     epj     double         Julian epoch (TT)
**
**  Returned:
**     vec     double[3]      ecliptic pole unit vector
**
**  Notes:
**
**  1) The returned vector is with respect to the J2000.0 mean equator
**     and equinox.
**
**  2) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLtpequ(double epj, double veq[3])
/*
**  - - - - - - - - - -
**   e r a L t p e q u
**  - - - - - - - - - -
**
**  Long-term precession of the equator.
**
**  Given:
**     epj     double         Julian epoch (TT)
**
**  Returned:
**     veq     double[3]      equator pole unit vector
**
**  Notes:
**
**  1) The returned vector is with respect to the J2000.0 mean equator
**     and equinox.
**
**  2) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNum00a(double date1, double date2, double rmatn[3][3])
/*
**  - - - - - - - - - -
**   e r a N u m 0 0 a
**  - - - - - - - - - -
**
**  Form the matrix of nutation for a given date, IAU 2000A model.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rmatn        double[3][3]    nutation matrix
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(true) = rmatn * V(mean), where
**     the p-vector V(true) is with respect to the true equatorial triad
**     of date and the p-vector V(mean) is with respect to the mean
**     equatorial triad of date.
**
**  3) A faster, but slightly less accurate result (about 1 mas), can be
**     obtained by using instead the eraNum00b function.
**
**  Called:
**     eraPn00a     bias/precession/nutation, IAU 2000A
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.222-3 (p114).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNum00b(double date1, double date2, double rmatn[3][3])
/*
**  - - - - - - - - - -
**   e r a N u m 0 0 b
**  - - - - - - - - - -
**
**  Form the matrix of nutation for a given date, IAU 2000B model.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rmatn        double[3][3]    nutation matrix
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(true) = rmatn * V(mean), where
**     the p-vector V(true) is with respect to the true equatorial triad
**     of date and the p-vector V(mean) is with respect to the mean
**     equatorial triad of date.
**
**  3) The present function is faster, but slightly less accurate (about
**     1 mas), than the eraNum00a function.
**
**  Called:
**     eraPn00b     bias/precession/nutation, IAU 2000B
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.222-3 (p114).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNum06a(double date1, double date2, double rmatn[3][3])
/*
**  - - - - - - - - - -
**   e r a N u m 0 6 a
**  - - - - - - - - - -
**
**  Form the matrix of nutation for a given date, IAU 2006/2000A model.
**
**  Given:
**     date1,date2   double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rmatn         double[3][3]    nutation matrix
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(true) = rmatn * V(mean), where
**     the p-vector V(true) is with respect to the true equatorial triad
**     of date and the p-vector V(mean) is with respect to the mean
**     equatorial triad of date.
**
**  Called:
**     eraObl06     mean obliquity, IAU 2006
**     eraNut06a    nutation, IAU 2006/2000A
**     eraNumat     form nutation matrix
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.222-3 (p114).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNumat(double epsa, double dpsi, double deps, double rmatn[3][3])
/*
**  - - - - - - - - -
**   e r a N u m a t
**  - - - - - - - - -
**
**  Form the matrix of nutation.
**
**  Given:
**     epsa        double         mean obliquity of date (Note 1)
**     dpsi,deps   double         nutation (Note 2)
**
**  Returned:
**     rmatn       double[3][3]   nutation matrix (Note 3)
**
**  Notes:
**
**
**  1) The supplied mean obliquity epsa, must be consistent with the
**     precession-nutation models from which dpsi and deps were obtained.
**
**  2) The caller is responsible for providing the nutation components;
**     they are in longitude and obliquity, in radians and are with
**     respect to the equinox and ecliptic of date.
**
**  3) The matrix operates in the sense V(true) = rmatn * V(mean),
**     where the p-vector V(true) is with respect to the true
**     equatorial triad of date and the p-vector V(mean) is with
**     respect to the mean equatorial triad of date.
**
**  Called:
**     eraIr        initialize r-matrix to identity
**     eraRx        rotate around X-axis
**     eraRz        rotate around Z-axis
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.222-3 (p114).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNut00a(double date1, double date2, double *dpsi, double *deps)
/*
**  - - - - - - - - - -
**   e r a N u t 0 0 a
**  - - - - - - - - - -
**
**  Nutation, IAU 2000A model (MHB2000 luni-solar and planetary nutation
**  with free core nutation omitted).
**
**  Given:
**     date1,date2   double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps     double   nutation, luni-solar + planetary (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The nutation components in longitude and obliquity are in radians
**     and with respect to the equinox and ecliptic of date.  The
**     obliquity at J2000.0 is assumed to be the Lieske et al. (1977)
**     value of 84381.448 arcsec.
**
**     Both the luni-solar and planetary nutations are included.  The
**     latter are due to direct planetary nutations and the
**     perturbations of the lunar and terrestrial orbits.
**
**  3) The function computes the MHB2000 nutation series with the
**     associated corrections for planetary nutations.  It is an
**     implementation of the nutation part of the IAU 2000A precession-
**     nutation model, formally adopted by the IAU General Assembly in
**     2000, namely MHB2000 (Mathews et al. 2002), but with the free
**     core nutation (FCN - see Note 4) omitted.
**
**  4) The full MHB2000 model also contains contributions to the
**     nutations in longitude and obliquity due to the free-excitation
**     of the free-core-nutation during the period 1979-2000.  These FCN
**     terms, which are time-dependent and unpredictable, are NOT
**     included in the present function and, if required, must be
**     independently computed.  With the FCN corrections included, the
**     present function delivers a pole which is at current epochs
**     accurate to a few hundred microarcseconds.  The omission of FCN
**     introduces further errors of about that size.
**
**  5) The present function provides classical nutation.  The MHB2000
**     algorithm, from which it is adapted, deals also with (i) the
**     offsets between the GCRS and mean poles and (ii) the adjustments
**     in longitude and obliquity due to the changed precession rates.
**     These additional functions, namely frame bias and precession
**     adjustments, are supported by the ERFA functions eraBi00  and
**     eraPr00.
**
```

```
**  6) The MHB2000 algorithm also provides "total" nutations, comprising
**     the arithmetic sum of the frame bias, precession adjustments,
**     luni-solar nutation and planetary nutation.  These total
**     nutations can be used in combination with an existing IAU 1976
**     precession implementation, such as eraPmat76,  to deliver GCRS-
**     to-true predictions of sub-mas accuracy at current dates.
**     However, there are three shortcomings in the MHB2000 model that
**     must be taken into account if more accurate or definitive results
**     are required (see Wallace 2002):
**
**       (i) The MHB2000 total nutations are simply arithmetic sums,
**           yet in reality the various components are successive Euler
**           rotations.  This slight lack of rigor leads to cross terms
**           that exceed 1 mas after a century.  The rigorous procedure
**           is to form the GCRS-to-true rotation matrix by applying the
**           bias, precession and nutation in that order.
**
**      (ii) Although the precession adjustments are stated to be with
**           respect to Lieske et al. (1977), the MHB2000 model does
**           not specify which set of Euler angles are to be used and
**           how the adjustments are to be applied.  The most literal
**           and straightforward procedure is to adopt the 4-rotation
**           epsilon_0, psi_A, omega_A, xi_A option, and to add DPSIPR
**           to psi_A and DEPSPR to both omega_A and eps_A.
**
**     (iii) The MHB2000 model predates the determination by Chapront
**           et al. (2002) of a 14.6 mas displacement between the
**           J2000.0 mean equinox and the origin of the ICRS frame.  It
**           should, however, be noted that neglecting this displacement
**           when calculating star coordinates does not lead to a
**           14.6 mas change in right ascension, only a small second-
**           order distortion in the pattern of the precession-nutation
**           effect.
**
**     For these reasons, the ERFA functions do not generate the "total
**     nutations" directly, though they can of course easily be
**     generated by calling eraBi00, eraPr00 and the present function
**     and adding the results.
**
**  7) The MHB2000 model contains 41 instances where the same frequency
**     appears multiple times, of which 38 are duplicates and three are
**     triplicates.  To keep the present code close to the original MHB
**     algorithm, this small inefficiency has not been corrected.
**
**  Called:
**     eraFal03     mean anomaly of the Moon
**     eraFaf03     mean argument of the latitude of the Moon
**     eraFaom03    mean longitude of the Moon's ascending node
**     eraFame03    mean longitude of Mercury
**     eraFave03    mean longitude of Venus
**     eraFae03     mean longitude of Earth
**     eraFama03    mean longitude of Mars
**     eraFaju03    mean longitude of Jupiter
**     eraFasa03    mean longitude of Saturn
**     eraFaur03    mean longitude of Uranus
**     eraFapa03    general accumulated precession in longitude
**
**  References:
**
**     Chapront, J., Chapront-Touze, M. & Francou, G. 2002,
**     Astron.Astrophys. 387, 700
**
**     Lieske, J.H., Lederle, T., Fricke, W. & Morando, B. 1977,
**     Astron.Astrophys. 58, 1-16
**
**     Mathews, P.M., Herring, T.A., Buffet, B.A. 2002, J.Geophys.Res.
**     107, B4.  The MHB_2000 code itself was obtained on 9th September
**     2002 from ftp//maia.usno.navy.mil/conv2000/chapter5/IAU2000A.
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
```

```
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**     Wallace, P.T., "Software for Implementing the IAU 2000
**     Resolutions", in IERS Workshop 5.1 (2002)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNut00b(double date1, double date2, double *dpsi, double *deps)
/*
**  - - - - - - - - - -
**   e r a N u t 0 0 b
**  - - - - - - - - - -
**
**  Nutation, IAU 2000B model.
**
**  Given:
**     date1,date2   double    TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps     double    nutation, luni-solar + planetary (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The nutation components in longitude and obliquity are in radians
**     and with respect to the equinox and ecliptic of date.  The
**     obliquity at J2000.0 is assumed to be the Lieske et al. (1977)
**     value of 84381.448 arcsec.  (The errors that result from using
**     this function with the IAU 2006 value of 84381.406 arcsec can be
**     neglected.)
**
**     The nutation model consists only of luni-solar terms, but
**     includes also a fixed offset which compensates for certain long-
**     period planetary terms (Note 7).
**
**  3) This function is an implementation of the IAU 2000B abridged
**     nutation model formally adopted by the IAU General Assembly in
**     2000.  The function computes the MHB_2000_SHORT luni-solar
**     nutation series (Luzum 2001), but without the associated
**     corrections for the precession rate adjustments and the offset
**     between the GCRS and J2000.0 mean poles.
**
**  4) The full IAU 2000A (MHB2000) nutation model contains nearly 1400
**     terms.  The IAU 2000B model (McCarthy & Luzum 2003) contains only
**     77 terms, plus additional simplifications, yet still delivers
**     results of 1 mas accuracy at present epochs.  This combination of
**     accuracy and size makes the IAU 2000B abridged nutation model
**     suitable for most practical applications.
**
**     The function delivers a pole accurate to 1 mas from 1900 to 2100
**     (usually better than 1 mas, very occasionally just outside
**     1 mas).  The full IAU 2000A model, which is implemented in the
**     function eraNut00a (q.v.), delivers considerably greater accuracy
**     at current dates;  however, to realize this improved accuracy,
**     corrections for the essentially unpredictable free-core-nutation
**     (FCN) must also be included.
**
**  5) The present function provides classical nutation.  The
**     MHB_2000_SHORT algorithm, from which it is adapted, deals also
```

```
**     with (i) the offsets between the GCRS and mean poles and (ii) the
**     adjustments in longitude and obliquity due to the changed
**     precession rates.  These additional functions, namely frame bias
**     and precession adjustments, are supported by the ERFA functions
**     eraBi00  and eraPr00.
**
**  6) The MHB_2000_SHORT algorithm also provides "total" nutations,
**     comprising the arithmetic sum of the frame bias, precession
**     adjustments, and nutation (luni-solar + planetary).  These total
**     nutations can be used in combination with an existing IAU 1976
**     precession implementation, such as eraPmat76,  to deliver GCRS-
**     to-true predictions of mas accuracy at current epochs.  However,
**     for symmetry with the eraNut00a  function (q.v. for the reasons),
**     the ERFA functions do not generate the "total nutations"
**     directly.  Should they be required, they could of course easily
**     be generated by calling eraBi00, eraPr00 and the present function
**     and adding the results.
**
**  7) The IAU 2000B model includes "planetary bias" terms that are
**     fixed in size but compensate for long-period nutations.  The
**     amplitudes quoted in McCarthy & Luzum (2003), namely
**     Dpsi = -1.5835 mas and Depsilon = +1.6339 mas, are optimized for
**     the "total nutations" method described in Note 6.  The Luzum
**     (2001) values used in this ERFA implementation, namely -0.135 mas
**     and +0.388 mas, are optimized for the "rigorous" method, where
**     frame bias, precession and nutation are applied separately and in
**     that order.  During the interval 1995-2050, the ERFA
**     implementation delivers a maximum error of 1.001 mas (not
**     including FCN).
**
**  References:
**
**     Lieske, J.H., Lederle, T., Fricke, W., Morando, B., "Expressions
**     for the precession quantities based upon the IAU /1976/ system of
**     astronomical constants", Astron.Astrophys. 58, 1-2, 1-16. (1977)
**
**     Luzum, B., private communication, 2001 (Fortran code
**     MHB_2000_SHORT)
**
**     McCarthy, D.D. & Luzum, B.J., "An abridged model of the
**     precession-nutation of the celestial pole", Cel.Mech.Dyn.Astron.
**     85, 37-49 (2003)
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J., Astron.Astrophys. 282, 663-683 (1994)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNut06a(double date1, double date2, double *dpsi, double *deps)
/*
**  - - - - - - - - - -
**   e r a N u t 0 6 a
**  - - - - - - - - - -
**
**  IAU 2000A nutation with adjustments to match the IAU 2006
**  precession.
**
**  Given:
**     date1,date2   double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps     double   nutation, luni-solar + planetary (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The nutation components in longitude and obliquity are in radians
**     and with respect to the mean equinox and ecliptic of date,
**     IAU 2006 precession model (Hilton et al. 2006, Capitaine et al.
**     2005).
**
**  3) The function first computes the IAU 2000A nutation, then applies
**     adjustments for (i) the consequences of the change in obliquity
**     from the IAU 1980 ecliptic to the IAU 2006 ecliptic and (ii) the
**     secular variation in the Earth's dynamical form factor J2.
**
**  4) The present function provides classical nutation, complementing
**     the IAU 2000 frame bias and IAU 2006 precession.  It delivers a
**     pole which is at current epochs accurate to a few tens of
**     microarcseconds, apart from the free core nutation.
**
**  Called:
**     eraNut00a    nutation, IAU 2000A
**
**  References:
**
**     Chapront, J., Chapront-Touze, M. & Francou, G. 2002,
**     Astron.Astrophys. 387, 700
**
**     Lieske, J.H., Lederle, T., Fricke, W. & Morando, B. 1977,
**     Astron.Astrophys. 58, 1-16
**
**     Mathews, P.M., Herring, T.A., Buffet, B.A. 2002, J.Geophys.Res.
**     107, B4.  The MHB_2000 code itself was obtained on 9th September
**     2002 from ftp//maia.usno.navy.mil/conv2000/chapter5/IAU2000A.
**
**     Simon, J.-L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G., Laskar, J. 1994, Astron.Astrophys. 282, 663-683
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M. 1999,
```

```
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**     Wallace, P.T., "Software for Implementing the IAU 2000
**     Resolutions", in IERS Workshop 5.1 (2002)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNut80(double date1, double date2, double *dpsi, double *deps)
/*
**  - - - - - - - - - -
**   e r a N u t 8 0
**  - - - - - - - - - -
**
**  Nutation, IAU 1980 model.
**
**  Given:
**     date1,date2   double    TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi          double    nutation in longitude (radians)
**     deps          double    nutation in obliquity (radians)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The nutation components are with respect to the ecliptic of
**     date.
**
**  Called:
**     eraAnpm      normalize angle into range +/- pi
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.222 (p111).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraNutm80(double date1, double date2, double rmatn[3][3])
/*
**  - - - - - - - - - -
**   e r a N u t m 8 0
**  - - - - - - - - - -
**
**  Form the matrix of nutation for a given date, IAU 1980 model.
**
**  Given:
**     date1,date2    double          TDB date (Note 1)
**
**  Returned:
**     rmatn          double[3][3]    nutation matrix
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(true) = rmatn * V(mean),
**     where the p-vector V(true) is with respect to the true
**     equatorial triad of date and the p-vector V(mean) is with
**     respect to the mean equatorial triad of date.
**
**  Called:
**     eraNut80     nutation, IAU 1980
**     eraObl80     mean obliquity, IAU 1980
**     eraNumat     form nutation matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraObl06(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a O b l 0 6
**  - - - - - - - - -
**
**  Mean obliquity of the ecliptic, IAU 2006 precession model.
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double   obliquity of the ecliptic (radians, Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result is the angle between the ecliptic and mean equator of
**     date date1+date2.
**
**  Reference:
**
**     Hilton, J. et al., 2006, Celest.Mech.Dyn.Astron. 94, 351
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraObl80(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a O b l 8 0
**  - - - - - - - - -
**
**  Mean obliquity of the ecliptic, IAU 1980 model.
**
**  Given:
**     date1,date2   double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                   double    obliquity of the ecliptic (radians, Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result is the angle between the ecliptic and mean equator of
**     date date1+date2.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Expression 3.222-1 (p114).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraP06e(double date1, double date2,
             double *eps0, double *psia, double *oma, double *bpa,
             double *bqa, double *pia, double *bpia,
             double *epsa, double *chia, double *za, double *zetaa,
             double *thetaa, double *pa,
             double *gam, double *phi, double *psi)
/*
**  - - - - - - - -
**   e r a P 0 6 e
**  - - - - - - - -
**
**  Precession angles, IAU 2006, equinox based.
**
**  Given:
**     date1,date2   double   TT as a 2-part Julian Date (Note 1)
**
**  Returned (see Note 2):
**     eps0          double   epsilon_0
**     psia          double   psi_A
**     oma           double   omega_A
**     bpa           double   P_A
**     bqa           double   Q_A
**     pia           double   pi_A
**     bpia          double   Pi_A
**     epsa          double   obliquity epsilon_A
**     chia          double   chi_A
**     za            double   z_A
**     zetaa         double   zeta_A
**     thetaa        double   theta_A
**     pa            double   p_A
**     gam           double   F-W angle gamma_J2000
**     phi           double   F-W angle phi_J2000
**     psi           double   F-W angle psi_J2000
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) This function returns the set of equinox based angles for the
**     Capitaine et al. "P03" precession theory, adopted by the IAU in
**     2006.  The angles are set out in Table 1 of Hilton et al. (2006):
**
**     eps0   epsilon_0   obliquity at J2000.0
**     psia   psi_A       luni-solar precession
**     oma    omega_A     inclination of equator wrt J2000.0 ecliptic
**     bpa    P_A         ecliptic pole x, J2000.0 ecliptic triad
**     bqa    Q_A         ecliptic pole -y, J2000.0 ecliptic triad
**     pia    pi_A        angle between moving and J2000.0 ecliptics
**     bpia   Pi_A        longitude of ascending node of the ecliptic
**     epsa   epsilon_A   obliquity of the ecliptic
**     chia   chi_A       planetary precession
**     za     z_A         equatorial precession: -3rd 323 Euler angle
**     zetaa  zeta_A      equatorial precession: -1st 323 Euler angle
```

```
**     thetaa theta_A     equatorial precession: 2nd 323 Euler angle
**     pa     p_A         general precession
**     gam    gamma_J2000 J2000.0 RA difference of ecliptic poles
**     phi    phi_J2000   J2000.0 codeclination of ecliptic pole
**     psi    psi_J2000   longitude difference of equator poles, J2000.0
**
**     The returned values are all radians.
**
**  3) Hilton et al. (2006) Table 1 also contains angles that depend on
**     models distinct from the P03 precession theory itself, namely the
**     IAU 2000A frame bias and nutation.  The quoted polynomials are
**     used in other ERFA functions:
**
**     . eraXy06  contains the polynomial parts of the X and Y series.
**
**     . eraS06  contains the polynomial part of the s+XY/2 series.
**
**     . eraPfw06  implements the series for the Fukushima-Williams
**       angles that are with respect to the GCRS pole (i.e. the variants
**       that include frame bias).
**
**  4) The IAU resolution stipulated that the choice of parameterization
**     was left to the user, and so an IAU compliant precession
**     implementation can be constructed using various combinations of
**     the angles returned by the present function.
**
**  5) The parameterization used by ERFA is the version of the Fukushima-
**     Williams angles that refers directly to the GCRS pole.  These
**     angles may be calculated by calling the function eraPfw06.  ERFA
**     also supports the direct computation of the CIP GCRS X,Y by
**     series, available by calling eraXy06.
**
**  6) The agreement between the different parameterizations is at the
**     1 microarcsecond level in the present era.
**
**  7) When constructing a precession formulation that refers to the GCRS
**     pole rather than the dynamical pole, it may (depending on the
**     choice of angles) be necessary to introduce the frame bias
**     explicitly.
**
**  8) It is permissible to re-use the same variable in the returned
**     arguments.  The quantities are stored in the stated order.
**
**  Reference:
**
**     Hilton, J. et al., 2006, Celest.Mech.Dyn.Astron. 94, 351
**
**  Called:
**     eraObl06     mean obliquity, IAU 2006
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPb06(double date1, double date2,
             double *bzeta, double *bz, double *btheta)
/*
**  - - - - - - - -
**   e r a P b 0 6
**  - - - - - - - -
**
**  This function forms three Euler angles which implement general
**  precession from epoch J2000.0, using the IAU 2006 model.  Frame
**  bias (the offset between ICRS and mean J2000.0) is included.
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     bzeta        double   1st rotation: radians cw around z
**     bz           double   3rd rotation: radians cw around z
**     btheta       double   2nd rotation: radians ccw around y
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The traditional accumulated precession angles zeta_A, z_A,
**     theta_A cannot be obtained in the usual way, namely through
**     polynomial expressions, because of the frame bias.  The latter
**     means that two of the angles undergo rapid changes near this
**     date.  They are instead the results of decomposing the
**     precession-bias matrix obtained by using the Fukushima-Williams
**     method, which does not suffer from the problem.  The
**     decomposition returns values which can be used in the
**     conventional formulation and which include frame bias.
**
**  3) The three angles are returned in the conventional order, which
**     is not the same as the order of the corresponding Euler
**     rotations.  The precession-bias matrix is
**     R_3(-z) x R_2(+theta) x R_3(-zeta).
**
**  4) Should zeta_A, z_A, theta_A angles be required that do not
**     contain frame bias, they are available by calling the ERFA
**     function eraP06e.
**
**  Called:
**     eraPmat06    PB matrix, IAU 2006
**     eraRz        rotate around Z-axis
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPfw06(double date1, double date2,
              double *gamb, double *phib, double *psib, double *epsa)
/*
**  - - - - - - - - -
**   e r a P f w 0 6
**  - - - - - - - - -
**
**  Precession angles, IAU 2006 (Fukushima-Williams 4-angle formulation).
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     gamb         double   F-W angle gamma_bar (radians)
**     phib         double   F-W angle phi_bar (radians)
**     psib         double   F-W angle psi_bar (radians)
**     epsa         double   F-W angle epsilon_A (radians)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) Naming the following points:
**
**           e = J2000.0 ecliptic pole,
**           p = GCRS pole,
**           E = mean ecliptic pole of date,
**     and   P = mean pole of date,
**
**     the four Fukushima-Williams angles are as follows:
**
**        gamb = gamma_bar = epE
**        phib = phi_bar = pE
**        psib = psi_bar = pEP
**        epsa = epsilon_A = EP
**
**  3) The matrix representing the combined effects of frame bias and
**     precession is:
**
**        PxB = R_1(-epsa).R_3(-psib).R_1(phib).R_3(gamb)
**
**  4) The matrix representing the combined effects of frame bias,
**     precession and nutation is simply:
**
**        NxPxB = R_1(-epsa-dE).R_3(-psib-dP).R_1(phib).R_3(gamb)
**
**     where dP and dE are the nutation components with respect to the
**     ecliptic of date.
**
**  Reference:
**
**     Hilton, J. et al., 2006, Celest.Mech.Dyn.Astron. 94, 351
**
```

```
**  Called:
**     eraObl06     mean obliquity, IAU 2006
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPmat00(double date1, double date2, double rbp[3][3])
/*
**  - - - - - - - - - - -
**   e r a P m a t 0 0
**  - - - - - - - - - - -
**
**  Precession matrix (including frame bias) from GCRS to a specified
**  date, IAU 2000 model.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rbp          double[3][3]    bias-precession matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rbp * V(GCRS), where
**     the p-vector V(GCRS) is with respect to the Geocentric Celestial
**     Reference System (IAU, 2000) and the p-vector V(date) is with
**     respect to the mean equatorial triad of the given date.
**
**  Called:
**     eraBp00      frame bias and precession matrices, IAU 2000
**
**  Reference:
**
**     IAU: Trans. International Astronomical Union, Vol. XXIVB;  Proc.
**     24th General Assembly, Manchester, UK.  Resolutions B1.3, B1.6.
**     (2000)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPmat06(double date1, double date2, double rbp[3][3])
/*
**  - - - - - - - - - -
**   e r a P m a t 0 6
**  - - - - - - - - - -
**
**  Precession matrix (including frame bias) from GCRS to a specified
**  date, IAU 2006 model.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rbp          double[3][3]    bias-precession matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rbp * V(GCRS), where
**     the p-vector V(GCRS) is with respect to the Geocentric Celestial
**     Reference System (IAU, 2000) and the p-vector V(date) is with
**     respect to the mean equatorial triad of the given date.
**
**  Called:
**     eraPfw06     bias-precession F-W angles, IAU 2006
**     eraFw2m      F-W angles to r-matrix
**
**  References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPmat76(double date1, double date2, double rmatp[3][3])
/*
**  - - - - - - - - - -
**   e r a P m a t 7 6
**  - - - - - - - - - -
**
**  Precession matrix from J2000.0 to a specified date, IAU 1976 model.
**
**  Given:
**     date1,date2 double       ending date, TT (Note 1)
**
**  Returned:
**     rmatp       double[3][3] precession matrix, J2000.0 -> date1+date2
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = RMATP * V(J2000),
**     where the p-vector V(J2000) is with respect to the mean
**     equatorial triad of epoch J2000.0 and the p-vector V(date)
**     is with respect to the mean equatorial triad of the given
**     date.
**
**  3) Though the matrix method itself is rigorous, the precession
**     angles are expressed through canonical polynomials which are
**     valid only for a limited time span.  In addition, the IAU 1976
**     precession rate is known to be imperfect.  The absolute accuracy
**     of the present formulation is better than 0.1 arcsec from
**     1960AD to 2040AD, better than 1 arcsec from 1640AD to 2360AD,
**     and remains below 3 arcsec for the whole of the period
**     500BC to 3000AD.  The errors exceed 10 arcsec outside the
**     range 1200BC to 3900AD, exceed 100 arcsec outside 4200BC to
**     5600AD and exceed 1000 arcsec outside 6800BC to 8200AD.
**
**  Called:
**     eraPrec76    accumulated precession angles, IAU 1976
**     eraIr        initialize r-matrix to identity
**     eraRz        rotate around Z-axis
**     eraRy        rotate around Y-axis
**     eraCr        copy r-matrix
**
**  References:
**
**     Lieske, J.H., 1979, Astron.Astrophys. 73, 282.
**      equations (6) & (7), p283.
**
**     Kaplan,G.H., 1981. USNO circular no. 163, pA2.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn00(double date1, double date2, double dpsi, double deps,
             double *epsa,
             double rb[3][3], double rp[3][3], double rbp[3][3],
             double rn[3][3], double rbpn[3][3])
/*
**  - - - - - - - -
**   e r a P n 0 0
**  - - - - - - - -
**
**  Precession-nutation, IAU 2000 model:  a multi-purpose function,
**  supporting classical (equinox-based) use directly and CIO-based
**  use indirectly.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**     dpsi,deps    double          nutation (Note 2)
**
**  Returned:
**     epsa         double          mean obliquity (Note 3)
**     rb           double[3][3]    frame bias matrix (Note 4)
**     rp           double[3][3]    precession matrix (Note 5)
**     rbp          double[3][3]    bias-precession matrix (Note 6)
**     rn           double[3][3]    nutation matrix (Note 7)
**     rbpn         double[3][3]    GCRS-to-true matrix (Note 8)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The caller is responsible for providing the nutation components;
**     they are in longitude and obliquity, in radians and are with
**     respect to the equinox and ecliptic of date.  For high-accuracy
**     applications, free core nutation should be included as well as
**     any other relevant corrections to the position of the CIP.
**
**  3) The returned mean obliquity is consistent with the IAU 2000
**     precession-nutation models.
**
**  4) The matrix rb transforms vectors from GCRS to J2000.0 mean
**     equator and equinox by applying frame bias.
**
**  5) The matrix rp transforms vectors from J2000.0 mean equator and
**     equinox to mean equator and equinox of date by applying
**     precession.
**
**  6) The matrix rbp transforms vectors from GCRS to mean equator and
**     equinox of date by applying frame bias then precession.  It is
**     the product rp x rb.
**
**  7) The matrix rn transforms vectors from mean equator and equinox of
**     date to true equator and equinox of date by applying the nutation
**     (luni-solar + planetary).
**
```

```
**  8) The matrix rbpn transforms vectors from GCRS to true equator and
**     equinox of date.  It is the product rn x rbp, applying frame
**     bias, precession and nutation in that order.
**
**  9) It is permissible to re-use the same array in the returned
**     arguments.  The arrays are filled in the order given.
**
**  Called:
**     eraPr00      IAU 2000 precession adjustments
**     eraObl80     mean obliquity, IAU 1980
**     eraBp00      frame bias and precession matrices, IAU 2000
**     eraCr        copy r-matrix
**     eraNumat     form nutation matrix
**     eraRxr       product of two r-matrices
**
**  Reference:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn00a(double date1, double date2,
              double *dpsi, double *deps, double *epsa,
              double rb[3][3], double rp[3][3], double rbp[3][3],
              double rn[3][3], double rbpn[3][3])
/*
**  - - - - - - - - -
**   e r a P n 0 0 a
**  - - - - - - - - -
**
**  Precession-nutation, IAU 2000A model:  a multi-purpose function,
**  supporting classical (equinox-based) use directly and CIO-based
**  use indirectly.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps    double          nutation (Note 2)
**     epsa         double          mean obliquity (Note 3)
**     rb           double[3][3]    frame bias matrix (Note 4)
**     rp           double[3][3]    precession matrix (Note 5)
**     rbp          double[3][3]    bias-precession matrix (Note 6)
**     rn           double[3][3]    nutation matrix (Note 7)
**     rbpn         double[3][3]    GCRS-to-true matrix (Notes 8,9)
**
**  Notes:
**
**  1)  The TT date date1+date2 is a Julian Date, apportioned in any
**      convenient way between the two arguments.  For example,
**      JD(TT)=2450123.7 could be expressed in any of these ways,
**      among others:
**
**             date1          date2
**
**          2450123.7           0.0       (JD method)
**          2451545.0       -1421.3       (J2000 method)
**          2400000.5       50123.2       (MJD method)
**          2450123.5           0.2       (date & time method)
**
**      The JD method is the most natural and convenient to use in
**      cases where the loss of several decimal digits of resolution
**      is acceptable.  The J2000 method is best matched to the way
**      the argument is handled internally and will deliver the
**      optimum resolution.  The MJD method and the date & time methods
**      are both good compromises between resolution and convenience.
**
**  2)  The nutation components (luni-solar + planetary, IAU 2000A) in
**      longitude and obliquity are in radians and with respect to the
**      equinox and ecliptic of date.  Free core nutation is omitted;
**      for the utmost accuracy, use the eraPn00  function, where the
**      nutation components are caller-specified.  For faster but
**      slightly less accurate results, use the eraPn00b function.
**
**  3)  The mean obliquity is consistent with the IAU 2000 precession.
**
**  4)  The matrix rb transforms vectors from GCRS to J2000.0 mean
**      equator and equinox by applying frame bias.
**
**  5)  The matrix rp transforms vectors from J2000.0 mean equator and
**      equinox to mean equator and equinox of date by applying
**      precession.
**
**  6)  The matrix rbp transforms vectors from GCRS to mean equator and
**      equinox of date by applying frame bias then precession.  It is
**      the product rp x rb.
**
**  7)  The matrix rn transforms vectors from mean equator and equinox
**      of date to true equator and equinox of date by applying the
**      nutation (luni-solar + planetary).
**
```

```
**  8)  The matrix rbpn transforms vectors from GCRS to true equator and
**      equinox of date.  It is the product rn x rbp, applying frame
**      bias, precession and nutation in that order.
**
**  9)  The X,Y,Z coordinates of the IAU 2000A Celestial Intermediate
**      Pole are elements (3,1-3) of the GCRS-to-true matrix,
**      i.e. rbpn[2][0-2].
**
**  10) It is permissible to re-use the same array in the returned
**      arguments.  The arrays are filled in the order given.
**
**  Called:
**     eraNut00a    nutation, IAU 2000A
**     eraPn00      bias/precession/nutation results, IAU 2000
**
**  Reference:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn00b(double date1, double date2,
              double *dpsi, double *deps, double *epsa,
              double rb[3][3], double rp[3][3], double rbp[3][3],
              double rn[3][3], double rbpn[3][3])
/*
**  - - - - - - - - -
**   e r a P n 0 0 b
**  - - - - - - - - -
**
**  Precession-nutation, IAU 2000B model:  a multi-purpose function,
**  supporting classical (equinox-based) use directly and CIO-based
**  use indirectly.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps    double          nutation (Note 2)
**     epsa         double          mean obliquity (Note 3)
**     rb           double[3][3]    frame bias matrix (Note 4)
**     rp           double[3][3]    precession matrix (Note 5)
**     rbp          double[3][3]    bias-precession matrix (Note 6)
**     rn           double[3][3]    nutation matrix (Note 7)
**     rbpn         double[3][3]    GCRS-to-true matrix (Notes 8,9)
**
**  Notes:
**
**  1)  The TT date date1+date2 is a Julian Date, apportioned in any
**      convenient way between the two arguments.  For example,
**      JD(TT)=2450123.7 could be expressed in any of these ways,
**      among others:
**
**             date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**      The JD method is the most natural and convenient to use in
**      cases where the loss of several decimal digits of resolution
**      is acceptable.  The J2000 method is best matched to the way
**      the argument is handled internally and will deliver the
**      optimum resolution.  The MJD method and the date & time methods
**      are both good compromises between resolution and convenience.
**
**  2)  The nutation components (luni-solar + planetary, IAU 2000B) in
**      longitude and obliquity are in radians and with respect to the
**      equinox and ecliptic of date.  For more accurate results, but
**      at the cost of increased computation, use the eraPn00a function.
**      For the utmost accuracy, use the eraPn00  function, where the
**      nutation components are caller-specified.
**
**  3)  The mean obliquity is consistent with the IAU 2000 precession.
**
**  4)  The matrix rb transforms vectors from GCRS to J2000.0 mean
**      equator and equinox by applying frame bias.
**
**  5)  The matrix rp transforms vectors from J2000.0 mean equator and
**      equinox to mean equator and equinox of date by applying
**      precession.
**
**  6)  The matrix rbp transforms vectors from GCRS to mean equator and
**      equinox of date by applying frame bias then precession.  It is
**      the product rp x rb.
**
**  7)  The matrix rn transforms vectors from mean equator and equinox
**      of date to true equator and equinox of date by applying the
**      nutation (luni-solar + planetary).
**
```

```
**  8)  The matrix rbpn transforms vectors from GCRS to true equator and
**      equinox of date.  It is the product rn x rbp, applying frame
**      bias, precession and nutation in that order.
**
**  9)  The X,Y,Z coordinates of the IAU 2000B Celestial Intermediate
**      Pole are elements (3,1-3) of the GCRS-to-true matrix,
**      i.e. rbpn[2][0-2].
**
**  10) It is permissible to re-use the same array in the returned
**      arguments.  The arrays are filled in the stated order.
**
**  Called:
**     eraNut00b    nutation, IAU 2000B
**     eraPn00      bias/precession/nutation results, IAU 2000
**
**  Reference:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003).
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn06(double date1, double date2, double dpsi, double deps,
             double *epsa,
             double rb[3][3], double rp[3][3], double rbp[3][3],
             double rn[3][3], double rbpn[3][3])
/*
**  - - - - - - - - -
**   e r a P n 0 6
**  - - - - - - - - -
**
**  Precession-nutation, IAU 2006 model:  a multi-purpose function,
**  supporting classical (equinox-based) use directly and CIO-based use
**  indirectly.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**     dpsi,deps    double          nutation (Note 2)
**
**  Returned:
**     epsa         double          mean obliquity (Note 3)
**     rb           double[3][3]    frame bias matrix (Note 4)
**     rp           double[3][3]    precession matrix (Note 5)
**     rbp          double[3][3]    bias-precession matrix (Note 6)
**     rn           double[3][3]    nutation matrix (Note 7)
**     rbpn         double[3][3]    GCRS-to-true matrix (Note 8)
**
**  Notes:
**
**  1)  The TT date date1+date2 is a Julian Date, apportioned in any
**      convenient way between the two arguments.  For example,
**      JD(TT)=2450123.7 could be expressed in any of these ways,
**      among others:
**
**             date1          date2
**
**          2450123.7           0.0       (JD method)
**          2451545.0       -1421.3       (J2000 method)
**          2400000.5       50123.2       (MJD method)
**          2450123.5           0.2       (date & time method)
**
**      The JD method is the most natural and convenient to use in
**      cases where the loss of several decimal digits of resolution
**      is acceptable.  The J2000 method is best matched to the way
**      the argument is handled internally and will deliver the
**      optimum resolution.  The MJD method and the date & time methods
**      are both good compromises between resolution and convenience.
**
**  2)  The caller is responsible for providing the nutation components;
**      they are in longitude and obliquity, in radians and are with
**      respect to the equinox and ecliptic of date.  For high-accuracy
**      applications, free core nutation should be included as well as
**      any other relevant corrections to the position of the CIP.
**
**  3)  The returned mean obliquity is consistent with the IAU 2006
**      precession.
**
**  4)  The matrix rb transforms vectors from GCRS to J2000.0 mean
**      equator and equinox by applying frame bias.
**
**  5)  The matrix rp transforms vectors from J2000.0 mean equator and
**      equinox to mean equator and equinox of date by applying
**      precession.
**
**  6)  The matrix rbp transforms vectors from GCRS to mean equator and
**      equinox of date by applying frame bias then precession.  It is
**      the product rp x rb.
**
**  7)  The matrix rn transforms vectors from mean equator and equinox
**      of date to true equator and equinox of date by applying the
**      nutation (luni-solar + planetary).
**
```

```
**  8)  The matrix rbpn transforms vectors from GCRS to true equator and
**      equinox of date.  It is the product rn x rbp, applying frame
**      bias, precession and nutation in that order.
**
**  9)  The X,Y,Z coordinates of the Celestial Intermediate Pole are
**      elements (3,1-3) of the GCRS-to-true matrix, i.e. rbpn[2][0-2].
**
**  10) It is permissible to re-use the same array in the returned
**      arguments.  The arrays are filled in the stated order.
**
**  Called:
**     eraPfw06     bias-precession F-W angles, IAU 2006
**     eraFw2m      F-W angles to r-matrix
**     eraCr        copy r-matrix
**     eraTr        transpose r-matrix
**     eraRxr       product of two r-matrices
**
**  References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn06a(double date1, double date2,
              double *dpsi, double *deps, double *epsa,
              double rb[3][3], double rp[3][3], double rbp[3][3],
              double rn[3][3], double rbpn[3][3])
/*
**  - - - - - - - - -
**   e r a P n 0 6 a
**  - - - - - - - - -
**
**  Precession-nutation, IAU 2006/2000A models:  a multi-purpose function,
**  supporting classical (equinox-based) use directly and CIO-based use
**  indirectly.
**
**  Given:
**     date1,date2  double          TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsi,deps    double          nutation (Note 2)
**     epsa         double          mean obliquity (Note 3)
**     rb           double[3][3]    frame bias matrix (Note 4)
**     rp           double[3][3]    precession matrix (Note 5)
**     rbp          double[3][3]    bias-precession matrix (Note 6)
**     rn           double[3][3]    nutation matrix (Note 7)
**     rbpn         double[3][3]    GCRS-to-true matrix (Notes 8,9)
**
**  Notes:
**
**  1)  The TT date date1+date2 is a Julian Date, apportioned in any
**      convenient way between the two arguments.  For example,
**      JD(TT)=2450123.7 could be expressed in any of these ways,
**      among others:
**
**             date1          date2
**
**          2450123.7           0.0       (JD method)
**          2451545.0       -1421.3       (J2000 method)
**          2400000.5       50123.2       (MJD method)
**          2450123.5           0.2       (date & time method)
**
**      The JD method is the most natural and convenient to use in
**      cases where the loss of several decimal digits of resolution
**      is acceptable.  The J2000 method is best matched to the way
**      the argument is handled internally and will deliver the
**      optimum resolution.  The MJD method and the date & time methods
**      are both good compromises between resolution and convenience.
**
**  2)  The nutation components (luni-solar + planetary, IAU 2000A) in
**      longitude and obliquity are in radians and with respect to the
**      equinox and ecliptic of date.  Free core nutation is omitted;
**      for the utmost accuracy, use the eraPn06 function, where the
**      nutation components are caller-specified.
**
**  3)  The mean obliquity is consistent with the IAU 2006 precession.
**
**  4)  The matrix rb transforms vectors from GCRS to mean J2000.0 by
**      applying frame bias.
**
**  5)  The matrix rp transforms vectors from mean J2000.0 to mean of
**      date by applying precession.
**
**  6)  The matrix rbp transforms vectors from GCRS to mean of date by
**      applying frame bias then precession.  It is the product rp x rb.
**
**  7)  The matrix rn transforms vectors from mean of date to true of
**      date by applying the nutation (luni-solar + planetary).
**
**  8)  The matrix rbpn transforms vectors from GCRS to true of date
**      (CIP/equinox).  It is the product rn x rbp, applying frame bias,
**      precession and nutation in that order.
**
```

```
**  9)  The X,Y,Z coordinates of the IAU 2006/2000A Celestial
**      Intermediate Pole are elements (3,1-3) of the GCRS-to-true
**      matrix, i.e. rbpn[2][0-2].
**
**  10) It is permissible to re-use the same array in the returned
**      arguments.  The arrays are filled in the stated order.
**
**  Called:
**     eraNut06a    nutation, IAU 2006/2000A
**     eraPn06      bias/precession/nutation results, IAU 2006
**
**  Reference:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPnm00a(double date1, double date2, double rbpn[3][3])
/*
**  - - - - - - - - - - -
**   e r a P n m 0 0 a
**  - - - - - - - - - - -
**
**  Form the matrix of precession-nutation for a given date (including
**  frame bias), equinox-based, IAU 2000A model.
**
**  Given:
**     date1,date2  double     TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rbpn         double[3][3]    classical NPB matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rbpn * V(GCRS), where
**     the p-vector V(date) is with respect to the true equatorial triad
**     of date date1+date2 and the p-vector V(GCRS) is with respect to
**     the Geocentric Celestial Reference System (IAU, 2000).
**
**  3) A faster, but slightly less accurate result (about 1 mas), can be
**     obtained by using instead the eraPnm00b function.
**
**  Called:
**     eraPn00a     bias/precession/nutation, IAU 2000A
**
**  Reference:
**
**     IAU: Trans. International Astronomical Union, Vol. XXIVB;  Proc.
**     24th General Assembly, Manchester, UK.  Resolutions B1.3, B1.6.
**     (2000)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPnm00b(double date1, double date2, double rbpn[3][3])
/*
**  - - - - - - - - - - -
**   e r a P n m 0 0 b
**  - - - - - - - - - - -
**
**  Form the matrix of precession-nutation for a given date (including
**  frame bias), equinox-based, IAU 2000B model.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rbpn        double[3][3] bias-precession-nutation matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rbpn * V(GCRS), where
**     the p-vector V(date) is with respect to the true equatorial triad
**     of date date1+date2 and the p-vector V(GCRS) is with respect to
**     the Geocentric Celestial Reference System (IAU, 2000).
**
**  3) The present function is faster, but slightly less accurate (about
**     1 mas), than the eraPnm00a function.
**
**  Called:
**     eraPn00b     bias/precession/nutation, IAU 2000B
**
**  Reference:
**
**     IAU: Trans. International Astronomical Union, Vol. XXIVB;  Proc.
**     24th General Assembly, Manchester, UK.  Resolutions B1.3, B1.6.
**     (2000)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPnm06a(double date1, double date2, double rnpb[3][3])
/*
**  - - - - - - - - - -
**   e r a P n m 0 6 a
**  - - - - - - - - - -
**
**  Form the matrix of precession-nutation for a given date (including
**  frame bias), IAU 2006 precession and IAU 2000A nutation models.
**
**  Given:
**     date1,date2 double       TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     rnpb        double[3][3] bias-precession-nutation matrix (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rnpb * V(GCRS), where
**     the p-vector V(date) is with respect to the true equatorial triad
**     of date date1+date2 and the p-vector V(GCRS) is with respect to
**     the Geocentric Celestial Reference System (IAU, 2000).
**
**  Called:
**     eraPfw06     bias-precession F-W angles, IAU 2006
**     eraNut06a    nutation, IAU 2006/2000A
**     eraFw2m      F-W angles to r-matrix
**
**  Reference:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPnm80(double date1, double date2, double rmatpn[3][3])
/*
**  - - - - - - - - -
**   e r a P n m 8 0
**  - - - - - - - - -
**
**  Form the matrix of precession/nutation for a given date, IAU 1976
**  precession model, IAU 1980 nutation model.
**
**  Given:
**     date1,date2    double         TDB date (Note 1)
**
**  Returned:
**     rmatpn         double[3][3]   combined precession/nutation matrix
**
**  Notes:
**
**  1) The TDB date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TDB)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The matrix operates in the sense V(date) = rmatpn * V(J2000),
**     where the p-vector V(date) is with respect to the true equatorial
**     triad of date date1+date2 and the p-vector V(J2000) is with
**     respect to the mean equatorial triad of epoch J2000.0.
**
**  Called:
**     eraPmat76    precession matrix, IAU 1976
**     eraNutm80    nutation matrix, IAU 1980
**     eraRxr       product of two r-matrices
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 3.3 (p145).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPom00(double xp, double yp, double sp, double rpom[3][3])
/*
**  - - - - - - - - - - -
**   e r a P o m 0 0
**  - - - - - - - - - - -
**
**  Form the matrix of polar motion for a given date, IAU 2000.
**
**  Given:
**     xp,yp    double    coordinates of the pole (radians, Note 1)
**     sp       double    the TIO locator s' (radians, Note 2)
**
**  Returned:
**     rpom     double[3][3]   polar-motion matrix (Note 3)
**
**  Notes:
**
**  1) The arguments xp and yp are the coordinates (in radians) of the
**     Celestial Intermediate Pole with respect to the International
**     Terrestrial Reference System (see IERS Conventions 2003),
**     measured along the meridians to 0 and 90 deg west respectively.
**
**  2) The argument sp is the TIO locator s', in radians, which
**     positions the Terrestrial Intermediate Origin on the equator.  It
**     is obtained from polar motion observations by numerical
**     integration, and so is in essence unpredictable.  However, it is
**     dominated by a secular drift of about 47 microarcseconds per
**     century, and so can be taken into account by using s' = -47*t,
**     where t is centuries since J2000.0.  The function eraSp00
**     implements this approximation.
**
**  3) The matrix operates in the sense V(TRS) = rpom * V(CIP), meaning
**     that it is the final rotation when computing the pointing
**     direction to a celestial source.
**
**  Called:
**     eraIr        initialize r-matrix to identity
**     eraRz        rotate around Z-axis
**     eraRy        rotate around Y-axis
**     eraRx        rotate around X-axis
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPr00(double date1, double date2, double *dpsipr, double *depspr)
/*
**  - - - - - - - -
**   e r a P r 0 0
**  - - - - - - - -
**
**  Precession-rate part of the IAU 2000 precession-nutation models
**  (part of MHB2000).
**
**  Given:
**     date1,date2    double  TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     dpsipr,depspr  double  precession corrections (Notes 2,3)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The precession adjustments are expressed as "nutation
**     components", corrections in longitude and obliquity with respect
**     to the J2000.0 equinox and ecliptic.
**
**  3) Although the precession adjustments are stated to be with respect
**     to Lieske et al. (1977), the MHB2000 model does not specify which
**     set of Euler angles are to be used and how the adjustments are to
**     be applied.  The most literal and straightforward procedure is to
**     adopt the 4-rotation epsilon_0, psi_A, omega_A, xi_A option, and
**     to add dpsipr to psi_A and depspr to both omega_A and eps_A.
**
**  4) This is an implementation of one aspect of the IAU 2000A nutation
**     model, formally adopted by the IAU General Assembly in 2000,
**     namely MHB2000 (Mathews et al. 2002).
**
**  References:
**
**     Lieske, J.H., Lederle, T., Fricke, W. & Morando, B., "Expressions
**     for the precession quantities based upon the IAU (1976) System of
**     Astronomical Constants", Astron.Astrophys., 58, 1-16 (1977)
**
**     Mathews, P.M., Herring, T.A., Buffet, B.A., "Modeling of nutation
**     and precession   New nutation series for nonrigid Earth and
**     insights into the Earth’s interior", J.Geophys.Res., 107, B4,
**     2002.  The MHB2000 code itself was obtained on 9th September 2002
**     from ftp://maia.usno.navy.mil/conv2000/chapter5/IAU2000A.
**
**     Wallace, P.T., "Software for Implementing the IAU 2000
**     Resolutions", in IERS Workshop 5.1 (2002).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPrec76(double date01, double date02, double date11, double date12,
               double *zeta, double *z, double *theta)
/*
**  - - - - - - - - - - -
**   e r a P r e c 7 6
**  - - - - - - - - - - -
**
**  IAU 1976 precession model.
**
**  This function forms the three Euler angles which implement general
**  precession between two dates, using the IAU 1976 model (as for the
**  FK5 catalog).
**
**  Given:
**     date01,date02   double    TDB starting date (Note 1)
**     date11,date12   double    TDB ending date (Note 1)
**
**  Returned:
**     zeta            double    1st rotation: radians cw around z
**     z               double    3rd rotation: radians cw around z
**     theta           double    2nd rotation: radians ccw around y
**
**  Notes:
**
**  1) The dates date01+date02 and date11+date12 are Julian Dates,
**     apportioned in any convenient way between the arguments daten1
**     and daten2.  For example, JD(TDB)=2450123.7 could be expressed in
**     any of these ways, among others:
**
**           daten1        daten2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 method is best matched to the way the
**     argument is handled internally and will deliver the optimum
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**     The two dates may be expressed using different methods, but at
**     the risk of losing some resolution.
**
**  2) The accumulated precession angles zeta, z, theta are expressed
**     through canonical polynomials which are valid only for a limited
**     time span.  In addition, the IAU 1976 precession rate is known to
**     be imperfect.  The absolute accuracy of the present formulation
**     is better than 0.1 arcsec from 1960AD to 2040AD, better than
**     1 arcsec from 1640AD to 2360AD, and remains below 3 arcsec for
**     the whole of the period 500BC to 3000AD.  The errors exceed
**     10 arcsec outside the range 1200BC to 3900AD, exceed 100 arcsec
**     outside 4200BC to 5600AD and exceed 1000 arcsec outside 6800BC to
**     8200AD.
**
**  3) The three angles are returned in the conventional order, which
**     is not the same as the order of the corresponding Euler
**     rotations.  The precession matrix is
**     R_3(-z) x R_2(+theta) x R_3(-zeta).
**
**  Reference:
**
**     Lieske, J.H., 1979, Astron.Astrophys. 73, 282, equations
**     (6) & (7), p283.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraS00(double date1, double date2, double x, double y)
/*
**  - - - - - - -
**   e r a S 0 0
**  - - - - - - -
**
**  The CIO locator s, positioning the Celestial Intermediate Origin on
**  the equator of the Celestial Intermediate Pole, given the CIP's X,Y
**  coordinates.  Compatible with IAU 2000A precession-nutation.
**
**  Given:
**     date1,date2   double    TT as a 2-part Julian Date (Note 1)
**     x,y           double    CIP coordinates (Note 3)
**
**  Returned (function value):
**                   double    the CIO locator s in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The CIO locator s is the difference between the right ascensions
**     of the same point in two systems:  the two systems are the GCRS
**     and the CIP,CIO, and the point is the ascending node of the
**     CIP equator.  The quantity s remains below 0.1 arcsecond
**     throughout 1900-2100.
**
**  3) The series used to compute s is in fact for s+XY/2, where X and Y
**     are the x and y components of the CIP unit vector;  this series
**     is more compact than a direct series for s would be.  This
**     function requires X,Y to be supplied by the caller, who is
**     responsible for providing values that are consistent with the
**     supplied date.
**
**  4) The model is consistent with the IAU 2000A precession-nutation.
**
**  Called:
**     eraFal03     mean anomaly of the Moon
**     eraFalp03    mean anomaly of the Sun
**     eraFaf03     mean argument of the latitude of the Moon
**     eraFad03     mean elongation of the Moon from the Sun
**     eraFaom03    mean longitude of the Moon's ascending node
**     eraFave03    mean longitude of Venus
**     eraFae03     mean longitude of Earth
**     eraFapa03    general accumulated precession in longitude
**
**  References:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
```

```
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraS00a(double date1, double date2)
/*
**  - - - - - - - -
**   e r a S 0 0 a
**  - - - - - - - -
**
**  The CIO locator s, positioning the Celestial Intermediate Origin on
**  the equator of the Celestial Intermediate Pole, using the IAU 2000A
**  precession-nutation model.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    the CIO locator s in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The CIO locator s is the difference between the right ascensions
**     of the same point in two systems.  The two systems are the GCRS
**     and the CIP,CIO, and the point is the ascending node of the
**     CIP equator.  The CIO locator s remains a small fraction of
**     1 arcsecond throughout 1900-2100.
**
**  3) The series used to compute s is in fact for s+XY/2, where X and Y
**     are the x and y components of the CIP unit vector;  this series
**     is more compact than a direct series for s would be.  The present
**     function uses the full IAU 2000A nutation model when predicting
**     the CIP position.  Faster results, with no significant loss of
**     accuracy, can be obtained via the function eraS00b, which uses
**     instead the IAU 2000B truncated model.
**
**  Called:
**     eraPnm00a    classical NPB matrix, IAU 2000A
**     eraBnp2xy    extract CIP X,Y from the BPN matrix
**     eraS00       the CIO locator s, given X,Y, IAU 2000A
**
**  References:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
```

```
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraS00b(double date1, double date2)
/*
**  - - - - - - - -
**   e r a S 0 0 b
**  - - - - - - - -
**
**  The CIO locator s, positioning the Celestial Intermediate Origin on
**  the equator of the Celestial Intermediate Pole, using the IAU 2000B
**  precession-nutation model.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    the CIO locator s in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The CIO locator s is the difference between the right ascensions
**     of the same point in two systems.  The two systems are the GCRS
**     and the CIP,CIO, and the point is the ascending node of the
**     CIP equator.  The CIO locator s remains a small fraction of
**     1 arcsecond throughout 1900-2100.
**
**  3) The series used to compute s is in fact for s+XY/2, where X and Y
**     are the x and y components of the CIP unit vector;  this series
**     is more compact than a direct series for s would be.  The present
**     function uses the IAU 2000B truncated nutation model when
**     predicting the CIP position.  The function eraS00a uses instead
**     the full IAU 2000A model, but with no significant increase in
**     accuracy and at some cost in speed.
**
**  Called:
**     eraPnm00b    classical NPB matrix, IAU 2000B
**     eraBnp2xy    extract CIP X,Y from the BPN matrix
**     eraS00       the CIO locator s, given X,Y, IAU 2000A
**
**  References:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
```

```
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraS06(double date1, double date2, double x, double y)
/*
**  - - - - - - - 
**   e r a S 0 6
**  - - - - - - - 
**
**  The CIO locator s, positioning the Celestial Intermediate Origin on
**  the equator of the Celestial Intermediate Pole, given the CIP's X,Y
**  coordinates.  Compatible with IAU 2006/2000A precession-nutation.
**
**  Given:
**     date1,date2   double    TT as a 2-part Julian Date (Note 1)
**     x,y           double    CIP coordinates (Note 3)
**
**  Returned (function value):
**                   double    the CIO locator s in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The CIO locator s is the difference between the right ascensions
**     of the same point in two systems:  the two systems are the GCRS
**     and the CIP,CIO, and the point is the ascending node of the
**     CIP equator.  The quantity s remains below 0.1 arcsecond
**     throughout 1900-2100.
**
**  3) The series used to compute s is in fact for s+XY/2, where X and Y
**     are the x and y components of the CIP unit vector;  this series
**     is more compact than a direct series for s would be.  This
**     function requires X,Y to be supplied by the caller, who is
**     responsible for providing values that are consistent with the
**     supplied date.
**
**  4) The model is consistent with the "P03" precession (Capitaine et
**     al. 2003), adopted by IAU 2006 Resolution 1, 2006, and the
**     IAU 2000A nutation (with P03 adjustments).
**
**  Called:
**     eraFal03     mean anomaly of the Moon
**     eraFalp03    mean anomaly of the Sun
**     eraFaf03     mean argument of the latitude of the Moon
**     eraFad03     mean elongation of the Moon from the Sun
**     eraFaom03    mean longitude of the Moon's ascending node
**     eraFave03    mean longitude of Venus
**     eraFae03     mean longitude of Earth
**     eraFapa03    general accumulated precession in longitude
**
**  References:
**
**     Capitaine, N., Wallace, P.T. & Chapront, J., 2003, Astron.
**     Astrophys. 432, 355
**
```

```
**     McCarthy, D.D., Petit, G. (eds.) 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraS06a(double date1, double date2)
/*
**  - - - - - - - -
**   e r a S 0 6 a
**  - - - - - - - -
**
**  The CIO locator s, positioning the Celestial Intermediate Origin on
**  the equator of the Celestial Intermediate Pole, using the IAU 2006
**  precession and IAU 2000A nutation models.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    the CIO locator s in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The CIO locator s is the difference between the right ascensions
**     of the same point in two systems.  The two systems are the GCRS
**     and the CIP,CIO, and the point is the ascending node of the
**     CIP equator.  The CIO locator s remains a small fraction of
**     1 arcsecond throughout 1900-2100.
**
**  3) The series used to compute s is in fact for s+XY/2, where X and Y
**     are the x and y components of the CIP unit vector;  this series is
**     more compact than a direct series for s would be.  The present
**     function uses the full IAU 2000A nutation model when predicting
**     the CIP position.
**
**  Called:
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**
**  References:
**
**     Capitaine, N., Chapront, J., Lambert, S. and Wallace, P.,
**     "Expressions for the Celestial Intermediate Pole and Celestial
**     Ephemeris Origin consistent with the IAU 2000A precession-
**     nutation model", Astron.Astrophys. 400, 1145-1154 (2003)
**
**     n.b. The celestial ephemeris origin (CEO) was renamed "celestial
**          intermediate origin" (CIO) by IAU 2006 Resolution 2.
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
```

```
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraSp00(double date1, double date2)
/*
**  - - - - - - - -
**   e r a S p 0 0
**  - - - - - - - -
**
**  The TIO locator s', positioning the Terrestrial Intermediate Origin
**  on the equator of the Celestial Intermediate Pole.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    the TIO locator s' in radians (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The TIO locator s' is obtained from polar motion observations by
**     numerical integration, and so is in essence unpredictable.
**     However, it is dominated by a secular drift of about
**     47 microarcseconds per century, which is the approximation
**     evaluated by the present function.
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraXy06(double date1, double date2, double *x, double *y)
/*
**  - - - - - - - -
**   e r a X y 0 6
**  - - - - - - - -
**
**  X,Y coordinates of celestial intermediate pole from series based
**  on IAU 2006 precession and IAU 2000A nutation.
**
**  Given:
**     date1,date2  double     TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     x,y          double     CIP X,Y coordinates (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The X,Y coordinates are those of the unit vector towards the
**     celestial intermediate pole.  They represent the combined effects
**     of frame bias, precession and nutation.
**
**  3) The fundamental arguments used are as adopted in IERS Conventions
**     (2003) and are from Simon et al. (1994) and Souchay et al.
**     (1999).
**
**  4) This is an alternative to the angles-based method, via the ERFA
**     function eraFw2xy and as used in eraXys06a for example.  The two
**     methods agree at the 1 microarcsecond level (at present), a
**     negligible amount compared with the intrinsic accuracy of the
**     models.  However, it would be unwise to mix the two methods
**     (angles-based and series-based) in a single application.
**
**  Called:
**     eraFal03     mean anomaly of the Moon
**     eraFalp03    mean anomaly of the Sun
**     eraFaf03     mean argument of the latitude of the Moon
**     eraFad03     mean elongation of the Moon from the Sun
**     eraFaom03    mean longitude of the Moon's ascending node
**     eraFame03    mean longitude of Mercury
**     eraFave03    mean longitude of Venus
**     eraFae03     mean longitude of Earth
**     eraFama03    mean longitude of Mars
**     eraFaju03    mean longitude of Jupiter
**     eraFasa03    mean longitude of Saturn
**     eraFaur03    mean longitude of Uranus
**     eraFane03    mean longitude of Neptune
**     eraFapa03    general accumulated precession in longitude
**
**  References:
**
**     Capitaine, N., Wallace, P.T. & Chapront, J., 2003,
```

```
**     Astron.Astrophys., 412, 567
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**     Simon, J.L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G. & Laskar, J., Astron.Astrophys., 1994, 282, 663
**
**     Souchay, J., Loysel, B., Kinoshita, H., Folgueira, M., 1999,
**     Astron.Astrophys.Supp.Ser. 135, 111
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraXys00a(double date1, double date2,
               double *x, double *y, double *s)
/*
**  - - - - - - - - - -
**   e r a X y s 0 0 a
**  - - - - - - - - - -
**
**  For a given TT date, compute the X,Y coordinates of the Celestial
**  Intermediate Pole and the CIO locator s, using the IAU 2000A
**  precession-nutation model.
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     x,y          double   Celestial Intermediate Pole (Note 2)
**     s            double   the CIO locator s (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The Celestial Intermediate Pole coordinates are the x,y
**     components of the unit vector in the Geocentric Celestial
**     Reference System.
**
**  3) The CIO locator s (in radians) positions the Celestial
**     Intermediate Origin on the equator of the CIP.
**
**  4) A faster, but slightly less accurate result (about 1 mas for
**     X,Y), can be obtained by using instead the eraXys00b function.
**
**  Called:
**     eraPnm00a    classical NPB matrix, IAU 2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS00       the CIO locator s, given X,Y, IAU 2000A
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraXys00b(double date1, double date2,
               double *x, double *y, double *s)
/*
**  - - - - - - - - - -
**   e r a X y s 0 0 b
**  - - - - - - - - - -
**
**  For a given TT date, compute the X,Y coordinates of the Celestial
**  Intermediate Pole and the CIO locator s, using the IAU 2000B
**  precession-nutation model.
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     x,y          double   Celestial Intermediate Pole (Note 2)
**     s            double   the CIO locator s (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The Celestial Intermediate Pole coordinates are the x,y
**     components of the unit vector in the Geocentric Celestial
**     Reference System.
**
**  3) The CIO locator s (in radians) positions the Celestial
**     Intermediate Origin on the equator of the CIP.
**
**  4) The present function is faster, but slightly less accurate (about
**     1 mas in X,Y), than the eraXys00a function.
**
**  Called:
**     eraPnm00b    classical NPB matrix, IAU 2000B
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS00       the CIO locator s, given X,Y, IAU 2000A
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraXys06a(double date1, double date2,
               double *x, double *y, double *s)
/*
**  - - - - - - - - - -
**   e r a X y s 0 6 a
**  - - - - - - - - - -
**
**  For a given TT date, compute the X,Y coordinates of the Celestial
**  Intermediate Pole and the CIO locator s, using the IAU 2006
**  precession and IAU 2000A nutation models.
**
**  Given:
**     date1,date2  double  TT as a 2-part Julian Date (Note 1)
**
**  Returned:
**     x,y          double  Celestial Intermediate Pole (Note 2)
**     s            double  the CIO locator s (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The Celestial Intermediate Pole coordinates are the x,y components
**     of the unit vector in the Geocentric Celestial Reference System.
**
**  3) The CIO locator s (in radians) positions the Celestial
**     Intermediate Origin on the equator of the CIP.
**
**  4) Series-based solutions for generating X and Y are also available:
**     see Capitaine & Wallace (2006) and eraXy06.
**
**  Called:
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**
**  References:
**
**     Capitaine, N. & Wallace, P.T., 2006, Astron.Astrophys. 450, 855
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEe00(double date1, double date2, double epsa, double dpsi)
/*
**  - - - - - - - -
**   e r a E e 0 0
**  - - - - - - - -
**
**  The equation of the equinoxes, compatible with IAU 2000 resolutions,
**  given the nutation in longitude and the mean obliquity.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**     epsa         double    mean obliquity (Note 2)
**     dpsi         double    nutation in longitude (Note 3)
**
**  Returned (function value):
**                  double    equation of the equinoxes (Note 4)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The obliquity, in radians, is mean of date.
**
**  3) The result, which is in radians, operates in the following sense:
**
**        Greenwich apparent ST = GMST + equation of the equinoxes
**
**  4) The result is compatible with the IAU 2000 resolutions.  For
**     further details, see IERS Conventions 2003 and Capitaine et al.
**     (2002).
**
**  Called:
**     eraEect00    equation of the equinoxes complementary terms
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEe00a(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a E e 0 0 a
**  - - - - - - - - -
**
**  Equation of the equinoxes, compatible with IAU 2000 resolutions.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    equation of the equinoxes (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result, which is in radians, operates in the following sense:
**
**        Greenwich apparent ST = GMST + equation of the equinoxes
**
**  3) The result is compatible with the IAU 2000 resolutions.  For
**     further details, see IERS Conventions 2003 and Capitaine et al.
**     (2002).
**
**  Called:
**     eraPr00      IAU 2000 precession adjustments
**     eraObl80     mean obliquity, IAU 1980
**     eraNut00a    nutation, IAU 2000A
**     eraEe00      equation of the equinoxes, IAU 2000
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003).
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEe00b(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a E e 0 0 b
**  - - - - - - - - -
**
**  Equation of the equinoxes, compatible with IAU 2000 resolutions but
**  using the truncated nutation model IAU 2000B.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    equation of the equinoxes (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result, which is in radians, operates in the following sense:
**
**        Greenwich apparent ST = GMST + equation of the equinoxes
**
**  3) The result is compatible with the IAU 2000 resolutions except
**     that accuracy has been compromised for the sake of speed.  For
**     further details, see McCarthy & Luzum (2001), IERS Conventions
**     2003 and Capitaine et al. (2003).
**
**  Called:
**     eraPr00      IAU 2000 precession adjustments
**     eraObl80     mean obliquity, IAU 1980
**     eraNut00b    nutation, IAU 2000B
**     eraEe00      equation of the equinoxes, IAU 2000
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003)
**
**     McCarthy, D.D. & Luzum, B.J., "An abridged model of the
**     precession-nutation of the celestial pole", Celestial Mechanics &
**     Dynamical Astronomy, 85, 37-49 (2003)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEe06a(double date1, double date2)
/*
**  - - - - - - - - -
**   e r a E e 0 6 a
**  - - - - - - - - -
**
**  Equation of the equinoxes, compatible with IAU 2000 resolutions and
**  IAU 2006/2000A precession-nutation.
**
**  Given:
**     date1,date2  double    TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double    equation of the equinoxes (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result, which is in radians, operates in the following sense:
**
**        Greenwich apparent ST = GMST + equation of the equinoxes
**
**  Called:
**     eraAnpm      normalize angle into range +/- pi
**     eraGst06a    Greenwich apparent sidereal time, IAU 2006/2000A
**     eraGmst06    Greenwich mean sidereal time, IAU 2006
**
**  Reference:
**
**     McCarthy, D. D., Petit, G. (eds.), 2004, IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEect00(double date1, double date2)
/*
**  - - - - - - - - - - -
**   e r a E e c t 0 0
**  - - - - - - - - - - -
**
**  Equation of the equinoxes complementary terms, consistent with
**  IAU 2000 resolutions.
**
**  Given:
**     date1,date2  double   TT as a 2-part Julian Date (Note 1)
**
**  Returned (function value):
**                  double   complementary terms (Note 2)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The "complementary terms" are part of the equation of the
**     equinoxes (EE), classically the difference between apparent and
**     mean Sidereal Time:
**
**        GAST = GMST + EE
**
**     with:
**
**        EE = dpsi * cos(eps)
**
**     where dpsi is the nutation in longitude and eps is the obliquity
**     of date.  However, if the rotation of the Earth were constant in
**     an inertial frame the classical formulation would lead to
**     apparent irregularities in the UT1 timescale traceable to side-
**     effects of precession-nutation.  In order to eliminate these
**     effects from UT1, "complementary terms" were introduced in 1994
**     (IAU, 1994) and took effect from 1997 (Capitaine and Gontier,
**     1993):
**
**        GAST = GMST + CT + EE
**
**     By convention, the complementary terms are included as part of
**     the equation of the equinoxes rather than as part of the mean
**     Sidereal Time.  This slightly compromises the "geometrical"
**     interpretation of mean sidereal time but is otherwise
**     inconsequential.
**
**     The present function computes CT in the above expression,
**     compatible with IAU 2000 resolutions (Capitaine et al., 2002, and
**     IERS Conventions 2003).
**
**  Called:
**     eraFal03     mean anomaly of the Moon
**     eraFalp03    mean anomaly of the Sun
```

```
**     eraFaf03     mean argument of the latitude of the Moon
**     eraFad03     mean elongation of the Moon from the Sun
**     eraFaom03    mean longitude of the Moon’s ascending node
**     eraFave03    mean longitude of Venus
**     eraFae03     mean longitude of Earth
**     eraFapa03    general accumulated precession in longitude
**
**  References:
**
**     Capitaine, N. & Gontier, A.-M., Astron.Astrophys., 275,
**     645-650 (1993)
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astron.Astrophys., 406,
**     1135-1149 (2003)
**
**     IAU Resolution C7, Recommendation 3 (1994)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEqeq94(double date1, double date2)
/*
**  - - - - - - - - - - -
**   e r a E q e q 9 4
**  - - - - - - - - - - -
**
**  Equation of the equinoxes, IAU 1994 model.
**
**  Given:
**     date1,date2   double     TDB date (Note 1)
**
**  Returned (function value):
**                   double     equation of the equinoxes (Note 2)
**
**  Notes:
**
**  1) The date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The result, which is in radians, operates in the following sense:
**
**        Greenwich apparent ST = GMST + equation of the equinoxes
**
**  Called:
**     eraAnpm      normalize angle into range +/- pi
**     eraNut80     nutation, IAU 1980
**     eraObl80     mean obliquity, IAU 1980
**
**  References:
**
**     IAU Resolution C7, Recommendation 3 (1994).
**
**     Capitaine, N. & Gontier, A.-M., 1993, Astron.Astrophys., 275,
**     645-650.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraEra00(double dj1, double dj2)
/*
**  - - - - - - - - - -
**   e r a E r a 0 0
**  - - - - - - - - - -
**
**  Earth rotation angle (IAU 2000 model).
**
**  Given:
**     dj1,dj2   double    UT1 as a 2-part Julian Date (see note)
**
**  Returned (function value):
**               double    Earth rotation angle (radians), range 0-2pi
**
**  Notes:
**
**  1) The UT1 date dj1+dj2 is a Julian Date, apportioned in any
**     convenient way between the arguments dj1 and dj2.  For example,
**     JD(UT1)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**             dj1            dj2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  The date & time method is
**     best matched to the algorithm used:  maximum precision is
**     delivered when the dj1 argument is for 0hrs UT1 on the day in
**     question and the dj2 argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) The algorithm is adapted from Expression 22 of Capitaine et al.
**     2000.  The time argument has been expressed in days directly,
**     and, to retain precision, integer contributions have been
**     eliminated.  The same formulation is given in IERS Conventions
**     (2003), Chap. 5, Eq. 14.
**
**  Called:
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Capitaine N., Guinot B. and McCarthy D.D, 2000, Astron.
**     Astrophys., 355, 398-405.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGmst00(double uta, double utb, double tta, double ttb)
/*
**  - - - - - - - - - - -
**   e r a G m s t 0 0
**  - - - - - - - - - - -
**
**  Greenwich mean sidereal time (model consistent with IAU 2000
**  resolutions).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**     tta,ttb    double    TT as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich mean sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 and TT dates uta+utb and tta+ttb respectively, are both
**     Julian Dates, apportioned in any convenient way between the
**     argument pairs.  For example, JD=2450123.7 could be expressed in
**     any of these ways, among others:
**
**            Part A         Part B
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable (in the case of UT;  the TT is not at all critical
**     in this respect).  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     Rotation Angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) Both UT1 and TT are required, UT1 to predict the Earth rotation
**     and TT to predict the effects of precession.  If UT1 is used for
**     both purposes, errors of order 100 microarcseconds result.
**
**  3) This GMST is compatible with the IAU 2000 resolutions and must be
**     used only in conjunction with other IAU 2000 compatible
**     components such as precession-nutation and equation of the
**     equinoxes.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  5) The algorithm is from Capitaine et al. (2003) and IERS
**     Conventions 2003.
**
**  Called:
**     eraEra00     Earth rotation angle, IAU 2000
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
```

```
*/
```

```
double eraGmst06(double uta, double utb, double tta, double ttb)
/*
**  - - - - - - - - - -
**   e r a G m s t 0 6
**  - - - - - - - - - -
**
**  Greenwich mean sidereal time (consistent with IAU 2006 precession).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**     tta,ttb    double    TT as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich mean sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 and TT dates uta+utb and tta+ttb respectively, are both
**     Julian Dates, apportioned in any convenient way between the
**     argument pairs.  For example, JD=2450123.7 could be expressed in
**     any of these ways, among others:
**
**            Part A        Part B
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable (in the case of UT;  the TT is not at all critical
**     in this respect).  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     rotation angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) Both UT1 and TT are required, UT1 to predict the Earth rotation
**     and TT to predict the effects of precession.  If UT1 is used for
**     both purposes, errors of order 100 microarcseconds result.
**
**  3) This GMST is compatible with the IAU 2006 precession and must not
**     be used with other precession models.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  Called:
**     eraEra00     Earth rotation angle, IAU 2000
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Reference:
**
**     Capitaine, N., Wallace, P.T. & Chapront, J., 2005,
**     Astron.Astrophys. 432, 355
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGmst82(double dj1, double dj2)
/*
**  - - - - - - - - - -
**   e r a G m s t 8 2
**  - - - - - - - - - -
**
**  Universal Time to Greenwich mean sidereal time (IAU 1982 model).
**
**  Given:
**     dj1,dj2    double    UT1 Julian Date (see note)
**
**  Returned (function value):
**                double    Greenwich mean sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 date dj1+dj2 is a Julian Date, apportioned in any
**     convenient way between the arguments dj1 and dj2.  For example,
**     JD(UT1)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**             dj1            dj2
**
**         2450123.7          0          (JD method)
**          2451545        -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5         0.2         (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  The date & time method is
**     best matched to the algorithm used:  maximum accuracy (or, at
**     least, minimum noise) is delivered when the dj1 argument is for
**     0hrs UT1 on the day in question and the dj2 argument lies in the
**     range 0 to 1, or vice versa.
**
**  2) The algorithm is based on the IAU 1982 expression.  This is
**     always described as giving the GMST at 0 hours UT1.  In fact, it
**     gives the difference between the GMST and the UT, the steady
**     4-minutes-per-day drawing-ahead of ST with respect to UT.  When
**     whole days are ignored, the expression happens to equal the GMST
**     at 0 hours UT1 each day.
**
**  3) In this function, the entire UT1 (the sum of the two arguments
**     dj1 and dj2) is used directly as the argument for the standard
**     formula, the constant term of which is adjusted by 12 hours to
**     take account of the noon phasing of Julian Date.  The UT1 is then
**     added, but omitting whole days to conserve accuracy.
**
**  Called:
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Transactions of the International Astronomical Union,
**     XVIII B, 67 (1983).
**
**     Aoki et al., Astron.Astrophys., 105, 359-361 (1982).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGst00a(double uta, double utb, double tta, double ttb)
/*
**  - - - - - - - - - -
**   e r a G s t 0 0 a
**  - - - - - - - - - -
**
**  Greenwich apparent sidereal time (consistent with IAU 2000
**  resolutions).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**     tta,ttb    double    TT as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich apparent sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 and TT dates uta+utb and tta+ttb respectively, are both
**     Julian Dates, apportioned in any convenient way between the
**     argument pairs.  For example, JD=2450123.7 could be expressed in
**     any of these ways, among others:
**
**            Part A        Part B
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable (in the case of UT;  the TT is not at all critical
**     in this respect).  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     Rotation Angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) Both UT1 and TT are required, UT1 to predict the Earth rotation
**     and TT to predict the effects of precession-nutation.  If UT1 is
**     used for both purposes, errors of order 100 microarcseconds
**     result.
**
**  3) This GAST is compatible with the IAU 2000 resolutions and must be
**     used only in conjunction with other IAU 2000 compatible
**     components such as precession-nutation.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  5) The algorithm is from Capitaine et al. (2003) and IERS
**     Conventions 2003.
**
**  Called:
**     eraGmst00    Greenwich mean sidereal time, IAU 2000
**     eraEe00a     equation of the equinoxes, IAU 2000A
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
```

```
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGst00b(double uta, double utb)
/*
**  - - - - - - - - - -
**   e r a G s t 0 0 b
**  - - - - - - - - - -
**
**  Greenwich apparent sidereal time (consistent with IAU 2000
**  resolutions but using the truncated nutation model IAU 2000B).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich apparent sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 date uta+utb is a Julian Date, apportioned in any
**     convenient way between the argument pair.  For example,
**     JD=2450123.7 could be expressed in any of these ways, among
**     others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     Rotation Angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) The result is compatible with the IAU 2000 resolutions, except
**     that accuracy has been compromised for the sake of speed and
**     convenience in two respects:
**
**     . UT is used instead of TDB (or TT) to compute the precession
**       component of GMST and the equation of the equinoxes.  This
**       results in errors of order 0.1 mas at present.
**
**     . The IAU 2000B abridged nutation model (McCarthy & Luzum, 2001)
**       is used, introducing errors of up to 1 mas.
**
**  3) This GAST is compatible with the IAU 2000 resolutions and must be
**     used only in conjunction with other IAU 2000 compatible
**     components such as precession-nutation.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  5) The algorithm is from Capitaine et al. (2003) and IERS
**     Conventions 2003.
**
**  Called:
**     eraGmst00    Greenwich mean sidereal time, IAU 2000
**     eraEe00b     equation of the equinoxes, IAU 2000B
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Capitaine, N., Wallace, P.T. and McCarthy, D.D., "Expressions to
**     implement the IAU 2000 definition of UT1", Astronomy &
**     Astrophysics, 406, 1135-1149 (2003)
**
```

```
**     McCarthy, D.D. & Luzum, B.J., "An abridged model of the
**     precession-nutation of the celestial pole", Celestial Mechanics &
**     Dynamical Astronomy, 85, 37-49 (2003)
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGst06(double uta, double utb, double tta, double ttb,
                double rnpb[3][3])
/*
**  - - - - - - - - -
**   e r a G s t 0 6
**  - - - - - - - - -
**
**  Greenwich apparent sidereal time, IAU 2006, given the NPB matrix.
**
**  Given:
**     uta,utb  double        UT1 as a 2-part Julian Date (Notes 1,2)
**     tta,ttb  double        TT as a 2-part Julian Date (Notes 1,2)
**     rnpb     double[3][3]  nutation x precession x bias matrix
**
**  Returned (function value):
**              double        Greenwich apparent sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 and TT dates uta+utb and tta+ttb respectively, are both
**     Julian Dates, apportioned in any convenient way between the
**     argument pairs.  For example, JD=2450123.7 could be expressed in
**     any of these ways, among others:
**
**            Part A        Part B
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable (in the case of UT;  the TT is not at all critical
**     in this respect).  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     rotation angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) Both UT1 and TT are required, UT1 to predict the Earth rotation
**     and TT to predict the effects of precession-nutation.  If UT1 is
**     used for both purposes, errors of order 100 microarcseconds
**     result.
**
**  3) Although the function uses the IAU 2006 series for s+XY/2, it is
**     otherwise independent of the precession-nutation model and can in
**     practice be used with any equinox-based NPB matrix.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  Called:
**     eraBpn2xy    extract CIP X,Y coordinates from NPB matrix
**     eraS06       the CIO locator s, given X,Y, IAU 2006
**     eraAnp       normalize angle into range 0 to 2pi
**     eraEra00     Earth rotation angle, IAU 2000
**     eraEors      equation of the origins, given NPB matrix and s
**
**  Reference:
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```c
double eraGst06a(double uta, double utb, double tta, double ttb)
/*
**  - - - - - - - - - -
**   e r a G s t 0 6 a
**  - - - - - - - - - -
**
**  Greenwich apparent sidereal time (consistent with IAU 2000 and 2006
**  resolutions).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**     tta,ttb    double    TT as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich apparent sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 and TT dates uta+utb and tta+ttb respectively, are both
**     Julian Dates, apportioned in any convenient way between the
**     argument pairs.  For example, JD=2450123.7 could be expressed in
**     any of these ways, among others:
**
**            Part A        Part B
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable (in the case of UT;  the TT is not at all critical
**     in this respect).  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     rotation angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) Both UT1 and TT are required, UT1 to predict the Earth rotation
**     and TT to predict the effects of precession-nutation.  If UT1 is
**     used for both purposes, errors of order 100 microarcseconds
**     result.
**
**  3) This GAST is compatible with the IAU 2000/2006 resolutions and
**     must be used only in conjunction with IAU 2006 precession and
**     IAU 2000A nutation.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  Called:
**     eraPnm06a    classical NPB matrix, IAU 2006/2000A
**     eraGst06     Greenwich apparent ST, IAU 2006, given NPB matrix
**
**  Reference:
**
**     Wallace, P.T. & Capitaine, N., 2006, Astron.Astrophys. 459, 981
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraGst94(double uta, double utb)
/*
**  - - - - - - - - -
**   e r a G s t 9 4
**  - - - - - - - - -
**
**  Greenwich apparent sidereal time (consistent with IAU 1982/94
**  resolutions).
**
**  Given:
**     uta,utb    double    UT1 as a 2-part Julian Date (Notes 1,2)
**
**  Returned (function value):
**                double    Greenwich apparent sidereal time (radians)
**
**  Notes:
**
**  1) The UT1 date uta+utb is a Julian Date, apportioned in any
**     convenient way between the argument pair.  For example,
**     JD=2450123.7 could be expressed in any of these ways, among
**     others:
**
**             uta            utb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in cases
**     where the loss of several decimal digits of resolution is
**     acceptable.  The J2000 and MJD methods are good compromises
**     between resolution and convenience.  For UT, the date & time
**     method is best matched to the algorithm that is used by the Earth
**     Rotation Angle function, called internally:  maximum precision is
**     delivered when the uta argument is for 0hrs UT1 on the day in
**     question and the utb argument lies in the range 0 to 1, or vice
**     versa.
**
**  2) The result is compatible with the IAU 1982 and 1994 resolutions,
**     except that accuracy has been compromised for the sake of
**     convenience in that UT is used instead of TDB (or TT) to compute
**     the equation of the equinoxes.
**
**  3) This GAST must be used only in conjunction with contemporaneous
**     IAU standards such as 1976 precession, 1980 obliquity and 1982
**     nutation.  It is not compatible with the IAU 2000 resolutions.
**
**  4) The result is returned in the range 0 to 2pi.
**
**  Called:
**     eraGmst82    Greenwich mean sidereal time, IAU 1982
**     eraEqeq94    equation of the equinoxes, IAU 1994
**     eraAnp       normalize angle into range 0 to 2pi
**
**  References:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**     IAU Resolution C7, Recommendation 3 (1994)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraPvstar(double pv[2][3], double *ra, double *dec,
              double *pmr, double *pmd, double *px, double *rv)
/*
**  - - - - - - - - - -
**   e r a P v s t a r
**  - - - - - - - - - -
**
**  Convert star position+velocity vector to catalog coordinates.
**
**  Given (Note 1):
**     pv     double[2][3]   pv-vector (au, au/day)
**
**  Returned (Note 2):
**     ra     double         right ascension (radians)
**     dec    double         declination (radians)
**     pmr    double         RA proper motion (radians/year)
**     pmd    double         Dec proper motion (radians/year)
**     px     double         parallax (arcsec)
**     rv     double         radial velocity (km/s, positive = receding)
**
**  Returned (function value):
**            int            status:
**                              0 = OK
**                             -1 = superluminal speed (Note 5)
**                             -2 = null position vector
**
**  Notes:
**
**  1) The specified pv-vector is the coordinate direction (and its rate
**     of change) for the date at which the light leaving the star
**     reached the solar-system barycenter.
**
**  2) The star data returned by this function are "observables" for an
**     imaginary observer at the solar-system barycenter.  Proper motion
**     and radial velocity are, strictly, in terms of barycentric
**     coordinate time, TCB.  For most practical applications, it is
**     permissible to neglect the distinction between TCB and ordinary
**     "proper" time on Earth (TT/TAI).  The result will, as a rule, be
**     limited by the intrinsic accuracy of the proper-motion and
**     radial-velocity data;  moreover, the supplied pv-vector is likely
**     to be merely an intermediate result (for example generated by the
**     function eraStarpv), so that a change of time unit will cancel
**     out overall.
**
**     In accordance with normal star-catalog conventions, the object’s
**     right ascension and declination are freed from the effects of
**     secular aberration.  The frame, which is aligned to the catalog
**     equator and equinox, is Lorentzian and centered on the SSB.
**
**     Summarizing, the specified pv-vector is for most stars almost
**     identical to the result of applying the standard geometrical
**     "space motion" transformation to the catalog data.  The
**     differences, which are the subject of the Stumpff paper cited
**     below, are:
**
**     (i) In stars with significant radial velocity and proper motion,
**     the constantly changing light-time distorts the apparent proper
**     motion.  Note that this is a classical, not a relativistic,
**     effect.
**
**     (ii) The transformation complies with special relativity.
**
**  3) Care is needed with units.  The star coordinates are in radians
**     and the proper motions in radians per Julian year, but the
**     parallax is in arcseconds; the radial velocity is in km/s, but
**     the pv-vector result is in au and au/day.
**
**  4) The proper motions are the rate of change of the right ascension
**     and declination at the catalog epoch and are in radians per Julian
**     year.  The RA proper motion is in terms of coordinate angle, not
```

```
**     true angle, and will thus be numerically larger at high
**     declinations.
**
**  5) Straight-line motion at constant speed in the inertial frame is
**     assumed.  If the speed is greater than or equal to the speed of
**     light, the function aborts with an error status.
**
**  6) The inverse transformation is performed by the function eraStarpv.
**
**  Called:
**     eraPn        decompose p-vector into modulus and direction
**     eraPdp       scalar product of two p-vectors
**     eraSxp       multiply p-vector by scalar
**     eraPmp       p-vector minus p-vector
**     eraPm        modulus of p-vector
**     eraPpp       p-vector plus p-vector
**     eraPv2s      pv-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Reference:
**
**     Stumpff, P., 1985, Astron.Astrophys. 144, 232-240.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraStarpv(double ra, double dec,
              double pmr, double pmd, double px, double rv,
              double pv[2][3])
/*
**  - - - - - - - - - - -
**   e r a S t a r p v
**  - - - - - - - - - - -
**
**  Convert star catalog coordinates to position+velocity vector.
**
**  Given (Note 1):
**     ra     double        right ascension (radians)
**     dec    double        declination (radians)
**     pmr    double        RA proper motion (radians/year)
**     pmd    double        Dec proper motion (radians/year)
**     px     double        parallax (arcseconds)
**     rv     double        radial velocity (km/s, positive = receding)
**
**  Returned (Note 2):
**     pv     double[2][3]  pv-vector (au, au/day)
**
**  Returned (function value):
**            int           status:
**                              0 = no warnings
**                              1 = distance overridden (Note 6)
**                              2 = excessive speed (Note 7)
**                              4 = solution didn’t converge (Note 8)
**                           else = binary logical OR of the above
**
**  Notes:
**
**  1) The star data accepted by this function are "observables" for an
**     imaginary observer at the solar-system barycenter.  Proper motion
**     and radial velocity are, strictly, in terms of barycentric
**     coordinate time, TCB.  For most practical applications, it is
**     permissible to neglect the distinction between TCB and ordinary
**     "proper" time on Earth (TT/TAI).  The result will, as a rule, be
**     limited by the intrinsic accuracy of the proper-motion and
**     radial-velocity data;  moreover, the pv-vector is likely to be
**     merely an intermediate result, so that a change of time unit
**     would cancel out overall.
**
**     In accordance with normal star-catalog conventions, the object’s
**     right ascension and declination are freed from the effects of
**     secular aberration.  The frame, which is aligned to the catalog
**     equator and equinox, is Lorentzian and centered on the SSB.
**
**  2) The resulting position and velocity pv-vector is with respect to
**     the same frame and, like the catalog coordinates, is freed from
**     the effects of secular aberration.  Should the "coordinate
**     direction", where the object was located at the catalog epoch, be
**     required, it may be obtained by calculating the magnitude of the
**     position vector pv[0][0-2] dividing by the speed of light in
**     au/day to give the light-time, and then multiplying the space
**     velocity pv[1][0-2] by this light-time and adding the result to
**     pv[0][0-2].
**
**     Summarizing, the pv-vector returned is for most stars almost
**     identical to the result of applying the standard geometrical
**     "space motion" transformation.  The differences, which are the
**     subject of the Stumpff paper referenced below, are:
**
**     (i) In stars with significant radial velocity and proper motion,
**     the constantly changing light-time distorts the apparent proper
**     motion.  Note that this is a classical, not a relativistic,
**     effect.
**
**     (ii) The transformation complies with special relativity.
**
**  3) Care is needed with units.  The star coordinates are in radians
```

```
**     and the proper motions in radians per Julian year, but the
**     parallax is in arcseconds; the radial velocity is in km/s, but
**     the pv-vector result is in au and au/day.
**
**  4) The RA proper motion is in terms of coordinate angle, not true
**     angle.  If the catalog uses arcseconds for both RA and Dec proper
**     motions, the RA proper motion will need to be divided by cos(Dec)
**     before use.
**
**  5) Straight-line motion at constant speed, in the inertial frame,
**     is assumed.
**
**  6) An extremely small (or zero or negative) parallax is interpreted
**     to mean that the object is on the "celestial sphere", the radius
**     of which is an arbitrary (large) value (see the constant PXMIN).
**     When the distance is overridden in this way, the status,
**     initially zero, has 1 added to it.
**
**  7) If the space velocity is a significant fraction of c (see the
**     constant VMAX), it is arbitrarily set to zero.  When this action
**     occurs, 2 is added to the status.
**
**  8) The relativistic adjustment involves an iterative calculation.
**     If the process fails to converge within a set number (IMAX) of
**     iterations, 4 is added to the status.
**
**  9) The inverse transformation is performed by the function
**     eraPvstar.
**
**  Called:
**     eraS2pv      spherical coordinates to pv-vector
**     eraPm        modulus of p-vector
**     eraZp        zero p-vector
**     eraPn        decompose p-vector into modulus and direction
**     eraPdp       scalar product of two p-vectors
**     eraSxp       multiply p-vector by scalar
**     eraPmp       p-vector minus p-vector
**     eraPpp       p-vector plus p-vector
**
**  Reference:
**
**     Stumpff, P., 1985, Astron.Astrophys. 144, 232-240.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraFk52h(double r5, double d5,
              double dr5, double dd5, double px5, double rv5,
              double *rh, double *dh,
              double *drh, double *ddh, double *pxh, double *rvh)
/*
**  - - - - - - - - -
**   e r a F k 5 2 h
**  - - - - - - - - -
**
**  Transform FK5 (J2000.0) star data into the Hipparcos system.
**
**  Given (all FK5, equinox J2000.0, epoch J2000.0):
**     r5      double    RA (radians)
**     d5      double    Dec (radians)
**     dr5     double    proper motion in RA (dRA/dt, rad/Jyear)
**     dd5     double    proper motion in Dec (dDec/dt, rad/Jyear)
**     px5     double    parallax (arcsec)
**     rv5     double    radial velocity (km/s, positive = receding)
**
**  Returned (all Hipparcos, epoch J2000.0):
**     rh      double    RA (radians)
**     dh      double    Dec (radians)
**     drh     double    proper motion in RA (dRA/dt, rad/Jyear)
**     ddh     double    proper motion in Dec (dDec/dt, rad/Jyear)
**     pxh     double    parallax (arcsec)
**     rvh     double    radial velocity (km/s, positive = receding)
**
**  Notes:
**
**  1) This function transforms FK5 star positions and proper motions
**     into the system of the Hipparcos catalog.
**
**  2) The proper motions in RA are dRA/dt rather than
**     cos(Dec)*dRA/dt, and are per year rather than per century.
**
**  3) The FK5 to Hipparcos transformation is modeled as a pure
**     rotation and spin;  zonal errors in the FK5 catalog are not
**     taken into account.
**
**  4) See also eraH2fk5, eraFk5hz, eraHfk5z.
**
**  Called:
**     eraStarpv    star catalog data to space motion pv-vector
**     eraFk5hip    FK5 to Hipparcos rotation and spin
**     eraRxp       product of r-matrix and p-vector
**     eraPxp       vector product of two p-vectors
**     eraPpp       p-vector plus p-vector
**     eraPvstar    space motion pv-vector to star catalog data
**
**  Reference:
**
**     F.Mignard & M.Froeschle, Astron.Astrophys., 354, 732-739 (2000).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraFk5hip(double r5h[3][3], double s5h[3])
/*
**  - - - - - - - - - -
**   e r a F k 5 h i p
**  - - - - - - - - - -
**
**  FK5 to Hipparcos rotation and spin.
**
**  Returned:
**     r5h   double[3][3]  r-matrix: FK5 rotation wrt Hipparcos (Note 2)
**     s5h   double[3]     r-vector: FK5 spin wrt Hipparcos (Note 3)
**
**  Notes:
**
**  1) This function models the FK5 to Hipparcos transformation as a
**     pure rotation and spin;  zonal errors in the FK5 catalogue are
**     not taken into account.
**
**  2) The r-matrix r5h operates in the sense:
**
**           P_Hipparcos = r5h x P_FK5
**
**     where P_FK5 is a p-vector in the FK5 frame, and P_Hipparcos is
**     the equivalent Hipparcos p-vector.
**
**  3) The r-vector s5h represents the time derivative of the FK5 to
**     Hipparcos rotation.  The units are radians per year (Julian,
**     TDB).
**
**  Called:
**     eraRv2m      r-vector to r-matrix
**
**  Reference:
**
**     F.Mignard & M.Froeschle, Astron.Astrophys., 354, 732-739 (2000).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraFk5hz(double r5, double d5, double date1, double date2,
              double *rh, double *dh)
/*
**  - - - - - - - - -
**   e r a F k 5 h z
**  - - - - - - - - -
**
**  Transform an FK5 (J2000.0) star position into the system of the
**  Hipparcos catalogue, assuming zero Hipparcos proper motion.
**
**  Given:
**     r5           double   FK5 RA (radians), equinox J2000.0, at date
**     d5           double   FK5 Dec (radians), equinox J2000.0, at date
**     date1,date2  double   TDB date (Notes 1,2)
**
**  Returned:
**     rh           double   Hipparcos RA (radians)
**     dh           double   Hipparcos Dec (radians)
**
**  Notes:
**
**  1) This function converts a star position from the FK5 system to
**     the Hipparcos system, in such a way that the Hipparcos proper
**     motion is zero.  Because such a star has, in general, a non-zero
**     proper motion in the FK5 system, the function requires the date
**     at which the position in the FK5 system was determined.
**
**  2) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  3) The FK5 to Hipparcos transformation is modeled as a pure
**     rotation and spin;  zonal errors in the FK5 catalogue are not
**     taken into account.
**
**  4) The position returned by this function is in the Hipparcos
**     reference system but at date date1+date2.
**
**  5) See also eraFk52h, eraH2fk5, eraHfk5z.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraFk5hip    FK5 to Hipparcos rotation and spin
**     eraSxp       multiply p-vector by scalar
**     eraRv2m      r-vector to r-matrix
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraPxp       vector product of two p-vectors
**     eraC2s       p-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
**  Reference:
**
**     F.Mignard & M.Froeschle, 2000, Astron.Astrophys. 354, 732-739.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
```

```
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraH2fk5(double rh, double dh,
              double drh, double ddh, double pxh, double rvh,
              double *r5, double *d5,
              double *dr5, double *dd5, double *px5, double *rv5)
/*
**  - - - - - - - - -
**   e r a H 2 f k 5
**  - - - - - - - - -
**
**  Transform Hipparcos star data into the FK5 (J2000.0) system.
**
**  Given (all Hipparcos, epoch J2000.0):
**     rh      double    RA (radians)
**     dh      double    Dec (radians)
**     drh     double    proper motion in RA (dRA/dt, rad/Jyear)
**     ddh     double    proper motion in Dec (dDec/dt, rad/Jyear)
**     pxh     double    parallax (arcsec)
**     rvh     double    radial velocity (km/s, positive = receding)
**
**  Returned (all FK5, equinox J2000.0, epoch J2000.0):
**     r5      double    RA (radians)
**     d5      double    Dec (radians)
**     dr5     double    proper motion in RA (dRA/dt, rad/Jyear)
**     dd5     double    proper motion in Dec (dDec/dt, rad/Jyear)
**     px5     double    parallax (arcsec)
**     rv5     double    radial velocity (km/s, positive = receding)
**
**  Notes:
**
**  1) This function transforms Hipparcos star positions and proper
**     motions into FK5 J2000.0.
**
**  2) The proper motions in RA are dRA/dt rather than
**     cos(Dec)*dRA/dt, and are per year rather than per century.
**
**  3) The FK5 to Hipparcos transformation is modeled as a pure
**     rotation and spin;  zonal errors in the FK5 catalog are not
**     taken into account.
**
**  4) See also eraFk52h, eraFk5hz, eraHfk5z.
**
**  Called:
**     eraStarpv    star catalog data to space motion pv-vector
**     eraFk5hip    FK5 to Hipparcos rotation and spin
**     eraRv2m      r-vector to r-matrix
**     eraRxp       product of r-matrix and p-vector
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraPxp       vector product of two p-vectors
**     eraPmp       p-vector minus p-vector
**     eraPvstar    space motion pv-vector to star catalog data
**
**  Reference:
**
**     F.Mignard & M.Froeschle, Astron.Astrophys., 354, 732-739 (2000).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraHfk5z(double rh, double dh, double date1, double date2,
              double *r5, double *d5, double *dr5, double *dd5)
/*
**  - - - - - - - - -
**   e r a H f k 5 z
**  - - - - - - - - -
**
**  Transform a Hipparcos star position into FK5 J2000.0, assuming
**  zero Hipparcos proper motion.
**
**  Given:
**     rh            double    Hipparcos RA (radians)
**     dh            double    Hipparcos Dec (radians)
**     date1,date2   double    TDB date (Note 1)
**
**  Returned (all FK5, equinox J2000.0, date date1+date2):
**     r5            double    RA (radians)
**     d5            double    Dec (radians)
**     dr5           double    FK5 RA proper motion (rad/year, Note 4)
**     dd5           double    Dec proper motion (rad/year, Note 4)
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) The proper motion in RA is dRA/dt rather than cos(Dec)*dRA/dt.
**
**  3) The FK5 to Hipparcos transformation is modeled as a pure rotation
**     and spin;  zonal errors in the FK5 catalogue are not taken into
**     account.
**
**  4) It was the intention that Hipparcos should be a close
**     approximation to an inertial frame, so that distant objects have
**     zero proper motion;  such objects have (in general) non-zero
**     proper motion in FK5, and this function returns those fictitious
**     proper motions.
**
**  5) The position returned by this function is in the FK5 J2000.0
**     reference system but at date date1+date2.
**
**  6) See also eraFk52h, eraH2fk5, eraFk5zhz.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraFk5hip    FK5 to Hipparcos rotation and spin
**     eraRxp       product of r-matrix and p-vector
**     eraSxp       multiply p-vector by scalar
**     eraRxr       product of two r-matrices
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraPxp       vector product of two p-vectors
**     eraPv2s      pv-vector to spherical
**     eraAnp       normalize angle into range 0 to 2pi
**
```

```
**  Reference:
**
**     F.Mignard & M.Froeschle, 2000, Astron.Astrophys. 354, 732-739.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraStarpm(double ra1, double dec1,
              double pmr1, double pmd1, double px1, double rv1,
              double ep1a, double ep1b, double ep2a, double ep2b,
              double *ra2, double *dec2,
              double *pmr2, double *pmd2, double *px2, double *rv2)
/*
**  - - - - - - - - - - -
**   e r a S t a r p m
**  - - - - - - - - - - -
**
**  Star proper motion:  update star catalog data for space motion.
**
**  Given:
**     ra1    double     right ascension (radians), before
**     dec1   double     declination (radians), before
**     pmr1   double     RA proper motion (radians/year), before
**     pmd1   double     Dec proper motion (radians/year), before
**     px1    double     parallax (arcseconds), before
**     rv1    double     radial velocity (km/s, +ve = receding), before
**     ep1a   double     "before" epoch, part A (Note 1)
**     ep1b   double     "before" epoch, part B (Note 1)
**     ep2a   double     "after" epoch, part A (Note 1)
**     ep2b   double     "after" epoch, part B (Note 1)
**
**  Returned:
**     ra2    double     right ascension (radians), after
**     dec2   double     declination (radians), after
**     pmr2   double     RA proper motion (radians/year), after
**     pmd2   double     Dec proper motion (radians/year), after
**     px2    double     parallax (arcseconds), after
**     rv2    double     radial velocity (km/s, +ve = receding), after
**
**  Returned (function value):
**            int        status:
**                          -1 = system error (should not occur)
**                           0 = no warnings or errors
**                           1 = distance overridden (Note 6)
**                           2 = excessive velocity (Note 7)
**                           4 = solution didn't converge (Note 8)
**                        else = binary logical OR of the above warnings
**
**  Notes:
**
**  1) The starting and ending TDB dates ep1a+ep1b and ep2a+ep2b are
**     Julian Dates, apportioned in any convenient way between the two
**     parts (A and B).  For example, JD(TDB)=2450123.7 could be
**     expressed in any of these ways, among others:
**
**             epna          epnb
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) In accordance with normal star-catalog conventions, the object's
**     right ascension and declination are freed from the effects of
**     secular aberration.  The frame, which is aligned to the catalog
**     equator and equinox, is Lorentzian and centered on the SSB.
**
**     The proper motions are the rate of change of the right ascension
**     and declination at the catalog epoch and are in radians per TDB
**     Julian year.
```

```
**
**     The parallax and radial velocity are in the same frame.
**
**  3) Care is needed with units.  The star coordinates are in radians
**     and the proper motions in radians per Julian year, but the
**     parallax is in arcseconds.
**
**  4) The RA proper motion is in terms of coordinate angle, not true
**     angle.  If the catalog uses arcseconds for both RA and Dec proper
**     motions, the RA proper motion will need to be divided by cos(Dec)
**     before use.
**
**  5) Straight-line motion at constant speed, in the inertial frame,
**     is assumed.
**
**  6) An extremely small (or zero or negative) parallax is interpreted
**     to mean that the object is on the "celestial sphere", the radius
**     of which is an arbitrary (large) value (see the eraStarpv
**     function for the value used).  When the distance is overridden in
**     this way, the status, initially zero, has 1 added to it.
**
**  7) If the space velocity is a significant fraction of c (see the
**     constant VMAX in the function eraStarpv), it is arbitrarily set
**     to zero.  When this action occurs, 2 is added to the status.
**
**  8) The relativistic adjustment carried out in the eraStarpv function
**     involves an iterative calculation.  If the process fails to
**     converge within a set number of iterations, 4 is added to the
**     status.
**
**  Called:
**     eraStarpv    star catalog data to space motion pv-vector
**     eraPvu       update a pv-vector
**     eraPdp       scalar product of two p-vectors
**     eraPvstar    space motion pv-vector to star catalog data
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraEceq06(double date1, double date2, double dl, double db,
               double *dr, double *dd)
/*
**  - - - - - - - - - - -
**   e r a E c e q 0 6
**  - - - - - - - - - - -
**
**  Transformation from ecliptic coordinates (mean equinox and ecliptic
**  of date) to ICRS RA,Dec, using the IAU 2006 precession model.
**
**  Given:
**     date1,date2 double TT as a 2-part Julian date (Note 1)
**     dl,db       double ecliptic longitude and latitude (radians)
**
**  Returned:
**     dr,dd       double ICRS right ascension and declination (radians)
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) No assumptions are made about whether the coordinates represent
**     starlight and embody astrometric effects such as parallax or
**     aberration.
**
**  3) The transformation is approximately that from ecliptic longitude
**     and latitude (mean equinox and ecliptic of date) to mean J2000.0
**     right ascension and declination, with only frame bias (always
**     less than 25 mas) to disturb this classical picture.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraEcm06     J2000.0 to ecliptic rotation matrix, IAU 2006
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraC2s       unit vector to spherical coordinates
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraEcm06(double date1, double date2, double rm[3][3])
/*
**  - - - - - - - - -
**   e r a E c m 0 6
**  - - - - - - - - -
**
**  ICRS equatorial to ecliptic rotation matrix, IAU 2006.
**
**  Given:
**     date1,date2  double         TT as a 2-part Julian date (Note 1)
**
**  Returned:
**     rm           double[3][3]   ICRS to ecliptic rotation matrix
**
**  Notes:
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  1) The matrix is in the sense
**
**        E_ep = rm x P_ICRS,
**
**     where P_ICRS is a vector with respect to ICRS right ascension
**     and declination axes and E_ep is the same vector with respect to
**     the (inertial) ecliptic and equinox of date.
**
**  2) P_ICRS is a free vector, merely a direction, typically of unit
**     magnitude, and not bound to any particular spatial origin, such
**     as the Earth, Sun or SSB.  No assumptions are made about whether
**     it represents starlight and embodies astrometric effects such as
**     parallax or aberration.  The transformation is approximately that
**     between mean J2000.0 right ascension and declination and ecliptic
**     longitude and latitude, with only frame bias (always less than
**     25 mas) to disturb this classical picture.
**
**  Called:
**     eraObl06     mean obliquity, IAU 2006
**     eraPmat06    PB matrix, IAU 2006
**     eraIr        initialize r-matrix to identity
**     eraRx        rotate around X-axis
**     eraRxr       product of two r-matrices
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraEqec06(double date1, double date2, double dr, double dd,
               double *dl, double *db)
/*
**  - - - - - - - - - -
**   e r a E q e c 0 6
**  - - - - - - - - - -
**
**  Transformation from ICRS equatorial coordinates to ecliptic
**  coordinates (mean equinox and ecliptic of date) using IAU 2006
**  precession model.
**
**  Given:
**     date1,date2 double TT as a 2-part Julian date (Note 1)
**     dr,dd       double ICRS right ascension and declination (radians)
**
**  Returned:
**     dl,db       double ecliptic longitude and latitude (radians)
**
**  1) The TT date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
**
**  2) No assumptions are made about whether the coordinates represent
**     starlight and embody astrometric effects such as parallax or
**     aberration.
**
**  3) The transformation is approximately that from mean J2000.0 right
**     ascension and declination to ecliptic longitude and latitude
**     (mean equinox and ecliptic of date), with only frame bias (always
**     less than 25 mas) to disturb this classical picture.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraEcm06     J2000.0 to ecliptic rotation matrix, IAU 2006
**     eraRxp       product of r-matrix and p-vector
**     eraC2s       unit vector to spherical coordinates
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLteceq(double epj, double dl, double db, double *dr, double *dd)
/*
**  - - - - - - - - - -
**   e r a L t e c e q
**  - - - - - - - - - -
**
**  Transformation from ecliptic coordinates (mean equinox and ecliptic
**  of date) to ICRS RA,Dec, using a long-term precession model.
**
**  Given:
**     epj     double     Julian epoch (TT)
**     dl,db   double     ecliptic longitude and latitude (radians)
**
**  Returned:
**     dr,dd   double     ICRS right ascension and declination (radians)
**
**  1) No assumptions are made about whether the coordinates represent
**     starlight and embody astrometric effects such as parallax or
**     aberration.
**
**  2) The transformation is approximately that from ecliptic longitude
**     and latitude (mean equinox and ecliptic of date) to mean J2000.0
**     right ascension and declination, with only frame bias (always
**     less than 25 mas) to disturb this classical picture.
**
**  3) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraLtecm     J2000.0 to ecliptic rotation matrix, long term
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraC2s       unit vector to spherical coordinates
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLtecm(double epj, double rm[3][3])
/*
**  - - - - - - - - -
**   e r a L t e c m
**  - - - - - - - - -
**
**  ICRS equatorial to ecliptic rotation matrix, long-term.
**
**  Given:
**     epj     double         Julian epoch (TT)
**
**  Returned:
**     rm      double[3][3]   ICRS to ecliptic rotation matrix
**
**  Notes:
**
**  1) The matrix is in the sense
**
**        E_ep = rm x P_ICRS,
**
**     where P_ICRS is a vector with respect to ICRS right ascension
**     and declination axes and E_ep is the same vector with respect to
**     the (inertial) ecliptic and equinox of epoch epj.
**
**  2) P_ICRS is a free vector, merely a direction, typically of unit
**     magnitude, and not bound to any particular spatial origin, such
**     as the Earth, Sun or SSB.  No assumptions are made about whether
**     it represents starlight and embodies astrometric effects such as
**     parallax or aberration.  The transformation is approximately that
**     between mean J2000.0 right ascension and declination and ecliptic
**     longitude and latitude, with only frame bias (always less than
**     25 mas) to disturb this classical picture.
**
**  3) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  Called:
**     eraLtpequ    equator pole, long term
**     eraLtpecl    ecliptic pole, long term
**     eraPxp       vector product
**     eraPn        normalize vector
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraLteqec(double epj, double dr, double dd, double *dl, double *db)
/*
**  - - - - - - - - - -
**   e r a L t e q e c
**  - - - - - - - - - -
**
**  Transformation from ICRS equatorial coordinates to ecliptic
**  coordinates (mean equinox and ecliptic of date) using a long-term
**  precession model.
**
**  Given:
**     epj     double     Julian epoch (TT)
**     dr,dd   double     ICRS right ascension and declination (radians)
**
**  Returned:
**     dl,db   double     ecliptic longitude and latitude (radians)
**
**  1) No assumptions are made about whether the coordinates represent
**     starlight and embody astrometric effects such as parallax or
**     aberration.
**
**  2) The transformation is approximately that from mean J2000.0 right
**     ascension and declination to ecliptic longitude and latitude
**     (mean equinox and ecliptic of date), with only frame bias (always
**     less than 25 mas) to disturb this classical picture.
**
**  3) The Vondrak et al. (2011, 2012) 400 millennia precession model
**     agrees with the IAU 2006 precession at J2000.0 and stays within
**     100 microarcseconds during the 20th and 21st centuries.  It is
**     accurate to a few arcseconds throughout the historical period,
**     worsening to a few tenths of a degree at the end of the
**     +/- 200,000 year time span.
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraLtecm     J2000.0 to ecliptic rotation matrix, long term
**     eraRxp       product of r-matrix and p-vector
**     eraC2s       unit vector to spherical coordinates
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**
**  References:
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2011, New precession
**    expressions, valid for long time intervals, Astron.Astrophys. 534,
**    A22
**
**    Vondrak, J., Capitaine, N. and Wallace, P., 2012, New precession
**    expressions, valid for long time intervals (Corrigendum),
**    Astron.Astrophys. 541, C1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraG2icrs ( double dl, double db, double *dr, double *dd )
/*
**  - - - - - - - - - -
**   e r a G 2 i c r s
**  - - - - - - - - - -
**
**  Transformation from Galactic Coordinates to ICRS.
**
**  Given:
**     dl     double      galactic longitude (radians)
**     db     double      galactic latitude (radians)
**
**  Returned:
**     dr     double      ICRS right ascension (radians)
**     dd     double      ICRS declination (radians)
**
**  Notes:
**
**  1) The IAU 1958 system of Galactic coordinates was defined with
**     respect to the now obsolete reference system FK4 B1950.0.  When
**     interpreting the system in a modern context, several factors have
**     to be taken into account:
**
**     . The inclusion in FK4 positions of the E-terms of aberration.
**
**     . The distortion of the FK4 proper motion system by differential
**       Galactic rotation.
**
**     . The use of the B1950.0 equinox rather than the now-standard
**       J2000.0.
**
**     . The frame bias between ICRS and the J2000.0 mean place system.
**
**     The Hipparcos Catalogue (Perryman & ESA 1997) provides a rotation
**     matrix that transforms directly between ICRS and Galactic
**     coordinates with the above factors taken into account.  The
**     matrix is derived from three angles, namely the ICRS coordinates
**     of the Galactic pole and the longitude of the ascending node of
**     the galactic equator on the ICRS equator.  They are given in
**     degrees to five decimal places and for canonical purposes are
**     regarded as exact.  In the Hipparcos Catalogue the matrix
**     elements are given to 10 decimal places (about 20 microarcsec).
**     In the present ERFA function the matrix elements have been
**     recomputed from the canonical three angles and are given to 30
**     decimal places.
**
**  2) The inverse transformation is performed by the function eraIcrs2g.
**
**  Called:
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**     eraS2c       spherical coordinates to unit vector
**     eraTrxp      product of transpose of r-matrix and p-vector
**     eraC2s       p-vector to spherical
**
**  Reference:
**     Perryman M.A.C. & ESA, 1997, ESA SP-1200, The Hipparcos and Tycho
**     catalogues.  Astrometric and photometric star catalogues
**     derived from the ESA Hipparcos Space Astrometry Mission.  ESA
**     Publications Division, Noordwijk, Netherlands.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraIcrs2g ( double dr, double dd, double *dl, double *db )
/*
**  - - - - - - - - - -
**   e r a I c r s 2 g
**  - - - - - - - - - -
**
**  Transformation from ICRS to Galactic Coordinates.
**
**  Given:
**     dr     double      ICRS right ascension (radians)
**     dd     double      ICRS declination (radians)
**
**  Returned:
**     dl     double      galactic longitude (radians)
**     db     double      galactic latitude (radians)
**
**  Notes:
**
**  1) The IAU 1958 system of Galactic coordinates was defined with
**     respect to the now obsolete reference system FK4 B1950.0.  When
**     interpreting the system in a modern context, several factors have
**     to be taken into account:
**
**     . The inclusion in FK4 positions of the E-terms of aberration.
**
**     . The distortion of the FK4 proper motion system by differential
**       Galactic rotation.
**
**     . The use of the B1950.0 equinox rather than the now-standard
**       J2000.0.
**
**     . The frame bias between ICRS and the J2000.0 mean place system.
**
**     The Hipparcos Catalogue (Perryman & ESA 1997) provides a rotation
**     matrix that transforms directly between ICRS and Galactic
**     coordinates with the above factors taken into account.  The
**     matrix is derived from three angles, namely the ICRS coordinates
**     of the Galactic pole and the longitude of the ascending node of
**     the galactic equator on the ICRS equator.  They are given in
**     degrees to five decimal places and for canonical purposes are
**     regarded as exact.  In the Hipparcos Catalogue the matrix
**     elements are given to 10 decimal places (about 20 microarcsec).
**     In the present ERFA function the matrix elements have been
**     recomputed from the canonical three angles and are given to 30
**     decimal places.
**
**  2) The inverse transformation is performed by the function eraG2icrs.
**
**  Called:
**     eraAnp       normalize angle into range 0 to 2pi
**     eraAnpm      normalize angle into range +/- pi
**     eraS2c       spherical coordinates to unit vector
**     eraRxp       product of r-matrix and p-vector
**     eraC2s       p-vector to spherical
**
**  Reference:
**     Perryman M.A.C. & ESA, 1997, ESA SP-1200, The Hipparcos and Tycho
**     catalogues.  Astrometric and photometric star catalogues
**     derived from the ESA Hipparcos Space Astrometry Mission.  ESA
**     Publications Division, Noordwijk, Netherlands.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraEform ( int n, double *a, double *f )
/*
**  - - - - - - - - -
**   e r a E f o r m
**  - - - - - - - - -
**
**  Earth reference ellipsoids.
**
**  Given:
**     n    int         ellipsoid identifier (Note 1)
**
**  Returned:
**     a    double      equatorial radius (meters, Note 2)
**     f    double      flattening (Note 2)
**
**  Returned (function value):
**          int         status:  0 = OK
**                              -1 = illegal identifier (Note 3)
**
**  Notes:
**
**  1) The identifier n is a number that specifies the choice of
**     reference ellipsoid.  The following are supported:
**
**        n    ellipsoid
**
**        1     ERFA_WGS84
**        2     ERFA_GRS80
**        3     ERFA_WGS72
**
**     The n value has no significance outside the ERFA software.  For
**     convenience, symbols ERFA_WGS84 etc. are defined in erfam.h.
**
**  2) The ellipsoid parameters are returned in the form of equatorial
**     radius in meters (a) and flattening (f).  The latter is a number
**     around 0.00335, i.e. around 1/298.
**
**  3) For the case where an unsupported n value is supplied, zero a and
**     f are returned, as well as error status.
**
**  References:
**
**     Department of Defense World Geodetic System 1984, National
**     Imagery and Mapping Agency Technical Report 8350.2, Third
**     Edition, p3-2.
**
**     Moritz, H., Bull. Geodesique 66-2, 187 (1992).
**
**     The Department of Defense World Geodetic System 1972, World
**     Geodetic System Committee, May 1974.
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     p220.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraGc2gd ( int n, double xyz[3],
               double *elong, double *phi, double *height )
/*
**  - - - - - - - - -
**   e r a G c 2 g d
**  - - - - - - - - -
**
**  Transform geocentric coordinates to geodetic using the specified
**  reference ellipsoid.
**
**  Given:
**     n       int        ellipsoid identifier (Note 1)
**     xyz     double[3]  geocentric vector (Note 2)
**
**  Returned:
**     elong   double     longitude (radians, east +ve, Note 3)
**     phi     double     latitude (geodetic, radians, Note 3)
**     height  double     height above ellipsoid (geodetic, Notes 2,3)
**
**  Returned (function value):
**            int         status:  0 = OK
**                                -1 = illegal identifier (Note 3)
**                                -2 = internal error (Note 3)
**
**  Notes:
**
**  1) The identifier n is a number that specifies the choice of
**     reference ellipsoid.  The following are supported:
**
**        n    ellipsoid
**
**        1     ERFA_WGS84
**        2     ERFA_GRS80
**        3     ERFA_WGS72
**
**     The n value has no significance outside the ERFA software.  For
**     convenience, symbols ERFA_WGS84 etc. are defined in erfam.h.
**
**  2) The geocentric vector (xyz, given) and height (height, returned)
**     are in meters.
**
**  3) An error status -1 means that the identifier n is illegal.  An
**     error status -2 is theoretically impossible.  In all error cases,
**     all three results are set to -1e9.
**
**  4) The inverse transformation is performed in the function eraGd2gc.
**
**  Called:
**     eraEform     Earth reference ellipsoids
**     eraGc2gde    geocentric to geodetic transformation, general
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraGc2gde ( double a, double f, double xyz[3],
                double *elong, double *phi, double *height )
/*
**  - - - - - - - - - -
**   e r a G c 2 g d e
**  - - - - - - - - - -
**
**  Transform geocentric coordinates to geodetic for a reference
**  ellipsoid of specified form.
**
**  Given:
**     a       double     equatorial radius (Notes 2,4)
**     f       double     flattening (Note 3)
**     xyz     double[3]  geocentric vector (Note 4)
**
**  Returned:
**     elong   double     longitude (radians, east +ve)
**     phi     double     latitude (geodetic, radians)
**     height  double     height above ellipsoid (geodetic, Note 4)
**
**  Returned (function value):
**             int        status:  0 = OK
**                                -1 = illegal f
**                                -2 = illegal a
**
**  Notes:
**
**  1) This function is based on the GCONV2H Fortran subroutine by
**     Toshio Fukushima (see reference).
**
**  2) The equatorial radius, a, can be in any units, but meters is
**     the conventional choice.
**
**  3) The flattening, f, is (for the Earth) a value around 0.00335,
**     i.e. around 1/298.
**
**  4) The equatorial radius, a, and the geocentric vector, xyz,
**     must be given in the same units, and determine the units of
**     the returned height, height.
**
**  5) If an error occurs (status < 0), elong, phi and height are
**     unchanged.
**
**  6) The inverse transformation is performed in the function
**     eraGd2gce.
**
**  7) The transformation for a standard ellipsoid (such as ERFA_WGS84) can
**     more conveniently be performed by calling eraGc2gd, which uses a
**     numerical code to identify the required A and F values.
**
**  Reference:
**
**     Fukushima, T., "Transformation from Cartesian to geodetic
**     coordinates accelerated by Halley's method", J.Geodesy (2006)
**     79: 689-693
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraGd2gc ( int n, double elong, double phi, double height,
               double xyz[3] )
/*
**  - - - - - - - - -
**   e r a G d 2 g c
**  - - - - - - - - -
**
**  Transform geodetic coordinates to geocentric using the specified
**  reference ellipsoid.
**
**  Given:
**     n       int        ellipsoid identifier (Note 1)
**     elong   double     longitude (radians, east +ve)
**     phi     double     latitude (geodetic, radians, Note 3)
**     height  double     height above ellipsoid (geodetic, Notes 2,3)
**
**  Returned:
**     xyz     double[3]  geocentric vector (Note 2)
**
**  Returned (function value):
**             int        status:  0 = OK
**                                -1 = illegal identifier (Note 3)
**                                -2 = illegal case (Note 3)
**
**  Notes:
**
**  1) The identifier n is a number that specifies the choice of
**     reference ellipsoid.  The following are supported:
**
**        n    ellipsoid
**
**        1     ERFA_WGS84
**        2     ERFA_GRS80
**        3     ERFA_WGS72
**
**     The n value has no significance outside the ERFA software.  For
**     convenience, symbols ERFA_WGS84 etc. are defined in erfam.h.
**
**  2) The height (height, given) and the geocentric vector (xyz,
**     returned) are in meters.
**
**  3) No validation is performed on the arguments elong, phi and
**     height.  An error status -1 means that the identifier n is
**     illegal.  An error status -2 protects against cases that would
**     lead to arithmetic exceptions.  In all error cases, xyz is set
**     to zeros.
**
**  4) The inverse transformation is performed in the function eraGc2gd.
**
**  Called:
**     eraEform     Earth reference ellipsoids
**     eraGd2gce    geodetic to geocentric transformation, general
**     eraZp        zero p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraGd2gce ( double a, double f, double elong, double phi,
                double height, double xyz[3] )
/*
**  - - - - - - - - - -
**   e r a G d 2 g c e
**  - - - - - - - - - -
**
**  Transform geodetic coordinates to geocentric for a reference
**  ellipsoid of specified form.
**
**  Given:
**     a       double     equatorial radius (Notes 1,4)
**     f       double     flattening (Notes 2,4)
**     elong   double     longitude (radians, east +ve)
**     phi     double     latitude (geodetic, radians, Note 4)
**     height  double     height above ellipsoid (geodetic, Notes 3,4)
**
**  Returned:
**     xyz     double[3]  geocentric vector (Note 3)
**
**  Returned (function value):
**             int        status:  0 = OK
**                                -1 = illegal case (Note 4)
**  Notes:
**
**  1) The equatorial radius, a, can be in any units, but meters is
**     the conventional choice.
**
**  2) The flattening, f, is (for the Earth) a value around 0.00335,
**     i.e. around 1/298.
**
**  3) The equatorial radius, a, and the height, height, must be
**     given in the same units, and determine the units of the
**     returned geocentric vector, xyz.
**
**  4) No validation is performed on individual arguments.  The error
**     status -1 protects against (unrealistic) cases that would lead
**     to arithmetic exceptions.  If an error occurs, xyz is unchanged.
**
**  5) The inverse transformation is performed in the function
**     eraGc2gde.
**
**  6) The transformation for a standard ellipsoid (such as ERFA_WGS84) can
**     more conveniently be performed by calling eraGd2gc,  which uses a
**     numerical code to identify the required a and f values.
**
**  References:
**
**     Green, R.M., Spherical Astronomy, Cambridge University Press,
**     (1985) Section 4.5, p96.
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992),
**     Section 4.22, p202.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraD2dtf(const char *scale, int ndp, double d1, double d2,
             int *iy, int *im, int *id, int ihmsf[4])
/*
**  - - - - - - - - -
**   e r a D 2 d t f
**  - - - - - - - - -
**
**  Format for output a 2-part Julian Date (or in the case of UTC a
**  quasi-JD form that includes special provision for leap seconds).
**
**  Given:
**     scale     char[]  time scale ID (Note 1)
**     ndp       int     resolution (Note 2)
**     d1,d2     double  time as a 2-part Julian Date (Notes 3,4)
**
**  Returned:
**     iy,im,id  int     year, month, day in Gregorian calendar (Note 5)
**     ihmsf     int[4]  hours, minutes, seconds, fraction (Note 1)
**
**  Returned (function value):
**               int     status: +1 = dubious year (Note 5)
**                                0 = OK
**                               -1 = unacceptable date (Note 6)
**
**  Notes:
**
**  1) scale identifies the time scale.  Only the value "UTC" (in upper
**     case) is significant, and enables handling of leap seconds (see
**     Note 4).
**
**  2) ndp is the number of decimal places in the seconds field, and can
**     have negative as well as positive values, such as:
**
**     ndp         resolution
**     -4            1 00 00
**     -3            0 10 00
**     -2            0 01 00
**     -1            0 00 10
**      0            0 00 01
**      1            0 00 00.1
**      2            0 00 00.01
**      3            0 00 00.001
**
**     The limits are platform dependent, but a safe range is -5 to +9.
**
**  3) d1+d2 is Julian Date, apportioned in any convenient way between
**     the two arguments, for example where d1 is the Julian Day Number
**     and d2 is the fraction of a day.  In the case of UTC, where the
**     use of JD is problematical, special conventions apply:  see the
**     next note.
**
**  4) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The ERFA internal convention is that
**     the quasi-JD day represents UTC days whether the length is 86399,
**     86400 or 86401 SI seconds.  In the 1960-1972 era there were
**     smaller jumps (in either direction) each time the linear UTC(TAI)
**     expression was changed, and these "mini-leaps" are also included
**     in the ERFA convention.
**
**  5) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  6) For calendar conventions and limitations, see eraCal2jd.
**
**  Called:
**     eraJd2cal    JD to Gregorian calendar
**     eraD2tf      decompose days to hms
**     eraDat       delta(AT) = TAI-UTC
**
```

```
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraDat(int iy, int im, int id, double fd, double *deltat )
/*
**  - - - - - - -
**   e r a D a t
**  - - - - - - -
**
**  For a given UTC date, calculate Delta(AT) = TAI-UTC.
**
**     :------------------------------------------:
**     :                                          :
**     :                 IMPORTANT                :
**     :                                          :
**     :  A new version of this function must be  :
**     :  produced whenever a new leap second is  :
**     :  announced.  There are four items to     :
**     :  change on each such occasion:           :
**     :                                          :
**     :  1) A new line must be added to the set  :
**     :     of statements that initialize the    :
**     :     array "changes".                     :
**     :                                          :
**     :  2) The constant IYV must be set to the  :
**     :     current year.                        :
**     :                                          :
**     :  3) The "Latest leap second" comment     :
**     :     below must be set to the new leap    :
**     :     second date.                         :
**     :                                          :
**     :  4) The "This revision" comment, later,  :
**     :     must be set to the current date.     :
**     :                                          :
**     :  Change (2) must also be carried out     :
**     :  whenever the function is re-issued,     :
**     :  even if no leap seconds have been       :
**     :  added.                                  :
**     :                                          :
**     :  Latest leap second:  2016 December 31   :
**     :                                          :
**     :__________________________________________:
**
**  Given:
**     iy     int      UTC:  year (Notes 1 and 2)
**     im     int            month (Note 2)
**     id     int            day (Notes 2 and 3)
**     fd     double         fraction of day (Note 4)
**
**  Returned:
**     deltat double   TAI minus UTC, seconds
**
**  Returned (function value):
**            int      status (Note 5):
**                       1 = dubious year (Note 1)
**                       0 = OK
**                      -1 = bad year
**                      -2 = bad month
**                      -3 = bad day (Note 3)
**                      -4 = bad fraction (Note 4)
**                      -5 = internal error (Note 5)
**
**  Notes:
**
**  1) UTC began at 1960 January 1.0 (JD 2436934.5) and it is improper
**     to call the function with an earlier date.  If this is attempted,
**     zero is returned together with a warning status.
**
**     Because leap seconds cannot, in principle, be predicted in
**     advance, a reliable check for dates beyond the valid range is
**     impossible.  To guard against gross errors, a year five or more
**     after the release year of the present function (see the constant
**     IYV) is considered dubious.  In this case a warning status is
```

```
**     returned but the result is computed in the normal way.
**
**     For both too-early and too-late years, the warning status is +1.
**     This is distinct from the error status -1, which signifies a year
**     so early that JD could not be computed.
**
**  2) If the specified date is for a day which ends with a leap second,
**     the TAI-UTC value returned is for the period leading up to the
**     leap second.  If the date is for a day which begins as a leap
**     second ends, the TAI-UTC returned is for the period following the
**     leap second.
**
**  3) The day number must be in the normal calendar range, for example
**     1 through 30 for April.  The "almanac" convention of allowing
**     such dates as January 0 and December 32 is not supported in this
**     function, in order to avoid confusion near leap seconds.
**
**  4) The fraction of day is used only for dates before the
**     introduction of leap seconds, the first of which occurred at the
**     end of 1971.  It is tested for validity (0 to 1 is the valid
**     range) even if not used;  if invalid, zero is used and status -4
**     is returned.  For many applications, setting fd to zero is
**     acceptable;  the resulting error is always less than 3 ms (and
**     occurs only pre-1972).
**
**  5) The status value returned in the case where there are multiple
**     errors refers to the first error detected.  For example, if the
**     month and day are 13 and 32 respectively, status -2 (bad month)
**     will be returned.  The "internal error" status refers to a
**     case that is impossible but causes some compilers to issue a
**     warning.
**
**  6) In cases where a valid result is not available, zero is returned.
**
**  References:
**
**  1) For dates from 1961 January 1 onwards, the expressions from the
**     file ftp://maia.usno.navy.mil/ser7/tai-utc.dat are used.
**
**  2) The 5ms timestep at 1961 January 1 is taken from 2.58.1 (p87) of
**     the 1992 Explanatory Supplement.
**
**  Called:
**     eraCal2jd    Gregorian calendar to JD
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraDtdb(double date1, double date2,
               double ut, double elong, double u, double v)
/*
**  - - - - - - - -
**   e r a D t d b
**  - - - - - - - -
**
**  An approximation to TDB-TT, the difference between barycentric
**  dynamical time and terrestrial time, for an observer on the Earth.
**
**  The different time scales - proper, coordinate and realized - are
**  related to each other:
**
**            TAI             <-  physically realized
**             :
**          offset            <-  observed (nominally +32.184s)
**             :
**            TT              <-  terrestrial time
**             :
**    rate adjustment (L_G)   <-  definition of TT
**             :
**            TCG             <-  time scale for GCRS
**             :
**      "periodic" terms      <-  eraDtdb  is an implementation
**             :
**    rate adjustment (L_C)   <-  function of solar-system ephemeris
**             :
**            TCB             <-  time scale for BCRS
**             :
**    rate adjustment (-L_B)  <-  definition of TDB
**             :
**            TDB             <-  TCB scaled to track TT
**             :
**      "periodic" terms      <-  -eraDtdb is an approximation
**             :
**            TT              <-  terrestrial time
**
**  Adopted values for the various constants can be found in the IERS
**  Conventions (McCarthy & Petit 2003).
**
**  Given:
**     date1,date2   double  date, TDB (Notes 1-3)
**     ut            double  universal time (UT1, fraction of one day)
**     elong         double  longitude (east positive, radians)
**     u             double  distance from Earth spin axis (km)
**     v             double  distance north of equatorial plane (km)
**
**  Returned (function value):
**                   double  TDB-TT (seconds)
**
**  Notes:
**
**  1) The date date1+date2 is a Julian Date, apportioned in any
**     convenient way between the two arguments.  For example,
**     JD(TT)=2450123.7 could be expressed in any of these ways,
**     among others:
**
**            date1          date2
**
**         2450123.7           0.0       (JD method)
**         2451545.0       -1421.3       (J2000 method)
**         2400000.5       50123.2       (MJD method)
**         2450123.5           0.2       (date & time method)
**
**     The JD method is the most natural and convenient to use in
**     cases where the loss of several decimal digits of resolution
**     is acceptable.  The J2000 method is best matched to the way
**     the argument is handled internally and will deliver the
**     optimum resolution.  The MJD method and the date & time methods
**     are both good compromises between resolution and convenience.
```

```
**
**     Although the date is, formally, barycentric dynamical time (TDB),
**     the terrestrial dynamical time (TT) can be used with no practical
**     effect on the accuracy of the prediction.
**
**  2) TT can be regarded as a coordinate time that is realized as an
**     offset of 32.184s from International Atomic Time, TAI.  TT is a
**     specific linear transformation of geocentric coordinate time TCG,
**     which is the time scale for the Geocentric Celestial Reference
**     System, GCRS.
**
**  3) TDB is a coordinate time, and is a specific linear transformation
**     of barycentric coordinate time TCB, which is the time scale for
**     the Barycentric Celestial Reference System, BCRS.
**
**  4) The difference TCG-TCB depends on the masses and positions of the
**     bodies of the solar system and the velocity of the Earth.  It is
**     dominated by a rate difference, the residual being of a periodic
**     character.  The latter, which is modeled by the present function,
**     comprises a main (annual) sinusoidal term of amplitude
**     approximately 0.00166 seconds, plus planetary terms up to about
**     20 microseconds, and lunar and diurnal terms up to 2 microseconds.
**     These effects come from the changing transverse Doppler effect
**     and gravitational red-shift as the observer (on the Earth’s
**     surface) experiences variations in speed (with respect to the
**     BCRS) and gravitational potential.
**
**  5) TDB can be regarded as the same as TCB but with a rate adjustment
**     to keep it close to TT, which is convenient for many applications.
**     The history of successive attempts to define TDB is set out in
**     Resolution 3 adopted by the IAU General Assembly in 2006, which
**     defines a fixed TDB(TCB) transformation that is consistent with
**     contemporary solar-system ephemerides.  Future ephemerides will
**     imply slightly changed transformations between TCG and TCB, which
**     could introduce a linear drift between TDB and TT;  however, any
**     such drift is unlikely to exceed 1 nanosecond per century.
**
**  6) The geocentric TDB-TT model used in the present function is that of
**     Fairhead & Bretagnon (1990), in its full form.  It was originally
**     supplied by Fairhead (private communications with P.T.Wallace,
**     1990) as a Fortran subroutine.  The present C function contains an
**     adaptation of the Fairhead code.  The numerical results are
**     essentially unaffected by the changes, the differences with
**     respect to the Fairhead & Bretagnon original being at the 1e-20 s
**     level.
**
**     The topocentric part of the model is from Moyer (1981) and
**     Murray (1983), with fundamental arguments adapted from
**     Simon et al. 1994.  It is an approximation to the expression
**     ( v / c ) . ( r / c ), where v is the barycentric velocity of
**     the Earth, r is the geocentric position of the observer and
**     c is the speed of light.
**
**     By supplying zeroes for u and v, the topocentric part of the
**     model can be nullified, and the function will return the Fairhead
**     & Bretagnon result alone.
**
**  7) During the interval 1950-2050, the absolute accuracy is better
**     than +/- 3 nanoseconds relative to time ephemerides obtained by
**     direct numerical integrations based on the JPL DE405 solar system
**     ephemeris.
**
**  8) It must be stressed that the present function is merely a model,
**     and that numerical integration of solar-system ephemerides is the
**     definitive method for predicting the relationship between TCG and
**     TCB and hence between TT and TDB.
**
**  References:
**
**     Fairhead, L., & Bretagnon, P., Astron.Astrophys., 229, 240-247
```

```
**     (1990).
**
**     IAU 2006 Resolution 3.
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Moyer, T.D., Cel.Mech., 23, 33 (1981).
**
**     Murray, C.A., Vectorial Astrometry, Adam Hilger (1983).
**
**     Seidelmann, P.K. et al., Explanatory Supplement to the
**     Astronomical Almanac, Chapter 2, University Science Books (1992).
**
**     Simon, J.L., Bretagnon, P., Chapront, J., Chapront-Touze, M.,
**     Francou, G. & Laskar, J., Astron.Astrophys., 282, 663-683 (1994).
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraDtf2d(const char *scale, int iy, int im, int id,
             int ihr, int imn, double sec, double *d1, double *d2)
/*
**  - - - - - - - - -
**   e r a D t f 2 d
**  - - - - - - - - -
**
**  Encode date and time fields into 2-part Julian Date (or in the case
**  of UTC a quasi-JD form that includes special provision for leap
**  seconds).
**
**  Given:
**     scale     char[]  time scale ID (Note 1)
**     iy,im,id  int     year, month, day in Gregorian calendar (Note 2)
**     ihr,imn   int     hour, minute
**     sec       double  seconds
**
**  Returned:
**     d1,d2     double  2-part Julian Date (Notes 3,4)
**
**  Returned (function value):
**               int     status: +3 = both of next two
**                               +2 = time is after end of day (Note 5)
**                               +1 = dubious year (Note 6)
**                                0 = OK
**                               -1 = bad year
**                               -2 = bad month
**                               -3 = bad day
**                               -4 = bad hour
**                               -5 = bad minute
**                               -6 = bad second (<0)
**
**  Notes:
**
**  1) scale identifies the time scale.  Only the value "UTC" (in upper
**     case) is significant, and enables handling of leap seconds (see
**     Note 4).
**
**  2) For calendar conventions and limitations, see eraCal2jd.
**
**  3) The sum of the results, d1+d2, is Julian Date, where normally d1
**     is the Julian Day Number and d2 is the fraction of a day.  In the
**     case of UTC, where the use of JD is problematical, special
**     conventions apply:  see the next note.
**
**  4) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The ERFA internal convention is that
**     the quasi-JD day represents UTC days whether the length is 86399,
**     86400 or 86401 SI seconds.  In the 1960-1972 era there were
**     smaller jumps (in either direction) each time the linear UTC(TAI)
**     expression was changed, and these "mini-leaps" are also included
**     in the ERFA convention.
**
**  5) The warning status "time is after end of day" usually means that
**     the sec argument is greater than 60.0.  However, in a day ending
**     in a leap second the limit changes to 61.0 (or 59.0 in the case
**     of a negative leap second).
**
**  6) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  7) Only in the case of continuous and regular time scales (TAI, TT,
**     TCG, TCB and TDB) is the result d1+d2 a Julian Date, strictly
**     speaking.  In the other cases (UT1 and UTC) the result must be
**     used with circumspection;  in particular the difference between
**     two such results cannot be interpreted as a precise time
**     interval.
**
**  Called:
```

```
**     eraCal2jd    Gregorian calendar to JD
**     eraDat       delta(AT) = TAI-UTC
**     eraJd2cal    JD to Gregorian calendar
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTaitt(double tai1, double tai2, double *tt1, double *tt2)
/*
**  - - - - - - - - -
**   e r a T a i t t
**  - - - - - - - - -
**
**  Time scale transformation:  International Atomic Time, TAI, to
**  Terrestrial Time, TT.
**
**  Given:
**     tai1,tai2  double    TAI as a 2-part Julian Date
**
**  Returned:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Note:
**
**     tai1+tai2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tai1 is the Julian
**     Day Number and tai2 is the fraction of a day.  The returned
**     tt1,tt2 follow suit.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTaiut1(double tai1, double tai2, double dta,
              double *ut11, double *ut12)
/*
**  - - - - - - - - - -
**   e r a T a i u t 1
**  - - - - - - - - - -
**
**  Time scale transformation:  International Atomic Time, TAI, to
**  Universal Time, UT1.
**
**  Given:
**     tai1,tai2  double    TAI as a 2-part Julian Date
**     dta        double    UT1-TAI in seconds
**
**  Returned:
**     ut11,ut12  double    UT1 as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tai1+tai2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tai1 is the Julian
**     Day Number and tai2 is the fraction of a day.  The returned
**     UT11,UT12 follow suit.
**
**  2) The argument dta, i.e. UT1-TAI, is an observed quantity, and is
**     available from IERS tabulations.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTaiutc(double tai1, double tai2, double *utc1, double *utc2)
/*
**  - - - - - - - - - -
**   e r a T a i u t c
**  - - - - - - - - - -
**
**  Time scale transformation:  International Atomic Time, TAI, to
**  Coordinated Universal Time, UTC.
**
**  Given:
**     tai1,tai2  double   TAI as a 2-part Julian Date (Note 1)
**
**  Returned:
**     utc1,utc2  double   UTC as a 2-part quasi Julian Date (Notes 1-3)
**
**  Returned (function value):
**                int      status: +1 = dubious year (Note 4)
**                                  0 = OK
**                                 -1 = unacceptable date
**
**  Notes:
**
**  1) tai1+tai2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tai1 is the Julian
**     Day Number and tai2 is the fraction of a day.  The returned utc1
**     and utc2 form an analogous pair, except that a special convention
**     is used, to deal with the problem of leap seconds - see the next
**     note.
**
**  2) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The convention in the present
**     function is that the JD day represents UTC days whether the
**     length is 86399, 86400 or 86401 SI seconds.  In the 1960-1972 era
**     there were smaller jumps (in either direction) each time the
**     linear UTC(TAI) expression was changed, and these "mini-leaps"
**     are also included in the ERFA convention.
**
**  3) The function eraD2dtf can be used to transform the UTC quasi-JD
**     into calendar date and clock time, including UTC leap second
**     handling.
**
**  4) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  Called:
**     eraUtctai    UTC to TAI
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTcbtdb(double tcb1, double tcb2, double *tdb1, double *tdb2)
/*
**  - - - - - - - - - -
**   e r a T c b t d b
**  - - - - - - - - - -
**
**  Time scale transformation:  Barycentric Coordinate Time, TCB, to
**  Barycentric Dynamical Time, TDB.
**
**  Given:
**     tcb1,tcb2  double    TCB as a 2-part Julian Date
**
**  Returned:
**     tdb1,tdb2  double    TDB as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tcb1+tcb2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tcb1 is the Julian
**     Day Number and tcb2 is the fraction of a day.  The returned
**     tdb1,tdb2 follow suit.
**
**  2) The 2006 IAU General Assembly introduced a conventional linear
**     transformation between TDB and TCB.  This transformation
**     compensates for the drift between TCB and terrestrial time TT,
**     and keeps TDB approximately centered on TT.  Because the
**     relationship between TT and TCB depends on the adopted solar
**     system ephemeris, the degree of alignment between TDB and TT over
**     long intervals will vary according to which ephemeris is used.
**     Former definitions of TDB attempted to avoid this problem by
**     stipulating that TDB and TT should differ only by periodic
**     effects.  This is a good description of the nature of the
**     relationship but eluded precise mathematical formulation.  The
**     conventional linear relationship adopted in 2006 sidestepped
**     these difficulties whilst delivering a TDB that in practice was
**     consistent with values before that date.
**
**  3) TDB is essentially the same as Teph, the time argument for the
**     JPL solar system ephemerides.
**
**  Reference:
**
**     IAU 2006 Resolution B3
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTcgtt(double tcg1, double tcg2, double *tt1, double *tt2)
/*
**  - - - - - - - - -
**   e r a T c g t t
**  - - - - - - - - -
**
**  Time scale transformation:  Geocentric Coordinate Time, TCG, to
**  Terrestrial Time, TT.
**
**  Given:
**     tcg1,tcg2  double    TCG as a 2-part Julian Date
**
**  Returned:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Note:
**
**     tcg1+tcg2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tcg1 is the Julian
**     Day Number and tcg22 is the fraction of a day.  The returned
**     tt1,tt2 follow suit.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),.
**     IERS Technical Note No. 32, BKG (2004)
**
**     IAU 2000 Resolution B1.9
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTdbtcb(double tdb1, double tdb2, double *tcb1, double *tcb2)
/*
**  - - - - - - - - - -
**   e r a T d b t c b
**  - - - - - - - - - -
**
**  Time scale transformation:  Barycentric Dynamical Time, TDB, to
**  Barycentric Coordinate Time, TCB.
**
**  Given:
**     tdb1,tdb2  double    TDB as a 2-part Julian Date
**
**  Returned:
**     tcb1,tcb2  double    TCB as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tdb1+tdb2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tdb1 is the Julian
**     Day Number and tdb2 is the fraction of a day.  The returned
**     tcb1,tcb2 follow suit.
**
**  2) The 2006 IAU General Assembly introduced a conventional linear
**     transformation between TDB and TCB.  This transformation
**     compensates for the drift between TCB and terrestrial time TT,
**     and keeps TDB approximately centered on TT.  Because the
**     relationship between TT and TCB depends on the adopted solar
**     system ephemeris, the degree of alignment between TDB and TT over
**     long intervals will vary according to which ephemeris is used.
**     Former definitions of TDB attempted to avoid this problem by
**     stipulating that TDB and TT should differ only by periodic
**     effects.  This is a good description of the nature of the
**     relationship but eluded precise mathematical formulation.  The
**     conventional linear relationship adopted in 2006 sidestepped
**     these difficulties whilst delivering a TDB that in practice was
**     consistent with values before that date.
**
**  3) TDB is essentially the same as Teph, the time argument for the
**     JPL solar system ephemerides.
**
**  Reference:
**
**     IAU 2006 Resolution B3
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTdbtt(double tdb1, double tdb2, double dtr,
             double *tt1, double *tt2 )
/*
**  - - - - - - - - -
**   e r a T d b t t
**  - - - - - - - - -
**
**  Time scale transformation:  Barycentric Dynamical Time, TDB, to
**  Terrestrial Time, TT.
**
**  Given:
**     tdb1,tdb2  double    TDB as a 2-part Julian Date
**     dtr        double    TDB-TT in seconds
**
**  Returned:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tdb1+tdb2 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where tdb1 is the Julian
**     Day Number and tdb2 is the fraction of a day.  The returned
**     tt1,tt2 follow suit.
**
**  2) The argument dtr represents the quasi-periodic component of the
**     GR transformation between TT and TCB.  It is dependent upon the
**     adopted solar-system ephemeris, and can be obtained by numerical
**     integration, by interrogating a precomputed time ephemeris or by
**     evaluating a model such as that implemented in the ERFA function
**     eraDtdb.   The quantity is dominated by an annual term of 1.7 ms
**     amplitude.
**
**  3) TDB is essentially the same as Teph, the time argument for the
**     JPL solar system ephemerides.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     IAU 2006 Resolution 3
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTttai(double tt1, double tt2, double *tai1, double *tai2)
/*
**  - - - - - - - - -
**   e r a T t t a i
**  - - - - - - - - -
**
**  Time scale transformation:  Terrestrial Time, TT, to International
**  Atomic Time, TAI.
**
**  Given:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned:
**     tai1,tai2  double    TAI as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Note:
**
**     tt1+tt2 is Julian Date, apportioned in any convenient way between
**     the two arguments, for example where tt1 is the Julian Day Number
**     and tt2 is the fraction of a day.  The returned tai1,tai2 follow
**     suit.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTttcg(double tt1, double tt2, double *tcg1, double *tcg2)
/*
**  - - - - - - - - -
**   e r a T t t c g
**  - - - - - - - - -
**
**  Time scale transformation:  Terrestrial Time, TT, to Geocentric
**  Coordinate Time, TCG.
**
**  Given:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned:
**     tcg1,tcg2  double    TCG as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Note:
**
**     tt1+tt2 is Julian Date, apportioned in any convenient way between
**     the two arguments, for example where tt1 is the Julian Day Number
**     and tt2 is the fraction of a day.  The returned tcg1,tcg2 follow
**     suit.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     IAU 2000 Resolution B1.9
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTttdb(double tt1, double tt2, double dtr,
             double *tdb1, double *tdb2)
/*
**  - - - - - - - - -
**   e r a T t t d b
**  - - - - - - - - -
**
**  Time scale transformation:  Terrestrial Time, TT, to Barycentric
**  Dynamical Time, TDB.
**
**  Given:
**     tt1,tt2    double    TT as a 2-part Julian Date
**     dtr        double    TDB-TT in seconds
**
**  Returned:
**     tdb1,tdb2  double    TDB as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tt1+tt2 is Julian Date, apportioned in any convenient way between
**     the two arguments, for example where tt1 is the Julian Day Number
**     and tt2 is the fraction of a day.  The returned tdb1,tdb2 follow
**     suit.
**
**  2) The argument dtr represents the quasi-periodic component of the
**     GR transformation between TT and TCB.  It is dependent upon the
**     adopted solar-system ephemeris, and can be obtained by numerical
**     integration, by interrogating a precomputed time ephemeris or by
**     evaluating a model such as that implemented in the ERFA function
**     eraDtdb.   The quantity is dominated by an annual term of 1.7 ms
**     amplitude.
**
**  3) TDB is essentially the same as Teph, the time argument for the JPL
**     solar system ephemerides.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     IAU 2006 Resolution 3
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTtut1(double tt1, double tt2, double dt,
             double *ut11, double *ut12)
/*
**  - - - - - - - - -
**   e r a T t u t 1
**  - - - - - - - - -
**
**  Time scale transformation:  Terrestrial Time, TT, to Universal Time,
**  UT1.
**
**  Given:
**     tt1,tt2    double    TT as a 2-part Julian Date
**     dt         double    TT-UT1 in seconds
**
**  Returned:
**     ut11,ut12  double    UT1 as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) tt1+tt2 is Julian Date, apportioned in any convenient way between
**     the two arguments, for example where tt1 is the Julian Day Number
**     and tt2 is the fraction of a day.  The returned ut11,ut12 follow
**     suit.
**
**  2) The argument dt is classical Delta T.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraUt1tai(double ut11, double ut12, double dta,
              double *tai1, double *tai2)
/*
**  - - - - - - - - - -
**   e r a U t 1 t a i
**  - - - - - - - - - -
**
**  Time scale transformation:  Universal Time, UT1, to International
**  Atomic Time, TAI.
**
**  Given:
**     ut11,ut12  double    UT1 as a 2-part Julian Date
**     dta        double    UT1-TAI in seconds
**
**  Returned:
**     tai1,tai2  double    TAI as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) ut11+ut12 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where ut11 is the Julian
**     Day Number and ut12 is the fraction of a day.  The returned
**     tai1,tai2 follow suit.
**
**  2) The argument dta, i.e. UT1-TAI, is an observed quantity, and is
**     available from IERS tabulations.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraUt1tt(double ut11, double ut12, double dt,
             double *tt1, double *tt2)
/*
**  - - - - - - - - -
**   e r a U t 1 t t
**  - - - - - - - - -
**
**  Time scale transformation:  Universal Time, UT1, to Terrestrial
**  Time, TT.
**
**  Given:
**     ut11,ut12  double    UT1 as a 2-part Julian Date
**     dt         double    TT-UT1 in seconds
**
**  Returned:
**     tt1,tt2    double    TT as a 2-part Julian Date
**
**  Returned (function value):
**                int       status:  0 = OK
**
**  Notes:
**
**  1) ut11+ut12 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where ut11 is the Julian
**     Day Number and ut12 is the fraction of a day.  The returned
**     tt1,tt2 follow suit.
**
**  2) The argument dt is classical Delta T.
**
**  Reference:
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraUt1utc(double ut11, double ut12, double dut1,
              double *utc1, double *utc2)
/*
**  - - - - - - - - - -
**   e r a U t 1 u t c
**  - - - - - - - - - -
**
**  Time scale transformation:  Universal Time, UT1, to Coordinated
**  Universal Time, UTC.
**
**  Given:
**     ut11,ut12  double   UT1 as a 2-part Julian Date (Note 1)
**     dut1       double   Delta UT1: UT1-UTC in seconds (Note 2)
**
**  Returned:
**     utc1,utc2  double   UTC as a 2-part quasi Julian Date (Notes 3,4)
**
**  Returned (function value):
**                int      status: +1 = dubious year (Note 5)
**                                  0 = OK
**                                 -1 = unacceptable date
**
**  Notes:
**
**  1) ut11+ut12 is Julian Date, apportioned in any convenient way
**     between the two arguments, for example where ut11 is the Julian
**     Day Number and ut12 is the fraction of a day.  The returned utc1
**     and utc2 form an analogous pair, except that a special convention
**     is used, to deal with the problem of leap seconds - see Note 3.
**
**  2) Delta UT1 can be obtained from tabulations provided by the
**     International Earth Rotation and Reference Systems Service.  The
**     value changes abruptly by 1s at a leap second;  however, close to
**     a leap second the algorithm used here is tolerant of the "wrong"
**     choice of value being made.
**
**  3) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The convention in the present
**     function is that the returned quasi JD day UTC1+UTC2 represents
**     UTC days whether the length is 86399, 86400 or 86401 SI seconds.
**
**  4) The function eraD2dtf can be used to transform the UTC quasi-JD
**     into calendar date and clock time, including UTC leap second
**     handling.
**
**  5) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  Called:
**     eraJd2cal    JD to Gregorian calendar
**     eraDat       delta(AT) = TAI-UTC
**     eraCal2jd    Gregorian calendar to JD
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraUtctai(double utc1, double utc2, double *tai1, double *tai2)
/*
**  - - - - - - - - - -
**   e r a U t c t a i
**  - - - - - - - - - -
**
**  Time scale transformation:  Coordinated Universal Time, UTC, to
**  International Atomic Time, TAI.
**
**  Given:
**     utc1,utc2  double   UTC as a 2-part quasi Julian Date (Notes 1-4)
**
**  Returned:
**     tai1,tai2  double   TAI as a 2-part Julian Date (Note 5)
**
**  Returned (function value):
**                int      status: +1 = dubious year (Note 3)
**                                  0 = OK
**                                 -1 = unacceptable date
**
**  Notes:
**
**  1) utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**     convenient way between the two arguments, for example where utc1
**     is the Julian Day Number and utc2 is the fraction of a day.
**
**  2) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The convention in the present
**     function is that the JD day represents UTC days whether the
**     length is 86399, 86400 or 86401 SI seconds.  In the 1960-1972 era
**     there were smaller jumps (in either direction) each time the
**     linear UTC(TAI) expression was changed, and these "mini-leaps"
**     are also included in the ERFA convention.
**
**  3) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  4) The function eraDtf2d converts from calendar date and time of day
**     into 2-part Julian Date, and in the case of UTC implements the
**     leap-second-ambiguity convention described above.
**
**  5) The returned TAI1,TAI2 are such that their sum is the TAI Julian
**     Date.
**
**  Called:
**     eraJd2cal    JD to Gregorian calendar
**     eraDat       delta(AT) = TAI-UTC
**     eraCal2jd    Gregorian calendar to JD
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraUtcut1(double utc1, double utc2, double dut1,
              double *ut11, double *ut12)
/*
**  - - - - - - - - - -
**   e r a U t c u t 1
**  - - - - - - - - - -
**
**  Time scale transformation:  Coordinated Universal Time, UTC, to
**  Universal Time, UT1.
**
**  Given:
**     utc1,utc2  double   UTC as a 2-part quasi Julian Date (Notes 1-4)
**     dut1       double   Delta UT1 = UT1-UTC in seconds (Note 5)
**
**  Returned:
**     ut11,ut12  double   UT1 as a 2-part Julian Date (Note 6)
**
**  Returned (function value):
**                int      status: +1 = dubious year (Note 3)
**                                  0 = OK
**                                 -1 = unacceptable date
**
**  Notes:
**
**  1) utc1+utc2 is quasi Julian Date (see Note 2), apportioned in any
**     convenient way between the two arguments, for example where utc1
**     is the Julian Day Number and utc2 is the fraction of a day.
**
**  2) JD cannot unambiguously represent UTC during a leap second unless
**     special measures are taken.  The convention in the present
**     function is that the JD day represents UTC days whether the
**     length is 86399, 86400 or 86401 SI seconds.
**
**  3) The warning status "dubious year" flags UTCs that predate the
**     introduction of the time scale or that are too far in the future
**     to be trusted.  See eraDat for further details.
**
**  4) The function eraDtf2d converts from calendar date and time of
**     day into 2-part Julian Date, and in the case of UTC implements
**     the leap-second-ambiguity convention described above.
**
**  5) Delta UT1 can be obtained from tabulations provided by the
**     International Earth Rotation and Reference Systems Service.
**     It is the caller’s responsibility to supply a dut1 argument
**     containing the UT1-UTC value that matches the given UTC.
**
**  6) The returned ut11,ut12 are such that their sum is the UT1 Julian
**     Date.
**
**  References:
**
**     McCarthy, D. D., Petit, G. (eds.), IERS Conventions (2003),
**     IERS Technical Note No. 32, BKG (2004)
**
**     Explanatory Supplement to the Astronomical Almanac,
**     P. Kenneth Seidelmann (ed), University Science Books (1992)
**
**  Called:
**     eraJd2cal    JD to Gregorian calendar
**     eraDat       delta(AT) = TAI-UTC
**     eraUtctai    UTC to TAI
**     eraTaiut1    TAI to UT1
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraA2af(int ndp, double angle, char *sign, int idmsf[4])
/*
**  - - - - - - - -
**   e r a A 2 a f
**  - - - - - - - -
**
**  Decompose radians into degrees, arcminutes, arcseconds, fraction.
**
**  Given:
**     ndp     int     resolution (Note 1)
**     angle   double  angle in radians
**
**  Returned:
**     sign    char    '+' or '-'
**     idmsf   int[4]  degrees, arcminutes, arcseconds, fraction
**
**  Called:
**     eraD2tf      decompose days to hms
**
**  Notes:
**
**  1) The argument ndp is interpreted as follows:
**
**     ndp         resolution
**      :      ...0000 00 00
**     -7         1000 00 00
**     -6          100 00 00
**     -5           10 00 00
**     -4            1 00 00
**     -3            0 10 00
**     -2            0 01 00
**     -1            0 00 10
**      0            0 00 01
**      1            0 00 00.1
**      2            0 00 00.01
**      3            0 00 00.001
**      :            0 00 00.000...
**
**  2) The largest positive useful value for ndp is determined by the
**     size of angle, the format of doubles on the target platform, and
**     the risk of overflowing idmsf[3].  On a typical platform, for
**     angle up to 2pi, the available floating-point precision might
**     correspond to ndp=12.  However, the practical limit is typically
**     ndp=9, set by the capacity of a 32-bit int, or ndp=4 if int is
**     only 16 bits.
**
**  3) The absolute value of angle may exceed 2pi.  In cases where it
**     does not, it is up to the caller to test for and handle the
**     case where angle is very nearly 2pi and rounds up to 360 degrees,
**     by testing for idmsf[0]=360 and setting idmsf[0-3] to zero.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraA2tf(int ndp, double angle, char *sign, int ihmsf[4])
/*
**  - - - - - - - -
**   e r a A 2 t f
**  - - - - - - - -
**
**  Decompose radians into hours, minutes, seconds, fraction.
**
**  Given:
**     ndp     int     resolution (Note 1)
**     angle   double  angle in radians
**
**  Returned:
**     sign    char    '+' or '-'
**     ihmsf   int[4]  hours, minutes, seconds, fraction
**
**  Called:
**     eraD2tf      decompose days to hms
**
**  Notes:
**
**  1) The argument ndp is interpreted as follows:
**
**     ndp         resolution
**      :      ...0000 00 00
**     -7         1000 00 00
**     -6          100 00 00
**     -5           10 00 00
**     -4            1 00 00
**     -3            0 10 00
**     -2            0 01 00
**     -1            0 00 10
**      0            0 00 01
**      1            0 00 00.1
**      2            0 00 00.01
**      3            0 00 00.001
**      :            0 00 00.000...
**
**  2) The largest positive useful value for ndp is determined by the
**     size of angle, the format of doubles on the target platform, and
**     the risk of overflowing ihmsf[3].  On a typical platform, for
**     angle up to 2pi, the available floating-point precision might
**     correspond to ndp=12.  However, the practical limit is typically
**     ndp=9, set by the capacity of a 32-bit int, or ndp=4 if int is
**     only 16 bits.
**
**  3) The absolute value of angle may exceed 2pi.  In cases where it
**     does not, it is up to the caller to test for and handle the
**     case where angle is very nearly 2pi and rounds up to 24 hours,
**     by testing for ihmsf[0]=24 and setting ihmsf[0-3] to zero.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraAf2a(char s, int ideg, int iamin, double asec, double *rad)
/*
**  - - - - - - - -
**   e r a A f 2 a
**  - - - - - - - -
**
**  Convert degrees, arcminutes, arcseconds to radians.
**
**  Given:
**     s         char    sign:  '-' = negative, otherwise positive
**     ideg      int     degrees
**     iamin     int     arcminutes
**     asec      double  arcseconds
**
**  Returned:
**     rad       double  angle in radians
**
**  Returned (function value):
**               int     status:  0 = OK
**                                1 = ideg outside range 0-359
**                                2 = iamin outside range 0-59
**                                3 = asec outside range 0-59.999...
**
**  Notes:
**
**  1)  The result is computed even if any of the range checks fail.
**
**  2)  Negative ideg, iamin and/or asec produce a warning status, but
**      the absolute value is used in the conversion.
**
**  3)  If there are multiple errors, the status value reflects only the
**      first, the smallest taking precedence.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraAnp(double a)
/*
**  - - - - - - -
**   e r a A n p
**  - - - - - - -
**
**  Normalize angle into the range 0 <= a < 2pi.
**
**  Given:
**     a        double     angle (radians)
**
**  Returned (function value):
**              double     angle in range 0-2pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraAnpm(double a)
/*
**  - - - - - - - -
**   e r a A n p m
**  - - - - - - - -
**
**  Normalize angle into the range -pi <= a < +pi.
**
**  Given:
**     a        double     angle (radians)
**
**  Returned (function value):
**              double     angle in range +/-pi
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraD2tf(int ndp, double days, char *sign, int ihmsf[4])
/*
**  - - - - - - - -
**   e r a D 2 t f
**  - - - - - - - -
**
**  Decompose days to hours, minutes, seconds, fraction.
**
**  Given:
**     ndp     int     resolution (Note 1)
**     days    double  interval in days
**
**  Returned:
**     sign    char    '+' or '-'
**     ihmsf   int[4]  hours, minutes, seconds, fraction
**
**  Notes:
**
**  1) The argument ndp is interpreted as follows:
**
**     ndp         resolution
**      :      ...0000 00 00
**     -7         1000 00 00
**     -6          100 00 00
**     -5           10 00 00
**     -4            1 00 00
**     -3            0 10 00
**     -2            0 01 00
**     -1            0 00 10
**      0            0 00 01
**      1            0 00 00.1
**      2            0 00 00.01
**      3            0 00 00.001
**      :            0 00 00.000...
**
**  2) The largest positive useful value for ndp is determined by the
**     size of days, the format of double on the target platform, and
**     the risk of overflowing ihmsf[3].  On a typical platform, for
**     days up to 1.0, the available floating-point precision might
**     correspond to ndp=12.  However, the practical limit is typically
**     ndp=9, set by the capacity of a 32-bit int, or ndp=4 if int is
**     only 16 bits.
**
**  3) The absolute value of days may exceed 1.0.  In cases where it
**     does not, it is up to the caller to test for and handle the
**     case where days is very nearly 1.0 and rounds up to 24 hours,
**     by testing for ihmsf[0]=24 and setting ihmsf[0-3] to zero.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTf2a(char s, int ihour, int imin, double sec, double *rad)
/*
**  - - - - - - - -
**   e r a T f 2 a
**  - - - - - - - -
**
**  Convert hours, minutes, seconds to radians.
**
**  Given:
**     s         char    sign:  '-' = negative, otherwise positive
**     ihour     int     hours
**     imin      int     minutes
**     sec       double  seconds
**
**  Returned:
**     rad       double  angle in radians
**
**  Returned (function value):
**               int     status:  0 = OK
**                                1 = ihour outside range 0-23
**                                2 = imin outside range 0-59
**                                3 = sec outside range 0-59.999...
**
**  Notes:
**
**  1)  The result is computed even if any of the range checks fail.
**
**  2)  Negative ihour, imin and/or sec produce a warning status, but
**      the absolute value is used in the conversion.
**
**  3)  If there are multiple errors, the status value reflects only the
**      first, the smallest taking precedence.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
int eraTf2d(char s, int ihour, int imin, double sec, double *days)
/*
**  - - - - - - - -
**   e r a T f 2 d
**  - - - - - - - -
**
**  Convert hours, minutes, seconds to days.
**
**  Given:
**     s         char    sign:  '-' = negative, otherwise positive
**     ihour     int     hours
**     imin      int     minutes
**     sec       double  seconds
**
**  Returned:
**     days      double  interval in days
**
**  Returned (function value):
**               int     status:  0 = OK
**                                1 = ihour outside range 0-23
**                                2 = imin outside range 0-59
**                                3 = sec outside range 0-59.999...
**
**  Notes:
**
**  1)  The result is computed even if any of the range checks fail.
**
**  2)  Negative ihour, imin and/or sec produce a warning status, but
**      the absolute value is used in the conversion.
**
**  3)  If there are multiple errors, the status value reflects only the
**      first, the smallest taking precedence.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRx(double phi, double r[3][3])
/*
**  - - - - - -
**   e r a R x
**  - - - - - -
**
**  Rotate an r-matrix about the x-axis.
**
**  Given:
**     phi    double          angle (radians)
**
**  Given and returned:
**     r      double[3][3]    r-matrix, rotated
**
**  Notes:
**
**  1) Calling this function with positive phi incorporates in the
**     supplied r-matrix r an additional rotation, about the x-axis,
**     anticlockwise as seen looking towards the origin from positive x.
**
**  2) The additional rotation can be represented by this matrix:
**
**         (  1        0            0      )
**         (                               )
**         (  0   + cos(phi)   + sin(phi)  )
**         (                               )
**         (  0   - sin(phi)   + cos(phi)  )
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRy(double theta, double r[3][3])
/*
**  - - - - - -
**   e r a R y
**  - - - - - -
**
**  Rotate an r-matrix about the y-axis.
**
**  Given:
**     theta  double          angle (radians)
**
**  Given and returned:
**     r      double[3][3]    r-matrix, rotated
**
**  Notes:
**
**  1) Calling this function with positive theta incorporates in the
**     supplied r-matrix r an additional rotation, about the y-axis,
**     anticlockwise as seen looking towards the origin from positive y.
**
**  2) The additional rotation can be represented by this matrix:
**
**         (  + cos(theta)     0      - sin(theta)  )
**         (                                        )
**         (       0           1           0        )
**         (                                        )
**         (  + sin(theta)     0      + cos(theta)  )
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRz(double psi, double r[3][3])
/*
**  - - - - - -
**   e r a R z
**  - - - - - -
**
**  Rotate an r-matrix about the z-axis.
**
**  Given:
**     psi    double          angle (radians)
**
**  Given and returned:
**     r      double[3][3]    r-matrix, rotated
**
**  Notes:
**
**  1) Calling this function with positive psi incorporates in the
**     supplied r-matrix r an additional rotation, about the z-axis,
**     anticlockwise as seen looking towards the origin from positive z.
**
**  2) The additional rotation can be represented by this matrix:
**
**         (  + cos(psi)   + sin(psi)     0  )
**         (                                 )
**         (  - sin(psi)   + cos(psi)     0  )
**         (                                 )
**         (       0            0         1  )
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraCp(double p[3], double c[3])
/*
**  - - - - - -
**   e r a C p
**  - - - - - -
**
**  Copy a p-vector.
**
**  Given:
**     p        double[3]     p-vector to be copied
**
**  Returned:
**     c        double[3]     copy
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraCpv(double pv[2][3], double c[2][3])
/*
**  - - - - - - -
**   e r a C p v
**  - - - - - - -
**
**  Copy a position/velocity vector.
**
**  Given:
**     pv     double[2][3]    position/velocity vector to be copied
**
**  Returned:
**     c      double[2][3]    copy
**
**  Called:
**     eraCp        copy p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraCr(double r[3][3], double c[3][3])
/*
**  - - - - - -
**   e r a C r
**  - - - - - -
**
**  Copy an r-matrix.
**
**  Given:
**     r        double[3][3]    r-matrix to be copied
**
**  Returned:
**     c        double[3][3]    copy
**
**  Called:
**     eraCp        copy p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraP2pv(double p[3], double pv[2][3])
/*
**  - - - - - - - - -
**   e r a P 2 p v
**  - - - - - - - - -
**
**  Extend a p-vector to a pv-vector by appending a zero velocity.
**
**  Given:
**     p        double[3]       p-vector
**
**  Returned:
**     pv       double[2][3]    pv-vector
**
**  Called:
**     eraCp        copy p-vector
**     eraZp        zero p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPv2p(double pv[2][3], double p[3])
/*
**  - - - - - - - -
**   e r a P v 2 p
**  - - - - - - - -
**
**  Discard velocity component of a pv-vector.
**
**  Given:
**     pv      double[2][3]     pv-vector
**
**  Returned:
**     p       double[3]        p-vector
**
**  Called:
**     eraCp        copy p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraIr(double r[3][3])
/*
**  - - - - - -
**   e r a I r
**  - - - - - -
**
**  Initialize an r-matrix to the identity matrix.
**
**  Returned:
**     r       double[3][3]    r-matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraZp(double p[3])
/*
**  - - - - - -
**   e r a Z p
**  - - - - - -
**
**  Zero a p-vector.
**
**  Returned:
**     p        double[3]      p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraZpv(double pv[2][3])
/*
**  - - - - - - - -
**   e r a Z p v
**  - - - - - - - -
**
**  Zero a pv-vector.
**
**  Returned:
**     pv       double[2][3]      pv-vector
**
**  Called:
**     eraZp        zero p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraZr(double r[3][3])
/*
**  - - - - - -
**   e r a Z r
**  - - - - - -
**
**  Initialize an r-matrix to the null matrix.
**
**  Returned:
**     r        double[3][3]    r-matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRxr(double a[3][3], double b[3][3], double atb[3][3])
/*
**  - - - - - - -
**   e r a R x r
**  - - - - - - -
**
**  Multiply two r-matrices.
**
**  Given:
**     a        double[3][3]    first r-matrix
**     b        double[3][3]    second r-matrix
**
**  Returned:
**     atb      double[3][3]    a * b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Called:
**     eraCr        copy r-matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraTr(double r[3][3], double rt[3][3])
/*
**  - - - - - -
**   e r a T r
**  - - - - - -
**
**  Transpose an r-matrix.
**
**  Given:
**     r        double[3][3]    r-matrix
**
**  Returned:
**     rt       double[3][3]    transpose
**
**  Note:
**     It is permissible for r and rt to be the same array.
**
**  Called:
**     eraCr        copy r-matrix
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRxp(double r[3][3], double p[3], double rp[3])
/*
**  - - - - - - - -
**   e r a R x p
**  - - - - - - - -
**
**  Multiply a p-vector by an r-matrix.
**
**  Given:
**     r        double[3][3]    r-matrix
**     p        double[3]       p-vector
**
**  Returned:
**     rp       double[3]       r * p
**
**  Note:
**     It is permissible for p and rp to be the same array.
**
**  Called:
**     eraCp        copy p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRxpv(double r[3][3], double pv[2][3], double rpv[2][3])
/*
**  - - - - - - - -
**   e r a R x p v
**  - - - - - - - -
**
**  Multiply a pv-vector by an r-matrix.
**
**  Given:
**     r        double[3][3]    r-matrix
**     pv       double[2][3]    pv-vector
**
**  Returned:
**     rpv      double[2][3]    r * pv
**
**  Note:
**     It is permissible for pv and rpv to be the same array.
**
**  Called:
**     eraRxp       product of r-matrix and p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraTrxp(double r[3][3], double p[3], double trp[3])
/*
**  - - - - - - - -
**   e r a T r x p
**  - - - - - - - -
**
**  Multiply a p-vector by the transpose of an r-matrix.
**
**  Given:
**     r        double[3][3]   r-matrix
**     p        double[3]      p-vector
**
**  Returned:
**     trp      double[3]      r * p
**
**  Note:
**     It is permissible for p and trp to be the same array.
**
**  Called:
**     eraTr        transpose r-matrix
**     eraRxp       product of r-matrix and p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraTrxpv(double r[3][3], double pv[2][3], double trpv[2][3])
/*
**  - - - - - - - - -
**   e r a T r x p v
**  - - - - - - - - -
**
**  Multiply a pv-vector by the transpose of an r-matrix.
**
**  Given:
**     r        double[3][3]    r-matrix
**     pv       double[2][3]    pv-vector
**
**  Returned:
**     trpv     double[2][3]    r * pv
**
**  Note:
**     It is permissible for pv and trpv to be the same array.
**
**  Called:
**     eraTr        transpose r-matrix
**     eraRxpv      product of r-matrix and pv-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRm2v(double r[3][3], double w[3])
/*
**  - - - - - - - -
**   e r a R m 2 v
**  - - - - - - - -
**
**  Express an r-matrix as an r-vector.
**
**  Given:
**     r        double[3][3]    rotation matrix
**
**  Returned:
**     w        double[3]       rotation vector (Note 1)
**
**  Notes:
**
**  1) A rotation matrix describes a rotation through some angle about
**     some arbitrary axis called the Euler axis.  The "rotation vector"
**     returned by this function has the same direction as the Euler axis,
**     and its magnitude is the angle in radians.  (The magnitude and
**     direction can be separated by means of the function eraPn.)
**
**  2) If r is null, so is the result.  If r is not a rotation matrix
**     the result is undefined;  r must be proper (i.e. have a positive
**     determinant) and real orthogonal (inverse = transpose).
**
**  3) The reference frame rotates clockwise as seen looking along
**     the rotation vector from the origin.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraRv2m(double w[3], double r[3][3])
/*
**  - - - - - - - - -
**   e r a R v 2 m
**  - - - - - - - - -
**
**  Form the r-matrix corresponding to a given r-vector.
**
**  Given:
**     w        double[3]      rotation vector (Note 1)
**
**  Returned:
**     r        double[3][3]    rotation matrix
**
**  Notes:
**
**  1) A rotation matrix describes a rotation through some angle about
**     some arbitrary axis called the Euler axis.  The "rotation vector"
**     supplied to This function has the same direction as the Euler
**     axis, and its magnitude is the angle in radians.
**
**  2) If w is null, the unit matrix is returned.
**
**  3) The reference frame rotates clockwise as seen looking along the
**     rotation vector from the origin.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraPap(double a[3], double b[3])
/*
**  - - - - - - -
**   e r a P a p
**  - - - - - - -
**
**  Position-angle from two p-vectors.
**
**  Given:
**     a      double[3]  direction of reference point
**     b      double[3]  direction of point whose PA is required
**
**  Returned (function value):
**            double     position angle of b with respect to a (radians)
**
**  Notes:
**
**  1) The result is the position angle, in radians, of direction b with
**     respect to direction a.  It is in the range -pi to +pi.  The
**     sense is such that if b is a small distance "north" of a the
**     position angle is approximately zero, and if b is a small
**     distance "east" of a the position angle is approximately +pi/2.
**
**  2) The vectors a and b need not be of unit length.
**
**  3) Zero is returned if the two directions are the same or if either
**     vector is null.
**
**  4) If vector a is at a pole, the result is ill-defined.
**
**  Called:
**     eraPn        decompose p-vector into modulus and direction
**     eraPm        modulus of p-vector
**     eraPxp       vector product of two p-vectors
**     eraPmp       p-vector minus p-vector
**     eraPdp       scalar product of two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraPas(double al, double ap, double bl, double bp)
/*
**  - - - - - - -
**   e r a P a s
**  - - - - - - -
**
**  Position-angle from spherical coordinates.
**
**  Given:
**     al     double     longitude of point A (e.g. RA) in radians
**     ap     double     latitude of point A (e.g. Dec) in radians
**     bl     double     longitude of point B
**     bp     double     latitude of point B
**
**  Returned (function value):
**            double     position angle of B with respect to A
**
**  Notes:
**
**  1) The result is the bearing (position angle), in radians, of point
**     B with respect to point A.  It is in the range -pi to +pi.  The
**     sense is such that if B is a small distance "east" of point A,
**     the bearing is approximately +pi/2.
**
**  2) Zero is returned if the two points are coincident.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraSepp(double a[3], double b[3])
/*
**  - - - - - - - -
**   e r a S e p p
**  - - - - - - - -
**
**  Angular separation between two p-vectors.
**
**  Given:
**     a      double[3]    first p-vector (not necessarily unit length)
**     b      double[3]    second p-vector (not necessarily unit length)
**
**  Returned (function value):
**            double       angular separation (radians, always positive)
**
**  Notes:
**
**  1) If either vector is null, a zero result is returned.
**
**  2) The angular separation is most simply formulated in terms of
**     scalar product.  However, this gives poor accuracy for angles
**     near zero and pi.  The present algorithm uses both cross product
**     and dot product, to deliver full accuracy whatever the size of
**     the angle.
**
**  Called:
**     eraPxp       vector product of two p-vectors
**     eraPm        modulus of p-vector
**     eraPdp       scalar product of two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraSeps(double al, double ap, double bl, double bp)
/*
**  - - - - - - - -
**   e r a S e p s
**  - - - - - - - -
**
**  Angular separation between two sets of spherical coordinates.
**
**  Given:
**     al     double       first longitude (radians)
**     ap     double       first latitude (radians)
**     bl     double       second longitude (radians)
**     bp     double       second latitude (radians)
**
**  Returned (function value):
**            double       angular separation (radians)
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraSepp      angular separation between two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraC2s(double p[3], double *theta, double *phi)
/*
**  - - - - - - - -
**   e r a C 2 s
**  - - - - - - - -
**
**  P-vector to spherical coordinates.
**
**  Given:
**     p      double[3]    p-vector
**
**  Returned:
**     theta  double       longitude angle (radians)
**     phi    double       latitude angle (radians)
**
**  Notes:
**
**  1) The vector p can have any magnitude; only its direction is used.
**
**  2) If p is null, zero theta and phi are returned.
**
**  3) At either pole, zero theta is returned.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraP2s(double p[3], double *theta, double *phi, double *r)
/*
**  - - - - - - - -
**   e r a P 2 s
**  - - - - - - - -
**
**  P-vector to spherical polar coordinates.
**
**  Given:
**     p        double[3]    p-vector
**
**  Returned:
**     theta    double       longitude angle (radians)
**     phi      double       latitude angle (radians)
**     r        double       radial distance
**
**  Notes:
**
**  1) If P is null, zero theta, phi and r are returned.
**
**  2) At either pole, zero theta is returned.
**
**  Called:
**     eraC2s       p-vector to spherical
**     eraPm        modulus of p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPv2s(double pv[2][3],
             double *theta, double *phi, double *r,
             double *td, double *pd, double *rd)
/*
**  - - - - - - - -
**   e r a P v 2 s
**  - - - - - - - -
**
**  Convert position/velocity from Cartesian to spherical coordinates.
**
**  Given:
**     pv       double[2][3]  pv-vector
**
**  Returned:
**     theta    double        longitude angle (radians)
**     phi      double        latitude angle (radians)
**     r        double        radial distance
**     td       double        rate of change of theta
**     pd       double        rate of change of phi
**     rd       double        rate of change of r
**
**  Notes:
**
**  1) If the position part of pv is null, theta, phi, td and pd
**     are indeterminate.  This is handled by extrapolating the
**     position through unit time by using the velocity part of
**     pv.  This moves the origin without changing the direction
**     of the velocity component.  If the position and velocity
**     components of pv are both null, zeroes are returned for all
**     six results.
**
**  2) If the position is a pole, theta, td and pd are indeterminate.
**     In such cases zeroes are returned for all three.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraS2c(double theta, double phi, double c[3])
/*
**  - - - - - - -
**   e r a S 2 c
**  - - - - - - -
**
**  Convert spherical coordinates to Cartesian.
**
**  Given:
**     theta    double       longitude angle (radians)
**     phi      double       latitude angle (radians)
**
**  Returned:
**     c        double[3]    direction cosines
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraS2p(double theta, double phi, double r, double p[3])
/*
**  - - - - - - -
**   e r a S 2 p
**  - - - - - - -
**
**  Convert spherical polar coordinates to p-vector.
**
**  Given:
**     theta   double       longitude angle (radians)
**     phi     double       latitude angle (radians)
**     r       double       radial distance
**
**  Returned:
**     p       double[3]    Cartesian coordinates
**
**  Called:
**     eraS2c       spherical coordinates to unit vector
**     eraSxp       multiply p-vector by scalar
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraS2pv(double theta, double phi, double r,
             double td, double pd, double rd,
             double pv[2][3])
/*
**  - - - - - - - - 
**   e r a S 2 p v
**  - - - - - - - - 
**
**  Convert position/velocity from spherical to Cartesian coordinates.
**
**  Given:
**     theta    double          longitude angle (radians)
**     phi      double          latitude angle (radians)
**     r        double          radial distance
**     td       double          rate of change of theta
**     pd       double          rate of change of phi
**     rd       double          rate of change of r
**
**  Returned:
**     pv       double[2][3]    pv-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraPdp(double a[3], double b[3])
/*
**  - - - - - - - -
**   e r a P d p
**  - - - - - - - -
**
**  p-vector inner (=scalar=dot) product.
**
**  Given:
**     a      double[3]     first p-vector
**     b      double[3]     second p-vector
**
**  Returned (function value):
**            double        a . b
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
double eraPm(double p[3])
/*
**  - - - - - -
**   e r a P m
**  - - - - - -
**
**  Modulus of p-vector.
**
**  Given:
**     p      double[3]     p-vector
**
**  Returned (function value):
**            double        modulus
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPmp(double a[3], double b[3], double amb[3])
/*
**  - - - - - - -
**   e r a P m p
**  - - - - - - -
**
**  P-vector subtraction.
**
**  Given:
**     a        double[3]      first p-vector
**     b        double[3]      second p-vector
**
**  Returned:
**     amb      double[3]      a - b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPn(double p[3], double *r, double u[3])
/*
**  - - - - - -
**   e r a P n
**  - - - - - -
**
**  Convert a p-vector into modulus and unit vector.
**
**  Given:
**     p        double[3]      p-vector
**
**  Returned:
**     r        double         modulus
**     u        double[3]      unit vector
**
**  Notes:
**
**  1) If p is null, the result is null.  Otherwise the result is a unit
**     vector.
**
**  2) It is permissible to re-use the same array for any of the
**     arguments.
**
**  Called:
**     eraPm        modulus of p-vector
**     eraZp        zero p-vector
**     eraSxp       multiply p-vector by scalar
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPpp(double a[3], double b[3], double apb[3])
/*
**  - - - - - - - -
**   e r a P p p
**  - - - - - - - -
**
**  P-vector addition.
**
**  Given:
**     a        double[3]      first p-vector
**     b        double[3]      second p-vector
**
**  Returned:
**     apb      double[3]      a + b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPpsp(double a[3], double s, double b[3], double apsb[3])
/*
**  - - - - - - - -
**   e r a P p s p
**  - - - - - - - -
**
**  P-vector plus scaled p-vector.
**
**  Given:
**     a      double[3]     first p-vector
**     s      double        scalar (multiplier for b)
**     b      double[3]     second p-vector
**
**  Returned:
**     apsb   double[3]     a + s*b
**
**  Note:
**     It is permissible for any of a, b and apsb to be the same array.
**
**  Called:
**     eraSxp       multiply p-vector by scalar
**     eraPpp       p-vector plus p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvdpv(double a[2][3], double b[2][3], double adb[2])
/*
**  - - - - - - - - -
**   e r a P v d p v
**  - - - - - - - - -
**
**  Inner (=scalar=dot) product of two pv-vectors.
**
**  Given:
**     a        double[2][3]      first pv-vector
**     b        double[2][3]      second pv-vector
**
**  Returned:
**     adb      double[2]         a . b (see note)
**
**  Note:
**
**     If the position and velocity components of the two pv-vectors are
**     ( ap, av ) and ( bp, bv ), the result, a . b, is the pair of
**     numbers ( ap . bp , ap . bv + av . bp ).  The two numbers are the
**     dot-product of the two p-vectors and its derivative.
**
**  Called:
**     eraPdp       scalar product of two p-vectors
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvm(double pv[2][3], double *r, double *s)
/*
**  - - - - - - -
**   e r a P v m
**  - - - - - - -
**
**  Modulus of pv-vector.
**
**  Given:
**     pv     double[2][3]   pv-vector
**
**  Returned:
**     r      double         modulus of position component
**     s      double         modulus of velocity component
**
**  Called:
**     eraPm        modulus of p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvmpv(double a[2][3], double b[2][3], double amb[2][3])
/*
**  - - - - - - - - -
**   e r a P v m p v
**  - - - - - - - - -
**
**  Subtract one pv-vector from another.
**
**  Given:
**     a       double[2][3]      first pv-vector
**     b       double[2][3]      second pv-vector
**
**  Returned:
**     amb     double[2][3]      a - b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Called:
**     eraPmp       p-vector minus p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvppv(double a[2][3], double b[2][3], double apb[2][3])
/*
**  - - - - - - - - -
**   e r a P v p p v
**  - - - - - - - - -
**
**  Add one pv-vector to another.
**
**  Given:
**     a        double[2][3]      first pv-vector
**     b        double[2][3]      second pv-vector
**
**  Returned:
**     apb      double[2][3]      a + b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Called:
**     eraPpp       p-vector plus p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvu(double dt, double pv[2][3], double upv[2][3])
/*
**  - - - - - - -
**   e r a P v u
**  - - - - - - -
**
**  Update a pv-vector.
**
**  Given:
**     dt       double           time interval
**     pv       double[2][3]     pv-vector
**
**  Returned:
**     upv      double[2][3]     p updated, v unchanged
**
**  Notes:
**
**  1) "Update" means "refer the position component of the vector
**     to a new date dt time units from the existing date".
**
**  2) The time units of dt must match those of the velocity.
**
**  3) It is permissible for pv and upv to be the same array.
**
**  Called:
**     eraPpsp      p-vector plus scaled p-vector
**     eraCp        copy p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvup(double dt, double pv[2][3], double p[3])
/*
**  - - - - - - - -
**   e r a P v u p
**  - - - - - - - -
**
**  Update a pv-vector, discarding the velocity component.
**
**  Given:
**     dt       double            time interval
**     pv       double[2][3]      pv-vector
**
**  Returned:
**     p        double[3]         p-vector
**
**  Notes:
**
**  1) "Update" means "refer the position component of the vector to a
**     new date dt time units from the existing date".
**
**  2) The time units of dt must match those of the velocity.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPvxpv(double a[2][3], double b[2][3], double axb[2][3])
/*
**  - - - - - - - - -
**   e r a P v x p v
**  - - - - - - - - -
**
**  Outer (=vector=cross) product of two pv-vectors.
**
**  Given:
**     a        double[2][3]      first pv-vector
**     b        double[2][3]      second pv-vector
**
**  Returned:
**     axb      double[2][3]      a x b
**
**  Notes:
**
**  1) If the position and velocity components of the two pv-vectors are
**     ( ap, av ) and ( bp, bv ), the result, a x b, is the pair of
**     vectors ( ap x bp, ap x bv + av x bp ).  The two vectors are the
**     cross-product of the two p-vectors and its derivative.
**
**  2) It is permissible to re-use the same array for any of the
**     arguments.
**
**  Called:
**     eraCpv       copy pv-vector
**     eraPxp       vector product of two p-vectors
**     eraPpp       p-vector plus p-vector
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraPxp(double a[3], double b[3], double axb[3])
/*
**  - - - - - - -
**   e r a P x p
**  - - - - - - -
**
**  p-vector outer (=vector=cross) product.
**
**  Given:
**     a        double[3]      first p-vector
**     b        double[3]      second p-vector
**
**  Returned:
**     axb      double[3]      a x b
**
**  Note:
**     It is permissible to re-use the same array for any of the
**     arguments.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraS2xpv(double s1, double s2, double pv[2][3], double spv[2][3])
/*
**  - - - - - - - - -
**   e r a S 2 x p v
**  - - - - - - - - -
**
**  Multiply a pv-vector by two scalars.
**
**  Given:
**     s1     double         scalar to multiply position component by
**     s2     double         scalar to multiply velocity component by
**     pv     double[2][3]   pv-vector
**
**  Returned:
**     spv    double[2][3]   pv-vector: p scaled by s1, v scaled by s2
**
**  Note:
**     It is permissible for pv and spv to be the same array.
**
**  Called:
**     eraSxp       multiply p-vector by scalar
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraSxp(double s, double p[3], double sp[3])
/*
**  - - - - - - -
**   e r a S x p
**  - - - - - - -
**
**  Multiply a p-vector by a scalar.
**
**  Given:
**     s      double        scalar
**     p      double[3]     p-vector
**
**  Returned:
**     sp     double[3]     s * p
**
**  Note:
**     It is permissible for p and sp to be the same array.
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```

```
void eraSxpv(double s, double pv[2][3], double spv[2][3])
/*
**  - - - - - - - -
**   e r a S x p v
**  - - - - - - - -
**
**  Multiply a pv-vector by a scalar.
**
**  Given:
**     s       double          scalar
**     pv      double[2][3]    pv-vector
**
**  Returned:
**     spv     double[2][3]    s * pv
**
**  Note:
**     It is permissible for pv and spv to be the same array
**
**  Called:
**     eraS2xpv     multiply pv-vector by two scalars
**
**  Copyright (C) 2013-2018, NumFOCUS Foundation.
**  Derived, with permission, from the SOFA library.  See notes at end of file.
*/
```