SUBROUTINES FOR CALCULATING PROPER DIRECTION
by L Lindegren

A subroutine package 'proper.f' has been written, by which the proper
direction of a star or planet is calculated. It consists of 9 subprograms
arranged as follows:

```
proper
  |   |
sun  earth  geosat  codirp  codirs  natdir  prpdir
    /  |    |
satorb  fourie
```

No reference is made to external routines or files, except that geocentric
orbital data for the satellite are supposed to be available in a special
disk file 'satorb'. (A separate program generating test data on such a
file is also available.) The procedure is invoked through the statement

```
CALL proper(T, opar,   u, ierr)
```

in which T is the time in seconds from J1988.0 TDT, opar(1:20) is a double-
precision array of object parameters (see below), and u(1:3) a DP array
in which the proper direction with respect to N is returned. The error
flag (ierr) becomes negative if T is outside the permitted range; normal
return is ierr = 0.

A very brief description of the subprograms used by subroutine
proper is given below. For details, see program listings at the end of
this note.
SUBROUTINE sun : calculates the barycentric position of the Sun \( \mathbf{h}_S \) at the given time \( T \) (see Lindegren, 1984 June 15)

SUBROUTINE earth : calculates the barycentric position and velocity of the Earth \( \mathbf{h}_E, \mathbf{v}_E \) at the given time \( T \) (see Lindegren, 1984 June 8)

SUBROUTINE geosat : calculates the geocentric position and velocity of the satellite (observer), \( \mathbf{g}_0 \) and \( \mathbf{g}_0' \), at the given time \( T \). Fourier coefficients etc for these are read from the disk file 'satorb'.

SUBROUTINE codirp : calculates the coordinate direction \( \mathbf{\hat{u}} \) to a planet at time \( T \), given object parameters in opar()

SUBROUTINE codirs : calculates the coordinate direction \( \mathbf{\hat{u}} \) to a star at time \( T \) from object parameters in opar()

SUBROUTINE natdir : calculates the natural direction \( \mathbf{\hat{u}} \) of the object by applying gravitational deflexion by the Sun and Earth to the coordinate direction

SUBROUTINE prpdir : calculates the proper direction \( \mathbf{u} \) of the object by Lorentz transformation to the proper frame

FUNCTION fourie : calculates the sum of a Fourier series

The array of object parameters, opar(1:20), is an attempt to summarize the state of knowledge on any particular object, stellar or planetary, in a standardized form suitable for rapid calculations. The length of this array (20) and the precise contents of it, as described below, are only preliminary. Concerning the parameters for planetary objects (asteroids), it should be remembered that a given planet is formally regarded as a different object, with its own ID number, for each time it enters the FOV. Provisionally, it is furthermore assumed that planetary objects are given ID numbers in excess of 90,000,000, while stellar objects have ID numbers below this limit.
Contents of the object parameters array opar(i), i = 1 to 20

<table>
<thead>
<tr>
<th>i</th>
<th>opar(i) [stellar object]</th>
<th>opar(i) [planetary object]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>object ID number ($&lt; 9 \cdot 10^7$)</td>
<td>object ID number ($&gt; 9 \cdot 10^7$)</td>
</tr>
<tr>
<td>2</td>
<td>reference epoch ($T_o$)</td>
<td>reference epoch ($T_o$)</td>
</tr>
<tr>
<td>3</td>
<td>$\cos \alpha_o$ ($\alpha_o = \text{barycentric R.A. at } T_o$)</td>
<td>x-component of $\vec{u}$ at $T_o$</td>
</tr>
<tr>
<td>4</td>
<td>$\sin \alpha_o$</td>
<td>y-component of $\vec{u}$ at $T_o$</td>
</tr>
<tr>
<td>5</td>
<td>$\cos \delta_o$ ($\delta_o = \text{barycentric Dec. at } T_o$)</td>
<td>z-component of $\vec{u}$ at $T_o$</td>
</tr>
<tr>
<td>6</td>
<td>$\sin \delta_o$</td>
<td>$\dot{x}$-component of $\vec{u}$ at $T_o$ [s⁻¹]</td>
</tr>
<tr>
<td>7</td>
<td>$\mu_\alpha \cos \delta_o$ [rad s⁻¹]</td>
<td>$\dot{y}$-component of $\vec{u}$ at $T_o$ [s⁻¹]</td>
</tr>
<tr>
<td>8</td>
<td>$\mu_\delta$ [rad s⁻¹]</td>
<td>$\dot{z}$-component of $\vec{u}$ at $T_o$ [s⁻¹]</td>
</tr>
<tr>
<td>9</td>
<td>$H$ [rad]</td>
<td>$p = \text{inverse distance at } T_o$ [m⁻¹]</td>
</tr>
<tr>
<td>10</td>
<td>$\rho = \text{radial velocity}$ [m s⁻¹]</td>
<td>$\dot{p} = \text{derivative of } p$ [m⁻¹ s⁻¹]</td>
</tr>
<tr>
<td>11</td>
<td>$\Delta \alpha \cos \delta_o$ [rad]</td>
<td>(TBD)</td>
</tr>
<tr>
<td>12</td>
<td>$\Delta \delta$ [rad]</td>
<td>(TBD)</td>
</tr>
<tr>
<td>13-18</td>
<td>(TBD)</td>
<td>(TBD)</td>
</tr>
<tr>
<td>19</td>
<td>$H = \frac{1}{2}(B + V)$ [mag]</td>
<td>$H = \frac{1}{2}(B + V)$ at $T_o$ [mag]</td>
</tr>
<tr>
<td>20</td>
<td>$B-V$ [mag]</td>
<td>$B-V$ [mag]</td>
</tr>
</tbody>
</table>

NOTE: opar(13) - (20) are not used in the present subroutines.
The circumstance that a planetary object 'exists' for only the 20 seconds or so it takes to travel across the FOV means that its coordinate direction and inverse distance (which are the quantities needed to calculate the natural direction) are adequately represented by \(\tilde{\mathbf{u}}(T_o), \tilde{\mathbf{u}}(T_o), p(T_o),\) and \(\hat{p}(T_o),\) where \(T_o =\) reference epoch such that \(|T - T_o| \lesssim 10\) s, and \(p =\) inverse distance. It is assumed that a special procedure (yet to be written) will recognize a planet the first time it is observed in a certain FOV crossing, assign its ID number, compute the object parameters for the entire crossing, and incorporate these with the current object catalogue. This could be done immediately prior to the IDT Preprocessing.

In writing the present subroutines, special care was taken in order to get the basic transformations in a form which is both rigorous and computationally efficient. It was also assumed that successive calls are usually made with \(T\) within the same 12\(^{th}\) interval (say), or even with exactly the same \(T\) for several objects observed in the same frame. Thus, the calculation of ephemeris data \((b_S, b_B, \dot{b}_B, g_0, \dot{g}_0)\) is automatically suppressed if these were already obtained in the last call (with the same \(T\)). This is quite important, as it turns out that subroutine 'earth' is by far the most time-consuming part of the calculation:

<table>
<thead>
<tr>
<th>Subroutine called by 'proper'</th>
<th>Execution time [ms] on HP9000</th>
</tr>
</thead>
<tbody>
<tr>
<td>sun</td>
<td>0.5</td>
</tr>
<tr>
<td>earth</td>
<td>36.2</td>
</tr>
<tr>
<td>geosat (provided 'satorb' need not be referenced)</td>
<td>6.8</td>
</tr>
<tr>
<td>codir, codirp</td>
<td>0.9 (star), 0.1 (planet)</td>
</tr>
<tr>
<td>natdir</td>
<td>1.4 (star), 2.5 (planet)</td>
</tr>
<tr>
<td>prpdir</td>
<td>1.2</td>
</tr>
<tr>
<td>total for 'proper', same (T):</td>
<td>3.6 (star), 3.9 (planet)</td>
</tr>
<tr>
<td>new (T):</td>
<td>47.6 (star), 48.1 (planet)</td>
</tr>
</tbody>
</table>

With about 4 objects per frame, the average execution time would be some 15 ms per object.

If necessary, the execution time for calculating the Earth ephemeris could be brought down to about 2.3 ms, by using a Chebyshev representation (degree = 8, maximum error about 1 m and 1 mm/s) on each 10 day interval. Subroutines for calculating the Chebyshev coefficients and evaluating the polynomials exist on our computer.
SUBROUTINE proper(T, opar, u, ierr)

HIPPARCOS - NDAC - General routines - Proper direction

Calculates the proper direction to an object at a given time, using a set of 'object parameters' contained in the array opar().

Input:
- T - time from J1988.0 TDT [s]
- opar(1) - object ID number
  < 9d7 for a stellar object
  > 9d7 for a planetary object
- opar(2) - reference epoch (TD) [s]
- opar(i), i=3-18 - these are different for stars and planets, see SUBROUTINE codirs and codirp, respectively, for explanation
- opar(19) - H-magnitude of object
- opar(20) - B-V colour index

Output:
- u(i), i=1-3 - proper direction to object
- ierr = 0 on normal return
  < 0 on error condition:
    ierr = -DCBA, where
    A = 1 if error in SUBROUTINE sun
    B = 1 if error in SUBROUTINE earth
    C = 1 if error in SUBROUTINE geosat
    D = 1 if error in SUBROUTINE codirp

Execution time per call (HP9000):
- 48 ms if T is not the same as on previous call
- 3.6 ms (star) if T has not been changed since the last call
- 3.9 ms (planet)

L Lindegren 1984 July 26

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION opar(20), u(3), uco(3), unat(3)
DIMENSION be(3), bedot(3), bo(3), bodot(3), bs(3), go(3),
    godot(3), ho(3)
SAVE Tprev, bo, bodot, go, ho
DATA Tprev/-1d8/

If this is the first call with the present value of T, then calculate and save:
- bo() = barycentric position of observer
- bodot() = barycentric velocity of observer
- go() = geocentric position of observer
- ho() = heliocentric position of observer

IF (T .NE. Tprev) THEN
    CALL sun(T, bs, ierr1)
    CALL earth(T, be, bedot, ierr2)
    CALL geosat(T, go, godot, ierr3)
DO 100 i = 1, 3
    bo(i) = bs(i) + go(i)
    bodot(i) = bedot(i) + godot(i)
    ho(i) = bo(i) - bs(i)
100 CONTINUE
Tprev = T
ENDIF
SUBROUTINE proper, ct'd

To calculate the coordinate direction, use codirp for a planet; codirs for a star

IF (opar(1) .GT. 9d7) THEN
    CALL codirp(T, opar, uco, p, ierr4)
ELSE
    CALL codirs(T, opar, bo, uco, p)
ENDIF

Calculate natural direction unat():

CALL natdir(uco, p, ho, go, unat)

Calculate proper direction u():

CALL prpdir(unat, bodot, u)

ierr = ierr1 + 10*(ierr2 + 10*(ierr3 + 10*ierr4))

RETURN
END
**SUBROUTINE** geosat(T, go, godot, ierr)

**HIPPARCOS - NDAC - General routines - Satellite ephemeris**

Calculates the geocentric equatorial position and velocity of the satellite at a given time.

Orbital parameters for contiguous time intervals (of some 12h) are supposed to be available in the file 'satorb'.

**Input:**

\[ T = \text{time in seconds from J1988.0 (12h TDT)} \]

**Output:**

\[ \text{go}(i) = \text{geocentric rectangular equatorial coordinates [m]} \]
\[ \text{godot}(i) = \text{geocentric equatorial velocity components [m/s]} \]
\[ \text{ierr} = \text{0 on normal return, -1 if } T \text{ is outside the range defined by 'satorb'} \]

**NOTE:** This routine is in a very preliminary shape

**Execution time:** 6.8 ms if \( T \) is in same interval as on prev. call
1000 ms if \( T \) is random among 100 intervals

L Lindegren 1984 July 24

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION go(3), godot(3), p(64)
SAVE p, ireo
DATA p/DdO, -1d0, 62*d0/1, ireo/0/
1 CONTINUE
IF ((p(1) .LE. T) .AND. (T .LE. p(2))) THEN

\[ T \text{ is in an interval for which the parameters have been saved from a previous call:} \]

\[ DT = T - p(3) \]
\[ f = DT*p(4) \]
\[ D0 100 \quad j = 1, 3 \]
\[ i0 = 10*j-4 \]
\[ \text{go}(j) = p(i0) + p(i0-1)*DT + \text{fourie}(f, 4, p(i0+1), p(i0+5)) \]
\[ i1 = i0+30 \]
\[ \text{godot}(j) = p(i1) + p(i1-1)*DT + \text{fourie}(f, 4, p(i1+1), p(i1+5)) \]

100 CONTINUE
ierr = 0
ELSE
SUBROUTINE geosat, ct'd

T is not in the same interval as on previous call:
search file 'satorb' for orbital parameters of relevant interval

lu = 10
OPEN(UNIT=lu, FILE='satorb', ACCESS='DIRECT', RECL=8*64)
199 CONTINUE
irecd = irec
200 CONTINUE
irec = irec + 1
READ(UNIT=lu, REC=irec, ERR=299) p
IF ((T .LT. p(1)) .OR. (T .GT. p(2))) GOTO 200
CLOSE(UNIT=lu)
GOTO 1
299 CONTINUE
IF (irecd .NE. 0) THEN
  irec = 0
  GOTO 199
ELSE
  ierr = -1
  irecd = 0
  DO 210 i = 1, 3
    go(i) = 0d0
    godot(i) = 0d0
  210 CONTINUE
ENDIF
CLOSE(UNIT=lu)
ENDIF

RETURN
END
SUBROUTINE codirp(T, opar, uco, p, ierr)

HIPPARCOS - NDAC - General routines - Coordinate direction to planet

Calculates the coordinate direction to a planet at a given time

**Input:**
- T: time from J1988.0 TDT [s]
- opar(1): object ID number (> 9d7 for a planet)
- opar(2): reference epoch (TO) [s]
- opar(3)-(5): coordinate direction at TO [s]
- opar(6)-(8): derivative of coordinate direction [1/s]
- opar(9): inverse distance at TO [m]
- opar(10): derivative of inverse distance [1/m/s]
- opar(11)-(20): (not used by this subroutine)

**Output:**
- uco(i), i=1-3: coordinate direction at time T [1/m]
- p: inverse distance at time T [1/m]
- ierr: = 0 on normal return, -1 if abs(T-TO) > 100 s

Execution time per call (HP9000): 0.13 ms

L Lindegren 1984 July 26

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION opar(20), uco(3)

d = T - opar(2)
uc0(1) = opar(3) + d*opar(6)
uc0(2) = opar(4) + d*opar(7)
uc0(3) = opar(5) + d*opar(8)
p = opar(9) + d*opar(10)

ierr = 0
IF (dabs(d) .GT. 1d2) ierr = -1

RETURN
END
SUBROUTINE oodirs(T, opar, bo, uco, p)

HIPPARCOS - NDAC - General routines - Coordinate direction to a star

Calculates the coordinate direction towards a star at a given time from its astrometric parameters and the observer's barycentric coordinates.

Input:
- T [s] - time from J1988.0 TDT
- opar(1) [s] - object ID number (< 9d7 for a star)
- opar(2) - reference epoch (T0)
- opar(3) - cos(RAO); RAO = barycentric R.A. at T0
- opar(4) - sin(RAO)
- opar(5) - cos(Dec0); Dec0 = barycentric Dec. at T0
- opar(6) - sin(Dec0)
- opar(7) - p.m. in R.A.; times cos(Dec0) [rad/s]
- opar(8) - p.m. in Dec. [rad/s]
- opar(9) - parallax [rad]
- opar(10) - radial velocity [m/s]
- opar(11) - correction in R.A.; times cos(Dec0) [rad]
- opar(12) - correction in Dec. [rad]
- opar(13)-(20) - (not used by this subroutine)
- bo(i), i=1-3 - barycentric coordinates of observer [m]

Output:
- uco(i), i=1-3 - coordinate direction
- p - inverse distance ( = 0)

Execution time per call (HP9000): 0.93 ms

L Lindegré 1984 July 26

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION opar(20), bo(3), uco(3)
PARAMETER (au = 1.49597870d11)

d = T - opar(2)
p = opar(9)/au
q = 1d0 + d*p*opar(10)

da = opar(11) + d*opar(7)
 dd = opar(12) + d*opar(8)

uco(1) = q*opar(5)*opar(3)-da*opar(4)-dd*opar(6)*opar(3)-p*bo(1)
uco(2) = q*opar(5)*opar(4)+da*opar(3)-dd*opar(6)*opar(4)-p*bo(2)
uco(3) = q*opar(6) +dd*opar(5) -p*bo(3)

unorm = 1d0/dsqr(uco(1)**2 + uco(2)**2 + uco(3)**2)

uco(1) = unorm*uco(1)
uco(2) = unorm*uco(2)
uco(3) = unorm*uco(3)

p = 0d0

RETURN
END
SUBROUTINE natau(uco, p, ho, go, unat)

HIPPARCOS - NDAC - General routines - Gravitational deflexion

Calculates the natural direction to an object by applying gravitational deflexion by the Sun and Earth. The object may be at a finite distance (p > 0).

Source: Vectorial Astrometry, (2.5.5) (modified)

Input:
  uco(i), i=1,2,3 - coordinate direction to object
  p - inverse distance to object [1/m].
  p may be set = 0 for non-Solar System objects, with some saving in computing time.
  ho(i), i=1,2,3 - heliocentric coordinate of observer [m]
  go(i), i=1,2,3 - geocentric coordinate of observer [m]

Output:
  unat(i), i=1,2,3 - natural direction to object

Execution time per call (HP9000): 1.4 ms with p = 0
                                   2.5 ms with p > 0

L Lindegren 1984 July 25

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION uco(3), ho(3), go(3), unat(3)
PARAMETER (o = 299792458d0, c2 = c*o,
          , 6c = 1.32712438d20, Ge = 3.986005d14,
          , s = 2d0*6c/c2, e = 2d0*Ge/o2)

h = dsqrt(ho(1)**2 + ho(2)**2 + ho(3)**2)
g = dsqrt(go(1)**2 + go(2)**2 + go(3)**2)

(p = 0) for a star, (p > 0) for a planet:

IF (p .LE. 0d0) THEN
  qh = h
  qg = g
ELSE
  qh = h*(h*p+dsqrt((uco(1)+ho(1)*p)**2+(uco(2)+ho(2)*p)**2 +
                   (uco(3)+ho(3)*p)**2))
  qg = g*(g*p+dsqrt((uco(1)+go(1)*p)**2+(uco(2)+go(2)*p)**2 +
                   (uco(3)+go(3)*p)**2))
ENDIF

as = s/(h*qh + uco(1)*ho(1) + uco(2)*ho(2) + uco(3)*ho(3))
ae = e/(g*qg + uco(1)*go(1) + uco(2)*go(2) + uco(3)*go(3))

D1 = ho(1)*as + go(1)*ae
D2 = ho(2)*as + go(2)*ae
D3 = ho(3)*as + go(3)*ae

AD = 1d0 - uco(1)*D1 - uco(2)*D2 - uco(3)*D3

unat(1) = uco(1)*AD + D1
unat(2) = uco(2)*AD + D2
unat(3) = uco(3)*AD + D3

RETURN
END
SUBROUTINE prpdire(unat, v, u)

* HIPPARCOS - NDAC - General routines - Stellar aberration *

* Calculates the proper direction to an object by applying stellar *
* aberration (Lorentz transformation). *

* Source: Vectorial Astrometry, (2.5.8) (modified) *

* Input: *
  * unat(i), i=1,2,3 - natural direction to object *
  * v(i), i=1,2,3 - velocity of observer [m/s] *

* Output: *
  * u(i), i=1,2,3 - proper direction to object *

* Execution time per call (HP9000): 1.2 ms *

L Lindegren 1984 July 25

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION unat(3), v(3), u(3)
PARAMETER (c = 299792458d0, c2 = c*c)

e = dsqrt(c2 - v(1)**2 - v(2)**2 - v(3)**2)
f = (1d0 + (unat(1)*v(1)+unat(2)*v(2)+unat(3)*v(3))/(c+e))/e

u(1) = unat(1) + f*v(1)
u(2) = unat(2) + f*v(2)
u(3) = unat(3) + f*v(3)

unorm = 1d0/dsqr(u(1)**2 + u(2)**2 + u(3)**2)

u(1) = unorm*u1)
u(2) = unorm*u(2)
u(3) = unorm*u(3)

RETURN
END
DOUBLE PRECISION FUNCTION fourie(x, n, a, b)

HIPPARCOS - NDAC - General routines - Fourier series

Evaluates the Fourier series

\[ \text{fourie} = a(1)\cos(x) + a(2)\cos(2x) + \ldots + a(n)\cos(nx) + b(1)\sin(x) + b(2)\sin(2x) + \ldots + b(n)\sin(nx) \]

using Clenshaw's algorithm.

Restriction: \( 0 \leq n \leq 20 \) (NO CHECK IS MADE!)

Execution time: approx 0.74 + 0.075*n ms per call

L. Lindegren 1984 July 23

IMPLICIT DOUBLE PRECISION (a-h,o-z)
DIMENSION c(22), s(22), a(n), b(n)
y = dcos(x)
z = 2d0*y
k2 = n + 2
k1 = n + 1
c(k2) = 0d0
s(k2) = 0d0
c(k1) = 0d0
s(k1) = 0d0
DO 100 k = n, 1, -1
   c(k) = z*c(k1) - c(k2) + a(k)
   s(k) = z*s(k1) - s(k2) + b(k)
   k2 = k2 - 1
   k1 = k1 - 1
100 CONTINUE

fourie = y*c(1) - c(2) + s(1)*d*sin(x)

RETURN
END