

Computer Methods for Converting Between True-of-Date and EME2000 Cartesian Coordinates

This memo describes a Matlab script called `eme_tod.m` and a Fortran computer program called `eme_tod.f` that can be used to transform between Earth-centered inertial (ECI) true-of-date Cartesian coordinates and the corresponding coordinates in the Earth mean equator and equinox of J2000 (EME2000) coordinate system. The software implemented within these two applications is based on several SPICE library routines provided by JPL (<http://naif.jpl.nasa.gov/naif/>).

The following is a brief description of each coordinate system. Each reference system is a right-handed Cartesian set of three orthogonal axes.

ECI True-of-date Coordinate System

The origin of the true-of-date ECI inertial coordinate system is the geocenter and the fundamental plane is the Earth's true-of-date equator. The x-axis of this system is aligned with the true-of-date vernal equinox, the y-axis is advanced 90 degrees along the Earth's equator, and the z-axis is along the true-of-date spin axis of the Earth.

EME2000 Coordinate System

The origin of the ECI inertial coordinate system is the geocenter and the fundamental plane is the Earth's mean equator. The z-axis of this system is normal to the Earth's mean equator at epoch J2000, the x-axis is parallel to the vernal equinox of the Earth's mean orbit at epoch J2000 and the y-axis completes the right-handed coordinate system. The epoch J2000 is the Julian Ephemeris Date (JED) 2451545.0 (January 1, 2000, 12 hours ephemeris time).

Using the Software

The software is data-driven by a simple ASCII input file created by the user. To run the Fortran version of the program, simply type `eme_tod *.*` at a DOS command line, where `*.*` is the name of a compatible input data file. If you do not include an input file on the command line, the software will interactively ask you for the name of a data file with the following prompt;

```
program eme_tod

convert between true-of-date and eme2000 state vectors

please input the name of the data file
```

The Matlab version of the software will display a file manager window that will allow the user to select an input file. By default, the script display all file names with a `*.in` extension. However, the software will accept compatible data files with any filename extension.

Please note that the leap second data file `"naif0008.tls"` and frame definition data file `"earth_tod.tf"` must be available to the `eme_tod` computer program in order for the software to function properly.

Typical Data File

The following is a typical input data file for this program. This data file is based on the example given on pages 17-18 of AAS 06-134, "Implementation Issues Surrounding the New IAU Reference Systems for Astrodynamics". This program also provides the classical orbital elements and Keplerian orbital period for each set of state vectors. The user can input a value for the gravitational constant of the Earth which is used in the calculation of orbital elements. Please note that all input and output is metric.

Each data item within an input file is preceded by one or more lines of annotation text. Do not delete any of these annotation lines or increase or decrease the number of lines reserved for each comment. However, you may change them to reflect your own explanation. The annotation line also includes the correct units and when appropriate, the valid range of the input. ASCII text input is not case sensitive but must be spelled correctly.

```
*****
data file for eme_tod computer program
converts between true-of-date and eme2000 state vectors
*****

type of coordinate conversion
(1 = eme2000-to-true-of-date, 2 = true-of-date-to-eme2000)
-----
1

UTC epoch
-----
Apr 6, 2004 07:51:28.386009

x-component of position vector (kilometers)
-----
5.1025096000e+003

y-component of position vector (kilometers)
-----
+6.1230115200e+003

z-component of position vector (kilometers)
-----
+6.3781363000e+003

x-component of velocity vector (kilometers/second)
-----
-4.7432195996e+000

y-component of velocity vector (kilometers/second)
-----
+7.9053660026e-001

z-component of velocity vector (kilometers/second)
-----
+5.5337561903e+000

gravitational constant (km**3/sec**2)
-----
398600.4415
```

The following is the program output for this example.

```
eme2000-to-true-of-date conversion
=====
```

```
UTC epoch           Apr 6, 2004 07:51:28.386009
UTC Julian date     2453101.8274119
TDB Julian date     2453101.82815476
```

```
true-of-date state vector and orbital elements
-----
```

rx (km)	ry (km)	rz (km)	rmag (km)
0.5094514780D+04	0.6127366461D+04	0.6380344533D+04	0.1020820733D+05
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.4746088567D+01	0.7860772220D+00	0.5531931288D+01	0.7331134828D+01
sma (km)	eccentricity	inclination (deg)	argper (deg)
0.1637058685D+05	0.4249757137D+00	0.6309754662D+02	0.2139676547D+01
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.2628891413D+02	0.4235721450D+02	0.4449689105D+02	0.3474210292D+03

```
eme2000 state vector and orbital elements
-----
```

rx (km)	ry (km)	rz (km)	rmag (km)
0.5102509600D+04	0.6123011520D+04	0.6378136300D+04	0.1020820733D+05
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.4743219600D+01	0.7905366003D+00	0.5533756190D+01	0.7331134828D+01
sma (km)	eccentricity	inclination (deg)	argper (deg)
0.1637058685D+05	0.4249757137D+00	0.6310562739D+02	0.2116170060D+01
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.2624806701D+02	0.4235721450D+02	0.4447338456D+02	0.3474210292D+03

The following are brief definitions of information output by the software.

EME2000 = Earth mean equator and equinox of J2000

UTC = coordinated universal time

TDB = barycentric dynamical time

rx (km) = x-component of the position vector in kilometers

ry (km) = y-component of the position vector in kilometers
rz (km) = z-component of the position vector in kilometers
vx (kps) = x-component of the velocity vector in kilometers/second
vy (kps) = y-component of the velocity vector in kilometers/second
vz (kps) = z-component of the velocity vector in kilometers/second

sma (km) = semimajor axis in kilometers
eccentricity = orbital eccentricity (non-dimensional)
inclination (deg) = orbital inclination in degrees
argper (deg) = argument of perigee in degrees
raan (deg) = right ascension of the ascending node in degrees
true anomaly (deg) = true anomaly in degrees
arglat (deg) = argument of latitude in degrees
period (min) = orbital period in minutes

Matlab Source Code

The following is a listing of the Matlab source code. Calls to Spice functions are in bold font.

```
% eme_tod.m  
  
% this Matlab script converts between  
% true-of-date and eme2000 state vectors  
% using Matlab spice routines  
  
% February 26, 2007  
  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
  
clear all;  
  
% load leap seconds data  
cspace_furnsh('naif0008.tls');  
  
% load Earth true-of-date coordinate frame definition  
cspace_furnsh('earth_tod.tf');
```

```

% select data file

clc;

[filename, pathname] = uigetfile('*.in', 'Please select the data file to read');

% read data file contents

[fid, itype, utc_epoch, r, v, emu] = readdata(filename);

% compute time in ephemeris seconds since j2000

etime = cspice_str2et(utc_epoch);

if (itype == 1)
    % eme2000-to-true-of-date conversion

    txt1 = 'eme2000-to-true-of-date conversion';
    txt2 = '=====';

    r2000 = r;
    v2000 = v;

    % compute transformation matrix

    xform = cspice_sxform('J2000', 'EARTH_TOD', etime);

    for i = 1:1:3
        sv_2000(i) = r2000(i);

        sv_2000(i + 3) = v2000(i);
    end

    sv_tod = xform * sv_2000';

    rtod = sv_tod(1:3);

    vtod = sv_tod(4:6);
else
    % true-of-date-to-eme2000 conversion

    txt1 = 'true-of-date-to-eme2000 conversion';
    txt2 = '=====';

    rtod = r;
    vtod = v;

    for i = 1:1:3
        sv_tod(i) = rtod(i);

        sv_tod(i + 3) = vtod(i);
    end

    % compute transformation matrix

    xform = cspice_sxform('EARTH_TOD', 'J2000', etime);

    sv_2000 = xform * sv_tod';

```

```

    r2000 = sv_2000(1:3);

    v2000 = sv_2000(4:6);
end

% display results

disp(' ');

disp(txt1);
disp(txt2);

fprintf('\nUTC epoch          %s\n', utc_epoch);

jdstr = cspice_et2utc(etime, 'J', 12);

fprintf('\nUTC Julian date %s\n', jdstr(3:18));

fprintf('\n\ntrue-of-date state vector and orbital elements');
fprintf('\n-----\n');

svprintl(rtod, vtod);

oev_tod = eci2orbl (emu, rtod, vtod);

oeprintl(emu, oev_tod);

fprintf('\n\neme2000 state vector and orbital elements');
fprintf('\n-----\n');

svprintl(r2000, v2000);

oev_2000 = eci2orbl (emu, r2000, v2000);

oeprintl(emu, oev_2000);

disp(' ');

```

Fortran Source Code

The following is a listing of the Fortran source code. Calls to Spice subroutines are in bold font.

```

    program eme_tod

c     this Fortran program converts between
c     true-of-date and eme2000 state vectors
c     using Fortran spice routines

c     February 28, 2007

c     *****

    implicit none

```

```

character*(10) from, to, format
character*(40) txt1, txt2, inputfile$, utc_epoch, jdstr
double precision emu, eph_time
integer i, itype, prec
integer(2) istatus
c dimension arrays
double precision r(3), v(3), rtod(3), vtod(3)
double precision r2000(3), v2000(3), sv_2000(6), sv_tod(6)
double precision oev_tod(6), oev_2000(6), xform (6, 6)
c load leap seconds data
call ldpool('naif0008.tls')
c load Earth true-of-date coordinate frame definition
call ldpool('earth_tod.tf')
c select data file
c if present, use command line argument #1 for input file
call getarg(1, inputfile$, istatus)
if (istatus .eq. -1) then
c *****
c input filename not on command line
c request name of simulation definition input file
c *****
print *, ' '
print *, 'program eme_tod'
print *, ' '
print *,
& 'convert between true-of-date and eme2000 state vectors'
print *, ' '
print *,
& 'please input the name of the data file'
read (*, *) inputfile$
end if
c read data file contents
call readdata(inputfile$, itype, utc_epoch, r, v, emu)
c compute time in ephemeris seconds since j2000
call str2et(utc_epoch, eph_time)

```

```

c      if (itype .eq. 1) then
          eme2000-to-true-of-date conversion

          txt1 = 'eme2000-to-true-of-date conversion'
          txt2 = '=====

          do i =1 , 3
              r2000(i) = r(i)

              v2000(i) = v(i)
          end do

          do i = 1, 3
              sv_2000(i) = r2000(i)

              sv_2000(i + 3) = v2000(i)
          end do

c      compute transformation matrix

          from = 'J2000'

          to = 'EARTH_TOD'

          call sxform(from, to, eph_time, xform)

          call mxvg(xform, sv_2000, 6, 6, sv_tod)

          do i = 1, 3
              rtod(i) = sv_tod(i)

              vtod(i) = sv_tod(i + 3)
          end do
      else
c      true-of-date-to-eme2000 conversion

          txt1 = 'true-of-date-to-eme2000 conversion'
          txt2 = '=====

          do i = 1, 3
              rtod(i) = r(i)

              vtod(i) = v(i)
          end do

          do i = 1, 3
              sv_tod(i) = rtod(i)

              sv_tod(i + 3) = vtod(i)
          end do

c      compute transformation matrix

          from = 'EARTH_TOD'

          to = 'J2000'

          call sxform(from, to, eph_time, xform)

```

```

    call mxvg(xform, sv_tod, 6, 6, sv_2000)

    do i = 1, 3
        r2000(i) = sv_2000(i)

        v2000(i) = sv_2000(i + 3)
    end do

end if

c    display results

print *, ' '
print *, ' '
print *, txt1
print *, txt2
print *, ' '

print *, 'UTC epoch          ', utc_epoch

print *, ' '

call et2utc(eph_time, 'J', 15, jdstr)

print *, 'UTC Julian date    ', jdstr(3:18)
print *, ' '
print *, 'TDB Julian date ', 2451545.0d0 + eph_time / 86400.0d0

print *, ' '
print *, ' '
print *, 'true-of-date state vector and orbital elements'
print *, '-----'

call svprint1(rtod, vtod)

call eci2orb1 (emu, rtod, vtod, oev_tod)

call oeprint1(emu, oev_tod)

print *, ' '
print *, ' '
print *, 'eme2000 state vector and orbital elements'
print *, '-----'

call svprint1(r2000, v2000)

call eci2orb1 (emu, r2000, v2000, oev_2000)

call oeprint1(emu, oev_2000)

print *, ' '

end

```

Earth true-of-date frame definition

The contents of the Earth true-of-date frame definition file are as follows;

```
\begintext

Earth true-of-date frame definition

\begindata

FRAME_EARTH_TOD           = 398600
FRAME_398600_NAME         = 'EARTH_TOD'
FRAME_398600_CLASS        = 5
FRAME_398600_CLASS_ID    = 398600
FRAME_398600_CENTER       = 399
FRAME_398600_RELATIVE     = 'J2000'
FRAME_398600_DEF_STYLE    = 'PARAMETERIZED'
FRAME_398600_FAMILY       = 'TRUE_EQUATOR_AND_EQUINOX_OF_DATE'
FRAME_398600_PREC_MODEL   = 'EARTH_IAU_1976'
FRAME_398600_NUT_MODEL    = 'EARTH_IAU_1980'
FRAME_398600_ROTATION_STATE = 'INERTIAL'
```

Please consult the Spice documentation for additional information about frame definitions.

Algorithm Resources

Astronomical Algorithms, Jean Meeus, Willmann-Bell, Inc., 1991.

NOVAS (Naval Observatory Vector Astrometry Subroutines) software package, U.S. Naval Observatory, 1992.

Explanatory Supplement to the Astronomical Almanac, Edited by P. K. Seidelmann, University Science Books, 1992.

“JPL Planetary Ephemeris DE410”, E. M. Standish, JPL IOM 312.N-03-009, 24 April 2003.

“IERS Conventions (2003)”, IERS Technical Note 32, November 2003.

J. H. Lieske, “Precession Matrix Based on IAU (1976) System of Astronomical Constants”, *Astronomy and Astrophysics*, 73, 282-284 (1979)

Appendix

Precession and Nutation

This appendix summarizes the numerical methods used in the IAU 1976 precession and IAU 1980 nutation algorithms.

Precession

Precession is the slow drift of the Earth's rotational axis due mainly to the gravitational attraction of the Sun and Moon. The precession matrix transforms coordinates referred to the mean Earth equator and equinox of J2000 to coordinates measured with respect to the mean Earth equator and equinox of date.

According to J. H. Lieske, "Precession Matrix Based on IAU (1976) System of Astronomical Constants", *Astronomy and Astrophysics*, 73, 282-284 (1979), the fundamental precession matrix is defined by

$$\mathbf{P} = \begin{bmatrix} \cos z_a \cos \theta_a \cos \zeta_a - \sin z_a \sin \zeta_a & -\cos z_a \cos \theta_a \sin \zeta_a - \sin z_a \cos \zeta_a & -\cos z_a \sin \theta_a \\ \sin z_a \cos \theta_a \cos \zeta_a + \cos z_a \sin \zeta_a & -\sin z_a \cos \theta_a \sin \zeta_a + \cos z_a \cos \zeta_a & -\sin z_a \sin \theta_a \\ \sin \theta_a \cos \zeta_a & -\sin \theta_a \sin \zeta_a & \cos \theta_a \end{bmatrix}$$

The precession angles are given by

$$\begin{aligned} \zeta_a &= (2306.2181 + 1.39656T - 0.000139T^2)t + (0.30188 - 0.000344T)t^2 + 0.017998t^3 \\ z_a &= (2306.2181 + 1.39656T - 0.000139T^2)t + (1.09468 + 0.000066T)t^2 + 0.018203t^3 \\ \theta_a &= (2004.3109 - 0.85330T - 0.000217T^2)t + (-0.42665 - 0.000217T)t^2 - 0.041833t^3 \end{aligned}$$

where the unit of these angular arguments is arc seconds and the fundamental time arguments are as follows

$$T = (JED_1 - 2451545) / 36525$$

$$t = (JED_1 - JED_2) / 36525$$

In these two equations JED_1 is the Julian Date of the first epoch and JED_2 is the Julian Date of the second epoch, both measured on the ephemeris time scale.

The precession matrix can also be expressed as a combination of elementary rotations according to the following matrix multiplications

$$\mathbf{P} = \mathbf{R}_3(-z_a) \mathbf{R}_2(\theta_a) \mathbf{R}_3(-\zeta_a)$$

Nutation

The nutation matrix rotates coordinates referred to the mean Earth equator and equinox of date to coordinates referred to the true Earth equator and equinox of date. This part of the total coordinate transformation includes both the nutation in obliquity $\Delta\varepsilon$ and the nutation in longitude $\Delta\psi$. These periodic perturbations are caused by external forces acting on the Earth's pole or axis of rotation.

The nutation in longitude is determined from a series of the form

$$\Delta\psi = \sum_{i=1}^n S_i \sin A_i$$

Likewise, the nutation in obliquity is determined from

$$\Delta\varepsilon = \sum_{i=1}^n C_i \cos A_i$$

where

$$A_i = a_i l + b_i l' + c_i F + d_i D + e_i \Omega$$

and l, l', F, D and Ω are fundamental arguments. The IAU 1980 nutation theory contains a total of 108 terms.

The nutation matrix is defined by

$$\mathbf{N} = \begin{bmatrix} \cos \Delta\psi & -\sin \Delta\psi \cos \varepsilon_0 & -\sin \Delta\psi \sin \varepsilon_0 \\ \sin \Delta\psi \cos \varepsilon & \cos \Delta\psi \cos \varepsilon \cos \varepsilon_0 + \sin \varepsilon \sin \varepsilon_0 & \cos \Delta\psi \cos \varepsilon \cos \varepsilon_0 - \sin \varepsilon \cos \varepsilon_0 \\ \sin \Delta\psi \sin \varepsilon & \cos \Delta\psi \sin \varepsilon \cos \varepsilon_0 - \cos \varepsilon \sin \varepsilon_0 & \cos \Delta\psi \sin \varepsilon \sin \varepsilon_0 + \cos \varepsilon \cos \varepsilon_0 \end{bmatrix}$$

In this matrix ε_0 is the mean obliquity of the ecliptic and $\varepsilon = \varepsilon_0 + \Delta\varepsilon$ is the true obliquity.

The nutation matrix can also be expressed as a combination of elementary rotations according to

$$\mathbf{N} = \mathbf{R}_1(-\varepsilon) \mathbf{R}_3(-\Delta\psi) \mathbf{R}_1(+\varepsilon_0)$$

The mean obliquity of the ecliptic is calculated from

$$\varepsilon_0 = 23^{\circ}26'21.448 - 46.8150T - 0.00059T^2 + 0.001813T^3$$

where T is the time in Julian centuries given by $T = (JD - 2451545.0) / 36525$ and JD is the Julian Date on the universal time scale.

Elementary Rotation Matrices

The three rotation matrices for the x , y and z axes are defined by

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & \sin \theta \\ 0 & -\sin \theta & \cos \theta \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \quad R_z(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where θ is the rotation angle.