Ephemeris Subsystem SPK

Focused on reading SPK files

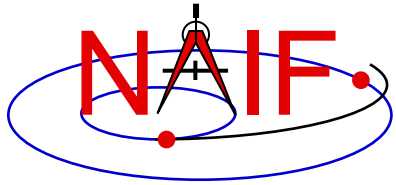## January 2017

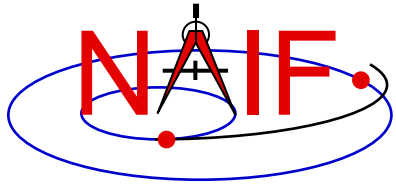**Navigation and Ancillary Information Facility**

- **SPK is probably unlike any previous ephemeris or trajectory representation you've used or heard about.**

- **We think you'll find it to be more capable than other ephemeris system architectures.**
  - **As such, it's also a bit more complicated to grasp.**

- *Don't panic! Shortly you'll be reading SPK files like a pro.*

Overview of
SPICE Ephemeris Data

A Picture is worth …

- **We'll start with a mostly pictorial view of ephemeris data and SPK files, just to ease you into this topic.**

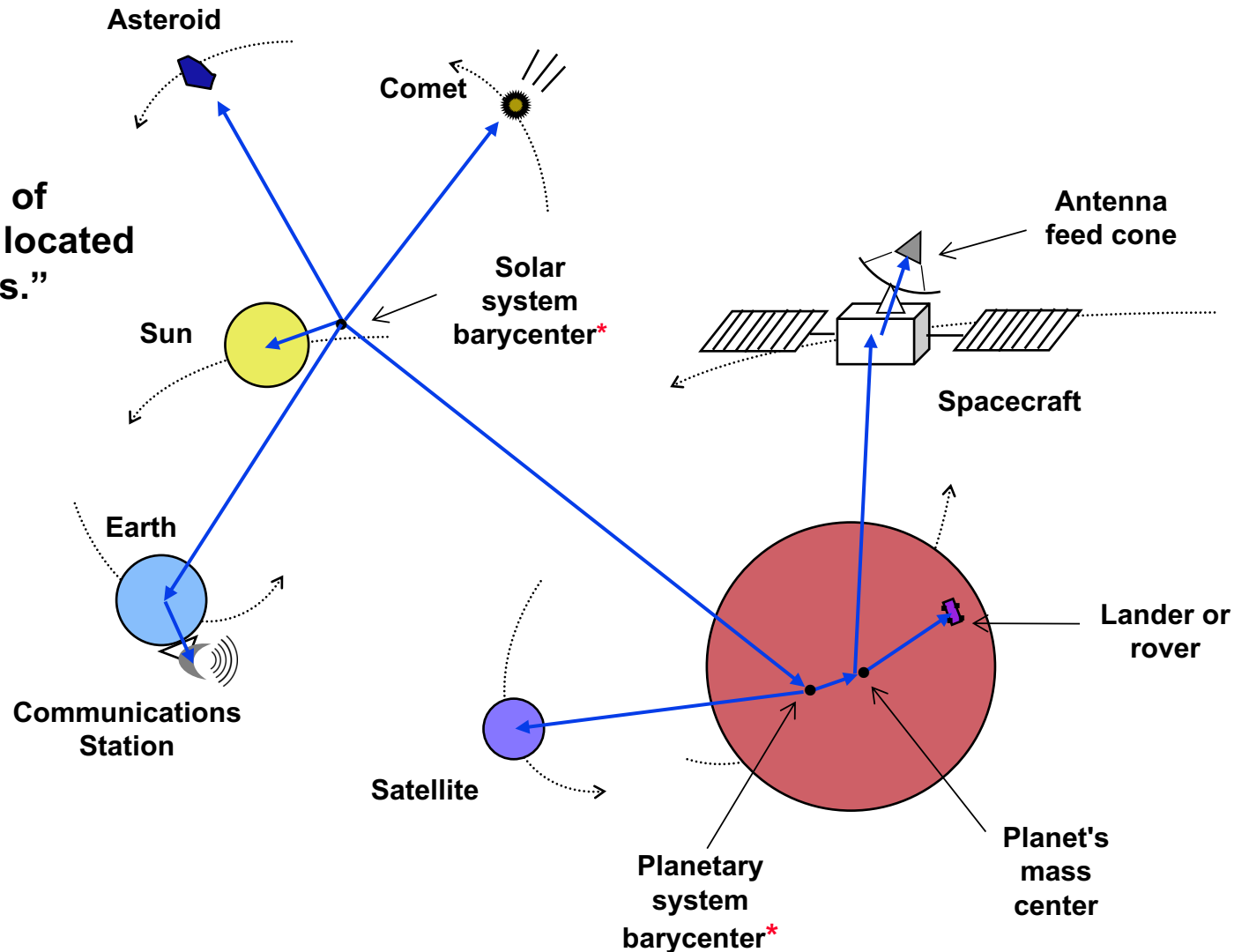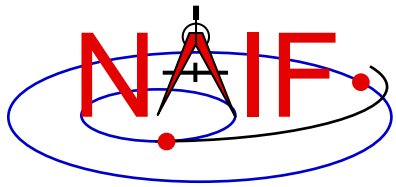**Navigation and Ancillary Information Facility**

- **An SPK file contains ephemeris (trajectory) data for "ephemeris objects."**
  - "Ephemeris" means position and velocity as a function of time
    - » Position + velocity is often referred to as "state"
- **"Ephemeris objects" are spacecraft, planets, satellites, comets and asteroids.**
  - The following are also ephemeris objects:
    - » the center of mass of our solar system (solar system barycenter)
    - » the center of mass of a planet/satellite system (planet barycenter)
    - » a rover on the surface of a body
    - » a camera on top of a mast on a lander
    - » a transmitter cone on a spacecraft
    - » a deep space communications antenna on the earth
- **A single SPK file can contain data for multiple ephemeris objects, and often does**
- **See the next page for a pictorial representation of some of these objects.**

Navigation and Ancillary Information Facility

The head and the tail of every **blue arrow** are located at "ephemeris objects."

Asteroid

Comet

Solar system barycenter*

Antenna feed cone

Sun

Spacecraft

Earth

Lander or rover

*A barycenter is the center of mass of a system of bodies, such as Saturn plus all of Saturn's satellites, or, the sun and all the planets, satellites, comets and asteroids in the solar system.

Communications Station

Satellite

Planetary system barycenter*

Planet's mass center
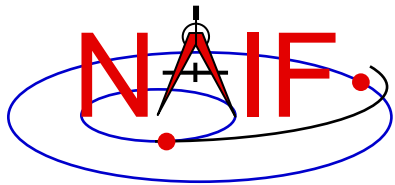
Imagine Some Ephemeris Data

epoch_1, x1, y1, z1, vx1, vy1, vz1
epoch_2, x2, y2, z2, vx2, vy2, vz2
epoch_3, x3, y3, z3, vx3, vy3, vz3
epoch_4, x4, y4, z4, vx4, vy4, vz4
.... etc. ....
.... etc. ....
epoch_n, xn, yn, zn, vxn, vyn, vzn

Perhaps this is an ASCII table or an Excel spreadsheet containing rows of time-tagged Cartesian state vectors.

"epoch" = time

It may not be written inside the table or spreadsheet, but perhaps an interface agreement somehow tells you:

- what object–the "target"–this ephemeris is for
- what is the name of the reference frame ("coordinate frame") in which the data are given
- what is the center of motion of the target
- what time system is being used for the epochs
- maybe also what are the start and stop times of the file
    - » meaning, what are "epoch_1" and "epoch_n"

Navigation and Ancillary Information Facility
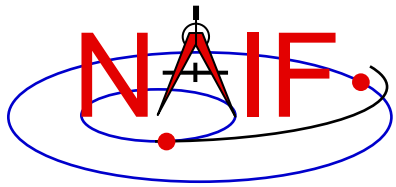
```
epoch_1,  x1,  y1,  z1,  vx1,  vy1,  vz1
epoch_2,  x2,  y2,  z2,  vx2,  vy2,  vz2
epoch_3,  x3,  y3,  z3,  vx3,  vy3,  vz3
epoch_4,  x4,  y4,  z4,  vx4,  vy4,  vz4
 ................. etc. ..................
 ................. etc. ..................
epoch_n,  xn,  yn,  zn,  vxn,  vyn,  vzn
```

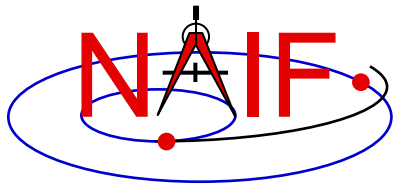⇐ **We'll represent that simple ephemeris data as a "block" like this.**

Navigation and Ancillary Information Facility

```
epoch_1,  x1,  y1,  z1,  vx1,  vy1,  vz1
epoch_2,  x2,  y2,  z2,  vx2,  vy2,  vz2
epoch_3,  x3,  y3,  z3,  vx3,  vy3,  vz3
epoch_4,  x4,  y4,  z4,  vx4,  vy4,  vz4
 ..................  etc. ...................
 ..................  etc. ...................
epoch_n,  xn,  yn,  zn,  vxn,  vyn,  vzn
```

We'll represent that simple ephemeris file as a "block" like this.

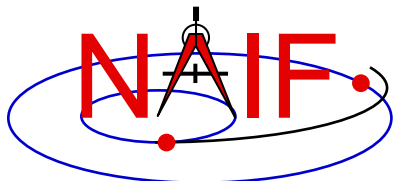**This becomes the basis of a "segment" in an SPK file.**

An SPK "Segment"

Target, Ref Frame ID, Center of Motion, T_{start}, T_{stop}

epoch_1, x1, y1, z1, vx1, vy1, vz1
epoch_2, x2, y2, z2, vx2, vy2, vz2
epoch_3, x3, y3, z3, vx3, vy3, vz3
epoch_4, x4, y4, z4, vx4, vy4, vz4
.... etc. ....
.... etc. ....
epoch_n, xn, yn, zn, vxn, vyn, vzn

One segment

We attach the following meta-data to complete the segment:

– **what is the object this ephemeris is for - the "target"**
– **what is the ID of the reference frame ("coordinate frame") in which the data are given**
– **what is the center of motion of the target - the "observer"**
– **maybe also what are the start and stop times of the file, T_{start} and T_{stop}**
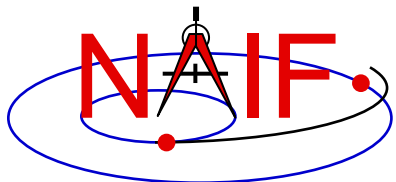   » **meaning, what are "epoch_1" and "epoch_n"**

# A Simple SPK File

Some other stuff
Target, Ref Frame ID, Center of Motion, T_{start}, T_{stop}
epoch_1, x1, y1, z1, vx1, vy1, vz1   **epoch_2, x2, y2, z2, vx2, vy2, vz2**   **epoch_3, x3, y3, z3, vx3, vy3, vz3**   **epoch_4, x4, y4, z4, vx4, vy4, vz4**   **.................. etc. ....**   **.................. etc. ....**   **epoch_n, xn, yn, zn, vxn, vyn, vzn**

- **This very simple SPK file is made up of a single segment containing ephemeris data:**
  - **for a single object (perhaps a spacecraft, an asteroid, or …whatever)**
  - **given in a single reference frame ("coordinate frame")**
  - **having a single center of motion**
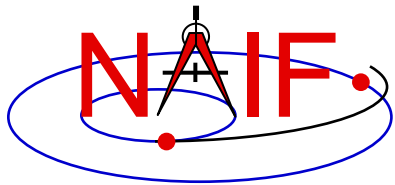 - **with data spanning from T_{start} to T_{stop}**

A More Substantial SPK File

Some other stuff

Target, Ref Frame ID, Center of Motion, T_1, T_2

Target, Ref Frame ID, Center of Motion, T_3, T_4

Target, Ref Frame ID, Center of Motion, T_5, T_6

```
epoch_1,  x1,  y1,  z1,  vx1,  vy1,  vz1
epoch_2,  x2,  y2,  z2,  vx2,  vy2,  vz2
epoch_3,  x3,  y3,  z3,  vx3,  vy3,  vz3
epoch_4,  x4,  y4,  z4,  vx4,  vy4,  vz4
................  etc. ...................
................  etc. ...................
epoch_n,  xn,  yn,  zn,  vxn,  vyn,  vzn
```
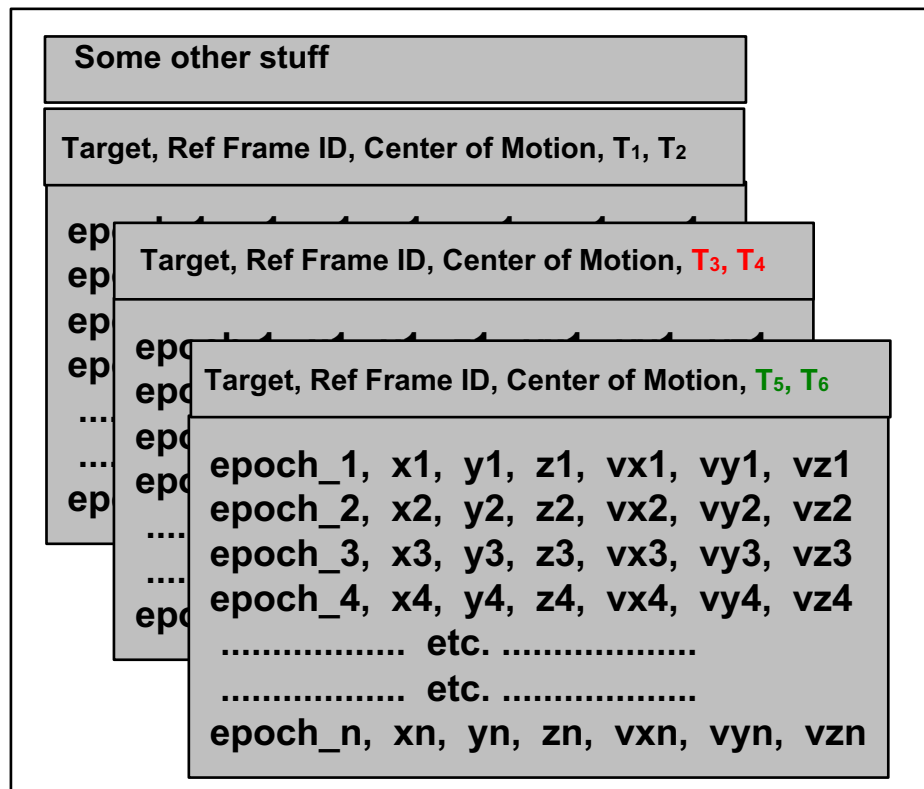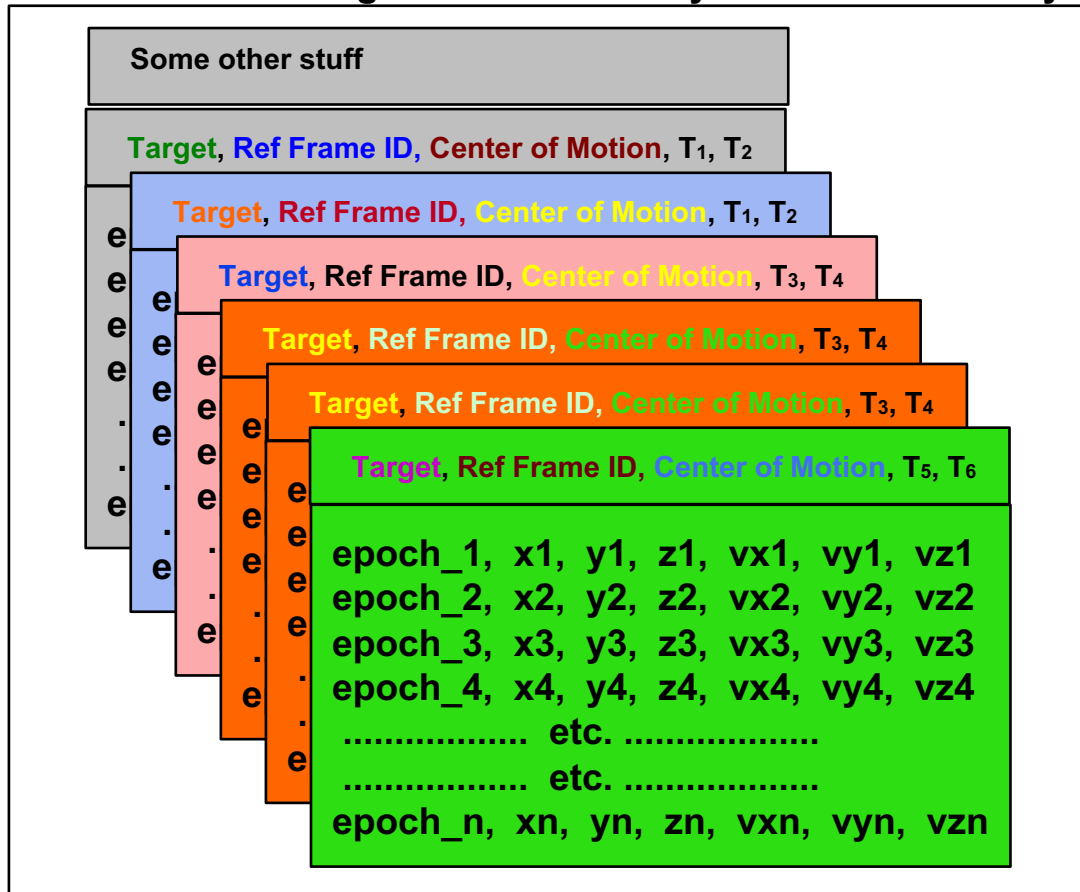
- **This more substantial SPK is made up of multiple segments containing ephemeris data:**
  - **for a single object (perhaps a spacecraft, an asteroid, or …???)**
  - **given in a single reference frame ("coordinate frame")**
  - **having a single center of motion**
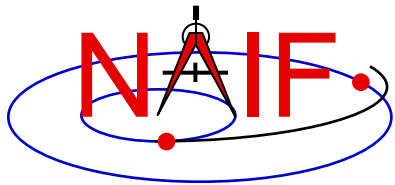 - **with data spanning from T_1 to T_6**

Navigation and Ancillary Information Facility

Some other stuff

Target, **Ref Frame ID**, **Center of Motion**, T_1, T_2

Target, **Ref Frame ID**, **Center of Motion**, T_1, T_2

Target, Ref Frame ID, **Center of Motion**, T_3, T_4

Target, Ref Frame ID, **Center of Motion**, T_3, T_4

Target, Ref Frame ID, **Center of Motion**, T_3, T_4

Target, Ref Frame ID, **Center of Motion**, T_5, T_6

epoch_1, x1, y1, z1, vx1, vy1, vz1
epoch_2, x2, y2, z2, vx2, vy2, vz2
epoch_3, x3, y3, z3, vx3, vy3, vz3
epoch_4, x4, y4, z4, vx4, vy4, vz4
.... etc. ....
.... etc. ....
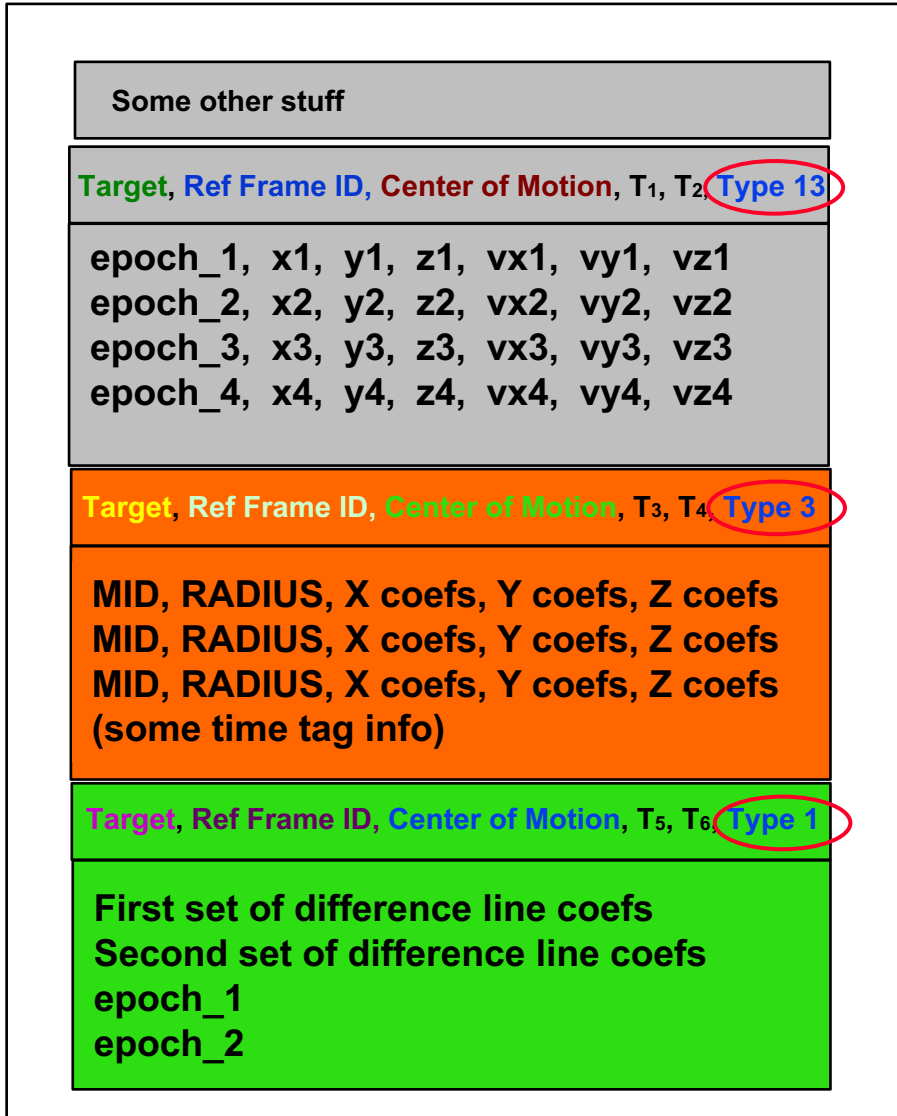epoch_n, xn, yn, zn, vxn, vyn, vzn

- **This even more substantial SPK contains multiple segments having:**
  - **several objects**
  - **several reference frames**
  - **several centers of motion**
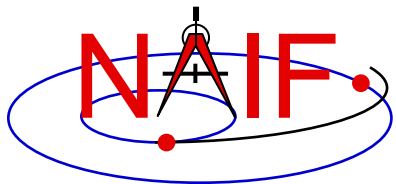 - **several pairs of start and stop times**

Some other stuff

Target, **Ref Frame ID**, **Center of Motion**, T_1, T_2, **Type 13**

```
epoch_1,  x1,  y1,  z1,  vx1,  vy1,  vz1
epoch_2,  x2,  y2,  z2,  vx2,  vy2,  vz2
epoch_3,  x3,  y3,  z3,  vx3,  vy3,  vz3
epoch_4,  x4,  y4,  z4,  vx4,  vy4,  vz4
```

Target, **Ref Frame ID**, **Center of Motion**, T_3, T_4, **Type 3**

MID, RADIUS, X coefs, Y coefs, Z coefs
MID, RADIUS, X coefs, Y coefs, Z coefs
MID, RADIUS, X coefs, Y coefs, Z coefs
(some time tag info)

Target, **Ref Frame ID**, **Center of Motion**, T_5, T_6, **Type 1**

First set of difference line coefs
Second set of difference line coefs
epoch_1
epoch_2

- **Each segment can contain a different type of ephemeris data (as long as it's been built into the SPK subsystem). Examples:**
  - **Discrete state vectors**
  - **Chebyshev polynomials**
  - **Difference lines (unique to JPL)**
  - **Etc., etc.**

- **Each segment has the SPK Type stored in its meta-data record.**

- **Toolkit software knows how to evaluate each Type – no worries for you!**

Navigation and Ancillary Information Facility

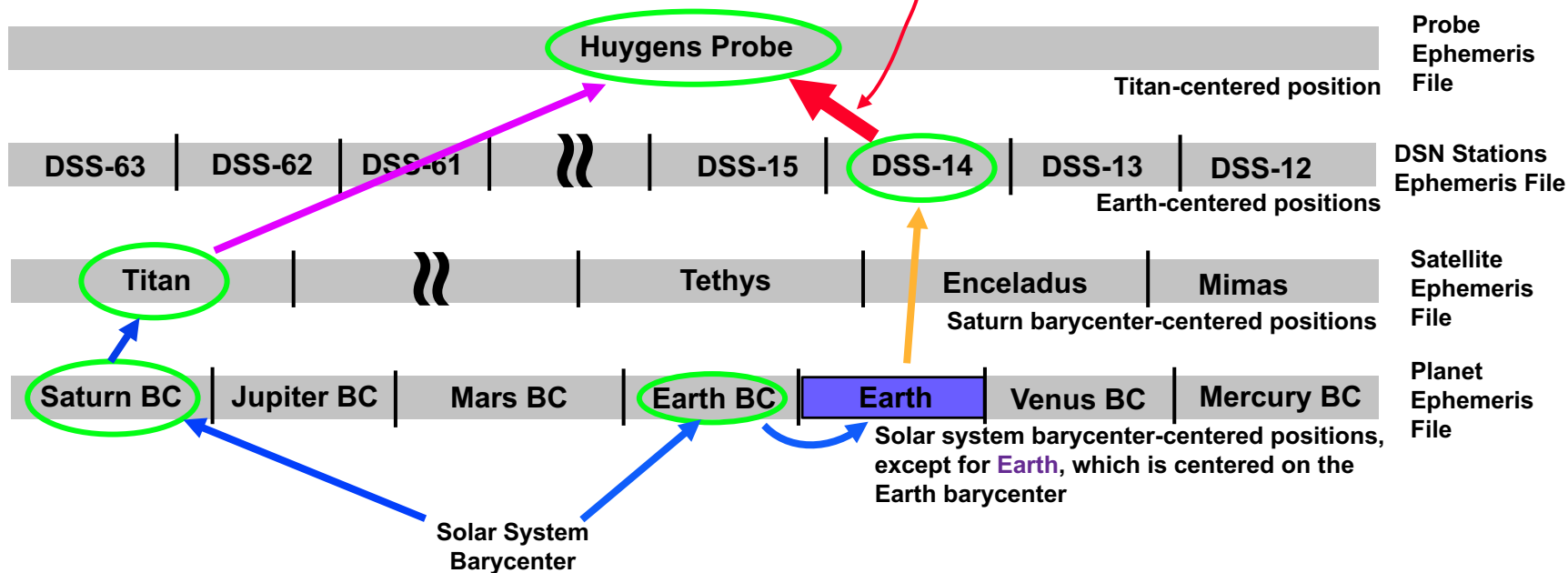| **Cassini**, Ref Frame ID, **Saturn bc**, T_1, T_2, Type 13 |
|---|
| epoch_1, x1, y1, z1, vx1, vy1, vz1
epoch_2, x2, y2, z2, vx2, vy2, vz2
epoch_3, x3, y3, z3, vx3, vy3, vz3
epoch_4, x4, y4, z4, vx4, vy4, vz4 |

- **Within the time bounds (T_1, T_2) of a segment, SPICE software will return a result–a state vector consisting of position and velocity–at any epoch… not just at the epochs of the ephemeris records**

- **In the example above, SPICE will return the position and velocity (the state) of the Cassini spacecraft relative to the Saturn barycenter at any time t where: $T_1 \leq t \leq T_2$**
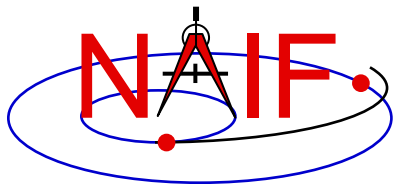
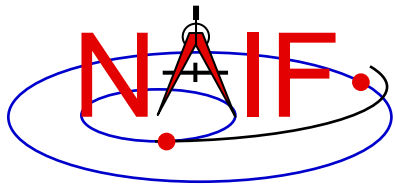- **SPICE automatically searches across all loaded SPK files to find the segments needed to compute the vectors needed to obtain the result the customer has asked for. SPICE chains these together using addition and subtraction.**
  - **In this example the user wants the position of the Huygens probe sitting on the surface of Titan as seen from Deep Space Station 14.**
  - **SPICE computes this by chaining the gold, blue and violet chunks.**

Probe Ephemeris File

Titan-centered position

DSN Stations Ephemeris File

Earth-centered positions

Satellite Ephemeris File

Saturn barycenter-centered positions

Planet Ephemeris File

Solar system barycenter-centered positions, except for Earth, which is centered on the Earth barycenter

Solar System Barycenter

Huygens Probe

DSS-63 | DSS-62 | DSS-61 | DSS-15 | DSS-14 | DSS-13 | DSS-12

Titan | Tethys | Enceladus | Mimas

Saturn BC | Jupiter BC | Mars BC | Earth BC | Earth | Venus BC | Mercury BC

- **As part of the "chaining" process just mentioned…**
  - **position vectors are automatically rotated into a consistent reference frame**
  - **the final vector is rotated into the output reference frame requested by the user**

Reference Frames Used

International Celestial
Reference Frame
(`J2000`)

Titan body-fixed frame
(`IAU_TITAN`)

International Terrestrial
Reference Frame
(`ITRF93`)

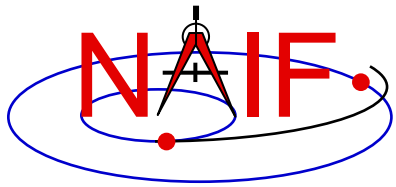DSS-14 topocentric
reference frame
(`DSS–14_TOPO`)

Ephemeris Segments Used

Titan

Huygens Probe
(after landing)

Saturn Barycenter

Deep Space
Station 14

Earth

Solar
System
Barycenter

Earth barycenter
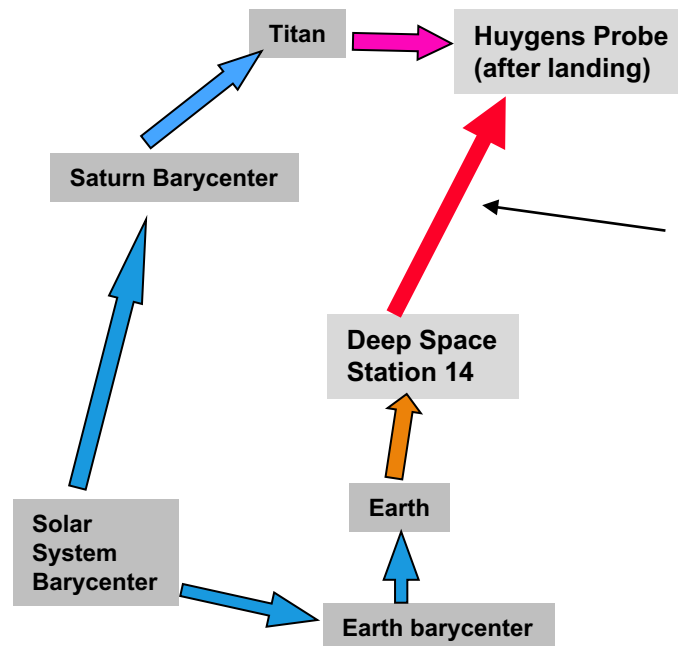
The position of the Huygens Probe relative to DSS-14, at time t, given in the DSS-14 topocentric reference frame, can be determined using a SINGLE Toolkit subroutine call.

A single subroutine call does it all!

```
CALL SPKPOS ('HUYGENS_PROBE', t, 'DSS-14_TOPO', LT+S, 'DSS-14', POSITION, LT)
```

17

- ## Within a single SPK file…

  - The segments in an SPK file **need not be ordered** according to time or body.

  - But segment order **does imply priority**. If two segments from the same SPK file both contain data for a given target and time that satisfy a request, the SPK system selects the segment for which the **physical location** is positioned **later** in the file.

    » The centers of motion, frames and SPK types are irrelevant to this selection.

- ## If using two or more SPK files…

  - Segments from SPK files loaded **later** have higher priority: if two segments from two different SPK files both contain data for a given target and time, the SPK system selects the segment from the SPK file that was loaded later.

   » The centers of motion, frames and SPK types are irrelevant to this selection.
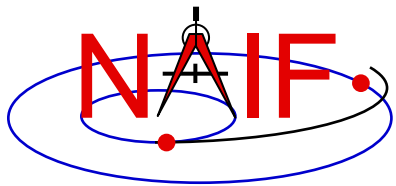
- **When you read an SPK file you specify which ephemeris object is to be the "target" and which is to be the "observer."**

- **The SPK system returns the state of the target relative to the observer.**
  - **The position data point from the "observer" to the "target."**
  - **The velocity is that of the "target" relative to the "observer."**
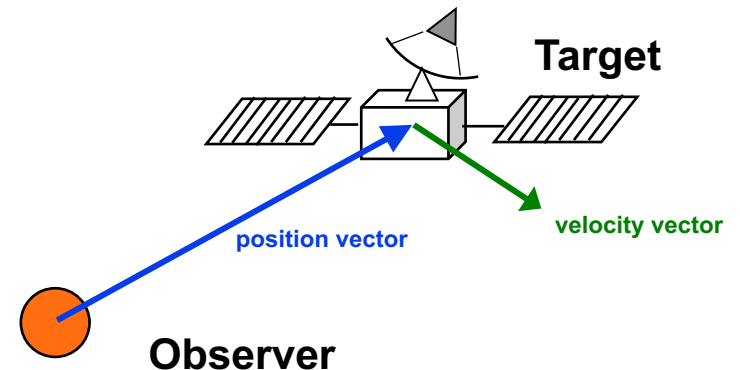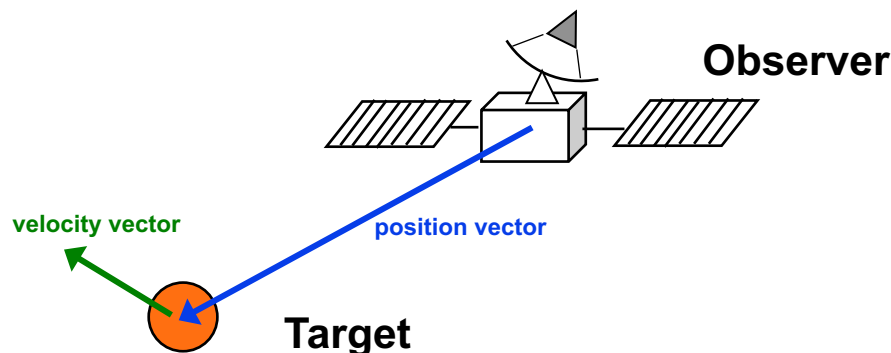
- **Any ephemeris object can be a target or an observer!**
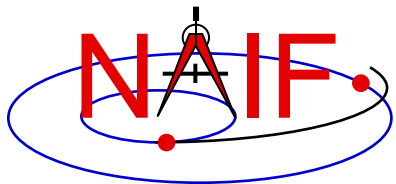
Navigation and Ancillary Information Facility

- **The time period over which an SPK file provides data for an ephemeris object is called the "coverage" or "time coverage" for that object.**
  - **An SPK file's coverage for an object consists of one or more time intervals.**
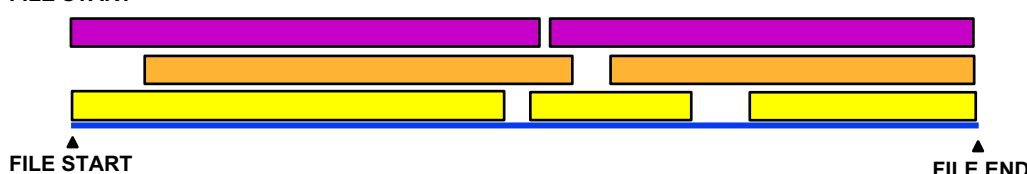 - **Often the coverage for all objects in an SPK file is a single, common time interval.**

SPK file containing data for one object with no data gaps

FILE START

FILE END

SPK file containing data for one object, with two data gaps
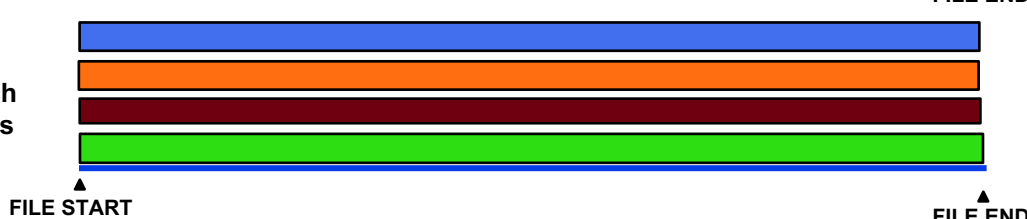
FILE START

FILE END
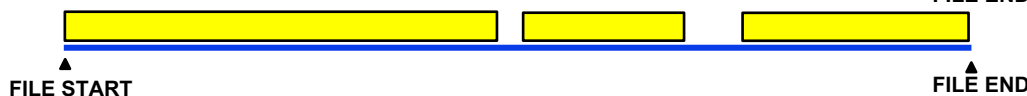
SPK file containing data for three objects, each having different data gaps
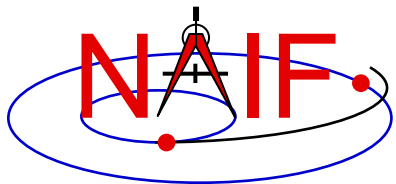
FILE START

FILE END

SPK file containing data for three objects, each having different coverage but with no data gaps

FILE START

FILE END

SPK file containing data for several objects, each having the same coverage and with no data gaps
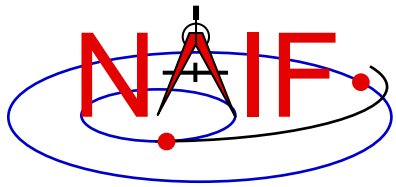
FILE START

FILE END

- **For any request time within any time interval comprising the coverage for an object (i.e. the three green stripes below), the SPK subsystem can return a vector representing the state of that object relative to its center of motion.**
  - The SPK system will automatically interpolate ephemeris data to produce a state vector at the request time.
  - To a user's program, the ephemeris data appear to be **continuous** over each time interval, even if the data stored inside the SPK file are discrete.

- **The SPK subsystem will *not* return a result for a request time falling within a data gap. Data gaps can only occur between segments.**

"Results" will be returned by the SPK reader API for any request time falling within these three intervals

FILE START   FILE END
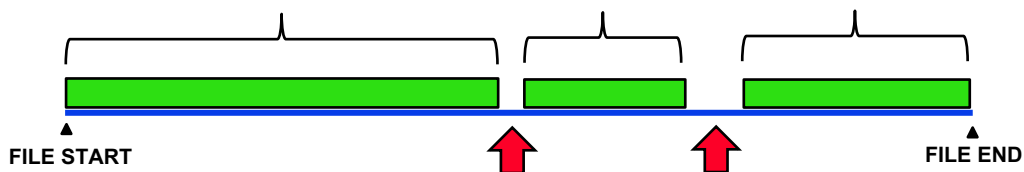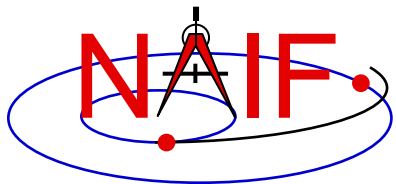
Note: each of the green stripes above could consist of one or more segments.

No results will be returned by the SPK subsystem for any request time falling within these two data gaps

- **All ephemeris data have an associated reference frame**
  - **The frame specification is provided by the SPK producer**

- **A program reading an SPK file specifies relative to what reference frame the output state or position vectors are to be given; you're not stuck with using the frame the SPK producer used**
  - **This output frame you select must be known to the program**
    - » **"Known" means either a built-in frame (hard coded in SPICE) or one fully specified at run-time**
    - » **The user's program may need to have access to additional SPICE data in order to construct the specified frame**
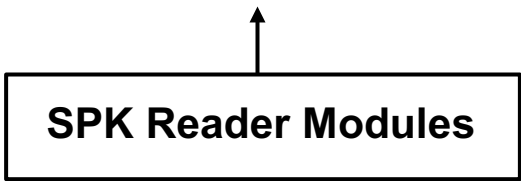
SPK Data Type Concept

SPK files may contain various mathematical representations of ephemeris data ("data types"), but the high-level user interfaces (SPK "readers") are type-independent:
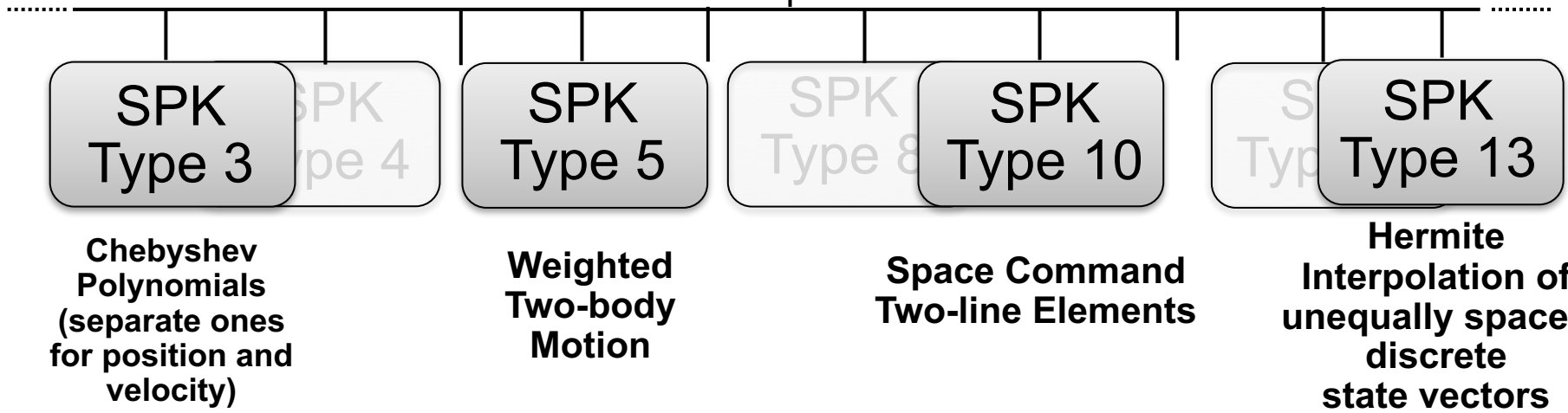
Some SPK reader APIs do not return velocity

```
X, Y, Z, dX, dY, dZ at requested time
```

The most used SPK reader APIs are these:

SPK Reader Modules

```
SPKEZR (to get position and velocity)
SPKPOS (to get position only)
SPKGEO (a low-level SPK reader)
```

SPK Type 3	SPK Type 4	SPK Type 5	SPK Type 8	SPK Type 10	SPK Type ...	SPK Type 13

Chebyshev Polynomials (separate ones for position and velocity)

Weighted Two-body Motion

Space Command Two-line Elements

Hermite Interpolation of unequally spaced discrete state vectors

Why Have Multiple Data Types?

- **To allow an SPK producer to choose an ephemeris representation well-suited for his application. For example:**

    - Weighted two-body extrapolation (type 5) yields compact files; may be used with sparse data. Only accurate for motion that is well approximated by the two-body model.

    - SPK files based on sliding-window Lagrange and Hermite interpolation (types 9 and 13) are easy to create. Position can be made arbitrarily accurate with sufficiently small time separation of states.

    - Chebyshev polynomials (types 2, 3, 14) yield the best combination of evaluation speed and accuracy. But the file creator must do more work to use these data types.

- **To replicate data originally provided in another format.**

    - Types 1, 2, 3, 8, 10, 14, 15, 17 and 18 were developed to enable accurate duplication of data obtained from original ephemeris developers.
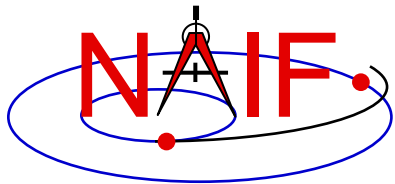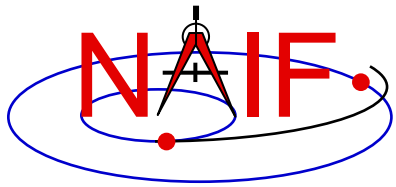
Widely Used SPK Data Types

- **Type 1 (Modified divided difference arrays)**
  - **Used by JPL orbit determination software for spacecraft ephemerides**

- **Type 2 (Chebyshev polynomials for position, velocity given by differentiation)**
  - **Used for JPL planetary ephemerides**

- **Type 3 (Separate Chebyshev polynomials for position and velocity)**
  - **Used for JPL satellite ephemerides**

- **Type 5 (Weighted two-body extrapolation)**
  - **Used for comets and asteroids, as well as for sparse data sets where a piecewise two-body approximation is acceptable**

- **Type 10 (Space command two-line elements)**
  - **Used for some earth orbiters**

- **Types 9 and 13 (Sliding-window Lagrange and Hermite interpolation of unequally-spaced states)**
  - **Used by non-JPL ephemeris producers and by MKSPK users**

- **Type 18, 19 (Sliding window Hermite or Lagrange interpolation)**
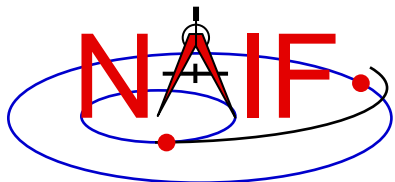 - **Used in SPKs made by ESA planetary missions (through Rosetta)**

Navigation and Ancillary Information Facility

• For planets

- A planet and its satellites orbit the planet system's barycenter
  - » For example, the planet Jupiter (599) and each of Jupiter's satellites (501 - 5xx) orbit the Jupiter system barycenter (5)

- Because Mercury and Venus have no satellites, their barycenters (1 and 2) are at exactly the same locations as their mass centers (199 and 299)
  - » Therefore SPICE ephemeris objects 199 and 299 as well as 1 and 2 are found in a planet ephemeris file

- Because the masses of Phobos and Deimos are so small compared to the mass of Mars, the mass center for Mars (499) was treated as being located at the Mars barycenter (4)
  - » Starting in 2013 with the JPL planetary ephemeris named DE430 this is no longer the case; there is a very small offset of about 20 cm

• For the solar system

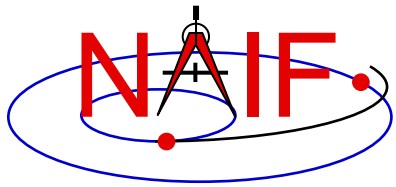- Planet system barycenters (i.e. 1 through 9) and the sun (10) orbit the solar system barycenter (0)

Barycenter Offset Magnitude

Navigation and Ancillary Information Facility

Body Mass Center ID	System Barycenter ID	Barycenter offset from body mass center (km)*	Offset as % of body radius*
Sun (10)	SSB (0)	1,378,196	**198**%
Mercury (199)	M. BC (1)	0	0
Venus (299)	V. BC (2)	0	0
Earth (399)	E. BC (3)	4942	**77**%
Mars (499)	M. BC (4)	0.0002	~ 0
Jupiter (599)	J. BC (5)	220	0.3%
Saturn (699)	S. BC (6)	312	0.5%
Uranus (799)	U. BC (7)	43	0.17%
Neptune (899)	N. BC (8)	74	0.3%
Pluto (999)**	P. BC (9)	2056	**172**%

* Estimated maximum values over the time range 2000-2050.
** For ephemeris purposes, Pluto is still treated as a planet.

- **A single SPK file can hold data for one ephemeris object, or for many ephemeris objects**

- **The objects in a given SPK file need not all be of the same type**
  - **One might find data for a spacecraft, some planets, and some satellites all in one file, split across multiple segments**

- **This is illustrated in the next three charts**

Navigation and Ancillary Information Facility

Planet Ephemeris   Asteroid Ephemeris   Merged Planet[4] and Satellite Ephemeris

0	*Solar System BC*
1	Merc. BC
199[1]	Mercury
2	Venus BC
299[1]	Venus
3	Earth BC
301[2]	Moon
399[2]	Earth
4	Mars BC
5	Jupiter BC
6	Saturn BC
7	Uranus BC
8	Neptune BC
9	Pluto BC*
10[3]	Sun

10	*Sun*
2000001	Ceres

5	*Jupiter BC*
3[4]	Earth BC
10[4]	Sun
399[4]	Earth
501	Io
502	Europa
503	Ganymede
504	Callisto
505	Amalthea
514	Thebe
515	Adrastea
516	Metis
599	Jupiter

Notes:
(1) Mercury and Venus planet locations are included in planet ephemerides since there are no satellite ephemerides for these planets.
(2) The Moon and Earth locations are included in each planetary ephemeris because of historical ephemeris production techniques.
(3) The Sun's location is included in each planetary ephemeris because of historical ephemeris production techniques.
(4) For user convenience, NAIF usually merges into a planet's satellite ephemeris files the locations of the earth, the earth barycenter and the sun.

* For ephemeris purposes, Pluto is still treated as a planet.

The objects in blue italic font are the center of motion for the remaining objects in each file. These is no trajectory data present for these centers of motion.
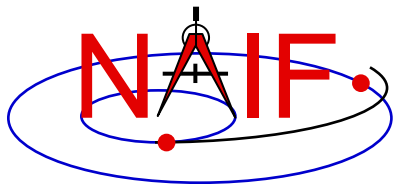
Examples of Flight Project SPK Files

Navigation and Ancillary Information Facility

This <u>made-up</u> example shows four collections of SPK files for the Cassini mission

Cassini Orbiter

6 = Saturn bc

-82 = Cassini S/C

One object

Huygens Probe

6 = Saturn bc

-150 = Huygens Probe

One object

Planets

0 = solar system bc

3 = Earth barycenter
6 = Saturn barycenter
399 = Earth mass center
301 = Moon

Multiple objects

Satellites - 1

6 = Saturn bc

601 = Mimas
602 = Enceladus
603 = Tethys
604 = Dione
605 = Rhea
606 = Titan
607 = Hyperion
608 = Iapetus
609 = Phoebe
699 = Saturn mass center

Multiple objects

610 = Janus
611 = Epimetheus
:
617 = Pandora
699 = Saturn mass center

Multiple objects

Satellites - 2

- **The user's program must "load" as many of these SPK files as needed to satisfy her/his requirements.**

- **Sometimes a project NAV team combines (merges) several of these collections before releasing them, making the user's job easier.**

- **Objects in blue font are the centers of motion for the remaining objects; these don't have ephemeris data included in the file.**

See the next page for a graphical representation of this collection of SPKs
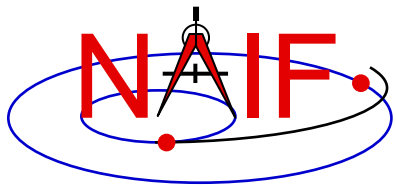
Possible* SPK File Time Coverages for the Previous Example

Navigation and Ancillary Information Facility
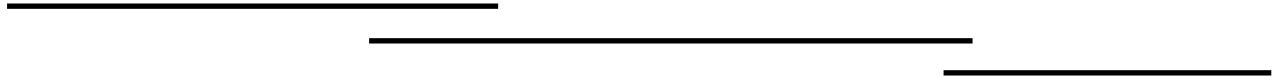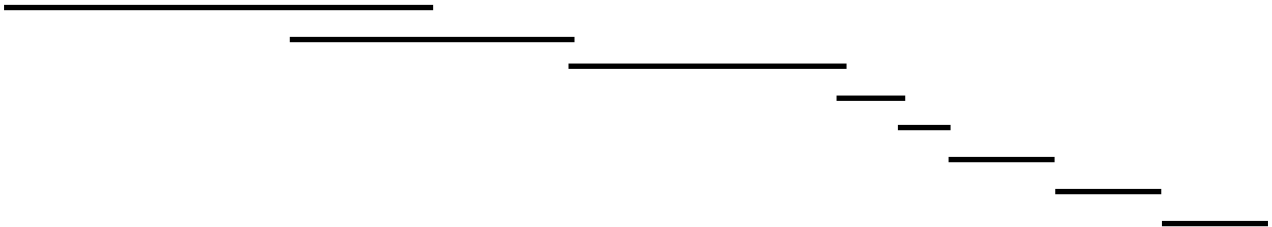
Each bar represents a separate file

Planet:

Satellite - 1:
(Major satellites)

Satellite - 2:
(Minor satellites)

Orbiter :

Probe :

Time line:

cruise phase ← → orbit phase

Launch   **Orbit Insertion** **Probe Release**   **End of Mission**
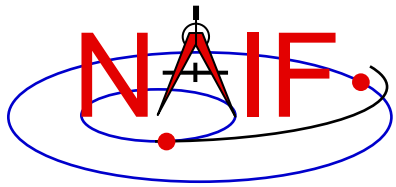
* Note: This was not the real Cassini scenario–it is simply an illustration of some of the possibilities for ephemeris delivery on a planetary mission.

- **An SPK file may contain positions of tracking stations, observatories, rovers, etc.**
  - **The object could be stationary or moving**
  - **Usually such SPKs contain ephemeris data given in the body-fixed reference frame**

- **One reads this file the same as for any other SPK file**
  - **Use the name or NAIF ID of the antenna, observatory or rover as the "target" or "observer" in an SPK reader argument list**
  - **Also requires use of a SPICE PCK file if you request vectors to be returned in an inertial frame such as J2000**
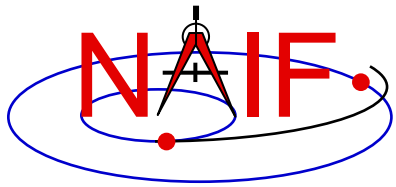   - » **Needed to rotate body-fixed vectors to the J2000 frame**

Using SPK Files

Retrieving Position or State Vectors

- **To retrieve position or state vectors of ephemeris objects one normally needs two kinds of SPICE kernels**
  - **Ephemeris kernel(s)   (SPK)**
  - **Leapseconds kernel   (LSK)**
    - » **Used to convert between Coordinated Universal Time (UTC) and Barycentric Dynamical Time (TDB, also called Ephemeris Time, ET)**
      - • **This conversion is done outside of the SPK subsystem**
- **Retrieving ephemeris data from an SPK file is usually called "reading" the file**
  - **This term is not very accurate since the SPK "reader" software also performs interpolation, and may chain together data from multiple sources, do frame transformations and perform aberration corrections**
- **State and position vectors retrieved from an SPK file by the SPK "reader" routines are of the form:**
  - **X,Y, Z, dX, dY, dZ   for a state vector (km, km/sec)**
  - **X, Y, Z   for a position vector (km)**

Retrieving a State Vector

Fortran syntax used here

Initialization…typically done <u>once</u> per program execution

Tell your program which SPICE files to use ("loading" files)

CALL FURNSH ('spk_file_name')

CALL FURNSH ('leapseconds_file_name')

It's better to replace these two calls with a single call to a "furnsh kernel" containing the names of all kernel files to load.

Loop... do as many times as you need to

Convert UTC time to ephemeris time (TDB), if needed

CALL STR2ET ('utc_string', *tdb*)

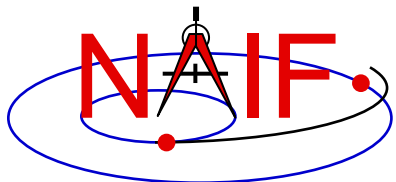**Retrieve state vector from the SPK file at your requested time**

CALL SPKEZR (target, tdb, 'frame', 'correction', observer, *state, light time*)

inputs   outputs

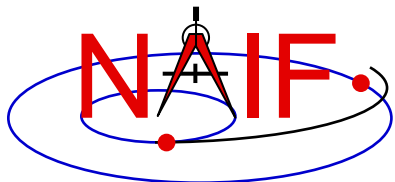**Now use the returned state vector in other SPICE routines to compute observation geometry of interest.**

Navigation and Ancillary Information Facility

INPUTS

- **TARGET* and OBSERVER*: Character names or NAIF IDs for the end point and origin of the state vector (Cartesian position and velocity vectors) to be returned.**
  - The position component of the requested state vector points from observer to target.

- **TDB: The time at the observer at which the state vector is to be computed. The time system used is Ephemeris Time (ET), now generally called Barycentric Dynamical Time (TDB).**

- **FRAME: The SPICE name for the reference frame in which the output state vector is to be given. SPK software will automatically convert ephemeris data to the frame you specified (if needed). SPICE must know the named frame. If it is not a built-in frame SPICE must have sufficient data at run-time to construct it.**

* Character names work for the target and observer inputs only if built into SPICE or if registered using the SPICE ID-body name mapping facility. Otherwise use the SPICE numeric ID in quotes, as a character string.

Navigation and Ancillary Information Facility

- **CORRECTION: Specification of what kind of aberration correction(s), if any, to apply in computing the output state vector.**

  – **Use 'LT+S' to obtain the apparent state of the target as seen by the observer. 'LT+S' invokes light time and stellar aberration corrections. ('CN+S' is better in some cases.)**

  – **Use 'NONE' to obtain the uncorrected (aka "geometric") state, as given by the source SPK file or files.**
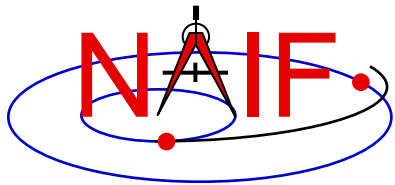
  See the header for subroutine SPKEZR, the document SPK Required Reading, or the "Fundamental Concepts" tutorial for details. See the backup charts for examples of aberration correction magnitudes.

OUTPUTS

- *STATE*: **This is the Cartesian state vector you requested. Contains 6 components: three for position (x,y,z) and three for velocity (dx, dy, dz) of the target with respect to the observer. The position component of the state vector points from the *observer* to the *target*.**

- *LIGHT TIME*: **The one-way light time between the (optionally aberration-corrected) position of target and the geometric position of the observer at the specified epoch.**

LT + S = light time plus stellar aberration
CN + S = converged Newtonian light time plus stellar aberration

A Simple Example of Retrieving a State Vector

Navigation and Ancillary Information Facility

Initialization - typically do this just <u>once</u> per program execution

```
CALL FURNSH ( 'NAIF0010.TLS' )
CALL FURNSH ( 'CASSINI_MERGED.BSP' )
```

It's better to replace these two calls with a single call to a "furnsh kernel" containing the names of all kernel files to load.
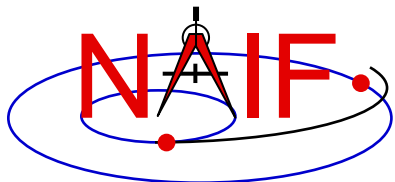
Repeat in a loop if/as needed to solve your particular problem

```
CALL STR2ET ('2004 NOV 21 02:40:21.3', TDB )
CALL SPKEZR ('TITAN', TDB, 'J2000', 'LT+S', 'CASSINI',
                   STATE,  LT )
```

(Insert additional code here to make derived computations such as spacecraft sub-latitude and longitude, lighting angles, etc. Use more SPICE subroutines to help.)

In this example we get the state (STATE) of Titan as seen from the Cassini spacecraft at the UTC epoch 2004 NOV 21 02:40:21.3. The state vector is returned in the J2000 inertial reference frame (which in SPICE is the same as the ICRF frame) and has been corrected for both light time and stellar aberration (LT+S). The one-way light time (LT) is also returned.

A SPICE leapseconds file (NAIF0010.TLS) is used, as is a SPICE ephemeris file (CASSINI_MERGED.BSP) containing ephemeris data for Cassini (-82), Saturn barycenter (6), Saturn mass center (699), Saturn's satellites (6xx) and the sun (10), relative to the solar system barycenter.
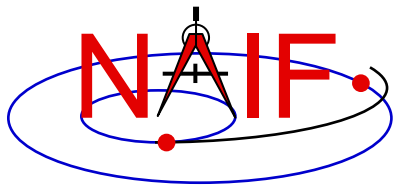
Navigation and Ancillary Information Facility

- **SPKPOS is the position-only analog of SPKEZR**
    - **The arguments of SPKPOS are identical to those of SPKEZR, except that SPKPOS returns a 3-component position vector instead of a 6-component state vector**
    - **SPKPOS executes more quickly than SPKEZR when stellar aberration corrections are used**
    - **SPKPOS can be used when reference frame transformations of velocity are not possible due to absence of C-kernel angular velocity data**
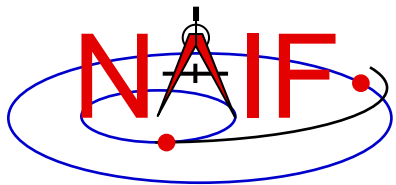
Navigation and Ancillary Information Facility

- **To get state vectors referenced to a non-inertial reference frame, or when the data within the SPK file are provided in a non-inertial frame, typically more kernels will be needed.**

  - **To get the state of an object relative to a body in the body's IAU body-fixed reference frame you'll need:**
    - » **PCK file containing orientation data for the body**
    - » **SPK(s) for the object and body**
    - » **LSK**

  - **To get the state of an object in a spacecraft-fixed reference frame you'll need:**
    - » **FK, CK and SCLK for the spacecraft**
    - » **SPK(s) for the spacecraft and object**
    - » **LSK**

A Slightly More Complex Example
Retrieving a State Vector

Obtain the state of Titan relative to Cassini in the Titan body-fixed reference frame

Initialization...typically <u>once</u> per program execution

Tell your program which SPICE files to use ("loading" files)

```
CALL FURNSH   ('CASSINI_MERGED.BSP')

CALL FURNSH   ('NAIF0010.TLS')

CALL FURNSH   ('NAIF0010.TPC')
```

It's better to replace these three calls with a single call to a "furnsh kernel" containing the names of all kernel files to load.

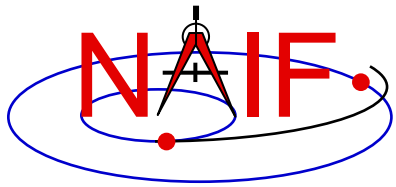**Loop... do as many times as you need**

Convert UTC time to ephemeris time (TDB), if needed

```
CALL STR2ET ( '2004 NOV 21 02:40:21.3', TDB)
```

Get state vector from SPK file at requested time, in planet's IAU body-fixed frame

```
CALL SPKEZR ('TITAN', TDB, 'IAU_TITAN', 'LT+S', 'CASSINI',
   STATE, LT)


(Insert additional code here to make derived computations such as
   spacecraft sub-latitude and longitude, lighting angles, etc. Use
   more SPICE subroutines to help.
```

- **The SPK subsystem contains routines for computing the state of an ephemeris object with respect to a fixed point or one moving with constant, non-zero velocity, in a specified reference frame.**
  - **The point may act as either the target or the observer.**
  - **The center of motion of the point must be an ephemeris object.**
- **The SPK routines providing this capability are:**
  - **SPKCPT (SPK, constant position target)**
  - **SPKCPO (SPK, constant position observer)**
  - **SPKCVT (SPK, constant velocity target)**
  - **SPKCVO (SPK, constant velocity observer)**
- **These routines may provide a convenient alternative to creating SPK files for surface points:**
  - **when there's a need to compute the locations on the fly**
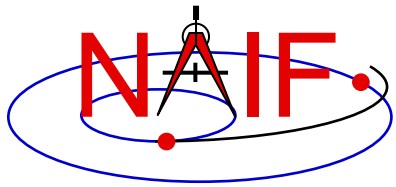 - **when the number of surface points is large**
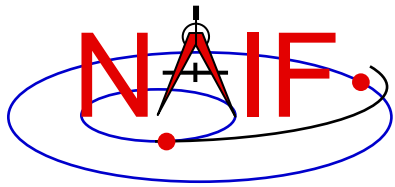
Navigation and Ancillary Information Facility

- ## You can subset an SPK, or merge two or more SPKs
    - The subset or merge may be keyed off of objects, or time, or both
    - Merging only portions of SPKs is possible

- ## You can read data from just one, or many* SPK files in your application program
    - Don't forget the precedence rule: data in a later loaded file take precedence over data from an earlier loaded file

- ## You can convert an SPK that is in non-native binary format to native binary format if you need to add data or comments
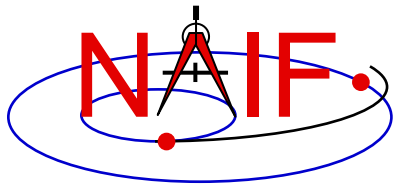
* The allowed number of simultaneously loaded DAF- and DAS-based files is set to 5000 in N66 Toolkits. "DAF" is Double Precision Array File. SPKs are based on the DAF architecture.

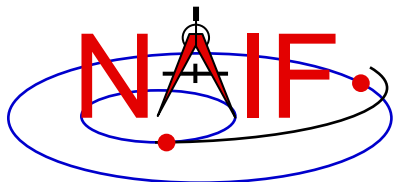**Navigation and Ancillary Information Facility**

- **The SPK producer should have provided descriptive meta-data inside an SPK file, in the "comment area"**
  - The comments should say when/why/how and for what purpose the file was made
  - Additional useful information could also be provided by the producer
    - » Example: when and why any data gaps are present

- **These comments may be extracted using an API (subroutine) or viewed using a SPICE utility program.**
  - API: SPCEC
  - Utility program: commnt -r

SPK Utility Programs

- **The following SPK utility programs are included in the Toolkit:**

BRIEF	summarizes coverage for one or more SPK files
SPACIT	generates segment-by-segment summary of an SPK file
COMMNT	reads, appends, or deletes comments in an SPK file
MKSPK	converts ephemeris data provided in a text file into an SPK file
SPKDIFF	compares two SPK files
SPKMERGE	subsets an spk, or merges one or more SPK files or pieces

- **These additional SPK utility programs are provided on the NAIF Web site (http://naif.jpl.nasa.gov/naif/utilities.html)**

SPY	validates, inspects, and analyses SPK files
PINPOINT	creates an SPK file for fixed locations (ground stations, etc)
BSPIDMOD	alters body IDs in an SPK file
DAFMOD	alters body or frame IDs in an SPK file
DAFCAT	concatenates together SPK files
BFF	displays binary file format of an SPK file
BINGO	converts SPK files between big- and little-endian formats

Navigation and Ancillary Information Facility

- **A summary of the contents and time coverage of an SPK file can be made using the SPICE Toolkit utility *"brief"***
    - **See the brief User's Guide for details**

```
% brief 070413BP_SCPSE_07097_07121.bsp



Summary for: 070413BP_SCPSE_07097_07121.bsp

Bodies: CASSINI (-82)            PLUTO BARYCENTER (9)     TETHYS (603)
        MERCURY BARYCENTER (1)   SUN (10)                DIONE (604)
        VENUS BARYCENTER (2)     MERCURY (199)           RHEA (605)
        EARTH BARYCENTER (3)     VENUS (299)             TITAN (606)
        MARS BARYCENTER (4)      MOON (301)              HYPERION (607)
        JUPITER BARYCENTER (5)   EARTH (399)             IAPETUS (608)
        SATURN BARYCENTER (6)    MARS (499)              PHOEBE (609)
        URANUS BARYCENTER (7)    MIMAS (601)             SATURN (699)
        NEPTUNE BARYCENTER (8)   ENCELADUS (602)

        Start of Interval (ET)               End of Interval (ET)
        ------------------------------       ------------------------------
        2007 APR 07 16:22:23.000             2007 MAY 01 09:34:03.000
```
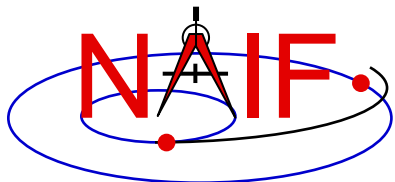
Note, the default time system is ET, not UTC!

Navigation and Ancillary Information Facility

- **Use of the "-c" option in brief will show you the center of motion for each object**
  - **This is often useful in diagnosing an SPK chaining problem**
    - » **See the "Problems" section at the end of this tutorial for more information**

```
% brief -c 070413BP_SCPSE_07097_07121.bsp

Bodies: CASSINI (-82) w.r.t. SATURN BARYCENTER (6)
        MERCURY BARYCENTER (1) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        VENUS BARYCENTER (2) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        EARTH BARYCENTER (3) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        MARS BARYCENTER (4) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        JUPITER BARYCENTER (5) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        SATURN BARYCENTER (6) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        URANUS BARYCENTER (7) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        NEPTUNE BARYCENTER (8) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        PLUTO BARYCENTER (9) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        SUN (10) w.r.t. SOLAR SYSTEM BARYCENTER (0)
        MERCURY (199) w.r.t. MERCURY BARYCENTER (1)
        VENUS (299) w.r.t. VENUS BARYCENTER (2)
        MOON (301) w.r.t. EARTH BARYCENTER (3)
        EARTH (399) w.r.t. EARTH BARYCENTER (3)
        MARS (499) w.r.t. MARS BARYCENTER (4)
        MIMAS (601) w.r.t. SATURN BARYCENTER (6)
        ENCELADUS (602) w.r.t. SATURN BARYCENTER (6)
        TETHYS (603) w.r.t. SATURN BARYCENTER (6)
        DIONE (604) w.r.t. SATURN BARYCENTER (6)
        RHEA (605) w.r.t. SATURN BARYCENTER (6)
        TITAN (606) w.r.t. SATURN BARYCENTER (6)
        HYPERION (607) w.r.t. SATURN BARYCENTER (6)
        IAPETUS (608) w.r.t. SATURN BARYCENTER (6)
        PHOEBE (609) w.r.t. SATURN BARYCENTER (6)
        SATURN (699) w.r.t. SATURN BARYCENTER (6)
        Start of Interval (ET)          End of Interval (ET)
        ----------------------------    ----------------------------
        2007 APR 07 16:22:23.000        2007 MAY 01 09:34:03.000
```

NAIF
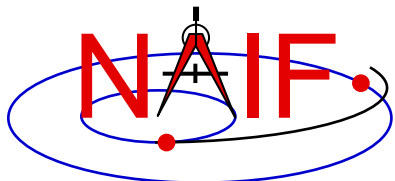
Navigation and Ancillary Information Facility

- **A detailed summary can be made using the Toolkit utility "SPACIT"**
- **See the SPACIT User's Guide for details**

```
Summary for SPK file: sat240.bsp
Leapseconds File    : /kernels/gen/lsk/leapseconds.ker
Summary Type        : Entire File


-----------------------------------------------------------------------------
   Segment ID     : SAT240
   Target Body    : Body 601, MIMAS
   Center Body    : Body 6, SATURN BARYCENTER
   Reference frame: Frame 1, J2000
   SPK Data Type  : Type 3
      Description : Fixed Width, Fixed Order Chebyshev Polynomials: Pos, Vel
   UTC Start Time : 1969 DEC 31 00:00:00.000
   UTC Stop Time  : 2019 DEC 02 00:00:00.000
   ET Start Time  : 1969 DEC 31 00:00:41.183
   ET Stop time   : 2019 DEC 02 00:01:05.183
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
   Segment ID     : SAT240
   Target Body    : Body 602, ENCELADUS
   Center Body    : Body 6, SATURN BARYCENTER
   Reference frame: Frame 1, J2000
   SPK Data Type  : Type 3
      Description : Fixed Width, Fixed Order Chebyshev Polynomials: Pos, Vel
   UTC Start Time : 1969 DEC 31 00:00:00.000
   UTC Stop Time  : 2019 DEC 02 00:00:00.000
   ET Start Time  : 1969 DEC 31 00:00:41.183
   ET Stop time   : 2019 DEC 02 00:01:05.183
-----------------------------------------------------------------------------
                                      :
                                      :
```
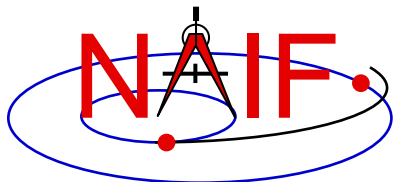
(This is a partial output; not all data could be displayed on this chart)
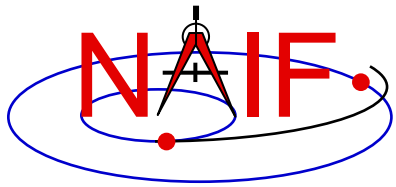
Summarizing an SPK at the API Level

- **Call SPKOBJ to find the set of objects for which a specified SPK provides data.**
  - INPUT: an SPK file name and initialized SPICE integer "Set" data structure. The set may optionally contain ID codes obtained from previous calls.
  - OUTPUT: the input set, to which have been added (via set union) the ID codes of objects for which the specified SPK provides coverage.

        CALL SPKOBJ (SPK, IDSET)

- **Call SPKCOV to find the window of times for which a specified SPK file provides coverage for a specified body:**
  - INPUT: an SPK file name, body ID code and initialized SPICE double precision "Window" data structure. The window may optionally contain coverage data from previous calls.
  - OUTPUT: the input window, to which have been added (via window union) the sequence of start and stop times of segment coverage intervals of the specified SPK, expressed as seconds past J2000 TDB.

        CALL SPKCOV (SPK, IDCODE, COVER)

- **See the headers of these routines for example programs.**
- **Also see the CELLS, SETS and WINDOWS Required Reading for background information on these SPICE data types.**

Additional Information on SPK

- **For more information about SPK, look at the following:**
  - **The remainder (the BACKUP section) of this tutorial**
  - **Most Useful Routines document**
  - **SPK Required Reading document**
  - **Headers of the subroutines mentioned**
  - **Using Frames tutorial**
  - **BRIEF and SPKDIFF User's Guides**

- **Related documents:**
  - **NAIF_IDS Required Reading**
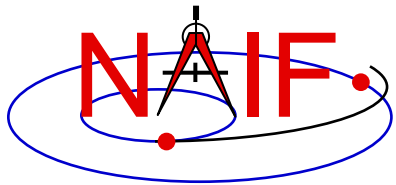 - **Frames Required Reading**
  - **Time Required Reading**
  - **Kernel Required Reading**
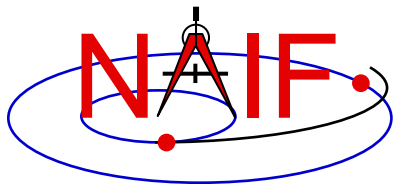
Backup

- **Problems Using SPK Files**

- **Don't Mix Planet Ephemerides**

- **Effect of Aberration Corrections**

- **Examples of Retrieving State Vectors**

- **SPK File Structure**

- **The file, or files, you loaded do not contain data for <u>both</u> your target and observer bodies**
  - You may have loaded the wrong file, or assumed the file contains data that it doesn't
  - You may not have loaded all the files needed
- **The file, or files, you loaded do not cover the <u>time</u> at which you requested a state vector**
  - This could occur if you've been given a file coverage summary in calendar ET form and you mistook this for UTC
    (ET = UTC + DELTAET, where DELTAET is about 69 seconds as of 1/17)
  - This could occur if you are requesting a light-time corrected state and the SPK files being used do not have data at the time that is one-way light-time away* from your ET epoch of interest
    - » * Earlier, for the receive case; later, for the transmission case
- **In the above situations you'll get an error message like the following:**

```
SPICE(SPKINSUFFDATA) - -
Insufficient ephemeris data has been loaded to
compute the state of xxx relative to yyy.
```
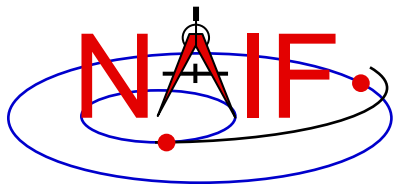
- **You have requested aberration-corrected states but the SPK file or files you loaded do not contain sufficient data to relate both your target and observer bodies back to the solar system barycenter, which is required for this calculation.**
    - **You may not have loaded all the files needed**
    - **You may have assumed the file contains data that it doesn't**

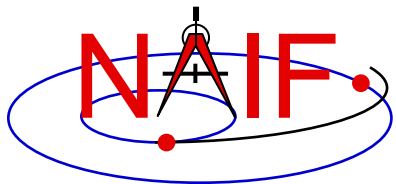- **In the above situations you'll get an error message like the following:**

```
SPICE(SPKINSUFFDATA) - -
Insufficient ephemeris data has been loaded to
compute the state of xxx relative to yyy.
```

- **An infrequent problem occurs when your SPK file(s) contain data for both target and observer, and cover the period of interest, but ephemeris data for an intermediate body needed to chain the target and observer together is missing.**

  – **Example: You load a spacecraft SPK containing ephemeris for Cassini (-82) relative to the solar system barycenter (0), and you load a satellite SPK containing the ephemeris for Titan (606) and Saturn (699) relative to the Saturn barycenter (6). But you forgot to load a planet SPK file that contains data for the Saturn barycenter (6) relative to the solar system barycenter (0). The SPK software cannot "connect" Cassini to Titan or to Saturn. (See the drawing on the next page.)**

  – **In this case, knowing what is the "Center Body" of movement for each target body is important; this is shown in SPACIT, and also in BRIEF summaries when the -c command line option is used.**

Navigation and Ancillary Information Facility
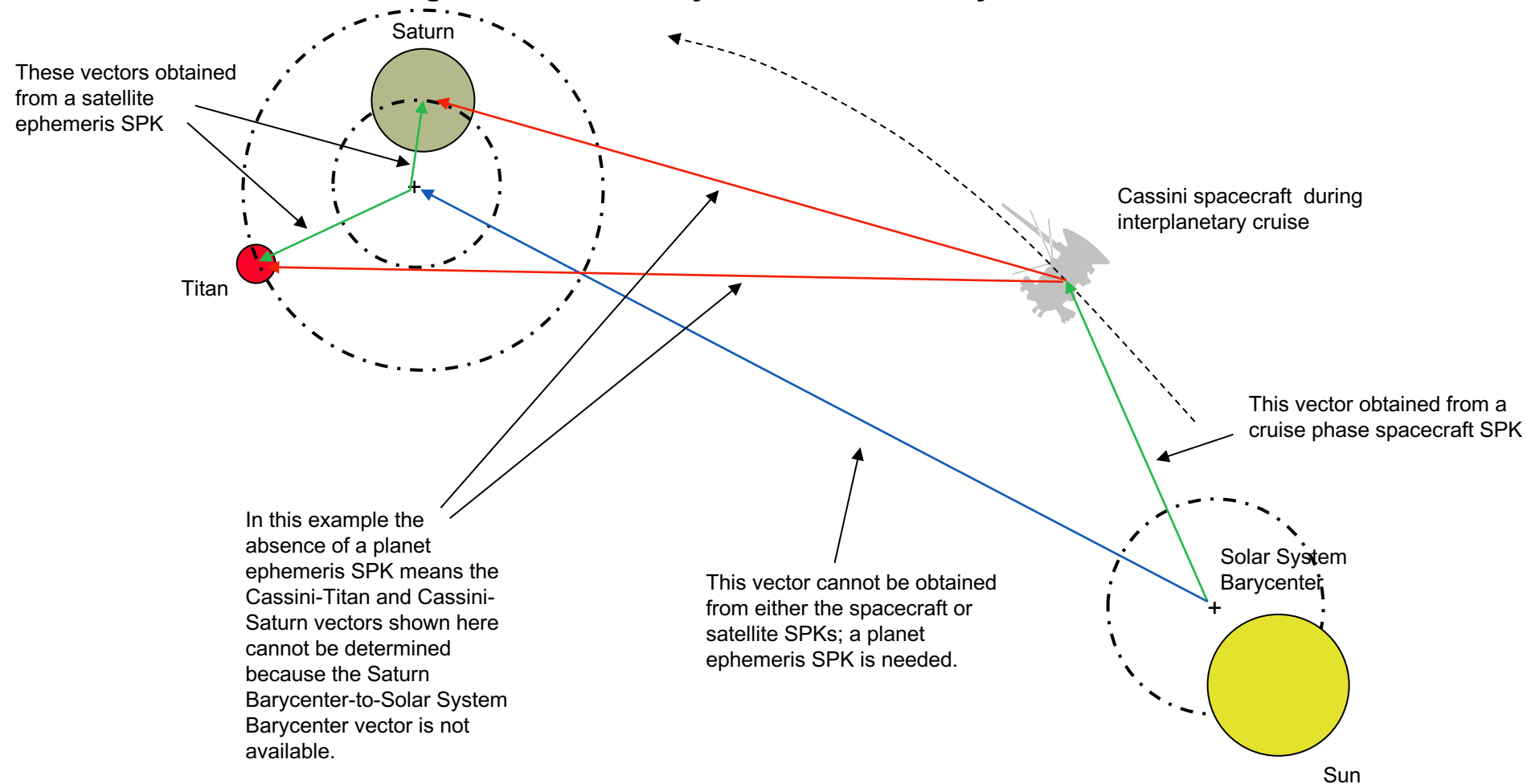
Saturn

These vectors obtained from a satellite ephemeris SPK

Titan

Cassini spacecraft during interplanetary cruise

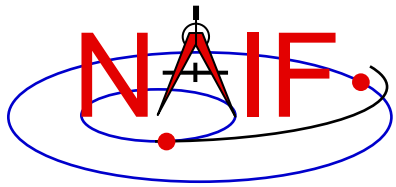This vector obtained from a cruise phase spacecraft SPK

In this example the absence of a planet ephemeris SPK means the Cassini-Titan and Cassini-Saturn vectors shown here cannot be determined because the Saturn Barycenter-to-Solar System Barycenter vector is not available.

This vector cannot be obtained from either the spacecraft or satellite SPKs; a planet ephemeris SPK is needed.

Solar System Barycenter

Sun

Given…

———— available in SPK files

———— not available in SPK files
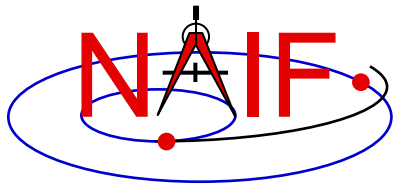
Therefore…

———— can't be computed

SPK Subsystem

55

- **You see an error message to the effect that pole RA (right ascension) data cannot be found**
  - You are requesting results in a body-fixed frame, but you have not loaded a SPICE PCK file that defines this frame.
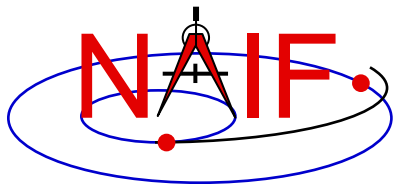
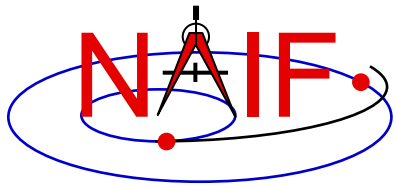**Navigation and Ancillary Information Facility**

- **Segment Masking: You've loaded sufficient data to chain together the target and observer, but the SPK subsystem can't make the connection.**
  - **This can happen when a high-priority segment that can't be connected to both target and observer "masks" a lower-priority segment that can be connected.**
  - **Example: you want the state of earth as seen from the Galileo orbiter at a specified ephemeris time ET1.**
    - » **You have loaded SPK files providing:**
      - **the state of the Galileo orbiter relative to the asteroid Gaspra**
      - **the state of the orbiter relative to the sun**
      - **the state of the earth relative to the earth-moon barycenter**
      - **the states of the sun and earth-moon barycenter relative to the solar system barycenter**
    - » **If an SPK segment for the orbiter relative to Gaspra covering ET1 has higher priority than the segment for the orbiter relative to the sun covering ET1, no connection between the orbiter and the earth will be made.**
    - » **Solution:**
      - **Load an SPK file providing the ephemeris of Gaspra relative to the sun or the solar system barycenter (for a time interval containing ET1)**

SPK Subsystem

Navigation and Ancillary Information Facility

- **Other missing data… not obvious.**
  - You may need CK (and SCLK), FK or PCK data to construct your requested output frame.
- **Mistaking ET for UTC, or vice-versa: very common**
- **Using light time corrections requires target ephemeris data at the light time-corrected epoch.**
  - If you're working near the beginning of an SPK file, the light time-corrected epoch may occur earlier than available data.
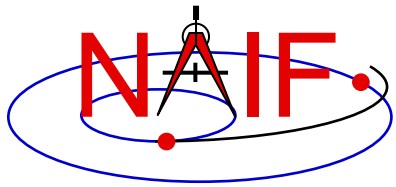
Navigation and Ancillary Information Facility

- **You've assumed that:**

  negative sign

  state (observer, target)   =   - state (target, observer)

- **This is NOT true unless you have requested geometric states in both cases (i.e. no light time or stellar aberration corrections are applied)**
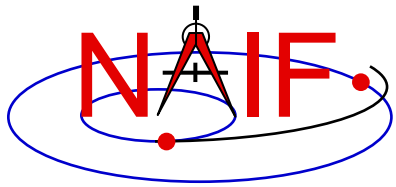
- **With each new version of the JPL planetary ephemeris, the solar system barycenter moves with respect to the planets**

- **Changes in relative planet positions are much smaller than changes in the planet locations relative to the solar system barycenter**
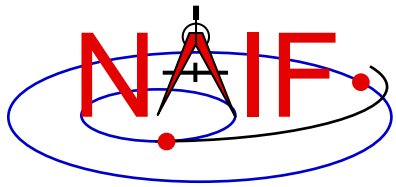

S.S.Barycenter

S.S.Barycenter

Earth

Mars
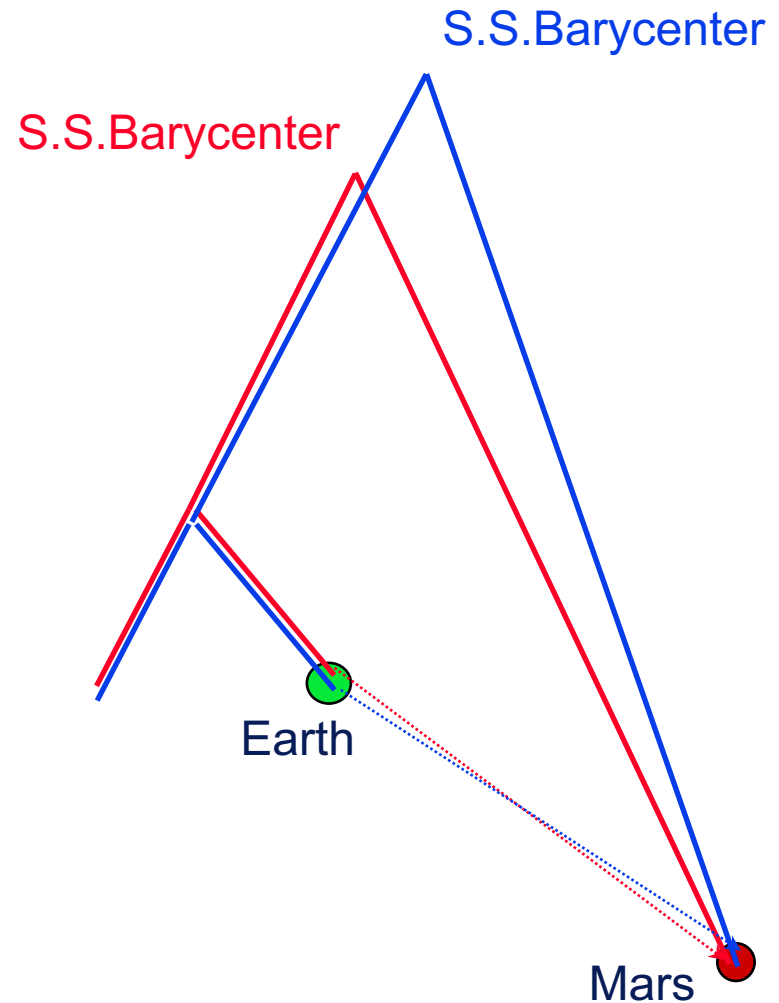
Navigation and Ancillary Information Facility

- **SPICE allows you to "load" different planetary ephemerides (or portions of them)**

  » **You can potentially subtract the solar system barycenter-relative positions from different ephemerides to get relative states**

- **Don't mix planetary ephemerides**

- **For JPL flight projects, a consistent set of ephemerides is provided to the mission team**
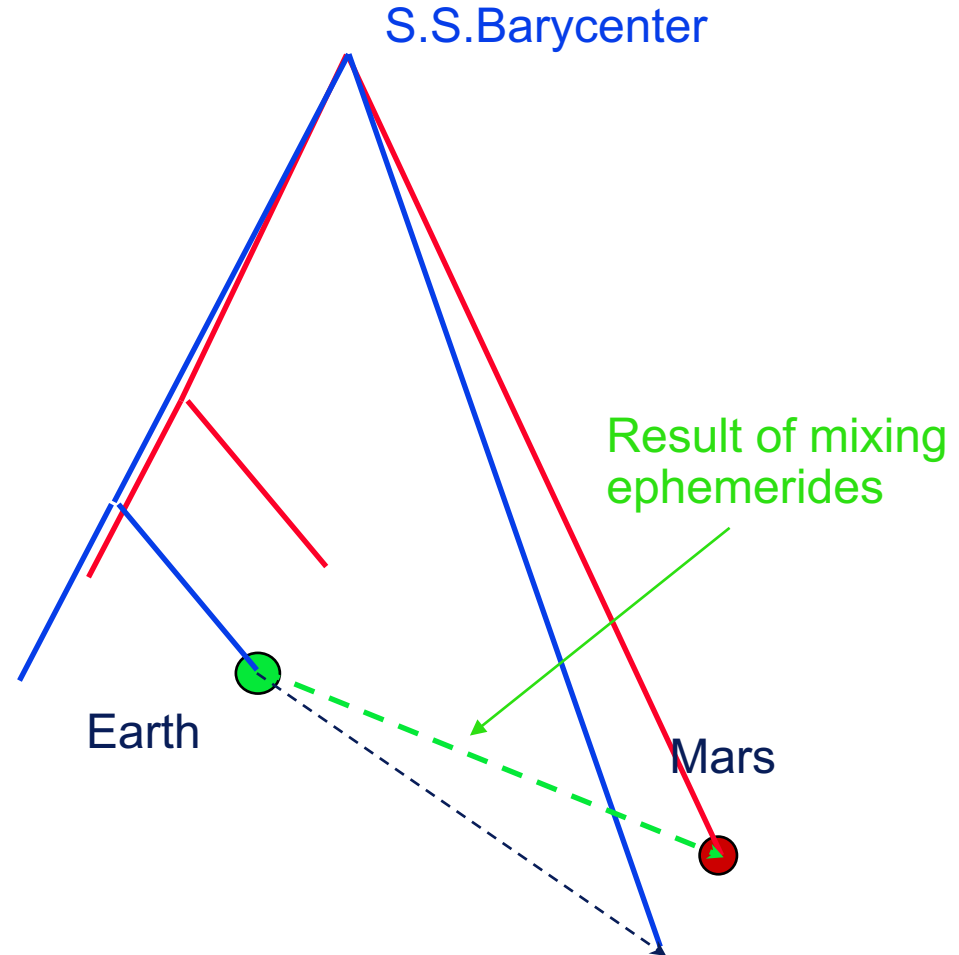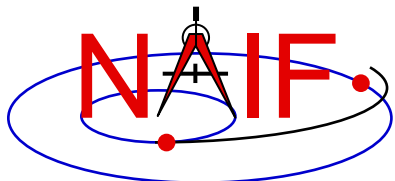
S.S.Barycenter

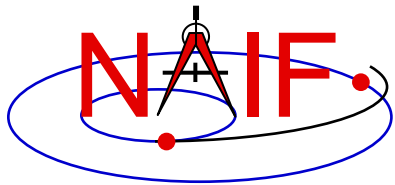Result of mixing ephemerides

Earth

Mars

- **Angular offsets between corrected and uncorrected position vectors over the time span 2004 Jan 1 to 2005 Jan1**
    - **Mars as seen from MEX:**
        - » **LT+S vs NONE:** .0002 to .0008 degrees
        - » **LT vs NONE:** .0006 to .0047 degrees
    - **Earth as seen from MEX:**
        - » **LT+S vs NONE:** .0035 to .0106 degrees
        - » **LT vs NONE:** .0000 to .0057 degrees
    - **MEX as seen from Earth:**
        - » **LT+S vs NONE:** .0035 to .0104 degrees
        - » **LT vs NONE:** .0033 to .0048 degrees
    - **Sun as seen from Mars:**
        - » **LT+S vs NONE:** .0042 to .0047 degrees
        - » **LT vs NONE:** .0000 to .0000 degrees

Effect of Aberration Corrections - 2

- **Angular offsets between corrected and uncorrected position vectors over the time span 2004 Jan 1 to 2008 Jan1**
  - **Saturn as seen from CASSINI:**
    - » **LT+S vs NONE: .0000 to .0058 degrees**
    - » **LT vs NONE: .0001 to .0019 degrees**
  - **Titan as seen from CASSINI:**
    - » **LT+S vs NONE: .0000 to .0057 degrees**
    - » **LT vs NONE: .0000 to .0030 degrees**
  - **Earth as seen from CASSINI:**
    - » **LT+S vs NONE: .0000 to .0107 degrees**
    - » **LT vs NONE: .0000 to .0058 degrees**
  - **CASSINI as seen from Earth:**
    - » **LT+S vs NONE: .0000 to .0107 degrees**
    - » **LT vs NONE: .0000 to .0059 degrees**
  - **Sun as seen from CASSINI:**
    - » **LT+S vs NONE: .0000 to .0059 degrees**
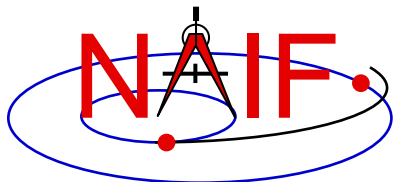   - » **LT vs NONE: .0000 to .0000 degrees**

- **Example: find the geometric state of the MGS orbiter relative to Mars at the observation epoch ET, expressed in the J2000 reference frame.**
  - **CALL SPKEZR ('MGS', ET, 'J2000', 'NONE', 'MARS', STATE, LT)**
  - **The SPK subsystem locates an SPK segment containing the ephemeris of the orbiter relative to Mars covering epoch ET, interpolates the ephemeris data at ET, and returns the interpolated state vector.**

- **Example: find the geometric state of Titan relative to the earth at epoch ET, expressed in the J2000 reference frame.**
  - **CALL SPKEZR ('TITAN', ET, 'J2000', 'NONE', 'EARTH', STATE, LT)**
  - **The SPK subsystem looks up and interpolates ephemeris data to yield:**
    - » **The state of the earth relative to the earth-moon barycenter (A)**
    - » **The state of the earth-moon barycenter relative to the solar system barycenter (B)**
    - » **The state of Titan relative to the Saturn system barycenter at ET (C)**
    - » **The state of the Saturn system barycenter relative to the solar system barycenter at ET (D)**
 - **SPKEZR then returns the state vector**
    - » **C + D – (A + B)**

- **Example: find the apparent state of the Cassini orbiter relative to the DSN station DSS-14, expressed in the topocentric reference frame centered at DSS-14, at a specified observation epoch ET.**

  - **CALL SPKEZR ('CASSINI',   ET,   'DSS-14_TOPO',**

    'LT+S',   'DSS-14',   STATE,   LT   )

  - **The SPK subsystem looks up and interpolates ephemeris data to yield:**

    - » **The state of DSS-14 relative to the earth in the ITRF93 terrestrial reference frame (A)**

    - » **The state at ET of the earth relative to the earth-moon barycenter in the J2000 reference frame (B)**

    - » **The state at ET of the earth-moon barycenter relative to the solar system barycenter in the J2000 frame (C)**

    - » **The state at the light time-corrected epoch ET-LT of the Cassini orbiter relative to the Saturn system barycenter (other centers are possible) in the J2000 frame (D)**
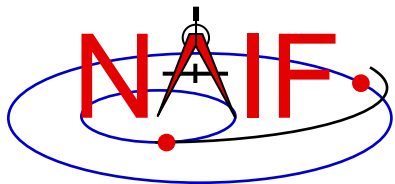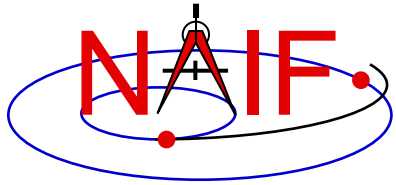
Navigation and Ancillary Information Facility

- » **The state at ET-LT of the Saturn system barycenter relative to the solar system barycenter in the J2000 frame (E)**
- – **The SPK subsystem also looks up transformation matrices to map states:**
  - » **From the J2000 frame to the ITRF93 terrestrial (earth body-fixed) frame at the observation epoch ET (T1)**
  - » **From the ITRF93 terrestrial frame to the DSS-14-centered topocentric frame (T2)**
- – **SPKEZR then computes the J2000-relative, light-time corrected observer-target state vector**
  - » $E + D - ((T1)^{-1} *A + B + C))$
- – **SPKEZR corrects this vector for stellar aberration**
  - » **Call the result "V_J2000_apparent"**
- – **and finally returns the requested state vector in the DSS-14 topocentric reference frame**
  - » **STATE = T2 * T1 * V_J2000_apparent**

- **A description of the SPK file structure is shown near the beginning of the "Making an SPK" tutorial.**