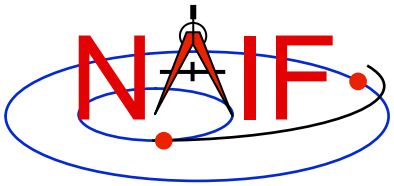


Navigation and Ancillary Information Facility

Introduction to the SPICE Ephemeris Subsystem SPK

Focused on reading SPK files

March 2010

SPICE Ephemeris Data

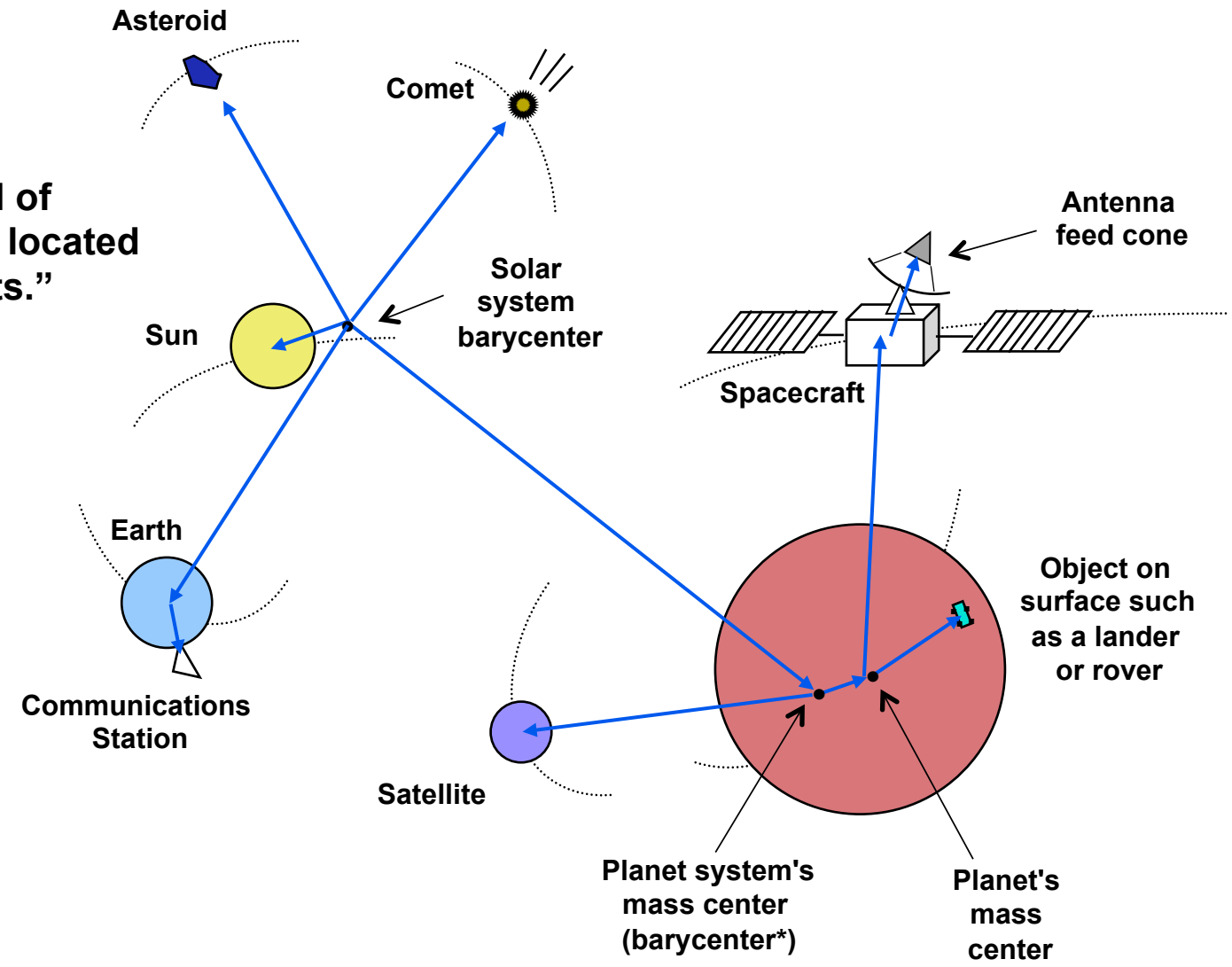
Navigation and Ancillary Information Facility

- **An SPK file contains ephemeris (trajectory) data for "ephemeris objects."**
  - **"Ephemeris" means position and velocity as a function of time.**
- **Spacecraft, planets, satellites, comets and asteroids are the obvious kinds of "ephemeris objects," but many other possibilities exist, such as:**
  - **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**
  - **the center of mass of a planet/satellite system (planet barycenter)**
  - **the center of mass of our solar system (solar system barycenter)**
- **See the next page for a pictorial representation of some of these objects.**

Examples of Ephemeris Objects

Navigation and Ancillary Information Facility

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



*A barycenter is the center of mass of a set of bodies, such as Saturn plus all of Saturn's satellites.

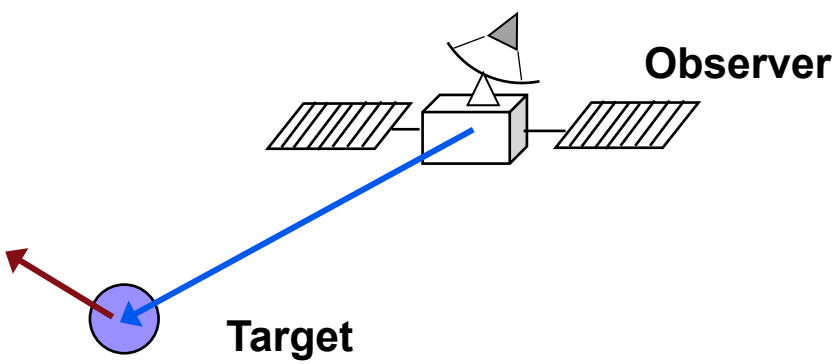
Inside an SPK: Bodies and Centers of Motion

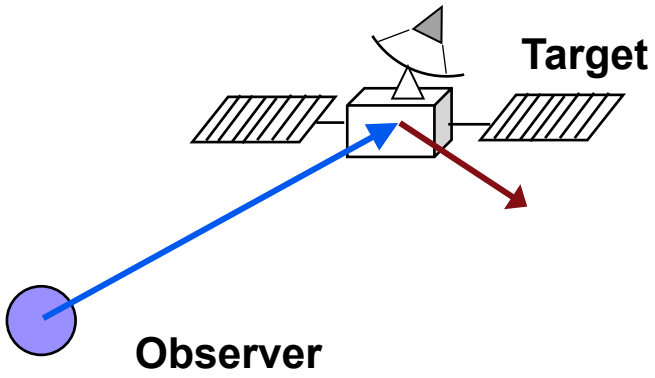
- **Inside an SPK file ephemeris objects come in pairs: a "body" and its "center of motion."**
  - **The ephemeris is given for the body moving relative to the center of motion.**
    - » **For the position component, the vector points TO the body FROM the center of motion.**
  - **There can be, and often are, multiple such pairs within an SPK file.**

Reading an SPK: Observers and Targets

Navigation and Ancillary Information Facility

- 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."





Caution: state (observer, target) ≠ - state (target, observer)

unless the state is geometric (no aberration corrections).

SPK File Coverage

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.
    - » Example: a planetary SPK file such as de421.bsp
    - » Counterexample: Cassini tour SPK with merged Huygens probe ephemeris

- For any request time within any time interval comprising the coverage for an object, the SPK system can return a vector representing the state of that body 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.

Reference Frames as Used in Writing and Reading SPKs

Navigation and Ancillary Information Facility

On Writing

- **All ephemeris data in an SPK file have an associated reference frame**
  - **There could be multiple such frames, each for a different portion of the data**
  - **For the ephemeris data to be useful, this/these frames must be "known" to any program that will subsequently read the ephemeris data**

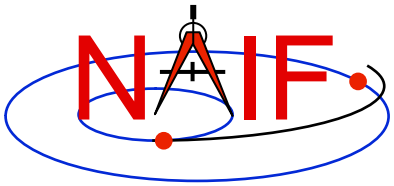
On Reading

- **The application "reading" an SPK file(s) must specify relative to what reference frame the output state or position vectors are to be given**
  - **This frame must be "known" to the SPICE-based program**

- **"Known" means either a built-in frame ("hard coded") or one fully specified at run-time**
  - **The user's program may need to have access to additional SPICE data in order to construct some of these frames**

Using SPK Files

A Brief Introduction



Retrieving Position or State Vectors

- To retrieve position or state vectors of ephemeris objects from an SPK file one normally needs two kinds of SPICE kernels
  - Ephemeris kernel(s) (SPK)
    - » Sometimes just one is needed
    - » Sometimes two or more are needed to chain together the "target" and "observer" you have selected
  - Leapseconds kernel (LSK)
    - » Used to convert between Coordinated Universal Time (UTC) and Ephemeris Time (ET)
    - » Usually needed since most people work with UTC time
- 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 and/or 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
  - X, Y, Z for a position vector

Retrieving a State Vector

Navigation and Ancillary Information Facility

Initialization…typically done once per program execution

Fortran syntax used here

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

```
CALL FURNSH  ('spk_file_name')
CALL FURNSH  ('leapseconds_file_name')
```

Better yet, 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

Use the returned state vector in other SPICE routines to compute observation geometry of interest.

Arguments of SPKEZR - 1

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 your output state vector is to be given. SPK software will automatically convert data to the frame you specify (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.**

Arguments of SPKEZR - 2

- **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.**
  - **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.**

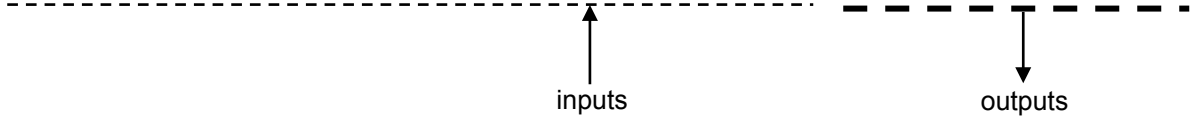
Retrieving a Position Vector

Navigation and Ancillary Information Facility

- The SPICE routine 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
- When velocity data are not needed, using SPKPOS offers several advantages over using SPKEZR
  - 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

Retrieve a position vector from the SPK file at your requested time

CALL SPKPOS (target, tdb, 'frame', 'correction', observer, *positn, light time*)



Use the returned position vector in other SPICE routines

A Simple Example of Retrieving State Vectors

Navigation and Ancillary Information Facility

Initialization - typically do this just once per program execution

```
CALL FURNSH ( 'NAIF0009.TLS' )
CALL FURNSH ( 'HUYGENS_3_MERGE.BSP' )
```

Better to use a "furnsh kernel" instead of these individual FURNSH statements

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', 'HUYGENS PROBE',
             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 Huygens probe 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 (NAIF0009.TLS) is used, as is a SPICE ephemeris file (HUYGENS_3_MERGE.BSP) containing ephemeris data for the Huygens probe (-150), Saturn barycenter (6), Saturn mass center (699), Saturn's satellites (6xx) and the sun (10), relative to the solar system barycenter.

A Slightly More Complex Example - 1 Kernel Data Needed

- **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 planet in the planet's IAU body-fixed reference frame you'll need:**
    - **PcK file containing orientation data for the planet**
    - **SPK(s) for the object, planet, and (typically) planet barycenter**
    - **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 - 2
Retrieving State Vectors

Obtaining a state vector in a body-fixed reference frame

Initialization...typically once per program execution

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

```
CALL FURNSH  ('spk_file_name')
CALL FURNSH  ('leapseconds_file_name')
CALL FURNSH  ('pck_file_name')
```

Better to use a "furnsh kernel" instead of these three individual FURNSH statements

Loop... do as many times as you need

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

```
CALL STR2ET ( 'utc_string', tdb)
```

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

```
CALL SPKEZR (target, tdb, 'IAU_<body_name>', 'correction',
  observer, state, lightime)


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

Chaining Data

- If needed, the SPK software will automatically chain together two or more state vectors needed to connect your "target" to your "observer." (See next chart.)
- SPK software can chain together state vectors provided by a single SPK file, or by multiple SPK files.
- In doing the chaining, if needed the SPK software will also transform the various state vectors into a common reference frame for addition or subtraction, then transform the result to the reference frame you have selected for output.
  - Your selected output reference frame must be one known to the SPICE system, and your application program must have available all needed SPICE data to construct this reference frame.
- See the chaining example on the next page.

Example of Chaining

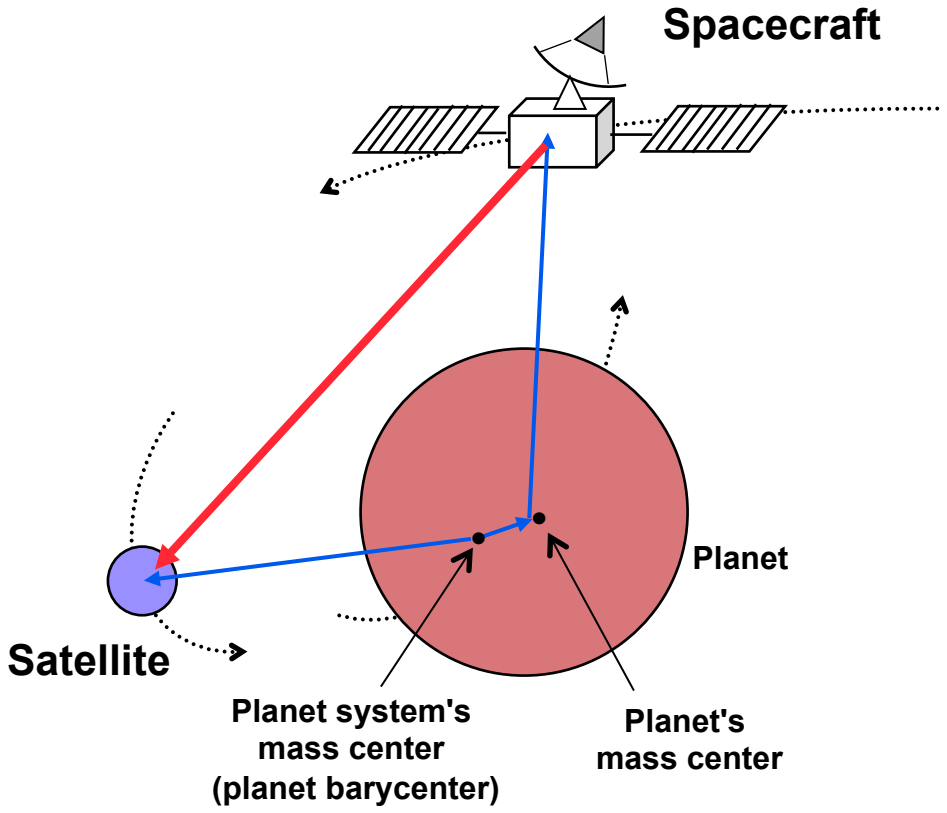
Navigation and Ancillary Information Facility

Suppose you ask for the position of the satellite relative to the spacecraft.

Your SPK may not contain exactly the ephemeris you want.

But, if all the needed data are available in your SPK file, the SPK subsystem will chain together the position vectors indicated by the three blue arrows–the data explicitly contained in an SPK file–to give you the position vector indicated by the red arrow–the one you asked for.

This might require the loading of two SPK files, one containing data for the spacecraft relative to the planet mass center, and another containing data for the planet mass center and the satellite relative to the planet barycenter.



"Planet system" = the planet and all of it's satellites

More About SPK Files

Here we provide some more details on the design and structure of SPK files.

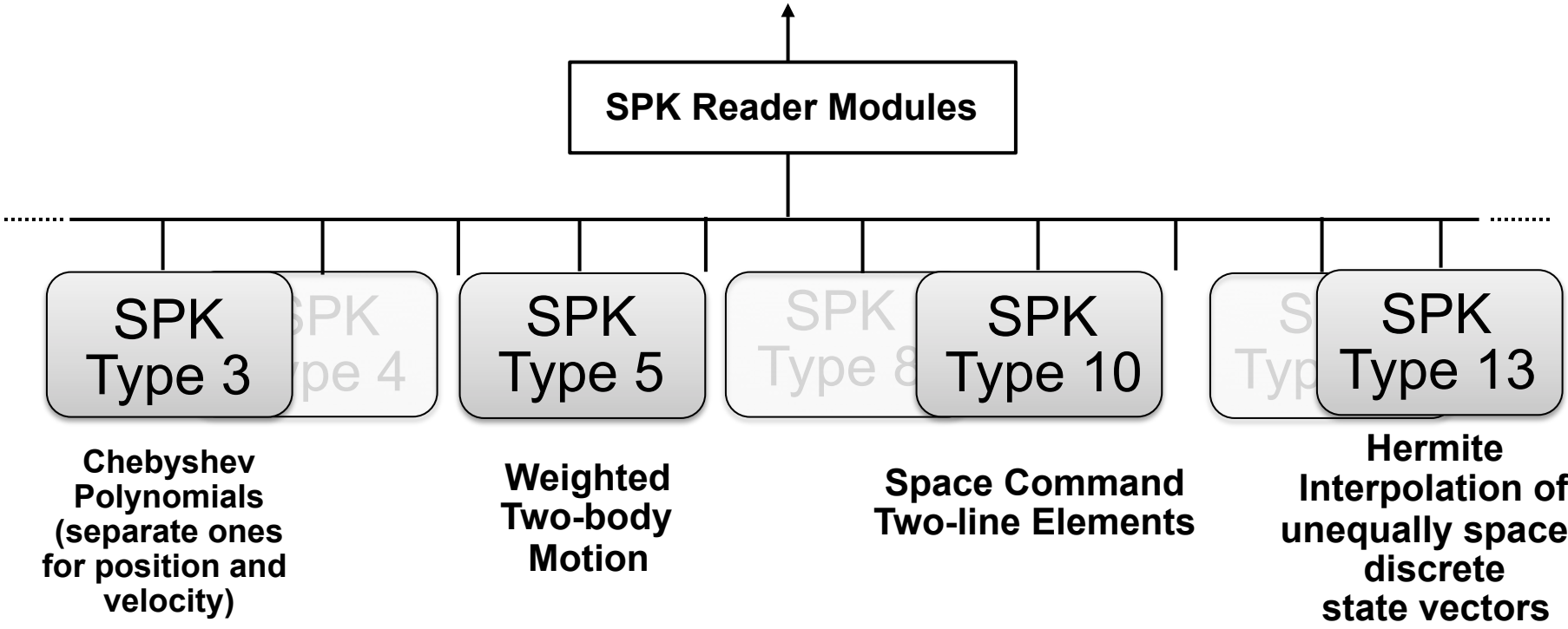
However, still more information is provided in the "Making an SPK" tutorial.

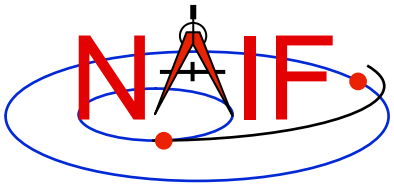
SPK Data Type Concept

Navigation and Ancillary Information Facility

SPK files contain various mathematical representations of ephemeris data ("data types"), but the high-level user interfaces (SPK "readers") are type-independent:

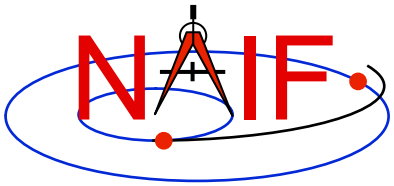
```
CALL SPKEZR ('target', time, 'frame', 'correction', 'observer', state,   light_time)

CALL SPKPOS ('target', time, 'frame', 'correction', 'observer', positn, light_time)
```





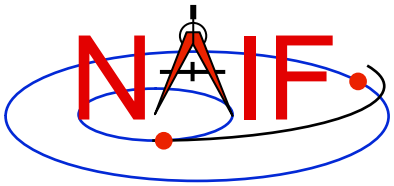
Why Have Multiple Data Types?

- To allow SPK producers to choose an ephemeris representation well-suited for their applications. 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. The file creator must do more work to use these data types.
- To replicate data originally provided in other formats.
  - Types 1, 2, 3, 8, 10, 14, 15, 17 and 18 were developed to enable accurate duplication of data obtained from ephemeris developers.



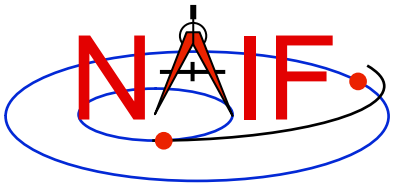
Widely Used SPK Data Types

- **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 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 (Sliding window Hermite or Lagrange interpolation)**
  - Used in SPKs made by ESA missions



SPK Segments - 1

- **An SPK file is mostly made up of one or more blocks of ephemeris data called "segments."**
  - **Each segment contains position and velocity data for one body, relative to one center of motion, given in one reference frame, using one mathematical representation (called the SPK "data type"), for a specified time interval.**
  - **Data for one body, relative to one center of motion, in one reference frame, using one SPK data type may be placed in many segments. This is often the case for spacecraft.**
  - **You can observe the segment-by-segment contents of an SPK file using the SPACIT utility program.**



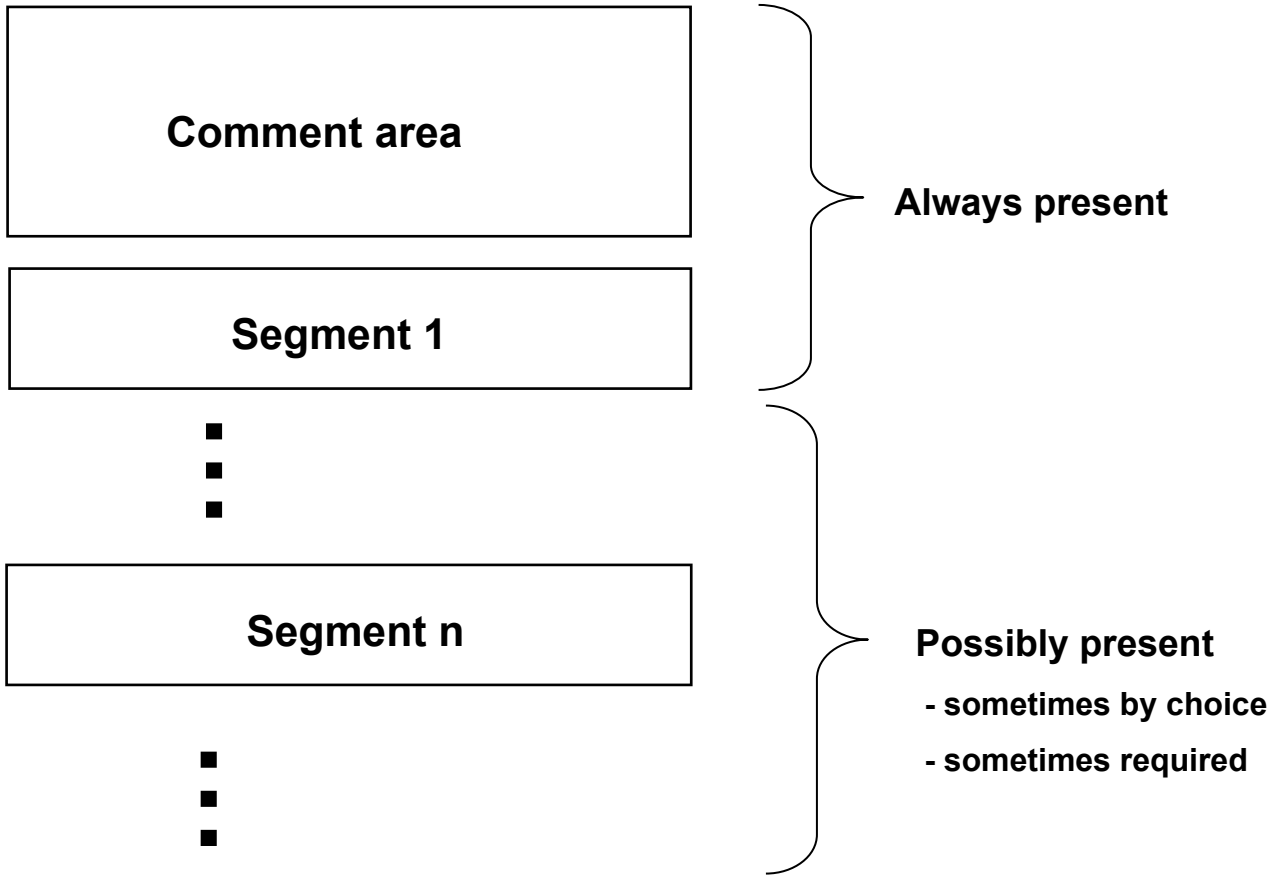
SPK Segments - 2

- An SPK segment, by definition, contains a continuous representation of ephemeris data over an interval of time.
  - You can request a position or state vector at any time within the covered interval.
    - » Achieved through interpolation of discrete data, or by evaluating an analytical function, depending on the SPK type.
- However, interpolation across segments and extrapolation beyond the time bounds of a segment are not permitted by the SPK "readers."

SPK File Structure: The User's View

Navigation and Ancillary Information Facility

Logical Organization of an SPK File



See the tutorial named Making an SPK for details about segment architecture and contents.

SPK Segment Order and Priority

Navigation and Ancillary Information Facility

- **Within a single SPK file…**
  - The segments in an SPK file need not be ordered according to time or body.
  - Segment order does imply priority. When 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 located later in the file.
    - » The centers of motion, frames and SPK types are irrelevant to this selection.

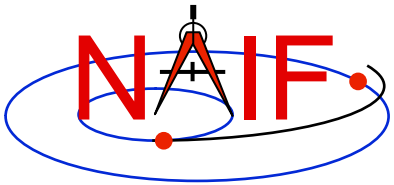
- **If using two or more SPK files…**
  - Segments from SPK files loaded later have higher priority: when two segments from two different SPK files both contain data for a given target and time, the SPK system selects the segment from the SPK file that was loaded later.

Planets and Planet Barycenters

Navigation and Ancillary Information Facility

- A planet and its satellites orbit the planet system's barycenter
  - For example, the Jupiter mass center (599) and each of Jupiter's satellites (501 - 5xx) orbit the Jupiter system barycenter (5)
- Planet system barycenters (i.e. 1 through 9) and the sun (10) orbit the solar system barycenter (0)
- Because Mercury and Venus have no satellites, their barycenters (1 and 2) occupy the same locations as their mass centers (199 and 299)*
- Because the masses of Phobos and Deimos are so small compared to the mass of Mars, the mass center for Mars (499)* is treated as equivalent to the Mars barycenter (4)

* These equivalences hold true ONLY in the SPK subsystem, not in the PCK subsystem



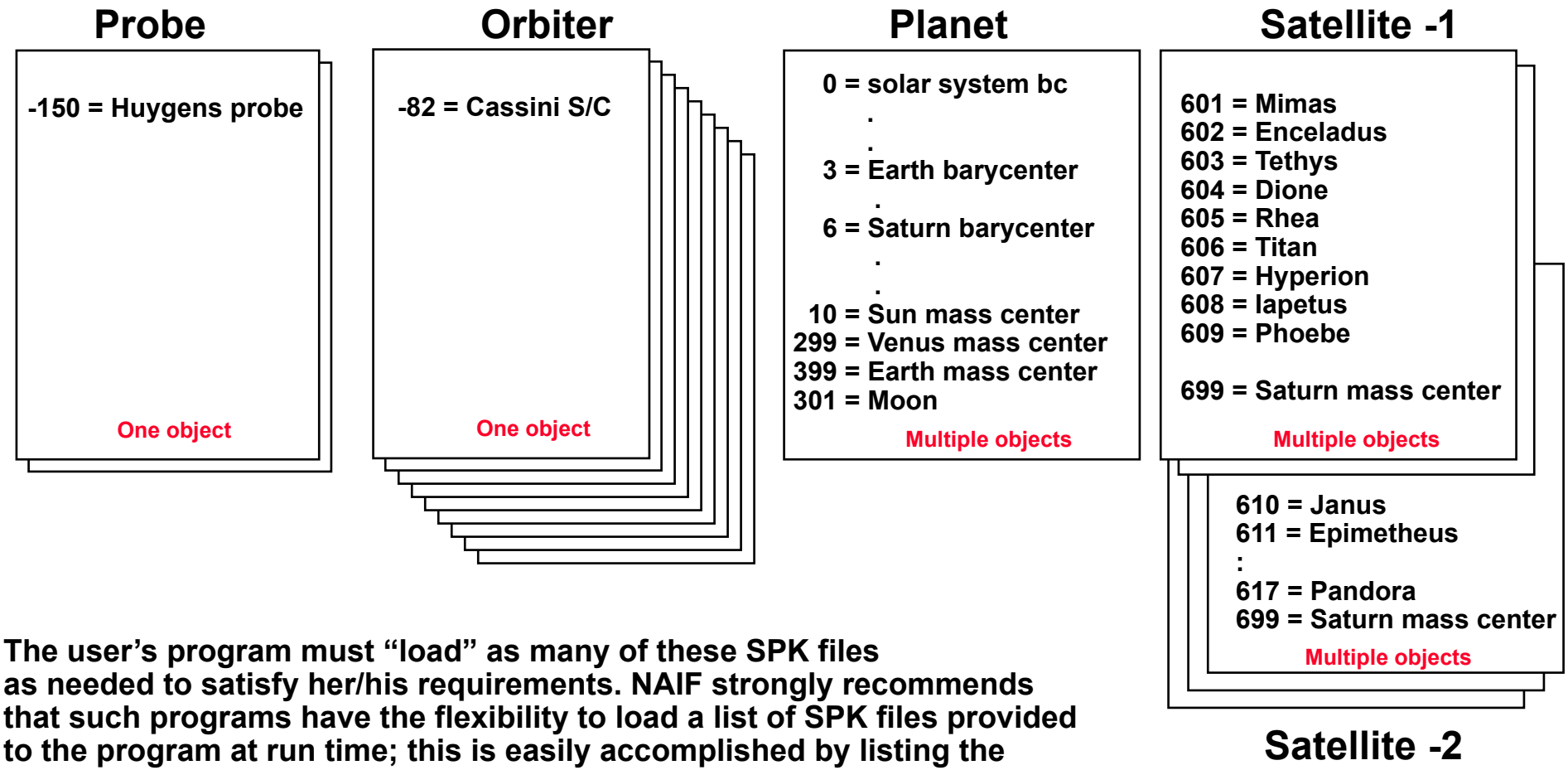
SPK File Ephemeris Contents

- **A single SPK file can hold data for one ephemeris object, or for many ephemeris objects.**
  - **This is illustrated in the next three charts**

Example of Flight Project SPK Files

Navigation and Ancillary Information Facility

This made up example shows four collections of SPK files for the Cassini-Huygens mission.

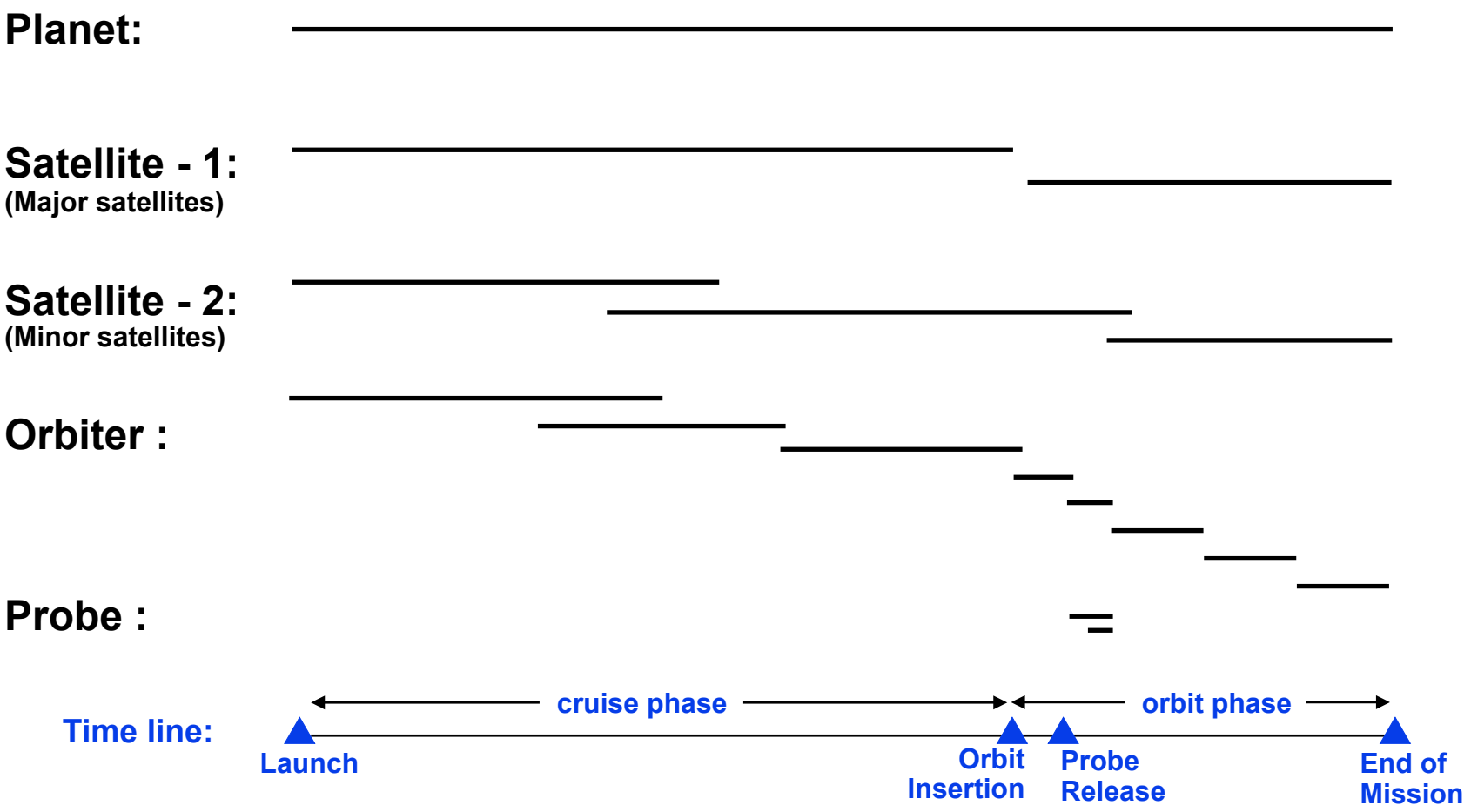


The user's program must "load" as many of these SPK files as needed to satisfy her/his requirements. NAIF strongly recommends that such programs have the flexibility to load a list of SPK files provided to the program at run time; this is easily accomplished by listing the SPK files in a "furnsh kernel" using the Toolkit's FURNSH routine.

Possible* SPK File Time Coverages for the Previous Example

Navigation and Ancillary Information Facility

Each bar represents a separate file



* Note: This is likely not a real Cassini scenario; it is simply an illustration of some of the possibilities for ephemeris delivery on a planetary mission.

Examples of Generic SPK File Contents

Navigation and Ancillary Information Facility

de421.bsp
Planet Ephemeris

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

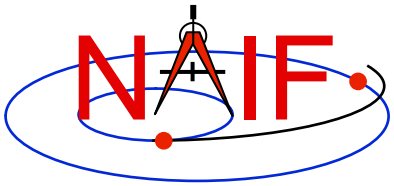
my_asteroids.bsp
Asteroid Ephemeris

2000253	Mathilde
2000433	Eros

jup230.bsp
Merged Planet and Satellite Ephemeris

3	Earth BC
399	Earth
5	Jupiter BC
501	Io
502	Europa
503	Ganymede
504	Callisto
505	Amalthea
514	Thebe
599	Jupiter
10	Sun

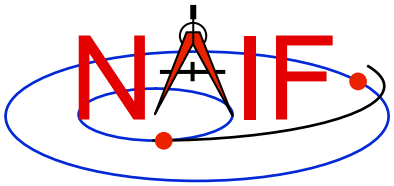
Generic SPK files can be obtained from the NAIF server.



SPKs for Objects Located on the Surface of a Natural Body

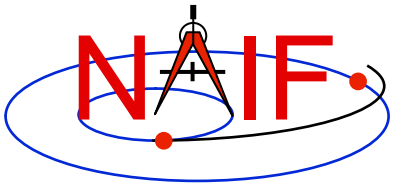
- **An SPK file may contain positions of tracking stations, observatories, roving vehicles, etc.**
  - **The object could be stationary or moving**

- **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**



Understanding an SPK File

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



Manipulating and Using SPK Files

- You can subset an SPK file, or merge two or more files
  - The merge may key off of objects, or time, or both
- 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

(* The allowed number of simultaneously loaded DAF-based files is currently set to 1000.)

Summarizing an SPK File - 1

Navigation and Ancillary Information Facility

- A brief summary can be made using the SPICE Toolkit utility “brief”
  - Summary is for objects present and their start and stop epochs
- At your command line prompt, type the program name (with path), followed by the name of the binary SPK file that is to be summarized
- See the brief User’s Guide or on-line help (%brief -h) for more details

```
% brief 070413BP_SCPSE_07097_07121.bsp
Brief.  Version: 2.3.1        (SPICE Toolkit N0061)


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 is ET, not UTC!

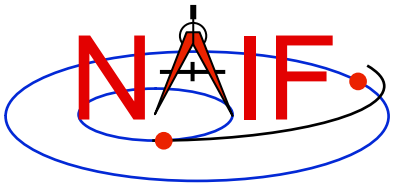
Summarizing an SPK File - 2

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 File - 3

Navigation and Ancillary Information Facility

Summarizing an SPK at the API Level

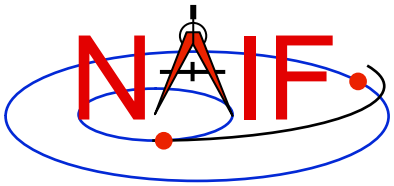
- **Call SPKOBJ to find the set of objects for which a specified SPK provides data.**
  - **INPUT: an SPK file name and initialized SPICE integer "Set" data structure. The set may optionally contain ID codes obtained from previous calls.**
  - **OUTPUT: the input set, to which have been added (via set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



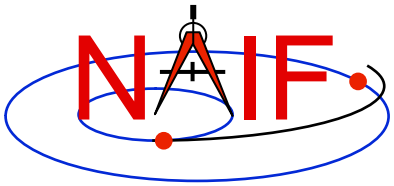
SPK Utility Programs

- **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 or merges one or more SPK files

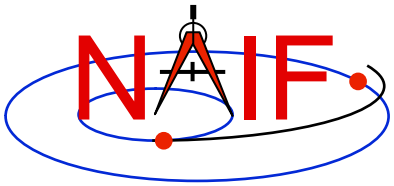
- **These additional SPK utility programs are provided on the NAIF Web site (http://naif.jpl.nasa.gov/naif/utilities.html)**

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 IEEE and PC binary formats



Additional Information on SPK

- For more information about SPK, look at the following:
  - Backup slides in this tutorial
  - STATES cookbook program (source code) and its User's Guide
  - 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

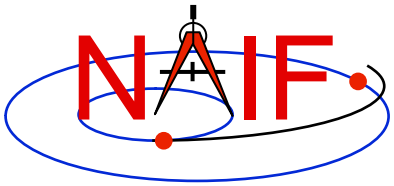
- Problems Using SPK Files
- Don’t Mix Planet Ephemerides
- Barycenters and Mass Centers
- Effect of Aberration Corrections
- Retrieving State Vectors: "Under the Hood”
- SPK File Structure

Problems Using SPK Files - 1

Navigation and Ancillary Information Facility

- The file, or files, you loaded do not contain data for both 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 time 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 65 secs as of 10/1/08)
  - 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.
```



Problems Using SPK Files - 2

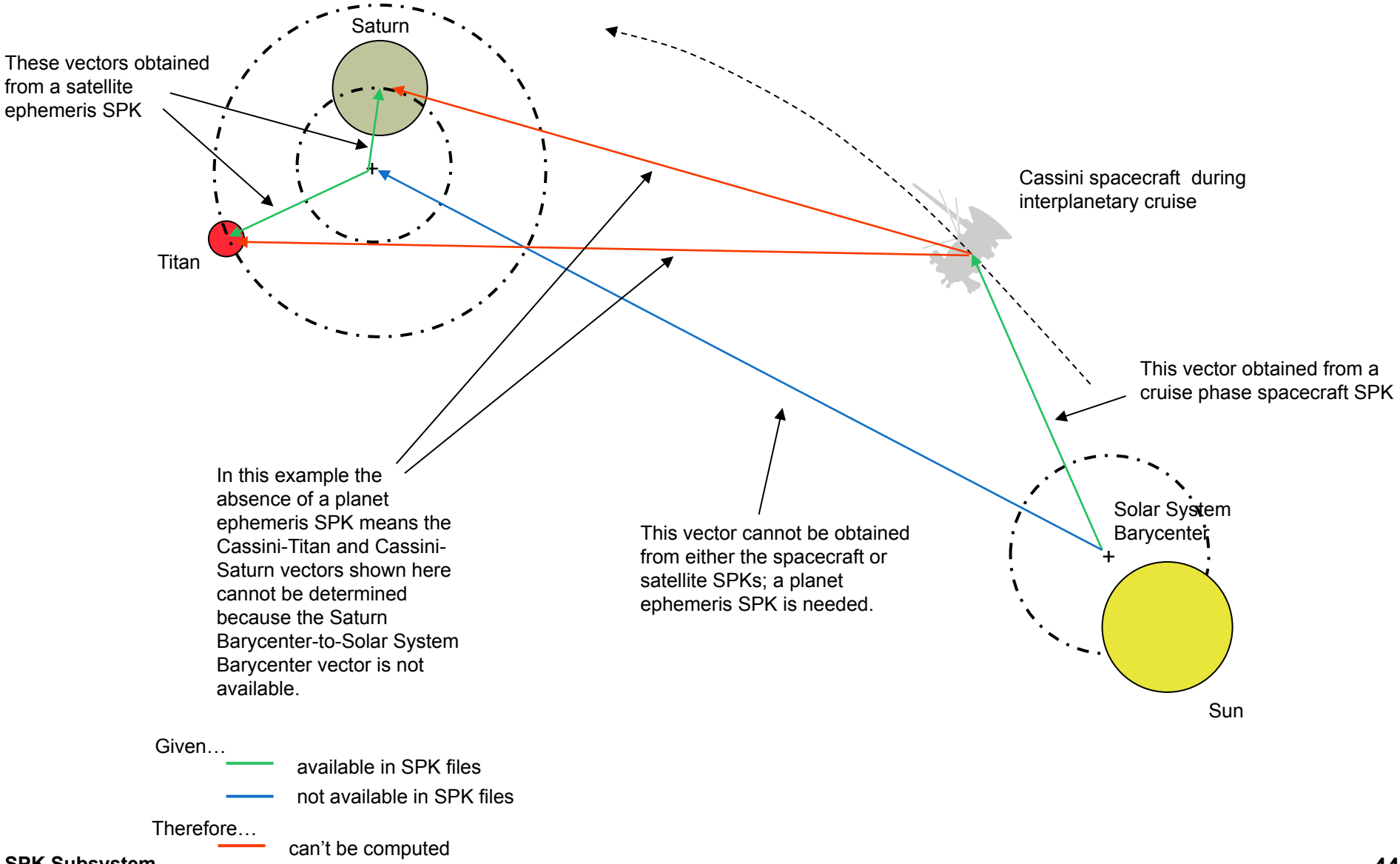
- **You have requested aberration-corrected states but the file, or files, you loaded do not contain sufficient data to relate both your target and observer bodies back to the solar system barycenter.**
  - **You may not have loaded all the files needed**
  - **You may have assumed the file contains data that it doesn't**
- **In the above situations you'll get an error message like the following:**

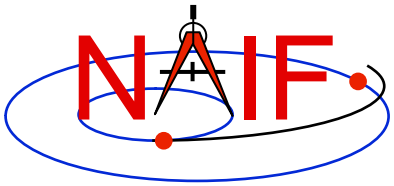
```
SPICE(SPKINSUFFDATA) - -
Insufficient ephemeris data has been loaded to
compute the state of xxx relative to yyy.
```

Problems Using SPK Files – 3a

- **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 link 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 relative to the solar system barycenter. 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 in BRIEF summaries if the -c command line option is used.**

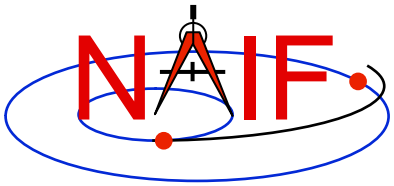
Problems Using SPK Files - 3b (drawing)





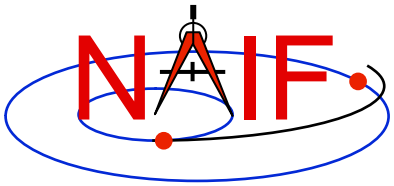
Problems Using SPK Files - 4

- **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.**



Problems Using SPK Files - 5

- **Segment Masking: You've loaded sufficient data to "connect" 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)**



Problems Using SPK Files - 6

- **Other missing data… not obvious.**
  - **You may need CK (and SCLK), FK or PCK data to construct a state (or position) vector in your requested output frame.**
- **Mistaking ET for UTC, or vice-versa.**
- **You must have loaded sufficient SPKs to be able to chain states to the solar system barycenter if doing aberration corrections.**
- **Using light time corrections requires target ephemeris data at the light time-corrected epoch.**
  - **If you're working near the beginning of an SPK, the light time-corrected epoch may occur earlier than available data.**

Problems Using SPK Files - 7

Navigation and Ancillary Information Facility

- **You've assumed that:**

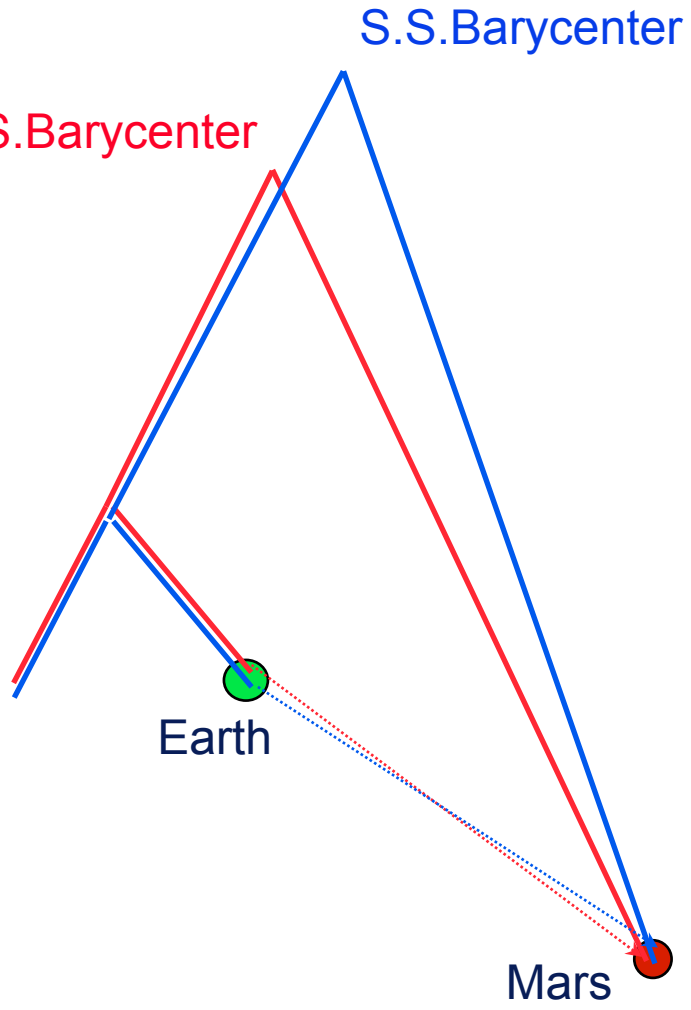
  state (observer, target) = - state (target, observer)

- **This is NOT true unless you have requested geometric states in both cases (i.e. no light time or stellar aberration corrections are applied)**

Comparing Apples and Oranges-1

Navigation and Ancillary Information Facility

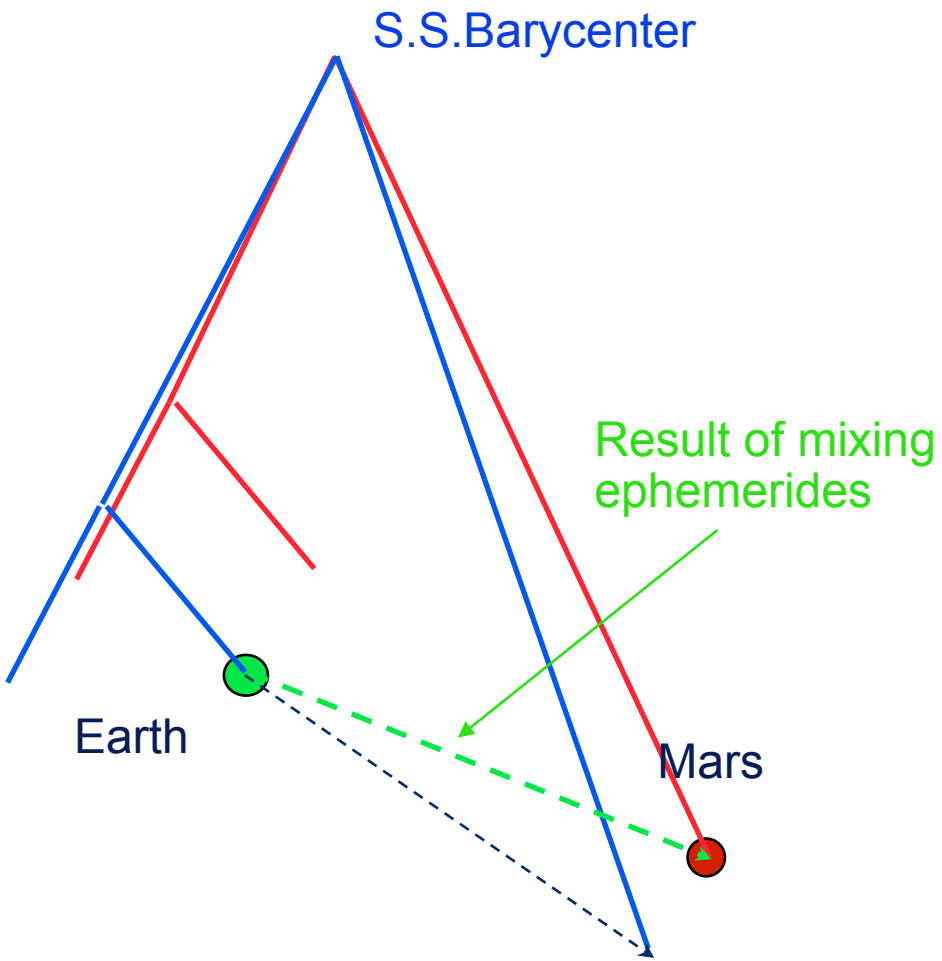
- With each new integration of the solar system, the solar system barycenter moves w.r.t. the planets
- Planet to planet offset variations are much smaller than the barycenter to planet variations

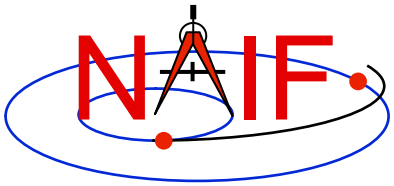


Comparing Apples and Oranges-2

Navigation and Ancillary Information Facility

- SPICE allows you to "load" different planetary ephemerides (or portions of them)
  - » Potentially can subtract the solar system barycenter-relative positions from different ephemerides to get relative states
- Don't mix planetary ephemerides
- For missions, a consistent set of ephemerides is provided





Barycenter Offset Magnitude

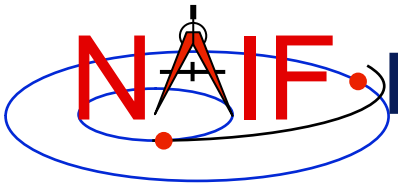
Navigation and Ancillary Information Facility

Body Mass Center	System Barycenter	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	~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)	2080	174%

* Estimated maximum values over the time range 2000-2050

Effect of Aberration Corrections - 1

- 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

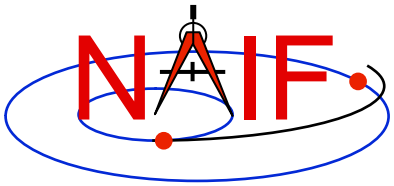


Effect of Aberration Corrections - 2

- **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**

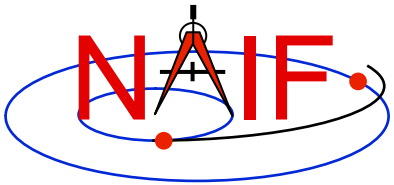
Retrieving State Vectors:
"Under the Hood" - 1

- **Example: find the geometric state of the MGS orbiter relative to Mars the at 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 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)**



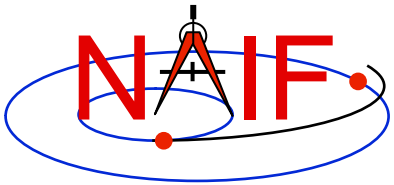
Retrieving State Vectors:
"Under the Hood" - 2

- **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)**



Retrieving State Vectors:
"Under the Hood" - 3

  - » 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



SPK File Structure

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