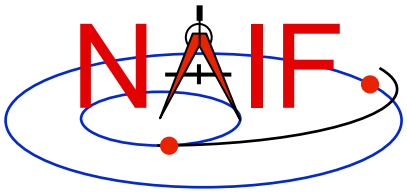


Navigation and Ancillary Information Facility

Making an SPK File

March 2010

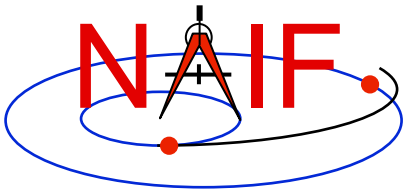
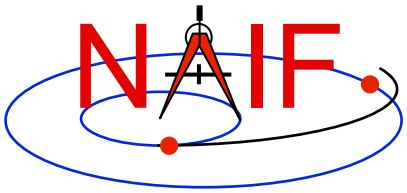


Table of Contents

Navigation and Ancillary Information Facility



Purpose

- **This tutorial provides guidance for producing (writing) an SPK file using software provided by NAIF:**
  - **the MKSPK application program**

  or
  - **SPK writer modules from the SPICELIB (FORTRAN) or CSPICE (C-language) library, or from the Icy (IDL) system**
    - » **Only partial implementation in Icy**
    - » **No SPK writers implemented in Mice**

Scope

- **This tutorial addresses production of SPK files**
  - **For general purposes**
  - **For use with NASA's Deep Space Network (the "SPS")**

- **(Note: This tutorial does *not* address SPK production by JPL navigation teams using the NIOSPK application, which was specially built to process JPL's NAVIO-format ephemeris/trajectory files.)**
  - **Those NAV teams may simply learn how to use the NIOSPK program and any useful SPK-related utilities.**

Background Assumptions

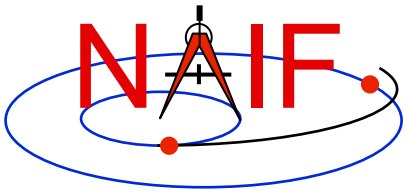
Navigation and Ancillary Information Facility

- It is assumed the reader has some familiarity with the SPICE system, and with basic ideas of orbital mechanics.
  - The SPICE Overview tutorial is available at:

    ftp://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/
- It is assumed the reader has read the "SPK Tutorial" that characterizes much of the SPK subsystem, but with emphasis on reading SPK files.
  - The SPK "reading" tutorial is available at:

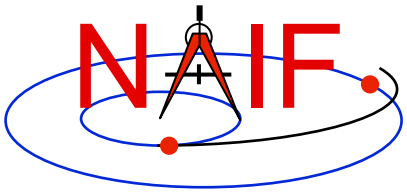
    ftp://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/ (named 19_spk)
- It is assumed the reader has available the SPK reference document entitled "SPK Required Reading," supplied with each copy of the SPICE Toolkit (.../doc/spk.req)
  - SPK Required Reading is also available at:

    http://naif.jpl.nasa.gov/naif/documentation.html



SPK References

- **References for SPK production**
  - **"Making an SPK" tutorial (this document)**
  - **"SPK (Ephemeris System)" tutorial (focused on reading an SPK)**
  - **"SPK Required Reading" (spk.req)**
  - **"MKSPK Users Guide" (mkspk.ug)**
  - **The source code "headers" provided as part of the SPK writer modules (subroutines)**
  - **"SPKMERGE User's Guide" (spkmerge.ug)**
  - **"SPY User's Guide" (spy.ug)**



Brief Overview - 1

- **Understand the physics of your data and how that relates to SPK type. For instance:**
  - **Type 5 implies an orbit well approximated by a sequence of one or more conic orbits.**
  - **Types 9 and 13 fit data regardless of the expected physics.**
    - » **Caution: a good fit in the mathematical realm may not respect the physics of the trajectory. For example, fitting polynomials to an excessively sparse set of states for a planetary orbiter could result in an interpolated path that intersects the planet.**

Brief Overview - 2

Navigation and Ancillary Information Facility

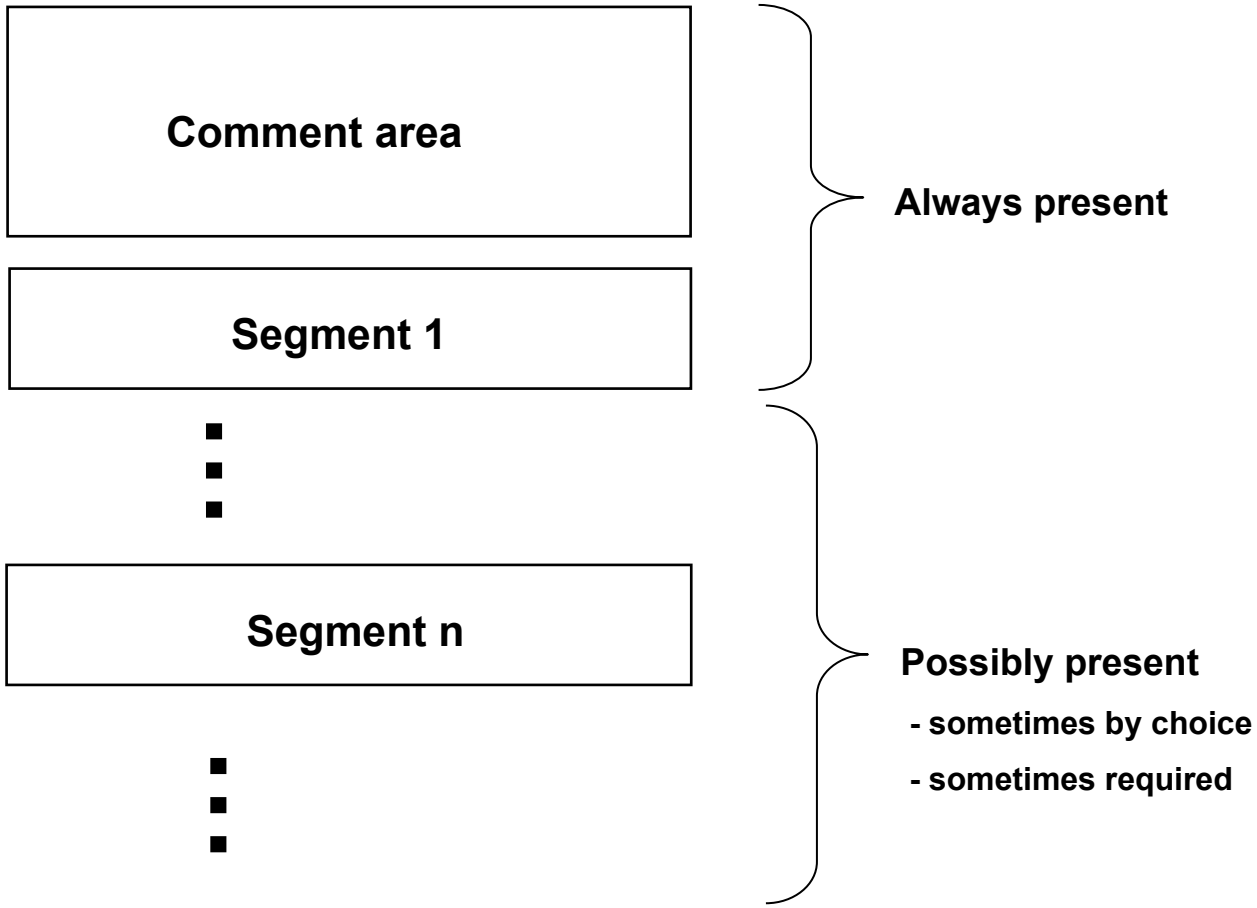
- **Ordinarily, use the NAIF MKSPK application to create SPKs from Cartesian state data or conic elements.**
  - **Depending on your source data, SPK types 5, 9, 10, and 13 will satisfy the requirements for most users.**
    - » **Type 5, yields compact SPK files when the trajectory is well approximated by piecewise two-body motion. May be the best choice for planetary or solar orbiters when available state data are sparse.**
    - » **Type 9, a good, general choice**
    - » **Type 13, when you have very accurate velocity data**
    - » **Type 10 applies ONLY to Two Line Element Sets (TLEs).**
- **Alternatively, use the Toolkit's SPK writing subroutines in your own production program.**
- **Caution: an SPK made for use by the NASA DSN has special requirements, discussed later on.**

Summary of SPK Architecture

SPK File Structure: The User's View

Navigation and Ancillary Information Facility

Logical Organization of an SPK File

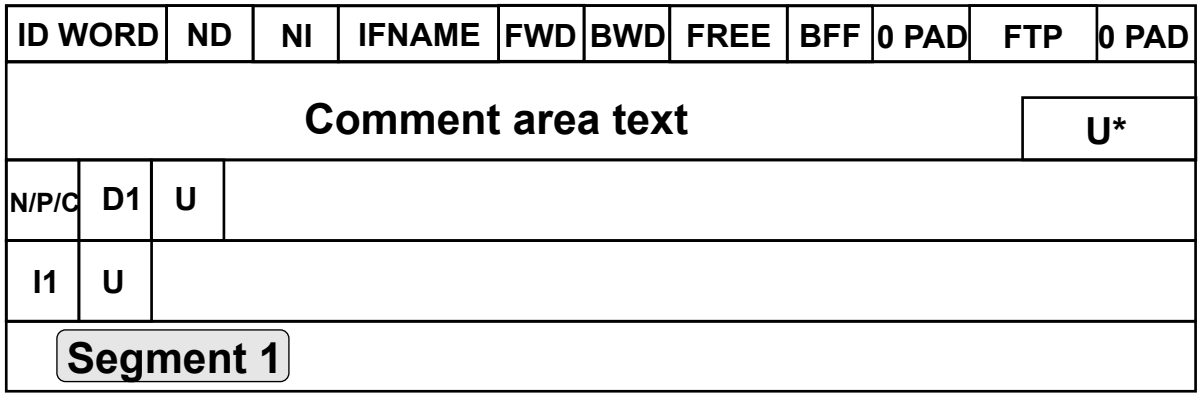


SPK File Structure - 1

Navigation and Ancillary Information Facility

A minimal SPK file, containing only one segment

Records are fixed-length: 1024 bytes



File record: One record.

Comment area: Present only if used. If used, one or more records.

Descriptor record: Contains 1 to 25 segment descriptors. One record.

Segment ID record: Contains 1 to 25 segment IDs. One record.

Data segment: One or more records.

ID WORD: Indicates file architecture and type

ND, NI: Number of d.p. and integer descriptor components

IFNAME: Internal file name

FWD, BWD: Forward and backward linked list pointers

FREE: First free DAF address

BFF: Binary file format ID

FTP: FTP corruption test string

N/P/C: Next, previous record pointers and descriptor count

Dn: Descriptor for segment n

In: Segment ID for segment n

U: Unused space

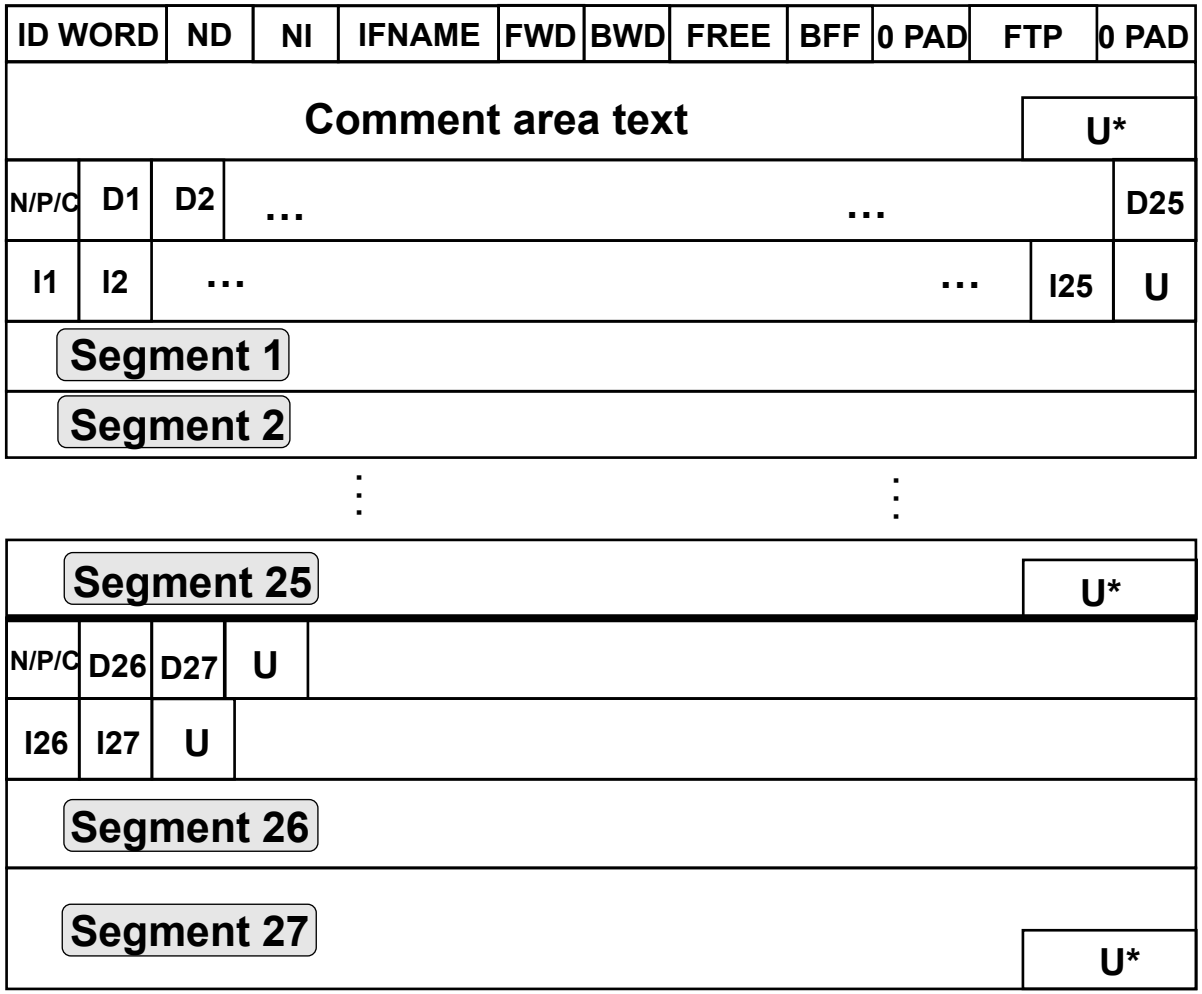
U*: Possibly unused space

SPK File Structure - 2

Navigation and Ancillary Information Facility

An SPK file containing 27 segments

Records are fixed-length: 1024 bytes



File record: One record

Comment area: Always present but could be empty. One or more records.

Descriptor record: Contains 1 to 25 segment descriptors. One record.

Segment ID record: Contains 1 to 25 segment IDs. One or more records.

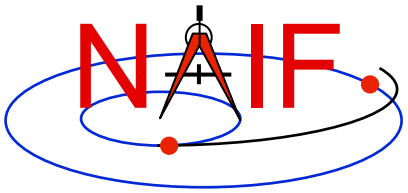
Data segments: One or more records per segment. (Up to 25 segments.)

Descriptor record: Contains 1 to 25 segment descriptors. One record.

Segment ID record: Contains 1 to 25 segment IDs. One or more records.

Data segments: One or more records per segment. (Up to 25 segments.)

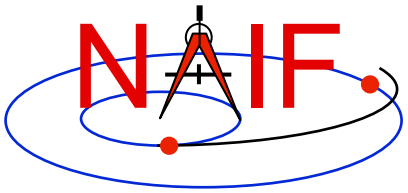
Diagram not to scale



SPK File Structure - Description

Navigation and Ancillary Information Facility

- **File record**
  - Contents
    - » Internal file name (set by file creator)
    - » Architecture and binary file format identifiers
    - » File structure parameters
    - » FTP transmission corruption detection string
  - Used by SPK reader and writer subroutines
- **Comment Area**
  - A place where "metadata" (data about data) may be placed to help a user of the SPK file understand the circumstances of its production and any recommendations about for what uses it was intended
- **Descriptor record and Segment ID record**
  - One of each of these is needed for every collection of 1-to-25 segments
- **Segment[s]**
  - Collection[s] of ephemeris data
    - » Minimum of one segment
    - » Maximum:
      - The practical maximum is a few thousand segments
      - Serious performance degradation occurs above 30000 segments for a single body
      - Absolute limits are imposed by the range of the INTEGER data type for your computer
  - Numerous SPK types may be used within an SPK file, but only one SPK type may appear within a given segment
  - Segments of different types may be intermixed within a given SPK file



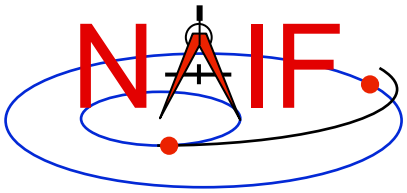
What is an SPK Segment?

Navigation and Ancillary Information Facility

- **A segment is a collection of information:**
  - » **providing ephemeris (position and velocity) of a single object**
  - » **given relative to a single center of motion**
  - » **specified in a single reference frame known to SPICE**
    - **Either built-in ("hard coded") or defined in a loaded frames kernel (FK)**
  - » **covering a specified, continuous period of time, and**
  - » **using a single SPK data type.**
  - **– Example: ephemeris for the Voyager 2 spacecraft, relative to the center of the Neptunian system (Neptune's barycenter), given in the J2000 inertial reference frame, covering a specific period of time, and using the Hermite interpolation with variable length intervals SPK type (type 13)**

- **An SPK segment must contain enough data to yield an object's state at any epoch within the time bounds associated with the segment**
  - **– This has implications that depend on the SPK type being produced**

Discussion applicable to all SPK production methods



The SPK Family

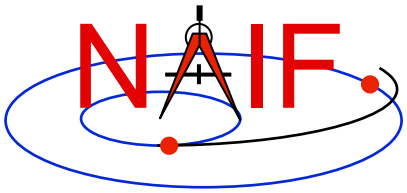
Navigation and Ancillary Information Facility

Type	Description	Notes
1	Modified divided difference arrays	Unique form produced at JPL; not likely to be useful to others.
2	Chebyshev polynomials for position; fixed length time intervals.	Velocity is obtained by differentiation. Used at JPL for planets. Evaluates quickly.
3	Chebyshev polynomials for position and velocity; fixed length time intervals	Separate polynomial sets for position and velocity. Used at JPL for natural satellites.
4	Special form used only by Hubble Space Telescope	Not available for general use.
5	Discreet states using weighted two-body propagation	Ok if motion very closely approximates two-body motion.
6	Special form of trigonometric expansion of elements for Phobos and Deimos	Not available for general use.
7	Precessing elements	Not available for general use.
8	Lagrange interpolation of position and velocity; fixed length intervals between states	Use Type 9 unless state spacing is truly uniform when measured in the TDB system.
9	Lagrange interpolation of position and velocity; variable length intervals between states	Versatile type; easy to use with MKSPK.
10	Weighted two-line element sets (Space Command)	Handles both “near-earth” and “deep space” versions.
11	Not used	
12	Hermite interpolation; fixed length intervals between states	Use Type 13 unless state spacing is truly uniform when measured in the TDB system.
13	Hermite interpolation; variable length intervals between states	Versatile type; easy to use with MKSPK. Use for DSN support.
14	Chebyshev polynomials for position and velocity, variable length time intervals	The most flexible of the Chebyshev types.
15	Precessing conic elements propagator	
16	Special form used by ESA’s Infrared Space Observatory	Not available for general use.
17	Equinoctial elements	Used for some satellites.
18	Emulation of ESOC’s “DDID” format	Used for SMART-1, MEX, VEX, and Rosetta

Recommended SPK Data Types - 1

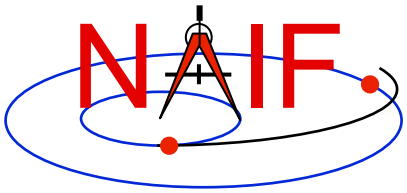
Navigation and Ancillary Information Facility

- SPK type 2 (Chebyshev polynomials for position, velocity given by differentiation) Used at JPL for planetary ephemerides.
- SPK type 3 (Separate Chebyshev polynomials for position and velocity) Used at JPL for satellite ephemerides.
- SPK type 5 (Weighted two-body extrapolation) Often used for comets and asteroids, as well as for sparse data sets where a two-body approximation is acceptable.
- SPK types 9 and 13 (Sliding-window Lagrange and Hermite interpolation of unequally-spaced states) Often used by non-JPL ephemeris producers and by users of NAIF's "Make SPK" (MKSPK) application.
- SPK type 10 (weighted Space Command two-line element extrapolation) Often used for earth orbiters.
- SPK type 14 (Separate Chebyshev polynomials for position and velocity, with variable time steps) This is the most flexible Chebyshev data type.
- SPK type 15 (Precessing conic elements) Provides a very compact ephemeris representation; limited to orbits where this type of approximation is valid.
- SPK type 17 (Equinoctial elements) Most suited for representation of ephemerides of natural satellites in equatorial or near-equatorial orbits.



Recommended SPK Data Types - 2

- **Each type has certain properties that may promote or limit its usefulness in a particular application. These properties include, but are not limited to the following.**
  - » **Ability to model the actual ephemeris to be represented with the accuracy required for your application.**
  - » **Consistency between velocity and derivative of position.**
  - » **Evaluation speed (performance).**
  - » **Compactness (file size).**
  - » **Availability of SPICE software needed to write files of that type.**
- **Users are encouraged to consult with NAIF about the suitability of an SPK type for a particular purpose.**



Creating Multiple SPK Segments

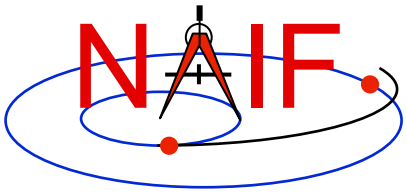
Navigation and Ancillary Information Facility

- Each SPK segment must have a single object, center of motion, reference frame and SPK data type.
- Limiting segment size to 10,000 states or "packets of ephemeris data" can improve performance when searching within a segment.
  - Absolute limits on segment size depend on the size of the INTEGER data type.
- For good SPK reading performance, the total number of segments for any given body in a file should be kept under the dimension of the SPKBSR segment buffer, currently set to 30,000.
  - More details about reading efficiency are provided at the end of this tutorial.
  - When reading data from multiple SPK files, a more stringent limit applies: the total number of loaded segments for any body, possibly contributed by multiple files, should be less than the SPKBSR segment buffer size.
  - For best efficiency, the total number of segments loaded should be less than this buffer size.
- One may elect to initiate a new segment (or more) as the means for modeling a propulsive maneuver.
  - This is because the SPK reader modules will NOT allow interpolation over a segment boundary.
- When starting a new segment you may use a new segment identifier, for instance to indicate a new trajectory leg after a maneuver.
  - Can only be done if using SPK write modules–not if using the MKSPK application.

Choosing Polynomial Degree

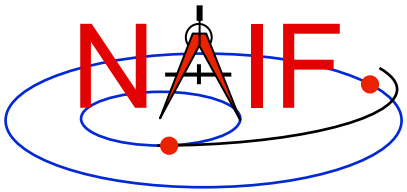
Navigation and Ancillary Information Facility

- **If you make a type 8 or 9 (Lagrange interpolation) or a type 12 or 13 (Hermite interpolation) SPK file you must specify the degree of the interpolating polynomial that the SPK reader subroutine will use.**
  - **This choice needs some consideration about desired accuracy, file size and evaluation speed (performance).**
  - **This choice is also affected by the "smoothness" of the orbit data you wish to represent with an SPK file.**
  - **The allowed range of degree is 1-to-15. In addition, to ensure position and velocity continuity over the time span covered by the orbit data:**
    - » **for types 8 and 9, the polynomial degree must be odd.**
    - » **for types 12 and 13, the polynomial degree must be 3-mod-4, meaning degree 3, 7, 11 or 15.**

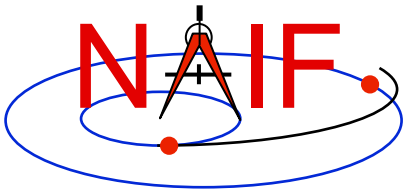


Reference Frame

- **No matter the SPK Type, all ephemeris data must be provided in a well identified reference frame.**
- **Any reference frame known to the SPICE system, whether built-in (hard coded) or defined at run time through a Frames Kernel (FK), may be used for ephemeris data placed in an SPK file.**
- **Some examples of typical reference frames used:**
  - **Inertial (non-rotating):**
    - **Earth Mean Equator and Equinox of J2000 (EME2000, a.k.a. J2000)**
    - **Ecliptic of J2000**
  - **Body fixed (non-inertial)**
    - **ITRF93 (for Earth)**
    - **MOON_ME (MOON_MeanEarth)**



SPK Production Methods

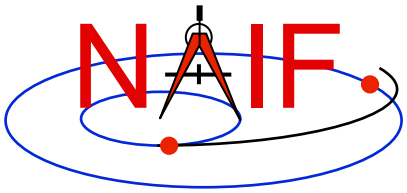


Choices for Making an SPK File

- **There are two methods available for making an SPK file.**
  1. **Take a data file produced by your own trajectory propagator program and input this into the conversion utility (MKSPK) provided by NAIF that outputs an SPK file.**
  2. **Incorporate the appropriate SPK writer modules (subroutines) into your own code.**
     - » **Add these routines to your trajectory estimator/ propagator.**

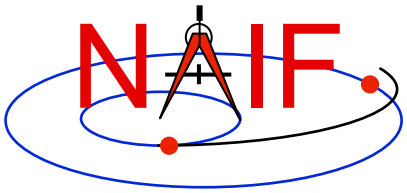
     or...

     - » **Write your own "post-processor" conversion utility, similar to MKSPK described above.**
- **Both methods are described in the next few pages.**



Making Your Choice - 1

- **Using the MKSPK program provided in the Toolkit could be easiest for "simple" situations.**
  - **Provides considerable flexibility for accepting a wide assortment of input data formats.**
  - **Does allow one to make multi-segment SPK files when the target, center of motion, reference frame, or SPK type changes, but not as straight forward as it could/should be.**
    - » **Best done through multiple program executions (although one could be tricky and accomplish this in a single execution).**
    - » **A future version of MKSPK may better accommodate this.**
    - » **Note: production of multiple segments in type 5, 8, 9, 12 and 13 files when the amount of input data requires so, is automatically handled.**



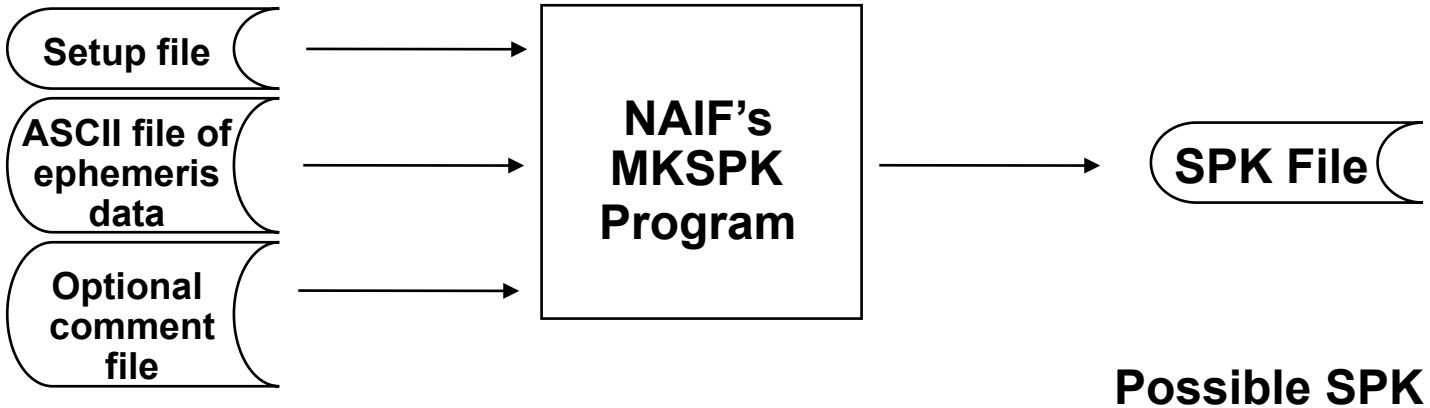
Making Your Choice - 2

- **Using the SPK "writer" modules found in SPICELIB, CSPICE, and Icy offers the greatest flexibility and user control.**
  - **Using these requires that you write your own program.**
  - **You'll likely need to use some additional SPICE modules as well.**

Using NAIF's MKSPK Application Program

Using the MKSPK Utility - 1

Navigation and Ancillary Information Facility



Suitable kinds of input ephemeris data are:

- Table of Cartesian state vectors
- Table of conic elements
- One or more sets of equinoctial elements
- One or more sets of Space Command two-line elements

Possible SPK data types produced are:

- Type 05
- Type 08
- Type 09
- Type 10
- Type 12
- Type 13
- Type 15
- Type 17

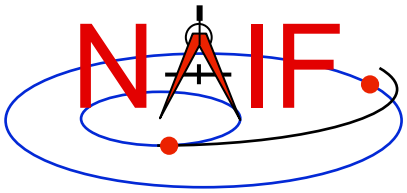
Using the MKSPK Utility - 2

Navigation and Ancillary Information Facility

This table indicates which SPK types can be made from the four kinds of input data accepted by MKSPK

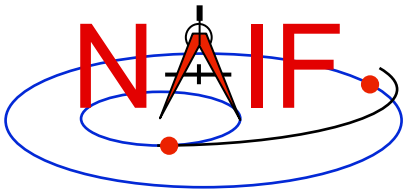
SPK Type Produced by MKSPK -->	5	8	9	10	12	13	15	17
Input Data Type								
Cartesian state vectors	Y	Y	Y	N	Y	Y	Y	Y
Conic elements	Y	Y	Y	N	Y	Y	Y	Y
Equinoctial elements	N	N	N	N	N	N	N	Y
Space Command Two-line elements	N	N	N	Y	N	N	N	N

Y = Yes N = No



Using the MKSPK Utility - 3

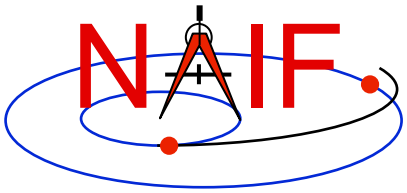
- **MKSPK will produce a file consisting of one or more segments as needed.**
  - **It will write up to 10,000 data points in one segment.**
  - **For multi-segment files based on types 5, 8, 9, 12 and 13 the program will repeat sufficient data points at both sides of each interior segment boundary to ensure the SPK file will provide a continuous ephemeris through the segment boundary epoch.**
- **You can use MKSPK to add a new segment to an existing SPK file.**
- **You can use SPKMERGE to merge two or more SPK files made from separate executions of MKSPK.**
  - **It's important to fully understand how SPKMERGE works if you do this.**



Using the MKSPK Utility - 4

- **MKSPK does not provide direct/specific means for including propulsive maneuvers within an SPK file.**
  - **Instead, use either of these two methods.**
    - » **Append a new SPK segment to an existing SPK file, using MKSPK.**
    - » **Merge a collection of SPK files, using SPKMERGE.**

Using SPK “Writer” Modules



Using SPK Writer Routines

Navigation and Ancillary Information Facility

- The next several charts outline how to use the“SPK writer” modules available in the Toolkit libraries.
  - SPICELIB (FORTRAN)
  - CSPICE (C)
    - » All types supported except Type 1
  - Icy (IDL)
    - » All types supported except Type 1, 15, 17, 18
  - Mice (MATLAB)
    - » Currently no SPK writer modules are supported

- These routines could be embedded in your existing trajectory propagator program, or they could be used to build a separate conversion program analogous to MKSPK.

What Routines To Use - 1

For all except SPK type 14

SPKOPN	Open a new SPK file. (Use SPKOPA to append to existing file.)
SPKWxx	Write a segment of SPK type xx
.	
.	
[SPKWxx]	[Write more segments]
.	[Repeat as needed]
.	
SPKCLS	Close the file

[…] indicates possible multiple occurrences

These routine names are for the FORTRAN (SPICELIB) Toolkit. For CSPICE the names are the same but are in lower case and have an "_c" appended. For Icy, module names are case-insensitive and have "cspice_" prepended.

What Routines To Use - 2

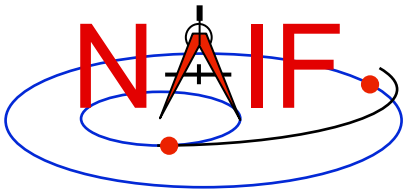
Navigation and Ancillary Information Facility

For SPK type 14

SPKOPN, SPKOPA	Open file to add data
SPK14B	Begin a new segment
SPK14A	Add data to segment
[SPK14A]	Add more data
SPK14E	End the segment
SPK14B	Begin a new segment
SPK14A	Add data to segment
[SPK14A]	Add more data
SPK14E	End the segment
etc.	
SPKCLS	Close the file

Repeat as needed

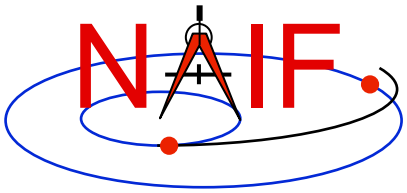
[…] indicates possible multiple occurrences



Close the SPK File

- **Once you have completed the addition of all data to your SPK file, be sure to call the SPKCLS routine to close the file.**
  - **Failure to properly close an SPK file will result in a problem file having been produced.**
- **This point is emphasized here because it has been a frequent problem.**

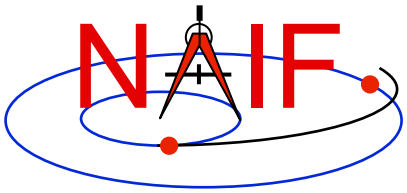
Finishing Up



Not Quite Done Yet

- **You've now used either MKSPK or the appropriate SPK writer routines to produce an SPK file. To complete the job you should consider the following.**
  - **Add comments (metadata) to the comment area of the SPK file.**
    - » **This could have been done during execution of MKSPK.**
    - » **It can be done after the SPK has been created by using the Toolkit's "commnt" utility program.**
  - **Validate the file before sending it off to your customer.**
  - **Consider if there is a need to merge this newly made SPK file with others.**

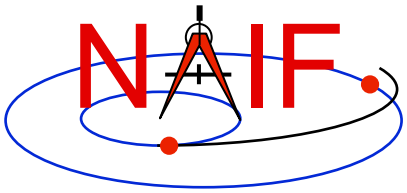
- **See the next several charts for more information on these subjects.**



Add Comments (metadata)

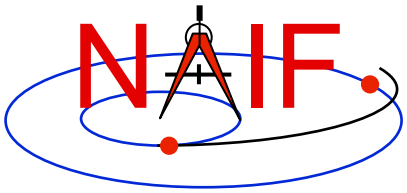
Navigation and Ancillary Information Facility

- **It is recommended (but not a technical requirement) that the producer of an SPK file add to the file, in the "comment area," appropriate descriptive information.**
  - **When, how and by whom the file was created.**
  - **Intended use for the file.**
  - **Cautions or restrictions on how the file is to be used.**
- **The comments might also include some of these topics.**
  - **Time coverage.**
  - **Ephemeris objects included.**
  - **Type(s) of data used (in the sense of reconstructed versus predicted).**
  - **Any available estimates of accuracy.**
  - **Sources of the data used to produce this SPK file.**
  - **Name(s) of previously generated SPK file(s) being replaced by this file.**
  - **Any knowledge of plans for future updates to (replacements for) this file.**
  - **Name and version number of your SPK production program.**
  - **Type of platform (hardware/OS/compiler) on which the SPK file was generated.**



How to Add Comments to an SPK

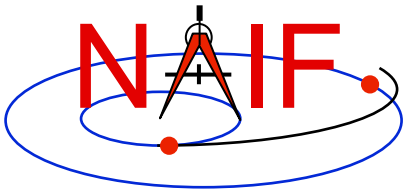
- **Several means are available for adding comments (metadata) to an SPK file.**
  - **An option in the MKSPK program allows comments supplied in a separate text file to be added to the comment area during MKSPK execution.**
  - **Use the "COMMNT" utility program from the SPICE Toolkit.**
    - **This may be run as an interactive program or in command line mode within a script.**
  - **If using FORTRAN, C or IDL you can use APIs.**
    - **Not currently supported in MATLAB.**



Validate the SPK File

Navigation and Ancillary Information Facility

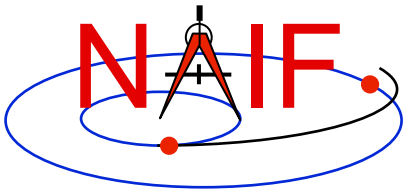
- **Validation of SPK files is recommended**
  - **Caution is needed more for one-of-a-kind files than for those generated in a previously tested, unchanging process.**
  - **Some SPICE utility programs might help with this validation.**
    - » **SPY: can do a variety of structure and semantic checks.**
      - **SPY is available from the Utilities link of the NAIF website.**
    - » **SPKDIFF: used to statistically compare two supposedly similar SPK files.**
      - **SPKDIFF is available in each Toolkit package and also from the Utilities link of the NAIF website.**
  - **Consider writing your own validation program.**
  - **Caution: successfully running an SPK summary program (e.g. BRIEF or SPACIT) or successfully running the format conversion program (TOXFR or SPACIT) is a positive sign, but is not a sufficient test.**



Validate the Overall Process

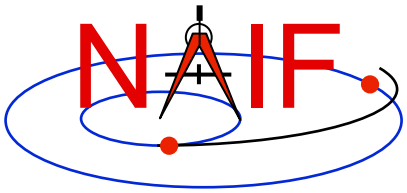
- **When you first start producing SPK files, or when changing the SPK "type" used or the kind of mission (trajectory) to be represented, validation (or revalidation) of the overall process is advised.**
  - **Validation of not only SPKs, but of end products derived from SPKs, is advised.**

- **Consider writing a program that compares states from your source data with states extracted from your new SPK file.**
  - **Do this using interpolated states from your source data–not only the states placed in the SPK file.**
  - **Verify a uniformly good fit on the whole time interval covered by the file.**



Make a Merged SPK File ?

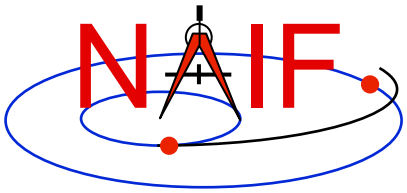
- **Sometimes it is helpful to customers if portions of two or more SPK files are merged into just one.**
  - **(Sometimes the opposite is true, so be careful!)**
- **If making a merged product is appropriate, use the SPICE utility SPKMERGE.**
  - **Read the SPKMERGE User's Guide.**
    - » **Be especially aware of how SPKMERGE directives affect the precedence order of the data being merged. (This is different from the precedence order that applies when one reads an SPK file or files.)**
  - **Carefully examine your results (probably using either BRIEF or SPACIT) to help verify you got what you expected.**
- **If you've made a merged SPK file, check to see that the included comments are appropriate.**



Get Help

Navigation and Ancillary Information Facility

- **If your project provides funding to NAIF for help with development:**
  - **Ask JPL's NAIF team for assistance with:**
    - » **picking the SPK type to be used**
    - » **picking the method for producing SPK files**
    - » **designing tests to validate the process**
  - **Ask NAIF for samples of SPK files from other missions that could help you check your process.**

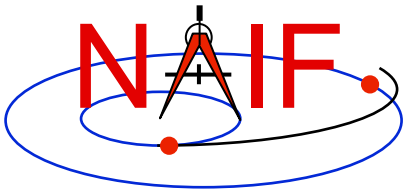


Allowed SPK Types and Their Restrictions for Interfaces with the Service Preparation System (SPS) of NASA's Deep Space Network

DSN Interface Overview

Navigation and Ancillary Information Facility

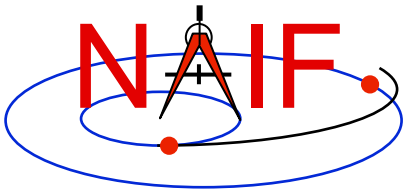
- **SPKs prepared for use in the DSN/SPS may be used in one or more of five software sets:**
  - **Metric Predicts Generator (MPG)**
    - » **Used for view period generation, DSN scheduling and DSN metric predicts (antenna pointing and tuning of the transmitters and receivers)**
  - **Telecomm Preditcs (UTP/TFP)**
    - » **Subsystem for prediction and analysis of telecommunications signal levels**
  - **Radiometric Modeling and Calibration Subsystem (RMC)**
    - » **Used to calibrate atmospheric effects on radio waves**
  - **Delta Differenced One-way Range (Delta-DOR) subsystem**
    - » **A special tracking data type providing additional precision to spacecraft navigation**
- **All SPKs delivered to the SPS must pass through a front-end validation program that has some restrictions.**
- **SPK files intended for use in any of these software sets may face some restrictions. See the next pages.**
  - **(Note: The restrictions that apply as of October 2008 are far less than before this date, due to full retirement of the SPS predecessor–the NSS.)**



SPS Validation Gate

Navigation and Ancillary Information Facility

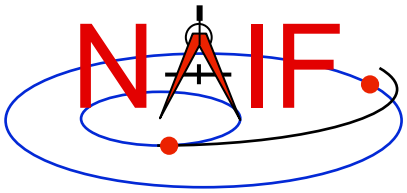
- **The SPS' front-end validation tool requires:**
  - **An SPK contain data for only one spacecraft**
    - » **The presence of non-spacecraft ephemeris data is ok**
  - **An SPK have no data gaps for the spacecraft**
  - **The spacecraft SPK must be of Type 1 or Type 13**



What SPK Type to Use for Interfaces with the DSN?

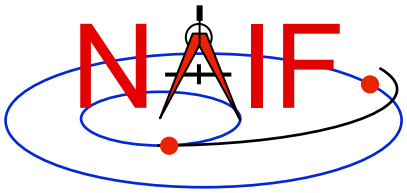
Navigation and Ancillary Information Facility

- **The Metric Predicts Generator does not inherently place any restrictions on the SPK files used.**
  - **However, current rules of the SPS nevertheless restrict the SPK choice to only Type 1 or Type 13. (Other types not yet fully, formally tested.)**
  - **This restriction may be lifted in the near future: contact a DSN representative for the latest news.**
  - **Note: Only JPL NAV teams are able to produce Type 1 SPKs.**
- **The telecommunications prediction software does not inherently place any restrictions on the SPK files used.**
- **The radiometric modeling and calibration software requires only Type 1 or Type 13 SPKs**
  - **This restriction will be lifted approximately January 2009: contact a DSN representative for the latest news.**
- **The delta-DOR software requires only Type 1 or Type 13 SPKs**
  - **This restriction is likely to be lifted by approximately November 2009, maybe even sooner.**



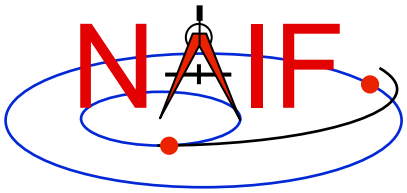
Issues Affecting SPK Reading Efficiency

The way you write an SPK file could substantially affect how quickly your customer's software will be able to read the file.



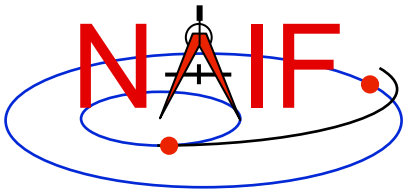
SPK Reading: Efficiency Issues - 1

- **SPK file creators should design files to support efficient read access.**
  - **This requires knowledge of how SPK file attributes impact efficiency.**
- **When you store "large" amounts (>10^7 states or data packets) of ephemeris data in one or more SPK files, SPK reading efficiency may be affected by:**
  - **SPK segment size**
  - **Number of segments for a body in one SPK file**
  - **Number of segments for a body contributed by multiple SPK files**
  - **The number of loaded segments for all bodies**
  - **The number of loaded files**



SPK Reading: Efficiency Issues - 2

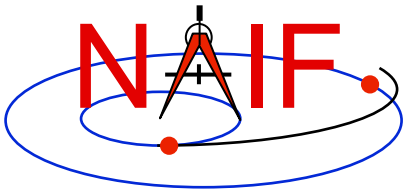
- **Segment size**
  - **When a segment contains more than 10,000 states or data packets, the SPK readers will generally take longer to search the segment for requested data.**
    - » **When the segment is larger than this size, more records are read to look up segment directory information. If these records are not buffered, more physical records are read from the SPK file.**
  - **There is a trade-off between segment size and numbers of segments and files.**
    - » **It can be preferable to have large segments rather than have "too many" segments or files. (Read on)**



SPK Reading: Efficiency Issues - 3

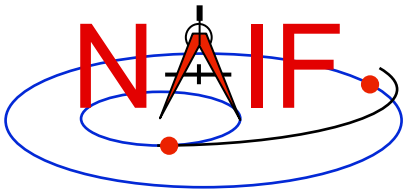
- **Number of segments for a body in one SPK file**
  - **An SPK file MUST not contain more segments for one body than can be "buffered" at one time.**
    - » **The SPK reading system buffers coverage descriptions ("segment descriptors") for segments it has examined to satisfy previous requests for state data.**
      - **Don't confuse descriptor buffering with data buffering.**
        - **The SPK reading system also buffers segment DATA, as opposed to segment descriptors, but this is not relevant to this discussion.**
    - » **One fixed-size buffer is used for all SPK segments.**
      - **The size of this buffer is given by the parameter "STSIZE," declared in the SPKBSR suite of routines.**
      - **STSIZE is currently set by NAIF to 30,000.**
        - **NAIF recommends that users NOT change this parameter, since maintenance problems may result.**
    - » **Unsurprisingly, the system works best when all needed segment descriptors are buffered simultaneously.**

continues



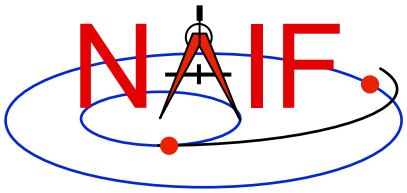
SPK Reading: Efficiency Issues - 4

- **Number of segments for a body in one SPK file, continued:**
  - » **The buffering scheme is "lazy": no descriptors are buffered for segments that haven't been examined.**
    - **But when an SPK file is searched for data for a specified body, descriptor data for ALL segments in the file for that body are buffered.**
  - » **The buffering algorithm can "make room" in the buffer by discarding unneeded, buffered descriptor data.**
    - **A "least cost" algorithm decides which buffered data to discard.**
  - » **When more buffer room is needed than can be found:**
    - **The SPK reading system reads data directly from SPK files without buffering descriptor information.**
    - **This is NOT an error case: the SPK system will continue to provide correct answers.**
    - **BUT: the system will run VERY SLOWLY.**
      - **This situation is analogous to "thrashing" in a virtual-memory operating system.**
      - **If buffer overflow occurs frequently, the SPK reading system may be too slow to be of practical use.**



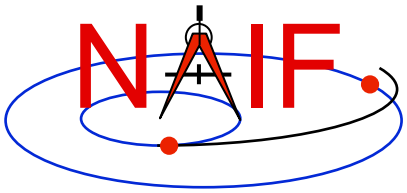
SPK Reading: Efficiency Issues - 5

- **Number of segments for a body contributed by multiple SPK files:**
  - **Buffer overflow can occur if too many segments for one body are contributed by multiple loaded SPK files.**
    - » **Overflow can take longer to occur than in the single-SPK case, due to lazy buffering: files that haven't been searched don't consume buffer space.**
      - **Thus an impending overflow problem may not be detected early in a program run.**
  - **User applications can avoid buffer overflow if data are appropriately spread across multiple SPK files.**
    - » **Applications can avoid buffer overflow by:**
      - **loading only those files of immediate interest**
      - **unloading files once they're no longer needed**



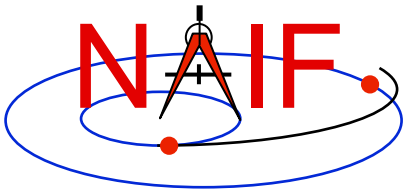
SPK Reading: Efficiency Issues - 6

- **Number of segments for all bodies, contributed by all loaded SPK files:**
  - **Buffer overflow does not result from loading SPK files contributing more than STSIZE segments for different bodies.**
  - **However, if the total number of loaded segments for bodies of interest exceeds STSIZE, thrashing can occur as descriptor data are repeatedly discarded from the buffer and then re-read.**
    - » **Loaded segments for bodies for which data are not requested do not contribute to the problem.**
  - **For best efficiency, load only files contributing fewer than a total of STSIZE segments for all bodies of interest.**
    - » **When more than STSIZE segments are needed, applications should process data in batches: unload files containing unneeded data in order to make room for new files.**



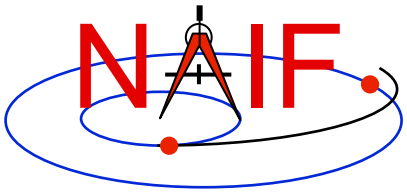
SPK Reading: Efficiency Issues - 7

- **Number of loaded SPK files:**
  - Up to 1000 SPK files may be loaded at one time by an application.
    - » The "1000" limit applies to DAF-based files, so loaded C-kernels and binary PCK kernels count against this limit.
  - But loading large numbers of SPK files hurts efficiency:
    - » Since operating systems usually allow a process to open much fewer than 1000 files, the SPK system must open and close files via the host system's file I/O API in order to provide a "virtual" view of 1000 open files.
      - The more such file I/O, the slower an application runs.
    - » Loading a large number of SPK files could result in a buffering problem if too many segments are loaded for bodies of interest.



SPK Reading: Efficiency Issues - 8

- **Recommendations**
  - **Limit segment counts to avoid buffer overflow and thrashing**
    - » **Never have more than STSIZE segments for one body in an SPK file and never have more than STSIZE segments for one body loaded simultaneously.**
    - » **Don't require users to have more than STSIZE segments loaded at one time.**
    - » **If necessary, use larger segments to enable smaller segment counts.**
  - **Consider distributing SPK data across multiple files:**
    - » **so as to make selective SPK loading convenient**
      - **facilitate processing data in batches**
    - » **so that loading very large numbers of SPK files at once is unnecessary**



SPK Reading: Further Usability Issues

- **We've discussed reading efficiency in terms of application execution speed; other usability concerns include:**
  - **ease with which files can be transferred between systems**
  - **simplicity of file management required of user applications**
  - **ease with which files of interest can be identified by users, both for current use and in an archival setting**