

---

# **SpiceyPy Documentation**

*Release 2.1.0*

**Andrew Annex**

**Nov 09, 2017**



---

## Contents

---

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Common Issues</b>	<b>5</b>
2.1	SSL Alert Handshake Issue . . . . .	5
<b>3</b>	<b>How to install from source (for bleeding edge updates)</b>	<b>7</b>
<b>4</b>	<b>Cassini Position Example</b>	<b>9</b>
<b>5</b>	<b>Lessons</b>	<b>13</b>
5.1	Basics of SpiceyPy . . . . .	13
<b>6</b>	<b>SpiceyPy package</b>	<b>15</b>
6.1	spiceypy.spiceypy module . . . . .	15
6.2	spiceypy.utils.support_types module . . . . .	157
6.3	spiceypy.utils.libspice module . . . . .	163
<b>7</b>	<b>Indices and tables</b>	<b>165</b>
	<b>Python Module Index</b>	<b>167</b>



This is the documentation for SpiceyPy. The documentation for each function in the wrapper is in large part copied from the “Abstract” and “Brief\_I/O” sections of the corresponding CSPICE function documentation. Each wrapper function has a link back to the corresponding original CSPICE function documentation hosted at the NAIF website. For more in-depth information about SPICE, please visit the NAIF website or [click here](#) to view the entire CSPICE documentation.

The intent of the function doc-strings is to serve only as a quick reference to what the parameter’s expected types are for the purpose of getting started with the wrapper. As each function has a link to the CSPICE documentation for that function, more detailed explanations are deferred to the NAIF via those links.

Contents:



# CHAPTER 1

---

## Installation

---

SpiceyPy is currently supported on Mac, Linux, and Windows systems. If you are new to python, it is a good idea to read a bit about it first [docs.python-guide.org](https://docs.python-guide.org). For new installations of python, it is encouraged to install and or update: pip, setuptools, wheel, numpy, six, certifi:

```
pip install -U pip setuptools wheel
pip install -U numpy six certifi
```

Then to install SpiceyPy, simply run:

```
pip install spiceypy
```

If you use anaconda/miniconda/conda run:

```
conda install -c https://conda.anaconda.org/andrewannex spiceypy
```

If no error was returned you have successfully installed SpiceyPy. To verify this you can list the installed packages via this pip command:

```
pip list
```

You should see spiceypy in the output of this command. Or you can start a python interpreter and try importing SpiceyPy like so:

```
import spiceypy

# print out the toolkit version installed
print(spiceypy.tkvrns('TOOLKIT'))
```

This should print out the toolkit version without any errors. You have now verified that SpiceyPy is installed.





### 2.1 SSL Alert Handshake Issue

In early 2017, JPL updated to a TLS1.2 certificate and enforced https connections causing installation issues for users, in particular for macOS users, with OpenSSL versions older than 1.0.1g. This is because older versions of OpenSSL still distributed in some environments which are incompatible with TLS1.2. As of late 2017 SpiceyPy has been updated with a strategy that can mitigate this issue on some systems, but it may not be totally reliable due to known deficiencies in setuptools and pip.

Another solution is to configure a new python installation that is linked against a newer version of OpenSSL, the easiest way to do this is to install python using homebrew, once this is done spiceypy can be installed to this new installation of python (IMHO this is the best option).

If your python 3.6 distribution was installed from the packages available at python.org an included command `Install Certificates.command` should be run before attempting to install SpiceyPy again. That command installs the certifi package that can also be install using pip.

Alternatively, installing an anaconda or miniconda python distribution and installing SpiceyPy using the conda command above is another possible work around.

Users continuing to have issues should report an issue to the github repository.

Supporting links:

<https://bugs.python.org/issue29065> <https://github.com/requests/requests/issues/2022> <https://pyfound.blogspot.com/2017/01/time-to-upgrade-your-python-tls-v12.html> <https://www.python.org/dev/peps/pep-0518> <https://github.com/AndrewAnnex/SpiceyPy/pull/202>



---

## How to install from source (for bleeding edge updates)

---

**Attention:** If you have used the pip or conda install commands above you do not need to do any of the following commands. Installing from source is intended for advanced users. Users on machines running Windows should take note that attempting to install from source will require software such as visual studio and additional environment configuration. Given the complexity of this Windows users are highly encouraged to stick with the releases made available through PyPi/Anaconda Cloud.

If you wish to install from source, first simply clone the repository by running the following in your favorite shell:

```
git clone git@github.com:AndrewAnnex/SpiceyPy.git
```

If you do not have git, you can also directly download the source code from the GitHub repo for SpiceyPy at <https://github.com/AndrewAnnex/SpiceyPy>

To install the library, simply change into the root directory of the project and then run:

```
python setup.py install
```

The installation script will download the appropriate version of the SPICE toolkit for your system, and will build a shared library from the included static library files. Then the installation script will install SpiceyPy along with the generated shared library into your site-packages directory.



---

## Cassini Position Example

---

Below is an example that uses spiceypy to plot the position of the Cassini spacecraft relative to the barycenter of Saturn.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

First import spiceypy and test it out.

```
import spiceypy as spice
```

```
# Print out the toolkit version
spice.tkvrns("TOOLKIT")
```

```
'CSPICE_N0066'
```

We will need to load some kernels. You will need to download the following kernels from the NAIF servers via the links provided. After the kernels have been downloaded to a common directory write a metakernel containing the file names for each downloaded kernel (provided after the links). I named the metakernel 'cassMetaK.txt' for this example. For more on defining meta kernels in spice, please consult the [Kernel Required Reading](#).

- naif0009.tls
- cas00084.tsc
- cpck05Mar2004.tpc
- cas\_v37.tf
- 04135\_04171pc\_psv2.bc
- 030201AP\_SK\_SM546\_T45.bsp
- cas\_iss\_v09.ti
- 020514\_SE\_SAT105.bsp

- 981005\_PLTEPH-DE405S.bsp

```
# The meta kernel file contains entries pointing to the following SPICE kernels,
↳ which the user needs to download.
# https://naif.jpl.nasa.gov/pub/naif/generic_kernels/lsk/a_old_versions/naif0009.tls
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/sclk/cas00084.tsc
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/pck/cpck05Mar2004.tpc
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/fk/release.11/cas_v37.tf
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/ck/04135_04171pc_psiv2.bc
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/spk/030201AP_SK_SM546_T45.bsp
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/ik/release.11/cas_iss_v09.ti
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/spk/020514_SE_SAT105.bsp
# https://naif.jpl.nasa.gov/pub/naif/CASSINI/kernels/spk/981005_PLTEPH-DE405S.bsp
#
# The following is the contents of a metakernel that was saved with
# the name 'cassMetaK.txt'.
# \begindata
# KERNELS_TO_LOAD=(
# 'naif0009.tls',
# 'cas00084.tsc',
# 'cpck05Mar2004.tpc',
# '020514_SE_SAT105.bsp',
# '981005_PLTEPH-DE405S.bsp',
# '030201AP_SK_SM546_T45.bsp',
# '04135_04171pc_psiv2.bc',
# 'cas_v37.tf',
# 'cas_iss_v09.ti')
# \begintext
#
spice.furnsh("./cassMetaK.txt")
```

```
step = 4000
# we are going to get positions between these two dates
utc = ['Jun 20, 2004', 'Dec 1, 2005']

# get et values one and two, we could vectorize str2et
etOne = spice.str2et(utc[0])
etTwo = spice.str2et(utc[1])
print("ET One: {}, ET Two: {}".format(etOne, etTwo))
```

```
ET One: 140961664.18440723, ET Two: 186667264.18308285
```

```
# get times
times = [x*(etTwo-etOne)/step + etOne for x in range(step)]

# check first few times:
print(times[0:3])
```

```
[140961664.18440723, 140973090.5844069, 140984516.98440656]
```

```
# check the documentation on spkpos before continueing
help(spice.spkpos)
```

```
Help on function spkpos in module spiceypy.spiceypy:
```

```

spkpos(targ, et, ref, abcorr, obs)
    Return the position of a target body relative to an observing
    body, optionally corrected for light time (planetary aberration)
    and stellar aberration.

    http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkpos_c.html

    :param targ: Target body name.
    :type targ: str
    :param et: Observer epoch.
    :type et: float or List of Floats
    :param ref: Reference frame of output position vector.
    :type ref: str
    :param abcorr: Aberration correction flag.
    :type abcorr: str
    :param obs: Observing body name.
    :type obs: str
    :return:
        Position of target,
        One way light time between observer and target.
    :rtype: tuple

```

```

#Run spkpos as a vectorized function
positions, lightTimes = spice.spkpos('Cassini', times, 'J2000', 'NONE', 'SATURN_
↳BARYCENTER')

# Positions is a 3xN vector of XYZ positions
print("Positions: ")
print(positions[0])

# Light times is a N vector of time
print("Light Times: ")
print(lightTimes[0])

```

```

Positions:
[-5461446.61080924 -4434793.40785864 -1200385.93315424]
Light Times:
23.8062238783

```

```

# Clean up the kernels
spice.kclear()

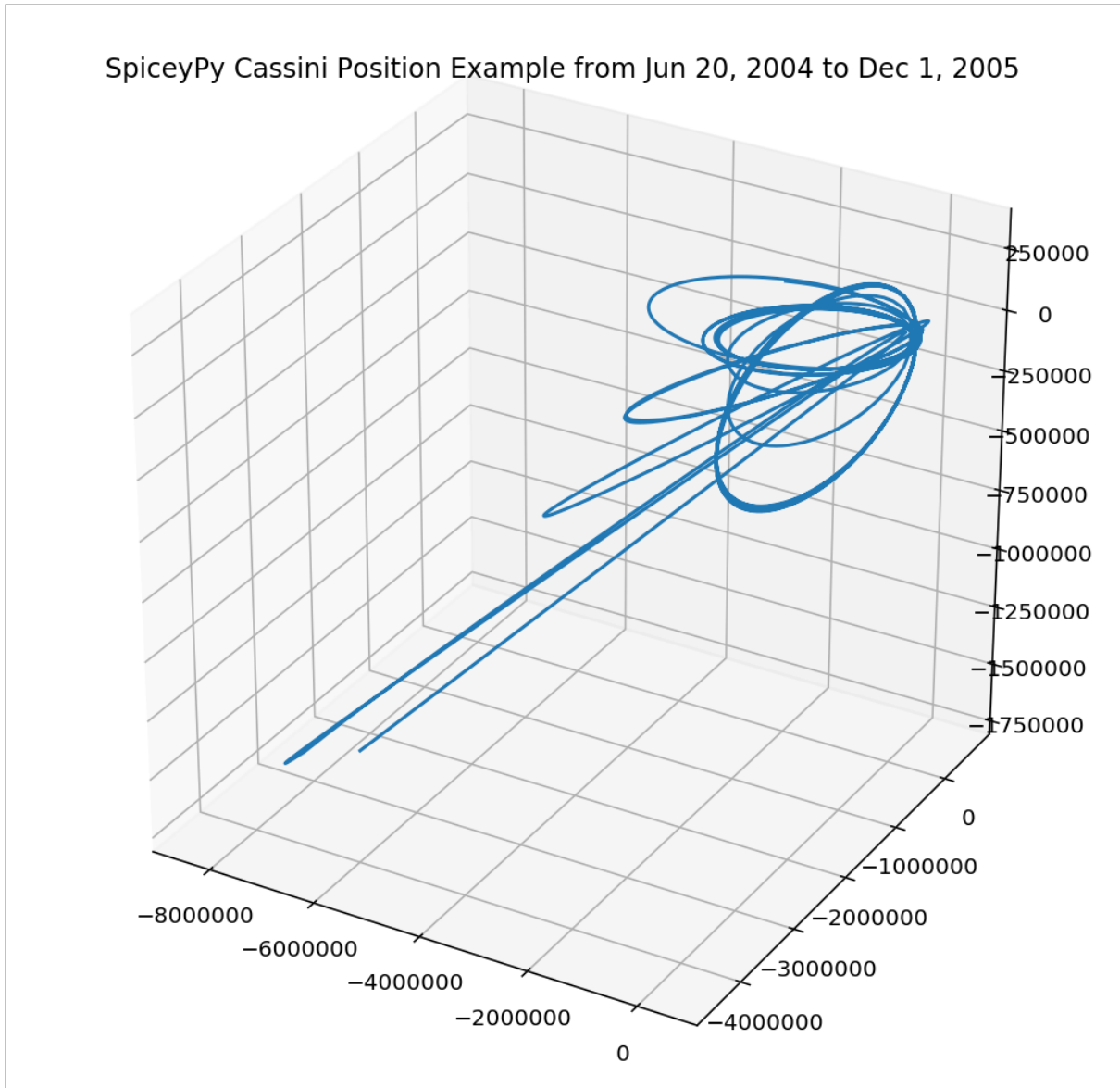
```

We will use matplotlib's 3D plotting to visualize Cassini's coordinates.

```

fig = plt.figure(figsize=(9, 9))
ax = fig.add_subplot(111, projection='3d')
ax.plot(positions.T[0], positions.T[1], positions.T[2])
plt.title('SpiceyPy Cassini Position Example from Jun 20, 2004 to Dec 1, 2005')
plt.show()

```





Here listed are the various SPICE lessons provided by the NAIF translated to use python code examples. Please refer to the NAIF lessons for the kernel files needed.

Contents:

## 5.1 Basics of SpiceyPy

### 5.1.1 Environment Set-up

Follow the installation instructions provided in installation.

### 5.1.2 Confirm SpiceyPy installation

There are multiple ways to verify your SpiceyPy installation. The first test is to simply run

```
pip list
```

You should see SpiceyPy in the list of your installed packages. If SpiceyPy is not present in the list then a configuration issue in your environment caused SpiceyPy to be installed in a non-standard way. Note this is an error prone to systems with multiple installed python versions.

If SpiceyPy is present in the pip list, then SpiceyPy is installed. Another verification step is within the python REPL run:

```
import spiceypy as spice
```

The version of the installed cspice toolkit (note: not SpiceyPy's version) should be printed out. Otherwise the Python interpreter should output an explanatory error message.

### 5.1.3 A simple example program

The following calls the SPICE function `spiceypy.spiceypy.tkvrns()` which outputs the version of cspice that SpiceyPy is wrapping.

```
import spiceypy as spice

spice.tkvrns('TOOLKIT')
```

This should output the following string:

```
'CSPICE_N0066'
```

### 5.1.4 Exceptions

SpiceyPy by default checks the spice error system for errors after all function calls and will raise `SpiceyErrors` (a python exception) when spice indicates an error. The exception message is a string that follows the format used elsewhere in spice and includes the toolkit version, the short description, explanation, long format description, and traceback (of spice calls). [Read the NAIF tutorial on exceptions here.](#)

Also, by default SpiceyPy captures the ‘found’ flags some functions return as it is not idiomatic to python and instead through a `SpiceyError` exception. This can be temporarily disabled using the `spiceypy.spiceypy.no_found_check()` context manager that allows the found flag to be returned to the user for action. Outside the context SpiceyPy functions will revert to default behavior.

```
import spiceypy as spice

spice.bodc2n(-9991) # will raise an exception

with spice.no_found_check():
    name, found = spice.bodc2n(-9991) # found is now available, no exception raised!
    assert not found # found is going to be False in this case.

spice.bodc2n(-9991) # will raise an exception again
```

There is also an accompanying context manager for enabling the default spiceypy behavior within a code block like so:

```
import spiceypy as spice

spice.bodc2n(-9991) # will raise an exception

with spice.found_check():
    name = spice.bodc2n(-9991) # will also raise an exception
```

In addition, for advanced users there are two function `spiceypy.spiceypy.found_check_off()` and `spiceypy.spiceypy.found_check_on()` which will disable and enable the behavior without use of the context manager. Additionally, a method `spiceypy.spiceypy.get_found_catch_state()` allows users to query the current state of found flag catching setting.

## 6.1 spiceypy.spiceypy module

The MIT License (MIT)

Copyright (c) [2015-2017] [Andrew Annex]

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

`spiceypy.spiceypy.appendc(item, cell)`

Append an item to a character cell.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/appndc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/appndc_c.html)

### Parameters

- **item** (*str* or *list*) – The item to append.
- **cell** (`spiceypy.utils.support_types.SpiceCell`) – The cell to append to.

`spiceypy.spiceypy.appendd(item, cell)`

Append an item to a double precision cell.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/appndd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/appndd_c.html)

**Parameters**

- **item** (*Union[float, Iterable[float]]*) – The item to append.
- **cell** (*spiceypy.utils.support\_types.SpiceCell*) – The cell to append to.

`spiceypy.spiceypy.appendi(item, cell)`

Append an item to an integer cell.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/appndi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/appndi_c.html)

**Parameters**

- **item** (*Union[float, Iterable[int]]*) – The item to append.
- **cell** (*spiceypy.utils.support\_types.SpiceCell*) – The cell to append to.

`spiceypy.spiceypy.axisar(axis, angle)`

Construct a rotation matrix that rotates vectors by a specified angle about a specified axis.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/axisar\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/axisar_c.html)

**Parameters**

- **axis** (*3 Element vector (list, tuple, numpy array)*) – Rotation axis.
- **angle** (*float*) – Rotation angle, in radians.

**Returns** Rotation matrix corresponding to axis and angle.

**Return type** `numpy array ((3, 3))`

`spiceypy.spiceypy.b1900()`

Return the Julian Date corresponding to Besselian Date 1900.0.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/b1900\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/b1900_c.html)

**Returns** The Julian Date corresponding to Besselian Date 1900.0.

**Return type** `float`

`spiceypy.spiceypy.b1950()`

Return the Julian Date corresponding to Besselian Date 1950.0.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/b1950\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/b1950_c.html)

**Returns** The Julian Date corresponding to Besselian Date 1950.0.

**Return type** `float`

`spiceypy.spiceypy.badkpv(caller, name, comp, insize, divby, intype)`

Determine if a kernel pool variable is present and if so that it has the correct size and type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/badkpv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/badkpv_c.html)

**Parameters**

- **caller** (*str*) – Name of the routine calling this routine.
- **name** (*str*) – Name of a kernel pool variable.
- **comp** (*str*) – Comparison operator.
- **insize** (*int*) – Expected size of the kernel pool variable.
- **divby** (*int*) – A divisor of the size of the kernel pool variable.
- **intype** (*str*) – Expected type of the kernel pool variable

**Returns** returns false if the kernel pool variable is OK.

**Return type** *bool*

`spiceypy.spiceypy.bltfm(frmcls, outCell=None)`

Return a SPICE set containing the frame IDs of all built-in frames of a specified class.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bltfm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bltfm_c.html)

**Parameters**

- **frmcls** (*int*) – Frame class.
- **outCell** (`spiceypy.utils.support_types.SpiceCell`) – Optional SpiceInt Cell that is returned

**Returns** Set of ID codes of frames of the specified class.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.bodc2n(code, lenout=256)`

Translate the SPICE integer code of a body into a common name for that body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodc2n\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodc2n_c.html)

**Parameters**

- **code** (*int*) – Integer ID code to be translated into a name.
- **lenout** (*int*) – Maximum length of output name.

**Returns** A common name for the body identified by code.

**Return type** *str*

`spiceypy.spiceypy.bodc2s(code, lenout=256)`

Translate a body ID code to either the corresponding name or if no name to ID code mapping exists, the string representation of the body ID value.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodc2s\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodc2s_c.html)

**Parameters**

- **code** (*int*) – Integer ID code to translate to a string.
- **lenout** (*int*) – Maximum length of output name.

**Returns** String corresponding to ‘code’.

**Return type** *str*

`spiceypy.spiceypy.boddef(name, code)`

Define a body name/ID code pair for later translation via `bodn2c()` or `bodc2n()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/boddef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/boddef_c.html)

**Parameters**

- **name** (*str*) – Common name of some body.
- **code** (*int*) – Integer code for that body.

`spiceypy.spiceypy.bodfnd(body, item)`

Determine whether values exist for some item for any body in the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodfnd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodfnd_c.html)

**Parameters**

- **body** (*int*) – ID code of body.

- **item** (*str*) – Item to find (“RADII”, “NUT\_AMP\_RA”, etc.).

**Returns** True if the item is in the kernel pool, and is False if it is not.

**Return type** *bool*

`spiceypy.spiceypy.bodn2c` (*name*)

Translate the name of a body or object to the corresponding SPICE integer ID code.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodn2c\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodn2c_c.html)

**Parameters** **name** (*str*) – Body name to be translated into a SPICE ID code.

**Returns** SPICE integer ID code for the named body.

**Return type** *int*

`spiceypy.spiceypy.bods2c` (*name*)

Translate a string containing a body name or ID code to an integer code.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bods2c\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bods2c_c.html)

**Parameters** **name** (*str*) – String to be translated to an ID code.

**Returns** Integer ID code corresponding to name.

**Return type** *int*

`spiceypy.spiceypy.bodvar` (*body, item, dim*)

Deprecated: This routine has been superseded by `bodvcd()` and `bodvrd()`. This routine is supported for purposes of backward compatibility only.

Return the values of some item for any body in the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodvar\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodvar_c.html)

**Parameters**

- **body** (*int*) – ID code of body.
- **item** (*str*) – Item for which values are desired, (“RADII”, “NUT\_PREC\_ANGLES”, etc.)
- **dim** (*int*) – Number of values returned.

**Returns** values

**Return type** Array of floats

`spiceypy.spiceypy.bodvcd` (*bodyid, item, maxn*)

Fetch from the kernel pool the double precision values of an item associated with a body, where the body is specified by an integer ID code.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodvcd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodvcd_c.html)

**Parameters**

- **bodyid** (*int*) – Body ID code.
- **item** (*str*) – Item for which values are desired, (“RADII”, “NUT\_PREC\_ANGLES”, etc.)
- **maxn** (*int*) – Maximum number of values that may be returned.

**Returns** dim, values

**Return type** tuple

`spiceypy.spiceypy.bodvrd` (*bodynm, item, maxn*)

Fetch from the kernel pool the double precision values of an item associated with a body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bodvrd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bodvrd_c.html)

#### Parameters

- **bodynm** (*str*) – Body name.
- **item** (*str*) – Item for which values are desired, (“RADII”, “NUT\_PREC\_ANGLES”, etc.)
- **maxn** (*int*) – Maximum number of values that may be returned.

**Returns** tuple of (dim, values)

**Return type** tuple

`spiceypy.spiceypy.brcktd` (*number, end1, end2*)

Bracket a number. That is, given a number and an acceptable interval, make sure that the number is contained in the interval. (If the number is already in the interval, leave it alone. If not, set it to the nearest endpoint of the interval.)

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/brcktd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/brcktd_c.html)

#### Parameters

- **number** (*float*) – Number to be bracketed.
- **end1** (*float*) – One of the bracketing endpoints for number.
- **end2** (*float*) – The other bracketing endpoint for number.

**Returns** value within an interval

**Return type** float

`spiceypy.spiceypy.brckti` (*number, end1, end2*)

Bracket a number. That is, given a number and an acceptable interval, make sure that the number is contained in the interval. (If the number is already in the interval, leave it alone. If not, set it to the nearest endpoint of the interval.)

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/brckti\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/brckti_c.html)

#### Parameters

- **number** (*int*) – Number to be bracketed.
- **end1** (*int*) – One of the bracketing endpoints for number.
- **end2** (*int*) – The other bracketing endpoint for number.

**Returns** value within an interval

**Return type** int

`spiceypy.spiceypy.bscho` (*value, ndim, lenvals, array, order*)

Do a binary search for a given value within a character string array, accompanied by an order vector. Return the index of the matching array entry, or -1 if the key value is not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bscho\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bscho_c.html)

#### Parameters

- **value** (*str*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.

- **lenvals** (*int*) – String length.
- **array** (*list of strings*) – Character string array to search.
- **order** (*Array of ints*) – Order vector.

**Returns** index

**Return type** int

`spiceypy.spiceypy.bschoi` (*value, ndim, array, order*)

Do a binary search for a given value within an integer array, accompanied by an order vector. Return the index of the matching array entry, or -1 if the key value is not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bschoi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bschoi_c.html)

**Parameters**

- **value** (*int*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of ints*) – Integer array to search.
- **order** (*Array of ints*) – Order vector.

**Returns** index

**Return type** int

`spiceypy.spiceypy.bsrchc` (*value, ndim, lenvals, array*)

Do a binary search for a given value within a character string array. Return the index of the first matching array entry, or -1 if the key value was not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bsrchc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bsrchc_c.html)

**Parameters**

- **value** (*str*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.
- **lenvals** (*int*) – String length.
- **array** (*list of strings*) – Character string array to search.

**Returns** index

**Return type** int

`spiceypy.spiceypy.bsrchd` (*value, ndim, array*)

Do a binary search for a key value within a double precision array, assumed to be in increasing order. Return the index of the matching array entry, or -1 if the key value is not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bsrchd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bsrchd_c.html)

**Parameters**

- **value** (*float*) – Value to find in array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of floats*) – Array to be searched.

**Returns** index

**Return type** int



`spiceypy.spiceypy.bsrchi` (*value, ndim, array*)

Do a binary search for a key value within an integer array, assumed to be in increasing order. Return the index of the matching array entry, or -1 if the key value is not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/bsrchi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/bsrchi_c.html)

**Parameters**

- **value** (*int*) – Value to find in array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of ints*) – Array to be searched.

**Returns** *index*

**Return type** *int*

`spiceypy.spiceypy.card` (*cell*)

Return the cardinality (current number of elements) in a cell of any data type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/card\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/card_c.html)

**Parameters** **cell** (`spiceypy.utils.support_types.SpiceCell`) – Input cell.

**Returns** the number of elements in a cell of any data type.

**Return type** *int*

`spiceypy.spiceypy.ccifrm` (*frclss, clssid, lenout=256*)

Return the frame name, frame ID, and center associated with a given frame class and class ID.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ccifrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ccifrm_c.html)

**Parameters**

- **frclss** (*int*) – Class of frame.
- **clssid** (*int*) – Class ID of frame.
- **lenout** (*int*) – Maximum length of output string.

**Returns** the frame name, frame ID, center.

**Return type** *tuple*

`spiceypy.spiceypy.cgv2e1` (*center, vec1, vec2*)

Form a SPICE ellipse from a center vector and two generating vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cgv2e1\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cgv2e1_c.html)

**Parameters**

- **center** (*3-Element Array of floats*) – Center Vector
- **vec1** (*3-Element Array of floats*) – Vector 1
- **vec2** (*3-Element Array of floats*) – Vector 2

**Returns** *Ellipse*

**Return type** `spiceypy.utils.support_types.Ellipse`

`spiceypy.spiceypy.chbder` (*cp, degp, x2s, x, nderiv*)

Given the coefficients for the Chebyshev expansion of a polynomial, this returns the value of the polynomial and its first nderiv derivatives evaluated at the input X.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/chbder\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/chbder_c.html)

**Parameters**

- **cp** (*Array of floats*) – degp+1 Chebyshev polynomial coefficients.
- **degp** (*int*) – Degree of polynomial.
- **x2s** (*Array of floats*) – Transformation parameters of polynomial.
- **x** (*float*) – Value for which the polynomial is to be evaluated
- **nderiv** (*int*) – The number of derivatives to compute

**Returns** Array of the derivatives of the polynomial

**Return type** Array of floats

`spiceypy.spiceypy.checkForSpiceError(f)`

Internal function to check :param f: :raise stypes.SpiceyError:

`spiceypy.spiceypy.chkIn(module)`

Inform the SPICE error handling mechanism of entry into a routine.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/chkin\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/chkin_c.html)

**Parameters** **module** (*str*) – The name of the calling routine.

`spiceypy.spiceypy.chkout(module)`

Inform the SPICE error handling mechanism of exit from a routine.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/chkout\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/chkout_c.html)

**Parameters** **module** (*str*) – The name of the calling routine.

`spiceypy.spiceypy.cidfrm(cent, lenout=256)`

Retrieve frame ID code and name to associate with a frame center.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cidfrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cidfrm_c.html)

**Parameters**

- **cent** (*int*) – An object to associate a frame with.
- **lenout** (*int*) – Available space in output string ffname.

**Returns** frame ID code, name to associate with a frame center.

**Return type** tuple

`spiceypy.spiceypy.ckcls(handle)`

Close an open CK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckcls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckcls_c.html)

**Parameters** **handle** (*int*) – Handle of the CK file to be closed.

`spiceypy.spiceypy.ckcov(ck, idcode, needav, level, tol, timsys, cover=None)`

Find the coverage window for a specified object in a specified CK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckcov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckcov_c.html)

**Parameters**

- **ck** (*str*) – Name of CK file.
- **idcode** (*int*) – ID code of object.
- **needav** (*bool*) – Flag indicating whether angular velocity is needed.
- **level** (*str*) – Coverage level: (SEGMENT OR INTERVAL)

- **tol** (*float*) – Tolerance in ticks.
- **timesys** (*str*) – Time system used to represent coverage.
- **cover** (*Optional SpiceCell*) – Window giving coverage for idcode.

**Returns** coverage window for a specified object in a specified CK file

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.ckgp` (*inst, sclkdp, tol, ref*)

Get pointing (attitude) for a specified spacecraft clock time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckgp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckgp_c.html)

**Parameters**

- **inst** (*int*) – NAIF ID of instrument, spacecraft, or structure.
- **sclkdp** (*float*) – Encoded spacecraft clock time.
- **tol** (*float*) – Time tolerance.
- **ref** (*str*) – Reference frame.

**Returns** C-matrix pointing data, Output encoded spacecraft clock time

**Return type** tuple

`spiceypy.spiceypy.ckgpav` (*inst, sclkdp, tol, ref*)

Get pointing (attitude) and angular velocity for a specified spacecraft clock time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckgpav\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckgpav_c.html)

**Parameters**

- **inst** (*int*) – NAIF ID of instrument, spacecraft, or structure.
- **sclkdp** (*float*) – Encoded spacecraft clock time.
- **tol** (*float*) – Time tolerance.
- **ref** (*str*) – Reference frame.

**Returns** C-matrix pointing data, Angular velocity vector, Output encoded spacecraft clock time.

**Return type** tuple

`spiceypy.spiceypy.cklpf` (*filename*)

Load a CK pointing file for use by the CK readers. Return that file's handle, to be used by other CK routines to refer to the file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cklpf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cklpf_c.html)

**Parameters** **filename** (*str*) – Name of the CK file to be loaded.

**Returns** Loaded file's handle.

**Return type** int

`spiceypy.spiceypy.ckobj` (*ck, outCell=None*)

Find the set of ID codes of all objects in a specified CK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckobj\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckobj_c.html)

**Parameters**

- **ck** (*str*) – Name of CK file.

- **outCell** (*Optional spiceypy.utils.support\_types.SpiceCell*) – Optional user provided Spice Int cell.

**Returns** Set of ID codes of objects in CK file.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.ckopn` (*filename, ifname, ncomch*)

Open a new CK file, returning the handle of the opened file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckopn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckopn_c.html)

**Parameters**

- **filename** (*str*) – The name of the CK file to be opened.
- **ifname** (*str*) – The internal filename for the CK.
- **ncomch** (*int*) – The number of characters to reserve for comments.

**Returns** The handle of the opened CK file.

**Return type** `int`

`spiceypy.spiceypy.ckupf` (*handle*)

Unload a CK pointing file so that it will no longer be searched by the readers.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckupf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckupf_c.html)

**Parameters** **handle** (*int*) – Handle of CK file to be unloaded

`spiceypy.spiceypy.ckw01` (*handle, begtim, endtim, inst, ref, avflag, segid, nrec, sclkdp, quats, avvs*)

Add a type 1 segment to a C-kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckw01\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckw01_c.html)

**Parameters**

- **handle** (*int*) – Handle of an open CK file.
- **begtim** (*float*) – The beginning encoded SCLK of the segment.
- **endtim** (*float*) – The ending encoded SCLK of the segment.
- **inst** (*int*) – The NAIF instrument ID code.
- **ref** (*str*) – The reference frame of the segment.
- **avflag** (*bool*) – True if the segment will contain angular velocity.
- **segid** (*str*) – Segment identifier.
- **nrec** (*int*) – Number of pointing records.
- **sclkdp** (*Array of floats*) – Encoded SCLK times.
- **quats** (*Nx4-Element Array of floats*) – Quaternions representing instrument pointing.
- **avvs** (*Nx3-Element Array of floats*) – Angular velocity vectors.

`spiceypy.spiceypy.ckw02` (*handle, begtim, endtim, inst, ref, segid, nrec, start, stop, quats, avvs, rates*)

Write a type 2 segment to a C-kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckw02\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckw02_c.html)

**Parameters**

- **handle** (*int*) – Handle of an open CK file.

- **begtim** (*float*) – The beginning encoded SCLK of the segment.
- **endtim** (*float*) – The ending encoded SCLK of the segment.
- **inst** (*int*) – The NAIF instrument ID code.
- **ref** (*str*) – The reference frame of the segment.
- **segid** (*str*) – Segment identifier.
- **nrec** (*int*) – Number of pointing records.
- **start** (*Array of floats*) – Encoded SCLK interval start times.
- **stop** (*Array of floats*) – Encoded SCLK interval stop times.
- **quats** (*Nx4-Element Array of floats*) – Quaternions representing instrument pointing.
- **avvs** (*Nx3-Element Array of floats*) – Angular velocity vectors.
- **rates** (*Array of floats*) – Number of seconds per tick for each interval.

`spiceypy.spiceypy.ckw03` (*handle, begtim, endtim, inst, ref, avflag, segid, nrec, sclkdp, quats, avvs, nints, starts*)

Add a type 3 segment to a C-kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckw03\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckw03_c.html)

#### Parameters

- **handle** (*int*) – Handle of an open CK file.
- **begtim** (*float*) – The beginning encoded SCLK of the segment.
- **endtim** (*float*) – The ending encoded SCLK of the segment.
- **inst** (*int*) – The NAIF instrument ID code.
- **ref** (*str*) – The reference frame of the segment.
- **avflag** (*bool*) – True if the segment will contain angular velocity.
- **segid** (*str*) – Segment identifier.
- **nrec** (*int*) – Number of pointing records.
- **sclkdp** (*Array of floats*) – Encoded SCLK times.
- **quats** (*Nx4-Element Array of floats*) – Quaternions representing instrument pointing.
- **avvs** (*Nx3-Element Array of floats*) – Angular velocity vectors.
- **nints** (*int*) – Number of intervals.
- **starts** (*Array of floats*) – Encoded SCLK interval start times.

`spiceypy.spiceypy.ckw05` (*handle, subtype, degree, begtim, endtim, inst, ref, avflag, segid, sclkdp, packets, rate, nints, starts*)

Write a type 5 segment to a CK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ckw05\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ckw05_c.html)

#### Parameters

- **handle** (*int*) – Handle of an open CK file.
- **subtype** (*int*) – CK type 5 subtype code. Can be: 0, 1, 2, 3 see naif docs via link above.

- **degree** (*int*) – Degree of interpolating polynomials.
- **begtim** (*float*) – The beginning encoded SCLK of the segment.
- **endtim** (*float*) – The ending encoded SCLK of the segment.
- **inst** (*int*) – The NAIF instrument ID code.
- **ref** (*str*) – The reference frame of the segment.
- **avflag** (*bool*) – True if the segment will contain angular velocity.
- **segid** (*str*) – Segment identifier.
- **sclkd** (*Array of floats*) – Encoded SCLK times.
- **packts** (*Some NxM vector of floats*) – Array of packets.
- **rate** (*float*) – Nominal SCLK rate in seconds per tick.
- **nints** (*int*) – Number of intervals.
- **starts** (*Array of floats*) – Encoded SCLK interval start times.

`spiceypy.spiceypy.cleared()`

`spiceypy.spiceypy.clight()`

Return the speed of light in a vacuum (IAU official value, in km/sec).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/clight\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/clight_c.html)

**Returns** The function returns the speed of light in vacuum (km/sec).

**Return type** float

`spiceypy.spiceypy.clpool()`

Remove all variables from the kernel pool. Watches on kernel variables are retained.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/clpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/clpool_c.html)

`spiceypy.spiceypy.cltext(fname)`

Internal undocumented command for closing a text file opened by RDTEXT.

No URL available; relevant lines from SPICE source:

FORTRAN SPICE, rdtext.f:

```
C$Procedure  CLTEXT ( Close a text file opened by RDTEXT)
      ENTRY  CLTEXT ( FILE )
      CHARACTER*(*)      FILE
C   VARIABLE  I/O  DESCRIPTION
C   -----  ---  -----
C   FILE      I   Text file to be closed.
```

CSPICE, rdtext.c:

```
/* $Procedure  CLTEXT ( Close a text file opened by RDTEXT) */
/* Subroutine */ int cltext_(char *file, ftnlen file_len)
```

**Parameters** **fname** (*str*) – Text file to be closed.

`spiceypy.spiceypy.cmprss(delim, n, instr, lenout=256)`

Compress a character string by removing occurrences of more than N consecutive occurrences of a specified character.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cmprss\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cmprss_c.html)

**Parameters**

- **delim** (*str*) – Delimiter to be compressed.
- **n** (*int*) – Maximum consecutive occurrences of delim.
- **instr** (*str*) – Input string.
- **lenout** (*Optional int*) – Optional available space in output string.

**Returns** Compressed string.

**Return type** `str`

`spiceypy.spiceypy.cnmfrm` (*cname, lenout=256*)

Retrieve frame ID code and name to associate with an object.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cnmfrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cnmfrm_c.html)

**Parameters**

- **cname** (*int*) – Name of the object to find a frame for.
- **lenout** (*int*) – Maximum length available for frame name.

**Returns** The ID code of the frame associated with `cname`, The name of the frame with ID `frcode`.

**Return type** `tuple`

`spiceypy.spiceypy.conics` (*elts, et*)

Determine the state (position, velocity) of an orbiting body from a set of elliptic, hyperbolic, or parabolic orbital elements.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/conics\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/conics_c.html)

**Parameters**

- **elts** (*8-Element Array of floats*) – Conic elements.
- **et** (*float*) – Input time.

**Returns** State of orbiting body at `et`.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.convrt` (*x, inunit, outunit*)

Take a measurement `X`, the units associated with `X`, and units to which `X` should be converted; return `Y` the value of the measurement in the output units.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/convrt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/convrt_c.html)

**Parameters**

- **x** (*float*) – Number representing a measurement in some units.
- **inunit** (*str*) – The units in which `x` is measured.
- **outunit** (*str*) – Desired units for the measurement.

**Returns** The measurement in the desired units.

**Return type** `float`

`spiceypy.spiceypy.copy` (*cell*)

Copy the contents of a `SpiceCell` of any data type to another cell of the same type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/copy\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/copy_c.html)

**Parameters** `cell` (`spiceypy.utils.support_types.SpiceCell`) – Cell to be copied.

**Returns** New cell

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.cpos` (*string, chars, start*)

Find the first occurrence in a string of a character belonging to a collection of characters, starting at a specified location, searching forward.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cpos\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cpos_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **chars** (*str*) – A collection of characters.
- **start** (*int*) – Position to begin looking for one of chars.

**Returns** The index of the first character of *str* at or following index *start* that is in the collection *chars*.

**Return type** `int`

`spiceypy.spiceypy.cposr` (*string, chars, start*)

Find the first occurrence in a string of a character belonging to a collection of characters, starting at a specified location, searching in reverse.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cposr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cposr_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **chars** (*str*) – A collection of characters.
- **start** (*int*) – Position to begin looking for one of chars.

**Returns** The index of the last character of *str* at or before index *start* that is in the collection *chars*.

**Return type** `int`

`spiceypy.spiceypy.cvpool` (*agent*)

Indicate whether or not any watched kernel variables that have a specified agent on their notification list have been updated.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cvpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cvpool_c.html)

**Parameters** **agent** (*str*) – Name of the agent to check for notices.

**Returns** True if variables for “agent” have been updated.

**Return type** *bool*

`spiceypy.spiceypy.cyllat` (*r, lonc, z*)

Convert from cylindrical to latitudinal coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cyllat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cyllat_c.html)

**Parameters**

- **r** (*float*) – Distance of point from z axis.
- **lonc** (*float*) – Cylindrical angle of point from XZ plane(radians).
- **z** (*float*) – Height of point above XY plane.

**Returns** Distance, Longitude (radians), and Latitude of point (radians).

**Return type** `tuple`



`spiceypy.spiceypy.cylrec(r, lon, z)`

Convert from cylindrical to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cylrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cylrec_c.html)

**Parameters**

- **r** (*float*) – Distance of a point from z axis.
- **lon** (*float*) – Angle (radians) of a point from xZ plane.
- **z** (*float*) – Height of a point above xY plane.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.cylsph(r, lonc, z)`

Convert from cylindrical to spherical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/cylsph\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/cylsph_c.html)

**Parameters**

- **r** (*float*) – Rectangular coordinates of the point.
- **lonc** (*float*) – Angle (radians) of point from XZ plane.
- **z** (*float*) – Height of point above XY plane.

**Returns** Distance of point from origin, Polar angle (co-latitude in radians) of point, Azimuthal angle (longitude) of point (radians).

**Return type** tuple

`spiceypy.spiceypy.dafac(handle, buffer)`

Add comments from a buffer of character strings to the comment area of a binary DAF file, appending them to any comments which are already present in the file's comment area.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafac\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafac_c.html)

**Parameters**

- **handle** (*int*) – handle of a DAF opened with write access.
- **buffer** (*list[str]*) – Buffer of comments to put into the comment area.

`spiceypy.spiceypy.dafbbs(handle)`

Begin a backward search for arrays in a DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafbbs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafbbs_c.html)

**Parameters** **handle** (*int*) – Handle of DAF to be searched.

`spiceypy.spiceypy.dafbfs(handle)`

Begin a forward search for arrays in a DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafbfs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafbfs_c.html)

**Parameters** **handle** (*int*) – Handle of file to be searched.

`spiceypy.spiceypy.dafcls(handle)`

Close the DAF associated with a given handle.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafcls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafcls_c.html)

**Parameters** **handle** (*int*) – Handle of DAF to be closed.

`spiceypy.spiceypy.dafcs` (*handle*)

Select a DAF that already has a search in progress as the one to continue searching.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/dafcs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/dafcs_c.html)

**Parameters** `handle` (*int*) – Handle of DAF to continue searching.

`spiceypy.spiceypy.dafdc` (*handle*)

Delete the entire comment area of a specified DAF file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/dafdc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/dafdc_c.html)

**Parameters** `handle` (*int*) – The handle of a binary DAF opened for writing.

`spiceypy.spiceypy.dafec` (*handle, bufsiz, lenout=256*)

Extract comments from the comment area of a binary DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/dafec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/dafec_c.html)

**Parameters**

- **handle** (*int*) – Handle of binary DAF opened with read access.
- **bufsiz** (*int*) – Maximum size, in lines, of buffer.
- **lenout** (*int*) – Length of strings in output buffer.

**Returns** Number of extracted comment lines, buffer where extracted comment lines are placed, Indicates whether all comments have been extracted.

**Return type** tuple

`spiceypy.spiceypy.daffna` ()

Find the next (forward) array in the current DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/daffna\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/daffna_c.html)

**Returns** True if an array was found.

**Return type** *bool*

`spiceypy.spiceypy.daffpa` ()

Find the previous (backward) array in the current DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/daffpa\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/daffpa_c.html)

**Returns** True if an array was found.

**Return type** *bool*

`spiceypy.spiceypy.dafgda` (*handle, begin, end*)

Read the double precision data bounded by two addresses within a DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/dafgda\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/dafgda_c.html)

**Parameters**

- **handle** (*int*) – Handle of a DAF.
- **begin** (*int*) – Initial address within file.
- **end** (*int*) – Final address within file.

**Returns** Data contained between begin and end.

**Return type** Array of floats

`spiceypy.spiceypy.dafgh()`

Return (get) the handle of the DAF currently being searched.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafgh\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafgh_c.html)

**Returns** Handle for current DAF.

**Return type** int

`spiceypy.spiceypy.dafgn(lenout=256)`

Return (get) the name for the current array in the current DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafgn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafgn_c.html)

**Parameters** `lenout` (*int*) – Length of array name string.

**Returns** Name of current array.

**Return type** str

`spiceypy.spiceypy.dafgs(n=125)`

Return (get) the summary for the current array in the current DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafgs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafgs_c.html)

**Parameters** `n` – Optional length N for result Array.

**Returns** Summary for current array.

**Return type** Array of floats

`spiceypy.spiceypy.dafgsr(handle, recno, begin, end)`

Read a portion of the contents of (words in) a summary record in a DAF file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafgsr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafgsr_c.html)

**Parameters**

- **handle** (*int*) – Handle of DAF.
- **recno** (*int*) – Record number; word indices are 1-based, 1 to 128 inclusive.
- **begin** (*int*) – Index of first word to read from record, will be clamped > 0.
- **end** (*int*) – Index of last word to read, will be clamped < 129

**Returns** Contents of request sub-record

**Return type** float numpy.ndarray

`spiceypy.spiceypy.dafopr(fname)`

Open a DAF for subsequent read requests.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafopr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafopr_c.html)

**Parameters** `fname` (*str*) – Name of DAF to be opened.

**Returns** Handle assigned to DAF.

**Return type** int

`spiceypy.spiceypy.dafopw(fname)`

Open a DAF for subsequent write requests.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafopw\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafopw_c.html)

**Parameters** `fname` (*str*) – Name of DAF to be opened.

**Returns** Handle assigned to DAF.

**Return type** int

`spiceypy.spiceypy.dafps` (*nd, ni, dc, ic*)

Pack (assemble) an array summary from its double precision and integer components.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafps\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafps_c.html)

**Parameters**

- **nd** (*int*) – Number of double precision components.
- **ni** (*int*) – Number of integer components.
- **dc** (*Array of floats*) – Double precision components.
- **ic** (*Array of ints*) – Integer components.

**Returns** Array summary.

**Return type** Array of floats

`spiceypy.spiceypy.dafrda` (*handle, begin, end*)

Read the double precision data bounded by two addresses within a DAF.

Deprecated: This routine has been superseded by `dafgda()` and `dafgsr()`. This routine is supported for purposes of backward compatibility only.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafrda\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafrda_c.html)

**Parameters**

- **handle** (*int*) – Handle of a DAF.
- **begin** (*int*) – Initial address within file.
- **end** (*int*) – Final address within file.

**Returns** Data contained between begin and end.

**Return type** Array of floats

`spiceypy.spiceypy.dafrfr` (*handle, lenout=256*)

Read the contents of the file record of a DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafrfr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafrfr_c.html)

**Parameters**

- **handle** (*int*) – Handle of an open DAF file.
- **lenout** (*int*) – Available room in the output string

**Returns** Number of double precision components in summaries, Number of integer components in summaries, Internal file name, Forward list pointer, Backward list pointer, Free address pointer.

**Return type** tuple

`spiceypy.spiceypy.dafrs` (*insum*)

Change the summary for the current array in the current DAF.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafrs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafrs_c.html)

**Parameters** **insum** (*Array of floats*) – New summary for current array.

`spiceypy.spiceypy.dafus` (*insum, nd, ni*)

Unpack an array summary into its double precision and integer components.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dafus\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dafus_c.html)

**Parameters**

- **insum** (*Array of floats*) – Array summary.
- **nd** (*int*) – Number of double precision components.
- **ni** (*int*) – Number of integer components.

**Returns** Double precision components, Integer components.

**Return type** tuple

`spiceypy.spiceypy.dasac` (*handle, buffer*)

Add comments from a buffer of character strings to the comment area of a binary DAS file, appending them to any comments which are already present in the file's comment area.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasac\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasac_c.html)

**Parameters**

- **handle** (*int*) – DAS handle of a file opened with write access.
- **buffer** (*Array of strs*) – Buffer of lines to be put into the comment area.

`spiceypy.spiceypy.dascls` (*handle*)

Close a DAS file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dascls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dascls_c.html)

**Parameters** **handle** (*int*) – Handle of an open DAS file.

`spiceypy.spiceypy.dasdc` (*handle*)

Delete the entire comment area of a previously opened binary DAS file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasdc\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasdc_c.html)

**Parameters** **handle** (*int*) – The handle of a binary DAS file opened for writing.

`spiceypy.spiceypy.dasec` (*handle, bufsiz=256, buflen=256*)

Extract comments from the comment area of a binary DAS file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasec_c.html)

**Parameters**

- **handle** (*int*) – Handle of binary DAS file open with read access.
- **bufsiz** (*int*) – Maximum size, in lines, of buffer.
- **buflen** (*int*) – Line length associated with buffer.

**Returns** Number of comments extracted from the DAS file, Buffer in which extracted comments are placed, Indicates whether all comments have been extracted.

**Return type** tuple

`spiceypy.spiceypy.dashfn` (*handle, lenout=256*)

Return the name of the DAS file associated with a handle.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dashfn\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dashfn_c.html)

**Parameters**

- **handle** (*int*) – Handle of a DAS file.
- **lenout** (*int*) – Length of output file name string.

**Returns** Corresponding file name.

**Return type** str

`spiceypy.spiceypy.dasonw` (*fname*, *ftype*, *ifname*, *ncomch*)  
Internal undocumented command for creating a new DAS file

**Parameters**

- **fname** (*str*) – filename
- **ftype** (*str*) – type
- **ifname** (*str*) – internal file name
- **ncomch** (*int*) – amount of comment area

**Returns** Handle to new DAS file

**Return type** int

`spiceypy.spiceypy.dasopr` (*fname*)  
Open a DAS file for reading.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasopr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasopr_c.html)

**Parameters** **fname** (*str*) – Name of a DAS file to be opened.

**Returns** Handle assigned to the opened DAS file.

**Return type** int

`spiceypy.spiceypy.dasopw` (*fname*)  
Open a DAS file for writing.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasopw\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasopw_c.html) :param *fname*: Name of a DAS file to be opened. :type *fname*: str :return: Handle assigned to the opened DAS file.

`spiceypy.spiceypy.dasrfr` (*handle*, *lenout*=256)  
Return the contents of the file record of a specified DAS file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dasrfr\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dasrfr_c.html)

**Parameters**

- **handle** (*int*) – DAS file handle.
- **lenout** (*str*) – length of output str

**Returns** ID word, DAS internal file name, Number of reserved records in file,

Number of characters in use in reserved rec. area, Number of comment records in file, Number of characters in use in comment area. :rtype: tuple

`spiceypy.spiceypy.dcyldr` (*x*, *y*, *z*)  
This routine computes the Jacobian of the transformation from rectangular to cylindrical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dcyldr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dcyldr_c.html)

**Parameters**

- **x** (*float*) – X-coordinate of point.
- **y** (*float*) – Y-coordinate of point.
- **z** (*float*) – Z-coordinate of point.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.deltet` (*epoch, eptype*)

Return the value of Delta ET (ET-UTC) for an input epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/deltet\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/deltet_c.html)

**Parameters**

- **epoch** (*float*) – Input epoch (seconds past J2000).
- **eptype** (*str*) – Type of input epoch (“UTC” or “ET”).

**Returns** Delta ET (ET-UTC) at input epoch.

**Return type** float

`spiceypy.spiceypy.det` (*m1*)

Compute the determinant of a double precision 3x3 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/det\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/det_c.html)

**Parameters** **m1** (*3x3-Element Array of floats*) – Matrix whose determinant is to be found.

**Returns** The determinant of the matrix.

**Return type** float

`spiceypy.spiceypy.dgeodr` (*x, y, z, re, f*)

This routine computes the Jacobian of the transformation from rectangular to geodetic coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dgeodr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dgeodr_c.html)

**Parameters**

- **x** (*float*) – X-coordinate of point.
- **y** (*float*) – Y-coordinate of point.
- **z** (*float*) – Z-coord
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.diags2` (*symmat*)

Diagonalize a symmetric 2x2 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/diags2\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/diags2_c.html)

**Parameters** **symmat** (*2x2-Element Array of floats*) – A symmetric 2x2 matrix.

**Returns** A diagonal matrix similar to *symmat*, A rotation used as the similarity transformation.

**Return type** tuple

`spiceypy.spiceypy.diff` (*a, b*)

Take the difference of two sets of any data type to form a third set. [http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/diff\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/diff_c.html)

**Parameters**

- **a** (`spiceypy.utils.support_types.SpiceCell`) – First input set.
- **b** (`spiceypy.utils.support_types.SpiceCell`) – Second input set.

**Returns** Difference of a and b.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.dlabbs` (*handle*)

Begin a backward segment search in a DLA file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dlabbs\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dlabbs_c.html)

**Parameters** **handle** (*int*) – Handle of open DLA file.

**Returns** Descriptor of last segment in DLA file

**Return type** *spiceypy.utils.support\_types.SpiceDLADescr*

`spiceypy.spiceypy.dlabfs` (*handle*)

Begin a forward segment search in a DLA file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dlabfs\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dlabfs_c.html)

**Parameters** **handle** (*int*) – Handle of open DLA file.

**Returns** Descriptor of next segment in DLA file

**Return type** *spiceypy.utils.support\_types.SpiceDLADescr*

`spiceypy.spiceypy.dlafns` (*handle, descr*)

Find the segment following a specified segment in a DLA file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dlafns\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dlafns_c.html)

**Parameters**

- **handle** (*c\_int*) – Handle of open DLA file.
- **descr** (*spiceypy.utils.support\_types.SpiceDLADescr*) – Descriptor of a DLA segment.

**Returns** Descriptor of next segment in DLA file

**Return type** *spiceypy.utils.support\_types.SpiceDLADescr*

`spiceypy.spiceypy.dlafps` (*handle, descr*)

Find the segment preceding a specified segment in a DLA file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dlafps\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dlafps_c.html)

**Parameters**

- **handle** (*c\_int*) – Handle of open DLA file.
- **descr** (*spiceypy.utils.support\_types.SpiceDLADescr*) – Descriptor of a segment in DLA file.

**Returns** Descriptor of previous segment in DLA file

**Return type** *spiceypy.utils.support\_types.SpiceDLADescr*

`spiceypy.spiceypy.dlatdr` (*x, y, z*)

This routine computes the Jacobian of the transformation from rectangular to latitudinal coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dlatdr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dlatdr_c.html)

**Parameters**

- **x** (*float*) – X-coordinate of point.
- **y** (*float*) – Y-coordinate of point.



- **z** (*float*) – Z-coord

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.dp2hx` (*number*, *lenout*=256)

Convert a double precision number to an equivalent character string using base 16 “scientific notation.”

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dp2hx\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dp2hx_c.html)

**Parameters**

- **number** (*float*) – D.p. number to be converted.
- **lenout** (*int*) – Available space for output string.

**Returns** Equivalent character string, left justified.

**Return type** str

`spiceypy.spiceypy.dpgrdr` (*body*, *x*, *y*, *z*, *re*, *f*)

This routine computes the Jacobian matrix of the transformation from rectangular to planetographic coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dpgrdr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dpgrdr_c.html)

**Parameters**

- **body** (*str*) – Body with which coordinate system is associated.
- **x** (*float*) – X-coordinate of point.
- **y** (*float*) – Y-coordinate of point.
- **z** (*float*) – Z-coordinate of point.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.dpmax` ()

Return the value of the largest (positive) number representable in a double precision variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dpmax\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dpmax_c.html)

**Returns** The largest (positive) number representable in a double precision variable.

**Return type** float

`spiceypy.spiceypy.dpmin` ()

Return the value of the smallest (negative) number representable in a double precision variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dpmin\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dpmin_c.html)

**Returns** The smallest (negative) number that can be represented in a double precision variable.

**Return type** float

`spiceypy.spiceypy.dpr` ()

Return the number of degrees per radian.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dpr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dpr_c.html)

**Returns** The number of degrees per radian.

**Return type** float

`spiceypy.spiceypy.drdcyl` (*r, lon, z*)

This routine computes the Jacobian of the transformation from cylindrical to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/drdcyl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/drdcyl_c.html)

**Parameters**

- **r** (*float*) – Distance of a point from the origin.
- **lon** (*float*) – Angle of the point from the xz plane in radians.
- **z** (*float*) – Height of the point above the xy plane.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.drdgeo` (*lon, lat, alt, re, f*)

This routine computes the Jacobian of the transformation from geodetic to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/drddgeo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/drddgeo_c.html)

**Parameters**

- **lon** (*float*) – Geodetic longitude of point (radians).
- **lat** (*float*) – Geodetic latitude of point (radians).
- **alt** (*float*) – Altitude of point above the reference spheroid.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.drdlat` (*r, lon, lat*)

Compute the Jacobian of the transformation from latitudinal to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/drddlat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/drddlat_c.html)

**Parameters**

- **r** (*float*) – Distance of a point from the origin.
- **lon** (*float*) – Angle of the point from the XZ plane in radians.
- **lat** (*float*) – Angle of the point from the XY plane in radians.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.drdpgr` (*body, lon, lat, alt, re, f*)

This routine computes the Jacobian matrix of the transformation from planetographic to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/drddpgr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/drddpgr_c.html)

**Parameters**

- **body** (*str*) – Body with which coordinate system is associated.
- **lon** (*float*) – Planetographic longitude of a point (radians).
- **lat** (*float*) – Planetographic latitude of a point (radians).
- **alt** (*float*) – Altitude of a point above reference spheroid.

- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.drdsph(r, colat, lon)`

This routine computes the Jacobian of the transformation from spherical to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/drdsph\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/drdsph_c.html)

**Parameters**

- **r** (*float*) – Distance of a point from the origin.
- **colat** (*float*) – Angle of the point from the positive z-axis.
- **lon** (*float*) – Angle of the point from the xy plane.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.dskb02(handle, dladsc)`

Return bookkeeping data from a DSK type 2 segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskb02\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskb02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor

**Returns** bookkeeping data from a DSK type 2 segment

**Return type** tuple

`spiceypy.spiceypy.dskcls(handle, optimiz=False)`

Close a DSK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskcls\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskcls_c.html)

**Parameters**

- **handle** (*int*) – Handle assigned to the opened DSK file.
- **optimiz** (*bool*) – Flag indicating whether to segregate the DSK.

**Returns**

`spiceypy.spiceypy.dskd02(handle, dladsc, item, start, room)`

Fetch double precision data from a type 2 DSK segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskd02\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskd02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor
- **item** (*int*) – Keyword identifying item to fetch
- **start** (*int*) – Start index

- **room** (*int*) – Amount of room in output array

**Returns** Array containing requested item

**Return type** `numpy.ndarray`

`spiceypy.spiceypy.dskgd` (*handle, dladsc*)

Return the DSK descriptor from a DSK segment identified by a DAS handle and DLA descriptor.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskgd\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskgd_c.html)

**Parameters**

- **handle** (*int*) – Handle assigned to the opened DSK file.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA segment descriptor.

**Returns** DSK segment descriptor.

**Return type** `stypes.SpiceDSKDescr`

`spiceypy.spiceypy.dskgt1` (*keywrđ*)

Retrieve the value of a specified DSK tolerance or margin parameter.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskgt1\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskgt1_c.html)

**Parameters** **keywrđ** (*int*) – Code specifying parameter to retrieve.

**Returns** Value of parameter.

**Return type** `float`

`spiceypy.spiceypy.dski02` (*handle, dladsc, item, start, room*)

Fetch integer data from a type 2 DSK segment.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dski02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dski02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor.
- **item** (*int*) – Keyword identifying item to fetch.
- **start** (*int*) – Start index.
- **room** (*int*) – Amount of room in output array.

**Returns** Array containing requested item.

**Return type** `array`

`spiceypy.spiceypy.dskmi2` (*vrtces, plates, finscl, corscl, worksz, voxpsz, voxlsz, makvtl, spxisz*)

Make spatial index for a DSK type 2 segment. The index is returned as a pair of arrays, one of type `int` and one of type `float`. These arrays are suitable for use with the DSK type 2 writer `dskw02`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskmi2\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskmi2_c.html)

**Parameters**

- **vrtces** (*NxM-Element Array of floats*) – Vertices
- **plates** (*NxM-Element Array of ints*) – Plates
- **finscl** (*float*) – Fine voxel scale

- **corscl** (*int*) – Coarse voxel scale
- **worksz** (*int*) – Workspace size
- **voxpsz** (*int*) – Voxel plate pointer array size
- **voxlsz** (*int*) – Voxel plate list array size
- **makvtl** (*bool*) – Vertex plate list flag
- **spxisz** (*int*) – Spatial index integer component size

**Returns** double precision and integer components of the spatial index of the segment.

**Return type** tuple

`spiceypy.spiceypy.dskn02` (*handle, dladsc, plid*)

Compute the unit normal vector for a specified plate from a type 2 DSK segment.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskn02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskn02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor.
- **plid** (*int*) – Plate ID.

**Returns** plate's unit normal vector.

**Return type** 3-Element Array of floats.

`spiceypy.spiceypy.dskobj` (*dsk*)

Find the set of body ID codes of all objects for which topographic data are provided in a specified DSK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskobj\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskobj_c.html)

**Parameters** **dsk** (*str*) – Name of DSK file.

**Returns** Set of ID codes of objects in DSK file.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.dskopn` (*fname, ifname, ncomch*)

Open a new DSK file for subsequent write operations.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskopn\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskopn_c.html)

**Parameters**

- **fname** (*str*) – Name of a DSK file to be opened.
- **ifname** (*str*) – Internal file name.
- **ncomch** (*int*) – Number of comment characters to allocate.

**Returns** Handle assigned to the opened DSK file.

**Return type** int

`spiceypy.spiceypy.dskp02` (*handle, dladsc, start, room*)

Fetch triangular plates from a type 2 DSK segment.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskp02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskp02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle.

- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor.
- **start** (*int*) – Start index.
- **room** (*int*) – Amount of room in output array.

**Returns** Array containing plates.

`spiceypy.spiceypy.dskrb2` (*vrtnes, plates, corsys, corpar*)

Determine range bounds for a set of triangular plates to be stored in a type 2 DSK segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskrb2\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskrb2_c.html)

**Parameters**

- **vrtnes** (*NxM-Element Array of floats*) – Vertices
- **plates** (*NxM-Element Array of ints*) – Plates
- **corsys** (*int*) – DSK coordinate system code
- **corpar** (*N-Element Array of floats*) – DSK coordinate system parameters

**Returns** Lower and Upper bound on range of third coordinate

**Return type** tuple

`spiceypy.spiceypy.dsksrfr` (*dsk, bodyid*)

Find the set of surface ID codes for all surfaces associated with a given body in a specified DSK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dsksrfr\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dsksrfr_c.html)

**Parameters**

- **dsk** (*str*) – Name of DSK file.
- **bodyid** (*int*) – Integer body ID code.

**Returns** Set of ID codes of surfaces in DSK file.

`spiceypy.spiceypy.dskstl` (*keywrdr, dpval*)

Set the value of a specified DSK tolerance or margin parameter.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskstl\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskstl_c.html)

**Parameters**

- **keywrdr** (*int*) – Code specifying parameter to set.
- **dpval** (*float*) – Value of parameter.

**Returns**

`spiceypy.spiceypy.dskv02` (*handle, dladsc, start, room*)

Fetch vertices from a type 2 DSK segment.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskv02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskv02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor.
- **start** (*int*) – Start index.
- **room** (*int*) – Amount of room in output array.

**Returns** Array containing vertices.

**Return type** Room x 3-Element Array of floats

`spiceypy.spiceypy.dskw02` (*handle, center, surfid, dclass, fname, corsys, corpar, mncor1, mxcor1, mncor2, mxcor2, mncor3, mxcor3, first, last, vrtces, plates, spaixd, spaixi*)  
Write a type 2 segment to a DSK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskw02\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskw02_c.html)

#### Parameters

- **handle** (*int*) – Handle assigned to the opened DSK file
- **center** (*int*) – Central body ID code
- **surfid** (*int*) – Surface ID code
- **dclass** (*int*) – Data class
- **fname** (*str*) – Reference frame
- **corsys** (*int*) – Coordinate system code
- **corpar** (*N-Element Array of floats*) – Coordinate system parameters
- **mncor1** (*float*) – Minimum value of first coordinate
- **mxcor1** (*float*) – Maximum value of first coordinate
- **mncor2** (*float*) – Minimum value of second coordinate
- **mxcor2** (*float*) – Maximum value of second coordinate
- **mncor3** (*float*) – Minimum value of third coordinate
- **mxcor3** (*float*) – Maximum value of third coordinate
- **first** (*float*) – Coverage start time
- **last** (*float*) – Coverage stop time
- **vrtces** (*NxM-Element Array of floats*) – Vertices
- **plates** (*NxM-Element Array of ints*) – Plates
- **spaixd** (*N-Element Array of floats*) – Double precision component of spatial index
- **spaixi** (*N-Element Array of ints*) – Integer component of spatial index

`spiceypy.spiceypy.dskx02` (*handle, dladsc, vertex, raydir*)

Determine the plate ID and body-fixed coordinates of the intersection of a specified ray with the surface defined by a type 2 DSK plate model.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskx02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskx02_c.html)

#### Parameters

- **handle** (*int*) – Handle of DSK kernel containing plate model.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor of plate model segment.
- **vertex** (*3-Element Array of floats*) – Ray’s vertex in the body fixed frame.
- **raydir** (*3-Element Array of floats*) – Ray direction in the body fixed frame.

**Returns** ID code of the plate intersected by the ray, Intercept, and Flag indicating whether intercept exists.

**Return type** tuple

`spiceypy.spiceypy.dskxsi` (*pri, target, srflst, et, fixref, vertex, raydir*)

Compute a ray-surface intercept using data provided by multiple loaded DSK segments. Return information about the source of the data defining the surface on which the intercept was found: DSK handle, DLA and DSK descriptors, and DSK data type-dependent parameters.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskxsi\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskxsi_c.html)

**Parameters**

- **pri** (*bool*) – Data prioritization flag.
- **target** (*str*) – Target body name.
- **srflst** (*list of str*) – Surface ID list.
- **et** (*float*) – Epoch, expressed as seconds past J2000 TDB.
- **fixref** (*str*) – Name of target body-fixed reference frame.
- **vertex** (*3-Element Array of floats*) – Vertex of ray.
- **raydir** (*3-Element Array of floats*) – Direction vector of ray.

**Returns** Intercept point, Handle of segment contributing surface data, DLADSC, DSKDSC, Double precision component of source info, Integer component of source info

**Return type** tuple

`spiceypy.spiceypy.dskxv` (*pri, target, srflst, et, fixref, vtxarr, dirarr*)

Compute ray-surface intercepts for a set of rays, using data provided by multiple loaded DSK segments.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskxv\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskxv_c.html)

**Parameters**

- **pri** (*bool*) – Data prioritization flag.
- **target** (*str*) – Target body name.
- **srflst** (*list of str*) – Surface ID list.
- **et** (*float*) – Epoch, expressed as seconds past J2000 TDB.
- **fixref** (*str*) – Name of target body-fixed reference frame.
- **vtxarr** (*Nx3-Element Array of floats*) – Array of vertices of rays.
- **dirarr** (*Nx3-Element Array of floats*) – Array of direction vectors of rays.

**Returns** Intercept point array and Found flag array.

**Return type** tuple

`spiceypy.spiceypy.dskz02` (*handle, dladsc*)

Return plate model size parameters—plate count and vertex count—for a type 2 DSK segment.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dskz02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dskz02_c.html)

**Parameters**

- **handle** (*int*) – DSK file handle.
- **dladsc** (`spiceypy.utils.support_types.SpiceDLADescr`) – DLA descriptor.

**Returns** Number of vertices, Number of plates.



**Return type** tuple

`spiceypy.spiceypy.dsphdr(x, y, z)`

This routine computes the Jacobian of the transformation from rectangular to spherical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dsphdr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dsphdr_c.html)

**Parameters**

- **x** (*float*) – X-coordinate of point.
- **y** (*float*) – Y-coordinate of point.
- **z** (*float*) – Z-coordinate of point.

**Returns** Matrix of partial derivatives.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.dtpool(name)`

Return the data about a kernel pool variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dtpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dtpool_c.html)

**Parameters** **name** (*str*) – Name of the variable whose value is to be returned.

**Returns** Number of values returned for name, Type of the variable “C”, “N”, or “X”.

**Return type** tuple

`spiceypy.spiceypy.ducrass(s1, s2)`

Compute the unit vector parallel to the cross product of two 3-dimensional vectors and the derivative of this unit vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ducrass\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ducrass_c.html)

**Parameters**

- **s1** (*6-Element Array of floats*) – Left hand state for cross product and derivative.
- **s2** (*6-Element Array of floats*) – Right hand state for cross product and derivative.

**Returns** Unit vector and derivative of the cross product.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.dvcrass(s1, s2)`

Compute the cross product of two 3-dimensional vectors and the derivative of this cross product.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvcrass\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvcrass_c.html)

**Parameters**

- **s1** (*6-Element Array of floats*) – Left hand state for cross product and derivative.
- **s2** (*6-Element Array of floats*) – Right hand state for cross product and derivative.

**Returns** State associated with cross product of positions.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.dvdot(s1, s2)`

Compute the derivative of the dot product of two double precision position vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvdot\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvdot_c.html)

**Parameters**

- **s1** (*6-Element Array of floats*) – First state vector in the dot product.
- **s2** (*6-Element Array of floats*) – Second state vector in the dot product.

**Returns** The derivative of the dot product.

**Return type** float

`spiceypy.spiceypy.dvhat(s1)`

Find the unit vector corresponding to a state vector and the derivative of the unit vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvhat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvhat_c.html)

**Parameters** **s1** (*6-Element Array of floats*) – State to be normalized.

**Returns** Unit vector  $s1 / \text{abs}(s1)$ , and its time derivative.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.dvnorm(state)`

Function to calculate the derivative of the norm of a 3-vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvnorm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvnorm_c.html)

**Parameters** **state** (*6-Element Array of floats*) – A 6-vector composed of three coordinates and their derivatives.

**Returns** The derivative of the norm of a 3-vector.

**Return type** float

`spiceypy.spiceypy.dvpool(name)`

Delete a variable from the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvpool_c.html)

**Parameters** **name** (*str*) – Name of the kernel variable to be deleted.

`spiceypy.spiceypy.dvsep(s1, s2)`

Calculate the time derivative of the separation angle between two input states, S1 and S2.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/dvsep\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/dvsep_c.html)

**Parameters**

- **s1** (*6-Element Array of floats*) – State vector of the first body.
- **s2** (*6-Element Array of floats*) – State vector of the second body.

**Returns** The time derivative of the angular separation between S1 and S2.

**Return type** float

`spiceypy.spiceypy.edlimb(a, b, c, viewpt)`

Find the limb of a triaxial ellipsoid, viewed from a specified point.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/edlimb\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/edlimb_c.html)

**Parameters**

- **a** (*float*) – Length of ellipsoid semi-axis lying on the x-axis.
- **b** (*float*) – Length of ellipsoid semi-axis lying on the y-axis.
- **c** (*float*) – Length of ellipsoid semi-axis lying on the z-axis.

- **viewpt** (*3-Element Array of floats*) – Location of viewing point.

**Returns** Limb of ellipsoid as seen from viewing point.

**Return type** *spiceypy.utils.support\_types.Ellipse*

`spiceypy.spiceypy.edterm` (*trmtyp, source, target, et, fixref, abcorr, obsrvr, npts*)

Compute a set of points on the umbral or penumbral terminator of a specified target body, where the target shape is modeled as an ellipsoid.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/edterm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/edterm_c.html)

#### Parameters

- **trmtyp** (*str*) – Terminator type.
- **source** (*str*) – Light source.
- **target** (*str*) – Target body.
- **et** (*str*) – Observation epoch.
- **fixref** (*str*) – Body-fixed frame associated with target.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Observer.
- **npts** (*int*) – Number of points in terminator set.

**Returns** Epoch associated with target center, Position of observer in body-fixed frame, Terminator point set.

**Return type** tuple

`spiceypy.spiceypy.ekacec` (*handle, segno, recno, column, nvals, cvals, isnull*)

Add data to a character column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekacec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekacec_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be added.
- **column** (*str*) – Column name.
- **nvals** (*int*) – Number of values to add to column.
- **cvals** (*list of str.*) – Character values to add to column.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekaced` (*handle, segno, recno, column, nvals, dvals, isnull*)

Add data to an double precision column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekaced\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekaced_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be added.
- **column** (*str*) – Column name.

- **nvals** (*int*) – Number of values to add to column.
- **dvals** (*Array of floats*) – Double precision values to add to column.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekacei` (*handle, segno, recno, column, nvals, ival, isnull*)  
Add data to an integer column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekacei\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekacei_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be added.
- **column** (*str*) – Column name.
- **nvals** (*int*) – Number of values to add to column.
- **ival** (*Array of ints*) – Integer values to add to column.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekac1c` (*handle, segno, column, vallen, cvals, entszs, nlf1gs, rcptrs, wkindx*)  
Add an entire character column to an EK segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekac1c\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekac1c_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Number of segment to add column to.
- **column** (*str*) – Column name.
- **vallen** (*int*) – Length of character values.
- **cvals** (*list of str.*) – Character values to add to column.
- **entszs** (*Array of ints*) – Array of sizes of column entries.
- **nlf1gs** (*Array of bools*) – Array of null flags for column entries.
- **rcptrs** (*Array of ints*) – Record pointers for segment.
- **wkindx** (*Array of ints*) – Work space for column index.

**Returns** Work space for column index.

**Return type** Array of ints

`spiceypy.spiceypy.ekac1d` (*handle, segno, column, dvals, entszs, nlf1gs, rcptrs, wkindx*)  
Add an entire double precision column to an EK segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekac1d\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekac1d_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Number of segment to add column to.
- **column** (*str*) – Column name.
- **dvals** (*Array of floats*) – Double precision values to add to column.

- **entszs** (*Array of ints*) – Array of sizes of column entries.
- **nlflgs** (*Array of bools*) – Array of null flags for column entries.
- **rcptrs** (*Array of ints*) – Record pointers for segment.
- **wkindx** (*Array of ints*) – Work space for column index.

**Returns** Work space for column index.

**Return type** Array of ints

`spiceypy.spiceypy.ekaccli` (*handle, segno, column, ival, entszs, nlflgs, rcptrs, wkindx*)  
Add an entire integer column to an EK segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekaccli\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekaccli_c.html)

#### Parameters

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Number of segment to add column to.
- **column** (*str*) – Column name.
- **ivals** (*Array of ints*) – Integer values to add to column.
- **nlflgs** (*Array of bools*) – Array of null flags for column entries.
- **rcptrs** (*Array of ints*) – Record pointers for segment.
- **wkindx** (*Array of ints*) – Work space for column index.

**Returns** Work space for column index.

**Return type** Array of ints

`spiceypy.spiceypy.ekappr` (*handle, segno*)  
Append a new, empty record at the end of a specified E-kernel segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekappr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekappr_c.html)

#### Parameters

- **handle** (*int*) – File handle.
- **segno** (*int*) – Segment number.

**Returns** Number of appended record.

**Return type** int

`spiceypy.spiceypy.ekbseg` (*handle, tabnam, cnames, decls*)  
Start a new segment in an E-kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekbseg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekbseg_c.html)

#### Parameters

- **handle** (*int*) – File handle.
- **tabnam** (*str*) – Table name.
- **cnames** (*list of str.*) – Names of columns.
- **decls** (*list of str.*) – Declarations of columns.

**Returns** Segment number.

**Return type** int

`spiceypy.spiceypy.ekccnt` (*table*)

Return the number of distinct columns in a specified, currently loaded table.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekccnt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekccnt_c.html)

**Parameters** `table` (*str*) – Name of table.

**Returns** Count of distinct, currently loaded columns.

**Return type** `int`

`spiceypy.spiceypy.ekcii` (*table, index, lenout=256*)

Return attribute information about a column belonging to a loaded EK table, specifying the column by table and index.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekcii\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekcii_c.html)

**Parameters**

- **table** (*str*) – Name of table containing column.
- **index** (*int*) – Index of column whose attributes are to be found.
- **lenout** – Maximum allowed length of column name.

**Returns** Name of column, Column attribute descriptor.

**Return type** `tuple`

`spiceypy.spiceypy.ekcls` (*handle*)

Close an E-kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekcls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekcls_c.html)

**Parameters** `handle` (*int*) – EK file handle.

`spiceypy.spiceypy.ekdelr` (*handle, segno, recno*)

Delete a specified record from a specified E-kernel segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekdelr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekdelr_c.html)

**Parameters**

- **handle** (*int*) – File handle.
- **segno** (*int*) – Segment number.
- **recno** (*int*) – Record number.

`spiceypy.spiceypy.ekffld` (*handle, segno, rcptrs*)

Complete a fast write operation on a new E-kernel segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekffld\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekffld_c.html)

**Parameters**

- **handle** (*int*) – File handle.
- **segno** (*int*) – Segment number.
- **rcptrs** (*Array of ints*) – Record pointers.

`spiceypy.spiceypy.ekfind` (*query, lenout=256*)

Find E-kernel data that satisfy a set of constraints.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekfind\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekfind_c.html)

**Parameters**

- **query** (*str*) – Query specifying data to be found.
- **lenout** (*int*) – Declared length of output error message string.

**Returns** Number of matching rows, Flag indicating whether query parsed correctly, Parse error description.

**Return type** tuple

`spiceypy.spiceypy.ekgc` (*selidx, row, element, lenout=256*)

Return an element of an entry in a column of character type in a specified row.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekgc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekgc_c.html)

**Parameters**

- **selidx** (*int*) – Index of parent column in SELECT clause.
- **row** (*int*) – Row to fetch from.
- **element** (*int*) – Index of element, within column entry, to fetch.
- **lenout** (*int*) – Maximum length of column element.

**Returns** Character string element of column entry, Flag indicating whether column entry was null.

**Return type** tuple

`spiceypy.spiceypy.ekgd` (*selidx, row, element*)

Return an element of an entry in a column of double precision type in a specified row.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekgd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekgd_c.html)

**Parameters**

- **selidx** (*int*) – Index of parent column in SELECT clause.
- **row** (*int*) – Row to fetch from.
- **element** (*int*) – Index of element, within column entry, to fetch.

**Returns** Double precision element of column entry, Flag indicating whether column entry was null.

**Return type** tuple

`spiceypy.spiceypy.ekgi` (*selidx, row, element*)

Return an element of an entry in a column of integer type in a specified row.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekgi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekgi_c.html)

**Parameters**

- **selidx** (*int*) – Index of parent column in SELECT clause.
- **row** (*int*) – Row to fetch from.
- **element** (*int*) – Index of element, within column entry, to fetch.

**Returns** Integer element of column entry, Flag indicating whether column entry was null.

**Return type** tuple

`spiceypy.spiceypy.ekifld` (*handle, tabnam, ncols, nrows, cnmlen, cnames, declen, decls*)

Initialize a new E-kernel segment to allow fast writing.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/ekifld\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/ekifld_c.html)

**Parameters**

- **handle** (*int*) – File handle.

- **tabnam** (*str*) – Table name.
- **ncols** (*int*) – Number of columns in the segment.
- **nrows** (*int*) – Number of rows in the segment.
- **cnmlen** (*int*) – Length of names in in column name array.
- **cnames** (*list of str.*) – Names of columns.
- **declen** (*int*) – Length of declaration strings in declaration array.
- **decls** (*list of str.*) – Declarations of columns.

**Returns** Segment number, Array of record pointers.

**Return type** tuple

`spiceypy.spiceypy.ekinsr` (*handle, segno, recno*)

Add a new, empty record to a specified E-kernel segment at a specified index.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekinsr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekinsr_c.html)

**Parameters**

- **handle** (*int*) – File handle.
- **segno** (*int*) – Segment number.
- **recno** (*int*) – Record number.

`spiceypy.spiceypy.eklef` (*fname*)

Load an EK file, making it accessible to the EK readers.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eklef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eklef_c.html)

**Parameters** **fname** (*str*) – Name of EK file to load.

**Returns** File handle of loaded EK file.

**Return type** int

`spiceypy.spiceypy.eknelt` (*selidx, row*)

Return the number of elements in a specified column entry in the current row.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eknelt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eknelt_c.html)

**Parameters**

- **selidx** (*int*) – Index of parent column in SELECT clause.
- **row** (*int*) – Row containing element.

**Returns** The number of elements in entry in current row.

**Return type** int

`spiceypy.spiceypy.eknseg` (*handle*)

Return the number of segments in a specified EK.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eknseg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eknseg_c.html)

**Parameters** **handle** (*int*) – EK file handle.

**Returns** The number of segments in the specified E-kernel.

**Return type** int



`spiceypy.spiceypy.ekntab()`

Return the number of loaded EK tables.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekntab\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekntab_c.html)

**Returns** The number of loaded EK tables.

**Return type** int

`spiceypy.spiceypy.ekopn(fname, ifname, ncomch)`

Open a new E-kernel file and prepare the file for writing.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekopn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekopn_c.html)

**Parameters**

- **fname** (*str*) – Name of EK file.
- **ifname** (*str*) – Internal file name.
- **ncomch** (*int*) – The number of characters to reserve for comments.

**Returns** Handle attached to new EK file.

**Return type** int

`spiceypy.spiceypy.ekopr(fname)`

Open an existing E-kernel file for reading.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekopr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekopr_c.html)

**Parameters** **fname** (*str*) – Name of EK file.

**Returns** Handle attached to EK file.

**Return type** int

`spiceypy.spiceypy.ekops()`

Open a scratch (temporary) E-kernel file and prepare the file for writing.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekops\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekops_c.html)

**Returns** Handle attached to new EK file.

**Return type** int

`spiceypy.spiceypy.ekopw(fname)`

Open an existing E-kernel file for writing.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekopw\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekopw_c.html)

**Parameters** **fname** (*str*) – Name of EK file.

**Returns** Handle attached to EK file.

**Return type** int

`spiceypy.spiceypy.ekpsel(query, msglen, tablen, collen)`

Parse the SELECT clause of an EK query, returning full particulars concerning each selected item.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekpsel\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekpsel_c.html) note: oddly docs at url are incomplete/incorrect.

**Parameters**

- **query** (*str*) – EK query.
- **msglen** (*int*) – Available space in the output error message string.

- **tablen** (*int*) – UNKNOWN? Length of Table?
- **collen** – UNKNOWN? Length of Column?

**Returns** Number of items in SELECT clause of query, Begin positions of expressions in SELECT clause, End positions of expressions in SELECT clause, Data types of expressions, Classes of expressions, Names of tables qualifying SELECT columns, Names of columns in SELECT clause of query, Error flag, Parse error message.

**Return type** tuple

`spiceypy.spiceypy.ekrcec` (*handle, segno, recno, column, lenout, nelts=100*)  
Read data from a character column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekrcec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekrcec_c.html)

**Parameters**

- **handle** (*int*) – Handle attached to EK file.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record from which data is to be read.
- **column** (*str*) – Column name.
- **lenout** (*int*) – Maximum length of output strings.
- **nelts** (*int*) – Number of elements to allow for (default=100)

**Returns** Number of values in column entry, Character values in column entry, Flag indicating whether column entry is null.

**Return type** tuple

`spiceypy.spiceypy.ekrccd` (*handle, segno, recno, column, nelts=100*)  
Read data from a double precision column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekrccd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekrccd_c.html)

**Parameters**

- **handle** (*int*) – Handle attached to EK file.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record from which data is to be read.
- **column** (*str*) – Column name.

**Returns** Number of values in column entry, Float values in column entry, Flag indicating whether column entry is null.

**Return type** tuple

`spiceypy.spiceypy.ekrcei` (*handle, segno, recno, column, nelts=100*)  
Read data from an integer column in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekrcei\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekrcei_c.html)

**Parameters**

- **handle** (*int*) – Handle attached to EK file.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record from which data is to be read.
- **column** (*str*) – Column name.

**Returns** Number of values in column entry, Integer values in column entry, Flag indicating whether column entry is null.

**Return type** tuple

`spiceypy.spiceypy.ekssum(handle, segno)`

Return summary information for a specified segment in a specified EK.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekssum\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekssum_c.html)

**Parameters**

- **handle** (*int*) – Handle of EK.
- **segno** (*int*) – Number of segment to be summarized.

**Returns** EK segment summary.

**Return type** `spiceypy.utils.support_types.SpiceEKSegSum`

`spiceypy.spiceypy.ektnam(n, lenout=256)`

Return the name of a specified, loaded table.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ektnam\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ektnam_c.html)

**Parameters**

- **n** (*int*) – Index of table.
- **lenout** (*int*) – Maximum table name length.

**Returns** Name of table.

**Return type** str

`spiceypy.spiceypy.ekucec(handle, segno, recno, column, nvals, cvals, isnull)`

Update a character column entry in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekucec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekucec_c.html)

**Parameters**

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be updated.
- **column** (*str*) – Column name.
- **nvals** (*int*) – Number of values in new column entry.
- **cvals** (*list of str.*) – Character values comprising new column entry.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekuced(handle, segno, recno, column, nvals, dvals, isnull)`

Update a double precision column entry in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekuced\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekuced_c.html)

**Parameters**

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be updated.
- **column** (*str*) – Column name.

- **nvals** (*int*) – Number of values in new column entry.
- **dvals** (*Array of floats*) – Double precision values comprising new column entry.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekupei` (*handle, segno, recno, column, nvals, ival, isnull*)  
Update an integer column entry in a specified EK record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekupei\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekupei_c.html)

**Parameters**

- **handle** (*int*) – EK file handle.
- **segno** (*int*) – Index of segment containing record.
- **recno** (*int*) – Record to which data is to be updated.
- **column** (*str*) – Column name.
- **nvals** (*int*) – Number of values in new column entry.
- **ival** (*Array of ints*) – Integer values comprising new column entry.
- **isnull** (*bool*) – Flag indicating whether column entry is null.

`spiceypy.spiceypy.ekuef` (*handle*)

Unload an EK file, making its contents inaccessible to the EK reader routines, and clearing space in order to allow other EK files to be loaded.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ekuef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ekuef_c.html)

**Parameters** **handle** (*int*) – Handle of EK file.

`spiceypy.spiceypy.el2cgv` (*ellipse*)

Convert an ellipse to a center vector and two generating vectors. The selected generating vectors are semi-axes of the ellipse.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/el2cgv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/el2cgv_c.html)

**Parameters** **ellipse** (`spiceypy.utils.support_types.Ellipse`) – An Ellipse

**Returns** Center and semi-axes of ellipse.

**Return type** tuple

`spiceypy.spiceypy.elemc` (*item, inset*)

Determine whether an item is an element of a character set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/elemc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/elemc_c.html)

**Parameters**

- **item** (*str*) – Item to be tested.
- **inset** (`spiceypy.utils.support_types.SpiceCell`) – Set to be tested.

**Returns** True if item is an element of set.

**Return type** *bool*

`spiceypy.spiceypy.elemd` (*item, inset*)

Determine whether an item is an element of a double precision set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/elemd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/elemd_c.html)

**Parameters**

- **item** (*float*) – Item to be tested.
- **inset** (`spiceypy.utils.support_types.SpiceCell`) – Set to be tested.

**Returns** True if item is an element of set.

**Return type** *bool*

`spiceypy.spiceypy.elemi` (*item, inset*)

Determine whether an item is an element of an integer set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eleme\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eleme_c.html)

**Parameters**

- **item** (*int*) – Item to be tested.
- **inset** (`spiceypy.utils.support_types.SpiceCell`) – Set to be tested.

**Returns** True if item is an element of set.

**Return type** *bool*

`spiceypy.spiceypy.eqncpv` (*et, epoch, eqel, rapol, decpol*)

Compute the state (position and velocity of an object whose trajectory is described via equinoctial elements relative to some fixed plane (usually the equatorial plane of some planet).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eqncpv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eqncpv_c.html)

**Parameters**

- **et** (*float*) – Epoch in seconds past J2000 to find state.
- **epoch** (*float*) – Epoch of elements in seconds past J2000.
- **eqel** (*9-Element Array of floats*) – Array of equinoctial elements
- **rapol** (*float*) – Right Ascension of the pole of the reference plane.
- **decpol** (*float*) – Declination of the pole of the reference plane.

**Returns** State of the object described by eqel.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.eqstr` (*a, b*)

Determine whether two strings are equivalent.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eqstr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eqstr_c.html)

**Parameters**

- **a** (*str*) – Arbitrary character string.
- **b** (*str*) – Arbitrary character string.

**Returns** True if A and B are equivalent.

**Return type** *bool*

`spiceypy.spiceypy.erract` (*op, lenout, action=None*)

Retrieve or set the default error action. spiceypy sets the default error action to “report” on init.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/erract\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/erract_c.html)

**Parameters**

- **op** (*str*) – operation, “GET” or “SET”.
- **lenout** (*int*) – Length of list for output.

- **action** (*str*) – Error response action.

**Returns** Error response action.

**Return type** *str*

`spiceypy.spiceypy.errch` (*marker, string*)

Substitute a character string for the first occurrence of a marker in the current long error message.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/errch\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/errch_c.html)

**Parameters**

- **marker** (*str*) – A substring of the error message to be replaced.
- **string** (*str*) – The character string to substitute for marker.

`spiceypy.spiceypy.errdev` (*op, lenout, device*)

Retrieve or set the name of the current output device for error messages.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/errdev\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/errdev_c.html)

**Parameters**

- **op** (*str*) – The operation, “GET” or “SET”.
- **lenout** (*int*) – Length of device for output.
- **device** (*str*) – The device name.

**Returns** The device name.

**Return type** *str*

`spiceypy.spiceypy.errdp` (*marker, number*)

Substitute a double precision number for the first occurrence of a marker found in the current long error message.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/errdp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/errdp_c.html)

**Parameters**

- **marker** (*str*) – A substring of the error message to be replaced.
- **number** (*float*) – The d.p. number to substitute for marker.

`spiceypy.spiceypy.errint` (*marker, number*)

Substitute an integer for the first occurrence of a marker found in the current long error message.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/errint\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/errint_c.html)

**Parameters**

- **marker** (*str*) – A substring of the error message to be replaced.
- **number** (*int*) – The integer to substitute for marker.

`spiceypy.spiceypy.errprt` (*op, lenout, inlist*)

Retrieve or set the list of error message items to be output when an error is detected.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/errprt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/errprt_c.html)

**Parameters**

- **op** (*str*) – The operation, “GET” or “SET”.
- **lenout** (*int*) – Length of list for output.
- **inlist** (*list of str.*) – Specification of error messages to be output.

**Returns** A list of error message items.

**Return type** list of str.

`spiceypy.spiceypy.esrchc` (*value*, *array*)

Search for a given value within a character string array. Return the index of the first equivalent array entry, or -1 if no equivalent element is found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/esrchc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/esrchc_c.html)

**Parameters**

- **value** (*str*) – Key value to be found in array.
- **array** (*list of str.*) – Character string array to search.

**Returns** The index of the first array entry equivalent to value, or -1 if none is found.

**Return type** int

`spiceypy.spiceypy.et21st` (*et*, *body*, *lon*, *typein*, *timlen=256*, *ampmlen=256*)

Given an ephemeris epoch, compute the local solar time for an object on the surface of a body at a specified longitude.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/et21st\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/et21st_c.html)

**Parameters**

- **et** (*float*) – Epoch in seconds past J2000 epoch.
- **body** (*int*) – ID-code of the body of interest.
- **lon** (*float*) – Longitude of surface point (RADIANS).
- **typein** (*str*) – Type of longitude “PLANETOCENTRIC”, etc.
- **timlen** (*int*) – Available room in output time string.
- **ampmlen** (*int*) – Available room in output amp string.

**Returns** Local hour on a “24 hour” clock, Minutes past the hour, Seconds past the minute, String giving local time on 24 hour clock, String giving time on A.M. / P.M. scale.

**Return type** tuple

`spiceypy.spiceypy.et2utc` (*et*, *formatStr*, *prec*, *lenout=256*)

Convert an input time from ephemeris seconds past J2000 to Calendar, Day-of-Year, or Julian Date format, UTC.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/et2utc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/et2utc_c.html)

**Parameters**

- **et** (*float*) – Input epoch, given in ephemeris seconds past J2000.
- **formatStr** (*str*) – Format of output epoch.
- **prec** (*int*) – Digits of precision in fractional seconds or days.
- **lenout** (*int*) – The length of the output string plus 1.

**Returns** Output time string in UTC

**Return type** str

`spiceypy.spiceypy.etcalf` (*et*, *lenout=256*)

Convert from an ephemeris epoch measured in seconds past the epoch of J2000 to a calendar string format using a formal calendar free of leapseconds.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/etcalf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/etcalf_c.html)

**Parameters**

- **et** (*Union[float, Iterable[float]]*) – Ephemeris time measured in seconds past J2000.
- **lenout** (*int*) – Length of output string.

**Returns** A standard calendar representation of et.

**Return type** *str*

`spiceypy.spiceypy.eul2m(angle3, angle2, angle1, axis3, axis2, axis1)`

Construct a rotation matrix from a set of Euler angles.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eul2m\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eul2m_c.html)

**Parameters**

- **angle3** (*float*) – Rotation angle about third rotation axis (radians).
- **angle2** (*float*) – Rotation angle about second rotation axis (radians).
- **angle1** (*float*) – Rotation angle about first rotation axis (radians).
- **axis3** (*int*) – Axis number of third rotation axis.
- **axis2** (*int*) – Axis number of second rotation axis.
- **axis1** (*int*) – Axis number of first rotation axis.]

**Returns** Product of the 3 rotations.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.eul2xf(eulang, axisa, axisb, axisc)`

This routine computes a state transformation from an Euler angle factorization of a rotation and the derivatives of those Euler angles.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/eul2xf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/eul2xf_c.html)

**Parameters**

- **eulang** (*6-Element Array of floats*) – An array of Euler angles and their derivatives.
- **axisa** (*int*) – Axis A of the Euler angle factorization.
- **axisb** (*int*) – Axis B of the Euler angle factorization.
- **axisc** (*int*) – Axis C of the Euler angle factorization.

**Returns** A state transformation matrix.

**Return type** 6x6-Element Array of floats

`spiceypy.spiceypy.exists(fname)`

Determine whether a file exists.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/exists\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/exists_c.html)

**Parameters** **fname** – Name of the file in question.

**Returns** True if the file exists, False otherwise.

**Return type** *bool*



`spiceypy.spiceypy.exppool(name)`

Confirm the existence of a kernel variable in the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/exppool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/exppool_c.html)

**Parameters** `name` (*str*) – Name of the variable whose value is to be returned.

**Returns** True when the variable is in the pool.

**Return type** *bool*

`spiceypy.spiceypy.failed()`

True if an error condition has been signalled via `sigerr_c`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/failed\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/failed_c.html)

**Returns** a boolean

**Return type** *bool*

`spiceypy.spiceypy.fn21un(fname)`

Internal undocumented command for mapping name of open file to its FORTRAN (F2C) logical unit.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/FORTRAN/spicelib/fn21un.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/FORTRAN/spicelib/fn21un.html)

**Parameters** `fname` (*str*) – name of the file to be mapped to its logical unit.

**Returns** the FORTRAN (F2C) logical unit associated with the filename.

**Return type** *int*

`spiceypy.spiceypy.found_check()`

Temporarily enables spiceypy default behavior which raises exceptions for false found flags for certain spice functions. All spice functions executed within the context manager will check the found flag return parameter and the found flag will be removed from the return for the given function. For Example `bodc2n` in spiceypy is normally called like:

```
name = spice.bodc2n(399)
```

With the possibility that an exception is thrown in the even of a invalid ID:

```
name = spice.bodc2n(-999991) # throws a SpiceyError
```

With this function however, we can use it as a context manager to do this:

```
with spice.found_check():
    found = spice.bodc2n(-999991) # will raise an exception!
```

Within the context any spice functions called that normally check the found flags will pass through the check without raising an exception if they are false.

`spiceypy.spiceypy.found_check_off()`

Method that turns off found catching

`spiceypy.spiceypy.found_check_on()`

Method that turns on found catching

`spiceypy.spiceypy.fovray(inst, raydir, rframe, abcorr, observer, et)`

Determine if a specified ray is within the field-of-view (FOV) of a specified instrument at a given time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/fovray\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/fovray_c.html)

**Parameters**

- `inst` (*str*) – Name or ID code string of the instrument.

- **raydir** (*3-Element Array of floats*) – Ray’s direction vector.
- **rframe** (*str*) – Body-fixed, body-centered frame for target body.
- **abcorr** (*str*) – Aberration correction flag.
- **observer** (*str*) – Name or ID code string of the observer.
- **et** (*float*) – Time of the observation (seconds past J2000).

**Returns** Visibility flag

**Return type** *bool*

`spiceypy.spiceypy.fovtrg` (*inst, target, tshape, tframe, abcorr, observer, et*)

Determine if a specified ephemeris object is within the field-of-view (FOV) of a specified instrument at a given time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/fovtrg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/fovtrg_c.html)

**Parameters**

- **inst** (*str*) – Name or ID code string of the instrument.
- **target** (*str*) – Name or ID code string of the target.
- **tshape** (*str*) – Type of shape model used for the target.
- **tframe** (*str*) – Body-fixed, body-centered frame for target body.
- **abcorr** (*str*) – Aberration correction flag.
- **observer** (*str*) – Name or ID code string of the observer.
- **et** (*float*) – Time of the observation (seconds past J2000).

**Returns** Visibility flag

**Return type** *bool*

`spiceypy.spiceypy.frame` (*x*)

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/frame\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/frame_c.html)

**Parameters** **x** (*3-Element Array of floats*) – Input vector. A parallel unit vector on output.

**Returns** a tuple of 3 list[3]

**Return type** tuple

`spiceypy.spiceypy.frinfo` (*frcode*)

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/frinfo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/frinfo_c.html)

**Parameters** **frcode** (*int*) – the idcode for some frame.

**Returns** a tuple of attributes associated with the frame.

**Return type** tuple

`spiceypy.spiceypy.frmnam` (*frcode, lenout=256*)

Retrieve the name of a reference frame associated with a SPICE ID code.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/frmnam\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/frmnam_c.html)

**Parameters**

- **frcode** (*int*) – an integer code for a reference frame
- **lenout** (*int*) – Maximum length of output string.

**Returns** the name associated with the reference frame.

**Return type** str

`spiceypy.spiceypy.ftncls` (*unit*)

Close a file designated by a Fortran-style integer logical unit.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ftncls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ftncls_c.html)

**Parameters** `unit` (*int*) – Fortran-style logical unit.

`spiceypy.spiceypy.furnsh` (*path*)

Load one or more SPICE kernels into a program.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/furnsh\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/furnsh_c.html)

**Parameters** `path` (*str or list of str*) – one or more paths to kernels

`spiceypy.spiceypy.gcpool` (*name, start, room, lenout=256*)

Return the character value of a kernel variable from the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gcpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gcpool_c.html)

**Parameters**

- **name** (*str*) – Name of the variable whose value is to be returned.
- **start** (*int*) – Which component to start retrieving for name.
- **room** (*int*) – The largest number of values to return.
- **lenout** (*int*) – The length of the output string.

**Returns** Values associated with name.

**Return type** list of str

`spiceypy.spiceypy.gdpool` (*name, start, room*)

Return the d.p. value of a kernel variable from the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gdpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gdpool_c.html)

**Parameters**

- **name** (*str*) – Name of the variable whose value is to be returned.
- **start** (*int*) – Which component to start retrieving for name.
- **room** (*int*) – The largest number of values to return.

**Returns** Values associated with name.

**Return type** list of float

`spiceypy.spiceypy.georec` (*lon, lat, alt, re, f*)

Convert geodetic coordinates to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/georec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/georec_c.html)

**Parameters**

- **lon** (*float*) – Geodetic longitude of point (radians).
- **lat** (*float*) – Geodetic latitude of point (radians).
- **alt** (*float*) – Altitude of point above the reference spheroid.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Rectangular coordinates of point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.get_found_catch_state()`

Returns the current found catch state

**Returns**

`spiceypy.spiceypy.getelm(frstyr, lineIn, lines)`

Given a the “lines” of a two-line element set, parse the lines and return the elements in units suitable for use in SPICE software.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/getelm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/getelm_c.html)

**Parameters**

- **frstyr** (*int*) – Year of earliest representable two-line elements.
- **lineIn** (*int*) – Length of strings in lines array.
- **lines** (*list of str*) – A pair of “lines” containing two-line elements.

**Returns** The epoch of the elements in seconds past J2000, The elements converted to SPICE units.

**Return type** tuple

`spiceypy.spiceypy.getfat(file)`

Determine the file architecture and file type of most SPICE kernel files.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/getfat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/getfat_c.html)

**Parameters** **file** (*str*) – The name of a file to be examined.

**Returns** The architecture of the kernel file, The type of the kernel file.

**Return type** tuple

`spiceypy.spiceypy.getfov(instid, room, shapelen=256, framelen=256)`

This routine returns the field-of-view (FOV) parameters for a specified instrument.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/getfov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/getfov_c.html)

**Parameters**

- **instid** (*int*) – NAIF ID of an instrument.
- **room** (*int*) – Maximum number of vectors that can be returned.
- **shapelen** (*int*) – Space available in the string shape.
- **framelen** (*int*) – Space available in the string frame.

**Returns** Instrument FOV shape, Name of the frame in which FOV vectors are defined, Boresight vector, Number of boundary vectors returned, FOV boundary vectors.

**Return type** tuple

`spiceypy.spiceypy.getmsg(option, lenout=256)`

Retrieve the current short error message, the explanation of the short error message, or the long error message.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/getmsg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/getmsg_c.html)

**Parameters**

- **option** (*str*) – Indicates type of error message.
- **lenout** (*int*) – Available space in the output string msg.

**Returns** The error message to be retrieved.

**Return type** *str*

`spiceypy.spiceypy.gfbail()`

Indicate whether an interrupt signal (SIGINT) has been received.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/gfbail\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/gfbail_c.html)

**Returns** True if an interrupt signal has been received by the GF handler.

**Return type** *bool*

`spiceypy.spiceypy.gfclrh()`

Clear the interrupt signal handler status, so that future calls to `gfbail()` will indicate no interrupt was received.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/gfclrh\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/gfclrh_c.html)

`spiceypy.spiceypy.gfdist(target, abcorr, obsrvr, relate, refval, adjust, step, nintvls, cnfine, result)`

Return the time window over which a specified constraint on observer-target distance is met.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/spice/gfdist\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/spice/gfdist_c.html)

#### Parameters

- **target** (*str*) – Name of the target body.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Adjustment value for absolute extrema searches.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvls** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is confined.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfevnt()`

`spiceypy.spiceypy.gffove()`

`spiceypy.spiceypy.gfilum(method, angtyp, target, illumn, fixref, abcorr, obsrvr, spoint, relate, refval, adjust, step, nintvls, cnfine, result)`

Return the time window over which a specified constraint on the observed phase, solar incidence, or emission angle at a specified target body surface point is met.

#### Parameters

- **method** (*str*) – Shape model used to represent the surface of the target body.
- **angtyp** (*str*) – The type of illumination angle for which a search is to be performed.
- **target** (*str*) – Name of a target body.
- **illumn** (*str*) – Name of the illumination source.
- **fixref** (*str*) – Name of the body-fixed, body-centered reference frame associated with the target body.

- **abcorr** (*str*) – The aberration corrections to be applied.
- **obsrvr** (*str*) – Name of an observing body.
- **spoint** (*3-Element Array of floats*) – Body-fixed coordinates of a target surface point.
- **relate** (*str*) – Relational operator used to define a constraint on a specified illumination angle.
- **refval** (*float*) – Reference value used with ‘relate’ to define an equality or inequality to be satisfied by the specified illumination angle.
- **adjust** (*float*) – Parameter used to modify searches for absolute extrema.
- **step** (*float*) – Step size to be used in the search.
- **nintvls** (*int*) – Number of intervals that can be accommodated by each of the dynamically allocated workspace windows used internally by this routine.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – Window that confines the time period over which the specified search is conducted. This can be updated by `gfilum`
- **result** (`spiceypy.utils.support_types.SpiceCell`) – Window of intervals in the confinement window that the illumination angle constraint is satisfied.

`spiceypy.spiceypy.gfint` (*sigcode*)

Respond to the interrupt signal SIGINT: save an indication that the signal has been received. This routine restores itself as the handler for SIGINT.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfint\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfint_c.html)

**Parameters** `sigcode` (*int*) – Interrupt signal ID code.

`spiceypy.spiceypy.gfocce` ()

`spiceypy.spiceypy.gfoclt` (*occtyp, front, fshape, fframe, back, bshape, bframe, abcorr, obsrvr, step, cnfine, result*)

Determine time intervals when an observer sees one target occulted by, or in transit across, another.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfoclt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfoclt_c.html)

**Parameters**

- **occtyp** (*str*) – Type of occultation.
- **front** (*str*) – Name of body occulting the other.
- **fshape** (*str*) – Type of shape model used for front body.
- **fframe** (*str*) – Body-fixed, body-centered frame for front body.
- **back** (*str*) – Name of body occulted by the other.
- **bshape** (*str*) – Type of shape model used for back body.
- **bframe** (*str*) – Body-fixed, body-centered frame for back body.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **step** (*float*) – Step size in seconds for finding occultation events.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.

- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfpa` (*target, illmin, abcorr, obsrvr, relate, refval, adjust, step, nintvals, cnfine, result*)

Determine time intervals for which a specified constraint on the phase angle between an illumination source, a target, and observer body centers is met.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfpa\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfpa_c.html)

#### Parameters

- **target** (*str*) – Name of the target body.
- **illmin** (*str*) – Name of the illuminating body.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Adjustment value for absolute extrema searches.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfposc` (*target, inframe, abcorr, obsrvr, crdsys, coord, relate, refval, adjust, step, nintvals, cnfine, result*)

Determine time intervals for which a coordinate of an observer-target position vector satisfies a numerical constraint.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfposc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfposc_c.html)

#### Parameters

- **target** (*str*) – Name of the target body.
- **inframe** (*str*) – Name of the reference frame for coordinate calculations.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **crdsys** (*str*) – Name of the coordinate system containing COORD
- **coord** (*str*) – Name of the coordinate of interest
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Adjustment value for absolute extrema searches.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.

- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfrefn(t1, t2, s1, s2)`

For those times when we can't do better, we use a bisection method to find the next time at which to test for state change.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrefn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrefn_c.html)

#### Parameters

- **t1** (*float*) – One of two values bracketing a state change.
- **t2** (*float*) – The other value that brackets a state change.
- **s1** (*bool*) – State at t1.
- **s2** (*bool*) – State at t2.

**Returns** New value at which to check for transition.

**Return type** float

`spiceypy.spiceypy.gfrepf()`

Finish a GF progress report.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrepf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrepf_c.html)

`spiceypy.spiceypy.gfrepf(window, begmss, endmss)`

This entry point initializes a search progress report.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrepf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrepf_c.html)

#### Parameters

- **window** (`spiceypy.utils.support_types.SpiceCell`) – A window over which a job is to be performed.
- **begmss** (*str*) – Beginning of the text portion of the output message.
- **endmss** (*str*) – End of the text portion of the output message.

`spiceypy.spiceypy.gfrepf(ivbeg, ivend, time)`

This function tells the progress reporting system how far a search has progressed.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrepf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrepf_c.html)

#### Parameters

- **ivbeg** (*float*) – Start time of work interval.
- **ivend** (*float*) – End time of work interval.
- **time** (*float*) – Current time being examined in the search process.

`spiceypy.spiceypy.gfrfov(inst, raydir, rframe, abcorr, obsrvr, step, cnfine, result)`

Determine time intervals when a specified ray intersects the space bounded by the field-of-view (FOV) of a specified instrument.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrfov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrfov_c.html)

#### Parameters

- **inst** (*str*) – Name of the instrument.
- **raydir** (*3-Element Array of Float.*) – Ray's direction vector.
- **rframe** (*str*) – Reference frame of ray's direction vector.



- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **step** (*float*) – Step size in seconds for finding FOV events.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfrr` (*target, abcorr, obsrvr, relate, refval, adjust, step, nintvals, cnfine, result*)  
Determine time intervals for which a specified constraint on the observer-target range rate is met.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfrr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfrr_c.html)

#### Parameters

- **target** (*str*) – Name of the target body.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Adjustment value for absolute extrema searches.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfsep` (*targ1, shape1, inframe1, targ2, shape2, inframe2, abcorr, obsrvr, relate, refval, adjust, step, nintvals, cnfine, result*)

Determine time intervals when the angular separation between the position vectors of two target bodies relative to an observer satisfies a numerical relationship.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfsep\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfsep_c.html)

#### Parameters

- **targ1** (*str*) – Name of first body.
- **shape1** (*str*) – Name of shape model describing the first body.
- **inframe1** (*str*) – The body-fixed reference frame of the first body.
- **targ2** (*str*) – Name of second body.
- **shape2** (*str*) – Name of the shape model describing the second body.
- **inframe2** (*str*) – The body-fixed reference frame of the second body.
- **abcorr** (*str*) – Aberration correction flag
- **obsrvr** (*str*) – Name of the observing body.
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.

- **adjust** (*float*) – Absolute extremum adjustment value.
- **step** (*float*) – Step size in seconds for finding angular separation events.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfsntc` (*target, fixref, method, abcorr, obsrvr, dref, dvec, crdsys, coord, relate, refval, adjust, step, nintvals, cnfine, result*)

Determine time intervals for which a coordinate of an surface intercept position vector satisfies a numerical constraint.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfsntc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfsntc_c.html)

#### Parameters

- **target** (*str*) – Name of the target body.
- **fixref** (*str*) – Body fixed frame associated with the target.
- **method** (*str*) – Name of method type for surface intercept calculation.
- **abcorr** (*str*) – Aberration correction flag
- **obsrvr** (*str*) – Name of the observing body.
- **dref** (*str*) – Reference frame of direction vector of *dvec*.
- **dvec** (*3-Element Array of floats*) – Pointing direction vector from the observer.
- **crdsys** (*str*) – Name of the coordinate system containing COORD.
- **coord** (*str*) – Name of the coordinate of interest
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Absolute extremum adjustment value.
- **step** (*float*) – Step size in seconds for finding angular separation events.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gfsstp` (*step*)

Set the step size to be returned by `gfstep()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfsstp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfsstp_c.html)

**Parameters** **step** (*float*) – Time step to take.

`spiceypy.spiceypy.gfstep` (*time*)

Return the time step set by the most recent call to `gfsstp()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfstep\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfstep_c.html)

**Parameters** **time** (*float*) – Ignored ET value.

**Returns** Time step to take.

**Return type** float

`spiceypy.spiceypy.gfstol` (*value*)

Override the default GF convergence value used in the high level GF routines.

Default value is 1.0e-6

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfstol\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfstol_c.html)

**Parameters** **value** (*float*) – Double precision value returned or to store.

`spiceypy.spiceypy.gfsubc` (*target, fixref, method, abcorr, obsrvr, crdsys, coord, relate, refval, adjust, step, nintvals, cnfine, result*)

Determine time intervals for which a coordinate of an subpoint position vector satisfies a numerical constraint.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfsubc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfsubc_c.html)

**Parameters**

- **target** (*str*) – Name of the target body.
- **fixref** (*str*) – Body fixed frame associated with the target.
- **method** (*str*) – Name of method type for subpoint calculation.
- **abcorr** (*str*) – Aberration correction flag
- **obsrvr** (*str*) – Name of the observing body.
- **crdsys** (*str*) – Name of the coordinate system containing COORD.
- **coord** (*str*) – Name of the coordinate of interest
- **relate** (*str*) – Relational operator.
- **refval** (*float*) – Reference value.
- **adjust** (*float*) – Adjustment value for absolute extrema searches.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvals** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

`spiceypy.spiceypy.gftfov` (*inst, target, tshape, tframe, abcorr, obsrvr, step, cnfine*)

Determine time intervals when a specified ephemeris object intersects the space bounded by the field-of-view (FOV) of a specified instrument.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gftfov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gftfov_c.html)

**Parameters**

- **inst** (*str*) – Name of the instrument.
- **target** (*str*) – Name of the target body.
- **tshape** (*str*) – Type of shape model used for target body.
- **tframe** (*str*) – Body-fixed, body-centered frame for target body.
- **abcorr** (*str*) – Aberration correction flag.
- **obsrvr** (*str*) – Name of the observing body.

- **step** (*float*) – Step size in seconds for finding FOV events.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.

**Returns** SpiceCell containing set of time intervals, within the confinement period, when the target body is visible

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.gfudb` (*udfuns, udfunb, step, cnfine, result*)  
Perform a GF search on a user defined boolean quantity.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfudb\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfudb_c.html)

**Parameters**

- **udfuns** (*ctypes.CFunctionType*) – Name of the routine that computes a scalar quantity of interest corresponding to an ‘et’.
- **udfunb** (*ctypes.CFunctionType*) – Name of the routine returning the boolean value corresponding to an ‘et’.
- **step** (*float*) – Step size used for locating extrema and roots.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

**Returns** result

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.gfuds` (*udfuns, udqdec, relate, refval, adjust, step, nintvls, cnfine, result*)  
Perform a GF search on a user defined scalar quantity.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gfuds\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gfuds_c.html)

**Parameters**

- **udfuns** (*ctypes.CFunctionType*) – Name of the routine that computes the scalar quantity of interest at some time.
- **udqdec** (*ctypes.CFunctionType*) – Name of the routine that computes whether the scalar quantity is decreasing.
- **relate** (*str*) – Operator that either looks for an extreme value (max, min, local, absolute) or compares the geometric quantity value and a number.
- **refval** (*float*) – Value used as reference for scalar quantity condition.
- **adjust** (*float*) – Allowed variation for absolute extremal geometric conditions.
- **step** (*float*) – Step size used for locating extrema and roots.
- **nintvls** (*int*) – Workspace window interval count.
- **cnfine** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window to which the search is restricted.
- **result** (`spiceypy.utils.support_types.SpiceCell`) – SPICE window containing results.

**Returns** result

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.gipool` (*name, start, room*)

Return the integer value of a kernel variable from the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gipool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gipool_c.html)

**Parameters**

- **name** (*str*) – Name of the variable whose value is to be returned.
- **start** (*int*) – Which component to start retrieving for name.
- **room** (*int*) – The largest number of values to return.

**Returns** Values associated with name.

**Return type** list of int

`spiceypy.spiceypy.gnpool` (*name, start, room, lenout=256*)

Return names of kernel variables matching a specified template.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/gnpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/gnpool_c.html)

**Parameters**

- **name** (*str*) – Template that names should match.
- **start** (*int*) – Index of first matching name to retrieve.
- **room** (*int*) – The largest number of values to return.
- **lenout** (*int*) – Length of strings in output array kvars.

**Returns** Kernel pool variables whose names match name.

**Return type** list of str

`spiceypy.spiceypy.halfpi` ()

Return half the value of pi (the ratio of the circumference of a circle to its diameter).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/halfpi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/halfpi_c.html)

**Returns** Half the value of pi.

**Return type** float

`spiceypy.spiceypy.hrmint` (*xvals, yvals, x*)

Evaluate a Hermite interpolating polynomial at a specified abscissa value.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/hrmint\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/hrmint_c.html)

**Parameters**

- **xvals** (*Array of floats*) – Abscissa values.
- **yvals** (*Array of floats*) – Ordinate and derivative values.
- **x** (*int*) – Point at which to interpolate the polynomial.

**Returns** Interpolated function value at x and the Interpolated function's derivative at x

**Return type** tuple

`spiceypy.spiceypy.hx2dp` (*string*)

Convert a string representing a double precision number in a base 16 scientific notation into its equivalent double precision number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/hx2dp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/hx2dp_c.html)

**Parameters** `string (str)` – Hex form string to convert to double precision.

**Returns** Double precision value to be returned, Or Error Message.

**Return type** float or str

`spiceypy.spiceypy.ident()`

This routine returns the 3x3 identity matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ident\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ident_c.html)

**Returns** The 3x3 identity matrix.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.illum(target, et, abcorr, obsrvr, spoint)`

Deprecated: This routine has been superseded by the CSPICE routine `ilumin`. This routine is supported for purposes of backward compatibility only.

Find the illumination angles at a specified surface point of a target body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/illum\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/illum_c.html)

**Parameters**

- **target** (`str`) – Name of target body.
- **et** (`float`) – Epoch in ephemeris seconds past J2000.
- **abcorr** (`str`) – Desired aberration correction.
- **obsrvr** (`str`) – Name of observing body.
- **spoint** (`3-Element Array of floats`) – Body-fixed coordinates of a target surface point.

**Returns** Phase angle, Solar incidence angle, and Emission angle at the surface point.

**Return type** tuple

`spiceypy.spiceypy.illumf(method, target, ilusrc, et, fixref, abcorr, obsrvr, spoint)`

Compute the illumination angles—phase, incidence, and emission—at a specified point on a target body. Return logical flags indicating whether the surface point is visible from the observer’s position and whether the surface point is illuminated.

The target body’s surface is represented using topographic data provided by DSK files, or by a reference ellipsoid.

The illumination source is a specified ephemeris object.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/illumf\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/illumf_c.html)

**Parameters**

- **method** (`str`) – Computation method.
- **target** (`str`) – Name of target body.
- **ilusrc** (`str`) – Name of illumination source.
- **et** (`float`) – Epoch in ephemeris seconds past J2000.
- **fixref** (`str`) – Body-fixed, body-centered target body frame.
- **abcorr** (`str`) – Desired aberration correction.
- **obsrvr** (`str`) – Name of observing body.

- **spoint** (*3-Element Array of floats*) – Body-fixed coordinates of a target surface point.

**Returns** Target surface point epoch, Vector from observer to target surface point, Phase angle at the surface point, Source incidence angle at the surface point, Emission angle at the surface point, Visibility flag, Illumination flag

**Return type** tuple

`spiceypy.spiceypy.illumg` (*method, target, ilusrc, et, fixref, abcorr, obsrvr, spoint*)

Find the illumination angles (phase, incidence, and emission) at a specified surface point of a target body.

The surface of the target body may be represented by a triaxial ellipsoid or by topographic data provided by DSK files.

The illumination source is a specified ephemeris object. param method: Computation method.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/illumg\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/illumg_c.html)

#### Parameters

- **target** (*str*) – Name of target body.
- **ilusrc** (*str*) – Name of illumination source.
- **et** (*float*) – Epoch in ephemeris seconds past J2000.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Desired aberration correction.
- **obsrvr** (*str*) – Name of observing body.
- **spoint** (*3-Element Array of floats*) – Body-fixed coordinates of a target surface point.

**Returns** Target surface point epoch, Vector from observer to target surface point, Phase angle at the surface point, Source incidence angle at the surface point, Emission angle at the surface point,

**Return type** tuple

`spiceypy.spiceypy.ilumin` (*method, target, et, fixref, abcorr, obsrvr, spoint*)

Find the illumination angles (phase, solar incidence, and emission) at a specified surface point of a target body.

This routine supersedes illum.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ilumin\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ilumin_c.html)

#### Parameters

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Desired aberration correction.
- **obsrvr** (*str*) – Name of observing body.
- **spoint** (*3-Element Array of floats*) – Body-fixed coordinates of a target surface point.

**Returns** Target surface point epoch, Vector from observer to target surface point, Phase angle, Solar incidence angle, and Emission angle at the surface point.

**Return type** tuple

`spiceypy.spiceypy.inedpl(a, b, c, plane)`

Find the intersection of a triaxial ellipsoid and a plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/inedpl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/inedpl_c.html)

**Parameters**

- **a** (*float*) – Length of ellipsoid semi-axis lying on the x-axis.
- **b** (*float*) – Length of ellipsoid semi-axis lying on the y-axis.
- **c** (*float*) – Length of ellipsoid semi-axis lying on the z-axis.
- **plane** (`spiceypy.utils.support_types.Plane`) – Plane that intersects ellipsoid.

**Returns** Intersection ellipse.

**Return type** `spiceypy.utils.support_types.Ellipse`

`spiceypy.spiceypy.inelpl(ellips, plane)`

Find the intersection of an ellipse and a plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/inelpl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/inelpl_c.html)

**Parameters**

- **ellips** – A SPICE ellipse.
- **plane** (`spiceypy.utils.support_types.Plane`) – A SPICE plane.

**Returns** Number of intersection points of plane and ellipse, Point 1, Point 2.

**Return type** tuple

`spiceypy.spiceypy.inrypl(vertex, direct, plane)`

Find the intersection of a ray and a plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/inrypl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/inrypl_c.html)

**Parameters**

- **vertex** (*3-Element Array of floats*) – Vertex vector of ray.
- **direct** (*3-Element Array of floats*) – Direction vector of ray.
- **plane** (`spiceypy.utils.support_types.Plane`) – A SPICE plane.

**Returns** Number of intersection points of ray and plane, Intersection point, if `nxpts == 1`.

**Return type** tuple

`spiceypy.spiceypy.insrtc(item, inset)`

Insert an item into a character set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/insrtc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/insrtc_c.html)

**Parameters**

- **item** (*str or list of str*) – Item to be inserted.
- **inset** (`spiceypy.utils.support_types.SpiceCell`) – Insertion set.

`spiceypy.spiceypy.insrtd(item, inset)`

Insert an item into a double precision set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/insrtd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/insrtd_c.html)



**Parameters**

- **item** (*Union[float, Iterable[float]]*) – Item to be inserted.
- **inset** (*spiceypy.utils.support\_types.SpiceCell*) – Insertion set.

`spiceypy.spiceypy.insrti(item, inset)`

Insert an item into an integer set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/insrti\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/insrti_c.html)

**Parameters**

- **item** (*Union[float, Iterable[int]]*) – Item to be inserted.
- **inset** (*spiceypy.utils.support\_types.SpiceCell*) – Insertion set.

`spiceypy.spiceypy.inter(a, b)`

Intersect two sets of any data type to form a third set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/inter\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/inter_c.html)

**Parameters**

- **a** (*spiceypy.utils.support\_types.SpiceCell*) – First input set.
- **b** (*spiceypy.utils.support\_types.SpiceCell*) – Second input set.

**Returns** Intersection of a and b.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.intmax()`

Return the value of the largest (positive) number representable in a int variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/intmax\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/intmax_c.html)

**Returns** The largest (positive) number representable in a Int variable.

**Return type** int

`spiceypy.spiceypy.intmin()`

Return the value of the smallest (negative) number representable in a SpiceInt variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/intmin\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/intmin_c.html)

**Returns** The smallest (negative) number representable in a Int variable.

**Return type** int

`spiceypy.spiceypy.invert(m)`

Generate the inverse of a 3x3 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/invert\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/invert_c.html)

**Parameters** **m** (*3x3-Element Array of floats*) – Matrix to be inverted.

**Returns** Inverted matrix  $(m)^{-1}$

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.invort(m)`

Given a matrix, construct the matrix whose rows are the columns of the first divided by the length squared of the the corresponding columns of the input matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/invort\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/invort_c.html)

**Parameters** **m** (*3x3-Element Array of floats*) – A 3x3 Matrix.

**Returns**  $m$  after transposition and scaling of rows.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.isordv(array, n)`

Determine whether an array of  $n$  items contains the integers 0 through  $n-1$ .

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/isordv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/isordv_c.html)

**Parameters**

- **array** (*Array of ints*) – Array of integers.
- **n** (*int*) – Number of integers in array.

**Returns** The function returns True if the array contains the integers 0 through  $n-1$ , otherwise it returns False.

**Return type** *bool*

`spiceypy.spiceypy.isrchc(value, ndim, lenvals, array)`

Search for a given value within a character string array. Return the index of the first matching array entry, or -1 if the key value was not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/isrchc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/isrchc_c.html)

**Parameters**

- **value** (*str*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.
- **lenvals** (*int*) – String length.
- **array** (*list of str*) – Character string array to search.

**Returns** The index of the first matching array element or -1 if the value is not found.

**Return type** *int*

`spiceypy.spiceypy.isrchr(value, ndim, array)`

Search for a given value within a double precision array. Return the index of the first matching array entry, or -1 if the key value was not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/isrchr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/isrchr_c.html)

**Parameters**

- **value** (*float*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of floats*) – Double Precision array to search.

**Returns** The index of the first matching array element or -1 if the value is not found.

**Return type** *int*

`spiceypy.spiceypy.isrchi(value, ndim, array)`

Search for a given value within an integer array. Return the index of the first matching array entry, or -1 if the key value was not found.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/isrchi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/isrchi_c.html)

**Parameters**

- **value** (*int*) – Key value to be found in array.
- **ndim** (*int*) – Dimension of array.

- **array** (*Array of ints*) – Integer array to search.

**Returns** The index of the first matching array element or -1 if the value is not found.

**Return type** *int*

`spiceypy.spiceypy.isrot` (*m, ntol, dtol*)

Indicate whether a 3x3 matrix is a rotation matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/isrot\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/isrot_c.html)

**Parameters**

- **m** (*3x3-Element Array of floats*) – A matrix to be tested.
- **ntol** (*float*) – Tolerance for the norms of the columns of m.
- **dtol** (*float*) – Tolerance for the determinant of a matrix whose columns are the unitized columns of m.

**Returns** True if and only if m is a rotation matrix.

**Return type** *bool*

`spiceypy.spiceypy.iswhsp` (*string*)

Return a boolean value indicating whether a string contains only white space characters.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/iswhsp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/iswhsp_c.html)

**Parameters** **string** (*str*) – String to be tested.

**Returns** the boolean value True if the string is empty or contains only white space characters; otherwise it returns the value False.

**Return type** *bool*

`spiceypy.spiceypy.j1900` ()

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/j1900\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/j1900_c.html)

**Returns** Julian Date of 1899 DEC 31 12:00:00

**Return type** *float*

`spiceypy.spiceypy.j1950` ()

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/j1950\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/j1950_c.html)

**Returns** Julian Date of 1950 JAN 01 00:00:00

**Return type** *float*

`spiceypy.spiceypy.j2000` ()

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/j2000\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/j2000_c.html)

**Returns** Julian Date of 2000 JAN 01 12:00:00

**Return type** *float*

`spiceypy.spiceypy.j2100` ()

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/j2100\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/j2100_c.html)

**Returns** Julian Date of 2100 JAN 01 12:00:00

**Return type** *float*

`spiceypy.spiceypy.jyear` ()

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/jyear\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/jyear_c.html)

**Returns** number of seconds in a julian year

**Return type** float

`spiceypy.spiceypy.kclear()`

Clear the KEEPER subsystem: unload all kernels, clear the kernel pool, and re-initialize the subsystem. Existing watches on kernel variables are retained.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kclear\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kclear_c.html)

`spiceypy.spiceypy.kdata(which, kind, fillen=256, typlen=256, srclen=256)`

Return data for the *n*th kernel that is among a list of specified kernel types.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kdata\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kdata_c.html)

**Parameters**

- **which** (*int*) – Index of kernel to fetch from the list of kernels.
- **kind** (*str*) – The kind of kernel to which fetches are limited.
- **fillen** (*int*) – Available space in output file string.
- **typlen** (*int*) – Available space in output kernel type string.
- **srclen** (*int*) – Available space in output source string.

**Returns** The name of the kernel file, The type of the kernel, Name of the source file used to load file, The handle attached to file.

**Return type** tuple

`spiceypy.spiceypy.kinfo(file, typlen=256, srclen=256)`

Return information about a loaded kernel specified by name.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kinfo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kinfo_c.html)

**Parameters**

- **file** (*str*) – Name of a kernel to fetch information for
- **typlen** (*int*) – Available space in output kernel type string.
- **srclen** (*int*) – Available space in output source string.

**Returns** The type of the kernel, Name of the source file used to load file, The handle attached to file.

**Return type** tuple

`spiceypy.spiceypy.kplfrm(frmcls, outCell=None)`

Return a SPICE set containing the frame IDs of all reference frames of a given class having specifications in the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kplfrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kplfrm_c.html)

**Parameters**

- **frmcls** (*int*) – Frame class.
- **outCell** (`spiceypy.utils.support_types.SpiceCell`) – Optional output Spice Int Cell

**Returns** Set of ID codes of frames of the specified class.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.ktotal(kind)`

Return the current number of kernels that have been loaded via the KEEPER interface that are of a specified type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kttotal\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kttotal_c.html)

**Parameters** **kind** (*str*) – A list of kinds of kernels to count.

**Returns** The number of kernels of type kind.

**Return type** int

`spiceypy.spiceypy.kxtrct` (*keywd*, *terms*, *nterms*, *instring*, *termlen=256*, *stringlen=256*, *substrlen=256*)

Locate a keyword in a string and extract the substring from the beginning of the first word following the keyword to the beginning of the first subsequent recognized terminator of a list.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/kxtrct\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/kxtrct_c.html)

**Parameters**

- **keywd** (*str*) – Word that marks the beginning of text of interest.
- **terms** (*Array of str*) – Set of words, any of which marks the end of text.
- **nterms** (*int*) – Number of terms.
- **instring** (*str*) – String containing a sequence of words.
- **termlen** (*int*) – Length of strings in string array term.
- **stringlen** (*int*) – Available space in argument string.
- **substrlen** (*int*) – Available space in output substring.

**Returns** String containing a sequence of words, String from end of keywd to beginning of first terms item found.

**Return type** tuple

`spiceypy.spiceypy.lastnb` (*string*)

Return the zero based index of the last non-blank character in a character string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lastnb\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lastnb_c.html)

**Parameters** **string** (*str*) – Input character string.

**Returns**

**rtype**

`spiceypy.spiceypy.latcyl` (*radius*, *lon*, *lat*)

Convert from latitudinal coordinates to cylindrical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/latcyl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/latcyl_c.html)

**Parameters**

- **radius** – Distance of a point from the origin.
- **lon** – Angle of the point from the XZ plane in radians.
- **lat** – Angle of the point from the XY plane in radians.

**Returns** (r, lonc, z)

**Return type** tuple

`spiceypy.spiceypy.latrec` (*radius*, *longitude*, *latitude*)

Convert from latitudinal coordinates to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/latrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/latrec_c.html)

**Parameters**

- **radius** (*float*) – Distance of a point from the origin.
- **longitude** (*float*) – Longitude of point in radians.
- **latitude** (*float*) – Latitude of point in radians.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.latsph` (*radius, lon, lat*)

Convert from latitudinal coordinates to spherical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/latsph\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/latsph_c.html)

**Parameters**

- **radius** – Distance of a point from the origin.
- **lon** – Angle of the point from the XZ plane in radians.
- **lat** – Angle of the point from the XY plane in radians.

**Returns** (rho colat, lons)

**Return type** tuple

`spiceypy.spiceypy.latsrf` (*method, target, et, fixref, lonlat*)

Map array of planetocentric longitude/latitude coordinate pairs to surface points on a specified target body.

The surface of the target body may be represented by a triaxial ellipsoid or by topographic data provided by DSK files.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/latsrf\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/latsrf_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in TDB seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **lonlat** (*A 2xM-Element Array of floats*) – Array of longitude/latitude coordinate pairs.

**Returns** Array of surface points.

**Return type** A 3xM-Element Array of floats

`spiceypy.spiceypy.lcase` (*instr, lenout=256*)

Convert the characters in a string to lowercase.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lcase\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lcase_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **lenout** (*int*) – Maximum length of output string.

**Returns** Output string, all lowercase.

**Return type** str

`spiceypy.spiceypy.ldpool` (*filename*)

Load the variables contained in a NAIF ASCII kernel file into the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ldpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ldpool_c.html)

**Parameters** `filename` (*str*) – Name of the kernel file.

`spiceypy.spiceypy.lgrind` (*xvals, yvals, x*)

Evaluate a Lagrange interpolating polynomial for a specified set of coordinate pairs, at a specified abscissa value. Return the value of both polynomial and derivative.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lgrind\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lgrind_c.html)

**Parameters**

- **xvals** (*N-Element Array of floats*) – Abscissa values.
- **yvals** (*N-Element Array of floats*) – Ordinate values.
- **x** (*float*) – Point at which to interpolate the polynomial.

**Returns** Polynomial value at x, Polynomial derivative at x.

**Return type** tuple

`spiceypy.spiceypy.limbpt` (*method, target, et, fixref, abcorr, corloc, obsrvr, refvec, rolstp, ncuts, schstp, soltol, maxn*)

Find limb points on a target body. The limb is the set of points of tangency on the target of rays emanating from the observer. The caller specifies half-planes bounded by the observer-target center vector in which to search for limb points.

The surface of the target body may be represented either by a triaxial ellipsoid or by topographic data.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/limbpt\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/limbpt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Aberration correction.
- **corloc** (*str*) – Aberration correction locus.
- **obsrvr** (*str*) – Name of observing body.
- **refvec** (*3-Element Array of floats*) – Reference vector for cutting half-planes.
- **rolstp** (*float*) – Roll angular step for cutting half-planes.
- **ncuts** (*int*) – Number of cutting half-planes.
- **schstp** (*float*) – Angular step size for searching.
- **soltol** (*float*) – Solution convergence tolerance.
- **maxn** (*int*) – Maximum number of entries in output arrays.

**Returns** Counts of limb points corresponding to cuts, Limb points, Times associated with limb points, Tangent vectors emanating from the observer

**Return type** tuple

`spiceypy.spiceypy.lmpool` (*cvals*)

Load the variables contained in an internal buffer into the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lmpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lmpool_c.html)

**Parameters** `cvals` (*list of str*) – list of strings.

`spiceypy.spiceypy.lparse` (*inlist, delim, nmax*)

Parse a list of items delimited by a single character.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lparse\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lparse_c.html)

**Parameters**

- **inlist** (*list*) – list of items delimited by `delim`.
- **delim** (*str*) – Single character used to delimit items.
- **nmax** (*int*) – Maximum number of items to return.

**Returns** Items in the list, left justified.

**Return type** list of str

`spiceypy.spiceypy.lparsm` (*inlist, delims, nmax, lenout=None*)

Parse a list of items separated by multiple delimiters.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lparsm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lparsm_c.html)

**Parameters**

- **inlist** (*list of strings*) – list of items delimited by `delims`.
- **delims** (*str*) – Single characters which delimit items.
- **nmax** (*int*) – Maximum number of items to return.
- **lenout** (*int*) – Optional Length of strings in item array.

**Returns** Items in the list, left justified.

**Return type** list of strings

`spiceypy.spiceypy.lparss` (*inlist, delims, NMAX=20, LENGTH=50*)

Parse a list of items separated by multiple delimiters, placing the resulting items into a set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lparss\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lparss_c.html)

**Parameters**

- **inlist** – list of items delimited by `delims`.
- **delims** (*str*) – Single characters which delimit items.
- **NMAX** (*int*) – Optional `nmax` of spice set.
- **LENGTH** (*int*) – Optional length of strings in spice set

**Returns** Set containing items in the list, left justified.

**Return type**

`spiceypy.spiceypy.lspcn` (*body, et, abcorr*)

Compute  $L_s$ , the planetocentric longitude of the sun, as seen from a specified body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lspcn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lspcn_c.html)

**Parameters**

- **body** (*str*) – Name of central body.



- **et** (*float*) – Epoch in seconds past J2000 TDB.
- **abcorr** (*str*) – Aberration correction.

**Returns** planetocentric longitude of the sun

**Return type** float

`spiceypy.spiceypy.lstlec` (*string, n, lenvals, array*)

Given a character string and an ordered array of character strings, find the index of the largest array element less than or equal to the given string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstlec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstlec_c.html)

**Parameters**

- **string** (*str*) – Upper bound value to search against.
- **n** (*int*) – Number elements in array.
- **lenvals** (*int*) – String length.
- **array** (*list*) – Array of possible lower bounds.

**Returns** index of the last element of array that is lexically less than or equal to string.

**Return type** int

`spiceypy.spiceypy.lstled` (*x, n, array*)

Given a number *x* and an array of non-decreasing floats find the index of the largest array element less than or equal to *x*.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstled\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstled_c.html)

**Parameters**

- **x** (*float*) – Value to search against.
- **n** (*int*) – Number elements in array.
- **array** (*list*) – Array of possible lower bounds

**Returns** index of the last element of array that is less than or equal to *x*.

**Return type** int

`spiceypy.spiceypy.lstlei` (*x, n, array*)

Given a number *x* and an array of non-decreasing ints, find the index of the largest array element less than or equal to *x*.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstlei\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstlei_c.html)

**Parameters**

- **x** (*int*) – Value to search against.
- **n** (*int*) – Number elements in array.
- **array** (*list*) – Array of possible lower bounds

**Returns** index of the last element of array that is less than or equal to *x*.

**Return type** int

`spiceypy.spiceypy.lstltc` (*string, n, lenvals, array*)

Given a character string and an ordered array of character strings, find the index of the largest array element less than the given string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstltc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstltc_c.html)

**Parameters**

- **string** (*int*) – Upper bound value to search against.
- **n** (*int*) – Number elements in array.
- **lenvals** (*int*) – String length.
- **array** (*list*) – Array of possible lower bounds

**Returns** index of the last element of array that is lexically less than string.

**Return type** int

`spiceypy.spiceypy.lstltd(x, n, array)`

Given a number *x* and an array of non-decreasing floats find the index of the largest array element less than *x*.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstltd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstltd_c.html)

**Parameters**

- **x** (*float*) – Value to search against
- **n** (*int*) – Number elements in array
- **array** (*list*) – Array of possible lower bounds

**Returns** index of the last element of array that is less than *x*.

**Return type** int

`spiceypy.spiceypy.lstlti(x, n, array)`

Given a number *x* and an array of non-decreasing int, find the index of the largest array element less than *x*.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lstlti\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lstlti_c.html)

**Parameters**

- **x** (*int*) – Value to search against
- **n** (*int*) – Number elements in array
- **array** (*list*) – Array of possible lower bounds

**Returns** index of the last element of array that is less than *x*.

**Return type** int

`spiceypy.spiceypy.ltime(etobs, obs, direct, targ)`

This routine computes the transmit (or receive) time of a signal at a specified target, given the receive (or transmit) time at a specified observer. The elapsed time between transmit and receive is also returned.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ltime\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ltime_c.html)

**Parameters**

- **etobs** (*float*) – Epoch of a signal at some observer
- **obs** (*int*) – NAIF ID of some observer
- **direct** (*str*) – Direction the signal travels ( “->” or “<-” )
- **targ** (*int*) – NAIF ID of the target object

**Returns** epoch and time

**Return type** tuple

`spiceypy.spiceypy.lx4dec` (*string*, *first*)

Scan a string from a specified starting position for the end of a decimal number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lx4dec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lx4dec_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **first** (*int*) – First character to scan from in string.

**Returns** last and nchar

**Return type** tuple

`spiceypy.spiceypy.lx4num` (*string*, *first*)

Scan a string from a specified starting position for the end of a number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lx4num\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lx4num_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **first** (*int*) – First character to scan from in string.

**Returns** last and nchar

**Return type** tuple

`spiceypy.spiceypy.lx4sgn` (*string*, *first*)

Scan a string from a specified starting position for the end of a signed integer.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lx4sgn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lx4sgn_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **first** (*int*) – First character to scan from in string.

**Returns** last and nchar

**Return type** tuple

`spiceypy.spiceypy.lx4uns` (*string*, *first*)

Scan a string from a specified starting position for the end of an unsigned integer.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lx4uns\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lx4uns_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **first** (*int*) – First character to scan from in string.

**Returns** last and nchar

**Return type** tuple

`spiceypy.spiceypy.lxqstr` (*string*, *qchar*, *first*)

Lex (scan) a quoted string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/lxqstr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/lxqstr_c.html)

**Parameters**

- **string** (*str*) – String to be scanned.
- **qchar** (*char* (*string of one char*)) – Quote delimiter character.

- **first** (*int*) – Character position at which to start scanning.

**Returns** last and nchar

**Return type** tuple

`spiceypy.spiceypy.m2eu1` (*r, axis3, axis2, axis1*)

Factor a rotation matrix as a product of three rotations about specified coordinate axes.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/m2eu1\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/m2eu1_c.html)

**Parameters**

- **r** (*3x3-Element Array of floats*) – A rotation matrix to be factored
- **axis3** (*int*) – third rotation axes.
- **axis2** (*int*) – second rotation axes.
- **axis1** (*int*) – first rotation axes.

**Returns** Third, second, and first Euler angles, in radians.

**Return type** tuple

`spiceypy.spiceypy.m2q` (*r*)

Find a unit quaternion corresponding to a specified rotation matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/m2q\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/m2q_c.html)

**Parameters** **r** (*3x3-Element Array of floats*) – A rotation matrix to be factored

**Returns** A unit quaternion representing the rotation matrix

**Return type** 4-Element Array of floats

`spiceypy.spiceypy.matchi` (*string, templ, wstr, wchr*)

Determine whether a string is matched by a template containing wild cards. The pattern comparison is case-insensitive.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/matchi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/matchi_c.html)

**Parameters**

- **string** (*str*) – String to be tested.
- **templ** (*str*) – Template (with wild cards) to test against string.
- **wstr** (*str of length 1*) – Wild string token.
- **wchr** (*str of length 1*) – Wild character token.

**Returns** The function returns True if string matches templ, else False

**Return type** *bool*

`spiceypy.spiceypy.matchw` (*string, templ, wstr, wchr*)

Determine whether a string is matched by a template containing wild cards.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/matchw\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/matchw_c.html)

**Parameters**

- **string** (*str*) – String to be tested.
- **templ** (*str*) – Template (with wild cards) to test against string.
- **wstr** (*str of length 1*) – Wild string token.
- **wchr** (*str of length 1*) – Wild character token.

**Returns** The function returns True if string matches templ, else False

**Return type** *bool*

`spiceypy.spiceypy.megu(m1)`

Set one double precision 3x3 matrix equal to another.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mequ\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mequ_c.html)

**Parameters** **m1** (*3x3-Element Array of floats*) – input matrix.

**Returns** Output matrix equal to m1.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.mequg(m1, nr, nc)`

Set one double precision matrix of arbitrary size equal to another.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mequg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mequg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – Input matrix.
- **nr** (*int*) – Row dimension of m1.
- **nc** (*int*) – Column dimension of m1.

**Returns** Output matrix equal to m1

**Return type** NxM-Element Array of floats

`spiceypy.spiceypy.mtxm(m1, m2)`

Multiply the transpose of a 3x3 matrix and a 3x3 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mtx\\_m\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mtx_m_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **m2** (*3x3-Element Array of floats*) – 3x3 double precision matrix.

**Returns** The produce m1 transpose times m2.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.mtxmg(m1, m2, ncol1, nr1r2, ncol2)`

Multiply the transpose of a matrix with another matrix, both of arbitrary size.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mtxmg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mtxmg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – nr1r2 X ncol1 double precision matrix.
- **m2** (*NxM-Element Array of floats*) – nr1r2 X ncol2 double precision matrix.
- **ncol1** (*int*) – Column dimension of m1 and row dimension of mout.
- **nr1r2** (*int*) – Row dimension of m1 and m2.
- **ncol2** (*int*) – Column dimension of m2.

**Returns** Transpose of m1 times m2.

**Return type** NxM-Element Array of floats

`spiceypy.spiceypy.mtxv(m1, vin)`

Multiplies the transpose of a 3x3 matrix on the left with a vector on the right.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mtxv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mtxv_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **vin** (*3-Element Array of floats*) – 3-dimensional double precision vector.

**Returns** 3-dimensional double precision vector.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.mtxvg(m1, v2, ncol1, nr1r2)`

Multiply the transpose of a matrix and a vector of arbitrary size.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mtxvg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mtxvg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – Left-hand matrix to be multiplied.
- **v2** (*Array of floats*) – Right-hand vector to be multiplied.
- **ncol1** (*int*) – Column dimension of m1 and length of vout.
- **nr1r2** (*int*) – Row dimension of m1 and length of v2.

**Returns** Product vector m1 transpose \* v2.

**Return type** Array of floats

`spiceypy.spiceypy.mxm(m1, m2)`

Multiply two 3x3 matrices.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxm_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **m2** (*3x3-Element Array of floats*) – 3x3 double precision matrix.

**Returns** 3x3 double precision matrix.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.mxmg(m1, m2, nrow1, ncol1, ncol2)`

Multiply two double precision matrices of arbitrary size.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxmg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxmg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – nrow1 X ncol1 double precision matrix.
- **m2** (*NxM-Element Array of floats*) – ncol1 X ncol2 double precision matrix.
- **nrow1** (*int*) – Row dimension of m1
- **ncol1** (*int*) – Column dimension of m1 and row dimension of m2.
- **ncol2** (*int*) – Column dimension of m2

**Returns** nrow1 X ncol2 double precision matrix.

**Return type** NxM-Element Array of floats

`spiceypy.spiceypy.mxmt(m1, m2)`

Multiply a 3x3 matrix and the transpose of another 3x3 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxmt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxmt_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **m2** (*3x3-Element Array of floats*) – 3x3 double precision matrix.

**Returns** The product m1 times m2 transpose.

**Return type** float

`spiceypy.spiceypy.mxmtg(m1, m2, nrow1, nc1c2, nrow2)`

Multiply a matrix and the transpose of a matrix, both of arbitrary size.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxmtg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxmtg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – Left-hand matrix to be multiplied.
- **m2** (*NxM-Element Array of floats*) – Right-hand matrix whose transpose is to be multiplied
- **nrow1** (*int*) – Row dimension of m1 and row dimension of mout.
- **nc1c2** (*int*) – Column dimension of m1 and column dimension of m2.
- **nrow2** (*int*) – Row dimension of m2 and column dimension of mout.

**Returns** Product matrix.

**Return type** NxM-Element Array of floats

`spiceypy.spiceypy.mxv(m1, vin)`

Multiply a 3x3 double precision matrix with a 3-dimensional double precision vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxv_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **vin** (*3-Element Array of floats*) – 3-dimensional double precision vector.

**Returns** 3-dimensional double precision vector.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.mxvlg(m1, v2, nrow1, nc1r2)`

Multiply a matrix and a vector of arbitrary size.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/mxvlg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/mxvlg_c.html)

**Parameters**

- **m1** (*NxM-Element Array of floats*) – Left-hand matrix to be multiplied.
- **v2** (*Array of floats*) – Right-hand vector to be multiplied.
- **nrow1** (*int*) – Row dimension of m1 and length of vout.
- **nc1r2** (*int*) – Column dimension of m1 and length of v2.

**Returns** Product vector m1\*v2

**Return type** Array of floats

`spiceypy.spiceypy.namfrm` (*fname*)

Look up the frame ID code associated with a string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/namfrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/namfrm_c.html)

**Parameters** `fname` (*str*) – The name of some reference frame.

**Returns** The SPICE ID code of the frame.

**Return type** `int`

`spiceypy.spiceypy.ncpos` (*string, chars, start*)

Find the first occurrence in a string of a character NOT belonging to a collection of characters, starting at a specified location searching forward.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ncpos\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ncpos_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **chars** (*str*) – A collection of characters.
- **start** (*int*) – Position to begin looking for one not in chars.

**Returns** `index`

**Return type** `int`

`spiceypy.spiceypy.ncposr` (*string, chars, start*)

Find the first occurrence in a string of a character NOT belonging to a collection of characters, starting at a specified location, searching in reverse.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ncposr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ncposr_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **chars** (*str*) – A collection of characters.
- **start** (*int*) – Position to begin looking for one of chars.

**Returns** `index`

**Return type** `int`

`spiceypy.spiceypy.nearpt` (*positn, a, b, c*)

locates the point on the surface of an ellipsoid that is nearest to a specified position. It also returns the altitude of the position above the ellipsoid.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/nearpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/nearpt_c.html)

**Parameters**

- **positn** (*3-Element Array of floats*) – Position of a point in bodyfixed frame.
- **a** (*float*) – Length of semi-axis parallel to x-axis.
- **b** (*float*) – Length of semi-axis parallel to y-axis.
- **c** (*float*) – Length on semi-axis parallel to z-axis.

**Returns** Point on the ellipsoid closest to `positn`, Altitude of `positn` above the ellipsoid.

**Return type** `tuple`



`spiceypy.spiceypy.no_found_check()`

Temporarily disables spiceypy default behavior which raises exceptions for false found flags for certain spice functions. All spice functions executed within the context manager will no longer check the found flag return parameter and the found flag will be included in the return for the given function. For Example `bodc2n` in spiceypy is normally called like:

```
name = spice.bodc2n(399)
```

With the possibility that an exception is thrown in the even of a invalid ID:

```
name = spice.bodc2n(-999991) # throws a SpiceyError
```

With this function however, we can use it as a context manager to do this:

```
with spice.no_found_check():
    name, found = spice.bodc2n(-999991) # found is false, no exception raised!
```

Within the context any spice functions called that normally check the found flags will pass through the check without raising an exception if they are false.

`spiceypy.spiceypy.npedln(a, b, c, linept, linedr)`

Find nearest point on a triaxial ellipsoid to a specified line and the distance from the ellipsoid to the line.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/npedln\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/npedln_c.html)

#### Parameters

- **a** (*float*) – Length of ellipsoid’s semi-axis in the x direction
- **b** (*float*) – Length of ellipsoid’s semi-axis in the y direction
- **c** (*float*) – Length of ellipsoid’s semi-axis in the z direction
- **linept** (*3-Element Array of floats*) – Length of ellipsoid’s semi-axis in the z direction
- **linedr** (*3-Element Array of floats*) – Direction vector of line

**Returns** Nearest point on ellipsoid to line, Distance of ellipsoid from line

**Return type** tuple

`spiceypy.spiceypy.npelpt(point, ellips)`

Find the nearest point on an ellipse to a specified point, both in three-dimensional space, and find the distance between the ellipse and the point. [http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/npelpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/npelpt_c.html)

#### Parameters

- **point** (*3-Element Array of floats*) – Point whose distance to an ellipse is to be found.
- **ellips** (`spiceypy.utils.support_types.Ellipse`) – An ellipse.

**Returns** Nearest point on ellipsoid to line, Distance of ellipsoid from line

**Return type** tuple

`spiceypy.spiceypy.nplnpt(linept, lindir, point)`

Find the nearest point on a line to a specified point, and find the distance between the two points.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/nplnpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/nplnpt_c.html)

#### Parameters

- **linept** (*3-Element Array of floats*) – Point on a line

- **lindir** (*3-Element Array of floats*) – line’s direction vector
- **point** (*3-Element Array of floats*) – A second point.

**Returns** Nearest point on the line to point, Distance between point and pnear

**Return type** tuple

`spiceypy.spiceypy.nvc2pl` (*normal, constant*)  
Make a plane from a normal vector and a constant.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/nvc2pl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/nvc2pl_c.html)

**Parameters**

- **normal** (*3-Element Array of floats*) – A normal vector defining a plane.
- **constant** (*float*) – A constant defining a plane.

**Returns** plane

**Return type** `spiceypy.utils.support_types.Plane`

`spiceypy.spiceypy.nvp2pl` (*normal, point*)  
Make a plane from a normal vector and a point.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/nvp2pl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/nvp2pl_c.html)

**Parameters**

- **normal** (*3-Element Array of floats*) – A normal vector defining a plane.
- **point** (*3-Element Array of floats*) – A point defining a plane.

**Returns** plane

**Return type** `spiceypy.utils.support_types.Plane`

`spiceypy.spiceypy.occult` (*target1, shape1, frame1, target2, shape2, frame2, abcorr, observer, et*)  
Determines the occultation condition (not occulted, partially, etc.) of one target relative to another target as seen by an observer at a given time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/occult\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/occult_c.html)

**Parameters**

- **target1** (*str*) – Name or ID of first target.
- **shape1** (*str*) – Type of shape model used for first target.
- **frame1** (*str*) – Body-fixed, body-centered frame for first body.
- **target2** (*str*) – Name or ID of second target.
- **shape2** (*str*) – Type of shape model used for second target.
- **frame2** (*str*) – Body-fixed, body-centered frame for second body.
- **abcorr** (*str*) – Aberration correction flag.
- **observer** (*str*) – Name or ID of the observer.
- **et** (*float*) – Time of the observation (seconds past J2000).

**Returns** Occultation identification code.

**Return type** int

`spiceypy.spiceypy.ordc` (*item, inset*)

The function returns the ordinal position of any given item in a character set. If the item does not appear in the set, the function returns -1.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ordc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ordc_c.html)

**Parameters**

- **item** (*str*) – An item to locate within a set.
- **inset** (*SpiceCharCell*) – A set to search for a given item.

**Returns** the ordinal position of item within the set

**Return type** int

`spiceypy.spiceypy.ordd` (*item, inset*)

The function returns the ordinal position of any given item in a double precision set. If the item does not appear in the set, the function returns -1.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ordd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ordd_c.html)

**Parameters**

- **item** (*float*) – An item to locate within a set.
- **inset** (*SpiceDoubleCell*) – A set to search for a given item.

**Returns** the ordinal position of item within the set

**Return type** int

`spiceypy.spiceypy.orderc` (*array, ndim=None*)

Determine the order of elements in an array of character strings.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/orderc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/orderc_c.html)

**Parameters**

- **array** (*Array of strings.*) – Input array.
- **ndim** (*int*) – Optional Length of input array

**Returns** Order vector for array.

**Return type** array of ints

`spiceypy.spiceypy.orderd` (*array, ndim=None*)

Determine the order of elements in a double precision array.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/orderd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/orderd_c.html)

**Parameters**

- **array** (*Array of floats*) – Input array.
- **ndim** (*int*) – Optional Length of input array

**Returns** Order vector for array.

**Return type** array of ints

`spiceypy.spiceypy.orderi` (*array, ndim=None*)

Determine the order of elements in an integer array.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/orderi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/orderi_c.html)

**Parameters**

- **array** (*Array of ints*) – Input array.
- **ndim** (*int*) – Optional Length of input array

**Returns** Order vector for array.

**Return type** array of ints

`spiceypy.spiceypy.ordi` (*item, inset*)

The function returns the ordinal position of any given item in an integer set. If the item does not appear in the set, the function returns -1.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ordi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ordi_c.html)

**Parameters**

- **item** (*int*) – An item to locate within a set.
- **inset** (*SpiceIntCell*) – A set to search for a given item.

**Returns** the ordinal position of item within the set

**Return type** int

`spiceypy.spiceypy.oscelt` (*state, et, mu*)

Determine the set of osculating conic orbital elements that corresponds to the state (position, velocity) of a body at some epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/oscelt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/oscelt_c.html)

**Parameters**

- **state** (*Float Array of 6 elements.*) – State of body at epoch of elements.
- **et** (*float*) – Epoch of elements.
- **mu** (*float*) – Gravitational parameter (GM) of primary body.

**Returns** Equivalent conic elements

**Return type** Float Array of 8 elements.

`spiceypy.spiceypy.oscltx` (*state, et, mu*)

Determine the set of osculating conic orbital elements that corresponds to the state (position, velocity) of a body at some epoch. In addition to the classical elements, return the true anomaly, semi-major axis, and period, if applicable.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/oscltx\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/oscltx_c.html)

**Parameters**

- **state** (*6-Element Array of floats*) – State of body at epoch of elements.
- **et** (*float*) – Epoch of elements.
- **mu** (*float*) – Gravitational parameter (GM) of primary body.

**Returns** Extended set of classical conic elements.

`spiceypy.spiceypy.pckcls` (*handle*)

Close an open PCK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckcls\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckcls_c.html)

**Parameters** **handle** (*int*) – Handle of the PCK file to be closed.

`spiceypy.spiceypy.pckcov` (*pck, idcode, cover*)

Find the coverage window for a specified reference frame in a specified binary PCK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckcov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckcov_c.html)

#### Parameters

- **pck** (*str*) – Name of PCK file.
- **idcode** (*int*) – Class ID code of PCK reference frame.
- **cover** (*SpiceCell*) – Window giving coverage in pck for idcode.

`spiceypy.spiceypy.pckfrm` (*pck, ids*)

Find the set of reference frame class ID codes of all frames in a specified binary PCK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckfrm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckfrm_c.html)

#### Parameters

- **pck** (*str*) – Name of PCK file.
- **ids** (*SpiceCell*) – Set of frame class ID codes of frames in PCK file.

`spiceypy.spiceypy.pcklof` (*filename*)

Load a binary PCK file for use by the readers. Return the handle of the loaded file which is used by other PCK routines to refer to the file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pcklof\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pcklof_c.html)

**Parameters** **filename** (*str*) – Name of the file to be loaded.

**Returns** Loaded file's handle.

**Return type** `int`

`spiceypy.spiceypy.pckopn` (*name, ifname, ncomch*)

Create a new PCK file, returning the handle of the opened file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckopn\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckopn_c.html)

#### Parameters

- **name** (*str*) – The name of the PCK file to be opened.
- **ifname** (*str*) – The internal filename for the PCK.
- **ncomch** (*int*) – The number of characters to reserve for comments.

**Returns** The handle of the opened PCK file.

**Return type** `int`

`spiceypy.spiceypy.pckuof` (*handle*)

Unload a binary PCK file so that it will no longer be searched by the readers.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckuof\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckuof_c.html)

**Parameters** **handle** (*int*) – Handle of PCK file to be unloaded

`spiceypy.spiceypy.pckw02` (*handle, classid, ffname, first, last, segid, intlen, n, polydg, cdata, btime*)

Write a type 2 segment to a PCK binary file given the file handle, frame class ID, base frame, time range covered by the segment, and the Chebyshev polynomial coefficients.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pckw02\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pckw02_c.html)

#### Parameters

- **handle** (*int*) – Handle of binary PCK file open for writing.

- **classid** (*int*) – Frame class ID of body-fixed frame.
- **frname** (*str*) – Name of base reference frame.
- **first** (*float*) – Start time of interval covered by segment.
- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **intlen** (*float*) – Length of time covered by logical record.
- **n** (*int*) – Number of logical records in segment.
- **polydg** (*int*) – Chebyshev polynomial degree.
- **cdata** (*N-Element Array of floats*) – Array of Chebyshev coefficients.
- **btime** (*float*) – Begin time of first logical record.

`spiceypy.spiceypy.pcpool` (*name, cvals*)

This entry point provides toolkit programmers a method for programmatically inserting character data into the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pcpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pcpool_c.html)

#### Parameters

- **name** (*str*) – The kernel pool name to associate with *cvals*.
- **cvals** (*Array of str*) – An array of strings to insert into the kernel pool.

`spiceypy.spiceypy.pdpool` (*name, dvals*)

This entry point provides toolkit programmers a method for programmatically inserting double precision data into the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pdpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pdpool_c.html)

#### Parameters

- **name** (*str*) – The kernel pool name to associate with *dvals*.
- **dvals** (*SpiceCell*) – An array of values to insert into the kernel pool.

`spiceypy.spiceypy.pgrrec` (*body, lon, lat, alt, re, f*)

Convert planetographic coordinates to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pgrrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pgrrec_c.html)

#### Parameters

- **body** (*str*) – Body with which coordinate system is associated.
- **lon** (*float*) – Planetographic longitude of a point (radians).
- **lat** (*float*) – Planetographic latitude of a point (radians).
- **alt** (*float*) – Altitude of a point above reference spheroid.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.phaseq` (*et, target, illmn, obsrvr, abcorr*)

Compute the apparent phase angle for a target, observer, illuminator set of ephemeris objects.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/phaseq\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/phaseq_c.html)

**Parameters**

- **et** (*float*) – Ephemeris seconds past J2000 TDB.
- **target** (*str*) – Target body name.
- **illmn** (*str*) – Illuminating body name.
- **obsrvr** (*str*) – Observer body.
- **abcorr** (*str*) – Aberration correction flag.

**Returns** Value of phase angle.

**Return type** float

`spiceypy.spiceypy.pi` ()

Return the value of pi (the ratio of the circumference of a circle to its diameter).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pi_c.html)

**Returns** value of pi.

**Return type** float

`spiceypy.spiceypy.pipool` (*name, ivals*)

This entry point provides toolkit programmers a method for programmatically inserting integer data into the kernel pool.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pipool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pipool_c.html)

**Parameters**

- **name** (*str*) – The kernel pool name to associate with values.
- **ivals** (*Array of ints*) – An array of integers to insert into the pool.

`spiceypy.spiceypy.pjelpl` (*elin, plane*)

Project an ellipse onto a plane, orthogonally.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pjelpl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pjelpl_c.html)

**Parameters**

- **elin** (`spiceypy.utils.support_types.Ellipse`) – A SPICE ellipse to be projected.
- **plane** (`supporttypes.Plane`) – A plane onto which elin is to be projected.

**Returns** A SPICE ellipse resulting from the projection.

**Return type** `spiceypy.utils.support_types.Ellipse`

`spiceypy.spiceypy.pl2nvc` (*plane*)

Return a unit normal vector and constant that define a specified plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pl2nvc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pl2nvc_c.html)

**Parameters** **plane** (`supporttypes.Plane`) – A SPICE plane.

**Returns** A normal vector and constant defining the geometric plane represented by plane.

**Return type** tuple

`spiceypy.spiceypy.pl2nvp` (*plane*)

Return a unit normal vector and point that define a specified plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pl2nvp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pl2nvp_c.html)

**Parameters** `plane` (*supporttypes.Plane*) – A SPICE plane.

**Returns** A unit normal vector and point that define plane.

**Return type** tuple

`spiceypy.spiceypy.pl2psv` (*plane*)

Return a point and two orthogonal spanning vectors that generate a specified plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pl2psv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pl2psv_c.html)

**Parameters** `plane` (*supporttypes.Plane*) – A SPICE plane.

**Returns** A point in the input plane and two vectors spanning the input plane.

**Return type** tuple

`spiceypy.spiceypy.pltar` (*vrctes, plates*)

Compute the total area of a collection of triangular plates.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pltar\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pltar_c.html)

**Parameters**

- **vrctes** (*Nx3-Element Array of floats*) – Array of vertices.
- **plates** (*Nx3-Element Array of ints*) – Array of plates.

**Returns** total area of the set of plates

**Return type** float

`spiceypy.spiceypy.pltexp` (*iverts, delta*)

Expand a triangular plate by a specified amount. The expanded plate is co-planar with, and has the same orientation as, the original. The centroids of the two plates coincide.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pltexp\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pltexp_c.html)

**Parameters**

- **iverts** (*3x3-Element Array of floats*) – Vertices of the plate to be expanded.
- **delta** (*double*) – Fraction by which the plate is to be expanded.

**Returns** Vertices of the expanded plate.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.pltnp` (*point, v1, v2, v3*)

Find the nearest point on a triangular plate to a given point.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pltnp\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pltnp_c.html)

**Parameters**

- **point** (*3-Element Array of floats*) – A point in 3-dimensional space.
- **v1** (*3-Element Array of floats*) – Vertices of a triangular plate.
- **v2** (*3-Element Array of floats*) – Vertices of a triangular plate.
- **v3** (*3-Element Array of floats*) – Vertices of a triangular plate.

**Returns** the nearest point on a triangular plate to a given point and distance



**Return type** tuple

`spiceypy.spiceypy.pltnrm(v1, v2, v3)`

Compute an outward normal vector of a triangular plate. The vector does not necessarily have unit length.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pltnrm\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pltnrm_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – Vertices of a plate.
- **v2** (*3-Element Array of floats*) – Vertices of a plate.
- **v3** (*3-Element Array of floats*) – Vertices of a plate.

**Returns** Plate's outward normal vector.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.pltvol(vrtces, plates)`

Compute the volume of a three-dimensional region bounded by a collection of triangular plates.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pltvol\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pltvol_c.html)

**Parameters**

- **vrtces** (*Nx3-Element Array of floats*) – Array of vertices.
- **plates** (*Nx3-Element Array of ints*) – Array of plates.

**Returns** the volume of the spatial region bounded by the plates.

**Return type** float

`spiceypy.spiceypy.polyds(coeffs, deg, nderiv, t)`

Compute the value of a polynomial and its first n derivatives at the value t.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/polyds\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/polyds_c.html)

**Parameters**

- **coeffs** (*N-Element Array of floats*) – Coefficients of the polynomial to be evaluated.
- **deg** (*int*) – Degree of the polynomial to be evaluated.
- **nderiv** (*int*) – Number of derivatives to compute.
- **t** (*float*) – Point to evaluate the polynomial and derivatives

**Returns** Value of polynomial and derivatives.

**Return type** nderiv-Element Array of floats

`spiceypy.spiceypy.pos(string, substr, start)`

Find the first occurrence in a string of a substring, starting at a specified location, searching forward.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pos\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pos_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **substr** (*str*) – Substring to locate in the character string.
- **start** (*int*) – Position to begin looking for substr in string.

**Returns** The index of the first occurrence of substr in string at or following index start.

**Return type** int

`spiceypy.spiceypy.posr` (*string, substr, start*)

Find the first occurrence in a string of a substring, starting at a specified location, searching backward.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/posr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/posr_c.html)

**Parameters**

- **string** (*str*) – Any character string.
- **substr** (*str*) – Substring to locate in the character string.
- **start** (*int*) – Position to begin looking for substr in string.

**Returns** The index of the last occurrence of substr in string at or preceding index start.

**Return type** int

`spiceypy.spiceypy.prop2b` (*gm, pvinit, dt*)

Given a central mass and the state of massless body at time  $t_0$ , this routine determines the state as predicted by a two-body force model at time  $t_0 + dt$ .

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/prop2b\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/prop2b_c.html)

**Parameters**

- **gm** (*float*) – Gravity of the central mass.
- **pvinit** (*6-Element Array of floats*) – Initial state from which to propagate a state.
- **dt** (*float*) – Time offset from initial state to propagate to.

**Returns** The propagated state.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.prsdp` (*string*)

Parse a string as a double precision number, encapsulating error handling.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/prsdpc.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/prsdpc.html)

**Parameters** **string** (*str*) – String representing a d.p. number.

**Returns** D.p. value obtained by parsing string.

**Return type** float

`spiceypy.spiceypy.prsint` (*string*)

Parse a string as an integer, encapsulating error handling.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/prsint\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/prsint_c.html)

**Parameters** **string** (*str*) – String representing an integer.

**Returns** Integer value obtained by parsing string.

**Return type** int

`spiceypy.spiceypy.psv2pl` (*point, span1, span2*)

Make a CSPICE plane from a point and two spanning vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/psv2pl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/psv2pl_c.html)

**Parameters**

- **point** (*3-Element Array of floats*) – A Point.
- **span1** (*3-Element Array of floats*) – First Spanning vector.

- **span2** (*3-Element Array of floats*) – Second Spanning vector.

**Returns** A SPICE plane.

**Return type** supporttypes.Plane

`spiceypy.spiceypy.pxform` (*fromstr, tostr, et*)

Return the matrix that transforms position vectors from one specified frame to another at a specified epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pxform\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pxform_c.html)

**Parameters**

- **fromstr** (*str*) – Name of the frame to transform from.
- **tostr** (*str*) – Name of the frame to transform to.
- **et** (*float*) – Epoch of the rotation matrix.

**Returns** A rotation matrix.

**Return type** 3x3 Element Array of floats

`spiceypy.spiceypy.pxfrm2` (*frame\_from, frame\_to, etfrom, etto*)

Return the 3x3 matrix that transforms position vectors from one specified frame at a specified epoch to another specified frame at another specified epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/pxfrm2\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/pxfrm2_c.html)

**Parameters**

- **frame\_from** (*str*) – Name of the frame to transform from.
- **frame\_to** (*str*) – Name of the frame to transform to.
- **etfrom** (*float*) – Evaluation time of frame\_from.
- **etto** (*float*) – Evaluation time of frame\_to.

**Returns** A position transformation matrix from frame\_from to frame\_to

**Return type** 3x3 Element Array of floats

`spiceypy.spiceypy.q2m` (*q*)

Find the rotation matrix corresponding to a specified unit quaternion.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/q2m\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/q2m_c.html)

**Parameters** **q** (*4-Element Array of floats*) – A unit quaternion.

**Returns** A rotation matrix corresponding to q

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.qcktrc` (*tracelen=256*)

Return a string containing a traceback.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/qcktrc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/qcktrc_c.html)

**Parameters** **tracelen** (*int*) – Maximum length of output traceback string.

**Returns** A traceback string.

**Return type** str

`spiceypy.spiceypy.qdq2av` (*q, dq*)

Derive angular velocity from a unit quaternion and its derivative with respect to time.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/qdq2av\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/qdq2av_c.html)

**Parameters**

- **q** (*4-Element Array of floats*) – Unit SPICE quaternion.
- **dq** (*4-Element Array of floats*) – Derivative of q with respect to time

**Returns** Angular velocity defined by q and dq.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.qxq(q1, q2)`

Multiply two quaternions.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/qxq\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/qxq_c.html)

**Parameters**

- **q1** (*4-Element Array of floats*) – First SPICE quaternion.
- **q2** (*4-Element Array of floats*) – Second SPICE quaternion.

**Returns** Product of q1 and q2.

**Return type** 4-Element Array of floats

`spiceypy.spiceypy.radrec(inrange, re, dec)`

Convert from range, right ascension, and declination to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/radrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/radrec_c.html)

**Parameters**

- **inrange** (*float*) – Distance of a point from the origin.
- **re** (*float*) – Right ascension of point in radians.
- **dec** (*float*) – Declination of point in radians.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.rav2xf(rot, av)`

This routine determines a state transformation matrix from a rotation matrix and the angular velocity of the rotation.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rav2xf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rav2xf_c.html)

**Parameters**

- **rot** (*3x3-Element Array of floats*) – Rotation matrix.
- **av** (*3-Element Array of floats*) – Angular velocity vector.

**Returns** State transformation associated with rot and av.

**Return type** 6x6-Element Array of floats

`spiceypy.spiceypy.raxisa(matrix)`

Compute the axis of the rotation given by an input matrix and the angle of the rotation about that axis.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/raxisa\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/raxisa_c.html)

**Parameters** **matrix** (*3x3-Element Array of floats*) – Rotation matrix.

**Returns** Axis of the rotation, Angle through which the rotation is performed

**Return type** tuple

`spiceypy.spiceypy.rdttext` (*file*, *lenout*=256)

Read the next line of text from a text file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rdttext\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rdttext_c.html)

**Parameters**

- **file** (*str*) – Name of text file.
- **lenout** (*int*) – Available room in output line.

**Returns** Next line from the text file, End-of-file indicator

**Return type** tuple

`spiceypy.spiceypy.reccyl` (*rectan*)

Convert from rectangular to cylindrical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reccyl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reccyl_c.html)

**Parameters** **rectan** (*3-Element Array of floats*) – Rectangular coordinates of a point.

**Returns** Distance from z axis, Angle (radians) from xZ plane, Height above xY plane.

**Return type** tuple

`spiceypy.spiceypy.recgeo` (*rectan*, *re*, *f*)

Convert from rectangular coordinates to geodetic coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/recgeo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/recgeo_c.html)

**Parameters**

- **rectan** (*3-Element Array of floats*) – Rectangular coordinates of a point.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Geodetic longitude (radians), Geodetic latitude (radians), Altitude above reference spheroid

**Return type** tuple

`spiceypy.spiceypy.reclat` (*rectan*)

Convert from rectangular coordinates to latitudinal coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reclat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reclat_c.html)

**Parameters** **rectan** (*3-Element Array of floats*) – Rectangular coordinates of a point.

**Returns** Distance from the origin, Longitude in radians, Latitude in radians

**Return type** tuple

`spiceypy.spiceypy.recpgr` (*body*, *rectan*, *re*, *f*)

Convert rectangular coordinates to planetographic coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/recpgr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/recpgr_c.html)

**Parameters**

- **body** (*str*) – Body with which coordinate system is associated.
- **rectan** (*3-Element Array of floats*) – Rectangular coordinates of a point.
- **re** (*float*) – Equatorial radius of the reference spheroid.
- **f** (*float*) – Flattening coefficient.

**Returns** Planetographic longitude (radians), Planetographic latitude (radians), Altitude above reference spheroid

**Return type** tuple

`spiceypy.spiceypy.recrad` (*rectan*)

Convert rectangular coordinates to range, right ascension, and declination.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/recrad\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/recrad_c.html)

**Parameters** `rectan` (*3-Element Array of floats*) – Rectangular coordinates of a point.

**Returns** Distance of the point from the origin, Right ascension in radians, Declination in radians

**Return type** tuple

`spiceypy.spiceypy.recsph` (*rectan*)

Convert from rectangular coordinates to spherical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/recrad\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/recrad_c.html)

**Parameters** `rectan` (*3-Element Array of floats*) – Rectangular coordinates of a point.

**Returns** Distance from the origin, Angle from the positive Z-axis, Longitude in radians.

**Return type** tuple

`spiceypy.spiceypy.removec` (*item, inset*)

Remove an item from a character set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/removc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/removc_c.html)

**Parameters**

- `item` (*str*) – Item to be removed.
- `inset` (`spiceypy.utils.support_types.SpiceCell`) – Set to be updated.

`spiceypy.spiceypy.removevd` (*item, inset*)

Remove an item from a double precision set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/removd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/removd_c.html)

**Parameters**

- `item` (*float*) – Item to be removed.
- `inset` (`spiceypy.utils.support_types.SpiceCell`) – Set to be updated.

`spiceypy.spiceypy.removi` (*item, inset*)

Remove an item from an integer set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/removi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/removi_c.html)

**Parameters**

- `item` (*int*) – Item to be removed.
- `inset` (`spiceypy.utils.support_types.SpiceCell`) – Set to be updated.

`spiceypy.spiceypy.reordc` (*iorder, ndim, lenvals, array*)

Re-order the elements of an array of character strings according to a given order vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reordc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reordc_c.html)

**Parameters**

- `iorder` (*Array of ints*) – Order vector to be used to re-order array.

- **ndim** (*int*) – Dimension of array.
- **lenvals** (*int*) – String length.
- **array** (*Array of strs*) – Array to be re-ordered.

**Returns** Re-ordered Array.

**Return type** Array of strs

`spiceypy.spiceypy.reordd` (*iorder, ndim, array*)

Re-order the elements of a double precision array according to a given order vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reordd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reordd_c.html)

**Parameters**

- **iorder** (*Array of ints*) – Order vector to be used to re-order array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of floats*) – Array to be re-ordered.

**Returns** Re-ordered Array.

**Return type** Array of floats

`spiceypy.spiceypy.reordi` (*iorder, ndim, array*)

Re-order the elements of an integer array according to a given order vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reordi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reordi_c.html)

**Parameters**

- **iorder** (*Array of ints*) – Order vector to be used to re-order array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of ints*) – Array to be re-ordered.

**Returns** Re-ordered Array.

**Return type** Array of ints

`spiceypy.spiceypy.reordl` (*iorder, ndim, array*)

Re-order the elements of a logical (Boolean) array according to a given order vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reordl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reordl_c.html)

**Parameters**

- **iorder** (*Array of ints*) – Order vector to be used to re-order array.
- **ndim** (*int*) – Dimension of array.
- **array** (*Array of ints*) – Array to be re-ordered.

**Returns** Re-ordered Array.

**Return type** Array of bools

`spiceypy.spiceypy.repmc` (*instr, marker, value, lenout=None*)

Replace a marker with a character string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmc_c.html)

**Parameters**

- **instr** (*str*) – Input string.

- **marker** (*str*) – Marker to be replaced.
- **value** (*str*) – Replacement value.
- **lenout** (*int*) – Optional available space in output string

**Returns** Output string.

**Return type** *str*

`spiceypy.spiceypy.repmtc` (*instr, marker, value, repcase, lenout=None*)  
Replace a marker with the text representation of a cardinal number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmtc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmtc_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **marker** (*str*) – Marker to be replaced.
- **value** (*int*) – Replacement value.
- **repcase** (*str*) – Case of replacement text.
- **lenout** (*int*) – Optional available space in output string

**Returns** Output string.

**Return type** *str*

`spiceypy.spiceypy.repmd` (*instr, marker, value, sigdig*)  
Replace a marker with a double precision number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmd_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **marker** (*str*) – Marker to be replaced.
- **value** (*float*) – Replacement value.
- **sigdig** (*int*) – Significant digits in replacement text.

**Returns** Output string.

**Return type** *str*

`spiceypy.spiceypy.repmf` (*instr, marker, value, sigdig, informat, lenout=None*)  
Replace a marker in a string with a formatted double precision value.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmf\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmf_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **marker** (*str*) – Marker to be replaced.
- **value** (*float*) – Replacement value.
- **sigdig** (*int*) – Significant digits in replacement text.
- **informat** (*str*) – Format ‘E’ or ‘F’.
- **lenout** (*int*) – Optional available space in output string.

**Returns** Output string.



**Return type** *str*

`spiceypy.spiceypy.repmi` (*instr*, *marker*, *value*, *lenout=None*)  
 Replace a marker with an integer.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmi_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **marker** (*str*) – Marker to be replaced.
- **value** (*int*) – Replacement value.
- **lenout** (*int*) – Optional available space in output string.

**Returns** Output string.

**Return type** *str*

`spiceypy.spiceypy.repmot` (*instr*, *marker*, *value*, *repcase*, *lenout=None*)  
 Replace a marker with the text representation of an ordinal number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/repmot\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/repmot_c.html)

**Parameters**

- **instr** (*str*) – Input string.
- **marker** (*str*) – Marker to be replaced.
- **value** (*int*) – Replacement value.
- **repcase** (*str*) – Case of replacement text.
- **lenout** (*int*) – Optional available space in output string.

**Returns** Output string.

**Return type** *str*

`spiceypy.spiceypy.reset` ()

Reset the SPICE error status to a value of “no error.” As a result, the status routine, `failed`, will return a value of `False`

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/reset\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/reset_c.html)

`spiceypy.spiceypy.return_c` ()

True if SPICE routines should return immediately upon entry.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/return\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/return_c.html)

**Returns** True if SPICE routines should return immediately upon entry.

**Return type** *bool*

`spiceypy.spiceypy.rotate` (*angle*, *iaxis*)

Calculate the 3x3 rotation matrix generated by a rotation of a specified angle about a specified axis. This rotation is thought of as rotating the coordinate system.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rotate\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rotate_c.html)

**Parameters**

- **angle** (*float*) – Angle of rotation (radians).
- **iaxis** (*int*) – Axis of rotation X=1, Y=2, Z=3.

**Returns** Resulting rotation matrix

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.rotmat` (*m1*, *angle*, *iaxis*)

Rotmat applies a rotation of angle radians about axis *iaxis* to a matrix. This rotation is thought of as rotating the coordinate system.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rotmat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rotmat_c.html)

**Parameters**

- **m1** (*3x3-Element Array of floats*) – Matrix to be rotated.
- **angle** (*float*) – Angle of rotation (radians).
- **iaxis** (*int*) – Axis of rotation X=1, Y=2, Z=3.

**Returns** Resulting rotated matrix.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.rotvec` (*v1*, *angle*, *iaxis*)

Transform a vector to a new coordinate system rotated by angle radians about axis *iaxis*. This transformation rotates *v1* by angle radians about the specified axis.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rotvec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rotvec_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – Vector whose coordinate system is to be rotated.
- **angle** (*float*) – Angle of rotation (radians).
- **iaxis** (*int*) – Axis of rotation X=1, Y=2, Z=3.

**Returns** the vector expressed in the new coordinate system.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.rpd` ()

Return the number of radians per degree.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rpd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rpd_c.html)

**Returns** The number of radians per degree, pi/180.

**Return type** float

`spiceypy.spiceypy.rquad` (*a*, *b*, *c*)

Find the roots of a quadratic equation.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/rquad\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/rquad_c.html)

**Parameters**

- **a** (*float*) – Coefficient of quadratic term.
- **b** (*float*) – Coefficient of linear term.
- **c** (*float*) – Constant.

**Returns** Root built from positive and negative discriminant term.

**Return type** tuple

`spiceypy.spiceypy.saelgv` (*vec1*, *vec2*)

Find semi-axis vectors of an ellipse generated by two arbitrary three-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/saelgv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/saelgv_c.html)

**Parameters**

- **vec1** (*3-Element Array of floats*) – First vector used to generate an ellipse.
- **vec2** (*3-Element Array of floats*) – Second vector used to generate an ellipse.

**Returns** Semi-major axis of ellipse, Semi-minor axis of ellipse.

**Return type** tuple

`spiceypy.spiceypy.scard` (*incard*, *cell*)

Set the cardinality of a SPICE cell of any data type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scard\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scard_c.html)

**Parameters**

- **incard** (*int*) – Cardinality of (number of elements in) the cell.
- **cell** (`spiceypy.utils.support_types.SpiceCell`) – The cell.

**Returns** The updated Cell.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.scdecd` (*sc*, *sclmdp*, *lenout=256*, *MXPART=None*)

Convert double precision encoding of spacecraft clock time into a character representation.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scdecd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scdecd_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft identification code.
- **sclmdp** (*float*) – Encoded representation of a spacecraft clock count.
- **lenout** (*int*) – Maximum allowed length of output SCLK string.
- **MXPART** (*int*) – Maximum number of spacecraft clock partitions.

**Returns** Character representation of a clock count.

**Return type** str

`spiceypy.spiceypy.sce2c` (*sc*, *et*)

Convert ephemeris seconds past J2000 (ET) to continuous encoded spacecraft clock “ticks”. Non-integral tick values may be returned.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sce2c\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sce2c_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft ID code.
- **et** (*float*) – Ephemeris time, seconds past J2000.

**Returns** SCLK, encoded as ticks since spacecraft clock start. sclmdp need not be integral.

**Return type** float

`spiceypy.spiceypy.sce2s` (*sc*, *et*, *lenout=256*)

Convert an epoch specified as ephemeris seconds past J2000 (ET) to a character string representation of a spacecraft clock value (SCLK).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sce2s\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sce2s_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft clock ID code.
- **et** (*float*) – Ephemeris time, specified as seconds past J2000.
- **lenout** (*int*) – Maximum length of output string.

**Returns** An SCLK string.

**Return type** str

`spiceypy.spiceypy.sce2t` (*sc, et*)

Convert ephemeris seconds past J2000 (ET) to integral encoded spacecraft clock (“ticks”). For conversion to fractional ticks, (required for C-kernel production), see the routine `sce2c` ().

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sce2t\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sce2t_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft ID code.
- **et** (*float*) – Ephemeris time, seconds past J2000.

**Returns** SCLK, encoded as ticks since spacecraft clock start.

**Return type** float

`spiceypy.spiceypy.scencd` (*sc, sclch, MXPART=None*)

Encode character representation of spacecraft clock time into a double precision number.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scencd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scencd_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft identification code.
- **sclch** (*str*) – Character representation of a spacecraft clock.
- **MXPART** (*int*) – Maximum number of spacecraft clock partitions.

**Returns** Encoded representation of the clock count.

**Return type** float

`spiceypy.spiceypy.scfmt` (*sc, ticks, lenout=256*)

Convert encoded spacecraft clock ticks to character clock format.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scfmt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scfmt_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft identification code.
- **ticks** (*float*) – Encoded representation of a spacecraft clock count.
- **lenout** (*int*) – Maximum allowed length of output string.

**Returns** Character representation of a clock count.

**Return type** str

`spiceypy.spiceypy.scpart` (*sc*)

Get spacecraft clock partition information from a spacecraft clock kernel file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scpart\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scpart_c.html)

**Parameters** **sc** (*int*) – NAIF spacecraft identification code.

**Returns** The number of spacecraft clock partitions, Array of partition start times, Array of partition stop times.

**Return type** tuple

`spiceypy.spiceypy.scs2e(sc, sclch)`

Convert a spacecraft clock string to ephemeris seconds past J2000 (ET).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/scs2e\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/scs2e_c.html)

**Parameters**

- **sc** (*int*) – NAIF integer code for a spacecraft.
- **sclch** (*str*) – An SCLK string.

**Returns** Ephemeris time, seconds past J2000.

**Return type** float

`spiceypy.spiceypy.sct2e(sc, sclkdp)`

Convert encoded spacecraft clock (“ticks”) to ephemeris seconds past J2000 (ET).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sct2e\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sct2e_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft ID code.
- **sclkdp** (*float*) – SCLK, encoded as ticks since spacecraft clock start.

**Returns** Ephemeris time, seconds past J2000.

**Return type** float

`spiceypy.spiceypy.sctiks(sc, clkstr)`

Convert a spacecraft clock format string to number of “ticks”.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sctiks\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sctiks_c.html)

**Parameters**

- **sc** (*int*) – NAIF spacecraft identification code.
- **clkstr** (*str*) – Character representation of a spacecraft clock.

**Returns** Number of ticks represented by the clock string.

**Return type** float

`spiceypy.spiceypy.sdifff(a, b)`

Take the symmetric difference of two sets of any data type to form a third set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sdifff\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sdifff_c.html)

**Parameters**

- **a** (`spiceypy.utils.support_types.SpiceCell`) – First input set.
- **b** (`spiceypy.utils.support_types.SpiceCell`) – Second input set.

**Returns** Symmetric difference of a and b.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.set_c(a, op, b)`

Given a relational operator, compare two sets of any data type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/set\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/set_c.html)

**Parameters**

- **a** (`spiceypy.utils.support_types.SpiceCell`) – First set.
- **op** (*str*) – Comparison operator.
- **b** (`spiceypy.utils.support_types.SpiceCell`) – Second set.

**Returns** The function returns the result of the comparison.

**Return type** *bool*

`spiceypy.spiceypy.setmsg` (*message*)

Set the value of the current long error message.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/setmsg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/setmsg_c.html)

**Parameters** **message** (*str*) – A long error message.

`spiceypy.spiceypy.shellc` (*ndim, lenvals, array*)

Sort an array of character strings according to the ASCII collating sequence using the Shell Sort algorithm.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/shelld\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/shelld_c.html)

**Parameters**

- **ndim** (*int*) – Dimension of the array.
- **lenvals** (*int*) – String length.
- **array** (*list of str.*) – The array to be sorted.

**Returns** The sorted array.

**Return type** list of str.

`spiceypy.spiceypy.shelld` (*ndim, array*)

Sort a double precision array using the Shell Sort algorithm.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/shelld\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/shelld_c.html)

**Parameters**

- **ndim** (*int*) – Dimension of the array.
- **array** (*Array of floats*) – The array to be sorted.

**Returns** The sorted array.

**Return type** Array of floats

`spiceypy.spiceypy.shelli` (*ndim, array*)

Sort an integer array using the Shell Sort algorithm.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/shelli\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/shelli_c.html)

**Parameters**

- **ndim** (*int*) – Dimension of the array.
- **array** (*Array of ints*) – The array to be sorted.

**Returns** The sorted array.

**Return type** Array of ints

`spiceypy.spiceypy.sigerr` (*message*)

Inform the CSPIICE error processing mechanism that an error has occurred, and specify the type of error.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sigerr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sigerr_c.html)

**Parameters** `message` (*str*) – A short error message.

`spiceypy.spiceypy.sincpt` (*method, target, et, fixref, abcorr, obsrvr, dref, dvec*)

Given an observer and a direction vector defining a ray, compute the surface intercept of the ray on a target body at a specified epoch, optionally corrected for light time and stellar aberration.

This routine supersedes `srfxpt` ().

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sincpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sincpt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.
- **dref** (*str*) – Reference frame of ray’s direction vector.
- **dvec** (*3-Element Array of floats*) – Ray’s direction vector.

**Returns** Surface intercept point on the target body, Intercept epoch, Vector from observer to intercept point.

**Return type** tuple

`spiceypy.spiceypy.size` (*cell*)

Return the size (maximum cardinality) of a SPICE cell of any data type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/size\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/size_c.html)

**Parameters** `cell` (`spiceypy.utils.support_types.SpiceCell`) – Input cell.

**Returns** The size of the input cell.

**Return type** int

`spiceypy.spiceypy.spd` ()

Return the number of seconds in a day.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spd_c.html)

**Returns** The number of seconds in a day.

**Return type** float

`spiceypy.spiceypy.sphcyl` (*radius, colat, slon*)

This routine converts from spherical coordinates to cylindrical coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sphcyl\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sphcyl_c.html)

**Parameters**

- **radius** (*float*) – Distance of point from origin.
- **colat** (*float*) – Polar angle (co-latitude in radians) of point.
- **slon** (*float*) – Azimuthal angle (longitude) of point (radians).

**Returns** Distance of point from z axis, angle (radians) of point from XZ plane, Height of point above XY plane.

**Return type** tuple

`spiceypy.spiceypy.sphlat` (*r, colat, lons*)

Convert from spherical coordinates to latitudinal coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sphlat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sphlat_c.html)

**Parameters**

- **r** (*float*) – Distance of the point from the origin.
- **colat** (*float*) – Angle of the point from positive z axis (radians).
- **lons** (*float*) – Angle of the point from the XZ plane (radians).

**Returns** Distance of a point from the origin, Angle of the point from the XZ plane in radians, Angle of the point from the XY plane in radians.

**Return type** tuple

`spiceypy.spiceypy.sphrec` (*r, colat, lon*)

Convert from spherical coordinates to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sphrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sphrec_c.html)

**Parameters**

- **r** (*float*) – Distance of a point from the origin.
- **colat** (*float*) – Angle of the point from the positive Z-axis.
- **lon** (*float*) – Angle of the point from the XZ plane in radians.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.spiceErrorCheck` (*f*)

Decorator for spiceypy hooking into spice error system. If an error is detected, an output similar to outmsg

**Returns**

**Return type**

`spiceypy.spiceypy.spiceFoundExceptionThrower` (*f*)

Decorator for wrapping functions that use status codes

`spiceypy.spiceypy.spk14a` (*handle, ncsets, coeffs, epochs*)

Add data to a type 14 SPK segment associated with handle. See also `spk14b()` and `spk14e()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spk14a\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spk14a_c.html)

**Parameters**

- **handle** (*int*) – The handle of an SPK file open for writing.
- **ncsets** (*int*) – The number of coefficient sets and epochs.
- **coeffs** (*Array of floats*) – The collection of coefficient sets.
- **epochs** (*Array of floats*) – The epochs associated with the coefficient sets.

`spiceypy.spiceypy.spk14b` (*handle, segid, body, center, framename, first, last, chbdeg*)

Begin a type 14 SPK segment in the SPK file associated with handle. See also `spk14a()` and `spk14e()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spk14b\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spk14b_c.html)

**Parameters**



- **handle** (*int*) – The handle of an SPK file open for writing.
- **segid** (*str*) – The string to use for segment identifier.
- **body** (*int*) – The NAIF ID code for the body of the segment.
- **center** (*int*) – The center of motion for body.
- **framename** (*str*) – The reference frame for this segment.
- **first** (*float*) – The first epoch for which the segment is valid.
- **last** (*float*) – The last epoch for which the segment is valid.
- **chbdeg** (*int*) – The degree of the Chebyshev Polynomial used.

`spiceypy.spiceypy.spk14e` (*handle*)

End the type 14 SPK segment currently being written to the SPK file associated with *handle*. See also `spk14a()` and `spk14b()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spk14e\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spk14e_c.html)

**Parameters** *handle* (*int*) – The handle of an SPK file open for writing.

`spiceypy.spiceypy.spkacs` (*targ, et, ref, abcorr, obs*)

Return the state (position and velocity) of a target body relative to an observer, optionally corrected for light time and stellar aberration, expressed relative to an inertial reference frame.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkacs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkacs_c.html)

#### Parameters

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Inertial reference frame of output state.
- **abcorr** (*str*) – Aberration correction flag.
- **obs** (*int*) – Observer.

**Returns** State of target, One way light time between observer and target, Derivative of light time with respect to time.

**Return type** tuple

`spiceypy.spiceypy.spkapo` (*targ, et, ref, sobs, abcorr*)

Return the position of a target body relative to an observer, optionally corrected for light time and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkapo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkapo_c.html)

#### Parameters

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Inertial reference frame of observer's state.
- **sobs** (*6-Element Array of floats*) – State of observer wrt. solar system barycenter.
- **abcorr** (*str*) – Aberration correction flag.

**Returns** Position of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkapp` (*targ, et, ref, sobs, abcorr*)

Deprecated: This routine has been superseded by `spkaps()`. This routine is supported for purposes of backward compatibility only.

Return the state (position and velocity) of a target body relative to an observer, optionally corrected for light time and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkapp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkapp_c.html)

#### Parameters

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Inertial reference frame of observer’s state.
- **sobs** (*6-Element Array of floats*) – State of observer wrt. solar system barycenter.
- **abcorr** (*str*) – Aberration correction flag.

**Returns** State of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkaps` (*targ, et, ref, abcorr, stobs, accobs*)

Given the state and acceleration of an observer relative to the solar system barycenter, return the state (position and velocity) of a target body relative to the observer, optionally corrected for light time and stellar aberration. All input and output vectors are expressed relative to an inertial reference frame.

This routine supersedes `spkapp()`.

SPICE users normally should call the high-level API routines `spkezr()` or `spkez()` rather than this routine.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkaps\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkaps_c.html)

#### Parameters

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Inertial reference frame of output state.
- **abcorr** (*str*) – Aberration correction flag.
- **stobs** (*6-Element Array of floats*) – State of the observer relative to the SSB.
- **accobs** (*6-Element Array of floats*) – Acceleration of the observer relative to the SSB.

**Returns** State of target, One way light time between observer and target, Derivative of light time with respect to time.

**Return type** tuple

`spiceypy.spiceypy.spkcls` (*handle*)

Close an open SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcls\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcls_c.html)

**Parameters** **handle** (*int*) – Handle of the SPK file to be closed.

`spiceypy.spiceypy.spkcov` (*spk, idcode, cover*)

Find the coverage window for a specified ephemeris object in a specified SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcov\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcov_c.html)

**Parameters**

- **spk** (*str*) – Name of SPK file.
- **idcode** (*int*) – ID code of ephemeris object.
- **cover** (`spiceypy.utils.support_types.SpiceCell`) – Window giving coverage in “spk” for “idcode”.

`spiceypy.spiceypy.spkcpo` (*target, et, outref, refloc, abcorr, obspos, obsctr, obsref*)

Return the state of a specified target relative to an “observer,” where the observer has constant position in a specified reference frame. The observer’s position is provided by the calling program rather than by loaded SPK files.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcpo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcpo_c.html)

**Parameters**

- **target** (*str*) – Name of target ephemeris object.
- **et** (*float*) – Observation epoch.
- **outref** (*str*) – Reference frame of output state.
- **refloc** (*str*) – Output reference frame evaluation locus.
- **abcorr** (*str*) – Aberration correction.
- **obspos** (*3-Element Array of floats*) – Observer position relative to center of motion.
- **obsctr** (*str*) – Center of motion of observer.
- **obsref** (*str*) – Frame of observer position.

**Returns** State of target with respect to observer, One way light time between target and observer.

**Return type** tuple

`spiceypy.spiceypy.spkcpt` (*trgpos, trgctr, trgref, et, outref, refloc, abcorr, obsrvr*)

Return the state, relative to a specified observer, of a target having constant position in a specified reference frame. The target’s position is provided by the calling program rather than by loaded SPK files.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcpt_c.html)

**Parameters**

- **trgpos** (*3-Element Array of floats*) – Target position relative to center of motion.
- **trgctr** (*str*) – Center of motion of target.
- **trgref** (*str*) – Observation epoch.
- **et** (*float*) – Observation epoch.
- **outref** (*str*) – Reference frame of output state.
- **refloc** (*str*) – Output reference frame evaluation locus.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** – Name of observing ephemeris object.

**Returns** State of target with respect to observer, One way light time between target and observer.

**Return type** tuple

`spiceypy.spiceypy.spkcvo` (*target, et, outref, refloc, abcorr, obssta, obsepc, obsctr, obsref*)

Return the state of a specified target relative to an “observer,” where the observer has constant velocity in a specified reference frame. The observer’s state is provided by the calling program rather than by loaded SPK files.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcvo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcvo_c.html)

**Parameters**

- **target** (*str*) – Name of target ephemeris object.
- **et** (*float*) – Observation epoch.
- **outref** (*str*) – Reference frame of output state.
- **refloc** (*str*) – Output reference frame evaluation locus.
- **abcorr** (*str*) – Aberration correction.
- **obssta** (*6-Element Array of floats*) – Observer state relative to center of motion.
- **obsepc** (*float*) – Epoch of observer state.
- **obsctr** (*str*) – Center of motion of observer.
- **obsref** (*str*) – Frame of observer state.

**Returns** State of target with respect to observer, One way light time between target and observer.

**Return type** tuple

`spiceypy.spiceypy.spkcv`t (*trgsta, trgepc, trgctr, trgref, et, outref, refloc, abcorr, obsrvr*)

Return the state, relative to a specified observer, of a target having constant velocity in a specified reference frame. The target’s state is provided by the calling program rather than by loaded SPK files.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkcv\\_t.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkcv_t.html)

**Parameters**

- **trgsta** (*6-Element Array of floats*) – Target state relative to center of motion.
- **trgepc** (*float*) – Epoch of target state.
- **trgctr** (*str*) – Center of motion of target.
- **trgref** (*str*) – Frame of target state.
- **et** (*float*) – Observation epoch.
- **outref** (*str*) – Reference frame of output state.
- **refloc** (*str*) – Output reference frame evaluation locus.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing ephemeris object.

**Returns** State of target with respect to observer, One way light time between target and observer.

**Return type** tuple

`spiceypy.spiceypy.spkez` (*targ, et, ref, abcorr, obs*)

Return the state (position and velocity) of a target body relative to an observing body, optionally corrected for light time (planetary aberration) and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkez\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkez_c.html)

**Parameters**

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Reference frame of output state vector.
- **abcorr** (*str*) – Aberration correction flag.
- **obs** (*int*) – Observing body.

**Returns** State of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkezp` (*targ, et, ref, abcorr, obs*)

Return the position of a target body relative to an observing body, optionally corrected for light time (planetary aberration) and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkezp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkezp_c.html)

#### Parameters

- **targ** (*int*) – Target body NAIF ID code.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Reference frame of output position vector.
- **abcorr** (*str*) – Aberration correction flag.
- **obs** (*int*) – Observing body NAIF ID code.

**Returns** Position of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkezr` (*targ, et, ref, abcorr, obs*)

Return the state (position and velocity) of a target body relative to an observing body, optionally corrected for light time (planetary aberration) and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkezr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkezr_c.html)

#### Parameters

- **targ** (*str*) – Target body name.
- **et** (*Union[float, Iterable[float]]*) – Observer epoch.
- **ref** (*str*) – Reference frame of output state vector.
- **abcorr** (*str*) – Aberration correction flag.
- **obs** (*str*) – Observing body name.

**Returns** State of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkgeo` (*targ, et, ref, obs*)

Compute the geometric state (position and velocity) of a target body relative to an observing body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkgeo\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkgeo_c.html)

#### Parameters

- **targ** (*int*) – Target body.
- **et** (*float*) – Target epoch.
- **ref** (*str*) – Target reference frame.

- **obs** (*int*) – Observing body.

**Returns** State of target, Light time.

**Return type** tuple

`spiceypy.spiceypy.spkggps` (*targ, et, ref, obs*)

Compute the geometric position of a target body relative to an observing body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkggps\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkggps_c.html)

**Parameters**

- **targ** (*int*) – Target body.
- **et** (*float*) – Target epoch.
- **ref** (*str*) – Target reference frame.
- **obs** (*int*) – Observing body.

**Returns** Position of target, Light time.

**Return type** tuple

`spiceypy.spiceypy.spk1ef` (*filename*)

Load an ephemeris file for use by the readers. Return that file's handle, to be used by other SPK routines to refer to the file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spk1ef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spk1ef_c.html)

**Parameters** **filename** (*str*) – Name of the file to be loaded.

**Returns** Loaded file's handle.

**Return type** int

`spiceypy.spiceypy.spk1tc` (*targ, et, ref, abcorr, stobs*)

Return the state (position and velocity) of a target body relative to an observer, optionally corrected for light time, expressed relative to an inertial reference frame.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spk1tc\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spk1tc_c.html)

**Parameters**

- **targ** (*int*) – Target body.
- **et** (*float*) – Observer epoch.
- **ref** (*str*) – Inertial reference frame of output state.
- **abcorr** (*str*) – Aberration correction flag.
- **stobs** (*6-Element Array of floats*) – State of the observer relative to the SSB.

**Returns** One way light time between observer and target, Derivative of light time with respect to time

**Return type** tuple

`spiceypy.spiceypy.spkobj` (*spk, outCell=None*)

Find the set of ID codes of all objects in a specified SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkobj\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkobj_c.html)

**Parameters**

- **spk** (*str*) – Name of SPK file.

- **outCell** (`spiceypy.utils.support_types.SpiceCell`) – Optional Spice Int Cell.

`spiceypy.spiceypy.spkopa` (*filename*)

Open an existing SPK file for subsequent write.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkopa\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkopa_c.html)

**Parameters** **filename** (*str*) – The name of an existing SPK file.

**Returns** A handle attached to the SPK file opened to append.

**Return type** `int`

`spiceypy.spiceypy.spkopn` (*filename, ifname, ncomch*)

Create a new SPK file, returning the handle of the opened file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkopn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkopn_c.html)

**Parameters**

- **filename** (*str*) – The name of the new SPK file to be created.
- **ifname** (*str*) – The internal filename for the SPK file.
- **ncomch** (*int*) – The number of characters to reserve for comments.

**Returns** The handle of the opened SPK file.

**Return type** `int`

`spiceypy.spiceypy.spkpds` (*body, center, framestr, typenum, first, last*)

Perform routine error checks and if all check pass, pack the descriptor for an SPK segment

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkpds\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkpds_c.html)

**Parameters**

- **body** (*int*) – The NAIF ID code for the body of the segment.
- **center** (*int*) – The center of motion for body.
- **framestr** (*str*) – The frame for this segment.
- **typenum** (*int*) – The type of SPK segment to create.
- **first** (*float*) – The first epoch for which the segment is valid.
- **last** (*float*) – The last epoch for which the segment is valid.

**Returns** An SPK segment descriptor.

**Return type** 5-Element Array of floats

`spiceypy.spiceypy.spkpos` (*targ, et, ref, abcorr, obs*)

Return the position of a target body relative to an observing body, optionally corrected for light time (planetary aberration) and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkpos\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkpos_c.html)

**Parameters**

- **targ** (*str*) – Target body name.
- **et** (*Union[float, Iterable[float]]*) – Observer epoch.
- **ref** (*str*) – Reference frame of output position vector.
- **abcorr** (*str*) – Aberration correction flag.

- **obs** (*str*) – Observing body name.

**Returns** Position of target, One way light time between observer and target.

**Return type** tuple

`spiceypy.spiceypy.spkpvn` (*handle, descr, et*)

For a specified SPK segment and time, return the state (position and velocity) of the segment's target body relative to its center of motion.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkpvn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkpvn_c.html)

**Parameters**

- **handle** (*int*) – File handle.
- **descr** (*5-Element Array of floats*) – Segment descriptor.
- **et** (*float*) – Evaluation epoch.

**Returns** Segment reference frame ID code, Output state vector, Center of state.

**Return type** tuple

`spiceypy.spiceypy.spksfs` (*body, et, idlen*)

Search through loaded SPK files to find the highest-priority segment applicable to the body and time specified.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spksfs\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spksfs_c.html)

**Parameters**

- **body** (*int*) – Body ID.
- **et** (*float*) – Ephemeris time.
- **idlen** (*int*) – Length of output segment ID string.

**Returns** Handle of file containing the applicable segment, Descriptor of the applicable segment, Identifier of the applicable segment.

**Return type** tuple

`spiceypy.spiceypy.spkssb` (*targ, et, ref*)

Return the state (position and velocity) of a target body relative to the solar system barycenter.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkssb\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkssb_c.html)

**Parameters**

- **targ** (*int*) – Target body.
- **et** (*float*) – Target epoch.
- **ref** (*str*) – Target reference frame.

**Returns** State of target.

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.spksub` (*handle, descr, identin, begin, end, newh*)

Extract a subset of the data in an SPK segment into a separate segment.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spksub\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spksub_c.html)

**Parameters**

- **handle** (*int*) – Handle of source segment.
- **descr** (*5-Element Array of floats*) – Descriptor of source segment.



- **identin** (*str*) – Identifier of source segment.
- **begin** (*int*) – Beginning (initial epoch) of subset.
- **end** (*int*) – End (final epoch) of subset.
- **newh** (*int*) – Handle of new segment.

`spiceypy.spiceypy.spkuds` (*descr*)

Unpack the contents of an SPK segment descriptor.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkuds\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkuds_c.html)

**Parameters** **descr** (*5-Element Array of floats*) – An SPK segment descriptor.

**Returns** The NAIF ID code for the body of the segment, The center of motion for body, The ID code for the frame of this segment, The type of SPK segment, The first epoch for which the segment is valid, The last epoch for which the segment is valid, Beginning DAF address of the segment, Ending DAF address of the segment.

**Return type** tuple

`spiceypy.spiceypy.spkuef` (*handle*)

Unload an ephemeris file so that it will no longer be searched by the readers.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkuef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkuef_c.html)

**Parameters** **handle** (*int*) – Handle of file to be unloaded

`spiceypy.spiceypy.spkw02` (*handle, body, center, inframe, first, last, segid, intlen, n, polydg, cdata, btime*)

Write a type 2 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw02\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw02_c.html)

**Parameters**

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **intlen** (*float*) – Length of time covered by logical record.
- **n** (*int*) – Number of coefficient sets.
- **polydg** (*int*) – Chebyshev polynomial degree.
- **cdata** (*Array of floats*) – Array of Chebyshev coefficients.
- **btime** (*float*) – Begin time of first logical record.

`spiceypy.spiceypy.spkw03` (*handle, body, center, inframe, first, last, segid, intlen, n, polydg, cdata, btime*)

Write a type 3 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw03\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw03_c.html)

**Parameters**

- **handle** (*int*) – Handle of SPK file open for writing.
- **body** (*int*) – NAIF code for ephemeris object.
- **center** (*int*) – NAIF code for the center of motion of the body.
- **inframe** (*str*) – Reference frame name.
- **first** (*float*) – Start time of interval covered by segment.
- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **intlen** (*float*) – Length of time covered by record.
- **n** (*int*) – Number of records in segment.
- **polydg** (*int*) – Chebyshev polynomial degree.
- **cdata** (*Array of floats*) – Array of Chebyshev coefficients.
- **btime** (*float*) – Begin time of first record.

`spiceypy.spiceypy.spkw05` (*handle, body, center, inframe, first, last, segid, gm, n, states, epochs*)

Write an SPK segment of type 5 given a time-ordered set of discrete states and epochs, and the gravitational parameter of a central body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw05\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw05_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **gm** (*float*) – Gravitational parameter of central body.
- **n** (*int*) – Number of states and epochs.
- **states** (*Nx6-Element Array of floats*) – States.
- **epochs** (*Array of floats*) – Epochs.

`spiceypy.spiceypy.spkw08` (*handle, body, center, inframe, first, last, segid, degree, n, states, epoch1, step*)

Write a type 8 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw08\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw08_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – NAIF code for an ephemeris object.
- **center** (*int*) – NAIF code for center of motion of “body”.
- **inframe** (*str*) – Reference frame name.
- **first** (*float*) – Start time of interval covered by segment.

- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **degree** (*int*) – Degree of interpolating polynomials.
- **n** (*int*) – Number of states.
- **states** (*Nx6-Element Array of floats*) – Array of states.
- **epoch1** (*float*) – Epoch of first state in states array.
- **step** (*float*) – Time step separating epochs of states.

`spiceypy.spiceypy.spkw09` (*handle, body, center, inframe, first, last, segid, degree, n, states, epochs*)  
Write a type 9 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw09\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw09_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – NAIF code for an ephemeris object.
- **center** (*int*) – NAIF code for center of motion of “body”.
- **inframe** (*str*) – Reference frame name.
- **first** (*float*) – Start time of interval covered by segment.
- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **degree** (*int*) – Degree of interpolating polynomials.
- **n** (*int*) – Number of states.
- **states** (*Nx6-Element Array of floats*) – Array of states.
- **epochs** (*Array of floats*) – Array of epochs corresponding to states.

`spiceypy.spiceypy.spkw10` (*handle, body, center, inframe, first, last, segid, consts, n, elems, epochs*)  
Write an SPK type 10 segment to the DAF open and attached to the input handle.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw10\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw10_c.html)

#### Parameters

- **handle** (*int*) – The handle of a DAF file open for writing.
- **body** (*int*) – The NAIF ID code for the body of the segment.
- **center** (*int*) – The center of motion for body.
- **inframe** (*str*) – The reference frame for this segment.
- **first** (*float*) – The first epoch for which the segment is valid.
- **last** (*float*) – The last epoch for which the segment is valid.
- **segid** (*str*) – The string to use for segment identifier.
- **consts** (*8-Element Array of floats*) – The array of geophysical constants for the segment.
- **n** (*int*) – The number of element/epoch pairs to be stored.
- **elems** (*Array of floats*) – The collection of “two-line” element sets.

- **epochs** (*Array of floats*) – The epochs associated with the element sets.

`spiceypy.spiceypy.spkw12` (*handle, body, center, inframe, first, last, segid, degree, n, states, epoch0, step*)

Write a type 12 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw12\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw12_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – NAIF code for an ephemeris object.
- **center** (*int*) – NAIF code for center of motion of body.
- **inframe** (*str*) – Reference frame name.
- **first** (*float*) – Start time of interval covered by segment.
- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **degree** (*int*) – Degree of interpolating polynomials.
- **n** (*int*) – Number of states.
- **states** (*Nx6-Element Array of floats*) – Array of states.
- **epoch0** (*float*) – Epoch of first state in states array.
- **step** (*float*) – Time step separating epochs of states.

`spiceypy.spiceypy.spkw13` (*handle, body, center, inframe, first, last, segid, degree, n, states, epochs*)

Write a type 13 segment to an SPK file.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw13\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw13_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – NAIF code for an ephemeris object.
- **center** (*int*) – NAIF code for center of motion of body.
- **inframe** (*str*) – Reference frame name.
- **first** (*float*) – Start time of interval covered by segment.
- **last** (*float*) – End time of interval covered by segment.
- **segid** (*str*) – Segment identifier.
- **degree** (*int*) – Degree of interpolating polynomials.
- **n** (*int*) – Number of states.
- **states** (*Nx6-Element Array of floats*) – Array of states.
- **epochs** (*Array of floats*) – Array of epochs corresponding to states.

`spiceypy.spiceypy.spkw15` (*handle, body, center, inframe, first, last, segid, epoch, tp, pa, p, ecc, j2flg, pv, gm, j2, radius*)

Write an SPK segment of type 15 given a type 15 data record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw15\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw15_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **epoch** (*float*) – Epoch of the periapse.
- **tp** (*3-Element Array of floats*) – Trajectory pole vector.
- **pa** (*3-Element Array of floats*) – Periapsis vector.
- **p** (*float*) – Semi-latus rectum.
- **ecc** (*float*) – Eccentricity.
- **j2flg** (*float*) – J2 processing flag.
- **pv** (*3-Element Array of floats*) – Central body pole vector.
- **gm** (*float*) – Central body GM.
- **j2** (*float*) – Central body J2.
- **radius** (*float*) – Equatorial radius of central body.

`spiceypy.spiceypy.spkw17` (*handle, body, center, inframe, first, last, segid, epoch, eqel, rapol, decpol*)  
Write an SPK segment of type 17 given a type 17 data record.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw17\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw17_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **epoch** (*float*) – Epoch of elements in seconds past J2000.
- **eqel** (*9-Element Array of floats*) – Array of equinoctial elements.
- **rapol** (*float*) – Right Ascension of the pole of the reference plane.
- **decpol** (*float*) – Declination of the pole of the reference plane.

`spiceypy.spiceypy.spkw18` (*handle, subtyp, body, center, inframe, first, last, segid, degree, packets, epochs*)  
Write a type 18 segment to an SPK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw18\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw18_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **subtyp** (*int*) – SPK type 18 subtype code.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **degree** (*int*) – Degree of interpolating polynomials.
- **packts** (*2D Array of floats*) – data packets
- **epochs** (*N-Element Array of floats*) – Array of epochs corresponding to states.

`spiceypy.spiceypy.spkw20` (*handle, body, center, inframe, first, last, segid, intlen, n, polydg, cdata, dscale, tscale, initjd, initfr*)

Write a type 20 segment to an SPK file.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/spkw20\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/spkw20_c.html)

#### Parameters

- **handle** (*int*) – Handle of an SPK file open for writing.
- **body** (*int*) – Body code for ephemeris object.
- **center** (*int*) – Body code for the center of motion of the body.
- **inframe** (*str*) – The reference frame of the states.
- **first** (*float*) – First valid time for which states can be computed.
- **last** (*float*) – Last valid time for which states can be computed.
- **segid** (*str*) – Segment identifier.
- **intlen** – Length of time covered by logical record (days).
- **n** – Number of logical records in segment.
- **polydg** – Chebyshev polynomial degree.
- **cdata** – Array of Chebyshev coefficients and positions.
- **dscale** – Distance scale of data.
- **tscale** – Time scale of data.
- **initjd** – Integer part of begin time (TDB Julian date) of first record.
- **initfr** – Fractional part of begin time (TDB Julian date) of first record.

`spiceypy.spiceypy.srfc2s` (*code, bodyid, srflen=256*)

Translate a surface ID code, together with a body ID code, to the corresponding surface name. If no such name exists, return a string representation of the surface ID code.

note: from NAIF if `isname` is false, this case is not treated as an error.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfc2s\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfc2s_c.html)

#### Parameters

- **code** (*int*) – Integer surface ID code to translate to a string.

- **bodyid** (*int*) – ID code of body associated with surface.
- **srflen** – Available space in output string.
- **srflen** – int

**Returns** String corresponding to surface ID code.

**Return type** str

`spiceypy.spiceypy.srfcss` (*code, bodstr, srflen=256*)

Translate a surface ID code, together with a body string, to the corresponding surface name. If no such surface name exists, return a string representation of the surface ID code.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfcss\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfcss_c.html)

#### Parameters

- **code** (*int*) – Integer surface ID code to translate to a string.
- **bodstr** (*str*) – Name or ID of body associated with surface.
- **srflen** – Available space in output string.
- **srflen** – int

**Returns** String corresponding to surface ID code.

**Return type** str

`spiceypy.spiceypy.srfnrm` (*method, target, et, fixref, srfpts*)

Map array of surface points on a specified target body to the corresponding unit length outward surface normal vectors.

The surface of the target body may be represented by a triaxial ellipsoid or by topographic data provided by DSK files.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfnrm\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfnrm_c.html)

#### Parameters

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in TDB seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **srfpts** (*3xM-Element Array of floats*) – Array of surface points.

**Returns** Array of outward, unit length normal vectors.

**Return type** 3xM-Element Array of floats

`spiceypy.spiceypy.srfrec` (*body, longitude, latitude*)

Convert planetocentric latitude and longitude of a surface point on a specified body to rectangular coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfrec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfrec_c.html)

#### Parameters

- **body** (*int*) – NAIF integer code of an extended body.
- **longitude** (*float*) – Longitude of point in radians.
- **latitude** (*float*) – Latitude of point in radians.

**Returns** Rectangular coordinates of the point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.srfs2c` (*srfstr*, *bodstr*)

Translate a surface string, together with a body string, to the corresponding surface ID code. The input strings may contain names or integer ID codes.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfs2c\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfs2c_c.html)

**Parameters**

- **srfstr** (*str*) – Surface name or ID string.
- **bodstr** (*str*) – Body name or ID string.

**Returns** Integer surface ID code.

**Return type** int

`spiceypy.spiceypy.srfscc` (*srfstr*, *bodyid*)

Translate a surface string, together with a body ID code, to the corresponding surface ID code. The input surface string may contain a name or an integer ID code.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfscc\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfscc_c.html)

**Parameters**

- **srfstr** (*str*) – Surface name or ID string.
- **bodyid** (*int*) – ID code of body associated with surface.

**Returns** Integer surface ID code.

**Return type** int

`spiceypy.spiceypy.srfxpt` (*method*, *target*, *et*, *abcorr*, *obsrvr*, *dref*, *dvec*)

Deprecated: This routine has been superseded by the CSPICE routine `sincpt()`. This routine is supported for purposes of backward compatibility only.

Given an observer and a direction vector defining a ray, compute the surface intercept point of the ray on a target body at a specified epoch, optionally corrected for light time and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/srfxpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/srfxpt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*Union[float, Iterable[float]]*) – Epoch in ephemeris seconds past J2000 TDB.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.
- **dref** (*str*) – Reference frame of input direction vector.
- **dvec** (*3-Element Array of floats*) – Ray’s direction vector.

**Returns** Surface intercept point on the target body, Distance from the observer to the intercept point, Intercept epoch, Observer position relative to target center.

**Return type** tuple

`spiceypy.spiceypy.ssize` (*newsiz*, *cell*)

Set the size (maximum cardinality) of a CSPICE cell of any data type.



[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ssize\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ssize_c.html)

**Parameters**

- **newsize** (*int*) – Size (maximum cardinality) of the cell.
- **cell** (`spiceypy.utils.support_types.SpiceCell`) – The cell.

**Returns** The updated cell.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.stelab` (*pobj, vobs*)

Correct the apparent position of an object for stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/stelab\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/stelab_c.html)

**Parameters**

- **pobj** (*3-Element Array of floats*) – Position of an object with respect to the observer.
- **vobs** (*3-Element Array of floats*) – Velocity of the observer with respect to the Solar System barycenter.

**Returns** Apparent position of the object with respect to the observer, corrected for stellar aberration.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.stpool` (*item, nth, contin, lenout=256*)

Retrieve the *nth* string from the kernel pool variable, where the string may be continued across several components of the kernel pool variable.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/stpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/stpool_c.html)

**Parameters**

- **item** (*str*) – Name of the kernel pool variable.
- **nth** (*int*) – Index of the full string to retrieve.
- **contin** (*str*) – Character sequence used to indicate continuation.
- **lenout** (*int*) – Available space in output string.

**Returns** A full string concatenated across continuations, The number of characters in the full string value.

**Return type** tuple

`spiceypy.spiceypy.str2et` (*time*)

Convert a string representing an epoch to a double precision value representing the number of TDB seconds past the J2000 epoch corresponding to the input epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/str2et\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/str2et_c.html)

**Parameters** **time** (*str*) – A string representing an epoch.

**Returns** The equivalent value in seconds past J2000, TDB.

**Return type** float

`spiceypy.spiceypy.subpnt` (*method, target, et, fixref, abcorr, obsrvr*)

Compute the rectangular coordinates of the sub-observer point on a target body at a specified epoch, optionally corrected for light time and stellar aberration.

This routine supersedes `subpt()`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/subpnt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/subpnt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.

**Returns** Sub-observer point on the target body, Sub-observer point epoch, Vector from observer to sub-observer point.

**Return type** tuple

`spiceypy.spiceypy.subpt` (*method, target, et, abcorr, obsrvr*)

Deprecated: This routine has been superseded by the CSPICE routine `subpnt` (). This routine is supported for purposes of backward compatibility only.

Compute the rectangular coordinates of the sub-observer point on a target body at a particular epoch, optionally corrected for planetary (light time) and stellar aberration. Return these coordinates expressed in the body-fixed frame associated with the target body. Also, return the observer's altitude above the target body.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/subpnt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/subpnt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*Union[float, Iterable[float]]*) – Epoch in ephemeris seconds past J2000 TDB.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.

**Returns** Sub-observer point on the target body, Altitude of the observer above the target body.

**Return type** tuple

`spiceypy.spiceypy.subslr` (*method, target, et, fixref, abcorr, obsrvr*)

Compute the rectangular coordinates of the sub-solar point on a target body at a specified epoch, optionally corrected for light time and stellar aberration.

This routine supersedes `subsol_c`.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/subslr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/subslr_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.

**Returns** Sub-solar point on the target body, Sub-solar point epoch, Vector from observer to sub-solar point.

**Return type** tuple

`spiceypy.spiceypy.subsol` (*method, target, et, abcorr, obsrvr*)

Deprecated: This routine has been superseded by the CSPICE routine `subslr()`. This routine is supported for purposes of backward compatibility only.

Determine the coordinates of the sub-solar point on a target body as seen by a specified observer at a specified epoch, optionally corrected for planetary (light time) and stellar aberration.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/subsol\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/subsol_c.html)

#### Parameters

- **method** (*str*) – Computation method.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **abcorr** (*str*) – Aberration correction.
- **obsrvr** (*str*) – Name of observing body.

**Returns** Sub-solar point on the target body.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.sumad` (*array*)

Return the sum of the elements of a double precision array.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sumad\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sumad_c.html)

**Parameters** **array** (*Array of floats*) – Input Array.

**Returns** The sum of the array.

**Return type** float

`spiceypy.spiceypy.sumai` (*array*)

Return the sum of the elements of an integer array.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sumai\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sumai_c.html)

**Parameters** **array** (*Array of ints*) – Input Array.

**Returns** The sum of the array.

**Return type** int

`spiceypy.spiceypy.surfnm` (*a, b, c, point*)

This routine computes the outward-pointing, unit normal vector from a point on the surface of an ellipsoid.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/surfnm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/surfnm_c.html)

#### Parameters

- **a** (*float*) – Length of the ellipsoid semi-axis along the x-axis.
- **b** (*float*) – Length of the ellipsoid semi-axis along the y-axis.
- **c** (*float*) – Length of the ellipsoid semi-axis along the z-axis.
- **point** (*3-Element Array of floats*) – Body-fixed coordinates of a point on the ellipsoid'

**Returns** Outward pointing unit normal to ellipsoid at point.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.surfpt` (*posin, u, a, b, c*)

Determine the intersection of a line-of-sight vector with the surface of an ellipsoid.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/surfpt\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/surfpt_c.html)

**Parameters**

- **posin** (*3-Element Array of floats*) – Position of the observer in body-fixed frame.
- **u** (*3-Element Array of floats*) – Vector from the observer in some direction.
- **a** (*float*) – Length of the ellipsoid semi-axis along the x-axis.
- **b** (*float*) – Length of the ellipsoid semi-axis along the y-axis.
- **c** (*float*) – Length of the ellipsoid semi-axis along the z-axis.

**Returns** Point on the ellipsoid pointed to by **u**.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.surfpv` (*stvrtn, stdir, a, b, c*)

Find the state (position and velocity) of the surface intercept defined by a specified ray, ray velocity, and ellipsoid.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/surfpv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/surfpv_c.html)

**Parameters**

- **stvrtn** (*6-Element Array of floats*) – State of ray's vertex.
- **stdir** (*6-Element Array of floats*) – State of ray's direction vector.
- **a** (*float*) – Length of the ellipsoid semi-axis along the x-axis.
- **b** (*float*) – Length of the ellipsoid semi-axis along the y-axis.
- **c** (*float*) – Length of the ellipsoid semi-axis along the z-axis.

**Returns** State of surface intercept.

**Return type** list

`spiceypy.spiceypy.swpool` (*agent, nnames, lenvals, names*)

Add a name to the list of agents to notify whenever a member of a list of kernel variables is updated.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/swpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/swpool_c.html)

**Parameters**

- **agent** (*str*) – The name of an agent to be notified after updates.
- **nnames** (*int*) – The number of variables to associate with agent.
- **lenvals** (*int*) – Length of strings in the names array.
- **names** (*list of str.*) – Variable names whose update causes the notice.

`spiceypy.spiceypy.sxform` (*instrng, tostring, et*)

Return the state transformation matrix from one frame to another at a specified epoch.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/sxform\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/sxform_c.html)

**Parameters**

- **instrng** (*str*) – Name of the frame to transform from.

- **tostring** (*str*) – Name of the frame to transform to.
- **et** (*Union[float, Iterable[float]]*) – Epoch of the state transformation matrix.

**Returns** A state transformation matrix.

**Return type** 6x6-Element Array of floats

`spiceypy.spiceypy.szpool` (*name*)

Return the kernel pool size limitations.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/szpool\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/szpool_c.html)

**Parameters** **name** (*str*) – Name of the parameter to be returned.

**Returns** Value of parameter specified by name,

**Return type** int

`spiceypy.spiceypy.termpt` (*method, ilusrc, target, et, fixref, abcorr, corloc, obsrvr, refvec, rolstp, ncuts, schstp, soltol, maxn*)

Find limb points on a target body. The limb is the set of points of tangency on the target of rays emanating from the observer. The caller specifies half-planes bounded by the observer-target center vector in which to search for limb points.

The surface of the target body may be represented either by a triaxial ellipsoid or by topographic data.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/limbpt\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/limbpt_c.html)

**Parameters**

- **method** (*str*) – Computation method.
- **ilusrc** (*str*) – Illumination source.
- **target** (*str*) – Name of target body.
- **et** (*float*) – Epoch in ephemeris seconds past J2000 TDB.
- **fixref** (*str*) – Body-fixed, body-centered target body frame.
- **abcorr** (*str*) – Aberration correction.
- **corloc** (*str*) – Aberration correction locus.
- **obsrvr** (*str*) – Name of observing body.
- **refvec** (*3-Element Array of floats*) – Reference vector for cutting half-planes.
- **rolstp** (*float*) – Roll angular step for cutting half-planes.
- **ncuts** (*int*) – Number of cutting half-planes.
- **schstp** (*float*) – Angular step size for searching.
- **soltol** (*float*) – Solution convergence tolerance.
- **maxn** (*int*) – Maximum number of entries in output arrays.

**Returns** Counts of terminator points corresponding to cuts, Terminator points, Times associated with terminator points, Terminator vectors emanating from the observer

**Return type** tuple

`spiceypy.spiceypy.timdef` (*action, item, lenout, value=None*)

Set and retrieve the defaults associated with calendar input strings.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/timdef\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/timdef_c.html)

**Parameters**

- **action** (*str*) – the kind of action to take “SET” or “GET”.
- **item** (*str*) – the default item of interest.
- **lenout** (*int*) – the length of list for output.
- **value** (*str*) – the optional string used if action is “SET”

**Returns** the value associated with the default item.

**Return type** str

`spiceypy.spiceypy.timeout(et, pictur, lenout=256)`

This vectorized routine converts an input epoch represented in TDB seconds past the TDB epoch of J2000 to a character string formatted to the specifications of a user’s format picture.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/timeout\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/timeout_c.html)

**Parameters**

- **et** (*Union[float, Iterable[float]]*) – An epoch in seconds past the ephemeris epoch J2000.
- **pictur** (*str*) – A format specification for the output string.
- **lenout** (*int*) – The length of the output string plus 1.

**Returns** A string representation of the input epoch.

**Return type** str or array of str

`spiceypy.spiceypy.tipbod(ref, body, et)`

Return a 3x3 matrix that transforms positions in inertial coordinates to positions in body-equator-and-prime-meridian coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tipbod\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tipbod_c.html)

**Parameters**

- **ref** (*str*) – ID of inertial reference frame to transform from.
- **body** (*int*) – ID code of body.
- **et** (*float*) – Epoch of transformation.

**Returns** Transformation (position), inertial to prime meridian.

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.tisbod(ref, body, et)`

Return a 6x6 matrix that transforms states in inertial coordinates to states in body-equator-and-prime-meridian coordinates.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tisbod\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tisbod_c.html)

**Parameters**

- **ref** (*str*) – ID of inertial reference frame to transform from.
- **body** (*int*) – ID code of body.
- **et** (*float*) – Epoch of transformation.

**Returns** Transformation (state), inertial to prime meridian.

**Return type** 6x6-Element Array of floats

`spiceypy.spiceypy.tkvrnsn` (*item*)

Given an item such as the Toolkit or an entry point name, return the latest version string.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tkvrnsn\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tkvrnsn_c.html)

**Parameters** *item* (*str*) – Item for which a version string is desired.

**Returns** the latest version string.

**Return type** *str*

`spiceypy.spiceypy.tparse` (*instring*, *lenout*=256)

Parse a time string and return seconds past the J2000 epoch on a formal calendar.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tparse\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tparse_c.html)

**Parameters**

- **instring** (*str*) – Input time string, UTC.
- **lenout** (*int*) – Available space in output error message string.

**Returns** Equivalent UTC seconds past J2000, Descriptive error message.

**Return type** *tuple*

`spiceypy.spiceypy.tpicttr` (*sample*, *lenout*=256, *lenerr*=256)

Given a sample time string, create a time format picture suitable for use by the routine timout.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tpicttr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tpicttr_c.html)

**Parameters**

- **sample** (*str*) – A sample time string.
- **lenout** (*int*) – The length for the output picture string.
- **lenerr** (*int*) – The length for the output error string.

**Returns** A format picture that describes sample, Flag indicating whether sample parsed successfully, Diagnostic returned if sample cannot be parsed

**Return type** *tuple*

`spiceypy.spiceypy.trace` (*matrix*)

Return the trace of a 3x3 matrix.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/trace\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/trace_c.html)

**Parameters** *matrix* (*3x3-Element Array of floats*) – 3x3 matrix of double precision numbers.

**Returns** The trace of matrix.

**Return type** *float*

`spiceypy.spiceypy.trcdep` ()

Return the number of modules in the traceback representation.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/trcdep\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/trcdep_c.html)

**Returns** The number of modules in the traceback.

**Return type** *int*

`spiceypy.spiceypy.trcnam` (*index*, *namlen*=256)

Return the name of the module having the specified position in the trace representation. The first module to check in is at index 0.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/trcnam\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/trcnam_c.html)

**Parameters**

- **index** (*int*) – The position of the requested module name.
- **namlen** (*int*) – Available space in output name string.

**Returns** The name at position *index* in the traceback.

**Return type** *str*

`spiceypy.spiceypy.trcoff()`

Disable tracing.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/trcoff\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/trcoff_c.html)

`spiceypy.spiceypy.tsetyr(year)`

Set the lower bound on the 100 year range.

Default value is 1969

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tsetyr\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tsetyr_c.html)

**Parameters** **year** (*int*) – Lower bound on the 100 year interval of expansion

`spiceypy.spiceypy.twopi()`

Return twice the value of pi (the ratio of the circumference of a circle to its diameter).

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/twopi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/twopi_c.html)

**Returns** Twice the value of pi.

**Return type** *float*

`spiceypy.spiceypy.twovec(axdef, indexa, plndef, indexp)`

Find the transformation to the right-handed frame having a given vector as a specified axis and having a second given vector lying in a specified coordinate plane.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/twovec\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/twovec_c.html)

**Parameters**

- **axdef** (*3-Element Array of floats*) – Vector defining a principal axis.
- **indexa** (*int*) – Principal axis number of *axdef* (X=1, Y=2, Z=3).
- **plndef** (*3-Element Array of floats*) – Vector defining (with *axdef*) a principal plane.
- **indexp** (*int*) – Second axis number (with *indexa*) of principal plane.

**Returns** Output rotation matrix.

**Return type** *3x3-Element Array of floats*

`spiceypy.spiceypy.txtopn(fname)`

Internal undocumented command for opening a new text file for subsequent write access.

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ftncls\\_c.html#Files](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ftncls_c.html#Files)    [https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ftncls\\_c.html#Examples](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ftncls_c.html#Examples)

**Parameters** **fname** (*str*) – name of the new text file to be opened.

**Returns** FORTRAN logical unit of opened file

**Return type** *int*



`spiceypy.spiceypy.tyear()`

Return the number of seconds in a tropical year.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/tyear\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/tyear_c.html)

**Returns** The number of seconds in a tropical year.

**Return type** float

`spiceypy.spiceypy.ucase(inchar, lenout=None)`

Convert the characters in a string to uppercase.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ucase\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ucase_c.html)

**Parameters**

- **inchar** (*str*) – Input string.
- **lenout** (*int*) – Optional Maximum length of output string.

**Returns** Output string, all uppercase.

**Return type** str

`spiceypy.spiceypy.ucrss(v1, v2)`

Compute the normalized cross product of two 3-vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/ucrss\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/ucrss_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – Left vector for cross product.
- **v2** (*3-Element Array of floats*) – Right vector for cross product.

**Returns** Normalized cross product  $v1 \times v2 / \text{abs}(v1 \times v2)$ .

**Return type** Array of floats

`spiceypy.spiceypy.uddc(udfunc, x, dx)`

SPICE private routine intended solely for the support of SPICE routines. Users should not call this routine directly due to the volatile nature of this routine.

This routine calculates the derivative of ‘udfunc’ with respect to time for ‘et’, then determines if the derivative has a negative value.

Use the `@spiceypy.utils.callbacks.SpiceUDFUNS` decorator to wrap a given python function that takes one parameter (float) and returns a float. For example:

```
@spiceypy.utils.callbacks.SpiceUDFUNS
def udfunc(et_in):
    pos, new_et = spice.spkpos("MERCURY", et_in, "J2000", "LT+S", "MOON")
    return new_et

deriv = spice.uddf(udfunc, et, 1.0)
```

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/uddc\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/uddc_c.html)

**Parameters**

- **udfunc** (*ctypes.CFunctionType*) – Name of the routine that computes the scalar value of interest.
- **x** (*float*) – Independent variable of ‘udfunc’.
- **dx** (*float*) – Interval from ‘x’ for derivative calculation.

**Returns** Boolean indicating if the derivative is negative.

**Return type** *bool*

`spiceypy.spiceypy.uddf (udfunc, x, dx)`

Routine to calculate the first derivative of a caller-specified function using a three-point estimation.

Use the `@spiceypy.utils.callbacks.SpiceUDFUNS` decorator to wrap a given python function that takes one parameter (float) and returns a float. For example:

```
@spiceypy.utils.callbacks.SpiceUDFUNS
def udfunc(et_in):
    pos, new_et = spice.spkpos("MERCURY", et_in, "J2000", "LT+S", "MOON")
    return new_et

deriv = spice.uddf(udfunc, et, 1.0)
```

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/uddf\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/uddf_c.html)

#### Parameters

- **udfunc** (*ctypes.CFunctionType*) – Name of the routine that computes the scalar value of interest.
- **x** (*float*) – Independent variable of ‘udfunc’.
- **dx** (*float*) – Interval from ‘x’ for derivative calculation.

**Returns** Approximate derivative of ‘udfunc’ at ‘x’

**Return type** *float*

`spiceypy.spiceypy.udf (x)`

No-op routine for with an argument signature matching udfuns. Always returns 0.0 .

[https://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/udf\\_c.html](https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/udf_c.html)

**Parameters** **x** (*float*) – Double precision value, unused.

**Returns** Double precision value, unused.

**Return type** *float*

`spiceypy.spiceypy.union (a, b)`

Compute the union of two sets of any data type to form a third set.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/union\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/union_c.html)

#### Parameters

- **a** (`spiceypy.utils.support_types.SpiceCell`) – First input set.
- **b** (`spiceypy.utils.support_types.SpiceCell`) – Second input set.

**Returns** Union of a and b.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.unitim (epoch, insys, outsys)`

Transform time from one uniform scale to another. The uniform time scales are TAI, TDT, TDB, ET, JED, JDTDB, JDTDT.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/unitim\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/unitim_c.html)

#### Parameters

- **epoch** (*float*) – An epoch to be converted.

- **insys** (*str*) – The time scale associated with the input epoch.
- **outsys** (*str*) – The time scale associated with the function value.

**Returns** The float in outsys that is equivalent to the epoch on the insys time scale.

**Return type** float

`spiceypy.spiceypy.unload(filename)`  
 Unload a SPICE kernel.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/unload\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/unload_c.html)

**Parameters** **filename** (*str*) – The name of a kernel to unload.

`spiceypy.spiceypy.unorm(v1)`  
 Normalize a double precision 3-vector and return its magnitude.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/unorm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/unorm_c.html)

**Parameters** **v1** (*3-Element Array of floats*) – Vector to be normalized.

**Returns** Unit vector of v1, Magnitude of v1.

**Return type** tuple

`spiceypy.spiceypy.unormg(v1, ndim)`  
 Normalize a double precision vector of arbitrary dimension and return its magnitude.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/unormg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/unormg_c.html)

**Parameters**

- **v1** (*Array of floats*) – Vector to be normalized.
- **ndim** (*int*) – This is the dimension of v1 and vout.

**Returns** Unit vector of v1, Magnitude of v1.

**Return type** tuple

`spiceypy.spiceypy.utc2et(utcstr)`  
 Convert an input time from Calendar or Julian Date format, UTC, to ephemeris seconds past J2000.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/utc2et\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/utc2et_c.html)

**Parameters** **utcstr** (*str*) – Input time string, UTC.

**Returns** Output epoch, ephemeris seconds past J2000.

**Return type** float

`spiceypy.spiceypy.vadd(v1, v2)`  
 Add two 3 dimensional vectors. [http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vadd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vadd_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector to be added.
- **v2** (*3-Element Array of floats*) – Second vector to be added.

**Returns** v1+v2

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vaddg(v1, v2, ndim)`  
 Add two n-dimensional vectors [http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vaddg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vaddg_c.html)

**Parameters**

- **v1** (*list [ndim]*) – First vector to be added.
- **v2** (*list [ndim]*) – Second vector to be added.
- **ndim** (*int*) – Dimension of v1 and v2.

**Returns** v1+v2

**Return type** list[ndim]

`spiceypy.spiceypy.valid` (*insize, n, inset*)

Create a valid CSPICE set from a CSPICE Cell of any data type.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/valid\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/valid_c.html)

**Parameters**

- **insize** (*int*) – Size (maximum cardinality) of the set.
- **n** (*int*) – Initial no. of (possibly non-distinct) elements.
- **inset** – Set to be validated.

**Returns** validated set

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.vcrss` (*v1, v2*)

Compute the cross product of two 3-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vcrss\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vcrss_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – Left hand vector for cross product.
- **v2** (*3-Element Array of floats*) – Right hand vector for cross product.

**Returns** Cross product v1 x v2.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vdist` (*v1, v2*)

Return the distance between two three-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vdist\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vdist_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector in the dot product.
- **v2** (*3-Element Array of floats*) – Second vector in the dot product.

**Returns** the distance between v1 and v2

**Return type** float

`spiceypy.spiceypy.vdistg` (*v1, v2, ndim*)

Return the distance between two vectors of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vdistg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vdistg_c.html)

**Parameters**

- **v1** (*list [ndim]*) – ndim-dimensional double precision vector.
- **v2** (*list [ndim]*) – ndim-dimensional double precision vector.
- **ndim** (*int*) – Dimension of v1 and v2.

**Returns** the distance between  $v1$  and  $v2$

**Return type** float

`spiceypy.spiceypy.vdot(v1, v2)`

Compute the dot product of two double precision, 3-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vdot\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vdot_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector in the dot product.
- **v2** (*3-Element Array of floats*) – Second vector in the dot product.

**Returns** dot product of  $v1$  and  $v2$ .

**Return type** float

`spiceypy.spiceypy.vdotg(v1, v2, ndim)`

Compute the dot product of two double precision vectors of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vdotg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vdotg_c.html)

**Parameters**

- **v1** (*list[ndim]*) – First vector in the dot product.
- **v2** (*list[ndim]*) – Second vector in the dot product.
- **ndim** (*int*) – Dimension of  $v1$  and  $v2$ .

**Returns** dot product of  $v1$  and  $v2$ .

**Return type** float

`spiceypy.spiceypy.vequ(v1)`

Make one double precision 3-dimensional vector equal to another.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vequ\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vequ_c.html)

**Parameters** **v1** (*3-Element Array of floats*) – 3-dimensional double precision vector.

**Returns** 3-dimensional double precision vector set equal to  $vin$ .

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vequg(v1, ndim)`

Make one double precision vector of arbitrary dimension equal to another.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vequg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vequg_c.html)

**Parameters**

- **v1** (*list[ndim]*) –  $ndim$ -dimensional double precision vector.
- **ndim** (*int*) – Dimension of  $vin$  (and also  $vout$ ).

**Returns**  $ndim$ -dimensional double precision vector set equal to  $vin$ .

**Return type** list[ndim]

`spiceypy.spiceypy.vhat(v1)`

Find the unit vector along a double precision 3-dimensional vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vhat\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vhat_c.html)

**Parameters** **v1** (*3-Element Array of floats*) – Vector to be unitized.

**Returns** Unit vector  $v / \text{abs}(v)$ .

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vhatg(v1, ndim)`

Find the unit vector along a double precision vector of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vhatg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vhatg_c.html)

**Parameters**

- **v1** (*list [ndim]*) – Vector to be normalized.
- **ndim** (*int*) – Dimension of v1 (and also vout).

**Returns** Unit vector  $v / \text{abs}(v)$ .

**Return type** list[ndim]

`spiceypy.spiceypy.vlcom(a, v1, b, v2)`

Compute a vector linear combination of two double precision, 3-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vlcom\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vlcom_c.html)

**Parameters**

- **a** (*float*) – Coefficient of v1
- **v1** (*3-Element Array of floats*) – Vector in 3-space
- **b** (*float*) – Coefficient of v2
- **v2** (*3-Element Array of floats*) – Vector in 3-space

**Returns** Linear Vector Combination  $a*v1 + b*v2$ .

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vlcom3(a, v1, b, v2, c, v3)`

This subroutine computes the vector linear combination  $a*v1 + b*v2 + c*v3$  of double precision, 3-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vlcom3\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vlcom3_c.html)

**Parameters**

- **a** (*float*) – Coefficient of v1
- **v1** (*3-Element Array of floats*) – Vector in 3-space
- **b** (*float*) – Coefficient of v2
- **v2** (*3-Element Array of floats*) – Vector in 3-space
- **c** (*float*) – Coefficient of v3
- **v3** (*3-Element Array of floats*) – Vector in 3-space

**Returns** Linear Vector Combination  $a*v1 + b*v2 + c*v3$

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vlcomg(n, a, v1, b, v2)`

Compute a vector linear combination of two double precision vectors of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vlcomg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vlcomg_c.html)

**Parameters**

- **n** (*int*) – Dimension of vector space
- **a** (*float*) – Coefficient of v1

- **v1** (*list[n]*) – Vector in n-space
- **b** (*float*) – Coefficient of v2
- **v2** (*list[n]*) – Vector in n-space

**Returns** Linear Vector Combination  $a*v1 + b*v2$

**Return type** list[n]

`spiceypy.spiceypy.vminug(vin, ndim)`

Negate a double precision vector of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vminug\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vminug_c.html)

**Parameters**

- **vin** (*Array of floats*) – ndim-dimensional double precision vector to be negated.
- **ndim** (*int*) – Dimension of vin.

**Returns** ndim-dimensional double precision vector equal to -vin.

**Return type** list[ndim]

`spiceypy.spiceypy.vminus(vin)`

Negate a double precision 3-dimensional vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vminus\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vminus_c.html)

**Parameters** **vin** (*3-Element Array of floats*) – Vector to be negated.

**Returns** Negated vector -v1.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vnorm(v)`

Compute the magnitude of a double precision, 3-dimensional vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vnorm\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vnorm_c.html)

**Parameters** **v** (*3-Element Array of floats*) – Vector whose magnitude is to be found.

**Returns** magnitude of v calculated in a numerically stable way

**Return type** float

`spiceypy.spiceypy.vnormg(v, ndim)`

Compute the magnitude of a double precision vector of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vnormg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vnormg_c.html)

**Parameters**

- **v** (*Array of floats*) – Vector whose magnitude is to be found.
- **ndim** (*int*) – Dimension of v

**Returns** magnitude of v calculated in a numerically stable way

**Return type** float

`spiceypy.spiceypy.vpack(x, y, z)`

Pack three scalar components into a vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vpack\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vpack_c.html)

**Parameters**

- **x** (*float*) – first scalar component

- **y** (*float*) – second scalar component
- **z** (*float*) – third scalar component

**Returns** Equivalent 3-vector.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vperp(a, b)`

Find the component of a vector that is perpendicular to a second vector. All vectors are 3-dimensional.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vperp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vperp_c.html)

**Parameters**

- **a** (*3-Element Array of floats*) – The vector whose orthogonal component is sought.
- **b** (*3-Element Array of floats*) – The vector used as the orthogonal reference.

**Returns** The component of a orthogonal to b.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vprjp(vin, plane)`

Project a vector onto a specified plane, orthogonally.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vprjp\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vprjp_c.html)

**Parameters**

- **vin** (*3-Element Array of floats*) – The projected vector.
- **plane** (`spiceypy.utils.support_types.Plane`) – Plane containing vin.

**Returns** Vector resulting from projection.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vprjpi(vin, projpl, invpl)`

Find the vector in a specified plane that maps to a specified vector in another plane under orthogonal projection.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vprjpi\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vprjpi_c.html)

**Parameters**

- **vin** (*3-Element Array of floats*) – The projected vector.
- **projpl** (`spiceypy.utils.support_types.Plane`) – Plane containing vin.
- **invpl** (`spiceypy.utils.support_types.Plane`) – Plane containing inverse image of vin.

**Returns** Inverse projection of vin.

**Return type** list

`spiceypy.spiceypy.vproj(a, b)`

Find the projection of one vector onto another vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vproj\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vproj_c.html)

**Parameters**

- **a** (*3-Element Array of floats*) – The vector to be projected.
- **b** (*3-Element Array of floats*) – The vector onto which a is to be projected.

**Returns** The projection of a onto b.



**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vrel(v1, v2)`

Return the relative difference between two 3-dimensional vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vrel\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vrel_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector
- **v2** (*3-Element Array of floats*) – Second vector

**Returns** the relative difference between v1 and v2.

**Return type** float

`spiceypy.spiceypy.vrelg(v1, v2, ndim)`

Return the relative difference between two vectors of general dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vrelg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vrelg_c.html)

**Parameters**

- **v1** (*Array of floats*) – First vector
- **v2** (*Array of floats*) – Second vector
- **ndim** (*int*) – Dimension of v1 and v2.

**Returns** the relative difference between v1 and v2.

**Return type** float

`spiceypy.spiceypy.vrotv(v, axis, theta)`

Rotate a vector about a specified axis vector by a specified angle and return the rotated vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vrotv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vrotv_c.html)

**Parameters**

- **v** (*3-Element Array of floats*) – Vector to be rotated.
- **axis** (*3-Element Array of floats*) – Axis of the rotation.
- **theta** (*float*) – Angle of rotation (radians).

**Returns** Result of rotating v about axis by theta

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vsc1(s, v1)`

Multiply a scalar and a 3-dimensional double precision vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsc1\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsc1_c.html)

**Parameters**

- **s** (*float*) – Scalar to multiply a vector
- **v1** (*3-Element Array of floats*) – Vector to be multiplied

**Returns** Product vector, s\*v1.

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vsc1g(s, v1, ndim)`

Multiply a scalar and a double precision vector of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsc1g\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsc1g_c.html)

**Parameters**

- **s** (*float*) – Scalar to multiply a vector
- **v1** (*Array of floats*) – Vector to be multiplied
- **ndim** (*int*) – Dimension of v1

**Returns** Product vector,  $s*v1$ .

**Return type** Array of floats

`spiceypy.spiceypy.vsep(v1, v2)`

Find the separation angle in radians between two double precision, 3-dimensional vectors. This angle is defined as zero if either vector is zero.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsep\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsep_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector
- **v2** (*3-Element Array of floats*) – Second vector

**Returns** separation angle in radians

**Return type** float

`spiceypy.spiceypy.vsepg(v1, v2, ndim)`

Find the separation angle in radians between two double precision vectors of arbitrary dimension. This angle is defined as zero if either vector is zero.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsepg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsepg_c.html)

**Parameters**

- **v1** (*Array of floats*) – First vector
- **v2** (*Array of floats*) – Second vector
- **ndim** (*int*) – The number of elements in v1 and v2.

**Returns** separation angle in radians

**Return type** float

`spiceypy.spiceypy.vsub(v1, v2)`

Compute the difference between two 3-dimensional, double precision vectors.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsub\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsub_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – First vector (minuend).
- **v2** (*3-Element Array of floats*) – Second vector (subtrahend).

**Returns** Difference vector,  $v1 - v2$ .

**Return type** 3-Element Array of floats

`spiceypy.spiceypy.vsubg(v1, v2, ndim)`

Compute the difference between two double precision vectors of arbitrary dimension.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vsubg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vsubg_c.html)

**Parameters**

- **v1** (*Array of floats*) – First vector (minuend).

- **v2** (*Array of floats*) – Second vector (subtrahend).
- **ndim** (*int*) – Dimension of v1, v2, and vout.

**Returns** Difference vector,  $v1 - v2$ .

**Return type** Array of floats

`spiceypy.spiceypy.vtmv(v1, matrix, v2)`

Multiply the transpose of a 3-dimensional column vector a 3x3 matrix, and a 3-dimensional column vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vtmv\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vtmv_c.html)

**Parameters**

- **v1** (*3-Element Array of floats*) – 3 dimensional double precision column vector.
- **matrix** (*3x3-Element Array of floats*) – 3x3 double precision matrix.
- **v2** (*3-Element Array of floats*) – 3 dimensional double precision column vector.

**Returns** the result of  $(v1^{**t} * matrix * v2)$ .

**Return type** float

`spiceypy.spiceypy.vtmvg(v1, matrix, v2, nrow, ncol)`

Multiply the transpose of a n-dimensional column vector a nxm matrix, and a m-dimensional column vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vtmvg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vtmvg_c.html)

**Parameters**

- **v1** (*Array of floats*) – n-dimensional double precision column vector.
- **matrix** (*NxM-Element Array of floats*) – nxm double precision matrix.
- **v2** (*Array of floats*) – m-dimensional double precision column vector.
- **nrow** (*int*) – Number of rows in matrix (number of rows in v1.)
- **ncol** (*int*) – Number of columns in matrix (number of rows in v2.)

**Returns** the result of  $(v1^{**t} * matrix * v2)$

**Return type** float

`spiceypy.spiceypy.vupack(v)`

Unpack three scalar components from a vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vupack\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vupack_c.html)

**Parameters** **v** (*3-Element Array of floats*) – Vector

**Returns** (x, y, z)

**Return type** tuple

`spiceypy.spiceypy.vzero(v)`

Indicate whether a 3-vector is the zero vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vzero\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vzero_c.html)

**Parameters** **v** (*3-Element Array of floats*) – Vector to be tested

**Returns** true if and only if v is the zero vector

**Return type** *bool*

`spiceypy.spiceypy.vzerog(v, ndim)`

Indicate whether a general-dimensional vector is the zero vector.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/vzerog\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/vzerog_c.html)

**Parameters**

- **v** (*Array of floats*) – Vector to be tested
- **ndim** (*int*) – Dimension of v

**Returns** true if and only if v is the zero vector

**Return type** *bool*

`spiceypy.spiceypy.wncard(window)`

Return the cardinality (number of intervals) of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wncard\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wncard_c.html)

**Parameters** **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window

**Returns** the cardinality of the input window.

**Return type** *int*

`spiceypy.spiceypy.wncomd(left, right, window)`

Determine the complement of a double precision window with respect to a specified interval.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wncomd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wncomd_c.html)

**Parameters**

- **left** (*float*) – left endpoints of complement interval.
- **right** (*float*) – right endpoints of complement interval.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window

**Returns** Complement of window with respect to left and right.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.wncond(left, right, window)`

Contract each of the intervals of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wncond\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wncond_c.html)

**Parameters**

- **left** (*float*) – Amount added to each left endpoint.
- **right** (*float*) – Amount subtracted from each right endpoint.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Window to be contracted

**Returns** Contracted Window.

**Return type** *spiceypy.utils.support\_types.SpiceCell*

`spiceypy.spiceypy.wndifd(a, b)`

Place the difference of two double precision windows into a third window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wndifd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wndifd_c.html)

**Parameters**

- **a** (`spiceypy.utils.support_types.SpiceCell`) – Input window A.

- **b** (`spiceypy.utils.support_types.SpiceCell`) – Input window B.

**Returns** Difference of a and b.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnelmd` (*point, window*)

Determine whether a point is an element of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnelmd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnelmd_c.html)

**Parameters**

- **point** (*float*) – Input point.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window

**Returns** returns True if point is an element of window.

**Return type** `bool`

`spiceypy.spiceypy.wnexpd` (*left, right, window*)

Expand each of the intervals of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnexpd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnexpd_c.html)

**Parameters**

- **left** (*float*) – Amount subtracted from each left endpoint.
- **right** (*float*) – Amount added to each right endpoint.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Window to be expanded.

**Returns** Expanded Window.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnextd` (*side, window*)

Extract the left or right endpoints from a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnextd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnextd_c.html)

**Parameters**

- **side** (*str*) – Extract left “L” or right “R” endpoints.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Window to be extracted.

**Returns** Extracted Window.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnfetd` (*window, n*)

Fetch a particular interval from a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnfetd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnfetd_c.html)

**Parameters**

- **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window
- **n** (*int*) – Index of interval to be fetched.

**Returns** Left, right endpoints of the nth interval.

**Return type** tuple

`spiceypy.spiceypy.wnfield` (*small, window*)

Fill small gaps between adjacent intervals of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnfield\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnfield_c.html)

**Parameters**

- **small** (*float*) – Limiting measure of small gaps.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Window to be filled

**Returns** Filled Window.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnfltd` (*small, window*)

Filter (remove) small intervals from a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnfltd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnfltd_c.html)

**Parameters**

- **small** (*float*) – Limiting measure of small intervals.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Window to be filtered.

**Returns** Filtered Window.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnincd` (*left, right, window*)

Determine whether an interval is included in a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnincd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnincd_c.html)

**Parameters**

- **left** (*float*) – Left interval
- **right** (*float*) – Right interval
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window

**Returns** Returns True if the input interval is included in window.

**Return type** `bool`

`spiceypy.spiceypy.wninsd` (*left, right, window*)

Insert an interval into a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wninsd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wninsd_c.html)

**Parameters**

- **left** (*float*) – Left endpoints of new interval.
- **right** (*float*) – Right endpoints of new interval.
- **window** (`spiceypy.utils.support_types.SpiceCell`) – Input window.

`spiceypy.spiceypy.wnintd` (*a, b*)

Place the intersection of two double precision windows into a third window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnintd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnintd_c.html)

**Parameters**

- **a** (`spiceypy.utils.support_types.SpiceCell`) – Input window A.

- `b` (`spiceypy.utils.support_types.SpiceCell`) – Input window B.

**Returns** Intersection of a and b.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnreld(a, op, b)`

Compare two double precision windows.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnreld\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnreld_c.html)

**Parameters**

- `a` (`spiceypy.utils.support_types.SpiceCell`) – First window.
- `op` (`str`) – Comparison operator.
- `b` (`spiceypy.utils.support_types.SpiceCell`) – Second window.

**Returns** The result of comparison: `a (op) b`.

**Return type** `bool`

`spiceypy.spiceypy.wnsumd(window)`

Summarize the contents of a double precision window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnsumd\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnsumd_c.html)

**Parameters** `window` (`spiceypy.utils.support_types.SpiceCell`) – Window to be summarized.

**Returns** Total measure of intervals in window, Average measure, Standard deviation, Location of shortest interval, Location of longest interval.

**Return type** tuple

`spiceypy.spiceypy.wnunid(a, b)`

Place the union of two double precision windows into a third window.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnunid\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnunid_c.html)

**Parameters**

- `a` (`spiceypy.utils.support_types.SpiceCell`) – Input window A.
- `b` (`spiceypy.utils.support_types.SpiceCell`) – Input window B.

**Returns** Union of a and b.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.wnvald(insize, n, window)`

Form a valid double precision window from the contents of a window array.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/wnvald\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/wnvald_c.html)

**Parameters**

- `insize` (`int`) – Size of window.
- `n` (`int`) – Original number of endpoints.
- `window` (`spiceypy.utils.support_types.SpiceCell`) – Input window.

**Returns** The union of the intervals in the input cell.

**Return type** `spiceypy.utils.support_types.SpiceCell`

`spiceypy.spiceypy.writln` (*line, unit*)

Internal undocumented command for writing a text line to a logical unit

No URL available; relevant lines from SPICE source:

FORTRAN SPICE, writln.f:

```
C$Procedure      WRITLN ( Write a text line to a logical unit )
  SUBROUTINE WRITLN ( LINE, UNIT )
  CHARACTER*(*)  LINE
  INTEGER        UNIT

C   Variable  I/O  Description
C   -----  ---  -----
C   LINE      I   The line which is to be written to UNIT.
C   UNIT      I   The Fortran unit number to use for output.
```

CSPICE, writln.c:

```
/* $Procedure      WRITLN ( Write a text line to a logical unit ) */
/* Subroutine */ int writln_(char *line, integer *unit, ftnlen line_len)
```

#### Parameters

- **line** (*str*) – The line which is to be written to UNIT.
- **unit** (*int*) – The Fortran unit number to use for output.

`spiceypy.spiceypy.xf2eul` (*xform, axisa, axisb, axisc*)

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xf2eul\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xf2eul_c.html)

#### Parameters

- **xform** (*list[6][6]*) – state transformation matrix
- **axisa** (*int*) – Axis A of the Euler angle factorization.
- **axisb** (*int*) – Axis B of the Euler angle factorization.
- **axisc** (*int*) – Axis C of the Euler angle factorization.

**Returns** (eulang, unique)

**Return type** tuple

`spiceypy.spiceypy.xf2rav` (*xform*)

This routine determines the rotation matrix and angular velocity of the rotation from a state transformation matrix. [http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xf2rav\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xf2rav_c.html)

**Parameters** **xform** (*list[6][6]*) – state transformation matrix

**Returns** rotation associated with xform, angular velocity associated with xform.

**Return type** tuple

`spiceypy.spiceypy.xfmsta` (*input\_state, input\_coord\_sys, output\_coord\_sys, body*)

Transform a state between coordinate systems.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xfmsta\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xfmsta_c.html)

#### Parameters

- **input\_state** (*6-Element Array of floats*) – Input state.
- **input\_coord\_sys** (*str*) – Current (input) coordinate system.



- **output\_coord\_sys** (*str*) – Desired (output) coordinate system.
- **body** (*str*) – Name or NAIF ID of body with which coordinates are associated (if applicable).

**Returns** Converted output state

**Return type** 6-Element Array of floats

`spiceypy.spiceypy.xpose(m)`

Transpose a 3x3 matrix

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xpose\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xpose_c.html)

**Parameters** *m* (*3x3-Element Array of floats*) – Matrix to be transposed

**Returns** Transposed matrix

**Return type** 3x3-Element Array of floats

`spiceypy.spiceypy.xpose6(m)`

Transpose a 6x6 matrix

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xpose6\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xpose6_c.html)

**Parameters** *m* (*list[6][6]*) – Matrix to be transposed

**Returns** Transposed matrix

**Return type** list[6][6]

`spiceypy.spiceypy.xposeg(matrix, nrow, ncol)`

Transpose a matrix of arbitrary size in place, the matrix need not be square.

[http://naif.jpl.nasa.gov/pub/naif/toolkit\\_docs/C/cspice/xposeg\\_c.html](http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/cspice/xposeg_c.html)

**Parameters**

- **matrix** (*NxM-Element Array of floats*) – Matrix to be transposed
- **nrow** (*int*) – Number of rows of input matrix.
- **ncol** (*int*) – Number of columns of input matrix

**Returns** Transposed matrix

**Return type** NxM-Element Array of floats

## 6.2 spiceypy.utils.support\_types module

The MIT License (MIT)

Copyright (c) [2015-2017] [Andrew Annex]

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT

HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The MIT License (MIT)

Copyright (c) 2013 Philipp Rasch

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

```
class spiceypy.utils.support_types.BoolArrayType
```

```
    Bases: object
```

```
        from_list (param)
```

```
        from_ndarray (param)
```

```
        from_param (param)
```

```
        from_tuple (param)
```

```
class spiceypy.utils.support_types.DataType
```

```
    Bases: object
```

```
    BOOL = 4
```

```
    CHR = 0
```

```
    DP = 1
```

```
    INT = 2
```

```
    SPICE_BOOL = 4
```

```
    SPICE_CHR = 0
```

```
    SPICE_DP = 1
```

```
    SPICE_INT = 2
```

```
    SPICE_TIME = 3
```

```
    TIME = 3
```

```
class spiceypy.utils.support_types.DoubleArrayType
```

```
    Bases: object
```

```
        from_array (param)
```

```
        from_list (param)
```

```
        from_ndarray (param)
```

```
        from_param (param)
```

```

    from_tuple (param)
class spiceypy.utils.support_types.DoubleMatrixType
    Bases: object
    from_list (param)
    from_matrix (param)
    from_ndarray (param)
    from_param (param)
    from_tuple (param)
class spiceypy.utils.support_types.Ellipse
    Bases: _ctypes.Structure
    center
    semi_major
    semi_minor
class spiceypy.utils.support_types.IntArrayType
    Bases: object
    from_array (param)
    from_list (param)
    from_ndarray (param)
    from_param (param)
    from_tuple (param)
class spiceypy.utils.support_types.IntMatrixType
    Bases: object
    from_list (param)
    from_matrix (param)
    from_ndarray (param)
    from_param (param)
    from_tuple (param)
class spiceypy.utils.support_types.Plane
    Bases: _ctypes.Structure
    constant
    normal
spiceypy.utils.support_types.SPICEBOOL_CELL (size)
spiceypy.utils.support_types.SPICECHAR_CELL (size, length)
spiceypy.utils.support_types.SPICEDOUBLE_CELL (size)
spiceypy.utils.support_types.SPICEINT_CELL (size)
spiceypy.utils.support_types.SPICETIME_CELL (size)

```

```

class spiceypy.utils.support_types.SpiceCell (dtype=None,      length=None,      size=None,
                                             card=None,        isSet=None,       base=None,
                                             data=None)

    Bases: ctypes.Structure

    CTRLBLOCK = 6

    DATATYPES_ENUM = {'char': 0, 'bool': 4, 'int': 2, 'time': 3, 'double': 1}

    DATATYPES_GET = [<function _char_getter>, <function _double_getter>, <function _int_getter>, <function _int_getter>]

    adjust
        Structure/Union member

    base
        Structure/Union member

    baseSize = 6

    classmethod bool (size)

    card
        Structure/Union member

    classmethod character (size, length)

    data
        Structure/Union member

    classmethod double (size)

    dtype
        Structure/Union member

    init
        Structure/Union member

    classmethod integer (size)

    isSet
        Structure/Union member

    is_bool ()

    is_char ()

    is_double ()

    is_int ()

    is_set ()

    is_time ()

    length
        Structure/Union member

    minCharLen = 6

    reset ()

    size
        Structure/Union member

    classmethod time (size)

class spiceypy.utils.support_types.SpiceDLADescr
    Bases: ctypes.Structure

```

**bwdptr**

**cbase**

**csize**

**dbase**

**dsize**

**fwdptr**

**ibase**

**isize**

**class** `spiceypy.utils.support_types.SpiceDSKDescr`  
Bases: `_ctypes.Structure`

**center**

**colmax**

**colmin**

**co2max**

**co2min**

**co3max**

**co3min**

**corpar**

**corsys**

**dclass**

**dtype**

**frmcde**

**start**

**stop**

**surfce**

**class** `spiceypy.utils.support_types.SpiceEKAttDsc`  
Bases: `_ctypes.Structure`

**cclass**

**dtype**

**indexd**

**nullok**

**size**

**strlen**

**class** `spiceypy.utils.support_types.SpiceEKDataType`  
Bases: `ctypes.c_int`

**SPICE\_BOOL = 4**

**SPICE\_CHR = 0**

**SPICE\_DP = 1**

**SPICE\_INT = 2**

**SPICE\_TIME = 3**

**class** spiceypy.utils.support\_types.**SpiceEKExprClass**

Bases: ctypes.c\_int

**SPICE\_EK\_EXP\_COL = 0**

**SPICE\_EK\_EXP\_EXPR = 2**

**SPICE\_EK\_EXP\_FUNC = 1**

**class** spiceypy.utils.support\_types.**SpiceEKSegSum**

Bases: \_ctypes.Structure

**cdescrs**

**cnames**

**ncols**

**nrows**

**tabnam**

**class** spiceypy.utils.support\_types.**SpiceSPK18Subtype**

Bases: ctypes.c\_int

**S18TP0 = 0**

**S18TP1 = 1**

**exception** spiceypy.utils.support\_types.**SpiceyError** (*value, found=None*)

Bases: Exception

SpiceyError wraps CSPICE errors. :type value: str

spiceypy.utils.support\_types.**cMatrixToNumpy** (*x*)

Convert a ctypes 2d array (or matrix) into a numpy array for python use :param x: thing to convert :return: numpy.ndarray

spiceypy.utils.support\_types.**cVectorToPython** (*x*)

Convert the c vector data into the correct python data type (numpy arrays or strings) :param x: :return:

spiceypy.utils.support\_types.**emptyBoolVector** (*n*)

spiceypy.utils.support\_types.**emptyCharArray** (*xLen=None, yLen=None*)

spiceypy.utils.support\_types.**emptyDoubleMatrix** (*x=3, y=3*)

spiceypy.utils.support\_types.**emptyDoubleVector** (*n*)

spiceypy.utils.support\_types.**emptyIntMatrix** (*x=3, y=3*)

spiceypy.utils.support\_types.**emptyIntVector** (*n*)

spiceypy.utils.support\_types.**emptySpiceEKDataTypeVector** (*n*)

spiceypy.utils.support\_types.**emptySpiceEKExprClassVector** (*n*)

spiceypy.utils.support\_types.**listToCharArray** (*inList, xLen=None, yLen=None*)

spiceypy.utils.support\_types.**listToCharArrayPtr** (*inList, xLen=None, yLen=None*)

spiceypy.utils.support\_types.**stringToCharP** (*inobject, inlen=None*)

**Parameters**

- **inobject** – input string, int for getting null string of length of int
- **inlen** – optional parameter, length of a given string can be specified

**Returns**

`spiceypy.utils.support_types.toBoolVector(x)`

`spiceypy.utils.support_types.toDoubleMatrix(x)`

`spiceypy.utils.support_types.toDoubleVector(x)`

`spiceypy.utils.support_types.toIntMatrix(x)`

`spiceypy.utils.support_types.toIntVector(x)`

`spiceypy.utils.support_types.toPythonString(inString)`

## 6.3 spiceypy.utils.libspice module

The MIT License (MIT)

Copyright (c) [2015-2017] [Andrew Annex]

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.





## CHAPTER 7

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



**S**

`spiceypy.spiceypy`, 15  
`spiceypy.utils.libspicehelper`, 163  
`spiceypy.utils.support_types`, 157



**A**

adjust (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 appndc() (in module spiceypy.spiceypy), 15  
 appnnd() (in module spiceypy.spiceypy), 15  
 appndi() (in module spiceypy.spiceypy), 16  
 axisar() (in module spiceypy.spiceypy), 16

**B**

b1900() (in module spiceypy.spiceypy), 16  
 b1950() (in module spiceypy.spiceypy), 16  
 badkpv() (in module spiceypy.spiceypy), 16  
 base (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 baseSize (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 bltfrm() (in module spiceypy.spiceypy), 17  
 bodc2n() (in module spiceypy.spiceypy), 17  
 bodc2s() (in module spiceypy.spiceypy), 17  
 boddef() (in module spiceypy.spiceypy), 17  
 bodfnd() (in module spiceypy.spiceypy), 17  
 bodn2c() (in module spiceypy.spiceypy), 18  
 bods2c() (in module spiceypy.spiceypy), 18  
 bodvar() (in module spiceypy.spiceypy), 18  
 bodvcd() (in module spiceypy.spiceypy), 18  
 bodvrd() (in module spiceypy.spiceypy), 18  
 BOOL (spiceypy.utils.support\_types.DataType attribute), 158  
 bool() (spiceypy.utils.support\_types.SpiceCell class method), 160  
 BoolArrayType (class in spiceypy.utils.support\_types), 158  
 brcktd() (in module spiceypy.spiceypy), 19  
 brckti() (in module spiceypy.spiceypy), 19  
 bschoc() (in module spiceypy.spiceypy), 19  
 bschoi() (in module spiceypy.spiceypy), 20  
 bsrhc() (in module spiceypy.spiceypy), 20  
 bsrhd() (in module spiceypy.spiceypy), 20  
 bsrchi() (in module spiceypy.spiceypy), 20

bwdptr (spiceypy.utils.support\_types.SpiceDLADescr attribute), 160

**C**

card (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 card() (in module spiceypy.spiceypy), 21  
 cbase (spiceypy.utils.support\_types.SpiceDLADescr attribute), 161  
 ccifrm() (in module spiceypy.spiceypy), 21  
 cclass (spiceypy.utils.support\_types.SpiceEKAttDsc attribute), 161  
 cdescrs (spiceypy.utils.support\_types.SpiceEKSegSum attribute), 162  
 center (spiceypy.utils.support\_types.Ellipse attribute), 159  
 center (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161  
 cgv2el() (in module spiceypy.spiceypy), 21  
 character() (spiceypy.utils.support\_types.SpiceCell class method), 160  
 chbder() (in module spiceypy.spiceypy), 21  
 checkForSpiceError() (in module spiceypy.spiceypy), 22  
 chkin() (in module spiceypy.spiceypy), 22  
 chkout() (in module spiceypy.spiceypy), 22  
 CHR (spiceypy.utils.support\_types.DataType attribute), 158  
 cidfrm() (in module spiceypy.spiceypy), 22  
 ckcls() (in module spiceypy.spiceypy), 22  
 kcov() (in module spiceypy.spiceypy), 22  
 ckgp() (in module spiceypy.spiceypy), 23  
 ckgpav() (in module spiceypy.spiceypy), 23  
 cklpf() (in module spiceypy.spiceypy), 23  
 ckobj() (in module spiceypy.spiceypy), 23  
 ckopn() (in module spiceypy.spiceypy), 24  
 ckupf() (in module spiceypy.spiceypy), 24  
 ckw01() (in module spiceypy.spiceypy), 24  
 ckw02() (in module spiceypy.spiceypy), 24  
 ckw03() (in module spiceypy.spiceypy), 25  
 ckw05() (in module spiceypy.spiceypy), 25

- clear() (in module spiceypy.spiceypy), 26
  - clight() (in module spiceypy.spiceypy), 26
  - clpool() (in module spiceypy.spiceypy), 26
  - cltext() (in module spiceypy.spiceypy), 26
  - cMatrixToNumpy() (in module spiceypy.utils.support\_types), 162
  - cmprss() (in module spiceypy.spiceypy), 26
  - cnames (spiceypy.utils.support\_types.SpiceEKSegSum attribute), 162
  - cnmfrm() (in module spiceypy.spiceypy), 27
  - co1max (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - co1min (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - co2max (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - co2min (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - co3max (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - co3min (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - conics() (in module spiceypy.spiceypy), 27
  - constant (spiceypy.utils.support\_types.Plane attribute), 159
  - convrt() (in module spiceypy.spiceypy), 27
  - copy() (in module spiceypy.spiceypy), 27
  - corpar (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - corsys (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - cpos() (in module spiceypy.spiceypy), 28
  - cpostr() (in module spiceypy.spiceypy), 28
  - csize (spiceypy.utils.support\_types.SpiceDLADDescr attribute), 161
  - CTRLBLOCK (spiceypy.utils.support\_types.SpiceCell attribute), 160
  - cVectorToPython() (in module spiceypy.utils.support\_types), 162
  - cvpool() (in module spiceypy.spiceypy), 28
  - cyllat() (in module spiceypy.spiceypy), 28
  - cylrec() (in module spiceypy.spiceypy), 28
  - cylsph() (in module spiceypy.spiceypy), 29
- ## D
- dafac() (in module spiceypy.spiceypy), 29
  - dafbbs() (in module spiceypy.spiceypy), 29
  - dafbfs() (in module spiceypy.spiceypy), 29
  - dafcls() (in module spiceypy.spiceypy), 29
  - dafcs() (in module spiceypy.spiceypy), 29
  - dafdc() (in module spiceypy.spiceypy), 30
  - dafec() (in module spiceypy.spiceypy), 30
  - daffna() (in module spiceypy.spiceypy), 30
  - daffpa() (in module spiceypy.spiceypy), 30
  - dafgda() (in module spiceypy.spiceypy), 30
  - dafgh() (in module spiceypy.spiceypy), 30
  - dafgn() (in module spiceypy.spiceypy), 31
  - dafgs() (in module spiceypy.spiceypy), 31
  - dafgsr() (in module spiceypy.spiceypy), 31
  - dafopr() (in module spiceypy.spiceypy), 31
  - dafopw() (in module spiceypy.spiceypy), 31
  - dafps() (in module spiceypy.spiceypy), 32
  - dafrda() (in module spiceypy.spiceypy), 32
  - dafrrf() (in module spiceypy.spiceypy), 32
  - dafrs() (in module spiceypy.spiceypy), 32
  - dafus() (in module spiceypy.spiceypy), 32
  - dasac() (in module spiceypy.spiceypy), 33
  - dascls() (in module spiceypy.spiceypy), 33
  - dasdc() (in module spiceypy.spiceypy), 33
  - dasec() (in module spiceypy.spiceypy), 33
  - dashfn() (in module spiceypy.spiceypy), 33
  - dasonw() (in module spiceypy.spiceypy), 34
  - dasopr() (in module spiceypy.spiceypy), 34
  - dasopw() (in module spiceypy.spiceypy), 34
  - dasrrf() (in module spiceypy.spiceypy), 34
  - data (spiceypy.utils.support\_types.SpiceCell attribute), 160
  - DataType (class in spiceypy.utils.support\_types), 158
  - DATATYPES\_ENUM (spiceypy.utils.support\_types.SpiceCell attribute), 160
  - DATATYPES\_GET (spiceypy.utils.support\_types.SpiceCell attribute), 160
  - dbase (spiceypy.utils.support\_types.SpiceDLADDescr attribute), 161
  - dclass (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161
  - dcylldr() (in module spiceypy.spiceypy), 34
  - deltet() (in module spiceypy.spiceypy), 34
  - det() (in module spiceypy.spiceypy), 35
  - dgeodr() (in module spiceypy.spiceypy), 35
  - diags2() (in module spiceypy.spiceypy), 35
  - diff() (in module spiceypy.spiceypy), 35
  - dlabbs() (in module spiceypy.spiceypy), 36
  - dlabfs() (in module spiceypy.spiceypy), 36
  - dlafns() (in module spiceypy.spiceypy), 36
  - dlafps() (in module spiceypy.spiceypy), 36
  - dlatdr() (in module spiceypy.spiceypy), 36
  - double() (spiceypy.utils.support\_types.SpiceCell class method), 160
  - DoubleArrayType (class in spiceypy.utils.support\_types), 158
  - DoubleMatrixType (class in spiceypy.utils.support\_types), 159
  - DP (spiceypy.utils.support\_types.DataType attribute), 158
  - dp2hx() (in module spiceypy.spiceypy), 37
  - dpgrdr() (in module spiceypy.spiceypy), 37
  - dpmax() (in module spiceypy.spiceypy), 37

- dpmn() (in module `spicepy.spiceypy`), 37  
 dpr() (in module `spicepy.spiceypy`), 37  
 drdcyl() (in module `spicepy.spiceypy`), 37  
 drdgeo() (in module `spicepy.spiceypy`), 38  
 drdlat() (in module `spicepy.spiceypy`), 38  
 drdpgr() (in module `spicepy.spiceypy`), 38  
 drdsph() (in module `spicepy.spiceypy`), 39  
 dsize (`spicepy.utils.support_types.SpiceDLADescr` attribute), 161  
 dskb02() (in module `spicepy.spiceypy`), 39  
 dskcls() (in module `spicepy.spiceypy`), 39  
 dskd02() (in module `spicepy.spiceypy`), 39  
 dskgd() (in module `spicepy.spiceypy`), 40  
 dskgtl() (in module `spicepy.spiceypy`), 40  
 dski02() (in module `spicepy.spiceypy`), 40  
 dskmi2() (in module `spicepy.spiceypy`), 40  
 dskn02() (in module `spicepy.spiceypy`), 41  
 dskobj() (in module `spicepy.spiceypy`), 41  
 dskopn() (in module `spicepy.spiceypy`), 41  
 dskp02() (in module `spicepy.spiceypy`), 41  
 dskrb2() (in module `spicepy.spiceypy`), 42  
 dskrsf() (in module `spicepy.spiceypy`), 42  
 dskstl() (in module `spicepy.spiceypy`), 42  
 dskv02() (in module `spicepy.spiceypy`), 42  
 dskw02() (in module `spicepy.spiceypy`), 43  
 dskx02() (in module `spicepy.spiceypy`), 43  
 dskxsi() (in module `spicepy.spiceypy`), 44  
 dskxv() (in module `spicepy.spiceypy`), 44  
 dskz02() (in module `spicepy.spiceypy`), 44  
 dsphdr() (in module `spicepy.spiceypy`), 45  
 dtpool() (in module `spicepy.spiceypy`), 45  
 dtype (`spicepy.utils.support_types.SpiceCell` attribute), 160  
 dtype (`spicepy.utils.support_types.SpiceDSKDescr` attribute), 161  
 dtype (`spicepy.utils.support_types.SpiceEKAttDsc` attribute), 161  
 ducrss() (in module `spicepy.spiceypy`), 45  
 dvcrrs() (in module `spicepy.spiceypy`), 45  
 dvdot() (in module `spicepy.spiceypy`), 45  
 dvhat() (in module `spicepy.spiceypy`), 46  
 dvnorm() (in module `spicepy.spiceypy`), 46  
 dvpool() (in module `spicepy.spiceypy`), 46  
 dvsep() (in module `spicepy.spiceypy`), 46
- ## E
- edlimb() (in module `spicepy.spiceypy`), 46  
 edterm() (in module `spicepy.spiceypy`), 47  
 ekacec() (in module `spicepy.spiceypy`), 47  
 ekaced() (in module `spicepy.spiceypy`), 47  
 ekacei() (in module `spicepy.spiceypy`), 48  
 ekaclc() (in module `spicepy.spiceypy`), 48  
 ekaclD() (in module `spicepy.spiceypy`), 48  
 ekacli() (in module `spicepy.spiceypy`), 49  
 ekappr() (in module `spicepy.spiceypy`), 49  
 ekbseg() (in module `spicepy.spiceypy`), 49  
 ekccnt() (in module `spicepy.spiceypy`), 49  
 ekcii() (in module `spicepy.spiceypy`), 50  
 ekcls() (in module `spicepy.spiceypy`), 50  
 ekdelr() (in module `spicepy.spiceypy`), 50  
 ekffld() (in module `spicepy.spiceypy`), 50  
 ekfind() (in module `spicepy.spiceypy`), 50  
 ekgc() (in module `spicepy.spiceypy`), 51  
 ekgd() (in module `spicepy.spiceypy`), 51  
 ekgi() (in module `spicepy.spiceypy`), 51  
 ekifld() (in module `spicepy.spiceypy`), 51  
 ekinsr() (in module `spicepy.spiceypy`), 52  
 eklef() (in module `spicepy.spiceypy`), 52  
 eknelt() (in module `spicepy.spiceypy`), 52  
 eknseg() (in module `spicepy.spiceypy`), 52  
 ekntab() (in module `spicepy.spiceypy`), 52  
 ekopn() (in module `spicepy.spiceypy`), 53  
 ekopr() (in module `spicepy.spiceypy`), 53  
 ekops() (in module `spicepy.spiceypy`), 53  
 ekopw() (in module `spicepy.spiceypy`), 53  
 ekpsel() (in module `spicepy.spiceypy`), 53  
 ekrcec() (in module `spicepy.spiceypy`), 54  
 ekrced() (in module `spicepy.spiceypy`), 54  
 ekrcei() (in module `spicepy.spiceypy`), 54  
 ekssum() (in module `spicepy.spiceypy`), 55  
 ektnam() (in module `spicepy.spiceypy`), 55  
 ekucec() (in module `spicepy.spiceypy`), 55  
 ekuced() (in module `spicepy.spiceypy`), 55  
 ekucei() (in module `spicepy.spiceypy`), 56  
 ekuef() (in module `spicepy.spiceypy`), 56  
 el2cgv() (in module `spicepy.spiceypy`), 56  
 elemc() (in module `spicepy.spiceypy`), 56  
 elemD() (in module `spicepy.spiceypy`), 56  
 elemi() (in module `spicepy.spiceypy`), 57  
 Ellipse (class in `spicepy.utils.support_types`), 159  
 emptyBoolVector() (in module `spicepy.utils.support_types`), 162  
 emptyCharArray() (in module `spicepy.utils.support_types`), 162  
 emptyDoubleMatrix() (in module `spicepy.utils.support_types`), 162  
 emptyDoubleVector() (in module `spicepy.utils.support_types`), 162  
 emptyIntMatrix() (in module `spicepy.utils.support_types`), 162  
 emptyIntVector() (in module `spicepy.utils.support_types`), 162  
 emptySpiceEKDataTypeVector() (in module `spicepy.utils.support_types`), 162  
 emptySpiceEKExprClassVector() (in module `spicepy.utils.support_types`), 162  
 eqncpv() (in module `spicepy.spiceypy`), 57  
 eqstr() (in module `spicepy.spiceypy`), 57

erract() (in module spiceypy.spiceypy), 57  
 errch() (in module spiceypy.spiceypy), 58  
 errdev() (in module spiceypy.spiceypy), 58  
 errdp() (in module spiceypy.spiceypy), 58  
 errint() (in module spiceypy.spiceypy), 58  
 errprt() (in module spiceypy.spiceypy), 58  
 esrhc() (in module spiceypy.spiceypy), 59  
 et2lst() (in module spiceypy.spiceypy), 59  
 et2utc() (in module spiceypy.spiceypy), 59  
 etcal() (in module spiceypy.spiceypy), 59  
 eul2m() (in module spiceypy.spiceypy), 60  
 eul2xf() (in module spiceypy.spiceypy), 60  
 exists() (in module spiceypy.spiceypy), 60  
 expool() (in module spiceypy.spiceypy), 60

## F

failed() (in module spiceypy.spiceypy), 61  
 fn2lun() (in module spiceypy.spiceypy), 61  
 found\_check() (in module spiceypy.spiceypy), 61  
 found\_check\_off() (in module spiceypy.spiceypy), 61  
 found\_check\_on() (in module spiceypy.spiceypy), 61  
 fovray() (in module spiceypy.spiceypy), 61  
 fovtrg() (in module spiceypy.spiceypy), 62  
 frame() (in module spiceypy.spiceypy), 62  
 frinfo() (in module spiceypy.spiceypy), 62  
 frmcdc (spiceypy.utils.support\_types.SpiceDSKDescr attribute), 161  
 frmnam() (in module spiceypy.spiceypy), 62  
 from\_array() (spiceypy.utils.support\_types.DoubleArrayType method), 158  
 from\_array() (spiceypy.utils.support\_types.IntArrayType method), 159  
 from\_list() (spiceypy.utils.support\_types.BoolArrayType method), 158  
 from\_list() (spiceypy.utils.support\_types.DoubleArrayType method), 158  
 from\_list() (spiceypy.utils.support\_types.DoubleMatrixType method), 159  
 from\_list() (spiceypy.utils.support\_types.IntArrayType method), 159  
 from\_list() (spiceypy.utils.support\_types.IntMatrixType method), 159  
 from\_matrix() (spiceypy.utils.support\_types.DoubleMatrixType method), 159  
 from\_matrix() (spiceypy.utils.support\_types.IntMatrixType method), 159  
 from\_ndarray() (spiceypy.utils.support\_types.BoolArrayType method), 158  
 from\_ndarray() (spiceypy.utils.support\_types.DoubleArrayType method), 158  
 from\_ndarray() (spiceypy.utils.support\_types.DoubleMatrixType method), 159  
 from\_ndarray() (spiceypy.utils.support\_types.IntArrayType method), 159

from\_ndarray() (spiceypy.utils.support\_types.IntMatrixType method), 159  
 from\_param() (spiceypy.utils.support\_types.BoolArrayType method), 158  
 from\_param() (spiceypy.utils.support\_types.DoubleArrayType method), 158  
 from\_param() (spiceypy.utils.support\_types.DoubleMatrixType method), 159  
 from\_param() (spiceypy.utils.support\_types.IntArrayType method), 159  
 from\_param() (spiceypy.utils.support\_types.IntMatrixType method), 159  
 from\_tuple() (spiceypy.utils.support\_types.BoolArrayType method), 158  
 from\_tuple() (spiceypy.utils.support\_types.DoubleArrayType method), 158  
 from\_tuple() (spiceypy.utils.support\_types.DoubleMatrixType method), 159  
 from\_tuple() (spiceypy.utils.support\_types.IntArrayType method), 159  
 from\_tuple() (spiceypy.utils.support\_types.IntMatrixType method), 159  
 ftncfs() (in module spiceypy.spiceypy), 63  
 furnsh() (in module spiceypy.spiceypy), 63  
 fwdptr (spiceypy.utils.support\_types.SpiceDLADescr attribute), 161

## G

gcpcpool() (in module spiceypy.spiceypy), 63  
 gdpcpool() (in module spiceypy.spiceypy), 63  
 georec() (in module spiceypy.spiceypy), 63  
 get\_found\_catch\_state() (in module spiceypy.spiceypy), 64  
 getelm() (in module spiceypy.spiceypy), 64  
 getfat() (in module spiceypy.spiceypy), 64  
 getfov() (in module spiceypy.spiceypy), 64  
 getmsg() (in module spiceypy.spiceypy), 64  
 gfbail() (in module spiceypy.spiceypy), 65  
 gfcrlh() (in module spiceypy.spiceypy), 65  
 gfdist() (in module spiceypy.spiceypy), 65  
 gfevnt() (in module spiceypy.spiceypy), 65  
 gffove() (in module spiceypy.spiceypy), 65  
 gflum() (in module spiceypy.spiceypy), 65  
 gfinth() (in module spiceypy.spiceypy), 66  
 gfocce() (in module spiceypy.spiceypy), 66  
 gfoclt() (in module spiceypy.spiceypy), 66  
 gfpfa() (in module spiceypy.spiceypy), 67  
 gfposc() (in module spiceypy.spiceypy), 67  
 gfnfn() (in module spiceypy.spiceypy), 68  
 gfrepf() (in module spiceypy.spiceypy), 68  
 gfrpi() (in module spiceypy.spiceypy), 68  
 gfrepu() (in module spiceypy.spiceypy), 68  
 gfrfov() (in module spiceypy.spiceypy), 68  
 gfrrr() (in module spiceypy.spiceypy), 69



gfsep() (in module spiceypy.spiceypy), 69  
 gfsntc() (in module spiceypy.spiceypy), 70  
 gfsstp() (in module spiceypy.spiceypy), 70  
 gfstep() (in module spiceypy.spiceypy), 70  
 gfstol() (in module spiceypy.spiceypy), 71  
 gfsubc() (in module spiceypy.spiceypy), 71  
 gftfov() (in module spiceypy.spiceypy), 71  
 gfudb() (in module spiceypy.spiceypy), 72  
 gfuds() (in module spiceypy.spiceypy), 72  
 gipool() (in module spiceypy.spiceypy), 73  
 gnpool() (in module spiceypy.spiceypy), 73

## H

halfpi() (in module spiceypy.spiceypy), 73  
 hrmint() (in module spiceypy.spiceypy), 73  
 hx2dp() (in module spiceypy.spiceypy), 73

## I

ibase (spiceypy.utils.support\_types.SpiceDLADescr attribute), 161  
 ident() (in module spiceypy.spiceypy), 74  
 illum() (in module spiceypy.spiceypy), 74  
 illumf() (in module spiceypy.spiceypy), 74  
 illumg() (in module spiceypy.spiceypy), 75  
 ilumin() (in module spiceypy.spiceypy), 75  
 indexd (spiceypy.utils.support\_types.SpiceEKAttDsc attribute), 161  
 inedpl() (in module spiceypy.spiceypy), 76  
 inelpl() (in module spiceypy.spiceypy), 76  
 init (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 inrypl() (in module spiceypy.spiceypy), 76  
 insrtc() (in module spiceypy.spiceypy), 76  
 insrtd() (in module spiceypy.spiceypy), 76  
 insrti() (in module spiceypy.spiceypy), 77  
 INT (spiceypy.utils.support\_types.DataType attribute), 158  
 IntArrayType (class in spiceypy.utils.support\_types), 159  
 integer() (spiceypy.utils.support\_types.SpiceCell class method), 160  
 inter() (in module spiceypy.spiceypy), 77  
 IntMatrixType (class in spiceypy.utils.support\_types), 159  
 intmax() (in module spiceypy.spiceypy), 77  
 intmin() (in module spiceypy.spiceypy), 77  
 invert() (in module spiceypy.spiceypy), 77  
 invort() (in module spiceypy.spiceypy), 77  
 is\_bool() (spiceypy.utils.support\_types.SpiceCell method), 160  
 is\_char() (spiceypy.utils.support\_types.SpiceCell method), 160  
 is\_double() (spiceypy.utils.support\_types.SpiceCell method), 160

is\_int() (spiceypy.utils.support\_types.SpiceCell method), 160  
 is\_set() (spiceypy.utils.support\_types.SpiceCell method), 160  
 is\_time() (spiceypy.utils.support\_types.SpiceCell method), 160  
 isize (spiceypy.utils.support\_types.SpiceDLADescr attribute), 161  
 isordv() (in module spiceypy.spiceypy), 78  
 isrhc() (in module spiceypy.spiceypy), 78  
 isrhd() (in module spiceypy.spiceypy), 78  
 isrhi() (in module spiceypy.spiceypy), 78  
 isrot() (in module spiceypy.spiceypy), 79  
 isSet (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 iswhsp() (in module spiceypy.spiceypy), 79

## J

j1900() (in module spiceypy.spiceypy), 79  
 j1950() (in module spiceypy.spiceypy), 79  
 j2000() (in module spiceypy.spiceypy), 79  
 j2100() (in module spiceypy.spiceypy), 79  
 jyear() (in module spiceypy.spiceypy), 79

## K

kclear() (in module spiceypy.spiceypy), 80  
 kdata() (in module spiceypy.spiceypy), 80  
 kinfo() (in module spiceypy.spiceypy), 80  
 kplfrm() (in module spiceypy.spiceypy), 80  
 kttotal() (in module spiceypy.spiceypy), 80  
 kxtrct() (in module spiceypy.spiceypy), 81

## L

lastnb() (in module spiceypy.spiceypy), 81  
 latcyl() (in module spiceypy.spiceypy), 81  
 latrec() (in module spiceypy.spiceypy), 81  
 latsph() (in module spiceypy.spiceypy), 82  
 latsrf() (in module spiceypy.spiceypy), 82  
 lcase() (in module spiceypy.spiceypy), 82  
 ldpool() (in module spiceypy.spiceypy), 82  
 length (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 lgrind() (in module spiceypy.spiceypy), 83  
 limbpt() (in module spiceypy.spiceypy), 83  
 listToCharArray() (in module spiceypy.utils.support\_types), 162  
 listToCharArrayPtr() (in module spiceypy.utils.support\_types), 162  
 lmpool() (in module spiceypy.spiceypy), 83  
 lparse() (in module spiceypy.spiceypy), 84  
 lparsm() (in module spiceypy.spiceypy), 84  
 lparss() (in module spiceypy.spiceypy), 84  
 lspcn() (in module spiceypy.spiceypy), 84  
 lstlec() (in module spiceypy.spiceypy), 85

lstled() (in module `spiceypy.spiceypy`), 85  
lstlei() (in module `spiceypy.spiceypy`), 85  
lstltc() (in module `spiceypy.spiceypy`), 85  
lstltd() (in module `spiceypy.spiceypy`), 86  
lstlti() (in module `spiceypy.spiceypy`), 86  
ltime() (in module `spiceypy.spiceypy`), 86  
lx4dec() (in module `spiceypy.spiceypy`), 86  
lx4num() (in module `spiceypy.spiceypy`), 87  
lx4sgn() (in module `spiceypy.spiceypy`), 87  
lx4uns() (in module `spiceypy.spiceypy`), 87  
lxqstr() (in module `spiceypy.spiceypy`), 87

## M

m2eul() (in module `spiceypy.spiceypy`), 88  
m2q() (in module `spiceypy.spiceypy`), 88  
matchi() (in module `spiceypy.spiceypy`), 88  
matchw() (in module `spiceypy.spiceypy`), 88  
mequ() (in module `spiceypy.spiceypy`), 89  
mequg() (in module `spiceypy.spiceypy`), 89  
minCharLen (`spiceypy.utils.support_types.SpiceCell` attribute), 160  
mtxm() (in module `spiceypy.spiceypy`), 89  
mtxmg() (in module `spiceypy.spiceypy`), 89  
mtxv() (in module `spiceypy.spiceypy`), 89  
mtxvg() (in module `spiceypy.spiceypy`), 90  
mxm() (in module `spiceypy.spiceypy`), 90  
mxmg() (in module `spiceypy.spiceypy`), 90  
mxmt() (in module `spiceypy.spiceypy`), 90  
mxmtg() (in module `spiceypy.spiceypy`), 91  
mxv() (in module `spiceypy.spiceypy`), 91  
mxvg() (in module `spiceypy.spiceypy`), 91

## N

namfrm() (in module `spiceypy.spiceypy`), 91  
ncols (`spiceypy.utils.support_types.SpiceEKSegSum` attribute), 162  
ncpos() (in module `spiceypy.spiceypy`), 92  
ncposr() (in module `spiceypy.spiceypy`), 92  
nearpt() (in module `spiceypy.spiceypy`), 92  
no\_found\_check() (in module `spiceypy.spiceypy`), 92  
normal (`spiceypy.utils.support_types.Plane` attribute), 159  
npedln() (in module `spiceypy.spiceypy`), 93  
npelpt() (in module `spiceypy.spiceypy`), 93  
nplnpt() (in module `spiceypy.spiceypy`), 93  
nrows (`spiceypy.utils.support_types.SpiceEKSegSum` attribute), 162  
nullok (`spiceypy.utils.support_types.SpiceEKAttDsc` attribute), 161  
nvc2pl() (in module `spiceypy.spiceypy`), 94  
nvp2pl() (in module `spiceypy.spiceypy`), 94

## O

occult() (in module `spiceypy.spiceypy`), 94  
ordc() (in module `spiceypy.spiceypy`), 94

ordd() (in module `spiceypy.spiceypy`), 95  
orderc() (in module `spiceypy.spiceypy`), 95  
orderd() (in module `spiceypy.spiceypy`), 95  
orderi() (in module `spiceypy.spiceypy`), 95  
ordi() (in module `spiceypy.spiceypy`), 96  
osclt() (in module `spiceypy.spiceypy`), 96  
oscltx() (in module `spiceypy.spiceypy`), 96

## P

pckcls() (in module `spiceypy.spiceypy`), 96  
pckcov() (in module `spiceypy.spiceypy`), 96  
pckfrm() (in module `spiceypy.spiceypy`), 97  
pcklof() (in module `spiceypy.spiceypy`), 97  
pckopn() (in module `spiceypy.spiceypy`), 97  
pckuof() (in module `spiceypy.spiceypy`), 97  
pckw02() (in module `spiceypy.spiceypy`), 97  
pcpool() (in module `spiceypy.spiceypy`), 98  
pdpool() (in module `spiceypy.spiceypy`), 98  
pgrrec() (in module `spiceypy.spiceypy`), 98  
phaseq() (in module `spiceypy.spiceypy`), 98  
pi() (in module `spiceypy.spiceypy`), 99  
pipool() (in module `spiceypy.spiceypy`), 99  
pjelpl() (in module `spiceypy.spiceypy`), 99  
pl2nvc() (in module `spiceypy.spiceypy`), 99  
pl2nvp() (in module `spiceypy.spiceypy`), 99  
pl2psv() (in module `spiceypy.spiceypy`), 100  
Plane (class in `spiceypy.utils.support_types`), 159  
pltar() (in module `spiceypy.spiceypy`), 100  
pltxp() (in module `spiceypy.spiceypy`), 100  
pltnp() (in module `spiceypy.spiceypy`), 100  
pltnrm() (in module `spiceypy.spiceypy`), 101  
pltvol() (in module `spiceypy.spiceypy`), 101  
polyds() (in module `spiceypy.spiceypy`), 101  
pos() (in module `spiceypy.spiceypy`), 101  
posr() (in module `spiceypy.spiceypy`), 101  
prop2b() (in module `spiceypy.spiceypy`), 102  
prsdp() (in module `spiceypy.spiceypy`), 102  
prsint() (in module `spiceypy.spiceypy`), 102  
psv2pl() (in module `spiceypy.spiceypy`), 102  
pxform() (in module `spiceypy.spiceypy`), 103  
pxfrm2() (in module `spiceypy.spiceypy`), 103

## Q

q2m() (in module `spiceypy.spiceypy`), 103  
qcktrc() (in module `spiceypy.spiceypy`), 103  
qdq2av() (in module `spiceypy.spiceypy`), 103  
qxq() (in module `spiceypy.spiceypy`), 104

## R

radrec() (in module `spiceypy.spiceypy`), 104  
rav2xf() (in module `spiceypy.spiceypy`), 104  
raxisa() (in module `spiceypy.spiceypy`), 104  
rdtext() (in module `spiceypy.spiceypy`), 104  
reccyl() (in module `spiceypy.spiceypy`), 105

recgeo() (in module spiceypy.spiceypy), 105  
 reclat() (in module spiceypy.spiceypy), 105  
 recpgr() (in module spiceypy.spiceypy), 105  
 recrad() (in module spiceypy.spiceypy), 106  
 recsph() (in module spiceypy.spiceypy), 106  
 removc() (in module spiceypy.spiceypy), 106  
 removd() (in module spiceypy.spiceypy), 106  
 removi() (in module spiceypy.spiceypy), 106  
 reordc() (in module spiceypy.spiceypy), 106  
 reordd() (in module spiceypy.spiceypy), 107  
 reordi() (in module spiceypy.spiceypy), 107  
 reordl() (in module spiceypy.spiceypy), 107  
 repmc() (in module spiceypy.spiceypy), 107  
 repmct() (in module spiceypy.spiceypy), 108  
 repmd() (in module spiceypy.spiceypy), 108  
 repmf() (in module spiceypy.spiceypy), 108  
 repmi() (in module spiceypy.spiceypy), 109  
 repmot() (in module spiceypy.spiceypy), 109  
 reset() (in module spiceypy.spiceypy), 109  
 reset() (spiceypy.utils.support\_types.SpiceCell method), 160  
 return\_c() (in module spiceypy.spiceypy), 109  
 rotate() (in module spiceypy.spiceypy), 109  
 rotmat() (in module spiceypy.spiceypy), 110  
 rotvec() (in module spiceypy.spiceypy), 110  
 rpd() (in module spiceypy.spiceypy), 110  
 rquad() (in module spiceypy.spiceypy), 110

## S

S18TP0 (spiceypy.utils.support\_types.SpiceSPK18Subtype attribute), 162  
 S18TP1 (spiceypy.utils.support\_types.SpiceSPK18Subtype attribute), 162  
 saelgv() (in module spiceypy.spiceypy), 110  
 scard() (in module spiceypy.spiceypy), 111  
 scdecdd() (in module spiceypy.spiceypy), 111  
 sce2c() (in module spiceypy.spiceypy), 111  
 sce2s() (in module spiceypy.spiceypy), 111  
 sce2t() (in module spiceypy.spiceypy), 112  
 scencdd() (in module spiceypy.spiceypy), 112  
 scfmt() (in module spiceypy.spiceypy), 112  
 scpart() (in module spiceypy.spiceypy), 112  
 scs2e() (in module spiceypy.spiceypy), 113  
 sct2e() (in module spiceypy.spiceypy), 113  
 sctiks() (in module spiceypy.spiceypy), 113  
 sdiff() (in module spiceypy.spiceypy), 113  
 semi\_major (spiceypy.utils.support\_types.Ellipse attribute), 159  
 semi\_minor (spiceypy.utils.support\_types.Ellipse attribute), 159  
 set\_c() (in module spiceypy.spiceypy), 113  
 setmsg() (in module spiceypy.spiceypy), 114  
 shellc() (in module spiceypy.spiceypy), 114  
 shelldd() (in module spiceypy.spiceypy), 114

shelli() (in module spiceypy.spiceypy), 114  
 sigerr() (in module spiceypy.spiceypy), 114  
 sincpt() (in module spiceypy.spiceypy), 115  
 size (spiceypy.utils.support\_types.SpiceCell attribute), 160  
 size (spiceypy.utils.support\_types.SpiceEKAttDsc attribute), 161  
 size() (in module spiceypy.spiceypy), 115  
 spd() (in module spiceypy.spiceypy), 115  
 sphcyl() (in module spiceypy.spiceypy), 115  
 sphlat() (in module spiceypy.spiceypy), 116  
 sphrec() (in module spiceypy.spiceypy), 116  
 SPICE\_BOOL (spiceypy.utils.support\_types.DataType attribute), 158  
 SPICE\_BOOL (spiceypy.utils.support\_types.SpiceEKDataType attribute), 161  
 SPICE\_CHR (spiceypy.utils.support\_types.DataType attribute), 158  
 SPICE\_CHR (spiceypy.utils.support\_types.SpiceEKDataType attribute), 161  
 SPICE\_DP (spiceypy.utils.support\_types.DataType attribute), 158  
 SPICE\_DP (spiceypy.utils.support\_types.SpiceEKDataType attribute), 161  
 SPICE\_EK\_EXP\_COL (spiceypy.utils.support\_types.SpiceEKExprClass attribute), 162  
 SPICE\_EK\_EXP\_EXPR (spiceypy.utils.support\_types.SpiceEKExprClass attribute), 162  
 SPICE\_EK\_EXP\_FUNC (spiceypy.utils.support\_types.SpiceEKExprClass attribute), 162  
 SPICE\_INT (spiceypy.utils.support\_types.DataType attribute), 158  
 SPICE\_INT (spiceypy.utils.support\_types.SpiceEKDataType attribute), 162  
 SPICE\_TIME (spiceypy.utils.support\_types.DataType attribute), 158  
 SPICE\_TIME (spiceypy.utils.support\_types.SpiceEKDataType attribute), 162  
 SPICEBOOL\_CELL() (in module spiceypy.utils.support\_types), 159  
 SpiceCell (class in spiceypy.utils.support\_types), 159  
 SPICECHAR\_CELL() (in module spiceypy.utils.support\_types), 159  
 SpiceDLADescr (class in spiceypy.utils.support\_types), 160  
 SPICEDOUBLE\_CELL() (in module spiceypy.utils.support\_types), 159  
 SpiceDSKDescr (class in spiceypy.utils.support\_types), 161  
 SpiceEKAttDsc (class in spiceypy.utils.support\_types), 161  
 SpiceEKDataType (class in spiceypy.utils.support\_types),

- 161
  - SpiceEKExprClass (class in `spiceypy.utils.support_types`), 162
  - SpiceEKSegSum (class in `spiceypy.utils.support_types`), 162
  - `spiceErrorCheck()` (in module `spiceypy.spiceypy`), 116
  - `spiceFoundExceptionThrower()` (in module `spiceypy.spiceypy`), 116
  - SPICEINT\_CELL() (in `spiceypy.utils.support_types`), 159
  - SpiceSPK18Subtype (class in `spiceypy.utils.support_types`), 162
  - SPICETIME\_CELL() (in `spiceypy.utils.support_types`), 159
  - SpiceyError, 162
  - `spiceypy.spiceypy` (module), 15
  - `spiceypy.utils.libspicehelper` (module), 163
  - `spiceypy.utils.support_types` (module), 157
  - `spk14a()` (in module `spiceypy.spiceypy`), 116
  - `spk14b()` (in module `spiceypy.spiceypy`), 116
  - `spk14e()` (in module `spiceypy.spiceypy`), 117
  - `spkacs()` (in module `spiceypy.spiceypy`), 117
  - `spkapo()` (in module `spiceypy.spiceypy`), 117
  - `spkapp()` (in module `spiceypy.spiceypy`), 117
  - `spkaps()` (in module `spiceypy.spiceypy`), 118
  - `spkcls()` (in module `spiceypy.spiceypy`), 118
  - `spkcov()` (in module `spiceypy.spiceypy`), 118
  - `spkcpo()` (in module `spiceypy.spiceypy`), 119
  - `spkcpt()` (in module `spiceypy.spiceypy`), 119
  - `spkcvv()` (in module `spiceypy.spiceypy`), 119
  - `spkcvv()` (in module `spiceypy.spiceypy`), 120
  - `spkez()` (in module `spiceypy.spiceypy`), 120
  - `spkezp()` (in module `spiceypy.spiceypy`), 121
  - `spkezr()` (in module `spiceypy.spiceypy`), 121
  - `spkgeo()` (in module `spiceypy.spiceypy`), 121
  - `spkgps()` (in module `spiceypy.spiceypy`), 122
  - `spklef()` (in module `spiceypy.spiceypy`), 122
  - `spkltc()` (in module `spiceypy.spiceypy`), 122
  - `spkobj()` (in module `spiceypy.spiceypy`), 122
  - `spkopa()` (in module `spiceypy.spiceypy`), 123
  - `spkopn()` (in module `spiceypy.spiceypy`), 123
  - `spkpds()` (in module `spiceypy.spiceypy`), 123
  - `spkpos()` (in module `spiceypy.spiceypy`), 123
  - `spkpvn()` (in module `spiceypy.spiceypy`), 124
  - `spksfs()` (in module `spiceypy.spiceypy`), 124
  - `spkssb()` (in module `spiceypy.spiceypy`), 124
  - `spksub()` (in module `spiceypy.spiceypy`), 124
  - `spkuds()` (in module `spiceypy.spiceypy`), 125
  - `spkuef()` (in module `spiceypy.spiceypy`), 125
  - `spkw02()` (in module `spiceypy.spiceypy`), 125
  - `spkw03()` (in module `spiceypy.spiceypy`), 125
  - `spkw05()` (in module `spiceypy.spiceypy`), 126
  - `spkw08()` (in module `spiceypy.spiceypy`), 126
  - `spkw09()` (in module `spiceypy.spiceypy`), 127
  - `spkw10()` (in module `spiceypy.spiceypy`), 127
  - `spkw12()` (in module `spiceypy.spiceypy`), 128
  - `spkw13()` (in module `spiceypy.spiceypy`), 128
  - `spkw15()` (in module `spiceypy.spiceypy`), 128
  - `spkw17()` (in module `spiceypy.spiceypy`), 129
  - `spkw18()` (in module `spiceypy.spiceypy`), 129
  - `spkw20()` (in module `spiceypy.spiceypy`), 130
  - `srfc2s()` (in module `spiceypy.spiceypy`), 130
  - `srfcss()` (in module `spiceypy.spiceypy`), 131
  - `srfnrm()` (in module `spiceypy.spiceypy`), 131
  - `srfreq()` (in module `spiceypy.spiceypy`), 131
  - `srfs2c()` (in module `spiceypy.spiceypy`), 132
  - `srfscc()` (in module `spiceypy.spiceypy`), 132
  - `srfxpt()` (in module `spiceypy.spiceypy`), 132
  - `ssize()` (in module `spiceypy.spiceypy`), 132
  - `start` (`spiceypy.utils.support_types.SpiceDSKDescr` attribute), 161
  - `stelab()` (in module `spiceypy.spiceypy`), 133
  - `stop` (`spiceypy.utils.support_types.SpiceDSKDescr` attribute), 161
  - `stpool()` (in module `spiceypy.spiceypy`), 133
  - `str2et()` (in module `spiceypy.spiceypy`), 133
  - `stringToCharP()` (in `spiceypy.utils.support_types`), 162
  - `strlen` (`spiceypy.utils.support_types.SpiceEKAttDsc` attribute), 161
  - `subpnt()` (in module `spiceypy.spiceypy`), 133
  - `subpt()` (in module `spiceypy.spiceypy`), 134
  - `subslr()` (in module `spiceypy.spiceypy`), 134
  - `subsol()` (in module `spiceypy.spiceypy`), 135
  - `sumad()` (in module `spiceypy.spiceypy`), 135
  - `sumai()` (in module `spiceypy.spiceypy`), 135
  - `surfce` (`spiceypy.utils.support_types.SpiceDSKDescr` attribute), 161
  - `surfnm()` (in module `spiceypy.spiceypy`), 135
  - `surfpt()` (in module `spiceypy.spiceypy`), 136
  - `surfpv()` (in module `spiceypy.spiceypy`), 136
  - `swpool()` (in module `spiceypy.spiceypy`), 136
  - `sxform()` (in module `spiceypy.spiceypy`), 136
  - `szpool()` (in module `spiceypy.spiceypy`), 137
- ## T
- `tabnam` (`spiceypy.utils.support_types.SpiceEKSegSum` attribute), 162
  - `termpt()` (in module `spiceypy.spiceypy`), 137
  - `timdef()` (in module `spiceypy.spiceypy`), 137
  - TIME (`spiceypy.utils.support_types.DataType` attribute), 158
  - `time()` (`spiceypy.utils.support_types.SpiceCell` class method), 160
  - `timeout()` (in module `spiceypy.spiceypy`), 138
  - `tipbod()` (in module `spiceypy.spiceypy`), 138
  - `tisbod()` (in module `spiceypy.spiceypy`), 138
  - `tkvrsn()` (in module `spiceypy.spiceypy`), 138

toBoolVector() (in module spiceypy.utils.support\_types), 163  
 toDoubleMatrix() (in module spiceypy.utils.support\_types), 163  
 toDoubleVector() (in module spiceypy.utils.support\_types), 163  
 toIntMatrix() (in module spiceypy.utils.support\_types), 163  
 toIntVector() (in module spiceypy.utils.support\_types), 163  
 toPythonString() (in module spiceypy.utils.support\_types), 163  
 tparse() (in module spiceypy.spiceypy), 139  
 tpictr() (in module spiceypy.spiceypy), 139  
 trace() (in module spiceypy.spiceypy), 139  
 tredep() (in module spiceypy.spiceypy), 139  
 trenam() (in module spiceypy.spiceypy), 139  
 trcoff() (in module spiceypy.spiceypy), 140  
 tsetyr() (in module spiceypy.spiceypy), 140  
 twopi() (in module spiceypy.spiceypy), 140  
 twovec() (in module spiceypy.spiceypy), 140  
 txtopn() (in module spiceypy.spiceypy), 140  
 tyear() (in module spiceypy.spiceypy), 140  
 vminus() (in module spiceypy.spiceypy), 147  
 vnorm() (in module spiceypy.spiceypy), 147  
 vnormg() (in module spiceypy.spiceypy), 147  
 vpack() (in module spiceypy.spiceypy), 147  
 vperp() (in module spiceypy.spiceypy), 148  
 vprjp() (in module spiceypy.spiceypy), 148  
 vprjpi() (in module spiceypy.spiceypy), 148  
 vproj() (in module spiceypy.spiceypy), 148  
 vrel() (in module spiceypy.spiceypy), 149  
 vrelg() (in module spiceypy.spiceypy), 149  
 vrotrv() (in module spiceypy.spiceypy), 149  
 vscl() (in module spiceypy.spiceypy), 149  
 vsclg() (in module spiceypy.spiceypy), 149  
 vsep() (in module spiceypy.spiceypy), 150  
 vsepg() (in module spiceypy.spiceypy), 150  
 vsub() (in module spiceypy.spiceypy), 150  
 vsubg() (in module spiceypy.spiceypy), 150  
 vtmv() (in module spiceypy.spiceypy), 151  
 vtmvg() (in module spiceypy.spiceypy), 151  
 vupack() (in module spiceypy.spiceypy), 151  
 vzero() (in module spiceypy.spiceypy), 151  
 vzerog() (in module spiceypy.spiceypy), 151

## U

ucase() (in module spiceypy.spiceypy), 141  
 ucrss() (in module spiceypy.spiceypy), 141  
 uddc() (in module spiceypy.spiceypy), 141  
 uddf() (in module spiceypy.spiceypy), 142  
 udf() (in module spiceypy.spiceypy), 142  
 union() (in module spiceypy.spiceypy), 142  
 unitim() (in module spiceypy.spiceypy), 142  
 unload() (in module spiceypy.spiceypy), 143  
 unorm() (in module spiceypy.spiceypy), 143  
 unormg() (in module spiceypy.spiceypy), 143  
 utc2et() (in module spiceypy.spiceypy), 143

## V

vadd() (in module spiceypy.spiceypy), 143  
 vaddg() (in module spiceypy.spiceypy), 143  
 valid() (in module spiceypy.spiceypy), 144  
 vcrss() (in module spiceypy.spiceypy), 144  
 vdist() (in module spiceypy.spiceypy), 144  
 vdistg() (in module spiceypy.spiceypy), 144  
 vdot() (in module spiceypy.spiceypy), 145  
 vdotg() (in module spiceypy.spiceypy), 145  
 vequ() (in module spiceypy.spiceypy), 145  
 vequg() (in module spiceypy.spiceypy), 145  
 vhat() (in module spiceypy.spiceypy), 145  
 vhatg() (in module spiceypy.spiceypy), 146  
 vlcom() (in module spiceypy.spiceypy), 146  
 vlcom3() (in module spiceypy.spiceypy), 146  
 vlcomg() (in module spiceypy.spiceypy), 146  
 vminus() (in module spiceypy.spiceypy), 147

## W

wncard() (in module spiceypy.spiceypy), 152  
 wncomd() (in module spiceypy.spiceypy), 152  
 wncond() (in module spiceypy.spiceypy), 152  
 wndifd() (in module spiceypy.spiceypy), 152  
 wnelmd() (in module spiceypy.spiceypy), 153  
 wnexpd() (in module spiceypy.spiceypy), 153  
 wnnextd() (in module spiceypy.spiceypy), 153  
 wnfetd() (in module spiceypy.spiceypy), 153  
 wnfiled() (in module spiceypy.spiceypy), 153  
 wnfltd() (in module spiceypy.spiceypy), 154  
 wnincd() (in module spiceypy.spiceypy), 154  
 wninsd() (in module spiceypy.spiceypy), 154  
 wnintd() (in module spiceypy.spiceypy), 154  
 wnreld() (in module spiceypy.spiceypy), 155  
 wnsumd() (in module spiceypy.spiceypy), 155  
 wnunid() (in module spiceypy.spiceypy), 155  
 wnvald() (in module spiceypy.spiceypy), 155  
 writln() (in module spiceypy.spiceypy), 155

## X

xf2eul() (in module spiceypy.spiceypy), 156  
 xf2rav() (in module spiceypy.spiceypy), 156  
 xfmsta() (in module spiceypy.spiceypy), 156  
 xpose() (in module spiceypy.spiceypy), 157  
 xpose6() (in module spiceypy.spiceypy), 157  
 xposeg() (in module spiceypy.spiceypy), 157