## 1.5 measuresdata - Module

Methods to obtain IERS data and make tables
include "measuresdata.g"

### Description

Module measuresdata provides scripts to maintain and update global data used by the Measures system. These include predicted and final IERS earth orientation data, JPL planetary ephemerides, leap second data, observatory positions, Earth magnetic field models, and others. The global data are stored as AIPS++tables, and are required by the Measures system for certain coordinate conversions, or to obtain the highest possible precision.
Scripts to convert external ephemerides (e.g. for comets and satellites) into a table are also provided (createcomet and createfromnorad.
A separate Glish script (`measuresiers.g`) is available to create and update all the IERS tables in one go. It could e.g. be used in a regularly running cron job (every 3 days) as:

```
glish -l measuresiers.g
```

When updating tables, the following should be observed:

- all data files, tables and scripts mentioned reside at $AIPSROOT/code/trial/apps/measures and can be copied from there if necessary
- all Glish scripts should be run in the directory where the tables that have to be upgraded or created are situated (copy them if necessary, and use the `aipsrc` resource file to indicate to the system where they can be found)
- for the upgrades to work the table files (i.e. the `Table/table.*` files) should be writable (`chmod +644 Table/table.*` if necessary)

There are a number of main separate routines provided for the upgrade of the different tables. They are:

- taiutc: checks if update of the leap seconds table is required (it changes at most once per half year). If a change has been possible, it reads the `tai-utc.dat` file from the USNO, and converts it into a table `TAI_UTC` in the current directory. The rows in the table are MJD (start of the leap second change); dUTC (the leap second value as from the MJD); Offset (necessary for earlier dates) and Multiplier (also for early dates). The table in the system will in most cases be up-to-date. (currently ftp:maia.usno.navy.mil/ser7/tai-utc.dat).

- ierseop97: checks if an update of the final IERS Earth Orientation Parameters data is necessary (it changes roughly once every week). If change is feasible, it reads the C04 solution from the IERS, and creates/updates the `IERSeop97` table in the current directory. It contains dUT, x, y (polar motion), LOD (length-of-day), dPsi and dEps (nutation) final values with their errors up till one to two weeks ago. The table in the system starts at 1962 (as will be a table created from scratch). (currently ftp:hpiers.obspm.fr/iers/eop/eopc04/eop97c04.xx).

- ierseop2000: checks if an update of the final IERS Earth Orientation Parameters data using the IAU2000 nutation/precession theory is necessary (it changes roughly once every week). If change is feasible, it reads the C04 solution from the IERS, and creates/updates the `IERSeop2000` table in the current directory. It contains dUT, x, y (polar motion), LOD (length-of-day), dPsi and dEps (nutation) final values with their errors up till one to two weeks ago. The table in the system starts at 1962 (as will be a table created from scratch). (currently ftp:hpiers.obspm.fr/iers/eop/eopc04/eop97c04_IAU2000.xx).

- ierspredict: checks if an update of the IERS predictions is necessary (the predictions change about once every three days). If appropriate it reads the predicted data for dUT, x and y from the USNO file `mark3,out` and updates the `IERSpredict` table. It contains dUT, x and y predicted for the next 3 months. Also the file `gpsrapid.out` is read, providing predictions for the next two weeks for dEps and dPsi (in future the last data will be either calculated or taken from JPL data). The table in the system starts at early 1997. A table created from scratch will start with the earliest data available at the ftp site. (currently ftp:maia.usno.navy.mil/ser7/mark3.out and ftp:maia.usno.navy.mil/ser7/gpsrapid.out).

- ierspredict2000: checks if an update of the IERS predictions using the IAU2000 theory is necessary (the predictions change about once every three days). If appropriate it reads the predicted data for dUT, x and y from the USNO file `mark3,out` and updates the `IERSpredict2000` table. It contains dUT, x and y predicted for the next 3 months. Also the file `gpsrapid.out` is read, providing predictions for the next two weeks for dEps and dPsi (in future the last data will be either calculated or taken from JPL data). The table in the system starts at early 1997. A table created from scratch will start with the earliest data available at the ftp site. (currently ftp:maia.usno.navy.mil/ser7/mark3.out and ftp:maia.usno.navy.mil/ser7/gpsrapid.out).

- jplde: checks if an update is necessary. The tables do not change, but to limit space, the standard table provided with the system is limited to the period from 1990 up to at least two full years after the current date. In general the system table will obey these rules, and an update will not be done. Tables for both DE200 (the IAU standard) and DE405 (the most recent ones) are in the system. If data for other periods is needed, the function can be run with start and end dates. The AIPS++system contains data for the period 1960-2020. If data outside this period is needed, the limits in the Glish script have to be changed. The data will be loaded from JPL in that case (at 5-7Mb per 20 year span). The table rows have an MJD and a large vector giving the Chebyshev polynomial coefficients for the planets. Access of the table (as for the other tables) can be made using functions provided in the `MeasTable class`, although, of course, the table can be read directly. (currently: ftp:navigator.jpl.nasa.gov/ephem/export/ascii/ascpxxxx.yyy)

- addobservatory: can add and delete entries in the `Observatories` table. It backups the old table; sort the entries on the observatory name, and capitalises the name and position type. In principle the editing of the table can in future be done by the tablebrowser, although no checks on the validity of the data will then be done. (currently only updated by hand).

- addline: can add and delete entries in the `Lines` table. It backups the old table; sort the entries on the spectral name, and capitalises the name. In principle the editing of the table can in future be done by the tablebrowser, although no checks on the validity of the data will then be done. (currently only updated by hand).

- addsource: can add and delete entries in the `Sources` table manually, or using the USNO VLBI reference source frame. It backs up the old table; sort the entries on the source name, and capitalises the name and direction type. In principle the editing of the table can in future be done by the tablebrowser, although no checks on the validity of the data will then be done. (currently ftp:maia.usno.navy.mil/rorf/refframe.one)

- createcomet: can create a table of e.g. cometary data from an input file. The input file should be (at least for now) in the "p120" format.

- createfromnorad: can create a table of e.g. satellite data from an input file. The input file should be (at least for now) in the "TRAKSTAR" ECI format.

- igrf: checks if update of the International Geomagnetic Reference Field models table is required (it changes every 5 years, with maybe a small update in between). If a change has been possible, it reads the `IGRF.COF` file from the Goddard data center, and converts it into a table `IGRF` in the current directory. The rows in the table are MJD (start of every 5 year period); COEF, the Schmidt-normalised spherical harmonic coefficients of the International Geomagnetic Reference Field, produced by the International Association of Geomagnetism and Aeronomy,

Working Group V-8: Analysis of the Global and Regional Geomagnetic Field and its Secular Variation, in the sequence:
g10, g11, h11, g20, g21, h21, g22, h22, g30, g31, h31, , g32, h32, g33, h33, ..., h1010; dCOEF, the annual change in the coefficients. The table in the system will in most cases be up-to-date. (currently ftp:ftp.ngdc.noaa.gov/SolidEarth/MainfldMag/Models/IAGA/IGRF.COF)

The tables are used in measures calculations. The `TAI_UTC` table is essential for any direct or indirect (e.g. coordinate conversions) work with the `MEpoch` classes. The `DE200` and `DE405` tables are essential for positions within the solar system, and very accurate coordinate conversions. The other IERS tables are necessary for high precision coordinate conversion. For lower precision (at the single dish level) they are not necessary. The IGRF tables are used for Earth' magnetic field calculations.
All tables described here have the following keywords:

- VS_CREATE: the time of creation of the original table
- VS_DATE: the time of the last update of the table
- VS_VERSION: xxxx.yyyy, where yyyy is updated at every update; xxxx at every new build
- VS_TYPE: a descriptive text of the table contents
- dMJD: the step between rows in the table in time (days), or zero if irregular
- MJD0: the date of the first row in the table - dMJD

and have a Type=IERS.
The DE tables have a few dozen more constants, specifying data used in the calculations. Access is provided through the MeasTable module.
The other methods provided are helper functions.

**Example**

```
> glish -l 'measuresiers.g'
```

will update the existing (or create if non-existent) tables with Earth Orientation parameters TAI_UTC, IERSeop97, IERSpredict, IERSeop2000, IERSpredict2000, and the Solar system DE200 and DE404 tables.
The following example will only update (create) the prediction tables:

```
- include 'measuresdata.g'
- ierspredict()
- exit
```

**Functions**

| | |
|---|---|
| addline | Add/delete entries to the Lines table |
| addobservatory | Add/delete entries to the Observatories table |
| addsource | Add/delete entries to the Sources table |
| createcomet | Creates an epherimis table from text input |
| createfromnorad | Creates an ephemeris table from text input |
| ierseop2000 | Get Earth Orientation Parameters solution |
| ierseop97 | Get Earth Orientation Parameters solution |
| ierspredict | Get predicted Earth Orientation Parameters |
| ierspredict2000 | Get predicted Earth Orientation Parameters |
| igrf | Get IGRF models |
| jplde | Get solar system parameter |
| taiutc | Get leap seconds |

### 1.5.1 taiutc - Function
**Package general**
**Module measuresdata**

Get leap seconds

**Synopsis**
taiutc()

**Description**
Load the leap second table TAI_UTC from USNO database if necessary. The full table is loaded each time

**Example**

```
- include 'measuresdata.g'
- taiutc()
T
```

**Arguments**

**Returns**
bool

---

### 1.5.2 ierseop97 - Function

**Package general**
**Module measuresdata**

Get Earth Orientation Parameters solution

**Synopsis**
ierseop97()

**Description**
Load the table with the IERS 97 C04 solution for the Earth Orientation Parameters. The table used is IERSeop97. The table will start at 1962, and run as far as calculations by the IERS have proceeded.

**Example**

```
- include 'measuresdata.g'
- ierseop97()
T
```

**Arguments**

**Returns**
bool

---

### 1.5.3 ierseop2000 - Function

**Package general**
**Module measuresdata**

Get Earth Orientation Parameters solution

**Synopsis**
ierseop2000()

**Description**
Load the table with the IERS 97 C04 solution for the Earth Orientation Parameters using the IAU2000 theories. The table used is IERSeop2000. The table will start at 1962, and run as far as calculations by the IERS have proceeded.

**Example**

```
- include 'measuresdata.g'
- ierseop2000()
T
```

**Arguments**

**Returns**
bool

---

### 1.5.4 ierspredict - Function

| | |
|---|---|
| **Package** | **general** |
| **Module** | **measuresdata** |

Get predicted Earth Orientation Parameters

**Synopsis**
ierspredict()

**Description**
Load the predicted Earth Orientation Parameters in the IERSpredict table.

**Example**

```
- include 'measuresdata.g'
- ierspredict()
T
```

**Arguments**

**Returns**
bool

---

### 1.5.5 ierspredict2000 - Function

**Package general**
**Module measuresdata**

Get predicted Earth Orientation Parameters

**Synopsis**
ierspredict2000()

**Description**
Load the predicted Earth Orientation Parameters in the IERSpredict2000 table using the IAU20000 theories.

**Example**

```
- include 'measuresdata.g'
- ierspredict2000()
T
```

**Arguments**

**Returns**
bool

---

### 1.5.6 jplde - Function

**Package general**
**Module measuresdata**

Get solar system parameter

**Synopsis**
jplde(v, st, nd)

**Description**
Get Solar system parameters from the JPL DE databases. As a default the table loaded will be from 1990 till at least two years from now. The default is indicated by a zero end time.

**Example**

```
- include 'measuresdata.g'
- jplde(405)
T
```

**Arguments**

| | |
|---|---|
| v | JPL table number<br>Allowed:200 or 405<br>Default: 200 |
| st | start year<br>Allowed:between 1960 and 2020<br>Default: 1990 |
| nd | end year<br>Allowed:between 1960 and 2020<br>Default: 0 |

**Returns**
bool

---

### 1.5.7 addobservatory - Function

**Package** general
**Module** measuresdata

Add/delete entries to the Observatories table

**Synopsis**
addobservatory(nam, typ, long, lat, ht, src, com, delnam, ctable)

**Description**
Add or delete entries from the `Observatories` table. A backup table will be made, the entries will be sorted on the name specified (capitalised).

**Example**

```
- include 'measuresdata.g'
- addobservatory('myplace','itrf','149.55deg','-30.1455deg','567m',
+ 'taken from AE','precision unknown')
T
```

**Arguments**

| | |
|---|---|
| nam | name of observatory<br>Allowed:string |
| typ | position type<br>Allowed:known type like itrf, wgs84<br>Default: " |
| long | longitude or x<br>Allowed:angle or length quantity<br>Default: " |
| lat | latitude or y<br>Allowed:angle or length quantity<br>Default: " |
| ht | height or z<br>Allowed:length quantity<br>Default: " |
| src | some comment about source of data<br>Allowed:string<br>Default: " |
| com | comment, e.g. about precision<br>Allowed:string<br>Default: " |
| delnam | indicate if delete in stead of add<br>Allowed:bool<br>Default: F |
| ctable | indicate that an empty table should be created<br>Allowed:bool<br>Default: F |

**Returns**
bool

---

### 1.5.8 addline - Function
**Package general**
**Module measuresdata**

Add/delete entries to the Lines table

**Synopsis**
addline(nam, freq, src, com, delnam, ctable)

**Description**
Add or delete spectral line entries from the `Lines` table. A backup table will be made, the entries will be sorted on the name specified (capitalised).

**Example**

```
- include 'measuresdata.g'
- addline('OH1720','1720.53MHz',
+ 'taken from WSRT list','precision unknown')
T
```

**Arguments**

| | | |
|---|---|---|
| nam | name of spectral line<br>Allowed:string |
| freq | frequency<br>Allowed:any wave energy quantity<br>Default: ” |
| src | some comment about source of data<br>Allowed:string<br>Default: ” |
| com | comment, e.g. about precision<br>Allowed:string<br>Default: ” |
| delnam | indicate if delete in stead of add<br>Allowed:bool<br>Default: F |
| ctable | indicate that an empty table should be created<br>Allowed:bool<br>Default: F |

**Returns**
bool

---

### 1.5.9 addsource - Function
**Package general**
**Module measuresdata**

Add/delete entries to the Sources table

**Synopsis**
addsource(nami, typi, longi, lati, srci, comi, delnam, ctable, vref)

**Description**
Add or delete entries from the `Sources` table. A backup table will be made, the entries will be sorted on the name specified (capitalised). The entries will either be manually, or, with vref set, obtained from the refframe.one VLBI table maintained by USNO.

**Example**

```
- include 'measuresdata.g'
```

```
- addsource('mysrc','B1950','149.55deg','-30.1455deg')
T
- addsource(vref=T) # get sources from USNO
```

**Arguments**

| | |
|---|---|
| nami | name of source<br>Allowed:string<br>Default: unknown |
| typi | position type<br>Allowed:known type like J2000, B1950, GALACTIC<br>Default: " |
| longi | longitude or x<br>Allowed:angle quantity<br>Default: " |
| lati | latitude or y<br>Allowed:angle quantity<br>Default: " |
| srci | some comment about source of data<br>Allowed:string<br>Default: " |
| comi | comment, e.g. about precision<br>Allowed:string<br>Default: " |
| delnam | indicate if delete in stead of add<br>Allowed:bool<br>Default: F |
| ctable | indicate that an empty table should be created<br>Allowed:bool<br>Default: F |
| vref | indicate that sources to be taken from USNO<br>Allowed:bool<br>Default: F |

**Returns**
bool

---

### 1.5.10 createcomet - Function

**Package general**
**Module measuresdata**

Creates an epherimis table from text input

**Synopsis**
createcomet(nam, fil)

**Description**
Create a table with ephemeris information in the current directory, using a text file in (for now) the "p120" format.

**Example**

```
- include 'measuresdata.g'
- createcomet('VGEO', '../indata/vgeo.txt')
successful readonly open of auto-locked table VGEO: 7 columns, 35 rows
deleted table VGEO.old
successful creation of auto-locked table VGEO: 7 columns, 0 rows
creating table VGEO with zero rows
VGEO table 0003.0001 now with 35 entries till 1997/12/21
T
```

**Arguments**

| | |
|---|---|
| nam | name of table to be created<br>Allowed:string |
| fil | path to input ascii list<br>Allowed:path as string |

**Returns**
bool

---

### 1.5.11 createfromnorad - Function
**Package general**
**Module measuresdata**

Creates an ephemeris table from text input

**Synopsis**

createfromnorad(nam, fil)

**Description**
Create a table with ephemeris information in the current directory, using a text file in (for now) the "TRAKSTAR" ECI format.

**Example**

```
- include 'measuresdata.g'
- createfromnorad('GAL','ECI21263.509')
NORMAL: successful readonly open of default-locked table GAL:
 7 columns, 21 rows
NORMAL: deleted table GAL.old
NORMAL: successful creation of default-locked table GAL:
 7 columns, 0  rows
NORMAL: created plain table GAL with zero rows
NORMAL: GAL table 0009.0001 now with 21 entries till 2006/01/05
<fail>: non-numeric operand in expression: ([b[i][5], b[i][6],
b[i][7]] * [b[i][8], b[i][9], b[i][10]])
        File:   ./measuresdata.g, Line 2541
        Stack:              sum(), ./measuresdata.g line 2541
                      as_double(), ./measuresdata.g line 2541
                  createfromnorad()
```

Note the error at end, due to a glish error.

**Arguments**

| | |
|---|---|
| nam | name of table to be created<br>Allowed:string |
| fil | path to input ascii list<br>Allowed:path as string |

**Returns**
bool

---

### 1.5.12 igrf - Function

**Package general**
**Module measuresdata**

Get IGRF models

**Synopsis**
igrf(fmt)

**Description**
Load the international Geomagnetic Reference Field model table IGRF (or DGRF) from the Goddard data center. The full table is loaded each time an update is necessary. The `fmt` argument selects the current format available: 2 was until 1999; 1 is from 2000 (till 2005?).

**Example**

```
- include 'measuresdata.g'
- igrf()
T
```

**Arguments**

| | |
|---|---|
| fmt | format of table from ftp<br>Allowed:code: 1 most recent<br>Default: 1 |

**Returns**
bool