1.3 bimams - Module

Module for BIMA measurement set access and manipulation
include "bimams.g"

Description

`bimams` provides an interface to BIMA-specific measurement sets. It provides several convenience functions for accessing information in measurement set subtables. `bimams` is a vertically integrated tool built on top of `ms` (another way of looking at it is that a `bimams` tool contains an `ms` tool). Thus, the usefulness of `bimams` is that it supplies a simple interface for observers to access and manipulate measurement set information that may be utilized more by BIMA than other telescopes.

Overview of `bimams` tool functions:

miscellaneous timerange

miscellaneous totalintegrationtime fieldids

spectral window chanwidths, startfreqs, endfreqs, namespw, nchans, nspw, reavg, reffreqs, spwid, spwidsbyname, spwids, spwnames, spwnames startfreqs

polarization polartable, polarids stokesid

table access ddtable ddid ddids fieldtable mcaddids sourcetable spwtable subtable

underlying tools ms

Tools

bimams	tool for manipulating BIMA measurement sets

1.3.1 bimams - Tool

Package bima
Module bimams

tool for manipulating BIMA measurement sets

Description

bimams is a tool which is built on top of an ms tool and provides easy access to information frequently requested by BIMA users.

Constructors

bimams	Construct a bimams tool

Functions

chanwidths	Get the channel widths of the specified type of windows
ddid	get the DATA_DESCRIPTION ID (row number) associated with a specified spectral window id and a polarization or polarization id.
ddids	get the DATA_DESCRIPTION IDs (row numbers) associated with specified spectral window ids and polarization ids. The spectral window and polarization ids are or-ed within themselves and then anded together.
ddtable	get the DATA_DESCRIPTION table associated with the ms
doptable	get the DOPPLER table associated with the ms
endfreqs	get the frequencies of the last channels in the multi-channel data spectral windows.
fieldids	Get the 1-based field ids associated with the specified source
fieldtable	Get the FIELD subtable as a table tool
mcaddids	Given a list of 1-based DATA DESCRIPTION ids which describe multi-channel data spectral windows, get a list of 1-based DATA DESCRIPTION ids which correspond to the multi-channel average windows.
ms	get the ms tool contained within this bimams tool.
namespw	Assign a name to a spectral window.
namespwid	Assign a name to a spectral window by specifying its id
nchans	Get the number of channels in each multi-channel data spectral window
nspw	Get the number of spectral windows
polarids	Get a list of polarization ids
polartable	Get the POLARIZATION subtable
reavg	Recompute sideband averages and channel averages based on channel data flags
reffreqs	Get a list of spectral window reference frequencies
sourcetable	Get the SOURCE subtable as a table tool
spwid	Get a spectral window id
spwids	Get a list of spectral window ids by specifying a type of spectral window and, optionally, a sideband
spwidsbyname	Get a list of spectral window ids by specifying a list of spectral window names
spwnames	Get a list of spectral window names by specifying the type of spectral window
spwtable	Get the SPECTRAL_WINDOW subtable as a table tool
startfreqs	Get the frequencies of the first channels in the multi-channel data spectral windows.
stokesid	Get the system-defined id for the specified stokes parameter.
subtable	Get the specified subtable of the underlying measurement set
timerange	Get the maximum and minimum times in the MS
totalintegrationtime	Get the total integration time of this MS

bimams - Constructor

Package	**bima**
Module	**bimams**
Tool	**bimams**

Construct a bimams tool

Synopsis
bimams(msname)

Description
This is used to construct a `bimas` tool associated with a measurement set. A Glish variable containing the tool functions of bimams is returned.

Arguments

msname	Name of the measurement set on disk to associate with this bimams tool Allowed:Disk measurement set name

Returns
bimams tool

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('sgrb2n.ms');
```

---

bimams.chanwidths - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the channel widths of the specified type of windows

Synopsis
chanwidths(type)

Description
Get a Glish vector containing the channel widths (in Hz) for the specified type of windows. The order of the values in the returned vector is the same as the order of the type of windows specified in the SPECTRAL_WINDOWS subtable. For the multichannel data, the assumption is made that the channel

width is constant within a given spectral window (note though that the measurement set specification permits varying channel widths within a window; in general however, channel widths within a spectral window are constant for BIMA data). The actual values which are returned are the widths of the first channels in each window.

Arguments

type	type of spectral window Allowed:'s' (sibeband average), 'c' (multi-channel data), or 'a' (multi-channel average) Default: 'c'

Returns
A Glish vector of doubles containing the channel widths of the multi-channel data spectral windows.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the widths of the channels in the multi-channel data windows of
# the underlying ms
bms.chanwidths();
```

The result is [-48828.125 -3125000 -3125000 -48828.125 -3125000 -3125000 48828.125 3125000 3125000 48828.125 3125000 3125000]

---

bimams.ddid - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the DATA_DESCRIPTION ID (row number) associated with a specified spectral window id and a polarization or polarization id.

Synopsis
ddid(spwid, stokes, polid)

Description
Retrieve the (1-based) DATA_DESCRIPTION ID (row number) based on the SPECTRAL_WINDOW ID and stokes parameter or polarization id. Returns -1 if the specified description doesn't exist in the table.

Arguments

spwid	The 1-based id (row number) of the spectral window in the SPECTRAL_WINDOWS subtable Allowed:Integer
stokes	The stokes parameter associated with the data description. Not used if polid is specified Allowed:String representation of a valid stokes parameter. Default: YY
polid	The 1-based id (row number) of the desired polarization in the POLARIZATION subtable. If specified, stokes is ignored. Allowed:Integer > 0

Returns
Integer (-1 if requested id does not exist)

Example

```
# get a data description id based on a spectral window id and a stokes
# parameter
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
#  get the data description id associated with spectral window id 10
# a stokes parameter of YY
bms.ddid(spwid=10,stokes='YY');
```

The result is 10.

bimams.ddids - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the DATA_DESCRIPTION IDs (row numbers) associated with specified spectral window ids and polarization ids. The spectral window and polarization ids are or-ed within themselves and then anded together.

Synopsis
ddids(spwids, polids)

Description
Retrieve (1-based) DATA_DESCRIPTION IDs (row numbers) based on SPECTRAL_WINDOW IDs and polarization ids.

Arguments

spwids	The 1-based ids (row numbers) of the spectral windows in the SPECTRAL_WINDOWS subtable Allowed:Vector of integers Default: unset (all spectral windows)
polids	The 1-based ids (row numbers) of the desired polarizations in the POLARIZATION subtable. Allowed:Vector of integers Default: unset (all polarizations)

Returns
Vector of integers (empty vector of no specified ids are found)

Example

```
# get a list of data description ids
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
#  get the data description id associated with spectral window ids 5
# and 10 and polarizaion id 0
bms.ddid(spwids=[5,10],polids=0);
```

The result is [5,10].

---

bimams.ddtable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the DATA_DESCRIPTION table associated with the ms

Synopsis
ddtable()

Description
Retrieve the DATA_DESCRIPTION subtable as a table tool.

Returns
Table tool

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# browse its DATA_DESCRIPTION subtable
bms.ddtable().browse();
```

---

bimams.doptable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the DOPPLER table associated with the ms

Synopsis
doptable()

Description
Retrieve the DOPPLER subtable as a table tool.

Returns
Table tool

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# browse its DOPPLER subtable
bms.doptable().browse();
```

---

bimams.endfreqs - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the frequencies of the last channels in the multi-channel data spectral windows.

Synopsis
endfreqs()

Description
Retrieve a Glish vector of doubles representing the end (last channel) frequencies (in Hz) in the multi-channel spectral windows. The order of the frequencies is the same as the order of the multi-channel windows in the SPECTRAL_WINDOWS table.

Returns
Vector of Doubles

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the end frequencies of the multi-channel spectral windows
bms.endfreqs();
```

The result is `[9.3162917e+10 9.30201938e+10 9.29201938e+10 9.2862413e+10 9.27202078e+10 9.26202078e+10 9.61141944e+10 9.62569176e+10 9.63569176e+10 9.64146984e+10 9.65569036e+10 9.66569036e+10]`

---

bimams.fieldids - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the 1-based field ids associated with the specified source

Synopsis
fieldids(sname)

Description
This function returns a vector of 1-based field ids associated with the specified source.

Arguments

sname	The source to which to get the field ids for Allowed:String Default: None

Returns
Vector of Ints

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
# the the 1-based field ids for the source 1733-130
- bms.fieldids('1733-130');
1
```

---

bimams.fieldtable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the FIELD subtable as a table tool

Synopsis
fieldtable()

Description
Get the FIELD subtable as a table tool.

Arguments

Returns
Table tool

Example

```
- # construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
- # get the FIELD table
- ft := bms.fieldtable();
- # browse it
- ft.browse();
```

---

bimams.mcaddids - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Given a list of 1-based DATA DESCRIPTION ids which describe multi-channel data spectral windows, get a list of 1-based DATA DESCRIPTION ids which correspond to the multi-channel average windows.

Synopsis
mcaddids(mcd)

Description
Given a list of 1-based DATA DESCRIPTION ids which describe multi-channel data spectral windows, get a list of 1-based DATA DESCRIPTION ids which correspond to the multi-channel average windows.

Arguments

mcd	vector of 1-based data description ids which refer to multichannel data windows. Allowed:Vector of integers

Returns
Vector of integers. If any input DATA DESCRIPTION ids do not have multi-channel average associations, -1 is in the corresponding the position in the returned vector

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# data description ids 1,3,5 are multichannel data.  1 and 3 have
# corresponding multichannel averages, while 5 does not.
bms.mcaddids([1,3,5);
```

The result is [15,17,-1].

---

bimams.ms - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

get the ms tool contained within this bimams tool.

Synopsis
ms()

Description
Retrieve the `ms` tool contained within this `bimams` tool (recall that a `bimams`

tool is essentially a container for an `ms` tool which supplies convience functions for retrieving information from the underlying `ms` tool).

Returns
ms tool

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the contained ms
true_ms := bms.ms();
true_ms.name();
```

The result of the last command is `neptune.ms`.

---

bimams.namespw - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Assign a name to a spectral window.

Synopsis
namespw(sb, type, mcn, name)

Description
Assign a name to a spectral window (that is, insert a value into the NAME column of the specified window's row in the SPECTRAL_WINDOWS table). After assigning a name, other functions can be used to retrieve information on spectral windows by specifying their names. BIMA vertical integration tools require that user-specified spectral window names be unique (this is not a requirement of the measurement set specification, however). Therefore, this function will not assign a name to a spectral window if another window different from the specified one already has that name. In this case, this function will fail.

Arguments

sb	The sideband in which the window is located. Allowed:'u' (usb) or 'l' (lsb)
type	type of spectral window Allowed:'s' (sibeband average), 'c' (multi-channel data), or 'a' (multi-channel average)
mcn	the 1-based location of this window in the SPECTRAL_WINDOWS table with respect to other windows of the same type and sideband (usually NOT the same as its row number in this table). Not used if type='s' Allowed:Integer from 1 to the number of spectral windows of the specified type in the specified sideband Default: 1
name	the name to assign to this window Allowed:String

Returns
Boolean: T if naming was successful.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# name the third multichannel data window in the lower sideband HCN
bms.namespw(sb='l', type='c', mcn=3, name='HCN');
# The result of the previous command is T if there is no window already
# named HCN. If there is, the function will fail.

# Now get the spectral window id of the window just named by using its
# name

bms.spwidsbyname('HCN');

# the result of the previous command is the spectral window id (the
# row number of the spectral window in the SPECTRAL_WINDOWS table. For
# example, 22.
```

---

bimams.namespwid - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Assign a name to a spectral window by specifying its id

Synopsis
namespwid(spwid, name)

Description
Assign a name to a spectral window (that is, insert a value into the NAME column of the specified window's row in the SPECTRAL_WINDOWS table) by specifying the window's id (ie, row number in the SPECTRAL_WINDOWS table). After assigning a name, other functions can be used to retrieve information on spectral windows by specifying their names. BIMA vertical integration tools require that user-specified spectral window names be unique (this is not a requirement of the measurement set specification, however). Therefore, this function will not assign a name to a spectral window if another window different from the specified one already has that name. In this case, this function will fail.

Arguments

spwid	The 1-based id of the spectral window (ie, the row number of the window in the SPECTRAL_WINDOWS table) Allowed:Integer >= 1 and <= number of spectral windows
name	the name to assign to this window Allowed:String

Returns
Boolean: T if naming was successful.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# name window number 3 CS
bms.namespw(spwid=3, name='CS');
# The result of the previous command is T if there is no window already
# named CS. If there is, the function will fail.
```

---

bimams.nchans - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the number of channels in each multi-channel data spectral window

Synopsis
nchans()

Description
Get a Glish vector of integers of the number of channels in the multi-channel data spectral windows. The order of the values is the same as that of the multi-channel data windows in the SPECTRAL_WINDOWS subtable.

Returns
vector of integers of the number of channels for the multi-channel data spectral windows

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the number of channels in the multi-channel data spectral
# windows of this measurement set
bimams.nchans();
```

The result of the last command will be something like `[256 32 32 256 32 32 256 32 32 256 32 32]`

---

bimams.nspw - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the number of spectral windows

Synopsis
nspw()

Description
Get the number of spectral windows in the SPECTRAL_WINDOWS subtable. This includes all tpes of spectral windows; that is, the returned value is simply the number of rows in this subtable

Returns
Integer: the number of rows in the SPECTRAL_WINDOWS subtable

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the number of spectral windows in this measurement set
bimams.nspw();
```

The result of the last command will be something like `26`.

---

bimams.polartable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the POLARIZATION subtable

Synopsis
polartable()

Description
Get the POLARIZATION subtable associated with the underlying measurement set. The returned quantity is a table tool.

Returns
Table tool containing the POLARIZATION subtable

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# browse the POLARIZATION subtable associated with the underlying
# measurement set
bimams.polartable().browse();
```

---

bimams.polarids - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a list of polarization ids

Synopsis
polarids(stokes)

Description
Get a 1-based vector of integers representing the polarization ids (row numbers) from the POLARIZATION subtable by specifying a stokes parameter. If no such polarization exists in the subtable, an empty vector is returned.

Arguments

stokes	The stokes parameter associated with the polarization id (e.g., 'YY', 'I') Allowed:String

Returns
Vector of integers: the corresponding polarization ids or an empty vector if the specified polarization is not found

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the polarization id associated with the YY stokes parameter
bimams.polarid('YY');
```

The last command will return the corresponding row in the POLARIZATION subtable, for example `1`

---

bimams.reavg - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Recompute sideband averages and channel averages based on channel data flags

Synopsis
reavg(out, dosort, reset, verbosity)

Description
This function recomputes the sideband averages and channel averages based on the channel data flags. It is currently implemented in glish and is rather slow. The plan is to implement it in C++ so it will run faster. In general, it should be run after channel data have been flagged. It is meant to be the AIPS++ analog of MIRIAD's `uvwide` task.

Yet to do: `edge` and `blankf` parameters, as in `uvwide` which allow the user to discard edge channels when recomputing averages, still must be added. Note that in `uvwide`, the default value of `blankf` is 0.033 which is why `uvwide` run with this default will compute different averages than the current implementation of `reavg`.

Arguments

out	the output dataset, if not specified the input dataset is modified and no new dataset is written Allowed:String Default: ”
dosort	sort the data when doing subqueries, if F, will speed up execution, but should only be set to F if you are sure the dataset is already sorted in time-data description id-baseline order (the order in which mirfiller writes data when filling) Allowed:Boolean Default: T
reset	should all average data be recomputed? By default (reset=T), all averages recomputed regardless of the value of the existing flags, otherwise (reset=F), only averages with F flags are recomputed (i.e. averages flagged bad are left untouched). Note that setting reset=F will not increase the speed of the function. Allowed:Boolean Default: T
verbosity	Controls the level of informational messages, 0 = run quietly, higher numbers = more messages Allowed:Integer Default: 1

Returns
T or fail

Example

```
# flag some data
- af := autoflag('neptune.ms');
- af.setselect(clip=[expr='ABS YY',min=0.001,max=50]);
# construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('neptune.ms');
# recompute sideband averages and print lots of messages during execution
- bms.reavg('neptune.ms.reavg',verbosity=3);
T
```

---

bimams.reffreqs - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a list of spectral window reference frequencies

Synopsis
reffreqs()

Description
Get a vector of doubles representing the reference frequencies (in Hz) of all the spectral windows in the SPECTRAL_WINDOWS table in the order they appear in this table.

Returns
Vector of doubles: the spectral window reference frequencies (in Hz)

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the reference frequencies of the spectral windows from the
# underlying measurement set
bimams.reffreqs;
```

The last command will return the reference frequencies (in Hz), e.g.,
[9.31753682e+10 9.31170688e+10 9.30170688e+10 9.28748642e+10
9.28170828e+10 9.27170828e+10 9.61017432e+10 9.61600426e+10
9.62600426e+10 9.64022472e+10 9.64600286e+10 9.65600286e+10]

---

bimams.sourcetable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the SOURCE subtable as a table tool

Synopsis
sourcetable()

Description

Get the SOURCE subtable as a table tool.

Arguments

Returns
Table tool

Example

```
- # construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
- # get the SOURCE table
- st := bms.sourcetable();
- # browse it
- st.browse();
```

---

bimams.spwid - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a spectral window id

Synopsis
spwid(sb, type, mcn)

Description
Get a 1-based spectral window id (row number) from the SPECTRAL_WINDOW subtable by specifying a sideband, spectral window type, and the position of the desired spectral window relative to other windows of the same type in the same sideband. If no window matching the specified parameters is found, -1 is returned.

Arguments

sb	The sideband in which the desired window is located Allowed:String: 'l' (lsb) or 'u' (usb)	
type	The type of window Allowed:String: 's' (sideband average), 'c' (multi-channel data), or 'a' (multi-channel average)	
mcn	The 1-based position of the desired window with respect to other windows of the same type in the same sideband. Usually NOT the same as the row number in the SPECTRAL_WINDOWS table Allowed:Integer: >= 1 and <= the total number of windows of the specified type in the specified sideband. Not used if type='s' Default: 1	

Returns
Integer: the id of the specified window or -1 if not found

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the spectral window id of the the third multi-channel average
# window in the upper sideband
bms.spwid(sb='u',type='a',mcn=3);
```

The last command will return the corresponding row in the SPECTRAL_WINDOWS subtable, for example `7` or `-1` if the specified window does not exist.

---

bimams.spwidsbyname - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a list of spectral window ids by specifying a list of spectral window names

Synopsis
spwidsbyname(names)

Description
Get a 1-based vector of spectral window ids (row numbers) from the SPECTRAL_WINDOW subtable by specifying a vector of names of the windows. A vector of integers of the same length as the input name vector is returned. For names which are not found in the spectral window table, `-1` will occupy that place in the returned vector.

Arguments

names	The names of the spectral windows Allowed:Vector of strings

Returns
Vector of integers: the ids (row numbers) of the specified windows (-1 for names which do not exist in the table)
.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# assign spectral window number 3 the name of HCN
bms.namespwid(id=3,name='HCN');
# assign spectral window number 7 the name of CS
bms.namespwid(id=7,name='CS');
# get the spectral window ids of the window named CS, HCN, and SiO
bms.spwidsbyname(names = ['CS','HCN','SiO']);
```

The last command returns `[7, 3, -1]` (the user has not yet named a window SiO, which is why the final value in the vector is `-1`).

---

bimams.spwids - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a list of spectral window ids by specifying a type of spectral window and, optionally, a sideband

Synopsis

spwids(type, sb)

Description
Get a 1-based vector of spectral window ids (row numbers) from the SPECTRAL_WINDOW subtable by specifying a type of spectral window (sideband average, multi-channel data, multi-channel average) and, optionally, a sideband. A vector of integers of the ids of all windows of the specified type in the specified sideband is returned.

Arguments

type	The spectral window type Allowed:String: 'A' (all types), 'a' (multi-channel average), 'c' (multi-channel data), or 's' (sideband average)
sb	The sideband. Not used if type='s' Allowed:String: 'b' (both), 'l' (lsb), 'u' (usb) Default: 'b'

Returns
Vector of integers: the ids (row numbers) of all the windows of the specified type.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the a list of ids of all the multi-channel data windows in both
# sidebands
bms.spwids(type='c', sb='b');
# The previous command returns a vector of integers representing the
# ids of all the multi-channel data windows, e.g., [4 6 8 10 12 14 16 18
# 20 22 24 26].

# get the ids for the multi-channel data windows from the upper
# sideband only
bms.spwids(type='c', sb='u');
# the previous command returns [16 18 20 22 24 26]
```

bimams.spwnames - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get a list of spectral window names by specifying the type of spectral window

Synopsis
spwnames(type)

Description
Get a vector of strings representing the names of all the spectral windows of a specified type.

Arguments

type		The type of spectral window
	Allowed:	String: 'all' (all types), 'a' (multi-channel averages), 'c' (multi-channel data), 's' (sideband averages)
	Default:	'all'

Returns
Vector of strings: the names of the spectral windows of the specified type in the order the windows appear in the SPECTRAL_WINDOWS subtable

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the names of the multi-channel data windows
bms.spwnames(type='c');
```

The last command returns a vector of the names of the multi-channel data windows, e.g., `none none HCN none none CS none none SiO none none none`. `none` is the default entry in the NAME column.

---

bimams.spwtable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the SPECTRAL_WINDOW subtable as a table tool

Synopsis
spwtable()

Description
Get the SPECTRAL_WINDOW subtable as a table tool.

Arguments

Returns
Table tool

Example

```
- # construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
- # get the SPECTRAL_WINDOW table
- spwt := bms.spwtable();
- # browse it
- spwt.browse();
```

---

bimams.startfreqs - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the frequencies of the first channels in the multi-channel data spectral windows.

Synopsis
startfreqs()

Description

Retrieve a Glish vector of doubles representing the start (first channel) frequencies (in Hz) in the multi-channel spectral windows. The order of the frequencies is the same as the order of the multi-channel windows in the SPECTRAL_WINDOWS table.

Returns
Vector of doubles of start frequencies (in Hz) for the multi-channel data spectral windows

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the starting frequencies of the multi-channel spectral windows
bms.startfreqs();
```

The result is a vector of doubles representing the start frequencies for the multi-channel data windows, e.g., `[9.31753682e+10 9.31170688e+10 9.30170688e+10 9.28748642e+10 9.28170828e+10 9.27170828e+10 9.61017432e+10 9.61600426e+10 9.62600426e+10 9.64022472e+10 9.64600286e+10 9.65600286e+10]`.

---

bimams.stokesid - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the system-defined id for the specified stokes parameter.

Synopsis
stokesid(stokes)

Description
Get the system-defined id for the specified stokes parameter as defined in `aips/implement/Measures/Stokes.h`.

Arguments

stokes	The stokes parameter (e.g., 'YY', 'I', etc.) Allowed:String: a stokes parameter

Returns
Integer: The system defined stokes id (see
`aips/implement/Measures/Stokes.h`
`) or -1 if the stokes parameter is not recognized`

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# get the stokes id for stokes parameter YY
bms.stokesid('YY');
```

The last command returns `12`.

---

bimams.subtable - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the specified subtable of the underlying measurement set

Synopsis
subtable(tname, readonly)

Description
Retrieve the specified subtable from the underlying measurement set as a table tool. By default, the table is opened in read-only mode, but it can be opened in read-write mode by setting the readonly parameter to F.

Arguments

tname	name of the subtable Allowed:String
readonly	open the table in read-only mode (no writing allowed) Allowed:Boolean Default: T

Returns
Table tool containing the specified subtable

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
bms := bimams('neptune.ms');
# browse the WEATHER table
bms.subtable('WEATHER').browse();
```

---

bimams.timerange - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the maximum and minimum times in the MS

Synopsis
timerange()

Description
This function returns the maximum and minimum times in the MS. The thing returned is a record with fields `max` and `min`. Each field contains a quantity corresponding to the maximum and minimum times, respectively.

Arguments

Returns
A record containing the maximum and minimum times
.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
# get the maximum and minimum times
- tra := bms.timerange();
- tra
[max=[value=4.38575597e+09, unit=s], min=[value=4.38573665e+09, unit=s]]
# now convert the maximum time to something meaningful to a human
- dq.time(tra.max,form='ymd');
1997/11/09/01:32:52.170
```

---

bimams.totalintegrationtime - Function

Package	**bima**
Module	**bimams**
Tool	**bimams**

Get the total integration time of this MS

Synopsis
totalintegrationtime(gaptime)

Description
This function returns the total integration time in the MS.

Arguments

gaptime	The minimum time between scans, if <0, 2 times the maximum value in the INTERVAL column is used Allowed:Double Default: -1

Returns
Quantity
.

Example

```
# construct a bimams tool and assign it to a Glish variable named bms
- bms := bimams('1733-130');
```