Chapter 1

BIMA

BIMA specific modules and tools

include "bima.g"

Description
Package for BIMA-related functionality

1.1 mirfiller - Module

Module for filling MIRIAD data
include "mirfiller.g"

Description

This module contains the `mirfiller` tool, which is used to load data in MIRIAD format in AIPS++ Measurement Sets. See the mirfiller tool documentation for details.

1.1.1 mirfiller - Tool

Package	**bima**
Module	**mirfiller**

a tool for filling MIRIAD data
include "mirfiller.g"

Description

`mirfiller` tool, which is used to load data in MIRIAD format in AIPS++ Measurement Sets. When a `mirfiller` tool is created, it is attached to a Miriad dataset; it can subsequently be used to fill several output measurement sets, perhaps with different data from the input MIRIAD dataset. The actually filling is commenced by calling the fill function. Restricting the filling to specific spectral windows can be controlled via the select function. `mirfiller` tries to be smart about figuring out which spectral windows pass to the output measurment sets by default, depending on what data it finds in the dataset. (At the moment, it tries to decide which wideband channels to pass; see section on MIRIAD layout below.) The default behavior can be changed via the `defpass` parameter given either to the `mirfiller` constructor or the `select()` function. See the description of select for more details.

Example

```
include 'mirfiller.g'

# open a miriad dataset; watch the logger for a summary and
#   mirfiller's guess as to what windows you'll probably want.
frommir := mirfiller('3c273')

# load the windows that mirfiller thinks you'll want.
frommir.fill('3c273.ms')

# load all the windows
frommir.select(defpass='all');
frommir.fill('3c273all.ms');

# load only the sideband averages to an MS called 3c273wide.ms
frommir.select(defpass='none', sbandav='all')
frommir.fill('3c273wide.ms')

# load only the first four spectral line windows and their
#   corresponding averages
frommir.select(defpass='none', splwin=[1:4], winav=[1:4])
frommir.fill('3c273_1-4.ms');
```

MIRIAD Dataset Spectral Layout and its Mapping to the Measurement Set

Each record in a Miriad dataset organizes its correlation data by spectral windows and so-called wideband channels. Each Miriad spectral window contains data from a band of frequencies, broken up into some number of channels. (Because each channel, then, is fairly narrow in bandwidth, these correlations are often referred to as "narrow-band" data.) The wideband channels, in general, contain single correlations, averaged over some band at potentially arbitrary frequencies; that is, they are bolometer-like measurements.

In Miriad datasets written by the BIMA telescope, the wideband channels are used to store wide-band averages of the spectral windows. More specifically, the first wideband channel contains the average of all spectral windows channels associated with the lower sideband (the first *Nwin/2* windows), and the second wideband channel contains the average of the upper sideband windows. The remaining wideband channels contain the averages of each of the spectral windows (in the same order). Thus, if the BIMA Miriad dataset contains *Nwin* spectral windows, there will be *Nwin+2* wideband channels. The first *Nwin/2* windows will be from the lower sideband, and the rest, from the upper sideband.

When the Miriad data is converted to an MS, each spectral window and each wideband channel is loaded into a separate MS spectral window. Thus, for a BIMA dataset with Nwin windows, the output MS (by default) will contain *Nwin*2+2* spectral windows. If all narrow and wideband data are present, `mirfiller` will store the relationships into the output measurement set so that they can be used appropriately in the calibration process. The select function can be used to further control the windows that get written.

Which windows that should get passed by default (i.e. where they are not explicitly specified by the select function) can depend on what "flavor" visibility dataset it is; that is, more specifically, where its been in its processing history and what the user wants to do with it. For example, if the data is calibrated, then the wideband data is usually no longer needed. (Calibrated or uncalibrated, the window averages are usually not needed.) Furthermore, if it has gone through much processing in Miriad, it is possible that some of the windows or wideband channels may have been discarded; thus, the convention for wideband channels described above is no longer valid. Finally, it's possible that the dataset did not originate from the BIMA telescope, and thus, the convention for wideband channels does not apply. `mirfiller` attempts to detect the "flavor" of the input dataset and adjusts its default window selection accordingly. See the select function for details.

Polarization

A Miriad dataset stores correlations for different polarization correlation types (e.g. RR, RL, etc.) in separate records, even if they were observed simultaneously. (Thus, it is up to the Miriad I/O software to properly combine records to form new Stokes correlations.) This is in contrast with AIPS++ which can store polarizations observed at the same time in the same record. Furthermore, BIMA can only observe one polarization per baseline at a time. `mirfiller` by default will load visibilities of different polarizations into separate polarization setups; thus, each out MS record will contain a single polarization. This is most appropriate for uncalibrated BIMA data.

(Normally, the polarization data are edited and calibrated separately; then, the data is smoothed in time to approximate simultaneous observations.) However, when the data contains multiple, simultaneously-observed polarizations–e.g., VLA data or calibrated and time-averaged BIMA data–it is possible to collect the polarizations into a single polarization setup record using the `joinpol` option (see setoptions()). Thus, when `joinpol` is set to True, each MS record will contain the visibilities for all polarizations. One caveat of this option is that data for any polarization not found for a given baseline will be set to zero. Thus, if you use set `joinpol=T` when loading the raw (unaveraged) BIMA data, it will be N times larger than if it were set to false, where N is the number of different polarizations.

mirfiller - Constructor

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

Construct a mirfiller tool

Synopsis
mirfiller(mirfile, preview, defpass)

Description
This creates a mirfiller tool attached to a Miriad dataset. The Miriad dataset is opened and, by default, is scanned to determine its contents. A summary of the contents is then sent to the logger, listing useful information such as the number of array configurations found, the source names, and, most importantly, the frequency setups found.
Based on what it finds in the dataset, `mirfiller` will determine what "flavor" Miriad visibility dataset it is. Currently recognized are:

- Uncalibrated BIMA data: this is data that has a frequency layout in convention used by the BIMA telescope. (See "MIRIAD Dataset Spectral Layout and its Mapping to the Measurement Set" section of the mirfiller tool description.)
- Calibrated BIMA data: Currently, this "flavor" is matched for any BIMA dataset that appears to be missing spectral line windows or window averages. The rationale is that this is an indication that the data has been calibrated and the undesired data have been dropped.
- Generic Miriad data: This is any dataset that appears not to follow frequency layout at all.

After giving the summary, `mirfiller` will report which catagory above it believes this dataset belongs. Based on this determination, it will set the default window passing mode as appropriate. This mode can be overridden with via `defpass` parameter, either here or in the `select()` function. The major aim of this mode is to minimize the amount that the user must specify in terms window selection for the most common uses of the data.

Arguments

mirfile	the file name of the Miriad dataset
preview	if true, the file will be scanned in full to determine its contents. It may be helpful to set this to false for very large files. In this case, only the beginning of the file is scanned, and the default defpass value is based on the limited information read. Default: T
defpass	a string indicating the default window passing mode to use. This mode affects the which kinds of windows get passed when they are not explicitly specified by the select() function. Allowed: 'default' – let mirfiller determine which mode to use based on what is found in the data; 'rawbima' – use a default appropriate for uncalibrated BIMA data: pass all spectral line windows and both side band averages but no window averages; 'calbima' – use a default appropriate for calibrated data: pass all spectral line windows but no wideband averages; 'all' – pass all windows and wideband channels by default; 'none' – pass no windows or wideband channels by default, require that they be specified explicitly with select(). Default: 'default'

Returns
mirfiller tool

Example
This example attaches a mirfiller tool to the miriad dataset called 3c273 and assigns it to a Glish variable named `frommir`.

```
frommir := mirfiller('3c273')
```

This causes a summary to be sent to the logger which will look something like this:

```
Miriad UV dataset: 3c273
 Max. no. of visibility records:    990
 Max. no. of spectral line windows: 12 (Max no. of channels: 1280)
 Max. no. of wide-band channels:    14
 No. of array configurations:       1
 No. of polarizations:              1
 Time Range: 1997/11/02/19:44:55 - 1997/11/02/20:03:35
Sources:
     3C273      12:29:06.70     +02.03.08.60 (1 field)
Frequency Setup:
   Mode 6, 1280 line channels, 14 wide channels
 Window  #chans   start freq.     increment    bandwidth    rest freq.
    1      256   93.17534 GHz     -48.83 kHz   12.50 MHz    93.17378 GHz
    2       32   93.11704 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    3       32   93.01704 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    4      256   92.87483 GHz     -48.83 kHz   12.50 MHz    96.41298 GHz
    5       32   92.81705 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    6       32   92.71705 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    7      256   96.10183 GHz      48.83 kHz   12.50 MHz    96.41298 GHz
    8       32   96.16013 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
    9       32   96.26013 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
   10      256   96.40233 GHz      48.83 kHz   12.50 MHz    96.41298 GHz
   11       32   96.46011 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
   12       32   96.56011 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
Input looks like a raw BIMA dataset; setting default defpass="rawbima"
```

---

mirfiller.select - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

select data from the input Miriad dataset.

Synopsis

select(defpass, splwin, winav, sbandav)

Description

Correllation data stored in a Miriad dataset are stored in spectral windows and so-called wideband channels. In BIMA data, the latter are used to store wideband averages. BIMA data contains an average for each window and one for each entire sideband. This function allows one to select which spectral line windows (via splwin) and which wideband channels (via winav and sbandav) to pass.

As a way of minimizing the amount the user has to type in selecting windows, mirfiller supports a notion of a *default window passing mode*, which can be adapted to the data found in the Miriad dataset. Controlled via the defpass parameter, it specifies for each type of input window or channel what to pass (usually all or none) if it is not specified via the splwin, winav, or sbandav parameters. See defpass below for currently available modes.

Each call to this function completely overrides previously selections.

Arguments

defpass	a string indicating the default window passing mode to use. This mode affects the which kinds of windows get passed when they are not explicitly specified by the other parameters. See general description above for more details. Allowed: 'default' – use the mode that was set (or assumed) at construction (see constructor description for more information); 'rawbima' – use a default appropriate for uncalibrated BIMA data: pass all spectral line windows and both side band averages but no window averages; 'calbima' – use a default appropriate for calibrated data: pass all spectral line windows but no wideband averages; 'all' – pass all windows and wideband channels by default; 'none' – pass no windows or wideband channels by default, require that they be specified explicitly with select(). Default: 'default'
splwin	a list of one-based spectral window indicies to load. Default: [], the defpass parameter controls how this is interpreted
winav	a list of one-based window average indicies to load. An index of 1 means load the average of the first spectral window. Default: [], the defpass parameter controls how this is interpreted
sbandav	a string indicating which sideband averages to load. Allowed: 'default' – allow the defpass parameter to control sideband average selection; 'lsb' – the lower sideband; 'usb' – the upper sideband; 'all' – both sidebands; 'none' – neither sideband. Default: 'default'

Example

```
# open a miriad dataset; watch the logger for a summary and
#   mirfiller's guess as to what windows you'll probably want.
frommir := mirfiller('3c273')

# load the windows that mirfiller thinks you'll want.
frommir.fill('3c273.ms')
```

```
# load all data into '3c273all.ms'
frommir.select(defpass='all');
frommir.fill('3c273all.ms')

# load only the sideband averages to an MS called 3c273wide.ms
frommir.select(defpass='none', sbandav='all')
frommir.fill('3c273wide.ms')

# load only the first four spectral line windows and their
#   corresponding averages
frommir.select(defpass='none', splwin=[1:4], winav=[1:4])
frommir.fill('3c273_1-4.ms');
```

---

mirfiller.summary - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

return and/or print a summary of the contents of the Miriad dataset

Synopsis
summary(header, verbose, preview)

Description
This function provides a summary of the contents of the Miriad dataset. By default, this information is printed as logger messages (unless the verbose parameter is false). Normally, this will cause the dataset to be scanned in its entirety, if it hasn't been already, to extract the information; by setting preview=F, the scanning will be prevented, and thus the summary may be based only on the first timestep in the dataset. Note that if the `mirfiller` tool was constructed with `preview=T` (the default) the dataset will be scanned then; thus, the preview parameter subsequently passed to `summary()` function will be ignored (since all the needed information has been gathered).
Some of the information can also be returned as a record via the header parameter. Fields that will be loaded into this record include:

- `nwide` — the maximum number of wide-band channels
- `nspect` — the maximum number of spectral line windows
- `nchan` — the maximum (total) number of channels

- `narray` — the number of array configurations found
- `npol` — the number of polarizations found
- `pols` — an array of the polarizations found
- `starttime` — the UTC starting time in seconds
- `endtime` — the UTC ending time in seconds
- `nrec` — the total number of visibility records found
- `cormode` — the correlator mode of the first correlator setup
- `telescope` — the names of all the telescopes encountered in the dataset.

Arguments

header	a record to hold the summary information Default: [=] (allowing the caller to ignore the returned information)
verbose	If true, detailed information regarding contents will be sent to the logger. Default: T
preview	if true, the entire file will be read (if it wasn't already scanned when the tool was constructed) to extract information. Set this to false to defeat this behavior (e.g. for very large files). In this case, only the first record of the file will be read; thus, some information may be inaccurate if the dataset is a concatonation of several original datasets. Default: T

Example

```
frommir := mirfiller('3c273', preview=F, quiet=T);  # preview=F prevents full scanning
hdr := [=];
frommir.summary(hdr, preview=T)   # preview=T causes scanning to happen now
```

This causes a summary to be sent to the logger which will look something like this:

```
Miriad UV dataset: 3c273
 Max. no. of visibility records:    990
 Max. no. of spectral line windows: 12 (Max no. of channels: 1280)
 Max. no. of wide-band channels:    14
```

```
 No. of array configurations:       1
 No. of polarizations:              1
 Time Range: 1997/11/02/19:44:55 - 1997/11/02/20:03:35
Sources:
     3C273      12:29:06.70     +02.03.08.60 (1 field)
Frequency Setup:
   Mode 6, 1280 line channels, 14 wide channels
 Window  #chans   start freq.     increment    bandwidth    rest freq.
    1      256   93.17534 GHz     -48.83 kHz   12.50 MHz    93.17378 GHz
    2       32   93.11704 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    3       32   93.01704 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    4      256   92.87483 GHz     -48.83 kHz   12.50 MHz    96.41298 GHz
    5       32   92.81705 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    6       32   92.71705 GHz   -3125.00 kHz  100.00 MHz    96.41298 GHz
    7      256   96.10183 GHz      48.83 kHz   12.50 MHz    96.41298 GHz
    8       32   96.16013 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
    9       32   96.26013 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
   10      256   96.40233 GHz      48.83 kHz   12.50 MHz    96.41298 GHz
   11       32   96.46011 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
   12       32   96.56011 GHz    3125.00 kHz  100.00 MHz    96.41298 GHz
Input looks like a raw BIMA dataset; setting default defpass="rawbima"
```

Furthermore, the `hdr` record now holds some information:

```
- print hdr
[nwide=14, nspect=12, nchan=1280, cormode=6, npol=1, pols=YY, starttime=4.38522e+09,
telescope=BIMA, scanned=T, endtime=4.38522e+09, narray=1, nrec=990]
```

---

mirfiller.getoptions - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

return the current filler options

Synopsis
getoptions()

Description
`mirfiller` supports a variety of options that can be set to control the filling, and this function returns their state as a Glish record. The currently supported options are:

- `scanlim` — the scan time jump limit (quantity). If the jump in time between two consecutive Miriad records is greater than this limit the scan number that gets written out for that record will be incremented. A change in source will always increment the scan number unless scanlim is negative; in this case, all records are forced to have the same scan number. The default is 5 minutes (`'5min'`).

- `obslim` — the observation ID time jump limit (quantity). The observation ID is meant to delimit two tracks that might appear in the same file. If the jump in time between two consecutive Miriad records is greater than this limit the scan number that gets written out for that record will be incremented. The ID will always be incremented if there is a change in telescope or array configuration unless obslim is negative, in which case, all records will be forced to have the same observation ID. The default is four hours (`'4h'`).

- `tilesize` — the tiling size to use for storing data in the MS using the TiledStorageManager. If the value is ¡= 0, the standard (non-tiled) storage manager will be used. The default is 32.

- `verbose` — If true, extra messages will be sent to the logger.

- `wideconv` — a string representing the convention to use for interpreting the wideband channels. The allowed values are "bima" and "none." In the "bima" convention, the first two wideband channels are the lower & upper sideband averages and the remaining channels represent averages of each of the spectral line windows. Currently, this option only affects the writing out of window associations: if wideconv="bima", the associations linking wideband windows to spectral line windows will be written according to the convention.

- `joinpol` — If false (the default), visibilities of different polarization correlation types will be loaded into different records. This is appropriate for uncalibrated BIMA data. If this option is true, the different polarizations observed at the same time will be gathered into a output record. This is appropriate for BIMA polarization measurement data that has been calibrated and time-averaged or data from a telescope with dual-polarization receivers. See section on Polarization in mirfiller module description for further discussion.

- `tsyswt` — If true (the default), the system temperatures will be used to estimate the theoretical noise for the data which will subsequently be used (automatically) as weights during imaging. If false, system temperature weighting will be disabled. (System temperatures will be loaded regardless.)

- `compress` — If true, the data will be written in compressed form. Compression reduces the size of the output MS by about a factor of two but also limits the dynamic range to 65000:1 (well above what is achievable with typical BIMA data). The default is currently false.

- `updmodelint` — A time quantity representing how often a new planet model should be written. Default is 8 hours (`'8h'`). For observations of planets, a record is entered into the SOURCE_MODEL column of the SOURCE subtable which points to the planet model. A new model is written after every interval, 'updmodelint' has passed.

Example

The example shows a Glish session demonstrating `getoptions()` and `setoptions()`. Text to the right of the "- " represents the user input.

```
- frommir := mirfiller('3c273')
- frommir.getoptions()
[tilesize=32, scanlim=[value=5, unit=min], obslim=[value=4, unit=h],
wideconv=bima, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=F]
- frommir.setoptions(verbose=T, scanlim='2min')
T
- frommir.getoptions()
[tilesize=32, scanlim=[value=2, unit=min], obslim=[value=4, unit=h],
wideconv=bima, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
- frommir.setoptions(obslim='0.5h')
T
- frommir.getoptions();
[tilesize=32, scanlim=[value=2, unit=min], obslim=[value=0.5, unit=h],
wideconv=bima, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
- frommir.setoptions(updmodelint='30min');
T
- frommir.getoptions();
[tilesize=32, scanlim=[value=5, unit=min], obslim=[value=4, unit=h],
wideconv=bima, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=0.5, unit=h], verbose=F]
- frommir.setoptions(verbose=T, reset=T)
T
- frommir.getoptions();
[tilesize=32, scanlim=[value=5, unit=min], obslim=[value=4, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
```

mirfiller.setoptions - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

return the current filler options

Synopsis

setoptions(scanlim, obslim, tilesize, verbose, wideconv, joinpol, tsyswt, compress, updmodelint, reset)

Description

This function allows one to set a number of options that control the filling. Normally, if an option value is not provided, the option will retain its previously set value (or its default if it has not been set by the user). This allows one to set options independently. This behavior can be overridden with `reset=T` which will reset all option values to their default before the newly provided values are set.

Arguments

scanlim	the scan time jump limit. If the jump in time between two consecutive Miriad records is greater than this limit the scan number that gets written out for that record will be incremented. A change in source will always increment the scan number unless scanlim is negative; in this case, all records are forced to have the same scan number. Default: 5 minutes
obslim	the observation ID time jump limit. The observation ID is meant to delimit two tracks that might appear in the same file. If the jump in time between two consecutive Miriad records is greater than this limit the scan number that gets written out for that record will be incremented. The ID will always be incremented if there is a change in telescope or array configuration unless obslim is negative, in which case, all records will be forced to have the same observation ID. Default: 5 minutes
tilesize	the tiling size to use for storing data in the MS using the TiledStorageManager. If the value is ¡= 0, the standard (non-tiled) storage manager will be used. Default: 32
verbose	If true, extra messages will be sent to the logger. Default: F
wideconv	a string representing the convention to use for interpreting the wideband channels. Currently, this option only affects the writing out of window associations in the spectral window subtable. Allowed: 'bima' – assume that the first two wideband channels are the lower & upper sideband averages and the remaining channels represent averages of each of the spectral line windows; the output wideband and spectral line windows will be linked accordingly in the spectral window subtable. 'none' – assume no convention for the wideband channels, and therefore, encode no associations. Default: the convention detected when mirfiller was constructed
joinpol	If false (the default), visibilities of different polarization correlation types will be loaded into different records. This is appropriate for uncalibrated BIMA data. If this option is true, the different polarizations observed at the same time will be gathered into a output record. This is appropriate for BIMA polarization measurement data that has been calibrated and time-averaged or data from a telescope with dual-polarization receivers. See section on Polarization in mirfiller module description for further discussion. Default: F
tsyswt	If true (the default), the system temperatures will be used to estimate the theoretical noise for the data which will subsequently be used (automatically) as weights during imaging. If false, system temperature weighting will be disabled. (System temperatures will be loaded regardless.) Default: T
compress	If true, the data will be written in compressed form. Compression reduces the size of the output MS by about a factor of two but also limits the dynamic range to 65000:1 (well above what is achievable with typical BIMA data). Default: F
updmodelint	the time interval after which the model for planet observations should be updated. Default: 8 hours
reset	If true, all above options are reset to their default values before modifying them to the new values Default: F

Example
The example shows a Glish session demonstrating `getoptions()` and `setoptions()`. Text to the right of the "- " (and before the "#") represents the user input; the text after "#" explain what's happening.

```
- frommir := mirfiller('3c273')
- frommir.getoptions()                    # see the default options
[tilesize=32, scanlim=[value=5, unit=min], obslim=[value=4, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
- frommir.setoptions(verbose=T, scanlim='2min') # set two options
T
- frommir.getoptions();                   # see the results
[tilesize=32, scanlim=[value=2, unit=min], obslim=[value=4, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
- frommir.setoptions(obslim='0.5h')          # set an additional option
T
- frommir.getoptions()
[tilesize=32, scanlim=[value=120, unit=min], obslim=[value=0.5, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
- frommir.setoptions(updmodelint='120min') # set model update interval to 120 minutes
T
- frommir.getoptions()
[tilesize=32, scanlim=[value=120, unit=min], obslim=[value=0.5, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=2, unit=h], verbose=T]
- frommir.setoptions(verbose=T, reset=T)    # now reset and set an option
T
- frommir.getoptions();
[tilesize=32, scanlim=[value=5, unit=min], obslim=[value=4, unit=h],
wideconv=none, joinpol=F, tsyswt=T, planetfit=4, movfield=T,
updmodelint=[value=8, unit=h], verbose=T]
```

---

mirfiller.fill - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

write selected data into a measurement set

Synopsis
fill(msfile, verbose)

Description
This function does the actual work of converting the Miriad dataset to a Measurment Set. A measurement set with the given name will be created and filled with data (previously selected with the select function) from the input Miriad dataset. When the filling is complete, a summary of the number of visibilities, spectral windows, fields, and polarizations will be written to the logger.
The fill function can be called several times, usually between calls to select. This allows one to write different parts of the Miriad dataset into different output datasets.
Enabling the verbose option (via the setoptions function) will cause `fill()` to print extra messages about the data it finds in the Miriad dataset. In particular, it will print out the timestamp for each new scan. (A scan is a group of integrations taken close together in time. What exactly constitutes a scan is controlled by the scanlim option; see setoptions for details.)

Arguments

msfile	the file name of the ouput MS dataset
verbose	If true, extra messages will be sent to the logger during this fill process. Setting this to true is equivalent to running setoptions(verbose=T) just before calling fill() and then setting it back again after fill() is finished. If the verbose option has been turned on via setoptions(), then extra messages are sent regardless of the parameter given here. Default: F

Example

```
# open a miriad dataset
frommir := mirfiller('3c273')

# load all data into '3c273all.ms'
frommir.fill('3c273all.ms')

# load only the sideband averages to an MS called 3c273wide.ms
frommir.select(splwin='none', winav='none')
frommir.fill('3c273wide.ms')
```

```
frommir.done();
```

mirfiller.done - Function

Package	**bima**
Module	**mirfiller**
Tool	**mirfiller**

shut down this filler

Synopsis
done()

Description
This will close the input Miriad dataset and shut down this tool. If this is the last open mirfiller tool open, the server will be shutdown.

Example

```
- frommir := mirfiller('3c273')
- frommir.done()                    # oops.
F
- print frommir                     # frommir is no longer a mirfiller tool
F
```

1.1.2 mirfillertest - Function

Package	**bima**
Module	**mirfiller**

Function to test mirfiller

Synopsis
mirfillertest(summary, details, logsummary, verbose, fill, mirdata, modeldata)

Description

`mirfillertest` is a function to test mirfiller. By default, it uses datasets in the AIPS++ data repository, although the user can can specify other datasets to use for the test. The basic strategy of the tests is to fill a `Miriad` dataset into MS2 format and then to test the MS against a fiducial MS that has been filled by mirfiller in the past. Tests include comparing column names, column keywords, and column data for the main MS table as well as all the associated subtables (except the HISTORY subtable). The function returns `T` if all tests pass, `F` otherwise. The tests (including filling) take 15 seconds on an unloaded Linux system with a 1.5 GHz Pentium 4 processor. The function will create a scratch directory in the current directory called `mirfillertester.work` where it will fill the `Miriad` dataset. For the default dataset, this will require about 5.5 MB of disk space.

Arguments

summary	out	Record in which to store test results Allowed:Any valid record Default: [=]
details	out	Record in which to store column data when column tests fail Allowed:Any valid record Default: [=]
logsummary	in	Send a summary of the results to the logger? Allowed:Boolean Default: F
verbose	in	Controls the amount of logger output, 0=none, higher numbers=more Allowed:Int ¿= 0 Default: 1
fill	in	Fill data? Allowed:Boolean Default: T
mirdata	in	Miriad dataset to fill Allowed:Any valid Miriad dataset name Default: ” (defaults to dataset in data repository)
modeldata	in	Fiducial measurement set for comparison Allowed:Any valid measurement set name Default: ” (defaults to dataset in data repository)

Returns
Boolean or fail

Example

```
- include 'mirfiller.g'
- mft_summary := [=];
- mft_details := [=];
# run the tests, being somewhat verbose putting results in mft_summary and
# mft_details, and have a summary sent to the logger
- mirfillertest(mft_summary,mft_details,T);
NORMAL: defaultos (dos) ready
NORMAL: table system ready
NORMAL: Begin filling  to mirfillertester.work/sgrb2n.test.ms
NORMAL: Starting server mirfiller
NORMAL: Server started: /appl/aips++/weekly/linux_gnu/bin/mirfiller
(AIPS++ version: 1.7 (build #173))
NORMAL:
Summary of Miriad UV dataset:
/appl/aips++/weekly/data/bima/dev/miriad/sgrb2n.small.ll
 Max. no. of visibility records:    396
 Max. no. of spectral line windows: 8 (Max no. of channels: 1024)
 Max. no. of wide-band channels:    10
 No. of array configurations:       1
 Telescopes: BIMA
 Polarizations found: YY
 Time Range: 1997/11/08/20:18:38 - 1997/11/08/20:22:41
Sources:
    SGRB2N      17:47:19.79     -28.22.17.30 (1 field)
Frequency Setup:
   Mode 4, 1024 line channels, 10 wide channels
 Window  #chans   start freq.     increment    bandwidth    rest freq.
    1      128   109.84321 GHz    -390.62 kHz   50.00 MHz    109.84363
    GHz
    2      128   109.67824 GHz    -390.62 kHz   50.00 MHz    109.68034
    GHz
    3      128   109.59827 GHz    -390.62 kHz   50.00 MHz    109.59875
    GHz
    4      128   109.29134 GHz    -390.62 kHz   50.00 MHz    109.40000
    GHz
    5      128   112.78287 GHz     390.62 kHz   50.00 MHz    112.83351
    GHz
    6      128   112.94784 GHz     390.62 kHz   50.00 MHz    113.00000
    GHz
    7      128   113.02781 GHz     390.62 kHz   50.00 MHz    113.00000
    GHz
    8      128   113.33475 GHz     390.62 kHz   50.00 MHz    113.38305
    GHz
NORMAL: Input looks like a raw BIMA dataset; setting default
```

```
defpass="rawbima"
NORMAL: Starting mirfiller::fill
NORMAL: Accepted 396 input MIRIAD records.
Loaded 7128 data records for
       409464 visibilities,
       18 spectral windows,
       1 polarization,
       1 field, and
       1 array configuration.
NORMAL: Finished mirfiller::fill
       2.57 real        1.31 user        1.21 system
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms: 22 columns, 7128 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms: 22 columns, 7128 rows
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/ANTENNA: 8 columns, 10 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/ANTENNA: 8 columns, 10 rows
ANTENNA
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/DATA_DESCRIPTION: 3 columns, 18
rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/DATA_DESCRIPTION: 3 columns, 18 rows
DATA_DESCRIPTION
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/FEED: 12 columns, 10 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/FEED: 12 columns, 10 rows
FEED
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/FLAG_CMD: 8 columns, 0 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/FLAG_CMD: 8 columns, 0 rows
FLAG_CMD
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/FIELD: 9 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/FIELD: 9 columns, 1 rows
FIELD
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/OBSERVATION: 9 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/OBSERVATION: 9 columns, 1 rows
OBSERVATION
```

```
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/POINTING: 9 columns, 0 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/POINTING: 9 columns, 0 rows
POINTING
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/POLARIZATION: 4 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/POLARIZATION: 4 columns, 1 rows
POLARIZATION
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/PROCESSOR: 5 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/PROCESSOR: 5 columns, 1 rows
PROCESSOR
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/SPECTRAL_WINDOW: 17 columns, 18
rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/SPECTRAL_WINDOW: 17 columns, 18 rows
SPECTRAL_WINDOW
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/STATE: 7 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/STATE: 7 columns, 1 rows
STATE
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/SOURCE: 12 columns, 1 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/SOURCE: 12 columns, 1 rows
SOURCE
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/DOPPLER: 4 columns, 8 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/DOPPLER: 4 columns, 8 rows
DOPPLER
NORMAL: successful readonly open of default-locked table
mirfillertester.work/sgrb2n.test.ms/SYSCAL: 8 columns, 180 rows
NORMAL: successful readonly open of default-locked table
/appl/aips++/data/bima/dev/ms/sgrb2n.ll.small.ms/SYSCAL: 8 columns, 180 rows
SYSCAL
NORMAL: All tests passed
NORMAL: T
NORMAL: MAIN.COLNAMES: T
NORMAL: MAIN.COLKEYWORDNAMES.UVW: T
NORMAL: MAIN.COLKEYWORDNAMES.FLAG: T
```

```
NORMAL: MAIN.COLKEYWORDNAMES.FLAG_CATEGORY: T
NORMAL: MAIN.COLKEYWORDNAMES.WEIGHT: T
NORMAL: MAIN.COLKEYWORDNAMES.SIGMA: T
NORMAL: MAIN.COLKEYWORDNAMES.ANTENNA1: T
NORMAL: MAIN.COLKEYWORDNAMES.ANTENNA2: T
NORMAL: MAIN.COLKEYWORDNAMES.ARRAY_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.DATA_DESC_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.EXPOSURE: T
NORMAL: MAIN.COLKEYWORDNAMES.FEED1: T
NORMAL: MAIN.COLKEYWORDNAMES.FEED2: T
NORMAL: MAIN.COLKEYWORDNAMES.FIELD_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: MAIN.COLKEYWORDNAMES.INTERVAL: T
NORMAL: MAIN.COLKEYWORDNAMES.OBSERVATION_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.PROCESSOR_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.SCAN_NUMBER: T
NORMAL: MAIN.COLKEYWORDNAMES.STATE_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.TIME: T
NORMAL: MAIN.COLKEYWORDNAMES.TIME_CENTROID: T
NORMAL: MAIN.COLKEYWORDNAMES.DATA: T
NORMAL: MAIN.COLVALUES.UVW: T
NORMAL: MAIN.COLVALUES.FLAG: T
NORMAL: MAIN.COLVALUES.FLAG_CATEGORY: T
NORMAL: MAIN.COLVALUES.WEIGHT: T
NORMAL: MAIN.COLVALUES.SIGMA: T
NORMAL: MAIN.COLVALUES.ANTENNA1: T
NORMAL: MAIN.COLVALUES.ANTENNA2: T
NORMAL: MAIN.COLVALUES.ARRAY_ID: T
NORMAL: MAIN.COLVALUES.DATA_DESC_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.EXPOSURE: T
NORMAL: MAIN.COLKEYWORDNAMES.FEED1: T
NORMAL: MAIN.COLKEYWORDNAMES.FEED2: T
NORMAL: MAIN.COLKEYWORDNAMES.FIELD_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: MAIN.COLKEYWORDNAMES.INTERVAL: T
NORMAL: MAIN.COLKEYWORDNAMES.OBSERVATION_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.PROCESSOR_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.SCAN_NUMBER: T
NORMAL: MAIN.COLKEYWORDNAMES.STATE_ID: T
NORMAL: MAIN.COLKEYWORDNAMES.TIME: T
NORMAL: MAIN.COLKEYWORDNAMES.TIME_CENTROID: T
NORMAL: MAIN.COLKEYWORDNAMES.DATA: T
NORMAL: MAIN.COLVALUES.UVW: T
NORMAL: MAIN.COLVALUES.FLAG: T
NORMAL: MAIN.COLVALUES.FLAG_CATEGORY: T
NORMAL: MAIN.COLVALUES.WEIGHT: T
```

```
NORMAL: MAIN.COLVALUES.SIGMA: T
NORMAL: MAIN.COLVALUES.ANTENNA1: T
NORMAL: MAIN.COLVALUES.ANTENNA2: T
NORMAL: MAIN.COLVALUES.ARRAY_ID: T
NORMAL: MAIN.COLVALUES.DATA_DESC_ID: T
NORMAL: MAIN.COLVALUES.EXPOSURE: T
NORMAL: MAIN.COLVALUES.FEED1: T
NORMAL: MAIN.COLVALUES.FEED2: T
NORMAL: MAIN.COLVALUES.FIELD_ID: T
NORMAL: MAIN.COLVALUES.FLAG_ROW: T
NORMAL: MAIN.COLVALUES.INTERVAL: T
NORMAL: MAIN.COLVALUES.OBSERVATION_ID: T
NORMAL: MAIN.COLVALUES.PROCESSOR_ID: T
NORMAL: MAIN.COLVALUES.SCAN_NUMBER: T
NORMAL: MAIN.COLVALUES.STATE_ID: T
NORMAL: MAIN.COLVALUES.TIME: T
NORMAL: MAIN.COLVALUES.TIME_CENTROID: T
NORMAL: MAIN.COLVALUES.DATA: T
NORMAL: MAIN.KEYWORDFIELDNAMES: T
NORMAL: ANTENNA.COLNAMES: T
NORMAL: ANTENNA.COLKEYWORDNAMES.OFFSET: T
NORMAL: ANTENNA.COLKEYWORDNAMES.POSITION: T
NORMAL: ANTENNA.COLKEYWORDNAMES.TYPE: T
NORMAL: ANTENNA.COLKEYWORDNAMES.DISH_DIAMETER: T
NORMAL: ANTENNA.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: ANTENNA.COLKEYWORDNAMES.MOUNT: T
NORMAL: ANTENNA.COLKEYWORDNAMES.NAME: T
NORMAL: ANTENNA.COLKEYWORDNAMES.STATION: T
NORMAL: ANTENNA.COLVALUES.OFFSET: T
NORMAL: ANTENNA.COLVALUES.POSITION: T
NORMAL: ANTENNA.COLVALUES.TYPE: T
NORMAL: ANTENNA.COLVALUES.DISH_DIAMETER: T
NORMAL: ANTENNA.COLVALUES.FLAG_ROW: T
NORMAL: ANTENNA.COLVALUES.MOUNT: T
NORMAL: ANTENNA.COLVALUES.NAME: T
NORMAL: ANTENNA.COLVALUES.STATION: T
NORMAL: ANTENNA.KEYWORDFIELDNAMES: T
NORMAL: DATA_DESCRIPTION.COLNAMES: T
NORMAL: DATA_DESCRIPTION.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: DATA_DESCRIPTION.COLKEYWORDNAMES.POLARIZATION_ID: T
NORMAL: DATA_DESCRIPTION.COLKEYWORDNAMES.SPECTRAL_WINDOW_ID: T
NORMAL: DATA_DESCRIPTION.COLVALUES.FLAG_ROW: T
NORMAL: DATA_DESCRIPTION.COLVALUES.POLARIZATION_ID: T
NORMAL: DATA_DESCRIPTION.COLVALUES.SPECTRAL_WINDOW_ID: T
NORMAL: DATA_DESCRIPTION.KEYWORDFIELDNAMES: T
NORMAL: FEED.COLNAMES: T
```

```
NORMAL: FEED.COLKEYWORDNAMES.POSITION: T
NORMAL: FEED.COLKEYWORDNAMES.BEAM_OFFSET: T
NORMAL: FEED.COLKEYWORDNAMES.POLARIZATION_TYPE: T
NORMAL: FEED.COLKEYWORDNAMES.POL_RESPONSE: T
NORMAL: FEED.COLKEYWORDNAMES.RECEPTOR_ANGLE: T
NORMAL: FEED.COLKEYWORDNAMES.ANTENNA_ID: T
NORMAL: FEED.COLKEYWORDNAMES.BEAM_ID: T
NORMAL: FEED.COLKEYWORDNAMES.FEED_ID: T
NORMAL: FEED.COLKEYWORDNAMES.INTERVAL: T
NORMAL: FEED.COLKEYWORDNAMES.NUM_RECEPTORS: T
NORMAL: FEED.COLKEYWORDNAMES.SPECTRAL_WINDOW_ID: T
NORMAL: FEED.COLKEYWORDNAMES.TIME: T
NORMAL: FEED.COLVALUES.POSITION: T
NORMAL: FEED.COLVALUES.BEAM_OFFSET: T
NORMAL: FEED.COLVALUES.POLARIZATION_TYPE: T
NORMAL: FEED.COLVALUES.POL_RESPONSE: T
NORMAL: FEED.COLVALUES.RECEPTOR_ANGLE: T
NORMAL: FEED.COLVALUES.ANTENNA_ID: T
NORMAL: FEED.COLVALUES.BEAM_ID: T
NORMAL: FEED.COLVALUES.FEED_ID: T
NORMAL: FEED.COLVALUES.INTERVAL: T
NORMAL: FEED.COLVALUES.NUM_RECEPTORS: T
NORMAL: FEED.COLVALUES.SPECTRAL_WINDOW_ID: T
NORMAL: FEED.COLVALUES.TIME: T
NORMAL: FEED.KEYWORDFIELDNAMES: T
NORMAL: FLAG_CMD.COLNAMES: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.APPLIED: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.COMMAND: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.INTERVAL: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.LEVEL: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.REASON: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.SEVERITY: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.TIME: T
NORMAL: FLAG_CMD.COLKEYWORDNAMES.TYPE: T
NORMAL: FLAG_CMD.COLVALUES.APPLIED: T
NORMAL: FLAG_CMD.COLVALUES.COMMAND: T
NORMAL: FLAG_CMD.COLVALUES.INTERVAL: T
NORMAL: FLAG_CMD.COLVALUES.LEVEL: T
NORMAL: FLAG_CMD.COLVALUES.REASON: T
NORMAL: FLAG_CMD.COLVALUES.SEVERITY: T
NORMAL: FLAG_CMD.COLVALUES.TIME: T
NORMAL: FLAG_CMD.COLVALUES.TYPE: T
NORMAL: FLAG_CMD.KEYWORDFIELDNAMES: T
NORMAL: FIELD.COLNAMES: T
NORMAL: FIELD.COLKEYWORDNAMES.DELAY_DIR: T
NORMAL: FIELD.COLKEYWORDNAMES.PHASE_DIR: T
```

```
NORMAL: FIELD.COLKEYWORDNAMES.REFERENCE_DIR: T
NORMAL: FIELD.COLKEYWORDNAMES.CODE: T
NORMAL: FIELD.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: FIELD.COLKEYWORDNAMES.NAME: T
NORMAL: FIELD.COLKEYWORDNAMES.NUM_POLY: T
NORMAL: FIELD.COLKEYWORDNAMES.SOURCE_ID: T
NORMAL: FIELD.COLKEYWORDNAMES.TIME: T
NORMAL: FIELD.COLVALUES.DELAY_DIR: T
NORMAL: FIELD.COLVALUES.PHASE_DIR: T
NORMAL: FIELD.COLVALUES.REFERENCE_DIR: T
NORMAL: FIELD.COLVALUES.CODE: T
NORMAL: FIELD.COLVALUES.FLAG_ROW: T
NORMAL: FIELD.COLVALUES.NAME: T
NORMAL: FIELD.COLVALUES.NUM_POLY: T
NORMAL: FIELD.COLVALUES.SOURCE_ID: T
NORMAL: FIELD.COLVALUES.TIME: T
NORMAL: FIELD.KEYWORDFIELDNAMES: T
NORMAL: OBSERVATION.COLNAMES: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.TIME_RANGE: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.LOG: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.SCHEDULE: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.OBSERVER: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.PROJECT: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.RELEASE_DATE: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.SCHEDULE_TYPE: T
NORMAL: OBSERVATION.COLKEYWORDNAMES.TELESCOPE_NAME: T
NORMAL: OBSERVATION.COLVALUES.TIME_RANGE: T
NORMAL: OBSERVATION.COLVALUES.LOG: T
NORMAL: OBSERVATION.COLVALUES.FLAG_ROW: T
NORMAL: OBSERVATION.COLVALUES.OBSERVER: T
NORMAL: OBSERVATION.COLVALUES.PROJECT: T
NORMAL: OBSERVATION.COLVALUES.RELEASE_DATE: T
NORMAL: OBSERVATION.COLVALUES.SCHEDULE_TYPE: T
NORMAL: OBSERVATION.COLVALUES.TELESCOPE_NAME: T
NORMAL: OBSERVATION.KEYWORDFIELDNAMES: T
NORMAL: POINTING.COLNAMES: T
NORMAL: POINTING.COLKEYWORDNAMES.DIRECTION: T
NORMAL: POINTING.COLKEYWORDNAMES.ANTENNA_ID: T
NORMAL: POINTING.COLKEYWORDNAMES.INTERVAL: T
NORMAL: POINTING.COLKEYWORDNAMES.NAME: T
NORMAL: POINTING.COLKEYWORDNAMES.NUM_POLY: T
NORMAL: POINTING.COLKEYWORDNAMES.TARGET: T
NORMAL: POINTING.COLKEYWORDNAMES.TIME: T
NORMAL: POINTING.COLKEYWORDNAMES.TIME_ORIGIN: T
NORMAL: POINTING.COLKEYWORDNAMES.TRACKING: T
```

```
NORMAL: POINTING.COLVALUES.DIRECTION: T
NORMAL: POINTING.COLVALUES.ANTENNA_ID: T
NORMAL: POINTING.COLVALUES.INTERVAL: T
NORMAL: POINTING.COLVALUES.NAME: T
NORMAL: POINTING.COLVALUES.NUM_POLY: T
NORMAL: POINTING.COLVALUES.TARGET: T
NORMAL: POINTING.COLVALUES.TIME: T
NORMAL: POINTING.COLVALUES.TIME_ORIGIN: T
NORMAL: POINTING.COLVALUES.TRACKING: T
NORMAL: POINTING.KEYWORDFIELDNAMES: T
NORMAL: POLARIZATION.COLNAMES: T
NORMAL: POLARIZATION.COLKEYWORDNAMES.CORR_TYPE: T
NORMAL: POLARIZATION.COLKEYWORDNAMES.CORR_PRODUCT: T
NORMAL: POLARIZATION.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: POLARIZATION.COLKEYWORDNAMES.NUM_CORR: T
NORMAL: POLARIZATION.COLVALUES.CORR_TYPE: T
NORMAL: POLARIZATION.COLVALUES.CORR_PRODUCT: T
NORMAL: POLARIZATION.COLVALUES.FLAG_ROW: T
NORMAL: POLARIZATION.COLVALUES.NUM_CORR: T
NORMAL: POLARIZATION.KEYWORDFIELDNAMES: T
NORMAL: PROCESSOR.COLNAMES: T
NORMAL: PROCESSOR.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: PROCESSOR.COLKEYWORDNAMES.MODE_ID: T
NORMAL: PROCESSOR.COLKEYWORDNAMES.TYPE: T
NORMAL: PROCESSOR.COLKEYWORDNAMES.TYPE_ID: T
NORMAL: PROCESSOR.COLKEYWORDNAMES.SUB_TYPE: T
NORMAL: PROCESSOR.COLVALUES.FLAG_ROW: T
NORMAL: PROCESSOR.COLVALUES.MODE_ID: T
NORMAL: PROCESSOR.COLVALUES.TYPE: T
NORMAL: PROCESSOR.COLVALUES.TYPE_ID: T
NORMAL: PROCESSOR.COLVALUES.SUB_TYPE: T
NORMAL: PROCESSOR.KEYWORDFIELDNAMES: T
NORMAL: SPECTRAL_WINDOW.COLNAMES: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.MEAS_FREQ_REF: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.CHAN_FREQ: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.REF_FREQUENCY: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.CHAN_WIDTH: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.EFFECTIVE_BW: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.RESOLUTION: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.FREQ_GROUP: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.FREQ_GROUP_NAME: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.IF_CONV_CHAIN: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.NAME: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.NET_SIDEBAND: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.NUM_CHAN: T
```

```
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.TOTAL_BANDWIDTH: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.ASSOC_SPW_ID: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.ASSOC_NATURE: T
NORMAL: SPECTRAL_WINDOW.COLKEYWORDNAMES.DOPPLER_ID: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.MEAS_FREQ_REF: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.CHAN_FREQ: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.REF_FREQUENCY: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.CHAN_WIDTH: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.EFFECTIVE_BW: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.RESOLUTION: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.FLAG_ROW: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.FREQ_GROUP: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.FREQ_GROUP_NAME: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.IF_CONV_CHAIN: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.NAME: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.NET_SIDEBAND: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.NUM_CHAN: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.TOTAL_BANDWIDTH: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.ASSOC_SPW_ID: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.ASSOC_NATURE: T
NORMAL: SPECTRAL_WINDOW.COLVALUES.DOPPLER_ID: T
NORMAL: SPECTRAL_WINDOW.KEYWORDFIELDNAMES: T
NORMAL: STATE.COLNAMES: T
NORMAL: STATE.COLKEYWORDNAMES.CAL: T
NORMAL: STATE.COLKEYWORDNAMES.FLAG_ROW: T
NORMAL: STATE.COLKEYWORDNAMES.LOAD: T
NORMAL: STATE.COLKEYWORDNAMES.OBS_MODE: T
NORMAL: STATE.COLKEYWORDNAMES.REF: T
NORMAL: STATE.COLKEYWORDNAMES.SIG: T
NORMAL: STATE.COLKEYWORDNAMES.SUB_SCAN: T
NORMAL: STATE.COLVALUES.CAL: T
NORMAL: STATE.COLVALUES.FLAG_ROW: T
NORMAL: STATE.COLVALUES.LOAD: T
NORMAL: STATE.COLVALUES.OBS_MODE: T
NORMAL: STATE.COLVALUES.REF: T
NORMAL: STATE.COLVALUES.SIG: T
NORMAL: STATE.COLVALUES.SUB_SCAN: T
NORMAL: STATE.KEYWORDFIELDNAMES: T
NORMAL: SOURCE.COLNAMES: T
NORMAL: SOURCE.COLKEYWORDNAMES.POSITION: T
NORMAL: SOURCE.COLKEYWORDNAMES.DIRECTION: T
NORMAL: SOURCE.COLKEYWORDNAMES.PROPER_MOTION: T
NORMAL: SOURCE.COLKEYWORDNAMES.CALIBRATION_GROUP: T
NORMAL: SOURCE.COLKEYWORDNAMES.CODE: T
NORMAL: SOURCE.COLKEYWORDNAMES.INTERVAL: T
NORMAL: SOURCE.COLKEYWORDNAMES.NAME: T
```

```
NORMAL: SOURCE.COLKEYWORDNAMES.NUM_LINES: T
NORMAL: SOURCE.COLKEYWORDNAMES.SOURCE_ID: T
NORMAL: SOURCE.COLKEYWORDNAMES.SPECTRAL_WINDOW_ID: T
NORMAL: SOURCE.COLKEYWORDNAMES.TIME: T
NORMAL: SOURCE.COLKEYWORDNAMES.REST_FREQUENCY: T
NORMAL: SOURCE.COLVALUES.POSITION: T
NORMAL: SOURCE.COLVALUES.DIRECTION: T
NORMAL: SOURCE.COLVALUES.PROPER_MOTION: T
NORMAL: SOURCE.COLVALUES.CALIBRATION_GROUP: T
NORMAL: SOURCE.COLVALUES.CODE: T
NORMAL: SOURCE.COLVALUES.INTERVAL: T
NORMAL: SOURCE.COLVALUES.NAME: T
NORMAL: SOURCE.COLVALUES.NUM_LINES: T
NORMAL: SOURCE.COLVALUES.SOURCE_ID: T
NORMAL: SOURCE.COLVALUES.SPECTRAL_WINDOW_ID: T
NORMAL: SOURCE.COLVALUES.TIME: T
NORMAL: SOURCE.COLVALUES.REST_FREQUENCY: T
NORMAL: SOURCE.KEYWORDFIELDNAMES: T
NORMAL: DOPPLER.COLNAMES: T
NORMAL: DOPPLER.COLKEYWORDNAMES.DOPPLER_ID: T
NORMAL: DOPPLER.COLKEYWORDNAMES.SOURCE_ID: T
NORMAL: DOPPLER.COLKEYWORDNAMES.TRANSITION_ID: T
NORMAL: DOPPLER.COLKEYWORDNAMES.VELDEF: T
NORMAL: DOPPLER.COLVALUES.DOPPLER_ID: T
NORMAL: DOPPLER.COLVALUES.SOURCE_ID: T
NORMAL: DOPPLER.COLVALUES.TRANSITION_ID: T
NORMAL: DOPPLER.COLVALUES.VELDEF: T
NORMAL: DOPPLER.KEYWORDFIELDNAMES: T
NORMAL: SYSCAL.COLNAMES: T
NORMAL: SYSCAL.COLKEYWORDNAMES.ANTENNA_ID: T
NORMAL: SYSCAL.COLKEYWORDNAMES.FEED_ID: T
NORMAL: SYSCAL.COLKEYWORDNAMES.INTERVAL: T
NORMAL: SYSCAL.COLKEYWORDNAMES.SPECTRAL_WINDOW_ID: T
NORMAL: SYSCAL.COLKEYWORDNAMES.TIME: T
NORMAL: SYSCAL.COLKEYWORDNAMES.TSYS: T
NORMAL: SYSCAL.COLKEYWORDNAMES.TSYS_FLAG: T
NORMAL: SYSCAL.COLKEYWORDNAMES.BIMA_JYPERK: T
NORMAL: SYSCAL.COLVALUES.ANTENNA_ID: T
NORMAL: SYSCAL.COLVALUES.FEED_ID: T
NORMAL: SYSCAL.COLVALUES.INTERVAL: T
NORMAL: SYSCAL.COLVALUES.SPECTRAL_WINDOW_ID: T
NORMAL: SYSCAL.COLVALUES.TIME: T
NORMAL: SYSCAL.COLVALUES.TSYS: T
NORMAL: SYSCAL.COLVALUES.TSYS_FLAG: T
NORMAL: SYSCAL.COLVALUES.BIMA_JYPERK: T
NORMAL: SYSCAL.KEYWORDFIELDNAMES: T
```

1.1.3 miriadtoms - Function
Package bima
Module mirfiller

fill a measurement set from a Miriad dataset

Synopsis
miriadtoms(msfile, mirfile, defpass, verbose)

Description
This global function provides a one-step method for filling Miriad data in the most typical circumstances, such as with simple raw BIMA data or calibrated BIMA data. If one needs to take control of the data selection, create several measurement sets from a single miriad dataset, or set special loading options, one should create a mirfiller tool.

Arguments

msfile	the output measurement set name
mirfile	the input Miriad dataset name
defpass	a string indicating the window passing mode to use. This mode affects the which kinds of windows get selected. Allowed: 'default' – let mirfiller determine which mode to use based on what is found in the data; 'rawbima' – use a default appropriate for uncalibrated BIMA data: pass all spectral line windows and both side band averages but no window averages; 'calbima' – use a default appropriate for calibrated data: pass all spectral line windows but no wideband averages; 'all' – pass all windows and wideband channels. Default: 'default'
verbose	if true, send lots of messages to the logger. A summary of the input miriad dataset will be printed, and filling will be done in verbose mode. Default: F