

## parmdbm

A parameter table is used by BBS to store its parameter values. The format is described in detail in [parmdb.pdf](#).

The table consists of two parts:

- The DEFAULTVALUES table containing default values for parameters. Usually they are used as the starting values for BBS when solving for parameters. Note that the default can be defined for a group of parameters. For example, a default for `gain:`, unless a more specific default is present. Note that the default value also defines if a parameter is solved for as scalar values or as a polynomial. See the [BBS documentation](#) for more info on parameter solving.
- The actual parameter values that are valid for a given freq/time domain. Normally they are filled in by BBS, but if needed, they can be manipulated as described below.

The parmdbm program can be used to operate on a parameter table. It can perform the following tasks:

- show parameter names
- show parameter values (actual or default)
- add a parameter value (actual or default)
- change a parameter value (default only)
- remove a parameter value (actual or default)
- export constant actual parameter values as default values to another parameter table (for cross calibration)

The interface is text based and quite basic, but it is the only way to create (default) values for the BBS parameters. Retrieval operations on a parameter table, in particular retrieving values for plotting purposes, can be done with the `lofar.parmdb` python module.

In principle parmdbm is interactive. Input redirection, for example

```
parmdbm < afile
```

where `afile` contains commands, can be used to create a parameter table in batch mode.

Furthermore note that parmdbm operates on a single parameter table. To operate on a distributed parameter table the tool `setupparmdb` or, if needed, `rundist` can be used.

## Commands

Several commands are available in parmdbm. Most commands can or must be followed by a parameter name (or Unix shell-type name pattern) telling which parameter(s) to show, add, update, or remove.

In commands where a parameter name pattern can be used, it defaults to `*` (i.e., all).

Almost all commands can or must be followed by one or more keyword=value pairs (e.g., telling the parameter table name in `open`) separated by commas. Optional keywords are shown in *italic* and the default value is given after the `=`.

A few examples are given at the end. Furthermore the `?` command can be used to get some help info.

The following table lists the commands (and synonyms), if a parameter name (pattern) needs to be

given, and the keywords it uses. The various keywords are described in more detail in a later table.

command	parmname	keywords	description
?			Give some help info.
open		tablename	Open a parameter table. If one is still open, it should be closed first.
create		tablename	Create an empty parameter table.
close			Close the open parameter table. It is done implicitly at the end.
quit exit stop EOF			End the parmdbm session.
adddef newdef insertdef	name	<i>type</i> <i>shape/nx/ny=1</i> <i>values</i> <i>mask</i> <i>pert=1e-6</i> <i>pertrel=false</i>	Add a new parameter default to the DEFAULTVALUES table. Note that BBS uses a default value if the parameter table does not contain a value for the given parameter and freq/time domain. The defaulting mechanism works such that it first looks for the entire parameter name. If not found, it strips off the last part (colons are used as separators), looks again, etc. until found. If not found at all, it uses 0. In this way a single default can be used for a bunch of similar parameters. The type given defines if BBS will solve a parameter as a polynomial or as a scalar.
updatedef	pattern	<i>type</i> <i>shape/nx/ny</i> <i>values</i> <i>mask</i> <i>pert</i> <i>pertrel</i>	Update the default values of parameters matching the name pattern. Only the keywords given are updated.
deletedef removedef erasedef	pattern		Remove the parameters from the DEFAULTVALUES table matching the name pattern.
showdef listdef	pattern (*)		Show the parameters in the DEFAULTVALUES table matching the name pattern.
namesdef	pattern (*)		Show the names of parameters in the DEFAULTVALUES table matching the name pattern.
add new insert	name	<i>type=""</i> <i>domain</i> <i>shape/nx/ny</i> <i>values</i> <i>mask=true</i> <i>errors=0</i>	Add a scalar or polynomial parameter with the given domain and value.
delete remove erase	pattern	<i>domain</i>	Remove the values of parameters matching the name pattern and domain.
show list	pattern (*)	<i>domain</i>	Show the parameters values table matching the name pattern and domain.
names	pattern (*)		Show the names of parameters in the table matching the name pattern.

command	parmname	keywords	description
range	pattern (*)		Show the overall range (in freq and time) of the parameters matching the name pattern.
export	pattern (*)	tablename <i>append=0</i>	Export values of parameters matching the name pattern to the parameter table with the given name. Optionally they can be appended to the table. It makes it possible to apply the solutions of a calibration observation to a target observation done at a different time. Only the following parameter values will be exported: - A constant parameter value (thus not varying in time or freq) is exported as a default value. - A parameter value varying in freq but not in time, is exported with the time domain set to infinite, so it can be applied to another time range. Note that if the time/freq domains of calibration and target observations are the same (e.g. by using multiple beams), no export needs to be done. The program <code>parmexportcal</code> can be used if the parameter values vary in time, see its <a href="#">documentation</a> .
clear			clear the table; all parameter values and defaults are removed.
set		<i>stepx</i> <i>stepy</i>	Overwrite one or both default grid steps to be used in <code>lofar.parmdb</code> for polynomial parameters. Initially <code>stepx=1000Hz</code> and <code>stepy=1s</code> .

Usually `parmdbm` will only be used to define some parameter default values as shown in the example at the end. Occasionally it will be used to look at a parameter value, although the `lofar.parmdb` python interface is much better suited for that. It will hardly ever be necessary to insert or update an actual value.

## Keywords

Keyword	Type	Description
type	string	Defines the type of parameter. <code>polc</code> means a 2-dim polynomial parameter (usually in freq/time) for which the coefficients have to be given in keyword values. Keyword <code>shape</code> defines the polynomial order in x and y. If the type is not given, the parameter has scalar values.
shape	integer vector	Defines the shape of the values in keyword values. For polynomials it defines the order+1. For scalars it defines the number of scalars given; they are placed on an regular grid in the box given by keyword domain. For example <code>shape=[2,3]</code> means <code>nx=nfreq=2</code> <code>ny=ntime=3</code> . Note that the shape can also be defined using keywords <code>nx</code> and/or <code>ny</code> . If none of them are given, the shape defaults to <code>[1,1]</code> .
nx	integer	Another way to define the shape in x (freq). It defaults to 1 if keyword <code>shape</code> is not given.
ny	integer	Another way to define the shape in y (time). It defaults to 1 if keyword <code>shape</code> is not given.

Keyword	Type	Description
domain	vector	The domain specifies the freq/time range for which a parameter value is valid or in which parameter values have to be searched. In principle an N-dim domain can be specified, but for BBS only 2-dim domain (freq/time) are used. A domain can be specified in two ways: 1. domain=[stx,endx,sty,endy,...] 2. domain=[st=[x,y,...], end=[x,y,...]] Instead of end one can also use size. In principle the values have to be given as floating point values; times in MJD seconds, frequencies in Hz. However, when using the first way, it is possible to specify times in the casacore MVTime format by enclosing it in quotes. For example: domain=[1e6, "15Mar2010/11:32:00", 1.1e6, "15Mar2010/11:33:00"]
values	double vector	The values (polynomial coefficient or scalars). They have to be given in order v[0,0], v[1,0], ... v[0,1], ... v[nx-1,ny-1].
errors	double vector	The errors on the values as found by the solver. Normally this is filled in by BBS.
mask	bool vector	An optional mask for polynomial coefficients. It tells which coefficients have to be solved for when solving for the parameter. The default is that the higher order coefficients are set to false, thus mask[i,j] = i+j<max(nx,ny)
pert	double	The perturbation to use for numerical derivatives. It defaults to 1e-6.
pertrel	bool	Tells if the perturbation is absolute or relative. For angles it should be absolute, otherwise relative. It defaults to true.

## Example

```
create tablename='parm.tab'
adddef phase values=0
adddef gain values=[1,0,0,0], nx=2, ny=2
showdef
quit
```

It creates a new parmtable and adds two defaults parameters. Phase will be a scalar parameter, while gain is a 2-dim polynomial. The parameters will not be used as such in BBS, but serve as parameter groups. BBS uses parameter names like phase:ST0:X. In this way only 2 parameter defaults have to be defined to serve all possible gain and phase parameters.

From:  
<https://www.astron.nl/lofarwiki/> - **LOFAR Wiki**

Permanent link:  
[https://www.astron.nl/lofarwiki/doku.php?id=public:user\\_software:documentation:parmdbm](https://www.astron.nl/lofarwiki/doku.php?id=public:user_software:documentation:parmdbm)

Last update: **2017-03-08 15:27**

