Introduction

This page describes the syntax of the BBS configuration file (a.k.a. parset). BBS consists of three main executables: `GlobalControl`, `KernelControl`, and `SolverControl`. Both `KernelControl` and `SolverControl` need only a very small subset of the information that can be put in the parset (see examples below). The `GlobalControl` parset is usually much larger because it describes the processing that BBS has to do. For testing purposes, it is often useful to run all three executables on a single node. In that case it is possible to create a single combined parset, because each executable will ignore all keys it does not understand.

Below you will find typical example parset for each of the three executables. The rest of this page discusses all the valid keys in more detail.

Example GlobalControl

```
Observation = L2007_03463.gds    # Global VDS-file

Strategy.InputColumn = DATA    # Input column
Strategy.Stations = []    # Stations to use in the calibration (all if empty)
Strategy.TimeWindow = []    # Time selection (date/time, all if empty)
Strategy.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Strategy.Correlation.Type = []    # Polarization product selection (all if empty)
Strategy.ChunkSize = 100    # Chunk size (timeslots) [SEE DOCUMENTATION]
Strategy.UseSolver = T    # Use global solver
Strategy.Steps = [solve_realimag, subtract, correct]

BBDB.Key = run00    # (Unique) name identifying the calibration session
BBDB.Host = cepmaster0    # Hostname or IP-address of postgresql server
BBDB.Port = 5432    # Port number where the postgresql server is listening
BBDB.Name = john    # Name of the database to use
BBDB.Username = postgres    # Username for accessing the postgresql server
BBDB.Password =     # Password for accessing the postgresql server

Step.solve_realimag.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.solve_realimag.Baselines.Station2 = []    #
Step.solve_realimag.Model.Sources = []    # Sources to include in the model (all if empty)
Step.solve_realimag.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in th
Step.solve_realimag.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.solve_realimag.Correlation.Type = []    # Polarization product selection (all if empty)
Step.solve_realimag.Operation = SOLVE    # Operation to perform
Step.solve_realimag.OutputData =     # Output column (no output if empty)
Step.solve_realimag.Solve.Parms = ["Gain:{11,22}:*"]    # Parameters to fit
Step.solve_realimag.Solve.ExclParms = []    # Parameters to exclude
Step.solve_realimag.Solve.CalibrationGroups = [2,3]    # Calibration group definition [SEE DOCUMENTA
Step.solve_realimag.Solve.CellSize.Freq = 0    # Solution cell size (channels)
Step.solve_realimag.Solve.CellSize.Time = 1    # Solution cell size (timeslots)
Step.solve_realimag.Solve.CellChunkSize = 10    # Cell chunk size (timeslots)
Step.solve_realimag.Solve.PropagateSolutions = T    # Propagate solutions to the next cell chunk?
Step.solve_realimag.Solve.Options.MaxIter = 10    # Maximal number of iterations
Step.solve_realimag.Solve.Options.EpsValue = 1e-9    # Convergence criterion
Step.solve_realimag.Solve.Options.EpsDerivative = 1e-9    # Convergence criterion
Step.solve_realimag.Solve.Options.ColFactor = 1e-9    # Colinearity factor
Step.solve_realimag.Solve.Options.LMFactor = 1.0    # Levenberg-Marquardt factor
Step.solve_realimag.Solve.Options.BalancedEqs = F    # Assume balanced equations?
Step.solve_realimag.Solve.Options.UseSVD = T    # Use singular value decomposition?

Step.subtract.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.subtract.Baselines.Station2 = []    #
Step.subtract.Model.Sources = []    # Sources to include in the model (all if empty)
Step.subtract.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in the mode
Step.subtract.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.subtract.Correlation.Type = []    # Polarization product selection (all if empty)
Step.subtract.Operation = SUBTRACT    # Operation to perform
Step.subtract.OutputData =     # Output column (no output if empty)

Step.correct.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.correct.Baselines.Station2 = []    #
Step.correct.Model.Sources = ["CasA"]    # Sources to include in the model (all if empty)
Step.correct.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in the model
Step.correct.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.correct.Correlation.Type = []    # Polarization product selection (all if empty)
Step.correct.Operation = CORRECT    # Operation to perform
Step.correct.OutputData = CORRECTED_DATA    # Output column (no output if empty)
```

Example KernelControl

```
ObservationPart.Filesystem = lioff021:/dev/sda10    # File system on which the part of the observati
ObservationPart.Path = /data/L2007_03463_SB0.MS    # Absolute path to the part of the observation to

BBDB.Key = run00    # (Unique) name identifying the calibration session
BBDB.Host = cepmaster0    # Hostname or IP-address of postgresql server
BBDB.Port = 5432    # Port number where the postgresql server is listening
BBDB.Name = john    # Name of the database to use
BBDB.Username = postgres    # Username for accessing the postgresql server
BBDB.Password =     # Password for accessing the postgresql server

ParmDB.Instrument = L2007_03463_SB0.instrument    # Instrument model parameters
ParmDB.Sky = L2007_03463_SB0.sky    # Sky model parameters
```

Example SolverControl

```
PortRange = [6500, 6599]  # Port range to try on start-up
ConnectionBacklog = 25    # Maximal number of pending connections

BBDB.Key = run00    # (Unique) name identifying the calibration session
BBDB.Host = cepmaster0    # Hostname or IP-address of postgresql server
BBDB.Port = 5432    # Port number where the postgresql server is listening
BBDB.Name = john    # Name of the database to use
BBDB.Username = postgres    # Username for accessing the postgresql server
BBDB.Password =     # Password for accessing the postgresql server
```

Global settings

BBDB [Relevant to `GlobalControl`, `KernelControl`, `SolverControl`]

> Information about the BlackBoard database.
>
> **Key** : *string*
>
> > Name that identifies the session.
>
> **Host** : *string*
>
> > Hostname or IP-address of the database server.
>
> **Port** : *integer*
>
> > Port number on which the database server is listening. For Postgres databases the default is 5432.
>
> **Name** : *string*
>
> > Name of the database.
>
> **User** : *string*
>
> > Username to access the database server.
>
> **Password** : *string*
>
> > Password to access the database server.

Observation : *string* [Relevant to `GlobalControl`]

> Global VDS file that describes the parts that compose the observation to be processed.

ObservationPart [Relevant to `KernelControl`]

> Describes the part (MS) of the observation to be processed.
>
> **Filesystem** : *string*

File system on which the part of the observation to process is located. **NB. Must match the file system specified in the global VDS-file exactly.**

Path : *string*

Absolute path to the part of the observation to process. **NB. Must match the path specified in the global VDS-file exactly.**

ParmDB [Relevant to `KernelControl`]

Information about the parameter databases (e.g. instrument model parameters, sky model parameters).

Instrument : *string*

Path to the instrument model parameter database.

Sky : *string*

Path to the sky model parameter database.

PortRange : *vector<integer>* [Relevant to `SolverControl`]

On start-up the specified range of ports will be searched for a free port on which to start listening.

ConnectionBacklog : *integer* [Relevant to `SolverControl`]

Maximal number of pending connections. The default is 10, but this may have to be increased when starting a large number of KernelControl processes.

Strategy

The strategy defines the operations that need to be performed on the data. It consists of one or more (multi-)steps.

InputColumn : *string*

Name of the column in the observation part that contains the input data. The default is “DATA”.

Stations : *vector<string>*

Names of the participating stations. All stations will be used if this field is left empty. The stations names must match the names stored in the measurement set. Shell style wildcards are recognized (?,*,{}).

```
Strategy.Stations = [CS010_dipole0, CS008*, CS00{1,16}_dip??e{0,4,8,12}]
```

Correlation : *Correlation*

Specifies which baselines and which polarization products to use.

TimeWindow : *vector<string>*

Time range to process, expressed as date/time string(s) (as returned by *msinfo*). All timeslots will be used if this field is left empty. Either a range or a start time can be provided.

```
Strategy.TimeWindow = [27-Jul-2007/16:05:04]
Strategy.TimeWindow = [27-Jul-2007/16:05:04, 28-Jul-2007/13:05:04]
```

ChunkSize : *integer*

Chunk size in timeslots. A chunk represents an amount of input data that is loaded into memory and processed as a whole. This is useful when the amount of visibility data is too large to fit into main memory. A value of zero means all.

UseSolver : *bool*

Will a global solver be used in this strategy?

Steps : *vector<string>*

The names of the steps that compose the strategy. It is an error to leave this field empty.

Step

A *single-step* describes one unit of work in the strategy. A step that is defined in terms of a number of other steps is known as a *multi-step*. The attributes of a multi-step should be interpreted as *default values* for the steps that compose the multi-step. These default values can always be overridden.

Steps : *vector<string>*

> The names of the steps that compose this step (for multi-steps), or absent (for single steps).

Baselines : *Baselines*

> Baselines to use.

Model : *Model*

> Model configuration to use.

Correlation : *Correlation*

> Specifies which baselines and which polarization products to use.

Operation : *string*

> The operation to be performed in this step. One of 'PREDICT', 'ADD', 'SUBTRACT', 'CORRECT', 'SOLVE'. Only relevant for single steps, should be absent for multi-steps.
> PREDICT - Simulate visibilities.
> ADD - Add simulated visibilities to the input visibilities.
> SUBTRACT - Subtract predicted visibilities from the input visibilities.
> CORRECT - Correct the input visibilities.
> SOLVE - Fit model parameters.

OutputColumn : *string*

> Column in the observation part wherein the output values of this step should be written. If left empty, no data will be written.

Single steps should define one of the following fields, depending on the value of ***Operation****:*

> **Solve** : *Solve*
>
> > Arguments of the SOLVE operation.

Example

```
Step.MultiStepExample.Steps = [solve_realimag, subtract]    # Steps that compose the multi-step.
Step.MultiStepExample.Baselines.Station1 = [CS016*, CS001*]    # Baseline selection (all if empty)
Step.MultiStepExample.Baselines.Station2 = [{CS008*,CS001*}, CS010*]

Step.solve_realimag.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.solve_realimag.Baselines.Station2 = []    #
Step.solve_realimag.Model.Sources = []    # Sources to include in the model (all if empty)
Step.solve_realimag.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in th
Step.solve_realimag.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.solve_realimag.Correlation.Type = []    # Polarization product selection (all if empty)
Step.solve_realimag.Operation = SOLVE    # Operation to perform
Step.solve_realimag.OutputData =     # Output column (no output if empty)
Step.solve_realimag.Solve.Parms = ["Gain:{11,22}:*"]    # Parameters to fit
Step.solve_realimag.Solve.ExclParms = []    # Parameters to exclude
Step.solve_realimag.Solve.CalibrationGroups = []    # Calibration group definition [SEE DOCUMENTATIO
Step.solve_realimag.Solve.CellSize.Freq = 0    # Solution cell size (channels)
Step.solve_realimag.Solve.CellSize.Time = 1    # Solution cell size (timeslots)
Step.solve_realimag.Solve.CellChunkSize = 10    # Cell chunk size (timeslots)
Step.solve_realimag.Solve.PropagateSolutions = T    # Propagate solutions to the next cell chunk?
Step.solve_realimag.Solve.Options.MaxIter = 10    # Maximal number of iterations
Step.solve_realimag.Solve.Options.EpsValue = 1e-9    # Convergence criterion
Step.solve_realimag.Solve.Options.EpsDerivative = 1e-9    # Convergence criterion
Step.solve_realimag.Solve.Options.ColFactor = 1e-9    # Colinearity factor
Step.solve_realimag.Solve.Options.LMFactor = 1.0    # Levenberg-Marquardt factor
Step.solve_realimag.Solve.Options.BalancedEqs = F    # Assume balanced equations?
Step.solve_realimag.Solve.Options.UseSVD = T    # Use singular value decomposition?

Step.subtract.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.subtract.Baselines.Station2 = []    #
Step.subtract.Model.Sources = []    # Sources to include in the model (all if empty)
Step.subtract.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in the mode
Step.subtract.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.subtract.Correlation.Type = []    # Polarization product selection (all if empty)
Step.subtract.Operation = SUBTRACT    # Operation to perform
Step.subtract.OutputData =     # Output column (no output if empty)

Step.correct.Baselines.Station1 = []    # Baseline selection (all if empty)
Step.correct.Baselines.Station2 = []    #
Step.correct.Model.Sources = ["CasA"]    # Sources to include in the model (all if empty)
Step.correct.Model.Components = [DIRECTIONAL_GAIN]    # Instrumental effects to include in the model
Step.correct.Correlation.Selection = CROSS    # Baseline selection (AUTO/CROSS/ALL)
Step.correct.Correlation.Type = []    # Polarization product selection (all if empty)
Step.correct.Operation = CORRECT    # Operation to perform
Step.correct.OutputData = CORRECTED_DATA    # Output column (no output if empty)
```

Correlation

Specifies which baselines and which polarization products to use.

Selection : *string*

> Baselines to use. Should be one of 'AUTO', 'CROSS', or 'ALL'.
>
> > AUTO - Use only auto correlations.
> > CROSS - Use only cross correlations.
> > ALL - Use auto and cross correlations both.

Type : *string*

> Polarization products to use. Should be one or more of 'XX', 'XY', 'YX', 'YY'. As an example, suppose we select 'XX' here and set **Selection** to 'AUTO', then the correlation of the X polarization signal of each station with itself will be used. However, if we set **Selection** to 'CROSS' then the correlation of the X polarization signal of station *A* with the X polarization signal of station *B* for each baseline *(A,B)* will be used (where A not equal to B).

Baselines

A baseline is a pair of stations. The first station of the pair is contained in **Station1**, the second in **Station2**. For example, suppose we want to select the following six baselines: (A, B), (A, C), (A, D), (B, C), (B, D), (C, D). Then **Station1** would contain [A, A, A, B, B, C] and **Station2** would contain [B, C, D, C, D, D]. The lengths of **Station1** and **Station2** should always be equal. If both fields are left empty, all baselines will be used. Shell style wildcards are recognized (?,*,{}).

Station1 : *vector<string>*

> One name for each baseline: the first station in the pair that forms the baseline.

Station2 : *vector<string>*

> One name for each baseline: the second station in the pair that forms the baseline.

```
Baselines.Station1 = [CS010*, "{CS008_dipole{4,8,12}, CS016*}"]
Baselines.Station2 = [CS008*, CS016_dipole?]

Step.solve0.Baselines.Station1 = ["{RT0,RT1,RT2,RT3,RT4,RT5,RT6}"]
Step.solve0.Baselines.Station2 = ["{RTA,RTB,RTC,RTD}"]
```

Model

Model configuration.

UsePhasors : *bool*

> If set to true, complex parameters are expressed as (amplitude, phase) components instead of (real, imaginary) components. As a consequence the model is extended with a conversion for each complex parameter from (amplitude, phase) to (real, imaginary).

Sources : *vector<string>*

> List of sources to include in the model. Shell style wildcards are allowed.

Components : *vector<string>*

> List of components to include in the model. Currently the following components are supported:
>
> *BANDPASS*
>
> > Multiply the sum of the source coherences by a *real diagonal matrix*. The naming convention for the associated parameters is `"Bandpass:{11,22}:<station name>"`.
>
> *GAIN*
>
> > Multiply the sum of the source coherences by a *complex matrix*. The naming convention for the associated parameters is `"Gain:{11,12,21,22}:{Real,Imag}|{Ampl,Phase}:<station name>"`. Depending on the value of **UsePhasors** either `{Real,Imag}` or `{Ampl,Phase}` is chosen.
>
> *DIRECTIONAL_GAIN*
>
> > Multiply the source coherence of each source with a source (direction) specific *complex matrix*. The naming convention for the associated parameters is `"Gain:{11,12,21,22}:{Real,Imag}|{Ampl,Phase}:<station name>:<source name>"`. Depending on the value of **UsePhasors** either `{Real,Imag}` or `{Ampl,Phase}` is chosen.
>
> *BEAM*
>
> > Multiply the source coherence of each source with a direction dependent *complex matrix* that is computed based on a specified beam model. The naming convention for the associated parameters is `"Orientation:<station name>"`.
>
> *IONOSPHERE*
>
> > Introduces a direction dependent phase shift that is computed based on a global (planar) phase screen. The naming convention for the associated parameters is `"MIM:{0,1,2,3,4,5}"`.

Beam.Type : *string*

> Select the beam model to use when **BEAM** is included in **Components**. The following beam models are currently supported:
>
> *HamakerDipole*
>
> > Model of the dipole voltage beam. This model is an implementation of J.P. Hamaker's memo "Mathematical-physical analysis of the generic dual-dipole antenna".
> >
> > **Beam.HamakerDipole.CoeffFile** : *string*
> >
> > > Path to a file with beam coefficients (obtained from fitting a numerical simulation to a polynomial basis).
>
> *YatawattaDipole*
>
> > Model of the dipole voltage beam based on Sarod Yatawatta's analytical model.
> >
> > **Beam.YatawattaDipole.ModuleTheta** : *string*
> >
> > > Path to a loadable module that computes the theta dependent part of the beam response.
> >
> > **Beam.YatawattaDipole.ModulePhi** : *string*
> >
> > > Path to a loadable module that computes the phi dependent part of the beam response.

Solve

Parms : *vector<string>*

> Parameters to fit. Shell style wildcards are recognized (?,*,{}).

```
Solve.Parms = ["gain:{11,22}:*"]
```

ExclParms : *vector<string>*

> Subset of the parameters selected by **Parms** that should *not* be fitted. For example, if we would like to solve for the gain (amplitude, phase) of each station, but we would also like to fix the phase of the first station (say, CS010_dipole0) this can be specified as follows:

```
Solve.Parms = ["gain:*"]
Solve.ExclParms = ["gain:*:phase:CS010_dipole0"]
```

CalibrationGroups : *vector<int>*

> A list of numbers that specifies the partitioning of kernels into calibration groups. The n-th number in the list gives the *number of kernels* in the n-th calibration group. For example, [3,1] signifies that there are two calibration groups, where the kernels with id 0, 1, 2, and 3 are assigned to calibration group 0 and the kernels with id 4 are assigned to calibration group 1. The sum of the numbers in the list must always equal the total number of KernelControl processes that participate in the calibration. An empty list means that there are no interdependencies and therefore each kernel can use it's own solver. The global solver will not be used in this case.

CellSize.Freq : *int*

> Solution grid cell size (frequency): A chunk is divided into solutions cells and a solution is computed for each cell independently. This parameter specifies the (nominal) number of channels that are grouped together in a cell. A value of zero (0) selects the entire frequency range. **NB.** When performing a global solve this parameter is ignored and the frequency range of the calibration group is used instead.

CellSize.Time : *int*

> Solution grid cell size (time): A chunk is divided into solutions cells and a solution is computed for each cell independently. This parameter specifies the (nominal) number of timeslots that are grouped together in a cell. A value of zero (0) selects the entire time range.

CellChunkSize : *int*

> Specifies the number of solution cells *along the time axis* to process simultaneously. A value of zero (0) selects all solution cells in the current chunk. Can be used to tune memory usage.

PropagateSolutions : *bool*

> If set to true, then the solutions of the previous *cell chunk* are used as initial values for the next *cell chunk*. **NB.** Even if set to true, solutions are *not* propagated to the next *chunk of visibility data*. Furthermore, no convergence check is done, so bad solutions may contaminate the solutions in the next *cell chunk* if this parameter is set to true.

Options.MaxIter : *int*

> Convergence criterion: Iteration is halted if the number of iterations exceeds this value. **NB.** There is a known off by one bug, so in practice a value of *x* results in *x+1* iterations.

Options.EpsValue : *double*

> Convergence criterion: Iteration is halted if the minimal relative change in the norm of the solutions is smaller than this value.

Options.EpsDerivative : *double*

> Convergence criterion: Iteration is halted if the maximum of the absolute difference between the observed and the predicted visibilities is smaller than this value.

Options.ColFactor : *double*

> Colinearity factor. Used to test for solvability of the normal equations. If not solvable an error is returned, unless **Options.UseSVD** is set to true (in which case Singular Value Decomposition is used to compute a solution). See http://aips2.nrao.edu/docs/scimath/implement/Fitting/LSQFit.html#LSQFit.

Options.LMFactor : *double*

> Initial Levenberg-Marquardt factor, which controls the balance between gradient descent and Newton-Raphson optimization. See also http://aips2.nrao.edu/docs/notes/224 (bottom of the page).

Options.BalancedEqs : *bool*

> If set to true, the Levenberg-Marquardt factor is added ("in some way?") to the diagonal elements of the normal equation matrix. Otherwise, the diagonal elements are multiplied by (1 + factor). See http://aips2.nrao.edu/docs/scimath/implement/Fitting/LSQFit.html#LSQFit.

Options.UseSVD : *bool*

> If set to true, Singular Value Decomposition is used to compute a solution when the normal equations cannot be triangularized.