

REAS 2.59 User's Manual

Tim Huege*

April 8, 2008

1 Introduction

REAS is a C++ code for the simulation of *Radio Emission from Air Showers*. It currently implements the geosynchrotron model [1; 3]. Inclusion of Cherenkov and end-point contributions is foreseen for the near future.

The initial version of the code, REAS1, calculated radio emission based on parameterised air showers. A number of results derived with REAS1 are available in the literature [4; 5].

The current version, REAS2, uses a sophisticated air shower model derived on a shower-to-shower basis from CORSIKA [2]. For details please consult [6].

The terms and conditions for the usage of REAS are detailed in section 7. In any case, please contact the authors if you find bugs or have programmed routines that would be useful to include in the REAS distribution.

2 Technical overview

Simulating radio emission from extensive air showers with REAS is a two-step process:

1. run a CORSIKA simulation for the parameters of interest
2. run a REAS simulation for a given CORSIKA simulation

This approach may seem somewhat complicated at first, yet it provides a number of advantages. In particular, changes in the radio emission model can be evaluated quickly without having to re-run the time-intensive CORSIKA simulations, and — maybe even more important — without being obscured by shower-to-shower fluctuations.

The following will describe the technical prerequisites in some more detail.

2.1 CORSIKA and COAST

To run simulations suitable as input for REAS, you will need two components: CORSIKA and COAST.

As of version 6.710, CORSIKA is already pre-configured appropriately to work with COAST version 3.0. Please download CORSIKA yourself from the official CORSIKA webpage¹.

*email: tim.huege@ik.fzk.de

¹<http://www-ik.fzk.de/corsika>

COAST is an interface code that fills histograms during the CORSIKA simulation run and stores them in a single ROOT data file per shower. A pre-configured version of COAST version 3.0 is available together with REAS from the REAS project webpage².

Last but not least, for COAST to run, ROOT³ has to be set up correctly. ROOT version 4 or 5 will work fine.

2.2 REAS

REAS is written in standard C++ and should compile on any Linux PC. It has no dependencies on external programme packages, with the exception of ROOT, which is needed to import the COAST-generated histogram files.

3 Installation

This section guides you through the installation steps necessary to run simulations with REAS. Please take special care of setting up all the environment variables correctly, both for compiling the codes and later on running the binaries. If you run into any problems, please double-check if all environment variables are set up correctly!

3.1 ROOT

ROOT has to be set up correctly for COAST and REAS to work. Please refer to the ROOT installation manual for further information. Please make sure that

- the environment variable *ROOTSYS* is set up correctly,
- *\$ROOTSYS/bin* is included in the *PATH*,
- and *\$ROOTSYS/lib* is included in the *LD_LIBRARY_PATH*

before continuing. (Note: Depending on your shell, after changing the *PATH* and *LD_LIBRARY_PATH* environment variables, you might have to execute the command *rehash*.)

3.2 COAST

After having downloaded the COAST package from the REAS project webpage, unzip and untar it into a new directory, e.g., */home/user/coast-v3r2*. In this example, set the environment variables

- *COAST_DIR* to the directory */home/user/coast-v3r2/install*
- *COAST_USER_LIB* to the directory */home/user/coast-v3r2/THRADIO*
- and include *\$COAST_DIR/lib* in the *LD_LIBRARY_PATH*

(Note: Depending on your shell, after changing the *LD_LIBRARY_PATH* environment variable, you might have to execute the command *rehash*.) Afterwards, execute the following steps:

²<http://www.timhuege.de/reas>

³<http://root.cern.ch>

```
cd /home/user/coast-v3r2
./configure
make install
```

This finishes the installation of COAST.

3.3 CORSIKA

With COAST being set up (including the setup of all environment variables), you can start compiling CORSIKA. To start, unzip and untar the CORSIKA files into a new directory, e.g., */home/user/corsika6710*. Then follow these steps

```
cd /home/user/corsika6710
./corsika-install
```

Please specify the options you want to use. Many options are compatible with COAST. The SLANT option, however, is not yet officially supported and the CURVED option does not yet work at all with COAST. (For further information on the individual options, please refer to the CORSIKA manual.)

To activate the COAST interface, in the screen where you can select the main CORSIKA options, please activate option

```
q - ROOT particle tRACKing option (Radio ...)
```

If everything is set up correctly, you should get the following message:

```
COAST package seems to be installed on your system.
Do you want to use this external COAST for the rootrack
option ? (y(default)/n)
```

Please answer with "y" and continue the compilation process as usual. If you get an error message instead of the above-cited message, you have probably not set up the environment variables correctly.

After finishing the compilation of CORSIKA, a CORSIKA binary will be created in the subdirectory *run*, ready for you to use. The usage of CORSIKA stays exactly the same as in the standard case. Please refer to the CORSIKA user's manual for further information. When you run a CORSIKA simulation with COAST enabled, there will be a message in the CORSIKA log stating so and at the end of the CORSIKA run, CORSIKA will create an additional file *DATxxxxx.y.hist.root* for each shower. This is the histogram file, which usually has a size somewhere between 15 and 30 Megabytes. The contents of these histogram files can be reviewed using some (unsupported) helper programs included in the COAST package called *histprofile* and *rootplot*.

3.4 REAS

Download the REAS source code from the project website, unzip and untar it into a directory of your choice, e.g., */home/user/reas259*. Change into that directory:

```
cd /home/user/reas259
```

Make sure ROOT is installed correctly. As a check, please test if the command

```
root-config --includir
```

outputs the correct path to the ROOT include files, which should be the same as you get with

```
echo $ROOTSYS/include
```

Next, run configure and make:

```
./configure CXXFLAGS=-O3  
make
```

(The optional argument to configure ensures that you get optimized code.) After compilation, the binaries for REAS and REASPlot can be found in the subdirectories REAS and REASPlot, respectively. Before running REAS or REASPlot, please again make sure that

- $\$ROOTSYS/lib$ is included in your $LD_LIBRARY_PATH$

3.5 Summary of environment variables

Setting up the environment variables correctly is the key to getting a working installation. To summarize, for a root installation in $/usr/local/root$ and the above example paths, in a *bash* shell, the environment variables should be set up like this:

```
export ROOTSYS=/usr/local/root  
export COAST_DIR=/home/user/coast-v3r2/install  
export COAST_USER_LIB=/home/user/coast-v3r2/THRadio  
export LD_LIBRARY_PATH=$ROOTSYS/lib:$COAST_DIR/lib:$LD_LIBRARY_PATH  
export PATH=$ROOTSYS/bin:$PATH
```

4 Running simulations

In this section, we provide a short introduction on how to set up a full chain of CORSIKA and REAS simulations.

4.1 Setup of CORSIKA simulations

The first step in the chain is to run an adequate CORSIKA simulation. In principle, this is no different from setting up any standard CORSIKA simulation. There are, however, some parameters of the CORSIKA steering card file that need special attention.

4.1.1 Singular values

Some of the parameters can be specified as a range in CORSIKA. If simulations are to be used with REAS, these, however, have to be set to specific values (and not a range). This applies to the keywords *ERANGE*, *THETAP* and *PHIP*. In practice, for these keywords you have to specify the same value twice, e.g., *THETAP 30.0 30.0* for an air shower with 30 degrees zenith angle.

4.1.2 Energy cutoffs

The recommended energy cutoff for electrons/positrons is 401 keV. (This is motivated by technical and physics aspects and will ensure correct results with good performance.) A corresponding entry would be *ECUTS 3.000E-01 3.000E-01 4.010E-04 4.010E-04*.

4.1.3 Thinning

Thinning can be used as usual. As a conservative guideline, 10^{-6} thinning with weight limitation [7] produces very high-quality air showers for use with REAS.

4.1.4 Longitudinal evolution

REAS imports the longitudinal evolution of an air shower from the *.long* file produced by CORSIKA. You therefore have to switch on its generation. An adequate sampling step is 5 g cm^{-2} . The corresponding entry would be *LONGIT 5. T T*.

4.1.5 Observation level

You need to specify an observation level, which sets the height (in cm asl) to which CORSIKA traces the air shower. Naturally, this level should be at least as low as the observation level for which you later want to calculate the radio emission with REAS. In fact, for inclined air showers, you should set the observation level in CORSIKA lower than the observation level of your radio antennas, maybe even to negative values. (Otherwise, there is a region "above the shower" that has not been traced with CORSIKA but will be relevant for radio emission.) This is related to the REAS keyword *ShowerEvolutionClippingDistance*.

4.1.6 Output directory

If you use the keyword *DIRECT* to write output to a directory other than the one where the CORSIKA binary resides, please make sure the directory path ends with a */*.

4.1.7 Multiple showers

You can run multiple showers in one run. Later on, for REAS, you have to pick a specific shower to simulate.

4.2 Running CORSIKA simulations

All parameters should be put in a steering card file. In this case, we call it *RUNxxxxx.inp*. Run the CORSIKA binary as usual with input from this file. You should divert the on-screen output to a file for archiving purposes, e.g., *RUNxxxxx.log*. If you receive error messages, please check again that all environment variables are set up correctly. Also, please make sure you read the CORSIKA user's manual for CORSIKA-related aspects.

Once the simulation has finished, you will have the following files, all needed for the later REAS simulation:

- *RUNxxxxxx.inp*: the CORSIKA steering card file
- *DATxxxxxx.long*: a file with all longitudinal evolution profiles, one per run
- *DATxxxxxx.y.hist*: a number of histogram files, one per shower in the run

The *DATxxxxxx.y.hist* files should usually have a size between 15 and 30 MB. If they are much smaller, this might point to a problem.

4.3 REAS configuration files

Once you have produced the necessary CORSIKA output, you can setup and run the corresponding REAS simulations. To set up a REAS simulation, you will need two configuration files. As an example, let us call our simulation *event* and our observer *all*. The configuration files would then be:

- *event.reas*: a file setting up the REAS simulation parameters
- *east.grnd* or *all.list*: a file setting up the active observer (i.e., radio antenna) positions; *grnd*-files specify observers on a regular grid, whereas *list*-files specify positions of arbitrarily spaced observers; see below for more details

Inside the *.reas* file, parameters are set in the manner

```
KeyWord=Value ; comment
```

where everything as of the semicolon is treated as a comment. In addition, lines starting with a *#* are treated as pure comments.

4.4 The *.reas* parameter file

When setting up REAS simulations, it is highly recommended that you start with the provided example configuration file (see also section 6) and only change the options that you need to alter. For reference, we still provide a complete list of the REAS keywords and their meaning. The order here is the same as in the parameter file.

4.4.1 ParameterFileVersion

This keyword sets an internal version number for the parameter file structure. It enables REAS to recognise when old parameter files have to be converted into a new format. This value must not be changed manually!

4.4.2 NumParticlesToCalculate

This parameter sets the maximum number of particles to be calculated in the simulation. (This is a quality criterium influencing the statistics of the results, not the normalisation of the pulses.) The simulation may end earlier if the appropriate options are set and the radio pulses converge, but it will not run beyond this number of particles. A conservative value for very high quality simulations is 1e9 particles. Lower values may be adequate and save some computation time. If you want to simulate low energy showers, you may have to

lower this number. Otherwise you will simulate the shower with more particles than there actually are. If this is the case, you will get a warning message that the shower will be overly smoothed

4.4.3 NumSimultaneousParticles

The code will calculate this number of particles “en bloque”. After each block, the intermediate result can be checked for convergence or scaled down to a coarser time resolution (see below). A good value for this parameter is 10000 or 100000 particles.

4.4.4 RandomSeed

This sets the random seed for the simulation. If set to a specific value, repeated runs will produce the same result. If set to -1, a random seed is generated from the current time in seconds. In contrast to air shower simulations, the radio results should not depend significantly on the random seed. (The shower-to-shower fluctuations are already contained in the air shower histograms.)

4.4.5 TrajectoryMonitoringFraction

This specifies the fraction of particles for which the trajectory is saved to the file *event.tracks*. A reasonable value is 0.0001, but depends on *NumParticlesToCalculate*.

4.4.6 AtmosphereModel

Sets the (parametrised) atmosphere model to be used for the simulation. Valid values are currently 0 (US Standard Atmosphere), 10 (Argentina Winter Atmosphere), 20 (European January Atmosphere) and 30 (South Pole March Atmosphere). These atmospheres have been taken from the CORSIKA user’s manual. To produce consistent simulation output, the same atmosphere should be used both in the CORSIKA simulation and the REAS simulation.

4.4.7 ObserverType

This parameter chooses between the setup of an observer configuration with a regular grid (grnd-file) or arbitrary observer positions (list-file).

4.4.8 ObserverHeight

Sets the height in cm asl of the observer grid. Only applicable to *ObserverType=0*, otherwise this value will be calculated from the observer positions in the list-file.

4.4.9 MaximumRadius

Sets up the maximum radius in cm of the regular grid in case of *ObserverType=0*.

4.4.10 NumAzimuthalBins

Sets up the number of azimuthal bins of the regular grid in case of *ObserverType=0*.

4.4.11 NumRadialBins

Sets up the number of radial bins of the regular grid in case of *ObserverType=0*. The distance between radial bins will be $d = \text{MaximumRadius}/\text{NumRadialBins}$. Their positions will be at $0.5d, 1.5d, \dots, \text{NumradialBins}-0.5d$.

4.4.12 CoreShiftToNorth

This parameter sets the air shower core shift in cm to north relative to the coordinates of the observers in the list-file. Only applicable to *ObserverType=1*.

4.4.13 CoreShiftToWest

This parameter sets the air shower core shift in cm to west relative to the coordinates of the observers in the list-file. Only applicable to *ObserverType=1*.

4.4.14 TimeLowerBoundary

Sets a global lower bound for the time window to be calculated. (Only applicable if *AutomaticTimeBoundaries=0* and not recommended, setting of *AutomaticTimeBoundaries* is preferred.) Value is in seconds, where 0 denotes the time when an imaginary leading particle propagating at the speed of light hits the plane given by *ObserverHeight*.

4.4.15 TimeUpperBoundary

Sets a global upper bound for the time window to be calculated. (Only applicable if *AutomaticTimeBoundaries=0* and not recommended, setting of *AutomaticTimeBoundaries* is preferred.) Value is in seconds, where 0 denotes the time when an imaginary leading particle propagating at the speed of light hits the plane given by *ObserverHeight*.

4.4.16 InnerTimeAlgorithm

Sets the algorithm for collecting the time series data. The recommended value is 3 for “ClassicGrid”.

4.4.17 InnerTimeResolution

Sets the sampling resolution in seconds for the calculation of the time series data. If *ReductionToOuterTimeGridToggle=1*, this refers only to the calculation blocks of *NumSimultaneousParticles*, otherwise this is the global resolution.

4.4.18 OuterTimeResolution

Only applicable if *ReductionToOuterTimeGridToggle=1*, which is not recommended. Sets the sampling resolution in seconds for the calculation of the time series data when merging blocks of *NumSimultaneousParticles* to a lower time resolution. Must not be smaller than *InnerTimeResolution*.

4.4.19 ReductionToOuterTimeGridToggle

If set to 1, after each block of *NumSimultaneousParticles*, the intermediate time series data is merged onto a coarser grid with time resolution *OuterTimeResolution*. Not available with *InnerTimeAlgorithm=3* and usually not recommended.

4.4.20 GroundLevelRefractiveIndex

Specifies the refractive index at 0 m asl. Not yet active (value will always be overwritten with unity).

4.4.21 AutomaticTimeBoundaries

If set to 0, the boundaries of the time windows are set globally by *TimeLowerBoundary* and *TimeUpperBoundary*. Otherwise, sets a time window in seconds for the calculation of the time series data that the code will position adequately for each individual observer. This hugely increases the simulation performance. A reasonable value is $4e-07$ in combination with *ResolutionReductionScale=5000*.

4.4.22 ResolutionReductionScale

If set to 0, all observers use the same sampling time resolution. Otherwise, sets a radial distance scale in cm on which the time resolution is repeatedly lowered. Activating this option significantly increases simulation performance, but the output files will not have a common sampling time scale. A recommended value is 5000 in combination with *AutomaticTimeBoundaries=4e-07*.

4.4.23 AutomaticBinInactivationToggle

If set to 1, the code will evaluate the convergence of the radio pulses for the individual observers throughout the simulation and inactivate observers once they have reached a result with good statistics. Highly recommended to increase computation performance. The details are configured with the keywords *RelativePrecisionGoal* and *NumPrecisionComparisons*. If set to 0, the simulation will run up to *NumParticlesToCalculate* for all of the observers.

4.4.24 RelativePrecisionGoal

Sets a fractional precision goal for *AutomaticBinInactivationToggle* against which the results of each observer is checked after each block of *NumSimultaneousParticles*. A very conservative value leading to high-quality data is 0.0005. Increasing this value will lead to better performance and may still provide good results.

4.4.25 NumPrecisionComparisons

Sets the number of consecutive successful checks of the *RelativePrecisionGoal* that are needed before an observer is inactivated in case of *AutomaticBinInactivationToggle=1*. A recommended value is 5.

4.4.26 TrajectoryPointsPerUnitPathDepth

Describes how many sampling points per g cm^{-2} should be calculated. This is a parameter with importance for the stability of the results, and it should not be changed lightly. Too high values will produce huge performance hits, too low values might lead to wrong results. The recommended value for standard applications is 3.

4.4.27 SmartSamplingToggle

If set to 1, sampling points on the particle trajectories will be set in an optimised way. Highly recommended to get good simulation performance. If set to 0, sampling of the trajectories will be on a dense equidistant grid, leading to very poor simulation performance.

4.4.28 ShowerType

Set to 1 for simulation of showers based on CORSIKA-generated histograms (REAS2). If set to 0, parameterised showers (REAS1) can still be calculated; there will however be no further user support for these.

4.4.29 DepthOfShowerMaximum

For *ShowerType=0* or *HistogrammedShowerEvolutionToggle=0*, specifies the slant depth of the shower maximum in g cm^{-2} . In case of *ShowerType=1* with *HistogrammedShowerEvolutionToggle=1*, this value is overwritten with the one taken from the CORSIKA *.inp* file.

4.4.30 ShowerEvolutionClippingDistance

For inclined showers, there exists a region “above” the shower that would be clipped if the shower evolution were only traced up to the point when the shower axis hits the *ObserverHeight*. Setting this parameter to values other than 0 makes sure that up to the specified lateral distance in cm no clipping occurs. This requires that the CORSIKA shower has been calculated with an appropriate *OBSLEV* value below the *ObserverHeight*.

4.4.31 MeanElectronPathDepth

Sets the mean length of electron paths in g cm^{-2} . This is an important parameter influencing the correctness of results. The usual value is 1, which should not be changed without special reasons.

4.4.32 MeanPositronPathDepth

Sets the mean length of positron paths in g cm^{-2} . This is an important parameter influencing the correctness of results. The usual value is 1, which should not be changed without special reasons. Not applicable for *ShowerType=0*.

4.4.33 TrajectoryLengthDistributionToggle

If set to 0, all trajectories for a given particle type have the same length, as specified by *MeanElectronPathDepth* and *MeanPositronPathDepth*. If set to 1, the path lengths are distributed exponentially and the aforementioned parameters denote the mean of the distribution. For REAS2 results, this value should be set to 0.

4.4.34 PathDepthProjectionToggle

If set to 1, path depths of particles with large inclination angles against the shower axis are corrected appropriately. No big impact, value of 1 is recommended.

4.4.35 SymmetricTrajectoriesToggle

Should be set to 0. If set to 1, particle trajectories are set up symmetrically around the point of particle creation, which is only relevant for testing purposes and does not work with all values of *PairWiseCreation*.

4.4.36 MagneticFieldStrength

Sets the strength of the magnetic field in Gauss.

4.4.37 MagneticFieldInclinationAngle

Sets the inclination angle of the magnetic field in degrees. Values larger than 0 are appropriate for the northern hemisphere, values smaller than 0 for the southern hemisphere.

4.4.38 MagneticFieldDeclinationAngle

Sets the angle in degrees east of north that the magnetic field points to.

4.4.39 PrimaryParticleEnergy

For *ShowerType=0*, specify the energy of the primary particle in eV. In case of *ShowerType=1*, this value is overwritten with the one taken from the CORSIKA *.inp* file.

4.4.40 EnergyDistribution

For *ShowerType=0* or *HistogrammedParticleEnergiesToggle=0*, specifies the parameterisation function for the particle energy distribution. Valid values are 0 (monoenergetic $\gamma = 60$), 1 (“old broken power-law”), 2 (monoenergetic $\gamma = 30$), 3 (monoenergetic $\gamma = 300$) and 4 (“new broken power-law”).

4.4.41 LowerGammaCutoff

For *ShowerType=0* or *HistogrammedParticleEnergiesToggle=0*, specifies the lower cutoff of the particle energy distribution.

4.4.42 UpperGammaCutoff

For *ShowerType=0* or *HistogrammedParticleEnergiesToggle=0*, specifies the upper cutoff of the particle energy distribution.

4.4.43 MinimumShowerAge

For *ShowerType=0* or *HistogrammedShowerEvolutionToggle=0*, specifies the shower age at which the shower evolution begins.

4.4.44 MaximumShowerAge

For *ShowerType=0* or *HistogrammedShowerEvolutionToggle=0*, specifies the shower age at which the shower evolution ends.

4.4.45 ShowerZenithAngle

For *ShowerType=0*, specify the zenith angle of the shower in degrees. In case of *ShowerType=1*, this value is overwritten with the one taken from the CORSIKA *.inp* file.

4.4.46 ShowerAzimuthAngle

For *ShowerType=0*, specifies the azimuth angle of the shower in degrees. In case of *ShowerType=1*, this value is overwritten with the one taken from the CORSIKA *.inp* file, unless *UseCorsikaAzimuthToggle=0*. The convention is such that 0 corresponds to a shower propagating to north, 90 corresponds to a shower propagating to west.

4.4.47 CorsikaFilePath

Sets the directory from which the CORSIKA files are read in. (Warning, cannot include any space characters!)

4.4.48 CorsikaParameterFile

For *ShowerType=1*, specify the name of the CORSIKA steering card file, e.g., RUNxxxxxx.inp.

4.4.49 CorsikaSlantOptionToggle

For *ShowerType=1*, this parameter has to be set to 0 if a CORSIKA simulation is used that was run without SLANT option, 1 if a simulation is used that was run with SLANT option.

4.4.50 SelectedCorsikaShower

For *ShowerType=1*, in a CORSIKA run with several showers, this selects which shower is to be simulated with REAS. If the CORSIKA run has only one shower, this must be set to 1.

4.4.51 UseCorsikaAzimuthToggle

For *ShowerType=1* and *UseCorsikaAzimuthToggle=1*, the azimuth angle for the shower to be simulated is imported from CORSIKA. Setting this parameter to 0 allows you to manually override the azimuth angle using the keyword *ShowerAzimuthAngle*.

4.4.52 HistogrammedShowerEvolutionToggle

In case of *ShowerType=1*, a value of 1 signifies that the shower evolution is taken from the CORSIKA histograms, whereas 0 uses a parameterised shower evolution.

4.4.53 HistogrammedLateralDistributionToggle

In case of *ShowerType=1*, a value of 1 signifies that the lateral particle distribution is taken from the CORSIKA histograms, whereas 0 uses a parameterised lateral distribution.

4.4.54 HistogrammedLongitudinalDistributionToggle

In case of *ShowerType=1*, a value of 1 signifies that the particle arrival time distribution is taken from the CORSIKA histograms, whereas 0 uses a parameterised arrival time distribution.

4.4.55 HistogrammedParticleEnergiesToggle

In case of *ShowerType=1*, a value of 1 signifies that the particle energy distribution is taken from the CORSIKA histograms, whereas 0 uses the parameterised energy distribution specified with the keyword *EnergyDistribution*.

4.4.56 HistogrammedPitchAngleDistributionToggle

In case of *ShowerType=1*, a value of 1 signifies that the angular particle momentum distribution is taken from the CORSIKA histograms, whereas 0 uses the parameterised angular particle momentum distribution.

4.4.57 PairWiseCreation

In case of *ShowerType=1*, selects how electrons and positrons are treated during particle creation. 0 means that electrons and positrons are always created in pairs, leading to a charge ratio of 1 between positrons and electrons. 1 means that particles are created in pairs when possible, reproducing the correct charge ratio. 2 means that electrons and positrons are created independently, reproducing the correct charge ratio. (A value of 2 automatically sets *HistogrammedShowerEvolutionToggle=1*.) Recommended value is 2.

4.4.58 ShowerEvolutionShift

In case of *ShowerType=1* and *HistogrammedShowerEvolutionToggle=1*, this parameter can be used to arbitrarily shift the CORSIKA-derived longitudinal evolution profile to earlier (negative values) or later (positive values) stages by the given atmospheric depth in g cm^{-2} .

4.4.59 ElectricFieldToggle

Will be used to switch on electric fields in the atmosphere. Not yet active.

4.4.60 ElectricFieldStrength

Will be used to set the atmospheric electric field strength. Not yet active.

4.4.61 ElectricFieldInclinationAngle

Will be used to set the electric field geometry. Not yet active.

4.4.62 ElectricFieldAzimuthAngle

Will be used to set the electric field geometry. Not yet active.

4.5 The .grnd and .list files

If you have selected *ObserverType=0*, you have to provide a *.grnd* file. Consider a case where we have specified *NumRadialBins=20* and *NumAzimuthalBins=8* in the *.reas* file. A corresponding *.grnd* file could be called *east.grnd* and contain the following entry:

```
AddGroundBinSegment = 0 4 19 7
```

The four numbers denote the numbers of the start radial bin, the start azimuth bin, the end radial bin and the end azimuth bin. In this case, the radial bin range 0 to 19 covers the complete radial range and the azimuthal bin range 4 to 7 covers the eastern half of the observer grid. (Azimuthal bin 0 is always to the north and then counting is counter-clockwise, in the mathematically positive sense.)

The idea to specify the active observers in a separate file facilitates the parallelisation of computations on several CPUs. You could create a complementary file *west.grnd* in which you would only calculate the western half of the observer grid and run this on a second CPU using the same *.reas* file. In addition, one *.grnd* file can consist of several *AddGroundBinSegment* statements if necessary.

In case you have selected *ObserverType=1*, you have to provide a *.list* file instead. This lists the arbitrary observer positions of a number of antennas. The syntax is like this:

```
AntennaPosition = -569900 2697300 143200 pole01
AntennaPosition = -577100 2690000 142800 pole02
AntennaPosition = -567300 2687400 142500 pole03
```

Each line denotes an antenna position. The columns signify the position to north, the position to west and the height asl, all in cm, followed by a unique name for the observer. The coordinate origin is arbitrary, but must be the same as the one used in the *CoreShiftToNorth* and *CoreShiftToWest* statements.

4.6 Running REAS simulations

Once you have the CORSIKA output files (see 4.2) as well as a *.reas* and a *.grnd* or *.list* file prepared, you can start the REAS simulation. In case we have the files *event.reas* and *east.grnd*, the command-line would be:

```
./reas event east
```

The simulation will produce on-screen output, but at the same time write to a log-file called *event_east.log*. Should this log-file already exist, the simulation will stop immediately with an error message. This may seem uncomfortable at first, but greatly facilitates running large sets of simulations without the risk of overwriting previous results.

During the simulation, there will be continuous output as the simulation progresses. Please note that the quoted “ETA” time largely overestimates the total computation time if *AutomaticBinInactivation=1* because it is not known in advance when observer bins will be inactivated.

Should you want to cancel a simulation, there are two possibilities: Hitting CTRL-C will cancel the simulation, but only after the current block of particles has been processed. (This may take some time depending on the simulation parameters.) The result that has been achieved up to that point will be appropriately normalised and saved to disk. If a simulation is already progressed by a huge fraction, canceling it in this way will lead to a result that might still be used, however with lower statistics for the observer bins. If you want to cancel the simulation immediately without saving any result, hit CTRL-\

4.7 REAS output files

During a REAS simulation run, a number of files are written to disk. All data are saved as ASCII-text. The data compress very well if you have to save disk space. Sticking to the example of *event.reas* and *east.grnd*, we get:

- *event.reas*: The input file is written back to disk after the simulation with the values that were actually used in the simulation run. In particular, any values imported from CORSIKA will be written to the corresponding fields in the *.reas* file.
- *event_east.log*: This holds a copy of the on-screen output of the simulation. In particular, you should check it for warning messages. Also, if you specified *RandomSeed=1* but want to know which random seed was actually used, you can look it up in here. As long as this file is present, REAS will deny any attempt to re-run the same simulation.
- *event_efflong*: This file contains the longitudinal evolution profile that REAS actually used (in case of *ShowerType=1* and *HistogrammedLongitudinalEvolutionToggle=1*).

- *event.efflong*: This file contains the longitudinal particle injection profile that REAS actually used (in case of *ShowerType=1* and *Histogrammed-LongitudinalEvolutionToggle=1*).
- *event.east.tracks*: This file contains the defined fraction of particle trajectories. The individual trajectories are separated by blank lines. The columns in the file specify the north, west and height coordinates of each particle as it flies on its trajectory. This can be easily plotted with gnuplot.
- *event.east.bins*: This is a helper file needed for data reduction with REASPlot. It lists the data files in the *event.east/* directory together with their x, y and z coordinates followed by a quantity describing when the bin has been shut down (either by CTRL + C or because the automatic bin inactivation has shut it down). If the quantity reads 1, then the bin has been active up to the end. If it reads 0.5, then the bin has been shut down after half the simulation.
- *event.east/*: This is the directory in which the main results (the time-series data for the individual ground bins) are saved.

The raw time-series data is then contained in one *raw.xxxcm.yyydeg.dat* file per groundbin situated in the *event.east/* directory. The columns denote the absolute time stamp and the north-, west-, and vertical component of the electric field. In case you had used *ObserverType=1* and a *.list* file, the individual files would be called *raw_antenna-id.dat* where the antenna-id was given in the *.list* file entries.

All quantities listed in REAS output files are in cgs units! Thus, once you know what kind of quantity (e.g., field strength, distance, time) is listed, there remain no ambiguities.

The raw time-series data represent pulses calculated for infinite bandwidth. To filter these data with a filter and generate useful plots these raw data have to be reduced further. This is done by the helper application REASPlot.

4.8 Data processing with REASPlot

One of the main purposes of REASPlot is to calculate frequency spectra and apply given filters to the raw, unlimited bandwidth time series data calculated by REAS. At the current time, you have to specify the desired filter at compile-time in the REASPlot source code, in the method *ShowerDataSet::ShowerDataSet()*. In the same method, you can specify one or more frequencies for which you want the spectral field strengths to be written out, e.g. for 10 MHz or 55 MHz.

The second purpose of REASPlot is to combine simulation data from a simulation that has been distributed over several CPUs into one data set and organise data in various organisational forms (e.g., as cuts along azimuth directions, as contour data, ...).

In our example, REASPlot would be used like this to produce processed data in the directory *event.filtered*:

```
reasplot event_filtered event east
```

In fact, if you have simulated the same shower on several CPUs, e.g., with the addition of complementary *northwest.grnd* and *southwest.grnd* files, you can combine these now into a complete data set using

```
reasplot event_filtered event east northwest northeast
```

4.9 REASPlot output files

The *event_filtered* directory will afterwards contain the filtered data for the individual ground bins and several additional derived data files. These are:

- *smooth_*.dat*: The filtered time-series data corresponding to the *raw_*.dat* files. The data format is, as in the raw files: absolute time stamp followed by north-, west- and vertical component of the electric field in cgs units.
- *nuspec_*.dat*: The spectrum associated to the time-series of the corresponding bin. The data are given as frequency followed by north-, west- and vertical component of the spectral component of the electric field, again in cgs units. Conversion to $\mu\text{Volt/m/MHz}$ is done later in the visualisation, e.g., with gnuplot. At the moment, only the absolute values are saved, the phase is not. Also, be careful about the Fourier transform convention used! Within REASPlot, a symmetrical convention with a factor of $1/\sqrt{2\pi}$ in both directions is used. In addition, note that the spectral values listed in these files refer to frequencies ν rather than cycle frequencies ω . If you want to plot values referring to cycle frequency ω rather than frequency ν , you have to take into account an extra factor $\sqrt{1/2\pi}$.
- *maxamp_xxxdeg.dat*: For the direction given by the azimuth angle xxx, the maximum amplitude of the filtered pulses is listed. (This strongly depends on the filter used and thus might not be the best quantity to use!) Please note: The algorithm identifies the total field strength maximum amplitude and then saves the electric field value of the north-, west- and vertical component at the corresponding time. This means that the values denoted are not the maximum amplitudes of the individual north-, west- and vertical components! The data listed in these files can be used to plot the lateral radio dependence in the given directions. The data are listed as radial distance from the shower centre followed by the north-, west- and vertical component field strength at the time-stamp where the maximum amplitude is reached. In an additional column, the corresponding time-stamp is denoted, which can be used to create plots of the electromagnetic front curvature.
- *nuspectral_xxxHz_yyydeg.dat*: Similar to the previous, but the spectral electric field corresponding to the frequency xxx Hz rather than the maximum amplitude is denoted. (The frequencies for which these files are created have to be set at compile-time in REASPlot, see above.) The data format is radial distance from the shower centre followed by the spectral electric field strength in the north-, west- and vertical component. The last column repeats the frequency.
- *maxamp_contour_*.dat*: Contains the same data as in the *maxamp_xxxdeg.dat* files, but rearranged in a way that is useful for creating contour plots with gnuplot. The * denotes whether the north-, west-, vertical component or the total field strength is tabulated. The data format is azimuth angle in radians, then the maximum field strength amplitude value followed by the

radial distance from the shower centre. The last column again lists the timestamp associated to the maximum amplitude value.

- *spectral_contour_xxxHz_*.dat*: Same as the previous but for the spectral electric field strength at xxx Hz. Correspondingly, the data format is the azimuth angle in radians, the spectral electric field strength at xxx Hz, the radial distance from the shower centre and in the last column the frequency.
- *passivation.m*: Contains data on when the individual bins have been deactivated during the calculation (again, 1 means that the bin was calculated for all particles right through to the end). Each line contains the azimuthal bins (from 0 to the last) for a specific radial position. Moving from line to line goes outward from the shower centre.

4.10 Data visualisation with gnuplot

The reduced data files produced by REASPlot can be easily used to visualise the results in a number of forms. Some example gnuplot scripts used to visualise the data are enclosed in the source code package.

5 Conventions

This section gives an overview of the conventions used by the REAS code.

5.1 Coordinates

Within the REAS code, the same coordinate conventions as in CORSIKA are being used. For spatial coordinates, this means that a right-handed coordinate system of x, y and z is used where x denotes the north direction, y denotes the west direction and z denotes the vertical direction. The same coordinate system applied to electric field vectors. For the azimuthal angles, 0 degrees denotes north, 90 degrees denotes west, i.e., counter-clockwise rotation.

Also, as in CORSIKA, an air shower is characterised by the direction into which it propagates — not the direction from which it is coming, as is usually done in experimental data. If an air shower has an azimuth angle of 0 degrees, this means that it is propagating to the north, i.e., coming from the south. If it has an azimuth angle of 90 degrees, this means that it is propagating to the west, i.e., coming from the east.

5.2 Units

Throughout the REAS code, cgs units are used. Exceptions are only made when importing values from other sources, and there should be a comment in the source code in these contexts.

All values being written out in data files by REAS are in cgs units.

6 Example parameter files

As a basis for running simulations with CORSIKA and REAS, we provide a set of example files here. They are also included in the REAS source code package in the subdirectory *examples*.

6.1 CORSIKA

A suitable parameter file *RUN000001.inp* for running a 10^{17} eV air shower looks like this:

```
RUNNR 1
EVTNR 1
SEED 3 0 0
SEED 4 0 0
SEED 5 0 0
PRMPAR 14
ERANGE 1.0E+8 1.0E+8
ESLOPE 0.000E+00
THETAP 30.0 30.0
PHIP 45.0 45.0
ECUTS 3.000E-01 3.000E-01 4.010E-04 4.010E-04
ELMFLG T T
THIN 1.000E-06 1.000E+02 0.000E+00
THINH 1.000E+00 1.000E+02
NSHOW 1
USER huege
HOST ik-linux
DIRECT '/home/huege/corsika/'
OBSLEV -50000.0
ECTMAP 1.000E+05
STEPFC 1.000E+00
MUMULT T
MUADDI T
PAROUT F F
MAXPRT 1
MAGNET 20.4 43.23
LONGI T 5. T T
RADNKG 5.000E+05
DATBAS F
EXIT
```

6.2 REAS

The following is a parameter file *event.reas* to simulate radio emission for this CORSIKA simulation. (Comments have been removed here to increase readability, but are included in the electronic version.)

```
# global parameters:

ParameterFileVersion = 20
```

```

NumParticlesToCalculate = 1000000000
NumSimultaneousParticles = 100000
RandomSeed = -1
TrajectoryMonitoringFraction = 0.0001
AtmosphereModel = 0

# parameters setting up the spatial observer configuration:

ObserverType = 0
ObserverHeight = 0
MaximumRadius = 50000
NumAzimuthalBins = 20
NumRadialBins = 8
CoreShiftToNorth = 0
CoreShiftToWest = 0

# parameters setting up the temporal observer configuration:

TimeLowerBoundary = -1
TimeUpperBoundary = 1
InnerTimeAlgorithm = 3
InnerTimeResolution = 2e-10
OuterTimeResolution = 2e-10
ReductionToOuterTimeGridToggle = 0
GroundLevelRefractiveIndex = 1

# parameters setting the optimisation strategies:

AutomaticTimeBoundaries = 4e-07
ResolutionReductionScale = 5000
AutomaticBinInactivationToggle = 1
RelativePrecisionGoal = 0.0005
NumPrecisionComparisons = 5
TrajectoryPointsPerUnitPathDepth = 3
SmartSamplingToggle = 1

# general air shower related parameters:

ShowerType = 1
DepthOfShowerMaximum = 640.0
ShowerEvolutionClippingDistance = 50000
MeanElectronPathDepth = 1
MeanPositronPathDepth = 1
TrajectoryLengthDistributionToggle = 0
PathDepthProjectionToggle = 1
SymmetricTrajectoriesToggle = 0
MagneticFieldStrength = 0.4780159935
MagneticFieldInclinationAngle = 64.73763227
MagneticFieldDeclinationAngle = 0

```

```

# parameters related to parametrised showers

PrimaryParticleEnergy = 1.0e+17
EnergyDistribution = 4
LowerGammaCutoff = 1.01
UpperGammaCutoff = 1000
MinimumShowerAge = 0.001
MaximumShowerAge = 2.249
ShowerZenithAngle = 30.0
ShowerAzimuthAngle = 45.0

# parameters specific to CORSIKA based showers

CorsikaFilePath = /home/huege/corsika
CorsikaParameterFile = RUN000001.inp
CorsikaSlantOptionToggle = 0
SelectedCorsikaShower = 1
UseCorsikaAzimuthToggle = 1
HistogrammedShowerEvolutionToggle = 1
HistogrammedLateralDistributionToggle = 1
HistogrammedLongitudinalDistributionToggle = 1
HistogrammedParticleEnergiesToggle = 1
HistogrammedPitchAngleDistributionToggle = 1
PairWiseCreation = 2
ShowerEvolutionShift = 0

# parameters related to electric field effects:

ElectricFieldToggle = 0
ElectricFieldStrength = 100
ElectricFieldInclinationAngle = 90
ElectricFieldAzimuthAngle = 0

The corresponding ground file east.grnd would look like this:

AddGroundBinSegment = 0 4 19 7

```

7 License

REAS is available to every scientist free of charge, but may not be used for commercial or military applications. You may not distribute the program or parts of it to other interested persons, but instead are asked to refer them to the official REAS webpage under <http://www.timhuege.de/reas> for information on how to obtain the most recent version of the source code. Only this way, we can keep an overview of who is working with the code, inform about bug fixes and coordinate changes. If you publish results based on REAS simulations, please cite the appropriate references mentioned during program startup. For more detailed copyright information, please read the copyright notice in the source code itself. For further information, please contact: Tim Huege (tim.huege@ik.fzk.de)

References

- [1] H. Falcke and P. W. Gorham. Detecting radio emission from cosmic ray air showers and neutrinos with a digital radio telescope. *Astropart. Physics*, 19:477–494, July 2003.
- [2] D. Heck, J. Knapp, J. N. Capdevielle, G. Schatz, and T. Thouw. CORSIKA: A Monte Carlo Code to Simulate Extensive Air Showers. FZKA Report 6019, Forschungszentrum Karlsruhe, 1998.
- [3] T. Huege and H. Falcke. Radio emission from cosmic ray air showers. Coherent geosynchrotron radiation. *Astronomy & Astrophysics*, 412:19–34, December 2003.
- [4] T. Huege and H. Falcke. Radio emission from cosmic ray air showers. Monte Carlo simulations. *Astronomy & Astrophysics*, 430:779–798, 2005.
- [5] T. Huege and H. Falcke. Radio emission from cosmic ray air showers: Simulation results and parametrization. *Astropart. Phys.*, 24:116, 2005.
- [6] T. Huege, R. Ulrich, and R. Engel. Monte Carlo simulations of geosynchrotron radio emission from CORSIKA-simulated air showers. *Astropart. Physics*, 27:392–405, 2007.
- [7] M. Kobal and Pierre Auger Collaboration. A thinning method using weight limitation for air-shower simulations. *Astroparticle Physics*, 15:259–273, June 2001.