

REAS 3.10 User's Manual

Tim Huege*

December 20, 2011

1 For former REAS3.0 users

If you have used REAS3.0 before, here is an executive summary of the most important changes between REAS3.0 and REAS3.1.

- REAS3.1 takes the varying atmospheric refractive index into account for signal calculation and propagation¹. This can lead to significant changes in the predicted signals, especially at high frequencies. The user specifies the refractive index at sea level via the keyword *GroundLevelRefractiveIndex*. Based on this, the refractive index is then scaled proportionally with atmospheric density according to the chosen atmospheric model. Please note that the default setting is now *GroundLevelRefractiveIndex* = 1.000292. If you want to reproduce REAS3.0 results, set *GroundLevelRefractiveIndex* = 1.0.
- REAS3.1 comes bundled with COAST v4r2 (while REAS3.0 came with COAST v4r1). COAST v4r2 produces histograms of the particle momentum angles which are binned logarithmically with respect to the shower axis. COAST v4r1 did the same with a linear binning. The logarithmic binning provides better resolution for very small angles, and consequently, the signal calculation becomes more realistic at very small lateral distances (say, below 20 m lateral distance for vertical 10^{17} eV showers measured at sea level)². It is strongly recommended to use COAST v4r2 in conjunction with REAS3.1. However, histograms created with COAST v4r1 can still be used with REAS3.1. (Caution: Never use COAST v4r2 histograms with REAS versions before 3.1 - results would be wrong.)
- REAS3.1 has been slightly optimized for execution speed when simulating with many observers.

*email: tim.huege@kit.edu

¹This functionality has been developed in collaboration with Clancy James.

²This change has been proposed by David Seckel, whom I would like to thank for manifold useful discussions.

2 For former REAS2 users

If you have used REAS2 before, here is an executive summary of the most important changes between REAS2 and REAS3.x.

New features:

- REAS3.x incorporates the “endpoint” formalism [6] for calculating the electromagnetic radiation from moving charges. As such it in particular incorporates radio emission associated with the time variation of the number of charged particles during the air shower evolution. For a discussion of this issue, please refer to [4].
- REAS3.x supports the usage of a curved atmosphere for the calculation of air showers inclined by more than $\approx 70^\circ$. Even extreme geometries such as 89° zenith angle can be simulated. Please keep in mind to use the CURVED option in CORSIKA for such geometries.
- REAS3.x has an improved definition of the convergence criteria used to determine when an observer result has converged. New options have been added to configure this behaviour.
- REAS3.x comes bundled with COAST v4, which supports many more options than COAST v3. In particular, the CORSIKA SLANT option should now be used by default.
- REAS3.x returns exit codes to the shell. If the simulation finished normally, an exit code of 0 is returned. In case of failures, an exit code of 10 is returned. This can be used to steer batch files more effectively.

To keep in mind:

- REAS3.x needs histograms created with COAST v4. Histograms created for REAS2 with COAST v3 are not usable.
- REAS3.x requires the user to set a three-dimensional core position, including a vertical height above sea level. This is a significant change to REAS2, where in case of using an antenna *.list* file the core z-component was derived from the mean altitude of the antennas. Failure to set the z-component in a REAS3.x simulation correctly will lead to wrong geometries!
- REAS3.x should be compiled with a modern compiler (gcc 4.x or newer), as performance of binaries compiled with gcc 3.x is much worse.
- REAS3.x has to be set up with very different values for some parameters as compared to REAS2. For example, the total number of simulated particles has to be increased very significantly, while the mean path depth for each particle has to be decreased very significantly. Please make sure you use the new reference parameter set as a basis for setting up your own simulations!

Discontinued features:

- REAS3.x no longer supports calculation of parameterized air showers. Options specific to these have been removed.
- REAS3.x no longer offers the different gridding mechanisms available in REAS2. The grid used in REAS3.x is a classic equidistant grid.
- REAS3.x no longer supports the use of `.grnd` files setting up a regular observer configuration. Everything is specified in `.list` files now.

3 Introduction

REAS is a C++ code for the simulation of *Radio Emission from Air Showers*.

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 [2; 3]. REAS2 made the transition to using a sophisticated air shower model derived on a shower-to-shower basis from CORSIKA [1]. For details about the implementation of REAS2 please consult [5].

REAS3.x uses the same CORSIKA-based air shower model as REAS2, but implements the radio emission calculation based on the “endpoint formalism” [8]. The endpoint formalism provides a universal way of calculating the electromagnetic radiation associated with arbitrarily moving particles and is discussed in more detail in [6]. Due to the universality of the approach, REAS therefore automatically incorporates all of the emission associated with the particle motion in air showers (except for information being lost in the air shower histogramming process). As of REAS 3.1, the effect of the atmospheric refractive index is also taken into account in the simulation, leading to “Cherenkov-like” amplification effects.

The terms and conditions for the usage of REAS are detailed in section 9. 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.

4 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.

4.1 CORSIKA and COAST

To run simulations suitable as input for REAS3.x, you will need two components: CORSIKA (version 6.960 or newer) and COAST (version v4 or newer).³

Please download CORSIKA yourself from the official CORSIKA webpage⁴.

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 4 is available together with REAS3.x from the REAS project webpage⁵. (Please note that the COAST version distributed with CORSIKA itself is not suitable for radio simulations, you need the radio-specific COAST version described here.)

Last but not least, for COAST to run, ROOT⁶ has to be set up correctly. ROOT version 4 or 5 will work fine.

4.2 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.

5 Installation

This section guides you through the installation steps necessary to run simulations with REAS3. 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!

5.1 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*.)

³For REAS3.0, COASTv4r1 must be used. For REAS3.1, the use of COASTv4r2 is strongly recommended, but backward compatibility with COASTv4r1 is still maintained.

⁴<http://www-ik.fzk.de/corsika>

⁵<http://www.timhuege.de/reas>

⁶<http://root.cern.ch>

5.2 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-v4r2`. In this example, set the environment variables

- `COAST_DIR` to the directory `/home/user/coast-v4r2/install`
- `COAST_USER_LIB` to the directory `/home/user/coast-v4r2/THRradio`
- 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:

```
cd /home/user/coast-v4r2
./configure
make install
```

This finishes the installation of COAST.

5.3 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/corsika6960`. Then follow these steps

```
cd /home/user/corsika6960
./coconut
```

Please specify the options you want to use. Many options are compatible with COAST. For further information on the individual options, please refer to the CORSIKA manual.

To activate the COAST THRradio interface, in the screen where you can select the main CORSIKA options, please activate option⁷

`p - ROOT particle TRACKing option`

If everything is set up correctly, you should be able to finish the compilation process as usual. If you get an error message instead, 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`.

⁷The name of this option has changed several times in different CORSIKA versions, but it has always been option "p". When CORSIKA shows which options have been activated, `ROOTTRACK` should be listed.

5.4 REAS

Download the REAS source code from the project website, unzip and untar it into a directory of your choice, e.g., `/home/user/reas30`. Change into that directory:

```
cd /home/user/reas30
```

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

```
root-config --incdir
```

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 make⁸:

```
./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`

Important notice: REAS3.x greatly profits from the optimization available in modern compilers. Tests have shown that REAS3.x binaries produced with gcc 4.x run twice as fast as binaries produced with gcc 3.x. Use of a modern compiler is therefore strongly recommended.

5.5 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-v4r2/install  
export COAST_USER_LIB=/home/user/coast-v4r2/THRradio  
export LD_LIBRARY_PATH=$ROOTSYS/lib:$COAST_DIR/lib:$LD_LIBRARY_PATH  
export PATH=$ROOTSYS/bin:$PATH
```

⁸Some Linux distributions (e.g., Debian) have install packages for ROOT that do not make use of the ROOTSYS environment variable. When you simply run `./configure` and `make` in such an environment, you will get error messages relating to missing files. To circumvent this problem, you have to specify the paths to the ROOT files explicitly when calling the `./configure` script, like this: `./configure CPPFLAGS=-I/usr/include/root/ LDFLAGS=-L/usr/lib/root CXXFLAGS=-O3`

6 Running simulations

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

6.1 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.

6.1.1 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.

6.1.2 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*.

6.1.3 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.

6.1.4 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 *LONGI T 5. T T*.

6.1.5 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*.

6.1.6 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 */*.

6.1.7 Multiple showers

You can run multiple showers in one run. Later on, for REAS, you have to pick a specific shower to simulate. In general, however, running one shower per CORSIKA run is preferred.

6.2 Running CORSIKA simulations

All parameters should be put in a steering card file. In this case, we call it *RUNxxxxxx.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., *RUNxxxxxx.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.

6.3 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
- *all.list*: a file setting up the active observer (i.e., radio antenna) positions, 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.

6.4 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 8) 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.

6.4.1 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!

6.4.2 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 $1e10$ particles. (This is a factor of 10 higher than what was recommended for REAS2!) 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

6.4.3 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 in REAS3.x is $1e6$ particles.

6.4.4 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.)

6.4.5 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.

6.4.6 CurvedGeometry

If switched to 1, REAS internally uses a curved atmosphere. This allows precise simulation of inclined air showers and is recommended for zenith angles above $\approx 70^\circ$. Naturally, you should switch on the CURVED option in CORSIKA as well.

6.4.7 EarthRadius

In case of *CurvedGeometry* set to 1, this specifies the radius of the Earth. The default value is the one used in CORSIKA.

6.4.8 CoreCoordinateNorth

This parameter sets the north coordinate of the air shower core position.

6.4.9 CoreCoordinateWest

This parameter sets the west coordinate of the air shower core position.

6.4.10 CoreCoordinateVertical

This parameter sets the vertical coordinate of the air shower core position. It is important to set this value consistently with the antenna coordinates provided in the *.list* file. (If *.list* files were used in REAS2, this coordinate was automatically assumed to be the mean antenna height above sea level. This is no longer done in REAS3!)

6.4.11 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 specified shower core.

6.4.12 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 specified shower core.

6.4.13 TimeResolution

Sets the sampling resolution in seconds for the calculation of the time series data.

6.4.14 GroundLevelRefractiveIndex

Specifies the refractive index at 0 m asl. As of REAS3.1, the default value is set to 1.000292. If you want to switch off refractive index effects, set this to 1.0.

6.4.15 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*.

6.4.16 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*. The scale set in this parameter also governs the feature activated with *AutomaticLambdaEnlargementToggle*.

6.4.17 AutomaticLambdaEnlargementToggle

If set to 0, all particle trajectories will span the path depth configured by the keyword *MeanPathDepth*. If set to 1, the path depth of the individual particle trajectories is varied dynamically as a function of the observer lateral distance from the core. Observers further away from the core will then use larger path depths than observers close to the core. This greatly improves the speed and numerical stability of the simulation and is therefore strongly recommended. The scale on which this dynamical scaling is applied is the one specified in *ResolutionReductionScale*. For further details, please see also *MeanPathDepth*.

6.4.18 AutomaticBinInactivationToggle

If set to 0, the simulation will run up to *NumParticlesToCalculate* for all of the observers. 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. This is highly recommended to increase computation performance. Please note that the algorithm judging the result quality has been significantly improved with respect to REAS2. In particular, the comparison is now done in the frequency domain, and the frequency range taken into account is defined by the user. If the signals for all frequency bins in the specified range change less than a user-specified fraction between *NumPrecisionComparisons* consecutive blocks of *NumSimultaneousParticles* or if the biggest change of any frequency bin is smaller than a certain fraction of the largest signal in all of the frequency bins (in other words, a given dynamic range has been reached), the result will be considered final. The keywords controlling this behaviour are *ComparisonLowestFrequency*, *ComparisonHighestFrequency*, *RequiredRelativePrecisionGoal*, *SufficientDynamicRange* and *NumPrecisionComparisons*. During the simulation, characters will be printed out on the console (but not the log file) after each block of particles. Each character corresponds to one active observer. If an “X” is printed out, then the change between two blocks was more than a factor of 10 larger than the required

precision goal. Digits of “1-9” denote that the change between two blocks was a factor of 1-9 larger than the required precision goal. A digit of “0” means that the precision goal has been met. If this happens *NumPrecisionComparison* times in a row, the result is considered final.

6.4.19 ComparisonLowestFrequency

This specifies the lowest frequency used in determining whether a simulation has converged or not as described in *AutomaticBinInactivationToggle*.

6.4.20 ComparisonHighestFrequency

This specifies the highest frequency used in determining whether a simulation has converged or not as described in *AutomaticBinInactivationToggle*. If you are not interested in results above a certain frequency, you can optimize calculation speed by setting this frequency as the high frequency limit.

6.4.21 RequiredRelativePrecision

Sets the fractional precision goal described in *AutomaticBinInactivationToggle*. A very conservative value leading to high-quality data is 0.001. Increasing this value will lead to better performance and may still provide good results.

6.4.22 SufficientDynamicRange

Sets the dynamic range requirement described in *AutomaticBinInactivationToggle*. A good value is 0.001.

6.4.23 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.

6.4.24 TrajectoryPointsPerUnitPathDepth

This parameter describes how finely a trajectory will be sampled with points. Any trajectory will always be sampled with at least two points, one startpoint and one endpoint. Usually this is sufficient and it is therefore safe to set this parameter to 0. However, you can sample the trajectory more finely by setting here a value > 0 .

6.4.25 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 core. 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 *CoreCoordinateVertical*.

6.4.26 MeanPathDepth

Sets the length of the individual trajectories, the so-called λ -parameter in g cm^{-2} . This is an important parameter influencing the correctness of results and should not be changed lightly. If *AutomaticLambdaEnlargementToggle* is set to 0, the value specified here is used for all observers. If *AutomaticLambdaEnlargementToggle* is set to 1, the value specified here is used for distances smaller than *ResolutionReductionScale*, and is increased by *MeanPathDepth* for every *ResolutionReductionScale* further lateral distance. It is strongly recommended to use *AutomaticLambdaEnlargementToggle* with a *MeanPathDepth* of 0.05 g cm^{-2} .

6.4.27 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.

6.4.28 PrimaryParticleEnergy

This value denotes the primary particle energy in eV. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.29 PrimaryParticleType

This value denotes the primary particle type in the CORSIKA convention for the keyword PRMPAR. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.30 ShowerZenithAngle

This value denotes the shower zenith angle. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.31 ShowerAzimuthAngle

This value denotes the shower azimuth angle. The convention is such that 0 corresponds to a shower propagating to north, 90 corresponds to a shower propagating to west. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.32 DepthOfShowerMaximum

This value denotes the atmospheric depth of the shower maximum. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.33 MagneticFieldStrength

This value denotes the strength of the magnetic field in Gauss. It is overwritten with the one taken from the CORSIKA *.inp* file.

6.4.34 MagneticFieldInclinationAngle

This value denotes the inclination angle of the magnetic field in degrees. It is overwritten with the one taken from the CORSIKA *.inp* file. Values larger than 0 are appropriate for the northern hemisphere, values smaller than 0 for the southern hemisphere.

6.4.35 CorsikaFilePath

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

6.4.36 CorsikaParameterFile

Specify the name of the CORSIKA steering card file, e.g., RUNxxxxxx.inp.

6.4.37 CorsikaSlantOptionToggle

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.

6.4.38 SelectedCorsikaShower

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.

6.4.39 ShowerEvolutionShift

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} . Not supported in case *CurvedGeometry* is set to 1.

6.4.40 ElectricFieldToggle

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

6.4.41 ElectricFieldStrength

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

6.4.42 ElectricFieldInclinationAngle

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

6.4.43 ElectricFieldAzimuthAngle

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

6.4.44 EventNumber

Keeps track of an event number to be used in the Auger Offline software package.

6.4.45 GPSSecs

Keeps track of a time stamp in seconds to be used in the Auger Offline software package.

6.4.46 GPSNanoSecs

Keeps track of a time stamp in nanoseconds to be used in the Auger Offline software package.

6.4.47 CoreEastingPampaAmarilla

Keeps track of the easting core coordinate to be used in the Auger Offline software package.

6.4.48 CoreNorthingPampaAmarilla

Keeps track of the northing core coordinate to be used in the Auger Offline software package.

6.4.49 CoreVerticalPampaAmarilla

Keeps track of the vertical core coordinate to be used in the Auger Offline software package.

6.4.50 Discontinued Keywords

The following keywords are no longer valid in REAS3. Some have been replaced, others have just become superfluous and have been removed: *TrajectoryMonitoringFraction*, *ObserverType*, *ObserverHeight*, *MaximumRadius*, *Bins*, *NumRadialBins*, *InnerTimeAlgorithm*, *InnerTimeResolution*, *OuterTimeResolution*, *ReductionToOuterTimeGridToggle*, *RelativePrecisionGoal*, *SmartSamplingToggle*, *ShowerType*, *MeanElectronPathDepth*, *MeanPositronPathDepth*, *TrajectoryLengthDistributionToggle*, *SymmetricTrajectoriesToggle*, *MagneticFieldDeclinationAngle*, *EnergyDistribution*, *LowerGammaCutoff*, *UpperGammaCutoff*, *MinimumShowerAge*, *MaximumShowerAge*, *UseCorsikaAzimuthToggle*, *HistogrammedShowerEvolutionToggle*, *HistogrammedLateralDistributionToggle*, *HistogrammedLongitudinalDistributionToggle*, *HistogrammedParticleEnergiesToggle*, *HistogrammedPitchAngleDistributionToggle*, *PairWiseCreation*

6.5 The observer .list file

To configure the observer locations, you have to provide a *.list* file. This lists the arbitrary observer positions of a number of antennas. The syntax is like this:

```

AntennaPosition = 10000      0 140000 pole_100m_0deg
AntennaPosition =      0 -10000 140000 pole_100m_270deg
AntennaPosition = 40000      0 140000 pole_400m_0deg
AntennaPosition =      0 -40000 140000 pole_400m_270deg

```

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 *CoreCoordinateNorth*, *CoreCoordinateWest* and *CoreCoordinateVertical* statements.

It is possible to distribute the observers into multiple files to facilitate the parallelisation of computations on several CPUs. You could for example create two files, one called *west.list* in which you would only specify the western half of your observer positions and one called *east.list* specifying the observer locations in the eastern half. These could then be run on two CPUs in parallel using the same *.reas* file.

6.6 Running REAS simulations

Once you have the CORSIKA output files (see 6.2) 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 *all.list*, the command-line would be:

```
./reas event all
```

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-\..

6.7 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 *all.list*, 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_all.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.
- *event_all.bins*: This is a helper file needed for data reduction with REAS-Plot. It lists the data files in the *event_all/* 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_all/*: 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_antenna-id.dat* file per observer where the antenna-id was given in the *.list* file entries. The columns in the file denote the absolute time stamp and the north-, west-, and vertical component of the electric field.

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.

6.8 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. Currently, 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 all
```

If you have simulated the same shower on several CPUs, e.g., by distributing the individual observers to files called *west.list* and *east.list*, you can combine the two sub-simulations now into a complete data set using

```
reasplot event_filtered event west east
```

6.9 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 $\mu\text{Volt}/\text{m}/\text{MHz}$ 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.
- *maxamp_summary.dat*: This file lists the maximum amplitudes of the filtered pulses for all observers. Its format is identical to the *maxamp_xxxdeg.dat* files: 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 timestamp is denoted, which can be used to create plots of the electromagnetic front curvature.
- *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.

6.10 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.

7 Conventions

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

7.1 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.

7.2 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.

8 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*.

8.1 CORSIKA

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

```
RUNNR 1
EVTNR 1
SEED 1 0 0
SEED 2 0 0
SEED 3 0 0
PRMPAR 14
ERANGE 1.000E+8 1.000E+8
ESLOPE 0.000E+00
THETAP 0. 0.
PHIP 0.000E+00 0.000E+00
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 iklx68
DIRECT './'
OBSLEV 140000.0
ECTMAP 1.000E+05
STEPFC 1.000E+00
MUMULT T
MUADDI T
PAROUT T F
MAXPRT 1
MAGNET 18.37 -13.84
LONGI T 5. T T
RADNKG 5.000E+05
DATBAS F
EXIT
```

8.2 REAS

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

```
# global parameters:

ParameterFileVersion = 22
NumParticlesToCalculate = 10000000000
NumSimultaneousParticles = 1000000
RandomSeed = -1
AtmosphereModel = 0
CurvedGeometry = 0
EarthRadius = 637131500

# parameters setting up the spatial observer configuration:

CoreCoordinateNorth = 0
CoreCoordinateWest = 0
CoreCoordinateVertical = 140000.0

# parameters setting up the temporal observer configuration:

TimeLowerBoundary = -1
TimeUpperBoundary = 1
TimeResolution = 2e-10
GroundLevelRefractiveIndex = 1.000292

# parameters setting the optimisation strategies:

AutomaticTimeBoundaries = 4e-07
ResolutionReductionScale = 5000
AutomaticLambdaEnlargementToggle = 1
AutomaticBinInactivationToggle = 1
ComparisonLowestFrequency = 0
ComparisonHighestFrequency = 100000000
RequiredRelativePrecision = 0.001
SufficientDynamicRange = 0.001
NumPrecisionComparisons = 5
TrajectoryPointsPerUnitPathDepth = 0
ShowerEvolutionClippingDistance = 100000
MeanPathDepth = 0.05
PathDepthProjectionToggle = 1

# parameters read from CORSIKA files
```

```

PrimaryParticleEnergy = 1e+17
PrimaryParticleType = 14
ShowerZenithAngle = 0
ShowerAzimuthAngle = 0
DepthOfShowerMaximum = 688.7855835
MagneticFieldStrength = 0.2300005435
MagneticFieldInclinationAngle = -36.99445349

# parameters specific to CORSIKA based showers

CorsikaFilePath = ./
CorsikaParameterFile = RUN000001.inp
CorsikaSlantOptionToggle = 1
SelectedCorsikaShower = 1
ShowerEvolutionShift = 0

# parameters related to electric field effects:

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

# event information for Offline simulations:

EventNumber = -1
GPSSecs = 0
GPSNanoSecs = 0
CoreEastingPampaAmarilla = 0
CoreNorthingPampaAmarilla = 0
CoreVerticalPampaAmarilla = 0

```

The corresponding ground file *all.list* would look like this:

```

AntennaPosition = 10000      0  140000  pole_100m_0deg
AntennaPosition =      0 -10000  140000  pole_100m_270deg
AntennaPosition = 40000      0  140000  pole_400m_0deg
AntennaPosition =      0 -40000  140000  pole_400m_270deg

```

9 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@kit.edu)

References

- [1] 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.
- [2] T. Huege and H. Falcke. Radio emission from cosmic ray air showers. Monte Carlo simulations. *Astronomy & Astrophysics*, 430:779–798, 2005.
- [3] T. Huege and H. Falcke. Radio emission from cosmic ray air showers: Simulation results and parametrization. *Astropart. Phys.*, 24:116, 2005.
- [4] T. Huege, M. Ludwig, O. Scholten, and K. D. de Vries. The convergence of EAS radio emission models and a detailed comparison of REAS3 and MGMR simulations. *NIM A*, in press, 2010. Proceedings of the 2010 ARENA conference, Nantes, France, arXiv:1009.0346.
- [5] 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.
- [6] C. W. James, H. Falcke, T. Huege, and M. Ludwig. An ‘endpoint’ formulation for the calculation of electromagnetic radiation from charged particle motion. *Phys Rev. E*, 84:056602, 2011. arXiv:1007.4146.
- [7] M. Kobaal and Pierre Auger Collaboration. A thinning method using weight limitation for air-shower simulations. *Astroparticle Physics*, 15:259–273, June 2001.
- [8] M. Ludwig and T. Huege. REAS3: Monte Carlo simulations of radio emission from cosmic ray air showers using an “end-point” formalism. *Astropart. Phys.*, 34:438–446, 2011. doi:10.1016/j.astropartphys.2010.10.012.