GENESIS 1.3

Version 4.5.0

User Manual

Mar-20

Contents

About this Manual

This manual covers the FEL simulation code GENESIS 1.3, Version 4. It explains how to compile the code and gives the description of the namelist elements in the main input deck and the definition of the lattice.

The source code can be found in a git repository

`https://github.com/svenreiche/Genesis-1.3-Version4`

Throughout this document different fonts are used to distinguish text from the plain text of the manual. The italic font and Greek symbols refer to mathematical and physical constants or expression, e.g.

$E = mc^2$

Sample input and output files as well as all names referring directly to GENESIS 1.3 are written in typewriter-style such as `GAMMA0` or `XLAMD`.

Installation

Version 4 of the code Genesis 1.3 is written fully in C++, avoiding the hybrid compilation of the previous versions. The only required libraries are OpenMPI and HDF5 with the support of parallel file input and output. This is typically guaranteed when HDF5 has been compiled with the support of OpenMPI. OpenMPI supports also the profiling and tracing of the code since it includes the library VampirTrace, which comes with the OpenMPI distribution. Note that profiling will not be necessary for the normal user and that it requires a commercial program to analyze the traces sufficiently. If available, Genesis 1.3 supports also the library FFTW to calculate the divergence of the radiation field. This can be controlled with a macro definition in the Makefile. If the macro is not defined, all relevant code, which uses FFTW is omitted.

The Source Code Distribution

Genesis is solely distributed by source text, its free, non-commercial and falls under the public license agreement. The distribution is in a compressed tar archive. To unpack, type at the location, where you want to have the source code installed:

`gunzip` *filename*`.tar.gz`

tar xvf *filename*`.tar`

where *filename* is the root name of the file of the source code package. However, it is recommended to pull the source code directly from the GIT repository. The link is:

`https://github.com/svenreiche/Genesis-1.3-Version4`

There should be several directories installed. The most important are `include` and `src` to hold the include- and source-files, `xgenesis` the postprocessor files for Matlab, `benchmark` some sample files to run with Genesis and `sdds2hdf` some utility programs to convert Elegant distribution in SDDS format into HDF5 files, which can be parsed by Genesis. The directory `pygenesis` holds Python files to support the work with Genesis, similar to `xgenesis` for Matlab.

Compilation

In the root directory there is the file `Makefile` which is the makefile for the compilation on Unix machines. This file can be used to help setting up the installation. In the file there is

the `OBJECTS` list which contains all required source code files. There are located in several directories in the subfolder `src`. To find them the compiler search path has to be extended as with the `VPATH` directive under unix. The compilation instruction should convert each individual source file into objects file and then link them with the HDF5 and OpenMPI libraries. Note that at some systems the required definition of the include and libraries files are wrapped in the compiler executive. E.g. in the given Makefile the compiler `h5c++` supports directly both required libraries. If this does not exist on your system and both libraries (HDF5 and OpenMPI) are not supported by the standard search path, they have to be set explicitly with the `-I`, `-L`, and `-l` directive for the include search path, library search path and the library name, respectively. Genesis 1.3 is supports the library FFTW enabled by setting the directive `-DFFTW`. It is needed if the user wants to calculate the instantaneous divergence of the radiation field during run time.

If the make file is set-up correctly, type `make` under Unix systems to compile, which should produce the executable `gencore` in the local directory. If source files have been changed then the code can be compiled again. Only files, which have been edited will be recompiled and then linked with the unchanged object files. If some include files have been changed it is recommended to recompile from scratch. The command `make clean` will clean up the directory by deleting all object files and the current executable. To install into your local library, which is the `bin` folder in your home directory, just type `make install`. You can edit the makefile to fit your system.

On platforms, which do not support gnu compiler directly, you have to build up your compilation instruction according to the Makefile.

To complete the installation it is recommended to install the scripts in the directory `sdds2hdf5` in a directory (e.g. `/bin`), which is included in your search path. The files will convert Elegant output distribution into HDF5 format. More info can be found later in the manual. Also, the directory `xgenesis` includes some functions, which allows Matlab to parse the output file of Genesis and to plot the results. They are not required for running Genesis itself. To use them add the given folder to your Matlab Path or place the file in a directory, which is already in the search path.

Running Genesis 1.3

Genesis 1.3 is a command line executable and requires a single input argument, which is the filename of the input deck:

`genesis4` [-o *output-rootname*] [-l *lattice-filename*] *input-filename*

The arguments for the lattice filename and the rootname for all output filenames are optional. If defined, they act as the default values in the `setup` namelist and thus can be omitted there. Any further information must be provided within the input deck. Starting Genesis in this way will effectively invoke a parallel job with only one single core. To start the code with just more than one core the launching command `mpirun` needs to be called, including the number of cores. In its simplest call it would be:

`mpirun -np ## genesis4` [-o *output-rootname*] [-l *lattice-filename*] *input-filename*

where `##` is the number of cores to be requested.

Once Genesis started the master core, which is the core with the MPI rank 0, will provide the output. The output can vary for different output decks. Normally it is reported, once the code generates or alters particle distributions or the radiaiton field, tracks the beam through the lattice or when a time window is specified.

The execution produces output files whenever a tracking command is issued or a field or particle dumped is requested. All in common is the root filename which is defined in the input deck. Normal output files have the extension `.out.h5` for the first tracking or `.Run#.out.h5` for the #th run (except for the first, where the addition of 'Run' is omitted). The file format is HDF5. The explicit format is described in a later chapter of this manual. Field or particle dumps are using as well the root filename as default but under certain circumstances this behavior can be overruled. For more information see the corresponding namelist elements in the next chapter.

For time-dependent simulations the code requires a large amount of memories to keep the entire radiation field and particle distribution in memory. This is the significant difference to previous versions where the code was able to progress through the bunch sequentially. However this excludes effects such long range fields, exchanging particles between slices, and large delaying chicanes in the older versions. For an estimate of the memory demand the radiation field requires

$$N_b = n_s * n_g^2 * 16 \text{ Bytes},$$

where n_s is the number of slices and n_g number of grid points. For the electron distribution it is

$$N_b = n_s * n_e * 48 \text{ Bytes},$$

with n_e the number of macro particles per slice.

Both numbers should be added up and for safety multiplied by a factor of 2 for some overhead (aligning arrays in memory, temporary working arrays, arrays to buffer the output information etc.). This will be distributed equally over all nodes. Non time-dependent simulation should be small enough to run on any computer.

In time-dependent runs the number of slices are equally distributed over all cores. The number of slices are even increased to guarantee symmetry, extending the time-window of the simulations. More information about that can be found in the next section. Often it is useful to calculate the number of generated slices in advance, in particular if advanced option such as subharmonic conversion are used. In the case of subharmonic conversion the slices per core should be an integer of the subharmonic number. Otherwise artefacts of numerically enhanced bunching causes wrong results.

The Main Input File

Unlike older version, which supported only a single namelist as input, the latest version splits up the namelist and processes them sequentially as they appear in the file. Therefore the order is important, e.g. you cannot define a time window after the electron beam or the field has been already generated. On the other hand the code supports multiple tracking in one run. For that the particle and field distribution is kept in memory and then reused in the next tracking. If a fresh bunch is required the old has to be discarded and then a new one has to be generated. In that sense it is possible to combine steady-state and time-dependent run in one input deck by the sequence: generate field, generate beam, track, define time window, generate beam, generate field, and track.

The namelists themselves always start with the ampersand & and the name of the name list and are terminated with `&end`. Each line consists out of an assignment of a supported variable, e.g `npart = 2048` in the namelist `&setup`. Empty lines and lines, starting with the pound symbol # are ignored. If the same variable occurs more than once the latest occurrence is used for the calculation. If Genesis encounters a variable name, which it does not recognize it stops execution and prints out a list of the supported variables.

The following describes all supported namelist with their variables, including its required format and default value)

setup

The namelist `setup` is a mandatory namelist and should be the first in the input deck. It contains the basic parameters to control the simulations. It can only be called once. If the user want to change some parameter the namelist `alter_setup` should be used.

- **rootname** (*string, < empty >*): The basic string, with which all output files will start, unless the output filename is directly overwritten (see `write` namelist)
- **lattice** (*string, < empty >*): The name of the file which contains the undulator lattice description. This can also include some relative paths if the lattice file is not in the same directory as the input file.
- **beamline** (*string, < empty >*): The name of the beamline, which has to be defined within the lattice file. For more information on the lattice file, see the next chapter.
- **gamma0** (*double, 11350.3*): The reference energy in unites of the electron rest mass. This is the reference energy which is used in the code at various place, mostly in the

calculation of the matching condition, the reference focusing strength of quadrupoles and undulator as well as the default value if an electron distribution is generated.

- **lambda0** (*double, 1e-10*): The reference wavelength in meter, which is used as the reference wavelength in steady-state simulation or for defining the sample distance in time-dependent runs. It also acts as the default value when field distributions are generated.

- **delz** (*double, 0.015*): Preferred integration stepsize in meter. Note that this is not a strict value because Genesis tries to optimized the stepsize according to the elements it can resolve. E.g. if an undulator is 1.99 m long but the preferred stepsize is 2 cm than it uses a stepsize which is the closest to preserve the number of integration step. In this case the preferred stepsize gives 99.5 steps which is than rounded to 100 and thus resulting in an actual stepsize of 1.99 cm. Note that outside of the undulator, which are free drifts for the radiation field, Genesis progresses the electron beam and radiation field in larger steps, namely one step per resolved element (drift, quadrupole, phase shifter).

- **seed** (*int, 123456789*): Seed to initialize the random number generator, which is used for shot noise calculation and undulator lattice errors, though it is recommended that the random number generator seed is redefined explicitly for undulator errors in its corresponding namelist.

- **npart** (*int, 8192*): Number of macro particles per slice. Note that the number must be a multiple of the used bins `nbins` otherwise Genesis will exit with an error. If one-for-one simulations are used, this parameter has no meaning.

- **nbins** (*int, 4*): Number of macro particles, which are grouped into beamlets for generating the correct shot noise. For one-for-one simulations this parameter has no meaning

- **one4one** (*bool, false*): Flag to enable or disable simulation to resolve each electron in the simulation. This is mandatory for certain features, such as sorting or slicing of particle distributions. If set to `true` other parameters such as `npart` and `nbins` are obsolete and do not need to be defined. It is recommended to estimate the number of electrons, which are generated in the simulations, because this can easily required memory beyond what is available on the computer.

- **shotnoise** (*bool, true*): Flag to enable the calculation of shotnoise per each slice during generation of the electron distribution. It is recommended to set the value to `false` for steady-state or scan simulations.

- **beam_global_stat** (*bool, false*): Flag to enable extra output of beam parameters of the entire bunch, such as energy, energy spread etc. The data are placed in the HDF5 group "Global" within the group "Beam" of the output file

- **field_global_stat** (*bool, false*): Flag for the field output, similar to `beam_global_stat`.

- **exclude_spatial_output** (*bool, false*): Flag to suppress the datasets in the output file for the x- and y-position and size (both Beam and Field) and px- and py-position (Beam only). This might be useful to reduce the file size of the output file, if these datasets are not needed for the post-processing

- **exclude_fft_output** (*bool, false*): Flag to suppress the datasets in the output file for the field divergence and pointing. Since it also disable the FFT calculation of the 2D wavefronts it speeds up the execution time slightly. If the code has been compiled without the support of the FFTW library this parametr has no effect.

- **exclude_intensity_output** (*bool, false*): Flag to suppress the datasets for the near and farfield intensity and phase for the radiaiton field. If excluded the output file size becomes smaller but no post-processing calculation of the spectra is possible.

- **exclude_energy_output** (*bool, false*): Flag to suppress the datasets in the output file for the mean energy and energy spread of the electron beam.

- **exclude_aux_output** (*bool, false*): Flag to suppress the auxiliary datasets in the output file. In the moment it is the long-range longitudinal electric field as seen by the electrons.

- **exclude_aux_output** (*bool, true*): Flag to reduce the size of the current dataset for the electron beam. Under most circumstances the current profile is constant and only the initial current profile is written out. However, simulation with `one4one` set to true and sorting events the current profile might change. Example are ESASE/HGHG schemes. By setting the flag to false the current profile is written out at each output step similar to radiation power and bunching profile

alter_setup

A namelist to change some parameters within the simulation, which have been defined alread by the `setup` namelist. The change values are stored in the `setup` module so that for another invocation of `alter_setup` some defaults values are use which have been defined in the preceding call of `alter_setup`

- **rootname** (*string, < taken from SETUP >*): The basic string, with which all output files will start, unless the output filename is directly overwritten (see `write` namelist)

- **beamline** (*string, < empty >*): The name of the beamline, which has to be defined within the lattice file. This way another beamline can be selected in the case the simulation has multiple stages

- **delz** (*double, < taken from SETUP >*): Preferred integration stepsize in meter. Note that this is not a strict value because Genesis tries to optimized the stepsize according to the elements it can resolve. E.g. if an undulator is 1.99 m long but the preferred

stepsize is 2 cm than it uses a stepsize which is the closes to preserve the number of integration step. In this case the preferred stepsize gives 99.5 steps which is than rounded to 100 and thus resulting in an actual stepsize of 1.99 cm. Note that outside of the undulator Genesis, which are free drifts for the radiation field, it progress the electron beam and radiation field in larger steps, namely one step per resolved element (drift, quadrupole, phase shifter).

- **harmonic** (*int, 1*): If the value is not 1 than a harmonic conversion is done. This has several consequences. The reference wavelength in `setup` is divided by the harmonic number, the sample rate in `time` is multiplied by the harmonic number, the ponderomotive phases of all macro particles are scaled with the harmonic number, all radiation fields, which are not identical to the harmonic numbers are deleted, while an existing harmonic field is changed to be at the fundamental wavelength

- **subharmonic** (*int, 1*): If the value is not 1 than a down conversion is done. It is similar to the action of `harmonics` but in the opposite directions. For the radiation field all field definitions are deleted except for the fundamental, which is converted to a harmonic. In this case the fundamental field needs to be defined before another tracking is called.

- **resample** (*bool, false*): If this is set to true and only if one-for-one simulations are used the harmonic and subharmonic conversion can re-sample to the new wavelength. In the case of up-conversion the slices are split and the total number of slices increases. Same with the radiation field. An previously existing harmonic field, which is now becoming the fundamental, is interpolated between the existing sample points (*still needs to be implemented*). If a new field is generated it has automatically the new number of slices. If also prevents that the sample rate is changed by remaining unchanged.

lattice

This namelist is used to change the raw lattice from the lattice file, such as generating errors in the position of the elements. The namelist can be defined several times to add more than one error source to the lattice.

- **zmatch** (*double, 0)* If the position within the undulator in meter is non-zero than Genesis tries to calculate the matched optics function for a periodic solution. In the case that it cannot find a solution than it will report it. Found solution will also be the default values for a succeeding beam generation, so that no explicit optical functions need to be defined any longer. If the lattice is highly non-periodic it is recommended to find the matching condition with an external program such as MAdX.

- **element** (*string, < empty >*): Name of the element type, which will be changed, e.g. `Undulator` if undulator modules are altered. Only the first 4 letters need to be defined. If there is no match, e.g. due to a type, nothing will be changed. It acts rather as a

filter than a mandatory element. Elements of the type MARKER are not supported to be changed.

- **field** (*string, < empty >)*: attribute name for a given element. The names are the same as in the definition of the lattice file. The field acts as a filter again. With non-matching events nothing will be changed.
- **value** (*doubl, 0, or sequence label*): The new value. If a reference to a sequence is used, values can be different depending on how many elements are changed. For a double the value would be the same for all elements affected.
- **instance** (*integer, 0*): The instances of affected elements. If a positive value is given, than only that element is change, where its occurence matches the number. E.g. for a value of 3 only the third element is selected. For a value of 0 all elements are changed. The ability to change more thanone but less than all is currently not supported.
- **add** (*bool, true*): If the value is true the changes are added to the existing value. For a value of false the old values are overwritten.

time

This namelist defines the time window/range for simulation with more than just one slice. For reference the complementary axis to the undulator axis, which is normally the position in the time frame, is expressed in a position s. Normally everything is aligned to the origin $s = 0$, in particular when external distributions are imported. Note that for parallel execution the number of slices per core should be the same for an efficient writing of the output files. Therefore Genesis extends the time-window to symmetrize the number of slices per core by extending it towards larger values of s. As an example with `XLAMDS=1e-6` and a length `SLEN=20e-6` a call of Genesis with 24 cores would generate a time-window of 24 microns because each core would have one slice, while 15 cores would expand it to 30 microns with 2 slices per core each.

this module defines also scans in either field or beam parameters if the corresponding flag is set. Technically it generates the beam and field as for time-dependence but disables slippage during simulations. That way the radiation field is kept in the same slice, acting as steady-state simulations.

- **s0** (*double, 0*): Starting point of the time-window in meters.
- **slen** (*double, 0*): Length of the time window in meters. Note that for parallel jobs this might be adjusted towards larger values.
- **sample** (*int, 1)*): Sample rate in units of the reference wavelength from the `setup` namelist, so that the number of slices is given by `SLEN / LAMBDA0/SAMPLE` after `SLEN` has been adjusted to fit the cluster size.

- **time** (*bool, true*): Flag to indicate time-dependent run. Note that time-dependent simulations are enabled already by using this namelist. This flag has the functionality to differentiate between time-dependent run and scans, which disable the slippage in the tracking. To restrict the simulation to steady-state the `time` namelist has to be omitted from the input deck.

profiles

Profiles are defining a dependence on the position in the time frame, which then can be used to defined the behavior of various parameters such as slice emittance, energy spread etc. All profiles are in common that they have to have a label with which they are referred to in other namelists. To indicate the reference to a profile and thus allows to distinguish Genesis between normal numbers the label name must have the additional character '@' in front. E.g. if the name of the label is `energy` then in the beam name list it is referred to by `gamma = @energy`.

profile_const

- **label** (`string, < empty >`): Name of the profile, which is used to refer to it in later calls of namelists
- **c0** (`double, 0`): constant value to be used.

profile_gauss

- **label** (`string, < empty >`): Name of the profile, which is used to refer to it in later calls of namelists
- **c0** (`double, 0`): Maximum function value of the Gaussian distribution
- **s0** (`double, 0`): Center point of the Gaussian distribution
- **sig** (`double, 0`): Standard deviation of the Gaussian distribution

profile_step

- **label** (`string, < empty >`): Name of the profile, which is used to refer to it in later calls of namelists
- **c0** (`double, 0`): Constant term
- **s_start** (`double, 0`): Starting point of the step function
- **s_end** (`double, 0`): Ending point of the step function

profile_polynom

- **label** (`string, < empty >`): Name of the profile, which is used to refer to it in later calls of namelists
- **c0** (`double, 0`): Constant term
- **c1** (`double, 0`): Term proportional to s
- **c2** (`double, 0`): Term proportional to s^2
- **c3** (`double, 0`): Term proportional to s^3
- **c4** (`double, 0`): Term proportional to s^4

profile_file

- **label** (`string, < empty >`): Name of the profile, which is used to refer to it in later calls of namelists
- **xdatal** (`string, < empty >`): Points to a dataset in an HDF5 file to define the s-position for the look-up table. The format is *filename/group1/.../groupn/datasetname*, where the naming of groups is not required if the dataset is at root level of the HDF5 file
- **ydata** (`string, < empty >`): Same as y data but for the function values of the look-up table.
- **isTime** (*bool, false*): If true the s-position is a time variable and therefore multiplied with the speed of light c to get the position in meters.
- **reverse** (*bool, false*): if true the order in the look-up table is reverse. This is sometimes needed because time and spatial coordinates differ sometimes by a minus sign.

beam

This namelist initiates the generation of the particle distribution to be kept in memory. Any time-dependence has to be defined before calling this namelist.

- **gamma** (*double, gamma0 or profile label*): Mean energy in units of the electron rest mass. If default value is given by the reference energy from the `setup` namelist.
- **delgam** (*double, 0 or profile label*): RMS energy spread in units of the electron rest mass.
- **current** (*double, 1000 or profile label*): Current in Amperes.

- **ex** (*double, 0.3e-6 or profile label*): Normalized emittance in x in units of meters
- **ey** (*double, 0.3e-6 or profile label*): Normalized emittance in y in units of meters
- **betax** (*double, 15 or matched value or profile label*): Initial beta-function in x in meters. If the matched command has been invoked before the default values are set to the results.
- **betay** (*double, 15 or matched value or profile label*): Initial beta-function in y in meters. If the matched command has been invoked before the default values are set to the results.
- **alphax** (*double, 0 or matched value or profile label*): Initial alpha-function in x. If the matched command has been invoked before the default values are set to the results.
- **alphay** (*double, 0 or matched value or profile label*): Initial alpha-function in y. If the matched command has been invoked before the default values are set to the results.
- **xcenter** (*double, 0 or profile label*): Initial centroid position in x in meter.
- **ycenter** (*double, 0 or profile label*): Initial centroid position in y in meter.
- **pxcenter** (*double, 0 or profile label*): Initial centroid momentum in x in units of $\gamma\beta_x$.
- **pycenter** (*double, 0 or profile label*): Initial centroid momentum in y in units of $\gamma\beta_y$.
- **xcenter** (*double, 0 or profile label*): Initial centroid position in x in meter.
- **bunch** (*double, 0 or profile label*): Initial bunching value
- **bunchphase** (*double, 0 or profile label*): Initial phase of the bunching
- **emod** (*double, 0 or profile label*): Initial energy modulation in units of the electron rest mass
- **emodphase** (*double, 0 or profile label*): Initial phase of the energy modulation

field

This namelist initiate the generation of the field distribution. It differs in one point from the generation of the beam. It can be called multiple times. If the variable `accumulate` is set to true, it does not delete the previous distribution but adds up the wavefronts. That way higher mode content in either spatial and time direction can be created.

- **lambda** (*double, lambda0*): Central frequency of the radiation mode. The default value is the reference wavelength from the `setup` namelist.
- **power** (*double, 0 or profile label*): Radiation power in Watts

- **phase** (*double, 0 or profile label*): radiation phase in rads. Note that a linear profile results in a shift in the radiation wavelength, which is also the method if for the variable `lambda` a different value than the reference wavelength is used. In case of conflicts the profile for the phase definition has priority.
- **waist_pos** (*double, 0 or profile label*): Position where the focal point is located relative to the undulator entrance. Negative values place it before, resulting in a diverging radiation field.
- **waist_size** (*double, 100e-9 or profile label*): Waist size according to the definition of w_0 according to Siegman's 'Laser' handbook
- **xcenter** (*double, 0*): Center position in x in meter of the Gauss-Hermite mode
- **ycenter** (*double, 0*): Center position in y in meter of the Gauss-Hermite mode
- **xangle** (*double, 0*): Injection angle in x in rad of the Gauss-Hermite mode
- **yangle** (*double, 0*): Injection angle in y in rad of the Gauss-Hermite mode
- **dgrid** (*double, 1e-3 or by existing field*): Grid extension from the center to one edge. The whole grid is twice as large with 0 as the center position
- **ngrid** (*int, 151 or by existing field*): Number of grid points in one dimension. This value should be odd to enforce a grid point directly on axis. Otherwise the convergence in the simulations could be worse.
- **harm** (*int, 1*): Harmonic number of the radiation field with respect to the reference wavelength.
- **nx** (*int, 0*): Mode number in x of the Gauss-Hermite mode
- **ny** (*int, 0*): Mode number in y of the Gauss-Hermite mode
- **accumulate** (*bool, false*): If set the generated field is added to an existing field instead of overwriting it.

importdistribution

This namelist controls the import of an external distribution which are generated from Elegant. This can be directly in SDDS format or converted to HDF5 format, which is done automatically by Genesis if the SDDS file format is indicated. However it is recommended to do the conversion manually before the start of genesis. The distribution has to provide all 6 dimensions while the charge is supplied in this namelist. When imported the longitudinal position is changed so that the last particles is at $s = 0$ micron.

Note that this namelist will be expanded in the future, to enable tilts and match/center to a core part of the beam

- **file** (*string, < empty >*): The file name of the distribution, including possible relative directories.

- **sdds** (*bool, false*): If set to true, the file is converted first to an HDF5 file by the shell script `sdds2hdf5-dist.sh`. A new file is generated with the additional extension `.h5` and then used for the parsing.

- **charge** (*double, 0*): Total charge of the distribution to calculate the current and individual charge per macro particle.

- **slicewidth** (*double, 0.01*): the fraction in length of the distribution which is used for reconstruction. E.g if the length is 10 micron and slic ewidth 0.02 then the reconstruction at the position $s = 4$ micron is using the particle in the distribution, which are located in the slice from 3.9 microns to 4.1 microns.

- **output** (*bool, false*). If set to true an output file is generated with the extension `'.ana.h5'` which contains the slice parameters of the particle distribution after the import and conversion has finished. Note that this option is obsolete because the same output is also included in the main output. Therefore this feature will be disabled soon.

- **center** (*bool, false*): If set to true the particle distribution is recentered in transverse position, momenta and energy.

- **gamma0** (*double, gamma0 from setup*): If centering is enabled, new center in energy in units of electron rest mass.

- **x0** (*double, 0*): If centering is enabled, new center in x in meter.

- **y0** (*double, 0*): If centering is enabled, new center in y in meter.

- **px0** (*double, 0*): If centering is enabled, new mean momentum in x in $\gamma\beta_x$.

- **py0** (*double, 0*): If centering is enabled, new mean momentum in y in $\gamma\beta_y$.

- **match** (*bool, false*): If set to true the particle distribution is matched to new optical function values.

- **betax** (*double, 15 or matched value*): If matching is enabled, new beta function in x in meter.

- **betay** (*double, 15 or matched value*): If matching is enabled, new beta function in y in meter.

- **alphax** (*double, 0 or matched value*): If matching is enabled, new alpha function in x.

- **alphay** (*double, 0 or matched value*): If matching is enabled, new alpha function in y.

importbeam

The modules controls the import of a Genesis 1.3 particle file to replace the internal generation of the particle distribution (note that the module `beam` should not be called). The routine defines also the parameter for a time-dependent run if the `time` namelist hasn't been defined yet.

- **file** (*string, < empty >*): File name of a hdf5 complient datafile to contain the slice-wise particle distribution. It has to follow the internal Genesis 1.3 syntax.
- **time** (*bool, true*): If the time window hasn't be defined it allows to run Genesis with the imported distribution in scan mode, when set to `false`. This would disable all slippage and long-range collective effects in the simulation

importfield

The modules controls the import of a Genesis 1.3 field file to replace the internal generation of the field distribution (note that the module `field` should only be called afterwards with the `accumulate` option enabled). The routine defines also the parameter for a time-dependent run if the `time` namelist hasn't been defined yet.

- **file** (*string, < empty >*): File name of a hdf5 complient datafile to contain the slice-wise particle distribution. It has to follow the internal Genesis 1.3 syntax.
- **harmonic** (*int, 1*) defines the harmonic for the given Genesis run.
- **time** (*bool, true*): If the time window hasn't be defined it allows to run Genesis with the imported distribution in scan mode, when set to `false`. This would disable all slippage and long-range collective effects in the simulation

efield

This namelist controls the short range space charge on a length scale of the resonant wavelength or shorter. The calculation is done on a radial azimuthal grid, centered to the centroid position of the electron slice

- **rmax** (*double, 0*): Scaling factor to define the grid size, which is given by the product of `rmax` and the maximum offset of the macro particles from its centroid
- **nz** (*int, 0*): Number of longitudinal Fourier component of the space charge field. Note that this should be not in conflict with the beamlet size.
- **nphi** (*int, 0*): Number of azimuthal modes in the calculation of the space charge field.
- **ngrid** (*int, 100*): Number of grid points of the radial grid.

sponrad

This enables the effect of spontaneous radiation outside of the frequency band of the FEL simulation.

- **seed** (*int, 1234*): Seed for random number generator to model the quantum fluctuation of hard photons.
- **doLoss** (*bool, false*): If set to true, electrons will loose energy due to the emission of spontaneous radiation within the undulator
- **doSpread** (*bool, false*): If set to true, the energy spread will increase due to the fluctuation in the emission of hard photons of the spontaneous radiation.

wake

Genesis supports the calculation of three types of wakefields by specifying the typical input parameters (e.g. gap length for the geometric wakefield). It first solves the single particle wake and then convolutes with the current distribution. Therefore it follows the change in the wakepotential if a chirped beams undergoes a compression in a chicane. In addition an external loss factor can be supplied, which can also refer to a profile. In this case it is treated as the full wake and subtracted from the particle energy directly.

- **loss** (*double, 0 or profile label*): Loss in eV/m. This is a global loss function (in particular if a profile is defined). It s function values $V(s)$ remains unchanged even if the current profile changes
- **radius** (*double, 2.5e-3*): Radius of the aperture if it is a round chanber or half the distance in the case of two parallel plates.
- **roundpipe** (*bool, true*): Flag to indicate the shape of the transverse cross-section of the aperture. If set to true a round aperture is assumed, otherwise the model has two parallel plates.
- **conductivity** (*double, 0*): Conductivity of the vacuum material for the resistive wall wakefield function
- **relaxation** (*double, 0*): Relaxation distance (aka the mean free path of the electron in the vacuum material) for the resistive wall wakefields
- **material** (*string, < empty >*): String literal to define conductivity and relaxation distance for either copper or aluminum by using the two character label 'CU' or 'AL' repectively. This overwrites also any explicit definition of the conductivity and relaxation value.

- **gap** (*double, 0*): Length in mm of a longitudinal gap in the aperture, exciting geometric wakes.
- **lgap** (*double, 1.0*): Effective length over which a single gap is applied. E.g. if there is a periodicity of 4.5 m at which there is always the same gap in the aperture for the geometrice wakes, then this value should be put to 4.5 m.
- **hrough** (*double, 0*): Amplitude in meters of a sinusoidal corrugation, modelling the effect of surface roughness wakes.
- **lrough** (*double, 1)*: period lengthin meters of the sinusoidal corrugation of the surface roughness model.
- **transient** (*bool, false*): If set to true, Genesis includes the catch-up length of the origin of the wakefield to the particle effects. E.g. particles do not see immediatly the wake from those closer ahead of them than those further away. The catch-up distance is the distance in the undulator added to the starting position `ztrans`. If set to false the steady-state model is used, effectively setting `ztrans` to infinity. Enabling transient calculation will update the wakefield at each integration step, which can slow down the calculations.
- **ztrans** (*double, 0*): Reference location of the first source of the wake fields. A positive value means that the condition for wakes (e.g. a small aperture in the vacuum chamber) has already started and there has been already some length to establish the wakes. For a value of zero the surce is right at the undulator start, while a negative value prevents any wake, till the interation position has passed that point.

sort

An empty namelist with no variables. It initiates the sorting and redistribution of particles only if one-for-one simulation is enabled. Note that harmonic conversion will automatically invokes sorting and therefore does not need to be called explicitly.

write

With this name list the field or particle distributions are dumped.

- **field** (*string, < empty >*): if a filename is defined, Genesis writes out the field distribution of all harmonics. The harmonics are indicated by the suffic `'.h#.'` where # is the harmonic number. The filename gets the extension `.fld.h5'` automatically
- **beam** (*string, < empty >*): if a filename is defined, Genesis writes out the particle distribution. The filename gets the extension `.par.h5` automatically

track

This namelist initiate the actually tracking through the undulator and then writing out the results.Normally all parameter should be defined before or defined in the lattice but the namelist allows some 'last minute' change of the behavior of the code

- **zstop** (*double, 1e9*): If `zstop` is shorter than the lattice length the tracking stops at the specified position
- **output_step** (*int, 1*): Defines the number of integration steps before the particle and field distribution is analyzed for output.
- **field_dump_step** (*int, 0*): Defines the number of integration steps before a field dump is written. Be careful because for time-dependent simulation it can generate many large output files.
- **beam_dump_step** (*int, 0*): Defines the number of integration steps before a particle dump is written. Be careful because for time-dependent simulation it can generate many large output files.
- **sort_step** (*int,0*): Defines the number of steps of integration before the particle distribution is sorted. Works only for one-4-one simulations.

The Lattice Files

In comparison to previous versions of Genesis the definition of the undulator lattice is now completely defined in a lattice file. The main input file just refers to it and might change some parameters, such as adding undulator errors etc. Another major change is that the rigid scheme of fixed integration steps is now broken up. The lattice does not need to be aligned to a single integration step. Instead the lattice can be closer now to the real lattice.

The format of the lattice file is similar to other codes such as ELEGANT or MADX, which makes it somehow easier to convert between the formats. In general the syntax is the following:

label: *element type* = { *parameter* = *value* [, $\cdots$]};

Following beamline elements are currently supported: `undulator, quadrupole, drift, corrector, chicane, phaseshifter` and `marker`. For the parsing of the elements Genesis only consider the first 4 letters. Therefore `undulator` and `undu` as element name are both valid. This applies only for the elements. Other tags, such as labels and parameter have to be exact. Upper or lower cases are ignored because all letters are converted to lower case prior to parsing.

Labels are used to identify elements and are referred to in the line element. More information is given at the end of this section.

Undulator

- **aw**: The dimensionless rms undulator parameter. For planar undulator this value is smaller by the factor $1/\sqrt{2}$ than its K-value, while for helical undulator rms and peak values are identical. Default value is zero.
- **lambdau**: Undulator period length in meter. Default is 0 m.
- **nwig**: Number of periods
- **helical**: Boolean flag whether the undulator is planar or helical. Default value is `false`, indicating a planar undulator. Note that setting it to true, does not change the roll-off parameters. To be consistent they have to be set directly.
- **kx**: Roll-off parameter of the quadratic term of the undulator field in x. There are normalized with respect to k_u^2. Default is 0
- **ky**: Roll-off parameter of the quadratic term of the undulator field in y. Default is 1

- **ax**: Offset of the undulator module in x in meter. Default is 0.
- **ay**: Offset of the undulator module in y in meter. Default is 0.
- **gradx**: Relative transverse gradient of undulator field in x $\equiv (1/a_w)\partial a_w/\partial x$. Default is 0.
- **grady**: Relative transverse gradient of undulator field in y $\equiv (1/a_w)\partial a_w/\partial y$. Default is 0.

Drift

- **l**: Length of the drift in meter. Default value is 0 m.

Quadrupole

- **l**: Length of the quadrupole in meter. Default value is 0 m.
- **k1**: Normalized focusing strength in $1/\mathrm{m}^2$. Default value is 0
- **dx**: Offset in x in meter. Default is zero.
- **dy**: Offset in y in meter. Default is zero.

Corrector

- **l**: Length of the corrector in meter. Default value is 0 m.
- **cx**: Kick angle in x in units of $\gamma\beta_x$. Default value is 0.
- **cy**: Kick angle in y in units of $\gamma\beta_y$. Default value is 0.

Chicane

- **l**: Length of the chicane, which consists out of 4 dipoles without focusing. The first and last are placed at the beginning and end of the reserved space. The inner ones are defined by the drift length in between. Any remaining distance, namely the length subtracted by 4 times the dipole length and twice the drift length are placed between the second and third dipole. Default value is 0 m.
- **lb**: Length of an individual dipole in meter. Default value is 0 m

- **ld**: Drift between the outer and inner dipoles, projected onto the undulator axis. The actual path length is longer by the factor $1/\cos\theta$, where θ is the bending angle of an individual dipole.

- **delay**: Path length difference between the straight path and the actual trajectory in meters. From this value the bending angle is calculated internally by Genesis. Default delay is 0 m.

Phaseshifter

- **l**: Length of the phase shifter in meter. Default value is 0 m.

- **phi**: Change in the ponderomotive phase of the electrons in units of rad. Default value is 0. Note that Genesis is doing an autophasing, so that the electrons at reference energy are not changing in ponderomotive phase in drifts.

Marker

- **dumpfield**: A non-zero value enforces the dump of the field distribution of this zero length element.

- **dumpbeam**: A non-zero value enforces the dump of the particle distribution.

- **sort**: A non-zero value enforces the sorting of particles, if one-for-one simulations are enabled.

- **stop**: A non-zero value stops the execution of the tracking module. Note that the output file still contains the full length with zeros as output for those integration steps which are no further calculated.

Line

The line uses the labels to define the order of the elements in the beam line. Elements can be referred to multiple times. Each time the line element creates an individual copy so that they are not treated as a single entity. That allows that in a later stage the elements can be changed differently, e.g. when errors in the alignment are generated. The line element can also contain other line elements. Genesis is unrolling the nested references, however stops after an iteration depth of 10. This catches the error that one line element uses another and vice versa.

Elements can be duplicated by a preceding multiplication. The following is an example

```
FODO: Line = {F,DRI,D,DRI};
LAT: Line = {6*FODO};
```

There is one difference between Genesis and Elegant/Madx. It allows a superposition of secondary elements (everything except of undulator modules) so that correction and focusing can be superimposed onto the undulator field. An example is

```
UND: Undulator = {aw=1, lambdau=0.02, nwig=100};
FODO: Line = {UND,F@0.2,D@1.2};
```

The postfix @ with the position in meter places the element directly there. Note that it also resets the counter. For a 10 cm long quadrupole the next position after `F@0.2` would be 0.3 m. To avoid overlapping with succeeding elements it is better to define the pointer with a marker element, e.g. the upper example can be improved with:

```
UND: Undulator = {aw=1, lambdau=0.02, nwig=100};
FODO: Line = {UND,F@0.2,D@1.2,M@2.0};
LAT: Line = {6*FODO};
```

If the absolute position places the element outside of an existing element than the missing space is padded with a drift section.

Output and Input Files

Beside the input and lattice file, Genesis uses exclusively the HDF5 format for the output files, which are the main output file, and the particle and field dump. HDF5 is a well supported format by many programs or programming languages which makes it easy to read in the data. As an example, Matlab requires only a single line to read an entire dataset such as the 2d array for the power, with sample points along the undulator and beam frame, already converted to the right data type. This should make it easy to write your own postprocesser. Nevertheless the Genesis distribution comes with some Matlab routines to ease the parsing and display of the output files.

Main Output File

The main output file is written when ever the `&track` namelist is called. The name is given by the specified rootname and the extension `'.out.h5'` for the first call. If there are more than one tracking commands the root is extended by the run number, such as `'.Run3.out.h5'` for the third time of tracking.

In the root level of the HDF5 files there are multiple groups, to organize the output data. The group `'Global'` list the basic configuration of the simulation such as reference energy and wavelength or whether it is a time-dependent or scan run. The group `Lattice` contains all lattice information. The undulator strength, quadrupole field and other are resolved with the resolution of the requested integration step size, which is given in the dataset `dz`. For the z-position there exist two dataset. The regular one `z` has the same length and describes the lattice quantities from the position `z[i]` to `z[i]+dz[i]` of the ith integration step. The dataset `zplot` is used for plotting the beam or field parameters along the undulator. Note that those are evalutated before the integration started, so that there can be one more entry than the lattice datasets. Also if the output is reduced by the `output_step` option in the tracking command, the length of `zplot` is shorter because it has to match the length of the beam and field parameters. Another group is `Meta` which contains some additional information, which are not stricktly related to the input case. These are the version number, creation date and user, which has evoked the simulations.

The main data is contained in the `Beam` and `Field record`. In the case that additional harmonics have been selected, there are additional Field groups, where the harmonic number has been added. As an example `Field3` would refer to the third harmonic. In the `Beam` group there are parameters, which are evaluated at each integration steps. They are: energy, energy spread, bunching, bunching phase, centroid position in x, y, px and py and the size in x and y. Other parameters are only evaluated at the beginning, which are the optical function,

emittance and current profile, but in the future they might be becoming also larger, e.g. when sorting is enabled and the current profile can change. In the `Field` group one can find the power, position and size in x and y as well as the on-axis intensity and phase in the near field (central grid point) or far field (summation over all grid points)

In general the units of the datasets should be given as attributes to the given dataset. This hasn't been implemented yet.

Particle and Field Dump

Particle dumps are following the data structure that each slice has its own 6D distribution with the datasets `gamma, theta, x, y, px,` and `py`. In additions the local current value is written as well. The slice number is encoded in the group name, which is the composition of `slice` and the 6 digit representation of the slice numbers. Preceding spaces are changed to zeros. E.g. the group name of the 7th slice is `slice000007`. The total number of slices is given by the dataset `slicecount`. In addition some extra information is given on the root level of the reference frame are defined by the starting position `refposition` and the sample length per slice `slicespacing`. The reference length `slicelength` can be used to convert the ponderomotive phase into a longitudinal position in units of meter. Whether the distribution has been generated with beamlets or resolving all electrons are indicated by the parameters `beamletsize` and `one4one`.

In the field dump has very similar structure than the beam dump, except that instead of `slicelength` the field is `wavelength` to distinguish between the fundamental or harmonic radiation, while the beam is always measured against the fundamental wavelength for the ponderomotive phase. The field `gridsize` defines the extension of the 2D grid from the origin to the edge along of the major axis. The slice groups contains the real and imaginary part of the wavefront, however for programming reason the data record is one dimensional with $\texttt{ngrid}^2$ elements. Any processing or display should convert it to a symmetric 2D array.

Particle Distribution

Genesis can import particle distribution which are generated by Elegant.However it does not support the parsing of SDDS files directly. Instead it provides a simple shell script to convert the typical SDDS distribution into a HDF5 file. This script expects that the SDDS toolkit is installed because it extracts the columns `t, p , x, y, xp,` and `yp` from the SDDS files and converts them into datasets with the same name in the HDF5 file. The short program will add the extension `'.h5'` to the newly created file. This format is the one Genesis expects to be imported. The corresponding charge is supplied in the namelist and therefore does not need to be part of the HDF5 file.

The Postprocessor XGENESIS

Although the output format of Genesis is HDF5, where entire datasets can be read with a single line in Matlab, the source code distribution also contains some Matlab functions, which help with the postprocessing. They can be found in the directory `xgenesis` from the root directory of the source code. It is recommended that the search path of Matlab extends to that directory or that the files are copied to a locations where the search path already points to.

The following is a brief description of the existing functions. More will come in the future.

xgeninit

The routine requires only the name of the output file, which then is parsed for the most essential data, which is the size in z and s and the dataset tree structure. This function needs to be called once. All other functions will make use of the data, which has been parsed here. If another file needs to be read then `xgeninit` needs to be called again.

xgenplot

The main routine to display data from the output file and returns the data. It can generate multiple plots at the same time and the return data type is a cell array, with each element referring to a dataset in the plot. The dataset itself is a cell array as well, where the first element is the x-coordinate and the second the y.

The function requires up to three input arguments.In special cases, e.g. results from steady-state simulations some options are restricted such as plotting along the time-window coordinate is not meaningful and thus not supported.

The first element mostly refers to the corresponding dataset in the input deck. The function uses regular expression to match the dataset names with the requested plot data. As an example plotting `'size'` will plot the sizes in x and y of the electron beam and each supported radiation field, so at least 4 plots and even more if harmonics were included. To restrict the plots the requested plot can be more explicit, according to the tree structure in the output file. As an example the input argument `'/Beam/.size'` would plot the x and y sizes of only the electron beam while `'Field.*/.size` does it for the radiation of the fundamental and all harmonics. Besides the datasets in the output file there is one derived dataset, which is the spectrum. The postprocessor can calculate it either from the near field intensity and

phase on axis or the equivalent pair in the far field. If only `'spectrum'` is requested than both are plotted. To select one of them, just specify as it would be done for the intensity, e.g. `'spectrum-nearfield'`.

If no further arguments are supplied then the first slice of the specified data is plotted along the undulator axis. This is in particular enforced for any lattice function, such as the undulator field. Also if the output data is from a steady-state file then the output is always along undulator axis, despite that a profile might be requested.

The second argument specify the plotting direction: either along undulator direction or along bunch frame or frequency. If the data is a 2D array, e.g. such as the power after a time-dependent run, than plotting a line need as the third argument the specification of the orthogonal position. Here the post processor chooses the line, which lies the closest to the specified point. The default direction and plotting mode is `'normal'` plotting along the undulator. To plot the power as an example at 3 microns of the bunch frame the plot command is
`xgeninit('power','normal',3e-6)`.
The direction along the bunch frame is indicated with `'profile'` such as
`xgeninit('power','profile',11.5)`
to plot around the position 11.5 m within the undulator lattice.

For 2D datasets from time-dependent simulations or scans `xgenplot` allows some condensed plotting along the undulator: `'max'` for the maximum value of the given profile, `'mean'` the mean value, and `'rms'` the rms value. The evolution of the radiation bandwidth is calculated best, using the `'rms'` option. In fact, RMS values are only meaningful for primary datasets, which are power profile, spectrum and current profile. For all other parameters, such as beam size it is recommended to use the mode `'weighted'` which is defined as

$$\bar{P}(z) = \frac{\int P(s,z) \cdot W(s,z) ds}{\int W(s,z) ds}$$

where P is the parameter to be display and W the weighting function. For electron beam parameters it is the current while for radiation parameter it is the power profile.

For a two-dimensional output the modes `'2d'` and `'2dnorm'` can be used. The latter differs from the former that for each step along the undulator the resulting profile is normalized by its mean value. This allows to exclude the dominant exponential change in some parameters such as power, bunching or energy.

xgenwigner

This function calculates the 2D wigner distribution at a position within the undulator, which is supplied as the input argument. The Wigner distribution is a very compact display of time and spectral properties, namely the projection onto the time axis is the power profile and the projection on the frequency axis is the power spectrum.

Where to Find Further Information

GENESIS 1.3
`https://gitlab.ethz.ch/Genesis/Genesis4`
S. Reiche, "Numerical Studies for a Single Pass High Gain Free-Electron Laser", DESY print, DESY-THESIS-2000-012 (2000)

FEL Basics
E.L. Saldin, E.A. Schneidmiller and M.V.Yurkov, "The Physics of Free Electron Lasers", Springer, (2000)
J.B. Murphy and C. Pellegrini, "Introduction to the Physics of the FEL", Proceedings of the South Padre Island Conference, Springer (1986) 163

Numeric Basics
W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery, "Numerical Recipes in FORTRAN", Cambridge University Press, (1988)
W.F. Ames, "Numerical Methods for Partial Differential Equations", Academic Press (1996)

C++
B. Stroustrup, "C++ Programming Language", Addison-Wesley (1997)

OpenMPI
`https://www.open-mpi.org/`

HDF5
`https://support.hdfgroup.org/HDF5/`