Documentation for WakeTrajDoc

Gilles Maynard

01 April 2017

Part I.

Using the code

The recommended way of using WakTraj is to have one directory for each calculation. In this directory you will have both the input files and the output ones, while the program exec is somewhere else (usually in a parent directory). WakeTraj has several options and therefore can have many input parameters. In order to reduce these parameter, main input data are written as namelist (see below), in whic h you have to specify only the parameters needed for your calculation. Moreover it is recommended to have a spreadsheet (Excell or equivalent) for each calculation, allowing to establish the relation between the various parameters, to also to make the conversion between normal units and reduced ones. An example of such spreadsheets is given in WakeTrajExamples.

Concerning the input data, bet aware taht several parameters are provided, concerning options presently not implemented. Generally you have not to specify the values of these parameters in your input file, the program will used default values, corresponding to the already implemented modules.

1. General input

The general input data are in the namelists of WakeAC_input.par : An example is given below

1.1. Numerical parameters

The first namelist is related to numerical parameter :

```
$ParNumeric
EpsCFL = 0.1d0 !< used to calculate the value of DeltaT from Courant Friedrichs Lewy Condition
EpsRK = 0.01d0 !< used to calculate the time step of a Runge-Kutta integration
IDIMR = 600 !< number of numerical cells in the transverse grid
IDIMZ = 1000 !< number of numerical cells in the longitudinal grid
ProjOrderR = 2 !< type of projector grid-particle, in the radial dimension allowed values are 1 or 2
ProjOrderZ = 2 !< type of projector grid-particle, in the longitudinal dimension, allowed values are 1 or 2
RadiusGridCoef = 15.d0 !< coefficient to determine the radius of the grid
TimeOn_cm = -4.d0 !< position in z at starting (z = 0 is the laser focal plane)
TimeEnd_cm = 5.09d0 !< position in z at the end of calculation
withLaser = .True. !< determine whereas there is (true) or not (false) a laser beam
withBeamParticle = .True. !< determine whereas there is (true) or not (false) an injected electron beam
withPlasma = .True. !< determine whereas there is (true) or not (false) a plasma
withTestParticle = .False. !< determine whereas there is (true) or not (false) test particles
ZLengthGrid = 23.964254d0 !< total length of the numerical grid, in reduced units
$END
```

Remarks on the namelist format:

- text '!<' and after are comments, not used by the program
- minuscules are Majuscule are equivalent
- Real number can be written as x.yyyDz or x.yyyyEz or x.yyyyy. D instead of E is for increasing precision
- you can change the order of the lines

- presence of first line ($ParNumeric) and last one ($End) are mandatory
- all other lines are optional (if you do not specify a value, default one is used by the program)
- The radius of the grid is given by `RadiusGridCoef x Waist of the laser`
- Logical parameters can be written as .True., .False., True, False, T, F
- For TimeOn and TimeEnd, you can specify the value either inreduced units (TimeOn=) or in cm (TimeOn_cm =) or in units of Rayleigh lengthes (TimeOn_Rayleigh =)
- For the length of the grid, you can specify either the value in reduce units (ZlengthGrid=) or in cm (ZlengthGrid_cm =)

1.2. Laser parameters

If withLaser=True, the next namelist yields the values of laser parameters :

`$Parlaser`

`LaserDuration`_fs = 38.2195d0 !< duration (1/e^2) of the laser pulse

`LaserEnergy_J` = 2.d0 !< energy in J of the laser pulse

`LaserGridPosition_Duration` = 3.65d0 !< position of the maximum laser intensity in the grid in units of the laser duration

`LaserWaist0_cm` = 50.46266472d-4 !< waist of the laser in cm

`SpaceGaussianLaser` = .True. !< determine whereas the laser is gaussian in space (True) or not (False)

`TimeGaussianLaser` = .True. !< determine whereas the laser is gaussian in time (True) or not (False)

`$END`

Here again one can provides values either in reduced units (laserDuration =) or in specific units specified after the '-' sign (laserDuration_fs is the laser duration in fs). LaserGridPosition_Duration, is the laser grid position expressed in units of the laser duration. Can be useful to be sure that all the duration of the laser fit within the grid.

For the LaserEnergy, one can provide either the energy, or the maximum intensity (LaserMaxIntensity_Wcm2 =) or the value of the maximum of the reduced potential (LaserAmax =). This will be useful when the laser is not gaussian.

1.3. Plasma parameters

If withPlasma=True, the next namelist yields the values of the plasma parameters

`$ParPlasma`

`DensPlasma_cm3` = 2.d17 !< reference density of the plasma target in cm^3

`$End`

The program will also need an ascii file WakeTrajDensityProfile.dat. The firts line of this file should have the value of the (integer) parameter FormeDensityProfile :

0 !< value of FormeDensityProfile
Then depending on the value of FormeDensityProfile, one needs to provide more data :

- =0; Uniform density profile, no more parameter is needed (the relative density is equal to 1 everywhere)
- =1; relative density = fond1 for position < T1_cm, then linear evolution up to fond2 between T1_cm and T2_cm, then uniform up to T3_cm, linear evolution up to fond3 between T3_cm and T4_cm then uniform at fond4 value : first additional line yields values of fond1, fond2 and fond3 (real values separated by a comma), and next line yields the values of T1_cm, T2_cm, T3_cm and T4_cm
- =2 third order Spline-like interpolation profile. First additional line yield the number NPar of positionsin cm, between which the interpolation is perfomed, The next NPar lines yield, for each position, the value of this position, the value of the relativde density profile, and its first, second and third derivative. At position smaller that the firts point and larger than the last one the density is uniform. Points should be equidistant.

1.4. Beam particles parameter

If withBeamParticle= True, the next namelist in WakeAC_input.par, yields the value of beam parameters (presently only electrons). First case is for reading an Astra File
$ParExternalBeam
`BeamPosition` = 9.d0 !< Position of the reference electron in reduced units (can also be given in cm)
`BeamEpsMultiply` = 5.d-3 !< used for creating new particles, in case of MultiplyBeamPart > 1, it yields the average size of the width of each Phase Space coordinates in terms of the width of the initial distribution
`MultiplyBeamPart` = 1 !< default value; If one wants to make a calculation with more particles than included in the ASTRA file, then use MultiplyBeamPart, so that the total number will be MultiplyBeamPart x NtBeamPartRead
`NcaseBeamInjection` = 1 !< used to specify which type of injection; =0: Gaussian beam; =1: reading of an ASTRA output
`NtBeamPartRead` = 100000 !< number of particles to be read
`NumBeam` = 1 !< number of beams (presently only one)
`BeamDeltaX_cm` = 0.d0 !<delta in x of the beam compare to the reference axis
`BeamDeltaY_cm` = 0.d0 !< delta in y of the beam compare to the reference axis
`BeamPhi_mrad` = 0.d0 !< angle in mrad between the projection of the beam axis in the transverse plane and the x axis
`BeamTeta_mrad` = 0.d0 !< angle in mrad between the beam axis and axis z
$END
The ASTRA file name should be 'External_BeamPart.dat', in which a first line has been added to describe the data, the other lines being the orignal ASTRA output

If one wants to perform a calculation with a gaussian beam then NcaseBeamInjection should be put to 0, the number of particles is NtBeamPartRead, moreover one has to specified the distribution average and width. This is done by adding in namelist ParExternalBeam, the following lines:

`BeamGamMoy` = 19.d0 !< average value of the Lorentz factor

`BeamGamWidth` = 1.d-3 !< rms average value for $\delta\gamma_e/\gamma_e$

`BeamDuration_fs` = 100.d0 !< rms value in fs for the duration of the beam, exists also in reduced units

`BeamZfocX_cm` = -4.d0 !< Give the position in cm where <x.px> = 0.d0 (close to the focal plane in x), default value is TimeOn_cm

`BeamZfocY_cm` = -4.d0 !< Give the position where <y.py> = 0.d0 (close to the focal plane in y), default value is TimeOn_cm

`BeamWidthTetaX_mrad` =0.d0 !< rms value of the width of the angle px/pz in mrad

`BeamWidthTetaY_mrad` = 0.d0 !< rms value of the width of the angle py/pz in mrad

`BeamWidthX_cm` = 5.d-3 !< rms value of the width in x, exists also in reduced units. Value coresponding to the focal plane in x

`BeamWidthY_cm` = 5.d-3 !< rms value of the width in y in cm, exists also in reduced units. Value corresponding to the focal plane in y

1.5. OutPut

Finaly in WakeAC_input.par, there is a namelist controlling the output :

$ParOutPut

`BeamNumPrint` = 20000 !< number of printed particles, used if LPrintBeamPartT2u = .True.

`DeltatSort_cm` = 0.1d0 !< length between two output

`LPrintPlasmaField` = .true. !< determine whereas to print (true) or not (false) the plasma wave fields

`LPrintBeamPartT2u` = .True. !< determine whereas to print (true) or not (false) the particles properties

$END

2. Output Files

- WakeTraj_output.out: yields the main initial parameters of the calculation
- BeamGamXXXX.out : provide information on the beam particles at the position = XXXX times DeltaSort_cm. The first line identifies each column
- PlElecRXXXX.out: 2D table of the radial electric field (reduced units) at the position = XXXX times DeltaSort_cm
- PlElecZXXXX.out: 2D table of the longitudinal electric field (reduced units) at the position = XXXX times DeltaSort_cm
- check_ExternalBeam.out: gives the initial state of the beam : number of particle, x, y,z, px, py, pz, rad, rad/RadiusGrid, Gamma, p_perp