LCODE user manual

K.V. Lotov, A.P. Sosedkin

February 8, 2022

Contents

1 Overview

LCODE is a freely-distributed code for simulations of particle beam-driven plasma wakefield acceleration. The code is 2-dimensional (2d3v), with both plane and axisymmetric geometries possible. In the code, the simulation window moves with the light velocity, and the quasi-static approximation is used for calculating plasma response. The beams are modeled by fully relativistic macro-particles. The plasma is modeled either by macro-particles (kinetic solver), or as the electron fluid (fluid solver). Transversely inhomogeneous plasmas, hot plasmas, non-neutral plasmas, and mobile ions are possible with the kinetic solver. The code is furnished with extensive diagnosing tools which include the possibility of in-flight graphical presentation of the results.

The essence of the quasi-static approximation is illustrated by Figure 1. When we calculate the plasma response, the beam is considered as a "rigid" (not evolving in time) distribution of charges and currents which propagates with the speed of light c. The fields generated by this beam depend on the longitudinal coordinate z and time t only in combination $\xi = z - ct$ and can be found layer-by-layer starting from the beam head. Since the beam is not changing, all particles started from some transverse position r_0 copy the motion of each other, and their parameters (transverse coordinate and momenta) can be found as functions of ξ. Thus, a plasma macroparticle in the quasi-static model is not a "big" particle, but a "particle tube," i.e., a group of real particles started from a given radius with a given initial momentum. This greatly reduces the memory required for storing plasma particles.

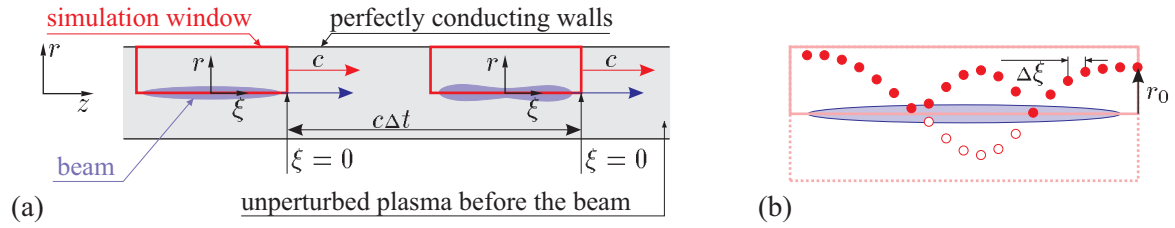

Figure 1: Geometry of the problem (a), and trajectory of a plasma particle in the simulation window (b).

The calculated fields are then used to modify the beam. For highly relativistic beams, the time step Δt for beam particles can be made large, which speeds up simulations several orders of magnitude. The quasi-static approximation is thus useful if and only if the time scale of beam evolution is much longer than the period of plasma wave.

Various details of LCODE and underlying physics are described in the following papers:

- K.V.Lotov, *Simulation of ultrarelativistic beam dynamics in plasma wake-field accelerator.* Phys. Plasmas **5** (1998), p.785-791. — The fluid plasma model.
- K.V.Lotov, *Fine wakefield structure in the blowout regime of plasma wakefield accelerators.* Phys. Rev. ST - Accel. Beams **6** (2003), p.061301. — The beam model and the kinetic plasma model.
- K.V.Lotov, *Blowout regimes of plasma wakefield acceleration.* Phys. Rev. E **69** (2004), p.046405. — Energy fluxes in the co-propagating window.

- K.V.Lotov, V.I.Maslov, I.N.Onishchenko, and E.N.Svistun, *Resonant excitation of plasma wakefields by a non-resonant train of short electron bunches.* Plasma Phys. Control. Fusion **52** (2010), p.065009. — Discussion on applicability of quasi-static codes to simulations of long beams.

- K.V. Lotov, A. Sosedkin, E.Mesyats, *Simulation of Self-modulating Particle Beams in Plasma Wakefield Accelerators.* Proceedings of IPAC2013 (Shanghai, China), p.1238-1240. — Upgrade of the kinetic plasma solver which was necessary for simulations of long beams.

2 Notation and units of measure

We use cylindrical coordinates (r, φ, ξ) for the axisymmetric geometry and Cartesian coordinates (x, y, ξ) for the plane geometry. The beam propagates in positive ξ-direction.

The code works with dimensionless quantities. Units of measure depend on some basic plasma density n_0. It is recommended to use the initial unperturbed plasma density as n_0. All times are in units of ω_p^{-1}, where $\omega_p = \sqrt{4\pi n_0 e^2/m}$ is the electron plasma frequency, e is the elementary charge, and m is the electron mass. All distances are in units of c/ω_p. The unit velocity is c. The notation used and units of measure for various quantities are given in Table 1.

Table 1: Notation, units of measure, and places of first appearance or definition for various quantities.

Notation	Quantity & place of definition	Unit
	Times:	ω_p^{-1}
t	Time in general (Sec. 1) or propagation time for the beam	
Δt	Main time step for the beam, Sec. 1, Sec. 5.1, **time-step**	
$\Delta t'$	Reduced time step for the beam, Sec. 5.1, **beam-substepping-energy**	
$t_{\max}$	Time limit for the run, Sec. 5.1, **time-limit**	
t_F	Period of the external beam focusing, Sec. 5.2, **foc-period**	
t_B	Oscillation period for the external magnetic field, Sec. 5.4, **magnetic-field-period**	
Δt_{out}	Periodicity of the detailed output, Sec. 5.6, **output-period**	
	Lengths:	c/ω_p
ξ	The co-moving coordinate, Sec. 1	
$\Delta\xi$	Longitudinal grid step, Sec. 3.1, Sec. 5.1, **xi-step**	
$\vec{r}_b$, r_b, x_b, ξ_b	Coordinates of a beam macro-particle, Sec. 3.3	
$\xi_{\max}$	Length of the simulation window, Sec. 5.1, **window-length**	
Δr	Transverse grid step, Sec. 5.1, **r-step**	
$r_{\max}$	Transverse size of the simulation window, Sec. 5.1, **window-width**	
r_p, r_{p2}	Width parameters for some plasma density profiles, Sec. 5.4.1, **plasma-profile**	
L_{trap}	Path limit for trapped plasma particles, Sec. 5.4.1, **trapped-path-limit**	
X_{b0}	Transverse displacement of the beam slice, Sec. 5.6.2, **f(xi)**	
R_b, X_b	Radius or half-width of the beam slice, Sec. 5.6.2, **f(xi)**	
ϵ	Emittance of the beam slice, Sec. 5.6.2, **f(xi)**	
ξ_{from}, ξ_{to}	Left and right boundaries of the subwindow, Sec. 5.6.1, **subwindow-xi-from**	
r_{from}, r_{to}	Bottom and top boundaries of the subwindow, Sec. 5.6.1, **subwindow-xi-from**	
r_{ax}, r_{aux}	Two transverse coordinates to output functions of ξ at, Sec. 5.6.2, **axis-radius**	
l_s	Length of the beam segment, Sec. 6.1, **length**	
$\delta\xi$	Distance to beginning of the beam segment, Sec. 6.1, **xishape**	
ξ_s	ξ-coordinate the segment begins at ($\xi_s < 0$), Sec. 6.1, **xishape**	
σ_r	Transverse size of the beam segment, Sec. 6.1, **xishape**	
σ_z	Length parameter for the Gaussian beam segment, Sec. 6.1, **xishape**	
x_0	Transverse displacement of the segment, Sec. 6.1, **vshift**	
	Velocities:	c
$\vec{v}$	Velocity of a plasma macro-particle or of an electron fluid element, Sec. 3.1, Sec. 3.2	
$\vec{v}_i$	Velocity of i-th plasma macro-particle, Sec. 3.1	
$\vec{v}_b$	Velocity of a beam macro-particle, Sec. 3.3	
	Momenta:	mc
$\vec{p}$	Momentum of a plasma macro-particle or of an electron fluid element, Sec. 3.1, Sec. 3.2	
$\vec{p}_b$	Momentum of a beam particle (not of a heavy macro-particle), Sec. 3.3	

Δp_{foc}	Extra momentum gained by a beam particle due to external focusing, Sec. 5.2, **focusing**	
$\vec{p}_e$	Momentum of a plasma electron, Sec. 5.6.1, **colormaps-full**	
$\vec{p}_i$	Momentum of a plasma ion, Sec. 5.6.1, **colormaps-full**	
$p_{b,\text{ref}}$	Reference value for displaying beam momentum, Sec. 5.6.4, **output-reference-energy**	
p_{b0}	Basic longitudinal momentum of beam particles in the segment, Sec. 6.1, **energy**	
p_a	Auxiliary value of the longitudinal momentum for the beam segment, Sec. 6.1, **espread**	
	Angular momentum:	mc^2/ω_p
M_b	Angular momentum of a beam particle, Sec. 7.1	
	Masses:	m
M	Mass of a plasma macro-particle, Sec. 3.1	
m_b	Mass of a beam particle, Sec. 3.3	
	Number densities:	n_0
n_e	Density of plasma electrons, Sec. 3.2, Sec. 5.6.1, **colormaps-full**	
n_i	Density of plasma ions, Sec. 3.2, Sec. 5.6.1, **colormaps-full**	
$n(r)$	Initial transverse profile of the plasma density, Sec. 5.4.1, **plasma-profile**	
n_{p2}	A parameter for some plasma density profiles, Sec. 5.4.1, **plasma-profile**	
	Charge densities:	en_0
ρ	Charge density of the plasma, Sec. 3.1	
ρ_b	Charge density of the beam, Sec. 3.1, Sec. 5.6.1, **colormaps-full**	
	Current densities:	ecn_0
$\vec{j}$	Total current density of plasma particles, Sec. 3.1	
$\vec{j}_b$	Current density of the beam, Sec. 3.1	
	Charges:	e
q	Charge of a plasma macro-particle, Sec. 3.1	
q_b	Charge of a beam particle, Sec. 3.3	
	Currents:	mc^3/e
I_{b0}	Base beam current, Sec. 5.2, **beam-current**	
I_b	Current of the beam slice, Sec. 6.1, **xishape**	
	Fields:	$E_0 \equiv mc\omega_p/e$
$\vec{E}$	Electric field in the plasma, Sec. 3.1	
$\vec{B}$	Magnetic field in the plasma, Sec. 3.1	
$\tilde{E}_r$, $\tilde{B}_r$	Auxiliary fields used in the kinetic plasma solver, Sec. 3.1	
B_0	External longitudinal magnetic field, Sec. 3.1, Sec. 5.4, **magnetic-field-type**	
B_{z0}	Variation amplitude for the external magnetic field, Sec. 5.4, **magnetic-field**	
E_{az}	Average longitudinal electric field acting on the beam slice, Sec. 5.6.2, **f(xi)**	
	Potential:	mc^2/e
Φ	Wakefield potential, Sec. 3.2	
	Energies:	mc^2
W_{kin}	Kinetic energy of a plasma electron, Sec. 3.1	
W_{ss}	Substepping energy for the beam, Sec. 5.2, **beam-substepping-energy**	
	Energy flux densities:	n_0mc^3
$\vec{S}$	Total energy flux density, Sec. 3.4, Sec. 5.6.1, **colormaps-full**	
$\vec{S}_f$	Collective energy flux density, Sec. 3.4, Sec. 5.6.1, **colormaps-full**	
$\vec{S}_e$	Electromagnetic energy flux density, Sec. 3.4, Sec. 5.6.1, **colormaps-full**	
	Energy fluxes:	n_0mc^5/ω_p^2
Ψ	Total energy flux along the simulation window, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
Ψ_w	Sum of Ψ and energy flux through window boundaries, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
Ψ_f	Collective energy flux along the simulation window, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
Ψ_e	Electromagnetic energy flux along the simulation window, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
	Energy densities:	n_0mc^2
W	Total energy density, Sec. 3.4, Sec. 5.6.1, **colormaps-full**	
W_f	Collective energy density, Sec. 3.4, Sec. 5.6.1, **colormaps-full**	
	Linear energy densities:	n_0mc^4/ω_p^2
W_{int}	Linear density of the total energy, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
dW_{int}	Difference between total and collective linear energy densities, Sec. 3.4, Sec. 5.6.2, **f(xi)**	
	Distribution functions:	
$f_\perp$	Sec. 6.1, **rshape**	$\omega_p/(m^2c^3)$
f_{4d}	Sec. 6.1, **rshape**	$\omega_p^2/(m^2c^4)$

$f_\parallel$	Sec. 6.1, **eshape**	$1/(mc)$
	Focusing strength:	$m\omega_p^2$
F_s	Strength of the external beam focusing, Sec. 5.2, **foc-strength**	
	Dimensionless:	
q_i	Charge of i-th plasma macro-particle, Sec. 3.1	
A	Normalization factor for calculation of plasma currents and charge densities, Sec. 3.1	
$\vec{e}_z$	Unit vector in z-direction, Sec. 3.2	
N	Quantity used by the fluid solver instead of the electron density, Sec. 3.2	
γ	Relativistic factor of a plasma particle or of a fluid element, Sec. 3.2	
N_r	Number of grid steps in the transverse direction, Sec. 5.1, **r-step**	
l	Number of twofold reductions of the time step for a low-energy beam particle, Sec. 5.2, **beam-substepping-energy**	
N_b	Number of beam macro-particles in the slice of current I_{b0}, Sec. 5.2.1, **beam-particles-in-layer**	
D_{ss}	Maximum sub-stepping depth allowed for the kinetic plasma model $(0 \dots 4)$, Sec. 5.4.1, **substepping-depth**	
A_{sub}	Sensitivity of the substepping trigger, Sec. 5.4.1, **substepping-sensivity**	
d_b	Fraction of beam particles to output, Sec. 5.5, **write-beam-particles**	
η_{draw}	Fraction of the simulation window to be drawn as the full window, Sec. 5.6.1,	
N_{mr}	Number of grid points to be merged transversely into a single pixel on colored maps, Sec. 5.6.1, **output-merging-r**	
$N_{m\xi}$	Number of grid points to be merged longitudinally into a single pixel on colored maps, Sec. 5.6.1, **output-merging-z**	
X_m, Y_m	Substitutes for axis dimensions, Sec. 5.6.2, Sec. 5.6.3	
D_b	Fraction of beam macro-particles to be drawn, Sec. 5.6.3, **draw-each**	
h_{fig}	Height of various pictures in pixels, Sec. 5.6.3, **beam-picture-height**	
N_{bins}	Number of histogram bins, Sec. 5.6.4, **histogram-bins**	
γ_{min}	Minimum relativistic factor for the plasma particle to be drawn, Sec. 5.6.5, **trajectories-min-energy**	
C_{step}	Color step for visualization of particle energies, Sec. 5.6.5, **trajectories-energy-step**	
N_{col}	Ordinal number of a color in the palette, Sec. 5.6.5, **trajectories-energy-step**	
α_b	Angular spread of the beam slice, Sec. 6, **angshape**	
α_0	Maximum angular spread in the segment, Sec. 6, **angspread**	
I_a	Maximum current in the beam segment, Sec. 6.1, **ampl**	

Throughout the manual, the following highlighting conventions are used:

`(filename.ext)` — names of various files (small typewriter font in parentheses);
`command -opt` — execution commands (typewriter font);
option — configuration option (boldface);

3 Underlying physics

3.1 Kinetic plasma model

The equations solved for the fields are Maxwell equations, which in the dimensionless variables take the form

$$\mathrm{rot}\,\vec{B} = \vec{j} + \vec{j}_b + \frac{\partial \vec{E}}{\partial t}, \qquad \mathrm{rot}\,\vec{E} = -\frac{\partial \vec{B}}{\partial t}, \qquad \mathrm{div}\,\vec{E} = \rho + \rho_b, \qquad \mathrm{div}\,\vec{B} = 0. \tag{1}$$

Under the quasi-static assumption

$$\frac{\partial}{\partial z} = -\frac{\partial}{\partial t} = \frac{\partial}{\partial \xi}, \tag{2}$$

equations (1) result in

$$\frac{1}{r}\frac{\partial}{\partial r} r E_r = \rho + \rho_b - \frac{\partial E_z}{\partial \xi}, \qquad \frac{1}{r}\frac{\partial}{\partial r} r B_r = -\frac{\partial B_z}{\partial \xi}, \tag{3}$$

$$\frac{1}{r}\frac{\partial}{\partial r} r (E_r - B_\varphi) = \rho - j_z, \qquad \frac{\partial E_z}{\partial r} = j_r, \qquad \frac{\partial B_z}{\partial r} = -j_\varphi, \qquad E_\varphi = -B_r. \tag{4}$$

Here we neglect the components j_{br} and $j_{b\varphi}$ of the beam current and put $j_{bz} = \rho_b$, since beam particles are assumed to move mostly in z-direction. To provide stability of the algorithm, we solve in finite differences, instead of (3), the following equations:

$$\frac{\partial}{\partial r}\frac{1}{r}\frac{\partial}{\partial r}rE_r - E_r = \frac{\partial(\rho+\rho_b)}{\partial r} - \frac{\partial j_r}{\partial \xi} - \tilde{E}_r, \qquad \frac{\partial}{\partial r}\frac{1}{r}\frac{\partial}{\partial r}rB_r - B_r = \frac{\partial j_\varphi}{\partial \xi} - \tilde{B}_r, \tag{5}$$

where $\tilde{E}_r$ and $\tilde{B}_r$ are some predictions for fields E_r and B_r. These equations are obtained by differentiation of (3) and substitution of (4) into the result. Subtraction of the fields (with or without the tildes) from both sides of the equalities does not produce a big error if the predictions are close to final fields. The boundary conditions for equations (4)–(5) are those of a perfectly conducting tube of the radius $r_{\max}$:

$$E_r(0) = B_r(0) = B_\varphi(0) = E_z(r_{\max}) = B_r(r_{\max}) = 0, \qquad \int_0^{r_{\max}} 2\pi r B_z\, dr = \pi r_{\max}^2 B_0, \tag{6}$$

where B_0 is an external longitudinal magnetic field, if any (the presence of this field does not change the axial symmetry of the system).

Each plasma macro-particle is characterized by 7 quantities: transverse coordinate (r), three components of the momentum (p_r, p_φ, and p_z), mass M, charge q, and ordinal number. Parameters of plasma macro-particles are initialized ahead of the beam (at $\xi = 0$) and then advanced slice-by-slice according to equations

$$\frac{d\vec{p}}{d\xi} = \frac{d\vec{p}}{dt}\frac{dt}{d\xi} = \frac{q}{v_z - 1}\left(\vec{E} + \left[\vec{v}\times\vec{B}\right]\right), \qquad \frac{dr}{d\xi} = \frac{v_r}{v_z - 1}, \qquad \vec{v} = \frac{\vec{p}}{\sqrt{M^2 + p^2}}. \tag{7}$$

If a particle hits the wall (at $r = r_{\max}$), it is returned to the simulation area to some near-wall location with zero momentum. Plasma current and charge density are obtained by summation over plasma macro-particles lying within a given radial interval:

$$\vec{j} = A\sum_i \frac{q_i\vec{v}_i}{1 - v_{z,i}}, \qquad \rho = A\sum_i \frac{q_i}{1 - v_{z,i}}, \tag{8}$$

where A is a normalization factor. The denominator in (8) appears since the contribution of a "particle tube" to density and current depends on the macro-particle speed in the simulation window.

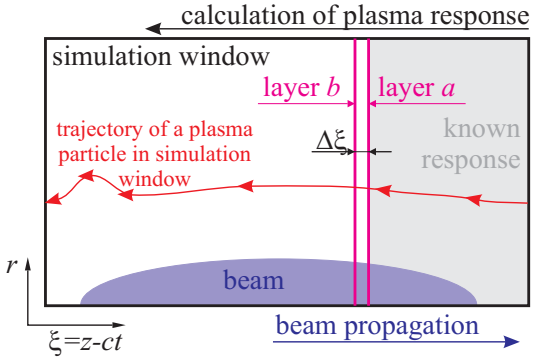

Figure 2: Calculation of plasma response in the quasi-static approximation.

The plasma response is calculated layer-by-layer towards the decreasing ξ (from right to left in Fig. 2). As far as for calculation of fields we need ξ-derivatives of currents, the following predictor-corrector scheme is used. We first move plasma particles from layer a to layer b by the fields of the layer a, then calculate currents in layer b, then calculate all fields in layer b, then move plasma particles from layer a to layer b by average fields of layers a and b, then again calculate currents and fields in layer b, then again move plasma particles from layer a to layer b by the average fields. When the fields are calculated first time, the radial fields from the previous layer are taken as $\tilde{E}_r$ and $\tilde{B}_r$. When the fields are calculated second time, the earlier found average radial fields are used as $\tilde{E}_r$ and $\tilde{B}_r$. Also, special efforts are made to suppress a small-scale (of the grid step size) plasma density noise.

The algorithm allows easy shortening of the ξ-step in the regions of a fine field structure. The shortening is made automatically if the plasma current density $|j_z|$ exceeds some threshold value.

In the plane geometry, instead of equations (4), (5) we solve

$$\frac{\partial^2 E_x}{\partial x^2} - E_x = \frac{\partial(\rho+\rho_b)}{\partial x} - \frac{\partial j_x}{\partial \xi} - \tilde{E}_x, \qquad \frac{\partial^2 B_x}{\partial x^2} - B_x = \frac{\partial j_y}{\partial \xi} - \tilde{B}_x, \tag{9}$$

$$\frac{\partial(E_x - B_y)}{\partial x} = \rho - j_z, \qquad \frac{\partial E_z}{\partial x} = j_x, \qquad \frac{\partial B_z}{\partial x} = -j_y, \qquad E_y = -B_x, \qquad \frac{\partial(E_x - B_y)}{\partial \xi} = j_x. \tag{10}$$

The last equation is used only at $x = 0.9\, r_{\max}$ to find the integration constant for B_y. The boundary conditions in the plane case are

$$E_z(0) = B_x(0) = E_z(r_{\max}) = B_r(r_{\max}) = 0, \qquad \int_0^{r_{\max}} B_z\, dx = r_{\max} B_0. \tag{11}$$

3.2 Fluid plasma model

In the fluid approximation, the plasma is characterized by the density n_e and momentum $\vec{p}$ of the electron component. Plasma ions are the immobile background of the density $n_i = 1$. Motion of the electron fluid is governed by the equation

$$\frac{\partial \vec{p}}{\partial t} + (\vec{v}\nabla)\vec{p} = -\vec{E} - \left[\vec{v} \times \vec{B}\right], \tag{12}$$

which, together with (1) and

$$\vec{j} = -n_e \vec{v}, \qquad \rho = 1 - n_e, \qquad \vec{v} = \vec{p}/\gamma, \qquad \gamma = \sqrt{1+p^2}, \tag{13}$$

forms the complete set of equations. This set has two constants of motion. The first one was derived by Khudik and Lotov[1]:

$$\vec{B} = \mathrm{rot}\, \vec{p} + n_e B_0 (\vec{e}_z - \vec{v}), \tag{14}$$

where B_0 is the unperturbed longitudinal magnetic field ahead of the beam. The second one is well known and comes from conservation of the generalized momentum:

$$\Phi = \gamma - p_z, \tag{15}$$

where Φ is the wakefield potential:

$$E_z = -\frac{\partial \Phi}{\partial \xi}, \qquad E_r - B_\varphi = -\frac{\partial \Phi}{\partial r}. \tag{16}$$

It is convenient to use the quantity $N = n_e(1 - v_z)$ instead of the electron density n_e and explicitly use the continuity equation which takes the form

$$\frac{\partial N}{\partial \xi} = \frac{1}{r}\frac{\partial}{\partial r}\frac{r N p_r}{\Phi}. \tag{17}$$

The final set of solved equations is, in the order of solving,

$$\frac{\partial \Phi}{\partial \xi} = -E_z, \qquad \frac{\partial p_r}{\partial \xi} = \frac{\partial p_z}{\partial r} + B_\varphi + \frac{N p_\varphi B_0}{\Phi}, \qquad N = 1 + \frac{1}{r}\frac{\partial}{\partial r} r \frac{\partial \Phi}{\partial r}, \tag{18}$$

$$\frac{\partial}{\partial r}\frac{1}{r}\frac{\partial}{\partial r} r p_\varphi - \frac{N p_\varphi}{\Phi} = -B_0 \frac{\partial N}{\partial r}, \qquad p_z = \frac{1 + p_r^2 + p_\varphi^2 - \Phi^2}{2\Phi}, \qquad \frac{\partial E_z}{\partial r} = -\frac{N p_r}{\Phi}, \tag{19}$$

$$B_r = -E_\varphi = -\frac{N p_r B_0}{\Phi} - \frac{\partial p_\varphi}{\partial \xi}, \qquad B_z = \frac{1}{r}\frac{\partial}{\partial r} r p_\varphi + B_0 N, \tag{20}$$

$$\frac{\partial}{\partial r}\frac{1}{r}\frac{\partial}{\partial r} r B_\varphi - \frac{N}{\Phi} B_\varphi = \frac{\partial j_b}{\partial r} - p_z \frac{\partial}{\partial r}\frac{N}{\Phi} + \frac{p_r}{\Phi}\frac{\partial}{r\partial r}\frac{r N p_r}{\Phi} + \frac{N p_r E_z}{\Phi^2} + \frac{N^2 p_\varphi B_0}{\Phi^2}. \tag{21}$$

Equations (18) are: the first equation of (16); the φ-component of (14) combined with (15); and the first equation of (4). Equations (19) are: the z-component of (14) differentiated with respect to r and combined with the third equation of (4); the definition of Φ (15) with the relativistic factor taken from (13); and the second equation of (4). Equations (20) are: the last equation of (4); the r-component of (14); and the z-component of (14). To obtain equation (21), we differentiate the z-component of the first equation in (1) with respect to r, use the expression (19) for $\partial E_z/\partial r$, and exclude ξ-derivatives of N, Φ, and p_r with the help of (17) and (18).

[1] V.N.Khudik and K.V.Lotov, *Ion channels produced by ultrarelativistic electron beams in a magnetized plasma.* Plasma Physics Reports, v.25 (1999), N 2, p.149-159.

In equations (18)–(21), only two quantities (Φ and p_r) need to be advanced to the next layer in ξ, all the others can be expressed in terms of Φ and p_r within the layer. The initial conditions for (18)–(21) are $\Phi = 1$ and $p_r = 0$. The necessary boundary conditions are

$$p_\varphi(0) = p_\varphi(r_{\max}) = 0, \qquad B_\varphi(0) = E_z(r_{\max}) = 0, \qquad \left.\frac{\partial(rB_\varphi)}{\partial r}\right|_{r=r_{\max}} = 0. \tag{22}$$

Equations (18)–(21) are solved with the predictor-corrector scheme.

In the plane geometry, the solved equations are

$$\frac{\partial\Phi}{\partial\xi} = -E_z, \qquad \frac{\partial p_x}{\partial\xi} = \frac{\partial p_z}{\partial x} + B_y + \frac{Np_yB_0}{\Phi}, \qquad N = 1 + \frac{\partial^2\Phi}{\partial x^2}, \qquad \frac{\partial^2 p_y}{\partial x^2} - \frac{Np_y}{\Phi} = -B_0\frac{\partial N}{\partial x}, \tag{23}$$

$$p_z = \frac{1 + p_x^2 + p_y^2 - \Phi^2}{2\Phi}, \qquad \frac{\partial E_z}{\partial x} = -\frac{Np_x}{\Phi}, \qquad B_x = -E_y = -\frac{Np_xB_0}{\Phi} - \frac{\partial p_y}{\partial\xi}, \qquad B_z = \frac{\partial p_y}{\partial x} + B_0N, \tag{24}$$

$$\frac{\partial^2 B_y}{\partial x^2} - \frac{N}{\Phi}B_y = \frac{\partial j_b}{\partial x} - p_z\frac{\partial}{\partial x}\frac{N}{\Phi} + \frac{p_x}{\Phi}\frac{\partial}{\partial x}\frac{Np_x}{\Phi} + \frac{Np_xE_z}{\Phi^2} + \frac{N^2p_yB_0}{\Phi^2}, \tag{25}$$

$$p_y(0) = p_y(r_{\max}) = 0, \qquad E_z(0) = E_z(r_{\max}) = 0, \qquad \left.\frac{\partial B_y}{\partial x}\right|_{x=0} = \left.\frac{\partial B_y}{\partial x}\right|_{x=r_{\max}} = 0. \tag{26}$$

3.3 Beam model

The beam is modeled by macro-particles. Each beam macro-particle is characterized by its longitudinal position ξ_b, transverse position r_b or x_b, three components of momentum $\vec{p}_b$, charge q_b, and mass m_b. Equations of motion for the macro-particles are

$$\frac{dr_b}{dt} = v_{br}, \qquad \frac{d\xi_b}{dt} = v_{bz} - 1, \qquad \frac{d\vec{p}_b}{dt} = q_b\vec{E} + q_b\left[\vec{v}_b \times \vec{B}\right], \qquad \vec{v}_b = \frac{\vec{p}_b}{\sqrt{m_b^2 + p_b^2}}. \tag{27}$$

These equations are solved with the modified Euler's method (midpoint method). The fields acting on the macro-particle are linearly interpolated to the predicted macro-particle location at the half time step. If a particle has a small longitudinal momentum and thus a high frequency of betatron oscillations, then the time step for this particle is automatically reduced.

With no external magnetic field ($B_0 = 0$), the angular momentum of beam particles must conserve, so the azimuthal component of the momentum $p_{b\varphi}$ is not changed according to (27), but reconstructed from the condition $r_b p_{b\varphi} = \text{const}$.

3.4 Energy fluxes and energy densities

In the presence of beams, there appears energy flows in the co-moving window[2]. These flows are composed by the energy flow in the laboratory frame and the energy transfer due to motion of the window. We can write the perturbation to the dimensionless flux density of the electromagnetic energy

$$\vec{S}_e = -\vec{e}_z\frac{E^2 + B^2 - B_0^2}{2} + \left[\vec{E} \times \vec{B}\right], \tag{28}$$

and the total energy flux density in the co-moving window

$$\vec{S} = \vec{S}_e + \sum(\gamma - 1)(\vec{v} - \vec{e}_z), \tag{29}$$

where the summation is carried out over plasma particles in the unit volume. For the fluid plasma model, the energy flux density is

$$\vec{S}_f = \vec{S}_e + n_e(\gamma - 1)(\vec{v} - \vec{e}_z). \tag{30}$$

The difference of the two, $\vec{S} - \vec{S}_f$, is the measure of the energy carried in the form of a thermal motion of plasma particles.

[2]K.V.Lotov, *Blowout regimes of plasma wakefield acceleration.* Phys. Rev. E, v.69 (2004), N 4, p.046405.

Integrating (28)–(30) across the simulation window gives us the energy fluxes against the z-axis:

$$\Psi_e = -\int_0^{r_{\max}} S_{ez}\, 2\pi r\, dr, \qquad \Psi = -\int_0^{r_{\max}} S_z\, 2\pi r\, dr, \qquad \Psi_f = -\int_0^{r_{\max}} S_{fz}\, 2\pi r\, dr. \tag{31}$$

The drive beam puts energy to some point of the simulation window, and then this energy flows backward or transversely until it exits the window or gets taken by a witness beam. The energy loss through transverse boundaries is taken into account by the quantity

$$\Psi_w = \Psi + \int_\xi^0 2\pi r_{\max} S_r(r_{\max}, \xi')d\xi', \tag{32}$$

which is the measure of beam-plasma energy exchange:

$$\frac{\partial \Psi_w}{\partial \xi} = \int_0^{r_{\max}} j_{bz} E_z\, 2\pi r\, dr. \tag{33}$$

The derivative $\partial\Psi_w/\partial\xi$ must be zero in the absence of beams; this can be used as a good test of precision for simulations. The difference between Ψ and Ψ_f can serve as a measure of the lost energy which cannot be retrieved by the accelerated beam. Formulae (31)–(33) are for the axisymmetric case, their modification for the plane geometry is straightforward.

The total energy density and fluid energy density are defined in the straightforward way:

$$W = \frac{E^2 + B^2 - B_0^2}{2} + \sum(\gamma - 1), \qquad W_f = \frac{E^2 + B^2 - B_0^2}{2} + n_e(\gamma - 1), \tag{34}$$

correspondingly; summation is over plasma particles the unit volume.

4 Running the code

4.1 Overview

To run LCODE, execute the file `(lcode.exe)` (Windows) or `(lcode)` (UNIX-like, execute with `./lcode`) in the folder where the input files are located (see 4.2). You should also provide configuration file(s) and/or options to LCODE, as described in (Sec. 4.3). LCODE will start creating the output files with the results of the simulations in the current directory. You may control the execution interactively as described in (Sec. 4.4). Upon completing the simulations LCODE process will exit, for the explanation of the exit codes refer to (Sec. 4.5). Using the MPI-enabled parallel version of LCODE is detailed in (Sec. 4.6).

4.2 Input files

Possible (optional) input files are:

`(lcode.cfg)`: the default configuration file (Sec. 5);
`(beamfile.bin)`: the beam state file (Sec. 7.1);
`(beamfile.bit)`: the continuation run indicator (Sec. 7.2);
`(plasma.bin)`: the plasma state file (Sec. 7.3);
`(fields.bin)`: the fields state file (Sec. 7.4);
arbitrarily named: other configuration files (Sec. 6), the beam constructor file (see **beam-profile**).

4.3 Configuration options and files

Additional command-line options can be provided after the executable name:

- Options `--option=value` overwrite options used in the main configuration file **lcode.cfg**.
- Directives `filename.ext` include other files containing configuration options.
- Special options interrupt the normal process of parsing command-line parameters and terminate execution of the program with the following actions:

`--help` **or** `--usage` outputs a brief description of all possible configuration options.

`--dump` **or** `--dump-config` outputs the already read values of configuration options to console. To save the output to a file (`config.cfg`), execute `lcode.exe --dump-config > config.cfg` (Windows) or `./lcode --dump-config > config.cfg` (UNIX-like).

`--dump-defconfig` outputs the the configuration file with default values of options to console. To save it to a file (`default.cfg`), execute `lcode.exe --dump-defconfig > default.cfg` (Windows) or `./lcode --dump-defconfig > default.cfg` (UNIX-like).

`--dump-docconfig` outputs the the configuration file with default values of options, but in a more detailed format. To save it to a file (`docconfig.cfg`), execute `lcode.exe --dump-docconfig > docconfig.cfg` (Windows) or `./lcode --dump-docconfig > docconfig.cfg` (UNIX-like).

`--beamfile-compose` allows concatenating and sorting files in beamfile.bin format (section 7.1). Please refer to the section 8.1 for details.

`--beamfile-inspect` allows inspecting the contents of the files in beamfile.bin format (section 7.1). Please refer to the section 8.2 for details.

If an option occurs several times in configuration files or command line, then the later value is used. The order of reading the values is the following:

1. Initially all options are set with default hard-coded values (the ones specified as default in this manual, Sec. 5).
2. The main configuration file 'lcode.cfg', if present, with all the files included. The file is read from the beginning to the end, with the included files, if any.
3. The command-line options processed left-to-right, if provided, with the included files, if any.

Here are some examples of using command-line options:

`./lcode`
Executes LCODE with the values read from lcode.cfg, if present. (UNIX-like)

`lcode.exe filename.cfg --beam-tune-charge=y`
The same as above, but also reads values from (`filename.cfg`) and explicitly defines the **beam-tune-charge** option. (Windows)

`lcode.exe filename1.cfg filename2.cfg --dump`
First reads values from (`lcode.cfg`), if available; then reads values from (`filename1.cfg`) and (`filename2.cfg`); then prints the resulting set of values to the screen. Simulations are not started. (Windows)

`./lcode filename1.cfg .cfg --time-limit=1000 --dump > new.cfg`
First reads values from (`lcode.cfg`), if available; then reads values from (`filename1.cfg`); modifies the value of th option time-limit; then writes the updated version of the config file to (`new.cfg`). Simulations are not started. (UNIX-like)

`./lcode --dump-defconfig > lcode.cfg`
Creates a default configuration file (`lcode.cfg`). Simulations are not started. (UNIX-like)

The option values are validated after reading. If some values are invalid, then the program attempts an automatic 'config recovery' which consists of resetting the invalid values to the hardcoded defaults. This action is indicated by warning messages. An example of a config recovery session is:

```
invalid histogram-bins=0 (Must be positive and <= 300) was reset to default 300
config recovered
Error filling config:  invalid config, some defaults used
The config is still valid though
Trying to use a recovered config
```

If the recovery has been completed successfully, the program attempts execution with the recovered config.

The final (possibly recovered) configuration values at the start of computations are written to (`lcode.runas.cfg`) to ease the diagnostics (see **save-config**, **save-config-filename**).

4.4 Interactive control

If the program is running in the terminal mode, its execution can be controlled by pressing some keys at the terminal:

- `','` (comma): pause the execution before the next time step, can be used repeatedly to pause after each time step;
- `'.'` (dot): stop the execution after finishing the time step, the run can be resumed afterwards;
- `' '` (space): pause the execution immediately;
- `'esc'` (escape): terminate the execution immediately;
- `'*'` (any key): resume the paused execution. Note that the feature is not tested in combination with the new experimental feature of performing parallel computations.

4.5 Exit codes

The exit codes are:

- `0` — execution ends successfully;
- `1` — execution fails, the program prints an error message and terminates;
- `2` — the program was stopped manually with the dot key;
- `3` — the program was interrupted manually with the escape key.

4.6 Parallel calculations with the MPI-enabled version

LCODE may be compiled with the ability of computing several time steps simultaneously by several processes executing in parallel. MPI (Message Passing Interface) is used for interprocess communication. In this mode the first process reads the beam file from disk and directs the new beam generation data to the input of the process 'trailing' it. The following processes also form a data processing chain, with the outputs of the former ones connected to the inputs of the latter ones. The last process writes the data to disk again.
Note that MPI-enabled version does not keep the whole beam in RAM and is considerably less RAM-hungry than the regular version. Also keeping the beam on disk removes the limitation on the amount of particles in the beam (see option **max-ram-megabytes** for details).
This way several cores of a single processors, multiple processors of a single system or even multiple processors of a distributed system with a shared storage could be used to speed up LCODE simulations by calculating more than one time step in a single pass.
This feature is still largely experimental, though it aims for full compatibility with the regular version of LCODE. The following known limitations apply:

- MPI-enabled version is only tested on Linux systems for now.
- Interactive process control is not tested in the MPI-enabled version. (Sec. 4.5).
- This is a relatively new feature and it is not heavily tested yet.

For the instructions on running the MPI-enabled version please contact your system administrator. Note that the MPI-enabled version must be compiled separately for different MPI execution environments. If you want to request a specific MPI-enabled build for your configuration, please specify your operating system version, MPI implementation vendor and version. If you are specifically designing your computer system to run LCODE, please consider using the latest stable Debian GNU/Linux system with OpenMPI, as this is the most tested platform by now.

5 Configuration file

The configuration file is a text file containing the list of options in the form

`option=value` or `option = value`.

The order of options is of no importance. If an option is specifies several times, the later value is used. Several options may coexist on the same line of the file, if they are separated by semicolons. Semicolons may be omitted in several cases, but it is strongly recommended not to rely on that. The configuration file can contain comments or empty lines which are ignored by the program, but improve visual readability of the file. Comments begin with `#` and extend to the end of line.

In this section, configuration options are described like this:
option, *type* (default value): brief description
Detailed description, if necessary.

Possible types of options are:

choice: a value from a predefined set.

float: a floating-point number, may be in scientific notation.

string: a string (multiple characters). Strings with spaces, semicolons or hash signs must be enclosed in double quotes, multiline strings must be enclosed in triple double quotes.

multichoice: multiple values from the predefined set, separated with commas.

y/n: 'yes' or 'no'. Brief forms 'y' and 'n' are also possible.

The following is the list of options grouped by purpose. The options present in `(lcode.runas.cfg)`, but not described here, are under development.

5.1 Simulation area

geometry, *choice* (c): The geometry of the problem:
'c' or 'axisymmetric': Axisymmetric (cylindrical) geometry
'p' or 'plane': Plane (2d Cartesian) geometry

window-width, *float* (5): Transverse size of the simulation window, $r_{\max}$
The radius in the cylindrical geometry or the full width in the plane geometry.

window-length, *float* (15): Length of the simulation window, $\xi_{\max}$

r-step, *float* (0.05): Transverse grid step, Δr
The number of grid steps in the transverse direction, N_r, must not be too large: $N_r = r_{\max}/\Delta r \leq 10000$. Otherwise Δr is automatically increased.

xi-step, *float* (0.05): Longitudinal grid step, $\Delta\xi$

time-limit, *float* (200.5): Time limit for the run, $t_{\max}$
Rounds to the nearest multiple of **time-step**. A special case of zero **time-limit** and **time-step** is allowed, in this mode LCODE generates the initial beam file and exits.

time-step, *float* (25): Main time step for the beam, Δt

continuation, *choice* (n): Mode of plasma continuation (Fig. 3):
'n' or 'no': Evolution of the beam. Every time step the beam enters an unperturbed plasma of a prescribed profile.
'y' or 'beam': Evolution of the beam sequence. Every time step (except the first one) the beam enters the perturbed plasma; the plasma state is taken from the end of the previous simulation window. Does not work for the fluid plasma model.
'Y' or 'longplasma': Evolution of the plasma. A long beam creates the wake, and long-term behavior of this wake is followed by the sequence of simulation windows. In this mode, only a rigid beam can extend to several simulation windows. Does not work for the fluid plasma model.
Latter two regimes need $\Delta t = \xi_{\max}$, otherwise Δt is automatically corrected.

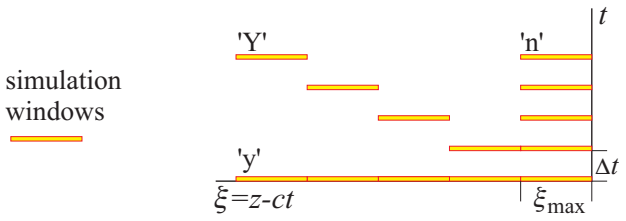

Figure 3: Illustration of continuation modes.

5.2 Particle beams

beam-current, *float* (0.1): Base beam current (in 17 kA), I_{b0}
The common multiplier (in units of $mc^3/e \approx 17\,\mathrm{kA}$) for dimensionless beam currents specified in **beam-profile**. In the plane geometry corresponds to the current through c/ω_p in the third dimension. Also the unit current for beam macro-particles.

rigid-beam, *y/n* (n): Switch for evolution of the beam
'y' or 'yes': A rigid (not evolving in time) distribution of the beam current.
'n' or 'no': The beam is modeled by macro-particles.

beam-substepping-energy, *float* (2): Substepping energy for the beam, W_{ss}
The threshold of reducing the time step for beam particles. For each beam particle, the minimal integer l is found that meets the condition $2^{2l}\sqrt{m_b^2+p_{bz}^2} > W_{\mathrm{ss}}$, and the reduced time step $\Delta t' = \Delta t/2^l$ is then determined. Plasma fields are calculated with periodicity Δt, and each beam particle is propagating in these fields with its own time step $\Delta t'$. This feature is particularly useful if low energy beam particles are present in the system, which otherwise would require undesirable reduction of the main time step Δt.

focusing, *choice* (n): External focusing for the beam:
'n' or 'no': No focusing
'c' or 'cosine': Cosine-varying focusing force. Each beam particle (located at radius r_b) at each time step gets the extra radial momentum $\Delta p_{\mathrm{foc}} = F_s r_b \Delta t \cos(2\pi t/t_F)$.
'r' or 'rectangular': Piecewise-constant focusing force. Each beam particle (located at radius r_b) at each time step gets the extra radial momentum $\Delta p_{\mathrm{foc}} = \pm F_s r_b \Delta t$, where "+" is chosen if the fractional part of $(t/t_F + 0.25)$ is less than 0.5, and "−" is chosen otherwise.
In the plane geometry, Δp_{foc} is the addition to x-momentum, and the distance between the particle and the midplane of the simulation window is used instead of r_b.

foc-period, *float* (100): Period of the external focusing, t_F
foc-strength, *float* (0.01): Strength of the external focusing, F_s

5.2.1 Newly generated beams

The following beam options determine parameters of newly generated particle beams or rigid beams. If the beam is allowed to evolve by **rigid-beam** and the files `(beamfile.bit)` and `(beamfile.bin)` are present in the working folder, then the beam state is imported from `(beamfile.bin)`, the time is read from `(beamfile.bit)`, and these options are ignored.

beam-particles-in-layer, *integer* (200): Number of beam particles in the layer (≤ 100000), N_b
A beam slice which has the current I_{b0} and length $\Delta\xi$ contains N_b macro-particles. Beam slices with lower currents have correspondingly fewer macro-particles.

beam-profile, *string* ("xishape=cos, length=1"): Initial distribution of beam particles
The name of the file that specifies the initial distribution of beam particles in 6d phase space, or the initial beam distribution itself (as a multiline description enclosed in triple quotes). If the value consists of a single line without spaces or equal signs, it is interpreted as a filename to read the beam distribution from; otherwise the value itself is considered the beam distribution description. The format of beam distribution is described in Sec. 6.

beam-tune-charge, *y/n* (n): Tuning the charge of beam particles to match the beam profile better
'n' or 'no': All beam particles are equally charged. Only the absolute value of the charge can be different. Consequently, the beam current can have only discrete values and therefore slightly differs from the specified value.
'y' or 'yes': The charge of particles in each beam slice is slightly modified to better match the specified beam profile. This option lowers the shot noise produced by the beam.

rng-seed, *integer* (1): Random generator seed
An integer number initiating the generator of random numbers. Different seeds generate different statistical ensembles for the beam.

5.3 Laser beams

LCODE currently supports either of the two laser beam models: a static laser beam model and an evolving laser beam model. The selection between two is done with the **laser-evolution** parameter; enabling the laser beam is done by the **laser** parameter.

laser, *choice* (n): Enable laser beams
Enables the laser beam (constant or evolving, see **laser-evolution**).

laser-evolution, *choice* (n): Enable laser beam evolution
Enables the laser beam evolution (requires **laser**).

5.3.1 Constant laser beam

Constant laser beam parameters are defined in file `(laser.dat)`. The format of the file is defined in Sec. 7.5.

5.3.2 Evolving laser beam

Evolving laser beam parameters are set directly in configuration file:

laser-amplitude, *float* (0.5): Laser amplitude
laser-xi-center, *float* (-5): Laser ξ center
The position of the center of the laser pulse in the window in ξ. Must be inside the window by at least **laser-xi-sigma**, i. e. not closer than **laser-xi-sigma** to both the front and the back of the window.

laser-xi-sigma, *float* (2): The width of the laser pulse in ξ
laser-r-autocenter, *choice* (n): Laser r autocenter
Ignore the value of **laser-r-center** and automatically position the laser-r-center at the middle of the window in r.

laser-r-center, *float* (0): Laser r center
The position of the center of the laser pulse in the window in r. Must be inside the window by at least **laser-r-sigma**, i. e. not closer than **laser-r-sigma** to the sides of the window. May be overridden with **laser-r-autocenter**.

laser-r-sigma, *float* (2): The width of the laser pulse in r
laser-wavenumber, *float* (250): Laser wavenumber, k_0
laser-steps-in-dt, *integer* (1): Laser substepping depth, N_{laser}
Laser substepping depth in relation to other solvers. Allows to make laser evolution steps N_{laser} times smaller than Δt and thus calculate N_{laser} laser evolution steps per one basic time step.

5.4 Plasma

plasma-model, *choice* (P): Plasma model:
'f' or 'fluid': The fluid model. It is the fastest one, but works only for the initially uniform plasma with immobile ions. Neither wave-breaking, nor near-wall plasma perturbations are allowed.
'P' or 'newparticles': New kinetic model described in Sec. 3.1.

magnetic-field, *float* (0): Variation amplitude for the external longitudinal magnetic field, B_{z0}
For zero B_{z0}, the code runs faster since some equations (which are identically zeros) are not solved.

magnetic-field-type, *choice* (c): Time dependence of the external magnetic field seen by the beam:
'c' or 'constant': Always $B_0 = B_{z0}$.
'r' or 'random': B_0 is a random value between 0 and B_{z0} for each time step.
'p' or 'periodic': $B_0 = B_{z0} \cos(2\pi t / t_B)$.

magnetic-field-period, *float* (200): Period of magnetic field oscillations, t_B
Used only with the periodic external magnetic field.

plasma-zshape, *string* (""): Longitudinal plasma density profile definition

This option defines longitudinal plasma density profile, i.e. plasma density as a function of time. A profile is defined in consecutive segments, one segment per line. Each line consists of a number that denotes the length of the segment (l), a number that defines an amplitude denominator (A), a letter that defines the shape of the segment and a shape parameter number P used for some of the shapes.

The available segment shape variants are:

'c' or 'c': Shifted cosine,	$f(t_m) = 0.5\,A(1 - \cos(2\pi t_m/l))$
't' or 't': OBSOLETE. Linear rise from 0 to A,	$f(t_m) = At_m/l$
'T' or 'T': OBSOLETE. Linear decrease fro A to 0,	$f(t_m) = A(1 - t_m/l)$
'l' or 'l': OBSOLETE. Constant,	$f(t_m) = A$
'L' or 'L': Linear rise/decrease from A to P,	$f(t_m) = A(1 - t_m/l) + P\,t_m/l$
'h' or 'half-cos': Rising half-period of the shifted cosine,	$f(t_m) = 0.5\,A(1 - \cos(\pi t_m/l))$
'b' or 'b': Decreasing half-period of the shifted cosine,	$f(t_m) = 0.5\,A(1 + \cos(\pi t_m/l))$
'g' or 'g': Decreasing part of Gaussian function with $\sigma = l/6$,	$f(t_m) = 0.5\,A\exp(-18\,\delta t_m^2/l^2)$
'G' or 'G': Parametrized curve, rising from 0 to A ($t \to$ inf),	$f(t_m) = A\,(1 - \exp(-t_m/P))$
'e' or 'e': Parametrized curve, rising exponentially from A,	$f(t_m) = A\exp(t_m/P)$
'E' or 'E': Parametrized curve, rising from $A/2$ to A ($t_m \to$ inf),	$f(t_m) = A(1 - 0.5\,\exp(-t_m/P))$

At each time step the plasma density is calculated as the value of the function defined with this option at the middle of the time step ($t_m = t - \delta t/2$). If the total length of the segments is too small, a density of 1 will be assumed for the undefined region. If the calculated plasma density is below 10^{-5}, it is set to 10^{-5}.

Example:
```
plasma-zshape = """
100 0 L 1.5
200 1.5 L 1.5
1500 3 E 150
"""
```

Here a profile is defined that rises linearly from 0 to 1.5 for the first 100 units of time (t_m from 0 to 100), stays 1.5 for the next 200 units of time (100 to 300), and rises from 1.5 to approx. 3 as $f(t_m - 100 - 200) = 3 - 1.5e^{(-t_m/150)}$ for the next 1500 units of time (300 to 1800). It defaults to 1 for the time steps in the undefined region after 1800 units of time.

In case this option gets enabled succesfully, LCODE will output the time t_m (not t!) and the calculated plasma density value at t_m to the `(plzshape.dat)` at each time step. Please check that this file gets created and actually contains the desired plasma profile to ensure you are using this option correctly.

5.4.1 Options specific to particle plasma models

These options have effect only if **plasma-model** is a kinetic one.

plasma-particles-per-cell, *int* (10): Number of plasma macro-particles per transverse grid cell

This option determines the number of plasma electron macro-particles per cell of transverse grid (total number of particles must be ≤ 100000). If mobile ions are enabled (**ion-model** = 'Y'), the same number of plasma ions macro-particles will be added to cell. Particles are added only to the cells, where plasma density according to **plasma-profile** is greater than zero.

plasma-particles-number, *int* (1000): Number of plasma macro-particles (≤ 100000)

All plasma electrons are modeled by this number of macro-particles. Mobile ions, if chosen, are modeled by the same number of heavier macro-particles. When this option is used **plasma-particles-per-cell** is not taken into account. This option should not be used and one should transition to **plasma-particles-per-cell** instead.

plasma-profile, *choice* (1): The initial transverse profile of the plasma density (Fig. 4):

'1' or 'uniform': Uniform (= 1) up to the walls.

'2' or 'stepwise': Uniform (= 1) up to r_p, zero at $r > r_p$.

'3' or 'gaussian': Gaussian $n(r) = \exp(-r^2/2r_p^2)$, zero at $r > 5r_p$; obtained by variation of macro-particles weights.

'4' or 'arbitrary': Arbitrary, with particle parameters imported from plasma-input-file and initial fields from fields-input-file.

'5' or 'channel': Zero up to r_p, uniform (= 1) at $r > r_p$.

'6' or 'sub-channel': Constant (n_{p2}) up to r_{p2}, then linear growth from n_{p2} to 1 between r_{p2} and r_p, then 1 at $r > r_p$; obtained by variation of macro-particles weights.

For the plane geometry, the distance to the midplane $|x - r_{\max}/2|$ is used instead of r.

plasma-input-filename, *string* ("plasma.bin"): Filename for importing plasma particles state.
fields-input-filename, *string* ("fields.bin"): Filename for importing initial fields state.
path-to-plasma-state, *string* ("."): Relative or absolute path to plasma state files.

Optional parameters for setting arbitrary plasma profile dependent on t. The filenames should be written in the form (fname*****.bin). Where (fname)s are specified by these options and (*****) correspond time step, where this profile is valid. If there is not available files for the current time step the closest plasma state will be chosen. In case of default values "plasma.bin" and "fields.bin" these files will be used for all time steps. This options are ignored in case of **continuation** = y/Y.

plasma-width, *float* (2): Main parameter of initial plasma density distributions, r_p
plasma-width-2, *float* (1): Auxiliary parameter of initial plasma density distributions, r_{p2}
plasma-density-2, *float* (0.5): Auxiliary parameter of initial plasma density distributions, n_{p2}

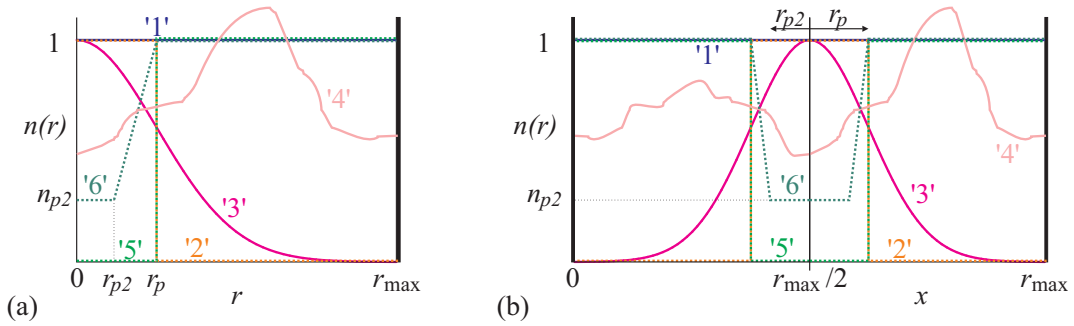

Figure 4: Illustration of possible plasma profiles in axisymmetric (a) and plane (b) geometries.

plasma-temperature, *float* (0): Initial temperature of mobile plasma particles in units of mc^2

ion-model, *choice* (y): Model of plasma ions:

'Y' or 'mobile': Half of plasma macro-particles are single-charged mobile ions initially located at the same positions as plasma electrons.

'y' or 'background': Ions are immobile background charge. If **plasma-profile** = '4', then the background ion density is the same as for uniform plasma.

'n' or 'absent': No ions, plasma electrons are initially at rest.

ion-mass, *float* (1836): Ion mass in units of the electron mass (for mobile ions)

substepping-depth, *integer* (3): Maximum sub-stepping depth allowed (0 . . . 4), $D_{\rm ss}$

Substepping is usually needed for strongly nonlinear wakefields when some plasma particles are close to trapping. If necessary, the longitudinal grid step can be automatically reduced up to $10^{D_{\rm ss}}$ times with respect to the basic **xi-step**.

substepping-sensivity, *float* (0.2): Sensitivity of substepping trigger, $A_{\rm sub}$.

If the longitudinal current density (j_z) is so high that the product of $|j_z|$ and ξ-step exceeds $A_{\rm sub}$ at some point, then the ξ-step is automatically divided by 10 (if allowed by **substepping-depth**). Reverse action (increasing the ξ-step) is taken when j_z gets small again.

trapped-path-limit, *float* (0): Path limit for trapped plasma particles, $L_{\rm trap}$

With this option, it is possible to treat trapping of plasma particles by the wakefield (to the extent allowed by the quasi-static approximation). The charge of a macro-particle is put to zero when the time spent by this macro-particle in the simulation window exceeds $L_{\rm trap}$. With this trick we obtain the correct plasma state and fields at the distance $L_{\rm trap}$ from the beam entrance to the plasma even if some plasma particles get trapped by the wakefield. It is necessary to put $L_{\rm trap} \gg \xi_{\max}$, otherwise the result will have no physical meaning. Zero value of $L_{\rm trap}$ switches this option off.

For every "trapped" macro-particle, 10 values are appended to file `(captured.pls)` (one particle per line):

1. Time
2. Number of the macro-particle (which contains information on its initial position)
3. Final ξ-coordinate
4. Final transverse coordinate

5-7. Final r-, φ-, and z-components of the particle momentum

8. Mass of the macro-particle
9. Charge-to-mass ratio
10. Relativistic factor

5.4.2 Option specific to the fluid plasma model

viscosity, *float* (0): Artificial viscosity

Artificial viscosity is used for suppressing high-frequency numerical noises or forcing the fluid model to work beyond the applicability area. Reasonable values are between 0 and 0.01.

5.5 Every-time-step diagnostics

If activated, these diagnostics work at each main time step Δt.

indication-line-format, *choice* (1): Format of the on-screen progress indication:

'1' or 'eachdt': One line each time step: time, total number of survived beam macro-particles, maximum and minimum electric field E_z on the axis.

'2' or 'eachdxi': One line each ξ-step: time, $|\xi|$, E_z on axis, total energy flux Ψ, number of beam macro-particles in this layer, number of sub-steps in ξ within the last ξ-step.

Location of the axis is determined by **axis-radius**. After finishing a time step (just before drawing pictures), the word "finished" is printed.

output-Ez-minmax, *y/n* (n): Write absolute extrema of the on-axis E_z into `(emaxf.dat)`
output-Phi-minmax, *y/n* (n): Write absolute extrema of the on-axis Φ into `(gmaxf.dat)`
output-Ez-local, *y/n* (n): Write local extrema of the on-axis E_z into `(elocf.dat)`
output-Phi-local, *y/n* (n): Write local extrema of the on-axis Φ into `(glocf.dat)`

If enabled, a line of 5 values is appended to the corresponding file immediately after an extremum is found:

1. Current time t
2. The maximum value of the quantity
3. ξ-coordinate of this maximum
4. The minimum value of the quantity (earlier found in case of local extrema)
5. ξ-coordinate of this minimum

Locations of the potential extrema are calculated more precisely than those of the field extrema. The latter are found as multiples of the grid step $\Delta\xi$.

write-beam-particles, *y/n* (n): Output of individual characteristics of beam particles

If the diagnostics is enabled, then the following information about selected beam particles is appended to file `(partic.swp)` at every time step, one line (9 values) per particle:

1. Time t,
2. Longitudinal coordinate ξ_b,
3. Transverse coordinate r_b or x_b,
4. Longitudinal momentum p_{bz},
5. Transverse momentum p_{br} or p_{bx},
6. Angular momentum M_b or third component of the momentum p_{by},
7. Charge-to-mass ratio (absolute value),
8. Current carried by the particle,
9. Ordinal number.

The same particle characteristics can be also extracted from `(beamfile.bin)`.

write-beam-particles-each, *integer* (1000): Fraction of beam particles to output, d_b

A beam particle is selected for output if its ordinal number is a multiple of d_b.

write-beam-particles-from, *float* (0): Right limit.

write-beam-particles-to, *float* (-10): Left limit

Only beam particles located between these two non-positive values are selected for output.

write-beam-particles-q-m-from, *float* (0): Lower limit for filtering beam particles by their q/m ratio

write-beam-particles-q-m-to, *float* (0): Upper limit for filtering beam particles by their q/m ratio

The interval of charge-to-mass ratios in which particles are written to `(partic.swp)`. Equal values of these parameters disable filtering.

output-lost-particles, *y/n* (n): Output of lost particles to `(beamlost.dat)`

If enabled, then beam particles which exit the simulation window are kept in file `(beamlost.dat)`. One line of 9 values is written for each particle. These values correspond to the last time particle was in the simulation window. The file format is the same as for `(partic.swp)`.

5.6 Periodical diagnostics

These diagnostics are periodically triggered with the time interval given by

output-period, *float* (100): Time periodicity of detailed output, Δt_{out}

If $\Delta t_{\text{out}} < \Delta t$, then each time step is diagnosed. The first time step is always diagnosed.

5.6.1 Colored maps

These keys control output of various quantities as functions of r and ξ, either in the form of a colored map, or in the form of a data array.

The colored maps are produced as separate files named `(??*****[w].png)` where `(??)` stands for two-character quantity abbreviation, `(*****)` is the time of output, and the optional suffix 'w' denotes subwindow output.

Data arrays have the same naming abbreviations, but different extension: `(??*****[m|w].swp)`. Suffix 'm' stands for full-window output, suffix 'w' denotes subwindow output. In the data files, one row is for one ξ-layer or sub-layer. Columns correspond to different transverse coordinates.

colormaps-full, *multichoice* (""): A list of quantities to output in the full window

colormaps-subwindow, *multichoice* (""): A list of quantities to output in the subwindow

Full window means the **drawn-portion** of the whole simulation window, the area of size $\xi_{\max} \times r_{\max}\eta_{\text{draw}}$. Subwindow size and location are controlled manually. Output of the following quantities is possible:

- 'Er': first transverse component of the electric field, E_r or E_x (`er*****[w]`.png/.swp)
- 'Ef': second transverse component of the electric field, E_φ or E_y (`ef*****[w]`.png/.swp)
- 'Ez': z-component of the electric field, E_z (`ez*****[w]`.png/.swp)
- 'Phi': wakefield potential, Φ (`fi*****[w]`.png/.swp)
- 'Bf': φ or y component of the magnetic field, B_φ or B_y (`bf*****[w]`.png/.swp)
- 'Bz': perturbation of the longitudinal magnetic field, $B_z - B_0$ (`bz*****[w]`.png/.swp)
- 'pr': first transverse component of the electron momentum, p_{er} or p_{ex} (`pr*****[w]`.png/.swp)
- 'pf': second transverse component of the electron momentum, $p_{e\varphi}$ or p_{ey} (`pf*****[w]`.png/.swp)
- 'pz': z-component of the electron momentum, p_{ez} (`pz*****[w]`.png/.swp)
- 'pri': first transverse component of the ion momentum, p_{ir} or p_{ix} (`ir*****[w]`.png/.swp)
- 'pfi': second transverse component of the ion momentum, $p_{i\varphi}$ or p_{iy} (`if*****[w]`.png/.swp)
- 'pzi': z-component of the ion momentum, p_{iz} (`iz*****[w]`.png/.swp)
- 'nb': charge density of particle beams, ρ_b (`nb*****[w]`.png/.swp)
- 'ne': perturbation of the plasma electron density, $n_e - 1$ (`ne*****[w]`.png/.swp)
- 'ni': perturbation of the plasma ion density, $n_i - 1$ (`ni*****[w]`.png/.swp)
- 'Sf': z-component of the total energy flux density (e.f.d.), S_z (`sf*****[w]`.png/.swp)
- 'Sr': r-component of the total e.f.d., S_r (`sr*****[w]`.png/.swp)
- 'dS': z-component of the thermal e.f.d., $S_z - S_{fz}$ (`sd*****[w]`.png/.swp)
- 'Sf2': r-weighted z-component of the total e.f.d., $2\pi r S_z$ (`2f*****[w]`.png/.swp)
- 'Sr2': r-weighted r-component of the total e.f.d., $2\pi r S_r$ (`2r*****[w]`.png/.swp)
- 'dS2': r-weighted z-component of the thermal e.f.d., $2\pi r(S_z - S_{fz})$ (`2d*****[w]`.png/.swp)
- 'Wf': total energy density, W (`wf*****[w]`.png/.swp)
- 'dW': thermal energy density, $W - W_f$ (`wd*****[w]`.png/.swp)
- 'SEB': z-component of the electromagnetic e.f.d., S_{ez} (`se*****[w]`.png/.swp)

If a quantity is undefined (like electron momentum at points of zero electron density), then zero is output at this point.

colormaps-type, *choice* (y): Controls whether functions of (r, ξ) are output as data files or as pictures:

'n' or 'numbers': Only data files.
'y' or 'pictures': Only pictures.
'F' or 'both': Both pictures and data files.

drawn-portion, *float* (1): Fraction of the simulation window to be referred as the full window, η_{draw}

A number between 0 and 1 (Fig. 5a). Used to exclude uninteresting near-wall regions from the output.

subwindow-xi-from, *float* (-5): Right boundary of the subwindow, ξ_{from}
subwindow-xi-to, *float* (-10): Left boundary of the subwindow, ξ_{to}
subwindow-r-from, *float* (0): Bottom boundary of the subwindow, r_{from}
subwindow-r-to, *float* (3): Top boundary of the subwindow, r_{to}

Four parameters controlling the size and position of the subwindow (Fig. 5b).

output-merging-r, *integer* (1): Merging in r, grid points, N_{mr}
output-merging-z, *integer* (1): Merging in ξ, grid points, $N_{m\xi}$

These numbers control output of one average value from several grid points (Fig. 5a). Values from the specified number of grid points (N_{mr} in r and $N_{m\xi}$ in ξ) are added together and divided by the number of points ($N_{mr}N_{m\xi}$). This procedure is made after the passage of the whole simulation window, when all the values are written to an auxiliary file. The subwindow output is not affected by these options.

palette, *choice* (d): Colormap palette: default/greyscale/hue/bluewhitered

Coloring method used to display values of functions on (r, ξ) plane (Fig. 6):

'd' or 'default': Distinctly-colored palette
'g' or 'greyscale': Greyscale palette
'h' or 'hue': Hue-equidistant palette
'b' or 'bluewhitered': Blue-white-red palette

Each palette has 8 distinct colors for positive values of quantities and 7 colors for negative values. One color is reserved for regions of zero density at plasma density maps (black for 'default' and 'greyscale' palettes, pure magenta for 'hue', and pure blue for 'bluewhitered').

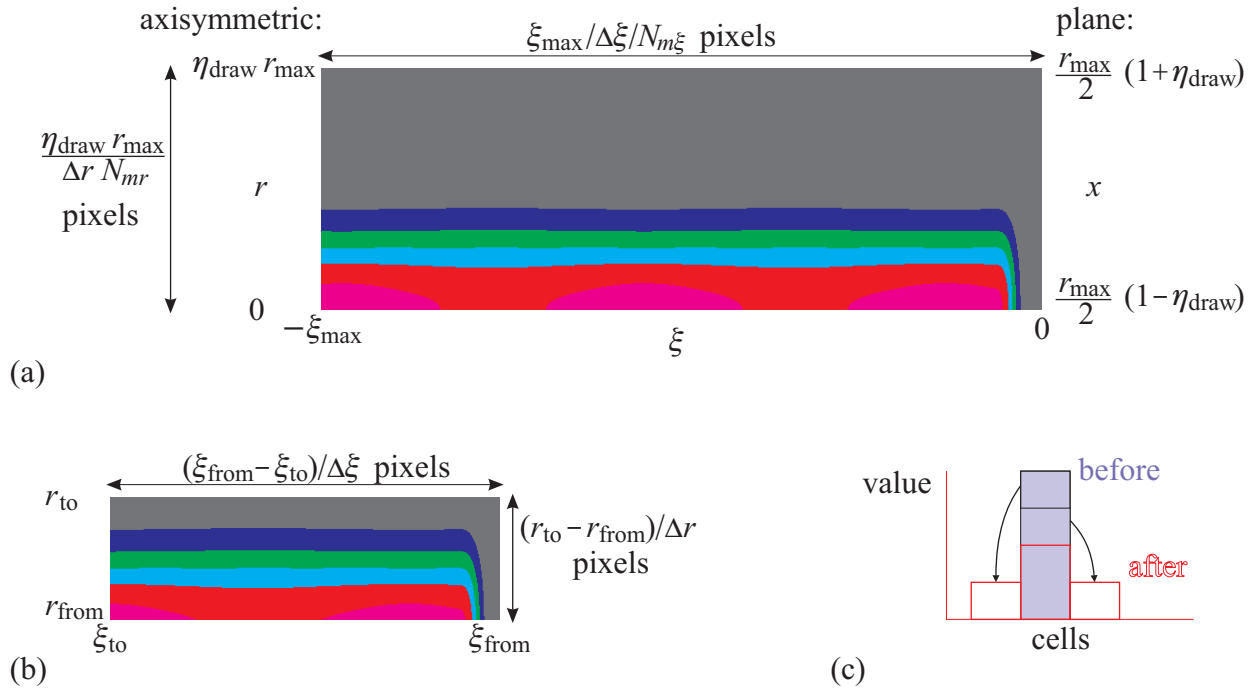

Figure 5: Physical and pixel sizes of the full-window colored map (a) and subwindow colored map (b); elementary smoothing procedure (c).

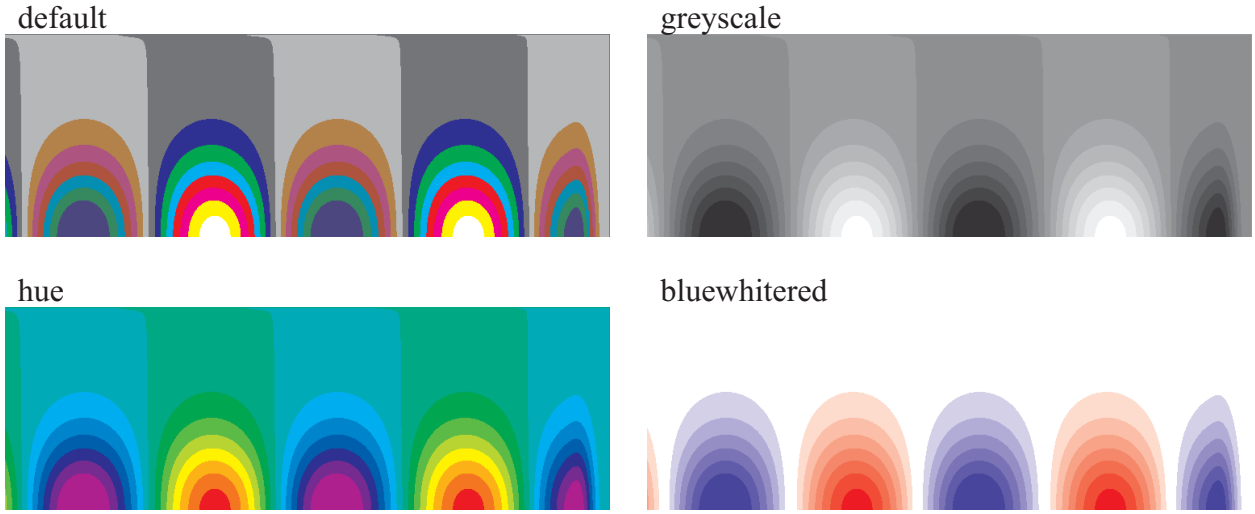

Figure 6: Examples of colormap palettes.

The following parameters (color steps) set up a correspondence between the increment of a quantity and color change on the picture:
E-step, *float* (0.1): Components of the electric field ('Er', 'Ef', 'Ez').
Phi-step, *float* (0.1): Wakefield potential ('Phi').
Bf-step, *float* (0.1): Transverse magnetic field ('Bf').
Bz-step, *float* (0.1): Longitudinal magnetic field ('Bz').
electron-momenta-step, *float* (0.1): Electron momenta ('pr', 'pf', 'pz').
ion-momenta-step, *float* (0.1): Ion momenta ('pri', 'pfi', 'pzi').
nb-step, *float* (0.1): Charge density of particle beams ('nb').
ne-step, *float* (0.1): Density of plasma electrons ('ne').
ni-step, *float* (0.01): Density of plasma ions ('ni').
flux-step, *float* (0.01): Components of energy flux densities ('Sf', 'Sr', 'dS', 'SEB').
r-corrected-flux-step, *float* (0.01): r-corrected energy flux densities ('Sf2', 'Sr2', 'dS2').
energy-step, *float* (0.01): Energy densities ('Wf', 'dW').

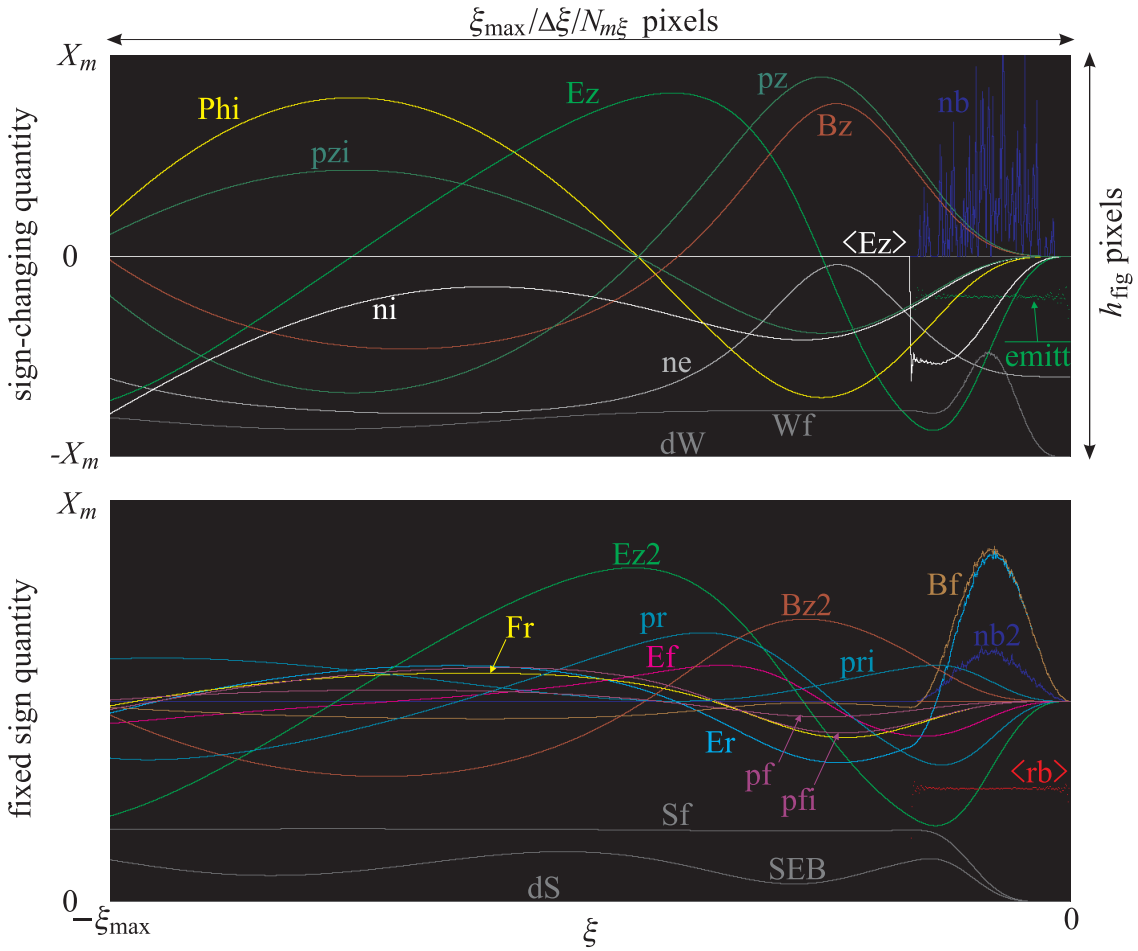

Figure 7: Physical and pixel sizes of the figures showing functions of ξ and colors used for displaying output quantities in the first group (top) and second group (bottom). The exact appearance of colors in the manual may depend on the viewing software.

5.6.2 Functions of ξ

These keys control output of various quantities as functions of ξ, either in the form of a graph, or in the form of a data array. The graphs are plotted in files named `(u*****.png)` or `(v*****.png)` by different colors (Fig. 7), where `(*****)` is the time of output in the 5-digit representation. The 5-digit representation is the integer part of time or its first five significant digits, if the time is greater than 10^6. The exact time and its 5-digit representation are appended to file `(times.dat)`. Corresponding data is scattered around `(xi_pre_*****.swp)`, where `pre` is a prefix for a physical quantity. Each file contains a single column with quantity values. The corresponding xi values are written to `(xi_*****.swp)`.

f(xi), *multichoice* (Ez,nb2,Ez2,dS,Sf,SEB): A list of quantities to output as functions of ξ.
The quantities can be listed in an arbitrary order. There are two groups of quantities, which differ by the picture they are plotted on. Indicated in braces are: the column number in the data array `(?*****.det)`, the color and style of the graph in `(?*****.png)`.

First group:				`(u*****.png/.det)`
'ne'	{2, grey line}	$n_e(r_{\rm ax})$	on-axis density of plasma electrons	`(xi_ne_*****.swp)`
'nb'	{3, blue line}	$\rho_b(r_{\rm ax})$	on-axis charge density of the beam	`(xi_nb_*****.swp)`
'Ez'	{4, green line}	$E_z(r_{\rm ax})$	on-axis z-component of the electric field	`(xi_Ez_*****.swp)`
'Eza'	{5, white line}	E_{az}	average longitudinal electric field acting on the beam slice	`(xi_Eza_*****.swp)`
'Bz'	{6, dark red line}	$B_z(r_{\rm ax}) - B_0$	on-axis z-component of the magnetic field	`(xi_Bz_*****.swp)`
'Phi'	{7, yellow line}	$\Phi(r_{\rm ax})$	on-axis wakefield potential	`(xi_Phi_*****.swp)`
'pz'	{8, dark green line}	$p_{ez}(r_{\rm ax})$	on-axis z-momentum of plasma electrons	`(xi_pz_*****.swp)`
'em'	{9, green points}	ϵ	emittance of the beam slice	`(xi_em_*****.swp)`
'dW'	{10, dark grey line}	$dW_{\rm int}$	thermal energy per unit length	`(xi_dW_*****.swp)`
'Wf'	{11, dark grey line}	$W_{\rm int}$	total energy per unit length	`(xi_Wf_*****.swp)`
'ni'	{12, white line}	$n_i(r_{\rm ax})$	on-axis density of plasma ions	`(xi_ni_*****.swp)`
'pzi'	{13, dark green line}	$p_{iz}(r_{\rm ax})$	on-axis z-momentum of plasma ions	`(xi_pzi_*****.swp)`
Second group:				`(v*****.png/.det)`
'nb2'	{2, blue line}	$\rho_b(r_{\rm aux})$	off-axis charge density of the beam	`(xi_nb2_*****.swp)`
'Er'	{3, cyan line}	E_r or $E_x(r_{\rm aux})$	off-axis transverse electric field	`(xi_Er_*****.swp)`
'Ez2'	{4, green line}	$E_z(r_{\rm aux})$	off-axis longitudinal electric field	`(xi_Ez2_*****.swp)`
'Bf'	{5, brown line}	B_φ or $B_y(r_{\rm aux})$	off-axis φ- or y component of the magnetic field	`(xi_Bf_*****.swp)`
'Bz2'	{6, dark red line}	$B_z(r_{\rm aux}) - B_0$	off-axis z-component of the magnetic field	`(xi_Bz2_*****.swp)`
'Fr'	{7, yellow line}		average focusing force, $(\Phi_{aux} - \Phi_{ax})(r_{\rm aux} - r_{\rm ax})$	`(xi_Fr_*****.swp)`
'pr'	{8, dark blue line}	p_{er} or $p_{ex}(r_{\rm aux})$	off-axis r- or x-momentum of plasma electrons	`(xi_pr_*****.swp)`
'pf'	{9, violet line}	$p_{e\varphi}$ or $p_{ey}(r_{\rm aux})$	off-axis φ- or y-momentum of plasma electrons	`(xi_pf_*****.swp)`
'rb'	{10, red points}	R_b or X_b	radius/width of the beam	`(xi_rb_*****.swp)`
'xb'	{11, dark red points}	0 or X_{b0}	transverse displacement of the beam (enabled by 'rb')	`(xi_xb_*****.swp)`
'dS'	{12, dark grey line}	$\Psi - \Psi_f$	integral thermal energy flux	`(xi_dS_*****.swp)`
'Sf'	{13, dark grey line}	Ψ	integral total energy flux	`(xi_Sf_*****.swp)`
'SEB'	{14, dark grey line}	Ψ_e	integral electromagnetic energy flux	`(xi_SEB_*****.swp)`
'pri'	{15, dark blue line}	p_{ir} or $p_{ix}(r_{\rm aux})$	off-axis r- or x-momentum of plasma ions	`(xi_pri_*****.swp)`
'pfi'	{16, violet line}	$p_{i\varphi}$ or $p_{iy}(r_{\rm aux})$	off-axis φ- or y-momentum of plasma ions	`(xi_pfi_*****.swp)`
'Ef'	{17, magenta line}	E_φ or $E_y(r_{\rm aux})$	off-axis φ- or y component of the electric field	`(xi_Ef_*****.swp)`
'Sw'	{18, dark grey line}	Ψ_w	integral total energy flux plus wall losses	`(xi_Sw_*****.swp)`

Where emittance ϵ of the beam slice is defined as:

generally: $\epsilon^2 = \left\langle \left(|\overrightarrow{r_{b\perp}}| - \langle|\overrightarrow{r_{b\perp}}|\rangle\right)^2 \right\rangle \left\langle \left(|\overrightarrow{p_{b\perp}}| - \langle|\overrightarrow{p_{b\perp}}|\rangle\right)^2 \right\rangle - \left\langle \left(|\overrightarrow{r_{b\perp}}| - \langle|\overrightarrow{r_{b\perp}}|\rangle\right)\left(|\overrightarrow{p_{b\perp}}| - \langle|\overrightarrow{p_{b\perp}}|\rangle\right) \right\rangle^2$,

cylindrical case: $\epsilon^2 = \langle r_b^2 \rangle \left(\langle p_{br}^2 \rangle + \langle p_{b\varphi}^2 \rangle \right) - \langle r_b p_{br} \rangle^2$,

plane case: $\epsilon^2 = \left(\langle x_b^2 \rangle - \langle x_b \rangle^2\right)\left(\langle p_{bx}^2 \rangle - \langle p_{bx} \rangle^2\right) - \left(\langle x_b p_{bx} \rangle - \langle x_b \rangle \langle p_{bx} \rangle\right)^2$.

Beam transverse displacement X_{b0} (is nonzero only in the plane geometry) is defined as $X_{b0} = \langle x_b \rangle$.
Beam radius R_b or half-width X_b is defined as: $R_b = \sqrt{\langle r_b^2 \rangle}$ or $X_b = \sqrt{\langle x_b^2 \rangle - X_{b0}^2}$.
The averaging is taken over all beam particles in the layer (the interval of width $\Delta\xi$, **xi-step**).

f(xi)-type, *choice* (n): Output mode for the functions of ξ:
'y' or 'pictures': Only pictures `(u*****.png)` and `(v*****.png)` are created.
'Y' or 'numbers': Only data files `(xi_pre_*****.swp)`.
'F' or 'both': Both pictures and data files are created.
'n' or 'n': No output of this kind.

axis-radius, *float* (0): Position of the 'probe' line 1, $r_{\rm ax}$
auxiliary-radius, *float* (1): Position of the 'probe' line 2, $r_{\rm aux}$
Two transverse positions at which functions of ξ are output. Both are measured from the geometrical axis (in cylindrical geometry) or middle of the simulation window (in plane geometry). It makes sense to put $r_{\rm ax} \neq 0$ only if the beam goes off-axis (in plane geometry) or if the plasma density is too noisy at the geometrical axis (in cylindrical geometry).

The following parameters (X_m) determine the size of the vertical axis when drawing functions of ξ (Fig. 7). The

axis is:
$(0,\ X_m)$ for fixed sign quantities,
$(-X_m,\ X_m)$ for sign-changing quantities.
For ion and electron densities, the axis type depends on X_m:
if $X_m \geq 1$, then $(0,\ X_m)$, the absolute value of the density is drawn,
if $X_m < 1$, then $(-X_m,\ X_m)$, the density perturbation is drawn.
E-scale, *float* (1): Most of the fields: 'Er','Ef', 'Ez', 'Ez2', '⟨Ez⟩', 'Bf', 'Fr'
Phi-scale, *float* (1): Wakefield potential: 'Phi'
Bz-scale, *float* (1): Longitudinal magnetic field: 'Bz', 'Bz2'
electron-momenta-scale, *float* (1): Electron momenta: 'pr, 'pf', 'pz'
ion-momenta-scale, *float* (1): Ion momenta: 'pri', 'pfi', 'pzi'
beam-radius-scale, *float* (5): Beam width and displacement: '⟨rb⟩'
nb-scale, *float* (1): Charge density of particle beams: 'nb', 'nb2'
ne-scale, *float* (2): Density of plasma electrons: 'ne'
ni-scale, *float* (1): Density of plasma ions: 'ni'
flux-scale, *float* (1): Integral energy fluxes: 'dS', 'Sf', 'SEB', 'Sw'
energy-scale, *float* (1): Energies per unit length: 'dW', 'Wf'
emittance-scale, *float* (5): Beam emittance: 'em'

5.6.3 Beam particle information as pictures

These keys control output of beam portraits in various parameter spaces. The selected beam macro-particles are drawn as cyan dots in pictures `(?*****.png)`, where `(?)` shows the content, and `(*****)` is the time.

output-beam-particles, *multichoice* (""): A list of beam projections to be drawn
Beam macro-particles can be output on the following planes in the following files:

'r':	(r, ξ) plane (real space)	`(p*****.png)`
'pr':	(p_{br}, ξ) plane (transverse momentum)	`(r*****.png)`
'pz':	(p_{bz}, ξ) plane (longitudinal phase space)	`(z*****.png)`
'M':	(M_b, ξ) plane (angular momentum)	`(m*****.png)`

draw-each, *integer* (20): Fraction of beam macro-particles to be drawn, D_b
A beam macro-particle is drawn if its ordinal number is a multiple of D_b.

beam-picture-height, *integer* (300): Height of figures in pixels, h_{fig}
This number determines the vertical size of f(xi) graphs, histograms, and beam portraits except the real space portrait. The latter has the same height as colored maps (equal to $\eta_{\text{draw}} r_{\max}/\Delta r/N_{mr}$).

output-reference-energy, *float* (1000): Reference value for displaying beam momentum, $p_{b,\text{ref}}$.

The following parameters (Y_m) determine the size of the vertical axis for corresponding beam portraits:
beam-pr-scale, *float* (100): Axis is $(-Y_m, Y_m)$ for r-momentum of beam particles ('pr')
beam-a-m-scale, *float* (100): Axis is $(0, Y_m)$ for the angular momentum of beam particles ('M')
beam-pz-scale, *float* (2000): Axis for z-momentum of beam particles ('pz')
The latter axis depends on the ratio between Y_m and **output-reference-energy** $p_{b,\text{ref}}$. If $Y_m > p_{b,\text{ref}}$, the axis is $(0, Y_m)$, otherwise it is $(p_{b,\text{ref}} - Y_m, p_{b,\text{ref}} + Y_m)$.

5.6.4 Beam information as histograms

These keys control output of beam characteristics in the histogram form, either as a picture `(d?*****.png)`, or as a data file `(d?*****.swp)`, where `(?)` shows the content, and `(*****)` is the time. If a macro-particle falls outside the chosen histogram interval, the particle is ignored.

Vertical size of the pictures in pixels is determined by **beam-picture-height** (Fig. 8a). The picture height in physical units equals the highest column. To retain the normalization, the line of 3 values is appended to file `(hystog.dat)` each time a histogram picture is produced. These values are:
1. Time
2. Histogram type (second character of the filename)
3. Height of the highest column.

The data files for histograms have 2 columns:

1. Number of the cell
2. Number of macro-particles in this cell.

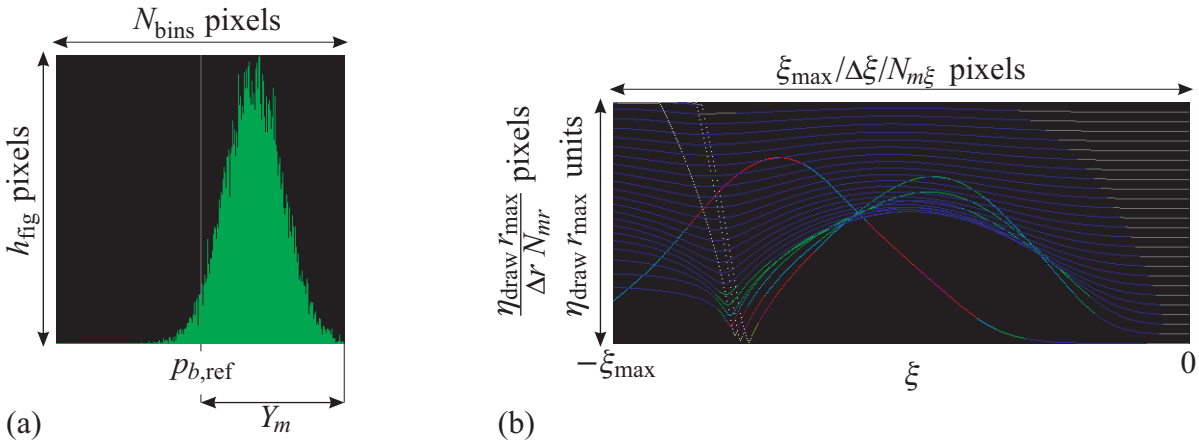

Figure 8: Exemplary histogram of beam longitudinal momentum (a) and trajectories of plasma particles in the blowout regime (b).

histogram-output, *multichoice* (""): Output of various beam properties in the histogram form
Available options, corresponding output quantities, keys determining histogram intervals, and corresponding file names are:

- 'r': transverse momentum, p_{br} or p_{bx}, **beam-pr-scale**, (`dr*****` .png .swp)
- 'z': longitudinal momentum, p_{bz}, **beam-pz-scale**, (`dz*****` .png .swp)
- 'M': angular or y-momentum, M_b or p_{by}, **beam-a-m-scale**, (`dm*****` .png .swp)
- 'a': angle between the particle momentum and z-axis, **beam-angle-scale**, (`da*****` .png .swp)

histogram-output-accel, *multichoice* (""): Output of histograms for accelerated particles only
The same as **histogram output**, but only beam macro-particles with z-momentum greater than **output-reference-energy** are taken into account. Available options, corresponding output quantities, keys determining histogram intervals, and corresponding file names are:

- 'r': transverse momentum, p_{br} or p_{bx}, **beam-pr-scale**, (`dR*****` .png .swp)
- 'z': longitudinal momentum, p_{bz}, **beam-pz-scale**, (`dZ*****` .png .swp)
- 'M': angular or y-momentum, M_b or p_{by}, **beam-a-m-scale**, (`dM*****` .png .swp)
- 'a': angle between the particle momentum and z-axis, **beam-angle-scale**, (`dA*****` .png .swp)

histogram-type, *choice* (y): Mode of the histogram output:
'y' or 'pictures': Pictures only,
'n' or 'data': Data files only,
'F' or 'both': Both pictures and data files.

histogram-bins, *integer* (300): Number of histogram bins (≤ 3000), $N_{\rm bins}$

beam-angle-scale, *float* (0.1): Axis size for the histogram of beam angular distribution
The axis is from zero to this value.

5.6.5 Trajectories of plasma particles

These diagnostics visualizes trajectories of plasma macro-particles in the simulation window, either as the picture `(pp*****.png)` (Fig. 8b), or as the data file `(*****.pls)`, where `(*****)` is the time. The diagnostics works for kinetic plasma models only. Each line of the data file contains 9 values corresponding to one position of one macro-particle:

1. Current ξ-coordinate
2. Number of the macro-particle
3. Transverse coordinate (r or x)

4-6. r-, φ-, and z-components of the particle momentum
7. Mass of the macro-particle
8. Charge-to-mass ratio
9. Relativistic factor

trajectories-draw, *choice* (n): Output mode for trajectories of plasma particles:
'n' or 'n': No output.
'y' or 'y': Picture mode. Trajectories of chosen plasma macro-particles on (r, ξ) plane are shown by dotted lines with a specified interval in ξ between dots. The dot color represents the gamma-factor γ of the particle. Sizes of the picture are the same as for full-window colored maps.
'"Y"' or '"Y"': Data mode. Information about chosen macro-particles is written to the data file with a specified interval in ξ between successive outputs.
'F' or 'F': Both picture and data file.

trajectories-each, *integer* (10): Fraction of plasma macro-particles to output
A trajectory of a plasma macro-particle can be drawn or written if the ordinal number of the macro-particle is a multiple of this number.

trajectories-spacing, *integer* (10): Spacing in ξ for consecutive output of macro-particle parameters
This interval is measured in units of the grid step $\Delta\xi$.

trajectories-min-energy, *float* (1): Minimum relativistic factor γ for the particle to be drawn, $\gamma_{\min}$
Particles with smaller relativistic factors are ignored. This is useful if you want to exclude weakly perturbed plasma regions from the output.

trajectories-energy-step, *float* (0.5): Color step for visualization of particle energies, C_{step}
The dot color in pictures is determined by $N_{\text{col}} = (\gamma - \gamma_{\min})/C_{\text{step}}$. The order of colors is the same as for the default palette of colored maps.

5.6.6 Detailed (substepped) plasma response

These keys control output of various plasma characteristics at a reduced ξ-step. The longitudinal subwindow for the output is the same as for subwindow colored maps (**subwindow-xi-from**, **subwindow-xi-to**, Sec. 5.6.1).

substepping-output-depth, *integer* (4): Substepping depth for the output (0..4)
How deep sub-stepping in ξ is included into the following output.

substepping-output-map, *y/n* (n): Substepped output of functions of (r, ξ)
If this option is enabled, then quantities chosen in **colormaps-subwindow** are also output with a reduced ξ-step into data files named `(??*****.det)`, where `(??)` is the two-character quantity abbreviation, and `(*****)` is the time. The subwindow for the output is the same as for subwindow colored maps. No merging of values is made. If this diagnostics is on, the file `(xi*****.det)` is also created, which has the same number of lines as data files and contains corresponding values of ξ, one value per line.

substepping-output-f(xi), *y/n* (n): Substepped output of functions of ξ
If enabled, then quantities chosen in **f(xi)** are also output with a reduced ξ-step into data files named `(u*****.det)` and `(v*****.det)`, where `(*****)` is the time. The file consists of several columns for numbers. The first column indicates ξ position, for the meaning of the other columns refer to the **f(xi)** option description.

substepping-output-particles, *y/n* (n): Substepped output of plasma particles
If enabled, information about plasma macro-particles is written to data file at each ξ-step or sub-step of the depth specified by **substepping-output-depth**. Particle filtering criteria imposed by **trajectories-each** and **trajectories-min-energy** are applicable.

substepping-output-particles-area, *choice* (f): Area for substepped output of plasma particles:
'n' or 'no': The output window is the same as for colored maps. Particle parameters are written to file `(*****.pls)`, where `(*****)` is the time.
'y' or 'yes': The output window is the same as subwindow for colored maps. Particle parameters are written to file `(s*****.pls)`, where `(*****)` is the time. The file format is the same as for `(*****.pls)`.

5.7 Saving run state

The following options control periodical savings of the run state.

saving-period, *float* (1000): Time interval between run saves
The run is saved if the time t differs from a multiple of the saving period by less then $\Delta t/2$. Saving is made after completion of the ξ-cycle and drawing all pictures.

save-beam, *y/n* (n): Saving the beam state
The state of the beam is saved in binary file `(tb*****.swp)`. The format of the file is the same as that of `(beamfile.bin)`. Option works for non-rigid beams only. Independently on this key, the beam state is also saved to `(beamfile.bin)` on reaching the time limit $t_{\max}$ or after manual interruption of the run (by the dot), so that the run can be continued without data loss.

save-plasma, *y/n* (n): Saving the plasma state
The plasma state at the end of the simulation window is saved to files `(pl*****.swp)` (plasma particles) and `(fl*****.swp)` (electric and magnetic fields). Formats of the files are the same as those of `(plasma.bin)` and `(fields.bin)` (Sec. 7.3, 7.4). Option works for kinetic plasma models only. Independently on this key, in regimes with plasma continuation the final plasma state is saved to `(plasma.bin)` and `(fields.bin)` on reaching the time limit $t_{\max}$ or after manual interruption of the run (by the dot), so that the run can be continued without data loss.

The run can be resumed from any saved time ***** by changing the content of `(beamfile.bit)` to *****, renaming `(tb*****.swp)` to `(beamfile.bin)`, and, if necessary, renaming `(pl*****.swp)` to `(plasma.bin)` and `(fl*****.swp)` to `(fields.bin)`. Any changes in code options (e.g., changes of diagnostics) will take effect on the resumed run.

5.8 Performance

max-ram-megabytes, *integer* (512): Allowed RAM usage for storing the beam, MB
This option defines the amount of RAM in megabytes that LCODE is allowed to use for storing the beam particles. This option is only useful for single-process calculations and is ignored for MPI-enabled version with more than one process. Set to zero if you don't want to store the beam particles in RAM; but keep in mind that it can result in performance loss and excessive disk usage. Note that setting the value too high may result in disabling this feature. In this case you should lower the value or set it to zero. Note that if it is set to a non-zero value that is not large enough to store the particle beam, LCODE will crash during the first time step of the computation. In this case you should increase the value or set it to zero.

5.9 Logging preferences

log-stdout-level, *choice* (d): Verbosity level for printing messages
log-file-level, *choice* (w): Verbosity level for logging to a file:
'"e"' or '"error"': Output error messages only
'"w"' or '"warning"': Output error and warning messages
'"i"' or '"info"': Output error, warning and info messages
'"d"' or '"debug"': Output error, warning, info and debug messages

log-filename, *string* ("lcode.log"): Name of the file to log to

log-file-clean, *y/n* (y): Remove empty log file
If enabled, then empty log files are automatically removed at the end of the run, if any.

save-config, *y/n* (y): Enable saving config to a file
save-config-filename, *string* ("lcode.runas.cfg"): Name of the file to save the current config to
This config includes the final version of all parameters, including those set by command-line switches. This may become useful in later debugging, for exact replication of the execution process, and especially after config recovery, as it reflects the automatic changes introduced by the recovery process.

5.10 Miscellaneous options

expected-commit-hash, *string* (""): Commit hash of the expected LCODE version
In case LCODE is executed with this option set to a commit hash, LCODE will warn user when its version differs from the expected one. If **save-config** option is enabled and **expected-commit-hash** is empty, LCODE will dump its built-in commit hash to **save-config-filename**.

6 Initial beam shape

There are two ways to define the initial distribution of beam particles in the six-dimensional phase space.

The first way is to specify all necessary parameters for each beam macro-particle. For this, information about the macro-particles must be written to the binary file `(beamfile.bin)` by an external program. The initial time must be written to the text file `(beamfile.bit)`. The code will read this information as the initial state of the beam.

The second way is to specify macroscopic parameters for individual beam segments. A segment is a beam piece of variable length l_s. The segments follow one by one starting from the beginning of the simulation window (at $\xi = 0$). A description of a segment has the format

```
parameter1=value1, parameter2=value2, ..., parameterN=valueN
```

If a parameter definition not followed by comma, then it is considered as the last one in the description of this segment. A description must contain at least one parameter defined. Segment descriptions follow one by one in **beam-profile** or in a separate file which **beam-profile** refers to.

Numerical parameters can be specified in scientific notation (like `1e6`) and postfixed with `PI` or `Pi` which acts like multiplication by 3.1415926.

If a parameter is not explicitly defined in the segment description, then it is assigned the default value. The initial default values are given in the following description. To modify default values for all subsequent segments, a special line prefixed with '`default:`' can be used. This line has the same syntax as segment descriptions, but does not define any segment and only changes the default values of some parameters. Re-definition of default values can be made several times.

6.1 Segment parameters

length, *float* (2PI): Length of the segment, l_s

ampl, *float* (0.5): Maximum current in the segment, I_a
Maximum current for this segment in units of **beam-current** I_{b0}.

xishape, *choice* (c): Dependence of the beam current on the longitudinal coordinate, $I_b(\xi)$
In the following variants, the formulae are applicable in the interval $l_s > \delta\xi > 0$, where $\delta\xi = \xi_s - \xi$, and the segment starts at $\xi_s \le 0$.

'c' or 'cos': Shifted cosine shape,	$I_b(\xi) = 0.5\, I_a I_{b0}(1 - \cos(2\pi\delta\xi/l_s))$.
't' or 't': Linear rise of the current,	$I_b(\xi) = I_a I_{b0}\, \delta\xi/l_s$.
'T' or 'T': Linear decrease of the current,	$I_b(\xi) = I_a I_{b0}(1 - \delta\xi/l_s)$.
'l' or 'l': Constant current,	$I_b(\xi) = I_a I_{b0}$.
'h' or 'half-cos': Rising half-period of the shifted cosine,	$I_b(\xi) = 0.5\, I_a I_{b0}(1 - \cos(\pi\delta\xi/l_s))$.
'b' or 'b': Decreasing half-period of the shifted cosine,	$I_b(\xi) = 0.5\, I_a I_{b0}(1 + \cos(\pi\delta\xi/l_s))$.
'g' or 'g': Decreasing part of Gaussian function with $\sigma_z = l_s/6$,	$I_b(\xi) = 0.5\, I_a I_{b0} \exp(-18\,\delta\xi^2/l_s^2)$.

radius, *float* (1): Radius of the beam segment, σ_r

energy, *float* (1000): Basic longitudinal momentum of beam particles, p_{b0}

vshift, *float* (0): Transverse displacement of the segment (used in plane geometry only), x_0

rshape, *choice* (g): Initial distribution of beam particles in the transverse phase space

Available options:

'g' or 'gaussian': Random distribution with the probability density

$$f_\perp(r_b, p_{br}, p_{b\varphi}) = \frac{r_b}{2\pi\sigma_r^2\alpha_b^2 p_{b0}^2}\exp\left(-\frac{r_b^2}{2\sigma_r^2}-\frac{p_{br}^2+p_{b\varphi}^2}{2\alpha_b^2 p_{b0}^2}\right), \quad \text{(axisymmetric beam)},$$

$$f_\perp(x_b, p_{bx}, p_{by}) = \frac{1}{(2\pi)^{3/2}\sigma_r\alpha_b^2 p_{b0}^2}\exp\left(-\frac{(x_b-x_0)^2}{2\sigma_r^2}-\frac{p_{bx}^2+p_{by}^2}{2\alpha_b^2 p_{b0}^2}\right), \quad \text{(plane geometry)}.$$

Here α_b is the angular spread of the beam slice, and x_b is measured from the central plane. For the Gaussian distribution, the beam current $I_b(\xi)$ and the beam density on the axis $\rho_b(0)$ are related as

$$I_b(\xi) = \frac{\rho_b(0)\sigma_r^2}{2}, \quad \text{(axisymmetric beam)},$$

$$I_b(\xi) = \frac{\rho_b(0)\sigma_r}{2\sqrt{2\pi}}, \quad \text{(plane geometry)}.$$

'r' or 'regular': The distribution with the same probability density as for 'gaussian', but with a regular distribution of beam particles. Means nothing for rigid beam mode.

'c' or 'cylinder': Cylinder-shaped distribution with the probability density

$$f_\perp(r_b, p_{br}, p_{b\varphi}) = \frac{r_b}{\pi\sigma_r^2\alpha_b^2 p_{b0}^2}\exp\left(-\frac{p_{br}^2+p_{b\varphi}^2}{2\alpha_b^2 p_{b0}^2}\right), \text{ for } r<\sigma_r,$$
$$0 \text{ otherwise.} \tag{35}$$

Also works in plane geometry:

$$f_\perp(x_b, p_{bx}, p_{by}) = \frac{\sqrt{{\sigma_r}^2-(x_b-x_0)^2}}{\pi^2\sigma_r^2\alpha_b^2 p_{b0}^2}\exp\left(-\frac{p_{bx}^2+p_{by}^2}{2\alpha_b^2 p_{b0}^2}\right), \text{ for } -\sigma_r+x_0<x<\sigma_r+x_0,$$
$$0 \text{ otherwise.} \tag{36}$$

'b' or 'brick': Brick-shaped distribution with the probability density

$$f_\perp(x_b, p_{bx}, p_{by}) = \frac{1}{2\pi\sigma_r\alpha_b^2 p_{b0}^2}\exp\left(-\frac{(x_b-x_0)^2}{2\sigma_r^2}-\frac{p_{bx}^2+p_{by}^2}{2\alpha_b^2 p_{b0}^2}\right), \text{ for } -\sigma_r+x_0<x<\sigma_r+x_0,$$
$$0 \text{ otherwise.} \tag{37}$$

Works in plane geometry only.

angspread, *float* (1e-5): Maximum angular spread in the segment, α_0

angshape, *choice* (l): Dependence of the angular spread on the longitudinal coordinate, $\alpha_b(\xi)$

In the following variants, the formulae are applicable in the interval $l_s > \delta\xi > 0$, where $\delta\xi = \xi_s - \xi$, and the segment starts at $\xi_s < 0$.

'c' or 'cos': Shifted cosine, $\alpha_b(\xi) = 0.5\,\alpha_0(1-\cos(2\pi\delta\xi/l_s))$.
't' or 't': Linear rise, $\alpha_b(\xi) = \alpha_0\,\delta\xi/l_s$.
'T' or 'T': Linear decrease, $\alpha_b(\xi) = \alpha_0(1-\delta\xi/l_s)$.
'l' or 'l': Constant, $\alpha_b(\xi) = \alpha_0$.
'h' or 'half-cos': Rising half-period of the shifted cosine, $\alpha_b(\xi) = 0.5\,\alpha_0(1-\cos(\pi\delta\xi/l_s))$.
'b' or 'b': Decreasing half-period of the shifted cosine, $\alpha_b(\xi) = 0.5\,\alpha_0(1+\cos(\pi\delta\xi/l_s))$.
'g' or 'g': Decreasing part of Gaussian function, $\alpha_b(\xi) = 0.5\,\alpha_0\exp(-18\,\delta\xi^2/l_s^2)$.

espread, *float* (0): Auxiliary value of the longitudinal momentum, p_a

eshape, *choice* (m): Longitudinal momentum distribution of beam particles, $f_\parallel(p_{bz})$

Available options:

'm' or 'monoenergetic': Monoenergetic, $f_\parallel(p_{bz}) = \delta(p_{bz} - p_{b0})$.

'u' or 'uniform': Uniform over the interval, $f_\parallel(p_{bz}) = (p_{b0} - p_a)^{-1}$ if $p_{b0} > p_{bz} > p_a$, $f_\parallel(p_{bz}) = 0$ otherwise.

'l' or 'linear': Linear energy growth from p_a to p_{b0}, $f_\parallel(p_{bz}) = \delta\big(p_{bz} - p_{b0}\,\delta\xi/l_s - p_a(1 - \delta\xi/l_s)\big)$.

'g' or 'gaussian': Gaussian distribution, $f_\parallel(p_{bz}) = \dfrac{e^{-(p_{bz}-p_{b0})^2/(2p_a^2)}}{\sqrt{2\pi}p_a}$

'N' or 'N': N monoenergetic fractions (N is a digit, $2 \ldots 9$),

$$f_\parallel(p_{bz}) = \sum_{i=0}^{N-1} \delta(p_{bz} - p_i), \qquad p_i = p_a + \frac{i}{N-1}(p_{b0} - p_a)$$

m/q, *float* (1): Absolute value of mass-to-charge ratio, compared to the electron

6.2 Example of a beam profile

```
default:  angspread=3e-3, energy=2000
xishape=cos, ampl=1, length=6PI
```

The first line redefines the default values for two options.

The second line defines the beam segment of the length 6π, current $I_b(\xi) = 0.5\, I_{b0}(1-\cos(|\xi|/3))$, and Gaussian transverse distribution (hardcoded default value) with $\sigma_r = 1$ (hardcoded default value) and $\alpha_b = 0.003$ (hardcoded default ξ-dependence, new default value). All particle in the segment have the same longitudinal momentum (hardcoded default distribution) that equals 2000 (new default value). Particles are either electrons or positrons, depending on the sign of I_{b0}.

7 Format of other input and output files

7.1 beamfile.bin

This file contains information about beam macro-particles in the binary form. For each particle, 8 values are are written, 8 bytes each (type double). These values are:

1. Longitudinal position, ξ_b.
2. Transverse position, r_b (cylindrical geometry) or x_b (plane geometry).
3. Longitudinal momentum, p_{bz}.
4. Transverse momentum, p_{br} (cylindrical geometry) or p_{bx} (plane geometry).
5. Angular momentum M_b (cylindrical geometry) or third momentum component p_{by} (plane geometry).
6. Absolute value of the charge-to-mass ratio, compared to electron.
7. Charge carried by the macro-particle. The charge unit is $\Delta\xi mc^2/(2e)$.
8. Ordinal number of the macro-particle.

As follows from the data format, the actual beam charge depends on the dimensionless longitudinal grid step **xi-step**. If the beam generated with some $\Delta\xi$ is used in a continuation run with different $\Delta\xi$, then the beam density seen by the code will differ from the original one by the ratio of grid steps.

The last macro-particle in `(beamfile.bin)` always has $\xi_b = -100000.0$ and serves as the end-of-file sign.

7.2 beamfile.bit

Contains the current time (a decimal number in the text form) and serves as the indicator of the continuation mode. If this file is present, the code tries to continue a previous run.

7.3 plasma.bin

File `(plasma.bin)` contains information about plasma macro-particles (one line for one macro-particle). It has 6 columns:

1. Transverse coordinate of the macro-particle (r or x) counted from the bottom of the simulation window.

2. First component of the transverse momentum (p_r or p_x) of the macro-particle (not of a single electron or ion). It equals the particle velocity times relativistic factor times "mass" of the macro-particle.

3. Second component of the transverse momentum (p_φ or p_y) of the macro-particle.

4. Longitudinal momentum (p_z) of the macro-particle.

5. "Mass" of the macro-particle. For uniform electrons of the unit density (i.e., of the density n_0), the "mass" is just the cross-section corresponding to this macro-particle (it is a ring in the cylindrical geometry or a rectangle in the plane geometry). For heavier particles, the "mass" is correspondingly greater.

6. "Charge" of the macro-particle. It is the "mass" times the real charge-to-mass ratio for particles of this sort.

When the run is normally terminated or interrupted in the regimes of a long plasma, the final parameters of all macro-particles are automatically saved to `(plasma.bin)`. If the run is then continued, the initial plasma parameters are taken from `(plasma.bin)`. The same happens if a new run is started with an arbitrary initial plasma profile.

7.4 fields.bin

File `(fields.bin)` contains information about the fields and plasma currents. Each line in `(fields.bin)` corresponds to a grid point. Number of lines is $N_r + 1$ (number of grid steps plus one). Each line contains 8 values:

$E_r,\ E_\varphi,\ E_z,\ B_\varphi,\ B_z,\ p_r,\ p_\varphi,\ p_z$ for cylindrical geometry and fluid plasma model,
$E_x,\ E_y,\ E_z,\ B_y,\ B_z,\ p_x,\ p_y,\ p_z$ for plane geometry and fluid plasma model,
$E_r,\ E_\varphi,\ E_z,\ B_\varphi,\ B_z,\ j_r,\ j_\varphi,\ j_z$ for cylindrical geometry and kinetic plasma model,
$E_x,\ E_y,\ E_z,\ B_y,\ B_z,\ j_x,\ j_y,\ j_z$ for plane geometry and kinetic plasma model.

Here p_α are components of the electron momentum; j_α are components of the total current density.

When the run is normally terminated or interrupted in the regimes of a long plasma, the final fields are automatically saved to `(fields.bin)`. For kinetic plasma models, the run can be then continued with the initial fields taken from `(plasma.bin)`. The same happens if a new run is started with an arbitrary initial plasma profile.

7.5 laser.dat

The laser.dat file is used for describing one or more non-evolving laser pulses (see **laser**, **laser-evolution**). The syntax is as follows:

```
Laser pulse 1,
radius, sigma_r:  1
length, sigma_z:  1
strength, a_0:  0.2
frequency, omega:  5
center (xi):  -5
last pulse (y/n):  y

draw a^2 intensity map (0,1,2,3):  1
draw on-axis strength :  y
```

Setting **last pulse** to 'n' allows to specify additional pulses, e. g.

```
Laser pulse 2,
radius, sigma_r:  3
length, sigma_z:  1
strength, a_0:  0.25
frequency, omega:  5
center (xi):  -8
last pulse (y/n):  y
```

7.6 r*****.det

This file contains auxiliary information for subwindow diagnostics (Sec. 5.6.1, 5.6.6). If any sub-window diagnostics is on, then 3 values are written (not appended) to file `(f*****.det)`, where `(*****)` is the time:

1. Transverse grid step.
2. Lower boundary of the sub-window.
3. Number of ξ-steps inside the sub-window.

7.7 times.dat

This file relates exact times of diagnostic outputs and their 5-digit representations. A line of two values is appended to `(times.dat)` on each output of functions of ξ (Sec. 5.6.2):

1. The integer used in the file name;
2. The corresponding time.

7.8 Diagnostics files

These files are generated by diagnostic tools described in corresponding sections of the manual, and their format is explained on the following pages:

File name	Section	Page
`(beamlost.dat)`	5.5	17
`(elocf.dat)`	5.5	16
`(emaxf.dat)`	5.5	16
`(glocf.dat)`	5.5	16
`(gmaxf.dat)`	5.5	16
`(hystog.dat)`	5.6.4	22
`(plzshape.dat)`	??	??
`(??*****.det)`	5.6.6	17
`(u*****.det)`	5.6.6	??
`(v*****.det)`	5.6.6	??
`(xi*****.det)`	5.6.6	24

File name	Section	Page
`(captured.pls)`	5.5	16
`(s*****.pls)`	5.6.6	23
`(*****.pls)`	5.6.5	23
`(??*****m.swp)`	5.6.1	17
`(??*****w.swp)`	5.6.1	17
`(d?*****.swp)`	5.6.4	23
`(fl*****.swp)`	5.7	29
`(partic.swp)`	5.5	17
`(pl*****.swp)`	5.7	28
`(tb*****.swp)`	5.7	28
`(xi_pre_*****.swp)`	5.6.2	??

7.9 Temporary files

Files `(aver.swp)`, `(digcol.swp)`, `(digcol2.swp)`, `(bfile.swp)`, `(info.swp)`, `(??*****.swp)` are temporary files and are always deleted in normal regimes

8 Tools built into LCODE

Working with files in beamfile.bin format (see 7.1) is complicated due to the fact that the format is binary-based. Frequent tasks like joining two beamfiles, for example to inject probe particles into the beam, are also complicated by this fact. When analyzing the result it's often desirable to work with plain-text decimal data. Though most of the general-purpose programming languages and most of the mathemathical software supports working with such array-like files, LCODE allows performing some of the most common operations without the need of external tools.

8.1 Beamfile composition/manipulation

beamfile-compose, *string* (""): Create a beamfile from other beamfiles

In case LCODE was executed with this option set to a filename, it doesn't enter the regular calculations mode. It creates a file with a specified filename by concatenating files, which names were specified in **beamfile-compose-from** and optionally sorting the result. The terminating stub particles of the input files do not make it into the resulting file and the new file is terminated with a newly created one.

Usage example:

```
lcode.exe --beamfile-compose=newbm.bin --beamfile-compose-from=bm1.bin,bm2.bin
--beamfile-compose-sort=x
```

Create a file `(newbm.bin)` by combining `(bm1.bin)` with `(bm2.bin)` and sorting the particles (Windows). It's advised to inspect the resulting beamfile (see section 8.2) at all times.

beamfile-compose-from, *string* (""): Input files for beam concatenation

Comma-separated file names for beamfile concatenation.

beamfile-compose-sort, *choice* (n): Sorting mode for beamfile composition

'n' or 'none': No sorting is performed, simple concatenation

'x' or 'xi': Particles are sorted by their ξ coordinate.

It's advised to inspect the resulting beamfile (see section 8.2).

beamfile-compose-filter, *choice* (a): Filtering to apply to beamfile composition

'a' or 'all': No filtering is performed, all particles get written to the resulting file

'p' or 'positive': Only positively charged particles get written to the resulting file

'n' or 'negative': Only negatively charged particles get written to the resulting file

beamfile-compose-stub, *choice* (y): Write stub particle

'y' or 'yes': The resulting file gets terminated with a stub particle

'n' or 'no': The resulting file does not get terminated with a stub particle

8.2 Beamfile inspection

beamfile-inspect, *string* (""): Allows inspecting files in beamfile.bin format

In case LCODE was executed with this option set to a filename, it doesn't enter the regular calculations mode. This option specifies the file name of the file use for beamfile inspection.

Usage examples:

`lcode.exe --beamfile-inspect=beamfile.bin`

Interactively browse the contents of the `(beamfile.bin)` (Windows).

`./lcode --beamfile-inspect=beamfile.bin --beamfile-inspect-mode=d > beamfile.dat`

Convert the entire `(beamfile.bin)` into decimal form and save the result into a file `(beamfile.dat)` (UNIX).

beamfile-inspect-mode, *choice* (n): Sorting mode for beamfile composition:

'i' or 'interactive': Interactive browsing of the beamfile contents. A pager-like program is started that displays decimal data from the file specified and allows scrolling the output buffer. Please refer to the interactive help accessible by pressing the 'h' button for instructions about using the interactive mode.

Note that this mode outputs only up to three significant digits for a nicer preview! For higher precision use the 'dump' mode or read the binary data directly from the beamfile.

'd' or 'dump': Dumps the beamfile contents in decimal form.

Note that converting the data into decimal form infers a slight precision loss!

's' or 'stats': Outputs beamfile statistics. Outputs the amount of the partices contained in the beamfile, the presence and location of the terminated stub particle, the amount of positively and negatively charged particles, the ξ-range, and other facts about the file.