# Warp examples

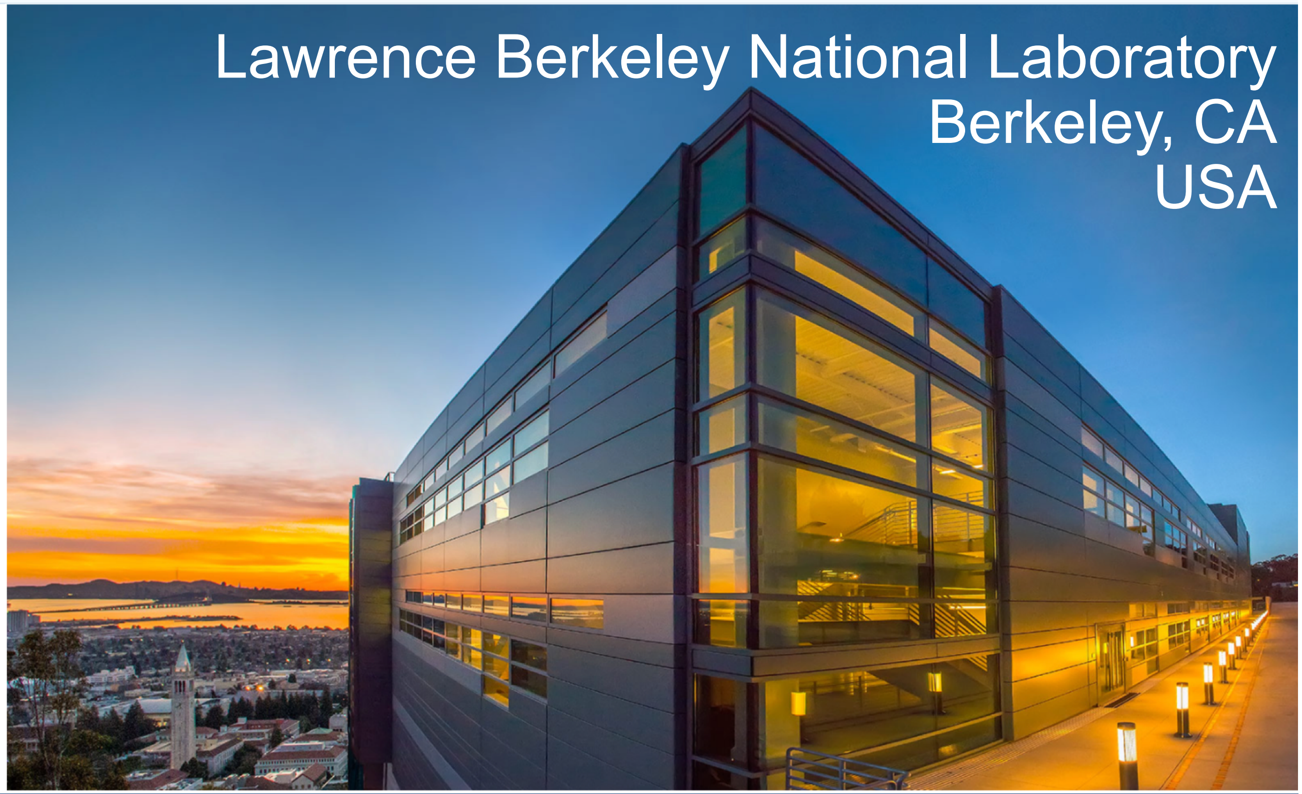
May 7-9, 2018

1st

BLAST
BERKELEY LAB ACCELERATOR SIMULATION TOOLKIT

workshop

Lawrence Berkeley National Laboratory
Berkeley, CA
USA



# Examples

- Pierce diode
- Solenoid transport
- Quadrupole transport 3D
- Quadrupole transport 2D-XY
- Laser injection & propagation in vacuum/dielectric

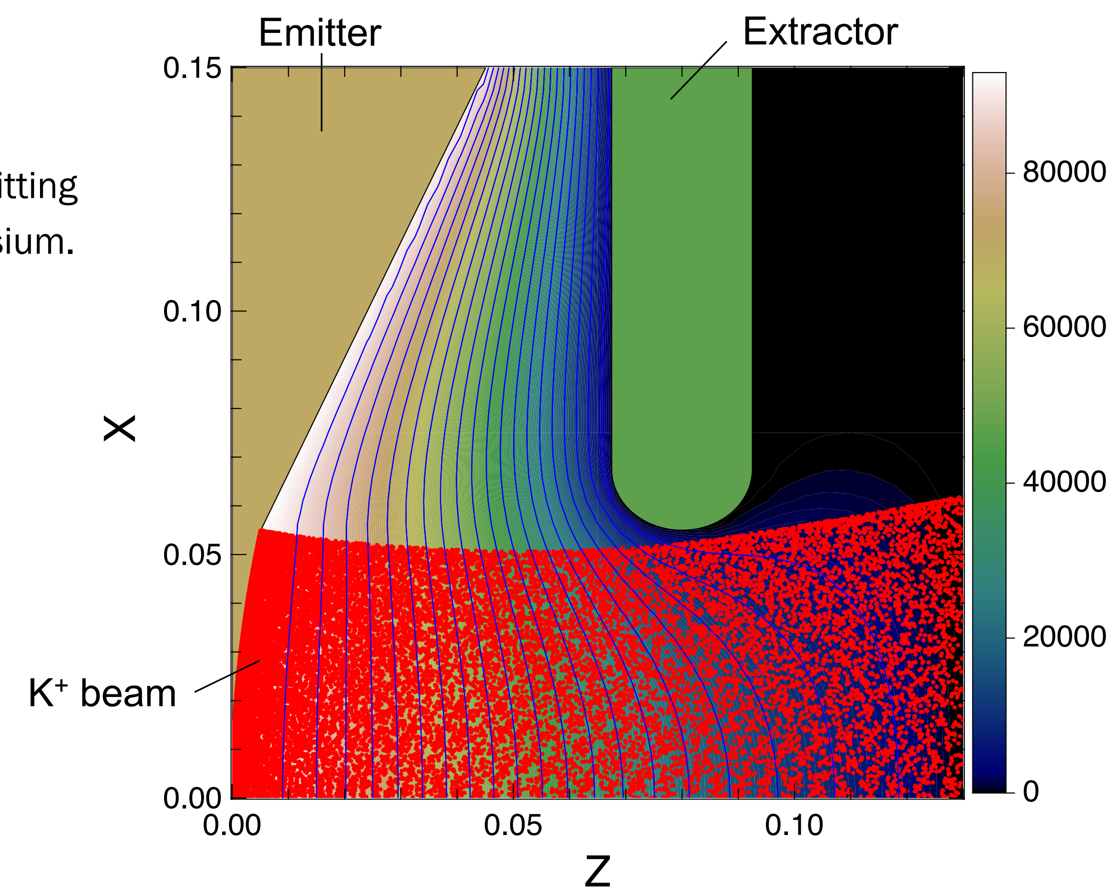
Examples are in warp_fbpic_hands_on/examples from github repository :

- git clone https://github.com/RemiLehe/warp_fbpic_hands_on.git.

# Pierce diode: intro

File Pierce_diode.py

Hot plate source emitting singly ionized potassium.



# Pierce diode: step-by-step

① In Pierce_diode, open file Pierce_diode.py and execute: “python –i Pierce_diode.py”

② Open cgm files and explore:

   a) “gist Pierce_diode.000.cgm &”

   b) “gist current.cgm &”

③ Read input script and try to understand every command

④ Comment “w3d.solvergeom = w3d.rzgeom”, uncomment “w3d.solvergeom = w3d.xyzgeom” and rerun; observe longer runtime but similar result

⑤ Reverse to RZ geometry

⑥ Set “steady_state_gun=True” and rerun. Simulation is now generating traces, converging to steady-state solutions faster than with time-dependent mode.

⑦ Set “w3d.l_inj_regular = True”, “top.npinject    = 15” and rerun with regularly spaced traces. This option can be used to enable faster simulations.

⑧ Change “diode_current = pi*source_radius**2*j” to “0.5*pi*source_radius**2*j”, then “2*pi*source_radius**2*j” and rerun each time. What do you observe?

# Pierce diode: step-by-step

⑨ Go back to original settings

- steady_state_gun=False
- diode_current = pi*source_radius**2*j
- (optional) w3d.l_inj_regular = False and top.npinject = 150

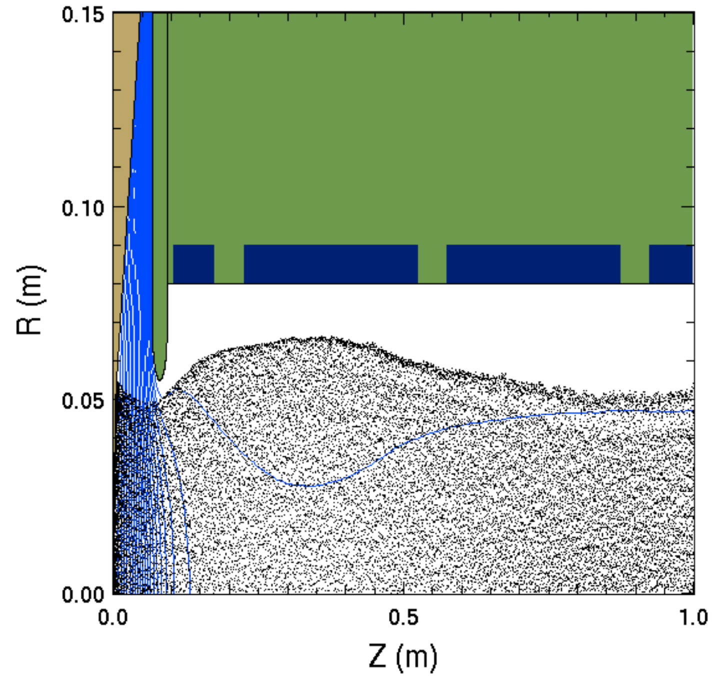
then change

- top.inject=1 ➔ top.inject=2 so that extracted current is automatically set at the Child-Langmuir limit, for a given voltage drop.

Rerun. Open the latest cgm file, page through and observe how the head of the beam has a larger current and touches the extractor. Can you explain why?

⑩ Set "l_constant_current = True" and rerun, observing how the injected current is now constant. Also observe the history of the applied voltage versus time.

# Solenoid transport



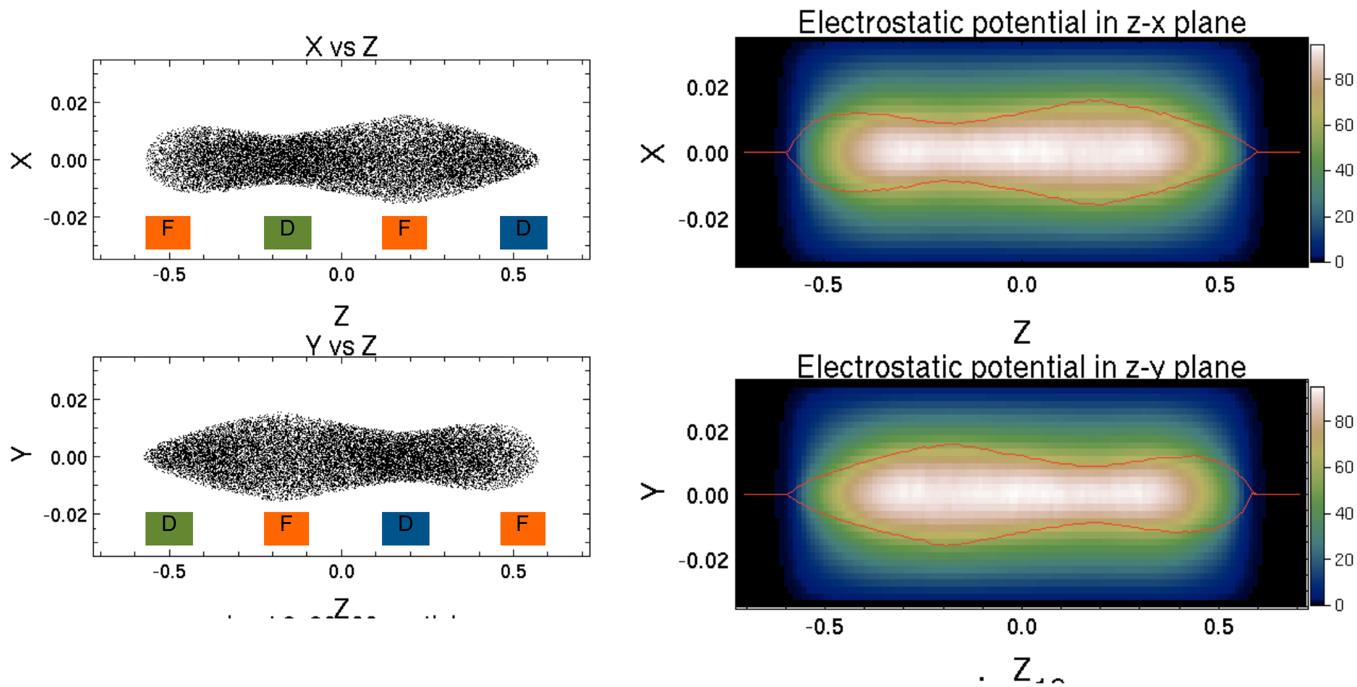

File Solenoid_transport.py:

- Example Pierce diode with subsequent solenoid transport.
- Hot plate source emitting singly ionized potassium.

# Solenoid transport: step-by-step

① In Solenoid_transport, open Solenoid_transport.py
② Execute file in interactive mode: “python –i Solenoid_transport.py”
③ Open cgm file and explore:
   a) “gist Solenoid_transport.000.cgm &”
④ Read input script and try to understand every command.
⑤ Change “l_solenoid = False” to “l_solenoid = True”. Rerun.
⑥ Select ‘window(1)’.
⑦ Type “fma()” to start next plot from empty page.
⑧ Type “rzplot(9)” to plot RZ view of beam, pipe and solenoids in upper half.
⑨ Type “ppzvtheta(view=10)” to plot particle projections of azimuthal velocity versus z.
⑩ Notice the correlations between the extremas of the azimuthal velocity and the positions of the solenoids.
⑪ Here again, faster simulations can be performed by setting “w3d.l_inj_regular = True”, “top.npinject = 15”.

# Quadrupole transport – 3D




File FODO3D.py - basic 3D simulation of an ion beam in a periodic FODO lattice:

- Sets up a periodic FODO lattice and creates a beam that is matched to the lattice.
- The beam is propagated one lattice period.

# FODO3D: step-by-step

① Into FODO3D, open FODO3D.py

② Execute file: "python –i FODO3D.py"

③ Open cgm file and explore:

  a) "gist FODO3D.000.cgm &"

④ Compare the slices emittance diagnostics to the whole emittance diagnostic. See how one is constant, the other oscillates.

⑤ Read input script and try to understand every command

⑥ Change 'w3d.distrbtn = "semigaus"' to 'w3d.distrbtn = "KV"' ; rerun & observe

⑦ Change 'w3d.distr_l = "gaussian"' to 'w3d.distr_l = "neuffer"' ; rerun & observe

⑧ Insert "beam.x0 = beam.a0/2" on the line following "beam.a0 = …"; rerun & observe

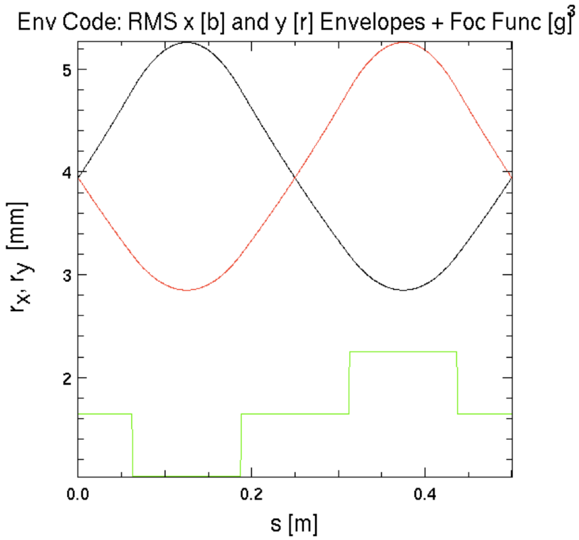
⑨ Check that you have the "ffmpeg" software installed: "which ffmpeg"

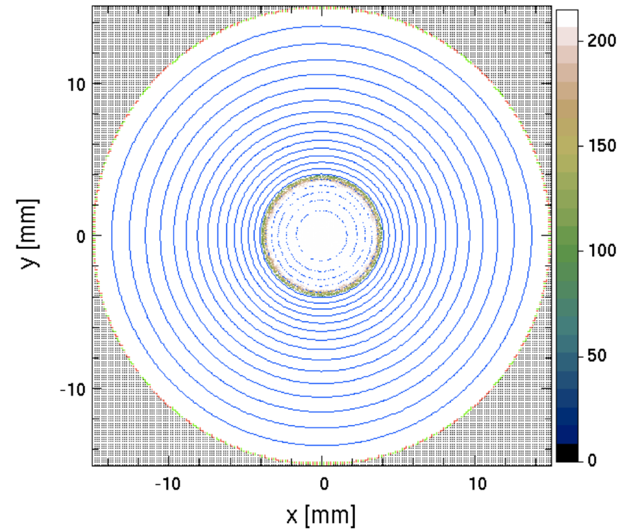
  - If not, download and install ffmpeg

⑩ Change "l_movieplot = False" to "l_movieplot = True" & rerun

  - If all goes well, after a few minutes, you should have a movie "movie.mp4"

# Quadrupole transport – XY




File xy-quad-mag-mg.py:

nonrelativistic Warp xy slice simulation of a $K^+$ ion beam with intense space-charge focused by a hard-edge magnetic quadrupole doublet focusing lattice.

# xy-quad-mag-mg: tasks

① Into XY-quad, open xy-quad-mag-mg.py
② Execute file: "python –i xy-quad-mag-mg.py"
③ Open cgm file and explore: "gist xy-quad-mag-mg.000.cgm &"
④ Read input script and try to understand every command.
⑤ Comment 'w3d.distrbtn = "SG"' and uncomment 'w3d.distrbtn = "KV"', rerun and compare to results using the KV vs SG distributions.
⑥ Change the initial emittance "emit = 10.e-6" to "emit = 10.e-7", rerun and observe effect on matching and emittance preservation.
⑦ Change switch "l_automatch = False" to "l_automatch = True", rerun and observe difference with previous run.
⑧ Change n_grid=200 to 400, rerun and observe differences.
⑨ With the simulation back at the python prompt, type 'dump()', then run for another 500 steps: "step(500)".
⑩ In another terminal, start python and type:

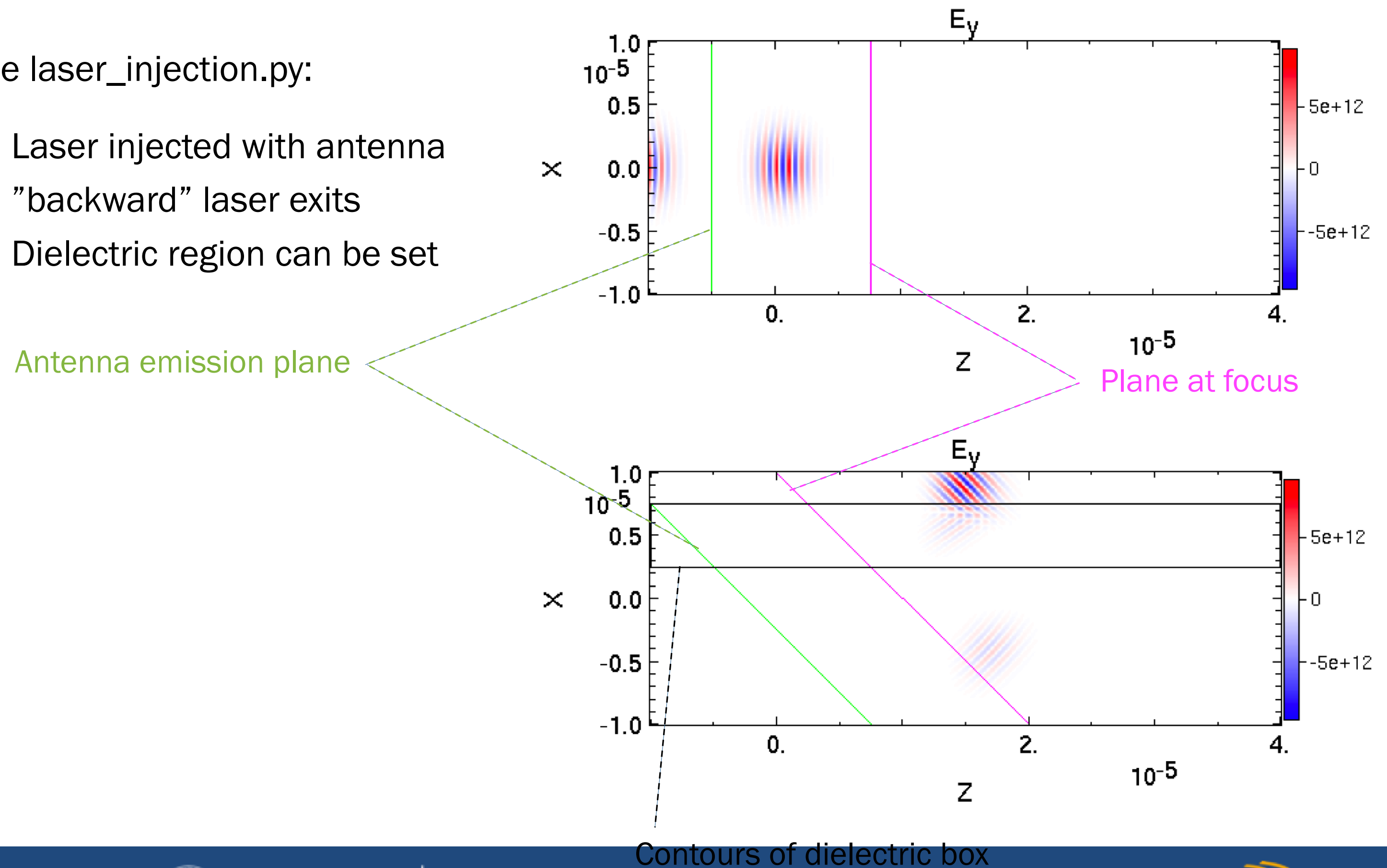
- from warp import *
- restart('xy-quad-mag-mg001000.dump')
- step(500)

Reopen "xy-quad-mag-mg.000.cgm" and compare to "xy-quad-mag-mg.001.cgm".

# Laser injection & propagation in vacuum/dielectric

File laser_injection.py:

- Laser injected with antenna
- ”backward” laser exits
- Dielectric region can be set



# Laser injection: step-by-step

① In Laser_injection, open Laser_injection.py

② Execute file in interactive mode: "python –i Laser_injection.py"

③ Read input script and try to understand every command

④ Check at beginning of scripts how to add optional arguments and their definitions

⑤ Rerun with a longer wavelength:

- python -i laser_injection.py –ll 2.e-6

⑥ Rerun with the laser impinging a dielectric at an angle of 45 degree:

- python -i laser_injection.py -lv '[1.,0.,1.]' -lp '[-5.e-6,0.,2.5e-6]' -er 1.5

⑦ Rerun with the laser born inside the dielectric:

- python -i laser_injection.py -lv '[1.,0.,1.]' -bp '[-5.e-6,5.e-6]' -er 1.5

⑧ Reducing the angle of incidence:

- python -i laser_injection.py -lv '[1.,0.,2.]' -bp '[-5.e-6,5.e-6]' -er 1.5
- python -i laser_injection.py -lv '[1.,0.,3.]' -bp '[-5.e-6,5.e-6]' -er 1.5
- Note: you may propagate the laser further with 'step(200)'
- Observe the total reflection with the latest run. What happens with -ll 2.e-6?

⑨ In the script, change laser_source_v from 0. to 0.5*clight and run

- python -i laser_injection.py (observe the Doppler effect)