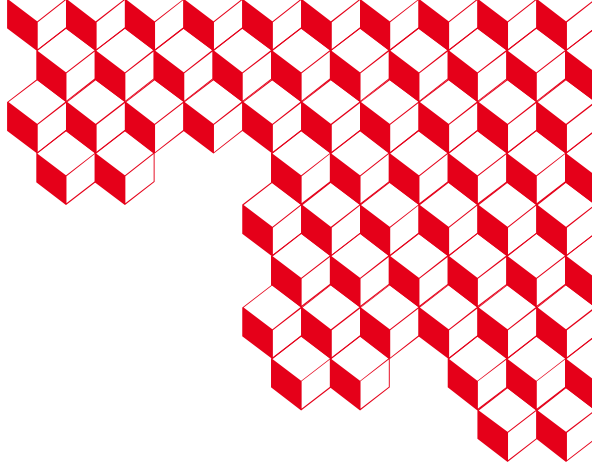


6th EARLI Project Meeting

Damien Minenna









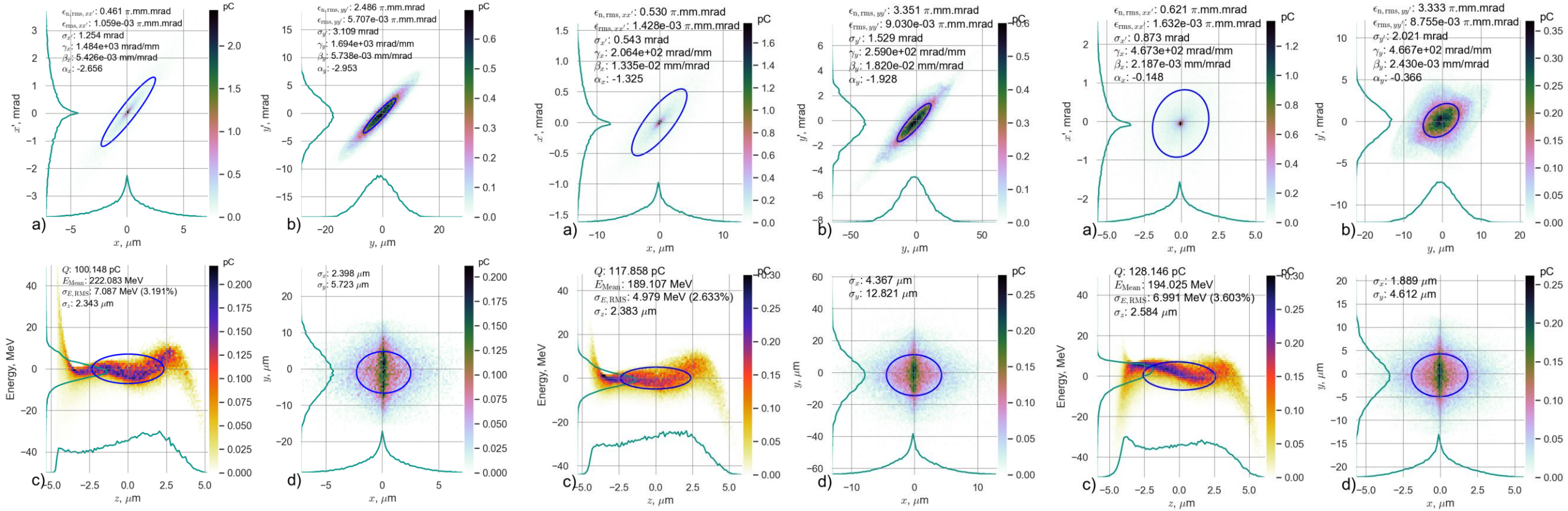
MAX-PLANCK-INSTITUT
FÜR PHYSIK

New recruitment

Internship

- **Laury Batista**
  - Master2 GI-PLATO | JUAS
  - Former internships on LWFA physics and beam dynamics (TraceWin)
- Transfer line design for LWFA
  - Design of a variety of TF using LPA beam
  - Examples
    - 4 quads
    - 6 quads
    - 9 quads + 2 dipoles
    - With various focalisation constraints

Best cases


$\epsilon_{n,rms,xx'}$: 0.461 π.mm.mrad
$\epsilon_{rms,xx'}$: 1.059e-03 π.mm.mrad
$\sigma_{x'}$: 1.254 mrad
γ_x: 1.484e+03 mrad/mm
β_x: 5.426e-03 mm/mrad
α_x: -2.656
$\epsilon_{n,rms,yy'}$: 2.486 π.mm.mrad
$\epsilon_{rms,yy'}$: 5.707e-03 π.mm.mrad
$\sigma_{y'}$: 3.109 mrad
γ_y: 1.694e+03 mrad/mm
β_y: 5.738e-03 mm/mrad
α_y: -2.953
$\epsilon_{n,rms,xx'}$: 0.530 π.mm.mrad
$\epsilon_{rms,xx'}$: 1.428e-03 π.mm.mrad
$\sigma_{x'}$: 0.543 mrad
γ_x: 2.064e+02 mrad/mm
β_x: 1.335e-02 mm/mrad
α_x: -1.325
$\epsilon_{n,rms,yy'}$: 3.351 π.mm.mrad
$\epsilon_{rms,yy'}$: 9.030e-03 π.mm.mrad
$\sigma_{y'}$: 1.529 mrad
γ_y: 2.590e+02 mrad/mm
β_y: 1.820e-02 mm/mrad
α_y: -1.928
$\epsilon_{n,rms,xx'}$: 0.621 π.mm.mrad
$\epsilon_{rms,xx'}$: 1.632e-03 π.mm.mrad
$\sigma_{x'}$: 0.873 mrad
γ_x: 4.673e+02 mrad/mm
β_x: 2.187e-03 mm/mrad
α_x: -0.148
$\epsilon_{n,rms,yy'}$: 3.333 π.mm.mrad
$\epsilon_{rms,yy'}$: 8.755e-03 π.mm.mrad
$\sigma_{y'}$: 2.021 mrad
γ_y: 4.667e+02 mrad/mm
β_y: 2.430e-03 mm/mrad
α_y: -0.366
Q: 100.148 pC
E_{Mean}: 222.083 MeV
$\sigma_{E,RMS}$: 7.087 MeV (3.191%)
σ_z: 2.343 μm
σ_x: 2.398 μm
σ_y: 5.723 μm
Q: 117.858 pC
E_{Mean}: 189.107 MeV
$\sigma_{E,RMS}$: 4.979 MeV (2.633%)
σ_z: 2.383 μm
σ_x: 4.367 μm
σ_y: 12.821 μm
Q: 128.146 pC
E_{Mean}: 194.025 MeV
$\sigma_{E,RMS}$: 6.991 MeV (3.603%)
σ_z: 2.584 μm
σ_x: 1.889 μm
σ_y: 4.612 μm
a)
b)
c)
d)
pC
x', mrad
y', mrad
Energy, MeV
x, μm
y, μm
z, μm

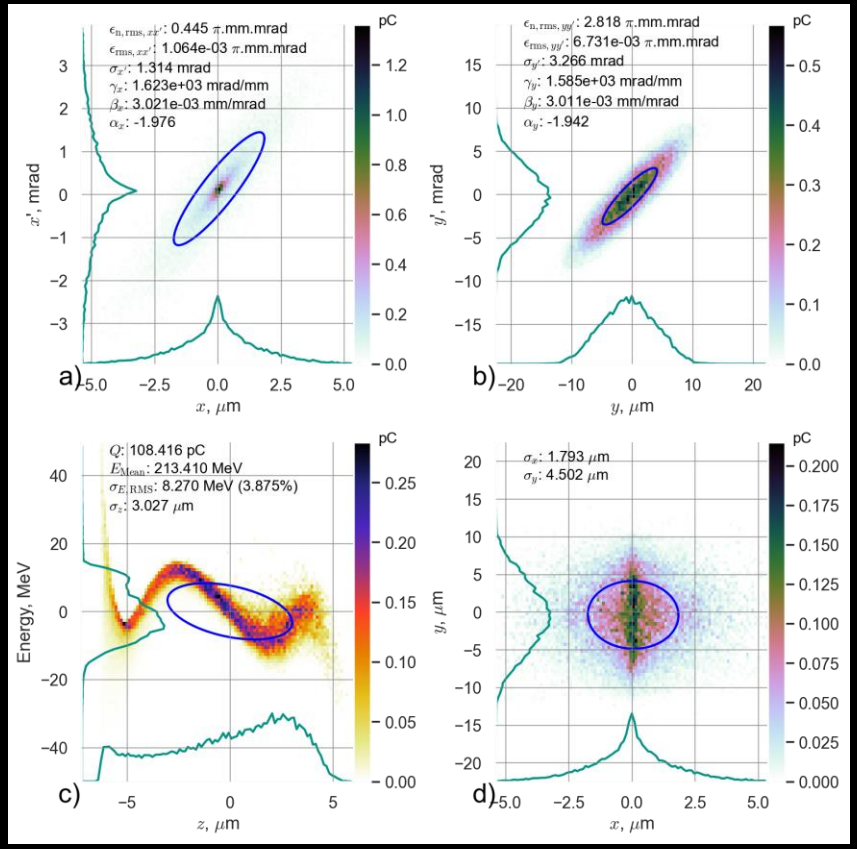
New numerical tool for LWFA design

Python package

- For beam dynamics and data analysis in laser-plasma simulations
- Interface between LPA and beam transport codes
  - Already take into account: **FBPIC, SMILEI, TraceWin**
  - Will take into account: **WarpX, MadX, ASTRA, …**
- Beam dynamics data analysis in plasmas
- Automatic launcher for optimisation runs (Genetic algorithm, Bayesian optimisation)
- Easy to install
  - Multi OS (tested on Windows | Mac | Linux)
  - Can be installed on a cluster (tested on TGCC IRENE and TOPAZE)
  - Very few dependencies (numpy | pandas | scipy required; matplotlib optional)
- Easy to use
  - GUI discussed
- Distribution
  - Open source discussed
  - Documentation ongoing

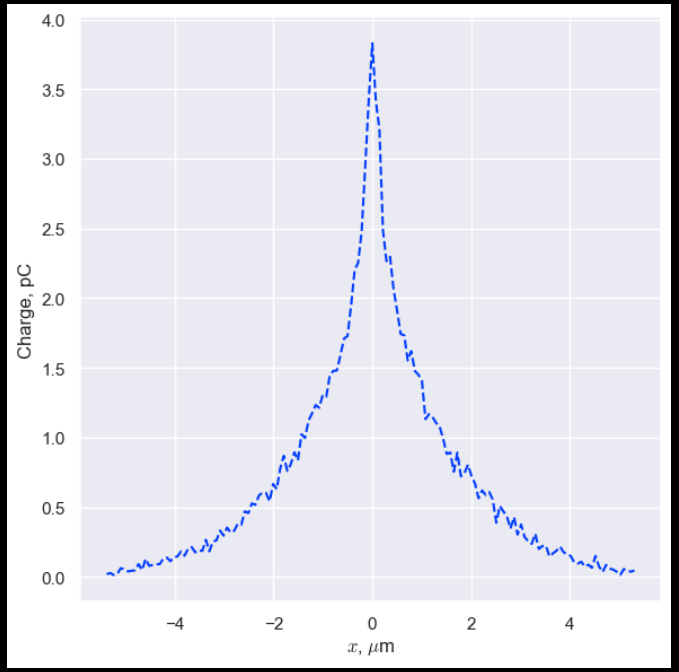
Example of uses

```
# skypiea
import skypiea
# Find all timesteps
steps = skypiea.Steps(directory='/…/…/',source='fbpic')
# Creation of the step class
step = skypiea.Step()
# Read data
step = steps.getParticle(step,timestep,species='elec')
# plot beam (6D beam)
_ = skypiea.plotbeam(step)
```



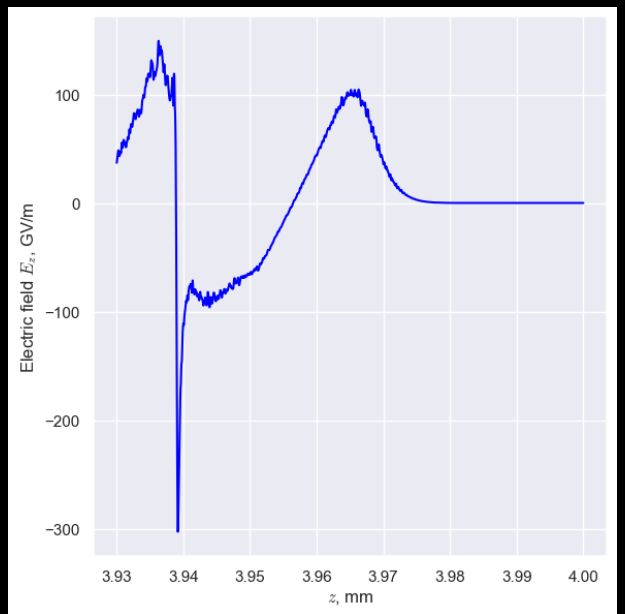
Example of uses

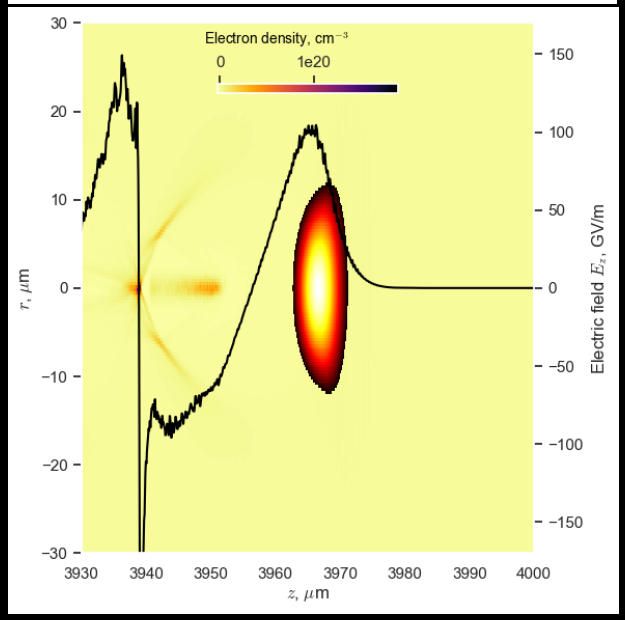
```
# Simple plot of the x distribution (dQ/dx). Note that x axis is convert in um
with xconv. step.sigma_x is the standard deviation (bunch length) in x.

H, xedges = step.hist1D('x',
                        xconv='um',
                        xrange=[-3*step.sigma_x*1e6,3*step.sigma_x*1e6],
                        bins=150,
                        # dx = 1, # Instead of bins
                        plot='plot',
                        # Keyword arguments
                        linestyle='--'
                        )
```

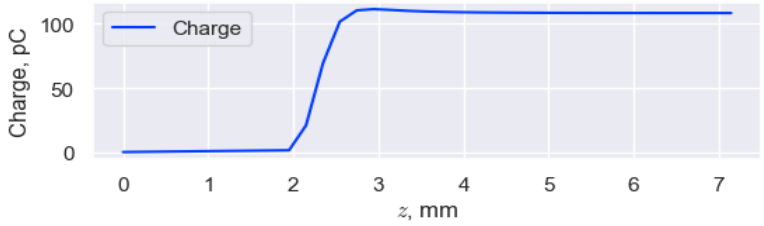


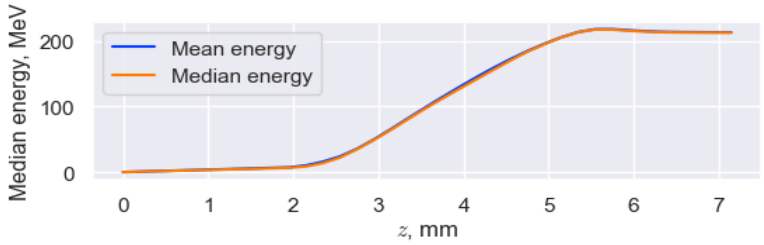
Example of uses

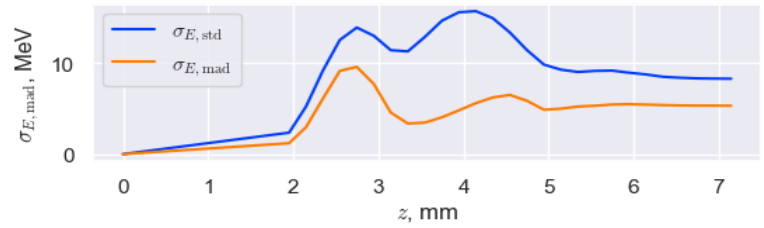
```
step = steps.getField(step,timestep)
step.plot1Dfield('zfield','Ez',xconv='mm',yconv='GV/m')
step.plot2Dfield('zfield','rfield','ne',
                 xconv='um',
                 yconv='um',
                 zconv='cm-3',
                 yrange=[-30,30],
                 cmap=cm.inferno_r,
                 ylabel=True,
                 Ez=True,
                 Ez_conv="GV/m",
                 Ez_range=[-170,170],
                 Ez_color="black",
                 Ez_ylabel=True,
                 Ey2d_env=True,
                 grid=False,
                 iscbar="cbar_reduced_b")
```

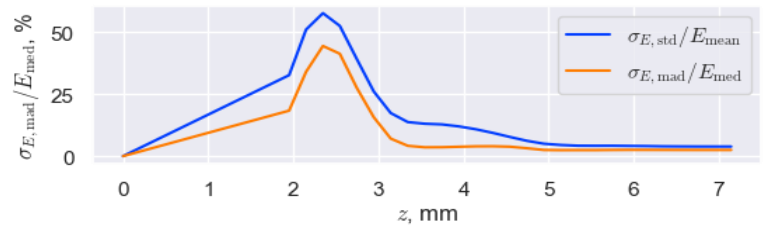


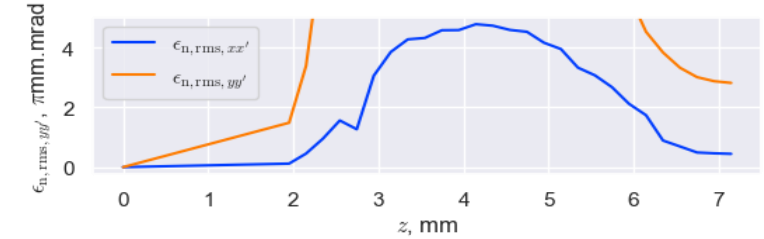


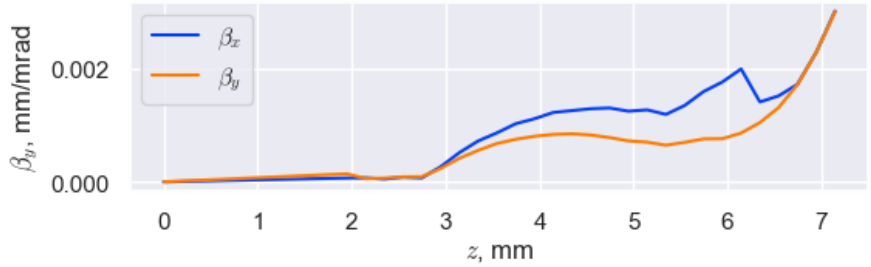

Charge, pC
Charge
z, mm

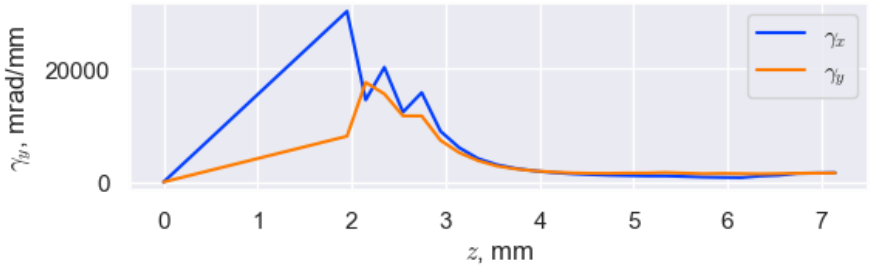

Median energy, MeV
Mean energy
Median energy
z, mm

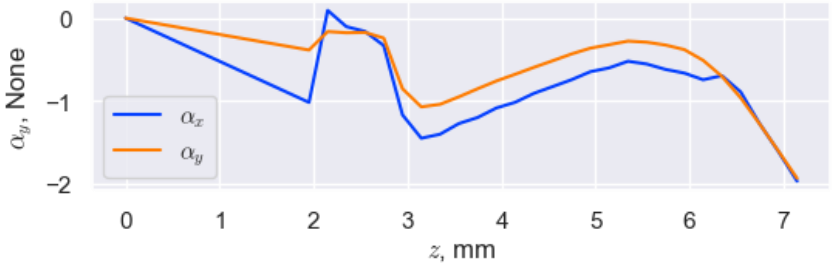

$\sigma_{E,\mathrm{mad}}$, MeV
$\sigma_{E,\mathrm{std}}$
$\sigma_{E,\mathrm{mad}}$
z, mm

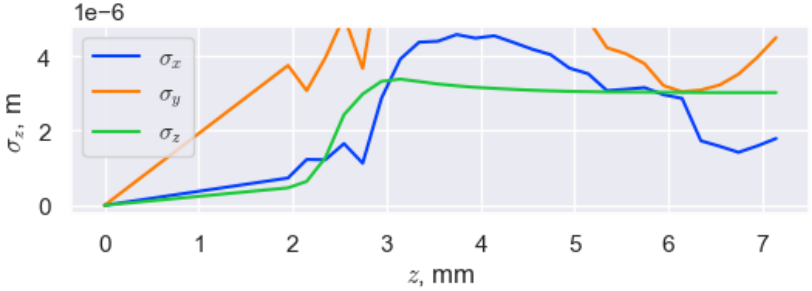

$\sigma_{E,\mathrm{mad}}/E_{\mathrm{med}}$, %
$\sigma_{E,\mathrm{std}}/E_{\mathrm{mean}}$
$\sigma_{E,\mathrm{mad}}/E_{\mathrm{med}}$
z, mm

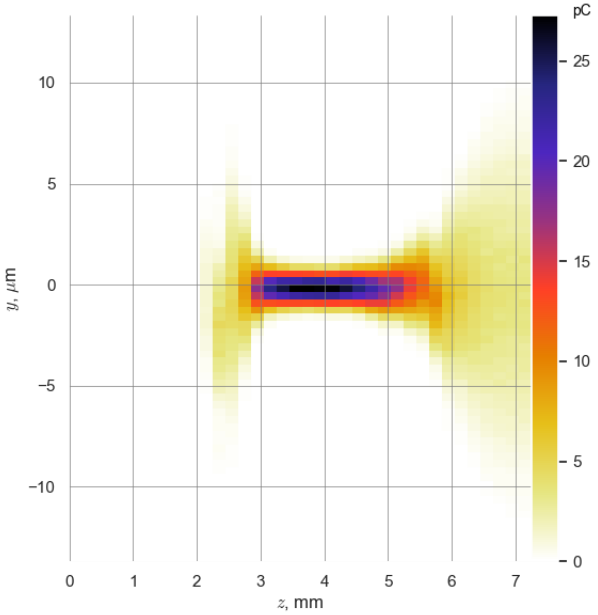

$\epsilon_{\mathrm{n,rms},yy'}$, πmm.mrad
$\epsilon_{\mathrm{n,rms},xx'}$
$\epsilon_{\mathrm{n,rms},yy'}$
z, mm

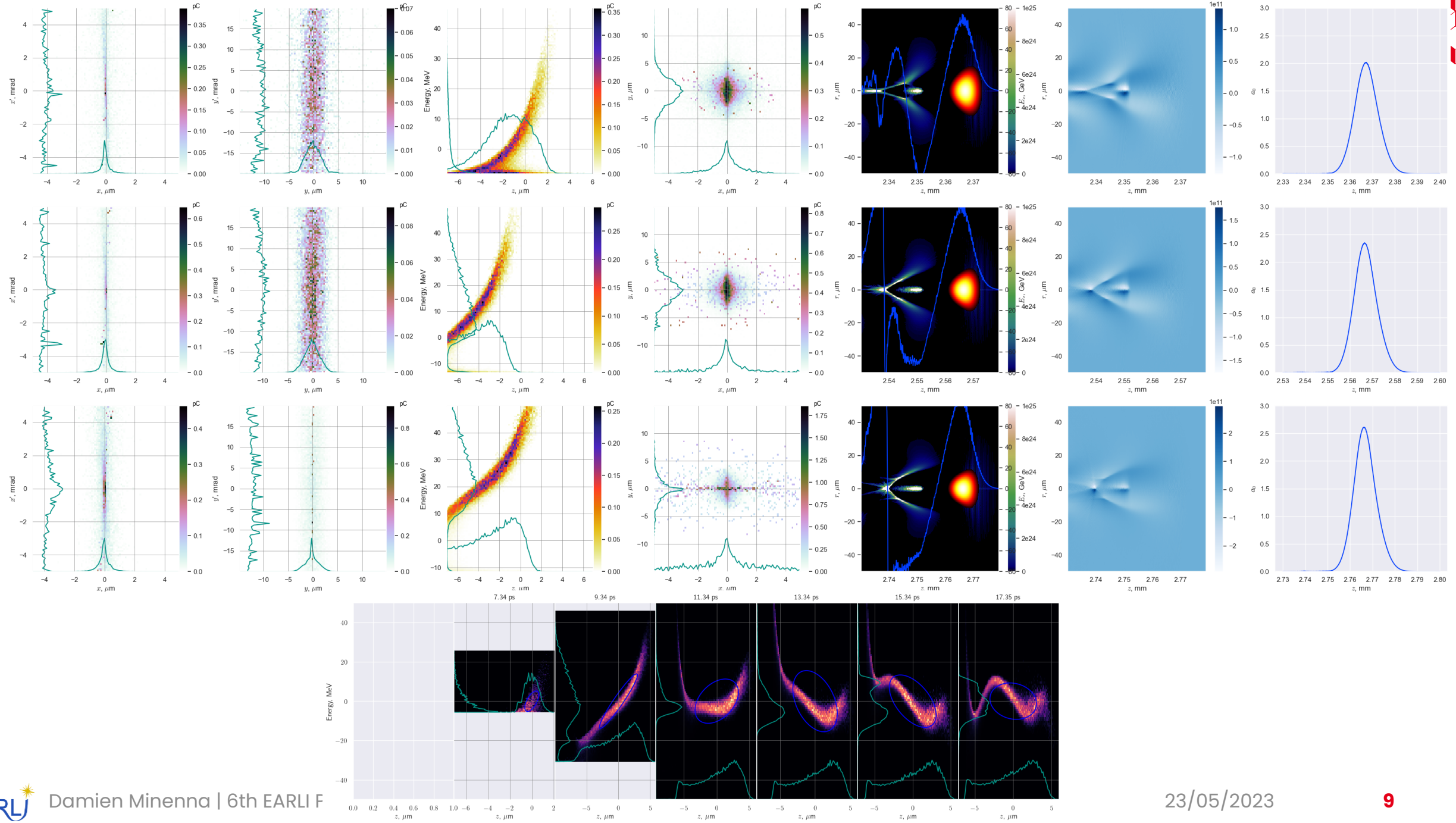

β_y, mm/mrad
β_x
β_y
z, mm


γ_y, mrad/mm
γ_x
γ_y
z, mm


α_y, None
α_x
α_y
z, mm

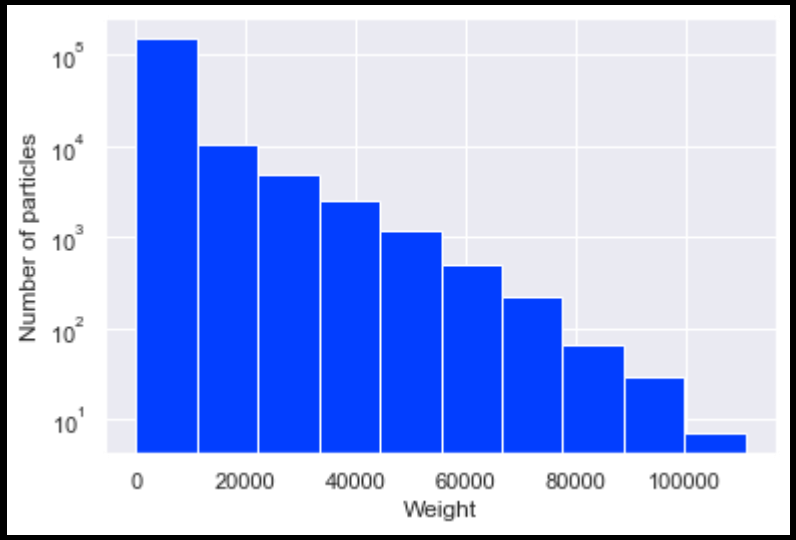

1e−6
σ_z, m
σ_x
σ_y
σ_z
z, mm

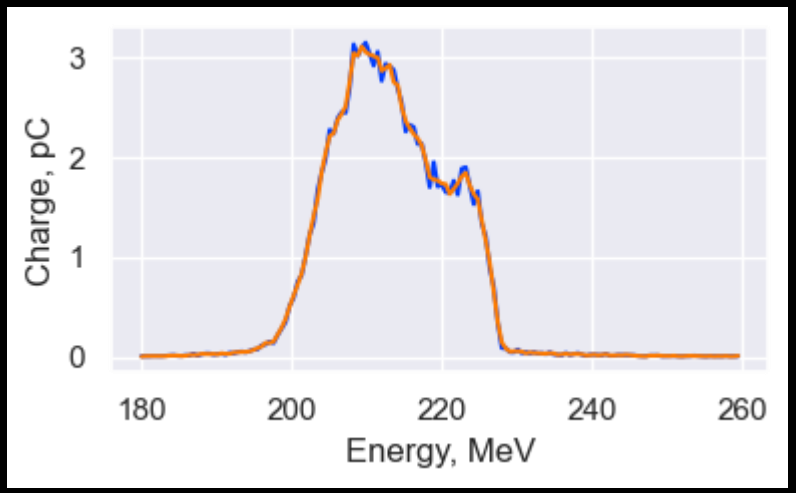

y, μm
pC
z, mm


x', mrad
x, μm
pC
y', mrad
y, μm
Energy, MeV
z, μm
y, μm
x, μm
r, μm
z, mm
E_z, GeV
1e25
1e11
a_0
7.34 ps
9.34 ps
11.34 ps
13.34 ps
15.34 ps
17.35 ps

To TraceWin

```
# Read dst
step = skypiea.read_dst("data.dst")

# Rewrite distribution with same weight
step_new = skypiea.noweight(step,N=100000)

# Write dst
step_new.writeTraceWin("tracewin.dst",freq)
```





Launcher

Launcher

- Generate automatically runs
  - Simulation scripts
  - Execution scripts
  - Analysis scripts
- Replacer
  - Ex: {{a0}} -> generated value
- Save everything
  - Input and output data
  - Scripts
  - Format csv
  - Toward ML approach

Genetic algorithm

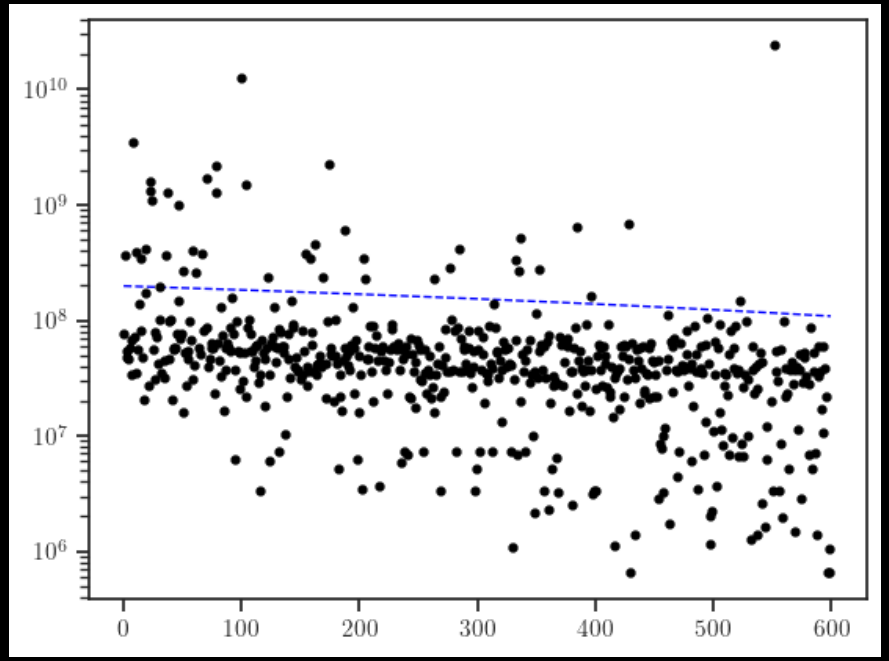
- More than 1000 runs with FBPIC

```
# SLURM script run.sh

#MSUB -n {{n_core}}
ccc_mprun python3 {{scriptpath}}

# -------
# Script FBPIC.py

a0 = {{a0}}
w0 = {{w0}}
```

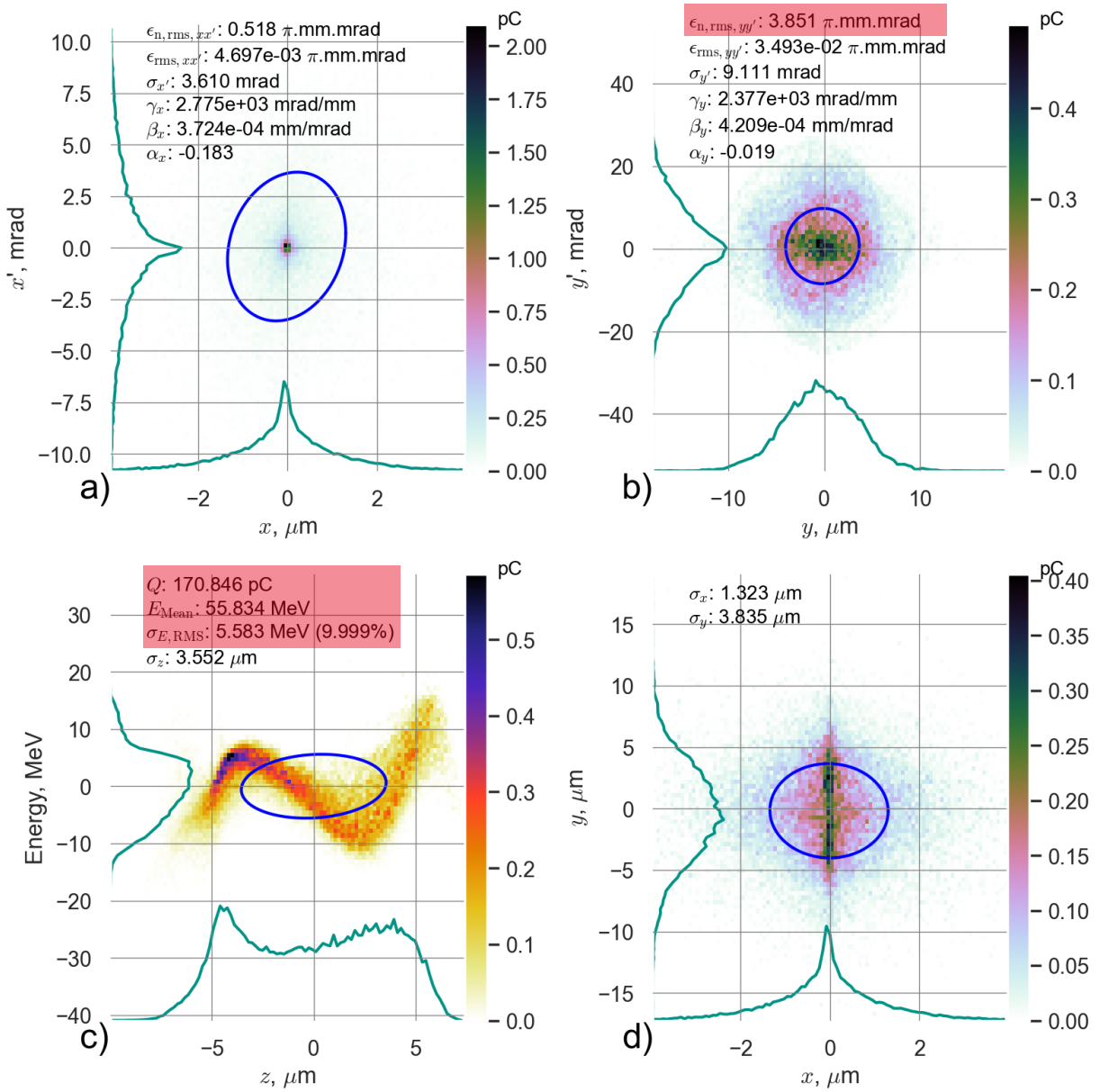
```
# SLURM script run.sh

#MSUB -n 8
ccc_mprun python3 /…/Run_00000958/script.py


# -------
# Script FBPIC.py

a0 = 1.3003218648997583
w0 = 2.095857887419 9e-05
```

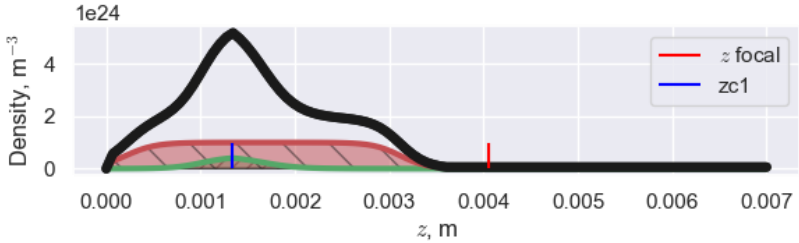
Genetic algorithm

```
INPUTS = [['a0'  , 1., 2.],
          ['w0'  , 12*um, 30.0*um],
          ['taua', 20*fs, 30*fs],
          ['zfoc', 2000*um, 4500*um],
          ['zc1' , 1000*um, 2200*um],
          ['mu1' , 300*um, 600*um],
          ['T11' , 60*um, 160*um],
          ['T12' , 60*um, 160*um],
          ['zc2' , 3000*um, 4000*um],
          ['mu2' , 800*um, 1600*um],
          ['T21' , 60*um, 160*um],
          ['T22' , 60*um, 160*um],
          ['amp1' , 2, 4],
          ['amp2' , 1.5, 2.5],
          ['ampn' , 0.01, 0.39],
          ['ampt' , 0.0001, 0.06]]

OUTPUTS = [['charge', 120., 50, 'or more'],['energy_std_perc', 2., 100, 'or
less'],['emit_norm_rms_y', 2., 50, 'or less']]
```



Best output with the genetic algorithm





Example of uses

```
angle = 45 * np.pi / 180 # Radian
step_new = step.rotationXY(angle)
```