LASY — Laser Manipulation Made Easy

Maxence Thévenet [1], Rob Shalloo [1], Igor Andriyash [2], Emily Archer [1], Timo Eichner [1], Luca Fedeli [3], Axel Huebl [4], Xingjian Hui [1], Sören Jalas [1], Manuel Kirchen [1], Remi Lehe [4], Andreas Maier [1], Alberto de la Ossa [1], Parth Patil [1], Kristjan Põder [1], Jean Luc Vay [4], Kale Weichman [5]

[1] Deutsches Elektronen-Synchrotron (DESY), Hamburg, Germany
[2] Laboratoire d'Optique Appliquée LOA, 181 Chemin de la Hunière 91762 Palaiseau, France
[3] Commissariat à l'Énergie Atomique CEA Paris-Saclay, 91191 Gif-sur-Yvette, France
[4] Lawrence Berkeley National Laboratory, 1 Cyclotron Rd, Berkeley, California 94720, USA
[5] Laboratory for Laser Energetics, University of Rochester, Rochester, NY, USA

Access the Code — GitHub

Checkout the Tutorial

`pip install lasy`

Motivation and Overview

`lasy` [1] **is an open source python library that streamlines the initialisation and manipulation of complex laser pulses for simulations of laser-plasma interactions.**

It was developed to address key challenges in laser-plasma simulation workflows:

- ✓ **Realistic laser profiles** are key for realistic simulations of laser-plasma interaction [2].
- ✓ **Laser manipulations** (conversions, propagation, etc.) are required and error-prone.
- ✓ **Start-to-end workflows** require interfacing simulation tools with different laser representations.

`lasy` **simplifies these workflows with modern programming methods:** (Open-source, Python, CI/CD, data standards).

Code Example

imports

physical description of laser

simulation parameters + create object

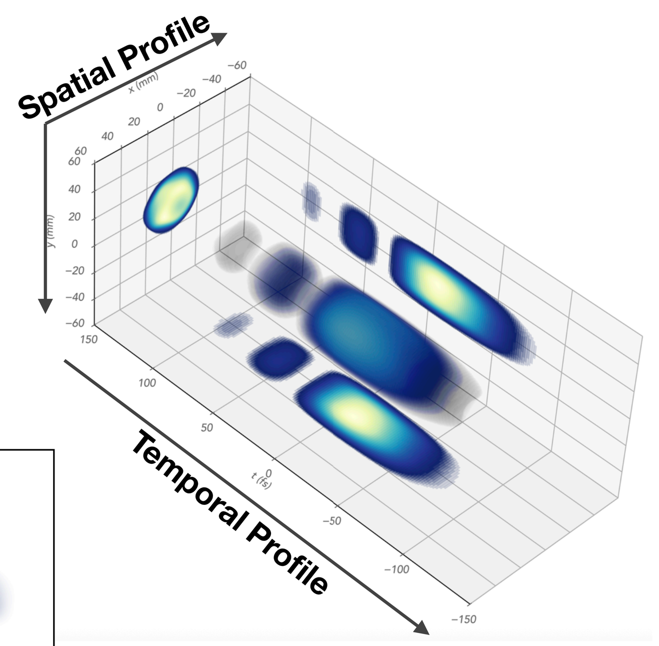
propagate + plot / save

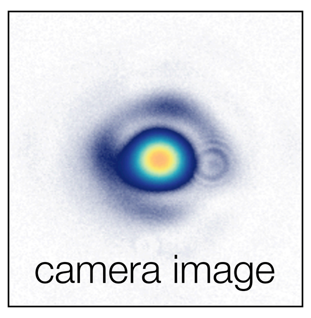
```python
from lasy.laser import Laser
from lasy.profiles.gaussian_profile import GaussianProfile

wavelength     = 800e-9  # Laser wavelength in meters
polarization   = (1, 0)  # Linearly polarized in the x direction
energy         = 1.5     # Energy of the laser pulse in joules
spot_size      = 25e-6   # Waist of the laser pulse in meters
pulse_duration = 30e-15  # Pulse duration of the laser in seconds
t_peak         = 0.0     # Location of the peak of the laser pulse in time

laser_profile = GaussianProfile(wavelength, polarization, energy, spot_size, pulse_duration, t_peak)

dimensions     = "rt"                              # Use cylindrical geometry
lo             = (0, -2.5 * pulse_duration)        # Lower bounds of the simulation box
hi             = (5 * spot_size, 2.5 * pulse_duration) # Upper bounds of the simulation box
num_points     = (300, 500)                        # Number of points in each dimension

laser = Laser(dimensions, lo, hi, num_points, laser_profile)

laser.propagate(-2e-6)                        # Propagate the pulse upstream of the focal plane

laser.show(envelope_type='intensity')         # Plot the intensity profile of the laser pulse
laser.write_to_file('test_laser', 'h5')       # Save the laser to file for use in a simulation
```

Defining a Laser Pulse

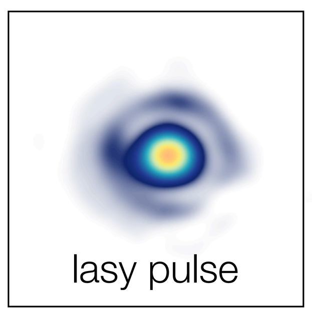
Can be defined as **separable** (in space and time) or as as profiles with **spatiotemporal couplings.**
These laser profiles can be:

- ✓ Analytic profiles
- ✓ Imported from experimental data
- ✓ Imported from another simulation via openPMD




camera image

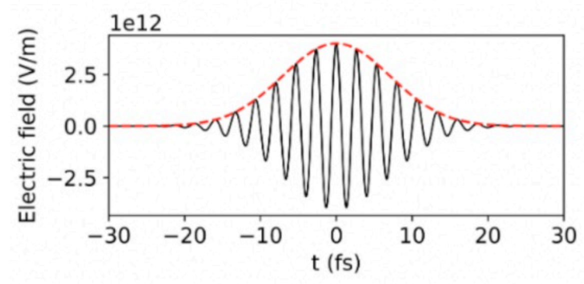
import experimental data + modal reconstruction

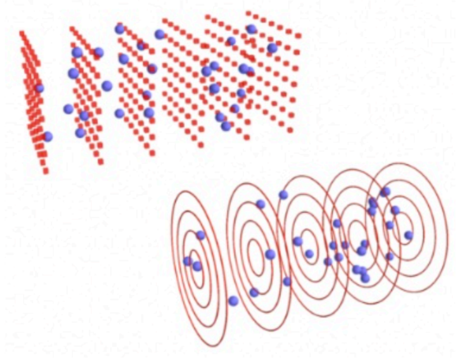

lasy pulse

How is a Laser Pulse Represented?

Simplified model covering broad use case:

- ✓ Temporal envelope approximation (internally)
- ✓ Paraxial approximation
- ✓ Constant polarisation
- ✓ Different geometries:
  - ✓ Cartesian (x, y, t)
  - ✓ Cylindrical with modal decomposition (m, r, t)





$\vec{E}, \vec{B}$: 6 components with fast oscillations

`lasy` : 1 component, no fast oscillations

Adding Optical Elements

```python
laser.apply_optics( ParabolicMirror( f=850e-3 ) )
```

Current elements included:

- ✓ Parabolic mirror
- ✓ Axicon
- ✓ Axiparabola
- ✓ Chromatic lens
- ✓ Intensity mask
- ✓ Dazzler
- ✓ Spectral phase
- ✓ Spectral filter
- ✓ Zernike aberrations

…and easy to add your own!

$$E(\vec{x}, \omega) \rightarrow E(\vec{x}, \omega)\, e^{i\phi(\vec{x}, \omega)}$$

angular frequency; laser envelope; spatial coordinates; phase added by optic

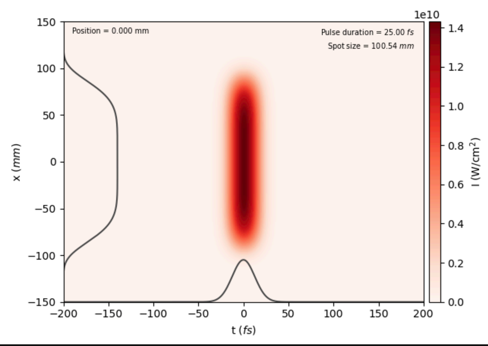
`lasy` also allows direct retrieval of the refractive index of desired materials from refractiveindex.info

Propagating the Laser Pulse

Propagation of beam can be easily simulated:

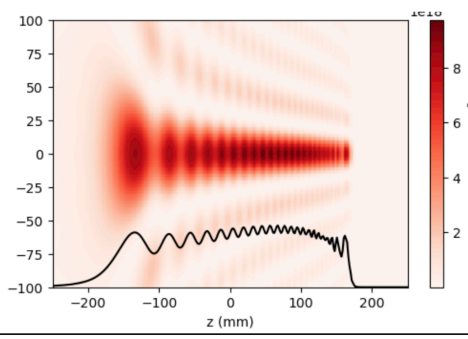
- ✓ Near field propagation (same grid)
- ✓ Far field propagation (resampled grid)
- ✓ Split-step propagation (nonlinear effects)
- ✓ Established libraries such as Axiprop [3,4]

```python
# define a resampled grid
lo_new         = (-100e-6,-100e-6,-80e-15)
hi_new         = (100e-6,100e-6,80e-15)
n_points_new   = (500,500,200)
grid_out       = Grid('xyt', lo_new, hi_new, n_points_new)

# Add propagator and propagate pulse
laser.add_propagator(FresnelChirpZPropagator())
laser.propagate( distance = 500e-3, grid_out=grid_out)
```



add phase + resampling propagator

`FresnelChirpZPropagator`



Save / View a Laser Pulse

```python
laser.write_to_file( file_prefix="kaldera_pulse",
                     file_format="h5",
                     write_dir="20250924_diags",
                     save_as_vector_potential=True)
```

Laser can be written to file following the openPMD [5] standard

Available formats:

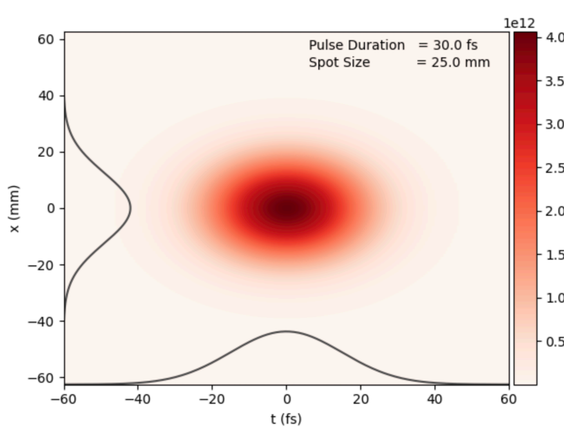
- ✓ HDF5
- ✓ ADIOS2

Envelope can be saved as:

- ✓ Electric field
- ✓ Vector potential

```python
laser.show(envelope_type='intensity')
```

Plotting and probing laser properties also possible

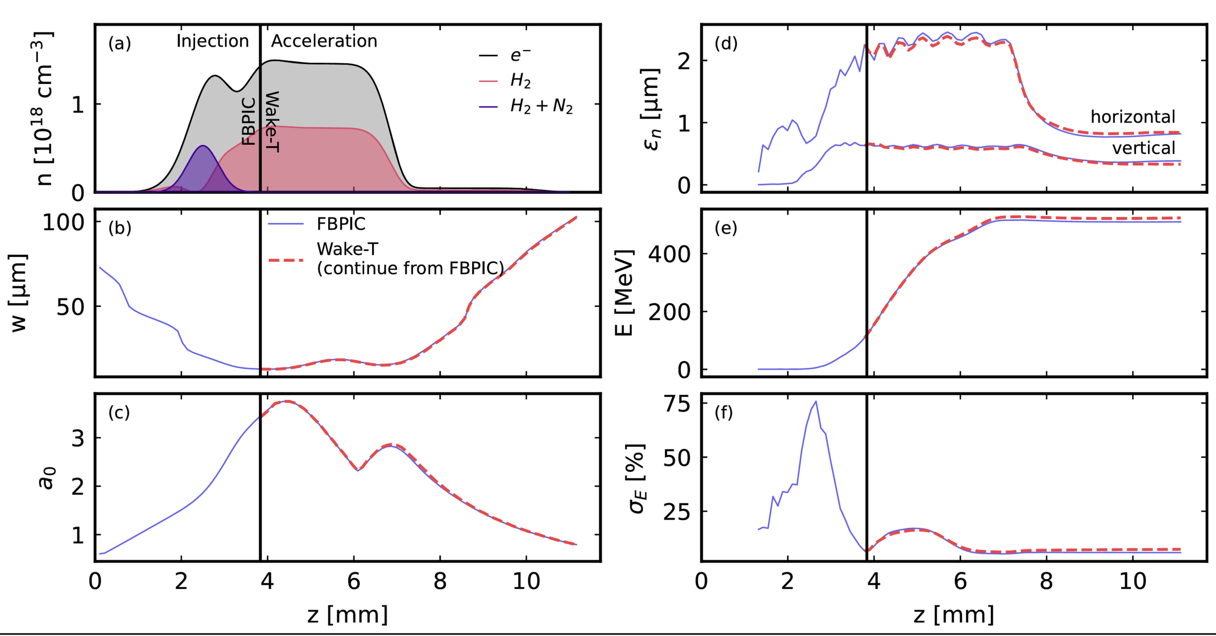


More in: `lasy.utils.laser_utils`

Handshake Between Different Simulation Codes

`lasy` makes it easier to combine codes with different laser representations

Example use case: Injection in full electromagnetic PIC (eg. FBPIC [6]) + acceleration in quasistatic PIC code (eg. Wake-T [7])



LASY Development

Hosted on github and powered by community contributions:

Angel Ferran Pousa, Alexander Sinn, Anna Liisa Puchert, Matt Streeter, David McMahon, Nadezhda Khachatrian, Cristina Mariani, Ryan Sandberg, Ilian Kara-Mostefa

Organisations and Funding

References

[1] M. Thévenet et al., arXiv:2403.12191v1 (2024)
[2] B. Beaurepaire et al. Phys. Rev. X 5.3, 031012 (2015)
[3] I. Andriyash, "Axiprop: simple-to-use optical propagation tool" (2020)
[4] K. Oubrerie et al., J. Opt. 24, 045503 (2022)
[5] A. Huebl et al., DOI:10.5281/zenodo.591699 (2015)
[6] R. Lehe et al., Comput. Phys. Commun. 203, 66 (2016)
[7] A. Ferran Pousa et al., Journ. Phys. 1350.1 IOP Publishing (2019)