Developing a Python-Based Post-Processor for GINGER-3D

&

Timing Analysis of GENESIS-4 and GINGER-3D

Shelley Tong [1], William Fawley [2], Michael Ehrlichman [2]

[1] University of California, Irvine,

[2] Accelerator Research Division, SLAC National Accelerator Laboratory

INTRODUCTION : FEL Simulations

➔Why Are FEL Simulations Important?

- Free electron lasers (FEL) equations can only be solved analytically using idealized and simplified models; more comprehensive descriptions requires numerical methods.
- Over the past 40 years, numerical codes and FEL simulations have been developed to model the complex interactions between electron beams and radiation within undulators, including extension to multiprocessor platforms.
- The main FEL simulation codes used at SLAC include GENESIS, GINGER-3D, and PUFFIN.

INTRODUCTION : GINGER-3D

➔What is GINGER-3D & Why Is It Necessary?

- GINGER-3D is an advanced FEL simulation code that extends the axisymmetric field solver (r-z) of the original GINGER to a full 3D model (x-y-z).
- This extension is necessary because FEL electron beam pulses and common "strong" quadrupole focusing are not axisymmetric, and FEL radiation often contains non-negligible, non-axisymmetric components (e.g., SASE startup).
- For example, GINGER-3D can study electron beams with tilts and offsets, as well as non-axisymmetric radiation patterns, such as orbital angular momentum modes.

Python-Based GINGER-3D Post-Processor

➔Why do we need a Python-Based Post-Processor?

- A post-processor enables users to visualize diagnostic results, extract input files, and analyze/compare outputs from different runs.
- The previous post-processor, built in Fortran, focused on production mode and was less interactive.
- Our current GUI-based post-processor is developed using Python for several reasons:
  1) Python is more widely known than Fortran, making it easier for most people to modify and develop the code further.
  2) Python strongly supports object-oriented programming, simplifying the development process.
  3) Python integrates better with GUI libraries, making the post-processor more interactive and user-friendly.

➔Key Outputs of GINGER-3D Post-Processor:

- Key output parameters of interest include:

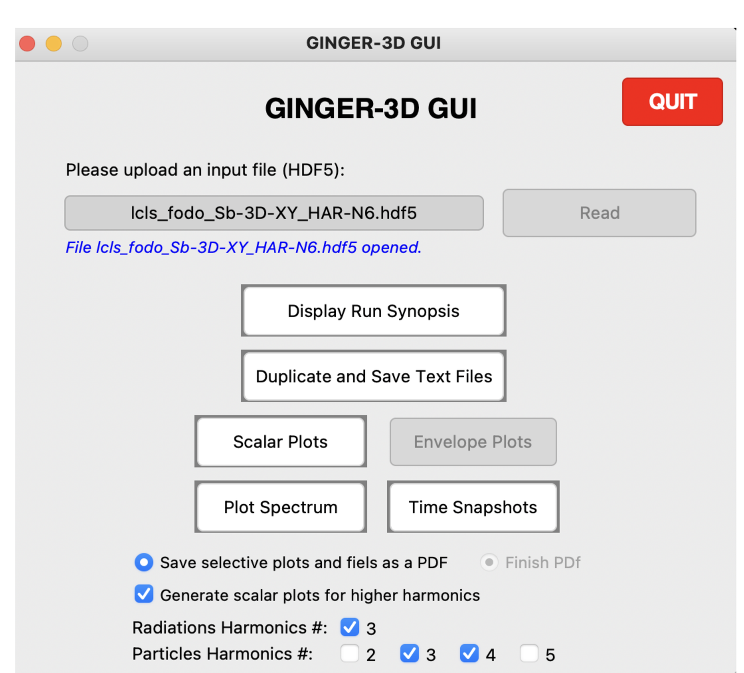
  Radiation power, spectra (both near-field and far-field), transverse profiles (both near-field and far-field), electron beam coherent microbunching energy loss, and instantaneous energy spread.

Post-Processor Functionality Design

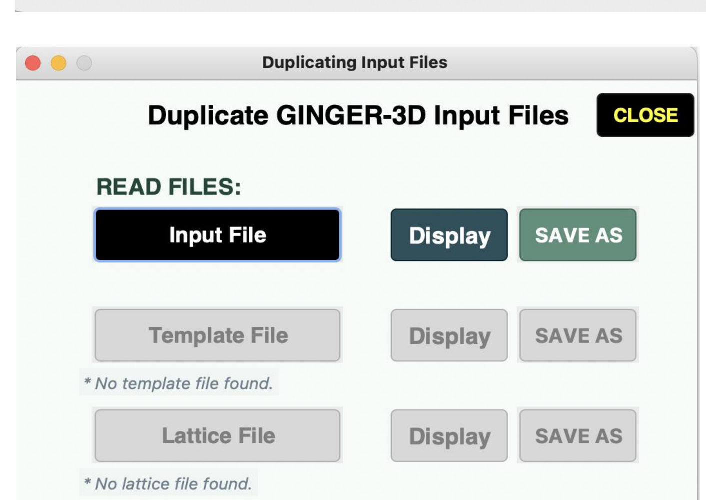
Our current GUI includes the following functionalities:

- **Display Key Parameters:** Shows details of the GINGER-3D run, such as numerical grid information, electron beam, magnetic undulator and focusing parameters, and radiation input parameters.
- **Access Input Files:** Display GINGER-3D the various input files embedded in the HDF5-formatted output file and permit duplication to new disk files.
- **Graphical Results:** Provides graphical visualization of all key outputs from the GINGER-3D run. These are are divided into four main categories: scalar quantities, particle envelope data, time snapshots, and spectra plots (implemented using Fast Fourier Transform (FFT) routines).

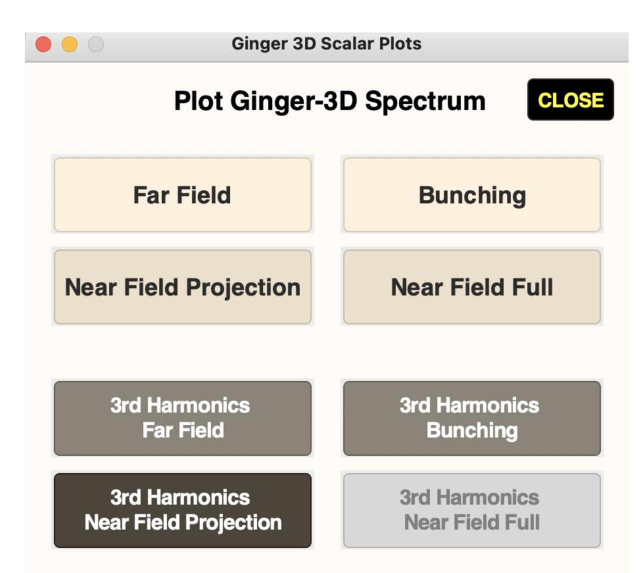
User Interface & Graphical Results



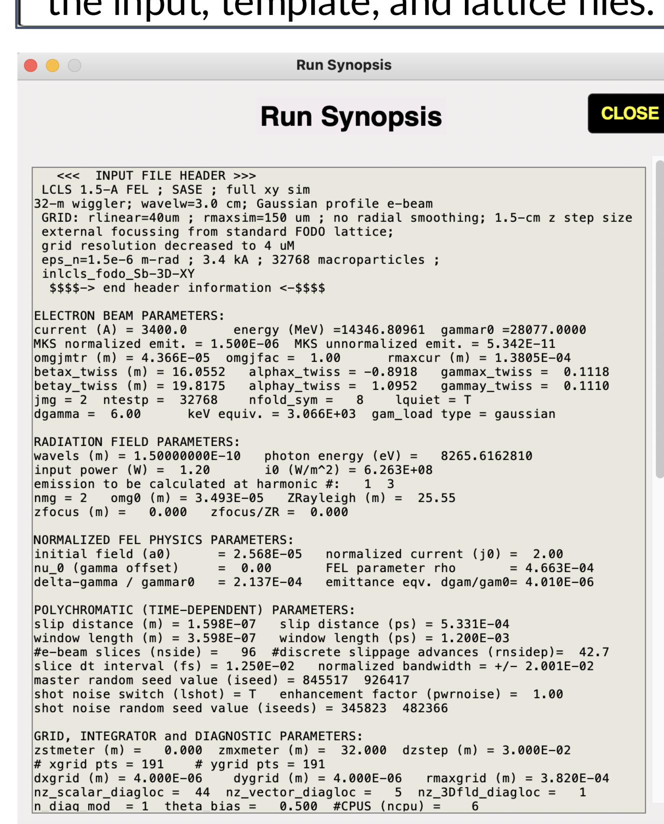
Main user interface showcasing different functionalities, including options to save figures as PDF files and plot higher harmonics.



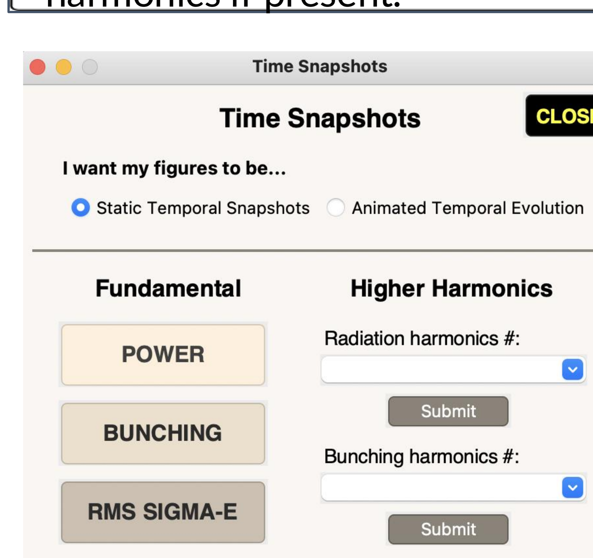
Allows the users to display and save the input, template, and lattice files.



Allows users to plot near- and far field radiation profiles and spectra, including higher harmonics if present.

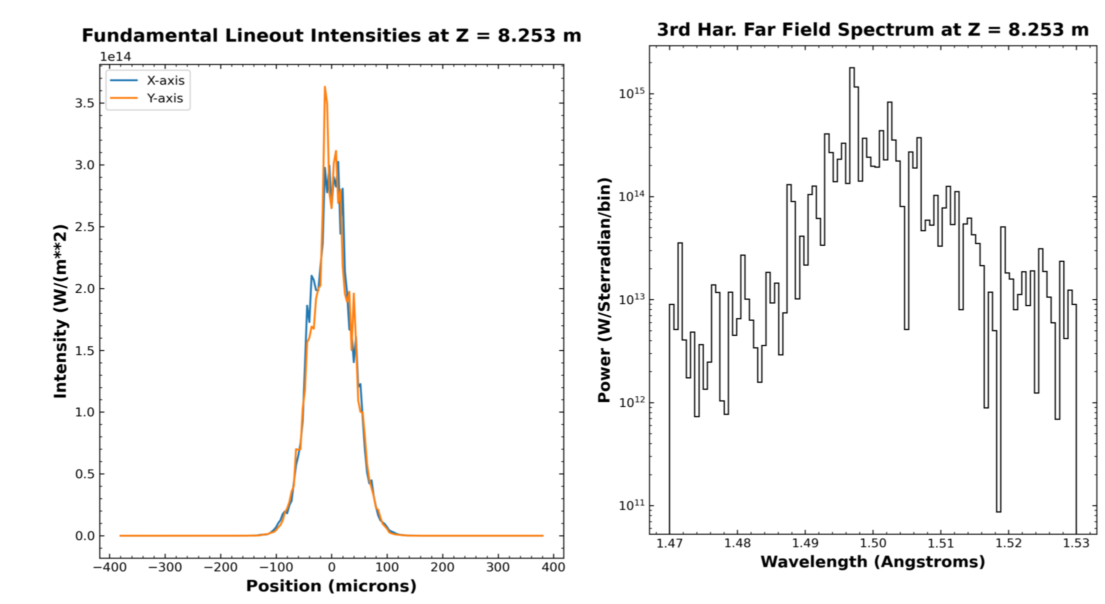


Displays summary of the run.

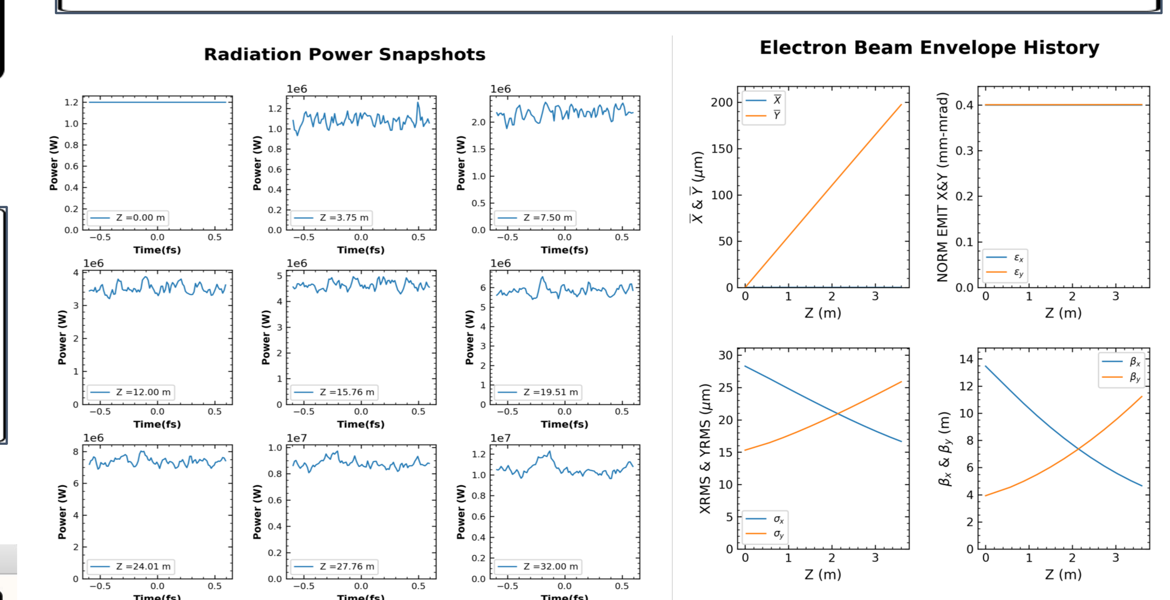


Enables users to plot static and animated time snapshots of scalar radiation and particle quantities.

Scan the QR codes to see the animated time snapshots!



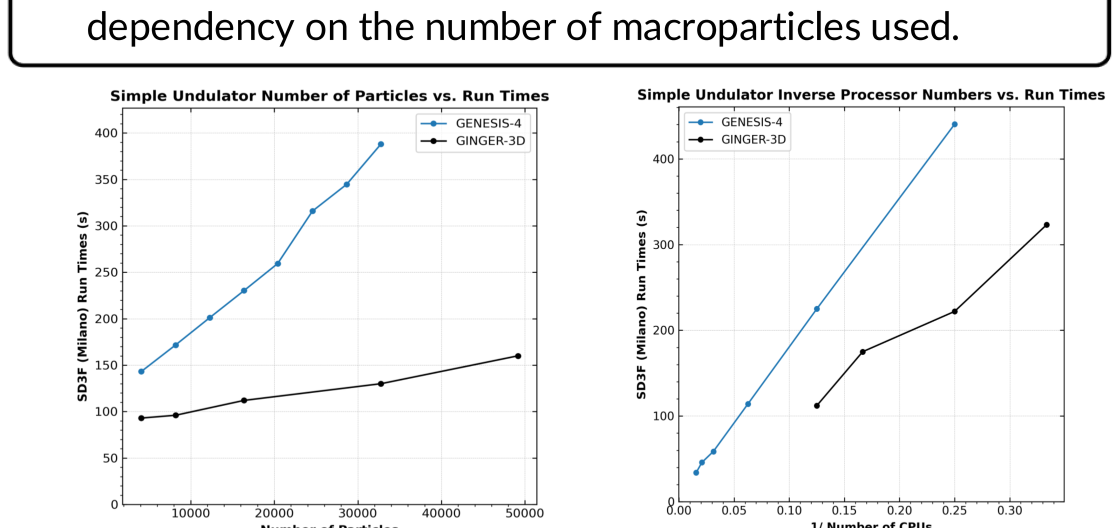
Examples of figures generated by the post-processor:
Left: Lineout intensity as a function of position on the x and y-axis.
Right: Far field power spectrum at the 3rd harmonics at Z = 8.25m.

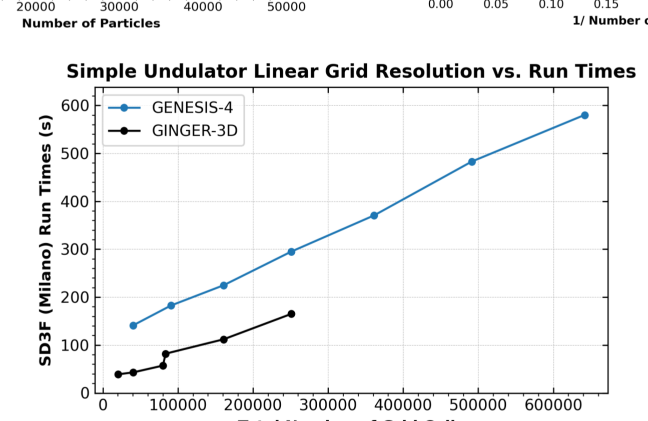


Left: Time snapshots of radiation power at various z positions.
Right: Z-histories of Electron beam envelope quantities

GINGER-3D & GENESIS-4 Runtime Analysis:

- In addition to designing and writing the new Python-based post-processor, I analyzed the runtimes of GINGER_3D relative to GENESIS-4, a widely-used FEL simulation code, by varying the number of grid points, particles, and processors on the multiprocessing SLAC Linux cluster S3DF.
- Plots of runtime vs. the number of macroparticles and grid points were generated using 8 processors.
- The results show that GINGER_3D is significantly more efficient, with runtimes approximately half of those of GENESIS-4. GINGER_3D also demonstrates less dependency on the number of macroparticles used.





ACKNOWLEDGEMENTS

Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.

REFERENCES

[1] Reiche, Sven. "FEL Simulations: History, Status and Outlook." *FEL 2010 - 32nd International Free Electron Laser Conference*, 2010.

[2] Fawley, William M. "A user manual for GINGER and its post-processor XPLOTGIN." (2002).

[3] Fawley, William M. "An enhanced ginger simulation code with harmonic emission and HDF5 io capabilities." (2006).