Intro. Lecture 01: Overview[*]

Prof. Steven M. Lund
Physics and Astronomy Department
Facility for Rare Isotope Beams (FRIB)
Michigan State University (MSU)

US Particle Accelerator School (USPAS) Lectures On
"Self-Consistent Simulations of Beam and Plasma Systems"
Steven M. Lund, Jean-Luc Vay, Remi Lehe, and Daniel Winklehner

US Particle Accelerator School Summer Session
Colorado State U, Ft. Collins, CO, 13-17 June, 2016
(Version 20160613)

[*] Research supported by:
FRIB/MSU, 2014 On via: U.S. Department of Energy Office of Science Cooperative Agreement DE-SC0000661and National Science Foundation Grant No. PHY-1102511
and
LLNL/LBNL, Pre 2014 via: US Dept. of Energy Contract Nos. DE-AC52-07NA27344 and DE-AC02-05CH11231

Abbreviated Outline
Introductory Lectures on Self-Consistent Simulations

1. Overview
2. Classes of Intense Beam Simulations
3. Overview of Basic Numerical Methods
4. Numerical Methods for Particle and Distribution Methods
5. Diagnostics
6. Initial Distributions
7. Numerical Convergence
8. Practical Considerations
9. Example Simulations

Outline
Introductory Lectures on Self-Consistent Simulations

1) Overview
   - A. Why Numerical Simulation?
   - B. Which Numerical Tools

2) Classes of Intense Beam Simulations
   - A. Overview
   - B. Particle Methods
   - C. Distribution Methods
   - D. Moment Methods
   - E. Hybrid Methods

3) Overview of Basic Numerical Methods
   - A. Discretization
   - B. Discrete Numerical Operations
   - C. Time Advance

Outline - 2

4) Numerical Methods for Particle and Distribution Methods
   - A. Overview
   - B. Integration of Equations of Motion
   - C. Field Solution
   - D. Weighting: Depositing Particles on the Field Mesh and Interpolating Gridded Fields to Particles
   - E. Computational Cycle for Particle in Cell Simulations

5) Diagnostics
   - A. Overview
   - B. Snapshot Diagnostics
   - C. History Diagnostics

6) Initial Distributions and Particle Loading
   - A. Overview
   - B. KV Load and the rms Equivalent Beam
   - C. Beam Envelope Matching
   - D. Semi-Gaussian Load
   - E. PseudoEquilibrium Distributions Based on Continuous Focusing Equilibria
   - F. Injection off a Source

Outline - 3

Overview
A. Why Numerical Simulation?

Builds intuition of intense beam physics

- "The purpose of computation is insight, not numbers."
  Richard Hamming, chief mathematician of the Manhattan Project and Turing Award recipient
- Advantages over laboratory experiments:
  - Full nonintrusive beam diagnostics are possible
  - Effects can be turned on and off

Allows analysis of more realistic situations than analytically tractable

- Realistic geometries
- Non-ideal distributions
- Combined effects
- Large amplitude (nonlinear) effects

Insight obtained can motivate analytical theories

- Suggest and test approximations and reduced models to most simply express relevant effects

---

Why Numerical Simulation? (2)

Can quantify expected performance of specific machines

- Machines and facilities expensive – important to have high confidence that systems will work as intended/promised to funding agencies
- Difficult to get proposals funded without significant motivation provided by simulation
  - Agencies want confidence that investment will work
- - Simulations for concept verification help credibility

Computers and numerical methods/libraries are becoming more powerful

Enables both analysis of more realistic problem modeling and/or better numerical convergence

- Bigger and faster hardware
  - Processor speed increasing
  - Parallel machine architectures
  - Greater memory
- More developed software
  - Improved numerical methods
  - Libraries of debugged code modules
  - Graphics and visualization tools
  - Data management and software maintenance tools

Why Numerical Simulation? (3)

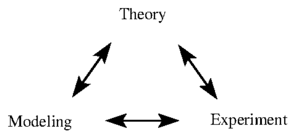
Simulations are increasingly powerful and valuable in the analysis of intense beams, but should not be used to exclusion

- Parametric scaling is *very* important in machine design
  - Often hard to understand best choices in physical aperture sizes, etc.
  - Although scaling can be explored with simulation, analytical theory often best illustrates the trade-offs, sensitivities, and relevant combinations of parameters
- Concepts often fail due to limits of technology (e.g., fabrication tolerances, material failures, and unanticipated properties) and hence full laboratory testing is vital
  - Many understood classes of errors can be probed with simulation
  - Unanticipated error sources are most dangerous!
  - Must understand both contemporary technology limits and limitations of numerical models to work effectively
- Economic realities often severely limit what can be constructed
  - Simulating something financially unattainable may serve little purpose
  - Need compelling evidence of improvements for major experiment funding

Why Numerical Simulation? (4)

The highest understanding and confidence is achieved when results from analytic theory, numerical simulation, and experiment all converge

- Motivates model simplifications and identification of relevant sensitivities
- Unfortunately, full agreement often hard in complex systems/problems. But gives strong confidence when it works!



Numerical simulation skills are highly sought in many areas of accelerator and beam physics

- Specialists readily employable
- Skills transfer easily to many fields of physics and engineering

B. Which Numerical Tools?

There are many simulation codes with a wide variety of scope and capabilities which evolve in time. This course is intended as a top-down review of contemporary methods commonly employed in numerical simulation of intense beams and plasmas.

- The topic of codes and preferences can at times (especially with developers!) border on discussions of religious preferences.

Numerous programming languages are employed in numerical simulations of intense beams and plasmas

- Most common today: Fortran (90, 2000, ...), C, C++, Java, ...
- Strengths and weaknesses depend on application, preferences, and history (legacy code)

Results are analyzed with a variety of graphics packages:

The well-known saying: “A picture is worth a thousand words” nicely summarizes the importance of good graphics in illustrating concepts.

- Commonly used: Matplotlib, Gnuplot, IDL, Narcisse, NCAR, Gist, Mathematica, Mathcad, MATLAB, Maple, Sage, ...
- Plot frames combine into movies to better illustrate evolutions
- Use can greatly simplify construction of beam visualization diagnostics
  - Many person-years of labor go into writing extensive graphics packages

Which Numerical Tools? (2)

A modern and flexible way to construct simulation packages is to link routines in fast, compiled code with an interactive interpreter

- Examples: Python, Yorick, Basis ...
- Python used in OS development and will not disappear anytime soon
  - Has many open and free numerical, data, scientific, and graphics packages
  - We will employ extensively in this course

Advantages of using interactive interpreters:

- Allows routines to be coded in mixed languages
  - Renders choice of programming languages less important
  - Discussing programming languages appears to border on religion …
- Flexible reconfiguration of code modules possible to readily adapt for unanticipated needs
  - Reduces need for recompilation and cumbersome structures for special uses
  - Aids cross-checking problems and debugging when switching numerical methods and parameters, etc.
- “Steering” of code during runs to address unanticipated side effects
  - Change diagnostics/methods in middle of long run based on results obtained
- Can facilitate modern, object-oriented structure for the problem description
- Allows use of wide variety of packages based on a users preference
  - Graphics/diagnostics, numerical methods (e.g., Scientific Python: SciPy), ...

Which Numerical Tools? (3)

Broadly overviewing programming languages and graphics packages is beyond the scope of this class. Here our goal is to survey numerical simulation methods employed without presenting details of specific implementations.

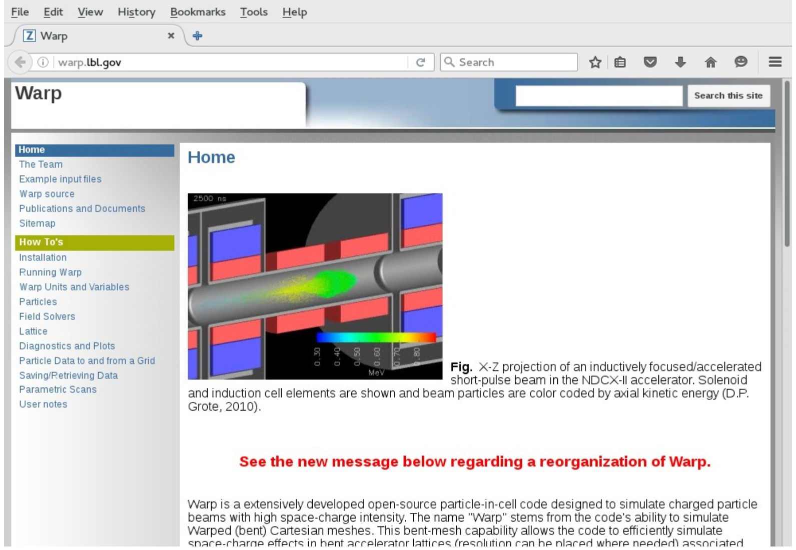
We will show examples based on the Warp particle-in-cell code developed for intense beam simulation at LLNL and LBNL: http://warp.lbl.gov

- Warp is so-named since it works on a “warped” Cartesian mesh with bends
- Alex Friedman (LLNL) original architect/developer in 1980s using LLNL steerable code philosophy; Dave Grote primary developer for many years
- Warp has evolved into a family of particle-in-cell code tools built around a common Python interpreter for flexible operation
- Open Sourced since 2013: free to modify, extend, redistribute
- Optimized for simulation of beams with self-consistent space-charge forces
- Actively maintained and extended:
  - Movers
  - Electrostatic Field Solvers
    Electromagnetic Field Solvers
    ⋮
  - Diagnostics
  - Multi-species
  - E-Cloud effects
  - Mesh Refinement
  - Dense Plasmas
  - Multipole Fields

More on Warp in parallel Warp/Python Lectures and Advanced Lectures

More information on the Warp code: http://warp.lbl.gov/

- Work in progress: being updated and extended year by year



Home

Fig. X-Z projection of an inductively focused/accelerated short-pulse beam in the NDCX-II accelerator. Solenoid and induction cell elements are shown and beam particles are color coded by axial kinetic energy (D.P. Grote, 2010).

See the new message below regarding a reorganization of Warp.

Warp is a extensively developed open-source particle-in-cell code designed to simulate charged particle beams with high space-charge intensity. The name "Warp" stems from the code's ability to simulate Warped (bent) Cartesian meshes. This bent-mesh capability allows the code to efficiently simulate space-charge effects in bent accelerator lattices (resolution can be placed where needed) associated

---

Corrections and suggestions for improvements welcome!

These notes will be corrected and expanded for reference and for use in future editions of US Particle Accelerator School (USPAS) and Michigan State University (MSU) courses. Contact:

Prof. Steven M. Lund
Facility for Rare Isotope Beams
Michigan State University
640 South Shaw Lane
East Lansing, MI 48824

lund@frib.msu.edu
(517) 908 – 7291 office
(510) 459 - 4045 mobile

Please provide corrections with respect to the present archived version at:

https://people.nscl.msu.edu/~lund/uspas/scs_2016

Redistributions of class material welcome. Please do not remove author credits.

References: For more information see:

These US Particle Accelerator School (USPAS) course notes are posted with updates, corrections, and supplemental material at:

https://people.nscl.msu.edu/~lund/uspas/scs_2016/

This course evolved from material originally presented in a related USPAS course :

JJ Barnard and SM Lund, *Beam Physics with Intense Space-Charge,* USPAS:
https://people.nscl.msu.edu/~lund/uspas/bpisc_2015/ 2015 Lectures + Info

- Also taught at the USPAS in 2011, 2008, 2006, 2004, and 2001 and a similar version at UC Berkeley in 2009
- This course serves as a reference for physics discussed in this course from a numerical modeling perspective.

References: Continued (2):

Numerical Methods

Forman S. Acton, *Numerical Methods that Work*, Harper and Row Publishers, New York (1970)

Steven E. Koonin, *Computational Physics*, Addison-Wesley Publishing Company (1986)

W. Press, B. Flannery, S. Teukolsky, W. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press (1992)

Particle Methods

C.K. Birdsall and A.B. Langdon, *Plasma Physics via Computer Simulation*, McGraw-Hill Book Company (1985)

R.W. Hockney and J.W. Eastwood, *Computer Simulation using Particles*, Institute of Physics Publishing (1988)

Review of Initial Distribution Loads

S. Lund, T. Kikuchi, and R. Davidson, “Generation of initial kinetic distributions for simulation of long-pulse charged particle beams with high space-charge intensity,” PRSTAB **12**, 114801 (2009)

Acknowledgments:

These lecture notes reflect input from numerous scientists and engineers who helped educate the author in accelerator physics over many years. Support enabling the long hours it took to produce these lecture notes were provided by the Facility for Rare Isotope Beams (FRIB) at Michigan State University (MSU), Lawrence Livermore National Laboratory (LLNL), and Lawrence Berkeley National Laboratory (LBNL). Special thanks are deserved to:

Rodger Bangerter
Martin Berz
John Barnard
Oliver Boine-Frankenheim
Richard Briggs
Ronald Davidson
Mikhail Dorf
Andy Faltens
Bill Fawley
Giuliano Franchetti
Alex Friedman
Dave Grote
Irving Haber
Klaus Halbach
Enrique Henestroza
Ingo Hoffmann
Dave Judd
Igor Kagonovich
Takashi Kikuchi
Rami Kishek
Joe Kwan
Ed Lee
Daniela Leitner
Steve Lidia
Guillaume Machicoane
Felix Marti
Hiromi Okamoto
Eduard Pozdeyez
Martin Reiser
Lou Reginato
Robert Ryne
Gian-Luca Sabbi
Peter Seidl
William Sharp
Peter Spiller
Edward Startsev
Ken Takayama
Jean-Luc Vay
Will Waldron
Tom Wangler
Jie Wei
Yoshi Yamazaki
Simon Yu
Pavel Zenkovich
Yan Zhang
Qiang Zhao

Acknowledgments Continued:

Special thanks are deserved for Alex Friedman and Dave Grote of the Lawrence Livermore and Lawrence Berkeley National Laboratories for help with these notes and extensively educating the author in simulation methods.

Sven Chilton (UCB) assisted in the development of part of these lecture notes and in generating some of the numerical examples and figures

Kei Fukushima, (Hiroshima U), Kazuya Osaki (Hiroshima U), Jonathan Wong (U. Hong Kong), and Albert Yuen (UCB) attend an informal version of a simulation course based on these notes at LBNL in 2012 which resulted in numerous improvements in material.

Rami Kishek (UMD) assisted teaching a version of this course and contributed to the simulation notes.

Irving Haber (UMD) helped educate the author on various simulation methods.

Michiel de Hoon helped with an early version of the lectures and with example Lagrangian methods.