

# ORBIT

**S. Cousineau, V. Danilov, J. Galambos, J. A. Holmes,  
A. Shishlo  
SNS**

**W. Chou, F. Ostiguy, L. Michelotti  
Fermilab**

# Motivation for ORBIT

---



- Model the beam dynamics of high intensity proton rings such as FNAL Booster, AGS Booster, PSR, and **SNS**.
- Treat collective effects due to space charge and wakefields in detail.
- Treat accelerator environment faithfully and in detail. ORBIT should be a quantitative design tool as well as a theoretical beam dynamics code.
- ORBIT is an open code. We give the source freely and encourage use and development by others.

# ORBIT: Present Capabilities

---



- ORBIT is a particle-tracking code in 6D phase space.
- ORBIT is designed to simulate real machines: it has detailed models for
  - transport through various types of lattice elements
  - injection foil and painting
  - RF and acceleration
  - 2.5D space charge with or without conducting wall beam pipe
  - longitudinal impedance and 1D longitudinal space charge
  - Transverse impedance - Danilov, Galambos, and Holmes
  - 3D space charge - Danilov and Holmes
  - apertures and collimation - Cousineau
- ORBIT has an excellent suite of routines for beam diagnostics.

# ORBIT: Present Needs

---



- Magnet lattices limited to linear matrices (symplectic), second order transport matrices (not symplectic), and thin lens multipoles. Need symplectic nonlinear maps.
- Parallel routines are old and broken. Need current and comprehensive parallelization.
- User environment is based on SuperCode, a nice but old and nonsupported, framework for code development and use. Need a new and standard user environment.

# ORBIT: Response to Needs - Fermilab Collaboration

---



- Need symplectic nonlinear maps: Leo Michelotti's library.
  - Responsibility: Michelotti and Holmes.
  - Create maps by parsing MAD input file.
  - Can split at arbitrary positions to do space charge or other desired operations.
- Need current and comprehensive parallelization.
  - Responsibility: Shishlo.
  - Will implement MPI parallelization of space charge, impedance calculations, and diagnostics.
- Need a new and standard user environment.
  - Responsibility: Ostiguy.
  - Will implement Python user interface.
- Development carried out using CVS to maintain uniformity.

# ORBIT: Electron Cloud Model

---



- We also plan to implement an electron cloud model in ORBIT
  - Responsibility: Cousineau, Danilov, and Holmes.
  - Advice and collaboration from Blaskiewicz, Furman, and Pivi.
  - Time scale for development is 6 months - 1 year.