Space-Charge Simulations of Non-Scaling FFAGs Using PTC

Dan T. Abell, George I. Bell, Andrey V. Sobol (Tech-X Corp.)
Alessandro Ruggiero, Dejan Trbojevic (BNL)
Étienne Forest (KEK)

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics under SBIR Grant No. DE-FG02-06ER8448.

PTC Now Handles Two Space-Charge Models

- **Ballistic expansion—SC kick—compression**

  The "standard" trick: use a ballistic transformation to convert a "pancake" of space-charge at fixed *s* to a particle bunch at fixed time. Kick the beam, then return to the pancake. Such an approximation can work well for short bunches.

- **Time-based tracking**

  For long bunches, the optics of the machine can significantly modify the bunch shape away from that given by the ballistic approximation. PTC gives us the tools to obtain a realistic bunch shape

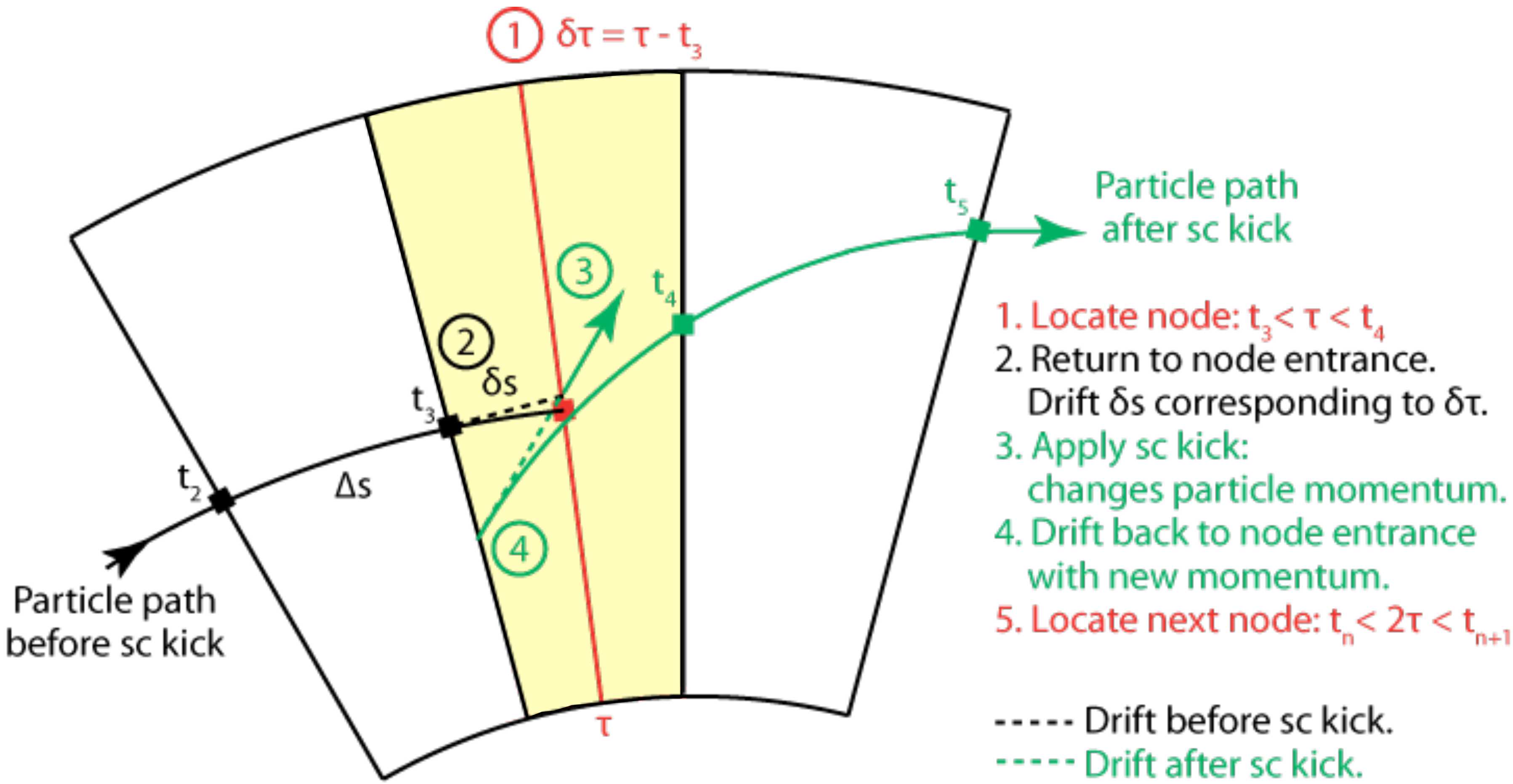
The How of Time-Based Tracking

Any high-order *s*-based integrator may be converted to a first-order time integrator *provided information is available about the three-dimensional environment*. The Polymorphic Tracking Code (PTC) has complete knowledge of the 3D environment about each node of integration.

PTC checks—node by node—to determine the integration node in which a particle reaches a given time τ. PTC identifies both the appropriate integration node and the amount of time $\delta\tau$ between the time t at the start of the integration node and τ. PTC then returns the particle to the integration node's entrance and drifts that particle forward by the distance δs corresponding to $\delta\tau$. If the element is a drift, the position of the particle at time τ is exact. Otherwise, the particle position is only first-order accurate. (See figure below.)

In the absence of collective effects, we recover the original PTC.

The How of Time-Based Tracking (cont.)



Because each integration node in PTC can report where it is in three-dimensional space, particles in a temporal_beam need only record the index of their current node.

We use particle sorting to make this algorithm more efficient.

PTC has Data Structures for Particles and Beams with Temporal Information

```
type temporal_beam
  integer :: n                                ! number of particles
  type(temporal_probe), pointer :: tp(:)      ! temporal_probe array
  real(dp) :: ent(3,3),a(3)
  real(dp) :: p0c, total_time  ! beam reference momentum and time
  type(integration_node), pointer :: c ! location near a(3)
  type(internal_state) :: state  ! defines ‘state’ of simulation
end type temporal_beam
```

A `temporal_beam` is a collection of `temporal_probes`.
It must be allocated: `call alloc(b,n,p0c)`
The initial beam frame and origin are defined in `ent` and `a`.
Set initial particle locations and momenta w/rt `ent` and `a`:

```
b%tp(i)%pos(1:3) = (x,y,z)
b%tp(i)%pos(4:6) = (px,py,pz)/p0c ! kinetic
```

Particles are placed on the correct integration nodes by

```
call position_temporal_beam(layout,b,state)
```

Example

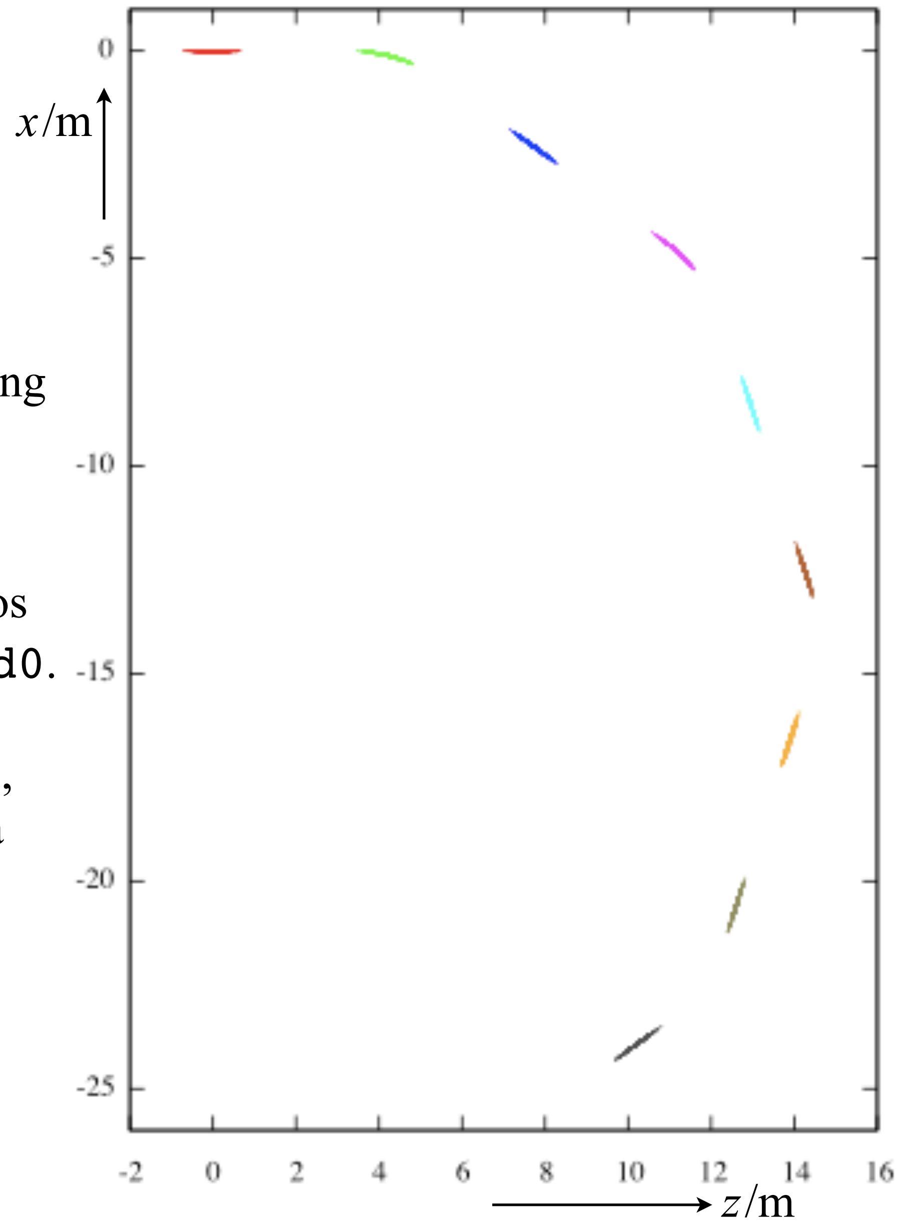
To track a temporal beam for a time c*`dt` we

```
call track_temporal_beam(B,cdt)
```

This call not only pushes the particles through the integration nodes, it also updates the data describing the full 6D particle information w/rt the global frame.

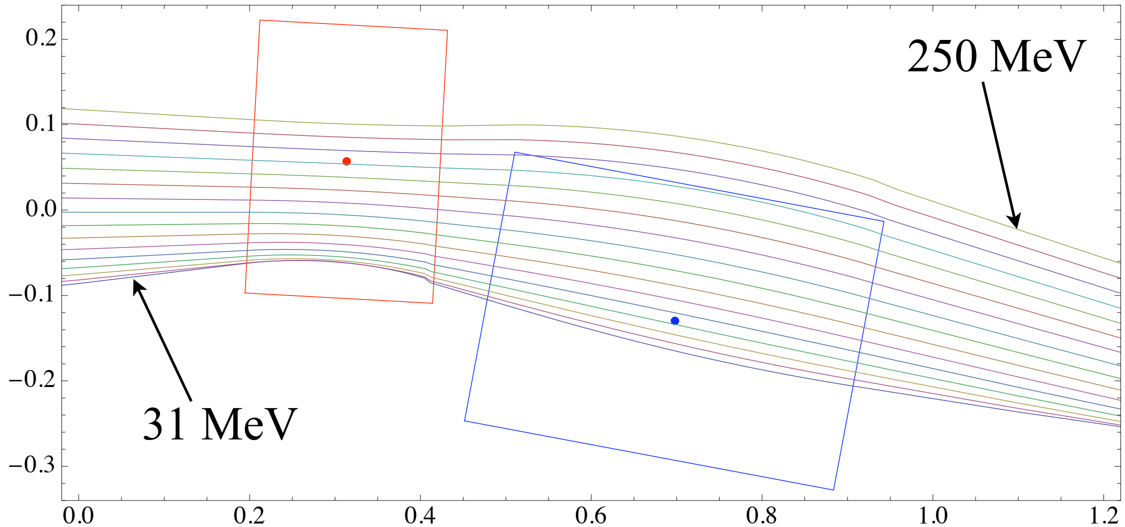
This figure shows a 20 cm beam in the Los Alamos PSR tracked for eight time-steps of `cdt = 5.0d0`.

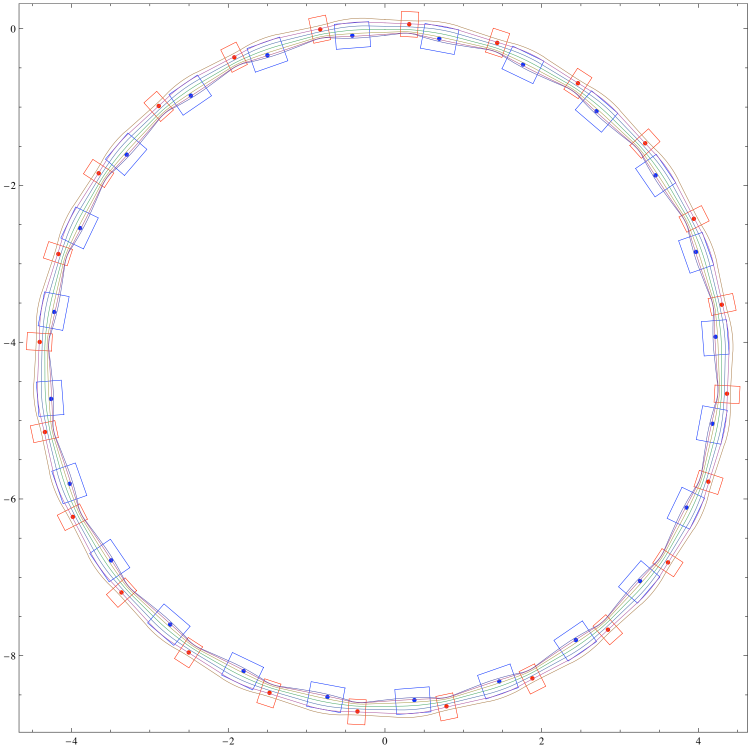
Note the curvature of some of the beam snapshots, present when part, or all, of the beam lies within a bending magnet.



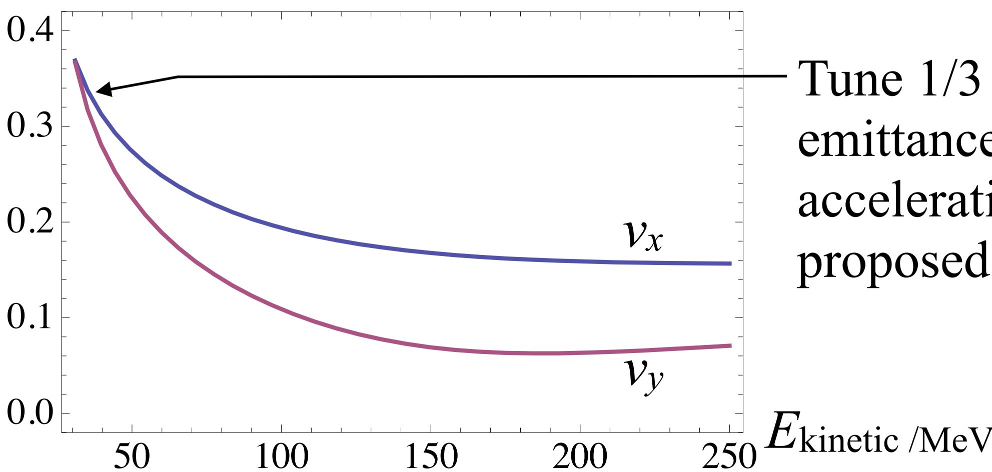
NS-FFAG for Hadron Therapy

Lattice comprising 24 cells, each with a pair of shifted quads.





Ring 2 of *PRSTAB* **10**, 054701 (2007).



Tune 1/3 causes significant particle loss for the emittance values discussed, unless the rate of acceleration is at least double, perhaps triple, the proposed value.

Summary

- **PTC is an excellent tool for modeling FFAGs, because it makes no approximations in the transverse or energy variables, and because it contains complete information about the 3D environment of the particle beam.**
- **Hard edges in the variable *s* lead to an unavoidable loss of symplecticity in the time domain. (Not an issue for FFAGs.)**
- **Poisson solvers implemented for both short and long bunches in PTC.**
- **We have compiled PTC for parallel machines (but we're not the first to do this).**
- **PTC can model realistic magnets (with fringes).**
- **Bring us your FFAG!**