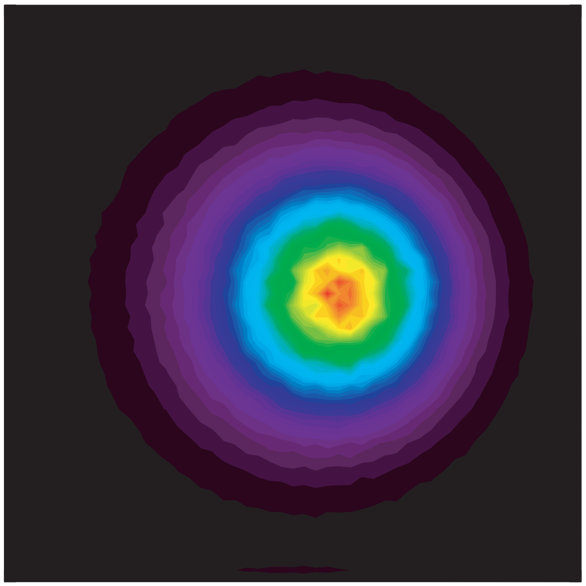
Progress on Energetic Particles in NIMROD

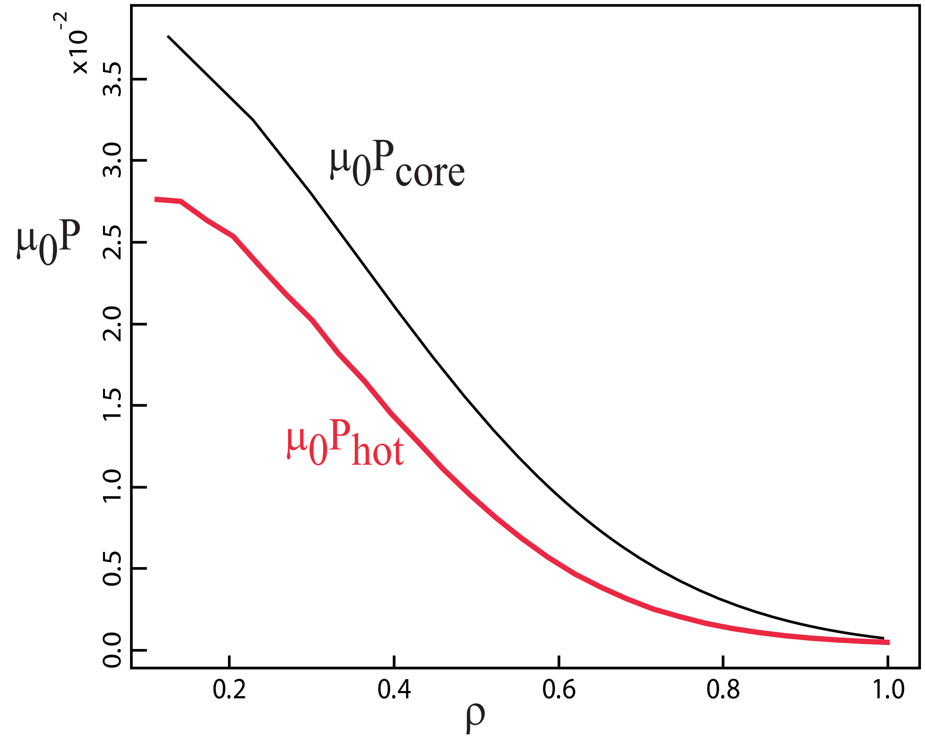
Charlson Kim, Scott Parker, Scott Kruger, Carl Sovenic
and the NIMROD Team

Outline

- ☐ Model equations
- ☐ Parallel performance
- ☐ Sorting for parallelism and minimizing cache conflicts
- ☐ Finite-element Particle-In-Cell
- ☐ Non-trivial equilibrium effects
- ☐ Recent results with δp_{hot}

Equilibrium hot particle pressure

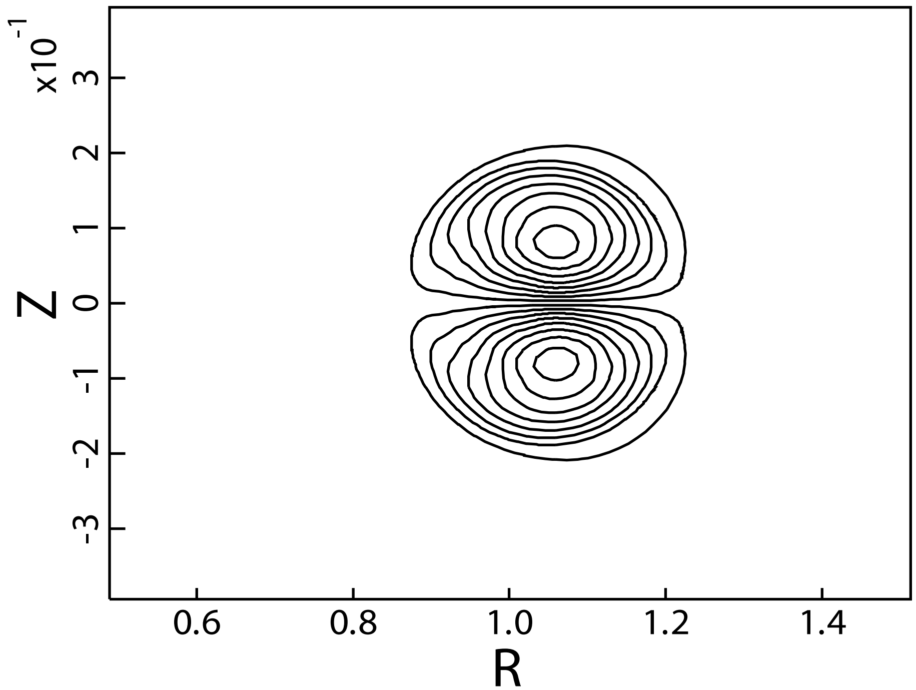




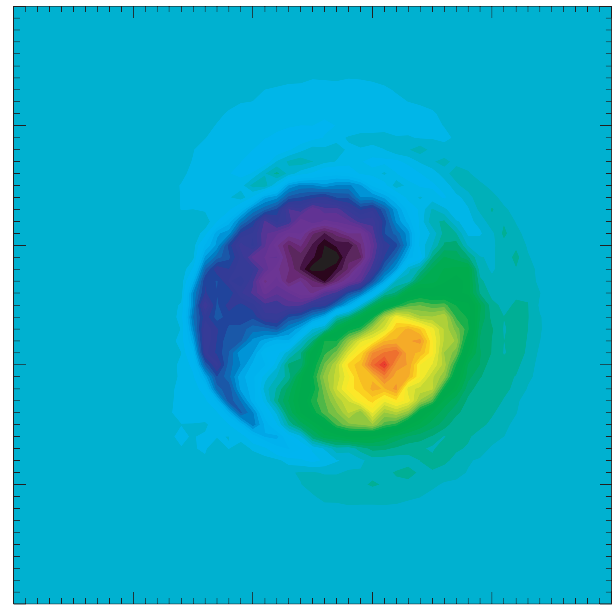
$f_0 \sim \exp(- \psi/\psi_0)$

Energetic pressure shows an eigenfunction is forming

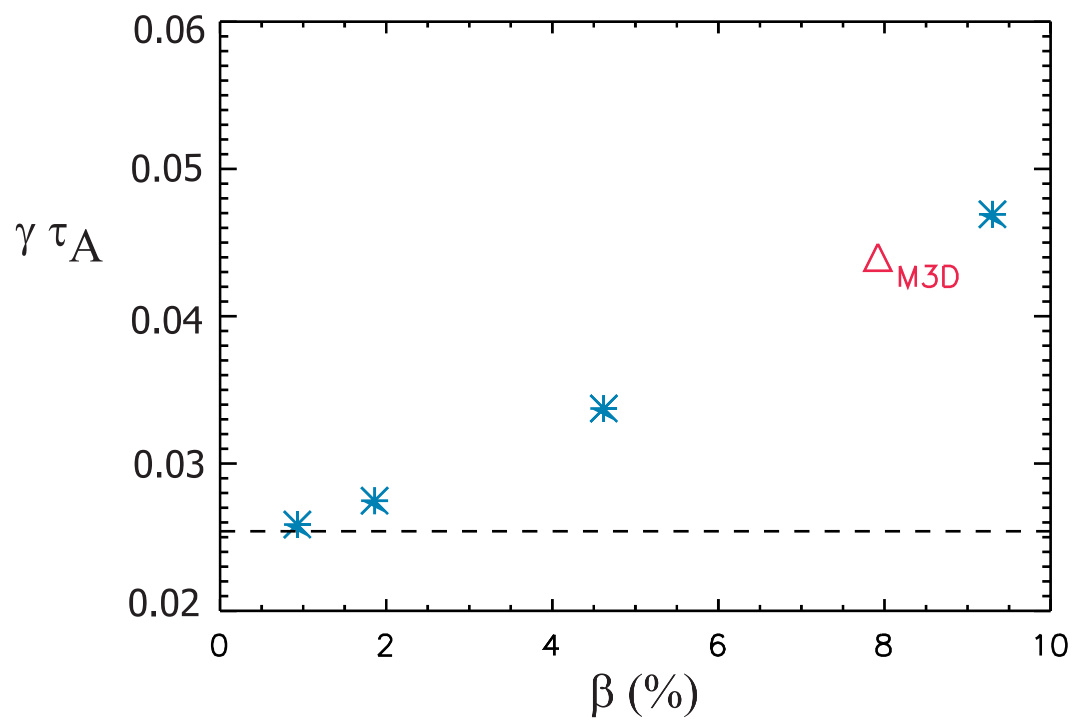
Re δP_{core} (n=1, R, Z)



Re δp_{hot} (n=1, R, Z)

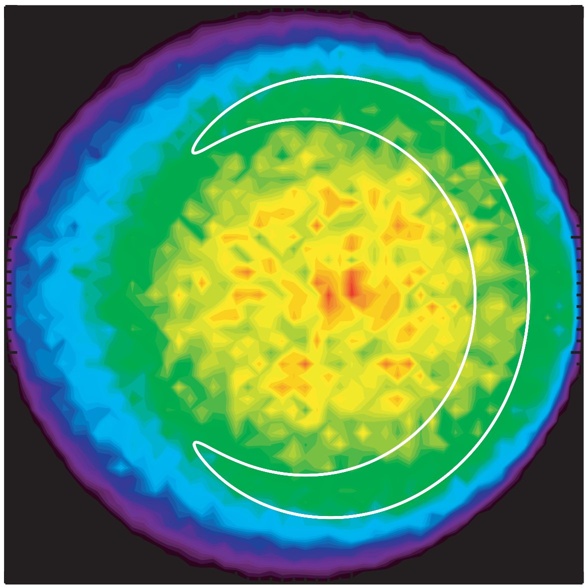


CEMM energetic particle internal kink benchmark

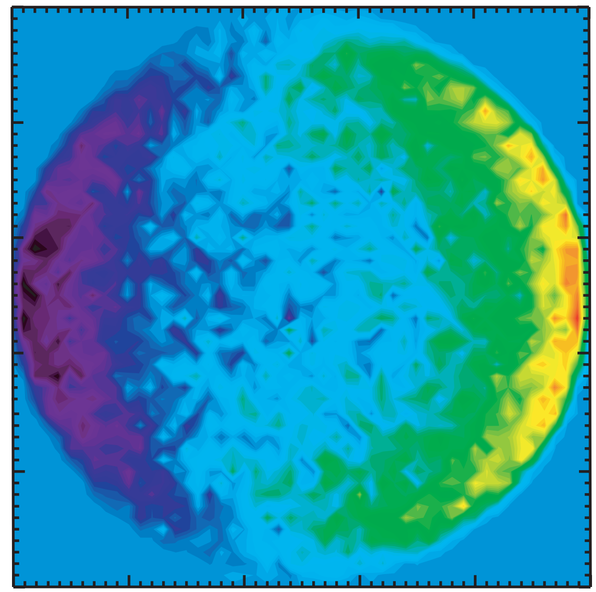


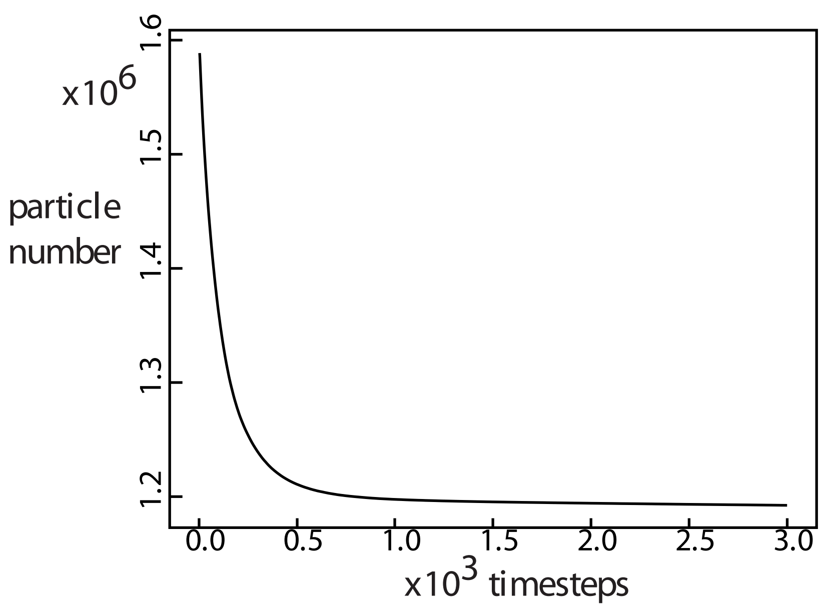
Equilibrium Flows are generated at the edge due to particle loss and pressure gradient

Pressure moment of marker particle distribution



Parallel flow moment of marker particle distribution





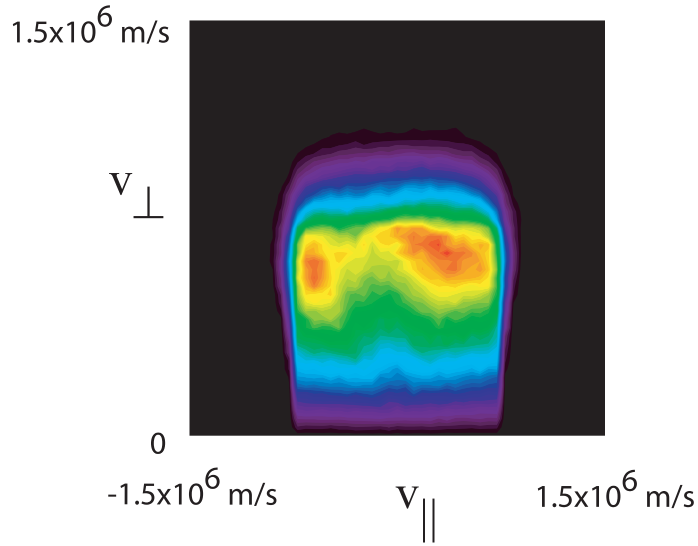
Pfirsch-Schluter Current:

$j_{||} = 2\ q/B\ dp/dr \cos\theta$

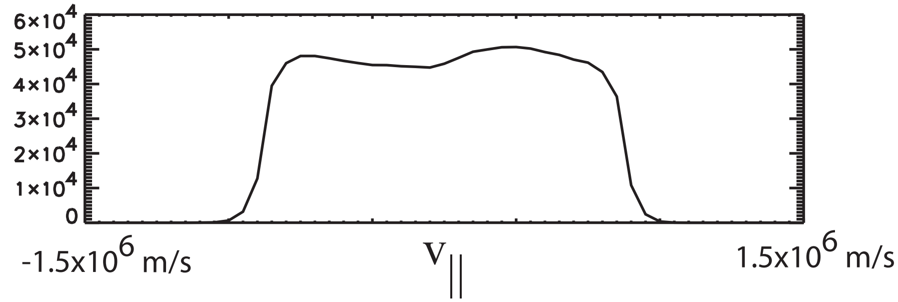
Miyamoto (1997)

Large orbit loss has slight effect on equilibrium distribution function

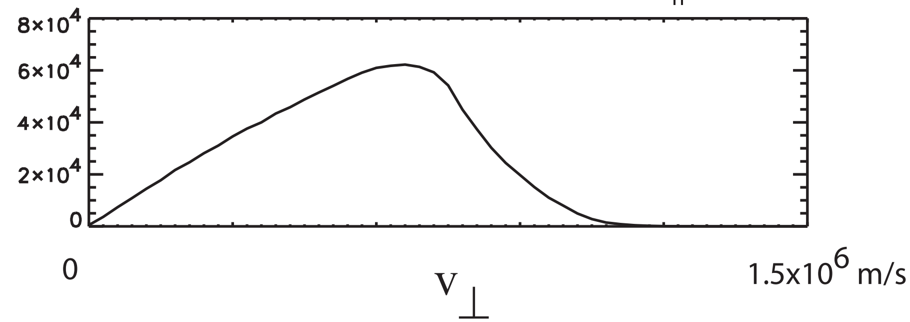
Marker particle distribution function



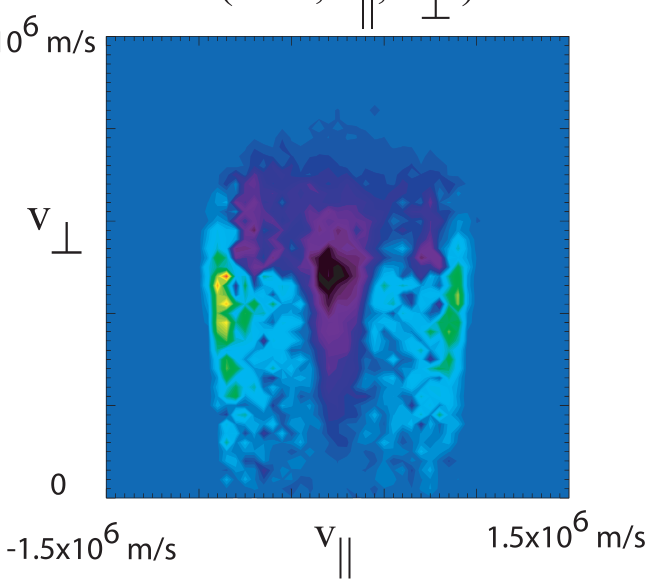
$v_{||}$ distribution averaged in $v_\perp$



$v_\perp$ distribution averaged in $v_{||}$



$\delta f\,(\,n{=}1,\, v_{||},\, v_\perp\,)$



Model Equations

Assume $n_{hot} \ll n_{bulk}$, **but** $\beta_{hot} \sim \beta_{bulk}$.

$$\rho \left(\frac{\partial}{\partial t} + \mathbf{V} \cdot \nabla \right) \mathbf{V} = \mathbf{J} \times \mathbf{B} - \nabla \cdot \mathbf{P} - \nabla \cdot \mathbf{P}_{hot}$$

$$\mathbf{P}_{hot} = \textstyle\int m \mathbf{v}' \mathbf{v}' \delta f \; d\mathbf{v}$$
$$\mathbf{v}' = \mathbf{v} - \mathbf{V}$$

Equations of motion

$$m \frac{du_\parallel}{dt} = -\hat{b} \cdot (\mu \nabla B + e\mathbf{E})$$
$$\frac{d\mathbf{R}}{dt} = \hat{b} u_\parallel + \frac{m}{eB^4} \left(u^2 + \frac{v_\perp^2}{2} \right) \left(\mathbf{B} \times \nabla \frac{B^2}{2} \right) + \frac{\mathbf{E} \times \mathbf{B}}{B^2}$$

Slowing down distribution using $f_0 = f_0(P_\zeta, \varepsilon)$

$$f_0 \;=\; \frac{A\exp(-\frac{P_\zeta}{P_0})}{\varepsilon^{3/2} + \varepsilon_0^{3/2}}$$

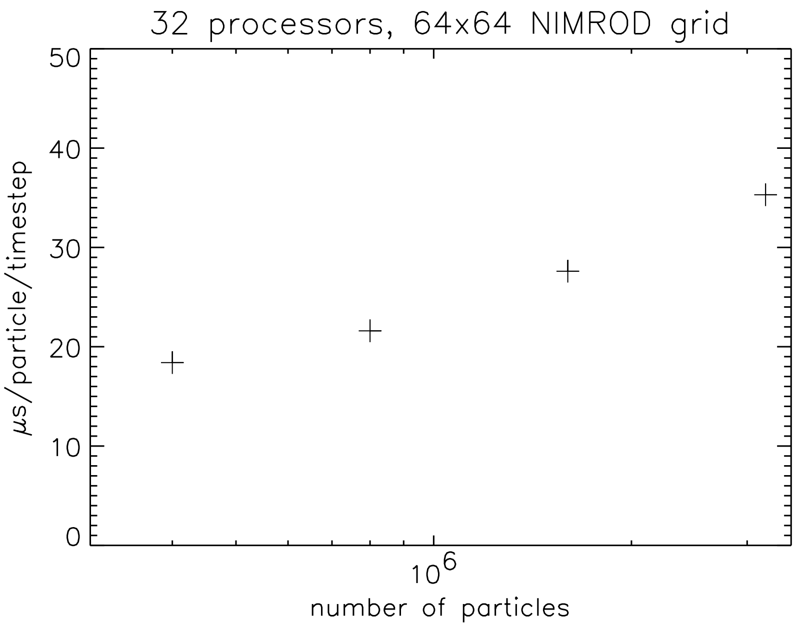
where $P_\zeta = g(\psi_p)\rho_\| - \psi_p$

$$\dot{\delta f} \;=\; -\mathbf{v_1} \cdot \nabla f_0 - e\mathbf{v} \cdot \mathbf{E}\partial_\varepsilon f_0$$

where

$$\mathbf{v_1} \;=\; \frac{\mathbf{E} \times \mathbf{B}}{B^2} + v_\| \frac{\delta\mathbf{B}}{B}$$

Parallel performance

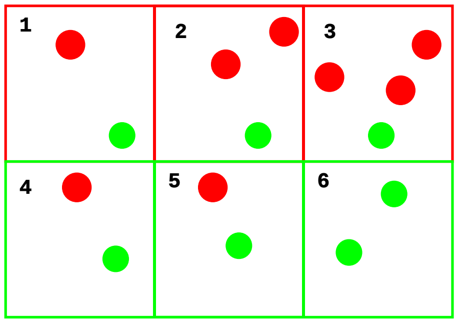


- particle performance at 10's μs/particle/timestep
- Algorithm is scalable, total cpu time $\propto \dfrac{1}{\text{no. of processors}}$
- Particle orbits exhibit near perfect energy conservation

Particle Sorting

- Sorting is important because:
  - sorting makes domain decomposition of particles trivial
  - cache thrashing is minimized
- Each processor does a "bucket" sort of it's own assigned particles
- Particles are sorted by their global logical (FE grid) coordinate
- Particles with a logical coordinate outside of the processor subdomain (r-block) are passed to their appropriate processor
- A locally sorted list of particles is finally tabulated on each processor

The Bucket Sort

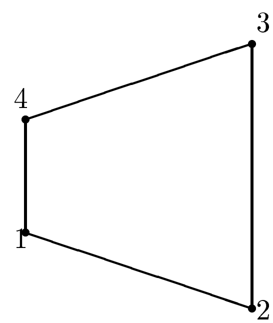


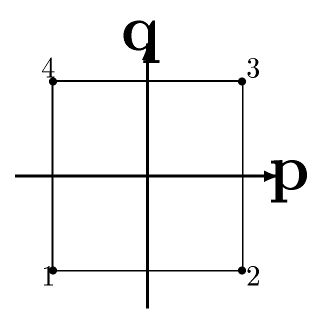
cell number	1	2	3	4	5	6
red count	1	2	3	1	1	0
green count	1	1	1	1	1	2
total count	2	3	4	2	2	2
local displacement	2	5	9	2	4	6

- The number of particles in each element is tabulated locally(red count, green count)
- A global sum gives the total number of particles in each element
- A displacement array is calculated locally
- Using the displacement array, the sorted particle list is filled
- Displacement is decremented for each particle placed in the sorted list
- Allows easy particle passing

Shape functions in logical space (p, q) [1]

$$N_1(p,q) = \frac{1}{4}(1-p)(1-q) \quad N_2(p,q) = \frac{1}{4}(1+p)(1-q)$$
$$N_3(p,q) = \frac{1}{4}(1+p)(1+q) \quad N_4(p,q) = \frac{1}{4}(1-p)(1+q)$$





where $-1 \leq p \leq 1$ and $-1 \leq q \leq 1$

shape function used for gather and scatter:

$$A_p = \sum_{i=1}^{4} N_i A_i \quad A_j = \sum_e \sum_p N_j A_p J_p$$

[1] Alejandro Allievi and Rodolfo Bermejo, JCP, 132, (1997)

Know particle's (R, Z), need to know (p, q)

- Search algorithm: If $-1 \leq p \leq 1$ and $-1 \leq q \leq 1$ is true, then the particle is in this element
- (p, q) are needed for particle deposition and field evaluation
- FE representation for (R, Z) need to be inverted

$$R = \sum_{i=1}^{4} R_i N_i(p, q), \quad Z = \sum_{i=1}^{4} Z_i N_i(p, q),$$

- Use Newton method to solve for (p, q) given (R, Z)

$$\left\{ \begin{array}{c} p^{k+1} \\ q^{k+1} \end{array} \right\} = \left\{ \begin{array}{c} p^{k} \\ q^{k} \end{array} \right\} + A(p_k, q_k) \left\{ \begin{array}{c} R - R^{k} \\ Z - Z^{k} \end{array} \right\}$$

- Iterate until $\sqrt{(R - R^k)^2 + (Z - Z^k)^2} < \epsilon$

The matrix on the right hand side is the inverse of the Jacobian relating the logical coordinates to the real coordinates

$$A(p_k, q_k) \equiv \frac{1}{\Delta^k} \begin{bmatrix} b_2 + b_3 p^k & -a_2 - a_3 p^k \\ -b_1 - b_3 q^k & a_1 + a + 3q^k \end{bmatrix} = \begin{pmatrix} \frac{\partial R}{\partial p} & \frac{\partial R}{\partial q} \\ \frac{\partial Z}{\partial p} & \frac{\partial Z}{\partial q} \end{pmatrix}^{-1}$$

$$a_1 = \frac{1}{4}(R_2 - R_1 + R_3 - R_4), \quad b_1 = \frac{1}{4}(Z_2 - Z_1 + Z_3 - Z_4),$$
$$a_2 = \frac{1}{4}(R_3 - R_1 + R_4 - R_2), \quad b_2 = \frac{1}{4}(Z_3 - Z_1 + Z_4 - Z_2),$$
$$a_3 = \frac{1}{4}(R_1 - R_2 + R_3 - R_4), \quad b_3 = \frac{1}{4}(Z_1 - Z_2 + Z_3 - Z_4),$$

$$\Delta^k = (a_1 b_2 - a_2 b_1) + (a_1 b_3 - a_3 b_1) p^k + (a_3 b_2 - a_2 b_3) q^k$$

Evaluating ∇B terms in eqs. of motion

Need to calculate $\frac{\partial B}{\partial R}$, $\frac{\partial B}{\partial Z}$, e.g.

$$\frac{\partial B}{\partial R} = \sum_{i=1}^{4} B_i \frac{\partial N_i(p,q)}{\partial R} = \sum_{i=1}^{4} B_i \left(\frac{\partial N_i}{\partial p}\frac{\partial p}{\partial R} + \frac{\partial N_i}{\partial q}\frac{\partial q}{\partial R} \right)$$

$$\begin{bmatrix} \frac{\partial p}{\partial R} & \frac{\partial p}{\partial Z} \\ \frac{\partial q}{\partial R} & \frac{\partial q}{\partial Z} \end{bmatrix} = \begin{bmatrix} \frac{\partial R}{\partial p} & \frac{\partial R}{\partial q} \\ \frac{\partial Z}{\partial p} & \frac{\partial Z}{\partial q} \end{bmatrix}^{-1}$$

The right hand matrix is easy to compute from

$$R = \sum_{i=1}^{4} R_i N_i(p,q)$$

$$\frac{\partial R}{\partial p} = \sum_{i=1}^{4} R_i \frac{\partial N_i}{\partial p}, \qquad \frac{\partial R}{\partial q} = \sum_{i=1}^{4} R_i \frac{\partial N_i}{\partial q}$$
$$\frac{\partial Z}{\partial p} = \sum_{i=1}^{4} Z_i \frac{\partial N_i}{\partial p}, \qquad \frac{\partial Z}{\partial q} = \sum_{i=1}^{4} Z_i \frac{\partial N_i}{\partial q}$$