Particles, NIMlite, and some other stuff

Charlson C. Kim

Plasma Science and Innovation Center
University of Washington, Seattle

Particles

- unify drift kinetic and Lorentz force particles
- common data type, initialization, field evaluation, search/sort, deposition
- unique pusher, moment calculator

>cd $(NIMHOME)/particle; ls -1 *.f

```
part_type_mod.f
particle_init.f
particle_interface.f
boris_particle.f
drift_particle.f
particle_deposit.f
particle_field.f
particle_find.f
```

part_type_mod.f

```
      TYPE :: particle
        REAL(r8), DIMENSION(0:2) :: R,Z,P,v1,v2,v3,w
        REAL(r8) :: g0(0:1),NS(0:3),GNS(1:2,0:3)
        INTEGER(i4) :: ix,iy,id,pass
      END TYPE particle
c
      TYPE :: species
        CHARACTER(10) :: name
        INTEGER(i4) :: number
        INTEGER(i4) :: MPI_PART,packsz,ntot
        REAL(r8) :: mass,charge,P0,en032,pres0
        TYPE(particle), DIMENSION(:), POINTER :: part
      END TYPE species
```

part_type_mod.f cont.

define MPI data type `MPI_PART` for `TYPE :: particle`

```
SUBROUTINE mpi_part_type_init(packsz,MPI_PART)
```

particle diagnostic routines

```
SUBROUTINE part_energy(fspec)
SUBROUTINE trace_part_bin(fpart,fnm,is)
SUBROUTINE part_phase(fspec)
```

particle_interface.f

routines that directly are directly called from nimrod

```
SUBROUTINE particle_load
SUBROUTINE adv_particle(dtp)
```

scale MHD pressure to preserve force balance

```
SUBROUTINE apl_betaf
```

add hot particle pressure to `vrhs`

```
SUBROUTINE phot_add(vrb,vint,bgr,alp,dpdr,dpdz,ind)
```

particle_init.f, particle_find

several options for loading for both spatial and velocity

```
c**particle_init.f:
      SUBROUTINE spatial_load3(fspec)
      SUBROUTINE velocity_load(fspec,distro)
```

- spatial load proportional to any field possible—(n_{eq}, p_{eq}) implemented
- velocity allows for mono-energetic, Maxwellian, slowing down
  - velocity load in Cartesian coordinates, then projected to appropriately

```
c**particle_find.f:
      SUBROUTINE part_find_init(pnm,ntt)
      SUBROUTINE part_find(pf,fspec,npl)
```

particle_deposit.f

```
SUBROUTINE phot_init
```

called at the beginning of orbit averaging

```
SUBROUTINE phot_zero
```

```
SUBROUTINE phot_cal(pcf,fnm,fpart)
```

called at the end of orbit averaging

```
SUBROUTINE phot_cal_seam
```

particle_field.f

```
SUBROUTINE field_init(nmal)
```

called at the beginning of orbit averaging, resorts field arrays putting Fourier index first

```
SUBROUTINE collapse_field
SUBROUTINE fourier_collect(nr,lphi,nq,f_coef,comp)
```

evaluation of field at particle location

```
SUBROUTINE befield(pf,fnm,fpart)
```

Linear Simulations of Tearing Modes in a RFP

- alpha model equilibrium $\nabla \times \mathbf{B} = \mu \mathbf{B} \quad \mu = 2\Theta \left[1 - \left(\frac{r}{a} \right)^{\alpha_0} \right]$
- parameters for straight cylinder

  $a = .5\text{m},\ B_0 = .3\text{T},\ \Theta = 1.75,\ \alpha_0 = 3,$

  $S = 1.e4,\ ka = 2,\ \gamma\tau_A = 1.3e-3$
- Boris push with orbit averaging to accommodates disparate time scales
- energetic ion density profile $\propto \exp\left[-\left(\frac{r}{0.45a} \right)^2 \right]$
- initialize with mono-energetic particles $\delta(\mathbf{v}_\perp - \mathbf{v}_0)$, only $\mathbf{v} \times \delta\mathbf{B}$ in weight equation
- use **only** perpendicular pressure for comparison with theory
- subcyling of particles and orbit average particle pressure

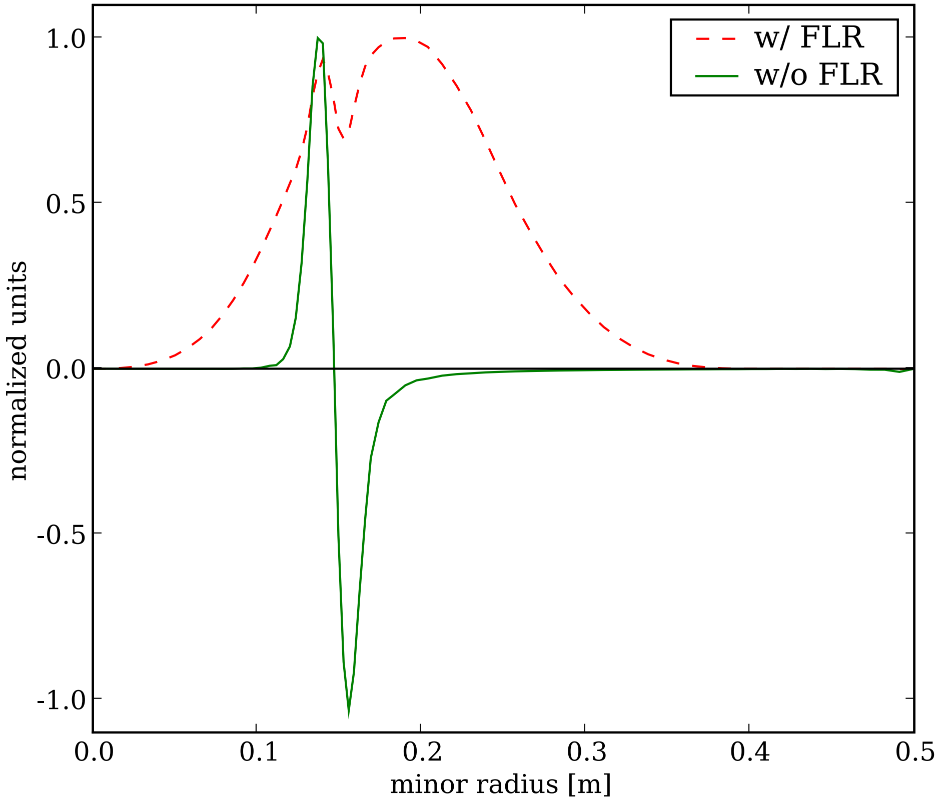
FLR Stabilization of RFP Tearing Mode

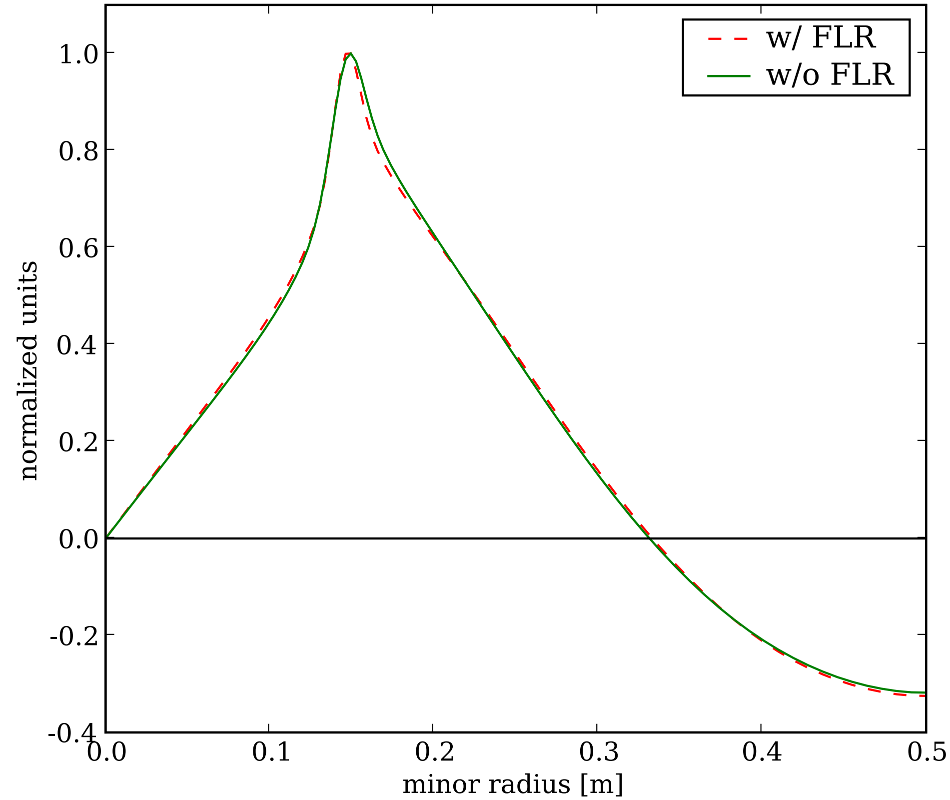
- stabilization with increasing $\mathbf{v}_\perp$

v_0(m/s)	L/a	$\gamma\tau_A$
base case	-	1.3×10^{-3}
1.0×10^6	.14	1.0×10^{-3}
1.5×10^6	.21	5.4×10^{-4}
2.0×10^6	.28	1.5×10^{-4}
2.5×10^6	.35	5.1×10^{-5}

- stabilization at $L/a \simeq 1/3$, where L is the Larmor diameter

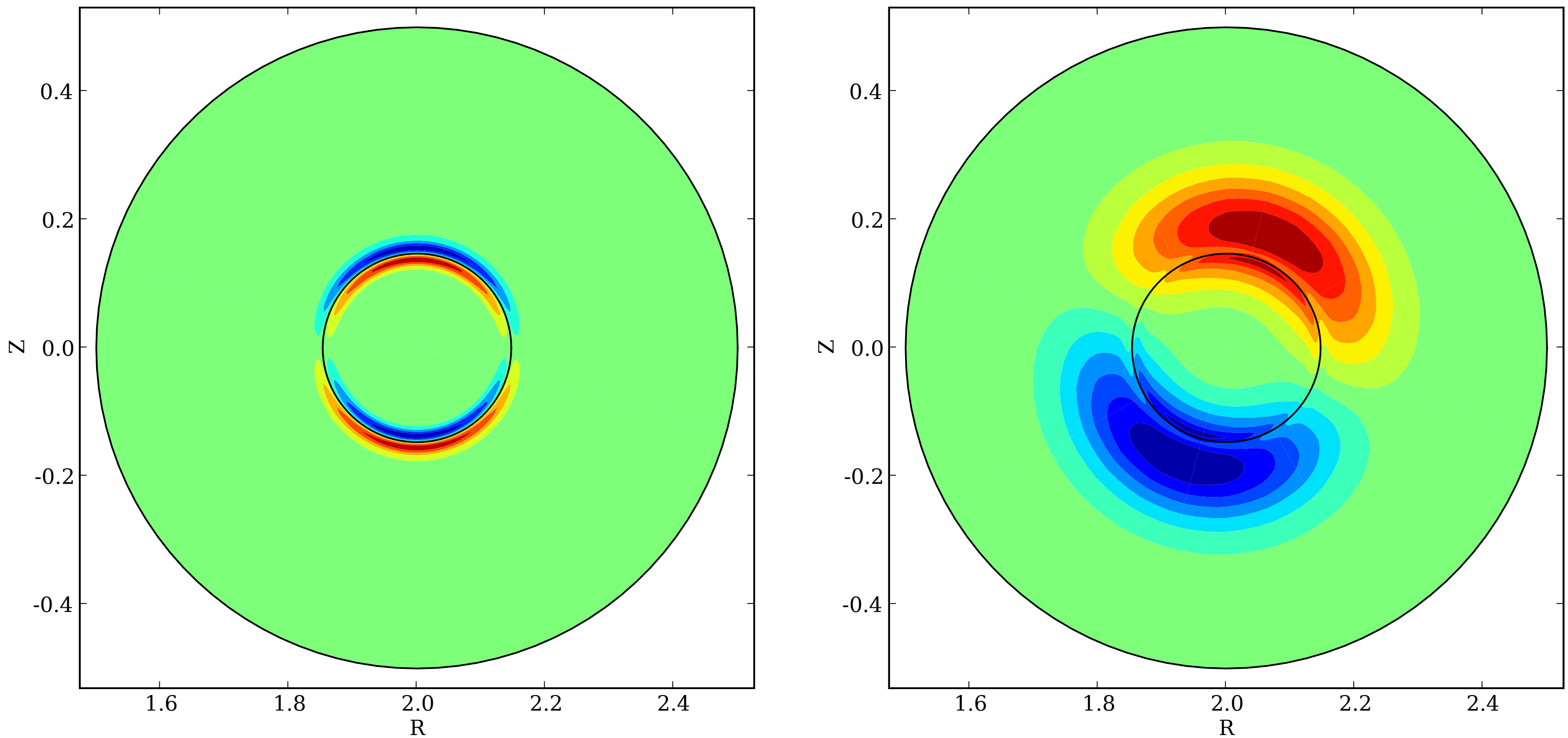
FLR Broadens Tangential Velocity Eigenmode Structure





- tangential velocity eigenmode substantially altered (left)
- magnetic eigenmode unaltered (right)

Comparison of V_ϕ Eigenmode



- inner circle shows resonance surface

NIMlite

- initially an attempt to better understand NIMROD by deconstructing to a smaller version
- the bulk of NIMROD is underlying "foundation" subroutines
- present incarnation is a reorganization and cosmetic changes
  - eliminate tblocks
  - almost eliminate bicubes
  - eliminate redundant compiles/links
  - move data specific I/O (from `dump.f`) to data definition files
  - reduce over used `USE` statements
  - break `nimrod.f` and `integrands.f` to `advance_*.f`

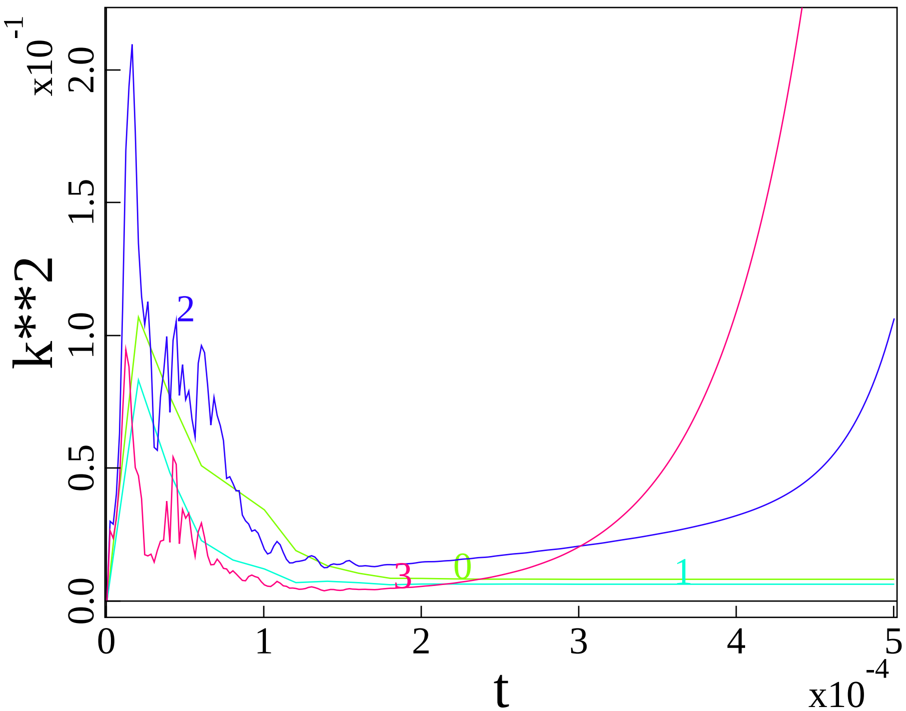
NIMlite cont.

- eliminate `INTERFACE` blocks
- build up library of function
  - for `lagr_quad`,`qp_type`,`matrix_type`
  - much of this is already there
- develop surface term subroutines
- solve for constitutive fields, e.g. $(p, \mathbf{E}, \mathbf{J})$
  - use of local evaluation is questionable
  - easier to apply boundary conditions

... and some other stuff.

- noise infestation seen at small dt, initially in δn
- worse for larger $\kappa_{\rm divb}$!!!

div(B) diagnostic vs. t



	dt	$\kappa_{\rm divb}$
0	1.0×10^{-6}	10.
1	1.0×10^{-6}	100.
2	1.0×10^{-7}	100.
3	1.0×10^{-7}	1000.

- tracked down to local calculation of p in `vrhs` routine
- call to `p_from_nt` and projecting to `qp` eliminates this