A new energetic particle module in M3D-C1 with GPU acceleration

Chang Liu

Princeton Plasma Physics Laboratory

CTTS Meeting

April 29 2020

Motivation of developing particle module in M3D-C1

- We want to have the same capability of the kinetic module of M3D-K code in M3D-C1, to study the interaction between energetic ions and MHD activities (Alfvén waves, kink/tearing modes etc).
- With more advanced finite-element representation and implicit time advance method, M3D-C1 can study the nonlinear problem with larger timestep and save computation time.
    - Explicit particle pushing can be accelerated using modern HPC with GPU, like in PIC codes.

Overview

1. Implementation of kinetic physics in M3D-C1 framework
2. GPU acceleration of particle pushing
3. Simulation results and benchmark with other codes
4. Summary

Outline

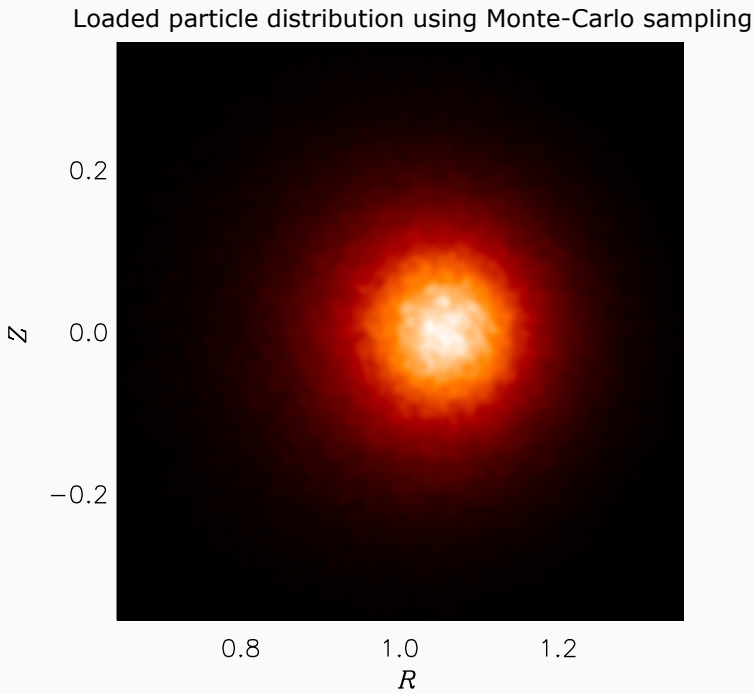
1. Implementation of kinetic physics in M3D-C1 framework
2. GPU acceleration of particle pushing
3. Simulation results and benchmark with other codes
4. Summary

Particles loading and initialization

Two ways to load particles

- In M3D-K, particles are loaded homogeneously in both real and momentum space. Each particle will then carry a f_0 that will appear in the weight equation.
  - It is OK for slowing-down distribution, but very expensive for Maxwellian distribution.
- In NIMROD, particles are loaded following f_0 through a Monte-Carlo sampling method.

- Some codes (like GTS) use homogeneous loading in configuration space, and Monte Carlo sampling in momentum space.
- We have tried all the methods and obtained similar result. The Monte-Carlo sampling method is the most efficient.



Particle pushing

$$\frac{d\mathbf{X}}{dt} = \frac{1}{B^*}\left(v_{\parallel}\mathbf{B}^* - \mathbf{b}\times\mathbf{E}^*\right)$$

$$m\frac{d\mathbf{v}_{\parallel}}{dt} = \frac{q}{B^*}\mathbf{B}^*\cdot\mathbf{E}^*$$

$$\mathbf{B}^* = \mathbf{B} + \frac{mv_{\parallel}}{e}\nabla\times\mathbf{b}, \qquad B^* = \mathbf{B}^*\cdot\mathbf{b}$$

$$\mathbf{E}^* = \mathbf{E} - \frac{mv_{\parallel}}{e}\frac{\partial\mathbf{b}}{\partial t} - \frac{\mu}{q}\nabla B$$

- Particle markers are advanced using 4th order Runge-Kutta.
- In guiding center mode, the fields are evaluated at the guiding center. In gyrokinetic mode, the fields are calculated using 4-point averaging along the gyro orbit.
- In the linear run, markers follow drift kinetic equations with equilibrium B fields only.
  - We have tested the energy and P_ϕ conservation in the long-term run. The error is less than 10^{-4} for 1ms, and it is not accumulating.

Weight evolution

$$\delta f \text{ method} \qquad \frac{d\delta f}{dt} = -\delta\dot{\mathbf{z}} \cdot \frac{\partial f_0}{\partial \mathbf{z}}$$

$$\frac{dw}{dt} = \frac{\delta\dot{f}}{f} = \frac{1-w}{f_0}\left(-\delta\mathbf{v}\cdot\nabla f_0 - \dot{\epsilon}\partial_\epsilon f_0\right)$$

- Here we use energy derivative ($\dot{\epsilon}$) to calculate weight evolution, which is not consistent with the guiding center equation ($\dot{v_{||}}$) but easier to implement. Will change to $\dot{v_{||}}$ in future.
- The change of Jacobian ($B^*_{||}$) can be taken into account by introducing a new weight $d = w + (1-w)\delta B^*_{||}/B^*_{||0}$, like in Belova (1997).

E.V. Belova, R.E. Denton, and A.A. Chan, J. Comp. Phys. 136, 324 (1997).

Particle deposition

- Parallel and perpendicular pressure are calculated from particles using δ-function deposition

$$\int \nu P_{||} g d\mathbf{x} = \sum_i m v_{i,||}^2 \nu(\mathbf{x}_i)$$

$$\int \nu P_{\perp} g d\mathbf{x} = \sum_i \mu_i B(\mathbf{x}_i) \nu(\mathbf{x}_i)$$

  - We can add a small diffusion to the obtained $P_{||}$ and $P_{\perp}$ to reduce noise, but it will break the energy conservation of the coupling scheme.
- The calculated $P_{||}$ and $P_{\perp}$ can be used for both pressure and current coupling to MHD equations.
  - Pressure coupling

$$\nabla \cdot \mathbf{P} = \nabla P_{\perp} + \nabla \cdot \left(P_{||} - P_{\perp}\right) \mathbf{bb}$$

  - Current coupling

$$\mathbf{J}_{hot} \times \mathbf{B} = \frac{P_{||}}{B^2} \mathbf{b} \times \nabla \times \mathbf{b} - \frac{P_{\perp}}{B^2} \nabla_{\perp} \ln B - \nabla \times \left(\frac{P_{\perp}}{B} \mathbf{b} \right) \times \mathbf{B}$$

    - This does not include current due to “moving dipole” effect.

J.W. Burby and C. Tronci, Plasma Phys. Control. Fusion 59, 045013 (2017).

Outline

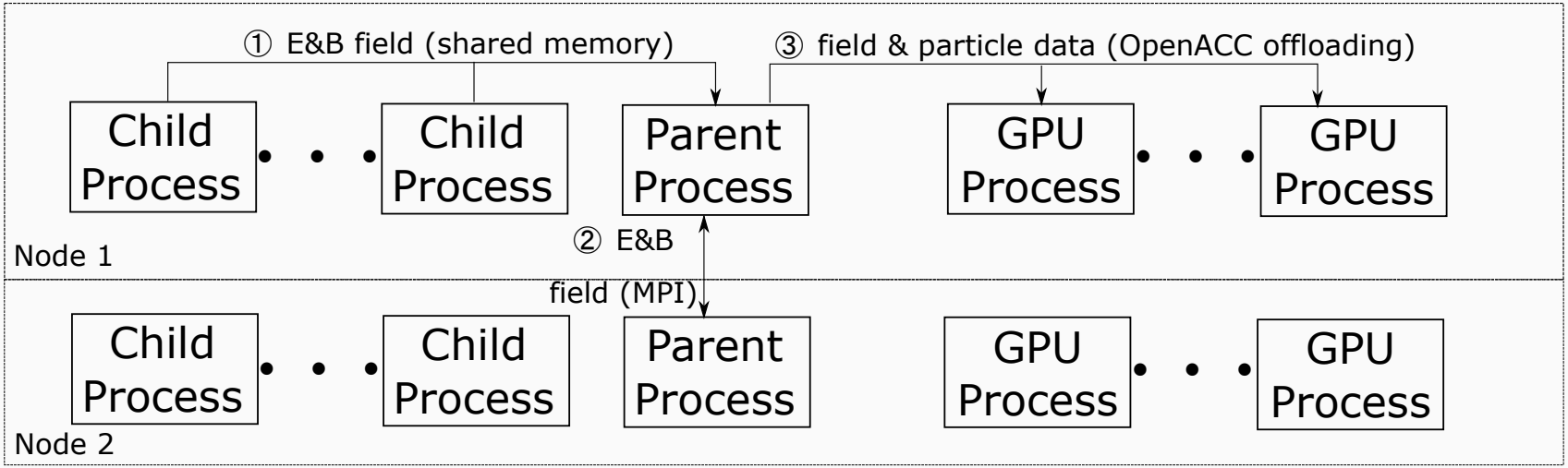
The importance of migrating code to GPU

- Many of the newly built supercomputers utilize GPU to reach high computation power.
  - In Traverse, a new cluster built by PPPL and Princeton University, 97% of computing power comes from GPU.
- GPU can be regarded as a co-processor with many cores and a shared memory.
  - Computation on a single GPU core is slower than on a single CPU core, especially for logical operations.
  - GPU should be used to do strongly parallel jobs with each job very simple, and particle pushing is indeed a suitable job.
- With the help of new API like OpenMP4 or OpenACC, it is now easier to migrate the existing code to run on GPUs.
  - Most of the migration work is related to communications between GPU and CPU (offloading), since they have separate memory.
  - Existing MPI structure of the code can complicate the work.

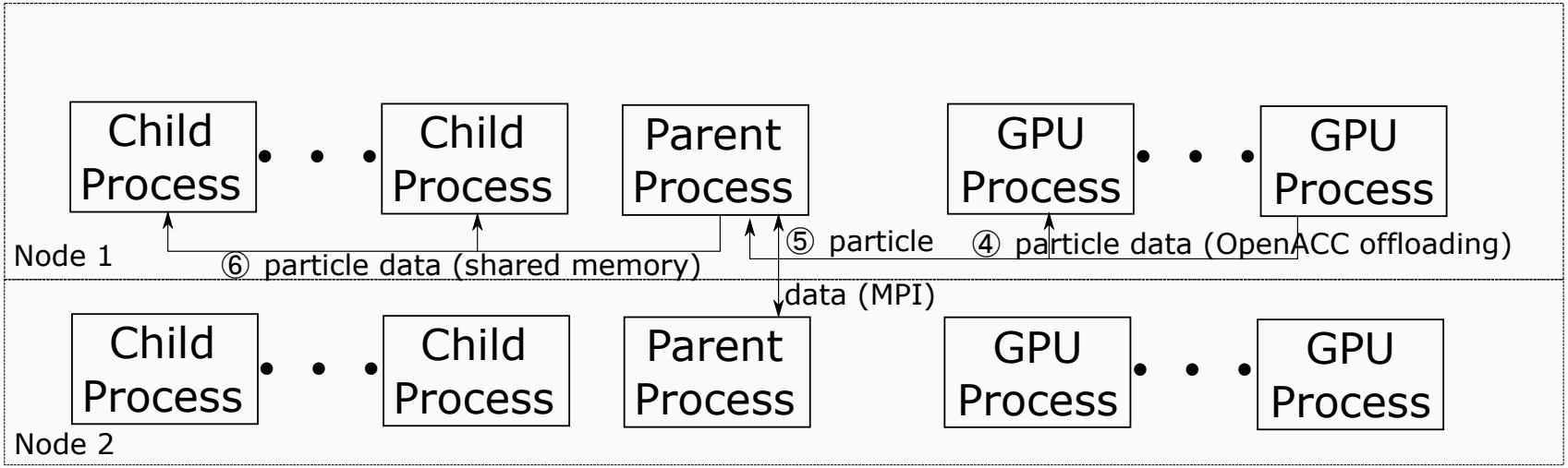
Combining distributed memory and shared memory

- Currently M3D-C1 use MPI processes for parallelization, with domain decomposition and distributed memory mode.
  - Each process only knows the information about a small subdomain of the whole 3D mesh.
- The most efficient way to push particles in GPU is to use particle-based data structure, and each particle is pushed independently.
  - This is more memory-consuming since every GPU needs the field information of the whole mesh. Fortunately for modern GPU with >16GB RAM, this is not a problem.
  - Previously we use a mesh-based data structure to store particle data. This leads a lot of communication due to particles particles moving from one mesh to the other.
- To incorporate the distributed-memory M3D-C1 and share-memory particle pushing, we exploit two methods for data sharing
  - Shared memory function (within one node) introduced in MPI-3.
  - MPI_Allgatherv between different nodes

Procedure and data flow in a particle pushing step



Procedure and data flow in a particle pushing step



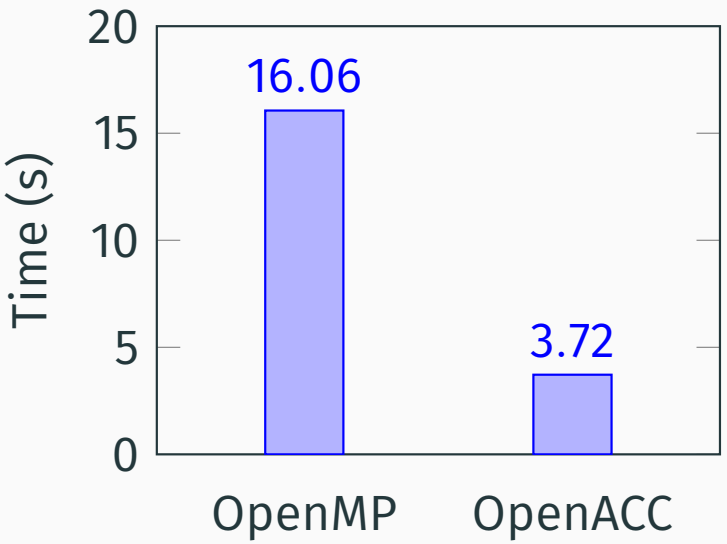
- Using GPU profiler, it is found that the data transfer time is < 0.1s for GPU offloading, which is ignorable compared to GPU computation time with subcycles.

Subcycles of particle pushing within MHD timestep

- Currently M3D-C1 can use timestep of tens of τ_A to simulate long-term phenomena, with the help of the advanced semi-implicit algorithm for velocity advance.
  - This is an improvement over the M3D code, which typically use much smaller timestep.
  - For particle pushing we use explicit RK4, thus the timestep is limited by particle speed.
- Here we use subcycles for particle pushing, which means that we push particle multiple times between two MHD timesteps.
  - GPU-CPU communications are only needed at the beginning and end of subcycles.
  - Fields are fixed during subcycles. This can be improved by utilizing information of time derivative of field.

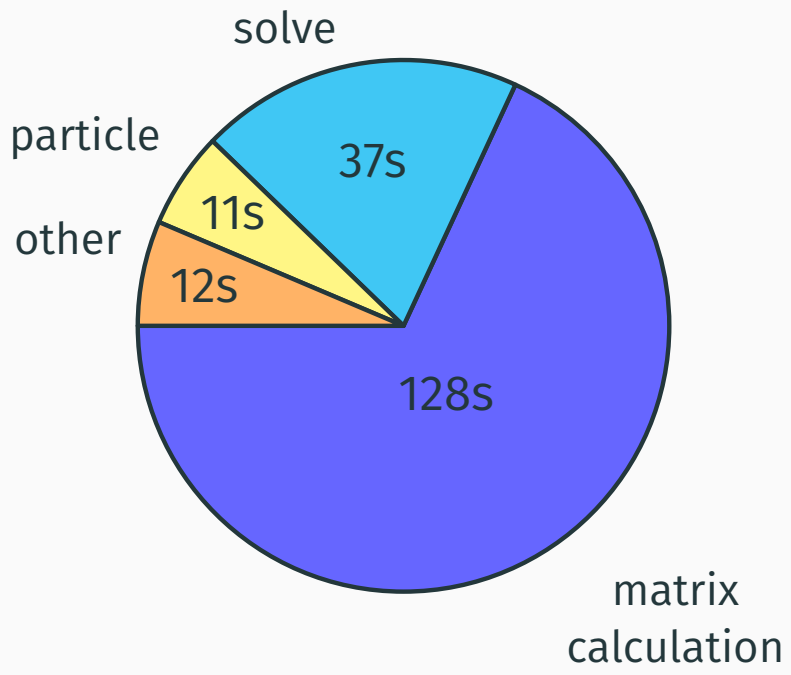
Performance benchmark of particle pushing on GPU and CPU

- The benchmark was done using the initial version of code, which only runs on one GPU. We compare it with OpenMP version running on Traverse CPU with 128 threads.
- According to the results, in current version (supporting multiple GPUs), we should get >16× speedup from GPU acceleration.
- Most of the time for particle pushing is spent in calculating the value of basis functions at each particle location, which requires a calculation of 5th order polynomials.



Linear subcycles in nonlinear MHD simulations

- In a nonlinear run of current version of code, most of the computation time is spent in matrix term calculation of MHD equations, especially evaluating the semi-implicit terms.
- To simulate the Alfvén wave excitation, we have to use MHD timesteps smaller than mode period, like $1\tau_A$ or less.
- To save time in matrix calculation, we keep using the same matrix for MHD evolution for several timesteps (~ 30), like in a linear run, and then recalculate the matrix and do the matrix factorization.
  - The particle current terms or pressure terms are added to equations as external momentum source, thus the wave-particle interaction is well treated.
  - However, the wave-wave interaction is absent.
  - The method can lead to numerical instabilities near the mode saturation. It is better to introduce self-adaptive timesteps.

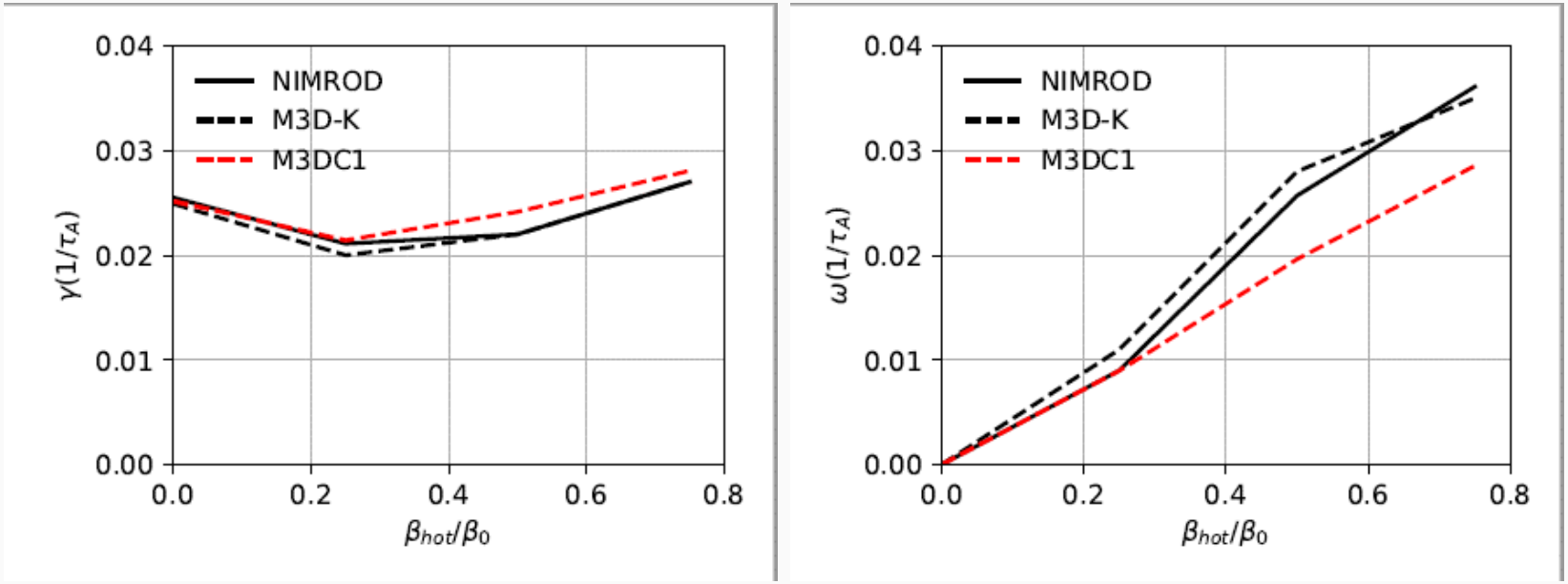


Outline

Fishbone simulation result agrees with M3D-K and NIMROD

$R/a = 2.8, \quad \beta_{total} = 0.08, \quad q_0 = 0.6, \quad q_a = 2.5$

$\hat{\rho}_h = v_0/(\Omega_h a) = 0.0125, v_0/v_A = 4$



- These results are obtained using pressure coupling scheme used in M3D-K and NIMROD. With current coupling, the growth rate increase significantly with β_h

G.Y. Fu, W. Park, H.R. Strauss, J. Breslau, J. Chen, S. Jardin, and L.E. Sugiyama, Phys. Plasmas 13, 052517 (2006).

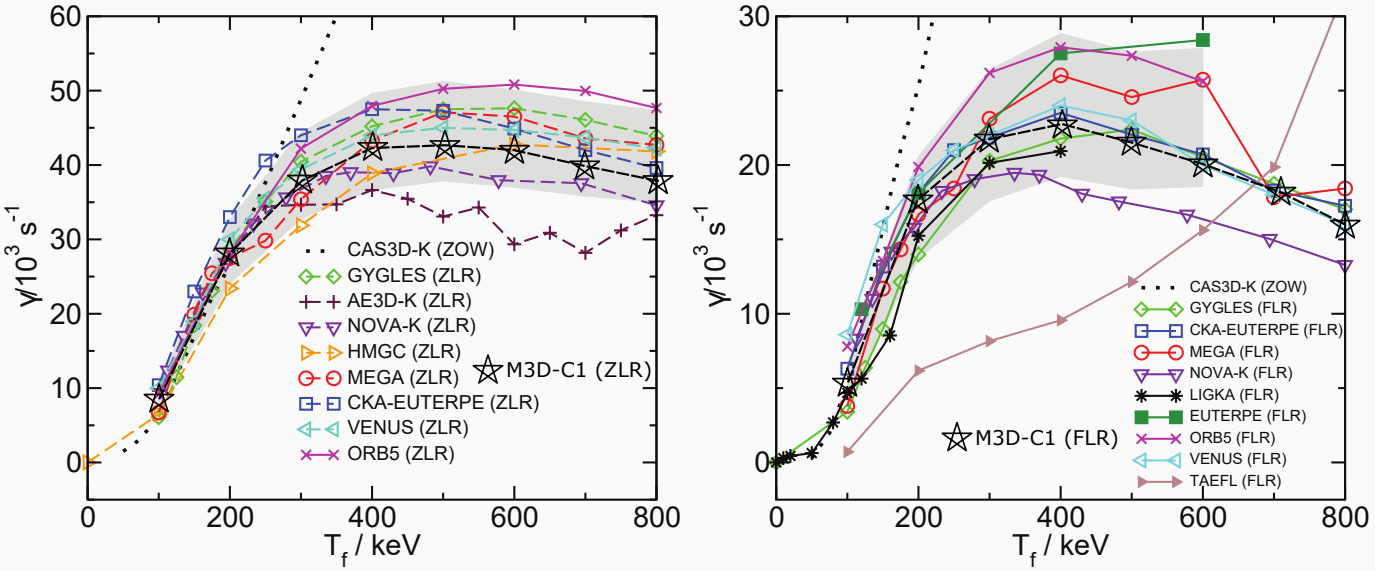
C.C. Kim and the NIMROD Team, Phys. Plasmas 15, 072507 (2008).

TAE linear simulation without and with FLR effects

- This is an ITPA collaborative effort to compare different codes and physical model. Several hybrid MHD, gyrokinetic and gyrofluid codes are benchmarked.

  $R/a = 10, \quad \beta \approx 0.2\%, \quad q = 1.71 + 0.16(r/a)^2$

  $n_f = c_3 \exp\left(-\frac{c_2}{c_1} \tanh \frac{\sqrt{s}-0.5}{c_2}\right)$



- In nonlinear simulations, an energetic particle driven acoustic-like mode is observed with growth rate of the same order.

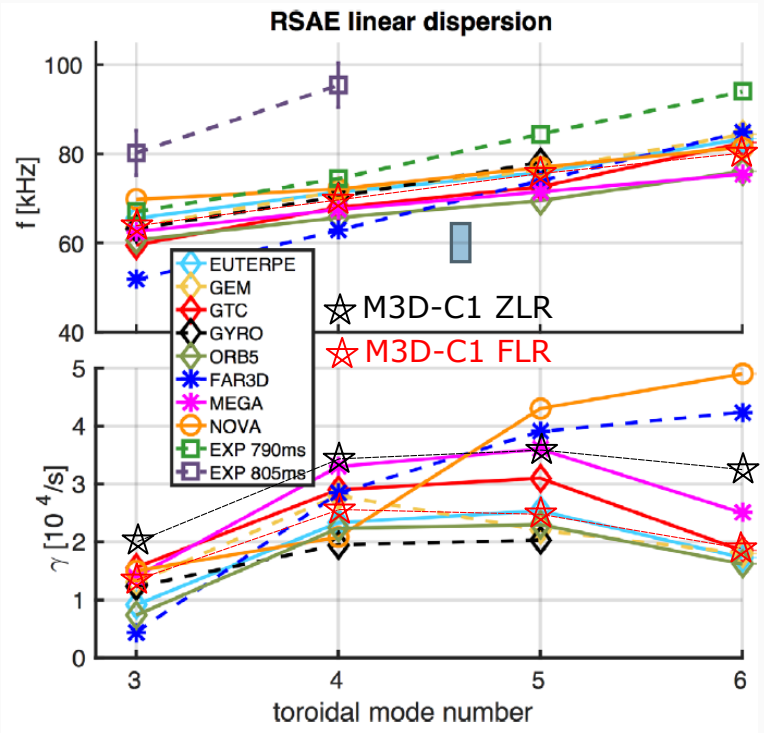
A. Könies, et al., Nucl. Fusion 58, 126027 (2018).

Simulation of RSAE driven in DIII-D tokamak

- Recently several MHD and gyrokinetic codes are employed to study the linear growth of reversed shear Alfvén eigenmode (RSAE) using DIII-D experimental parameters.

$R/a = 2.5, \quad q_{min} = 2.94, \quad B_0 = 2T, \quad n_0 = 3.29 \times 10^{13}\mathrm{cm}^{-3}, \quad T_e = 1.689\mathrm{keV}$
$n_f = 1.95 \times 10^{12}\mathrm{cm}^{-3}, \quad T_f = 24\mathrm{keV}$

- Including FLR effects leads to smaller mode growth rate, especially for high-k modes.
- We got almost the same results using pressure coupling or current coupling, meaning that the parallel dynamics are not important.
- Compressional effects ($\delta B_{||}$) are not important.

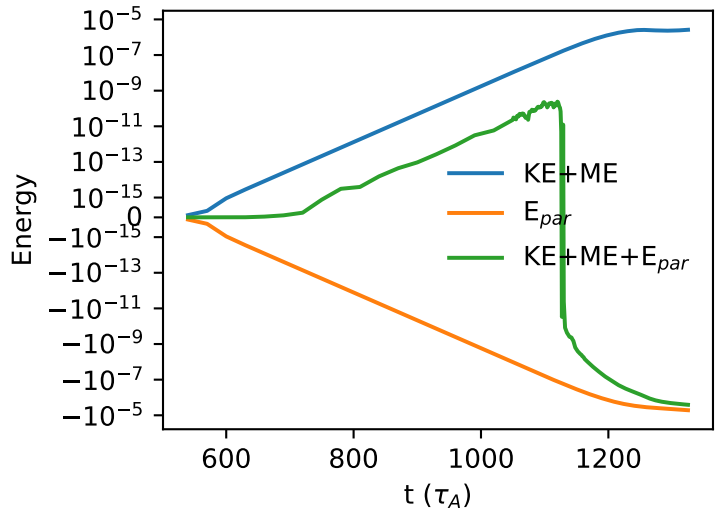


S. Taimourzadeh et al., Nucl. Fusion 59, 066006 (2019).

Energy conservation test

- We study the energy conservation in the nonlinear simulation of RSAE in DIII-D.
- Both the kinetic and magnetic energy are calculated using the perturbed field only, to reduce the noise coming from the equilibrium fields.

- Pressure is chosen to be very small.
- In energetic particle energy calculation, the contribution from full-f current is subtracted to reduce noise (Belova (1997))



- The total energy change is within 10% of MHD energy increase during linear growing stage, but in the saturation stage this error is significantly larger.
  - We think this is caused by the phase mixing related to continuum damping of RSAE, since high-k modes can be excited.

E.V. Belova, R.E. Denton, and A.A. Chan, J. Comp. Phys. 136, 324 (1997).

Outline

1. Implementation of kinetic physics in M3D-C1 framework
2. GPU acceleration of particle pushing
3. Simulation results and benchmark with other codes
4. Summary

Summary

- A new kinetic module coupled to M3D-C1 has been developed, which utilizes GPU for particle pushing, and can do both pressure coupling and current coupling.
- Linear benchmarks with other MHD and gyrokinetic codes are conducted, and good agreements are achieved.
  - The test of energy conservation is also successful in the linear stage.
- Future work
  - Continue working on nonlinear simulation of RSAE and benchmark with MEGA
  - Implement a full-orbit scheme for particle simulation, and use it to simulate high frequency Alfven waves like GAE and CAE.

Parallel pressure term in M3D-C1

$$
\begin{aligned}
\nabla \cdot (\alpha \mathbf{BB}) &= \mathbf{B} \cdot \nabla(\alpha \mathbf{B}) \\
&= \mathbf{BB} \cdot \nabla \alpha + \alpha \mathbf{B} \cdot \nabla \mathbf{B} \\
&= \mathbf{BB} \cdot \nabla \alpha + \frac{1}{2} \alpha \nabla B^2 - \alpha \mathbf{B} \times \nabla \times \mathbf{B}
\end{aligned}
$$

Parallel pressure term in M3D-C1

$$\mathbf{B} = \nabla\psi \times \nabla\phi - \nabla_\perp f' + F\nabla\phi$$

$$\begin{aligned}
\nu\nabla\varphi \cdot \nabla \times R^2\nabla \cdot (\alpha\mathbf{BB}) &= R^2\nabla_\perp\nu \times \nabla\varphi \cdot \nabla \cdot (\alpha\mathbf{BB}) \\
&= [\alpha,\psi](\nu,\psi) + \alpha' R^{-2}F(\nu,\psi) - (\alpha,f')(\nu,\psi) \\
&+ R^2[\alpha,\psi][\nu,f'] + \alpha' F[\nu,f'] - (\alpha,f')R^2[\nu,f'] \\
&+ \frac{1}{2}\alpha R^2[B^2,\nu] \\
&+ \alpha\Delta^*\psi[\nu,\psi] - \alpha\Delta^*\psi(\nu,f') + \alpha F[\nu,F^*] + \alpha F R^{-2}(\nu,\psi')
\end{aligned}$$

Parallel pressure term in M3D-C1

$$\nu R^2 \nabla\varphi \cdot \nabla \cdot (\alpha \mathbf{BB}) = \nu F[\alpha, \psi] + \nu F F \alpha' R^{-2} - \nu F(f', \alpha)$$

$$+ \nu \alpha B B'$$

$$- \alpha\nu[\psi, F^*] - \alpha\nu \frac{1}{R^2}(\psi, \psi') - \alpha\nu(f', F^*) - \alpha\nu[\psi', f']$$

Parallel pressure term in M3D-C1

$$
\begin{aligned}
\nu\nabla_\perp \cdot \left[R^{-2}\nabla\cdot(\alpha\mathbf{BB})\right] &= -\nabla_\perp\nu\cdot\left[R^{-2}\nabla\cdot(\alpha\mathbf{BB})\right] \\
&= -R^{-2}[\alpha,\psi][\nu,\psi] - \alpha' R^{-4}F[\nu,\psi] + (\alpha,f')R^{-2}[\nu,\psi] \\
&+ (\nu,f')R^{-2}[\alpha,\psi] + (\nu,f')R^{-4}\alpha' F - (\nu,f')R^{-2}(\alpha,f') \\
&- \frac{1}{2}\alpha R^{-2}(\nu,B^2)) \\
&+ R^{-4}\alpha\Delta^*\psi(\nu,\psi) + R^{-2}\alpha\Delta^*\psi[\nu,f'] \\
&+ FR^{-4}\alpha(\nu,F^*) + FR^{-4}\alpha[\psi',\nu]
\end{aligned}
$$

Parallel pressure term in M3D-C1

$$\nabla p = \frac{\mathbf{BB}}{B^2} \cdot \nabla p + (1 - \frac{\mathbf{BB}}{B^2}) \nabla p$$

$$\begin{aligned}
\nabla \cdot (\frac{p}{B^2}\mathbf{BB}) &= \mathbf{B} \cdot \nabla (\frac{p}{B^2}\mathbf{B}) \\
&= \mathbf{BB} \cdot \nabla \frac{p}{B^2} + \frac{p}{B^2}\mathbf{B} \cdot \nabla \mathbf{B} \\
&= \mathbf{BB} \cdot \nabla \frac{p}{B^2} + \frac{1}{2}\frac{p}{B^2}\nabla B^2 - \frac{p}{B^2}\mathbf{B} \times \nabla \times \mathbf{B} \\
&= \left(\frac{\mathbf{BB}}{B^2} \cdot \nabla p + \mathbf{BB} p \cdot \nabla \frac{1}{B^2} - \frac{1}{2} p \mathbf{BB} \cdot \nabla \frac{1}{B^2} \right) \\
&+ \left(-\frac{1}{2} p (B^2 - \mathbf{BB}) \cdot \nabla \frac{1}{B^2} + \frac{p}{B^2} \mathbf{J} \times \mathbf{B} \right)
\end{aligned}$$