ÖAW AUSTRIAN ACADEMY OF SCIENCES

IWF SPACE RESEARCH INSTITUTE

Recent advances in numerical algorithms for particle-in-cell simulations

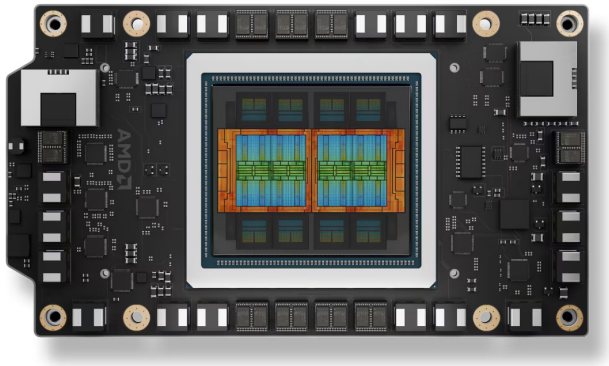
Seiji ZENITANI

Space Research Institute, Graz, AUSTRIA

T. Umeda (Hokkaido U.), T.-N. Kato (Rikkyo U.)

Two big issues in supercomputing

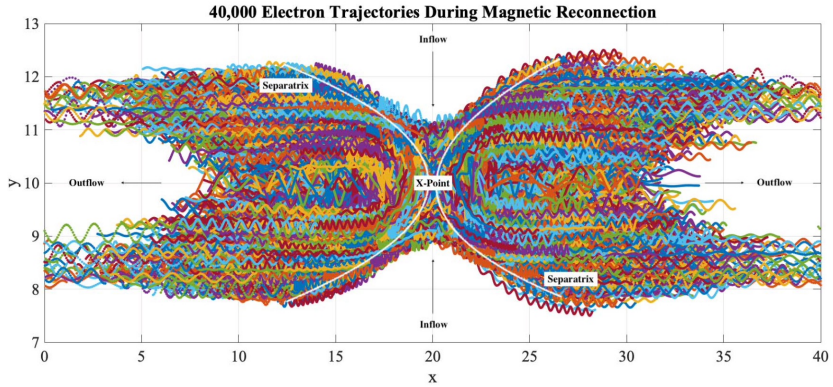
- 1. Accelerator
  - GPU (MN-core, NEC Aurora…)
  - Excellent performance-per-Watt
  - GPU-friendly algorithms are needed.

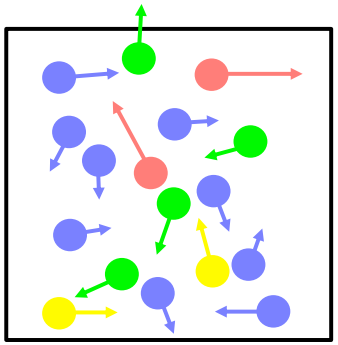


AMD MI350X

- 2. Arithmetically intense, high-accuracy algorithm
  - Computing cores are usually idle, waiting for data
  - Higher-accuracy algorithms are demanded to produce better results from a limited amount of data.

Particle-in-cell (PIC) simulation





Markidis+ 2020

1. Particle loader

2. Particle integrator

$$\boldsymbol{x}_n \rightarrow \boldsymbol{x}_{n+1}$$

$$\boldsymbol{v}_{n+\frac{1}{2}} \rightarrow \boldsymbol{v}_{n+\frac{3}{2}}$$

3. Charge deposition

$$x_n \rightarrow \overline{\rho}_{c,n}$$

$$\boldsymbol{v}_{n+\frac{1}{2}} \rightarrow \overline{\boldsymbol{j}}_{n+\frac{1}{2}}$$

4. Field solver

$$\boldsymbol{E}_n \rightarrow \boldsymbol{E}_{n+1}$$

$$\boldsymbol{B}_n \rightarrow \boldsymbol{B}_{n+1}$$

5. Field interpolation

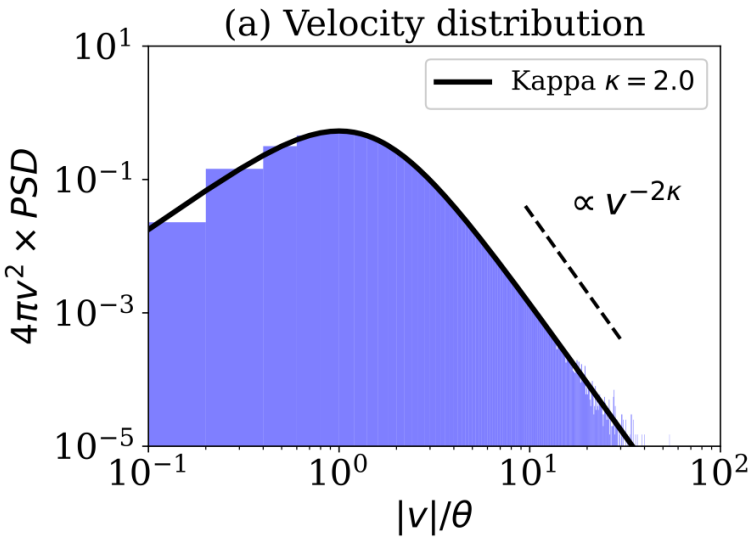
$$\boldsymbol{E}'_n$$

$$\boldsymbol{B}'_n$$

1. GPU-friendly Kappa generator

2. High-accuracy particle solver

1. Kappa generator



- Kappa distribution

$$f(\boldsymbol{v})d^3v = \frac{N_\kappa}{(\pi\kappa\theta^2)^{3/2}}\frac{\Gamma(\kappa+1)}{\Gamma(\kappa-1/2)}\left(1+\frac{\boldsymbol{v}^2}{\kappa\theta^2}\right)^{-(\kappa+1)}d^3v$$

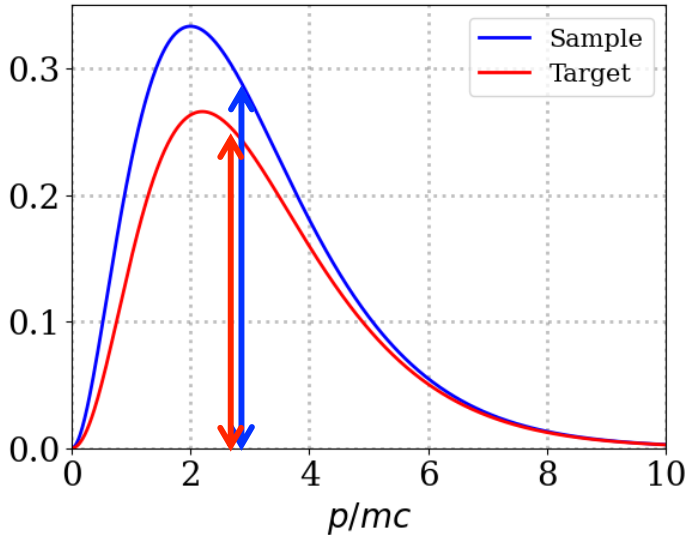
  - Maxwellian-like core + power-law tail

- Random number generation methods
  - Standard method (t-generator)
  - Pareto method
  - Bailey method
  - Galton board method
  - NEW method

Rejection sampling

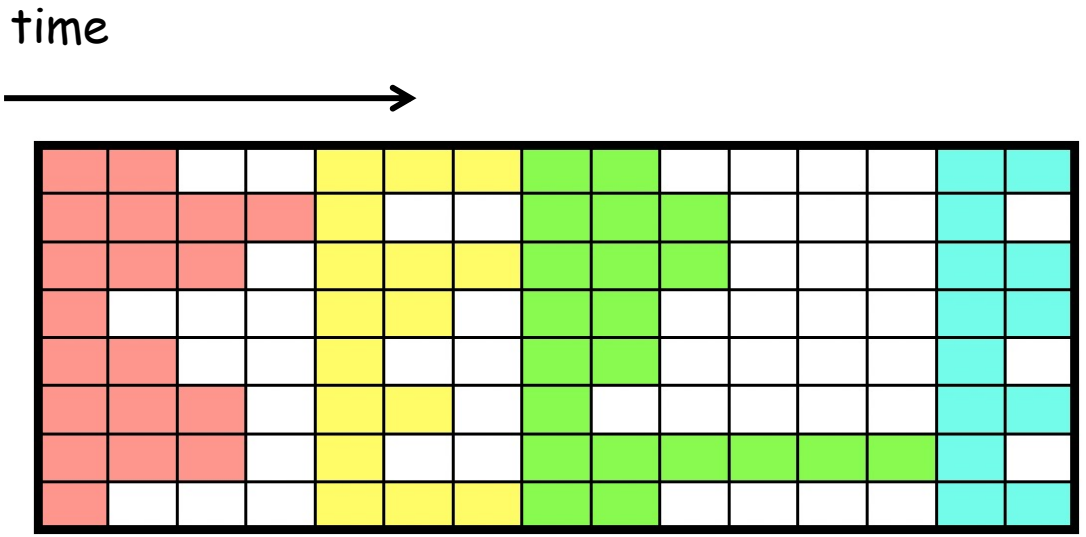
1D/2D only

GPUs are not good at rejection sampling



- Rejection sampling
  - The target distribution is taken out from the sampling distribution
  - In programming, it's an iteration loop
  - The iteration number differs from particle to particle

- GPU
  - A single instruction controls 32 threads simultaneously
  - All the threads wait for the slowest one
  - Inefficient



Approximate Kappa distribution

- Kappa distribution

$$f(\boldsymbol{v})d^3v = \frac{N_\kappa}{(\pi\kappa\theta^2)^{3/2}}\frac{\Gamma(\kappa+1)}{\Gamma(\kappa-1/2)}\left(1+\frac{\boldsymbol{v}^2}{\kappa\theta^2}\right)^{-(\kappa+1)}d^3v$$

- We have derived a good approximation

$$g(\boldsymbol{v})d^3v = \frac{3N_\kappa(a\theta^4+2bv^2\theta^2+bcv^4)}{4\pi v\theta^2\left(\theta^2+cv^2\right)^2} \times\sqrt{1-\left(1+\frac{(a\theta^2+bv^2)v^2}{\kappa^*(\theta^2+cv^2)\theta^2}\right)^{-\kappa^*}} \times\left(1+\frac{(a\theta^2+bv^2)v^2}{\kappa^*(\theta^2+cv^2)\theta^2}\right)^{-(\kappa^*+1)}d^3v$$

Random number generator
(there is no iteration loop)

$$\kappa^* \leftarrow \kappa - 1/2$$

$$a \leftarrow \frac{1}{\kappa}\left(\frac{2}{3B(3/2,\kappa^*)}\right)^{2/3}$$

$$c \leftarrow \frac{0.123\kappa^2 - 1.12\,\kappa + 2.56}{\kappa^2 - 7.89\,\kappa + 15.6}$$

$$b \leftarrow \left(\kappa^*\tfrac{3}{2}B(3/2,\kappa^*)\right)^{1/\kappa^*}\frac{\kappa^*}{\kappa}\,c$$

generate $U_1, U_2, U_3 \sim \mathrm{Uniform}(0,1)$

$$L \leftarrow -\kappa^*\left\{\left(1-U_1^{2/3}\right)^{-1/\kappa^*}-1\right\}$$

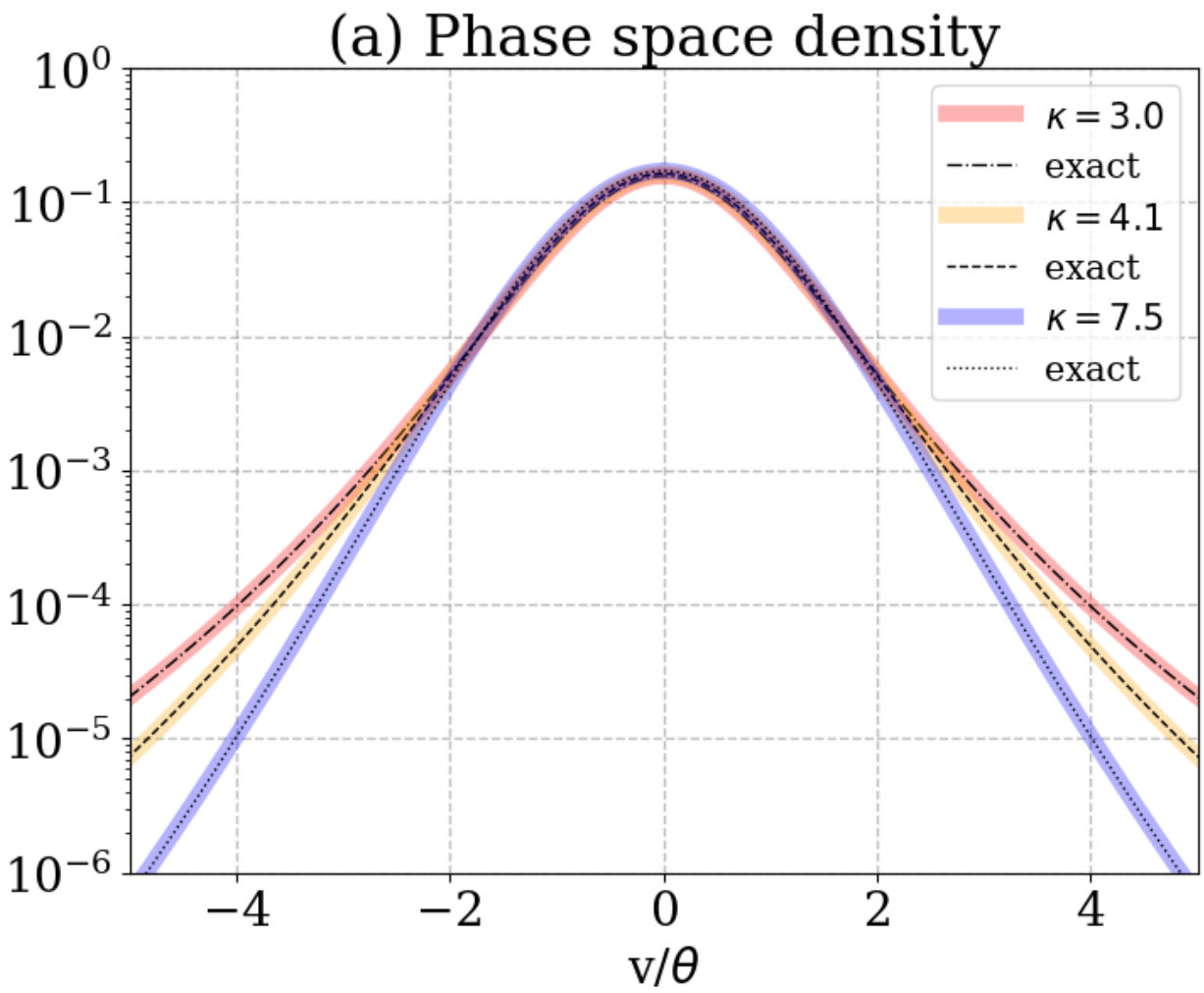
$$V \leftarrow \theta\sqrt{\frac{-2L}{(a+cL)+\sqrt{(a+cL)^2-4bL}}}$$

$$v_x \leftarrow V(2U_2-1)$$

$$v_y \leftarrow 2V\sqrt{U_2(1-U_2)}\cos(2\pi U_3)$$

$$v_z \leftarrow 2V\sqrt{U_2(1-U_2)}\sin(2\pi U_3)$$

How good is our approximation?



- δ(PSD) < PIC noise

$$N_p \lesssim 10^7$$

- Total number
  zero

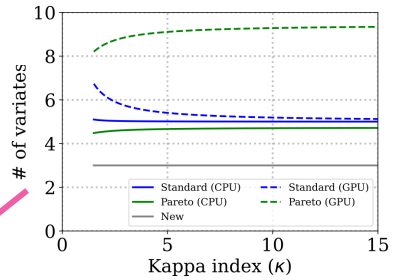
- Total energy

$$\frac{\Delta\mathcal{E}}{\mathcal{E}} \lesssim 10^{-2.5} \quad (\kappa \approx 4.1)$$

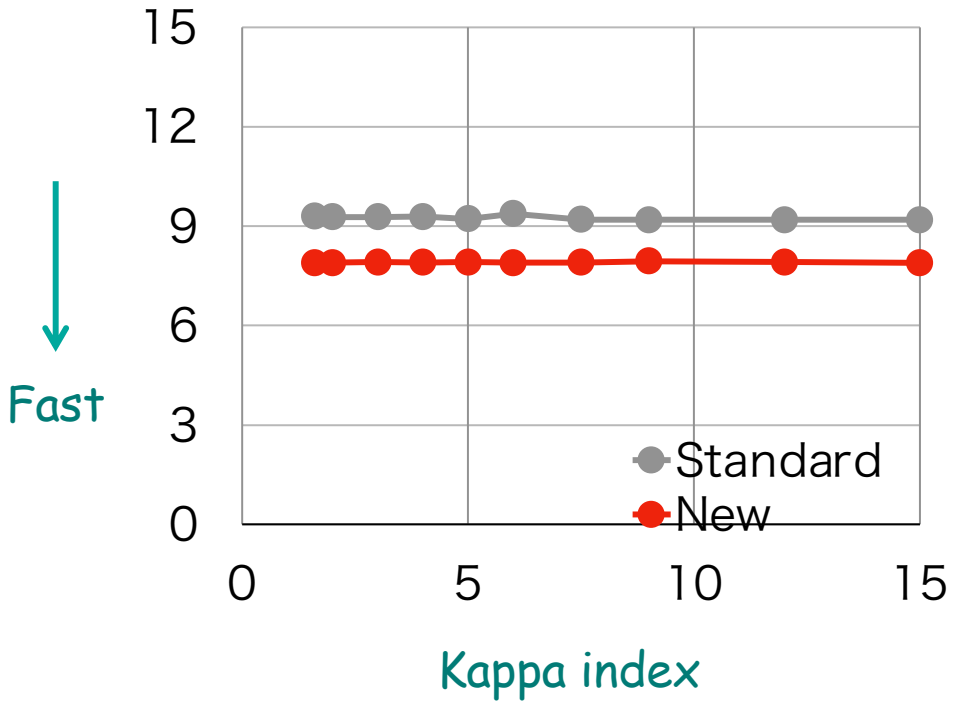
$$\frac{\Delta\mathcal{E}}{\mathcal{E}} \lesssim 10^{-3.5} \quad (\kappa \neq 4.1)$$

Numerical cost

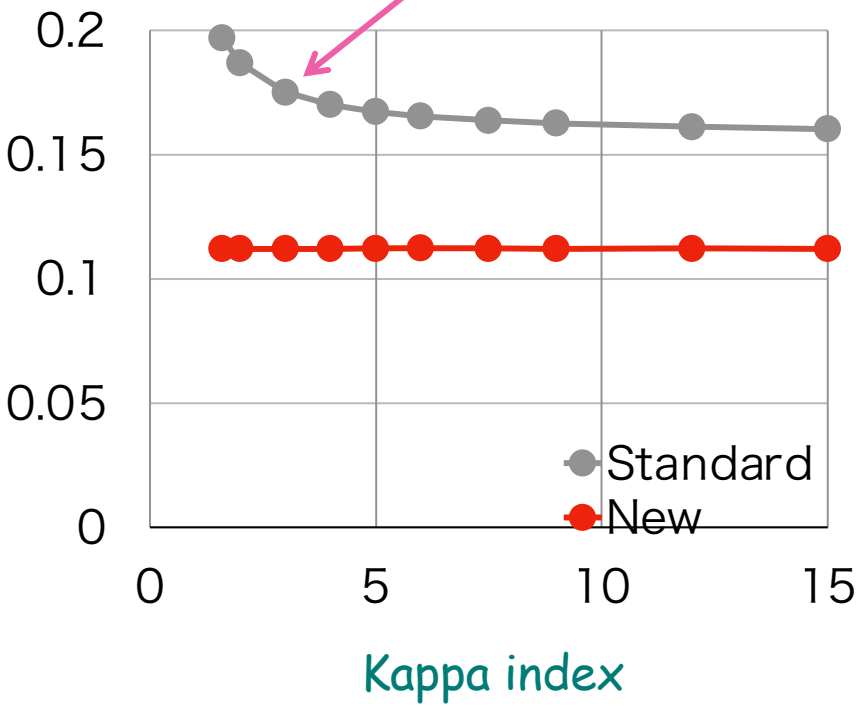
Theory



- CPU



- GPU



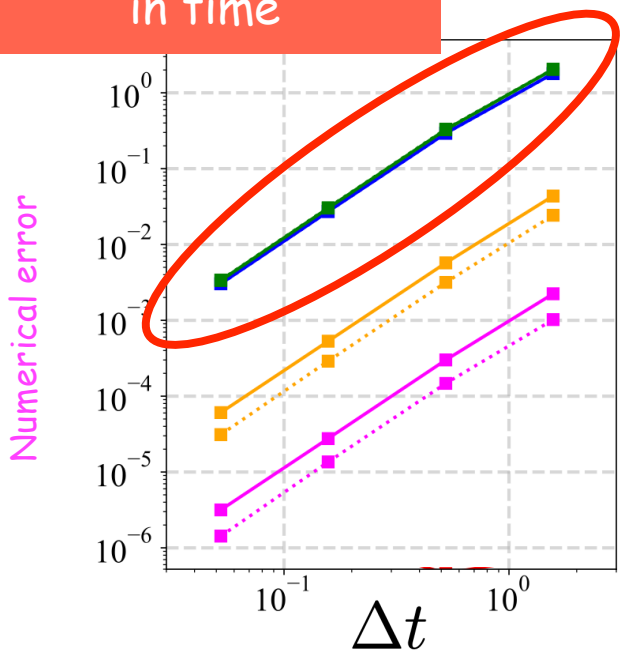
- New method runs constantly fast
- Standard method suffers from the loop problem on GPUs, in agreement with a theory (Ridley & Forget 2021)

2. Particle integrator (Boris solver)

$$\frac{\boldsymbol{x}^{t+\Delta t} - \boldsymbol{x}^{t}}{\Delta t} = \boldsymbol{v}^{t+\frac{1}{2}\Delta t}$$

$$m\,\frac{\boldsymbol{v}^{t+\frac{1}{2}\Delta t} - \boldsymbol{v}^{t-\frac{1}{2}\Delta t}}{\Delta t} = q\left(\boldsymbol{E}^{t} + \frac{\boldsymbol{v}^{t+\frac{1}{2}\Delta t} + \boldsymbol{v}^{t-\frac{1}{2}\Delta t}}{2} \times \boldsymbol{B}^{t}\right)$$

2nd-order accuracy in time



element E vector element B vector

$$\boldsymbol{e}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{E}, \quad \boldsymbol{t}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{B}$$

$$\boldsymbol{v}^{-} = \boldsymbol{v}^{t-\frac{1}{2}\Delta t} + \boldsymbol{e}_1$$

$$\boldsymbol{v}' = \boldsymbol{v}^{-} + \boldsymbol{v}^{-} \times \boldsymbol{t}_1$$

$$\boldsymbol{v}^{+} = \boldsymbol{v}^{-} + \frac{2}{1+t_1^2}\,\boldsymbol{v}' \times \boldsymbol{t}_1$$

$$\boldsymbol{v}^{t+\frac{1}{2}\Delta t} = \boldsymbol{v}^{+} + \boldsymbol{e}_1$$

Boris 1970

Solution 1: Subcycling

- We repeat the procedure n times with $\dfrac{\Delta t}{n}$
- We further accelerate the n-times calculation by using a formula

element E vector element B vector

$$\boldsymbol{e}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{E}, \quad \boldsymbol{t}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{B}$$

element E vector element B vector

$$\boldsymbol{e}_n \equiv \frac{q\Delta t}{2nm}\boldsymbol{E}, \quad \boldsymbol{t}_n \equiv \frac{q\Delta t}{2nm}\boldsymbol{B}$$

$$\begin{aligned}
\boldsymbol{v}^- &= \boldsymbol{v}^{t-\frac{1}{2}\Delta t} + \boldsymbol{e}_1 \\
\boldsymbol{v}' &= \boldsymbol{v}^- + \boldsymbol{v}^- \times \boldsymbol{t}_1 \\
\boldsymbol{v}^+ &= \boldsymbol{v}^- + \frac{2}{1+t_1^2}\,\boldsymbol{v}' \times \boldsymbol{t}_1 \\
\boldsymbol{v}^{t+\frac{1}{2}\Delta t} &= \boldsymbol{v}^+ + \boldsymbol{e}_1
\end{aligned}$$

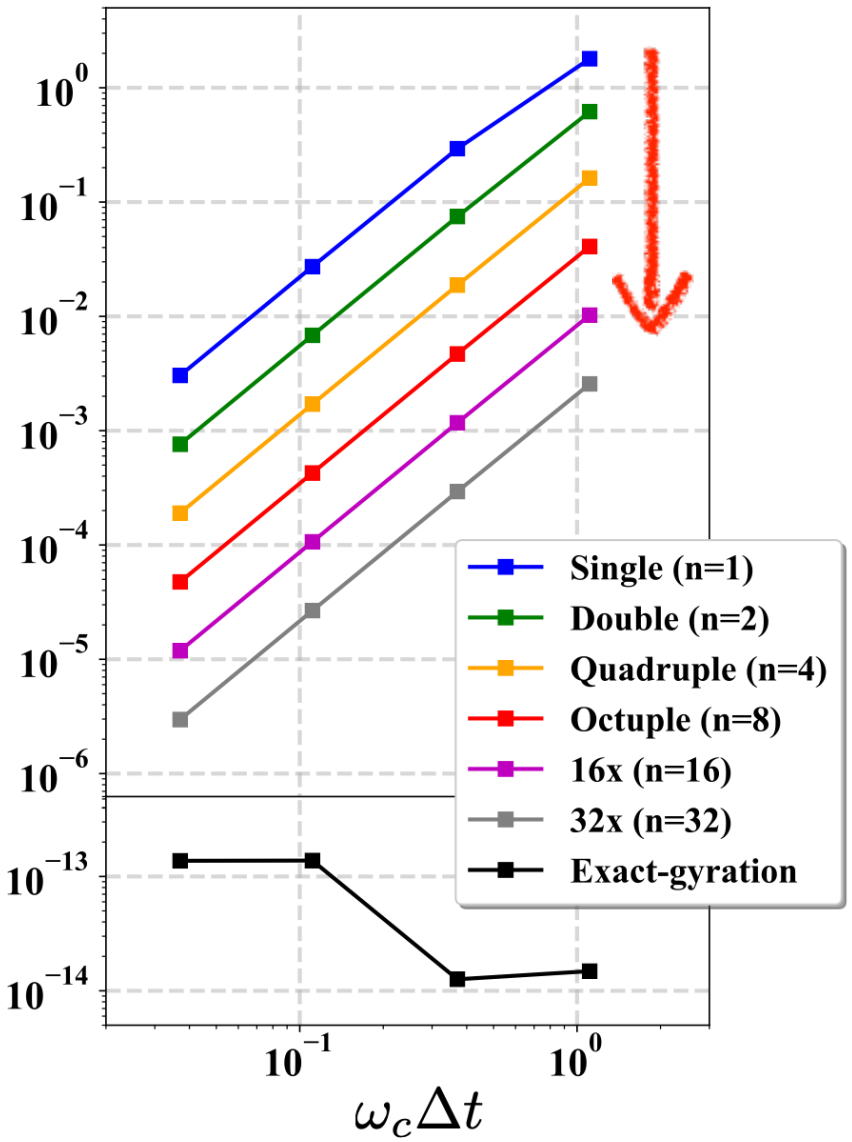
subcycling

$$\left.\begin{aligned}
\boldsymbol{v}^- &= \boldsymbol{v}^{(0)} + \boldsymbol{e}_n \\
\boldsymbol{v}' &= \boldsymbol{v}^- + \boldsymbol{v}^- \times \boldsymbol{t}_n \\
\boldsymbol{v}^+ &= \boldsymbol{v}^- + \tfrac{2}{1+t_n^2}\,\boldsymbol{v}' \times \boldsymbol{t}_n \\
\boldsymbol{v}^{(1)} &= \boldsymbol{v}^+ + \boldsymbol{e}_n \\
&\vdots \\
\boldsymbol{v}^- &= \boldsymbol{v}^{(n-1)} + \boldsymbol{e}_n \\
\boldsymbol{v}' &= \boldsymbol{v}^- + \boldsymbol{v}^- \times \boldsymbol{t}_n \\
\boldsymbol{v}^+ &= \boldsymbol{v}^- + \tfrac{2}{1+t_n^2}\,\boldsymbol{v}' \times \boldsymbol{t}_n \\
\boldsymbol{v}^{(n)} &= \boldsymbol{v}^+ + \boldsymbol{e}_n
\end{aligned}\right\} \times n \text{ times}$$

Numerical test

Numerical error

$$\frac{\|\delta \boldsymbol{v}\|}{\|\boldsymbol{v}\|}$$



$$\propto \left(\frac{\Delta t}{n}\right)^2$$

Solution 2: Higher-order corrections

element E vector element B vector

$$\boldsymbol{e}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{E}, \quad \boldsymbol{t}_1 \equiv \frac{q\Delta t}{2m}\boldsymbol{B}$$

Correction factor

$$f_1 \equiv \frac{\tan t_1}{t_1} = 1 + \frac{1}{3}t_1^2 + \frac{2}{15}t_1^4 + \frac{17}{315}t_1^6 + \frac{62}{2835}t_1^8 + \cdots$$

Gyrophase correction (Boris 1970)

$$\boldsymbol{t}_1 \leftarrow f_1 \boldsymbol{t}_1$$

Drift-speed correction

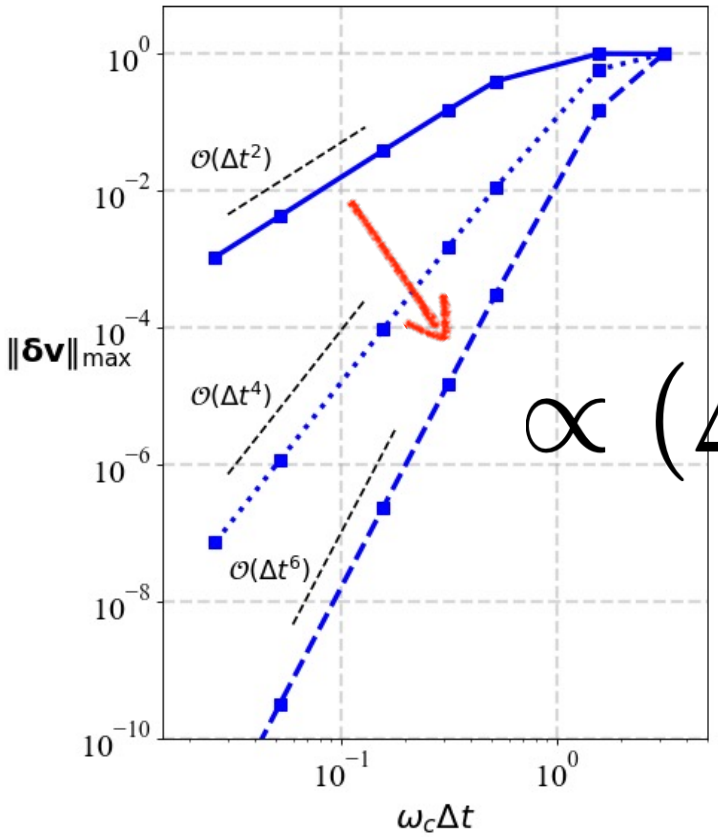
$$\boldsymbol{e}_1 \leftarrow f_1 \boldsymbol{e}_1 + \frac{1 - f_1}{t_1^2}(\boldsymbol{e}_1 \cdot \boldsymbol{t}_1)\boldsymbol{t}_1$$

$$\boldsymbol{v}^- = \boldsymbol{v}^{t-\frac{1}{2}\Delta t} + \boldsymbol{e}_1$$

$$\boldsymbol{v}' = \boldsymbol{v}^- + \boldsymbol{v}^- \times \boldsymbol{t}_1$$

$$\boldsymbol{v}^+ = \boldsymbol{v}^- + \frac{2}{1 + t_1^2}\,\boldsymbol{v}' \times \boldsymbol{t}_1$$

$$\boldsymbol{v}^{t+\frac{1}{2}\Delta t} = \boldsymbol{v}^+ + \boldsymbol{e}_1$$



$$\propto (\Delta t)^N$$

Solution 3: Combination of the two

- 0. Boris solver

$$\propto (\Delta t)^2$$

- 1. Subcycling

$$\propto \left(\frac{\Delta t}{n}\right)^2$$

- 2. Higher-order corrections

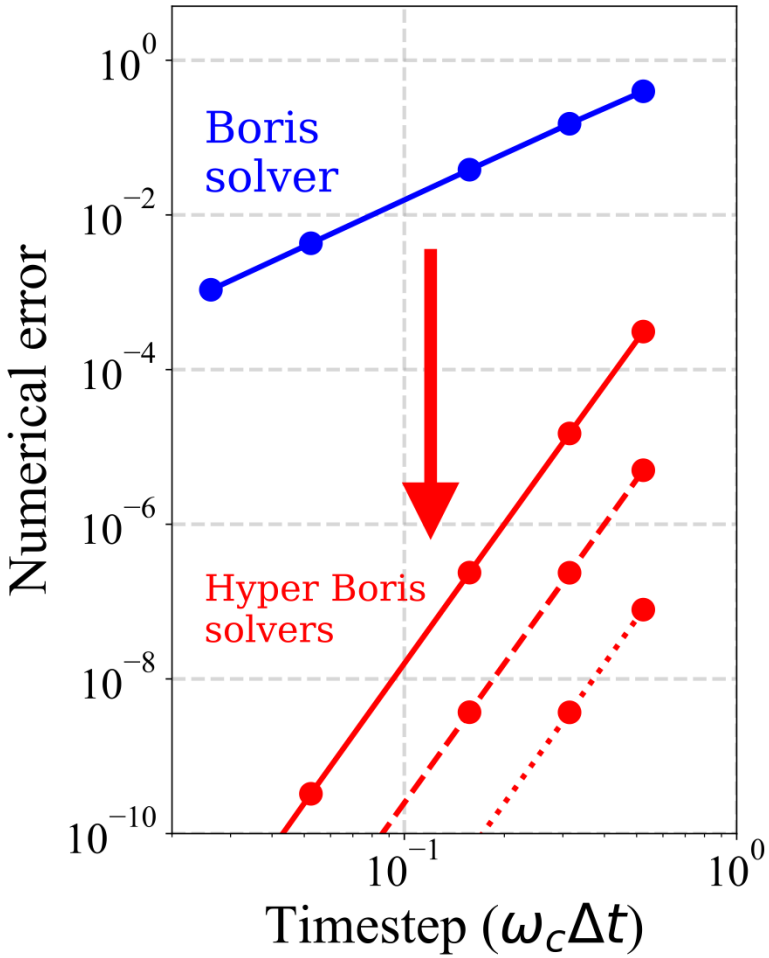
$$\propto (\Delta t)^N$$

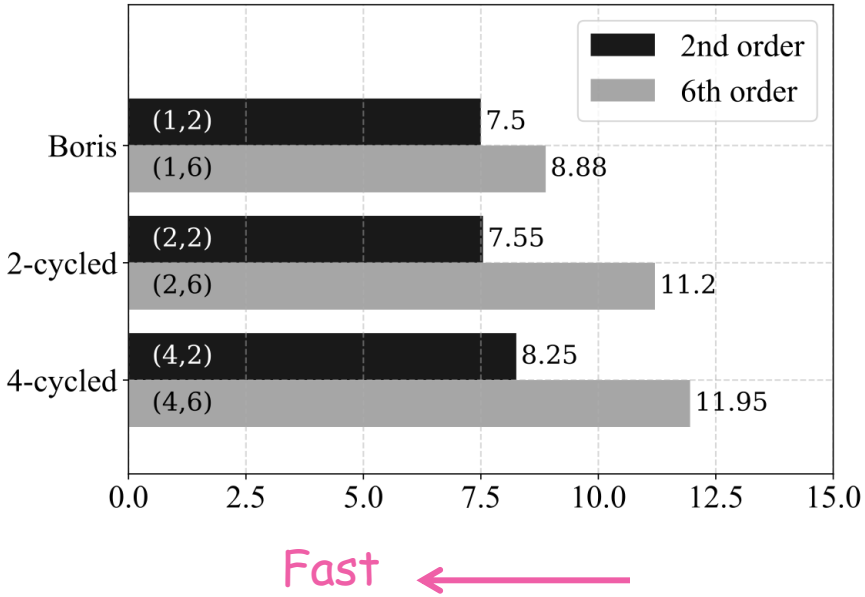
- 3. Hyper Boris solver

$$\propto \left(\frac{\Delta t}{n}\right)^N$$

Hyper Boris solver

Ultrahigh accuracy $\propto \left(\frac{\Delta t}{n}\right)^N$ Affordable computational cost





Fast ←

Zenitani & Kato 2025

Summary

- 1. Kappa generator
  - Approximate Kappa distribution
  - GPU-friendly, because it has no loop

- 2. Hyper Boris integrator
  - Subcycling + Higher-order correction = Ultrahigh accuracy
  - Affordable computational cost
  - Applicable to any vector problem of $\dfrac{d\boldsymbol{v}}{dt} = \boldsymbol{F} + \boldsymbol{v} \times \boldsymbol{R}$

- References:

1. S. Zenitani & T. Umeda, *Earth, Planets & Space* **78**, 119 (2026)
2. S. Zenitani & T.-N. Kato, *Comput. Phys. Commun.* **315**, 109695 (2025)


Thanks for
your hospitality!
谢谢澳门