Implementing Stabilized and Adaptive Time-Stepping Schemes in GENE-X

Ibrahim Almuslimani

École Polytechnique Fédérale de Lausanne (EPFL)

Swiss Plasma Center (SPC)

TSVV4 annual meeting
June 17, 2025

Swiss Plasma Center

EPFL

Outline

EPFL

Motivation: Accelerating GENE-X

EPFL

Primary Goal: To accelerate simulations in the GENE-X code by implementing more efficient time-stepping schemes.

Current Schemes and Limitations:

- **RK4:**
  - → A standard, fixed-step integrator.
  - → Can be inefficient as it must use a timestep small enough for the most challenging parts of the simulation, even when the physics is less dynamic.
- **Strang Splitting:**
  - ✅ Used to handle different physics (e.g., Vlasov terms with RK4, collisions terms with RK4 or RKC).
  - → Introduces additional spitting error.
  - → Can require more function evaluations than necessary.

Motivation: Accelerating GENE-X

EPFL

Proposed Solutions:

- PIROCK (Partitioned Runge-Kutta-Chebyshev):
  - ✓ A partitioned scheme that avoids splitting errors.
  - ✓ It is designed for problems with both stiff diffusion (collisions) and advection (Vlasov) terms.
  - ✓ It uses a stabilized method (ROCK2) for the stiff part, allowing for much larger timesteps in high-collisionality scenarios.
- RK43 (Adaptive RK4):
  - ✓ An adaptive-step integrator.
  - ✓ Estimates the local error at each step to automatically adjust the timestep size Δt.
  - ✓ This ensures the simulation runs as fast as possible while maintaining a desired level of accuracy, avoiding wasted time with unnecessarily small steps.

RK methods: General definition

EPFL

Consider the ODE

$$\frac{dy}{dt} = f(t, y), \qquad y(0) = y_0.$$

Runge Kutta integration $y_n \mapsto y_{n+1}$

$$\text{Internal stages} \rightarrow k_i = f(t_n + c_i \Delta t, y_n + \Delta t \sum_{j=1}^{s} a_{ij} k_j), \quad i = 1, \ldots, s,$$

$$\text{Solution} \rightarrow y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i k_i.$$

Butcher tableau representation

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^T \end{array} = \begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \ldots & a_{1s} \\ c_2 & a_{21} & a_{22} & \ldots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \ldots & a_{ss} \\ \hline & b_1 & b_2 & \ldots & b_s \end{array}$$

RK methods: General definition

EPFL

Consider the ODE

$$\frac{dy}{dt} = f(t, y), \qquad y(0) = y_0.$$

Runge Kutta integration $y_n \mapsto y_{n+1}$ *(Equivalent Definition)*

$$\text{Internal stages} \rightarrow K_i = y_n + \Delta t \sum_{j=1}^{s} a_{ij} f(t_n + c_j \Delta t, K_j), \quad i = 1, \ldots, s,$$

$$\text{Solution} \rightarrow y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i f(t_n + c_i \Delta t, K_i).$$

Butcher tableau representation

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^T \end{array} = \begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \ldots & a_{1s} \\ c_2 & a_{21} & a_{22} & \ldots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \ldots & a_{ss} \\ \hline & b_1 & b_2 & \ldots & b_s \end{array}$$

Examples of RK methods

EPFL

Explicit methods have lower triangular Butcher tableau:

$$\begin{array}{c|cccccc} 0 & 0 & & & & & \\ c_2 & a_{21} & 0 & & & & \\ c_3 & a_{31} & a_{32} & 0 & & & \\ \vdots & \vdots & \vdots & & \ddots & & \\ c_s & a_{s1} & \dots & \dots & a_{s,s-1} & 0 \\ \hline & b_1 & b_2 & \dots & \dots & b_s \end{array}$$

Common RK methods:

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

Explicit Euler

$$\begin{array}{c|c} 0 & 1 \\ \hline & 1 \end{array}$$

Implicit Euler

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

RK2

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

RK4

Stability of Runge-Kutta methods

EPFL

Consider the autonomous ODE

$$\frac{dy(t)}{dt} = f(y(t)), \qquad y(0) = y_0,$$

Stability function. A Runge-Kutta method with timestep Δt applied to the linear test problem

$$\frac{dy(t)}{dt} = \lambda y, \quad y(0) = y_0, \quad \lambda \in \mathbb{C},$$

yields $y_n = R(\Delta t\lambda)^n y_0$. The stability region is defined by:

$$\mathcal{S} := \{z \in \mathbb{C}; |R(z)| \leq 1\}.$$

Example. Explicit Euler: $y_{n+1} = y_n + \Delta t\lambda y_n$ $R(z) = 1 + z$. Stability condition: $|1 + \Delta t\lambda| \leq 1$ and for real eigenvalues $-2 \leq \Delta t\lambda \leq 0$ which leads for diffusion problems the famous CFL condition $\Delta t \leq C\Delta x^2$.

Stability regions

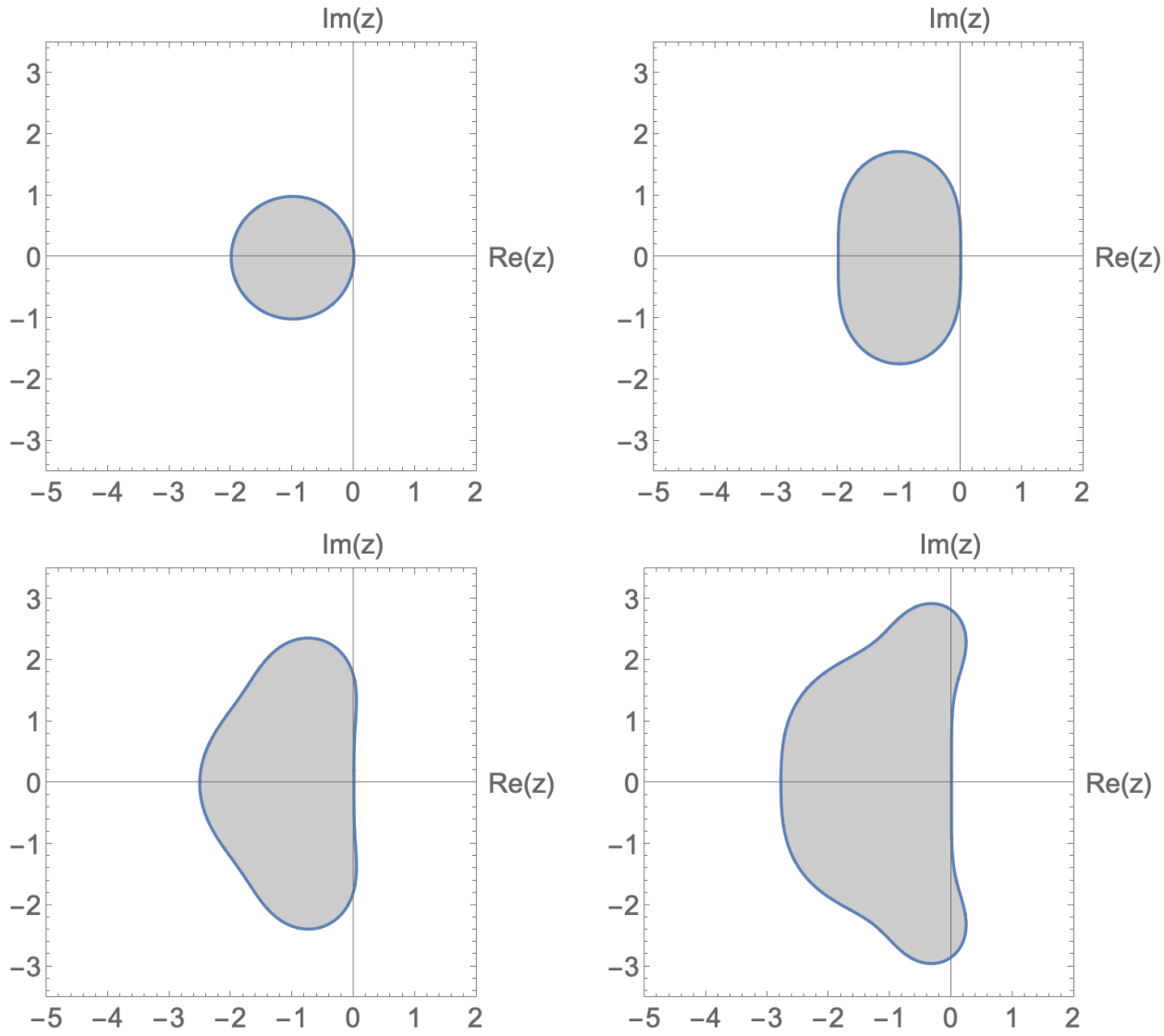


Figure: Stability regions of explicit Euler, RK2, RK3, and RK4 methods.

Explicit stabilized methods (Ideal for diffusion)

EPFL

Optimal 1^{st} order RKC method: Stability region contains $[-1.94s^2, 0]$, stability polynomial $R_s(z) = \frac{T_s(\omega_0+\omega_1 z)}{T_s(\omega_0)}$.

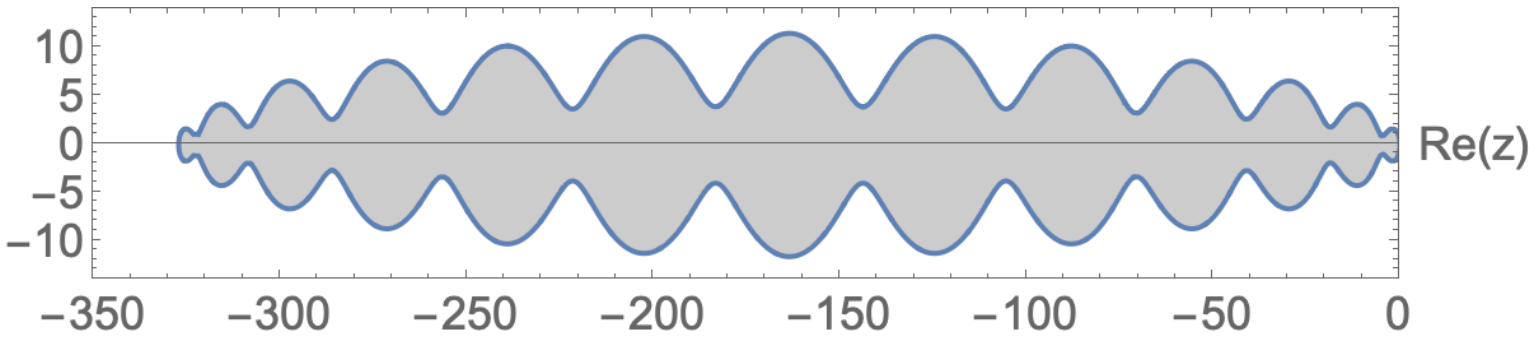


Figure: Stability region of 1^{st} order RKC for $s = 13$

Nearly optimal 2^{nd} order method ROCK2: Stability region contains $[-0.81s^2, 0]$, stability polynomial $R_s(z) = w_2(z)P_{s-1}(z)$.

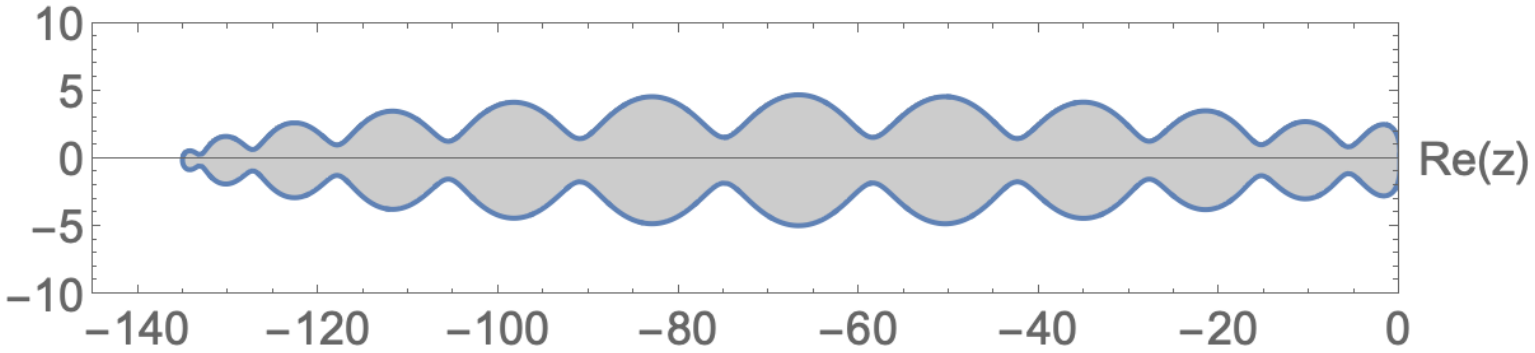


Figure: Stability region of ROCK2 for $s = 13$

Recurrence relations

EPFL

$$
\begin{aligned}
K_0 &= y_0, \\
K_1 &= y_0 + \Delta t \mu_1 f(y_0), \\
K_i &= \mu_i \Delta t f(K_{i-1}) + \nu_i K_{i-1} + \kappa_i K_{i-2}, \quad i = 2, \ldots, s \\
y_1 &= K_s,
\end{aligned}
$$

For a given step size Δt, the number of stages s is calculated adaptively such that $\lambda_{max} \Delta t \leq C_\eta s^2$,

$$
s \geq \sqrt{\frac{\Delta t \lambda_{\max}}{C_\eta}}.
$$

Splitting methods

EPFL

$$\frac{dy}{dt} = f(y) + g(y), \quad y(0) = y_0$$

where $f(y)$ and $g(y)$ are nonlinear operators (e.g., advection, collisions, neutrals).

Key idea: Split into subproblems with different physics,

$$\frac{dy}{dt} = f(y) \qquad \frac{dy}{dt} = g(y),$$

and evolve each via a separate solver:

$\Phi^f_{\Delta t}(y)$, $\Phi^g_{\Delta t}(y)$ denote flow maps of the split subsystems.

Lie Splitting (1st order):

$$y(t + \Delta t) \approx \Phi^g_{\Delta t} \circ \Phi^f_{\Delta t}(y(t))$$

Strang Splitting (2nd order):

$$y(t + \Delta t) \approx \Phi^f_{\Delta t/2} \circ \Phi^g_{\Delta t} \circ \Phi^f_{\Delta t/2}(y(t))$$

Partitioned (additive) RK methods

EPFL

$$\frac{dy}{dt} = f(y) + g(y), \qquad y(0) = y_0 \in \mathbb{R}^d.$$

Partitioned RK methods

$$K_i = y_n + \Delta t \sum_{j=1}^{s} a_{ij} f(K_j) + \Delta t \sum_{j=1}^{s} \hat{a}_{ij} g(K_j) \qquad i = 1, ..., s$$

$$y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i f(K_i) + \Delta t \sum_{i=1}^{s} \hat{b}_i g(K_i)$$

Second order conditions

$$\sum_{i=1}^{s} b_i = \sum_{i=1}^{s} \hat{b}_i = 1, \qquad \sum_{i,j=1}^{s} b_i a_{ij} = \sum_{i,j=1}^{s} \hat{b}_i \hat{a}_{ij} = \frac{1}{2} \text{ (standard 2nd order)}$$

$$\sum_{i,j=1}^{s} b_i \hat{a}_{ij} = \sum_{i,j=1}^{s} \hat{b}_i a_{ij} = \frac{1}{2} \text{ (Coupling order conditions)}$$

Advantages:

- ✓ No additional error coming from splitting.
- ✓ Less function evaluations needed compared to Strang-splitting.
- ✓ Ability to use error estimators for adaptive time-stepping.

PIROCK: ROCK2 and RK3 partition

[Abdulle and Vilmart 2013]

EPFL

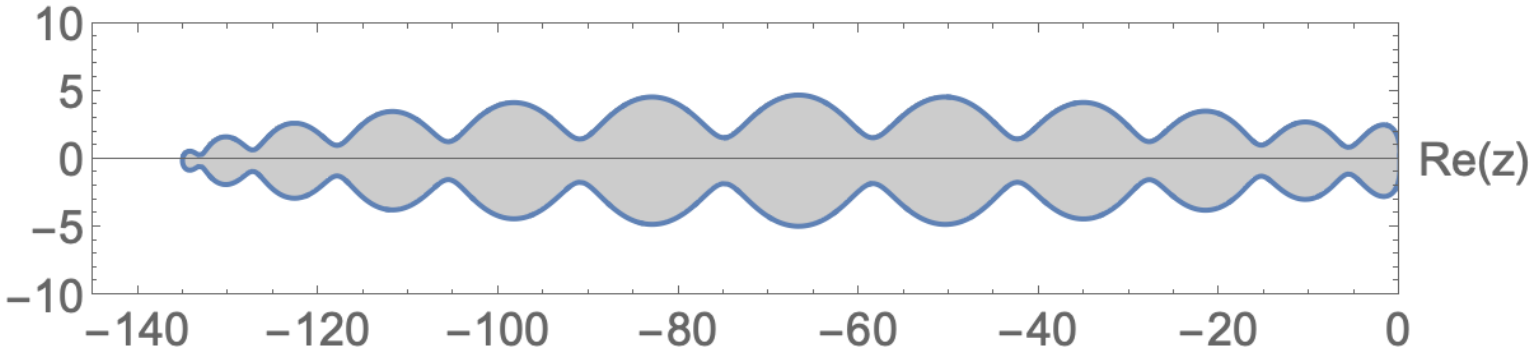


Figure: ROCK2 stability regions.

PIROCK is a partitioned RK method with ROCK2 for diffusion and RK3 for advection.

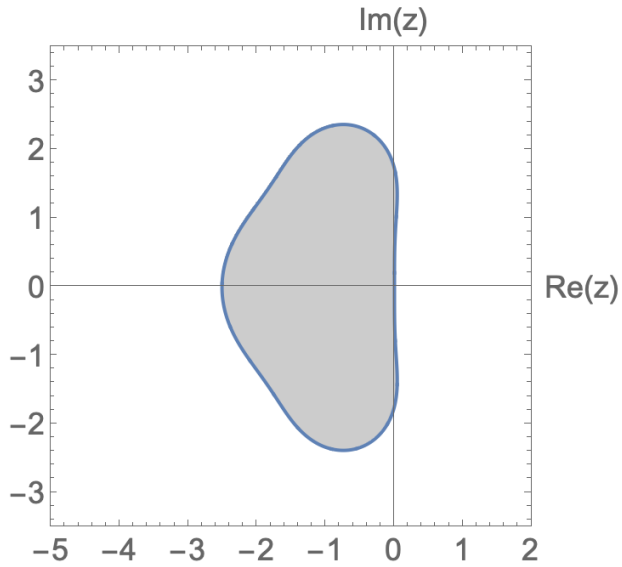


Figure: RK3 stability region.

PIROCK stability

EPFL

Test equation: $\frac{dy}{dt} = \lambda y + i\mu y$. Stability polynomial: $R(p, q)$ where $p = \Delta t\lambda$ represents a diffusion eigenvalue and $q = \Delta t\mu$ an advection eigenvalue.

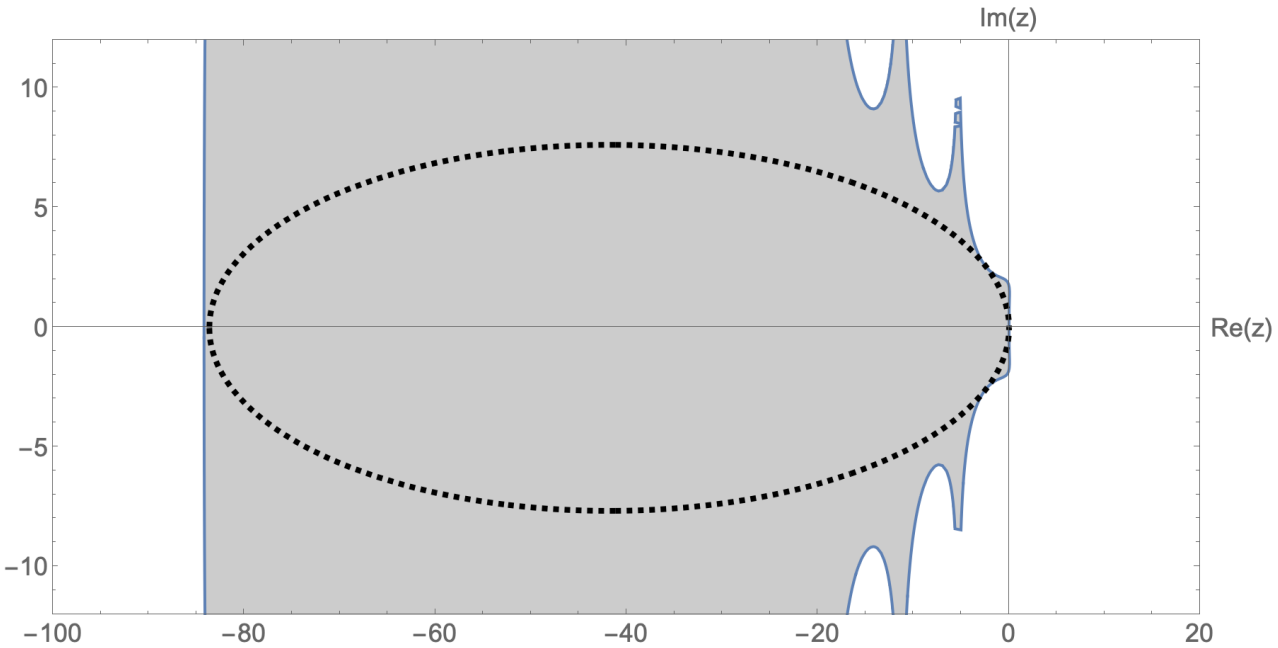


Figure: PIROCK stability region for $F_D - F_A$ coupling.

The stability region of PIROCK contains the intervals $[-0.43s^2, 0]$ and $[-i\sqrt{3}, i\sqrt{3}]$. The number of stages must satisfy

$$s \geq \sqrt{\frac{\lambda^{\mathrm{D}}_{\max}\Delta t}{0.43}}.$$

PIROCK implementation

EPFL

Consider the following ODE:

$$\frac{dy}{dt} = \underbrace{F_D(t, y(t))}_{\text{collisions + neutrals}} + \underbrace{F_A(t, y(t))}_{\text{Vlasov (plasma in GENE-X)}}, \quad t \geq t_0 \quad y(0) = y_0,$$

PIROCK scheme:

$$\begin{aligned}
K_0 &= y_n, \\
K_1 &= y_n + \alpha\mu_1 \Delta t F_D(t_n, y_n), \\
K_j &= \alpha\mu_j \Delta t F_D(t_n + c_{j-1}\Delta t, K_{j-1}) + \nu_j K_{j-1} + (1-\nu_j) K_{j-2}, \quad j = 2, \ldots, s-1.
\end{aligned}$$

Finishing procedure for diffusion:

$$\begin{aligned}
\bar{K}_{s-1} &= K_{s-2} + \sigma_s \Delta t F_D(t_n + c_{s-2}\Delta t, K_{s-2}), \\
\bar{K}_s &= \bar{K}_{s-1} + \sigma_s \Delta t F_D(t_n + \bar{c}_{s-1}\Delta t, \bar{K}_{s-1}).
\end{aligned}$$

Coupling with advection:

$$\begin{aligned}
K_{s+1} &= K_{s-1} + \Delta t(1-2\gamma) F_A(t_n + c_{s-1}\Delta t, K_{s-1}), \\
K_{s+2} &= K_{s-1} + \frac{\Delta t}{3} F_A(t_n + c_{s-1}\Delta t, K_{s-1}), \\
K_{s+3} &= K_{s-1} + \frac{2\Delta t}{3} F_A(t_n + c_{s+2}\Delta t, K_{s+2}).
\end{aligned}$$

Computation of y_{n+1} :

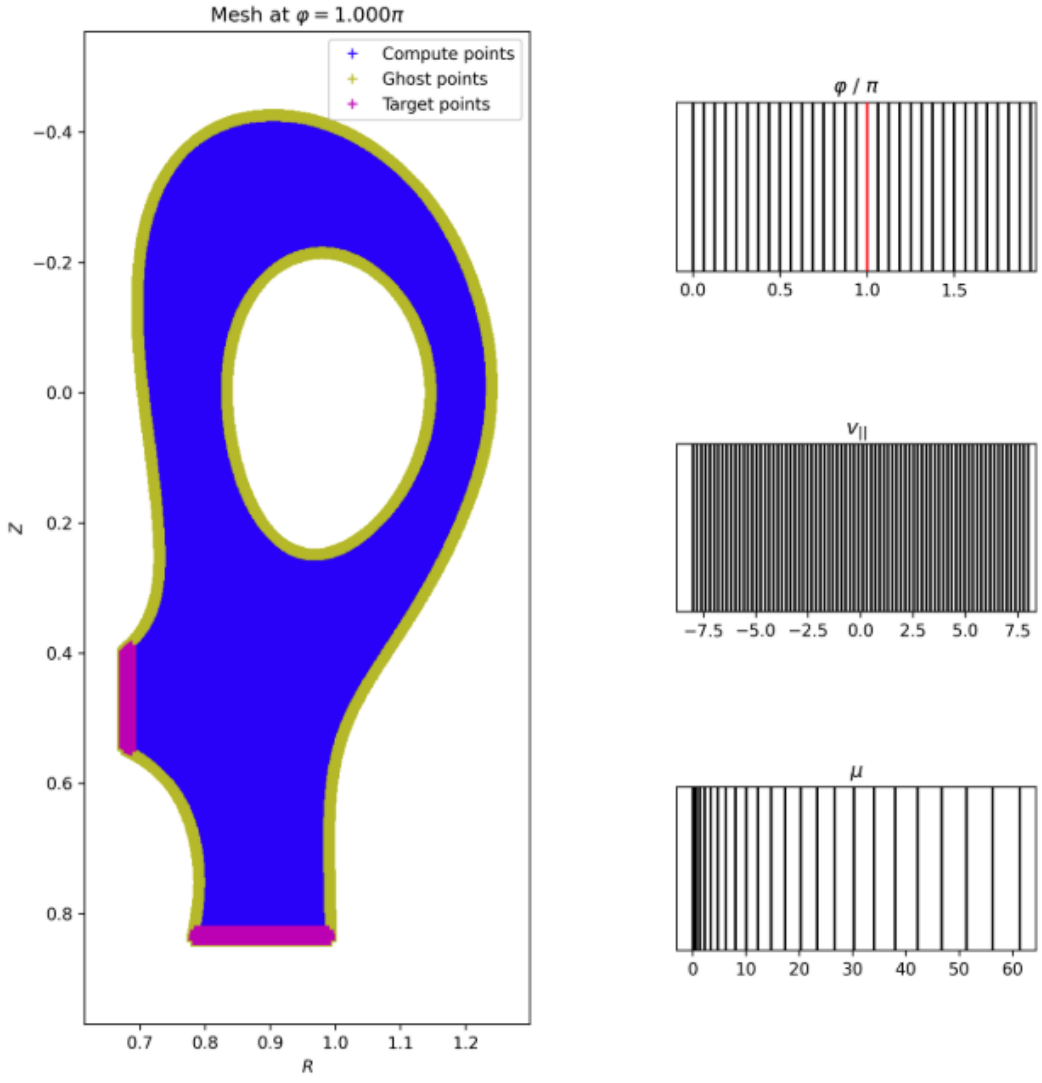
$$\begin{aligned}
y_{n+1} = \bar{K}_s &- \sigma \Delta t (1 - \frac{\tau_s}{\sigma_s^2})(F_D(t_n + \bar{c}_{s-1}\Delta t, \bar{K}_{s-1}) - F_D(t_n + c_{s-2}\Delta t, K_{s-2})) \\
&+ \frac{\Delta t}{4} F_A(t_n + c_{s-1}\Delta t, K_{s-1}) + \frac{3\Delta t}{4} F_A(t_n + c_{s+3}\Delta t, K_{s+3}) \\
&+ \frac{\Delta t}{2(1-2\gamma)} (F_D(t_n + c_{s+1}\Delta t, K_{s+1}) - F_D(t_n + c_{s-1}\Delta t, K_{s-1})).
\end{aligned}$$

Cost per step: $s+1$ F_D (collisions + neutrals) evaluations and **ONLY** 3 F_A **(Vlasov) evaluations.**

Test case: TCV-X21

EPFL

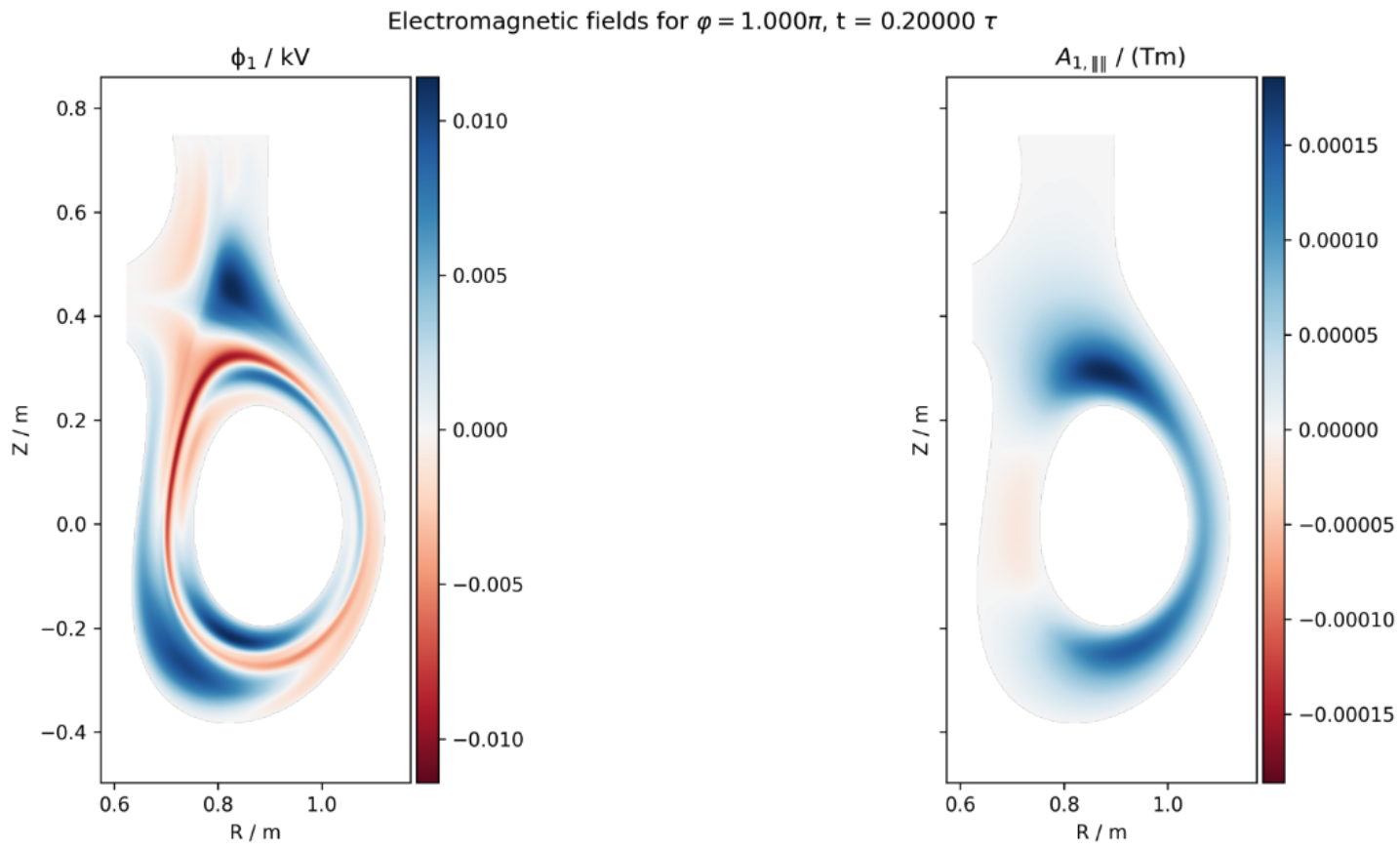
Mesh points: 500 in R, 600 in Z, 32 in ϕ, 60 in μ, 80 in $v_{\|}$. Total $\simeq 46 \times 10^9$ points. Normalization time $\tau \simeq 20\mu s$. LBD collision.



Preliminary test with PIROCK

EPFL

Simulation of 500 timesteps with $s = 5$, $\Delta t = 4 \times 10^{-4}$ on 64 nodes, 2 tasks / node, 56 CPU / task.



Average time / step $\simeq$ 112 sec.
Average time / step for Strang-splitting $\simeq$ 150 sec.
In addition, for strong collisions, Splitting will require smaller step.

Embedded RK methods RK$p(p-1)$:

$\mathbf{c}$	$\mathbf{A}$
	$\mathbf{b}^T$
	$\hat{\mathbf{b}}^T$

y_n has order p and $\hat{y}_n$ has order $p-1$.

Local error estimation:

$$err_{n+1} = \|y_{n+1} - \hat{y}_{n+1}\| \simeq C\Delta t_n^p.$$

Standard timestep update: We want $C\Delta t_{new}^p \simeq \text{tol}$. Replacing C by $err_{n+1}/\Delta t_n^p$ we get

$$\Delta t_{\text{new}} = \left(\frac{\text{tol}}{err_{n+1}}\right)^{1/p} \Delta t_n.$$

PI timestep control:

$$\Delta t_{\text{new}} = \left(\frac{\text{tol}}{err_{n+1}}\right)^{\alpha} \left(\frac{err_n}{\text{tol}}\right)^{\beta} \Delta t_n.$$

RK43

We go back to the non-partitioned equation:

$$\frac{dy}{dt} = f(y), \qquad y(0) = y_0.$$

The RK43 method is the couple:

0					
$\frac{1}{2}$	$\frac{1}{2}$				
$\frac{1}{2}$	0	$\frac{1}{2}$			
1	0	0	1		
1	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$	
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$	0
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{6}$

$$k_1 = f(y_n), \; k_2 = f(y_n + \frac{\Delta t}{2} k_1)$$

$$k_3 = f(y_n + \frac{\Delta t}{2} k_2), \; k_4 = f(y_n + \Delta t k_3)$$

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1^* = f(y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4))$$

$$\hat{y}_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_1^*)$$

The error estimation is $\|y_{n+1} - \hat{y}_{n+1}\| = \frac{\Delta t}{6}\|k_4 - k_1^*\|$. Note that $k_1^* = f(y_{n+1})$, so k_1^* is the k_1 of the next step, so it is free!

Basic adaptive algorithm

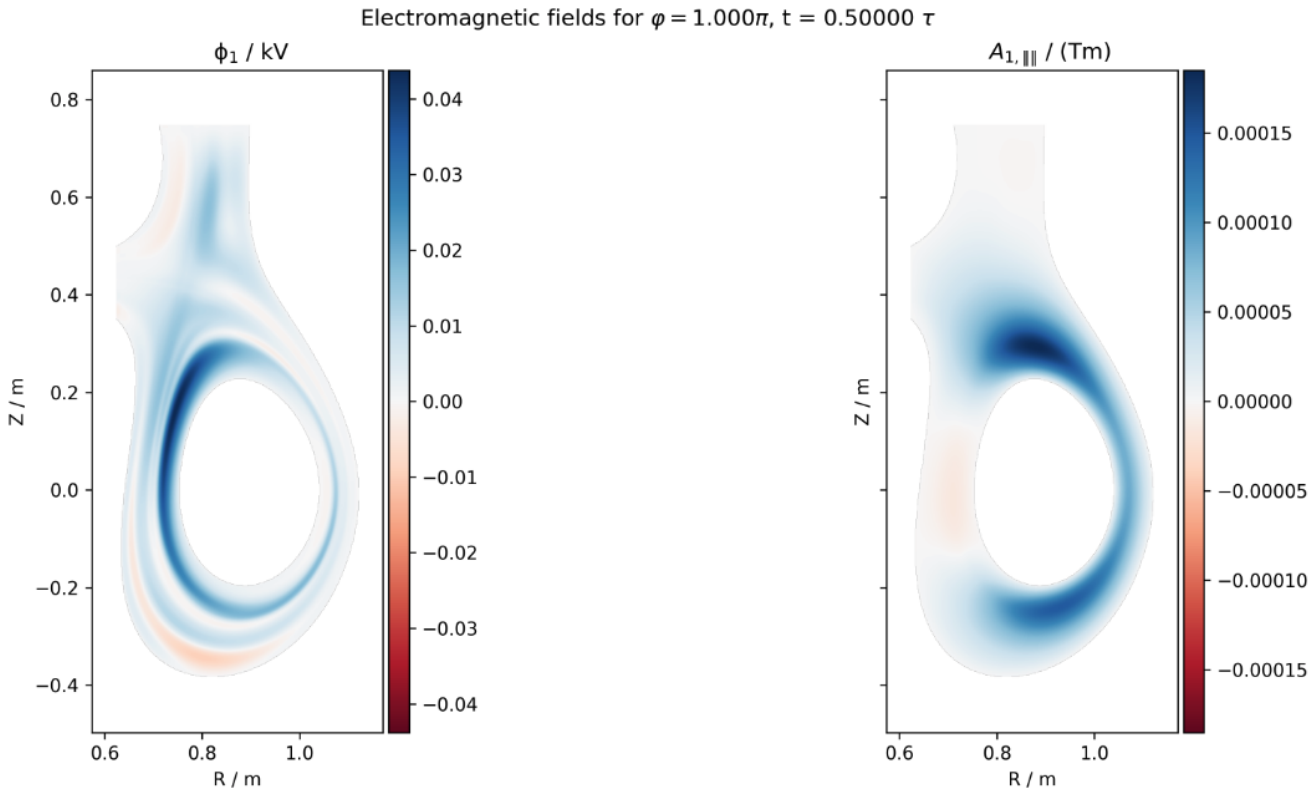
EPFL

Given y_0, Δt_0, $atol$, $rtol$, t_f, and other necessary parameters:

1: $t = t_0$, $y_{\text{now}} = y_0$, $\Delta t_{\text{now}} = \Delta t_0$, $nrej = 0$, $err_{\text{old}} = 1$
2: for ($n = 1$ to $n_{\max}$) do
3: if ($t + \Delta t_{\text{now}} \geq t_f$) then
4: $\Delta t_{\text{now}} = t_f - t$
5: end if
6: while (.true.) do
7: compute y_{next} and $\hat{y}_{\text{next}}$
8: $sc = atol + rtol \times \max(\|y_{\text{now}}\|, \|y_{\text{next}}\|)$
9: $err = \|y_{\text{next}} - \hat{y}_{\text{next}}\|/sc$
10: if ($err \leq 1$) then ▷ step is accepted
11: $t = t + \Delta t_{\text{now}}$
12: $fac = safe \times \left(\frac{1}{err}\right)^{\alpha} \left(err_{\text{old}}\right)^{\beta}$
13: $\Delta t_{\text{next}} = \Delta t_{\text{now}} \times \min(facmax, \max(facmin, fac))$
14: $\Delta t_{\text{now}} = \Delta t_{\text{next}}$, $y_{\text{now}} = y_{\text{next}}$, $err_{\text{old}} = err$
15: EXIT WHILE LOOP
16: else ▷ step is rejected
17: $nrej = nrej + 1$
18: $fac = safe \times \left(\frac{1}{err}\right)^{\alpha}$
19: $\Delta t_{\text{next}} = \Delta t_{\text{now}} \times \max(facmin, fac)$
20: $\Delta t_{\text{now}} = \Delta t_{\text{next}}$
21: end if
22: end while
23: if ($t = t_f$) then
24: print "Final time reached"
25: EXIT FOR LOOP
26: end if
27: end for

Preliminary test with RK43

Same mesh but with 24 points in μ. $\Delta t_0 = 4 \times 10^{-4}$, $atol = 10^{-3}$, $rtol = 10^{-1}$, $safe = 0.9$, $\alpha = 0.7/4 = 0.175$, $\beta = 0.4/4 = 0.1$, $t_{\text{final}} = 0.5$.



Number of accepted steps = 1065 and number of rejected steps = 1 (at the beginning). Maximum timestep used = 5×10^{-4} , while standard RK4 is not stable for this step size (needs $\Delta t = 2 \times 10^{-4}$).

Conclusion & Future Work

EPFL

☑ We have Implemented and tested two advanced time-stepping schemes in GENE-X:

→ **PIROCK** (Ready for production): Stabilized partitioned RK scheme for stiff problems (e.g., high-collisionality plasmas).

→ **RK43** (Almost ready to merge): Adaptive-timestep variant of classical RK4 for improved efficiency.

Key Findings

☑ **PIROCK** ∼25% reduction in time-per-step compared to Strang splitting; eliminates splitting error.

☑ **RK43** Automatically adapts Δt; enables significantly larger and safe timesteps.

Future Directions

■ Perform extensive benchmarking and validation across GENE-X test cases.

■ Finalize RK43 integration for production use.

■ Develop an adaptive version of PIROCK.

Conclusion & Future Work

EPFL

✓ We have Implemented and tested two advanced time-stepping schemes in GENE-X:

→ **PIROCK** (Ready for production): Stabilized partitioned RK scheme for stiff problems (e.g., high-collisionality plasmas).

→ **RK43** (Almost ready to merge): Adaptive-timestep variant of classical RK4 for improved efficiency.

Key Findings

✓ **PIROCK** ∼25% reduction in time-per-step compared to Strang splitting; eliminates splitting error.

✓ **RK43** Automatically adapts Δt; enables significantly larger and safe timesteps.

Future Directions

- Perform extensive benchmarking and validation across GENE-X test cases.
- Finalize RK43 integration for production use.
- Develop an adaptive version of PIROCK.

Conclusion & Future Work

EPFL

✓ We have Implemented and tested two advanced time-stepping schemes in GENE-X:

→ **PIROCK** (Ready for production): Stabilized partitioned RK scheme for stiff problems (e.g., high-collisionality plasmas).

→ **RK43** (Almost ready to merge): Adaptive-timestep variant of classical RK4 for improved efficiency.

Key Findings

✓ **PIROCK** ∼25% reduction in time-per-step compared to Strang splitting; eliminates splitting error.

✓ **RK43** Automatically adapts Δt; enables significantly larger and safe timesteps.

Future Directions

- Perform extensive benchmarking and validation across GENE-X test cases.
- Finalize RK43 integration for production use.
- Develop an adaptive version of PIROCK.

EPFL

THANK YOU !