Adaptive Physic-Based Preconditioning for hyperbolic systems.

Applications to wave and MHD models

E. Franck[1], A. Ratnani[2], E. Sonnendrücker[2], M. Hölzl[2]

28 may 2015

[1]INRIA Nancy Grand-Est and IRMA Strasbourg, TONUS team, France
[2]Max-Planck-Institut für Plasmaphysik, Garching, Germany

Outline

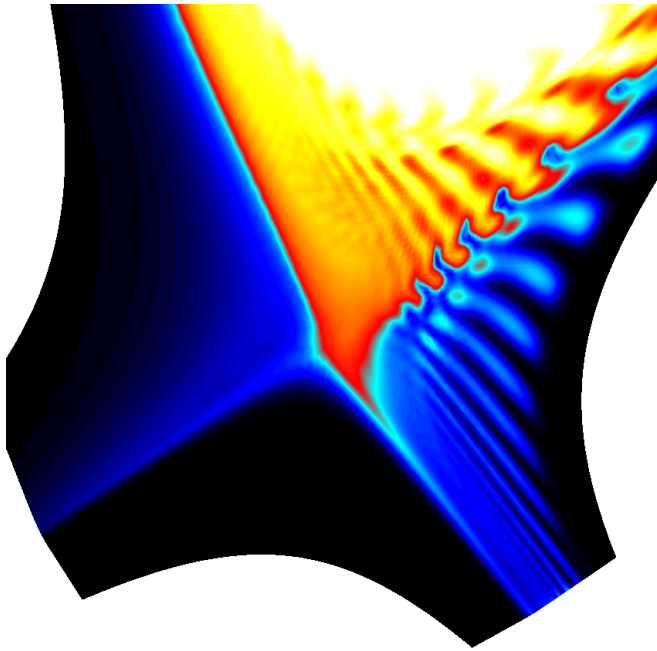
Mathematical context and JOREK code

Physical context : MHD and ELM's

- In the tokamak some instabilities can appear at the edge of the plasmas.
- The simulation to these instabilities is an important subject for ITER.
- Exemple of Edge Instabilities in the tokamak :
  - **Disruptions**: Violent edge instabilities which can damage seriously the tokamak.
  - **Edge Localized Modes (ELM's)**: Periodic edge instabilities which can damage the Tokamak.
- These instabilities are described by MHD models like

$$
\begin{cases}
\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0 \\
\rho \partial_t \mathbf{u} + \rho \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{J} \times \mathbf{B} - \nabla \cdot \overline{\overline{\Pi}} \\
\dfrac{1}{\gamma - 1} \partial_t p + \dfrac{1}{\gamma - 1} \mathbf{u} \cdot \nabla p + \dfrac{\gamma}{\gamma - 1} p \nabla \cdot \mathbf{u} + \nabla \cdot \mathbf{q} \\
= \frac{1}{\gamma - 1} \frac{m_i}{e\rho} \mathbf{J} \cdot \left(\nabla p_e - \gamma p_e \frac{\nabla \rho}{\rho} \right) - \overline{\overline{\Pi}} : \nabla \mathbf{u} + \eta |\mathbf{J}|^2 \\
\partial_t \mathbf{B} = -\nabla \times \left(-\mathbf{u} \times \mathbf{B} + \eta \mathbf{J} - \dfrac{m_i}{\rho e} \nabla p_e + \dfrac{m_i}{\rho e} (\mathbf{J} \times \mathbf{B}) \right) \\
\mu_0 \nabla \times \mathbf{B} = \mathbf{J}, \quad \nabla \cdot \mathbf{B} = 0
\end{cases}
$$

- ELM's Simulation



Time scheme in JOREK code

- The model is $\partial_t A(\mathbf{U}) = B(\mathbf{U}, t)$
- For the time stepping we use a Crank Nicholson or Gear scheme :

$$(1+\zeta)A(\mathbf{U}^{n+1}) - \zeta A(\mathbf{U}^{n}) + \zeta A(\mathbf{U}^{n-1}) = \theta \Delta t B(\mathbf{U}^{n+1}) + (1-\theta)\Delta t B(\mathbf{U}^{n}).$$

- Defining $G(\mathbf{U}) = (1+\zeta)A(\mathbf{U}) - \theta\Delta t B(\mathbf{U})$ and

$$b(\mathbf{U}^{n}, \mathbf{U}^{n-1}) = (1+2\zeta)A(\mathbf{U}^{n}) - \zeta A(\mathbf{U}^{n-1}) + (1-\theta)\Delta t B(\mathbf{U}^{n})$$

  we obtain the nonlinear problem

$$G(\mathbf{U}^{n+1}) = b(\mathbf{U}^{n}, \mathbf{U}^{n-1}).$$

- First order linearization

$$\left(\frac{\partial G(\mathbf{U}^{n})}{\partial \mathbf{U}^{n}}\right) \delta\mathbf{U}^{n} = -G(\mathbf{U}^{n}) + b(\mathbf{U}^{n}, \mathbf{U}^{n-1}) = R(\mathbf{U}^{n}),$$

  with $\delta\mathbf{U}^{n} = \mathbf{U}^{n+1} - \mathbf{U}^{n}$, and $J_n = \frac{\partial G(\mathbf{U}^{n})}{\partial \mathbf{U}^{n}}$ the Jacobian matrix of $G(\mathbf{U}^{n})$.

- First order linearization can be replaced by Newton method (more robust).

Linear Solvers

- Linear solver in JOREK: Preconditioning + GMRES iterative solver.
- Principle of the preconditioning (right) step:
  - Replace the problem $J_k \delta \mathbf{U}_k = R(\mathbf{U}^n)$ by $P_k (P_k^{-1} J_k) \delta \mathbf{U}_k = R(\mathbf{U}^n)$.
  - Solve the new system with two steps $P_k \delta \mathbf{U}_k^* = R(\mathbf{U}^n)$ and $(P_k^{-1} J_k) \delta \mathbf{U}_k = \delta \mathbf{U}_k^*$
- If P_k is easier to invert than J_k and $P_k \approx J_k$ the linear solving step is more robust and efficient.
- **Preconditioning** : algorithm to solve $P_k \mathbf{x} = \mathbf{b}$.

Physic-based Preconditioning of JOREK

- Extraction of the blocks which are associated with each toroidal harmonic (diagonal block).
- LU factorization of each block.
- Solve exactly with LU decomposition each subsystem associated with a block.
- Reconstruction of the solution of $P_k \mathbf{x} = \mathbf{b}$.

- **Physic-based preconditioning interpretation**: We neglect in the Jacobian the physical effects associated with the coupling between the Fourier mods (non diagonal block).

Advantages and defaults of the JOREK Preconditioning

Advantages of the Physic Based JOREK preconditioning

- Very efficient preconditioning in the linear phase.
- Efficient preconditioning for lot of test cases.
- Nice idea to construct a preconditioning using the knowledge of the physic and the discretization.

Defaults of the Physic Based JOREK preconditioning

- Preconditioning less efficient for case with strong coupling between Fourier modes).
- Important CPU cost in the nonlinear phase (factorization is computed often).
- Very important memory consumption (storage of LU decomposition and the Jacobian).
- Not independent to the toroidal discretization.

Aim: design a preconditioning which is:

- efficient in the linear phase (less than the previous one) and in the nonlinear phase,
- independent in the principle to the discretization,
- not so greedy in memory (Compatible with free matrix methods),
- adaptable to the difficulty of the test case.

Physic based preconditioning for Waves equations

Implicit scheme for Damped waves equations

- Damping wave equation (baby problem used for Inertial fusion confinement)

$$\begin{cases} \partial_t p + c\nabla\cdot\mathbf{u} = 0 \\ \partial_t \mathbf{u} + c\nabla p = -c\sigma\mathbf{u} \end{cases} \iff \begin{cases} \partial_t p + \frac{1}{\varepsilon}\nabla\cdot\mathbf{u} = 0 \\ \partial_t \mathbf{u} + \frac{1}{\varepsilon}\nabla p = -\frac{\sigma}{\varepsilon^2}\mathbf{u} \end{cases}$$

- with σ opacity, c light speed and $\varepsilon \approx \frac{1}{c} \approx \frac{1}{\sigma}$
- When $\varepsilon \longrightarrow 0$ the model can be approximated by $\partial_t p - \nabla\cdot(\frac{1}{\sigma}\nabla p) = 0$.
- This problem is stiff in time. CFL condition is $\Delta t \leq C_1\varepsilon h + C_2\varepsilon^2$.
- Simple way to solve this: implicit scheme but the model is ill-conditioned.
- Two sources of ill-conditioning: the stiff terms (which depend of ε) and the hyperbolic structure.

We propose a preconditioning (work of L. Chacon) which

- allows to treat the stiffness using operator splitting and reformulation of the equations (rewritting the hyperbolic system as a second order equation well-conditioned which can be solved easily),
- can be extend to the nonlinear hyperbolic system as MHD (and resistive MHD with additional splitting steps).

Construction of the preconditioning I

- First we implicit the equation

$$\left\{ \begin{array}{l} p^{n+1} + \theta \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^{n+1} = p^n - (1-\theta) \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^n \\ \mathbf{u}^{n+1} + \theta \dfrac{\Delta t}{\varepsilon} \nabla p^{n+1} + \theta \dfrac{\Delta t \sigma}{\varepsilon^2} \mathbf{u}^{n+1} = \mathbf{u}^n - (1-\theta) \dfrac{\Delta t}{\varepsilon} \nabla p^n - (1-\theta) \dfrac{\Delta t \sigma}{\varepsilon^2} \mathbf{u}^n \end{array} \right.$$

Construction of the preconditioning I

- Secondly we rewrite the equation

$$\left\{ \begin{array}{l} p^{n+1} + \theta \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^{n+1} = p^n - (1-\theta) \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^n \\ \mathbf{u}^{n+1} + \theta \dfrac{\alpha \Delta t}{\varepsilon} \nabla p^{n+1} = \alpha \mathbf{u}^n - (1-\theta) \dfrac{\alpha \Delta t}{\varepsilon} \nabla p^n - \alpha (1-\theta) \dfrac{\alpha \Delta t \sigma}{\varepsilon^2} \mathbf{u}^n \end{array} \right.$$

- with $\alpha = \dfrac{\varepsilon^2}{\varepsilon^2 + \theta \sigma \Delta t}$

Construction of the preconditioning I

- Secondly we rewrite the equation

$$\left\{ \begin{array}{l} p^{n+1} + \theta \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^{n+1} = p^n - (1-\theta) \dfrac{\Delta t}{\varepsilon} \nabla \cdot \mathbf{u}^n \\ \mathbf{u}^{n+1} + \theta \dfrac{\alpha \Delta t}{\varepsilon} \nabla p^{n+1} = \alpha \mathbf{u}^n - (1-\theta) \dfrac{\alpha \Delta t}{\varepsilon} \nabla p^n - \alpha (1-\theta) \dfrac{\alpha \Delta t \sigma}{\varepsilon^2} \mathbf{u}^n \end{array} \right.$$

- with $\alpha = \dfrac{\varepsilon^2}{\varepsilon^2 + \theta \sigma \Delta t}$
- The implicit system is given by

$$\begin{pmatrix} M & U \\ L & D \end{pmatrix} \begin{pmatrix} p^{n+1} \\ \mathbf{u^{n+1}} \end{pmatrix} = \begin{pmatrix} R_p \\ R_{\mathbf{u}} \end{pmatrix}$$

with $M = I_d$, $D = \begin{pmatrix} I_d & 0 \\ 0 & I_d \end{pmatrix}$, $U = \begin{pmatrix} \theta \dfrac{\Delta t}{\varepsilon} \partial_x & \dfrac{\Delta t}{\varepsilon} \partial_y \end{pmatrix}$ and $L = \begin{pmatrix} \alpha\theta \dfrac{\Delta t}{\varepsilon} \partial_x \\ \alpha\theta \dfrac{\Delta t}{\varepsilon} \partial_y \end{pmatrix}$

- The solution of the system is given by

$$\begin{pmatrix} p^{n+1} \\ \mathbf{u^{n+1}} \end{pmatrix} = \begin{pmatrix} M & U \\ L & D \end{pmatrix}^{-1} \begin{pmatrix} R_p \\ R_{\mathbf{u}} \end{pmatrix}$$

$$= \begin{pmatrix} I & M^{-1}U \\ 0 & I \end{pmatrix} \begin{pmatrix} M^{-1} & 0 \\ 0 & P_{schur}^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -LM^{-1} & I \end{pmatrix} \begin{pmatrix} R_p \\ R_{\mathbf{u}} \end{pmatrix}$$

with $P_{schur} = D - LM^{-1}U$.

Principle of the preconditioning II

- Using the previous Schur decomposition we can solve the implicit wave equation with the algorithm.

$$\left\{ \begin{array}{l} \text{Predictor :} \quad M_h p^* = R_p \\ \text{Velocity evolution :} \quad P_h \mathbf{u}^{n+1} = (-L_h p^* + R_{\mathbf{u}}) \\ \text{Corrector :} \quad M p^{n+1} = M_h p^* - U_h \mathbf{u}_{n+1} \end{array} \right.$$

- with the matrices:
  - M_h the mass matrix which discretize the Identity operator
  - U_h discretize the operator U and L_h the discretization of the L operator.
  - P_h discretize the positive and symmetric operator :

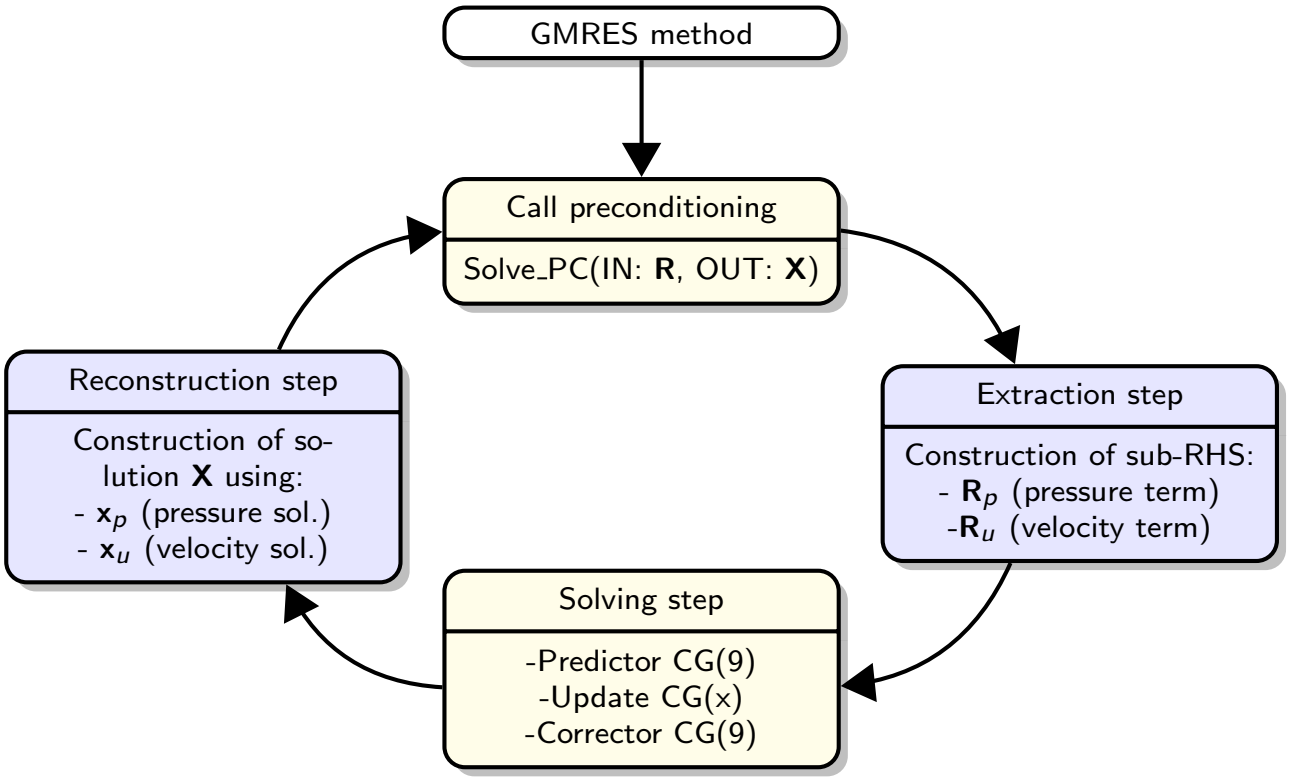
$$P_{Schur} = I_d - \theta^2 \frac{\alpha \Delta t^2}{\varepsilon^2} \begin{pmatrix} \partial_{xx} & \partial_{xy} \\ \partial_{yx} & \partial_{yy} \end{pmatrix}$$

- **Remark**: in this case, there is no approximation in the Schur, but in many cases (nonlinear models) we must use an approximation.

- The physic based preconditioning $PB(x)$ solves the previous algorithm with :
  - The Conjugate Gradient with $\varepsilon = 10^{-9}$ for predictor and correction step.
  - The Conjugate Gradient with $\varepsilon = 10^{-x}$ for velocity update step.

Algorithm of the PhyBas Preconditioning step

- Algorithm and implementation of the $PB(x)$ preconditioning:



- In this case we solve the sub-steps with a GC solver.
- We can use also Multi-grid (MG) methods or other methods efficient for symmetric and diagonal dominant matrix.

Results for Waves equation

- Comparison between iterative solver for test case in the diffusion limit $\sigma = 1$.

Mesh / solvers		GC	GC-PC	GMRES	GMRES-PC-Jacobi
Mesh 4*4, ε_1	cv iter	✗ -	✗ -	✗ -	✓ 27
Mesh 16*16, ε_1	cv iter	✗ -	✗ -	✗ -	✓ 1.5E+4
Mesh 4*4, ε_2	cv iter	✗ -	✗ -	✗ -	✓ 21000
Mesh 16*16, ε_2	cv iter	✗ -	✗ -	✗ -	✗ -

- $\varepsilon_1 = 10^{-5}$ and $\varepsilon_2 = 10^{-10}$.
- The solver tolerance is 10^{-10} for convergence and iter_max=100000. We compute the average of ten time iterations.
- The GC solver is unstable and cannot solve this type of problem.

A conclusion

- The results show that it is necessary to use a good preconditioning + robust solver (for general matrix).

Results for Waves equation

- Comparison between GMRES method with different preconditioning

Mesh / solvers		Jac	ILU(0)	ILU(4)	MG(2)	SOR	PB
Mesh4*4, ε_1	cv	✓	✓	✓	✓	✓	✓
	iter	27	11		38	8	1
	time	7.2 E-4	1.3E-3	7.7E-3	1.5E-2	1.4E-3	2.1E-3
4*4, ε_2	cv	✓	✓	✓	✗	✓	✓
	iter	2.1E+4	11	1	-	8	1
	time	3.6E-1	1.3E-3	7.7E-3	-	1.5E-3	2.1E-3
16*16, ε_1	cv	✓	✓	✓	✓	✓	✓
	iter	1.5E+4	18	9	140	20	1
	time	5.0E-0	2.3E-2	4.0E-1	5.0E-1	5.0E-2	2.1E-2
16*16, ε_2	cv	✗	✓	✓	✗	✓	✓
	iter	-	18	9	-	20	1
	time	-	2.3E-2	4.0E-1	-	5.0E-2	2.1E-2
64*64, ε_2	cv	✗	✗	✓	✗	✗	✓
	iter	-	-	632	-	-	1
	time	-	-	2.0E+1	-	-	4.2E-1

- ILU (Incomplete LU), MG (Multi-grids), SSOR, PB (our physic based PC).

A conclusion

- On fine grid our method is the fastest (and the implementation is not optimal).

Physic based preconditioning for MHD equations

Current Hole and preconditioning associated

- Current Hole : reduced problem in cartesian coordinates.
- The model

$$\begin{cases} \partial_t \psi = [\psi, u] + \eta \Delta \psi \\ \partial_t \Delta u = [\Delta u, u] + [\psi, \Delta \psi] + \nu \Delta^2 u \end{cases}$$

  with $w = \Delta u$ and $j = \Delta \psi$.
- In this formulation we split evolution and elliptic equations.
- For the time discretization we use a Cranck-Nicholson scheme and linearize the nonlinear system to obtain

$$\begin{pmatrix} M & U \\ L & D \end{pmatrix} \begin{pmatrix} \Delta\psi^n \\ \Delta u^n \end{pmatrix} = \begin{pmatrix} R_\psi \\ R_u \end{pmatrix}$$

or

$$\begin{pmatrix} I_d - \Delta t\theta[\cdot, u^n] - \Delta t\theta\Delta & -\Delta\theta[\psi^n, \cdot] \\ -\Delta t\theta[\psi^n, \Delta\cdot] - \Delta t\theta[\cdot, \Delta\psi^n] & \Delta - \Delta t\theta([\Delta\cdot, u^n] + [\cdot, \Delta u^n] + \Delta^2) \end{pmatrix} \begin{pmatrix} \delta\psi^n \\ \delta u^n \end{pmatrix} = \begin{pmatrix} R_\psi \\ R_u \end{pmatrix}$$

Design of the preconditioning for reduced MHD

PB-PC for Current Hole

$$\begin{cases} \text{Predictor :} & M\delta\psi_p^n = R_\psi \\ \text{potential update :} & P_{schur}\delta u^n = \left(-L\delta\psi_p^n + R_u\right) \\ \text{Corrector :} & M\delta\psi^n = M\delta\psi_p^n - U\delta u^n \\ \text{Current update :} & \delta z_j^n = \Delta\delta\psi^n \\ \text{Vorticity update :} & \delta w^n = \Delta\delta u^n \end{cases}$$

- The schur complement is given by $P_{schur} = D - LM^{-1}U$
- Two approximations for M^{-1}:
  - **Slow flow**: $M^{-1} = \Delta t$
  - **Arbitrary flow**: find M^* such that $UM^* \approx MU$. Consequently

$$P^{-1} = (D - LM^{-1}U)^{-1} \approx M^*(DM^* - LU)^{-1},$$

we obtain

$$\begin{cases} \text{potential update I :} & (DM^* - LU)\delta u^{**} = \left(-L\delta\psi_p^n + R_u\right) \\ \text{potential update II :} & \delta u^n = M^*\delta u^{**} \end{cases}$$

- **Last question** : Computation of the operator LU (second order form of the coupling hyperbolic operators).

Approximation of the Schur complement I

- Computation of Schur complement for (slow flow approximation $M^{-1} \approx \Delta t$)

$$P_{schur} = \frac{\Delta \delta u}{\Delta t} + \mathbf{u}^n \cdot \nabla(\Delta \delta u) + \delta \mathbf{u} \cdot \nabla(\Delta u^n) - \theta \nu \Delta^2 \delta u - \theta^2 \Delta t LU$$

- Operator $LU = \mathbf{B}^n \cdot \nabla(\Delta(\mathbf{B}^n \cdot \nabla \delta u)) + \frac{\partial j^n}{\partial \psi^n} \mathbf{B}^n_{pol} \cdot \nabla(\mathbf{B}^n \cdot \nabla \delta u)$.
- $\mathbf{B}^n \cdot \nabla \delta u = -[\psi^n, \delta u]$ and $\mathbf{u}^n \cdot \nabla \delta u = -[\delta u, u^n]$ et $\delta \mathbf{u} \cdot \nabla u^n = -[u^n, \delta u]$.
- **Remark**: the LU operator is the parabolization of coupling hyperbolic terms which contains only the Alfvén waves (rigorous proof missing).

Properties of LU operator

- We consider the L^2 space. The operator LU is not positive for all δu

$$< LU\delta u, \delta u >_{L^2} = \int |\nabla(\mathbf{B}^n \cdot \nabla \delta u)|^2 - \int \frac{\partial j^n}{\partial \psi^n} (\mathbf{B}^n_{pol} \cdot \nabla \delta u)(\mathbf{B}^n \cdot \nabla \delta u)$$

- The LU operator is not self-adjoint : $< LU\delta u, \delta v >_{L^2} \neq < \delta u, LU\delta v >_{L^2}$
- We propose the following approximation $LU^{approx} = \mathbf{B}^n \cdot \nabla(\Delta(\mathbf{B}^n \cdot \nabla \delta u))$.
- The operator LU^{approx} is positive and self-adjoint.

Solving the different steps of the PC

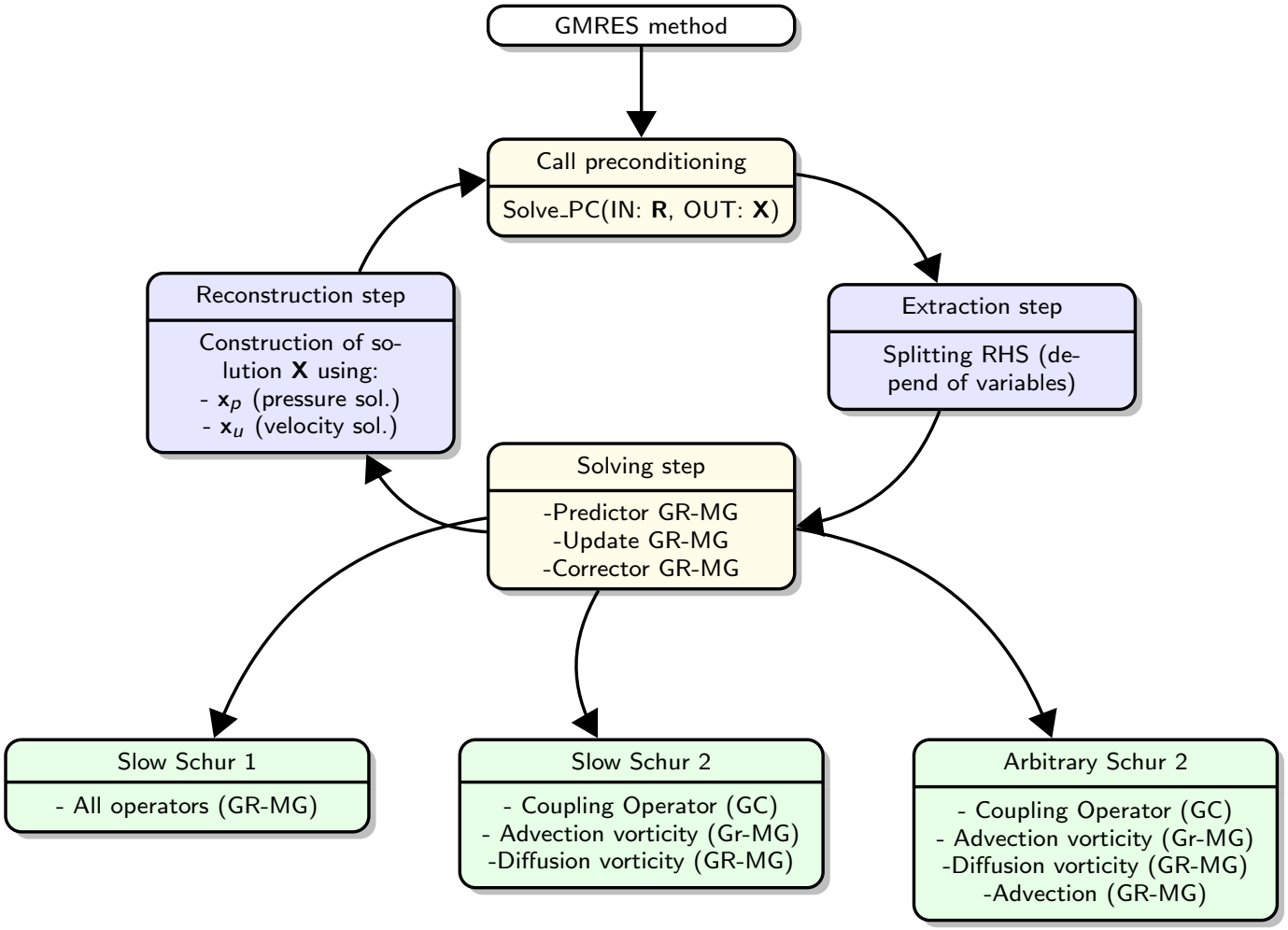
- **Question** How solve each step ?
- The first simple and efficient solver is to use the Multi-Grids methods (MG) efficient for second order and advection operators.
- But perhaps it can be more efficient to split some terms in the sub-systems to use the most adapted solver for each operator.
- Example for the Schur complement (L. Chacon paper) using a splitting and an approximation:

$$\begin{cases} \text{Schur solver I} : \Delta\delta u^* = RHS \\ \text{Schur solver II} : \left(\frac{I_d}{\Delta t} + \mathbf{u}^n \cdot \nabla I_d - \theta\nu\Delta\right)\delta u^{**} = \delta u^* \\ \text{Schur solver III} : \left(\frac{\Delta I_d}{\Delta t} - \mathbf{B}^n \cdot \nabla(\Delta(\mathbf{B}^n \cdot \nabla I_d))\right)\delta u^{n+1} = \delta u^{**} \end{cases}$$

  - MG methods are adapted for advection diffusion problems.
  - GC is more adapted for symmetric and positive anisotropic operator (smoother for MG are more complicated for anisotropic problem).
  - **L. Chacon remark**: to replace $\mathbf{B}^n \cdot \nabla(\Delta(\mathbf{B}^n \cdot \nabla I_d))$ by $\Delta(\mathbf{B}^n \cdot \nabla(\mathbf{B}^n \cdot \nabla I_d))$ generate noise.

Algorithm of the PhyBas Preconditioning step

- Algorithm and implementation of the $PB(x)$ preconditioning:



- In the future we will replace GRMES-MG by MG solvers.

Results for Current Hole Model

- Comparison between GMRES method with different preconditioning
- 50 time step in the linear phase (kink instability). $tol = 10^{-8}$, $iter\_max = 10000$.

Mesh / solvers		Jac	ILU(0)	ILU(4)	MG(2)	SOR	PB
Mesh 16*16 dt=0.5	cv	✗	✓	✓	✗	✓	✓
	iter	-	14	6	-	12	1
	time	-	1.2E-1	1.4E+0	-	1.8E-1	2.6E+0
Mesh 32*32 dt=1	cv	✗	✓	✓	✗	✗	✓
	iter	-	26	9	-	-	1
	time	-	6.8E-1	7.2E+0	-	-	9.8E+0

- On fine grid our method is robust (on finest grids it it necessary to increase k for ILU(k) method).

- This is not optimal because :
  - The matrices (7 in this case) are assembled one by one and not at the same time.
  - The extraction and reconstruction are made one by one.
  - The assembly of the matrices in Django are not optimal (PETSC configuration).
  - We solve each sub-system with a GMRES-MG(2) and not just a MG solver.

- 90% of the solving time comes from to the construction of the sub-matrices. In the future we will assume that it is possible to decrease this part by 10 or 20.

Extension for other MHD Models

- The Preconditioning is extendable to the other MHD problems

Extension to full MHD problem

- The matrix M contains advection and diffusion operators for ρ, T and $\mathbf{B}$
- To treat anisotropic operators splitting technics or adapted MG methods can be used.
- The LU operator (called ideal MHD force operator in the book of Schnack) is given by

$$(LU)\delta\mathbf{v} = [\mathbf{B}\times\nabla\times\nabla\times(I_d\times\mathbf{B}) - \mathbf{J}\times\nabla\times(I_d\times\mathbf{B}) - \nabla(I_d\cdot\nabla p + \gamma p\nabla\cdot I_d)]\,\delta\mathbf{v}$$

- This positive and self-adjoint operator contains the waves of MHD.
- A possible step to separate the waves can be added to solve easily this operator (work of S. Jardin)
- Extension of the method for the generalized Ohm's law is present in the literature.

- The equivalent can be written for reduced MHD models.

Physic based preconditioning for MHD equations

Problem of memory and "Matrix free" GMRES solver

- An important problem of the PC implicit scheme is the memory consumption.
- In the current JOREK version the PC implicit method prevent to use fine resolutions.
- **First idea**: used Free-Matrix method compatible with the previous PC algorithm.

Free Jacobian method

- In the iterative methods we replace $J\mathbf{X}$ (with J the Jacobian of $\mathbf{G}(\mathbf{U}^n)$) by

$$\frac{\mathbf{G}(\mathbf{U}^n + \epsilon\mathbf{X}) - \mathbf{G}(\mathbf{U}^n)}{\epsilon}$$

- With this method, it is not necessary to compute and store the full matrix.

Remark

- If the iterative method to solve the sub-steps of the PC is not compatible with "free Jacobian" method, we must store some sub-matrices of the PC.

Adaptive PhyBas preconditioning

Idea

- The PhyBas PC is based on physical approximations of the equations. We can also add approximations of the discretization in space.
- Indeed, we can use a less order approximation in the PC to reduce the size of the matrices and the storage and keep a good efficiency.

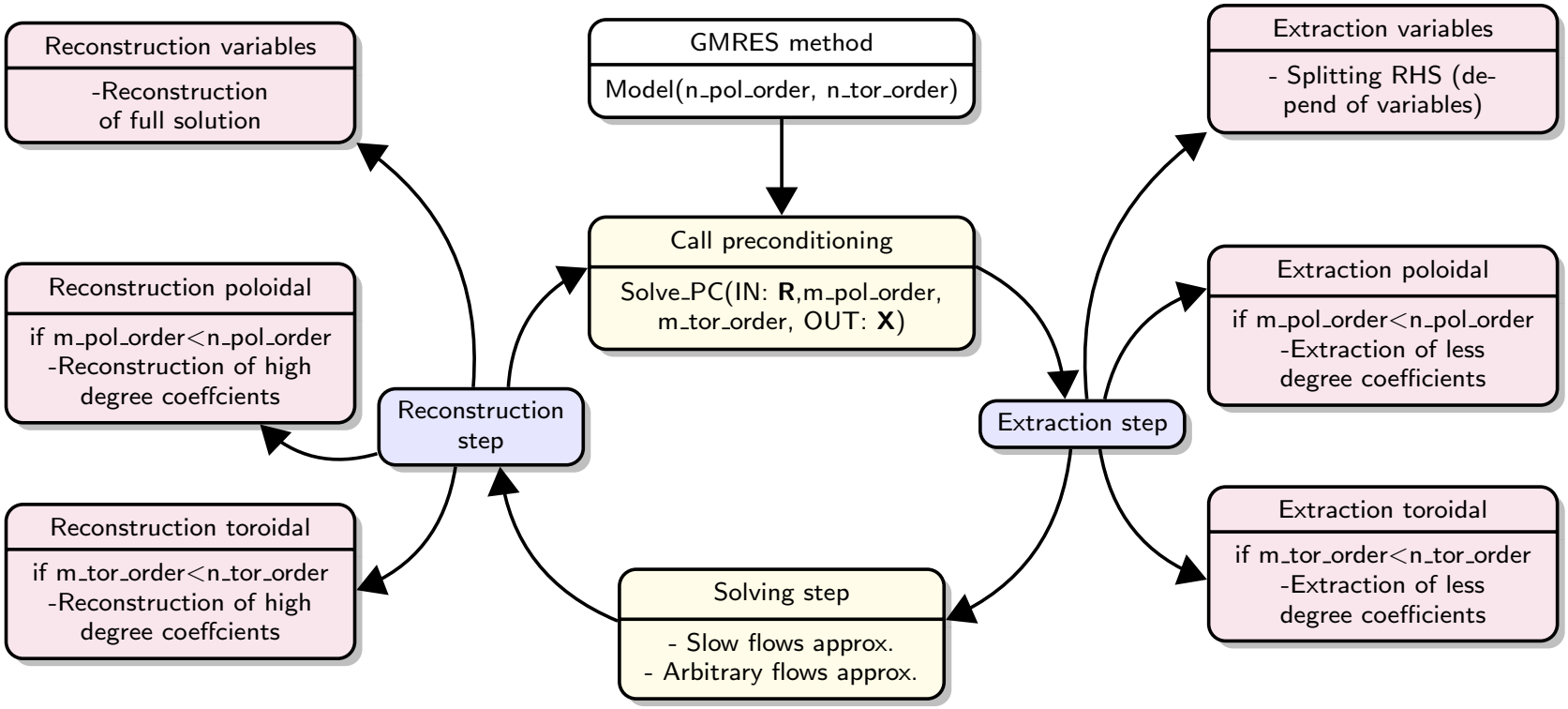
Applications to MHD PC

- We can call the preconditioning with
  - poloidal and toroidal orders of the B-Splines smaller than the orders used for the full model.
  - poloidal and toroidal regularity of the B-Splines different than the regularity used for the full model.
  - less Fourier harmonics than for the full model (we keep the coupling terms but neglect harmonics).
- Some restriction and interpolation operators must be added in the "extraction" and "reconstruction" steps.

- **Remark**: At the end, the user could choose the order and number of Harmonics for the PC (different that for the model) and adapt these parameters during the simulation.

Algorithm of the adaptive PhyBas Preconditioning step

- Algorithm and implementation of the $APB(x)$ preconditioning:



- In the future, it will be important to perform the extraction and reconstruction parts.

Conclusion

Conclusion

Conclusion:

- The idea to design a PC is to write the solving step as a suitability of simple operators (easy to invert) using splitting and reformulation (second order formulation) methods.
- The possible approximations gives the PC algorithm.
- **Problem**: the proposed method is dependent of the problem and use lot a methods (CG, MG, GMRES etc) $\Longrightarrow$ lot of work to treat all the models.

Possible approximations:

- **Solving approximation**: each sub step can be solve with a small accuracy.
- **Physical approximation**: each subsystem can be simplified to obtain well-conditioned operators (necessary in the MHD case).
- **Discretization approximation**: the systems of the PC can be solved with less order numerical methods or coarser grids (with extraction and reconstruction operator).
- **Multi-discretization approximation**: the PC models and the model can be discretized with different methods (finite element for PC and DG for the full system).

Others applications:

- **Shallow water equations and ocean flows**: Cemracs 2015 Project.
- **Radiative transfer**: project with CEA (DAM).

Program of works

Program June-August:

- Add a parallel multigrid solver in DJANGO.
- Optimisation of the PC (big optimization for matrices construction and extraction).
- Implementation of the PhyBas PC for the 2D and 3D cylindrical current Hole.
- Implementation of the JOREK PC for the 3D current Hole and comparison.

Program middle 2015- middle 2016:

- Implementation of the Free-Matrix methods.
- Optimisation and OpenMp parallelization.
- Validation of the 199 model interfacing JOREK and Django with restart files.
- Implementation of the adaptive PC (choice of the discretization can be different between the PC and the full model).

Program middle 2016 - end of 2017:

- Extension to the model 303 and Diamagnetic MHD.
- Implementation in JOREK (V 3.0 ?)

Call for help

- 3D Current Hole model :

$$\begin{cases} \partial_t \dfrac{1}{R^2}\psi = \dfrac{1}{R}[\psi, u] - \dfrac{F_0}{R^2}\partial_\phi u + \dfrac{\eta}{R^2} j \\ \nabla \cdot (\partial_t \nabla_{pol} u) = \dfrac{1}{R}[R^2 w, u] + \dfrac{1}{R}[\psi, j] - \dfrac{F_0}{R^2}\partial_\phi j + \nabla \cdot (\nu \nabla_{pol} w) \\ w = \nabla \cdot (\nabla_{pol} u), \quad j = \Delta^* \psi \end{cases}$$

- We want compare the new preconditioning and the preconditioning of JOREK using one of these model.

Find a test case

- Model: 3D Current Hole or 3D 199 model.
- Initialization : Grad-Safranov with analytical RHS + perturbation toroidal mods.
- Boundary condition : homogeneous Dirichlet.
- Mesh : Circle or D-shape.
- Physic : nonlinear coupling between the modes in the nonlinear phase.

Call for help

- 3D 199 model:

$$
\begin{cases}
\partial_t \dfrac{1}{R^2}\psi = \dfrac{1}{R}[\psi, u] - \dfrac{F_0}{R^2}\partial_\phi u + \dfrac{\eta}{R^2} j \\
\nabla \cdot (\widehat{\rho}\nabla_{pol}\partial_t u) = \dfrac{1}{2R}[R^2|\nabla_{pol} u|^2, \widehat{\rho}] + \dfrac{1}{R}[\widehat{\rho}R^2 w, u] - \dfrac{1}{R}[R^2, p] + \dfrac{1}{R}[\psi, j] - \dfrac{F_0}{R^2}\partial_\phi j \\
+\nabla \cdot (\nu \nabla_{pol} w) \\
\partial_t \rho = R[\rho, u] + 2\rho\partial_Z u \\
\partial_t T = R[T, u] + 2(\gamma - 1) T \partial_Z u \\
w = \nabla \cdot (\nabla_{pol} u), \quad j = \Delta^* \psi
\end{cases}
$$

- We want compare the new preconditioning and the preconditioning of JOREK using one of these model.

Find a test case

- Model: 3D Current Hole or 3D 199 model.
- Initialization : Grad-Safranov with analytical RHS + perturbation toroidal mods.
- Boundary condition : homogeneous Dirichlet.
- Mesh : Circle or D-shape.
- Physic : nonlinear coupling between the modes in the nonlinear phase.