Optimal preconditioning for the bidomain model

Charles Pierre

Laboratoire de Mathématiques Appliquées de Pau,

Université de Pau et des pays de l'Adour

Ottawa, May 2014

Outline

Optimal preconditioning

The bidomain model

Bidomain model preconditioning

Numerical results

Section 1

Optimal preconditioning

General problem

- We have to solve a linear problem,

$$Ax = b,$$

of large size $N \gg 1$ (N is the size of the matrix A).

- we search for a numerical method that optimises the resolution cost as,

$$N \longrightarrow +\infty.$$

- **Definition.** If we have :

$$\text{method cost} = O(f(N)),$$

we say that : *"f is the* **complexity** *of the method"* , (relatively to N and as $N \longrightarrow +\infty$).

Precisions on the problem $Ax = b$

- The matrix A comes from the discretusation of a PDE :
  - A is sparse,
  - the matrix vector product,

    $$x \longrightarrow Ax,$$

    has a **linear complexity**, i.e. $O(N)$.

- We resort to iterative solvers :
  - each iteration of the solver has a $O(N)$ complexity,
  - global cost $= O(\text{Iteration number} \times N)$,
  - **HERE : optimal complexity** $=O(N)$ (i.e. linear).

- As $N \longrightarrow +\infty$, the mesh size $h \longrightarrow 0$:
  - $\text{Cond}(A) \longrightarrow +\infty$ as $N \longrightarrow +\infty$,
  - it implies that iteration number $\longrightarrow +\infty$ as $N \longrightarrow +\infty$, **in absence of preconditioning**.

Precisions on the problem $Ax = b$

- **Assumption :** the matrix A is Symmetric Positive Definite, a standard iterative solver then is the **Preconditioned Conjugate Gradient** algorithm.

- To solve the preconditioned system,

$$PAx = Pb, \qquad P := \text{preconditioner},$$

  the PCG algorithm performs at each iteration :
  - 1 product $x \longrightarrow Ax \quad \Longrightarrow \quad O(N)$ complexity,
  - 1 product $x \longrightarrow Px \quad \Longrightarrow \quad O(N)$ complexity,
  - other less significant operations (2 scalar products, ...).

- **Proposition.** Let $\kappa = \text{Cond}(PA)$, then :

$$\text{iteration number} = O(\kappa^{1/2}) \quad \text{as} \quad \kappa \longrightarrow +\infty.$$

- **Corollary.** Optimal complexity is reached if we can bound κ independently of N (i.e. of the mesh size).

Illustartion 1 : elliptic problems

- Consider the Poisson problem,

$$-\Delta u = f,$$

  - following some classical discretisation method,
  - on a space grid of size h,
  - we have to solve,

$$Ax = b.$$

- **Property :** Cond(A) $= O(h^{-2})$.

$$\text{Elliptic equations} \Longrightarrow \text{complexity} = O(h^{-1}N).$$

- Complexity, in terms of N :

dim	without preconditioning	optimal
1	N^2	N
2	$N^{1+1/2}$	N
3	$N^{1+1/3}$	N

Illustartion 1 : elliptic problems

- Consider the Poisson problem,

$$-\Delta u = f,$$

  - following some classical discretisation method,
  - on a space grid of size h,
  - we have to solve,

$$Ax = b.$$

- **Property :** Cond(A) $= O(h^{-2})$.

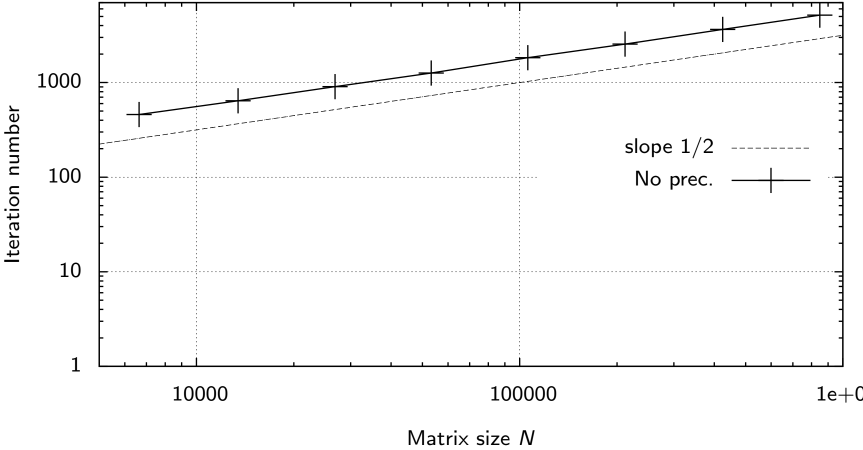
$$\text{Elliptic equations} \Longrightarrow \text{complexity} = O(h^{-1}N).$$

- Complexity, in terms of h :

dim	without preconditioning	optimal
1	h^{-2}	h^{-1}
2	h^{-3}	h^{-2}
3	h^{-4}	h^{-3}

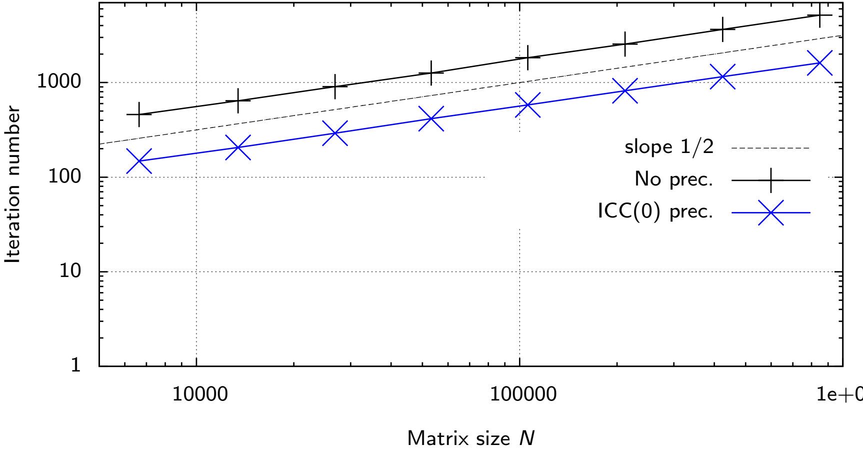
Illustartion 1 : elliptic problems

- Counting the iteration number,
- 8 meshes from 6 600 to 850 000 nodes.



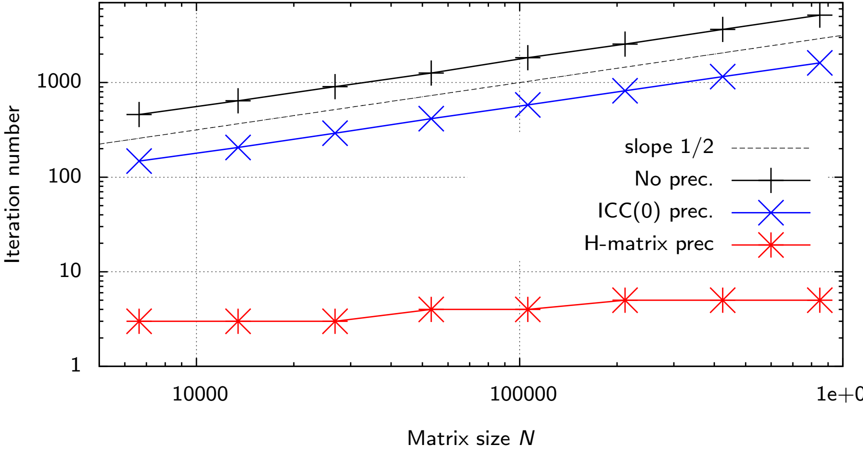
Illustartion 1 : elliptic problems

- Counting the iteration number,
- 8 meshes from 6 600 to 850 000 nodes.



Illustartion 1 : elliptic problems

- Counting the iteration number,
- 8 meshes from 6 600 to 850 000 nodes.



Illustartion 1 : elliptic problems

- Counting the iteration number,
- 8 meshes from 6 600 to 850 000 nodes.

N	6 600	105 000	850 000
no prec., inversion CPU	4.10^{-2}	4.8	208
H-matrix prec., inversion CPU	4.10^{-3}	0.12	1.2
H-matrix prec., assembling CPU	7.10^{-2}	1.7	17.2

Illustartion 2 : parabolic problems

- Consider the heat equation, $\partial_t u - \Delta u = 0$.
  - descretisd for instantce using an implicit Euler time scheme,
  - on a space/time grid of size h, Δ,
  - we have to solve at each time step n :

$$Au^{n+1} = b^n.$$

- **Property :** if $\Delta t = O(h)$ then Cond(A) $= O(h^{-1})$.

  Parabolic equations $\Longrightarrow$ complexity $= O(h^{-1/2}N)$.

- Complexity, in terms of N :

dim	without preconditioning	optimal
1	$N^{1+1/2}$	N
2	$N^{1+1/4}$	N
3	$N^{1+1/6}$	N

Illustartion 2 : parabolic problems

- Consider the heat equation, $\partial_t u - \Delta u = 0$.
  - descretisd for instantce using an implicit Euler time scheme,
  - on a space/time grid of size h, Δ,
  - we have to solve at each time step n :

$$Au^{n+1} = b^n.$$

- **Property :** if $\Delta t = O(h)$ then $\mathrm{Cond(A)} = O(h^{-1})$.

$$\text{Parabolic equations} \Longrightarrow \text{complexity} = O(h^{-1/2}N).$$

- Complexity, in terms of h :

dim	without preconditioning	optimal
1	$h^{-1.5}$	h^{-1}
2	$h^{-2.5}$	h^{-2}
3	$h^{-3.5}$	h^{-3}

- **Warning** : number of time steps $= O(1/h)$, the global complexity is multiplied by h^{-1}.

Optimal preconditioners

- For elliptic and parabolic equations,
  - multigrid preconditioners,
  - Hierarchical preconditioners (H-Matrix),

  are *nearly* optimal :
  - solver complexity $= O(N \log(N)^{\alpha})$,
  - preconditioner assembling/storage $= O(N)$.

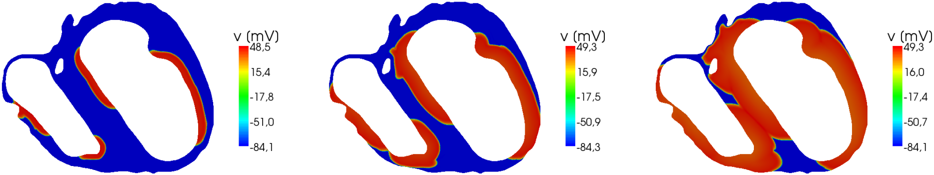
- References :
  - W. HackBush, *Multi-Grid Methods and Applications*
  - L. Grasedyck and W. Hackbusch, *Construction and Arithmetics of H-Matrices*

Section 2

The bidomain model

The bidomain model purpose, 1/3

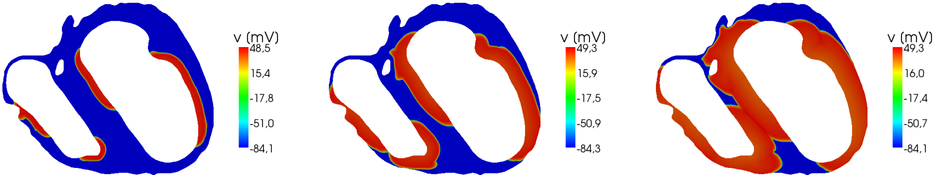
- Describes spreading of potential waves in the heart (2D slice) :



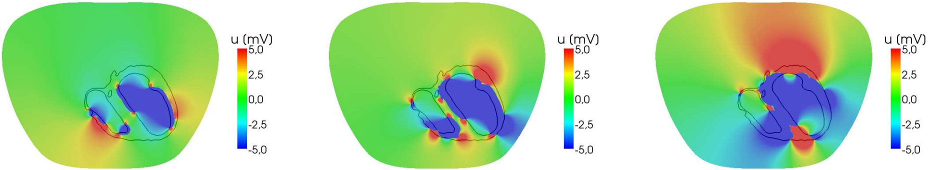
- First unknown v=*transmembrane potential*,
  - rest state, $v \simeq -90$ mV (in blue),
  - excited state, $v \simeq 50$ mV (in red),
  - propagation of sharp *depolarisation* wave fronts from rest to excited state.
- v is the difference between two potentials,

$$v = u_1 - u_e \quad \text{with} \quad \left\{ \begin{array}{ll} u_e & \text{extra-cellular potential} \\ u_i & \text{intra-cellular potential} \end{array} \right. ,$$

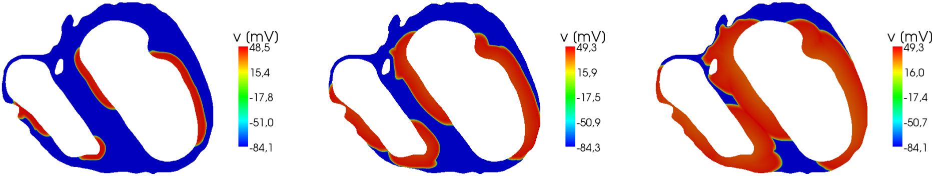
The bidomain model purpose, 2/3



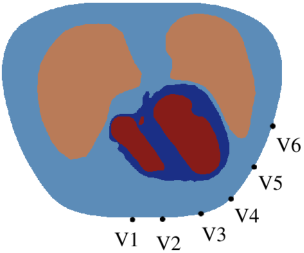
- Describes the extra-cellular potential evolution inside and outside the heart
- Second unknown u_e=*extra cellular potential*.

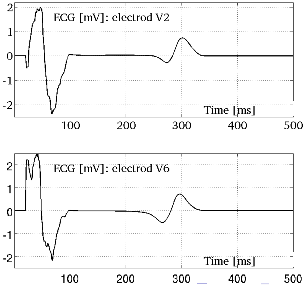


The bidomain model purpose, 3/3



- Describes body surface potential = electro-cardiogram (ECG).





Cellular modelling

- Models for the cellular transmembrane potential

$$v = u_1 - u_e, \quad \text{with :} \quad \begin{cases} u_e & \text{extra-cellular potential} \\ u_i & \text{intra-cellular potential} \end{cases} .$$

- Equations,

$$\begin{aligned} \partial_t v &= -I_{ion}(v, \mathbf{w}) + I_{stim}(x, t), \\ \partial_t \mathbf{w} &= g(v, \mathbf{w}), \end{aligned}$$

- with,
  - $\mathbf{w} \in \mathbb{R}^p$ = gating variable,
  - $I_{ion}(v, \mathbf{w})$ = reaction term
  - $I_{stim}(x, t)$ = source term (stimulation).

Equations : at the heart scale

Three unknowns :

- u_e extra cellular potential,
- v transmembrane potential,
- $\mathbf{w}$ gating variables.

For simplicity :

- the heart is isolated,
- the physical parameters are taken equal to 1, (excepting the conductivities).

Equations : at the heart scale

Three unknowns :

- u_e extra cellular potential,
- v transmembrane potential,
- $\mathbf{w}$ gating variables.

For simplicity :

- the heart is isolated,
- the physical parameters are taken equal to 1, (excepting the conductivities).

- First equation : elliptic type, current balance,

$$\mathrm{div}\Big(\; G_i \nabla u_i \; + \; G_e \nabla u_e \; \Big) = 0.$$

- $G_{i,e}$ = conductivity tensor of the intra/extra cellular media,
- $G_{i,e}$ non isotropic and heterogeneous, they follow the heart fibers,
- $\mathrm{div}\big(G_i \nabla u_i\big)$ current from the intra to the extra cellular media.

Equations : at the heart scale

Three unknowns :

- u_e extra cellular potential,
- v transmembrane potential,
- **w** gating variables.

For simplicity :

- the heart is isolated,
- the physical parameters are taken equal to 1, (excepting the conductivities).

- First equation : elliptic type, rewritten in u_e, v,

$$\mathrm{div}\Big((G_i + G_e)\nabla u_e\Big) + \mathrm{div}\big(G_i \nabla v\big) = 0.$$

Equations : at the heart scale

Three unknowns :

- u_e extra cellular potential,
- v transmembrane potential,
- $\mathbf{w}$ gating variables.

For simplicity :

- the heart is isolated,
- the physical parameters are taken equal to 1, (excepting the conductivities).

- First equation : elliptic type,

$$\mathrm{div}\big((G_i + G_e)\nabla u_e\big) + \mathrm{div}(G_i \nabla v) = 0.$$

- Second equation : parabolic type,

$$\partial_t v \ -\mathrm{div}(G_i \nabla u_e) - \ \mathrm{div}(G_i \nabla v) \ = \ I_{stim}(x,t) - I_{ion}(v, \mathbf{w}).$$

Equations : at the heart scale

Three unknowns :

- u_e extra cellular potential,
- v transmembrane potential,
- $\mathbf{w}$ gating variables.

For simplicity :

- the heart is isolated,
- the physical parameters are taken equal to 1, (excepting the conductivities).

- First equation : elliptic type,

$$\mathrm{div}\big((G_i + G_e)\nabla u_e\big) + \mathrm{div}(G_i \nabla v) = 0.$$

- Second equation : parabolic type,

$$\partial_t v \; - \mathrm{div}(G_i \nabla u_e) - \; \mathrm{div}(G_i \nabla v) \; = \; I_{stim}(x,t) - I_{ion}(v, \mathbf{w}).$$

- Third equation = cellular model,

$$\partial_t \mathbf{w} = g(v, \mathbf{w}).$$

Preconditioning the bidomain model : why ?

- Mathematical nature of the model :
  - coupling of elliptic/parabolic type equations,
  - conditioning in $O(h^{-2})$ at the discrete level.

Preconditioning the bidomain model : why ?

- Mathematical nature of the model :
  - coupling of elliptic/parabolic type equations,
  - conditioning in $O(h^{-2})$ at the discrete level.

- Particular case, **equal anisotropy ratio** : $G_i = \lambda G_e$:
  - the model reduces to a single parabolic equation,

$$\partial_t v \quad - \operatorname{div}(G_m \nabla v) \; = \; I_{stim}(x,t) - I_{ion}(v, \mathbf{w}),$$

  with $G_m = (G_i^{-1} + G_e^{-1})^{-1}$.

  - This simplified model is known as the **monodomain model**.
  - Conditioning in $O(h^{-1})$ at the discrete level.

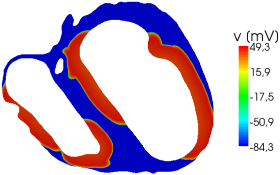
- General case :
  - one cannot get rid of the elliptic/parabolic nature of the model,
  - poor conditioning in $O(h^{-2})$ is inevitable.

Preconditioning the bidomain model : why ?

- Mathematical nature of the model :
  - coupling of elliptic/parabolic type equations,
  - conditioning in $O(h^{-2})$ at the discrete level.

- Physiological nature of depolarisation :
  - stiffness in space and in time of the potential waves,
  - resort to (very) fine grids in space and in time.

2D numerical illustration :



Preconditioning the bidomain model : why ?

- Mathematical nature of the model :
  - coupling of elliptic/parabolic type equations,
  - conditioning in $O(h^{-2})$ at the discrete level.

- Physiological nature of depolarisation :
  - stiffness in space and in time of the potential waves,
  - resort to (very) fine grids in space and in time.

- Conclusion :

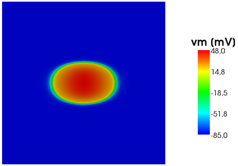
  ill conditioning + fine space and time grids

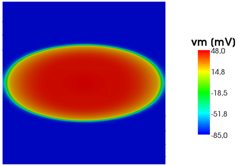
  $\Longrightarrow$ high computational costs

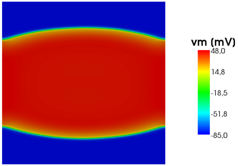
Preconditioning the bidomain model : why ?

Numerical illustration on a square :

- simulation of a (stiff) depolarisation wave,
- P^1 finite element + semi implicit Euler,
- 9 meshes from 6 600 to 1.7 10^6 nodes,
- counting the average iteration number for one time step inversion.

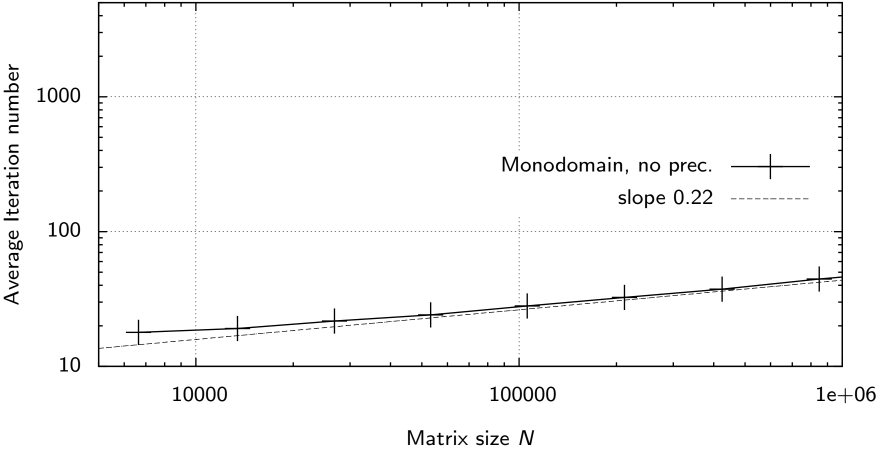






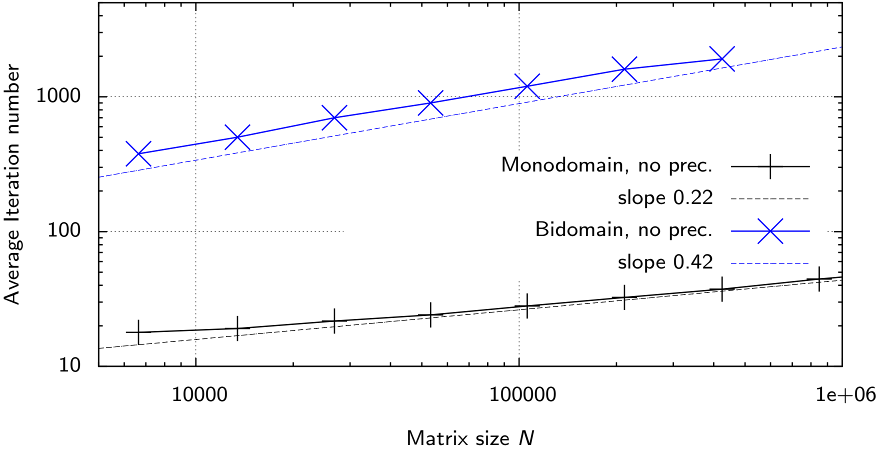
Preconditioning the bidomain model : why ?

Numerical illustration on a square :



Preconditioning the bidomain model : why ?

Numerical illustration on a square :



Preconditioning the bidomain model : why ?

Numerical illustration on a square :

- Average iteration number versus the mesh size N :

N	6 600	53 240	211 925	848 617
Monodomain, no prec.	17.9	24.1	32.5	44.5
Bidomain, no prec.	378	902	1604	2459

N	6 600	53 240	211 925	848 617
Monodomain, H-Mat prec.	3	3	3.3	4
Bidomain, H-Mat prec.	5	4.5	4	4.4

Preconditioning the bidomain model : why ?

Numerical illustration on a square :

- CPU ratio no prec./H-Mat prec. :

N	6 600	53 240	211 925	848 617
Monodomain	0.7	0.8	1.9	3.9
Bidomain	8.4	32.5	67	161

Section 3

Bidomain model preconditioning

Discretisation

- Continuous model,

$$\begin{aligned} \mathrm{div}\big((G_i + G_e)\nabla u_e\big) \;&+\; \mathrm{div}(G_i \nabla v) \;=\; 0 \\ \partial_t v \;-\; \mathrm{div}(G_i \nabla u_e) \;&-\; \mathrm{div}(G_i \nabla v) \;=\; f(x, t, \mathbf{w}). \end{aligned}$$

- For P^k Lagrange finite element we denote :
  - M the mass matrix,
  - S_α the stiffness matrix relatively to the conductivity tensor G_α.
- Numerical scheme, for a semi implicit Euler time stepping :

$$\begin{aligned} (S_i + S_e)U_e^{n+1} \;&+\; S_i V^{n+1} \;=\; 0 \\ M\frac{V^{n+1} - V^n}{\Delta t} \;+\; S_i U_e^{n+1} \;&+\; S_i V^{n+1} \;=\; F(\mathbf{W}^n, V^n, x, t^{n+1}). \end{aligned}$$

The linear problem

- To be solved at each time step,

$$A\, X^{n+1} = RHS^n. \tag{1}$$

- With :

$$A = \left[\begin{array}{cc} S_1 + S_e & S_i \\ S_i & \dfrac{1}{\Delta t} M + S_i \end{array} \right], \quad X^{n+1} = \left[\begin{array}{c} U_e^{n+1} \\ V^{n+1} \end{array} \right].$$

- **Property**, A is symmetric positive definite.

Bidomain preconditioners

(1) K. Mardal, B. Nielsen, X. Cai and A. Tveito,
An order optimal solver for the discretised bidomain equations
Num. Linear Algebra. Appl. 2007.

(2) L. Gerardo-Giorda, L. Mirabella, F. Nobile, M. Perego, A. Veneziani,
A model-based block-triangular preconditioner for the Bidomain system in electrocardiology
JCP 2009.

(3) C. Pierre,
Preconditioning the bidomain model with almost linear complexity,
JCP 2012.

Block Jacobi preconditioner

For the matrix $A = \begin{bmatrix} S_1 + S_e & S_i \\ S_i & \frac{1}{\Delta t} M + S_i \end{bmatrix}$,

Tveito et al. proposed in [1] the preconditioner,

$$P_J = \begin{bmatrix} (S_1 + S_e)^{-1} & 0 \\ 0 & (\frac{1}{\Delta t} M + S_i)^{-1} \end{bmatrix}.$$

- **Proposition 1.** P_J is an optimal preconditioner.
- Problem : $(S_1 + S_e)^{-1}$ and $(\frac{1}{\Delta t} M + S_i)^{-1}$ are unknown !
- **Proposition 2.** assuming that,
  - P_{ell} is an optimal preconditioner for $S_i + S_e$,
  - P_{para} is an optimal preconditioner for $\frac{1}{\Delta t} M + S_i$,

  then $\overline{P_J} = \begin{bmatrix} P_{ell} & 0 \\ 0 & P_{para} \end{bmatrix}$ is an optimal preconditioner for A.

The monodomain approximation

- Denoting $\mathcal{A}_i = \mathrm{div}(G_i \nabla \cdot)$, $\mathcal{A}_e = \mathrm{div}(G_e \nabla \cdot)$, the bidomain model reformulates into,

$$\partial_t v - \mathcal{A} v = F(v, \mathbf{w}, x, t), \quad \text{with} \quad \mathcal{A} = \left(\mathcal{A}_i^{-1} + \mathcal{A}_e^{-1}\right)^{-1}.$$

  - $\mathcal{A}$ is the **bidomain operator**,
  - $\mathcal{A}$ is symmetric positive and coercive but non local.

- Monodomain model approximation : $\mathcal{A} \simeq \mathrm{div}(G_m \nabla \cdot)$ with,

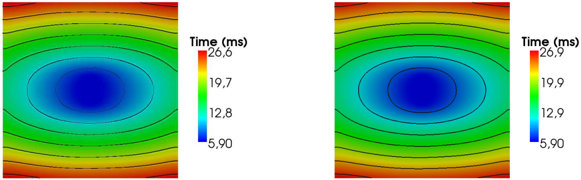
$$G_m = \left(G_i^{-1} + G_e^{-1}\right)^{-1}.$$

  - $\mathcal{A} = \mathrm{div}(G_m \nabla \cdot)$ in case of equal anisotropy ratio,
  - $\mathcal{A} \neq \mathrm{div}(G_m \nabla \cdot)$ in general.

The monodomain approximation

- Illustration (same test case as earlier on)

Activation times : bidomain vs monodomain



Relative error :
$e \simeq 1\%$,
in L^2 norm.

Block LU preconditioner

- The matrix $A = \begin{bmatrix} S_1 + S_e & S_i \\ S_i & \frac{1}{\Delta t} M + S_i \end{bmatrix}$, reads,

  $$A = LU, \quad L = \begin{bmatrix} S_i + S_e & 0 \\ S_i & K \end{bmatrix}, \quad U = \begin{bmatrix} 1 & (S_i + S_e)^{-1} S_i \\ 0 & 1 \end{bmatrix}$$

- K is the discretisation of the bidomain operator $\mathcal{A}$,

  $$K = \frac{M}{\Delta t} + \left(S_i^{-1} + S_e^{-1} \right)^{-1}$$

  - K is symmetric positive definite,
  - K^{-1} and K are full matrices in general.

- The monodomain approximation $\mathcal{A} \simeq \mathrm{div}(G_m \nabla \cdot)$ suggests :

  $$K \simeq K_m := \frac{M}{\Delta t} + S_m$$

  with S_m the stiffness matrix relatively to $G_m = \left(G_i^{-1} + G_e^{-1} \right)^{-1}$.

Block LU preconditioner

- We introduce the approximation $L \simeq L_m := \begin{bmatrix} S_i + S_e & 0 \\ S_i & K_m \end{bmatrix}$:

$$U^{-1} = \begin{bmatrix} 1 & -(S_i + S_e)^{-1} S_i \\ 0 & 1 \end{bmatrix}, \quad L_m^{-1} = \begin{bmatrix} (S_i + S_e)^{-1} & 0 \\ -K_m^{-1} S_i (S_i + S_e)^{-1} & K_m^{-1} \end{bmatrix}.$$

- The exact preconditionner P_{LU} is given by :

$$\boxed{P_{LU} := U^{-1} L_m^{-1}}.$$

- **Problem** K_m^{-1} and $(S_i + S_e)^{-1}$ are unknown. Let :
  - P_{ell} a preconditioner for $S_i + S_e$,
  - P_{para} a preconditioner for K_m.
- In practice we will use the preconditionner $\boxed{\overline{P}_{LU} := \overline{U}^{-1} \overline{L}_m^{-1}}$,

$$\overline{U}^{-1} = \begin{bmatrix} 1 & -P_{ell} S_i \\ 0 & 1 \end{bmatrix}, \quad \overline{L}_m^{-1} = \begin{bmatrix} P_{ell} & 0 \\ -P_{para} S_i P_{ell} & P_{para} \end{bmatrix}.$$

Solving $AX = Y$ in practice

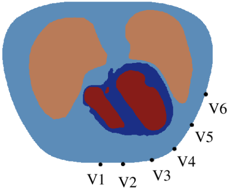
- Each iteration of the PCG algorithm performs :
  - One product $X \to AX = \begin{cases} 2 & \text{multiplication by} & S_i \\ 1 & \text{''} & S_i + S_e \\ 1 & \text{''} & M \end{cases}$
  - One product $X \to \overline{P}_{LU} X = \begin{cases} 2 & \text{multiplication by} & S_i \\ 2 & \text{''} & P_{ell} \\ 1 & \text{''} & P_{para} \end{cases}$
- One needs to define the preconditioners P_{ell} and P_{para} :
  - any preconditioner for elliptic or parabolic problems works,
  - we will use H-Matrix preconditioner.

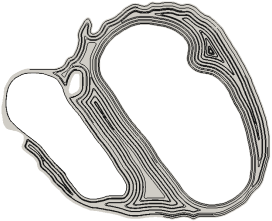
Section 4

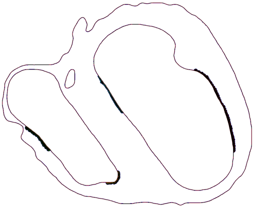
Numerical results

The 2D test case

- Segmented medical image (CT Scan),
- horizontal slice of the torso,
- heterogeneous thorax and anisotropic heart (fibrous organisation),
- 2 stimulation sites per ventricle (endo cardium),
- Luo and Rudy II ionic model, χ=1500 cm^{-1}.

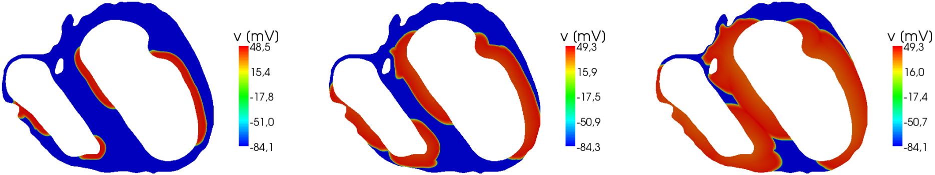




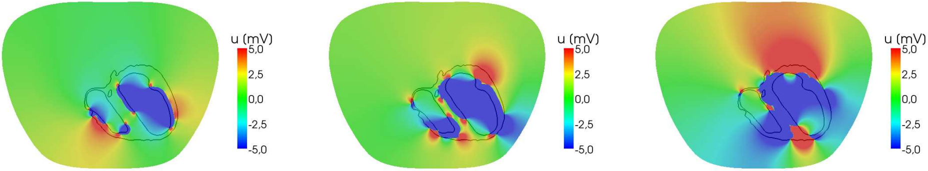


The 2D test case

- Four meshes considered : from 140 000 to 1 250 000 nodes.
- Transmembrane potential depolarisation wave,

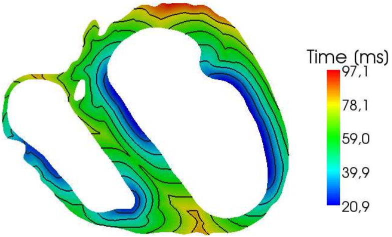


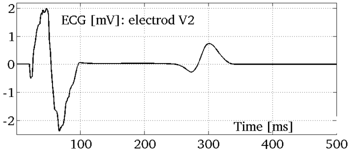
- extra-cellular potential field (same time instants),

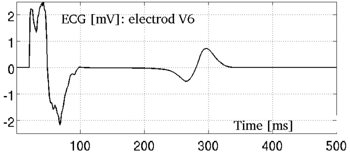


The 2D test case

- Four meshes considered : from 140 000 to 1 250 000 nodes.
- Activation time and ECG recordings,



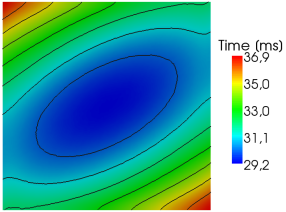




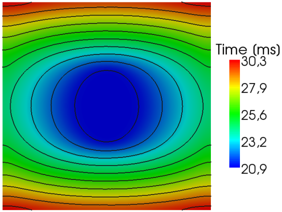
The 3D test case

- 1 cm^3 cubic slab of tissue,
- orthotropic anisotropy : horizontal fibres rotating of 90° from top to bottom,
- central stimulation,
- Luo and Rudy II ionic model, χ=500 cm^{-1},
- five meshes considered : from 500 to 1 250 000 nodes.

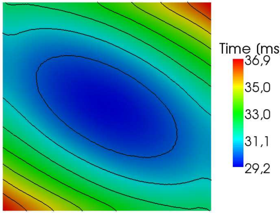
- Activation times :



$z = 1$



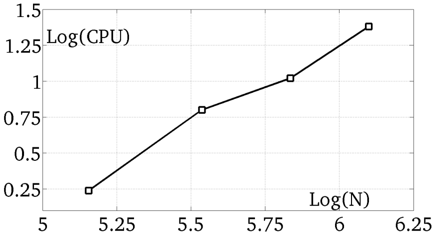
$z = 1/2$



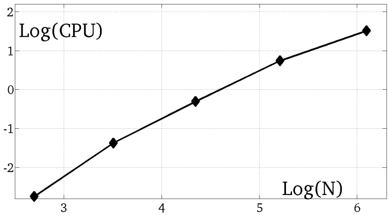
$z = 0$

Cost analysis

CPU dependence with the number N of nodes (Log/Log),



2D test case



3D test case

Cost analysis

- Log/Log growth rate of the CPU with N,

N	CPU	growth rate
143 053	1.57	
344 408	4.34	1.1
684 112	8.75	1.02
1 257 312	17.04	1.09

2D test case

N	CPU	growth rate
497	$1.8\ 10^{-3}$	
3 220	$4.2\ 10^{-2}$	1.7
22 256	$4.9\ 10^{-1}$	1.3
162 981	5.5	1.2
1 253 910	32.2	0.9

3D test case

Cost analysis

- Log/Log growth rate of the CPU with N,

N	CPU	growth rate
143 053	1.57	
344 408	4.34	1.1
684 112	8.75	1.02
1 257 312	17.04	1.09

2D test case

N	CPU	growth rate
497	$1.8\ 10^{-3}$	
3 220	$4.2\ 10^{-2}$	1.7
22 256	$4.9\ 10^{-1}$	1.3
162 981	5.5	1.2
1 253 910	32.2	0.9

3D test case

- Average number of iterations in PCG algorithm,

N	Iterations
143 053	3.0
344 408	3.0
684 112	3.0
1 257 312	3.0

2D test case

N	Iterations
497	2.0
3 220	2.7
22 256	3.0
162 981	3.0
1 253 910	2.0

3D test case

Cost calibration

- Linear complexity has been observed : this is asymptotic, we don't know the constants,
- to provide results that can be compared with other results, we calibrate the cost.
- **Calibration** = measure of the cost in terms of matrix/vector multiplication by A.
- This is relevent because linear complexity makes this measurment independent on N (matrix size).

$$\text{Inversion of } AX = Y \Leftrightarrow \left\{ \begin{array}{c} 50 \\ 80 \end{array} \right. \text{ matrix/vector multiplication in } \left\{ \begin{array}{c} 2D \\ 3D \end{array} \right. .$$

Conclusion

- Even with linear complexity for the linear system inversion, the global simulation cost remains large.

- Consider the following precision criterion :

  $e :=$ relative error on the activation times in L^2 norm.

  $e = O(h)$ here with h the mesh size (in space and time).

  in terms of precision, the global cost is

$$cost(e) = e^{DIM+1}$$

  → cubic complexity in 2D,

  → bi-quadratic complexity in 3D.