A hybrid parareal Monte-Carlo algorithm for parabolic problems

Jad Dabaghi, Yvon Maday, Andrea Zoia

CERMICS, École des Ponts ParisTech

Seminar CEA-DAM, November 22nd 2021

École des Ponts
ParisTech

informatics mathematics
Inria

Outline

Motivation

Simplified model for neutron transport in nuclear reactors.

Motivation

Simplified model for neutron transport in nuclear reactors.

Boltzmann equation $$\frac{1}{|v|}\frac{\partial \psi}{\partial t}(\boldsymbol{r}, E, \boldsymbol{\Omega}, t) + (\mathcal{A} - \mathcal{S} - \mathcal{F})\,\psi(\boldsymbol{r}, E, \boldsymbol{\Omega}, t) = Q(\boldsymbol{r}, \boldsymbol{\Omega}, E, t)$$

Monte Carlo simulation Lux & Koblinger (1991)

$\rightarrow$ intrinsically parallel : one replica $\Longleftrightarrow$ one processor

$\rightarrow$ preferred in large dimension

Motivation

Simplified model for neutron transport in nuclear reactors.

Boltzmann equation $$\frac{1}{|v|}\frac{\partial \psi}{\partial t}(\boldsymbol{r}, E, \boldsymbol{\Omega}, t) + (\mathcal{A} - \mathcal{S} - \mathcal{F})\,\psi(\boldsymbol{r}, E, \boldsymbol{\Omega}, t) = Q(\boldsymbol{r}, \boldsymbol{\Omega}, E, t)$$

Monte Carlo simulation Lux & Koblinger (1991)

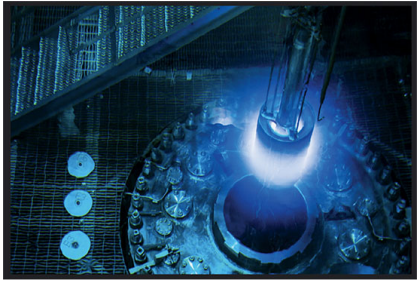
$\rightarrow$ intrinsically parallel : one replica $\Longleftrightarrow$ one processor

$\rightarrow$ preferred in large dimension

Can we speed-up a Monte Carlo resolution?

$\rightarrow$ time parallelization

Very complicated model: Start with a diffusion problem to understand the involved underlying mechanisms.



Outline

Model problem

Time-dependent diffusion equation with dirichlet boundary conditions:

$$\begin{cases} \partial_t u - \mathcal{D}\Delta u = 0 & \text{in} \quad \Omega \times [0,T], \\ u(\cdot,0) = u^0 & \text{in} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega \times [0,T]. \end{cases}$$

Model problem

Time-dependent diffusion equation with dirichlet boundary conditions:

$$\begin{cases} \partial_t u - \mathcal{D}\Delta u = 0 & \text{in} \quad \Omega \times [0, T], \\ u(\cdot, 0) = u^0 & \text{in} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega \times [0, T]. \end{cases}$$

Weak formulation: Find $u \in H_0^1(\Omega)$ such that

$$\langle \partial_t u, v \rangle_{H^{-1}(\Omega), H_0^1(\Omega)} + \mathcal{D} \int_\Omega \boldsymbol{\nabla} u \cdot \boldsymbol{\nabla} v \, \mathrm{dx} = 0 \quad \forall v \in H_0^1(\Omega)$$ Well-posed problem

Lions (1969), Dautrey & Lions (1985), Brezis (2011)

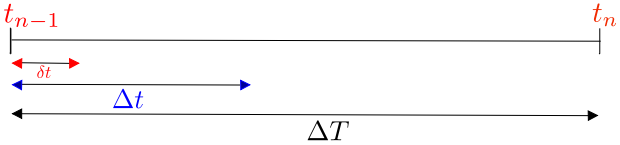
Parareal algorithm

Parallelization of the time variable ! Lions, Maday, Turinici (2001)

Parareal algorithm

Parallelization of the time variable ! Lions, Maday, Turinici (2001)

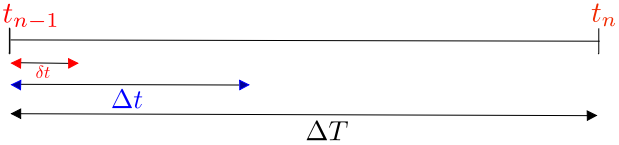
- Coarse sequential propagator $\mathcal{G}$ with associated time step Δt
- Fine parallel propagator $\mathcal{F}$ with associated time step δt, so that $\delta t \ll \Delta t$



Parareal algorithm

Parallelization of the time variable ! Lions, Maday, Turinici (2001)

- Coarse sequential propagator $\mathcal{G}$ with associated time step Δt
- Fine parallel propagator $\mathcal{F}$ with associated time step δt, so that $\delta t \ll \Delta t$



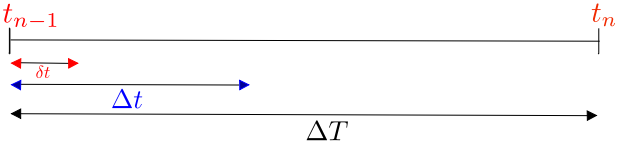
- Initialization : Compute a coarse solution at each time step n

$$\boldsymbol{U}^{n+1}_{k=0} := \mathcal{G}_{\Delta T}(\boldsymbol{U}^{n}_{k=0}), \quad \text{with} \quad \boldsymbol{U}^{0}_{k=0} = \boldsymbol{U}^{0} \qquad k : \text{ parareal iteration}$$

Parareal algorithm

Parallelization of the time variable ! Lions, Maday, Turinici (2001)

- Coarse sequential propagator $\mathcal{G}$ with associated time step Δt
- Fine parallel propagator $\mathcal{F}$ with associated time step δt, so that $\delta t \ll \Delta t$



- Initialization : Compute a coarse solution at each time step n

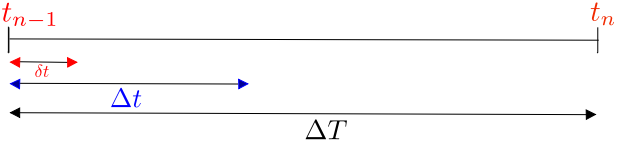
$$\boldsymbol{U}^{n+1}_{k=0} := \mathcal{G}_{\Delta T}(\boldsymbol{U}^{n}_{k=0}), \quad \text{with} \quad \boldsymbol{U}^{0}_{k=0} = \boldsymbol{U}^{0} \qquad k : \text{ parareal iteration}$$

- Compute parallel propagations: $\mathcal{F}_{\Delta T}(\boldsymbol{U}^{n}_{k}) \;\; \forall n = 0 \cdots N_t - 1$

Parareal algorithm

Parallelization of the time variable ! Lions, Maday, Turinici (2001)

- Coarse sequential propagator $\mathcal{G}$ with associated time step Δt
- Fine parallel propagator $\mathcal{F}$ with associated time step δt, so that $\delta t \ll \Delta t$



- Initialization : Compute a coarse solution at each time step n

$$\boldsymbol{U}^{n+1}_{k=0} := \mathcal{G}_{\Delta T}(\boldsymbol{U}^{n}_{k=0}), \quad \text{with} \quad \boldsymbol{U}^{0}_{k=0} = \boldsymbol{U}^{0} \qquad k : \text{ parareal iteration}$$

- Compute parallel propagations: $\mathcal{F}_{\Delta T}(\boldsymbol{U}^{n}_{k})\ \ \forall n = 0 \cdots N_t - 1$
- Parareal updates

$$\boxed{\boldsymbol{U}^{n}_{k} \approx u(t_n)} \qquad \boldsymbol{U}^{n+1}_{k+1} := \underbrace{\mathcal{G}_{\Delta T}(\boldsymbol{U}^{n}_{k+1})}_{\text{prediction}} + \underbrace{\mathcal{F}_{\Delta T}(\boldsymbol{U}^{n}_{k}) - \mathcal{G}_{\Delta T}(\boldsymbol{U}^{n}_{k})}_{\text{correction}} \quad \text{with} \quad \boldsymbol{U}^{0}_{k+1} := \boldsymbol{U}^{0}$$

Illustration of the procedure $n = 3$ and $k = 2$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}$, $u^2_{k=0}$, $u^3_{k=0}$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

3. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=1})$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

3. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=1})$
4. **Update:** $\boxed{u^1_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^1_{k=1})$, $\boxed{u^2_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^2_{k=1})$, $\boxed{u^3_{k=1}}$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

3. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=1})$
4. **Update:** $\boxed{u^1_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^1_{k=1})$, $\boxed{u^2_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^2_{k=1})$, $\boxed{u^3_{k=1}}$
5. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^1_{k=1}) - \mathcal{G}_{\Delta T}(u^1_{k=1}), \quad \mathcal{F}_{\Delta T}(u^2_{k=1}) - \mathcal{G}_{\Delta T}(u^2_{k=1}), \quad \mathcal{F}_{\Delta T}(u^3_{k=1}) - \mathcal{G}_{\Delta T}(u^3_{k=1})$$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

3. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=1})$
4. **Update:** $\boxed{u^1_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^1_{k=1})$, $\boxed{u^2_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^2_{k=1})$, $\boxed{u^3_{k=1}}$
5. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^1_{k=1}) - \mathcal{G}_{\Delta T}(u^1_{k=1}), \quad \mathcal{F}_{\Delta T}(u^2_{k=1}) - \mathcal{G}_{\Delta T}(u^2_{k=1}), \quad \mathcal{F}_{\Delta T}(u^3_{k=1}) - \mathcal{G}_{\Delta T}(u^3_{k=1})$$

6. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=2})$

Illustration of the procedure $n = 3$ and $k = 2$

1. Compute coarse approximations: $u^1_{k=0}, \quad u^2_{k=0}, \quad u^3_{k=0}$
2. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^0_{k=0}) - \mathcal{G}_{\Delta T}(u^0_{k=0}),\ \mathcal{F}_{\Delta T}(u^1_{k=0}) - \mathcal{G}_{\Delta T}(u^1_{k=0}),\ \mathcal{F}_{\Delta T}(u^2_{k=0}) - \mathcal{G}_{\Delta T}(u^2_{k=0})$$

3. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=1})$
4. **Update:** $\boxed{u^1_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^1_{k=1})$, $\boxed{u^2_{k=1}}$ and $\mathcal{G}_{\Delta T}(u^2_{k=1})$, $\boxed{u^3_{k=1}}$
5. Compute in parallel:

$$\mathcal{F}_{\Delta T}(u^1_{k=1}) - \mathcal{G}_{\Delta T}(u^1_{k=1}), \quad \mathcal{F}_{\Delta T}(u^2_{k=1}) - \mathcal{G}_{\Delta T}(u^2_{k=1}), \quad \mathcal{F}_{\Delta T}(u^3_{k=1}) - \mathcal{G}_{\Delta T}(u^3_{k=1})$$

6. Compute the prediction $\mathcal{G}_{\Delta T}(u^0_{k=2})$
7. **Update:** $\boxed{u^1_{k=2}}$ and $\mathcal{G}_{\Delta T}(u^1_{k=2})$ $\boxed{u^2_{k=2}}$ and $\mathcal{G}_{\Delta T}(u^2_{k=2})$ $\boxed{u^3_{k=2}}$

Some remarks on parareal

1. In general the two solvers are deterministic
2. Parareal converges for parabolic problems Gander, Vandewalle (2007)
3. Instability observed for hyperbolic problems Gander (2008)

Goal: Construct a hybrid parareal Monte Carlo algorithm for parabolic problems

Outline

Coarse propagator

$\mathcal{T}_h$: mesh of the domain Ω

$\mathcal{V}_h$: Lagrange nodes, $\mathcal{V}_h^{\text{int}}$: interior nodes,

$\mathcal{N}_h^{\text{int}}$: number of internal Lagrange nodes, N_e : number of elements

Coarse propagator

$\mathcal{T}_h$: mesh of the domain Ω

$\mathcal{V}_h$: Lagrange nodes, $\mathcal{V}_h^{\text{int}}$: interior nodes,

$\mathcal{N}_h^{\text{int}}$: number of internal Lagrange nodes, N_e : number of elements

The finite element propagator

$$X_h^p := \left\{ v_h \in \mathcal{C}^0(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h \right\} \subset H^1(\Omega)$$

$$X_{0h}^p := \left\{ v_h \in \mathcal{C}^0(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h,\ v_h|_{\partial\Omega} = 0 \right\} \subset H_0^1(\Omega)$$

Coarse propagator

$\mathcal{T}_h$: mesh of the domain Ω

$\mathcal{V}_h$: Lagrange nodes, $\mathcal{V}_h^{\text{int}}$: interior nodes,

$\mathcal{N}_h^{\text{int}}$: number of internal Lagrange nodes, N_e : number of elements

The finite element propagator

$$X_h^p := \left\{ v_h \in \mathcal{C}^0(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h \right\} \subset H^1(\Omega)$$

$$X_{0h}^p := \left\{ v_h \in \mathcal{C}^0(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h,\ v_h|_{\partial\Omega} = 0 \right\} \subset H_0^1(\Omega)$$

The discrete vector of unknowns : $\boldsymbol{U}_h^n \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}}$ satisfies $\boldsymbol{U}_h^n = \mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1})$ with

$$\mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1}) = [\mathbb{A}^n]^{-1} \times \boldsymbol{F}^{n-1}, \qquad \underbrace{[\mathbb{A}^n]^{-1}}_{\text{Stiffness matrix+mass matrix}} \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}, \mathcal{N}_h^{\text{int}}}, \quad \boldsymbol{F}^{n-1} \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}}$$

The cell centered finite volume propagator

$$\boldsymbol{U}_h^n := (\boldsymbol{U}_K^n)_{K\in\mathcal{T}_h}, \quad \textbf{one value per cell and time step}$$

The discrete vector of unknowns : $\boldsymbol{U}_h^n \in \mathbb{R}^{N_e}$ satisfies $\boldsymbol{U}_h^n = \mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1})$ with

$$\mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1}) = [\mathbb{A}^n]^{-1} \times \boldsymbol{F}^{n-1}, \quad \underbrace{[\mathbb{A}^n]^{-1}}_{\text{Sparse}} \in \mathbb{R}^{N_e,N_e}, \quad \boldsymbol{F}^{n-1} \in \mathbb{R}^{N_e}$$

The discontinuous Galerkin propagator

$\mathcal{N}_h^{\text{int}}$: total number of local internal degrees of freedom.

Discontinuous Galerkin space:

$$X_h^p := \left\{ v_h \in L^2(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h \right\} \not\subset H^1(\Omega),$$
$$X_{0h}^p := \left\{ v_h \in L^2(\Omega);\, v_h|_K \in \mathbb{P}_p(K)\ \forall K \in \mathcal{T}_h, v_h|_{\partial\Omega} = 0 \right\} \not\subset H_0^1(\Omega)$$

The discrete vector of unknowns : $\boldsymbol{U}_h^n \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}}$ satisfies $\boldsymbol{U}_h^n = \mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1})$ with

$$\mathcal{G}_{\Delta t}(\boldsymbol{U}_h^{n-1}) = [\mathbb{A}^n]^{-1} \times \boldsymbol{F}^{n-1}, \quad [\mathbb{A}^n]^{-1} \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}, \mathcal{N}_h^{\text{int}}}, \quad \boldsymbol{F}^{n-1} \in \mathbb{R}^{\mathcal{N}_h^{\text{int}}}$$

local matrix $[\mathbb{A}^n]_K^{-1}$ = stiffness matrix + mass matrix + consistency and stability terms.

Fine propagator : Monte-Carlo

Principle: It gives an approximation of

$$\int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_K \underbrace{f}_{PDF}(\boldsymbol{x}) g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

Fine propagator : Monte-Carlo

Principle: It gives an approximation of

$$\int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_K \underbrace{f}_{PDF}(\boldsymbol{x}) g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

Consider M particles and sample a collection $X_1, X_2, \cdots, X_M$ of M points from the PDF f. Denote by $\omega_i \in \mathbb{R}_+$ their statistical weight.

Compute $g(X_1)$,...,$g(X_M)$.

$$\int_K u(\boldsymbol{x})\,\mathrm{dx} = \overline{\mathbb{E}}\left[g(\boldsymbol{x})\right].$$

Fine propagator : Monte-Carlo

Principle: It gives an approximation of

$$\int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_K \underbrace{f}_{PDF}(\boldsymbol{x}) g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

Consider M particles and sample a collection $X_1, X_2, \cdots, X_M$ of M points from the PDF f. Denote by $\omega_i \in \mathbb{R}_+$ their statistical weight.

Compute $g(X_1)$,...,$g(X_M)$.

$$\int_K u(\boldsymbol{x})\,\mathrm{dx} = \overline{\mathbb{E}}\left[g(\boldsymbol{x})\right].$$

Law of large numbers: $\lim_{M\to+\infty} \frac{1}{M}\sum_{i=1}^{M} g(X_i) = \int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$

Fine propagator : Monte-Carlo

Principle: It gives an approximation of

$$\int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_K \underbrace{f}_{PDF}(\boldsymbol{x}) g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

Consider M particles and sample a collection $X_1, X_2, \cdots, X_M$ of M points from the PDF f. Denote by $\omega_i \in \mathbb{R}_+$ their statistical weight.

Compute $g(X_1)$,...,$g(X_M)$.

$$\int_K u(\boldsymbol{x})\,\mathrm{dx} = \overline{\mathbb{E}}\left[g(\boldsymbol{x})\right].$$

Law of large numbers: $\lim_{M\to+\infty} \frac{1}{M}\sum_{i=1}^{M} g(X_i) = \int_K u(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$

Central limit Theorem:

$$\text{error} \approx 1/\sqrt{M}.$$

Sampling in 1D

Direct inversion of the cumulative for a given PDF:

Sampling in 1D

Direct inversion of the cumulative for a given PDF:

$F : \Omega \to [0,1]$ such that $F(x) := \int_{-\infty}^{x} f(u)\,\mathrm{d}u$.

Let $\xi_1 \sim \mathcal{U}([0,1])$. Position of the particle: $X_i = F^{-1}(\xi_1)$.

Repeat M times the procedure.

Sampling in 1D

Direct inversion of the cumulative for a given PDF:

$F : \Omega \to [0,1]$ such that $F(x) := \int_{-\infty}^{x} f(u)\,\mathrm{d}u$.

Let $\xi_1 \sim \mathcal{U}([0,1])$. Position of the particle: $X_i = F^{-1}(\xi_1)$.

Repeat M times the procedure.

The table lookup method:

Probability each element: $\mathbb{P}([x_{i-1}, x_i]) = \int_{[x_{i-1},x_i]} f(x)\,\mathrm{d}x$,

Cumulative function: $F_i : \Omega \to [0,1],\ F_i = \sum_{j \le i} \mathbb{P}([x_{j-1}, x_j])$

Let $\xi_1 \sim \mathcal{U}([0,1])$. Identify the two intervals such that $F_{i-1} \le \xi_1 \le F_i$.

Position of the particle: $X_i = \dfrac{(x_i - x_{i-1})\,\xi_1 - x_i F_{i-1} + x_{i-1} F_i}{F_i - F_{i-1}}$.

Repeat M times the procedure.

Kernel Transport

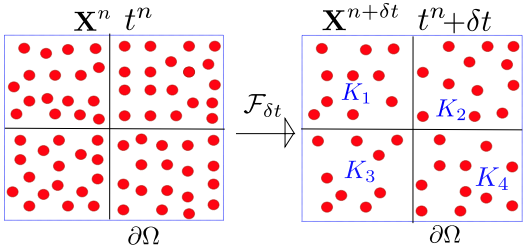
$(\boldsymbol{x}, t)$: position of the particle $\boldsymbol{x}$ at time t, $(\boldsymbol{x}', t')$: position of the particle $\boldsymbol{x}'$ at time t'

Density transition kernel:

$$T(\boldsymbol{x}', t' \to \boldsymbol{x}, t) := \frac{1}{\sqrt{2\pi\mathcal{D}\,(t - t')}} \exp\left(-\frac{(\boldsymbol{x} - \boldsymbol{x}')^2}{2\mathcal{D}\,(t - t')}\right).$$

Pratical formula for the brownian motion:

$$T(\boldsymbol{X}^{n+\delta t}, t^n + \delta t) = T(\boldsymbol{X}^n, t^n) + \sqrt{2\mathcal{D}\delta t}\,\mathcal{S}_n \quad \text{where} \quad \mathcal{S}_n \sim \mathcal{N}(0, 1)$$



Outline

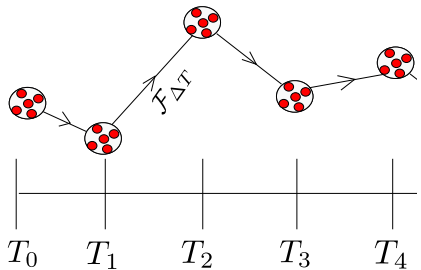
Hybrid parareal algorithm

Philosophy of the present work

Assume that 1000 processors are available.

Monte Carlo : We simulate 10^6 particules per processor. Precision : $1/\sqrt{10^9}$

1 replica = 1 processor



Required time for the propagation over a window ΔT: $\mathcal{F}_{\Delta T} = T/4$ seconds.

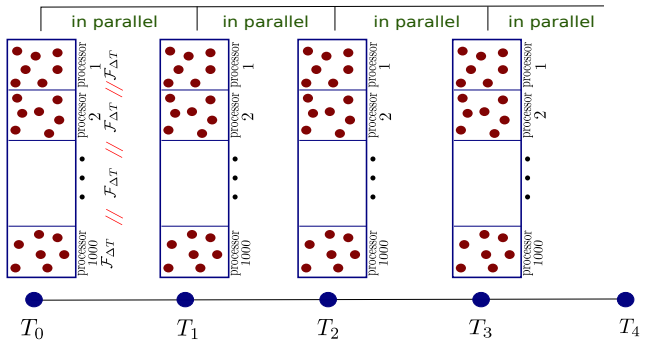
Total required time for each replica $= T$ seconds $\Rightarrow$ total time $\approx T$ secondes.

Assume now that 4000 processors are available.

precision: $1/\sqrt{4 \times 10^9}$ but statistical precision was already achieved...

Parareal : We employ the excess processors to parallelize the time variable !

- 4 **processors allocated to the time parallelization**
- For each time observable we have 1000 processors. For each of these processors we simulate 10^6 particles.



Precision :

$$\frac{1}{\sqrt{10^9}}$$

CPU cost :

$$\widetilde{T} = \frac{T}{4} \times \overline{k} \quad \text{with} \quad \overline{k} < 4$$

Other possibility: simulate a Borwnian motion on an interval $\left[0, \frac{4}{\overline{k}} T\right]$.

The hybrid algorithm

- **Coarse propagator :** Deterministic solver
- **Fine propagator:** Monte-Carlo solver: deterministic data + sampling + average

The numerical solution obtained for a replica $j \in [1, p]$ at parareal iteration k is denoted by $\boldsymbol{U}_{k,j}^{n+1}$

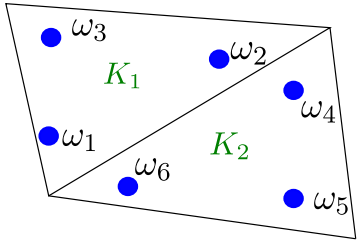
$$\boldsymbol{U}_{k}^{n+1} := \frac{1}{p} \sum_{j=1}^{p} \boldsymbol{U}_{k,j}^{n+1}$$

When $\boldsymbol{U}_{k,j}^{n+1}$ is computed, we need its statistical version for the computation of $\boldsymbol{U}_{k+1,j}^{n+2} = \mathcal{G}_{\Delta T}(\boldsymbol{U}_{k+1,j}^{n+1}) \times \dfrac{\mathcal{F}_{\Delta T}(\boldsymbol{U}_{k,j}^{n+1})}{\mathcal{G}_{\Delta T}(\boldsymbol{U}_{k,j}^{n+1})}$. Introduce bias in the Monte-Carlo solver.

Updating the statistical weights

Example: How avoid sampling $\boldsymbol{U}^3_{k=2} = \mathcal{G}_{\Delta T}(U^2_{k=2}) \times \frac{\mathcal{F}_{\Delta T}(\boldsymbol{U}^2_{k=1})}{\mathcal{G}_{\Delta T}(\boldsymbol{U}^2_{k=1})}$? We consider the statistical representation $\widetilde{\mathcal{F}_{\Delta T}(\boldsymbol{U}^2_{k=1})}$ before the average and we modify each of its particle weigths.

$$[\omega^3_{k=2}]_{i\in K} = [\omega_{\widetilde{\mathcal{F}_{\Delta T}(\boldsymbol{U}^2_{k=1})}}]_{i\in K} \times \left(\frac{\boldsymbol{U}^3_{k=2}}{\mathcal{F}_{\Delta T}(\boldsymbol{U}^2_{k=1})} \right) |_K$$



$$\omega_{1,2,3} \leftarrow \alpha_{K_1} \times \omega_{1,2,3} \qquad \alpha_{K_1} = \frac{[\mathbf{U}^2_{k=2}]|_{K_1}}{[\mathcal{F}_{\Delta T}(\mathbf{U}^1_{k=1})]|_{K_1}}$$

$$\omega_{4,5,6} \leftarrow \alpha_{K_2} \times \omega_{4,5,6} \qquad \alpha_{K_2} = \frac{[\mathbf{U}^2_{k=2}]|_{K_2}}{[\mathcal{F}_{\Delta T}(\mathbf{U}^1_{k=1})]|_{K_2}}$$

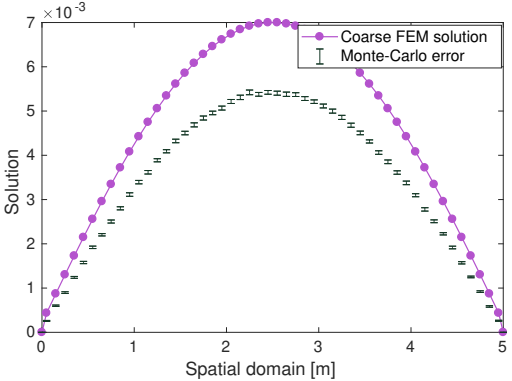
Outline

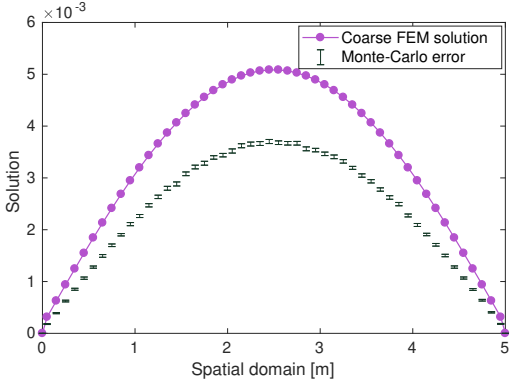
Numerical experiments

Final simulation time: $T = 10s$. **Diffusion coefficient:** $\mathcal{D} = 0.5m^2 \cdot s^{-1}$.

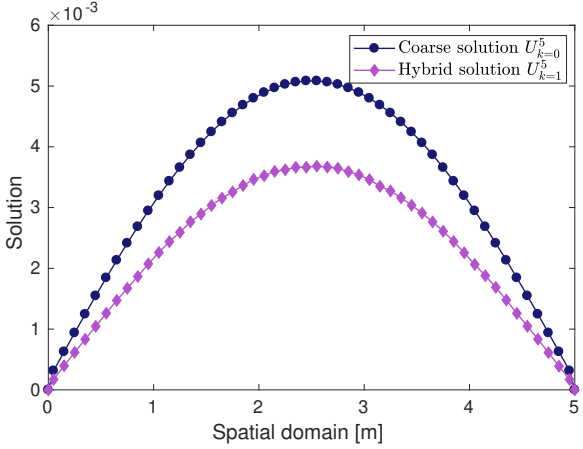
Coarse propagator: $\mathbb{P}_1$ FEM, $\Delta t = 2s$, **Fine propagator:** Monte-Carlo, $\delta t = 2 \times 10^{-4} s$.

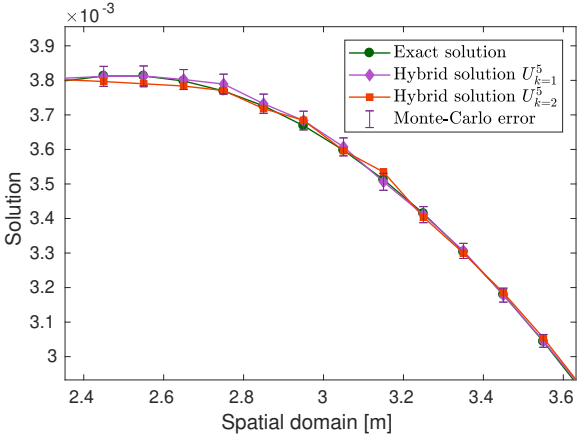
Initial condition: $u_0(x) = \frac{1}{L}$, **Number of particles:** 10^4, **Number of replicas:** 10^3





Hybrid solution





CPU time and convergence

Hybrid parareal resolution						
Number of processors time parallelization	Number of replicas for each parallel-in-time propagation	Number of particles for one replica j	CPU time $k=1$	CPU time $k=2$	Gain factor $k=1$	Gain factor $k=2$
5	10^2	10^5	335.76 s	537.16 s	4.92	3.04
5	10^3	10^4	33.05 s	53.1 s	4.96	3.09
5	10^4	10^3	3.39 s	5.49 s	4.97	3.07
5	10^5	10^2	0.35 s	0.58 s	5.08	3.02

A second test case

Final simulation time: $T = 14s$.

Deterministic propagator: $\mathbb{P}_1$ finite element, $\Delta t = 2s$.

Fine propagator: Monte-Carlo, $\delta t = 2 \times 10^{-4} s$.

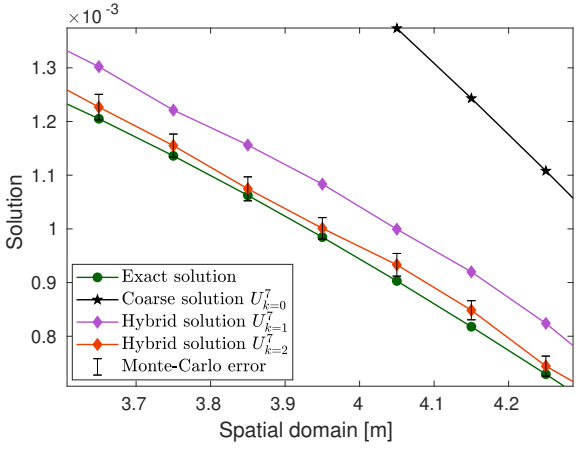
Diffusion coefficient MC: $\mathcal{D} = 0.5 m^2 \cdot s^{-1}$

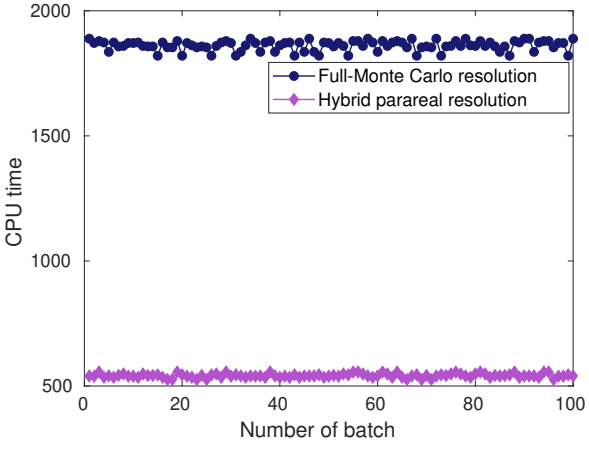
Diffusion coefficient FEM: $\mathcal{D} = 0.48 m^2 \cdot s^{-1}$

Initial condition: $u_0(x) = \frac{1}{L}\left(1 + \cos(\frac{\pi x}{L})\right)$.

Number of particles: 10^5, **Number of replicas:** 10^2

CPU time and convergence





A third test case

Final simulation time: $T = 50s$.

Deterministic propagator: $\mathbb{P}_1$ finite element, $\Delta t = 2s$.

Fine propagator: Monte-Carlo, $\delta t = 2 \times 10^{-3} s$.

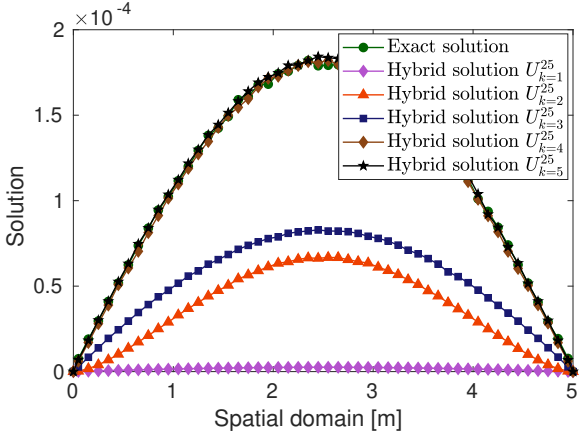
Diffusion coefficient MC: $\mathcal{D} = 0.25 m^2 \cdot s^{-1}$

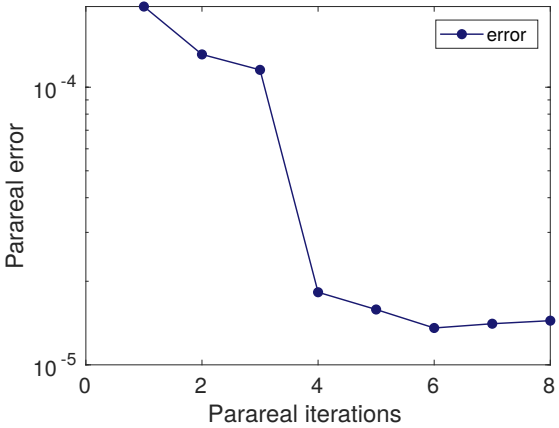
Diffusion coefficient FEM: $\mathcal{D} = 0.25 m^2 \cdot s^{-1}$

Initial condition: $u_0(x) = \dfrac{1}{L}$.

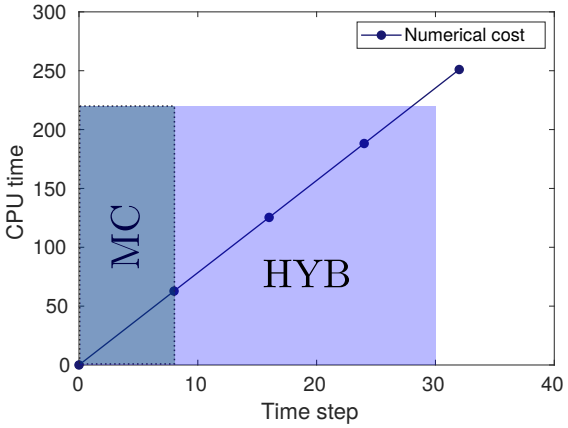
Number of particles: 10^5, **Number of replicas:** 10^3

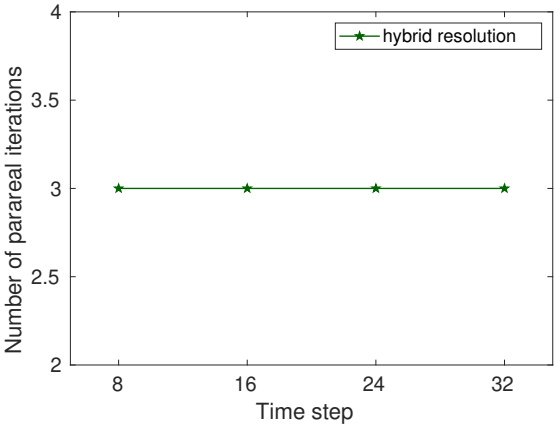
Convergence





Simulate longer times





J. DABAGHI, Y. MADAY, A. ZOIA, *A hybrid parareal Monte-Carlo algorithm for parabolic problems*. IN REVISION (2021)

Outline

Model problem

monokinetic 1D Boltzmann model

$$\frac{\partial \psi_+}{\partial t}(x,t) + \frac{\partial \psi_+}{\partial x}(x,t) + \Sigma_t \psi_+(x,t) = \left(\frac{\Sigma_s}{2} + \nu\frac{\Sigma_f}{2}\right)(\psi_+(x,t) + \psi_-(x,t))$$
$$\frac{\partial \psi_-}{\partial t}(x,t) - \frac{\partial \psi_-}{\partial x}(x,t) + \Sigma_t \psi_-(x,t) = \left(\frac{\Sigma_s}{2} + \nu\frac{\Sigma_f}{2}\right)(\psi_+(x,t) + \psi_-(x,t)).$$

- unknowns : ψ_+ and ψ_- : angular fluxes in direction $+1$ and -1
- $\Sigma_t = \Sigma_s + \Sigma_a + \Sigma_f$ (cross sections)
- ν : average number of neutrons emitted per fission

Fine propagator : The Monte Carlo algorithm

Fine propagator : The Monte Carlo algorithm

1. **Sampling the initial guess**

   → Position of particles (direct inversion of cumulative, table lookup, rejection...)

   → direction of particles. Select $\xi \sim \mathcal{U}([0,1])$. If $\xi \leq p^+$, $\omega_i = +1$. Otherwise, $\omega_i = -1$.

Fine propagator : The Monte Carlo algorithm

1. **Sampling the initial guess**

   $\rightarrow$ Position of particles (direct inversion of cumulative, table lookup, rejection...)

   $\rightarrow$ direction of particles. Select $\xi \sim \mathcal{U}([0,1])$. If $\xi \leq p^+$, $\omega_i = +1$. Otherwise, $\omega_i = -1$.

2. **Sampling the flights**

   Probability of collision between $s > 0$ and $s + ds > 0$ is $p(s)ds = \Sigma_t e^{-\Sigma_t s} ds$. The path is obtained by the direct inversion of the cumulative $s = -\frac{1}{\Sigma_t} \ln(1 - \xi)$ with $\xi \sim \mathcal{U}([0,1])$

Fine propagator : The Monte Carlo algorithm

1. **Sampling the initial guess**

   $\rightarrow$ Position of particles (direct inversion of cumulative, table lookup, rejection...)

   $\rightarrow$ direction of particles. Select $\xi \sim \mathcal{U}([0,1])$. If $\xi \leq p^+$, $\omega_i = +1$. Otherwise, $\omega_i = -1$.

2. **Sampling the flights**

   Probability of collision between $s > 0$ and $s + ds > 0$ is $p(s)ds = \Sigma_t e^{-\Sigma_t s} ds$. The path is obtained by the direct inversion of the cumulative $s = -\frac{1}{\Sigma_t} \ln(1 - \xi)$ with $\xi \sim \mathcal{U}([0,1])$

3. **Collision**

   $\xi_1 \sim \mathcal{U}([0,1])$. If $\xi_1 \leq \frac{\Sigma_s}{\Sigma_t}$ scattering event $\Rightarrow$ sample a new direction and flight. If $\frac{\Sigma_s}{\Sigma_t} < \xi_1 \leq \frac{\Sigma_s}{\Sigma_t} + \frac{\Sigma_a}{\Sigma_t}$ absorption event $\rightarrow$ the neutron is dead.
   If $\xi_1 > \frac{\Sigma_s}{\Sigma_t} + \frac{\Sigma_a}{\Sigma_t}$ fission event. The mother neutron is dead and $\overline{\nu}$ child neutrons are emitted. New directions and flights have to be sampled for each child neutrons.

Fine propagator : The Monte Carlo algorithm

1. **Sampling the initial guess**

   $\rightarrow$ Position of particles (direct inversion of cumulative, table lookup, rejection...)

   $\rightarrow$ direction of particles. Select $\xi \sim \mathcal{U}([0,1])$. If $\xi \leq p^+$, $\omega_i = +1$. Otherwise, $\omega_i = -1$.

2. **Sampling the flights**

   Probability of collision between $s > 0$ and $s + ds > 0$ is $p(s)ds = \Sigma_t e^{-\Sigma_t s} ds$. The path is obtained by the direct inversion of the cumulative $s = -\frac{1}{\Sigma_t} \ln(1 - \xi)$ with $\xi \sim \mathcal{U}([0,1])$

3. **Collision**

   $\xi_1 \sim \mathcal{U}([0,1])$. If $\xi_1 \leq \frac{\Sigma_s}{\Sigma_t}$ scattering event $\Rightarrow$ sample a new direction and flight. If $\frac{\Sigma_s}{\Sigma_t} < \xi_1 \leq \frac{\Sigma_s}{\Sigma_t} + \frac{\Sigma_a}{\Sigma_t}$ absorption event $\rightarrow$ the neutron is dead.
   If $\xi_1 > \frac{\Sigma_s}{\Sigma_t} + \frac{\Sigma_a}{\Sigma_t}$ fission event. The mother neutron is dead and $\overline{\nu}$ child neutrons are emitted. New directions and flights have to be sampled for each child neutrons.

4. **Monte Carlo computation**

   Compute an approximation of $\int_\Omega (\psi_+ + \psi_-)(x,t)\, dx = \int_\Omega \Phi(x,t)\, dx$

Coarse propagator : A reaction-diffusion model

$$\frac{\partial \phi}{\partial t}(x,t) - \mathcal{D}\frac{\partial^2 \phi}{\partial x^2}(x,t) + \Sigma_a \phi(x,t) + (\nu - 1)\Sigma_f \phi(x,t) = 0 \quad \text{in} \quad \Omega \times [0,T]$$

$$\phi(x,0) = \phi^0(x) \quad \text{in} \quad \Omega$$

$$\phi(x,t) = 0 \quad \text{on} \quad \partial\Omega \times]0,T[$$

Cell-centered finite volume method: a single constant value per cell: $\forall 1 \leq n \leq N_t - 1$ we let

$$\mathbf{\Phi}^n := (\phi_K^n)_{K \in \mathcal{T}_h} \in \mathbb{R}^{N_e} \quad \phi_K^n := \frac{1}{|K|}\int_K \phi^n(x)\, dx$$

By integration over the element K and using the Green's formula we obtain

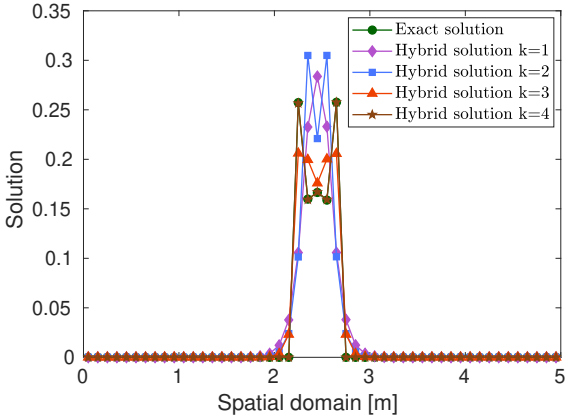
$$\frac{|K|}{\Delta t_n}\phi_K^n + \mathcal{D}\sum_{\sigma \in \mathcal{E}_K} \mathfrak{F}_{K,\sigma}^n - |\sigma|\frac{\phi_K^n}{d_{K\sigma}} + \Sigma_a \phi_K^n + (\nu - 1)\Sigma_f \phi_K^n = Q_K^{n-1} \quad \forall K \in \mathcal{T}_h.$$

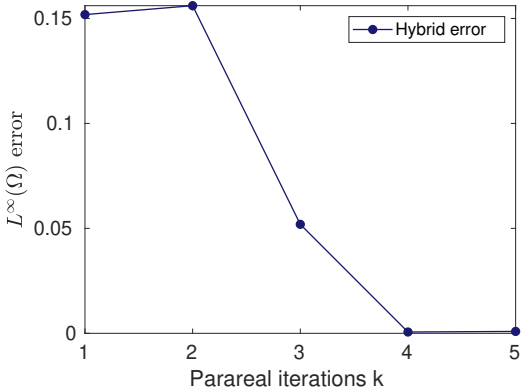
$\mathbf{\Phi}^n$ is solution to a linear system of equations

Numerical experiments

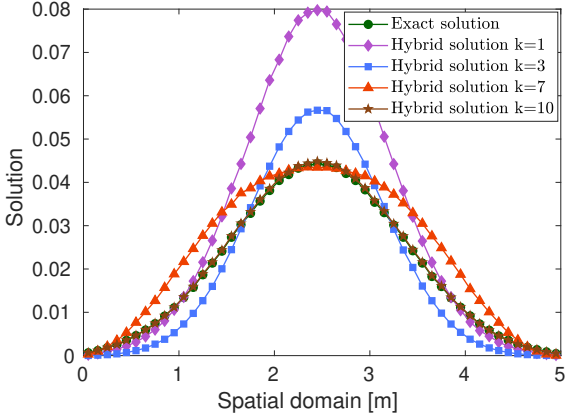
$\Sigma_a = \Sigma_f = 0$

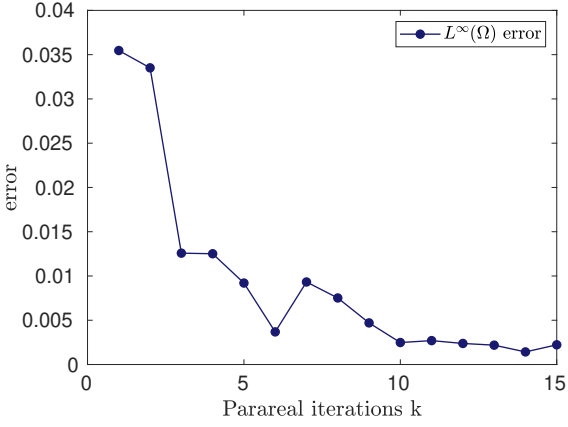
$n = 4$





$n = 80$





J. Dabaghi, Y. Maday, A. Zoia, *A hybrid parareal Monte-Carlo algorithm for the Boltzmann equation in neutronics.* In preparation (2021)

Outline

Conclusion

- We devised for the diffusion equation a hybrid parareal algorithm.
- Our approach reduces the CPU time of a Monte-Carlo simulation.

Ongoing work:

- Extension to the Boltzmann equation in neutronics with absorption and fission

Thank you for your attention!