A hybrid parareal Monte-Carlo algorithm for the parabolic time dependant diffusion equation

Jad Dabaghi, Yvon Maday, Andrea Zoia

Sorbonne Université & CEA Paris Saclay

WCCM, January 11-15, 2021

Outline

Motivation

Study the neutron transport in nuclear reactors

Motivation

Study the neutron transport in nuclear reactors

Model: Linear Boltzmann equation for the angular flux

$$\partial_t \Psi(t, \boldsymbol{x}, \boldsymbol{v}) + \boldsymbol{v} \cdot \nabla \Psi(t, \boldsymbol{x}, \boldsymbol{v}) + \sigma(\boldsymbol{x}, \boldsymbol{v}) \Psi(t, \boldsymbol{x}, \boldsymbol{v}) - \int_{\mathbb{R}^3} k(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{v}') \Psi(t, \boldsymbol{x}, \boldsymbol{v}') \, \mathrm{d}\boldsymbol{v}' = 0$$

Balance between the neutrons that are created and that disappear in the core.

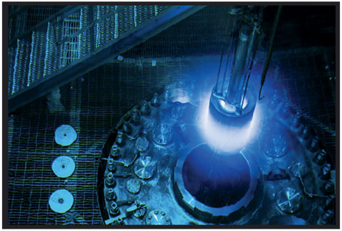
Motivation

Study the neutron transport in nuclear reactors

Model: Linear Boltzmann equation for the angular flux

$$\partial_t \Psi(t, \boldsymbol{x}, \boldsymbol{v}) + \boldsymbol{v} \cdot \nabla \Psi(t, \boldsymbol{x}, \boldsymbol{v}) + \sigma(\boldsymbol{x}, \boldsymbol{v}) \Psi(t, \boldsymbol{x}, \boldsymbol{v}) - \int_{\mathbb{R}^3} k(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{v}') \Psi(t, \boldsymbol{x}, \boldsymbol{v}') \, \mathrm{d}\boldsymbol{v}' = 0$$

Balance between the neutrons that are created and that disappear in the core.



Numerical resolution:

- Monte-Carlo approach
- Deterministic approach

Can we speed up a Monte-Carlo resolution?

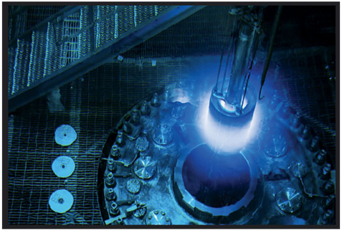
Motivation

Study the neutron transport in nuclear reactors

Model: Linear Boltzmann equation for the angular flux

$$\partial_t \Psi(t,\boldsymbol{x},\boldsymbol{v}) + \boldsymbol{v}\cdot\nabla\Psi(t,\boldsymbol{x},\boldsymbol{v}) + \sigma(\boldsymbol{x},\boldsymbol{v})\Psi(t,\boldsymbol{x},\boldsymbol{v}) - \int_{\mathbb{R}^3} k(\boldsymbol{x},\boldsymbol{v},\boldsymbol{v}')\Psi(t,\boldsymbol{x},\boldsymbol{v}')\,\mathrm{d}\boldsymbol{v}' = 0$$

Balance between the neutrons that are created and that disappear in the core.



Numerical resolution:

- Monte-Carlo approach
- Deterministic approach

Can we speed up a Monte-Carlo resolution?

Parareal-in-time algorithm $\Rightarrow$ important computational savings

Complicated problem... Start with a diffusion problem to understand the involved underlying mechanisms.

Outline

Model problem and parareal algorithm

Time dependent diffusion equation with dirichlet boundary conditions:

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

Model problem and parareal algorithm

Time dependent diffusion equation with dirichlet boundary conditions:

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

Parareal procedure: It constructs a sequence $\boldsymbol{u}_k^n := (\boldsymbol{u}_k^n)_{1\leq n\leq N}$ such that $\boldsymbol{u}_k^n \approx u^n$. It involves a coarse propagator $\mathcal{G}_{\Delta t}$ and a fine propagator $\mathcal{F}_{\delta t}$.

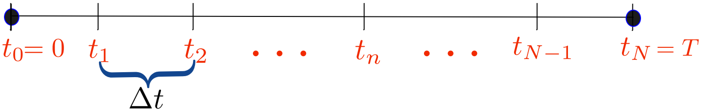
Model problem and parareal algorithm

Time dependent diffusion equation with dirichlet boundary conditions:

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

Parareal procedure: It constructs a sequence $\boldsymbol{u}_k^n := (\boldsymbol{u}_k^n)_{1 \leq n \leq N}$ such that $\boldsymbol{u}_k^n \approx u^n$. It involves a coarse propagator $\mathcal{G}_{\Delta t}$ and a fine propagator $\mathcal{F}_{\delta t}$.

Coarse time grid discretization:



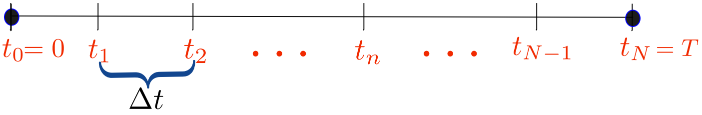
Model problem and parareal algorithm

Time dependent diffusion equation with dirichlet boundary conditions:

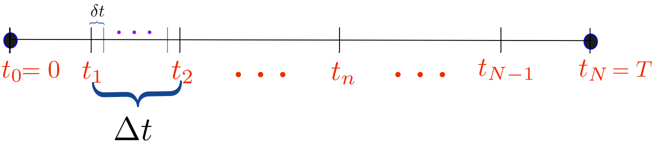
$$\begin{cases} \partial_t u - \mathcal{D}\Delta u = g & \text{in} \quad \Omega \times [0, T]\,, \\ u(\cdot, 0) = u^0 & \text{in} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega \times [0, T]. \end{cases}$$

Parareal procedure: It constructs a sequence $\boldsymbol{u}_k^n := (\boldsymbol{u}_k^n)_{1\leq n \leq N}$ such that $\boldsymbol{u}_k^n \approx u^n$. It involves a coarse propagator $\mathcal{G}_{\Delta t}$ and a fine propagator $\mathcal{F}_{\delta t}$.

Coarse time grid discretization:



Fine time grid discretization:



The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_k^n) - \mathcal{G}_{\Delta t}(u_k^n)\right]}_{\text{correction}}.$$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_k^n) - \mathcal{G}_{\Delta t}(u_k^n)\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4, k = 2$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_k^n) - \mathcal{G}_{\Delta t}(u_k^n)\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4, k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^0), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^0), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^0)$$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{[\mathcal{F}_{\delta t}(u_k^n) - \mathcal{G}_{\Delta t}(u_k^n)]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4, k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^0), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^0), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^0)$$

2. Compute: $u_{k\geq 1}^{1} = \mathcal{F}_{\delta t}(u^0)$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^{0}.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_{k}^{n}) - \mathcal{G}_{\Delta t}(u_{k}^{n})\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4, k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^{0}), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^{0}), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^{0})$$

2. Compute: $u_{k\geq 1}^{1} = \mathcal{F}_{\delta t}(u^{0})$
3. Compute in parallel:

$$\mathcal{F}_{\delta t}(u_{k=0}^{1}) - \mathcal{G}_{\Delta t}(u_{k=0}^{1}),\ \mathcal{F}_{\delta t}(u_{k=0}^{2}) - \mathcal{G}_{\Delta t}(u_{k=0}^{2}),\ \mathcal{F}_{\delta t}(u_{k=0}^{3}) - \mathcal{G}_{\Delta t}(u_{k=0}^{3})$$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_{k}^{n}) - \mathcal{G}_{\Delta t}(u_{k}^{n})\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4$, $k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^0), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^0), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^0)$$

2. Compute: $u_{k\geq 1}^{1} = \mathcal{F}_{\delta t}(u^0)$
3. Compute in parallel:

$$\mathcal{F}_{\delta t}(u_{k=0}^{1}) - \mathcal{G}_{\Delta t}(u_{k=0}^{1}),\ \mathcal{F}_{\delta t}(u_{k=0}^{2}) - \mathcal{G}_{\Delta t}(u_{k=0}^{2}),\ \mathcal{F}_{\delta t}(u_{k=0}^{3}) - \mathcal{G}_{\Delta t}(u_{k=0}^{3})$$

4. **Update:** $\boxed{u_{k=1}^{2}}$ and $\mathcal{G}_{\Delta t}(u_{k=1}^{2})$, $\boxed{u_{k=1}^{3}}$ and $\mathcal{G}_{\Delta t}(u_{k=1}^{3})$, and $\boxed{u_{k=1}^{4}}$

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^{0}.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_{k}^{n}) - \mathcal{G}_{\Delta t}(u_{k}^{n})\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4$, $k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^{0}), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^{0}), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^{0})$$

2. Compute: $u_{k\geq 1}^{1} = \mathcal{F}_{\delta t}(u^{0})$
3. Compute in parallel:

$$\mathcal{F}_{\delta t}(u_{k=0}^{1}) - \mathcal{G}_{\Delta t}(u_{k=0}^{1}),\ \mathcal{F}_{\delta t}(u_{k=0}^{2}) - \mathcal{G}_{\Delta t}(u_{k=0}^{2}),\ \mathcal{F}_{\delta t}(u_{k=0}^{3}) - \mathcal{G}_{\Delta t}(u_{k=0}^{3})$$

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

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

The parareal algorithm

$$u_{k=0}^{n+1} := \mathcal{G}_{\Delta t}(u_{k=0}^{n}), \qquad \text{where} \quad u_{k=0}^{0} := u^0.$$

$$u_{k+1}^{n+1} := \underbrace{\mathcal{G}_{\Delta t}(u_{k+1}^{n})}_{\text{prediction}} + \underbrace{\left[\mathcal{F}_{\delta t}(u_k^n) - \mathcal{G}_{\Delta t}(u_k^n)\right]}_{\text{correction}}.$$

Illustration of the procedure: $n = 4, k = 2$

1. Compute the coarse approximations:

$$u_{k=0}^{1} = \mathcal{G}_{\Delta t}(u^0), \quad u_{k=0}^{2} = \mathcal{G}_{\Delta t}^{2}(u^0), \quad u_{k=0}^{3} = \mathcal{G}_{\Delta t}^{3}(u^0)$$

2. Compute: $u_{k\geq 1}^{1} = \mathcal{F}_{\delta t}(u^0)$
3. Compute in parallel:

$$\mathcal{F}_{\delta t}(u_{k=0}^{1}) - \mathcal{G}_{\Delta t}(u_{k=0}^{1}),\ \mathcal{F}_{\delta t}(u_{k=0}^{2}) - \mathcal{G}_{\Delta t}(u_{k=0}^{2}),\ \mathcal{F}_{\delta t}(u_{k=0}^{3}) - \mathcal{G}_{\Delta t}(u_{k=0}^{3})$$

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

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

6. **Update:** $\boxed{u_{k=2}^{2}}$ and $\boxed{u_{k=2}^{3}}$ and $\boxed{u_{k=2}^{4}}$

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_{sp} : 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_{sp} : 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

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

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_{sp} : number of elements

The finite element propagator

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

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

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

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

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

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

Then,

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

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

Then,

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

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

Then,

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

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.

Rejection method:

Often used when the PDF f is hard to invert. Assume there exists a PDF $g : \Omega \to \mathbb{R}_+$ “easy” to simulate such that

$$f(x) \leq kg(x), \quad \text{where} \quad k \geq 1 \quad \text{is a constant.}$$

Set $\alpha(x) = \dfrac{f(x)}{kg(x)}$.

Compute the cumulative function $G : \Omega \to [0,1]$ associated to g.

Let $\xi \sim \mathcal{U}([0,1])$.

Find $X_i \in K$ using the direct inversion procedure.

Compute $\alpha(X_i)$.

Let $\xi_1 \sim \mathcal{U}([0,1])$. If $\xi_1 \leq \alpha(X_i)$ accept X_i. Otherwise reject and and come back to first step.

How define the transport of the particles?

Kernel Transport

$(\boldsymbol{x}, t)$: position of the particle $\boldsymbol{x}$ at time t

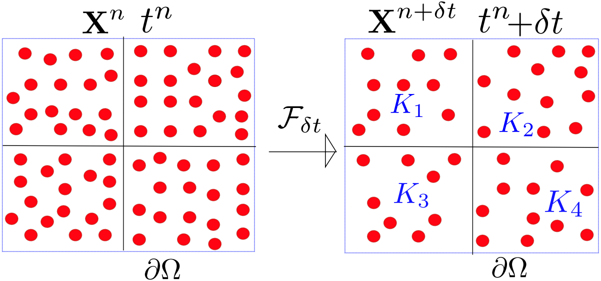
$(\boldsymbol{x}', t')$: position of the particle $\boldsymbol{x}'$ at time t'

Density transition kernel:

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

Pratical formula for the brownian motion:

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



Hybrid parareal algorithm

Coarse propagator : Deterministic solver

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

Consider p independent replicas and M' particles so that the total number of particles is $M = p \times M'$. The numerical solution obtained for a replica $j \in [1, p]$ at parareal iteration k is denoted by $\boldsymbol{U}_{k,j}^{n+1}$

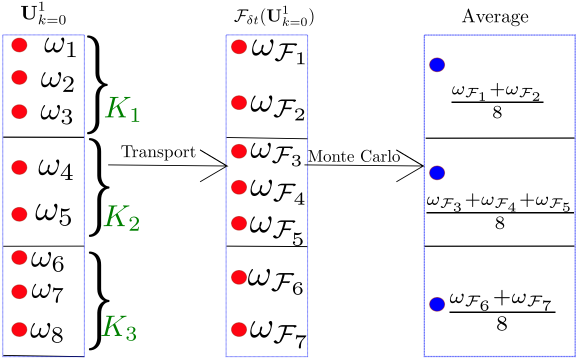
$$\boldsymbol{U}_{k}^{n+1} := \frac{1}{p} \sum_{j=1}^{p} \boldsymbol{U}_{k,j}^{n+1}$$

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

Updating the statistical weights

Example: How avoid sampling $\boldsymbol{U}^2_{k=1}$?

$$[\omega_1^2]_{i\in K} = [\omega_{\mathcal{F}}]_{i\in K} \times \left(\frac{\boldsymbol{U}^2_{k=1}}{\mathcal{F}(\boldsymbol{U}^1_{k=0})}\right)|_K$$



$$\mathrm{hist}(\omega^2_{k=1})|_{K_1} = \frac{1}{8}\left(\omega_{\mathcal{F}_1} + \omega_{\mathcal{F}_2}\right) \times \left(\frac{\boldsymbol{U}^2_{k=1}}{\mathcal{F}(\boldsymbol{U}^1_{k=0})}\right)|_{K_1} = \boldsymbol{U}^2_{k=1}$$

Outline

Numerical experiments

Ω: one-dimensional core with length $L = 5m$, **Final simulation time:** $T = 10s$.

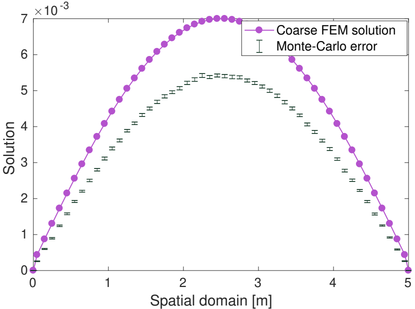
Deterministic propagator: $\mathbb{P}_1$ finite element, $\Delta t = 2s$.

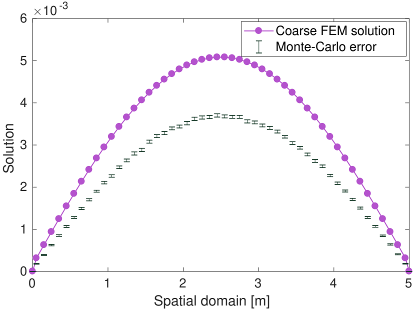
Fine propagator: Monte-Carlo, $\delta t = 2 \times 10^{-4}s$.

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

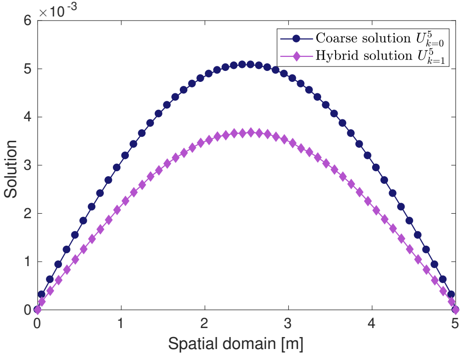
Initial condition: $u_0(x) = \frac{1}{L}$.

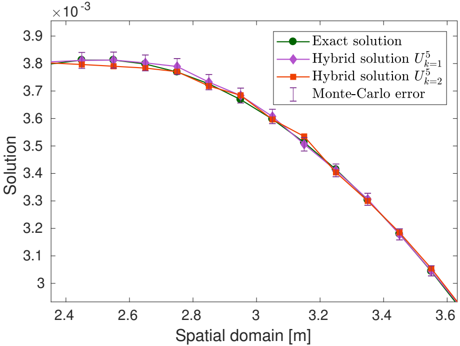
Number of particles: 10^4, **Number of replicas:** 10^3



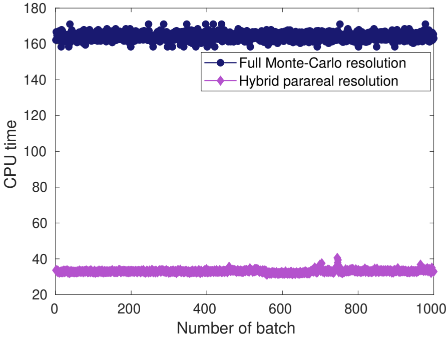


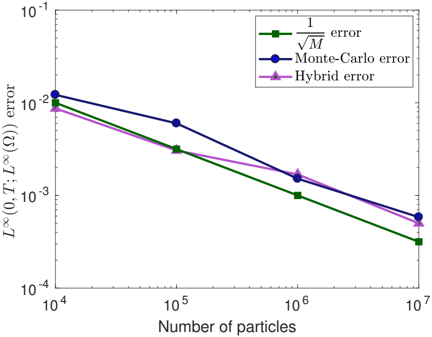
Hybrid solution





CPU time and convergence





Number of particles	Number of replica	Parallelized Monte-Carlo	Hybrid parallelized Monte-Carlo $k = 1$	Hybrid parallelized Monte-Carlo $k = 2$	Gain factor $k = 1$	Gain factor $k = 2$
10^5	10^2	1653.4 s	335.76 s	534.16 s	4.92	3.04
10^4	10^3	164.09 s	33.05 s	7.97 s	4.96	3.09
10^3	10^4	16.86 s	3.39 s	0.83 s	4.97	3.07
10^2	10^5	1.78 s	0.35 s	0.11 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, $\delta t = 2 \times 10^{-4} s$.

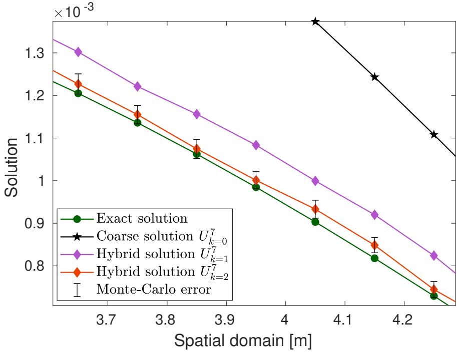
Diffusion coefficient MC: $\mathcal{D} = 0.5 m^2 \cdot s^{-1}$

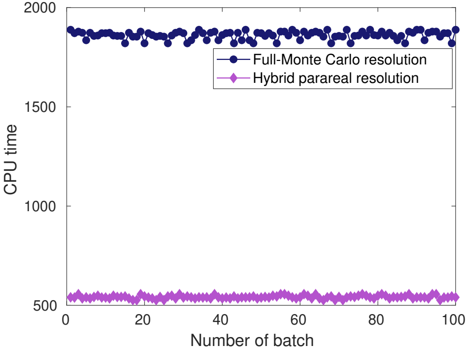
Diffusion coefficient FEM: $\mathcal{D} = 0.48 m^2 \cdot s^{-1}$

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





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 Boltzmann equation in neutronics

J. DABAGHI, Y. MADAY, A. ZOIA, *A hybrid parareal Monte-Carlo algorithm for the parabolic time dependant diffusion equation*. IN PREPARATION

Thank you for your attention!