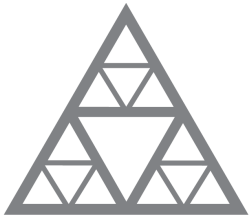
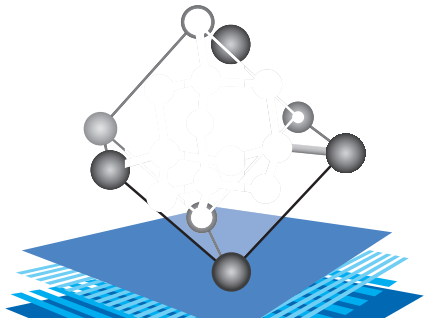




Parareal algorithms for molecular dynamics simulations

Frédéric Legoll

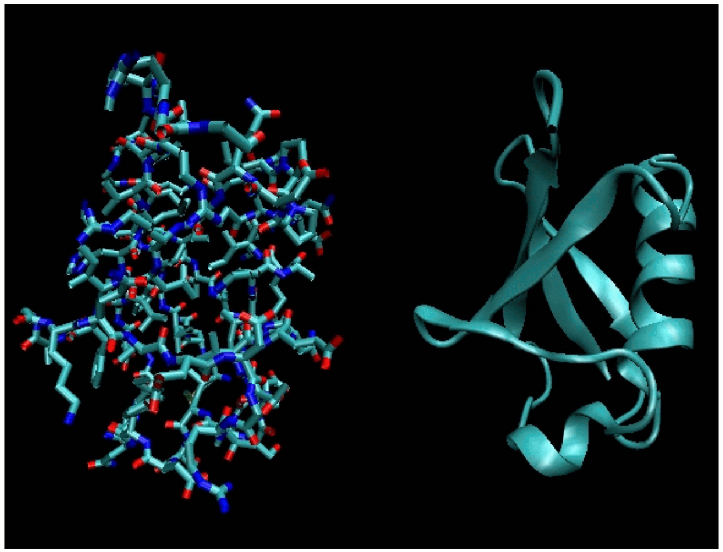
Ecole des Ponts & Inria Paris

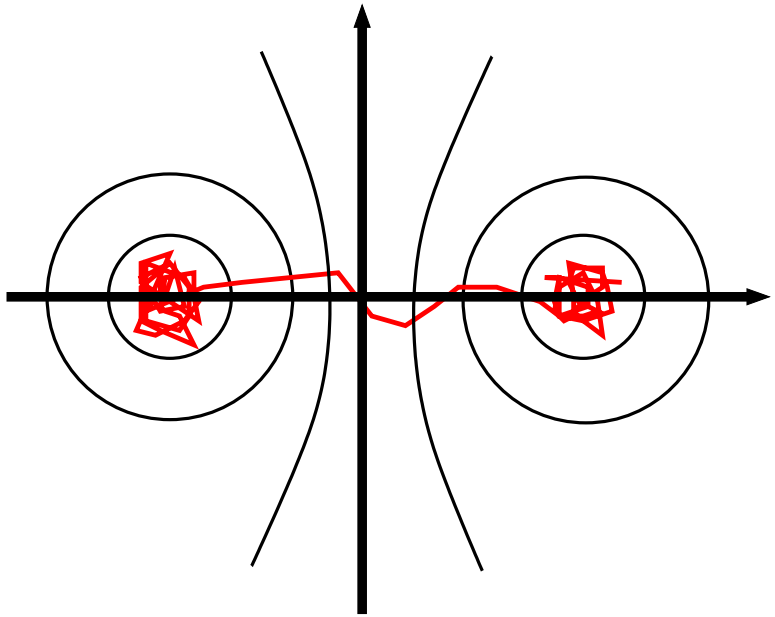
Joint with T. Lelièvre, D. Perez, U. Sharma and O. Gorynina

ANR QuAMProcs workshop, Inria Paris, 8-9 March 2022

Motivation

This work is motivated by molecular simulation, where we often have to simulate long trajectories of complex systems.





Typical dynamics: the Langevin equation

$$dq_t = p_t\,dt, \qquad dp_t = -\nabla V(q_t)\,dt - \gamma\,p_t\,dt + \sqrt{2\gamma\beta^{-1}}\,dW_t$$

Outline

- Since we have to simulate long-time trajectories, it seems attractive to use the parareal algorithm, which solves initial value problems by parallel-in-time computations (domain-decomposition fashion)

Outline

- Since we have to simulate long-time trajectories, it seems attractive to use the parareal algorithm, which solves initial value problems by parallel-in-time computations (domain-decomposition fashion)
- It turns out that this algorithm is not stable for MD problems when the time horizon is too large

Outline

- Since we have to simulate long-time trajectories, it seems attractive to use the parareal algorithm, which solves initial value problems by parallel-in-time computations (domain-decomposition fashion)
- It turns out that this algorithm is not stable for MD problems when the time horizon is too large
- We therefore introduce an adaptive parareal algorithm, which performs simulations on shorter time slabs and paste them together, thereby allowing for a significant CPU gain
- Application to the simulation in LAMMPS of self-interstitial atoms diffusing in a tungsten lattice

F.L., T. Lelièvre and U. Sharma, *An adaptive parareal algorithm: application to the simulation of molecular dynamics trajectories*, SIAM Journal on Scientific Computing 2022

O. Gorynina, F.L., T. Lelièvre and D. Perez, *Long-time simulation of diffusing self-interstitial atoms using the parareal algorithm*, in preparation

Parallel in time algorithm for ODEs

Parallel in time algorithm for ODEs

$$\frac{dx}{dt} = f(x), \qquad x \in \mathbb{R}^d$$

The parareal algorithm (Lions, Maday and Turinici, 2001) is based upon two integrators to propagate the system over a time ΔT:

- a fine, accurate integrator $\mathcal{F}_{\Delta T}$
- a cheap coarse integrator $\mathcal{C}_{\Delta T}$

For instance,

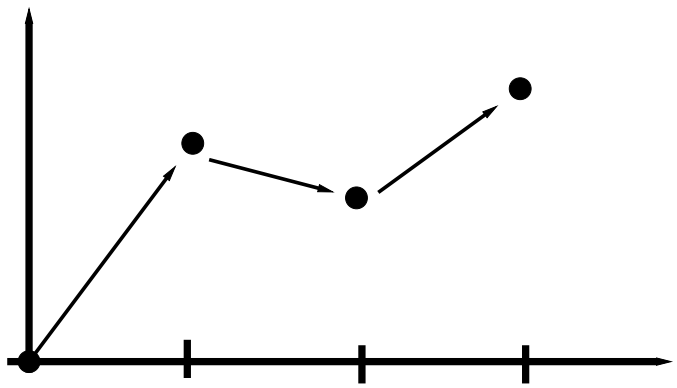
$$\mathcal{F}_{\Delta T} = (\Phi_{\delta t_F})^{\Delta T/\delta t_F} \quad \text{and} \quad \mathcal{C}_{\Delta T} = (\Phi_{\delta t_C})^{\Delta T/\delta t_C} \quad \text{with} \quad \delta t_F \ll \delta t_C$$

where $\Phi_{\delta t}$ is a one time step propagator

The parareal iterative procedure

- Initialization: coarse propagation that yields $\left\{x_n^{k=0}\right\}_n$:

$$\forall n, \quad x_{n+1}^{k=0} = \mathcal{C}_{\Delta T}(x_n^{k=0})$$

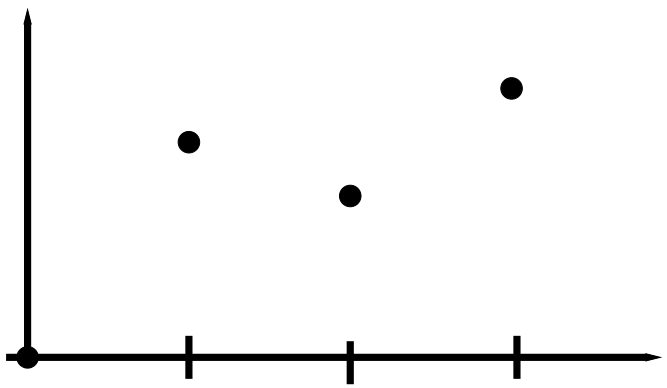


The parareal iterative procedure

- Initialization: coarse propagation that yields $\left\{x_n^{k=0}\right\}_n$:

$$\forall n, \quad x_{n+1}^{k=0} = \mathcal{C}_{\Delta T}(x_n^{k=0})$$

- Iterate over $k \geq 0$:



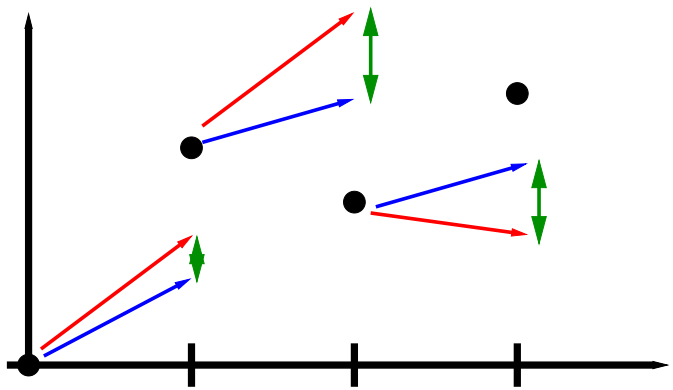
The parareal iterative procedure

- Initialization: coarse propagation that yields $\left\{x_n^{k=0}\right\}_n$:

$$\forall n, \quad x_{n+1}^{k=0} = \mathcal{C}_{\Delta T}(x_n^{k=0})$$

- Iterate over $k \geq 0$:
  - compute jumps (in parallel):

$$J_n^k = \mathcal{F}_{\Delta T}(x_n^k) - \mathcal{C}_{\Delta T}(x_n^k)$$



The parareal iterative procedure

- Initialization: coarse propagation that yields $\left\{x_n^{k=0}\right\}_n$:

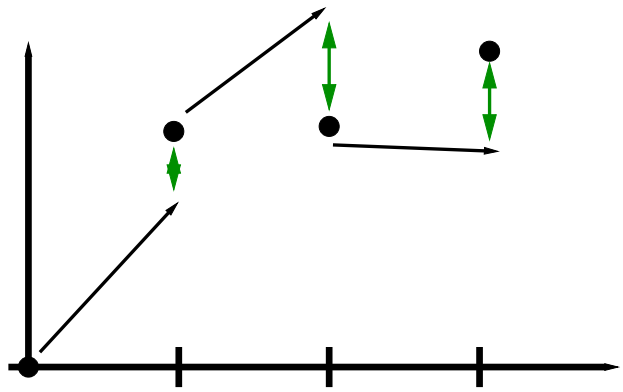
$$\forall n, \quad x_{n+1}^{k=0} = \mathcal{C}_{\Delta T}(x_n^{k=0})$$

- Iterate over $k \geq 0$:
  - compute jumps (in parallel):

$$J_n^k = \mathcal{F}_{\Delta T}(x_n^k) - \mathcal{C}_{\Delta T}(x_n^k)$$

  - sequential update to obtain $\left\{x_n^{k+1}\right\}_n$:

$$\forall n, \quad x_{n+1}^{k+1} = \mathcal{C}_{\Delta T}(x_n^{k+1}) + J_n^k$$



The parareal iterative procedure

- Initialization: coarse propagation that yields $\left\{x_n^{k=0}\right\}_n$:

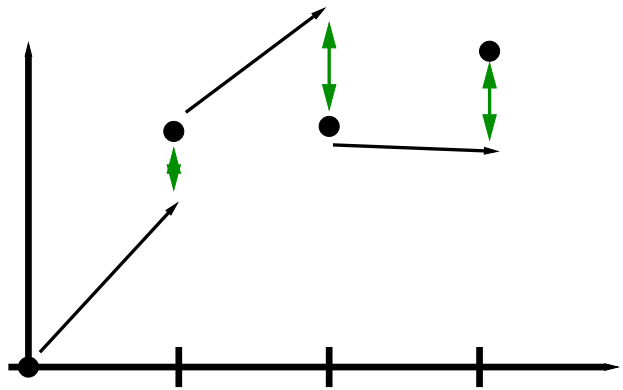
$$\forall n, \quad x_{n+1}^{k=0} = \mathcal{C}_{\Delta T}(x_n^{k=0})$$

- Iterate over $k \geq 0$:
  - compute jumps (in parallel):

$$J_n^k = \mathcal{F}_{\Delta T}(x_n^k) - \mathcal{C}_{\Delta T}(x_n^k)$$

  - sequential update to obtain $\left\{x_n^{k+1}\right\}_n$:

$$\forall n, \quad x_{n+1}^{k+1} = \mathcal{C}_{\Delta T}(x_n^{k+1}) + J_n^k$$



The fine solver is called only in the parallel part of the algorithm.

Parareal algorithm for MD simulations – 1

- The parareal iterations converge (when $k \to \infty$) to the solution of the reference dynamics

$$x_{n+1} = \mathcal{F}_{\Delta T}(x_n)$$

  This comes from the fact that $x_n^k = \mathcal{F}_{\Delta T}^n(x_0)$ whenever $k \geq n$.

- In practice, for many applications, convergence is observed in much fewer iterations

- In MD, we often run simulations with time steps chosen just below the stability limit (this often provides sufficient accuracy on the quantities of interest).

- There is hence no room for choosing $\delta t_C \gg \delta t_F$

Parareal algorithm for MD simulations – 2

- We thus turn to a different paradigm where $\mathcal{C}_{\Delta T}$ integrates a simpler dynamics than $\mathcal{F}_{\Delta T}$ (say with the same time step)
- In our setting, $\mathcal{F}_{\Delta T}$ integrates the original Langevin dynamics (with the reference potential $V_f \equiv V$) whereas $\mathcal{C}_{\Delta T}$ integrates a Langevin dynamics run on a simplified (cheaper to compute) potential V_c.
- The Gaussian increments (in the numerical scheme used to integrate the Langevin equation) are the same for $\mathcal{F}_{\Delta T}$ and $\mathcal{C}_{\Delta T}$ and over all parareal iterations (to ensure as best as possible trajectorial convergence)

Similar paradigm (in terms of V_f vs V_c) in [Baffico et al, PRE 2002]

Simulations on toy model problems

Two model problems

- A quadratic model in 1D:

$$V_f(q) = \frac{q^2}{2}, \qquad V_c(q) = \omega \frac{q^2}{2} \quad \text{for some } \omega > 0$$

Simple enough to be amenable to theoretical analysis, and exhibits the same issues as those appearing with more complex models.

Two model problems

- A quadratic model in 1D:

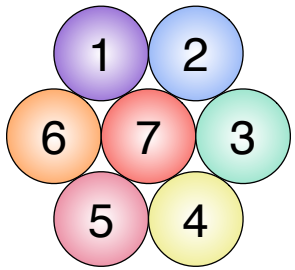
$$V_f(q) = \frac{q^2}{2}, \qquad V_c(q) = \omega \frac{q^2}{2} \quad \text{for some } \omega > 0$$

Simple enough to be amenable to theoretical analysis, and exhibits the same issues as those appearing with more complex models.

- A slightly less simple model: a 7-atom Lennard-Jones cluster in 2D:

$$V_f(q) = \frac{1}{2} \sum_{i,j \in \{1,\dots,7\},\ i \neq j} \phi_f(|q^i - q^j|), \qquad \phi_f(r) = r^{-12} - 2\, r^{-6}$$

$V_c \equiv$ harm. approx. of V_f at the global minimum (the initial condition is chosen in the corresponding well).



Convergence criteria

- Relative error between consecutive parareal trajectories:

$$E(k,N) = \frac{\sum_{n=1}^{N} |q_n^k - q_n^{k-1}|}{\sum_{n=1}^{N} |q_n^{k-1}|}.$$

- By construction, we have $q_n^k = q(n\Delta T)$ for any $k \geq n$, and thus $E(k,N) = 0$ for $k \geq N+1$
- We stop the algorithm at the first parareal iteration $\overline{k}$ for which

$$E(\overline{k},N) < \delta_{\text{conv}} = 10^{-5}$$

Convergence criteria

- Relative error between consecutive parareal trajectories:

$$E(k,N) = \frac{\sum_{n=1}^{N} |q_n^k - q_n^{k-1}|}{\sum_{n=1}^{N} |q_n^{k-1}|}.$$

- By construction, we have $q_n^k = q(n\Delta T)$ for any $k \geq n$, and thus $E(k,N) = 0$ for $k \geq N+1$
- We stop the algorithm at the first parareal iteration $\overline{k}$ for which

$$E(\overline{k},N) < \delta_{\text{conv}} = 10^{-5}$$

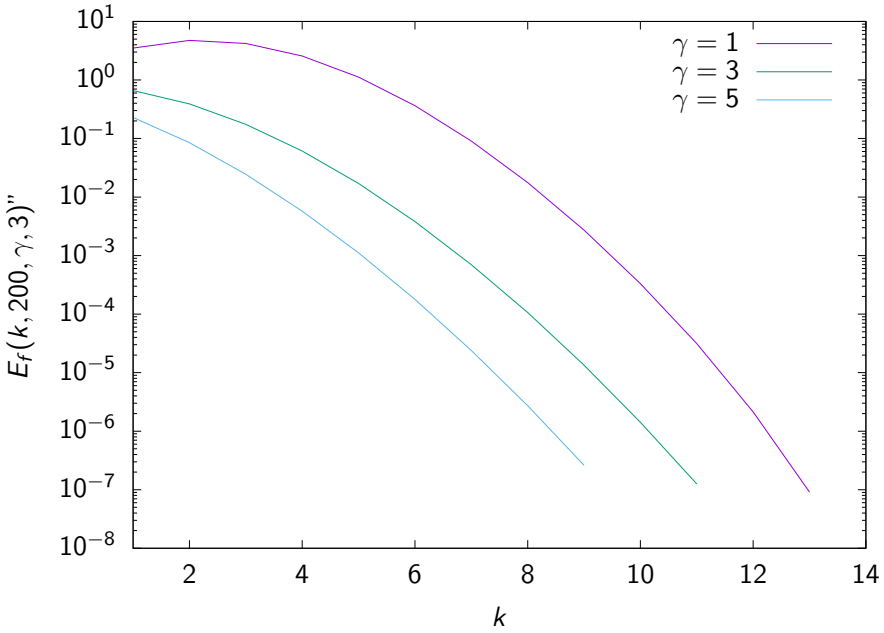
- The gain is then

$$\text{gain} = \frac{N}{\overline{k}} = \frac{\#\text{ fine propagations for a sequential algorithm}}{\#\text{ fine propagations for the parareal algorithm}}$$

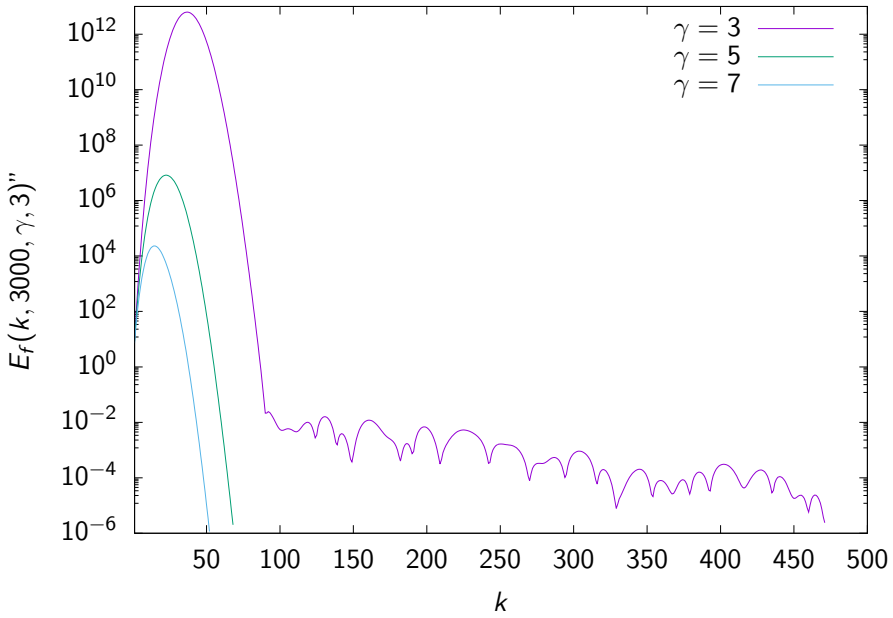
$E_f(k,N)$ = relative error with respect to the reference trajectory

Instability at large times

Plot of $E_f(k, N)$ as a function of k:



$N = 200$

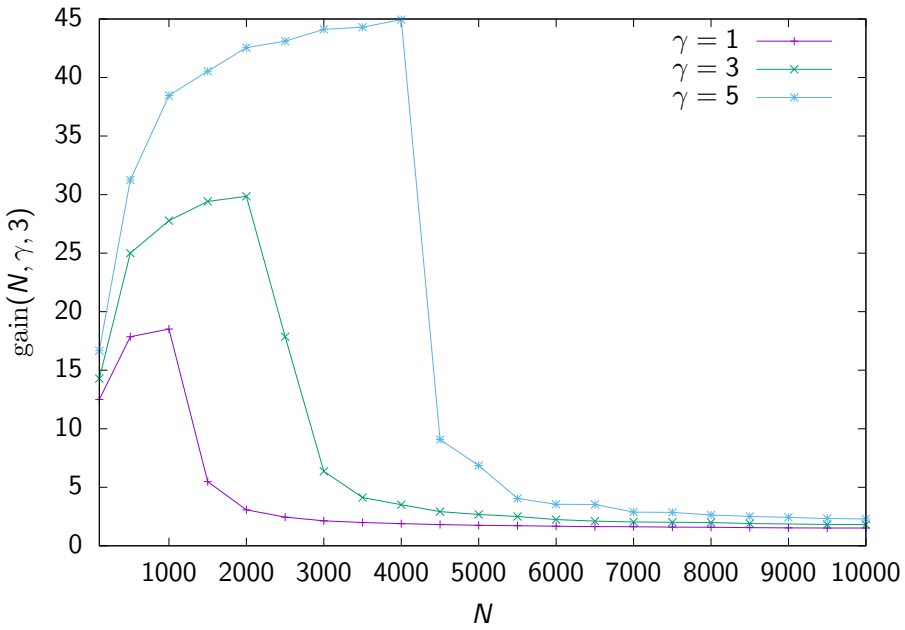


$N = 3000$

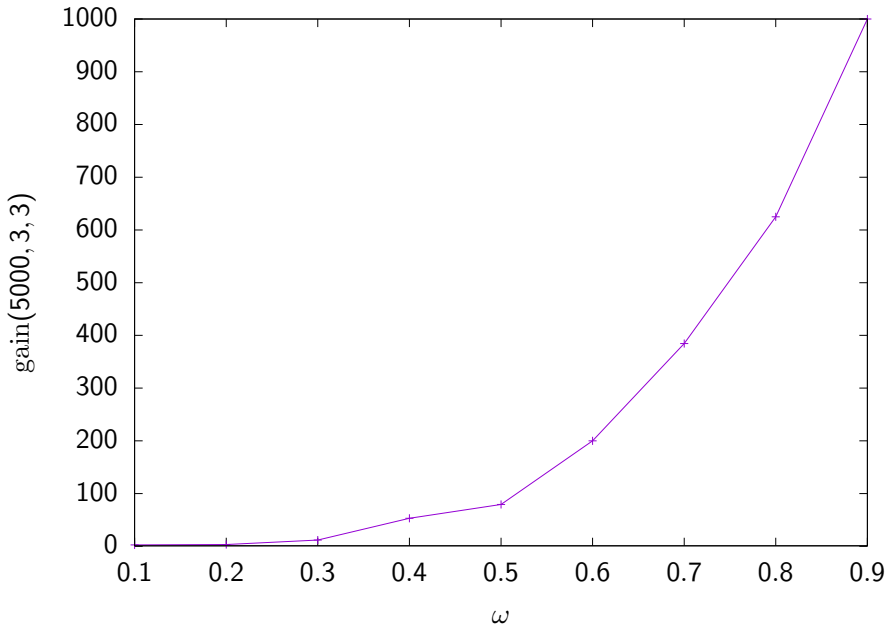
- For small N, fast convergence of the parareal iterations (gain ≈ 10)
- For large N, the error increases to large values when k increases (because the trajectory goes far away), before eventually converging

Harmonic model, $\omega = 0.1$, $\Delta T = 0.05$, $\beta = 3$

Gain as a function of N and ω



Gain as a function of N ($\omega = 0.1$)



Gain as a function of ω ($\gamma = 3$, $N = 5000$)

- Fixed coarse model: the gain decreases to 1 when N increases
- Fixed N: the gain decreases when the coarse model differs too much from the fine one

Harmonic model, $\Delta T = 0.05$, $\beta = 3$

Theoretical analysis on a toy problem – 1

- Reference problem: $V_f(x) = x^2/2$ and

$$\frac{dx}{dt} = -V_f'(x) = -x$$

Fine integrator (exact integrator over ΔT):

$$\mathcal{F}_{\Delta T}(x) = \exp(-\Delta T)\, x$$

- Coarse model: $V_c(x) = \omega\, x^2/2$ and

$$\frac{dx}{dt} = -V_c'(x) = -\omega\, x$$

Coarse integrator (exact integrator over ΔT):

$$\mathcal{C}_{\Delta T}(x) = \exp(-\omega \Delta T)\, x$$

Overdamped dynamics without noise

Theoretical analysis on a toy problem – 2

We study the relative error at the terminal point

$$R_{n,k} = \frac{x(n\Delta T) - x_n^k}{x(n\Delta T)} = \frac{x_n^n - x_n^k}{x_n^n}$$

as a function of k, for various choices

- of the trajectory length n
- and of

$$y = \frac{\mathcal{F}_{\Delta T}}{\mathcal{C}_{\Delta T}} - 1 \in (-1, \infty)$$

which quantifies how much the coarse and the fine model differ ($\omega = 1$ iff $y = 0$).

Formal result ($y > 0$)

In the case $y > 0$, then

$$0 < R_{n,k} \leq 1 \qquad \text{and} \qquad k \mapsto R_{n,k} \text{ is decreasing}$$

- If y is such that $y \leq c/n$ (the coarse model is very close to the fine; the longer the trajectory, the closer the models should be), then

$$k \mapsto R_{n,k} \quad \text{is convex}$$

Formal result ($y > 0$)

In the case $y > 0$, then

$$0 < R_{n,k} \leq 1 \qquad \text{and} \qquad k \mapsto R_{n,k} \text{ is decreasing}$$

- If y is such that $y \leq c/n$ (the coarse model is very close to the fine; the longer the trajectory, the closer the models should be), then

$$k \mapsto R_{n,k} \quad \text{is convex}$$

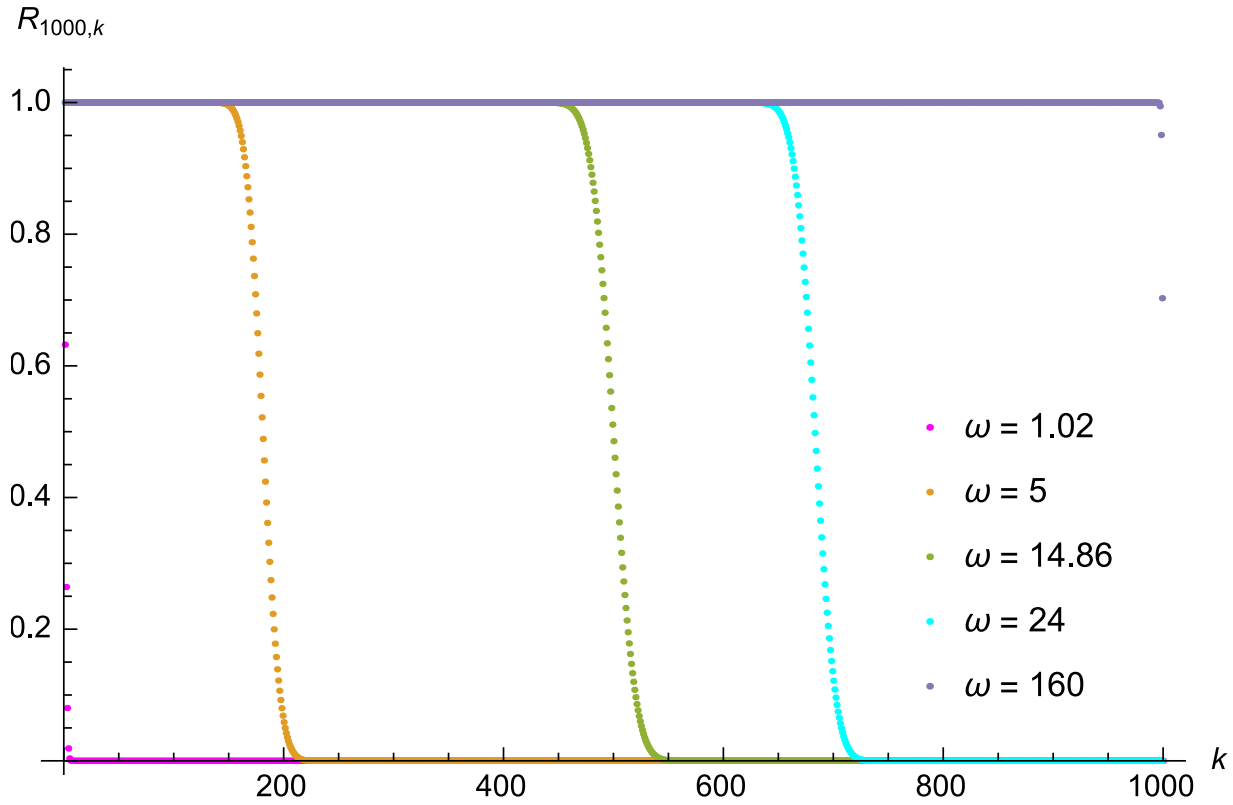
- If y is such that $y \geq c\,n$ (the coarse model is very different from the fine), then

$$k \mapsto R_{n,k} \quad \text{is concave}$$

- If y is in-between, then there exists $\overline{p} \approx \dfrac{n|y|}{1+|y|} \in [1, n]$ such that

$$k \mapsto R_{n,k} \text{ is concave for } 1 \leq k \leq \overline{p} \text{ and convex for } \overline{p} \leq k \leq n$$

Numerical illustration ($y > 0$)



Plot of $k \mapsto R_{n,k}$ for $n = 1000$

The error (as a function of k) is

- convex for $\omega = 1.02$ (excellent convergence)
- concave for $\omega = 160$ (error close to 100% for almost all k . . .)
- concave then convex for in-between ω (infl. point depends on y & n)

Formal result ($y \in (-1, 0)$)

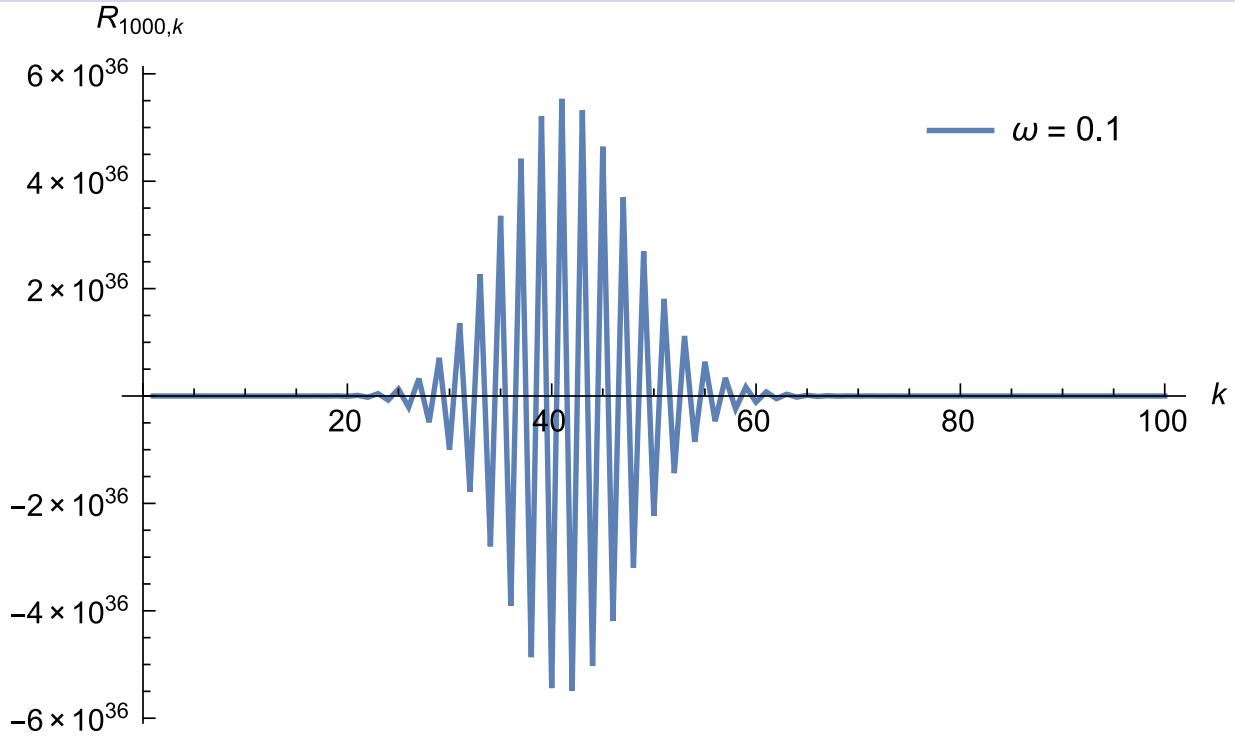
- If y is such that $|y| \leq c/n$, then the situation is fine: the sequence $k \mapsto |R_{n,k}|$ is decreasing

- Otherwise, the situation is bad. Along the parareal iterations, the error gets very large:

$$\max_{k \in \{0,\dots,n\}} |R_{n,k}| \geq \frac{C_y}{\sqrt{n}} (D_y)^n$$

  for some $D_y > 1$ only depending on y (which, we recall, quantifies how much the two models differ):
  - at fixed y, the error gets exponentially large as a function of n
  - the rate D_y increases when $|y| \in (0, 1)$ increases

Numerical illustration ($y \in (-1, 0)$)



Plot of $k \mapsto R_{n,k}$ for $n = 1000$

For $\omega = 0.1$ and $n = 1000$, we get $y < 0$ and $\overline{p} \approx 42$. We indeed observe that $|R_{n,k}|$ is maximal (and very large!) for $k = \overline{p} \approx \dfrac{n|y|}{1 + |y|}$.

These results provide a complete understanding of the error for this oversimplified model, which is illustrative of the general situation

Adaptive algorithm

Adaptive algorithm

Relative error $E \equiv$ relative error between consecutive trajectories

- On the time-slab $[0, N\Delta T]$, we run the parareal algorithm until E is
  - either smaller than the convergence threshold δ_{conv}
  - or larger than an explosion threshold δ_{expl} (attained at parareal iteration $\# \, k_{\mathrm{cur}}$)

Adaptive algorithm

Relative error $E \equiv$ relative error between consecutive trajectories

- On the time-slab $[0, N\Delta T]$, we run the parareal algorithm until E is
  - either smaller than the convergence threshold δ_{conv}
  - or larger than an explosion threshold δ_{expl} (attained at parareal iteration $\# \, k_{\mathrm{cur}}$)

- In the blow-up case, for the parareal iteration k_{cur}, we find the first time iteration $1 + \widetilde{m}_1 \leq N$ for which E exceeds δ_{expl}, and we shorten the slab to $[0, \widetilde{m}_1 \Delta T]$.

- We then proceed with the parareal iterations on the slab $[0, \widetilde{m}_1 \Delta T]$, that we possibly further shorten, until the relative error (on $[0, \widetilde{m}_1 \Delta T]$) is smaller than δ_{conv}.

Adaptive algorithm

Relative error $E \equiv$ relative error between consecutive trajectories

- On the time-slab $[0, N\Delta T]$, we run the parareal algorithm until E is
  - either smaller than the convergence threshold δ_{conv}
  - or larger than an explosion threshold δ_{expl} (attained at parareal iteration $\#\ k_{\mathrm{cur}}$)

- In the blow-up case, for the parareal iteration k_{cur}, we find the first time iteration $1 + \widetilde{m}_1 \leq N$ for which E exceeds δ_{expl}, and we shorten the slab to $[0, \widetilde{m}_1 \Delta T]$.

- We then proceed with the parareal iterations on the slab $[0, \widetilde{m}_1 \Delta T]$, that we possibly further shorten, until the relative error (on $[0, \widetilde{m}_1 \Delta T]$) is smaller than δ_{conv}.

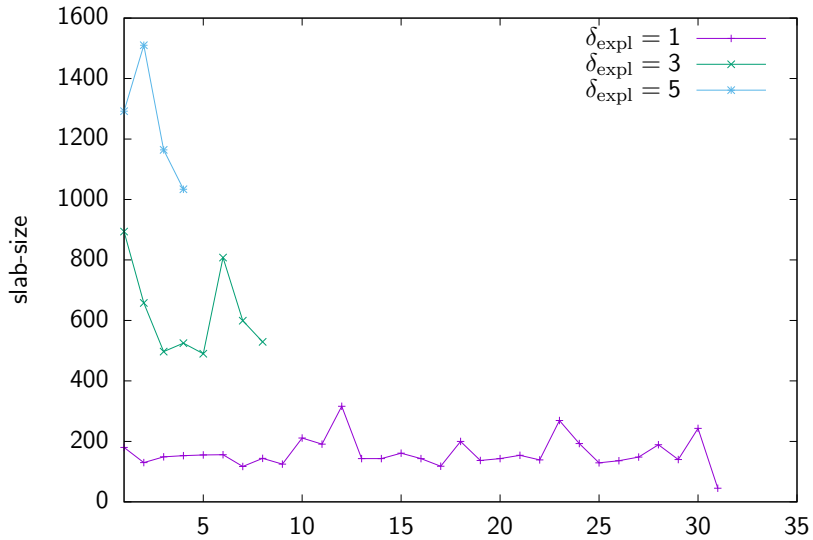
- Once we have converged on $[0, \widetilde{m}_1 \Delta T]$, we proceed with the next part of the time range and define the new (tentative) time-slab as $[\widetilde{m}_1 \Delta T, N\Delta T]$.

Explosion threshold

The slab sizes are such that $E \leq \delta_{\text{expl}}$:

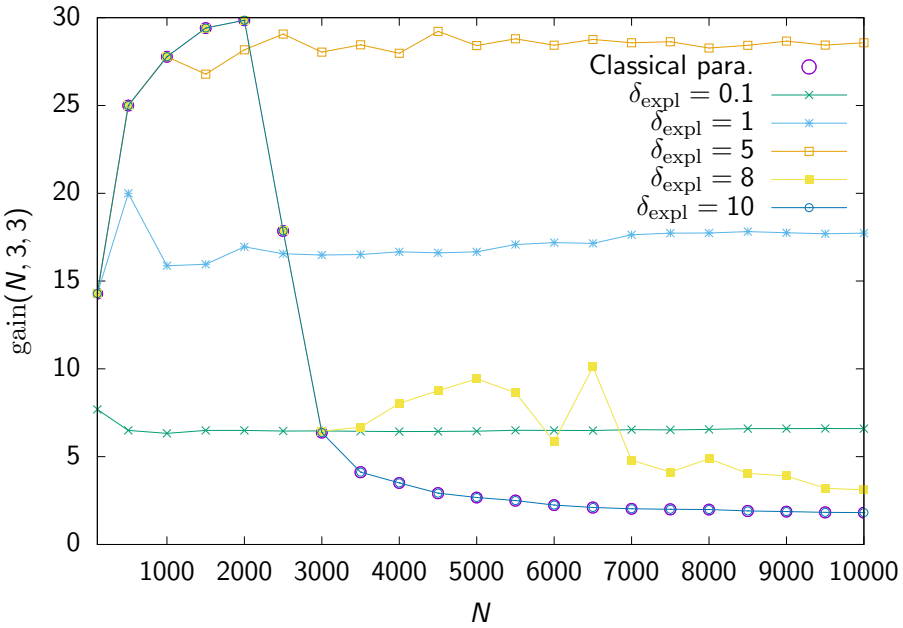
- if δ_{expl} is chosen large, the adaptive criterion is never triggered: vanilla parareal
- if δ_{expl} is chosen small, the slabs are short: no parallelism anymore
- the optimal choice of δ_{expl} is somewhere in-between

List of the sizes of the time-slabs found by the algorithm:

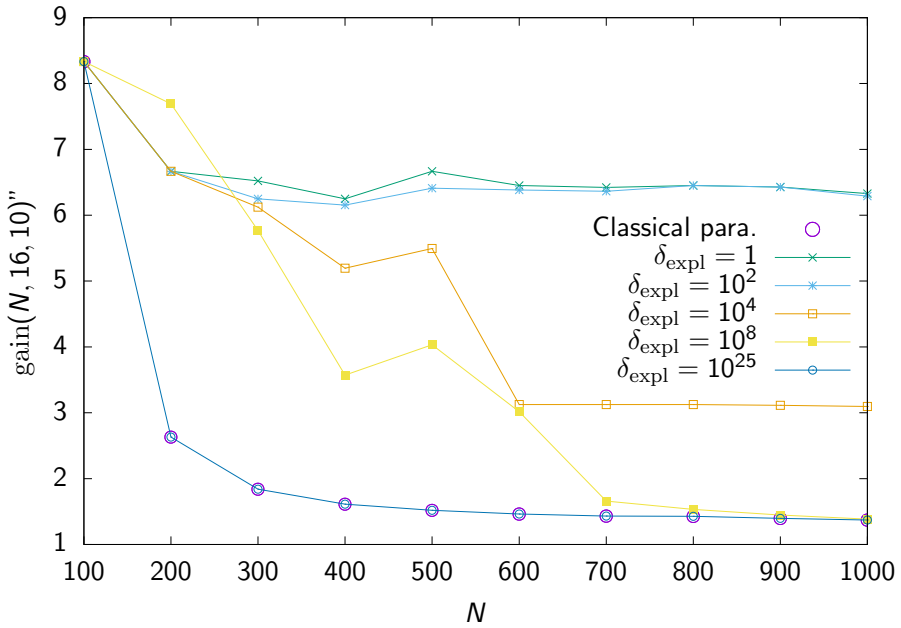


Harmonic model, $\omega = 0.1$, $\Delta T = 0.05$, $\beta = \gamma = 3$, $N = 5000$

Gain



Har-1d ($\omega = 0.1$)



LJ7-2d

- For moderate values of δ_{expl}, the gain seems independent of N
- For large N, the adaptive algorithm always outperforms the classical version (gain ≈ 30 for Har-1d, gain ≈ 7 for LJ7-2d)

Conclusions on this part

- The trajectories provided by the classical parareal algorithm are far away from the reference trajectories: not only trajectorial accuracy is poor, but statistical accuracy is poor as well
- The adaptive algorithm always outperforms the classical version

F.L., T. Lelièvre and U. Sharma, SIAM J. Scientific Computing 2022

Support of ANR (through project CINE-PARA) and of EuroHPC (through project TIME-X) are gratefully acknowledged

Applications to the simulation of self-interstitial atoms in tungsten

Joint work with O. Gorynina, T. Lelièvre and D. Perez

Our aim

- We consider a BCC lattice of 128 tungsten atoms (with periodic boundary conditions)
- We add a self-interstitial atom (SIA)
- At the temperature of 2000 K and for $\gamma^{-1} = 1$ ps:
  - the system is metastable
  - the activation energy is sufficiently small to observe several jumps of the SIA within affordable trajectories
- Quantities of interest:
  - list of the residence times $\equiv$ time spent by the SIA in a given well before jumping into another well (monitor "pathwise accuracy")
  - distribution of the residence times (monitor "statistical accuracy")

Time step: $\delta t = 2$ fs

Fine and coarse potentials

- fine (reference) potential V_f: SNAP potential [Thompson et al, J. Comput. Phys. 2015]:
  - an empirical potential with several hundreds of parameters
  - adjusted using Machine Learning techniques on DFT results (lattice constants, elastic constants, ...)
  - very accurate and very expensive!
- coarse potential V_c: EAM potential [Daw and Baskes, PRB 1984]

Computational time required to perform 5000 steps of Langevin scheme:

Potential	time (in sec. on a standard laptop)
V_c (EAM)	0.6923
V_f (SNAP)	1788

Although V_c is much cheaper than V_f, we will compute the actual gain, without assuming that Cost V_c is negligible vs Cost V_f.

Implementation in LAMMPS

$$x_{n+1}^{k+1} = \mathcal{C}_{\Delta T}(x_n^{k+1}) + \mathcal{F}_{\Delta T}(x_n^k) - \mathcal{C}_{\Delta T}(x_n^k)$$

We implemented the parareal algorithm in a non-intrusive manner in LAMMPS:

- master code: in Python
- for given $x_n^k = (q_n^k, p_n^k)$,
  - Python requests LAMMPS to advance the system (using either V_f or V_c) over a time ΔT (by possibly using several time steps δt), thereby computing $\mathcal{F}_{\Delta T}(x_n^k)$ and $\mathcal{C}_{\Delta T}(x_n^k)$
  - the jumps $\mathcal{F}_{\Delta T}(x_n^k) - \mathcal{C}_{\Delta T}(x_n^k)$ are computed by Python
- in the sequential part, Python first requests LAMMPS to compute $\mathcal{C}_{\Delta T}(x_n^{k+1})$ and second adds the jump to obtain x_{n+1}^{k+1}
- the parareal procedure is implemented in Python and LAMMPS is used as a legacy code

Time-stepping scheme of LAMMPS – 1

BBK scheme [Brunger, Brooks and Karplus, Chem. Phys. Lett. 1984]:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_{\ell-1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_{\ell+1}$$

- The same Gaussian increment is used at the third line to compute $p_{\ell+1}$ and at the first line of the next iteration to compute $p_{(\ell+1)+1/2}$

Time-stepping scheme of LAMMPS – 1

BBK scheme [Brunger, Brooks and Karplus, Chem. Phys. Lett. 1984]:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_{\ell-1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_{\ell+1}$$

- The same Gaussian increment is used at the third line to compute $p_{\ell+1}$ and at the first line of the next iteration to compute $p_{(\ell+1)+1/2}$
- Very easy to implement on the basis of a Verlet scheme:

$$p_{\ell+1/2} = p_\ell + \frac{\delta t}{2} F(q_\ell)$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

Compute the new force $F(q_{\ell+1})$

$$p_{\ell+1} = p_{\ell+1/2} + \frac{\delta t}{2} F(q_{\ell+1})$$

Here, the force F depends on q and p and is the sum of a potential force, a friction force and a random force

Time-stepping scheme of LAMMPS – 2

BBK scheme [Brunger, Brooks and Karplus, Chem. Phys. Lett. 1984]:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_{\ell-1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_{\ell+1}$$

For $\ell = 0$, the scheme needs to be adjusted since $p_{\ell-1/2}$ is not defined.

Time-stepping scheme of LAMMPS – 2

BBK scheme [Brunger, Brooks and Karplus, Chem. Phys. Lett. 1984]:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_{\ell-1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_{\ell+1}$$

  For $\ell = 0$, the scheme needs to be adjusted since $p_{\ell-1/2}$ is not defined.

- Scheme for $\ell = 0$:

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_\ell + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma\beta^{-1}\delta t} \, G_{\ell+1}$$

- In addition, to go from (q_0, p_0) to (q_1, p_1), two Gaussian increments G_0 and G_1 are used (in contrast to the next steps).

These specific features have no consequence only if many steps of the BBK algorithm are performed ...

Time-stepping scheme of LAMMPS – 3

Consider our parareal implementation, and assume $V \equiv 0$. To reach the final time $T = N\,\Delta T$ using a scheme with the time step $\delta t = \Delta T / L$, we do:

- For $n = 1$ to N do
  - enter LAMMPS
  - perform L time steps of the BBK scheme of length δt
  - exit LAMMPS

The equilibrium distribution of the $\{p_{n\,L}\}_{n \geq 0}$ (momenta when exiting LAMMPS) can be computed: it is a Gaussian distribution with variance

$$\sigma_L = \beta^{-1} \left(1 - \frac{1}{2L} \right) \quad \text{rather than } \beta^{-1}$$

Time-stepping scheme of LAMMPS – 3

Consider our parareal implementation, and assume $V \equiv 0$. To reach the final time $T = N\,\Delta T$ using a scheme with the time step $\delta t = \Delta T / L$, we do:

- For $n = 1$ to N do
  - enter LAMMPS
  - perform L time steps of the BBK scheme of length δt
  - exit LAMMPS

The equilibrium distribution of the $\{p_{n\,L}\}_{n \geq 0}$ (momenta when exiting LAMMPS) can be computed: it is a Gaussian distribution with variance

$$\sigma_L = \beta^{-1} \left(1 - \frac{1}{2L} \right) \quad \text{rather than } \beta^{-1}$$

L	β^{-1}	Empirical variance of $\{p_{n\,L}\}_{0 \leq n \leq N}$	N	σ_L
1	300	157	20000	150
1	600	303	20000	300
10	300	280	20000	285
100	300	303.46	20000	298

OK if $L \gg 1$, but nok if L is small!

Time-stepping scheme of LAMMPS – 4

Suggestion: use a time-dependent temperature:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma \, p_{\ell-1/2} + \frac{1}{2} \sqrt{2\gamma \beta_\ell^{-1} \delta t} \, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2} \, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma \, p_{\ell+1/2} + \frac{1}{2} \sqrt{2\gamma \beta_\ell^{-1} \delta t} \, G_{\ell+1}$$

For $\ell = 0$, we again replace $p_{\ell-1/2}$ by p_ℓ

Time-stepping scheme of LAMMPS – 4

Suggestion: use a time-dependent temperature:

- General expression (for $\ell \geq 1$):

$$p_{\ell+1/2} = p_\ell - \frac{\delta t}{2} \nabla V(q_\ell) - \frac{\delta t}{2} \gamma\, p_{\ell-1/2} + \frac{1}{2}\sqrt{2\gamma\beta_\ell^{-1}\delta t}\, G_\ell$$

$$q_{\ell+1} = q_\ell + p_{\ell+1/2}\, \delta t$$

$$p_{\ell+1} = p_{\ell+1/2} - \frac{\delta t}{2} \nabla V(q_{\ell+1}) - \frac{\delta t}{2} \gamma\, p_{\ell+1/2} + \frac{1}{2}\sqrt{2\gamma\beta_\ell^{-1}\delta t}\, G_{\ell+1}$$

  For $\ell = 0$, we again replace $p_{\ell-1/2}$ by p_ℓ

- We write $\beta_\ell^{-1} = C_\ell\, \beta^{-1}$ and identify C_ℓ such that

$$\text{Var } p_0 = \beta^{-1} \Longrightarrow \text{Var } p_\ell = \beta^{-1} \text{ for any } \ell \geq 1$$

  We find $C_0 = 2$ and $\sqrt{C_\ell} = \left(\sqrt{C_{\ell-1} + 8} - \sqrt{C_{\ell-1}}\right)/2$

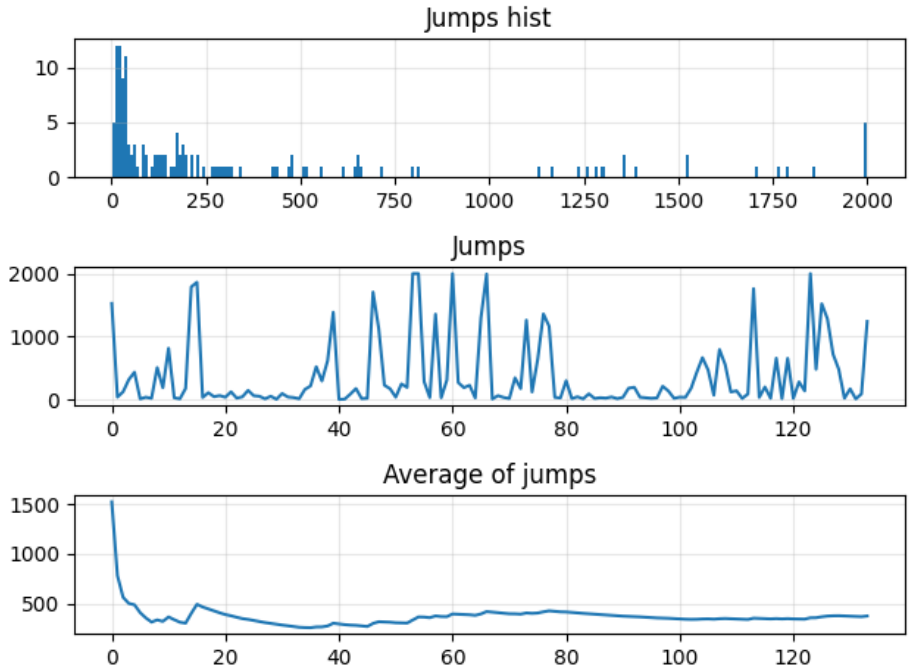
L	β^{-1}	Empirical variance of $\{p_{n\,L}\}_{0 \leq n \leq N}$	N
1	300	303	20000
10	300	294	20000
10	300	297	200000

Reference results

History of residence times on a single trajectory:

$$[122, 23, 27, 476, 14, 32, 560, 245] \quad \times \delta t$$

Distribution of residence times (25 trajectories of 2000 time steps):



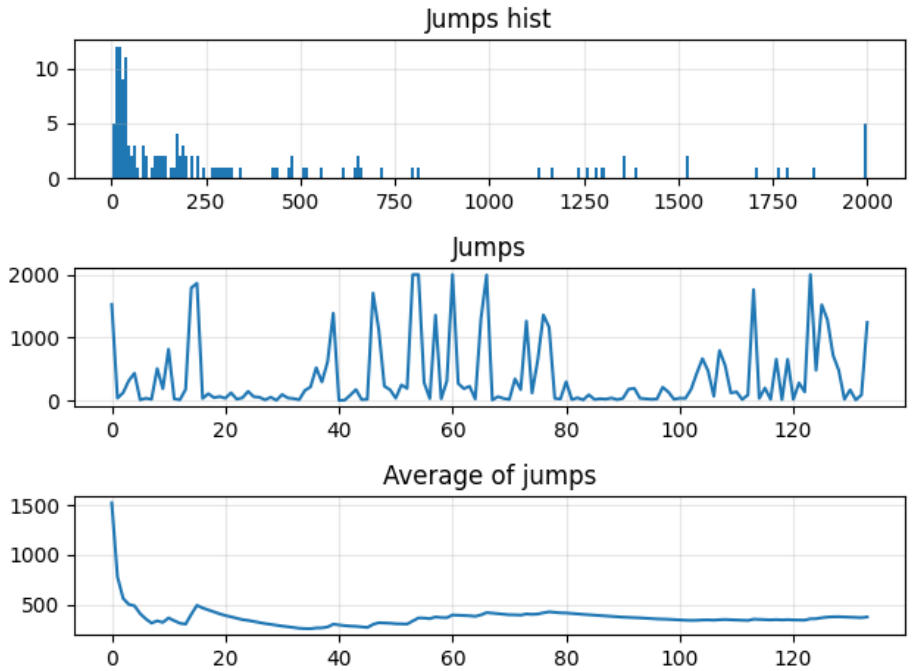
Mean residence time: $T_{\text{mean}} = 373 \times \delta t$, confidence interval $[278; 469] \times \delta t$

Reference results

History of residence times on a single trajectory:

$$[122, 23, 27, 476, 14, 32, 560, 245] \quad \times \delta t$$

Distribution of residence times (25 trajectories of 2000 time steps):



Mean residence time: $T_{\text{mean}} = 373 \times \delta t$, confidence interval $[278; 469] \times \delta t$

EAM trajectory is wrong: $T_{\text{mean}} = 91 \times \delta t$, confid. interval $[81; 100] \times \delta t$

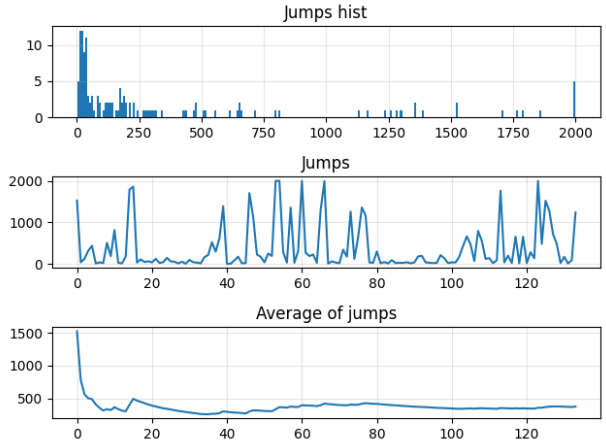
Parareal results, pathwise accuracy

		reference residence times
		[122, 23, 27, 476, 14, 32, 560, 245]
δ_{expl}	δ_{conv}	parareal residence times
0.35	10^{-5}	[63, 27, 16, 36, 19, 34, 332, 972]
0.35	10^{-10}	[122, 23, 27, 476, 14, 32, 575, 15, 28, 31, 156]

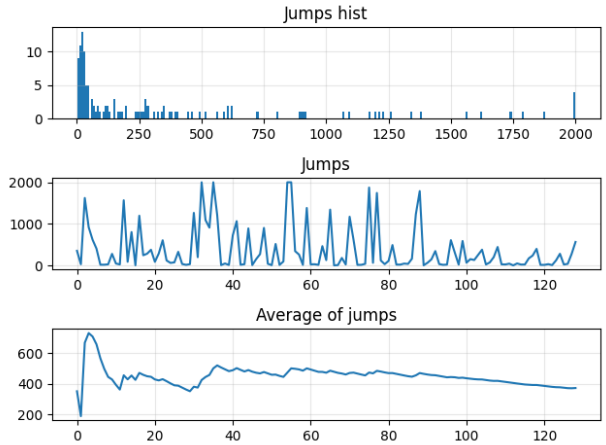
- For the convergence threshold $\delta_{\text{conv}} = 10^{-10}$, pathwise accuracy is reached.
- This is not the case for $\delta_{\text{conv}} = 10^{-5}$
- What about statistical accuracy?

Parareal results, statistical accuracy ($\delta_{\text{conv}} = 10^{-10}$)

Reference results: $T_{\text{mean}} = 373 \times \delta t$, confidence interval $[278; 469] \times \delta t$



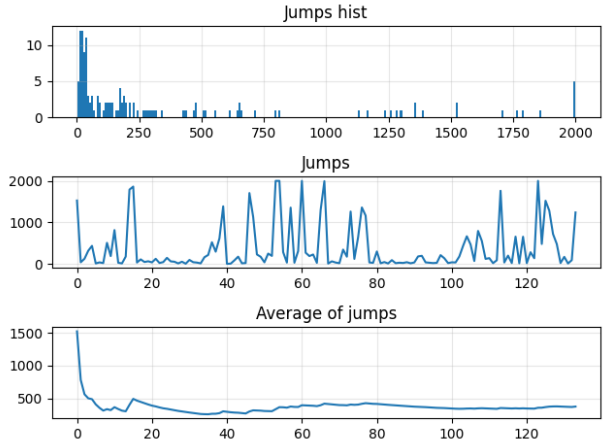
Parareal results: $T_{\text{mean}} = 372 \times \delta t$, confidence interval $[278; 466] \times \delta t$



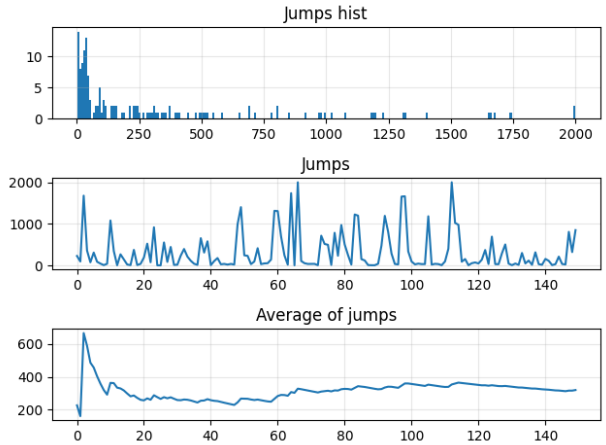
Excellent accuracy (even though noise and IC are different on reference and parareal results)

Parareal results, statistical accuracy ($\delta_{\text{conv}} = 10^{-3}$)

Reference results: $T_{\text{mean}} = 373 \times \delta t$, confidence interval $[278; 469] \times \delta t$



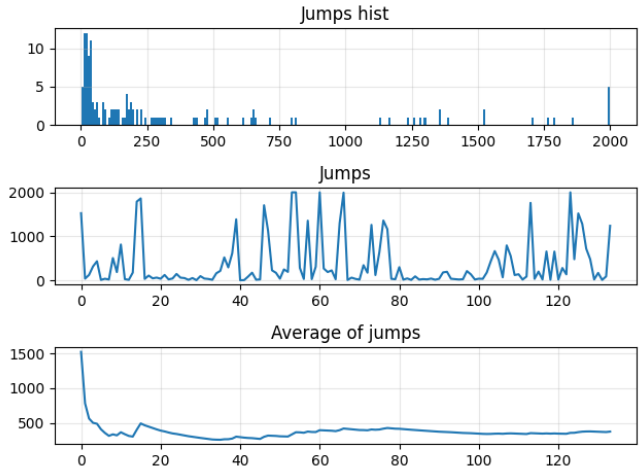
Parareal results: $T_{\text{mean}} = 320 \times \delta t$, confidence interval $[246; 394] \times \delta t$



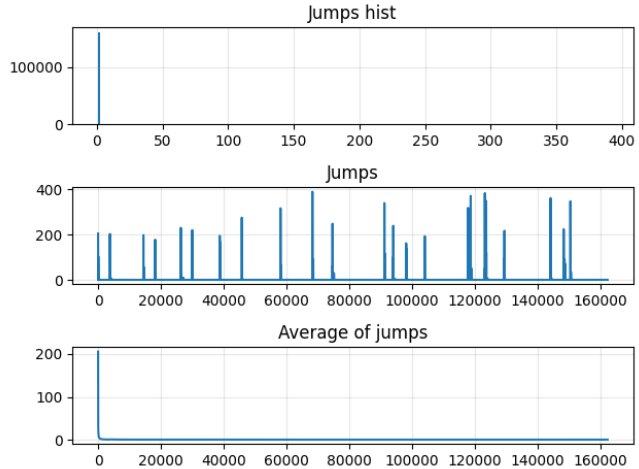
Very good statistical accuracy (overlapping confidence intervals!), while no pathwise accuracy for this value of δ_{conv}

Parareal results, statistical accuracy ($\delta_{\text{conv}} = 10^{-1}$)

Reference results: $T_{\text{mean}} = 373 \times \delta t$, confidence interval $[278; 469] \times \delta t$



Parareal results: $T_{\text{mean}} = 1.21 \times \delta t$, confidence interval $[1.19; 1.24] \times \delta t$



No accuracy at all (δ_{conv} is too large)

Gain

Gain ($\delta t = 2$ fs and $\delta_{\mathrm{conv}} = 10^{-3}$):

δ_{expl}	N_{final}	Theo. gain	Actual gain	# time-slabs to reach N_{final}
0.15	2000	4.73	4.68	90
0.20	2000	4.84	4.77	61
0.25	2000	5.06	4.95	38
0.30	2000	5.28	5.09	23
0.35	2000	5.45	5.12	13
0.40	2000	3.67	3.02	4

- The gain slightly decreases if δ_{conv} decreases.
- The gain increases if δt decreases: gain ≈ 16 for $\delta t = 0.5$ fs

Conclusions

- Non-intrusive implementation within LAMMPS is possible
  - this allows to consider realistic systems
  - needs appropriate adjustement of time-scheme
- If δ_{conv} is sufficiently small, then pathwise accuracy on the history of residence times
- Regime of intermediate δ_{conv} where no pathwise accuracy but very good statistical accuracy
- Significant computational gains are achieved on systems of physical interest

Support of ANR (through project CINE-PARA) and of EuroHPC (through project TIME-X) are gratefully acknowledged