Time-Parallel optimal control solver for parabolic equations

Mohamed Kamal RIAHI,
joint work with Yvon MADAY & Julien SALOMON

22 mai 2011

Optimal control's tools

The cost quadratic functional considered is as below

$$J(v) := \frac{1}{2}\|y(T) - y^{target}\|^2 + \frac{\alpha}{2}\int_0^T \|v\|^2 dt \tag{1}$$

The primal state variable $y(t, x; v)$ depends linearly on the control v through the heat equation starting form the initial data y_0
The first derivative of the Euler-Lagrange equations gives the following optimality system ;

$$\textit{Primal} \begin{cases} \partial_t y - \mu\Delta y = Bv \\ y(t=0) = y_0 \end{cases} \tag{2}$$

$$\textit{Dual} \begin{cases} \partial_t p + \mu\Delta p = 0 \\ p(t=T) = y(T) - y^{target} \end{cases} \tag{3}$$

$$\textit{gradient} \quad \nabla J(v) = \alpha v +^t Bp. \tag{4}$$

Summary

1. Introduction
   - Tools of : Optimal Control Solver
2. SITPOC : Parallel optimal control solver
   - Parallelization setting
3. Coupling SITPOC & Parareal
   - Parareal algorithm
   - PITPOC algorithm
4. Numerical experiments
5. Further reading
6. Conclusion

Parallelization of the optimal control solver

Now we aim to :

- Divide the global system into series of dependent time problems
- Solve iteratively sub problem independently, so that in the limit the solution of the global problem is obtained.
- The key ingredient is the introduction of the intermediate targets and initial conditions as follows.
  $\forall n \geq 0$
  - $$\lambda_n := y(t_n; v_n), \gamma_n := p(t_n; v_n) \text{on } [0, T] \times \Omega \tag{5}$$
  - $$\xi(t_n; v_n) := y(t_n; v_n) - p(t_n; v_n), \text{ on } [0, T] \times \Omega \tag{6}$$

Parallelization of the optimal control solver

Now we aim to :

- Divide the global system into series of dependent time problems
- Solve iteratively sub problem independently, so that in the limit the solution of the global problem is obtained.
- The key ingredient is the introduction of the intermediate targets and initial conditions as follows.
  $\forall n \geq 0$
  -
    $$\lambda_n := y(t_n; v_n), \gamma_n := p(t_n; v_n) \text{on } [0, T] \times \Omega \tag{5}$$
  -
    $$\xi(t_n; v_n) := y(t_n; v_n) - p(t_n; v_n), \text{on } [0, T] \times \Omega \tag{6}$$

Parallelization of the optimal control solver

Now we aim to :

- Divide the global system into series of dependent time problems
- Solve iteratively sub problem independently, so that in the limit the solution of the global problem is obtained.
- The key ingredient is the introduction of the intermediate targets and initial conditions as follows.
  $\forall n \geq 0$
  - $$\lambda_n := y(t_n; v_n), \gamma_n := p(t_n; v_n) \text{on } [0, T] \times \Omega \tag{5}$$
  - $$\xi(t_n; v_n) := y(t_n; v_n) - p(t_n; v_n), \text{on } [0, T] \times \Omega \tag{6}$$

Parallelization of the optimal control solver

Now we aim to :

- Divide the global system into series of dependent time problems
- Solve iteratively sub problem independently, so that in the limit the solution of the global problem is obtained.
- The key ingredient is the introduction of the intermediate targets and initial conditions as follows.
  $\forall n \geq 0$
  - $$\lambda_n := y(t_n; v_n), \gamma_n := p(t_n; v_n) \text{on } [0, T] \times \Omega \tag{5}$$
  - $$\xi(t_n; v_n) := y(t_n; v_n) - p(t_n; v_n), \text{on } [0, T] \times \Omega \tag{6}$$

Parallelization setting

That definition allows us to define sub cost functional as follows ;

$$J_n(w_n, \lambda_n, \xi_{n+1}) := \frac{1}{2}\|y_n(T_{n+1}^-) - \xi_{n+1}\|^2 + \frac{\alpha}{2}\int_{T_n}^{T_{n+1}} \|w_n\|^2 dt \qquad (7)$$

where $y_n(T_{n+1}^-)$ is the solution at time $t = T_{n+1}$ evolved from the initial data $y_n(T_n^+) = \lambda_n$ according to the PDE : $\partial_t y_n + \mu\Delta y_n = Bw_n$. Here we note that the local (on $I_n := [T_n, T_{n+1}]$) optimality system is :

$$\begin{cases} \partial_t y_n - \mu\Delta y_n = Bw_n \text{ on } I_n \times \Omega \\ y_n(t = n) = \lambda_n \end{cases} \qquad (8)$$

$$\begin{cases} \partial_t p_n + \mu\Delta p_n = 0 \text{ on } I_n \times \Omega \\ p_n(t_{n+1}^-) = y_n(t_{n+1}^-) - \xi_{n+1} \end{cases} \qquad (9)$$

$$\nabla J_n(w_n) = \alpha w_n +^t Bp_n. \qquad (10)$$

Theoretical support

Lemma (Consistence Lemma)

Let $\tau \in]0, T[$*, and the optimal control problem : Find* $w_\tau^\star \in \mathcal{H}$ *such that*

$$w_\tau^\star := argmin_{w\in\mathcal{H}} J_\tau(w)$$

where

$$J_\tau(w) := \frac{1}{2}\|y(\tau) - \xi^\star(\tau)\|^2 + \frac{\alpha}{2}\int_{T_n}^{T_{n+1}} \|w\|^2 \tag{11}$$

with $y(\tau)$ *the solution of Equation* (2)*. We have*

$$w_\tau^\star = v^\star_{I_{[0,\tau]}}$$

Intermediate targets *: With the notations above, denote by* $\xi^\star$ *the target trajectory defined by Equation* (6) *with* $y = y^\star$ *and* $p = p^\star$ *and by* $y_n^\star, p_n^\star, v_n^\star$ *the solutions of Equations (8–10) associated with* $v^\star$*. One has :*

$$v_n^\star = v^\star_{|_{I_n}}.$$

With an arbitrary subinterval index n

Play SITPOC algorithm

Over view of the Parareal algorithm

The parareal algorithm is an iterative preconditioned scheme that ensure convergence at its n^{th} iteration, this happens thanks to the pascal triangle behavior.

$$\lambda_{n+1}^{k+1} = \mathcal{G}_{\Delta T}(\lambda_n^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_n^k) - \mathcal{G}_{\Delta T}(\lambda_n^k) \tag{12}$$

- Compatibility with parallel architecture.
- No sleeping process (with some particular implementation).
- Fast convergence if it holds (stability question).
- For instance we get : when the error is about $1.E-4$

Nb processor	4#	8#	16#
Nb itrations	3	3	3
Wallclock mn : s	2 : 23	1 : 15	0 : 49

Over view of the Parareal algorithm

The parareal algorithm is an iterative preconditioned scheme that ensure convergence at its n^{th} iteration, this happens thanks to the pascal triangle behavior.

$$\lambda_{n+1}^{k+1} = \mathcal{G}_{\Delta T}(\lambda_n^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_n^k) - \mathcal{G}_{\Delta T}(\lambda_n^k) \tag{12}$$

- Compatibility with parallel architecture.
- No sleeping process (with some particular implementation).
- Fast convergence if it holds (stability question).
- For instance we get : when the error is about $1.E-4$

$Nb\ processor$	$4\#$	$8\#$	$16\#$
$Nb\ itrations$	3	3	3
$Wallclock\ mn:s$	$2:23$	$1:15$	$0:49$

Over view of the Parareal algorithm

The parareal algorithm is an iterative preconditioned scheme that ensure convergence at its n^{th} iteration, this happens thanks to the pascal triangle behavior.

$$\lambda_{n+1}^{k+1} = \mathcal{G}_{\Delta T}(\lambda_n^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_n^k) - \mathcal{G}_{\Delta T}(\lambda_n^k) \qquad (12)$$

- Compatibility with parallel architecture.
- No sleeping process (with some particular implementation).
- Fast convergence if it holds (stability question).
- For instance we get : when the error is about $1.E-4$

Nb processor	4#	8#	16#
Nb itrations	3	3	3
Wallclock mn : s	2 : 23	1 : 15	0 : 49

Over view of the Parareal algorithm

The parareal algorithm is an iterative preconditioned scheme that ensure convergence at its n^{th} iteration, this happens thanks to the pascal triangle behavior.

$$\lambda_{n+1}^{k+1} = \mathcal{G}_{\Delta T}(\lambda_n^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_n^k) - \mathcal{G}_{\Delta T}(\lambda_n^k) \tag{12}$$

- Compatibility with parallel architecture.
- No sleeping process (with some particular implementation).
- Fast convergence if it holds (stability question).
- For instance we get : when the error is about $1.E - 4$

$Nb\ processor$	$4\#$	$8\#$	$16\#$
$Nb\ itrations$	3	3	3
$Wallclock\ mn : s$	$2:23$	$1:15$	$0:49$

PITPOC algorithm

In this algorithm, our aim is to

- Reduce complexity by applying the coarse operator $\mathcal{G}$ instead of the fine operator $\mathcal{F}$
- Replace $y(T)(v)$ by λ_N on the functional J in order to hang over the target solution y^{target}

$$\Phi_{v^k,\lambda^k}(\theta) := \frac{1}{2}\|\lambda_N^{k+1}(\theta) - y^{target}\|^2 + \frac{\alpha}{2}\int_0^T \|v^{k+1}(\theta)\|^2 dt$$

- Use parallel information in order to correct predictor propagator in the sequential part of the algorithm
- Optimize relaxation of the coupled parareal-control algorithm

PITPOC algorithm

In this algorithm, our aim is to

- Reduce complexity by applying the coarse operator $\mathcal{G}$ instead of the fine operator $\mathcal{F}$
- Replace $y(T)(v)$ by λ_N on the functional J in order to hang over the target solution y^{target}

$$\Phi_{v^k,\lambda^k}(\theta) := \frac{1}{2}\|\lambda_N^{k+1}(\theta) - y^{target}\|^2 + \frac{\alpha}{2}\int_0^T \|v^{k+1}(\theta)\|^2 dt$$

- Use parallel information in order to correct predictor propagator in the sequential part of the algorithm
- Optimize relaxation of the coupled parareal-control algorithm

PITPOC algorithm

In this algorithm, our aim is to

- Reduce complexity by applying the coarse operator $\mathcal{G}$ instead of the fine operator $\mathcal{F}$
- Replace $y(T)(v)$ by λ_N on the functional J in order to hang over the target solution y^{target}

$$\Phi_{v^k,\lambda^k}(\theta) := \frac{1}{2}\|\lambda_N^{k+1}(\theta) - y^{target}\|^2 + \frac{\alpha}{2}\int_0^T \|v^{k+1}(\theta)\|^2 dt$$

- Use parallel information in order to correct predictor propagator in the sequential part of the algorithm
- Optimize relaxation of the coupled parareal-control algorithm

PITPOC algorithm

In this algorithm, our aim is to

- Reduce complexity by applying the coarse operator $\mathcal{G}$ instead of the fine operator $\mathcal{F}$
- Replace $y(T)(v)$ by λ_N on the functional J in order to hang over the target solution y^{target}

$$\Phi_{v^k,\lambda^k}(\theta) := \frac{1}{2}\|\lambda_N^{k+1}(\theta) - y^{target}\|^2 + \frac{\alpha}{2}\int_0^T \|v^{k+1}(\theta)\|^2 dt$$

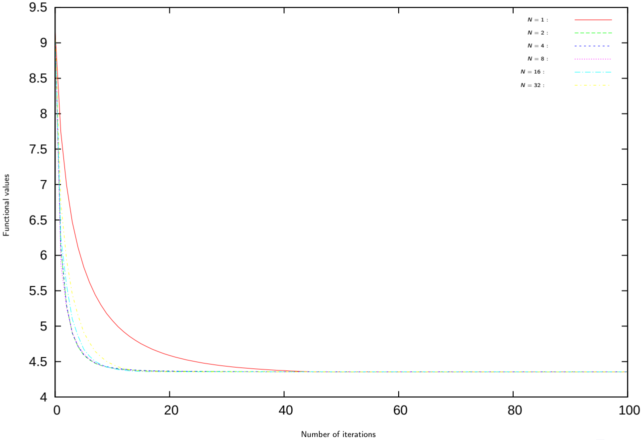
- Use parallel information in order to correct predictor propagator in the sequential part of the algorithm
- Optimize relaxation of the coupled parareal-control algorithm

Numerical experiments

Play PITPOC algorithm

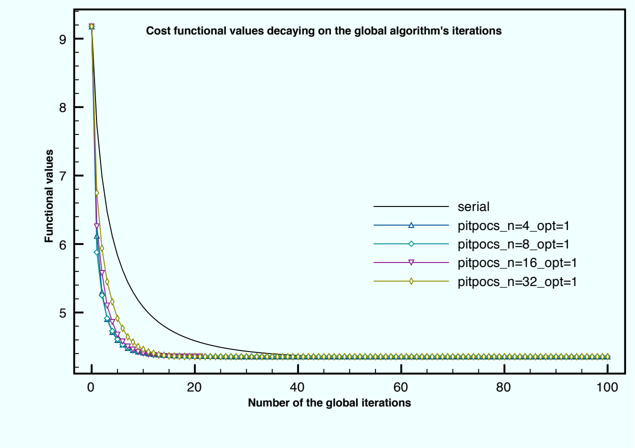
Speed up convergence I

Figure: SITPOC : Decaying against number of global iterations



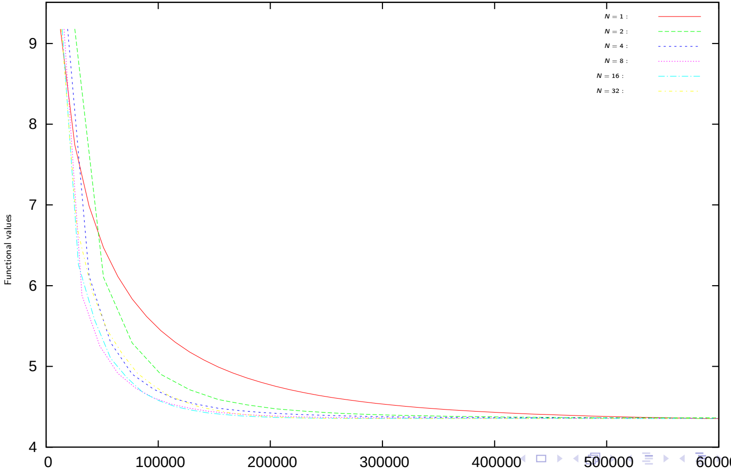
Speed up convergence II

Figure: PITPOC : Decaying against number of global iterations



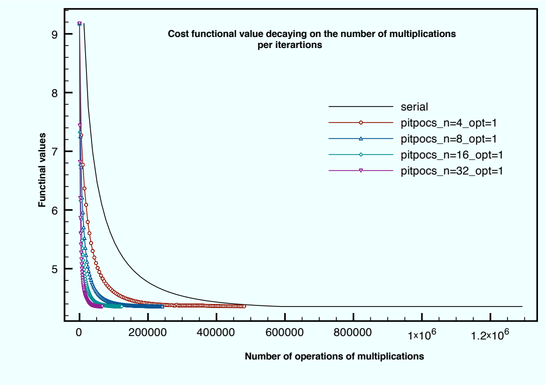
Speed up by reducing complexity I

Figure: Decaying against number of multiplications



Speed up by reducing complexity II

Figure: *Decaying of the functional value per complexity per processor*



Speed up the wallclock Simulation

Figure: *Elapsed real time for the simulation with SITPOC algorithm*

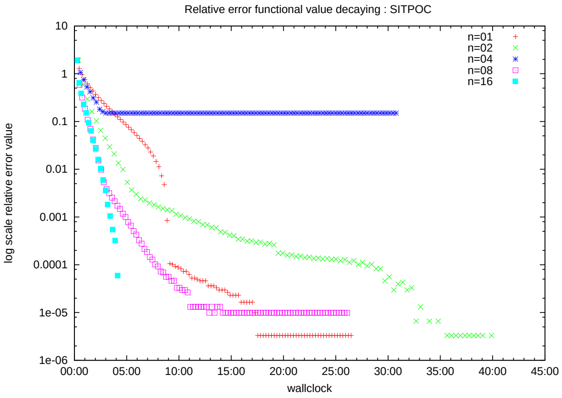
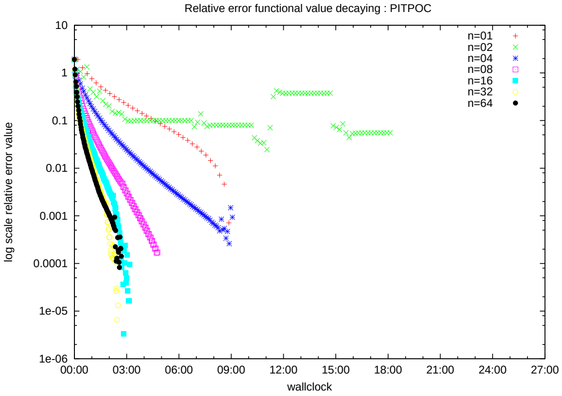


Figure: Elapsed real time for the simulation with PITPOC algorithm



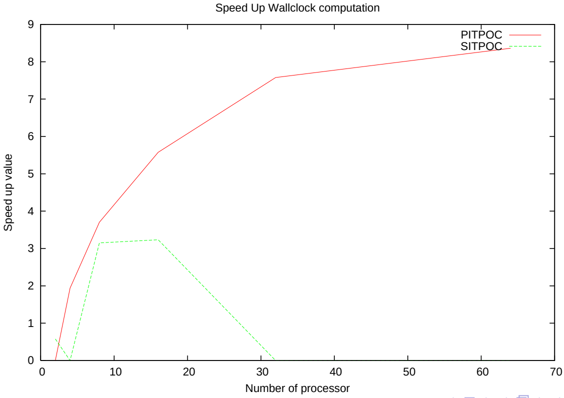
- We calculate the speed up using a serial and MPI simulation of the same problem with the same tools used as solvers.
- The reference here is the elapsed real time of an ordinary simulation (for instance optimal time step decent algorithm).
- The speed up formula reads

$$S_{p\#} = \frac{T_{1\#}(serial)}{T_{p\#}(MPI)}$$

Elapsed time to reach 1% of the result

Numer of processor#	1#	2#	4#	8#	16#	32#	64#
Timing SITPOC	08 : 05	13 : 59	*nan*	02 : 34	02 : 30	—	—
Timing PITPOC	08 : 05	*nan*	04 : 10	02 : 11	01 : 27	01 : 04	00 : 58

Figure:



For further reading

References

J.-L. Lions,Virtual and effetive control for distributed systems and decomposition of everything
J. Anal. Math. 80 257-297 ,2000

Y. Maday&G. Turinici, A parareal in time procedure for the control of partial differential equations
C. R. Math. Acad. Sci. Paris 335, 4 387-392, 2002.

G. Bal &Y. Maday A parareal time discretization for non-linear PDEs with application to the pricing of an american put ,
Springer,Lect Notes Comput. Sci. Eng. ,189-202, 2002.

J.-L. Lions &,&Y. Maday,&G Turinici, Résolution d'EDP par un shéma pararréel,
C. R. Acad. Sci Paris, I 332 , 661-668, 2001

Y. Maday& J. Salomon,& G. Turinici, Parareal in time control for quantum systems

SIAM J. Num Anal, 45(6) 2468-2482, 2007.

TAREK P. MATHEW&, MARCUS SARKIS,&, CHRISTIAN E. SCHAERER ANALYSIS OF BLOCK PARAREAL PRECONDITIONERS FOR PARABOLIC OPTIMAL CONTROL PROBLEMS,

Conclusion & perspectives

- Even if we are under the master-slaves net-framework an important speed up is shown
- Conjugate descent is applicable in the parallel resolution, and it may gives some more speed up.
- There is in algebraic interpretation for theses algorithms, that shows some relationship with the Jacobi process.
- Parareal could be coupled with others iterative solvers.
- We project to apply these parallel optimal control solvers to non-linear PDE

Conclusion & perspectives

- Even if we are under the master-slaves net-framework an important speed up is shown
- Conjugate descent is applicable in the parallel resolution, and it may gives some more speed up.
- There is in algebraic interpretation for theses algorithms, that shows some relationship with the Jacobi process.
- Parareal could be coupled with others iterative solvers.
- We project to apply these parallel optimal control solvers to non-linear PDE

Conclusion & perspectives

- Even if we are under the master-slaves net-framework an important speed up is shown
- Conjugate descent is applicable in the parallel resolution, and it may gives some more speed up.
- There is in algebraic interpretation for theses algorithms, that shows some relationship with the Jacobi process.
- Parareal could be coupled with others iterative solvers.
- We project to apply these parallel optimal control solvers to non-linear PDE

Conclusion & perspectives

- Even if we are under the master-slaves net-framework an important speed up is shown
- Conjugate descent is applicable in the parallel resolution, and it may gives some more speed up.
- There is in algebraic interpretation for theses algorithms, that shows some relationship with the Jacobi process.
- Parareal could be coupled with others iterative solvers.
- We project to apply these parallel optimal control solvers to non-linear PDE

Conclusion & perspectives

- Even if we are under the master-slaves net-framework an important speed up is shown
- Conjugate descent is applicable in the parallel resolution, and it may gives some more speed up.
- There is in algebraic interpretation for theses algorithms, that shows some relationship with the Jacobi process.
- Parareal could be coupled with others iterative solvers.
- We project to apply these parallel optimal control solvers to non-linear PDE

THANK YOU