Numerical control of wave and heat equations with reinforcement learning

S. Kadri Harouna

Laboratoire de Mathématiques, Image et Applications (MIA)
Avenue Michel Crépeau 17042 La Rochelle

Joint work with K. Ammari (University of Monastir, Tunisia) & Ghazi Bel Mufti (ESSAIT, University of Carthage, Tunisia).

Outline

Outline

Outline

Outline

Introduction

The initial-boundary value problem for the one-dimensional heat conduction that we considered is:

$$\begin{cases} \partial_t u(t,x) = \nu \partial_x^2 u(t,x) + f(t,x), \; x \in [0,1] \text{ and } t \in]0,T], \\ \\ u(0,x) = u_0(x), \end{cases} \tag{1}$$

where $\nu > 0$ is the diffusion coefficient, f is the source term and $T > 0$. Homogeneous Dirichlet boundary conditions are assumed: $u(t,0) = u(t,1) = 0$.

Objective

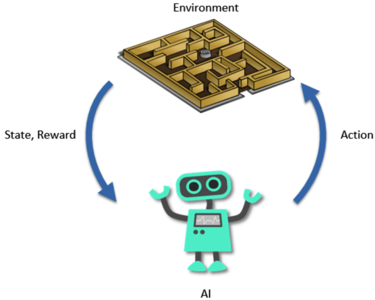
- Given a target $u_T \in L^2(0,1)$, find a source term $f(t,.) \in L^2(0,1)$, such that:

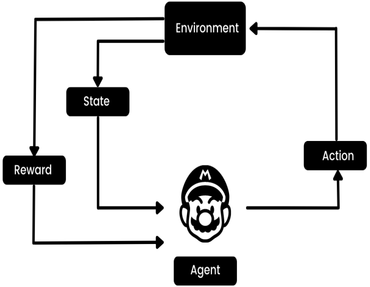
$$\|u(T,.) - u_T\|_{L^2(0,1)} \leq \epsilon \quad \text{for} \quad \epsilon > 0.$$

$\longrightarrow$ Numerical exact control remains elusive.

Introduction

Reinforcement Learning (RL) is a machine learning paradigm where an agent learns the optimal action for a given task through its repeated interaction with a dynamic environment that either rewards or punishes the agent's action.

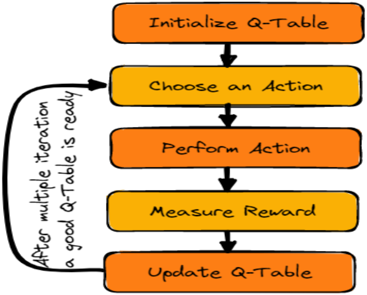




Introduction

Q-learning is a model-free, value-based, off-policy algorithm that will find the best series of actions based on the agent's current state. The Q stands for quality. Quality represents how valuable the action is in maximizing future rewards.

Q-Table: the agent maintains the Q-table of sets of states and actions.



$\longrightarrow$ **Objective:** to learn a Q-table of state and action.

Introduction

- States: s_t, the current position of the agent in the environment.

$$s_t = u(t, .)$$

- Action: a_t, a step taken by the agent in a particular state.

$$a_t = f(t, .)$$

- Rewards: R_t, for every action, the agent receives a reward and penalty.

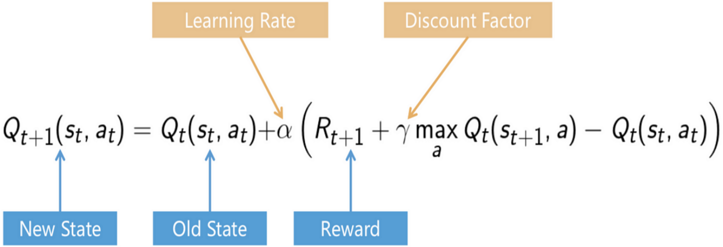
$$R_t = ?$$

- Episodes: the end of the stage, where agents can't take new action. It happens when the agent has achieved the goal or failed.

- $Q_t(s_{t+1}, a)$: expected optimal Q-value of doing the action in a particular state.

Introduction

Q-function uses the Bellman equation as a simple value iteration update, using the weighted average of the current value and the new information:



with $0 < \alpha \leq 1$ and $0 \leq \gamma \leq 1$.

Introduction

$\longrightarrow$ Is it possible to use this approach to solve the previous control problem?

$\longrightarrow$ How accurate is the method that results from this?

$\longrightarrow$ What kind of improvements can be made?

Introduction

[E. Hernández, D.Kalise, E. Otárola, 09]: Numerical approximation of the LQR problem in a strongly damped wave equation.

[M.A. Bucci, et al, 19]: Control of chaotic systems by deep reinforcement learning.

[K. Ammari,G. Bel Mufti, 23]: Controlling a dynamic system through reinforcement learning

[G. Novati, L. Mahadevan, P. Koumoutsakos, 19]: Controlled gliding and perching through deep-reinforcement-learning.

$\longrightarrow$ Wavelet approach satisfying physical boundary condition.

$\longrightarrow$ The use of wavelets is unnecessary, they provide numerical observability.

Outline

Biorthogonal wavelet basis of $L^2(\mathbb{R})$.

- $\cdots V_0 \subset \cdots \subset V_j \subset V_{j+1} \cdots \subset L^2(\mathbb{R}), \quad \cap V_j = \{0\}$
- $\forall\, j,\ V_j = \text{span} < \varphi_{j,k} = 2^{\frac{j}{2}} \varphi(2^j x - k),\ k \in \mathbb{Z} > \quad \varphi$ scaling function).

$\rightarrow$ There is another sequence $\tilde{V}_j$ with the same structure.

- $\cdots \tilde{V}_0 \subset \cdots \subset \tilde{V}_j \subset \tilde{V}_{j+1} \cdots \subset L^2(\mathbb{R}), \quad \cap \tilde{V}_j = \{0\}$
- $\forall\, j,\ \tilde{V}_j = \text{span} < \tilde{\varphi}_{j,k} = 2^{\frac{j}{2}} \tilde{\varphi}(2^j x - k),\ k \in \mathbb{Z} > \quad (\tilde{\varphi}$ dual scaling function).

Biorthogonality means:

- $\forall\, j,\ L^2(\mathbb{R}) = V_j \oplus \tilde{V}_j^{\perp} \qquad \Leftrightarrow \qquad \langle \varphi_{j,k}, \tilde{\varphi}_{j,k'} \rangle = \delta_{k,k'}$

Biorthogonal wavelet basis

Biorthogonal wavelet basis:

- Basis for the detail spaces: $W_j = V_{j+1} \cap \tilde{V}_j^{\perp}$ and $\tilde{W}_j = \tilde{V}_{j+1} \cap V_j^{\perp}$.

- $W_j = \text{span} < \psi_{j,k} = 2^{\frac{j}{2}} \psi(2^j x - k),\ k \in \mathbb{Z} >$ $\qquad$ (ψ wavelet generator)

$\longrightarrow$ The space $\tilde{W}_j$ has the same structure (with $\tilde{\psi}$ dual wavelet generator)

Finite dimensional spaces on $[0,1]$:

$$\# V_j = \# \tilde{V}_j = I_j < +\infty \quad \text{and} \quad \# W_j = \# \tilde{W}_j = 2^j$$

Biorthogonal wavelet basis

The multiscale projection of a function $f \in L^2(\mathbb{R})$ onto V_j and W_j are defined respectively by:

$$\mathcal{P}_j(f) = \sum_{k\in\mathbb{Z}} \langle f, \tilde{\varphi}_{j,k}\rangle \varphi_{j,k} \qquad \text{and} \qquad \mathcal{Q}_j(f) = \sum_{k\in\mathbb{Z}} \langle f, \tilde{\psi}_{j,k}\rangle \psi_{j,k}. \tag{2}$$

The two scales relation gives: $\mathcal{Q}_j(f) = \mathcal{P}_{j+1}(f) - \mathcal{P}_j(f)$. The multiscale decomposition of $f \in L^2(\mathbb{R})$ reads:

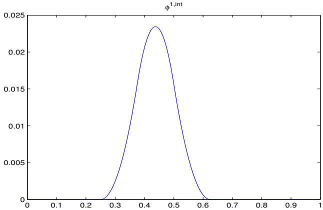
$$f = \mathcal{P}_j(f) + \sum_{\ell\geq j} \mathcal{Q}_\ell(f).$$

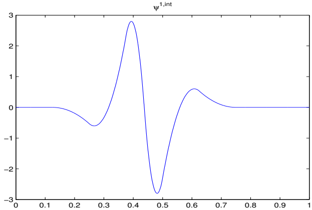
Given $f \in H^s(\mathbb{R})$, we have the following Jackson and Bernstein inequalities:

$\|\mathcal{P}_j(f) - f\|_{L^2(0,1)} \leq C2^{-js}\|f\|_{H^s(0,1)}$ and $\|\mathcal{P}_j(f)\|_{H^s(0,1)} \leq C2^{js}\|\mathcal{P}_j(f)\|_{L^2(0,1)}$, $s > 0$.

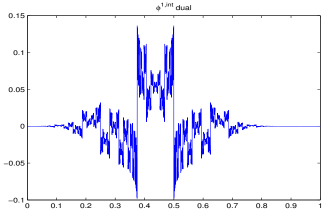
Biorthogonal B-Spline wavelets (3 vanishing moments)

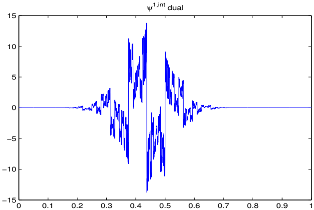
Primal scaling function (left) and associated wavelet (right):





Dual scaling function (left) and associated wavelet (right):





Wavelet basis satisfying boundary conditions

- Start with $(V_j^1, \tilde{V}_j^1)$ regular MRA of $L^2(0,1)$ associated to $(\varphi^1, \tilde{\varphi}^1)$.

- Polynomial reproduction at boundaries 0 and 1:

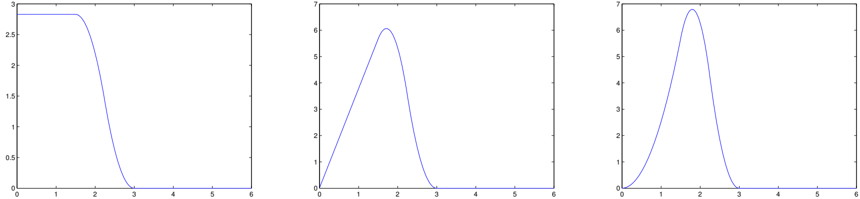
$$0 \leq \ell \leq r-1, \quad \frac{2^{j/2}(2^j x)^\ell}{\ell!} = \Phi_{j,\ell}^{1,\flat}(x) + \sum_{k=k_0}^{2^j-k_1} p_\ell^1(k)\, \varphi_{j,k}^1(x) + \Phi_{j,\ell}^{1,\sharp}(1-x)$$

with $\varphi_{j,k}$ internal scaling functions, $\Phi_{j,\ell}^{1,\flat}$ and $\Phi_{j,\ell}^{1,\sharp}$ the edge scaling functions.

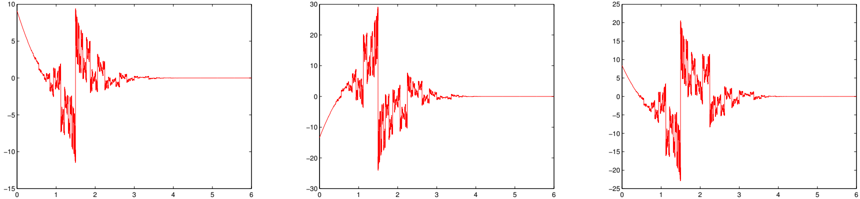
- Similar structure for $\tilde{V}_j^1$ with $\tilde{\varphi}_{j,k}$, $\tilde{\Phi}_{j,\ell}^{1,\flat}$ and $\tilde{\Phi}_{j,\ell}^{1,\sharp}$.

Wavelet basis satisfying boundary conditions

Edge 0 scaling function of V_j^1: B-Spline 3.3

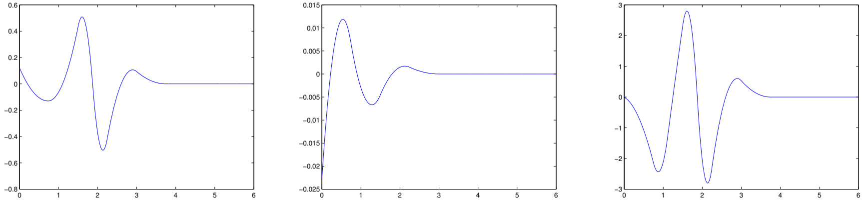


Edge 0 scaling function of $\tilde{V}_j^1$: B-Spline 3.3

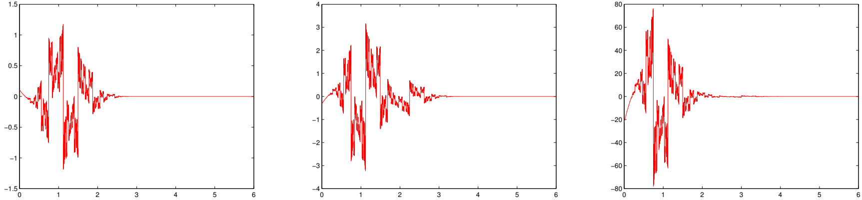


Wavelet basis satisfying boundary conditions

Edge 0 wavelets of W_j^1: B-Spline 3.3



Edge 0 wavelets of $\tilde{W}_j^1$: B-Spline 3.3



Wavelet basis satisfying boundary conditions

- Homogeneous boundary conditions:

- $f^{(\lambda)}(\alpha) = 0$ for $0 \leq \lambda \leq r-1$ and $\alpha = 0$ ou $\alpha = 1$

- It suffices to remove from V_j^1, the scaling functions:

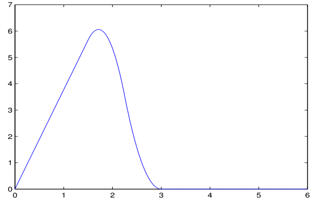
- $\{\Phi_{j,\ell}^{1,\flat}\}_{\ell=\lambda+1}$ if $\alpha = 0$ or $\{\Phi_{j,\ell}^{1,\sharp}\}_{\ell=\lambda+1}$ if $\alpha = 1$

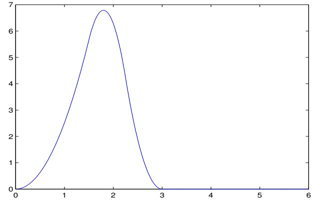
- The dual space dimension must be adjusted $\tilde{V}_j^1$:

- One can proceed similarly for $\tilde{V}_j^1$.

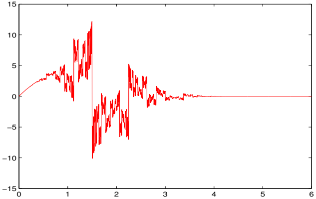
Wavelet basis satisfying boundary conditions

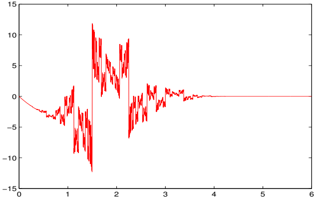
Edge 0 **scaling functions with Dirichlet:** **B-Spline** 3.3





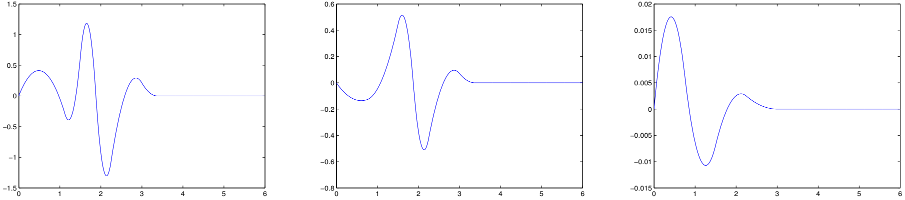
Edge 0 **dual scaling functions with Dirichlet:** **B-Spline** 3.3



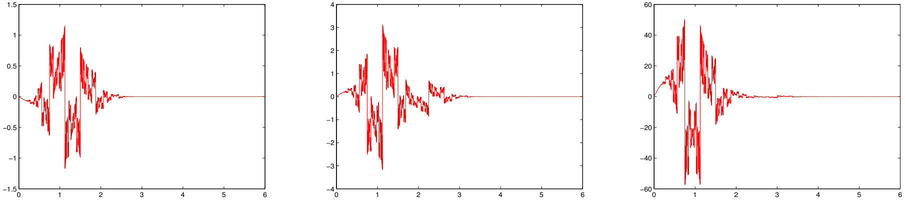


Wavelet basis satisfying boundary conditions

Edge 0 **wavelets with Dirichlet: B-Spline** 3.3



Edge 0 **dual wavelets with Dirichlet: B-Spline** 3.3



Outline

Wavelet-based Galerkin method for the heat equation

The solution $u_j \in V_j$ of (1) is searched in the following discrete form:

$$u_j(t,x) = \sum_{k=1}^{N_j} \langle u, \tilde{\psi}_{j,k} \rangle \psi_{j,k}(x) = \sum_{k=1}^{N_j} d_{j,k}(t)\psi_{j,k}(x). \quad (3)$$

For $m = 1, \ldots, N_j$, integration by part and the boundary conditions lead to:

$$\sum_{k=1}^{N_j} \left[d'_{j,k}(t)\langle \psi_{j,k}, \psi_{j,m} \rangle + \nu d_{j,k}(t) \langle \psi'_{j,k}, \psi'_{j,m} \rangle \right] = \langle f(t,.), \psi_{j,m} \rangle. \quad (4)$$

Thus, the coefficients $(d_{j,k})$ are solution of a differential system:

$$\mathcal{A}_j \left[d'_{j,k}(t) \right] + \mathcal{R}_j \left[d_{j,k}(t) \right] = \mathcal{A}_j \left[f_{j,k}(t) \right], \quad (5)$$

with

$$[\mathcal{A}_j]_{k,m} = \int_0^1 \psi_{j,k}(x)\psi_{j,m}(x)dx \quad \text{and} \quad [\mathcal{R}_j]_{k,m} = \nu \int_0^1 \psi'_{j,k}(x)\psi'_{j,m}(x)dx. \quad (6)$$

$\longrightarrow$ Symmetric and positive definite matrices with diagonal preconditioners.

Wavelet-based Galerkin method for the heat equation

Posteriori error estimate

Proposition

Let u and u_j be solutions of (1) and (4), respectively. If the initial conditions $u_0(x)$ and the wavelet basis are *regular enough*, then we have:

$$\forall t \in]0, T], \quad \|u_j - u\|_{L^2(0,1)} \leq C2^{-js}, \tag{7}$$

for all $j \geq j_{min}$ and $s > 0$.

$\longrightarrow$ j_{min} the smallest resolution to avoid boundary functions support overlapping

Wavelet-based Galerkin method for the heat equation

Then, if $\|u_j - \mathcal{P}_j(u_T)\|_{L^2(0,1)} \leq \frac{\epsilon}{2}$, using the triangle inequality, we have:

$$\begin{aligned} \|u(T) - u_T\|_{L^2(0,1)} &\leq \|u(T) - \mathcal{P}_j(u(T))\|_{L^2(0,1)} + \|u_T - \mathcal{P}_j(u_T)\|_{L^2(0,1)} \\ &+ \|u_j - \mathcal{P}_j(u_T)\|_{L^2(0,1)} \leq C2^{-js} + \frac{\epsilon}{2}, \end{aligned}$$

where the constant depends on the $H^s(0,1)$ norm of the data.

Taking the integer j large enough, $j \geq -\frac{\ln(\frac{\epsilon}{2C})}{s\ln(2)}$, allows to get:

$$\|u(T) - u_T\|_{L^2(0,1)} \leq \epsilon.$$

Given $d^T_{j,k} \sim \mathcal{P}_j(u^T)$, the control problem is thus reduced to find $[f_{j,k}(t)]$ such that:

$$\|d_{j,k}(T) - d^T_{j,k}(t)\|_{\ell^2} \leq \frac{\epsilon}{2}.$$

Wavelet-based Galerkin method for the heat equation

In the sequel, we assume

$$[f_{j,k}(t)] = \mathcal{B}_j\,[v_{j,k}(t)]\,, \text{ where } v_j = \sum_{k=1}^{N_j} v_{j,k}(t)\psi_{j,k}(x),$$

and $\mathcal{B}_j$ a suitable real matrix of rank less than N_j. Then, system (5) rewrites:

$$\left[d'_{j,k}(t)\right] + \mathcal{M}_j\,[d_{j,k}(t)] = \mathcal{B}_j\,[v_{j,k}(t)] \quad \text{with} \quad \mathcal{M}_j = \mathcal{A}_j^{-1}\mathcal{R}_j. \tag{8}$$

$\longrightarrow$ ODE system control: Kalman rank criterion for $\mathcal{M}_j$ and $\mathcal{B}_j$.

Time discretization of the ODE system

For a time step $\delta t > 0$ and integer $n \geq 0$, we search:

$$x_{t_n} \approx d_{j,k}(n\delta t) \text{ and } v_{t_n} \approx v_{j,k}(n\delta t).$$

An explicit Euler scheme leads to:

$$x_{t_{n+1}} = f(x_{t_n}, v_{t_n}) = A_{\delta t} x_{t_n} + B_{\delta t} v_{t_n}, \tag{9}$$

where

$$A_{\delta t} = I + \delta t \mathcal{M}_j \text{ and } B_{\delta t} = \delta t \mathcal{B}_j.$$

$\longrightarrow$ Implicite numerical schemes can be used.

ODE system control by reinforcement learning

To obtain control of equation (9), a linear feedback controller is usually designed:

$$v_{t_n} = P_{t_n} x_{t_n}.$$

The matrix P_{t_n} is a solution to an algebraic Riccati equation when minimising this quadratic cost function:

$$J_N = \frac{\delta t}{2} \sum_{n=0}^{N} [\langle E_{\delta t} x_{t_n}, x_{t_n} \rangle + \langle R_{\delta t} v_{t_n}, v_{t_n} \rangle] + \frac{1}{2} \langle E_N x_{t_N}, x_{t_N} \rangle, \quad T_N = N\delta t = T,$$

constrained by (9).

$\longrightarrow$ LQR regularization in the literature.

ODE system control by reinforcement learning

Alternatively, the linear feedback is obtained using an improved Q-learning policy approach:

$$r_{t_n} = r(x_{t_n}, v_{t_n}) =< x_{t_n}, E_{\delta t} x_{t_n} > + < v_{t_n}, R_{\delta t} v_{t_n} > . \tag{10}$$

In this case, the value of the total cost for x_{t_n} under policy P_{t_n} is:

$$V_{P_{t_n}}(x_{t_n}) = \sum_{i=0}^{N-1} \gamma^i r_{t_n+i} =< x_{t_n}, K_{t_n} x_{t_n} >, \quad 0 < \gamma < 1,$$

where K_{t_n} denotes the cost matrix related to the policy defined by P_{t_n}.

Thus, the Q-function is:

$$Q_{t_n}(x, v) = r(x, v) + \gamma V_{P_{t_n}}(f(x, v)).$$

ODE system control by reinforcement learning

The Q-function's value at the next time step is:

$$Q_{t_{n+1}}(x_{t_n}, v_{t_n}) = (1-\alpha)Q_{t_n}(x_{t_n}, v_{t_n}) + \alpha\left[r(x_{t_n}, v_{t_n}) + \gamma Q_{t_n}(x_{t_{n+1}}, v_{t_{n+1}})\right],$$

where

$$v_{t_{n+1}} = P_{t_{n+1}} x_{t_{n+1}}.$$

The matrix $P_{t_{n+1}}$ is the improved policy matrix computed from P_{t_n} such that:

$$P_{t_{n+1}} x = \arg\min_v [r(x, v) + \gamma V_{P_{t_n}}(f(x, v))]. \tag{11}$$

Using forward calculations, we see that:

$$P_{t_{n+1}} = -\gamma(R_{\delta t} + \gamma B^*_{\delta t} K_{t_n} B_{\delta t})^{-1} B^*_{\delta t} K_{t_n} A_{\delta t}$$

$\longrightarrow$ P_{t_n} and K_{t_n} are obtained by means of a dynamic programming procedure.

ODE system control by reinforcement learning

Recursive LQR algorithme via DP:

$$J_N = \frac{\delta t}{2} \sum_{k=0}^{N} [\langle E_{\delta t} x_{t_k}, x_{t_k} \rangle + \langle R_{\delta t} v_{t_k}, v_{t_k} \rangle] + \frac{1}{2} \langle E_N x_{t_N}, x_{t_N} \rangle.$$

$K_T = E_N$.

for $n = N$ *to* 1 **do**

$$K_{t_{n-1}} = E_{\delta t} + A_{\delta t}^* K_{t_n} A_{\delta t} - A_{\delta t}^* K_{t_n} B_{\delta t} (R_{\delta t} + \gamma B_{\delta t}^* K_{t_n} B_{\delta t})^{-1} B_{\delta t}^* K_{t_n} A_{\delta t}$$

for $n = 0$ *to* $N - 1$ **do**

$$P_{t_n} = -\gamma (R_{\delta t} + \gamma B_{\delta t}^* K_{t_{n+1}} B_{\delta t})^{-1} B_{\delta t}^* K_{t_{n+1}} A_{\delta t}$$

$$v_{t_n} = P_{t_n} x_{t_n}$$

end

end

ODE system control by reinforcement learning

Classical Q-learning algorithm

Input: $\mathcal{S}$, $\mathcal{A}$, α, γ
Output: $Q-$table
for *each episode* **do**
  Initialize the first state
  for *each step* **do**
    Given current state s, select action a with an ϵ-greedy policy
    Observe r and s' from the environment
    Update the Q-table:
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    Update s
    until end of the episode
  end
end

Special case:

$\longrightarrow Q_{t_{n+1}}(x_{t_n}, v_{t_n}) = Q_{t_n}(x_{t_n}, v_{t_n}) + \alpha\left[r(x_{t_n}, v_{t_n}) + \gamma Q_{t_n}(x_{t_{n+1}}, v_{t_{n+1}}) - Q_{t_n}(x_{t_n}, v_{t_n})\right]$

ODE system control by reinforcement learning

Linking with the Markov Decision Process

$$\mathcal{P}(s'|s=(t_n,x),v)=\delta[s'=(t_{n+1},x')],$$

$$r(s=(t_n,x),v)=\frac{\delta t}{2}\left[\langle E_{\delta t}x,x\rangle+\langle R_{\delta t}v,v\rangle\right],$$

$$x'=f(x,v)=A_{\delta t}x+B_{\delta t}v$$

with the terminal condition

$$\mathcal{P}(s'|s=(t_N,x),v)=\delta[s'=x_T],$$

$$r(s=(t_N,x),v)=\frac{1}{2}\langle E_N x,x\rangle,$$

Error estimate (7) suppose the control v is known. If $\bar{v}_j = (\bar{v}_{t_0}, \ldots, \bar{v}_{t_{N-1}})$ realizes the minimum of:

$$J_N(v_j) = \frac{\delta t}{2} \sum_{k=0}^{N} [\langle E_{\delta t} x_{t_k}, x_{t_k} \rangle + \langle R_{\delta t} v_{t_k}, v_{t_k} \rangle] + \frac{1}{2} \langle E_N x_{t_N}, x_{t_N} \rangle, \tag{12}$$

constrained by (9), we have:

Proposition

Let $\bar{v}_j$ be the minimum of (12) constrained by (9), for regular data, we have:

$$\|\bar{v} - \bar{v}_j\|_{L^2} \leq C 2^{-js}, \tag{13}$$

with $s > 0$ the Sobolev smoothness of u and $\bar{v}$ the control with the lowest quadratic cost.

$\longrightarrow$ This comes from the error on the adjoint model of (9).

Numerical results

To evaluate the Galerkin approximation, as analytical solution, we used:

$$u(t,x) = e^{-t}\sin(2\pi x). \tag{14}$$

Then, the initial condition is

$$u_0(x) = \sin(2\pi x),$$

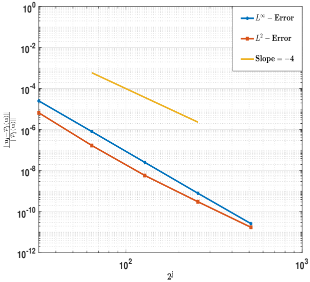
and the diffusion coefficient is set to $\nu = 1/(4\pi^2)$ in order to satisfy (1). The time step is $\delta t = 1/1000$.

For comparing the numerical and the exact solution, we use the following relative error:

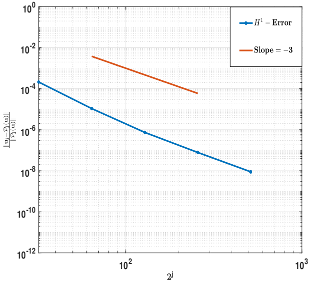
$$\mathbf{e}_j = \frac{\|\mathcal{P}_j[u(.,t)] - u_j(.,t)\|}{\|\mathcal{P}_j u(.,t)\|}.$$

Numerical results

Galerkin discretization error



(a)



(b)

Figure: Relative errors norm on the exact solution (14) at the simulation final time $T \approx 3$. Daubechies orthogonal generator with $r = 4$ vanishing moments.

Numerical results

The Q-learning algorithm was evaluated with an initial condition

$$x \mapsto u_0(x) = \sin(\pi x), \quad x \in [0,1].$$

The simulation final time is $T = 1$, $\delta t = 1/100$ and $\nu = 1/4\pi^2$. We aim to find $v_{j,k}$ such that $d_{j,k}(t)$ match the wavelet coefficients of

$$u(t,x) = \exp(1-t)\sin^3(2\pi x) + 8x(1-x)^2, \quad x \in [0,1], \tag{15}$$

at $t = T$.

The performance indicators considered are the ℓ^2 error:

$$er_j = \frac{\|d_{j,k}(T) - d_{j,k}^T\|_{\ell^2(\mathbb{Z})}}{\|d_{j,k}^T\|_{\ell^2(\mathbb{Z})}},$$

and the convergence ratio with respect to the change of the policy:

$$rt_j = \frac{\|d_{j,k}(t_n)\|_{\ell^2(\mathbb{Z})}}{\|d_{j,k}^T\|_{\ell^2(\mathbb{Z})}}, \quad 0 \leq n \leq N.$$

Hilbert Uniqness Method (HUM)

Continuous case:

$$v(t) = B^* e^{-(T-t)A^*} q_j^T,\ \mathbb{M} q_j^T = u^T - e^{-TA} u_0,\ \mathbb{M} = \int_0^T e^{-(T-s)A} B B^* e^{-(T-s)A^*} ds. \tag{16}$$

Discrete case:

Initialize: u_j^0, $w_j^0 = (A_{\delta t}^{N-1})^* q_j^T$ and $\bar{v}_j^0 = B_{\delta t}^* w_j^0$.

for $k = 1 \to N-1$ **do**

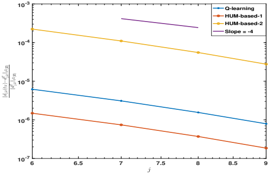
Update the control:
$w_j^k = (A_{\delta t}^{-1})^* w_j^{k-1}, \quad \bar{v}_j^k = B_{\delta t}^* w_j^k$

Update the solution:
$u_j^{k+1} = A_{\delta t} u_j^k + B_{\delta t} \bar{v}_j^k$.

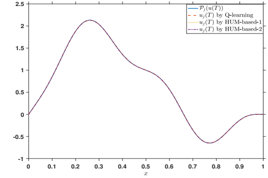
end

Algorithm 1: HUM control

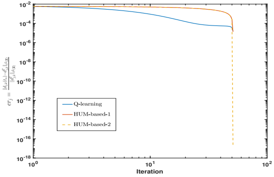
Numerical results



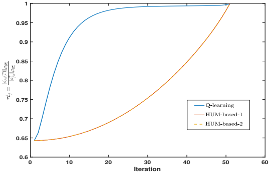
(a) $\frac{\|u_j(T)-\mathcal{P}_j(u^T)\|_{\ell^2}}{\|\mathcal{P}_j(u^T)\|_{\ell^2}}$



(b) $u_7(T)$

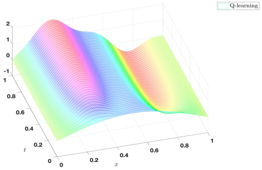


(c) Error er_j

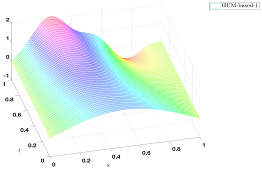


(d) ratio rt_j

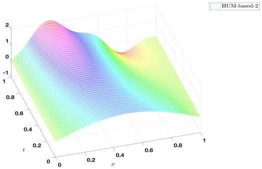
Numerical results



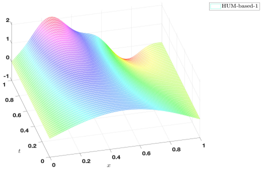
(e) Q-learning



(f) HUM -1



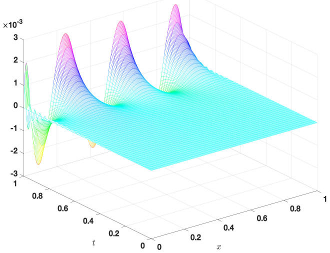
(g) Algorithm 1



(h) HUM-1

Figure: Evolution of $u_j(t_n)$ for $j = 7$ and $0 \leq t_n \leq T = 1$.

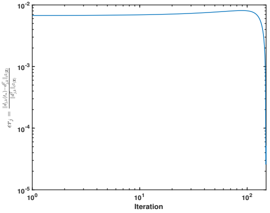
Numerical results



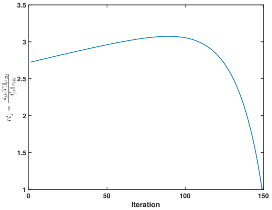
(a)

Figure: Residual error between the solutions presented on Figure **??**.

Numerical results: wave equation



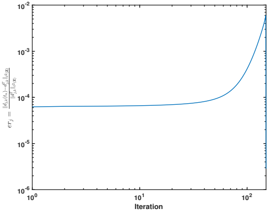
(a)



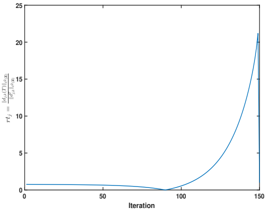
(b)

Figure: Relative error er_j (left) and the convergence ratio rt_j (right), according to the number of iterations for u_j: Q-learning method in the case of wave equation.

Numerical results: wave equation



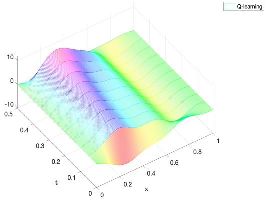
(a)



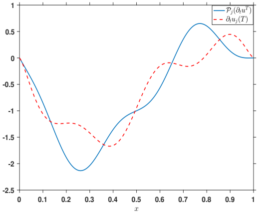
(b)

Figure: Relative error er_j (left) and the convergence ratio rt_j (right), according to the number of iterations for $\partial_t u_j$: Q-learning method in the case of wave equation.

Numerical results: wave equation



(a)



(b)

Figure: Time evolution of the solution $u_j(t_n)$ at grid points (left) $0 \leq t_n \leq 0.1$ and residual error $u_j(T) - \mathcal{P}_j(u^T)$ at grid points, for the wave equation with the proposed method and $j = 7$.

Numerical results: computational time

Two-dimensional space			
j	6	7	8
er_j	9.5593×10^{-4}	5.4462×10^{-4}	3.1054×10^{-4}
rt_j	0.9916161	0.9916162	0.9916028
CPU time in seconds	0.0700	0.1900	0.4800

Three-dimensional space			
j	6	7	8
er_j	8.1188×10^{-4}	4.5918×10^{-4}	2.5971×10^{-4}
rt_j	0.99157743	0.9915775	0.99157754
CPU time in seconds	1.8300	22.0200	243.3400

Table: Results obtained for heat equation using the Q-learning method in higher dimension.

Thank you for your attention

- K. Ammari, G. Bel Mufti, S. Kadri Harouna, *Reinforcement learning for the control of parabolic and hyperbolic differential equations*, in the pipeline.