Stochastic Control and Machine Learning : Lab sessions

Samy Mekkaoui

2025-11-09

Table of contents

Organisation

Welcome to the webpage of the practical sessions of the course **Apprentissage automatique et contrôle stochastique** taught by Professor H.Pham within the Master's program in Probability and Finance (M2-PF).

Logo of the M2-PF's Master

The practical sessions will be divided into three 3-hour sessions, each illustrating the concepts covered during the lecture.

- **Schedule**: The practical sessions will take place on ... each time in **room 15/25 104** at Jussieu.
- **Materials**: The practical sessions will be conducted in **Python**. You should bring your own laptop and have your own Python environment set up for each session.
- **Planning** :
  - **Lab work n°1** : About **Deep PDE Solver** for solving partial differential equations.
  - **Lab work n°2** : About **Reinforcement Learning** for stochastic control problems.
  - **Lab work n°3** : About **Generative IA** and **Schrodinger Bridge** for data generation.
- **Grade** : ... To be confirmed

Use this website

The site is structured into three sections that make up the course, each consisting of two chapters. The first chapter, titled **Course reminders**, presents the key theoretical concepts, followed by the second chapter, which contains the practical session instructions and a link to a **Jupyter notebook file** where you can write your code.

Note that the Jupyter notebooks files are designed to be **self-sufficient**, allowing you to work independently during the practical sessions.

For your information, this site is generated using Quarto, and the notes are available on this GitHub page. If you spot any errors on the site, feel free to report them through pull requests.

Instructors

The Lab sessions for this course have been prepared by Samy and Alexandre. We want to thank Huyên Pham for beeing our advisor and for the confidence in letting us preparing the lab works of his course.

The **prerequisites** for this course are the following:

- Good knowledge on **Probability Theory** and **Stochastic Processes**.
- Basics on **optimal control theory**.
- Basics on **Deep Learning**.
- Familarity with **Python** and potentially with **PyTorch**.

If you have any **questions**, you can either e-mail Samy or Alexandre

Acknowledgements

We would like to thank the following individuals and organizations without whose support this course would not be possible

- Professor Huyên Pham for the support and valuable feedbacks during the creation of this lab.
- Yadh Hafsi and Rémi Surat for their multiple feedbacks on the lectures notes and for their invaluable contributions on the labs.
- **Participants** in the course of your interest in Machine Learning for stochastic control and its applications.

Part I

Part n°1 : Deep Learning for PDE

1 Course reminders

The following is a reminder of the main theoretical results about **Deep Learning algorithms for PDE** seen during the course. full and self-content **PDF version** is also available at this link.

1.1 Some reminders on PDE and stochastic control

1.1.1 Stochastic control in a nutshell

1.1.2 The dynamic programming approach : HJB equation

1.2 Some reminders on neural networks

1.2.1 Feedforward neural networks (FFNN)

1.2.2 Other neural network architectures

1.2.2.1 Recurrent neural networks (RNN)

1.2.2.2 Long short term memory (LSTM)

1.3 Neural networks algorithms for PDE

1.3.1 Deep Galerkin Method

1.3.1.1 Algorithm Description

1.3.1.2

1.3.2 Deep BSDE Solver

1.3.2.1 A quick overview on BSDE

1.3.2.2 Algorithm Description

1.3.3 Deep BDP Solver

1.3.3.1 Algorithm Description

1.4 An extension : Deep Learning for MDP

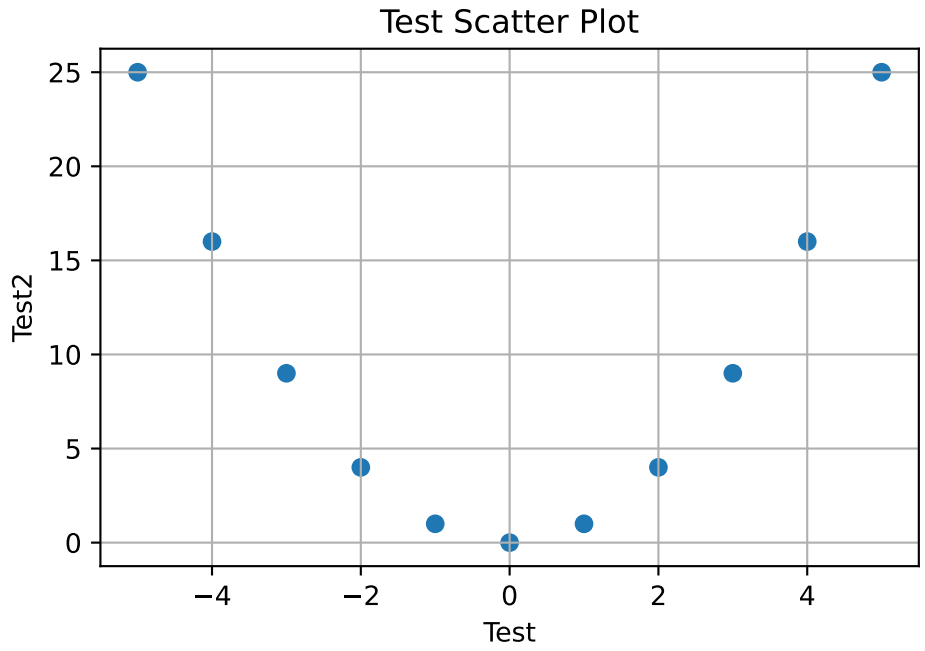
```
#  Import des Librairies

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Test
print(np.arange(5))
```

```python
def f(x) :
    return x**2

L=[i for i in range(-5,6)]

plt.scatter(L,[f(l) for l in L])
plt.xlabel("Test")
plt.ylabel("Test2")
plt.title("Test Scatter Plot")
plt.grid()

# Exemple simple de DataFrame
data = {
    'Nom': ['Alice', 'Bob', 'Charlie', 'David'],
    'Âge': [24, 30, 18, 22],
    'Ville': ['Paris', 'Lyon', 'Marseille', 'Toulouse'],
    'Score': [85.5, 90.0, 78.0, 88.5]
}

df = pd.DataFrame(data)
df
```

```
[0 1 2 3 4]
```

```
C:\Users\samym\anaconda3\lib\site-packages\IPython\core\formatters.py:342: FutureWarning:

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `Styler
```

	Nom	Âge	Ville	Score
0	Alice	24	Paris	85.5
1	Bob	30	Lyon	90.0
2	Charlie	18	Marseille	78.0
3	David	22	Toulouse	88.5



Definition 1.1. A noter qu'il faut être à l'aise sur l'utilisation de Quarto

2 Lab work n°1

You can download the **Lab work n°1** by clicking here. All the relevant informations are already provided on the notebook.

All the important information is already presented in the notebook. The following is a simple summary of what you will learn to do during the lab work and what is expected of you for the final submission if you choose this lab.

2.1 Summary of the Lab work n°1

The Lab work is divided into 3 parts :

- The first part is devoted to get more familiar with the PyTorch library for the ones who are not that familiar with the library.
- The second part is devoted to the implementation of the Deep Galerkin algorithm with so
- The third part is devoted to the implementation of the Deep BSDE Solver and to

2.1.1 Goals

The TP aims you to implement the various PDE algorithms seen during the class session :

- Get more familiar with the PyTorch library
- Implement the Deep Galerkin and Deep BSDE algorithm and benchmark it with some classic PDE for which we know explicit or semi explicit formulas.

2.1.2 Your expected work

2.1.3 Submission deadline

... : TBD.

2.2 Expected results

2.2.1 About the Deep Galerking Algorithm

2.2.2 About the Deep BSDE Solver

Part II

Part n°2 : Reinforcement Learning for stochastic control problems

3 Course reminders

The following is a reminder of the main theoretical results about **Reinforcement Learning for stochastic control problems** seen during the course. A full **PDF version** is also available at this link.

3.1 Some Foundations of Reinforcement Learning

We will introduce in the following the main concepts of Reinforcement Learning. If you want to look for more in depth theory, you can look at

3.1.1 Basics of Reinforcement Learning

Definition 3.1. A Markov Decision Process is a quadruplet given by $(\mathcal{X}, \mathcal{A}, P, r = (f, g))$ such that :

- $\mathcal{X}$ denotes the space of states on which the discrete time state process $(X_t)_{t\in\mathbb{N}}$
- $\mathcal{A}$ denotes the space of actions in which the control $(\alpha_t)_{t\in\mathbb{N}}$ is defined
- State dynamics given by :
$$X_{t+1} \sim P_t(X_t, \alpha_t)$$
with a probability transition given by an application $(t, x, a) \in \mathbb{N} \times \mathcal{X} \times \mathcal{A} \mapsto P_t(x, a, dx') \in \mathcal{P}(\mathcal{X})$.
- Reward given by a couple (f, g) such that :
  - $f(x, a)$ is a running reward obtained in state x when choosing the action a
  - Terminal reward $g(x)$
  - Discount factor $\beta \in [0, 1]$

Now, that we have defined the main components of a reinforcement learning problem, we can define the notion of policy

Definition 3.2. A policy $\pi = (\pi_t)_{t\in\mathbb{N}^*}$ is a sequence of actions choosen in a markovian setting with respect to the state variable. A policy π can be either :

- deterministic when $\pi_t : \mathcal{X} \mapsto \mathcal{A}$
- randomized when $\pi_t : \mathcal{X} \mapsto \mathcal{P}(\mathcal{A})$ meaning that π_t is a probability distribution of choosing an action at time t in state x.

We will say that a control $\alpha = (\alpha_t)_{t\in\mathbb{N}}$ is drawn from a policy π if for each $t \in \mathbb{N}$, we have :

- $\alpha_t = \pi_t(X_t)$ in the case of deterministic policies
- $\alpha_t \sim \pi_t(X_t)$ in the case of randomized policies.