Koopman operators for Reinforcement Learning: from value-function estimation to policy gradient

Francesco Zanini and Alessandro Chiuso
University of Padua

The Koopman operator framework allows to rewrite the problem in a functional space!

- State dynamics: \( s_{t+1} = f(s_t) \)
- Observable dynamics: \( \psi(s_{t+1}) = \psi(f(s_t)) = \psi_+(s_t) \)
- \( U: F \to F \), \( U(\psi(\cdot)) = \psi(f(\cdot)) = \psi_+(\cdot) \)

Why use the Koopman operator?

State dynamics may be overly complex, while the reward function is user-defined and is used to define the goal of the agent. It can be then safely assumed that reward is smooth and well-behaved.

Instead of learning system dynamics, a sequence of functions is learned, corresponding to the iterate composition of the reward function with the state evolution!

- \( r_t \) → \( r_{t+1} \) → \( r_{t+2} \)

Estimation

Kernel-regularized formulation of Koopman operator

Consider the problem of estimating the Koopman operator using a fixed dictionary of functions \( F_D \):

\[
\psi_+(\cdot) = U[\psi(\cdot)] = \pi_{F_D}[U[\psi(\cdot)]] + \text{res}(\cdot)
\]

from the measurement:

\[
s'_i = f(s_i) + \omega_i \quad \rightarrow \quad \psi(s'_i) = \begin{cases} 
\psi(f(s_i)) + \varepsilon \\
\psi_+(s_i) + \varepsilon
\end{cases}
\]

Then if \( \Phi(\cdot) \) is a row vector of basis functions for \( F_D \):

\[
\Phi(s'_i)\alpha = \Phi(s_i)\beta + \varepsilon
\]

The regularized solution using the RKHS defined by \( k(\cdot, \cdot) \) as dictionary is:

\[
U^{(r,k)} = \left[k(s_i, s) + \sigma^2\right]^{-1}k(s_i, \beta)
\]

So that:

\[
r_j(\cdot) = k(\cdot, s)\alpha \quad \rightarrow \quad r_{j+1}(\cdot) = k(\cdot, s)\beta = k(\cdot, s)U\alpha
\]

The formulation above is data-driven and has improved performances with respect to the fixed dictionary approximation.

Performance over 100 trials on different noise scenarios

- \( F \) – fixed dictionary case
- \( G \) – Gaussian kernel

Control

Main idea

Use the ability to predict future rewards in order to estimate the value-function for RL problem!

Main assumption:

Policy \( \pi \) is parametrized by \( \theta \) so that:

\[
a_t \sim \pi_\theta(s_t)
\]

- Parameter dependent Koopman operator: if \( \theta \) is fixed, the overall system can be seen as autonomous as every quantity depends on \( s_t \)
- Convenient expression for the value function:

\[
V^\theta(s) = \sum_{t=1}^{T+1} y^{t-1} U^t[r_\alpha(s)]
\]

Policy gradient techniques are allowed:

Gradient of the value-function w.r.t. \( \theta \):

\[
\nabla_\theta[V^\theta(s)] = \nabla_\theta k(x, x) \sum_{t=1}^{T+1} y^{t-1} r_t, \quad x = [s^T \theta^T]^T
\]

Policy update:

\[
\theta_{t+1} = \theta_t + \eta \nabla_\theta V^\theta(s_t)
\]