FACULTY OF SCIENCES, TETOUAN
Morocco

Understanding Reinforcement Learning with Gaussian Processes: A Variational Free Energy Perspective

Mehdi Attaoui

Master Artificial Intelligence and Machine Learning

Module: Reinforcement Learning

February 23, 2026

Contents

1 Introduction

Reinforcement Learning (RL) has become a fundamental framework for solving sequential decision-making problems, especially in domains such as robotics, control systems, and artificial intelligence. In RL, an agent interacts with an environment and learns a policy by trial and error in order to maximize a cumulative reward. Classical reinforcement learning methods have shown strong performance in small or discrete environments; however, their direct application becomes difficult when the state or action spaces are continuous or high-dimensional.

One major challenge in such settings is the need for function approximation. Instead of storing value functions or transition models in a tabular form, the agent must rely on approximators that are able to generalize from limited experience. Many existing approaches use parametric models such as neural networks, which are powerful but often lack explicit uncertainty estimation. This limitation can be problematic in reinforcement learning, where exploration and decision-making under uncertainty play a central role.

Gaussian Process (GP) regression offers an appealing alternative, as it provides a probabilistic framework that naturally models uncertainty in function approximation. By representing unknown functions as distributions, Gaussian processes allow the agent to reason not only about predictions but also about their confidence. For this reason, Gaussian processes have been increasingly studied in reinforcement learning, particularly for modeling system dynamics or value functions.

Despite these advantages, standard Gaussian process regression suffers from significant computational limitations. Its complexity grows cubically with the number of observed data points, which makes it unsuitable for long-term or online reinforcement learning scenarios. Updating the GP model as new experience is collected becomes computationally expensive and memory-intensive, limiting its practical applicability.

The paper by Kameda and Tanaka addresses these challenges by proposing a reinforcement learning framework that combines Gaussian process regression with variational free energy. The main idea of the paper is to reformulate Gaussian process inference using variational methods, enabling efficient online and mini-batch learning. By doing so, the authors aim to preserve the uncertainty-aware nature of Gaussian processes while significantly reducing the computational cost associated with standard GP-based reinforcement learning methods.

This report analyzes the motivation, mathematical formulation, and main contributions of the method proposed by Kameda and Tanaka. In particular, we focus on how variational free energy and sparse Gaussian process approximations are used to construct an efficient and scalable reinforcement learning algorithm.

2 Problem Statement and Motivation

In classical reinforcement learning, the value function is often represented using a table or a matrix that stores the value of each state or state–action pair. This approach works well when the number of states and actions is small. However, as the environment becomes larger or continuous, the size of this matrix increases rapidly, which leads to high memory usage and slow learning.

To overcome this limitation, function approximation methods are commonly used to represent the value function instead of storing it explicitly. Among these methods, Gaussian Process regression has been proposed as a flexible and powerful way to approximate value functions. It allows smooth function estimation and provides uncertainty information, which can be useful in reinforcement learning.

However, existing reinforcement learning methods based on Gaussian Processes are often difficult to use in practice. They usually involve high computational cost and complex online update algorithms, which makes them hard to implement and inefficient for large datasets.

The paper analyzed in this report suggests addressing this problem by using Gaussian Process regression combined with the Variational Free Energy method. This paper proposes a simpler and more efficient way to approximate the value function and integrate it into a Q-learning framework, while allowing online and mini-batch learning with reduced computational cost.

3 Background and Theoretical Foundations

This section provides the necessary background on Reinforcement Learning (RL) and Gaussian Process (GP) regression, which will prepare the reader for understanding the variational free energy approach introduced later.

3.1 Reinforcement Learning Background

Reinforcement Learning (RL) trains an agent to make decisions by interacting with an environment, receiving rewards, and learning which actions are best.

We model the environment as a Markov Decision Process (MDP): $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where

- $\mathcal{S}$: states, $\mathcal{A}$: actions
- P: transition probabilities, R: rewards
- γ: discount factor for future rewards

The Q-function measures expected total reward when starting in state s, taking action a, and following policy π:

$$Q^{\pi}(s,a) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\Big|\, s_0 = s, a_0 = a, \pi\Big]$$

Q-learning updates Q-values from experience:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)\Big]$$

Here, the term in brackets is the **temporal difference (TD) error**, showing how wrong our estimate was. Over time, this lets the agent learn the optimal Q-values without knowing the environment's rules.

3.2 Function Approximation in Reinforcement Learning

In RL, the Q-function can be very large or even infinite if the state or action space is big or continuous. Storing every possible Q-value in a table is impossible. That's where **function approximation** comes in: instead of storing every value, we use a function to estimate Q-values.

We can write:

$$Q(s,a) \approx \hat{Q}(s,a;\theta)$$

where $\hat{Q}$ is a function with parameters θ (like weights in a neural network, or coefficients in a linear model).

Example: Imagine a robot learning to move in a 2D grid. The robot's position is continuous (like (x, y) coordinates). A table can't store Q-values for every possible position. Instead, we use a function $\hat{Q}(x, y, a)$ to predict the value of each action at any position.

Common function approximators:

- Linear models: simple, fast, but may not capture complex patterns
- Neural networks: can approximate very complex functions, but may overfit or need lots of data
- Gaussian Processes: Bayesian, give uncertainty, avoid overfitting

Function approximation allows RL to scale to large or continuous environments, but introduces challenges, like stability and computational cost, especially for non-linear approximators.

3.3 Gaussian Process Regression

Gaussian Processes (GPs) are a special type of function approximator that work well for RL because they can **predict Q-values and their uncertainty** at the same time. This is useful for exploration: the agent can try actions where it is unsure, not just where the value looks high.

Using a GP, we can write the Q-function approximation as:

$$Q(s,a) \sim \mathcal{GP}(m(s,a), k((s,a),(s',a')))$$

- $m(s,a)$: mean function (usually zero) - $k((s,a),(s',a'))$: kernel measuring similarity between state-action pairs

When the agent observes a new transition (s,a,r,s'), the GP updates its prediction. The **predictive mean** $\mu(s,a)$ gives the estimated Q-value, and the **predictive variance** $\sigma^2(s,a)$ tells us how uncertain it is:

$$Q(s,a) \approx \mu(s,a), \quad \text{uncertainty} = \sigma^2(s,a)$$

example: Suppose our robot in a 2D grid hasn't tried action "move up" in the top-right corner. The GP predicts a Q-value with **high uncertainty**. The agent can choose this action to explore safely, rather than blindly following what it already knows.

Advantages of GPs in RL:

- Bayesian: naturally handles uncertainty
- Non-parametric: can adapt to new data without a fixed model size
- Avoid overfitting: predictions balance observed data and smoothness

The main challenge: updating GPs is **computationally expensive**, especially with many observations, which motivates methods like **variational free energy** to make online learning practical.

4 Variational Inference and Variational Free Energy

4.1 Bayesian Inference in Gaussian Processes

In Gaussian Process (GP) regression, Bayesian inference provides a principled way to update our beliefs about the function given new data. Given observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ and assuming Gaussian noise σ_n^2, the posterior distribution over function values at new points x_* is

$$p(f_* \mid \mathcal{D}) = \frac{p(\mathbf{y} \mid f) p(f_*)}{p(\mathbf{y})} \,,$$

where f_* denotes the function values at x_*. The predictive mean and variance are:

$$\mu_* = k_*^\top (K + \sigma_n^2 I)^{-1} \mathbf{y}, \quad \sigma_*^2 = k(x_*, x_*) - k_*^\top (K + \sigma_n^2 I)^{-1} k_* \,.$$

Exact Bayesian inference is often intractable in GP-based RL when the dataset grows, due to the cubic cost of inverting $K + \sigma_n^2 I$.

4.2 Variational Approximation

To address the computational cost, variational inference approximates the true posterior $p(f \mid \mathcal{D})$ with a simpler distribution $q(f)$ by minimizing the Kullback-Leibler (KL) divergence:

$$q^*(f) = \arg\min_q \, \mathrm{KL}[q(f) \,||\, p(f \mid \mathcal{D})] \,.$$

The approximation $q(f)$ is chosen to be tractable, often Gaussian, allowing efficient updates and predictions while maintaining uncertainty estimates.

4.3 Variational Free Energy

Variational Free Energy (VFE) is the objective function minimized in variational inference. It is defined as

$$\mathcal{F} = \mathrm{KL}[q(f) \,||\, p(f)] - \mathbb{E}_{q(f)}[\log p(\mathbf{y} \mid f)] \,,$$

which can be interpreted as a balance between:

- **Complexity**: how far the approximate posterior $q(f)$ is from the prior $p(f)$
- **Accuracy**: how well $q(f)$ explains the observed data

Minimizing $\mathcal{F}$ provides a tractable way to approximate the GP posterior for reinforcement learning, enabling **online updates** and **mini-batch learning** without expensive matrix inversions.

5 Sparse Gaussian Processes and Inducing Points

Gaussian Processes (GPs) are powerful but computationally expensive for large datasets due to the $\mathcal{O}(n^3)$ cost of inverting the kernel matrix. Sparse Gaussian Processes address this by introducing a small set of *inducing points* to summarize the dataset, reducing computation while retaining accuracy.

5.1 Inducing Variables and Inducing Points

Inducing points are a set of $m \ll n$ pseudo-inputs, denoted as $\mathbf{Z} = \{z_j\}_{j=1}^m$, with corresponding function values $\mathbf{u} = f(\mathbf{Z})$. They act as a compact representation of the full function. The joint distribution of observed function values $\mathbf{f}$ and inducing variables $\mathbf{u}$ is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{u} \end{bmatrix} \sim \mathcal{N}\Big(0, \begin{bmatrix} K_{nn} & K_{nm} \\ K_{mn} & K_{mm} \end{bmatrix}\Big),$$

where:

- K_{nn}: kernel matrix over training points
- K_{mm}: kernel matrix over inducing points
- K_{nm}, K_{mn}: cross-covariances between training and inducing points

This formulation allows the GP to be approximated using only the $m \times m$ matrix K_{mm} instead of the full $n \times n$ matrix.

5.2 Sparse Gaussian Process Approximation

Using variational inference, the posterior over $\mathbf{f}$ is approximated via the inducing variables $\mathbf{u}$:

$$q(\mathbf{f}) = \int p(\mathbf{f} \mid \mathbf{u}) q(\mathbf{u})\, d\mathbf{u},$$

where $q(\mathbf{u})$ is a variational distribution, often chosen Gaussian. The predictive mean and variance for a new input x_* become:

$$\mu_* = k_{*m} K_{mm}^{-1} \mathbf{m}, \quad \sigma_*^2 = k_{**} + k_{*m} K_{mm}^{-1} (\mathbf{S} - K_{mm}) K_{mm}^{-1} k_{m*},$$

with $\mathbf{m}$ and S being the mean and covariance of $q(\mathbf{u})$. This reduces computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(nm^2)$.

5.3 Mini-batch and Online Learning

Sparse GPs also enable **mini-batch** and **online learning**, crucial for RL with streaming data. Instead of using all n observations, we update the variational parameters $\mathbf{m}$ and $\mathbf{S}$ using a small batch of data $\mathcal{B} \subset \mathcal{D}$:

$$\mathcal{F}_{\mathcal{B}} = \mathrm{KL}[q(\mathbf{u}) \,||\, p(\mathbf{u})] - \sum_{(x_i, y_i) \in \mathcal{B}} \mathbb{E}_{q(f_i)}[\log p(y_i \mid f_i)].$$

This allows scalable updates, making GP-based RL feasible in large or continuous environments while maintaining uncertainty estimates and predictive performance.

6 Method Proposed by Kameda and Tanaka

6.1 Problem Formulation

The goal of the proposed method is to efficiently learn the optimal Q-function in a model-free reinforcement learning setting. Let the Q-function be approximated by a Gaussian Process (GP):

$$Q(s, a) \sim \mathcal{GP}(m(s, a), k((s, a), (s', a')))$$

The challenge is to update this GP online as new transitions (s_t, a_t, r_t, s_{t+1}) are observed, while keeping computation tractable in large or continuous state-action spaces. The method leverages **variational free energy** and **sparse GP approximation** to achieve this.

6.2 Gaussian Process Regression Using Variational Free Energy

The Q-function posterior is approximated using a set of inducing points $\mathbf{Z} = \{z_j\}_{j=1}^m$ with function values $\mathbf{u} = Q(\mathbf{Z})$. Using a variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$, the predictive distribution at a new state-action pair (s, a) is:

$$Q(s, a) \approx \mu(s, a) = k_{*m} K_{mm}^{-1} \mathbf{m}, \quad \sigma^2(s, a) = k_{**} + k_{*m} K_{mm}^{-1} (\mathbf{S} - K_{mm}) K_{mm}^{-1} k_{m*}.$$

The variational free energy (VFE) is minimized to find $\mathbf{m}$ and $\mathbf{S}$:

$$\mathcal{F} = \mathrm{KL}[q(\mathbf{u}) \,||\, p(\mathbf{u})] - \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})].$$

Here, $\mathbf{y}$ are the target Q-values obtained from temporal difference (TD) updates:

$$y_t = r_t + \gamma \max_{a'} \mu(s_{t+1}, a').$$

This allows the GP to incorporate new RL data efficiently while maintaining uncertainty estimates.

6.3 Variational Updates for Reinforcement Learning

The variational parameters $\mathbf{m}$ and $\mathbf{S}$ are updated incrementally as new transitions arrive. For a mini-batch $\mathcal{B}$ of transitions, the updates are obtained by minimizing the mini-batch free energy:

$$\mathcal{F}_{\mathcal{B}} = \mathrm{KL}[q(\mathbf{u}) \,||\, p(\mathbf{u})] - \sum_{(s_t, a_t, y_t) \in \mathcal{B}} \mathbb{E}_{q(f_t)}[\log p(y_t \mid f_t)].$$

This leads to efficient **online learning** of the Q-function:

- Each transition contributes to the update without requiring inversion of the full kernel matrix
- The predictive mean $\mu(s, a)$ is used as the Q-value in action selection
- The predictive variance $\sigma^2(s, a)$ can be used for exploration

By combining **sparse GP approximation** with **variational free energy**, the method achieves tractable, online, and scalable GP-based reinforcement learning.

7 Algorithm Description

The proposed algorithm combines **Q-learning** with **Gaussian Process regression** using **variational free energy (VFE)**, and processes data in **mini-batches** to be computationally efficient.

The algorithm works as follows:

1. At the start, the variational parameters of the GP are initialized using the offline VFE formulas. This gives the first estimates of the Q-function.
2. At each time step, the agent is in a state s_t and chooses an action a_t using an **ϵ-greedy policy**. This means it usually chooses the action with the highest predicted Q-value, but sometimes explores a random action.

3. After taking the action, the agent observes the reward r_t and the next state s_{t+1}. Using this information, the **target Q-value** is calculated:

$$y_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a')$$

4. The state-action pair (s_t, a_t) is combined into an input vector $x_t = \langle s_t, a_t \rangle$, and the pair (x_t, y_t) is added to a mini-batch dataset.

5. Once the mini-batch reaches the specified size N_{batch}, the algorithm updates the **variational parameters** of the GP (mean and covariance) using the mini-batch data. This is done using the VFE formulas, which allow the GP to efficiently incorporate multiple data points at once.

6. After updating, the mini-batch is cleared, and the process continues for the next time steps.

By using **mini-batches** instead of updating after every single transition, the algorithm reduces the computational burden of GP updates while keeping good accuracy. The combination of **Q-learning targets** and **GP uncertainty estimates** allows the agent to learn efficiently and explore the environment safely.

8 Numerical Experiments and Implementation

To test the proposed GP-based reinforcement learning method, we implemented a simple two-dimensional grid environment. The grid is 5×5, and the agent starts in the bottom-left corner at $(1, 1)$. The goal is located in the top-right corner at $(5, 5)$, where the agent receives a reward of 1. The agent can move in four directions: up, down, left, and right. To make the problem more realistic, movement is noisy: with a probability of 0.1, the agent stays in its current state instead of moving.

We compared three methods:

- The proposed **sparse GP with variational free energy** method
- **GPQ**, a standard GP-based Q-learning method
- **Tabular Q-learning**, a classic baseline

All methods used the **ϵ-greedy policy** for exploration, with $\epsilon = 0.3$, and the discount factor was set to $\gamma = 0.99$. For Q-learning, the learning rate was $\alpha = 0.1 \sim 0.5$. The GP methods used an RBF kernel:

$$k(x, x') = \exp\Big(-\frac{1}{2}\|x - x'\|^2\Big), \quad \sigma^2 = 1$$

and the proposed method used **36 inducing points** arranged in a grid pattern. Mini-batches of size $N_{\text{batch}} = 16$ were used to update the variational parameters.

The results show that all three methods were able to learn the optimal path to the goal. The proposed sparse GP method produced **value estimates very similar to GPQ**, indicating that it learned the Q-function accurately. Importantly, the proposed method was **faster to compute**, because it only requires updating two main equations per batch, compared to GPQ which involves more complex updates.

From these experiments, we can conclude several points:

- The sparse GP with variational free energy can **learn effectively in small discrete environments**, producing value functions comparable to existing GP-based methods.
- Using inducing points and mini-batches makes the method **computationally efficient**, reducing the time required to learn the Q-function.
- A limitation is that the **placement of inducing points matters**: if the points do not cover the state-action space well, the Q-function estimate may be less accurate. Increasing the number of inducing points improves coverage but increases computation.

Overall, this experiment demonstrates that the proposed method strikes a **balance between learning accuracy and computational efficiency**, making it suitable for online reinforcement learning in environments where full GP regression would be too expensive.

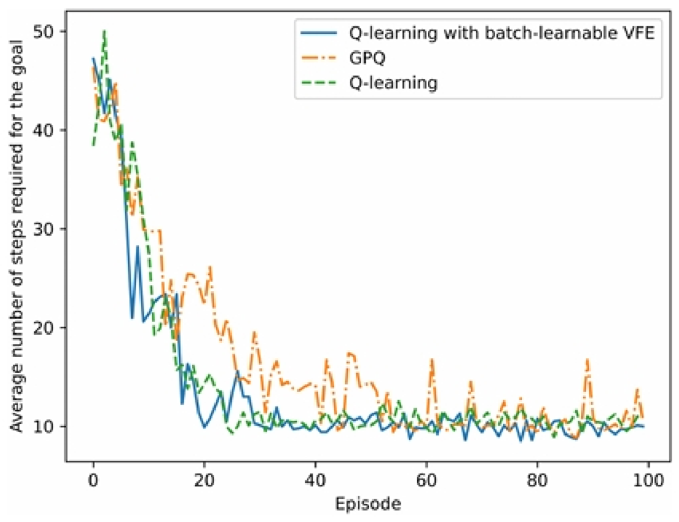

Figure 1: Figure 1: Average number of steps required to reach the goal over ten independent trials.

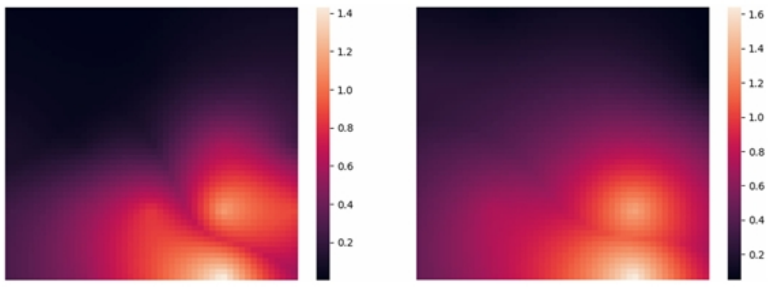

Figure 2: Figure 2: The resulting value estimates of the proposed method (left) and the resulting value estimates of GPQ (right) after 100 episodes. The goal is located at the bottom right.

9 Discussion

The numerical experiments show that the proposed sparse GP with variational free energy method can **learn accurate Q-functions** while being computationally efficient. By using inducing points and mini-batch updates, the algorithm avoids the heavy matrix inversions of standard GP methods. Compared with GPQ and tabular Q-learning, the method achieves similar performance in learning the optimal behavior in the grid environment.

An important observation is that the **uncertainty estimates provided by the GP** allow the agent to explore effectively. This means the agent can balance trying new actions and exploiting known good actions, which is crucial in reinforcement learning. Overall, the experiments confirm that **variational free energy provides a practical way to perform online GP regression** in RL without losing accuracy.

10 Limitations and Perspectives

Despite its advantages, the method has some limitations:

- The **placement and number of inducing points** significantly affect performance. Poorly placed points can reduce the accuracy of the Q-function approximation.
- The current experiments are on **small, discrete environments**. Applying the method to high-dimensional or more complex tasks may require careful selection of inducing points or additional approximations.
- While mini-batch learning improves computational efficiency, **scaling to very large datasets** may still require further optimizations.

Future perspectives include:

- Automatically selecting or adapting inducing points during learning to better cover the state-action space.
- Extending the method to **continuous control tasks** with higher-dimensional states and actions.
- Combining GP-based uncertainty estimates with **modern exploration strategies** to further improve learning efficiency.

11 Conclusion

In this report, we presented the reinforcement learning method proposed by Kameda and Tanaka, which combines **Gaussian Process regression** with **variational free energy** to approximate the Q-function. We explained how sparse GP and inducing points allow for **efficient online learning**, and how variational free energy provides a **tractable approximation of the posterior**.

The experiments in a simple 2D grid environment show that the method can learn optimal policies **accurately and efficiently**, while requiring fewer computations than standard GP methods. Limitations such as the choice of inducing points suggest directions for future work, but overall, the method provides a promising framework for **scalable, Bayesian reinforcement learning**.

References

[1] K. Kameda and F. Tanaka, *Reinforcement Learning with Gaussian Process Regression Using Variational Free Energy*, Journal of Intelligent Systems, 2023.