REINFORCE vs Actor-Critic

Both are policy gradient algorithms

**REINFORCE**

It optimizes the expected return by following the gradient of the log-policy scaled by the returns obtained from full episodes.

Algorithm summary:

-Collect full episodes using the correct policy (Monte Carlo)

-Compute return

-Update parameters with

-Update parameters with baseline

The baseline is used to reduce the variance, it can be either a static value (b=20) or estimated as the value function (like actor-critic).

The loss is computed by adding an entropy coefficient which encourages exploration.

Characteristics:

-Simple and unbiased gradient estimate

-No value function required

-High variance due to full-episode returns

-Inefficient sample usage (one update per episode)

-Convergence can be very slow

**Actor-Critic**

It combines REINFORCE and value-based methods. It uses two models:

Actor: learns the policy

Critic: learns a values function to estimate expected returns

It uses temporal-difference learning to estimate the advantage

In the updated version the advantages are computed with General Advantage Estimation (GAE) to balance bias-variance via λ.

Update rule

To stabilize learning multiple updates are used in the critc

Characteristics:

-Lower variance due to baseline V(s)

-Online updates possible

-Faster and more stable learning

-Biased gradient estimate due to bootstrapping

-Additional model must be trained (extra complexity)

-Sensitive to the quality of the value estimate

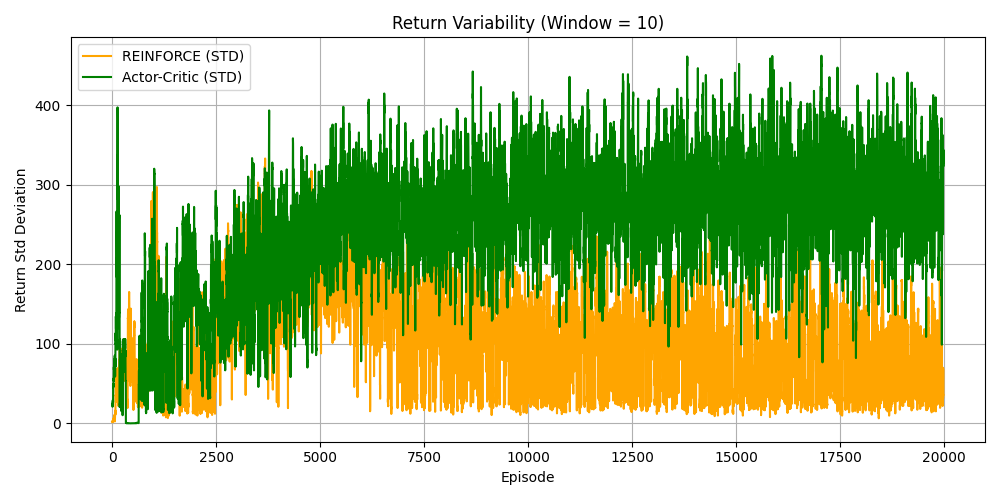
**Environment**

Observation space: continuous state vector Box([-inf -inf -inf -inf -inf -inf -inf -inf -inf -inf -inf], [inf inf inf inf inf inf inf inf inf inf inf], (11,), float64)

Action space: continuous joint torques Box([-1. -1. -1.], [1. 1. 1.], (3,), float32)

**Results** (20k stepsImmagine che contiene testo, Diagramma, diagramma, linea

Il contenuto generato dall'IA potrebbe non essere corretto.)



PPO

The previous methods have some problems, the updated policy can deviate too far from the old policy leading to poor performance or divergence.

This was solved by Trust Region Policy Optimization (TRPO) but it is too complex to implement.

PPO simplifies TRPO by introducing a clipped objective to control policy updates without complex constraints.

It uses:

The probability ratio of new vs old policy

The advantage estimate at time t

Clipped surrogate objective

This means that if the policy changes too much the update is clipped preventing over-optimization, it also ensures stable and conservative updates.

PPO can also train for multiple epochs using mini-batches improving sample efficiency

**Results** (1M steps)

No normalization (Source)

Immagine che contiene testo, Diagramma, diagramma, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

With normalization (Source)

Immagine che contiene testo, Diagramma, linea, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

With normalization (Target)

Immagine che contiene testo, Diagramma, diagramma, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

Test on source env (no norm)

Immagine che contiene testo, schermata, diagramma, Diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Test on target env (no norm)

Immagine che contiene testo, schermata, diagramma, Diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Test on source env (with norm)

Immagine che contiene testo, schermata, diagramma, Diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Test on target env (with norm)

Immagine che contiene testo, schermata, diagramma, Diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

SAC

It combines off-policy learning, entropy regularization, stochastic actor and twin q-learning.

SAC maximizes a modified objective that encourages exploration

Where is the entropy of the policy

And is the temperature parameter for controlling trade-off between reward and entropy

So, SAC tries to maximize both reward and entropy leading to stochastic policies that explore more broadly.

Components of SAC:

Soft Q-Function learns the expected return plus future entropy

The actor maximizes the soft value

SAC uses two Q-functions Q1 and Q2 and uses

Main steps of the algorithm:

Initialize:

-Replay buffer D

-Actor network

-Two Q-networks

-Target Q-networks

For each step

-Sample action

-Interact with environment, store in D

Sample mini-batch from D

Update Q-functions

-Compute target

-Minimize MSE loss

Update policy (actor)

Update temperature

Update target networks

SAC is particularly good when the action space is continuous, in sim-to-real scenarios and when high final performance and stability are required.