Reinforcement learning with automatic basis construction based on isometric feature mapping

Z. Huang et al.

Journal: Information Sciences
IF: 4.305

Background

- Value function approximation has always been a big issue in RL
- In small problems **tabular representation** is enough. But, as the problem grows bigger, representing the $V(s), Q(s,a), P(s,a), R(s)$ becomes drastically hard
- We are looking for some ways to represent $Q(s,a)$ (or $V(s)$) in a more compressed format

$$Q_{|\mathcal{S}||\mathcal{M}|*1}(s,a) = \Phi w = \sum_{i=1}^{l} \phi_i(s,a) w_i$$

Background

- Finding good basis functions is crucial to the success of our solution
- Some options:
  - Linear
  - Polynomials
  - Splines
  - RBFs
  - Fourier transforms
  - Laplacian analysis
- They need to be engineered!
- Can we automate the process?

Problem Definition

- Big MDPs, impossible to represent MDP with P and R in a regular way
- We have: a set of samples (trials or episodes)
- We want:

$$\vec{\phi} = [\phi_1(x), \phi_2(x), ..., \phi_l(x)]$$

  where

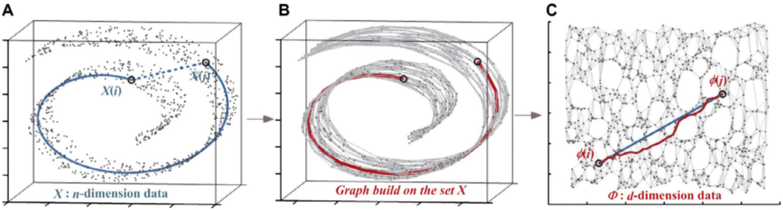
$$\tilde{V}(\mathcal{X}) = \vec{\phi}^T(\mathcal{X})\vec{W}$$

$$\mathcal{X} = \{x_1, x_2, ..., x_n\}$$

- This process is called *Learning Basis Functions*
- Given ϕs we can solve our MDP using standard solutions (LSPI, ALP,...)

Characteristics of the Solution

- Mapping X to Φ
- Isometric feature mapping (IFM)
- Preserves geodesic distance[1] in the neighborhood graph for input and output



[1] the distance between two vertices in a graph or the number of edges in a shortest path

Basis Learning Process

Three steps:

1. Determine neighbor points (K-nearest neighbors is used)
2. Geodesic distances ($d_{\mathcal{X}}(i, j)$) are estimated using shortest path
3. A multidimensional scaling method is applied to analyze the graph distance matrix

IFM Algorithm

Algorithm 1. *IFM* (K, ℓ, X)

\\\\ K: the number of the nearest neighbors of each state point;
\\\\ ℓ: the dimension of the basis function;
\\\\ X: the collected samples $\{x_i, i = 1, 2, \ldots, n\}$;
\\\\ *Knneighbor*: function to search the K nearest neighbors of the state x_i;
\\\\ *Eigen*: function to compute the bottom d eigenvectors of the matrix $\tau(D)$.

Initialize: $D = \{d(i,j)\}_{n\times n} \leftarrow \infty$
for $i = 1, 2, \ldots, n$ **do**
    $\{x_{i_1}, x_{i_2}, \ldots, x_{i_K}\} \leftarrow Knneighbor(K, x_i)$ \\\\ find K nearest neighbors of x_i.
    for $m = 1, 2, \ldots, K$ **do**
        $d(i, i_m) = \|x_i - x_{i_m}\|_2$
    end for
end for
for $m = 1, 2, \ldots, n$ **do**
    $d(i,j) \leftarrow \min\{d(i,j), d(i,m) + d(m,j)\}$
end for
$\tau(D) = -HSH/2$;
$\Phi \leftarrow Eigen(\tau(D), d)$;
return Φ

---

[1] $S_{ij} = D_{ij}^2$ and $H_{ij} = \delta_{ij} - 1/N$
[2] KNN has $\mathcal{O}(Knl)$ time complexity

IFM-API

- Sample Collection: Only a subset of samples are used to construct basis functions
  - × Trajectory-based Sub-sampling
  - ✓ Clustering-based Sub-sampling
- Basis Learning based on isometric feature mapping
- Obtain basis functions corresponding to the whole state spaces based on the known basis functions in X_s
- At the end, they combined IFM with LSPI

Thank You!