Bayesian Inverse Reinforcement Learning

Ramachandran, D., & Amir, E. (2007, January).
In *IJCAI* (Vol. 7, pp. 2586-2591).

理論談話会2020

5月19日

M2 石井 健太

目次

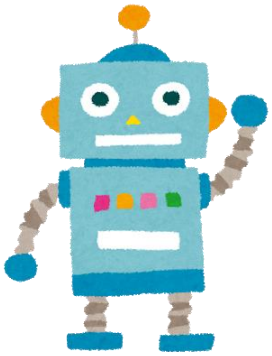
■Reinforcement Learningとは？

逐次的な意思決定ルールを学習する機械学習の一分野

各状態でどの程度報酬がもらえるかを設定し，
どう行動すればいいかを学習する

環境

- **観測**
- **報酬（given）**

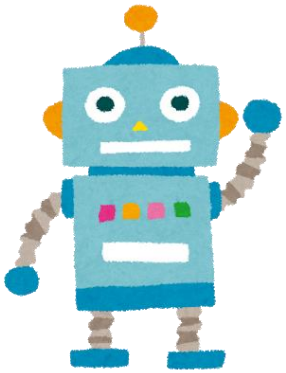


Agent

- **行動（学習対象）**

0. Preparation

■どのように行動すればいいかを学ぶ



Agent

例）報酬がたくさんもらえる行動をとりたい

目先の報酬が大きくても，その先報酬が小さいところばかりだったら困る…

将来まで通して報酬の期待値が大きくなるかどうかの指標を知ることができれば，Agentは報酬を最も多く獲得する可能性が高い行動ができる

価値関数という指標を導入

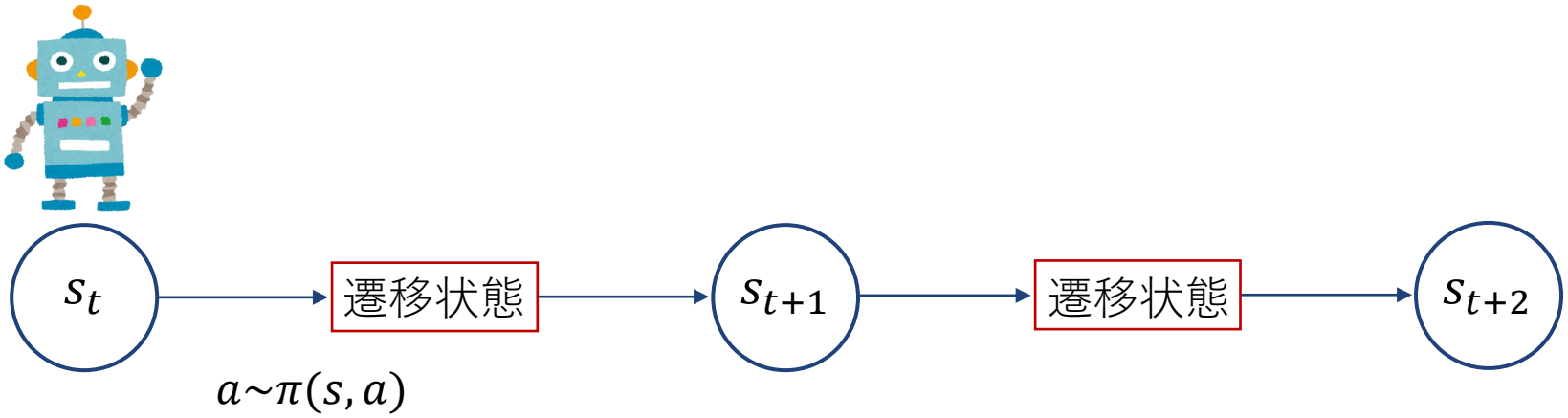
0. Preparation

■価値関数とは？

状態価値関数$V^{\pi}(s)$：状態sにいることの価値を評価する
特定の方策に関して定義される

方策$\pi(s,a)$：どのようなポリシーで行動するか.
(=ある状態sの時に行動aを取る確率)

【マルコフ決定過程】1. πに従って行動を選択



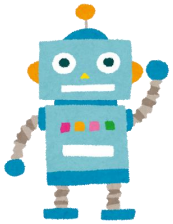
0. Preparation

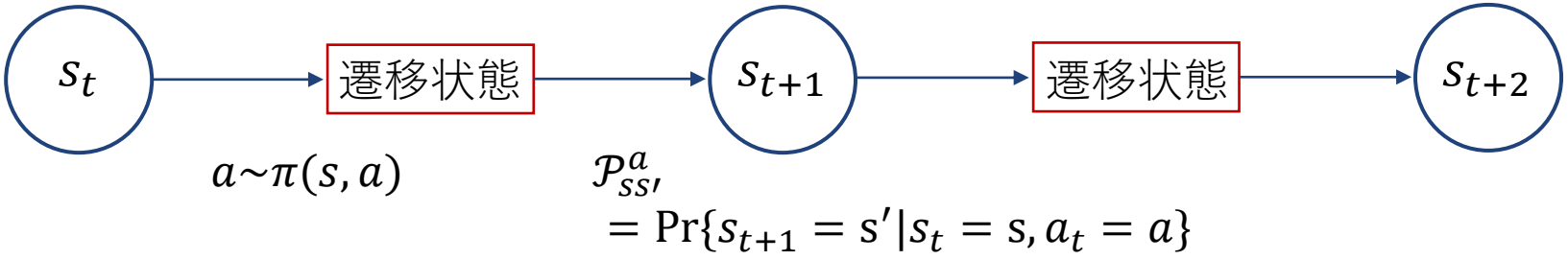
■価値関数とは？

状態価値関数$V^{\pi}(s)$：状態sにいることの価値を評価する
特定の方策に関して定義される

方策$\pi(s,a)$：どのようなポリシーで行動するか.
(=ある状態sの時に行動aを取る確率)

【マルコフ決定過程】2. 環境に影響を受け，確率的に次の状態が決まる





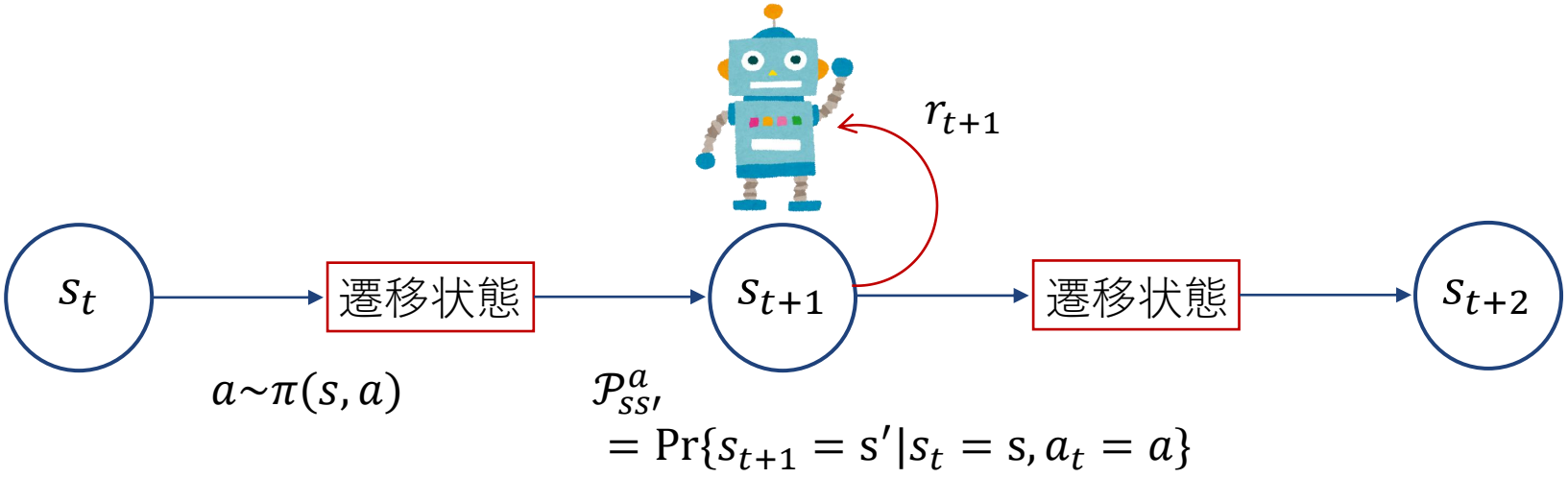
0. Preparation

■価値関数とは？

状態価値関数$V^{\pi}(s)$：状態sにいることの価値を評価する
特定の方策に関して定義される

方策$\pi(s,a)$：どのようなポリシーで行動するか.
(=ある状態sの時に行動aを取る確率)

【マルコフ決定過程】3. 次の状態を観測し，報酬を得る (以降も1から繰り返し)



0. Preparation

$$V^{\pi}(s) = E_{\pi}\left\{\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+1} | s_t = s\right\}$$

$$= E_{\pi}\left\{r_{t+1} + \gamma\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+2} | s_t = s\right\}$$

起こした行動aに対して想定される状態は$\mathcal{P}^{a}_{ss'}$に従って複数想定される

$$= \sum_{a}\pi(s,a)\sum_{s'}\mathcal{P}^{a}_{ss'}\left[\mathcal{R}^{a}_{ss'} + \gamma E_{\pi}\left\{\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+2} | s_{t+1} = s'\right\}\right]$$

再帰的

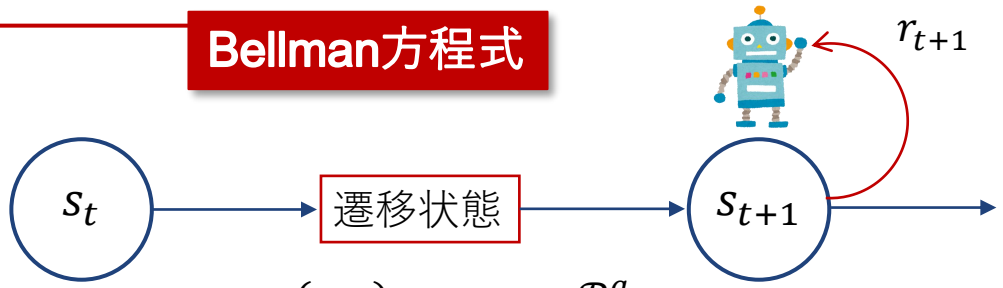
$$= \sum_{a}\pi(s,a)\sum_{s'}\mathcal{P}^{a}_{ss'}[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s')] \qquad (1)$$

Bellman方程式

γ: 時間割引率 $(0 < \gamma \leq 1)$

$\mathcal{R}^{a}_{ss'}$:報酬の期待値

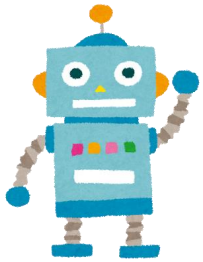
$(= E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\})$



■方策$\pi(s,a)$をどう決める？

1. 数理モデルを用いて直接的に規定するアプローチ
2. 報酬関数に基づいて間接的に方策を決定するアプローチ

⇒ここでは価値関数を用いる2の概要を説明する

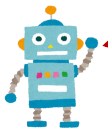


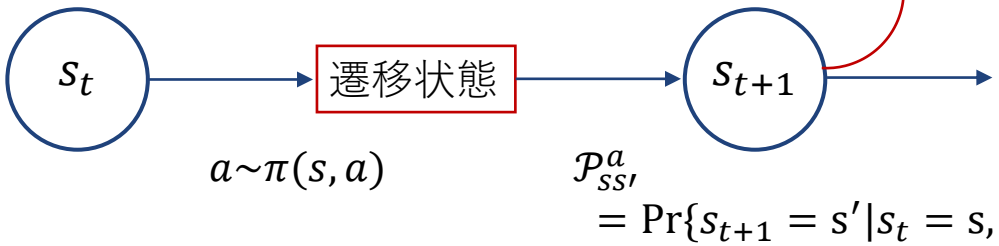
Agent

どの行動をとればよい状態価値関数を持つ状態に移行できるかを知りたい

⇒行動価値関数

$$Q(s,a)=\sum_{s'}\mathcal{P}^{a}_{ss'}[\mathcal{R}^{a}_{ss'}+\gamma V^{\pi}(s')] \quad (2)$$





0. Preparation

■方策$\pi(s,a)$をどう決める？

行動価値関数を基準に決める．下に例を挙げる．

・greedy policy model

行動価値関数最大となる行動をとる

$$\pi_{greedy}(s,a)=\begin{cases}1 & a=\underset{\mathrm{a}}{\operatorname{argmax}}\,Q(s,a)\\ 0 & \text{others}\end{cases} \tag{3}$$

・ε-greedy policy model

探索のため確率方策化

ε ：探索比率を決定するハイパーパラメータ

$|A|$：選択肢数

$$\pi_{\varepsilon-greedy}(s,a)=\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|A|} & a=\underset{\mathrm{a}}{\operatorname{argmax}}\,Q(s,a)\\ \dfrac{\varepsilon}{|A|} & \text{others}\end{cases} \tag{4}$$

0. Preparation

■方策$\pi(s,a)$をどう決める？

行動価値関数を基準に決める．下に例を挙げる．

・softmax policy model

最大値以外の行動価値関数も勘案した確率的方策

$$\pi_{softmax}(s,a) = \frac{exp(\beta Q(s,a))}{\sum_{a' \in A} exp(\beta Q(s,a'))} \quad (5)$$

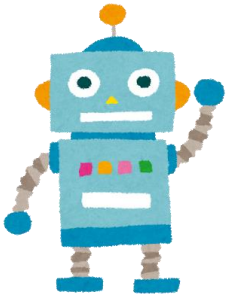
β ：ボルツマン分布の逆温度
(ランダム性を制御するハイパーパラメータ)

行動モデルの式形と同じ
特にRecursive Logit modelと等価になる

ランダム効用最大化理論からも証明できるので，
Fosgerau et al.(2013)を参照

0. Preparation

■ようやく本題



Agent

行動の学習の仕方はわかったけど…
報酬ってどうやって設定するの？

報酬の決め方

- モデラーが恣意的に
- モンテカルロ木探索

etc...

- プロ（Expert）から学ぼう
  ＝ Inverse Reinforcement Learning (IRL)

Expert

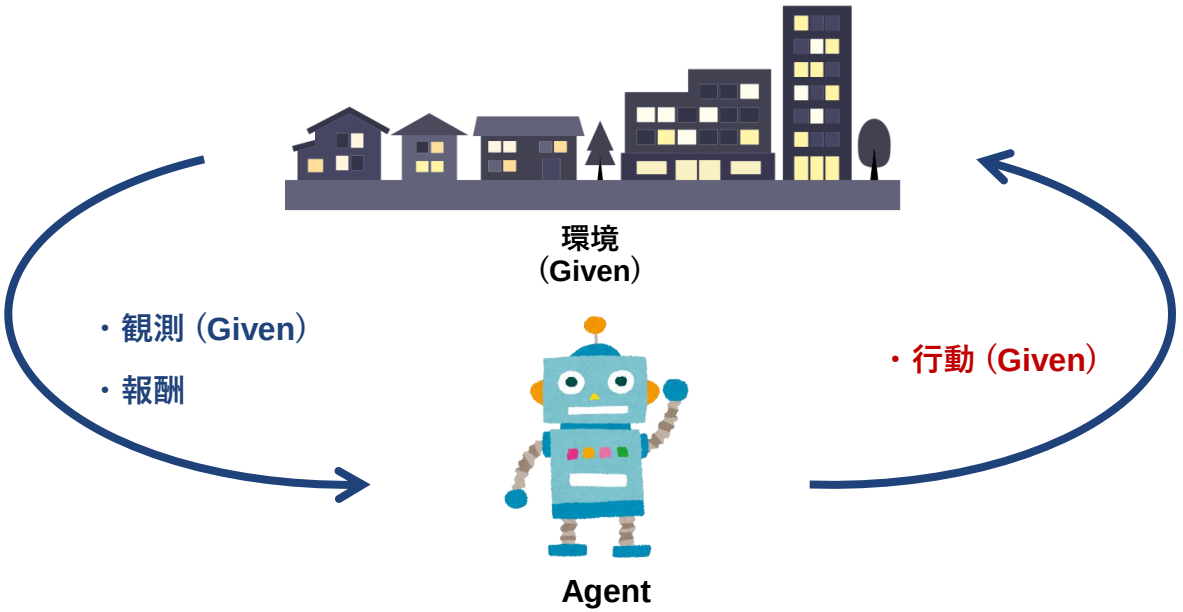
Expertの行動履歴を教師データとし，
報酬を推定する手法

■IRLの定義 (Russell, 1998による)

Agentが最適化する報酬関数を学習すること

Given)

1. Agentの時系列順の行動履歴
2. Agentへの入力 (観測)
3. モデルの環境



1. Introduction

■Task

大きく2つに分けられる

1. 報酬の学習

Agentの行動の選好自体に関心がある場合

各状態で得る報酬を推定する⇒行動モデルはこっち側のアプローチ

2. 徒弟学習 (Atkeson and Shaal,1997 など)

Agentの行動自体に関心がある場合

方策自体を直接推定する

報酬学習のほうが一般的に良い学習法

・環境が変化しても，報酬を学習していれば対応して計算ができる

・最適方策を報酬から決定できる

2. Bayesian IRL

■概要

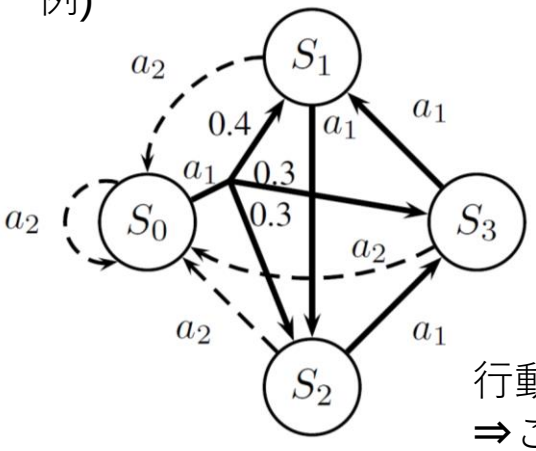
・報酬関数の事後分布をベイジアンアプローチで推定し，MAP解を求める

・事前分布を更新するためにevidenceとしてExpertの行動を用いる

■既往研究の主課題

情報が少ないため，報酬に関して1つの解しか得られない

例)



a_1が最適方策に従う行動だとする (Expertの経路とみなせる)

S1, S2, S3のそれぞれが最も高い報酬である可能性がある

行動選択確率分布に不確実性を入れる必要がある

⇒この問題をmax entropy法で解決した論文(Ziebart et al., 2008)もある

2. Bayesian IRL

■既往研究への優位性

1. 最適方策を完全に特定する必要はない
2. Expertが行動選択を誤らないという仮定が必要ない

⇒方策を確率分布で扱い，報酬も確率分布として推定する

3. 事前分布として他の事前知識を埋め込むことができる
4. 複数のExpertの情報を使用できる

⇒ベイジアンアプローチでevidenceとしてExpertの情報を扱う

2. Bayesian IRL

■Expertに仮定を置く

1. Expertは総獲得報酬の最大化を試みる
   =ε-greedy法などの環境探索は行わない

2. Expertの方策は時間経過に対して定常

つまりAgent χ の行動履歴 $O_{\mathcal{X}} = \{(s_1, a_1), (s_2, a_2) \ldots (s_k, a_k)\}$
の発生確率 $Pr_{\mathcal{X}}$ は，以下のように分解することができる

$$Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R}) = Pr_{\mathcal{X}}((s_1, a_1)|\boldsymbol{R}) Pr_{\mathcal{X}}((s_2, a_2)|\boldsymbol{R}) \ldots Pr_{\mathcal{X}}((s_k, a_k)|\boldsymbol{R}) \quad (6)$$

各状態での行動選択に分解できる

2. Bayesian IRL

■報酬関数の事後分布

evidence　事前分布

$$Pr_{\mathcal{X}}(\boldsymbol{R}|O_{\mathcal{X}}) = \frac{Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R})P_R(\boldsymbol{R})}{Pr(O_{\mathcal{X}})} \quad (7)$$

(∵ ベイズの定理より)

周辺分布 $= \int_{\boldsymbol{R}} Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R})P_R(\boldsymbol{R})$

・尤度

ここではAgentの方策をsoftmax policyに設定する

$$Pr_{\mathcal{X}}((s_i,a_i)|\boldsymbol{R}) = \frac{1}{Z_i}e^{\alpha_{\mathcal{X}}Q^*(s_i,a_i,\boldsymbol{R})} \quad (8)$$

↓ 行動履歴全体に拡大 (独立仮定を用いる)

$$Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R}) = \frac{1}{Z}e^{\alpha_{\mathcal{X}}E(O_{\mathcal{X}},\boldsymbol{R})} \quad (9)$$

where $E(O_{\mathcal{X}},\boldsymbol{R}) = \sum_i Q^*(s_i,a_i,\boldsymbol{R})$　　Z_i, Z は規格化定数

2. Bayesian IRL

・事前分布 (事前情報を反映できる)

evidence　　事前分布

$$\frac{Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R})P_R(\boldsymbol{R})}{Pr(O_{\mathcal{X}})}$$

周辺分布

1. 事前情報がない場合

一様分布を用いる

2. ほとんどの状態で報酬が微小な場合

ガウス分布 or ラプラス分布

$$P_{Gaussian}(\boldsymbol{R}(s)=r)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{r^2}{2\sigma^2}},\forall s\in S \qquad (10)$$

$$P_{Laplace}(\boldsymbol{R}(s)=r)=\frac{1}{2\sigma}e^{-\frac{|r|}{2\sigma}},\forall s\in S \qquad (11)$$

3. 多くの状態で微小 or 負の報酬．かつ目的地だけ正報酬である場合

β分布

$$P_{Beta}(\boldsymbol{R}(s)=r)=\frac{1}{(\frac{r}{R_{max}})^{\frac{1}{2}}(1-\frac{r}{R_{max}})^{\frac{1}{2}}},\forall s\in S \qquad (12)$$

2. Bayesian IRL

・周辺分布 $= \int_{\boldsymbol{R}} Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R})P_R(\boldsymbol{R})$

考慮する$\boldsymbol{R}$空間が広く，かつ連続
⇒計算が困難

evidence　事前分布

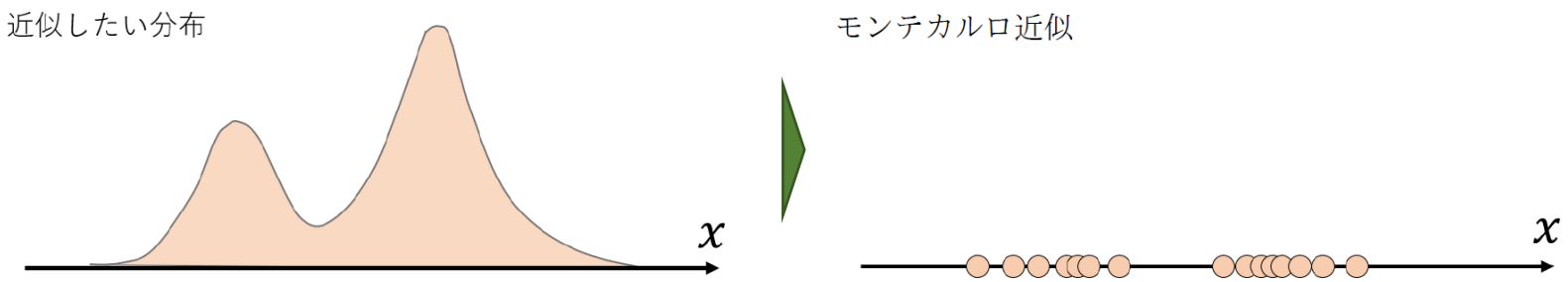
$$\frac{Pr_{\mathcal{X}}(O_{\mathcal{X}}|\boldsymbol{R})P_R(\boldsymbol{R})}{Pr(O_{\mathcal{X}})}$$

周辺分布

加えて分子の分布同士の積は分布形によっては困難

MCMC (Markov Chain Monte Carlo) を用いる

モンテカルロシミュレーションにより事後確率を計算する．
これにより事後分布を粒子近似が可能なので，正規化も容易



3. Inference

■損失関数

1. 報酬の学習の場合

$$L_{linear}(\boldsymbol{R}, \hat{\boldsymbol{R}}) = \| \boldsymbol{R} - \hat{\boldsymbol{R}} \|_1 \quad (13)$$

$$L_{SE}(\boldsymbol{R}, \hat{\boldsymbol{R}}) = \| \boldsymbol{R} - \hat{\boldsymbol{R}} \|_2 \quad (14)$$

$\boldsymbol{R}$が事後分布内から抽出されるのであれば,

・$\hat{\boldsymbol{R}}$ が事後分布の平均値⇒*SE loss*が最小化される[Berger, 1993]

・$\hat{\boldsymbol{R}}$ が事後分布の中央値⇒*linear loss*が最小化される

2.徒弟学習の場合

$$L^p_{policy}(\boldsymbol{R}, \pi) = \| \boldsymbol{V}^*(\boldsymbol{R}) - \boldsymbol{V}^{\pi}(\boldsymbol{R}) \|_p \quad (15)$$

$\boldsymbol{V}^*(\boldsymbol{R})$: 最適状態価値関数

$\boldsymbol{V}^{\pi}(\boldsymbol{R})$: 方策$\boldsymbol{\pi}$での状態価値関数

$\boldsymbol{p}$：ノルム

(15)を最小にする$\boldsymbol{\pi}$は報酬分布の期待値から得られる最適方策に一致

3. Inference

■損失関数

1. 報酬の学習の場合

$$L_{linear}(\boldsymbol{R}, \hat{\boldsymbol{R}}) = \| \boldsymbol{R} - \hat{\boldsymbol{R}} \|_1 \quad (13)$$

$$L_{SE}(\boldsymbol{R}, \hat{\boldsymbol{R}}) = \| \boldsymbol{R} - \hat{\boldsymbol{R}} \|_2 \quad (14)$$

$\boldsymbol{R}$が事後分布内から抽出されるのであれば

つまり報酬関数の事後分布を求めることは
両方の学習に寄与する

2.徒弟学習の場合

$$L^p_{policy}(\boldsymbol{R}, \pi) = \| \boldsymbol{V}^*(\boldsymbol{R}) - \boldsymbol{V}^{\pi}(\boldsymbol{R}) \|_p \quad (15)$$

$\boldsymbol{V}^*(\boldsymbol{R})$: 最適状態価値関数
$\boldsymbol{V}^{\pi}(\boldsymbol{R})$: 方策$\pi$での状態価値関数
p：ノルム

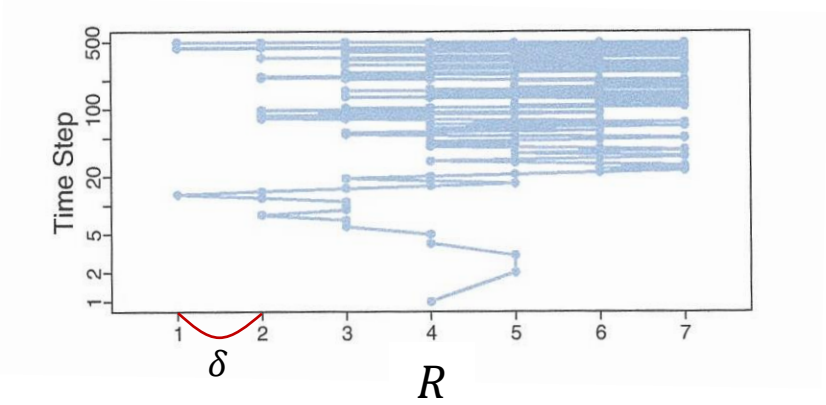
(15)を最小にするπは報酬分布の期待値から得られる最適方策に一致

4. Sampling and Rapid Convergence

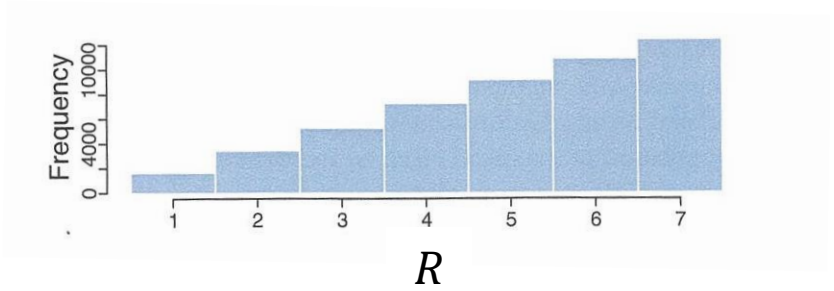
■MCMCの実装

⇒PolicyWalk というアルゴリズムを提案　(GridWalk [Vempala, 2005] の修正版)

➢MCMCのイメージ図



報酬の空間をδで離散化し，
隣接する点へ逐次移動していく



訪問頻度が近似したい分布の形状に

➢手順 (今回は事前分布に一様分布を採用)

1. 報酬ベクトル$\boldsymbol{R} \in \mathbb{R}/\delta$をランダムに取得
2. $\boldsymbol{R}$を基に価値関数を計算し，方策πの初期値とする

4. Sampling and Rapid Convergence

■MCMCの実装

前時期のπを用いることで
効率的に価値関数の探索が可能な手法

➢手順の続き

3. 以下を繰り返す

(a) $\boldsymbol{R}$の隣接点からランダムに報酬ベクトル$\tilde{\boldsymbol{R}}$を取得する

(b) 方策πに従い，行動価値関数$Q^{\pi}(s,a,\tilde{\boldsymbol{R}})$を計算する

(c) if $Q(s,\pi(s),\tilde{\boldsymbol{R}}) < Q(s,a,\tilde{\boldsymbol{R}})$ を満たす(s,a)が存在する

i. $\tilde{\boldsymbol{R}}, \pi$を用いて価値関数を計算し，方策$\tilde{\pi}$を計算する

ii. 確率$\min\left\{1, \frac{P(\tilde{\boldsymbol{R}},\tilde{\pi})}{P(\boldsymbol{R},\pi)}\right\}$ $\boldsymbol{R} := \tilde{\boldsymbol{R}}, \pi := \tilde{\pi}$と更新する

else:

i. 確率$\min\left\{1, \frac{P(\tilde{\boldsymbol{R}},\pi)}{P(\boldsymbol{R},\pi)}\right\}$ $\boldsymbol{R} := \tilde{\boldsymbol{R}}$と更新する

5. Experiments

■設定

N個の状態に対し，i.i.d.ガウス分布に従うよう報酬を生成

上記設定のMDPで2種類のシミュレーションを行い，これをExpertとする

Expert 1)

Q-learningにより方策を学習.
学習率の設定により，最適方策は学習できないが合理的に近い

Expert 2)

k ステップ先までで獲得する期待総報酬を最大化する方策を実行する
(kはhorizon timeより少し先に設定)

5. Experiments

■学習結果 (従来手法[Ng and Russell, 2000] と比較)

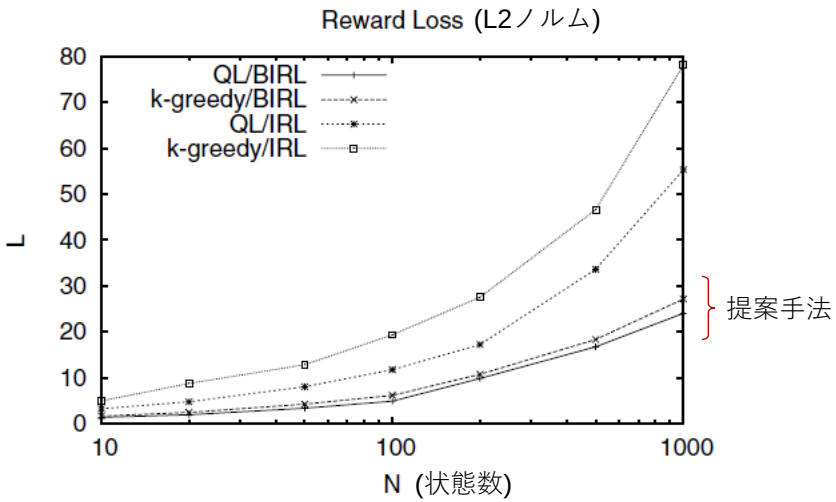


Figure 4: Reward Loss.

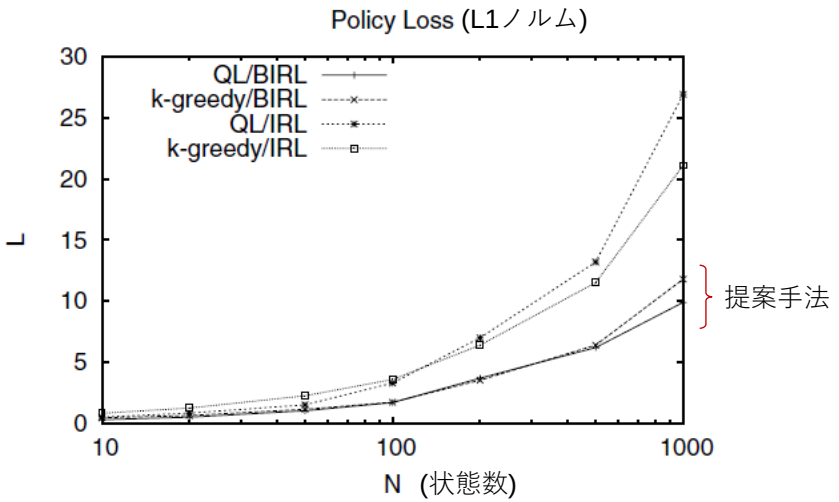


Figure 5: Policy Loss.

- どちらのExpertに対しても従来手法より精度が良い
- BIRL内でもQ-learningを用いるので,
  ExpertもQ-learningで価値関数を求めるもののほうが精度が良い

6. Related Work

■Physical science

物理システムの観測からモデルパラメータを推論する研究などがある [Tarantola, 2005]

■Control theory

カルマンが提起した，2次コストを持つ決定論的線形システムの目的関数を復元するという問題を解決している[Boyed et al., 1994]

7. Conclusions and Future Work

■まとめ

- ベイジアンアプローチからIRLを定式化することで,
  報酬関数や方策の推定精度を改善した

- BIRLの実行可能なアルゴリズムを提案した

■課題

- 背景知識を用いた特定のIRL問題のためにより有用な事前分布の検討

- 学習環境とExpertの環境が異なる場合のIRLの頑健性の検討

Appendix

■PolicyWalk algorithmの効率性

Lemma.

$F(\cdot)$はある正の数dについて$\{x \in \mathbb{R}^n | -d \leq x_i \leq d\}$で定義される正の実数値関数であり，
$\lambda \in [0,1]$, α, βに対して，　$f(x) = \log F(x)$ が以下を満たすとき，

$$|f(x) - f(y)| \leq \alpha \|x - y\|_\infty \qquad (16)$$

$$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda) f(y) - \beta \qquad (17)$$

F 上でのGridWalkを用いたマルコフ連鎖はステップ数$O\left(n^2 d^2 \alpha^2 e^{2\beta} \log \frac{1}{\epsilon}\right)$ で混合する
→ PolicyWalkでも同様

Proof.

[Applegate and Kannan, 1993]を参照

■PolicyWalk algorithmの効率性

$Theorem.$

状態集合S，行動集合A，遷移確率関数T，割引率γ，$|S| = N$で書かれるマルコフ決定過程において，事前分布に一様分布を仮定する．

$C = \{\boldsymbol{R} \in \mathbb{R}^n | -R_{max} \leq R_i \leq R_{max}\}, R_{max} = O(1/N)$ のとき，

$Pr_\chi(\boldsymbol{R}|O_\chi)$ はステップ数$O\left(N^2 \log\frac{1}{\epsilon}\right)$ でサンプリングが可能．(ϵはerror)

■PolicyWalk algorithmの効率性

$Proof.$

$$f(\boldsymbol{R}) = \alpha_\chi E(O_\chi, \boldsymbol{R}) = \alpha_\chi \textstyle\sum_i Q^*(s, a_i, \boldsymbol{R}) \quad (18)$$

$$f_\pi(\boldsymbol{R}) = \alpha_\chi \sum_i Q^\pi(s, a_i, \boldsymbol{R}) \quad (19)$$

f_πは報酬$\boldsymbol{R}$の線形関数であり，　すべての$\boldsymbol{R} \in C$において$f(\boldsymbol{R}) > f_\pi(\boldsymbol{R})$を満たす.

また,

$$\max_{s,a} Q^*(s,a) = \max_{s,a,\pi} Q^\pi(s,a) = \max_{s,a} V^\pi_{max}(s) \leq \frac{R_{max}}{1-\gamma}$$

(公比R_{max}の無限等比級数で抑える)

同様に,

$$\min_{s,a} Q^*(s,a) \geq -\frac{R_{max}}{1-\gamma}$$

よって$f(\boldsymbol{R}), f_\pi(\boldsymbol{R})$は,

$$f(\boldsymbol{R}) \leq \frac{\alpha_\chi N R_{max}}{1-\gamma}, \quad f_\pi(\boldsymbol{R}) \leq -\frac{\alpha_\chi N R_{max}}{1-\gamma}$$

(20)　　(21)

■PolicyWalk algorithmの効率性

$Proof$.続き

$$f_\pi(\boldsymbol{R}) \geq f(\boldsymbol{R}) - \frac{2\alpha_\chi N R_{max}}{1-\gamma} \qquad (22) \qquad (\because \text{式(20), 式(21)の両辺和をとる})$$

すべての$\boldsymbol{R_1}, \boldsymbol{R_2} \in C,\ \lambda \in [0,1]$において

$$\begin{aligned} f(\lambda \boldsymbol{R_1} + (\boldsymbol{1}-\lambda)\boldsymbol{R_2}) &\geq f_\pi\left(\lambda \boldsymbol{R_1} + (1-\lambda)\boldsymbol{R_2}\right) \\ &\geq \lambda f_\pi\left(\boldsymbol{R_1}\right) + (1-\lambda) f_\pi(\boldsymbol{R_2}) && (\because f_\pi \text{は線形関数}) \\ &\geq \lambda f(\boldsymbol{R_1}) + (1-\lambda)\lambda f(\boldsymbol{R_2}) - \frac{2\alpha_\chi N R_{max}}{1-\gamma} && (\because \text{式(22)}) \end{aligned}$$

$Lemma$ の式 (17)におけるβ

よってfは$Lemma$の条件を満たす.

$$\beta = \frac{2\alpha_\chi N R_{max}}{1-\gamma} = 2N \cdot \frac{O\left(\frac{1}{N}\right)}{1-\gamma} = O(1)$$

$$\alpha = \frac{|f(\boldsymbol{R_1}) - f(\boldsymbol{R_2})|}{\|\boldsymbol{R_1} - \boldsymbol{R_2}\|_\infty} = \frac{2\alpha_\chi N R_{max}}{(1-\gamma)O(\frac{1}{N})} = O(N)$$

$Lemma$より

必要サンプル数は$O\left(N^2 \log \frac{1}{\epsilon}\right)$

■Q-learning [森村哲郎，強化学習 (MLPシリーズ)を参考]

以下の特徴を持つ価値関数の計算手法の1つ

- ・オンライン学習
- ・環境のダイナミクスを把握する必要がない
- ・サンプリングを用いて計算可能
- ・前時期と現時期の情報のみで計算可能

行動価値関数を以下のように逐次更新していく

$$Q(s,a) \leftarrow (1-\alpha_t)Q(s,a) + \alpha_t \left\{ r' + \gamma \max_{a'} Q(s',a') \right\}$$

α_t：学習率
s'：次期の状態
a'：次期の行動

■Q-learning

【疑似コード】

$for\ each\ episode$：

$Initialize\ s$

$Repeat$

方策 (e.g.ε-$greedy$) に従い状態sにおいて行動aを選択

行動aを実行し，次期状態s'と報酬r'を観測

$$Q(s,a) \leftarrow (1-\alpha_t)Q(s,a) + \alpha_t\left\{r' + \gamma \max_{a'} Q(s',a')\right\}$$

$s \leftarrow s'$

$until\ s$が終端状態