Research Summary of Chao NI: literature reviews

February 2019

1 Model-free Policy search methods

1.1 policy gradient

1. Policy gradient: compute gradient descent from samples (episode based or step based).

$$D_{ep} = \{\theta_i, R_i\} \qquad or \qquad D_{st} = \{s_t, a_t, Q_t\} \tag{1}$$

The gradient descent update:

$$\omega_{k+1} = \omega_k + \alpha \nabla_\omega J_{\omega_k} \tag{2}$$

update policy parameters in the direction of the gradient.

2. Likilihood policy gradient: (Parameter Exploring Policy Gradient)

$$\nabla_\omega J_\omega \int \pi(\theta;\omega) d\theta = \int \pi(\theta;\omega) \nabla_\omega log\pi(\theta;\omega) R_\theta d\theta \approx \sum_{i=1}^{N} \nabla_\omega log\pi(\theta_i;\omega) R^{[i]} \tag{3}$$

Need samples!
Subtracting a baseline:

$$\nabla_\omega J_\omega = \sum_{i=1}^{N} \nabla_\omega \log \pi(\theta_i;\omega)(R^{[i]} - b) \tag{4}$$

The variance is huge because the reward is computed on all variables!!!
Step based algorithm could be more efficient with smaller variance.

3. Step based policy gradient methods:

- Trajectory distribution:

$$p(\tau, \theta) = p(s_1) \prod_{t=1}^{T} \pi(a_t|s_t;\theta) p(s_{t+1}|s_t, a_t) \tag{5}$$

- Return for a single trajectory:

$$R(\tau) = \sum_{i=1}^{T} r_t \tag{6}$$

- Expected long term reward could be written as the expectation over the trajectory distribution:

$$J_\theta = \mathbb{E}_{p(\tau;\theta)}[R(\tau)] = \int p(\tau;\theta)R(\tau)d\tau \tag{7}$$

Similarly, using the log-ratio trick:

$$\nabla_\theta J_\theta = \sum_{i=1}^{N} \nabla_\theta \log p(\tau^{[i]};\theta)R(\tau^{[i]}) \tag{8}$$

According to the trajectory distribution, the $\nabla_\theta \log p(\tau^{[i]};\theta)$ could be computed by:

$$\nabla_\theta \log p(\tau^{[i]};\theta) = \sum_{t=1}^{T} \nabla_\theta \log \pi(a_t|s_t;\theta) \tag{9}$$

and then plug it back

$$\begin{aligned} \nabla_\theta J &= \sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)R(\tau) \\ &= \sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)(\sum_{t=1}^{T} r_t^{[i]}) \end{aligned} \tag{10}$$

- The variance is still high because we still use the term $R(\tau)$. One simple observation is that the reward in the past is not correlated with actions in the future. $\mathbb{E}_{p(\tau)}[r_t \log \pi(a_h|s_h)] = 0, \quad \forall t < h$. Then the observation leads to the Policy Gradient Theorem:

$$\begin{aligned} \nabla_\theta^{PG} J &= \sum_{i=1}^{N}\sum_{t=1}^{T-1} \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)(\sum_{h=t}^{T-1} r_h^{[i]} + r_T^{[i]}) \\ &= \sum_{i=1}^{N}\sum_{t=1}^{T-1} \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)Q_h^{[i]} \end{aligned} \tag{11}$$

In this way, the variance is decreased and also, a baseline could be considered.

4. Choose metric to measure the distance.

- Euclidian distance $L_2(\pi_{k+1}, \pi_k) = ||\omega_{k+1}, \omega_k||$ or $||\theta_{k+1}, \theta_k||$

- choose learning rate so that $L_2(\cdot) < \epsilon$ or other methods like resulting learning rate: $\alpha_k = \frac{\epsilon}{||\nabla J||}$
- KL-divergence (Kullback Leibler): $KL(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$ denoted the distance between distributions. p(x) is often the reference distibution. This KL-divergence denoted the information gain if distribution q is chosed instead of p. The entropy gain:

$$KL(p||q) = H(p,q) - H(p) \tag{12}$$

- KL divergence can be approximately by the Fisher information matrix (2nd order Tayler approximation)

$$KL(p_{\theta+\Delta\theta}||p_\theta) \approx \Delta\theta' G(\theta) \Delta\theta \tag{13}$$

5. Natural Gradient: the natural gradient uses the Fisher information metric as metric, its goal is to find the direction maximally correlated with gradient.

$$\begin{aligned} \nabla_\theta^{NG} J = \arg\max_{\Delta\theta} & \Delta\theta' \nabla_\theta J \\ s.t. KL(p_{\theta+\Delta\theta}||p_\theta) & \approx \Delta\theta' G(\theta) \Delta\theta \leq \epsilon \end{aligned} \tag{14}$$

   - The solution of this optimization problem is given by: $\nabla_\theta^{NG} J \propto G(\theta)^{-1} \nabla_\theta J$ see [1] for details.
   - Estimate the reward to come by function approximation $f_\omega(s,a) = \phi(s,a)'\omega \approx (Q_h^{[i]}(s,a) - b_h(s^{[i]}))$ and replace the reward to come in Policy Gradient Theorem with baseline by this approximation function as the gradient. It can be shown that this gradient is still unbiased if: $\phi(s,a) = \nabla_\theta \log \pi(a|s)$.
   - Natural features are obtained by log-gradient.
     Advantage function, compatible function approximation.
   -

$$\nabla_\theta^{FA} J = \mathbb{E}_{p(\tau)}[\nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta) \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)^T]\omega = \mathcal{F}(\theta)\omega \tag{15}$$

     One hint is that the expectation over τ is equivalent to two summations over i and t (using trajectory distribution).
   - One can show that $\mathcal{F}(\theta)\omega$ equals $G(\theta)$, then the natural gradient of J equals ω.

$$\nabla_\theta^{NG} J = \omega \tag{16}$$

   - Episodic Natural Actor-critic: sum up the Bellman Equations,

$$\begin{aligned} Q_1^\pi(s_1^{[i]}, a_1^{[i]}) &= r(s_1^{[i]}, a_1^{[i]}) + V_2^\pi(s_2^{[i]}) \\ V_1^\pi(s_1^{[i]}) + \nabla_\theta \log \pi(a_1^{[i]}|s_1^{[i]};\theta)\omega &= r(s_1^{[i]}, a_1^{[i]}) + V_2^\pi(s_2^{[i]}) \end{aligned} \tag{17}$$

iterate around i and sum them up, cancel same terms in both sides and yield:

$$V^{\pi}(s_1^{[i]}) + \sum_{t=1}^{T-1} \nabla_\theta \log \pi(a_t^{[i]}|s_t^{[i]};\theta)\omega = \sum_{t=1}^{T} e(s_t^{[i]}, a_t^{[i]}) \tag{18}$$

Notice for a rollout i, in the above equation, the first term is a scalar, the second term is a multiplication of $p \times 1$ and $1 \times p$ where p is the dimension of the θ. So the left part of the equation could be merged and can be rewritten as form of $y_i = X_i\beta$, where $\beta = (\omega, J)$, and then use linear regression to solve the problem.

1.2 Weighted Maximum Likelihood Approiaches

1. Success-Matching principle: "human attempts to match not the best taken avction but the reward-weighted frequency of their actions and outcomes" [Arrow, 1958].

$$\pi_{new}(a|s) \propto f(r(s,a))\pi_{old}(a|s) \tag{19}$$

   - Episode-based success matching. $\theta^{[i]} \sim \pi(\theta;\omega_k)$ and $R^{[i]} = \sum_{i=1}^{T} r_i^{[i]}$;
   - Denotes the weight $w^{[i]} = f(R^{[i]})$ an example is exponential transformation: $w^{[i]} = \exp(\beta(R^{[i]} - \max R^{[i]}))$
   - The remaining problem is to find a new parametric distribution that fits the weighted distribution: $\pi(\theta;\omega_{k+1}) \sim w^{[i]}\pi(\theta;\omega_k) \propto p(\theta^{[i]})$

$$\begin{aligned} \omega_{k+1} &= \arg\min_{\omega} KL(p(\theta^{[i]})||\pi(\theta^{[i]};\omega)) \\ &\approx \arg\max_{\omega} \sum_i \omega^{[i]} \log \pi(\theta^{[i]};\omega) \end{aligned} \tag{20}$$

     hint: constant terms are canceled in the derivation. If the old policy is Gaussian, then the solution is explicit and very similar to the formulas learned in the statistical learning (mean and covariance update)
   - The reward function of swing pendulums is intersting. (not noticed before)

2. Information Theoretic methods: directly obtain the policy using relative entropy as metric:

$$\begin{aligned} &\max_{\pi} \sum_i \pi(\theta^{[i]})R(\theta^{[i]}) \\ &s.t. \quad KL(\pi(\theta)||q(\theta) \le \epsilon \\ &\sum_i \pi(\theta^{[i]}) = 1 \end{aligned} \tag{21}$$

   ϵcould be used to balance the exporation and exploitation trande-off. This optimization has a analytical solution: $\pi(\theta) \propto q(\theta)\exp(\frac{\mathcal{R}_\theta}{\eta})$

1.3 Contectual and Hierarchical Policy Search: (to be finished...)

2 Model-Based Policy Search Methods

2.1 Greedy Updates

1. learn dynamics model from the data set $D = \{(s_{1:T}^{[i]}, a_{1:T}^{[i]})\}$, and learn the model in form of the transition function. Some examples include:
   - RBF networks [2]
   - Gaussian Processes [4]
   - Deep neural nets [3]
2. Use learned model as simulator: sampling; approximate probabilistic inference;
3. Bound the update process for model-based policy research (choose metrics used in the model-based policy search)
   - Greedy methods: PILCO, this software deserves a large time to dive into. Gaussian Process, policy evaluation and policy update using policy gradient methods. Data efficient and good performance.
   - GP-REPS (Gaussian Process Relative Entropy Policy Research) Use this model to solve problems with contextual difference. $\mu(s)$ denotes the distribution of contexts. $\pi(\omega|s)$ denotes the parameter distribution which is used to describe the model.
   $$J = \sum_{s,\omega} \mu(s)\pi(\omega|s)\mathcal{R}_{sw} \tag{22}$$
     - From the initialization, observe the real data and execute the policy with $\omega^{[i]} = \pi(\omega|s^{[i]})$, estimate $\hat{\mu}^{[i]}$;
     - Generate artificial data to create the reward expectation;
     - Update the policy $p(s,\omega) = \pi(\omega|s)\mu(s)$: bounding the relative entropy between the old distribution and the new distribution; Policy update $\pi(\omega|s)$ by using knowledge of evidence maximization under the GP framework;

PILCO

Three major components of PILCO framework are: the dynamics model, analytical approximate policy evaluation, and the gradient based policy improvement. Gaussian Process is used as the key model, in the procedure, the parameters have to be learned are the means and the covariance matrix which can discribe the model clearly.

2.2 Learn the dynamic modle

$$x_t = f(x_{t-1}, u_{t-1}) \tag{23}$$

PILCO use tules (x_t, u_t) as the training set, and uses the difference $\Delta_t = x_t - x_{t-1}$ as the training target (label), The mean function is GP is specified as zero $m(x) = 0$ and the covariance function is defined as:

$$k(\tilde{x_p}, \tilde{x_q}) = \sigma_f^2 \exp(-\frac{1}{2}(\tilde{x_p} - \tilde{x_q})^T \Lambda^{-1}(\tilde{x_p} - \tilde{x_q})) + \delta_{pq}\sigma_w^2 \tag{24}$$

where $\tilde{x} = [x^T u^T]^T$, σ_f^2 is the variance of the latent function, σ_w^2 is the variance of the noise. $\Sigma = diag(l_1^2, ...l_{D+F}^2$, which depends on the characteristic lengths. There parameters can be learn by evidence maximization. The Gaussian Process yields a one-step prediction:

$$\begin{aligned} p(x_t|x_{t-1}, u_{t-1}) &= \mathcal{N}(x_t|u_t, \Sigma_t) \\ \mu_t &= x_{t-1} + \mathbb{E}_f(\Delta_t) \\ \Sigma_t &= var_f(\Delta_t) \end{aligned} \tag{25}$$

Given n training inputs $\tilde{X} = [\tilde{x}_1, ...\tilde{x}_n]$ and n training targets $y = [\Delta_1, ...\Delta_n]$, the mean and the variance of an known test input $\tilde{x}_*$ can be computed as:

$$m_f(\tilde{x}_*) = \mathbb{E}_f[\Delta_*] = k_*^T(K + \sigma^2 I)^{-1} y \tag{26}$$

$$\sigma_f^2(\delta_*) = var_f[\Delta_*] = k_{**} - k_*^T(K + \sigma^2 I)^{-1} k_* \tag{27}$$

where $k_* = k(\tilde{X}, \tilde{x}_*)$, $k_{**} = k(\tilde{x}_*, \tilde{x}_*)$ are covariance matrix respectively. (recall the matrix A,B,C and C')

2.3 Policy evaluation

For predicting the state of x_t from $p(x_{t-1})$, we require a joint distribution $p(\tilde{x}_{t-1}) = p(x_{t-1}, u_{t-1}) = p(x_{t-1}, \pi(x_{t-1}, \theta))$, this joint distribution can be approxmated by a Gaussian (mean and covariance). Assume $p(\tilde{x}_{t-1}) = \mathcal{N}(\tilde{x}_{t-1}|\tilde{\mu}_{t-1}, \tilde{\Sigma}_{t-1})$, then the distribution of the difference can be derived:

$$p(\Delta_{t-1}) = \int\int p(f(\tilde{x}_{t-1})|\tilde{x}_{t-1}) p(\tilde{x}_{t-1}) df d\tilde{x}_{t-1} \tag{28}$$

The distribution of this difference function could be also appriximated by a Gaussian Process, with mean μ_Δ and covariance matrix Σ_Δ. Then the predictive distribution is:

$$\begin{aligned} \mu_t &= \mu_{t-1} + \mu_\Delta \\ \Sigma_t &= \Sigma_{t-1} + \Sigma_\Delta + cov(x_{t-1}, \Delta_t) + cov(\Delta_t, x_{t-1}) \end{aligned} \tag{29}$$

Thus the predictive distribution is $p(x_t) = \mathcal{N}(x_t|\mu_t, \Sigma_t)$

2.4 Policy gradient using gradient descent

The long term cost:

$$J^{\pi}(\theta) = \sum_{t=0}^{T} \mathbb{E}_{x_t}[c(x_t)], \quad x_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \tag{30}$$

Compute the $\frac{dJ^{\pi}}{d\theta}$ and update θ.

3 Paper list

3.1 Read

1. Higgins, I., et al. (2017). "Darla: Improving zero-shot transfer in reinforcement learning."
2. Hessel, M., et al. (2017). "Rainbow: Combining improvements in deep reinforcement learning."
3. Kupcsik, A. G., et al. (2013). Data-efficient generalization of robot skills with contextual policy search. Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI 2013.
4. Deisenroth, M. and C. E. Rasmussen (2011). PILCO: A model-based and data-efficient approach to policy search. Proceedings of the 28th International Conference on machine learning (ICML-11).
5. Wen, M., et al. (2017). Learning from demonstrations with high-level side information. Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17.
6. Cubuktepe, M., et al. (2018). "Synthesis in pMDPs: A Tale of 1001 Parameters."

3.2 Prepare to read

1. Levine, S. and V. Koltun (2014). Learning complex neural network policies with trajectory optimization. International Conference on Machine Learning.
2. Van Hoof, H., et al. (2015). Learning of non-parametric control policies with high-dimensional state features. Artificial Intelligence and Statistics.

References

[1] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[2] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[3] Sergey Levine and Vladlen Koltun. Learning complex neural network policies with trajectory optimization. In *International Conference on Machine Learning*, pages 829–837, 2014.

[4] Herke Van Hoof, Jan Peters, and Gerhard Neumann. Learning of non-parametric control policies with high-dimensional state features. In *Artificial Intelligence and Statistics*, pages 995–1003, 2015.