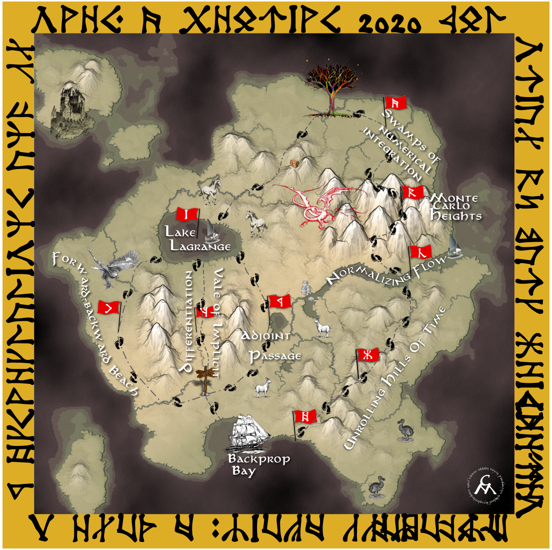
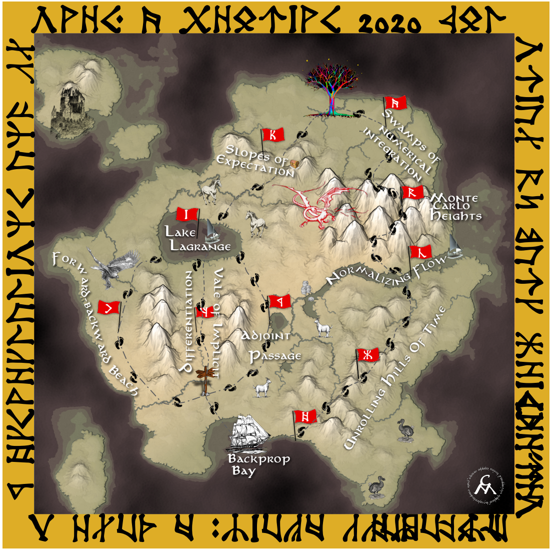

Swamps of Numerical Integration
Monte Carlo Heights
Lake Lagrange
Normalizing Flow
Forward-backward Beach
Differentiation
Vale of Implicit
Adjoint Passage
Unrolling Hills of Time
Backprop Bay


2020
Slopes of Expectation
Swamps of Numerical Integration
Monte Carlo Heights
Lake Lagrange
Normalizing Flow
Forward-backward Beach
Differentiation
Vale of Implicit
Adjoint Passage
Unrolling Hills of Time
Backprop Bay

Stochastic Gradient Estimation

Cheng Soon Ong
Marc Peter Deisenroth

December 2020

Journal of Machine Learning Research 21 (2020) 1-63 Submitted 03/19; Revised 04/20; Published 04/20

Monte Carlo Gradient Estimation in Machine Learning

Shakir Mohamed[*1] SHAKIR@GOOGLE.COM
Mihaela Rosca[*1 2] MIHAELACR@GOOGLE.COM
Michael Figurnov[*1] MFIGURNOV@GOOGLE.COM
Andriy Mnih[*1] AMNIH@GOOGLE.COM
[*]*Equal contributions;*
1 *DeepMind, London*
2 *University College London*

Setting

- Expected utility

$$\mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

- Distributional parameters $\boldsymbol{\theta}$

Setting

- Expected utility

$$\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \int U(\boldsymbol{x})p(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x}$$

- Distributional parameters $\boldsymbol{\theta}$
- Gradients w.r.t. distributional parameters $\boldsymbol{\theta}$ of an expected utility:

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})]$$

Setting

- Expected utility
$$\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \int U(\boldsymbol{x})p(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x}$$
- Distributional parameters $\boldsymbol{\theta}$
- Gradients w.r.t. distributional parameters $\boldsymbol{\theta}$ of an expected utility:
$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})]$$
- Sensitivity analysis ⏩ Explanation
- Training of machine learning models ⏩ Optimization

Where?

- **Variational inference.** Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})} \left[\log p(\boldsymbol{x}|\boldsymbol{z}) - \log \frac{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}{p(\boldsymbol{z})} \right]$$

Where?

- **Variational inference.** Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})} \left[\log p(\boldsymbol{x}|\boldsymbol{z}) - \log \frac{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}{p(\boldsymbol{z})} \right]$$

- **Reinforcement learning.** Gradient of expected long-term reward w.r.t. policy parameters $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})} \left[\sum_{t=0}^{T} \gamma^t r(\boldsymbol{x}_t, \boldsymbol{u}_t) \right], \qquad \text{trajectory } \boldsymbol{\tau} = (\boldsymbol{x}_0, \boldsymbol{u}_0, \ldots, \boldsymbol{x}_T, \boldsymbol{u}_T)$$

Where?

- **Variational inference.** Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})} \left[\log p(\boldsymbol{x}|\boldsymbol{z}) - \log \frac{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}{p(\boldsymbol{z})} \right]$$

- **Reinforcement learning.** Gradient of expected long-term reward w.r.t. policy parameters $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})} \left[\sum_{t=0}^{T} \gamma^t r(\boldsymbol{x}_t, \boldsymbol{u}_t) \right], \qquad \text{trajectory } \boldsymbol{\tau} = (\boldsymbol{x}_0, \boldsymbol{u}_0, \ldots, \boldsymbol{x}_T, \boldsymbol{u}_T)$$

- **Experimental design and Bayesian optimization.** Gradient of the probability of improvement w.r.t. designs $\boldsymbol{\theta}$ (where to measure next?):

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{y \sim p(y;\boldsymbol{\theta})} \left[\mathbb{1}_{\{y > y_{\text{best}}\}} \right]$$

Gradients of expectations

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})} [U(\boldsymbol{x})]$$

- If we can compute (an approximation of) the expected value analytically (requires deterministic approximate inference), we can use the **chain rule** to get the gradient
- Otherwise (e.g., stochastic approximate inference, mini-batching, Monte-Carlo estimation), we need to compute **gradients of a stochastic estimator**

Stochastic gradient estimators

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

- Derivatives of measures: Directly differentiate the measure $p(\boldsymbol{x};\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$
  - Score-function gradient estimators
  - Measure-valued gradient estimators

Stochastic gradient estimators

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

- Derivatives of measures: Directly differentiate the measure $p(\boldsymbol{x};\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$
  - Score-function gradient estimators
  - Measure-valued gradient estimators
- Derivatives of paths: Differentiate through path the parameters $\boldsymbol{\theta}$ take (via random variables $\boldsymbol{x}$ to U)
  - Pathwise gradient estimators

Stochastic gradient estimators

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

- Derivatives of measures: Directly differentiate the measure $p(\boldsymbol{x};\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$
  - Score-function gradient estimators
  - Measure-valued gradient estimators
- Derivatives of paths: Differentiate through path the parameters $\boldsymbol{\theta}$ take (via random variables $\boldsymbol{x}$ to U)
  - Pathwise gradient estimators
- Repeating pattern: Swap the order of differentiation and expectation
  - Gradients of deterministic quantities ✓
  - Monte-Carlo integration to compute the expectation ✓

Score-Function Gradient Estimators

Key insight

Key idea

Use log-derivative trick to turn the gradient of an expectation into an expectation of a gradient.

Key insight

Key idea

Use log-derivative trick to turn the gradient of an expectation into an expectation of a gradient.

- **Log-derivative trick**

$$\underbrace{\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}; \boldsymbol{\theta})}_{\text{score}} = \frac{\nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}; \boldsymbol{\theta})}{p(\boldsymbol{x}; \boldsymbol{\theta})} \quad \Rightarrow \quad \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}; \boldsymbol{\theta}) = p(\boldsymbol{x}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}; \boldsymbol{\theta})$$

Key insight

Key idea

Use log-derivative trick to turn the gradient of an expectation into an expectation of a gradient.

- **Log-derivative trick**

$$\underbrace{\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}; \boldsymbol{\theta})}_{\text{score}} = \frac{\nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}; \boldsymbol{\theta})}{p(\boldsymbol{x}; \boldsymbol{\theta})} \quad \Rightarrow \quad \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}; \boldsymbol{\theta}) = p(\boldsymbol{x}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}; \boldsymbol{\theta})$$

Score-function gradient estimator: Derivation

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Expectation as integration

Score-function gradient estimator: Derivation

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Expectation as integration

$$= \int U(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Move gradient inside

Score-function gradient estimator: Derivation

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Expectation as integration

$$= \int U(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Move gradient inside

$$= \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$$

Log-derivative trick

Score-function gradient estimator: Derivation

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] &= \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x} && \text{Expectation as integration} \\
&= \int U(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x} && \text{Move gradient inside} \\
&= \int U(\boldsymbol{x}) p(\boldsymbol{x};\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x} && \text{Log-derivative trick} \\
&= \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta})] && \text{Integral as expectation}
\end{aligned}
$$

Score-function gradient estimator

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x}) \underbrace{\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta})}_{\text{score}}]$$

- Turned gradient of an expectation into the expectation of a (deterministic) gradient
- Gradient is the expected utility-weighted score

Score-function gradient estimator

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x}) \underbrace{\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta})}_{\text{score}}]$$

- Turned gradient of an expectation into the expectation of a (deterministic) gradient
- Gradient is the expected utility-weighted score
- Monte-Carlo integration to get the gradient:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] \approx \frac{1}{S} \sum_{s=1}^{S} U(\boldsymbol{x}^{(s)}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}^{(s)};\boldsymbol{\theta}), \quad \boldsymbol{x}^{(s)} \sim p(\boldsymbol{x};\boldsymbol{\theta})$$

Properties: Score-function gradient estimator

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] \approx \frac{1}{S}\sum_{s=1}^{S} U(\boldsymbol{x}^{(s)}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}^{(s)};\boldsymbol{\theta}), \qquad \boldsymbol{x}^{(s)} \sim p(\boldsymbol{x};\boldsymbol{\theta})$$

- Single-sample estimation is OK, i.e., $S = 1$
- Any type of utility function U can be used (e.g., non-differentiable)
- p must be differentiable w.r.t. $\boldsymbol{\theta}$
- Must be able to sample easily from $p(\boldsymbol{x};\boldsymbol{\theta})$
- Discrete and continuous distributions are OK
- Techniques to control the variance of the estimator (e.g., Greensmith et al., 2004; Titsias & Lázaro-Gredilla, 2015)

Example: REINFORCE (Williams, 1992)

- Optimize parameters $\boldsymbol{\theta}$ of a (stochastic) policy $p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})$

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau})]$$

- $U(\boldsymbol{\tau})$: Long-term reward for trajectory $\boldsymbol{\tau}$

Example: REINFORCE (Williams, 1992)

- Optimize parameters $\boldsymbol{\theta}$ of a (stochastic) policy $p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau})]$$

- $U(\boldsymbol{\tau})$: Long-term reward for trajectory $\boldsymbol{\tau}$
- Trajectory distribution (with $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\epsilon}_t)$)

$$p(\boldsymbol{\tau};\boldsymbol{\theta}) = p(\boldsymbol{x}_0, \boldsymbol{u}_0, \dots, \boldsymbol{u}_T, \boldsymbol{x}_T;\boldsymbol{\theta}) = p(\boldsymbol{x}_0)\prod_{t=0}^{T} \underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{state transition}} \underbrace{p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})}_{\text{policy}}$$

Example: REINFORCE (2) (Williams, 1992)

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau}; \boldsymbol{\theta})}[U(\boldsymbol{\tau})] = \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau}; \boldsymbol{\theta})}[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\tau}; \boldsymbol{\theta})]$$

Example: REINFORCE (2) (Williams, 1992)

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau})] = \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\tau};\boldsymbol{\theta})]$$

$$= \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})} \left[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \left(\log p(\boldsymbol{x}_0) + \sum_{t=0}^{T} \log \underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{state transition}} + \log \underbrace{p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})}_{\text{policy}} \right) \right]$$

Example: REINFORCE (2) (Williams, 1992)

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau})] = \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\tau};\boldsymbol{\theta})]$$

$$= \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})} \left[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \left(\log p(\boldsymbol{x}_0) + \sum_{t=0}^{T} \log \underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{state transition}} + \log \underbrace{p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})}_{\text{policy}} \right) \right]$$

$$= \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})} \left[U(\boldsymbol{\tau}) \sum_{t=0}^{T} \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta}) \right]$$

Example: REINFORCE (2) (Williams, 1992)

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau})] = \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\tau};\boldsymbol{\theta})]$$

$$= \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}\left[U(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \left(\log p(\boldsymbol{x}_0) + \sum_{t=0}^{T} \log \underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{state transition}} + \log \underbrace{p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})}_{\text{policy}} \right)\right]$$

$$= \mathbb{E}_{\boldsymbol{\tau} \sim p(\boldsymbol{\tau};\boldsymbol{\theta})}\left[U(\boldsymbol{\tau}) \sum_{t=0}^{T} \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})\right]$$

- Markov property of state evolution
  - ⏩ Only need gradient of the log-policy at each time step
- Monte Carlo for expectation
- Can be used in model-free and model-based settings

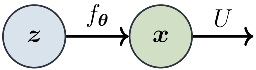
Applications

- Reinforcement learning (e.g., Williams, 1992; Sutton et al., 2000)
- (Black-box) variational inference (e.g., Paisley et al., 2012, Ranganath et al., 2014)
- Discrete-event systems (operations research)
- Computational finance

Pathwise Gradient Estimators

Setting

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})]$$



$$\boldsymbol{x} = f(\boldsymbol{z};\boldsymbol{\theta})$$

- Data $\boldsymbol{x}$ can be obtained by a deterministic transformation f (path) of a latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, where $p(\boldsymbol{z})$ has no tunable parameters, e.g., $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
- Distributional parameters of $p(\boldsymbol{x};\boldsymbol{\theta})$ are the parameters of the path f
- Push gradients through this path (chain rule)

Setting

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})} [U(\boldsymbol{x})]$$

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U} \qquad\qquad \boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

- Data $\boldsymbol{x}$ can be obtained by a deterministic transformation f (path) of a latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, where $p(\boldsymbol{z})$ has no tunable parameters, e.g., $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
- Distributional parameters of $p(\boldsymbol{x}; \boldsymbol{\theta})$ are the parameters of the path f
- Push gradients through this path (chain rule)

Key idea

Define a path from a latent variable $\boldsymbol{z}$ to data $\boldsymbol{x}$ and use the change-of-variables trick to turn the gradient of an expectation into an expectation of a gradient.

Derivation

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}; \boldsymbol{\theta})}[U(\boldsymbol{x})] = \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x}; \boldsymbol{\theta}) d\boldsymbol{x}$$

Expectation as integration

Derivation

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] &= \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x}; \boldsymbol{\theta}) d\boldsymbol{x} && \text{Expectation as integration} \\
&= \nabla_{\boldsymbol{\theta}} \int U(f(\boldsymbol{z}; \boldsymbol{\theta})) p(\boldsymbol{z}) d\boldsymbol{z} && \text{Change of variables}
\end{aligned}$$

Derivation

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] &= \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x}; \boldsymbol{\theta}) d\boldsymbol{x} && \text{Expectation as integration} \\
&= \nabla_{\boldsymbol{\theta}} \int U(f(\boldsymbol{z}; \boldsymbol{\theta})) p(\boldsymbol{z}) d\boldsymbol{z} && \text{Change of variables} \\
&= \int \nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta})) p(\boldsymbol{z}) d\boldsymbol{z} && \text{Move gradient inside}
\end{aligned}$$

Derivation

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] &= \nabla_{\boldsymbol{\theta}} \int U(\boldsymbol{x}) p(\boldsymbol{x}; \boldsymbol{\theta}) d\boldsymbol{x} && \text{Expectation as integration} \\
&= \nabla_{\boldsymbol{\theta}} \int U((\boldsymbol{z}; \boldsymbol{\theta})) p(\boldsymbol{z}) d\boldsymbol{z} && \text{Change of variables} \\
&= \int \nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta})) p(\boldsymbol{z}) d\boldsymbol{z} && \text{Move gradient inside} \\
&= \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\nabla_{\boldsymbol{\theta}} U(\underbrace{f(\boldsymbol{z}; \boldsymbol{\theta})}_{=\boldsymbol{x}})] && \text{Integral as expectation}
\end{aligned}$$

Pathwise gradient estimator

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}; \boldsymbol{\theta})}[U(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta}))]$$

- Turned gradient of an expectation into an expectation (w.r.t. a parameter-free distribution) of a (deterministic) gradient
  - Push parameters inside U

Pathwise gradient estimator

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta}))]$$

- Turned gradient of an expectation into an expectation (w.r.t. a parameter-free distribution) of a (deterministic) gradient
  - Push parameters inside U
- Monte-Carlo estimator of the gradient

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}^{(s)}; \boldsymbol{\theta})), \quad \boldsymbol{z}^{(s)} \sim p(\boldsymbol{z})$$

Pathwise gradient estimator

$$\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{x} \xrightarrow{U}$$

$$\boldsymbol{x} = f(\boldsymbol{z}; \boldsymbol{\theta})$$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta}))]$$

- Turned gradient of an expectation into an expectation (w.r.t. a parameter-free distribution) of a (deterministic) gradient
  - Push parameters inside U
- Monte-Carlo estimator of the gradient

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}^{(s)}; \boldsymbol{\theta})), \quad \boldsymbol{z}^{(s)} \sim p(\boldsymbol{z})$$

$$\nabla_{\boldsymbol{\theta}} U(f(\boldsymbol{z}; \boldsymbol{\theta})) = \nabla_{\boldsymbol{x}} U(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{z}; \boldsymbol{\theta})$$ Chain rule

Properties: Pathwise gradient estimator

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})] \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{x}} U(\boldsymbol{x}^{(s)}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{z}^{(s)}; \boldsymbol{\theta}), \quad \boldsymbol{z}^{(s)} \sim p(\boldsymbol{z})$$

- Single-sample estimation OK, i.e., $S = 1$.
- Utility U must be differentiable
- Path f must be differentiable
- Need to be able to sample from $p(\boldsymbol{z})$, but not from $p(\boldsymbol{x};\boldsymbol{\theta})$
- Often lower variance than score-function gradient estimator
- Control variability of path f to control variance of the estimator

Example: Bayesian optimization (Wilson et al., 2018)

- Inner loop of Bayesian optimization: Where to measure next?
  ⏩ Maximize acquisition function
- Many acquisition functions can be written as expected utilities

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{y};\boldsymbol{\theta})}[U(\boldsymbol{y})] = \int U(\boldsymbol{y}) p(\boldsymbol{y};\boldsymbol{\theta}) d\boldsymbol{y}, \qquad p(\boldsymbol{y};\boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$$

Example: Bayesian optimization (Wilson et al., 2018)

- Inner loop of Bayesian optimization: Where to measure next?
  ⏩ Maximize acquisition function
- Many acquisition functions can be written as expected utilities

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{y};\boldsymbol{\theta})}[U(\boldsymbol{y})] = \int U(\boldsymbol{y}) p(\boldsymbol{y};\boldsymbol{\theta}) d\boldsymbol{y}, \qquad p(\boldsymbol{y};\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Define path from $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ to $\boldsymbol{y}$ ⏩ Pathwise gradient estimation

Example: Bayesian optimization (Wilson et al., 2018)

Abbr.	Acquisition Function $\mathcal{L}$	Reparameterization	MM
EI	$\mathbb{E}_{\mathbf{y}}[\max(\mathrm{ReLU}(\mathbf{y}-\alpha))]$	$\mathbb{E}_{\mathbf{z}}[\max(\mathrm{ReLU}(\boldsymbol{\mu}+\mathbf{L}\mathbf{z}-\alpha))]$	Y
PI	$\mathbb{E}_{\mathbf{y}}[\max(\mathbb{1}^{-}(\mathbf{y}-\alpha))]$	$\mathbb{E}_{\mathbf{z}}[\max(\sigma(\frac{\boldsymbol{\mu}+\mathbf{L}\mathbf{z}-\alpha}{\tau}))]$	Y
SR	$\mathbb{E}_{\mathbf{y}}[\max(\mathbf{y})]$	$\mathbb{E}_{\mathbf{z}}[\max(\boldsymbol{\mu}+\mathbf{L}\mathbf{z})]$	Y
UCB	$\mathbb{E}_{\mathbf{y}}[\max(\boldsymbol{\mu}+\sqrt{\beta\pi/2}\lvert\boldsymbol{\gamma}\rvert)]$	$\mathbb{E}_{\mathbf{z}}[\max(\boldsymbol{\mu}+\sqrt{\beta\pi/2}\lvert\mathbf{L}\mathbf{z}\rvert)]$	Y
ES	$-\mathbb{E}_{\mathbf{y}_a}[\mathrm{H}(\mathbb{E}_{\mathbf{y}_b \mid \mathbf{y}_a}[\mathbb{1}^{+}(\mathbf{y}_b-\max(\mathbf{y}_b))])]$	$-\mathbb{E}_{\mathbf{z}_a}[\mathrm{H}(\mathbb{E}_{\mathbf{z}_b}[\mathrm{softmax}(\frac{\boldsymbol{\mu}_{b\mid a}+\mathbf{L}_{b\mid a}\mathbf{z}_b}{\tau})])]$	N
KG	$\mathbb{E}_{\mathbf{y}_a}[\max(\boldsymbol{\mu}_b+\boldsymbol{\Sigma}_{b,a}\boldsymbol{\Sigma}_{a,a}^{-1}(\mathbf{y}_a-\boldsymbol{\mu}_a))]$	$\mathbb{E}_{\mathbf{z}_a}[\max(\boldsymbol{\mu}_b+\boldsymbol{\Sigma}_{b,a}\boldsymbol{\Sigma}_{a,a}^{-1}\mathbf{L}_a\mathbf{z}_a)]$	N

Table 1: Examples of reparameterizable acquisition functions; the final column indicates whether they belong to the MM family (Section 3.2). Glossary: $\mathbb{1}^{+/-}$ denotes the right-/left-continuous Heaviside step function; ReLU and σ rectified linear and sigmoid nonlinearities, respectively; H the Shannon entropy; α an improvement threshold; τ a temperature parameter; $\mathbf{L}\mathbf{L}^{\top} \triangleq \boldsymbol{\Sigma}$ the Cholesky factor; and, residuals $\boldsymbol{\gamma} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. Lastly, non-myopic acquisition function (ES and KG) are assumed to be defined using a discretization. Terms associated with the query set and discretization are respectively denoted via subscripts a and b.

From Wilson et al. (2018)

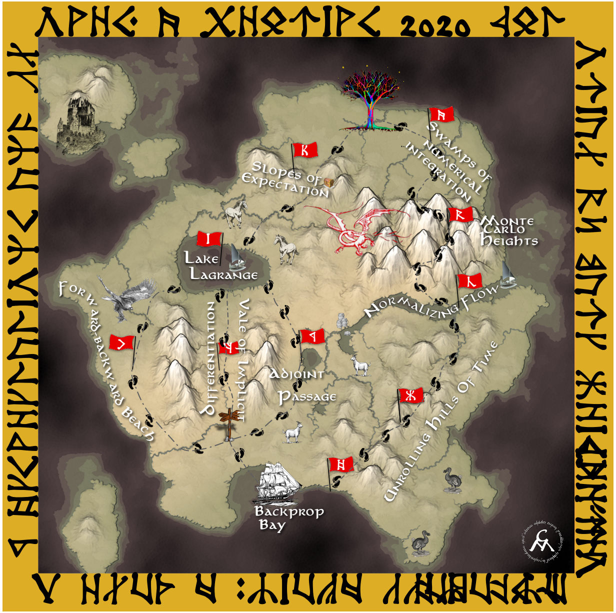
Application areas

- Bayesian optimization (e.g., Wilson et al., 2018)
- Normalizing flows (e.g., Rezende & Mohamed, 2015)
- Variational auto-encoders (e.g., Kingma & Welling, 2014; Rezende et al., 2014)
- Generative models (e.g., Goodfellow et al., 2014; Mohamed & Lakshminarayanan, 2016)
- Reinforcement learning (e.g., Heess et al., 2015)
- Probabilistic programming (e.g., Ritchie et al., 2016)

Summary

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{\theta})}[U(\boldsymbol{x})]$$

- Compute gradient of an expected utility
- Key idea: Swap order of differentiation and integration (expectation)
  ⏩ Use Monte Carlo methods to compute gradients
- Score-function gradient estimator using log-derivative trick
- Pathwise gradient estimator defines a parametrized path from a latent variable to the data



Slopes of Expectation
Swamps of Numerical Integration
Monte Carlo Heights
Lake Lagrange
Normalizing Flow
Forward-Backward Beach
Differentiation
Vale of Implicit
Adjoint Passage
Unrolling Hills of Time
Backprop Bay

References

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*.

Greensmith, E., Bartlett, P. L., and Baxter, J. (2004). Variance Reduction Techniques for Gradient Estimates inReinforcement Learning. *Journal of Machine Learning Research*, 5:1471–1530.

Heess, N., Wayne, G., andTimothy Lillicrap, D. S., Tassa, Y., and Erez, T. (2015). Learning Continuous Control Policies by Stochastic Value Gradients. In *Advances in Neural Information Processing Systems*.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *Proceedings of the International Conference on Learning Representations*.

Mohamed, S. and Lakshminarayanan, B. (2016). Learning in Implicit Generative Models. *arXiv:1610.03483*.

Paisley, J., Blei, D. M., and Jordan, M. I. (2012). Variational Bayesian Inference with Stochastic Search. In *Proceedings of the International Conference on Machine Learning*.

Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black Box Variational Inference. *arXiv:1401.0118*.

References (cont.)

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic Backpropagation and Variational Inference in Deep Latent Gaussian Models. In *Proceedings of the International Conference on Machine Learning*.

Ritchie, D., Horsfall, P., and Goodman, N. D. (2016). Deep Amortized Inference for Probabilistic Programs. *arXiv:1610.05735*.

Sutton, R. S., Mcallester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*.

Titsias, M. K. and Lázaro-Gredilla, M. (2015). Local Expectation Gradients for Black Box Variational Inference. In *Advances in Neural Information Processing Systems*.

Williams, R. J. (1992). Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8(3):229–256.

Wilson, J. T., Hutter, F., and Deisenroth, M. P. (2018). Maximizing Acquisition Functions for Bayesian Optimization. In *Advances in Neural Information Processing Systems*.