There and Back Again: A Tale of Expectations and Slopes. A Neurips 2020 Tutorial by Marc Deisenroth and Bay Backprop Heights Monte Carlo Lagrange Lake Differentiation No-rollbacking Flowing Thru the Swamps of Vale of Implicit Forwards-fully-backward Passage Adjoint Unrolling Hills Of Time
There and Back Again: A Tale of Expectations and Slopes. A Neurips 2020 Tutorial by Marc Deisenroth and Bay Backprop Heights Monte Carlo

Lagrange Lake Differentiation

Noremlization Flow

Expectation Slopes of integration Numerical Swamps of Vale of Implicit Forward-backward

Passage Adjoint Unrolling Hills

Of Time
Stochastic Gradient Estimation

Cheng Soon Ong
Marc Peter Deisenroth

December 2020
Monte Carlo Gradient Estimation in Machine Learning

Shakir Mohamed*1
Mihaela Rosca*1 2
Michael Figurnov*1
Andriy Mnih*1
*Equal contributions;
1 DeepMind, London
2 University College London

SHAKIR@GOOGLE.COM
MIHAELACR@GOOGLE.COM
MFIGURNOV@GOOGLE.COM
AMNIH@GOOGLE.COM
Setting

- Expected utility

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

- Distributional parameters \( \theta \)
Setting

- Expected utility

\[ E_{x \sim p(x; \theta)}[U(x)] = \int U(x)p(x; \theta) \, dx \]

- Distributional parameters \( \theta \)

- Gradients w.r.t. distributional parameters \( \theta \) of an expected utility:

\[ \nabla_{\theta} E_{x \sim p(x; \theta)}[U(x)] \]
Setting

- Expected utility

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

- Distributional parameters \( \theta \)

- Gradients w.r.t. distributional parameters \( \theta \) of an expected utility:

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

- Sensitivity analysis

- Training of machine learning models
Variational inference. Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\theta$:

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

- **Variational inference.** Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\theta$:
  \[
  \nabla_\theta \mathbb{E}_{z \sim q(z|x; \theta)} \left[ \log p(x|z) - \log \frac{q(z|x; \theta)}{p(z)} \right]
  \]

- **Reinforcement learning.** Gradient of expected long-term reward w.r.t. policy parameters $\theta$:
  \[
  \nabla_\theta \mathbb{E}_{\tau \sim p(\tau; \theta)} \left[ \sum_{t=0}^{T} \gamma^t r(x_t, u_t) \right], \quad \text{trajectory } \tau = (x_0, u_0, \ldots, x_T, u_T)
  \]
Variational inference. Gradient of evidence lower bound (ELBO) w.r.t. variational parameters $\theta$:

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

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

$$\nabla_{\theta} \mathbb{E}_{\tau \sim p(\tau;\theta)} \left[ \sum_{t=0}^{T} \gamma^t r(x_t, u_t) \right], \text{ trajectory } \tau = (x_0, u_0, \ldots, x_T, u_T)$$

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

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

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

- 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

\[ \nabla_{\theta} \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_{\theta} \int U(x)p(x; \theta) \, dx \]

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

Gradients of deterministic quantities
Monte-Carlo integration to compute the expectation
Stochastic gradient estimators

\[ \nabla_{\theta} \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_{\theta} \int U(x) p(x; \theta) \, dx \]

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

\[ \nabla_\theta \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_\theta \int U(x)p(x; \theta) \, dx \]

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

\[
\frac{\partial}{\partial \theta} \log p(x; \theta) = \mathbb{E}_{x \sim p(x; \theta)} \left[ \frac{\partial}{\partial \theta} \log p(x; \theta) \right]
\]
Key insight

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

Log-derivative trick

$$\nabla_{\theta} \log p(x; \theta) = \frac{\nabla_{\theta} p(x; \theta)}{p(x; \theta)}$$

$$\nabla_{\theta} p(x; \theta) = p(x; \theta) \nabla_{\theta} \log p(x; \theta)$$
Key insight

Key idea

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

Log-derivative trick

\[
\nabla_\theta \log p(x; \theta) = \frac{\nabla_\theta p(x; \theta)}{p(x; \theta)}
\]

\[
\nabla_\theta p(x; \theta) = p(x; \theta) \nabla_\theta \log p(x; \theta)
\]
Score-function gradient estimator: Derivation

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

Expectation as integration
Score-function gradient estimator: Derivation

\[ \nabla_{\theta} \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_{\theta} \int U(x) p(x; \theta) \, dx \]

\[ = \int U(x) \nabla_{\theta} p(x; \theta) \, dx \]

Expectation as integration

Move gradient inside
Score-function gradient estimator: Derivation

\[ \nabla_\theta \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_\theta \int U(x)p(x; \theta) \, dx = \int U(x) \nabla_\theta p(x; \theta) \, dx = \int U(x)p(x; \theta) \nabla_\theta \log p(x; \theta) \, dx \]

- **Expectation as integration**
- **Move gradient inside**
- **Log-derivative trick**
Score-function gradient estimator: Derivation

\[
\nabla_\theta E_{x \sim p(x;\theta)}[U(x)] = \nabla_\theta \int U(x)p(x;\theta)dx
\]

\[
= \int U(x)\nabla_\theta p(x;\theta)dx
\]

\[
= \int U(x)p(x;\theta)\nabla_\theta \log p(x;\theta)dx
\]

\[
= E_{x \sim p(x;\theta)}[U(x)\nabla_\theta \log p(x;\theta)]
\]
Score-function gradient estimator

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

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

$$\nabla_\theta E_{x \sim p(x; \theta)}[U(x)] = E_{x \sim p(x; \theta)}[U(x) \nabla_\theta \log p(x; \theta)]$$

- 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:

$$\nabla_\theta E_{x \sim p(x; \theta)}[U(x)] \approx \frac{1}{S} \sum_{s=1}^{S} U(x^{(s)}) \nabla_\theta \log p(x^{(s)}; \theta), \quad x^{(s)} \sim p(x; \theta)$$
Properties: Score-function gradient estimator

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

- 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. \( \theta \)
- Must be able to sample easily from \( p(x; \theta) \)
- 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 $\theta$ of a (stochastic) policy $p(u_t | x_t; \theta)$

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

- $U(\tau)$: Long-term reward for trajectory $\tau$
Example: REINFORCE (Williams, 1992)

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

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

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

- Trajectory distribution (with $\mathbf{x}_{t+1} = f(\mathbf{x}_t, u_t, \epsilon_t)$)

$$
p(\tau; \theta) = p(\mathbf{x}_0, u_0, \ldots, u_T, x_T; \theta) = p(\mathbf{x}_0) \prod_{t=0}^{T} p(x_{t+1}|\mathbf{x}_t, u_t) p(u_t|\mathbf{x}_t; \theta)
$$

- State transition
- Policy
Example: REINFORCE (2) (Williams, 1992)

\[ \nabla_{\theta} E_{\tau \sim p(\tau;\theta)}[U(\tau)] = E_{\tau \sim p(\tau;\theta)}[U(\tau) \nabla_{\theta} \log p(\tau;\theta)] \]
Example: REINFORCE (2) (Williams, 1992)

\[ \nabla_\theta \mathbb{E}_{\tau \sim p(\tau; \theta)}[U(\tau)] = \mathbb{E}_{\tau \sim p(\tau; \theta)}[U(\tau) \nabla_\theta \log p(\tau; \theta)] \\
= \mathbb{E}_{\tau \sim p(\tau; \theta)}[U(\tau) \nabla_\theta \left( \log p(x_0) + \sum_{t=0}^{T} \log p(x_{t+1}|x_t, u_t) + \log p(u_t|x_t; \theta) \right)] \\
\]

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
Example: REINFORCE (2) (Williams, 1992)

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

\[
= \mathbb{E}_{\tau \sim p(\tau; \theta)} \left[ U(\tau) \nabla_\theta \left( \log p(x_0) + \sum_{t=0}^{T} \log p(x_{t+1} | x_t, u_t) + \log p(u_t | x_t; \theta) \right) \right]
\]

\[
= \mathbb{E}_{\tau \sim p(\tau; \theta)} \left[ U(\tau) \sum_{t=0}^{T} \nabla_\theta \log p(u_t | x_t; \theta) \right]
\]

▶ 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
Example: REINFORCE (2) (Williams, 1992)

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

\[ = \mathbb{E}_{\tau \sim p(\tau; \theta)} \left[ U(\tau) \nabla_{\theta} \left( \log p(x_0) + \sum_{t=0}^{T} \log p(x_{t+1} | x_t, u_t) + \log p(u_t | x_t; \theta) \right) \right] \]

\[ = \mathbb{E}_{\tau \sim p(\tau; \theta)} \left[ U(\tau) \sum_{t=0}^{T} \nabla_{\theta} \log p(u_t | x_t; \theta) \right] \]

- 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
Data $\mathbf{x}$ can be obtained by a deterministic transformation $f$ (path) of a latent variable $z \sim p(z)$, where $p(z)$ has no tunable parameters, e.g., $p(z) = \mathcal{N}(0, I)$.

Distributional parameters of $p(\mathbf{x}; \theta)$ are the parameters of the path $f$.

Push gradients through this path (chain rule).

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

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

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

**Key idea**

Define a path from a latent variable \( z \) to data \( x \) and use the change-of-variables trick to turn the gradient of an expectation into an expectation of a gradient.
Derivation

\[ x = f(z; \theta) \]

\[ \nabla_\theta \mathbb{E}_{x \sim p(x; \theta)}[U(x)] = \nabla_\theta \int U(x)p(x; \theta) \, dx \]

Expectation as integration
Derivation

\[ x = f(z; \theta) \]

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

\[ = \nabla_{\theta} \int U(f(z; \theta))p(z)dz \]

Expectation as integration

Change of variables
Derivation

\[ x = f(z; \theta) \]

\[ \nabla_{\theta} E_{x \sim p(x; \theta)}[U(x)] = \nabla_{\theta} \int U(x)p(x; \theta)dx \]

= \[ \nabla_{\theta} \int U(f(z; \theta))p(z)dz \]

= \[ \int \nabla_{\theta} U(f(z; \theta))p(z)dz \]

Expectation as integration

Change of variables

Move gradient inside
Derivation

$$x = f(z; \theta)$$

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

$$= \nabla_\theta \int U((z; \theta))p(z)dz$$

$$= \int \nabla_\theta U(f(z; \theta))p(z)dz$$

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

Expectation as integration

Change of variables

Move gradient inside

Integral as expectation
Pathwise gradient estimator

\[
x = f(z; \theta)
\]

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

- 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

\[ x = f(z; \theta) \]

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

- 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

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

\[ x = f(z; \theta) \]

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

- 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

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

\[ \nabla_\theta U(f(z; \theta)) = \nabla_x U(x) \nabla_\theta f(z; \theta) \quad \text{Chain rule} \]
Properties: Pathwise gradient estimator

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

- 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 \( p(z) \), but not from \( p(x; \theta) \)
- 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

\[
\mathcal{L} = \mathbb{E}_{y \sim p(y; \theta)}[U(y)] = \int U(y)p(y; \theta)\,dy, \quad p(y; \theta) = \mathcal{N}(\mu, \Sigma)
\]
Inner loop of Bayesian optimization: Where to measure next?

Maximize acquisition function

Many acquisition functions can be written as expected utilities

\[
\mathcal{L} = \mathbb{E}_{y \sim p(y; \theta)}[U(y)] = \int U(y)p(y; \theta)dy,
\]

\( \text{where } p(y; \theta) = \mathcal{N}(\mu, \Sigma) \)

Define path from \( z \sim \mathcal{N}(0, I) \) to \( y \)  Pathwise gradient estimation
Example: Bayesian optimization (Wilson et al., 2018)

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Acquisition Function $\mathcal{L}$</th>
<th>Reparameterization</th>
<th>MM</th>
</tr>
</thead>
<tbody>
<tr>
<td>EI</td>
<td>$\mathbb{E}_y[\max(\text{ReLU}(y - \alpha))]$</td>
<td>$\mathbb{E}_z[\max(\text{ReLU}(\mu + \mathbf{L}z - \alpha))]$</td>
<td>$y$</td>
</tr>
<tr>
<td>PI</td>
<td>$\mathbb{E}_y[\max(1^- (y - \alpha))]$</td>
<td>$\mathbb{E}_z[\max(\frac{\sigma}{\tau}(\frac{\mu + \mathbf{L}z - \alpha}{\tau}))]$</td>
<td>$y$</td>
</tr>
<tr>
<td>SR</td>
<td>$\mathbb{E}_y[\max(y)]$</td>
<td>$\mathbb{E}_z[\max(\mu + \mathbf{L}z)]$</td>
<td>$y$</td>
</tr>
<tr>
<td>UCB</td>
<td>$\mathbb{E}_y[\max(\mu + \sqrt{\frac{\beta\pi}{2}}</td>
<td>\gamma</td>
<td>)]$</td>
</tr>
<tr>
<td>ES</td>
<td>$-\mathbb{E}<em>{y_a}[H(\mathbb{E}</em>{y_b}</td>
<td>y_a[1^+ (y_b - \max(y_b))])]$</td>
<td>$-\mathbb{E}<em>{z_a}[H(\mathbb{E}</em>{z_b}</td>
</tr>
<tr>
<td>KG</td>
<td>$\mathbb{E}<em>{y_a}[\max(\mu</em>{b} + \Sigma_{b,a} \Sigma^{-1}<em>{a,a} (y</em>{a} - \mu_{a}))]$</td>
<td>$\mathbb{E}<em>{z_a}[\max(\mu</em>{b} + \Sigma_{b,a} \Sigma^{-1}<em>{a,a} \mathbf{L}</em>{a}z_{a})]$</td>
<td>$N$</td>
</tr>
</tbody>
</table>

Table 1: Examples of reparameterizable acquisition functions; the final column indicates whether they belong to the MM family (Section 3.2). Glossary: $1^+/-$ denotes the right-/left-continuous Heaviside step function; ReLU and $\sigma$ rectified linear and sigmoid nonlinearities, respectively; $H$ the Shannon entropy; $\alpha$ an improvement threshold; $\tau$ a temperature parameter; $\mathbf{L}\mathbf{L}^\top \triangleq \Sigma$ the Cholesky factor; and, residuals $\gamma \sim \mathcal{N}(0, \Sigma)$. 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)
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
There and Back Again: A Tale of Expectations and Slopes. A Neurips 2020 Tutorial by Marc Deisenroth and Bay Backprop Heights Monte Carlo Lagrange Lake Differentiation Normalizing Flow Variational Integration Monte Carlo Heights Swamps of Implicit Backward-ward Swamps of Numerical Integration Expectation Slopes of Integration Numerical Swamps of Vale of Implicit Forward and Backward Adjoint Passage Adjoint Unrolling Hills Of Time


