UNIVERSITY OF CAMBRIDGE

Stochastic Approximation Theory

Yingzhen Li and Mark Rowland

November 26, 2015

Plan

UNIVERSITY OF CAMBRIDGE

- History and modern formulation of stochastic approximation theory
- In-depth look at stochastic gradient descent (SGD)
- Introduction to key ideas in stochastic approximation theory such as Lyapunov functions, quasimartingales, and also numerical solutions to differential equations.

Bibliography

Stochastic Approximation and Recursive Algorithms and Applications, Kushner & Lin (2003)

Online Learning and Stochastic Approximation, Léon Bottou (1998)

A Stochastic Approximation Algorithm, Robbins & Monro (1951)

Stochastic Estimation of the Maximisation of a Regression Function, Kiefer & Wolfowitz (1952)

Introduction to Stochastic Search and Optimization, Spall (2003)

Numerical Analysis is a well-established discipline...

c. 1800 - 1600 BC

Babylonians attempted to calculate $\sqrt{2}$, or in modern terms, find the roots of $x^2 - 2 = 0$.

1

1 http://www.math.ubc.ca/~cass/Euclid/ybc/ybc.html, Yale Babylonian Collection

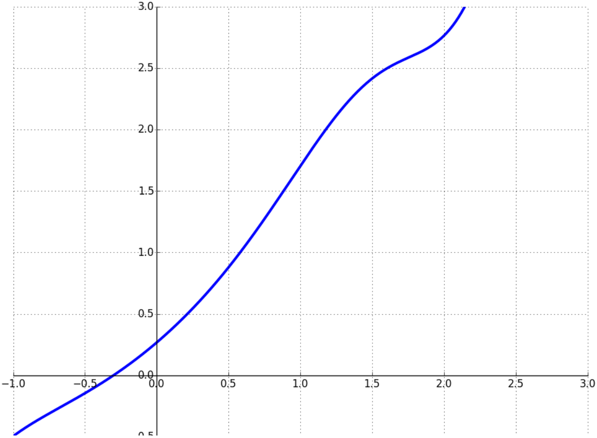
C17-C19

Huge number of numerical techniques developed as tools for the natural sciences:

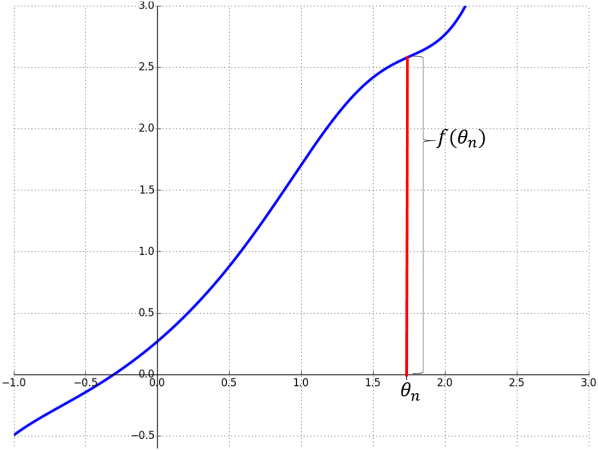
- Root-finding methods (e.g. Newton-Raphson)
- Numerical integration (e.g. Gaussian Quadrature)
- ODE solution (e.g. Euler method)
- Interpolation (e.g. Lagrange polynomials)

Focus is on situations where function evaluation is deterministic.

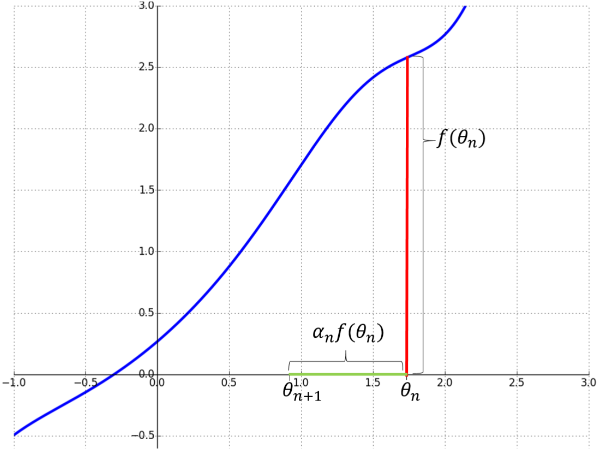
1951 Robbins and Monro publish "*A Stochastic Approximation Algorithm*", describing how to find the root of an increasing function $f : \mathbb{R} \to \mathbb{R}$ when only *noisy* estimates of the function's value at a given point are available.



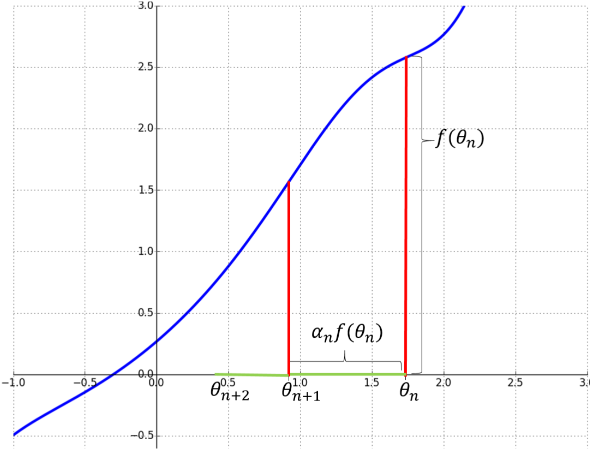
1951 Robbins and Monro publish "*A Stochastic Approximation Algorithm*", describing how to find the root of an increasing function $f : \mathbb{R} \to \mathbb{R}$ when only *noisy* estimates of the function's value at a given point are available.



1951 Robbins and Monro publish "*A Stochastic Approximation Algorithm*", describing how to find the root of an increasing function $f : \mathbb{R} \to \mathbb{R}$ when only *noisy* estimates of the function's value at a given point are available.



1951 Robbins and Monro publish *"A Stochastic Approximation Algorithm"*, describing how to find the root of an increasing function $f : \mathbb{R} \to \mathbb{R}$ when only *noisy* estimates of the function's value at a given point are available.



They provide an iterative scheme for root estimation in this context and prove convergence of the resulting estimate in L^2 and in probability.

Note also that if we treat f as the gradient of some function F, the Robbins-Monro algorithm can be viewed a minimisation procedure.

The Robbins-Monro paper establishes stochastic approximation as an area of numerical analysis in its own right.

They provide an iterative scheme for root estimation in this context and prove convergence of the resulting estimate in L^2 and in probability.

Note also that if we treat f as the gradient of some function F, the Robbins-Monro algorithm can be viewed a minimisation procedure.

The Robbins-Monro paper establishes stochastic approximation as an area of numerical analysis in its own right.

They provide an iterative scheme for root estimation in this context and prove convergence of the resulting estimate in L^2 and in probability.

Note also that if we treat f as the gradient of some function F, the Robbins-Monro algorithm can be viewed a minimisation procedure.

The Robbins-Monro paper establishes stochastic approximation as an area of numerical analysis in its own right.

1952

Motivated by the Robbins-Monro paper the year before, Kiefer and Wolfowitz publish "*Stochastic Estimation of the Maximisation of a Regression Function*".

Their algorithm is phrased as a maximisation procedure, in contrast to the Robbins-Monro paper, and uses central-difference approximations of the gradient to update the optimum estimator.

Present day

Stochastic approximation widely researched and used in practice:

- Original applications in root-finding and optimisation, often in the guise of stochastic gradient descent:
  - Neural networks
  - K-Means
  - ...
- Related ideas are used in :
  - Stochastic variational inference (Hoffman et al. (2013))
  - Psuedo-marginal Metropolis-Hastings (Beaumont (2003), Andrieu & Roberts (2009))
  - Stochastic Gradient Langevin Dynamics (Welling & Teh (2011))

Textbooks:

Stochastic Approximation and Recursive Algorithms and Applications, Kushner & Yin (2003)

Introduction to Stochastic Search and Optimization, Spall (2003)

Modern-day formulation

Stochastic approximation is now a mature area of numerical analysis, and the general problem it seeks to solve has the following form:

$$\text{Minimise } f(w) = \mathbb{E}\left[F(w, \xi)\right]$$

(over w in some domain W, and some random variable ξ).

This is an immensely flexible framework:

- $F(w, \xi) = f(w) + \xi$ models experimental/measurement error.
- $F(x, \xi) = (w^\top \phi(\xi_X) - \xi_Y)^2$ corresponds to (least squares) linear regression
- If $f(w) = \frac{1}{N} \sum_{i=1}^{N} g(w, x_i)$ and $F(w, \xi) = \frac{1}{Ks} \sum_{x \in \xi} g(w, x)$, with ξ a randomly-selected subset (of size K) of large data set corresponds to "stochastic" machine learning algorithms.

Modern-day formulation

Stochastic approximation is now a mature area of numerical analysis, and the general problem it seeks to solve has the following form:

$$\text{Minimise } f(w) = \mathbb{E}\left[F(w, \xi)\right]$$

(over w in some domain W, and some random variable ξ).

This is an immensely flexible framework:

- $F(w, \xi) = f(w) + \xi$ models experimental/measurement error.
- $F(x, \xi) = (w^\top \phi(\xi_X) - \xi_Y)^2$ corresponds to (least squares) linear regression
- If $f(w) = \frac{1}{N} \sum_{i=1}^{N} g(w, x_i)$ and $F(w, \xi) = \frac{1}{Ks} \sum_{x \in \xi} g(w, x)$, with ξ a randomly-selected subset (of size K) of large data set corresponds to "stochastic" machine learning algorithms.

We'll focus on proof techniques for stochastic gradient descent (SGD).

We'll derive conditions for SGD to convergence almost surely. We broadly follow the structure of Léon Bottou's paper[2].

Plan:

- Continuous gradient descent
- Discrete gradient descent
- Stochastic gradient descent

[2] *Online Learning and Stochastic Approximations* - Léon Bottou (1998)

Stochastic gradient descent

We'll focus on proof techniques for stochastic gradient descent (SGD).

We'll derive conditions for SGD to convergence almost surely. We broadly follow the structure of Léon Bottou's paper[2].

Plan:

- Continuous gradient descent
- Discrete gradient descent
- Stochastic gradient descent

[2] *Online Learning and Stochastic Approximations* - Léon Bottou (1998)

Continuous Gradient Descent

Let $C : \mathbb{R}^k \longrightarrow \mathbb{R}$ be differentiable. How do solutions $s : \mathbb{R} \to \mathbb{R}^k$ to the following ODE behave?

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

Example: $C(\mathbf{x}) = 5(x_1 + x_2)^2 + (x_1 - x_2)^2$

We can analytically solve the gradient descent equation for this example:

$$\nabla C(\mathbf{x}) = (12x_1 + 8x_2, 8x_1 + 12x_2)$$

So $(s_1', s_2') = (12s_1 + 8s_2, 8s_1 + 12s_2)$, and solving with $(s_1(0), s_2(0)) = (1.5, 0.5)$ gives

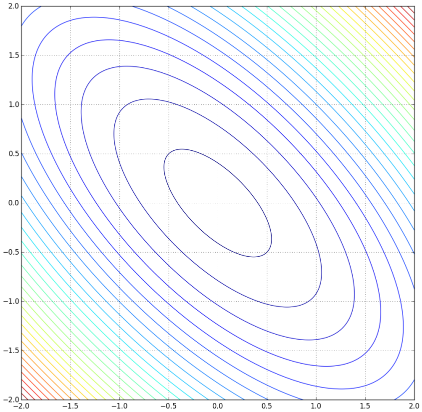
$$\begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \begin{pmatrix} e^{-20t} + 1.5e^{-4t} \\ e^{-20t} - 0.5e^{-4t} \end{pmatrix}$$

Continuous Gradient Descent

Let $C : \mathbb{R}^k \longrightarrow \mathbb{R}$ be differentiable. How do solutions $s : \mathbb{R} \to \mathbb{R}^k$ to the following ODE behave?

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

Example: $C(\mathbf{x}) = 5(x_1 + x_2)^2 + (x_1 - x_2)^2$

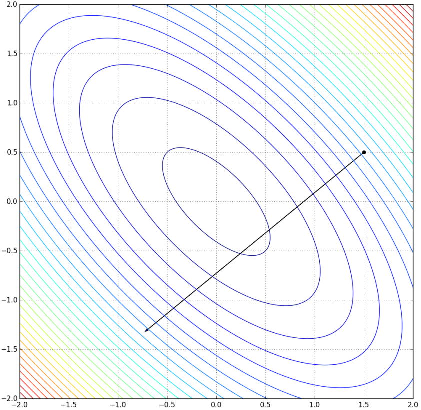


Continuous Gradient Descent

Let $C : \mathbb{R}^k \longrightarrow \mathbb{R}$ be differentiable. How do solutions $s : \mathbb{R} \to \mathbb{R}^k$ to the following ODE behave?

$$\frac{d}{dt} s(t) = -\nabla C(s(t))$$

Example: $C(\mathbf{x}) = 5(x_1 + x_2)^2 + (x_1 - x_2)^2$



Continuous Gradient Descent

Let $C : \mathbb{R}^k \longrightarrow \mathbb{R}$ be differentiable. How do solutions $s : \mathbb{R} \to \mathbb{R}^k$ to the following ODE behave?

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

Example: $C(\mathbf{x}) = 5(x_1 + x_2)^2 + (x_1 - x_2)^2$

We can analytically solve the gradient descent equation for this example:

$$\nabla C(\mathbf{x}) = (12x_1 + 8x_2, 8x_1 + 12x_2)$$

So $(s_1', s_2') = (12s_1 + 8s_2, 8s_1 + 12s_2)$, and solving with $(s_1(0), s_2(0)) = (1.5, 0.5)$ gives

$$\begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \begin{pmatrix} e^{-20t} + 1.5e^{-4t} \\ e^{-20t} - 0.5e^{-4t} \end{pmatrix}$$

Continuous Gradient Descent

Let $C : \mathbb{R}^k \longrightarrow \mathbb{R}$ be differentiable. How do solutions $s : \mathbb{R} \to \mathbb{R}^k$ to the following ODE behave?

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

Example: $C(\mathbf{x}) = 5(x_1 + x_2)^2 + (x_1 - x_2)^2$

We can analytically solve the gradient descent equation for this example:

$$\nabla C(\mathbf{x}) = (12x_1 + 8x_2, 8x_1 + 12x_2)$$

So $(s_1', s_2') = (12s_1 + 8s_2, 8s_1 + 12s_2)$, and solving with $(s_1(0), s_2(0)) = (1.5, 0.5)$ gives

$$\begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \begin{pmatrix} e^{-20t} + 1.5e^{-4t} \\ e^{-20t} - 0.5e^{-4t} \end{pmatrix}$$

Exact solution of gradient descent ODE

UNIVERSITY OF CAMBRIDGE

We'll start with some simplifying assumptions:

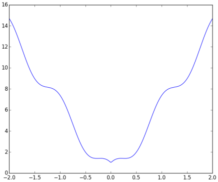
1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \quad \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$

(The second condition is weaker than convexity)

Proposition If $s : [0, \infty) \to X$ satisfies the differential equation

$$\frac{ds(t)}{dt} = -\nabla C(s(t))$$

then $s(t) \to x^\star$ as $t \to \infty$.



Example of non-convex function satisfying these conditions

We'll start with some simplifying assumptions:

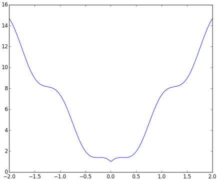
1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \quad \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$

(The second condition is weaker than convexity)

Proposition If $s : [0, \infty) \to X$ satisfies the differential equation

$$\frac{ds(t)}{dt} = -\nabla C(s(t))$$

then $s(t) \to x^\star$ as $t \to \infty$.



Example of non-convex function satisfying these conditions

Continuous Gradient Descent

We'll start with some simplifying assumptions:

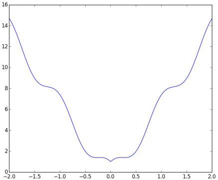
1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$

(The second condition is weaker than convexity)

Proposition If $s : [0, \infty) \to X$ satisfies the differential equation

$$\frac{ds(t)}{dt} = -\nabla C(s(t))$$

then $s(t) \to x^\star$ as $t \to \infty$.



Example of non-convex function satisfying these conditions

We'll start with some simplifying assumptions:

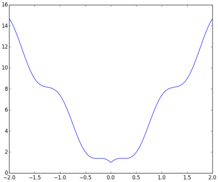
1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$

(The second condition is weaker than convexity)

Proposition If $s : [0, \infty) \to X$ satisfies the differential equation

$$\frac{ds(t)}{dt} = -\nabla C(s(t))$$

then $s(t) \to x^\star$ as $t \to \infty$.



Example of non-convex function satisfying these conditions

Continuous Gradient Descent

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

$h(t)$ **is a decreasing function**

$$\begin{aligned}\frac{d}{dt}h(t) &= \frac{d}{dt}\|s(t) - x^\star\|_2^2 \\ &= \frac{d}{dt}\left\langle s(t) - x^\star, s(t) - x^\star \right\rangle \\ &= 2\left\langle \frac{ds(t)}{dt}, s(t) - x^\star \right\rangle \\ &= -\left\langle s(t) - x^\star, \nabla C(s(t)) \right\rangle \leq 0\end{aligned}$$

Continuous Gradient Descent

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

$h(t)$ **is a decreasing function**

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

The limit of $h(t)$ must be 0

If not, there is some $K > 0$ such that $h(t) > K$ for all t. By assumption,

$$\inf_{x:h(x)>K} \langle x - x^\star, \nabla C(x) \rangle > 0$$

which means that $\inf_t h'(t) < 0$, contradicting the convergence of $h'(t)$ to 0.

Continuous Gradient Descent

1. Define Lyapunov function $h(t) = \|s(t) - x^\star\|_2^2$
2. $h(t)$ is a decreasing function
3. $h(t)$ converges to a limit, and $h'(t)$ converges to 0
4. The limit of $h(t)$ must be 0
5. $s(t) \to x^\star$

So gradient descent finds the global minimum for functions in the class stated in the theorem.

Solving general ODEs analytically is extremely difficult at best, and usually impossible.

To implement this strategy algorithmically to solve a minimisation problem, we'll need a method of solving the ODE numerically.

There is no one best way to do this!!

So gradient descent finds the global minimum for functions in the class stated in the theorem.

Solving general ODEs analytically is extremely difficult at best, and usually impossible.

To implement this strategy algorithmically to solve a minimisation problem, we'll need a method of solving the ODE numerically.

There is no one best way to do this!!

Continuous Gradient Descent

So gradient descent finds the global minimum for functions in the class stated in the theorem.

Solving general ODEs analytically is extremely difficult at best, and usually impossible.

To implement this strategy algorithmically to solve a minimisation problem, we'll need a method of solving the ODE numerically.

There is no one best way to do this!!

So gradient descent finds the global minimum for functions in the class stated in the theorem.

Solving general ODEs analytically is extremely difficult at best, and usually impossible.

To implement this strategy algorithmically to solve a minimisation problem, we'll need a method of solving the ODE numerically.

There is no one best way to do this!!

Discrete Gradient Descent

An intuitive, straightforward discretisation of

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

is

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_n) \qquad (ForwardEuler)$$

Although it is easy and quick to implement, it has theoretical (A-)instability issues.

An alternative method which is (A-)stable is the implicit Euler method:

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_{n+1})$$

So why settle for explicit discretisation? Implicit requires solution of non-linear system at each step, and practically explicit discretisation works.

An intuitive, straightforward discretisation of

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

is

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_n) \qquad (ForwardEuler)$$

Although it is easy and quick to implement, it has theoretical (A-)instability issues.

An alternative method which is (A-)stable is the implicit Euler method:

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_{n+1})$$

So why settle for explicit discretisation? Implicit requires solution of non-linear system at each step, and practically explicit discretisation works.

Discrete Gradient Descent

An intuitive, straightforward discretisation of

$$\frac{d}{dt}s(t) = -\nabla C(s(t))$$

is

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_n) \qquad (ForwardEuler)$$

Although it is easy and quick to implement, it has theoretical (A-)instability issues.

An alternative method which is (A-)stable is the implicit Euler method:

$$\frac{s_{n+1} - s_n}{\epsilon_n} = -\nabla C(s_{n+1})$$

So why settle for explicit discretisation? Implicit requires solution of non-linear system at each step, and practically explicit discretisation works.

Also have multistep methods, e.g.:

$$s_{n+2} = s_{n+1} + \epsilon_{n+2}\left(\frac{3}{2}\nabla C(s_{n+1}) - \frac{1}{2}\nabla C(s_n)\right)$$

This is the 2nd-order Adams-Bashforth method.

Similar (in)stability properties to forward Euler, but discretisation error of a smaller order of magnitude at each step ($\mathcal{O}(\epsilon^3)$ compared with $\mathcal{O}(\epsilon^2)$)

Discrete Gradient Descent

Examples of discretisation with $\epsilon = 0.1$

Examples of discretisation with $\epsilon = 0.07$

Discrete Gradient Descent

Examples of discretisation with $\epsilon = 0.01$

Proposition If $s_{n+1} = s_n - \epsilon_n \nabla C(s_n)$ and C satisfies

1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x-x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$
3. $\|\nabla C(x)\|_2^2 \leq A + B\|x - x^\star\|_2^2$ for some $A, B \geq 0$

then subject to $\sum_n \epsilon_n = \infty$ and $\sum_n \epsilon_n^2 < \infty$ we have $s_n \to x^\star$

Proof structure The proof is largely the same as for the continuous case, but the extra conditions in the statement deal with the errors introduced by discretisation.

UNIVERSITY OF CAMBRIDGE

Proposition If $s_{n+1} = s_n - \epsilon_n \nabla C(s_n)$ and C satisfies

1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$
3. $\|\nabla C(x)\|_2^2 \leq A + B\|x - x^\star\|_2^2$ for some $A, B \geq 0$

then subject to $\sum_n \epsilon_n = \infty$ and $\sum_n \epsilon_n^2 < \infty$ we have $s_n \to x^\star$

Proof structure The proof is largely the same as for the continuous case, but the extra conditions in the statement deal with the errors introduced by discretisation.

Discrete Gradient Descent

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^\infty h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

UNIVERSITY OF CAMBRIDGE

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
Because of the error that the discretisation introduced, h_n is not guaranteed to be decreasing - have to adjust proof.

UNIVERSITY OF CAMBRIDGE

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
Because of the error that the discretisation introduced, h_n is not guaranteed to be decreasing - have to adjust proof.

Idea: h_n may fluctuate, but if we can show that the cumulative 'up' movements aren't too big, we can still prove convergence of h_n.

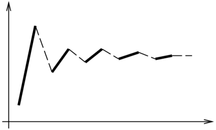


Illustration credit: *Online learning and stochastic approximation*, Léon Bottou (1998)

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^\infty h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
With some algebra, note that

$$\begin{aligned} h_{n+1} - h_n &= \langle s_{n+1} - x^\star, s_{n+1} - x^\star \rangle - \langle s_n - x^\star, s_n - x^\star \rangle \\ &= \langle s_{n+1}, s_{n+1} \rangle - \langle s_n, s_n \rangle - 2 \langle s_{n+1} - s_n, x^\star \rangle \\ &= \langle s_n - \epsilon_n \nabla C(s_n), s_n - \epsilon_n \nabla C(s_n) \rangle - \langle s_n, s_n \rangle + 2\epsilon_n \langle \nabla C(s_n), x^\star \rangle \\ &= -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n) \rangle + \epsilon_n^2 \|\nabla C(s_n)\|_2^2 \end{aligned}$$

UNIVERSITY OF CAMBRIDGE

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
Assuming $\|\nabla C(x)\|_2^2 \le A + B\|x - x^\star\|_2^2$, we get

$$\begin{aligned} h_{n+1} - h_n &\le -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n)\rangle + \epsilon_n^2(A + Bh_n) \\ \implies h_{n+1} - (1 + \epsilon_n^2 B)h_n &\le -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n)\rangle + \epsilon_n^2 A \\ &\le \epsilon_n^2 A \end{aligned}$$

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^\infty h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Show that $\sum_{n=1}^\infty h_n^+ < \infty$
Assuming $\|\nabla C(x)\|_2^2 \leq A + B\|x - x^\star\|_2^2$, we get

$$h_{n+1} - (1 - \epsilon_n^2 B) h_n \leq \epsilon_n^2 A$$

Writing $\mu_n = \prod_{i=1}^n \frac{1}{1+\epsilon_i^2 B}$, and $h_n' = \mu_n h_n$, we get

$$h_{n+1}' - h_n' \leq \epsilon_n^2 \mu_n A$$

Assuming $\sum_{n=1}^\infty \epsilon_n^2 < \infty$, μ_n converges away from 0, so RHS is summable.

UNIVERSITY OF CAMBRIDGE

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^\infty h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Show that this implies that h_n converges

If $\sum_{n=1}^\infty \max(0, h_{n+1} - h_n) < \infty$, then $\sum_{n=1}^\infty \min(0, h_{n+1} - h_n) < \infty$ (why?)

But $h_{n+1} = h_0 + \sum_{k=1}^n (\max(0, h_{k+1} - h_k) + \min(0, h_{k+1} - h_k))$

So $(h_n)_{n=1}^\infty$ converges.

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Show that h_n must converge to 0 Assume h_n converges to some positive number. Previously, we had

$$h_{n+1} - h_n = -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n) \rangle + \epsilon_n^2 \|\nabla C(s_n)\|_2^2$$

This is summable, and if we assume further that $\sum_{n=1}^{\infty} \epsilon_n = \infty$, then we get

$$\langle s_n - x^\star, \nabla C(s_n) \rangle \to 0$$

contradicting h_n converging away from 0.

Discrete Gradient Descent

1. Define Lyapunov sequence $h_n = \|s_n - x^\star\|_2^2$
2. Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
3. Show that $\sum_{n=1}^{\infty} h_n^+ < \infty$
4. Show that this implies that h_n converges
5. Show that h_n must converge to 0
6. $s_n \to x^\star$

Stochastic Gradient Descent

We're now ready to introduce the stochastic approximation of the gradient into our algorithm.

If C is a cost function averaged across a data set, often have true gradient of the form

$$\nabla C(x) = \frac{1}{N} \sum_{n=1}^{N} f_n(x)$$

and an approximation is formed by subsampling:

$$\widehat{\nabla C}(x) = \frac{1}{K} \sum_{k \in I_K} f_k(x) \qquad (I_k \sim \mathit{Unif}(\text{subsets of size K}))$$

We'll treat the more general case where the gradient estimates $\widehat{\nabla C}(x)$ are unbiased and independent.

The introduced randomness means that the Lyapunov sequence in our proof will now be a stochastic process, and we'll need some additional machinery to deal with it.

Stochastic Gradient Descent

We're now ready to introduce the stochastic approximation of the gradient into our algorithm.

If C is a cost function averaged across a data set, often have true gradient of the form

$$\nabla C(x) = \frac{1}{N} \sum_{n=1}^{N} f_n(x)$$

and an approximation is formed by subsampling:

$$\widehat{\nabla C}(x) = \frac{1}{K} \sum_{k \in I_K} f_k(x) \qquad (I_k \sim Unif(\text{subsets of size K}))$$

We'll treat the more general case where the gradient estimates $\widehat{\nabla C}(x)$ are unbiased and independent.

The introduced randomness means that the Lyapunov sequence in our proof will now be a stochastic process, and we'll need some additional machinery to deal with it.

Stochastic Gradient Descent

We're now ready to introduce the stochastic approximation of the gradient into our algorithm.

If C is a cost function averaged across a data set, often have true gradient of the form

$$\nabla C(x) = \frac{1}{N}\sum_{n=1}^{N} f_n(x)$$

and an approximation is formed by subsampling:

$$\widehat{\nabla C}(x) = \frac{1}{K}\sum_{k \in I_K} f_k(x) \qquad (I_k \sim \mathit{Unif}(\text{subsets of size K}))$$

We'll treat the more general case where the gradient estimates $\widehat{\nabla C}(x)$ are unbiased and independent.

The introduced randomness means that the Lyapunov sequence in our proof will now be a stochastic process, and we'll need some additional machinery to deal with it.

Let $(X_n)_{n\geq 0}$ be a stochastic process.

$\mathcal{F}_n$ denotes "the information describing the stochastic process up to time n" denoted mathematically by $\mathcal{F}_n = \sigma(X_m | m \leq n)$

Definition A stochastic process $(X_n)_{n=0}^{\infty}$ is a martingale[3] if

- $\mathbb{E}[|X_n|] < \infty$ for all n
- $\mathbb{E}[X_n | \mathcal{F}_m] = X_m$ for $n \geq m$

Martingale convergence theorem (Doob, 1953) If $(X_n)_{n=1}^{\infty}$ is a martingale, and $\sup \mathbb{E}[|X_n|] < \infty$, then $X_n \to X_\infty$ almost surely.

Quasimartingale convergence theorem (Fisk, 1965) If $(X_n)_{n=1}^{\infty}$ is a positive stochastic process, and

$$\sum_{n=1}^{\infty} \mathbb{E}\left[(\mathbb{E}[X_{n+1}|\mathcal{F}_n] - X_n)\, \mathbb{1}_{\{\mathbb{E}[X_{n+1}|\mathcal{F}_n] - X_n > 0\}}\right] < \infty$$

then $X_n \to X_\infty$ almost surely.

[3] on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_n)_{n=0}^{\infty}, \mathbb{P})$, with $\mathcal{F}_n = \sigma(X_m | m \leq n) \forall n$

Let $(X_n)_{n \geq 0}$ be a stochastic process.

$\mathcal{F}_n$ denotes "the information describing the stochastic process up to time n" denoted mathematically by $\mathcal{F}_n = \sigma(X_m | m \leq n)$

Definition A stochastic process $(X_n)_{n=0}^{\infty}$ is a martingale[3] if

- $\mathbb{E}[|X_n|] < \infty$ for all n
- $\mathbb{E}[X_n | \mathcal{F}_m] = X_m$ for $n \geq m$

Martingale convergence theorem (Doob, 1953) If $(X_n)_{n=1}^{\infty}$ is a martingale, and $\sup \mathbb{E}[|X_n|] < \infty$, then $X_n \to X_\infty$ almost surely.

Quasimartingale convergence theorem (Fisk, 1965) If $(X_n)_{n=1}^{\infty}$ is a positive stochastic process, and

$$\sum_{n=1}^{\infty} \mathbb{E}\left[\left(\mathbb{E}[X_{n+1}|\mathcal{F}_n] - X_n\right) \mathbb{1}_{\{\mathbb{E}[X_{n+1}|\mathcal{F}_n] - X_n > 0\}}\right] < \infty$$

then $X_n \to X_\infty$ almost surely.

[3] on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_n)_{n=0}^{\infty}, \mathbb{P})$, with $\mathcal{F}_n = \sigma(X_m | m \leq n) \forall n$

Let $(X_n)_{n \geq 0}$ be a stochastic process.

$\mathcal{F}_n$ denotes "the information describing the stochastic process up to time n" denoted mathematically by $\mathcal{F}_n = \sigma(X_m | m \leq n)$

Definition A stochastic process $(X_n)_{n=0}^{\infty}$ is a martingale[3] if

- $\mathbb{E}\left[|X_n|\right] < \infty$ for all n
- $\mathbb{E}\left[X_n | \mathcal{F}_m\right] = X_m$ for $n \geq m$

Martingale convergence theorem (Doob, 1953) If $(X_n)_{n=1}^{\infty}$ is a martingale, and $\sup \mathbb{E}\left[|X_n|\right] < \infty$, then $X_n \to X_\infty$ almost surely.
Quasimartingale convergence theorem (Fisk, 1965) If $(X_n)_{n=1}^{\infty}$ is a positive stochastic process, and

$$\sum_{n=1}^{\infty} \mathbb{E}\left[\left(\mathbb{E}\left[X_{n+1}|\mathcal{F}_n\right] - X_n\right) \mathbb{1}_{\{\mathbb{E}[X_{n+1}|\mathcal{F}_n] - X_n > 0\}}\right] < \infty$$

then $X_n \to X_\infty$ almost surely

[3] on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_n)_{n=0}^{\infty}, \mathbb{P})$, with $\mathcal{F}_n = \sigma(X_m | m \leq n) \forall n$

UNIVERSITY OF CAMBRIDGE

Proposition If $s_{n+1} = s_n - \epsilon_n H_n(s_n)$, with $H_n(s_n)$ an unbiased estimator for $\nabla C(s_n)$, and C satisfies

1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x-x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$
3. $\mathbb{E}\left[\|H_n(x)\|_2^2\right] \leq A + B\|x - x^\star\|_2^2$ for some $A, B \geq 0$ independent of n

then subject to $\sum_n \epsilon_n = \infty$ and $\sum_n \epsilon_n^2 < \infty$ we have $s_n \to x^\star$

Proof structure The proof is largely the same as for the deterministic discrete case, but the extra conditions in the statement deal with the control we need over the unbiased gradient estimators.

Proposition If $s_{n+1} = s_n - \epsilon_n H_n(s_n)$, with $H_n(s_n)$ an unbiased estimator for $\nabla C(s_n)$, and C satisfies

1. C has a unique minimiser $x^\star$
2. $\forall \epsilon > 0, \inf_{\|x - x^\star\|_2^2 > \epsilon} \langle x - x^\star, \nabla C(x) \rangle > 0$
3. $\mathbb{E}\left[\|H_n(x)\|_2^2\right] \leq A + B\|x - x^\star\|_2^2$ for some $A, B \geq 0$ independent of n

then subject to $\sum_n \epsilon_n = \infty$ and $\sum_n \epsilon_n^2 < \infty$ we have $s_n \to x^\star$

Proof structure The proof is largely the same as for the deterministic discrete case, but the extra conditions in the statement deal with the control we need over the unbiased gradient estimators.

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

Consider the positive variations $h_n^+ = \max(0, h_{n+1} - h_n)$
By exactly the same calculation as in the deterministic discrete case:

$$h_{n+1} - h_n = -2\epsilon_n \langle s_n - x^\star, H_n(s_n) \rangle + \epsilon_n^2 \|H_n(s_n)\|_2^2$$

So

$$\mathbb{E}\left[h_{n+1} - h_n | \mathcal{F}_n\right] = -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n) \rangle + \epsilon_n^2 \mathbb{E}\left[\|H_n(s_n)\|_2^2 | \mathcal{F}_n\right]$$

UNIVERSITY OF CAMBRIDGE

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

Show that h_n converges almost surely

Exactly the same as for discrete gradient descent:

- Assume $\mathbb{E}\left[\|H_n(x)\|_2^2\right] \leq A + B\|x - x^\star\|_2^2$
- Introduce $\mu_n = \prod_{i=1}^n \frac{1}{1+\epsilon_i^2 B}$ and $h'_n = \mu_n h_n$

Get:

$$\mathbb{E}\left[h'_{n+1} - h'_n \middle| \mathcal{F}_n\right] \leq \epsilon_n^2 \mu_n A$$

$$\Longrightarrow \mathbb{E}\left[(h'_{n+1} - h'_n)\mathbb{1}_{\mathbb{E}\left[h'_{n+1} - h'_n \middle| \mathcal{F}_n\right] > 0} \middle| \mathcal{F}_n\right] \leq \epsilon_n^2 \mu_n A$$

$\sum_{n=1}^\infty \epsilon_n^2 < \infty \implies$ Quasimartingale convergence $\implies (h'_n)_{n=1}^\infty$ converges a.s. $\implies (h_n)_{n=1}^\infty$ converges a.s.

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

Show that h_n must converge to 0 almost surely

From previous calculations:

$$\mathbb{E}\left[h_{n+1} - (1 - \epsilon_n^2 B)h_n \middle| \mathcal{F}_n\right] = -2\epsilon_n \langle s_n - x^\star, \nabla C(s_n)\rangle + \epsilon_n^2 A$$

$(h_n)_{n=1}^\infty$ converges, so sequence is summable a.s. . Assume $\sum_{n=1}^\infty \epsilon_n^2 < \infty$, so right term is summable a.s., so left term side is also summable a.s. :

$$\sum_{n=1}^\infty \epsilon_n \langle s_n - x^\star, \nabla C(s_n)\rangle < \infty \text{almost surely}$$

If we assume in addition that $\sum_{n=1}^\infty \epsilon_n = \infty$, this forces

$$\langle s_n - x^\star, \nabla C(s_n)\rangle \to 0 \text{almost surely}$$

And by our initial assumptions about C, $(h_n)_{n=1}^\infty$ must converge to 0 almost

Stochastic Gradient Descent

1. Define Lyapunov process $h_n = \|s_n - x^\star\|_2^2$
2. Consider the variations $h_{n+1} - h_n$
3. Show that h_n converges almost surely
4. Show that h_n must converge to 0 almost surely
5. $s_n \to x^\star$

Relaxing the pseudo-convexity conditions

Often the conditions on C in the preceding theorems are not satisfied in practice. One possible way of extending ideas above: Assume:

- C is a non-negative, three-times differentiable function.
- Robbins-Monro learning rate conditions hold: $\sum_{n=1}^{\infty} \epsilon_n = \infty$, $\sum_{n=1}^{\infty} \epsilon_n^2 < \infty$
- Gradient estimate H satisfies: $\mathbb{E}\left[\|H_n(x)\|^k\right] \leq A_k + B_k\|x\|^k$ for $k = 2, 3, 4$.
- There exists $D > 0$ such that $\inf_{\|x\|^2 > D} \langle x, \nabla C(x) \rangle > 0$

Idea for proof is then:

1. For a given start position, the sequence $(s_n)_{n=1}^{\infty}$ is confined to a bounded neighbourhood of 0 almost surely.
2. Introduce the Lyapunov function $h_n = C(s_n)$, and prove its almost-sure convergence.
3. Prove that $\nabla C(s_n)$ necessarily converges almost surely.

Note that this guarantees we settle at some critical point for the function (which may be a maximum, minimum, or saddle), rather than reaching the global optimum.

Often the conditions on C in the preceding theorems are not satisfied in practice. One possible way of extending ideas above: Assume:

- C is a non-negative, three-times differentiable function.
- Robbins-Monro learning rate conditions hold: $\sum_{n=1}^{\infty} \epsilon_n = \infty$, $\sum_{n=1}^{\infty} \epsilon_n^2 < \infty$
- Gradient estimate H satisfies: $\mathbb{E}\left[\|H_n(x)\|^k\right] \leq A_k + B_k\|x\|^k$ for $k = 2, 3, 4$.
- There exists $D > 0$ such that $\inf_{\|x\|^2 > D} \langle x, \nabla C(x) \rangle > 0$

Idea for proof is then:

1. For a given start position, the sequence $(s_n)_{n=1}^{\infty}$ is confined to a bounded neighbourhood of 0 almost surely.
2. Introduce the Lyapunov function $h_n = C(s_n)$, and prove its almost-sure convergence.
3. Prove that $\nabla C(s_n)$ necessarily converges almost surely.

Note that this guarantees we settle at some critical point for the function (which may be a maximum, minimum, or saddle), rather than reaching the global optimum.

Motivation

- So far we've told you why SGD is "safe" :)
- ...but Robbins-Monro is just a sufficient condition
- ...then how to choose learning rates to achieve
  - fast convergence
  - better local optimum

Motivation (cont.)

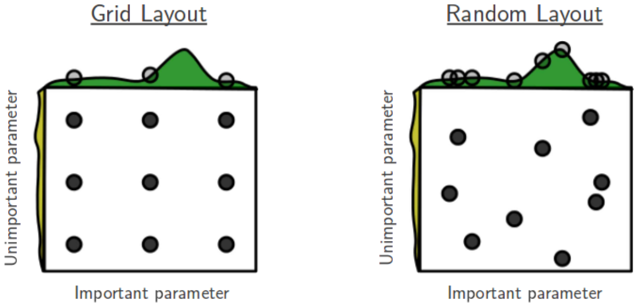


Figure: from [Bergstra and Bengio 2012]

- idea 1: search the best learning rate schedule
  - grid search, random search, cross validation
  - Bayesian optimization
- But I'm lazy and I don't want to spend too much time

Motivation (cont.)

- idea 2: let the learning rates adapt themselves
  - pre-conditioning with $|\texttt{diag}(H)|$ [Becker and LeCun 1988]
  - adaGrad [Duchi et al. 2010]
  - adaDelta [Zeiler 2012], RMSprop [Tieleman and Hinton 2012], ADAM [Kingma and Ba 2014]
  - Equilibrated SGD [Dauphin et al. 2015] (this year's NIPS!)
- now we'll talk a bit about online learning
  - a good tutorial [Shalev-Shwartz 2012]
  - milestone paper [Zinkevich 2003]
- ... and some practical comparisons

From batch to online learning

- Batch learning often assumes:
  - We've got the whole dataset $\mathcal{D}$
  - the cost function $C(\mathbf{w}; \mathcal{D}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[c(\mathbf{w}; \mathbf{x})]$
- SGD/mini-batch learning accelerate training in real time by
  - processing one/a mini-batch of datapoint each iteration
  - considering gradients with data point cost functions

$$\nabla C(\mathbf{w}; \mathcal{D}) \approx \frac{1}{M} \sum_{m=1}^{M} \nabla c(\mathbf{w}; \mathbf{x}_m)$$

From batch to online learning (cont.)

Let's forget the cost function on the batch for a moment:

- online gradient descent (OGD):
  - each iteration t we receive a loss $f_t(\mathbf{w})$
  - ...and a (noisy) (sub-)gradient $\mathbf{g}_t \in \partial f_t(\mathbf{w})$
  - ...then we update the weights with $\mathbf{w} \leftarrow \mathbf{w} - \eta \mathbf{g}_t$
- for SGD the received gradient is defined by

$$\mathbf{g}_t = \frac{1}{M} \sum_{m=1}^{M} \nabla c(\mathbf{w}; \mathbf{x}_m)$$

From batch to online learning (cont.)

From batch to online learning (cont.)

- Online learning assumes
  - each iteration t we receive a loss $f_t(\mathbf{w})$
    (in batch learning context $f_t(\mathbf{w}) = c(\mathbf{w}; \mathbf{x}_t)$)
  - ...then we learn $\mathbf{w}_{t+1}$ following some rules
- Performance is measured by **regret**

$$R_T^* = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \inf_{\mathbf{w} \in \mathcal{S}} \sum_{t=1}^{T} f_t(\mathbf{w}) \tag{1}$$

- General definition $R_T(\mathbf{u}) = \sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{u})]$
- SGD $\Leftrightarrow$ "follow the regularized leader" with L_2 regularization

From batch to online learning (cont.)

Follow the Leader (FTL)

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{S}} \sum_{i=1}^{t} f_i(\mathbf{w}) \tag{2}$$

Lemma (upper bound of regret)

Let $\mathbf{w}_1$, $\mathbf{w}_2$,... be the sequence produced by FTL, then $\forall \mathbf{u} \in \mathcal{S}$,

$$R_T(\mathbf{u}) = \sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{u})] \leq \sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t+1})] \,. \tag{3}$$

From batch to online learning (cont.)

A game that fools FTL

Let $w \in \mathcal{S} = [-1, 1]$ and the loss function at time t

$$f_t(w) = \begin{cases} -0.5w, & t = 1 \\ w, & t \textit{ is even} \\ -w, & t > 1 \textit{ and } t \textit{ is odd} \end{cases}$$

- FTL is easily fooled!

From batch to online learning (cont.)

`Follow the Regularized Leader (FTRL)`

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{i=1}^{t} f_i(\mathbf{w}) + \varphi(\mathbf{w}) \tag{4}$$

Lemma (upper bound of regret)

Let $\mathbf{w}_1$, $\mathbf{w}_2$,... *be the sequence produced by FTRL, then* $\forall \mathbf{u} \in \mathcal{S}$,

$$\sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{u})] \leq \varphi(\mathbf{u}) - \varphi(\mathbf{w}_1) + \sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t+1})] \, .$$

From batch to online learning (cont.)

- Let's assume the loss f_t are convex functions:

$$\forall \mathbf{w}, \mathbf{u} \in \mathcal{S}: \quad f_t(\mathbf{u}) - f_t(\mathbf{w}) \geq \langle \mathbf{u} - \mathbf{w}, \mathbf{g}(\mathbf{w}) \rangle, \forall \mathbf{g}(\mathbf{w}) \in \partial f_t(\mathbf{w})$$

- use linearisation $\tilde{f}_t(\mathbf{w}) = f_t(\mathbf{w}_t) + \langle \mathbf{w}, \mathbf{g}(\mathbf{w}) \rangle$ as a surrogate:

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - f_t(\mathbf{u}) \leq \sum_{t=1}^{T} \langle \mathbf{w}_t, \mathbf{g}_t \rangle - \langle \mathbf{u}, \mathbf{g}_t \rangle, \quad \forall \mathbf{g}_t \in \partial f_t(\mathbf{w}_t)$$

From batch to online learning (cont.)

Online Gradient Descent (OGD)

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t, \quad \mathbf{g}_t \in \partial f_t(\mathbf{w}_t) \tag{5}$$

- From FTRL to online gradient descent:
  - use linearisation as surrogate loss (upper bound LHS)
  - use L_2 regularizer $\varphi(\mathbf{w}) = \frac{1}{2\eta}||\mathbf{w} - \mathbf{w}_1||_2^2$
  - apply FTRL regret bound to the surrogate loss

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{S}} \sum_{i=1}^{t} \langle \mathbf{w}, \mathbf{g}_i \rangle + \varphi(\mathbf{w})$$

From batch to online learning (cont.)

Online Gradient Descent (OGD)

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t, \quad \mathbf{g}_t \in \partial f_t(\mathbf{w}_t) \tag{5}$$

Theorem (regret bound for online gradient descent)

Assume we run online gradient descent on convex loss functions f_1, f_2, ... f_T with regularizer $\varphi(\mathbf{w}) = \frac{1}{2\eta}||\mathbf{w} - \mathbf{w}_1||_2^2$. then for all $\mathbf{u} \in \mathcal{S}$,

$$R_T(\mathbf{u}) \leq \frac{1}{2\eta}||\mathbf{u} - \mathbf{w}_1||_2^2 + \eta \sum_{t=1}^{T} ||\mathbf{g}_t||_2^2, \quad \mathbf{g}_t \in \partial f_t(\mathbf{w}_t).$$

From batch to online learning (cont.)

Theorem (regret bound for online gradient descent)

Assume we run online gradient descent on convex loss functions f_1, f_2, ... f_T with regularizer $\varphi(\mathbf{w}) = \frac{1}{2\eta}||\mathbf{w} - \mathbf{w}_1||_2^2$. then for all $\mathbf{u} \in \mathcal{S}$,

$$R_T(\mathbf{u}) \leq \frac{1}{2\eta}||\mathbf{u} - \mathbf{w}_1||_2^2 + \eta \sum_{t=1}^{T} ||\mathbf{g}_t||_2^2, \quad \mathbf{g}_t \in \partial f_t(\mathbf{w}_t).$$

- keep in mind for later discussion of adaGrad:
  - I want φ to be strongly convex wrt. (semi-)norm $||\cdot||$
  - ...then the L_2 norm will be changed to $||\cdot||_*$
  - ...and we use Hölder's inequality for proof

adaGrad [Duchi et al. 2010]

Use **proximal** terms ψ_t!

- ψ_t is a strongly-convex function and

$$B_{\psi_t}(\mathbf{w}, \mathbf{w}_t) = \psi_t(\mathbf{w}) - \psi_t(\mathbf{w}_t) - \langle \nabla \psi_t(\mathbf{w}_t), \mathbf{w} - \mathbf{w}_t \rangle$$

- Primal-dual sub-gradient update:

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w}} \eta \langle \mathbf{w}, \bar{\mathbf{g}}_t \rangle + \eta \varphi(\mathbf{w}) + \frac{1}{t} \psi_t(\mathbf{w}). \tag{6}$$

- Proximal gradient/composite mirror descent:

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w}} \eta \langle \mathbf{w}, \mathbf{g}_t \rangle + \eta \varphi(\mathbf{w}) + B_{\psi_t}(\mathbf{w}, \mathbf{w}_t). \tag{7}$$

adaGrad [Duchi et al. 2010] (cont.)

adaGrad with diagonal matrices

$$G_t = \sum_{i=1}^{t} \mathbf{g}_t \mathbf{g}_t^T, \quad H_t = \delta I + \mathtt{diag}(G_t^{1/2}).$$

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{S}} \eta \langle \mathbf{w}, \bar{\mathbf{g}}_t \rangle + \eta \varphi(\mathbf{w}) + \frac{1}{2t} \mathbf{w}^T H_t \mathbf{w}. \tag{8}$$

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{S}} \eta \langle \mathbf{w}, \mathbf{g}_t \rangle + \eta \varphi(\mathbf{w}) + \frac{1}{2} (\mathbf{w} - \mathbf{w}_t)^T H_t (\mathbf{w} - \mathbf{w}_t). \tag{9}$$

- To get adaGrad from online gradient descent:
  - add a proximal term or a Bregman divergence to the objective
  - adapt the proximal term through time:

$$\psi_t(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T H_t \mathbf{w}$$

adaGrad [Duchi et al. 2010] (cont.)

- example: composite mirror descent on $\mathcal{S} = \mathbb{R}^D$, with $\varphi(\mathbf{w}) = \mathbf{0}$

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w}} \eta\langle \mathbf{w}, \mathbf{g}_t \rangle + \frac{1}{2}(\mathbf{w} - \mathbf{w}_t)^T[\delta I + \mathtt{diag}(G_t)](\mathbf{w} - \mathbf{w}_t)$$

$$\Rightarrow \quad \mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta \mathbf{g}_t}{\delta + \sqrt{\sum_{i=1}^t \mathbf{g}_i^2}} \tag{10}$$

adaGrad [Duchi et al. 2010] (cont.)

WHY???

- time-varying learning rate vs. time-varying proximal term
- I want the contribution of $||\mathbf{g}_t||_*^2$ induced by ψ_t be upper bounded by $||\mathbf{g}_t||_2$:
  - define $||\cdot||_{\psi_t} = \sqrt{\langle \cdot, H_t \cdot \rangle}$
  - $\psi_t(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T H_t \mathbf{w}$ is strongly convex wrt. $||\cdot||_{\psi_t}$
  - a little math can show that

$$\sum_{t=1}^{T} ||\mathbf{g}_t||_{\psi_t *}^2 \leq 2 \sum_{d=1}^{D} ||\mathbf{g}_{1:T,d}||_2$$

adaGrad [Duchi et al. 2010] (cont.)

Theorem (Thm. 5 in the paper)

Assume sequence $\{\mathbf{w}_t\}$ *is generated by adaGrad using primal-dual update (8) with* $\delta \geq \max_t ||\mathbf{g}_t||_\infty$, *then for any* $\mathbf{u} \in \mathcal{S}$,

$$R_T(\mathbf{u}) \leq \frac{\delta}{\eta}||\mathbf{u}||_2^2 + \frac{1}{\eta}||\mathbf{u}||_\infty^2 \sum_{d=1}^{D} ||\mathbf{g}_{1:T,d}||_2 + \eta \sum_{d=1}^{D} ||\mathbf{g}_{1:T,d}||_2.$$

For $\{\mathbf{w}_t\}$ *generated using composite mirror descent update (9),*

$$R_T(\mathbf{u}) \leq \frac{1}{2\eta} \max_{t \leq T} ||\mathbf{u} - \mathbf{w}_t||_2^2 \sum_{d=1}^{D} ||\mathbf{g}_{1:T,d}||_2 + \eta \sum_{d=1}^{D} ||\mathbf{g}_{1:T,d}||_2.$$

Improving adaGrad

- possible problems of adaGrad:
  - global learning rate η still need hand tuning
  - sensitive to initial conditions
- possible improving directions:
  - use truncated sum / running average
  - use (approximate) second-order information
  - add momentum
  - correct the bias of running average
- existing methods:
  - RMSprop [Tieleman and Hinton 2012]
  - adaDelta [Zeiler 2012]
  - ADAM [Kingma and Ba 2014]

Improving adaGrad (cont.)

RMSprop

$$G_t = \rho G_{t-1} + (1-\rho)\mathbf{g}_t\mathbf{g}_t^T, \quad H_t = (\delta I + \mathtt{diag}(G_t))^{1/2}.$$

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \eta\langle \mathbf{w}, \mathbf{g}_t\rangle + \frac{1}{2}(\mathbf{w}-\mathbf{w}_t)^T H_t(\mathbf{w}-\mathbf{w}_t).$$

$$\Rightarrow \quad \mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta \mathbf{g}_t}{\mathrm{RMS}[\mathbf{g}]_t}, \quad \mathrm{RMS}[\mathbf{g}]_t = \mathtt{diag}(H_t) \quad (11)$$

- possible improving directions:
  - **use truncated sum / running average**
  - use (approximate) second-order information
  - add momentum
  - correct the bias of running average

Improving adaGrad (cont.)

$$\begin{aligned}
\Delta \mathbf{w} \propto H^{-1}\mathbf{g} &\propto \frac{\partial f / \partial \mathbf{w}}{\partial^2 f / \partial \mathbf{w}^2} \\
\Rightarrow \quad \partial^2 f / \partial \mathbf{w}^2 &\propto \frac{\partial f / \partial \mathbf{w}}{\Delta \mathbf{w}} \\
\Rightarrow \quad \mathtt{diag}(H) &\approx \frac{\mathrm{RMS}[\mathbf{g}]_t}{\mathrm{RMS}[\Delta \mathbf{w}]_{t-1}}
\end{aligned}$$

- possible improving directions:
    - use truncated sum / running average
    - **use (approximate) second-order information**
    - add momentum
    - correct the bias of running average

Improving adaGrad (cont.)

adaDelta

$$\mathtt{diag}(H_t) = \frac{\mathrm{RMS}[\mathbf{g}]_t}{\mathrm{RMS}[\Delta \mathbf{w}]_{t-1}}$$

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}}{\mathrm{argmin}} \langle \mathbf{w}, \mathbf{g}_t \rangle + \frac{1}{2}(\mathbf{w} - \mathbf{w}_t)^T H_t (\mathbf{w} - \mathbf{w}_t).$$

$$\Rightarrow \quad \mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\mathrm{RMS}[\Delta \mathbf{w}]_{t-1}}{\mathrm{RMS}[\mathbf{g}]_t} \mathbf{g}_t \tag{12}$$

- possible improving directions:
  - use truncated sum / running average
  - **use (approximate) second-order information**
  - add momentum
  - correct the bias of running average

Improving adaGrad (cont.)

ADAM

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t, \quad \mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1-\beta_2)\mathbf{g}_t^2.$$

$$\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \quad \hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t}.$$

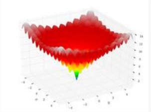
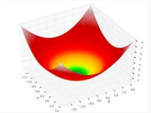
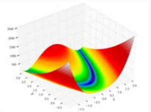
$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta \hat{\mathbf{m}}_t}{\hat{\mathbf{v}}_t + \delta}. \tag{13}$$

- possible improving directions:
  - use truncated sum / running average
  - use (approximate) second-order information
  - **add momentum**
  - **correct the bias of running average**

Demo time!

wikipedia page “test functions for optimization”
https://en.wikipedia.org/wiki/Test_functions_for_optimization

Test functions for single-objective optimization problems [edit]

Name	Plot	Formula
Ackley's function:		$f(x,y) = -20\exp\left(-0.2\sqrt{0.5\left(x^2+y^2\right)}\right) - \exp\left(0.5\left(\cos\left(2\pi x\right)+\cos\left(2\pi y\right)\right)\right) + e + 20$
Sphere function		$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$
Rosenbrock function		$f(\boldsymbol{x}) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2 + \left(x_i-1\right)^2\right]$
		

Learn the learning rates

- idea 3: learn the (global) learning rates as well!

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t \quad \Rightarrow \quad \mathbf{w}_t = \mathbf{w}(t, \eta, \mathbf{w}_1, \{\mathbf{g}_i\}_{i \leq t-1})$$

- $\mathbf{w}_t$ is a function of η
- learn η (with gradient descent) by

$$\eta = \text{argmin} \sum_{s \leq t} f_s(\mathbf{w}(s, \eta))$$

- Reverse-mode differentiation to back-track the learning process [Maclaurin et al. 2015]
- stochastic gradient descent to learn η on the fly [Massé and Ollivier 2015]

Summary

- Recent successes of machine learning rely on stochastic approximation and optimisation methods
- Theoretical analysis guarantees good behaviour
- Adaptive learning rates provides good performance in practice
- Future work will reduce labour on tuning hyper-parameters

References

Bergstra J, Bengio Y. Random search for hyper-parameter optimization. The Journal of Machine Learning Research, 2012, 13(1): 281-305.

S. Becker and Y. LeCun. Improving the convergence of back-propagation learning with second order methods. Tech. Rep., Department of Computer Science, University of Toronto, Toronto, ON, Canada, 1988.

Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research, 2011, 12: 2121-2159.

Zeiler M D. ADADELTA: An adaptive learning rate method. arXiv preprint arXiv:1212.5701, 2012.

References

Tieleman T, Hinton G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012, 4.

Kingma D, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

Dauphin Y N, de Vries H, Chung J, et al. RMSProp and equilibrated adaptive learning rates for non-convex optimization. arXiv preprint arXiv:1502.04390, 2015.

Zinkevich M. Online convex programming and generalized infinitesimal gradient ascent. ICML, 2003.

Reference

Shalev-Shwartz S. Online learning and online convex optimization. Foundations and Trends in Machine Learning, 2011, 4(2): 107-194.
Maclaurin D, Duvenaud D, Adams R P. Gradient-based Hyperparameter Optimization through Reversible Learning. arXiv preprint arXiv:1502.03492, 2015.
Massé P Y, Ollivier Y. Speed learning on the fly. arXiv preprint arXiv:1511.02540, 2015.