Modern Optimization Techniques

2. Unconstrained Optimization / 2.2. Stochastic Gradient Descent

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

Mon. 28.10.	(0)	0. Overview
		1. Theory
Mon. 4.11.	(1)	1. Convex Sets and Functions
		2. Unconstrained Optimization
Mon. 11.11.	(2)	2.1 Gradient Descent
Mon. 18.11.	(3)	2.2 Stochastic Gradient Descent
Mon. 25.11.	(4)	2.3 Newton's Method
Mon. 2.12.	(5)	2.4 Quasi-Newton Methods
Mon. 19.12.	(6)	2.5 Subgradient Methods
Mon. 16.12.	(7)	2.6 Coordinate Descent
	—	— *Christmas Break* —
		3. Equality Constrained Optimization
Mon. 6.1.	(8)	3.1 Duality
Mon. 13.1.	(9)	3.2 Methods
		4. Inequality Constrained Optimization
Mon. 20.1.	(10)	4.1 Primal Methods
Mon. 27.1.	(11)	4.2 Barrier and Penalty Methods
Mon. 3.2.	(12)	4.3 Cutting Plane Methods

Outline

Outline

Unconstrained Convex Optimization

$$\arg\min_{x \in \operatorname{dom} f} f(x)$$

- $\operatorname{dom} f \subseteq \mathbb{R}^N$ is convex and open (unconstrained optimization)
  - e.g., $\operatorname{dom} f = \mathbb{R}^N$
- f is convex

Stochastic Gradient

Gradient Descent makes use of the gradient

$$\nabla f(x)$$

Stochastic Gradient Descent: makes use of Stochastic Gradient only:

$$g(x) \sim p(g \in \mathbb{R}^N \mid x), \quad \mathbb{E}_p(g(x)) = \nabla f(x)$$

- for each point $x \in \mathbb{R}^N$:
  random variable over $\mathbb{R}^N$ with distribution p (conditional on x)
- on average yields the gradient (at each point)

Stochastic Gradient / Example: Big Sums

f is a "big sum":

$$f(x) = \frac{1}{C} \sum_{c=1}^{C} f_c(x)$$

with f_c convex, $\quad c = 1, \ldots, C$

g is the gradient of a random summand:

$$p(g \mid x) := \text{Unif}(\{\nabla f_c(x) \mid c = 1, \ldots, C\})$$

Stochastic Gradient / Example: Least Squares

$$\min_{x\in\mathbb{R}^N} f(x) := ||Ax-b||_2^2$$

- will find solution for $Ax = b$ if there is any (then $||Ax-b||_2 = 0$)
- otherwise will find the x where the difference $Ax - b$ of left and right side is as small as possible (in the squared L2 norm)
- is a big sum:

$$f(x) := ||Ax-b||_2^2 = \sum_{m=1}^{M}((Ax)_m - b_m)^2 = \sum_{m=1}^{M}(A_{m,.}x - b_m)^2$$

$$= \frac{1}{M}\sum_{m=1}^{M} f_m(x), \quad f_m(x) := M(A_{m,.}x - b_m)^2$$

- stochastic gradient g:
  - gradient for a random component m

Stochastic Gradient / Example: Supervised Learning

$$\min_{\theta \in \mathbb{R}^P} f(x) := \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, \hat{y}(x_n, \theta)) + \lambda ||\theta||_2^2$$

- where
  - $(x_n, y_n) \in \mathbb{R}^M \times \mathbb{R}^T$ are N training samples,
  - $\hat{y}$ is a parametrized model, e.g., logistic regression

    $$\hat{y}(x; \theta) := (1 + e^{-\theta^T x})^{-1}, \quad P := M, T := 1$$

  - ℓ is a loss, e.g., negative binomial loglikelihood:

    $$\ell(y, \hat{y}) := -y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

  - $\lambda \in \mathbb{R}_0^+$ is the regularization weight.
- will find parametrization with best trade-off between low loss and low model complexity

Stochastic Gradient / Example: Supervised Learning (2/2)

$$\min_{\theta \in \mathbb{R}^P} f(x) := \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, \hat{y}(x_n, \theta)) + \lambda ||\theta||_2^2$$

- where
  - $(x_n, y_n) \in \mathbb{R}^M \times \mathbb{R}^T$ are N training samples,
  - ...
- is a big sum:

$$f(\theta) := \frac{1}{N} \sum_{n=1}^{N} f_n(\theta), \quad f_n(\theta) := \ell(y_n, \hat{y}(x_n, \theta)) + \lambda ||\theta||_2^2$$

- stochastic gradient g:
  - gradient for a random sample n

Outline

Stochastic Gradient Descent

- the very same as Gradient Descent
- but use stochastic gradient $g(x)$ instead of exact gradient $\nabla f(x)$ in each step

```
min-sgd(f, p, x^(0), μ, K):
  for k := 1, ..., K:
    draw g^(k-1) ~ p(g | x)
    Δx^(k-1) := -g^(k-1)
    μ^(k-1) := μ(f, x^(k-1), Δx^(k-1))
    x^(k) := x^(k-1) + μ^(k-1) Δx^(k-1)
    if converged(...):
      return x^(k)
  raise exception "not converged in K iterations"
```

where

- p (distribution of the) stochastic gradient of f

Stochastic Gradient Descent / For Big Sums

```
min-sgd((f_c)_{c=1,...,C}, (∇f_c)_{c=1,...,C}, x^(0), μ, K):
  for k := 1, ..., K:
    draw c^(k−1) ∼ Unif(1, ..., C)
    g^(k−1) := ∇f_{c^(k−1)}(x^(k−1))
    Δx^(k−1) := −g^(k−1)
    μ^(k−1) := μ(f, x^(k−1), Δx^(k−1))
    x^(k) := x^(k−1) + μ^(k−1) Δx^(k−1)
    if converged(...):
      return x^(k)
  raise exception "not converged in K iterations"
```

where

- $(f_c)_{c=1,\ldots,C}$ objective function summands, $f := \frac{1}{C}\sum_{c=1}^{C} f_c$
- $(\nabla f_c)_{c=1,\ldots,C}$ gradients of the objective function summands

SGD / For Big Sums / Epochs

min-sgd$((f_c)_{c=1,\ldots,C},\ (\nabla f_c)_{c=1,\ldots,C},\ x^{(0)},\ \mu,\ K)$:
  $\mathcal{C} := (1, 2, \ldots, C)$
  $x^{(0,C)} := x^{(0)}$
  for $k := 1, \ldots, K$:
    randomly shuffle $\mathcal{C}$
    $x^{(k,0)} := x^{(k-1,C)}$
    for $i = 1, \ldots, C$:
      $g^{(k,i-1)} := \nabla f_{c_i}(x^{(k,i-1)})$
      $\Delta x^{(k,i-1)} := -g^{(k,i-1)}$
      $\mu^{(k,i-1)} := \mu(f, x^{(k,i-1)}, \Delta x^{(k,i-1)})$
      $x^{(k,i)} := x^{(k,i-1)} + \mu^{(k,i-1)} \Delta x^{(k,i-1)}$
  return $x^{(K,C)}$

where

- $(f_c)_{c=1,\ldots,C}$ objective function summands, $f := \frac{1}{C} \sum_{c=1}^{C} f_c$
- ...

Theorem (Convergence of Gradient Descent) [review]

If

(i) f is strongly convex,

(ii) the initial sublevel set $S := \{x \in \text{dom}\, f \mid f(x) \leq f(x^{(0)})\}$ is closed,

(iii) an exact line search is used,

then gradient descent converges, esp.

$$f(x^{(k)}) - p^* \leq (1 - \frac{m}{M})^k \; (f(x^{(0)}) - p^*)$$

$$||x^{(k)} - x^*||^2 \leq (1 - \frac{m}{M})^k \frac{||(\nabla f(x^{(0)}))||^2}{m^2}$$

Theorem (Convergence of SGD)

If

(i) f is strongly convex ($||\nabla^2 f(x)|| \succeq mI, m \in \mathbb{R}^+$),

(ii) the expected squared norm of its stochastic gradient g is uniformly bounded ($\exists G \in \mathbb{R}_0^+ \; \forall x : \mathbb{E}(||g(x)||^2) \leq G^2$) and

(iii) the step size $\mu^{(k)} := \frac{1}{m(k+1)}$ is used,

then SGD converges, esp.

$$\mathbb{E}_p(||x^{(k)} - x^*||^2) \leq \frac{1}{k+1} \max\{||x^{(0)} - x^*||^2, \frac{G^2}{m^2}\}$$

Convergence of SGD / Proof

$$f(x^*) - f(x) \geq \nabla f(x)^T (x^* - x) + \frac{m}{2} ||x^* - x||^2 \quad \text{str. conv. (i)}$$

$$f(x) - f(x^*) \geq \nabla f(x^*)^T (x - x^*) + \frac{m}{2} ||x - x^*||^2 = \frac{m}{2} ||x^* - x||^2$$

summing both yields

$$0 \geq \nabla f(x)^T (x^* - x) + m||x^* - x||^2$$

$$\nabla f(x)^T (x - x^*) \geq m||x^* - x||^2 \qquad (1)$$

$$\begin{aligned}
&\mathbb{E}(||x^{(k)} - x^*||^2) \\
&= \mathbb{E}(||x^{(k-1)} - \mu^{(k-1)} g^{(k-1)} - x^*||^2) \\
&= \mathbb{E}(||x^{(k-1)} - x^*||^2) - 2\mu^{(k-1)} \mathbb{E}((g^{(k-1)})^T (x^{(k-1)} - x^*)) + (\mu^{(k-1)})^2 \mathbb{E}(||g^{k-1}||^2) \\
&= \mathbb{E}(||x^{(k-1)} - x^*||^2) - 2\mu^{(k-1)} \mathbb{E}(\nabla f(x^{(k-1)})^T (x^{(k-1)} - x^*)) + (\mu^{(k-1)})^2 \mathbb{E}(||g^{k-1}||^2) \\
&\overset{(ii),(1)}{\leq} \mathbb{E}(||x^{(k-1)} - x^*||^2) - 2\mu^{(k-1)} m \mathbb{E}(||x^* - x^{(k-1)}||^2) + (\mu^{(k-1)})^2 G^2 \\
&= (1 - 2\mu^{(k-1)} m) \mathbb{E}(||x^{(k-1)} - x^*||^2) + (\mu^{(k-1)})^2 G^2 \qquad (2)
\end{aligned}$$

Convergence of SGD / Proof (2/2)

induction over k: $k := 0$:

$$||x^{(0)} - x^*||^2 \leq \frac{1}{1} L, \quad L := \max\{||x^{(0)} - x^*||^2, \frac{G^2}{m^2}\}$$

$k > 0$:

$$\begin{aligned}
\mathbb{E}(||x^{(k)} - x^*||^2) &\overset{(2)}{\leq} (1 - 2\mu^{(k-1)} m)\mathbb{E}(||x^{(k-1)} - x^*||^2) + (\mu^{(k-1)})^2 G^2 \\
&\overset{(iii)}{=} (1 - \frac{2}{k})\mathbb{E}(||x^{(k-1)} - x^*||^2) + \frac{G^2}{m^2 k^2} \\
&\overset{\text{ind.hyp.}}{\leq} (1 - \frac{2}{k})\frac{1}{k} L + \frac{G^2}{m^2 k^2} \\
&\overset{\text{def. } L}{\leq} (1 - \frac{2}{k})\frac{1}{k} L + \frac{1}{k^2} L \\
&= \frac{k-2}{k^2} L + \frac{1}{k^2} L = \frac{k-1}{k^2} L \leq \frac{1}{k+1} L
\end{aligned}$$

Outline

1. Stochastic Gradients
2. Stochastic Gradient Descent (SGD)
3. More on Line Search: Bold Driver
4. More on Line Search: AdaGrad

Choosing the step size for SGD

- The step size μ is a crucial parameter of gradient descent
- Given the low cost of the SGD update,
  using exact line search for the step size is a bad choice
- Possible alternatives:
  - Fixed step size
  - Exponentially decreasing step size
  - Backtracking / Armijo principle
  - Bold-Driver
  - Adagrad

Example: Body Fat prediction

We want to estimate the percentage of body fat based on various attributes:

- Age (years)
- Weight (lbs)
- Height (inches)
- Neck circumference (cm)
- Chest circumference (cm)
- Abdomen 2 circumference (cm)
- Hip circumference (cm)
- Thigh circumference (cm)
- Knee circumference (cm)
- ...

http://lib.stat.cmu.edu/datasets/bodyfat

Example: Body Fat prediction

The data is represented it as:

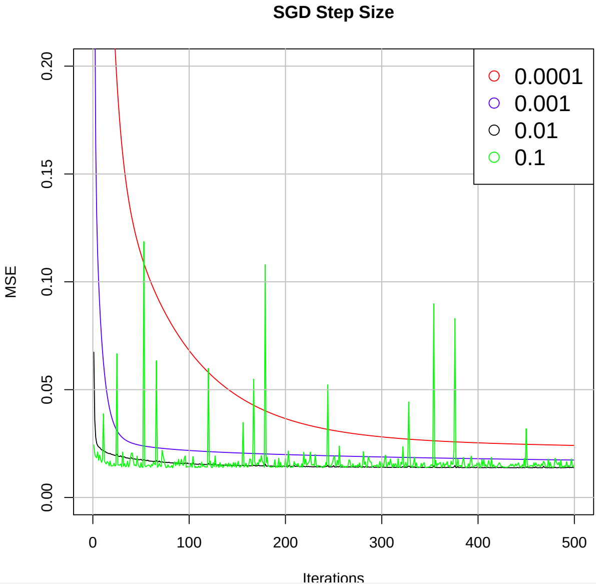
$$A = \begin{pmatrix} 1 & a_{1,1} & a_{1,2} & \dots & a_{1,M} \\ 1 & a_{2,1} & a_{2,2} & \dots & a_{2,M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{N,1} & a_{N,2} & \dots & a_{N,M} \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$$

with $N = 252$, $M = 14$

We can model the percentage of body fat y
as a linear combination of the body measurements with parameters $\mathbf{x}$:

$$\hat{y}_n = \mathbf{x}^T \mathbf{a_n} = x_0 1 + x_1 a_{n,1} + x_2 a_{n,2} + \ldots + x_M a_{n,M}$$

SGD - Fixed Step Size on the Body Fat dataset



Bold Driver Heuristic

- idea: use smaller step sizes closer to the minimum.
- adjust step size based on the value of $f(\mathbf{x}^{(k)}) - f(\mathbf{x}^{(k-1)})$
- if the value of $f(\mathbf{x})$ grows,
  the step size must decrease
- if the value of $f(\mathbf{x})$ decreases,
  the step size can increase for faster convergence
- adapt stepsize only **once after each epoch**,
  not for every (inner) iteration.

Bold Driver Heuristic — Update Rule

We need to define

- an **increase factor** $\mu^+ > 1$, e.g. $\mu^+ := 1.05$, and
- a **decay factor** $\mu^- \in (0, 1)$, e.g., $\mu^- := 0.5$.

Step size update rule:

- Cycle through the whole data and update the parameters
- Evaluate the objective function $f(\mathbf{x}^{(k)})$
- if $f(\mathbf{x}^{(k)}) < f(\mathbf{x}^{(k-1)})$ then $\mu \rightarrow \mu^+\mu$
- else $f(\mathbf{x}^{(k)}) > f(\mathbf{x}^{(k-1)})$ then $\mu \rightarrow \mu^-\mu$
- different from the bold driver heuristics for batch gradient descent, there is no way to evaluate $f(x + \mu\Delta x)$ for different μ.
  - stepsize μ is adapted once after the step has been done

Bold Driver

stepsize-bd($\mu, f_{\text{new}}, f_{\text{old}}, \mu^+, \mu^-$):
    if $f_{\text{new}} < f_{\text{old}}$
        $\mu := \mu^+ \mu$
    else
        $\mu := \mu^- \mu$
    return μ

where

- μ stepsize of last update
- $f_{\text{new}}, f_{\text{old}} = f(x^k), f(x^{k-1})$ function values before and after the last update
- μ^+, μ^- stepsize increase and decay factors

Considerations

- works well for a range of problems
- initial μ just needs to be large enough
- μ^+ and μ^- have to be adjusted to the problem at hand
  - often used values: $\mu^+ = 1.05$ and $\mu^- = 0.5$
- may lead to faster convergence

Outline

AdaGrad

- idea: adjust the step size
  individually for each variable to be optimized
- use information about past gradients
- often leads to faster convergence
- does not have parameters
  - such as μ^+ and μ^- for Bold Driver
- update stepsize for every inner iteration

AdaGrad - Update Rule

We have

$$\mathbf{g}(x) \sim p(\mathbf{g} \in \mathbb{R}^N \mid x), \quad \mathbb{E}_p(\mathbf{g}(x)) = \nabla f(x)$$

Update rule:

- Update the gradient square history

$$\mathbf{G}^{2,\text{next}} := \mathbf{G}^2 + \mathbf{g}(x) \odot \mathbf{g}(x)$$

- The step size for variable $\mathbf{x}_n$ is

$$\mu_n := \frac{\mu_0}{\sqrt{\mathbf{G^2}_n} + \epsilon}$$

- Update

$$\mathbf{x}^{\text{next}} := \mathbf{x} - \mu \odot \mathbf{g}(x)$$

$$\text{i.e., } \mathbf{x}_n^{\text{next}} := \mathbf{x}_n - \frac{\mu_0}{\sqrt{\mathbf{G^2}_n} + \epsilon}(g(x))_n$$

$\odot$ denotes the elementwise product, $\mathbf{G^2}$ a variable name, not a square.

AdaGrad

stepsize-adagrad$(\mathbf{g}, \mathbf{G^2}; \mu_0, \epsilon)$:

$\quad \mathbf{G^2} := \mathbf{G^2} + \mathbf{g} \circ \mathbf{g}$

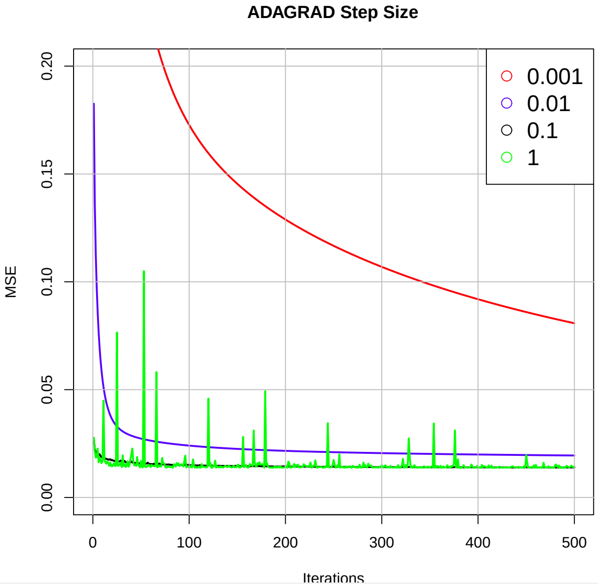
$\quad \mu_n := \frac{\mu_0}{\sqrt{\mathbf{G^2}_n + \epsilon}}$ for $n = 1, \ldots, N$

$\quad$ return $(\mu, \mathbf{G^2})$

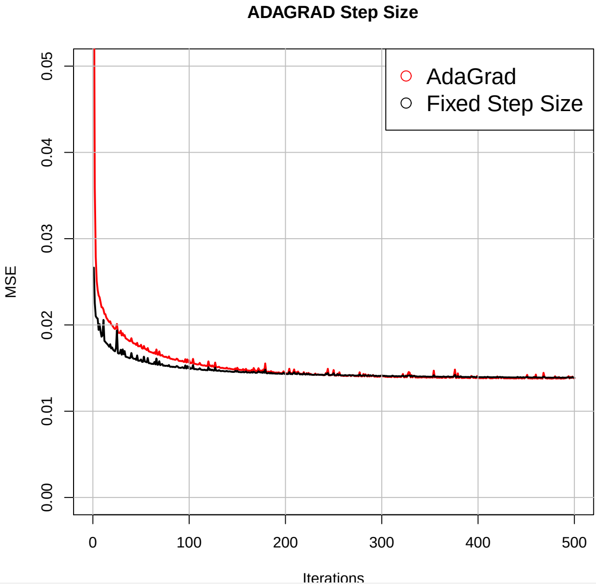
where

- returns a vector of stepsizes, one for each variable
- $\mathbf{g} \sim p(\mathbf{g} \in \mathbb{R}^N \mid \mathbf{x}), \quad \mathbb{E}_p(\mathbf{g}(\mathbf{x})) = \nabla f(x)$ current (stochastic) gradient
- $\mathbf{G}$ past gradient square history
- μ_0 initial stepsize

AdaGrad Step Size



AdaGrad vs Fixed Step Size



Adam

stepsize-adam$(\mathbf{g}, \mathbf{G}, \mathbf{G^2}, t; \mu_0, \beta_1, \beta_2, \epsilon)$:
  $\mathbf{G} := \beta_1 \mathbf{G} + (1-\beta_1)\mathbf{g}$
  $\mathbf{G^2} := \beta_2 \mathbf{G^2} + (1-\beta_2)\mathbf{g} \odot \mathbf{g}$
  $\mu_n := \mu_0 \frac{\mathbf{G}_n/(1-\beta_1^t)}{\sqrt{\mathbf{G^2}_n/(1-\beta_2^t)}+\epsilon}$ for $n = 1, \ldots, N$
  return $(\mu, \mathbf{G}, \mathbf{G^2})$

where

- returns a vector of stepsizes, one for each variable
- $\mathbf{g} \sim p(\mathbf{g} \in \mathbb{R}^N \mid \mathbf{x})$, $\mathbb{E}_p(\mathbf{g}(\mathbf{x})) = \nabla f(x)$ current (stochastic) gradient
- $\mathbf{G}, \mathbf{G^2}$ past gradient and gradient square history
- t iteration
- μ_0 initial stepsize

Note: Adagrad is a special case for $\beta_1 := 0, \beta_2 := \frac{1}{2}$ and iteration-dependent $\mu_0(t) := 2\mu_0^{\text{Adagrad}}/\sqrt{1-0.5^t}$.

Summary

- Stochastic Gradient Descent (SGD) is like Gradient Descent,
  - but instead of the exact gradient uses just a random vector called stochastic gradient
    - with expectation of the true/exact gradient.

- stochastic gradients occur naturally when the objective is a big sum
  - then the gradient of a uniformly random component is a stochastic gradient
  - e.g., objectives for most machine learning problems are big sums over instance-wise losses (and regularization terms).

- SGD converges with a rate of $1/k$ in the number of steps k.

Summary (2/2)

- step size and convergence critera have to be adapted
  - to aggregate over several update steps, e.g., an epoche
  - cannot test for different step sizes (like backtracking)

- Bold driver step size control:
  - update per epoche based on additional function evaluation.

- Adagrad step size control:
  - individual step size for each variable
  - $1/\sum g^2$ for past gradients.

Further Readings

- SGD is not covered in Boyd and Vandenberghe [2004].
- Leon Bottou, Frank E. Curtis, Jorge Nocedal (2016): *Stochastic Gradient Methods for Large-Scale Machine Learning*, ICML 2016 Tutorial, http://users.iems.northwestern.edu/~nocedal/ICML
- Francis Bach (2013): *Stochastic gradient methods for machine learning*, Microsoft Machine Learning Summit 2013, http://research.microsoft.com/en-us/um/cambridge/events/mls2013/downloads/stochastic_gradient.pdf
- for the convergence proof:
  Ji Liu (2014), Notes "Stochastic Gradient Descent",
  http://www.cs.rochester.edu/~jliu/CSC-576-2014fall.html

References

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.