Convergence rate of SGD

- **Theorem**:
  - (see Nemirovski et al '09 from readings)
  - Let *f* be a strongly convex stochastic function with param γ
  - Assume gradient of *f* is Lipschitz continuous and bounded

$\|\nabla f\|_2^2 \leq M^2$

$\forall x \quad \|\nabla f(w,x) - \nabla f(w',x)\|_2 \leq L \|w - w'\|_2 \qquad L > 0$

  - Then, for step sizes:

$\eta_t = K/t \qquad K > 0$

  - The expected loss decreases as O(1/t):

e.g. $K = 1/\gamma$

$$\underbrace{E\left[f(w^{(t)}) - f(w^*)\right]}_{\text{how much closer getting to } w^*} \leq \frac{1}{t} L \left(\frac{M^2}{\gamma^2} + \|w^{(0)} - w^*\|_2^2\right)$$

$O(1/t)$

Convergence rates for gradient descent/ascent versus SGD

$O(Nd \ln \frac{1}{\varepsilon})$ <- - -> $O(\frac{d}{\varepsilon})$

GD . . . | . . . SGD

N data points — mini batches 100 datapoints — 1 data point

- Number of Iterations to get to accuracy

$$\ell(\mathbf{w}^*) - \ell(\mathbf{w}) \leq \epsilon$$

- Gradient descent:
  - If func is strongly convex: O(ln(1/ϵ)) iterations
- Stochastic gradient descent:
  - If func is strongly convex: O(1/ϵ) iterations
- Seems exponentially worse, but much more subtle:
  - Total running time, e.g., for logistic regression:
    - Gradient descent: $O(\ln \frac{1}{\varepsilon})$ iterations, iteration $O(Nd)$, total: $O(Nd \ln \frac{1}{\varepsilon})$
    - SGD: $O(\frac{1}{\varepsilon})$ iterations, iteration $O(d)$, total: $O(d/\varepsilon)$
    - SGD can win when we have a lot of data
  - And, when analyzing true error, situation even more subtle… expected running time about the same, see readings

Motivating AdaGrad (Duchi, Hazan, Singer 2011)

- Assuming $\mathbf{w} \in \mathbb{R}^d$, standard stochastic (sub)gradient descent updates are of the form:

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta_t g_{t,i}$$

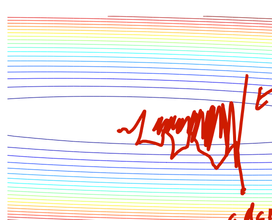
- step size
- learning rate

$\eta_{t,i} \leftarrow$ feature-specific step size

$g_{t,i} \leftarrow$ transforming by a factor a_i

- Should all features share the same learning rate?
- Often have high-dimensional feature spaces
  - Many features are irrelevant → Small learning rate
  - Rare features are often very informative
- Adagrad provides a feature-specific adaptive learning rate by incorporating knowledge of the geometry of past observations

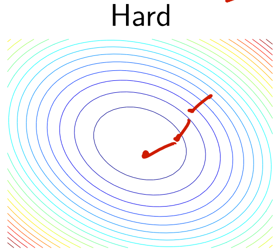
Why Adapt to Geometry?



non adaptive

adaptive

Hard



Nice

y_t	$x_{t,1}$	$x_{t,2}$	$x_{t,3}$
1	1	0	0
-1	.5	0	1
1	-.5	1	0
-1	0	0	0
1	.5	0	0
-1	1	0	0
1	-1	1	0
-1	-.5	0	1

1. Frequent, irrelevant
2. Infrequent, predictive
3. Infrequent, predictive

Examples from Duchi et al. ISMP 2012 slides

Not All Features are Created Equal

- Examples:

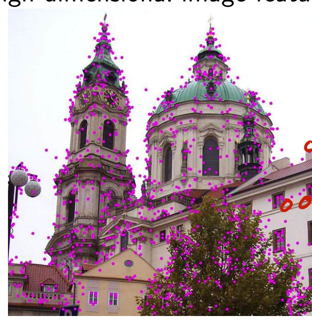
Text data:

```
The most unsung birthday
in American business and
technological history
this year may be the 50th
anniversary of the Xerox
914 photocopier.[a]
```

[a] *The Atlantic*, July/August 2010.

rare word

High-dimensional image features



corners less frequent & more informative

Images from Duchi et al. ISMP 2012 slides

Constrained Optimization

Original problem $\min_{w \in W} \ell(w)$

Projected Gradient

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta g_{t,i}$$

- Brief aside… e.g., $W \Rightarrow \|W\|_1 \le R$

- Consider an arbitrary feature space $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^d$

- If $\mathbf{w} \in \mathcal{W}$, can use ***projected gradient*** for (sub)gradient descent

$$\mathbf{w}^{(t+1)} = \arg\min_{w \in W} \|w - (w^{(t)} - \eta_t g_t)\|_2^2$$

efficient for some W

e.g. $W: \|w\|_2 \le R$

$\|w\|_1 \le R$

$\vdots$

W

$w^{(t+1)}$

$w^{(t)}$

$w^{(t)} - \eta_t g_t$

closest point in the space to $w^{(t)} - \eta_t g_t$

Regret Minimization

$R(T) \to 0$
$\Rightarrow w^{(t)}, w^{(t+1)} \dots$ as good as w^*

no-regret algorithm

- How do we assess the performance of an online algorithm?
- Algorithm iteratively predicts $\mathbf{w}^{(t)}$ — ad setting: $\hat{y}_t$ click?
- Incur ***loss*** $f_t(\mathbf{w}^{(t)})$ — either click or not
- ***Regret***:
  What is the total incurred loss of algorithm relative to the best choice of $\mathbf{w}$ that could have been made ***retrospectively***

typically $\frac{R(T)}{T} \to 0$ as $T \to \infty$

$$\underbrace{R(T)}_{\text{regret}} = \underbrace{\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)})}_{\text{cummulative loss based on sequence of choices}} - \underbrace{\overbrace{\inf_{\mathbf{w}\in\mathcal{W}}}^{w^*} \sum_{t=1}^{T} f_t(\mathbf{w})}_{\text{best single } w \text{ in retrospect}}$$

Regret Bounds for Standard SGD

- Standard projected gradient stochastic updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta g_t)||_2^2$$

$w^{(t+1)}$ $w^{(t)}$

- Standard regret bound:

same norm

$$\underbrace{\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)}_{R(T)} \le \frac{1}{2\eta}\underbrace{||\mathbf{w}^{(1)} - \mathbf{w}^*||_2^2}_{\text{error of where you started}} + \frac{\eta}{2}\sum_{t=1}^{T} \underbrace{||g_t||_2^2}_{\text{magnitude of gradients}}$$

Projected Gradient using Mahalanobis

$g_t = \begin{pmatrix} g_{t,1} \\ g_{t,2} \end{pmatrix}$ scale up, scale down

- Standard projected gradient stochastic updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta g_t)||_2^2$$

- What if instead of an L_2 metric for projection, we considered the ***Mahalanobis*** norm

$A = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}$ care more about w_1 gradient

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A^{-1} g_t)||_A^2$$

composed by proj. with A

L_2 ball: $||w||_2 \le R$

$\sqrt{w^T w} \le R$

$||w||_A \le R$

$\sqrt{w^T A w} \le R$

$A = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}$

$||w||_A \le R$ $A = \begin{pmatrix} \cdot & \cdot \\ \cdot & \cdot \end{pmatrix}$

$A \succeq 0$ positive semi-definite

Mahalanobis Regret Bounds

in 1 d: $||g_t||^2_{A^{-1}} = \frac{g_t^2}{a}$

min by $a \to \infty$

$tr(A) = \sum_i A_{ii}$

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A^{-1} g_t)||_A^2$$

- **What *A* to choose?**

$w^{*T} A w^*$ if $a \to \infty$ $\to \infty$

$||g_t||^2_{A^{-1}} = g_t^T A^{-1} g_t = \langle g_t, A^{-1} g_t \rangle$

- Regret bound now: $R(T)$ want to minimize

$$\sum_{t=1}^T f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \le \frac{1}{2\eta}||\cancel{\mathbf{w}^{(1)}}^{\,0} - \mathbf{w}^*||_A^2 + \frac{\eta}{2}\sum_{t=1}^T ||g_t||_{A^{-1}}^2$$

- What if we minimize upper bound on regret w.r.t. *A* in hindsight?

Choice of A: $||g_t||^2_{A^{-1}}$

$$\min_A \sum_{t=1}^T \langle g_t, A^{-1} g_t \rangle$$

avoid by not letting A get too big:

$tr(A) \le C$

Mahalanobis Regret Minimization

for Mahalanobis distance

- Objective: $g_t^T A^{-1} g_t$

$$\min_A \sum_{t=1}^{T} \langle g_t, A^{-1} g_t \rangle \quad \text{subject to } A \succeq 0, \text{tr}(A) \leq C$$

- Solution:

$$\boxed{A = c\left(\sum_{t=1}^{T} g_t g_t^T\right)^{\frac{1}{2}}}$$

if Q, $Q \succeq 0$, $\exists V$
$Q = V^T V$ ← square root matrix

outer product of gradient

For proof, see Appendix E, Lemma 15 of Duchi et al. 2011.
Uses "trace trick" and Lagrangian.

- *A* defines the norm of the metric space we should be operating in

AdaGrad Algorithm

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w} \in \mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A_t^{-1} g_t)||^2_{A_t}$$

- At time *t*, estimate optimal (sub)gradient modification *A* by

estimate of A at time t →

$$A_t = \left(\sum_{\tau=1}^{t} g_\tau g_\tau^T\right)^{\frac{1}{2}}$$

upto time t

← in d dims matrix $\sqrt{}$ is $O(d^3)$

- For *d* large, A_t is computationally intensive to compute. Instead,

diag(A_t) $\quad A_t = \begin{pmatrix} A_{ii} & 0 \\ 0 & \ddots \end{pmatrix}$

- Then, algorithm is a simple modification of normal updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w} \in \mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta \text{diag}(A_t)^{-1} g_t)||^2_{\text{diag}(A_t)}$$

$$A_{ii}^t = \sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}$$

weigh dimensions by sqrt of sum of gradients in that dim

$x_t = (0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0)$

AdaGrad in Euclidean Space

- For $\mathcal{W} = \mathbb{R}^d$,
  no constraints on w

  $w^{(t+1)} \leftarrow w^{(t)} - \eta\, \text{diag}(A_t)^{-1} g_t$

- For each feature dimension,

  adaptive step size

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta_{t,i} g_{t,i}$$

where

$$\eta_{t,i} = \eta / A_{t,ii}$$

in sparse case, step size larger when seeing a rare feature

- That is,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \frac{\eta}{\sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}} g_{t,i}$$

- Each feature dimension has it's own learning rate!
  - Adapts with *t*
  - Takes geometry of the past observations into account
  - Primary role of η is determining rate the first time a feature is encountered

AdaGrad Theoretical Guarantees

- AdaGrad regret bound:

$$\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \leq 2R_\infty \sum_{i=1}^{d} ||g_{1:T,j}||_2$$

$$R_\infty := \max_t ||\mathbf{w}^{(t)} - \mathbf{w}^*||_\infty$$

radius of space

- So, what does this mean in practice?
- Many cool examples. This really is used in practice!
- Let's just examine one…

AdaGrad Theoretical Example

$x = (0\,0\,0\,0\,1\,0 \cdots 0\,1\,0\,1)$

- Expect to out-perform when gradient vectors are sparse
- SVM hinge loss example:

$$f_t(\mathbf{w}) = [1 - y^t \langle \mathbf{x}^t, \mathbf{w} \rangle]_+ \quad \text{where} \quad \mathbf{x}^t \in \{-1, 0, 1\}^d$$

hinge loss

- If $x_j^t \neq 0$ with probability $\propto j^{-\alpha}, \quad \alpha > 1$

$j^{-\alpha}$ heavy tailed distribution

$R(T)/T$

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}\right)\right] - f(\mathbf{w}^*) = \mathcal{O}\left(\frac{||\mathbf{w}^*||_\infty}{\sqrt{T}} \cdot \max\{\log d, d^{1-\alpha/2}\}\right)$$

for d small

- Previously best known method:

same

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}\right)\right] - f(\mathbf{w}^*) = \mathcal{O}\left(\frac{||\mathbf{w}^*||_\infty}{\sqrt{T}} \cdot \sqrt{d}\right)$$

adagrad can be exp better in d

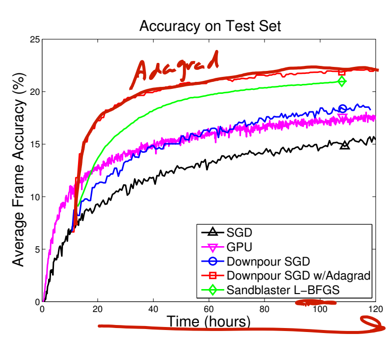
Neural Network Learning

- Very non-convex problem, but use SGD methods anyway

$$f(\cancel{x}\,w, \xi) = \log\left(1 + \exp\left(\langle [p(\langle w_1, \xi_1\rangle) \;\cdots\; p(\langle w_k, \xi_k\rangle)], \xi_0\rangle\right)\right)$$

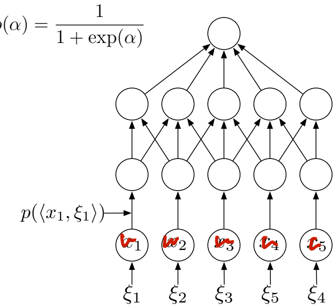
$$p(\alpha) = \frac{1}{1 + \exp(\alpha)}$$

accuracy



running time

(Dean et al. 2012)



Distributed, $d = 1.7 \cdot 10^9$ parameters. SGD and AdaGrad use 80 machines (1000 cores), L-BFGS uses 800 (10000 cores)

Images from Duchi et al. ISMP 2012 slides

What you should know about Logistic Regression (LR) and Click Prediction

- Click prediction problem:
  - Estimate probability of clicking
  - Can be modeled as logistic regression
- Logistic regression model: Linear model
- Gradient ascent to optimize conditional likelihood
- Overfitting + regularization
- Regularized optimization
  - Convergence rates and stopping criterion
- Stochastic gradient ascent for large/streaming data
  - Convergence rates of SGD
- AdaGrad motivation, derivation, and algorithm