CPSC 540: Machine Learning

SGD Convergence Rate

Mark Schmidt

University of British Columbia

Winter 2019
Last time: Stochastic sub-gradient

- We discussed minimizing finite sums,

\[ f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w), \]

when \( n \) is very large.

- For non-smooth \( f_i \), we discussed stochastic subgradient method,

\[ w^{k+1} = w^k - \alpha_k g_{i_k}, \]

for some \( g_{i_k} \in \partial f_{i_k}(w^k) \) for some random \( i_k \in \{1, 2, \ldots, n\} \).

  - May increase \( f \), but moves closer to \( w^* \) for small \( \alpha_k \) in expectation.

- Same rate as deterministic subgradient method but \( n \) times faster.
Stochastic Gradient Convergence Assumptions

- We’re going to analyze stochastic gradient rate under these assumptions:
  - $f$ is bounded below (not necessarily convex).
  - $\nabla f$ is $L$-Lipschitz continuous.
  - $\mathbb{E}[\|\nabla f_i(w)\|^2] \leq \sigma^2$ for some constant $\sigma^2$ and all $w$ (“variance” is bounded).
    - This bounds the worst-case effect of the “bad term”.

- Possible to relax noise bound to more-realistic $\mathbb{E}[\|\nabla f_i(w^k) - \nabla f(w^k)\|^2] \leq \sigma^2$.
  - Just get some extra terms in the result.

- Possible to show similar results for non-smooth functions.
  - Need something stronger than “bounded below” (“weak convexity” or “tame”).
Convergence Rate of Stochastic Gradient Method

- Let’s use the “variance” bound inside previous bound,

\[
\mathbb{E}[f(w^{k+1})] \leq f(w^k) - \alpha_k \|\nabla f(w^k)\|^2 + \alpha_k^2 \frac{L}{2} \mathbb{E}[\|\nabla f_i(w^k)\|^2]
\]

\[
\leq f(w^k) - \alpha_k \|\nabla f(w^k)\|^2 + \alpha_k^2 \frac{L\sigma^2}{2}
\]

- As before, re-arrange to get the gradient norm on the left side,

\[
\alpha_k \|\nabla f(w^k)\|^2 \leq f(w^k) - \mathbb{E}[f(w^{k+1})] + \alpha_k^2 \frac{L\sigma^2}{2}
\]

- Sum this up (and use iterated expectation) to get

\[
\sum_{k=1}^{t} \alpha_{k-1} \mathbb{E}[\|\nabla f(w^{k-1})\|^2] \leq \sum_{k=1}^{t} [\mathbb{E}f(w^{k-1}) - \mathbb{E}f(w^k)] + \sum_{k=1}^{t} \alpha_{k-1}^2 \frac{L\sigma^2}{2}
\]
Convergence Rate of Stochastic Gradient Method

- The bound from the previous slide:

\[
\sum_{k=1}^{t} \alpha_{k-1} \mathbb{E} \left\| \nabla f(w^{k-1}) \right\|^2 \leq \sum_{k=1}^{t} \left[ \mathbb{E} f(w^{k-1}) - \mathbb{E} f(w^k) \right] + \sum_{k=1}^{t} \alpha_{k-1}^2 \frac{L\sigma^2}{2}.
\]

- Applying the above operations gives

\[
\min_{k=0,1,\ldots,t-1} \left\{ \mathbb{E} \left\| \nabla f(w^k) \right\|^2 \right\} \sum_{k=0}^{t-1} \alpha_k \leq f(w^0) - \mathbb{E} f(w^t) + \frac{L\sigma^2}{2} \sum_{k=0}^{t-1} \alpha_k^2.
\]

- Using \( \mathbb{E} f(w^k) \geq f^* \) and dividing both sides by \( \sum_k \alpha_{k-1} \) gives

\[
\min_{k=0,1,\ldots,t-1} \left\{ \mathbb{E} \left\| \nabla f(w^k) \right\|^2 \right\} \leq \frac{f(w^0) - f^*}{\sum_{k=0}^{t-1} \alpha_k} + \frac{L\sigma^2}{2} \frac{\sum_{k=0}^{t-1} \alpha_k^2}{\sum_{k=0}^{t-1} \alpha_k}.
\]
Convergence Rate of Stochastic Gradient Method

- The final bound:
  \[ \min_{k=0,1,...,t-1} \{ \mathbb{E} \| \nabla f(w^k) \|^2 \} \leq \frac{f(w^0) - f^*}{\sum_{k=0}^{t-1} \alpha_k} + \frac{L\sigma^2}{2} \sum_{k=0}^{t-1} \frac{\alpha_k^2}{\sum_{k=0}^{t-1} \alpha_k}. \]

- If \( \sigma^2 = 0 \), then we could use a constant step-size and would get a \( O(1/t) \) rate.
  - Same as regular gradient descent.
- But due to stochasticity, convergence rate is determined by \( \sum_k \alpha_k^2 / \sum_k \alpha_k \).

- Classic decreasing step-sizes: set \( \alpha_k = \alpha/k \) for some \( \alpha \).
  - Gives \( \sum_k \alpha_k = O(\log(t)) \) and \( \sum_k \alpha_k^2 = O(1) \), so error at \( t \) is \( O(1/\log(t)) \).

- Bigger decreasing step-sizes: set \( \alpha_k = \alpha/\sqrt{k} \) for some \( \alpha \).
  - Gives \( \sum_k \alpha_k = O(\sqrt{k}) \) and \( \sum_k \alpha_k^2 = O(\log(k)) \), so error at \( t \) is \( O(\log(t)/\sqrt{t}) \).

- Constant step-sizes: set \( \alpha_k = \alpha \) for some \( \alpha \).
  - Gives \( \sum_k \alpha_k = k\alpha \) and \( \sum_k \alpha_k^2 = k\alpha^2 \), so error at \( t \) is \( O(1/t) + O(\alpha) \).
Convergence of Stochastic [Sub]Gradient under Strong Convexity

- You can get faster rates if $f$ is strongly-convex:
  - With decreasing $\alpha_k = 1/\mu k$ you get $O(1/t)$ for $t$ iterations (but not linear).
    - But be careful, if you over-estimate $\mu$ rate can be much worse.
    - Also, initial steps are huge (this approach only seems to work for binary SVMs).
  - With constant $\alpha_k = \alpha < 1/2\mu$ you get $O(\rho^k) + O(\alpha)$ for $t$ iterations.
    - Linear convergence up to some accuracy proportional to $\alpha$ for sufficiently small $\alpha$.

- For non-smooth strongly-convex $f$ you get similar results:
  - Setting $\alpha_k = 1/\mu k$ gives $O(\log(t)/t)$.
    - Can improve to $O(1/t)$ by using averaging of the last $t/2$ values of $w^k$.
  - Setting $\alpha_k = \alpha < 1/2\mu$ still gives $\mathbb{E}[\|w^k - w^*\|^2] = O(\rho^k) + O(\alpha)$.
    - Looks like linear convergence if far from solution (or gradients are similar).
    - No progress if close to solution or have high variance in gradients.
Stochastic Subgradient with Constant Step Size

- Expected distance with constant step-size and strong convexity (see bonus):

\[ \mathbb{E}[\|w^k - w^*\|^2] \leq (1 - 2\alpha\mu)^k\|w^0 - w^*\|^2 + \frac{\alpha\sigma^2}{2\mu} , \]

- First term looks like linear convergence, but second term does not go to zero.
Stochastic Subgradient with Constant Step Size

- Expected distance with constant step-size and strong convexity (see bonus):
  
  \[ \mathbb{E}[\|w^k - w^*\|^2] \leq (1 - 2\alpha \mu)^k \|w^0 - w^*\|^2 + \frac{\alpha \sigma^2}{2\mu}, \]

- First term looks like **linear convergence**, but second term does **not go to zero**.
Stochastic Subgradient with Constant Step Size

- Expected distance with constant step-size and strong convexity (see bonus):
  \[ \mathbb{E}[\|w_k - w^*\|^2] \leq (1 - 2\alpha\mu)^k\|w^0 - w^*\|^2 + \frac{\alpha\sigma^2}{2\mu}, \]

- First term looks like linear convergence, but second term does not go to zero.
Expected distance with constant step-size and strong convexity (see bonus):

\[ E[\|w^k - w^*\|^2] \leq (1 - 2\alpha \mu)^k \|w^0 - w^*\|^2 + \frac{\alpha \sigma^2}{2\mu}, \]

First term looks like **linear convergence**, but second term does **not go to zero**.
Expected distance with constant step-size and strong convexity (see bonus):

\[
\mathbb{E}[\|w^k - w^*\|^2] \leq (1 - 2\alpha\mu)^k\|w^0 - w^*\|^2 + \frac{\alpha\sigma^2}{2\mu},
\]

First term looks like \textit{linear convergence}, but second term does \textit{not go to zero}.

Theory justifies “\textit{divide the step-size in half if it looks like it’s stalled}” heuristic.

Halving \(\alpha\) divides radius of the ball around \(w^*\) in half (similar for non-convex).
For many datasets, our feature vectors $x^i$ are very sparse:

<table>
<thead>
<tr>
<th>“CPSC”</th>
<th>“Expedia”</th>
<th>“vicodin”</th>
<th>&lt;recipient name&gt;</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
</tbody>
</table>

Consider case where $d$ is huge but each row $x^i$ has at most $z$ non-zeroes:

- The $O(d)$ cost of stochastic subgradient might be too high.
- We can often modify stochastic subgradient to have $O(z)$ cost.

See bonus slides for details on this issue, and how to handle regularization.

- Various forms of “lazy updates” to deal with non-sparse gradient of regularizer.
Early Stopping

- It’s **hard to decide when to stop** stochastic gradient.
- Common heuristic is “**early stopping**”:
  - Every $m$ iterations, stop and **compute the validation error**.
  - Stop if the validation error starts increasing.

- This can be viewed as a form regularization ("stop overfitting before it happens").
“Hidden” Regularization in Neural Networks

- Fitting single-layer neural network with SGD and no regularization:

- Training error goes to 0 with enough units: we’re finding a global min.

- What should happen to training and test error for larger number of hidden units?
“Hidden” Regularization in Neural Networks

- Fitting single-layer neural network with SGD and no regularization:

![MNIST and CIFAR-10 plots](https://www.neyshabur.net/papers/inductive_bias_poster.pdf)

- Test error continues to go down!?! Where is fundamental trade-off??
- There exists global mins where large number of units leads to accuracy 0.
  - But among the global minima, SGD is somehow converging to “good” ones.
There is growing evidence that using SGD regularizes parameters.

Beyond empirical evidence, we know this happens in simpler cases.

Example:
- Consider a least squares problem where there exists a $w$ with $Xw = y$.
  - Residuals are all zero, we fit the data exactly.
- You run [stochastic] gradient descent starting from $w = 0$.
- Converges to solution $w^*$ of $Xw = y$ that has minimum L2-norm.
  - So using SGD is equivalent to L2-regularization here, but regularization is “implicit”.
Implicit Regularization of SGD

- There is growing evidence that using SGD regularizes parameters.

- Beyond empirical evidence, we know this happens in simpler cases.

- Example:
  - Consider a logistic regression problem where data is linearly separable.
  - We can fit the data exactly.
  - You run [stochastic] gradient descent starting from $w = 0$.
  - Converges to max-margin solution $w^*$ of the problem.
  - So using SGD is equivalent to encouraging large margin (and to SVM solutions).
Should we use Nesterov/Newton-like stochastic methods?
- These do not improve the $O(1/\epsilon)$ convergence rate.

In fact, there is a negative result due to Polyak and Ruppert:
- Classic result is that scaling by $\nabla^2 f(w^*)$ gives optimal asymptotic rate.
- You can get same rate without Hessian, by just averaging the later iterations:
  \[
  \bar{w}^t = \frac{1}{t-k} \sum_{k'=k}^t w^{k'}.
  \]

Practical averaging strategies:
- Could weight all iterations equally.
- Could ignore first half of the iterations then weight equally.
- Could weight proportional to $k$. 
**Stochastic Nesterov/Newton Methods?**

- Some positive results regarding stochastic Nesterov/Newton:
  - Nesterov/Newton can improve dependence on $L$ and $\mu$.

- Two-phase Newton-like method achieves $O(1/\epsilon)$ without strong-convexity.

- AdaGrad method,

  $$w^{k+1} = w^k + \alpha D g_{i_k}, \quad \text{with diagonal} \ D_{jj} = \sqrt{\sum_{k' = 1}^{k} \| \nabla_j f_{i_k}(w^{k'}) \|^2},$$

  improves "regret" but not optimization error.

- Some heuristic extensions of AdaGrad:
  - RMSprop: variant of AdaGrad where step-size does not go to zero.
  - Adam: variant where momentum is added.
  - Recent work: these AdaGrad variants have worse implicit regularization.
Outline

1. More Stochastic Subgradient

2. Mini-Batches and Batching
Better Methods for Smooth Objectives and Finite Datasets?

- **Stochastic methods:**
  - $O(1/\epsilon)$ iterations but requires 1 gradient per iterations.
  - Rates are unimprovable for general stochastic objectives.

- **Deterministic methods:**
  - $O(\log(1/\epsilon))$ iterations but requires $n$ gradients per iteration.
  - The faster rate is possible because $n$ is finite.

For finite $n$, can we design a better method?
Hybrid Deterministic-Stochastic

- Approach 1: control the sample size.
- Deterministic method uses all $n$ gradients,
  \[
  \nabla f(w^k) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w^k).
  \]
- Stochastic method approximates it with 1 sample,
  \[
  \nabla f_{i_k}(w^k) \approx \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w^k).
  \]
- A common variant is to use larger sample $B^k$ ("mini-batch"),
  \[
  \frac{1}{|B^k|} \sum_{i \in B^k} \nabla f_i(w^k) \approx \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w^k),
  \]
  particularly useful for vectorization/parallelization.
- For example, with 16 cores set $|B^k| = 16$ and compute 16 gradients at once.
Mini-Batching as Gradient Descent with Error

- The SG method with a sample $\mathcal{B}^k$ ("mini-batch") uses iterations
  \[ w^{k+1} = w^k - \frac{\alpha_k}{|\mathcal{B}^k|} \sum_{i \in \mathcal{B}^k} \nabla f_i(w^k). \]

- Let’s view this as a “gradient method with error”,
  \[ w^{k+1} = w^k - \alpha_k(\nabla f(w^k) + e^k), \]
  where $e^k$ is the difference between approximate and true gradient.

- If you use $\alpha_k = 1/L$, then using descent lemma this algorithm has
  \[ f(w^{k+1}) \leq f(w^k) - \frac{1}{2L} \| \nabla f(w^k) \|^2 + \frac{1}{2L} \| e^k \|^2, \]
  for any error $e^k$. 
  
  \[ \underline{\text{good}} \quad \underline{\text{bad}} \]
Effect of Error on Convergence Rate

- Our progress bound with $\alpha_k = 1/L$ and error in the gradient of $e^k$ is

$$f(w^{k+1}) \leq f(w^k) - \frac{1}{2L} \| \nabla f(w^k) \|^2 + \frac{1}{2L} \| e^k \|^2.$$

- Connection between “error-free” rate and “with error” rate:
  - If “error-free” rate is $O(1/k)$, you maintain this rate if $\| e^k \|^2 = O(1/k)$.
  - If “error-free” rate is $O(\rho^k)$, you maintain this rate if $\| e^k \|^2 = O(\rho^k)$.
  - If error goes to zero more slowly, then rate that it goes to zero becomes bottleneck.

- So to understanding effect of batch-size, need to know how $|B^k|$ affects $\| e^k \|^2$. 
Effect of Batch Size on Error

- Effect of batch size $|B^k|$ control error size $e^k$.
  - If we sample with replacement we get
    \[
    \mathbb{E}[\|e^k\|^2] = \frac{1}{|B^k|} \sigma^2,
    \]
    where $\sigma^2$ is the variance of the gradient norms.
  - “Doubling the batch size cuts the error in half”.
  - If we sample without replacement from a training set of size $n$ we get
    \[
    \mathbb{E}[\|e^k\|^2] = \frac{n - |B^k|}{n} \frac{1}{|B^k|} \sigma^2,
    \]
    which drives error to zero as batch size approaches $n$.
- For $O(\rho^k)$ linear convergence, need a schedule like $|B^{k+1}| = |B^k|/\rho$.
- For $O(1/k)$ sublinear convergence, need a schedule like $|B^{k+1}| = |B^k| + \text{const.}$
Batching: Growing-Batch-Size Methods

- The SG method with a sample \( B^k \) uses iterations

\[
 w^{k+1} = w^k - \frac{\alpha^k}{|B^k|} \sum_{i \in B^k} \nabla f_i(w^k).
\]

- For a fixed sample size \( |B^k| \), the rate is sublinear.
  - Still need step-size to go to zero to get convergence.

- But we can grow \( |B^k| \) to achieve a faster rate:
  - Early iterations are cheap like SG iterations.
  - Later iterations can use a sophisticated gradient method.
    - No need to set a magical step-size: use a line-search.
    - Can incorporate linear-time approximations to Newton.

- Another approach: at some point switch from stochastic to deterministic:
  - Often after a small number of passes (but hard to know when to switch).
Summary

- **Stochastic subgradient convergence rate:**
  - **Constant step-size:** subgradient quickly converges to approximate solution.
  - **Decreasing step-size:** subgradient slowly converges to exact solution.

- **Practical aspects of stochastic gradient methods:**
  - Sparse datasets, early stopping, “hidden regularization”.
  - Negative and positive results regarding second-order methods.

- **Increasing batch sizes:**
  - Leads to faster rate in terms of iterations.
  - Makes setting the step-size easier.

- Next time: new stochastic methods with fast rates batch size of 1.
The basic **stochastic** subgradient method (for random $i_t$ and $g_{i_t} \in \partial f_{i_t}(x^t)$):

$$x^{t+1} = x^t - \alpha g_{i_t},$$

We can't use descent lemma because $f$ is non-differentiable.

Since function value may not decrease, we analyze distance to $x^*$:

$$\|x^t - x^*\|^2 = \|(x^{t-1} - \alpha g_{i_t}) - x^*\|^2$$

$$= \|(x^{t-1} - x^*) - \alpha g_{i_t}\|^2$$

$$= \|x^{t-1} - x^*\|^2 - 2\alpha g_{i_t}^\top (x^{t-1} - x^*) + \alpha^2 \|g_{i_t}\|^2.$$

Take expectation with respect to $i_t$:

$$\mathbb{E}[\|x^t - x^*\|^2] = \mathbb{E}[\|x^{t-1} - x^*\|^2] - 2\alpha \mathbb{E}[g_{i_t}^\top (x^{t-1} - x^*)] + \alpha^2 \mathbb{E}[\|g_{i_t}\|^2]$$

$$= \underbrace{\|x^{t-1} - x^*\|^2}_{\text{old distance}} - 2\alpha \underbrace{g_t^\top (x^{t-1} - x^*)}_{\text{expected progress}} + \underbrace{\alpha^2 \mathbb{E}[\|g_{i_t}\|^2]}_{\text{"variance"}}.$$

where $g_t$ is a subgradient of $f$ at $w^k$ (expected progress is positive by convexity).
Convergence Rate of Stochastic Subgradient

- Our expected distance given $x^{t-1}$ is
  \[ \mathbb{E}[\|x^t - x^*\|^2] = \|x^{t-1} - x^*\|^2 - 2\alpha_t g_t^\top(x^{t-1} - x^*) + \alpha_t^2 \mathbb{E}[\|g_i\|^2]. \]

- It follows from strong-convexity that (next slide),
  \[ g_t^\top(x^{t-1} - x^*) \geq \mu \|x^{t-1} - x^*\|^2, \]
  which gives (assuming variance is bounded by constant $\sigma^2$):
  \[ \mathbb{E}[\|x^t - x^*\|^2] \leq \|x^{t-1} - x^*\|^2 - 2\alpha_t \mu \|x^{t-1} - x^*\|^2 + \alpha_t^2 \sigma^2 \]
  \[ = (1 - 2\alpha_t \mu) \|x^{t-1} - x^*\|^2 + \alpha_t^2 \sigma^2. \]

- With constant $\alpha_k = \alpha$ (with $\alpha < 2/\mu$) and applying recursively we get (with work)
  \[ \mathbb{E}[\|w^k - w^*\|^2] \leq (1 - 2\alpha \mu)^k \|w^0 - w^*\|^2 + \frac{\alpha \sigma^2}{2\mu}, \]
  where second term bounds a geometric series.
**Strong-Convexity Inequalities for Non-Differentiable $f$**

- A “first-order” relationship between subgradient and strong-convexity:
  - If $f$ is $\mu$-strongly convex then for all $x$ and $y$ we have
    \[ f(y) \geq f(x) + f'(y) \top (y - x) + \frac{\mu}{2} \| y - x \|^2, \]
    for $f'(y) \in \partial f(x)$.
  - The first-order definition of strong-convexity, but with subgradient replacing gradient.
  - Reversing $y$ and $x$ we can write
    \[ f(x) \geq f(y) + f'(x) \top (x - y) + \frac{\mu}{2} \| x - y \|^2, \]
    for $f'(x) \in \partial f(x)$.
  - Adding the above together gives
    \[ (f'(y) - f'(x)) \top (y - x) \geq \mu \| y - x \|^2. \]
  - Applying this with $y = x^{t-1}$ and subgradient $g_t$ and $x = x^*$ (which has $f'(x^*) = 0$ for some subgradient) gives
    \[ (g_t - 0) \top (x^{t-1} - x^*) \geq \mu \| x^{t-1} - x^* \|^2. \]
Convergence Rate of Stochastic Subgradient

- For full details of analyzing stochastic gradient under strong convexity, see:
  - Constant $\alpha_k$: http://circle.ubc.ca/bitstream/handle/2429/50358/stochasticGradientConstant.pdf.

- For both cases under PL, see Theorem 4 here:
Operations on Sparse Vectors

- Consider a vector \( g \in \mathbb{R}^d \) with at most \( z \) non-zeroes:

\[
g^T = [0 \ 0 \ 0 \ 1 \ 2 \ 0 \ -0.5 \ 0 \ 0 \ 0].
\]

- If \( z << d \), we can store the vector using \( O(z) \) storage instead of \( O(d) \):
  - Just store the non-zero values:

\[
g_{\text{value}}^T = [1 \ 2 \ -0.5].
\]

  - Store index of each non-zero ("pointer"):

\[
g_{\text{point}}^T = [4 \ 5 \ 7].
\]

- With this representation, we can do standard vector operations in \( O(z) \):
  - Compute \( \alpha g \) in \( O(z) \) by setting \( g_{\text{value}} = \alpha g_{\text{value}} \).
  - Compute \( w^T g \) in \( O(z) \) by multiplying \( g_{\text{value}} \) by \( w \) at positions \( g_{\text{point}} \).
Consider optimizing the hinge-loss,

$$\arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y^i (w^T x^i)\},$$

when \( d \) is huge but each \( x^i \) has at most \( z \) non-zeroes.

A stochastic subgradient method could use

$$w^{k+1} = w^k - \alpha_k g_k,$$

where

$$g_i = \begin{cases} -y^i x^i & \text{if } 1 - y^i (w^T x^i) > 0 \\ 0 & \text{otherwise} \end{cases}$$

Calculating \( w^{k+1} \) is \( O(z) \) since these are sparse vector operations.

So stochastic subgradient is fast if \( z \) is small even if \( d \) is large.

This is how you “train on all e-mails”: each e-mail has a limited number of words.
Stochastic Subgradient with Sparse Features

- But consider the L2-regularized hinge-loss in the same setting,

$$\arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i (w^T x^i)\} + \frac{\lambda}{2} \|w\|^2,$$

using a stochastic subgradient method,

$$w^{k+1} = w^k - \alpha_k g_{i_k} - \alpha_k \lambda w^k,$$

where $g_{i_k}$ is same as before.

- Problems is that $w^k$ could have $d$ non-zeroes:
  - So adding L2-regularization increases cost from $O(z)$ to $O(d)$?

- There are two standard ways to keep the cost at $O(z)$:
  - L2-regularization: use a $w^k = \beta^k v^k$ (scalar times vector) representation.
  - “Lazy” updates (which work for many regularizers).
But consider the L2-regularized hinge-loss in the same setting,

\[
\arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i (w^T x^i)\} + \frac{\lambda}{2} \|w\|^2,
\]

using a stochastic subgradient method,

\[
w^{k+1} = w^k - \alpha_k g_{i_k} - \alpha_k \lambda w^k, \text{ where } g_{i_k} \text{ is same as before}
\]

Problems is that \(w^t\) could have \(d\) non-zeroes:
- So adding L2-regularization increases cost from \(O(z)\) to \(O(d)\)?
- To use L2-regularization and keep \(O(z)\) cost, re-write iteration as

\[
w^{t+1} = w^t - \alpha_t g_{i_t} - \alpha_t \lambda w^t
\]

\[
= \frac{(1 - \alpha_t \lambda)w^t}{\text{changes scale of } w^t} - \underbrace{\alpha_t g_{i_t}}_{\text{sparse update}}.
\]
Stochastic Subgradient with Sparse Features

- Let's write the update as two steps
  \[ w^{t+\frac{1}{2}} = (1 - \alpha_t \lambda)w^t, \quad w^{t+1} = w^{t+\frac{1}{2}} - \alpha_t g_{it}. \]

- We can implement both steps in \( O(z) \) if we re-parameterize as
  \[ w^t = \beta^t v^t, \]
  for some scalar \( \beta^t \) and vector \( v^t \).

- For the first step we can use
  \[ \beta^{t+\frac{1}{2}} = (1 - \alpha_t \lambda)\beta^t, \quad v^{t+\frac{1}{2}} = v^t. \]
  which costs \( O(1) \).

- For the second step we can use
  \[ \beta^{t+1} = \beta^{t+\frac{1}{2}}, \quad v^{t+1} = v^{t+\frac{1}{2}} - \frac{\alpha_t}{\beta^{t+\frac{1}{2}}} g_{it}, \]
  which costs \( O(z) \).
Lazy Updates for Sparse Features with Dense Regularizers

Consider a feature $j$ that has been zero in the loss for 10 iterations (constant $\alpha$):

$$w_j^k = w_j^{k-1} - 0 - \alpha \lambda w_j^{k-1}$$

$$= (1 - \alpha \lambda)w_j^{k-1}$$

$$= (1 - \alpha \lambda)^2 w_j^{k-2}$$

$$\vdots$$

$$= (1 - \alpha \lambda)^{10} w_j^{k-10}.$$  

So we can apply 10 regularizer gradient steps in $O(1)$.

Lazy updates:

- If $j$ is zero in $g_{ik}$, do nothing.
- If $j$ is non-zero, apply all the old regularizer updates then do the gradient step.
  - Requires keeping a “checkpoint” of the last time each variable was updated.
Lazy Updates for Sparse Features with Dense Regularizers

- **Lazy updates** that track cumulative effects of simple updates.

- Consider the **stochastic proximal-gradient** for L1-regularization:
  - Soft-threshold operator with constant step-size $\alpha$ applies to each element,
    \[ w_{j}^{k+1} = \text{sign}(w_{j}^{k}) \max\{0, |w_{j}^{k}| - \alpha \lambda\}. \]
  - If all that happens to $w_{j}$ for 10 iterations is the proximal operator, we can use
    \[ w_{j}^{k+10} = \text{sign}(w_{j}^{k}) \max\{0, |w_{k}^{k}| - 10\alpha \lambda\}. \]

- **Digression:** **stochastic proximal-gradient** methods:
  - Same convergence rates as basic stochastic gradient method (doesn’t help).
  - Unlike deterministic proximal-gradient method, does not find final non-zero pattern in finite time.
    - Regularized dual averaging is a variant that has this property.