CS-GY 6763: Lecture 8
Second Order Conditions, Online and Stochastic Gradient Descent

NYU Tandon School of Engineering, Prof. Christopher Musco

Given a function f to minimize, assume we have:

- **Function oracle**: Evaluate $f(\mathbf{x})$ for any $\mathbf{x}$.
- **Gradient oracle**: Evaluate $\nabla f(\mathbf{x})$ for any $\mathbf{x}$.

Goal: Minimize the number of oracle calls to find $\tilde{\mathbf{x}}$ such that $f(\tilde{\mathbf{x}}) \leq \min_{\mathbf{x}} f(\mathbf{x}) + \epsilon$.

Prototype gradient descent method:

- Choose starting point $\mathbf{x}^{(0)}$.
- For $i = 0, \ldots, T$:
    - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f(\mathbf{x}^{(i)})$
- Return $\mathbf{x}^{(T)}$ (or similar).

step size
learning rate.

Intuition: Last time we showed that, for sufficiently small η, $f(\mathbf{x}^{(i+1)}) \leq f(\mathbf{x}^{(i)})$. So the algorithm eventually finds a (local) minimum. The question is, how fast.

Assume:

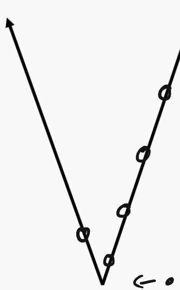
- f is convex.
- Lipschitz function: for all $\mathbf{x}$, $\|\nabla f(\mathbf{x})\|_2 \leq G$.
- Starting radius: $\|\mathbf{x}^* - \mathbf{x}^{(0)}\|_2 \leq R$.

Gradient descent:

- Choose number of steps T.
- Starting point $\mathbf{x}^{(0)}$. E.g. $\mathbf{x}^{(0)} = \vec{0}$.
- $\eta = \frac{R}{G\sqrt{T}}$
- For $i = 0, \ldots, T$:
  - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f(\mathbf{x}^{(i)})$
- Return $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}^{(i)}} f(\mathbf{x}^{(i)})$.

Claim (GD Convergence Bound)

If we run GD for $T \geq \frac{R^2 G^2}{\epsilon^2}$ iterations then $f(\hat{\mathbf{x}}) \leq f(\mathbf{x}^*) + \epsilon$.



Proof was made tricky by the fact that $f(\mathbf{x}^{(i)})$ does not improve monotonically. We can "overshoot" the minimum.

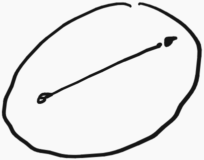
PROJECTED GRADIENT DESCENT

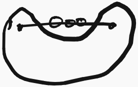
Given function $f(\mathbf{x})$, set $\mathcal{S}$, and access to projection oracle $P_{\mathcal{S}}(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathcal{S}} \|\mathbf{x}-\mathbf{y}\|_2$.

$\min_{x \in S} f(x)$

Projected gradient descent:

- Select starting point $\mathbf{x}^{(0)}$, $\eta = \frac{R}{G\sqrt{T}}$.
- For $i = 0, \ldots, T$:
    - $\mathbf{z} = \mathbf{x}^{(i)} - \eta \nabla f(\mathbf{x}^{(i)})$
    - $\mathbf{x}^{(i+1)} = P_{\mathcal{S}}(\mathbf{z})$
- Return $\hat{\mathbf{x}} = \arg\min_i f(\mathbf{x}^{(i)})$.



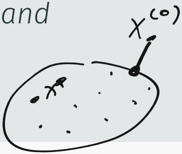


$\{c : f(x) = c\}$

Claim (PGD Convergence Bound)

If $f, \mathcal{S}$ are convex, $\|\nabla f(\mathbf{x})\|_2 \leq G$ for all $\mathbf{x} \in \mathcal{S}$ and $\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2 \leq R$.

If $T \geq \frac{R^2G^2}{\epsilon^2}$, then $f(\hat{\mathbf{x}}) \leq f(\mathbf{x}^*) + \epsilon$.



$\epsilon = .1$ $\frac{1}{(10000)^2}$

The previous bounds are optimal for convex first order optimization in general.

But in practice, the dependence on $1/\epsilon^2$ is pessimistic: gradient descent typically requires far fewer steps to reach ϵ error.

Previous bounds only make a very weak first order assumption:

$$\|\nabla f(x)\|_2 \leq G.$$

In practice, many function satisfy stronger assumptions.

$|f'(x)| \leq G$

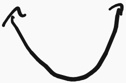
SECOND ORDER CONDITIONS

Today we will talk about assumptions that involve the second derivative of f.

$\alpha > 0$

In particular, we say that a scalar function f is α-strongly convex and β-smooth if for all x:

$0 \leq f''(x)$



$$\alpha \leq f''(x) \leq \beta.$$

> 0

We will give appropriate generalizations of these conditions to multi-dimensional functions shortly.

Take away: Having either an upper and lower bound on the second derivative helps convergence. Having both helps a lot.

$\frac{1}{\epsilon}$ instead of $1/\epsilon^2$

Take away: Having either an upper and lower bound on the second derivative helps convergence. Having both helps a lot.

Number of iterations for ϵ error:

	G-Lipschitz	β-smooth
R bounded start	$O\left(\frac{G^2R^2}{\epsilon^2}\right)$	$O\left(\frac{\beta R^2}{\epsilon}\right)$
α-strong convex	$O\left(\frac{G^2}{\alpha\epsilon}\right)$	$O\left(\frac{\beta}{\alpha}\log(1/\epsilon)\right)$

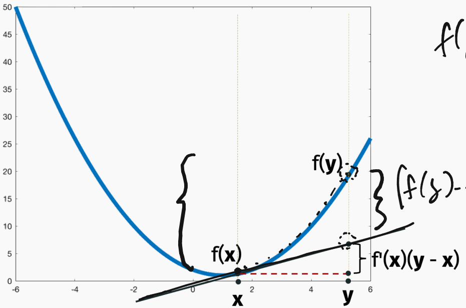
As we defined them so far, smoothness and strong convexity require f to be twice differentiable. On the other hand, gradient descent only requires first order differentiability.

SECOND ORDER CONDITIONS

Equivalent conditions: for y, x

$$\underline{f''(x) \leq \beta} \iff \left([f(y) - f(x)] - f'(x)(y - x) \leq \frac{\beta}{2}(y - x)^2 \right)$$

$$f''(x) \geq \alpha \iff [f(y) - f(x)] - f'(x)(y - x) \geq \underline{\frac{\alpha}{2}(y - x)^2}$$



$[f(y) - f(x)] - f'(x)(y - x)$

$f(y) - f(x) \geq f'(x)(y - x)$ for a convex function.

Recall: For all convex functions $[\underline{f(y)} - \underline{f(x)}] - f'(x)(y - x) \geq 0$.

Proof that $f''(x) \leq \beta \Rightarrow [f(y) - f(x)] - f'(x)(y - x) \leq \frac{\beta}{2}(y - x)^2$:

$$f(y) - f(x) = \int_x^y f'(t)\,dt$$

$$\leq \int_x^y f'(x) + (t-x)\beta \,dt$$

$$= f'(x)(y-x) + \int_x^y \beta(t-x)\,d$$

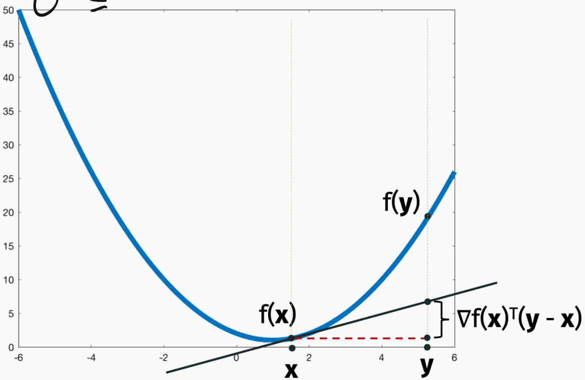
$$= f'(x)(y-x) + \underbrace{\beta \frac{(t-x)^2}{2}\Big|_x^y}_{= \frac{\beta}{2}(y-x)^2}$$

Proof for α-strongly convex is similar, as are the other directions.

A function is α-strongly convex and β-smooth if for all $\mathbf{x}, \mathbf{y}$:

$$\frac{\alpha}{2}\|\mathbf{y}-\mathbf{x}\|_2^2 \leq [f(\mathbf{y})-f(\mathbf{x})] - \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \leq \frac{\beta}{2}\|\mathbf{y}-\mathbf{x}\|_2^2$$

$0 \leq$



Definition (β-smoothness)

A function f is β smooth if, for all $\mathbf{x}, \mathbf{y}$

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq \beta \|\mathbf{x} - \mathbf{y}\|_2$$

I.e., the gradient function is a β-Lipschitz function.

We won't use this definition directly, but it's good to know. Easy to prove equivalency to previous definition (see Lem. 3.4 in Bubeck's book).

Theorem (GD convergence for β-smooth functions.)

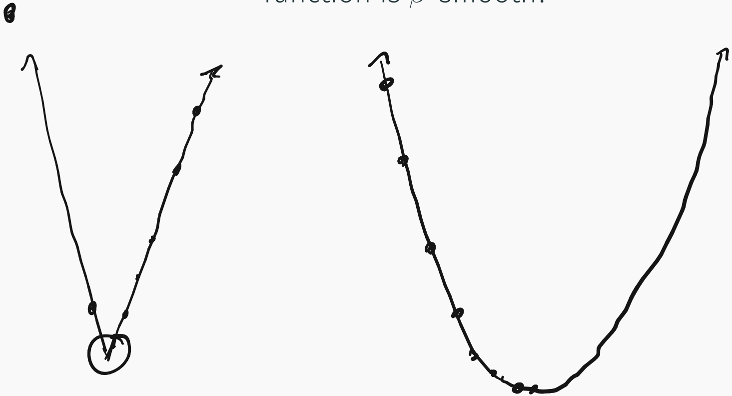
Let f be a β smooth convex function and assume we have $\|\mathbf{x}^* - \mathbf{x}^{(0)}\|_2 \leq R$. If we run GD for T steps, we have:

$$f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \frac{2\beta R^2}{T}$$

Corollary: If $T = O\left(\frac{\beta R^2}{\epsilon}\right)$ we have $f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \epsilon$.

Compare this to $T = \underline{O\left(\frac{G^2 R^2}{\epsilon^2}\right)}$ without a smoothness assumption.

Why do you think gradient descent might be faster when a function is β-smooth?



$\frac{1}{G}$

GUARANTEED PROGRESS

Previously learning rate/step size η depended on G. Now choose it based on β:

$$\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \frac{1}{\beta}\nabla f(\mathbf{x}^{(t)})$$

$-\frac{1}{\beta}\nabla f(x^{(t)})$

Progress per step of gradient descent:

1. $[f(\mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)})] - \nabla f(\mathbf{x}^{(t)})^T(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}) \leq \frac{\beta}{2}\|\mathbf{x}^{(t)} - \mathbf{x}^{(t+1)}\|_2^2$.

   $-(-\frac{1}{\beta}\nabla f(x^{(t)}) + \nabla f(x^{(t)}))$ $\quad \|1/\beta\, \nabla f(x^{(t)})\|_2^2$

2. $[f(\mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)})] + \frac{1}{\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{\beta}{2}\|\frac{1}{\beta}\nabla f(\mathbf{x}^{(t)})\|_2^2$.

   $\frac{1}{2\beta}\|\nabla f(x^{(t)})\|_2^2$

3. $f(\mathbf{x}^{(t)}) - f(\mathbf{x}^{(t+1)}) \geq \frac{1}{2\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2$.

Once we have the bound from the previous page, proving a convergence result isn't hard, but not obvious. A concise proof can be found in Page 15 in Garrigos and Gower's notes.

Theorem (GD convergence for β-smooth functions.)

Let f be a β smooth convex function and assume we have $\|\mathbf{x}^* - \mathbf{x}^{(1)}\|_2 \leq R$. If we run GD for T steps with $\eta = \frac{1}{\beta}$ we have:

$$f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \frac{2\beta R^2}{T}$$

Corollary: If $T = O\left(\frac{\beta R^2}{\epsilon}\right)$ we have $f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \epsilon$.

Where did we use convexity in this proof?

Progress per step of gradient descent:

1. $\left[f(\mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)})\right] - \nabla f(\mathbf{x}^{(t)})^T(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}) \leq \frac{\beta}{2}\|\mathbf{x}^{(t)} - \mathbf{x}^{(t+1)}\|_2^2$.

2. $\left[f(\mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)})\right] + \frac{1}{\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{\beta}{2}\|\frac{1}{\beta}\nabla f(\mathbf{x}^{(t)})\|_2^2$.

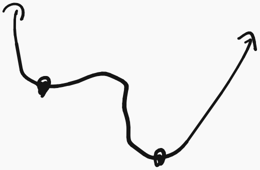
3. $f(\mathbf{x}^{(t)}) - f(\mathbf{x}^{(t+1)}) \geq \frac{1}{2\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2$.

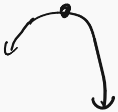
Definition (Stationary point)

For a differentiable function f, a stationary point is any $\mathbf{x}$ with:

$$\nabla f(\mathbf{x}) = \mathbf{0}$$

local/global minima - local/global maxima - saddle points



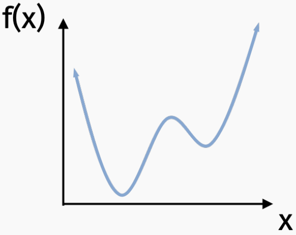


Theorem (Convergence to Stationary Point)

For any β-smooth differentiable function f (convex or not), if we run GD for T steps, we can find a point $\hat{\mathbf{x}}$ such that:

$$\|\nabla f(\hat{\mathbf{x}})\|_2^2 \leq \frac{2\beta}{T}\left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^*)\right)$$

Corollary: If $T \geq \frac{2\beta}{\epsilon}$, then $\|\nabla f(\hat{\mathbf{x}})\|_2^2 \leq \epsilon\left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^*)\right)$.



TELESCOPING SUM PROOF

Theorem (Convergence to Stationary Point)

For any β-smooth differentiable function f (convex or not), if we run GD for T steps, we can find a point $\hat{\mathbf{x}}$ such that:

$$\|\nabla f(\hat{\mathbf{x}})\|_2^2 \leq \frac{2\beta}{T}\left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^*)\right)$$

$f(x^{(0)}) - \cancel{f(x^{(1)})} + (\cancel{f(x^{(1)})}$

$- \cancel{f(x^{(2)})})$

$\vdots$

$+ (f(x^{(t-1)}) - f(x^{(t)}))$

We have that $\frac{1}{2\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq f(\mathbf{x}^{(t)}) - f(\mathbf{x}^{(t+1)})$. So:

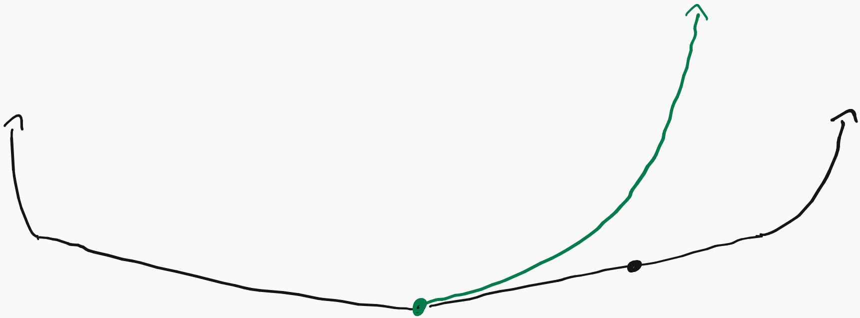
$$\sum_{t=0}^{T-1} \frac{1}{2\beta}\|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq f(\mathbf{x}^{(0)}) - f(\mathbf{x}^{(T)})$$

$$\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{2\beta}{T}\left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^*)\right)$$

$$\min_t \|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{2\beta}{T}\left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^*)\right)$$

BACK TO CONVEX FUNCTIONS

I said it was a bit tricky to prove that $f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \frac{2\beta R^{\cancel{2}}}{T}$ for convex functions. But we just easily proved that $\|\nabla f(\hat{\mathbf{x}})\|_2^2$ is small. Why doesn't this show we are close to the minimum?



STRONG CONVEXITY

Definition (α-strongly convex)

A convex function f is α-strongly convex if, for all $\mathbf{x}, \mathbf{y}$

$$[f(\mathbf{y}) - f(\mathbf{x})] - \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) \geq \frac{\alpha}{2}\|\mathbf{x} - \mathbf{y}\|_2^2$$

Compare to smoothness condition.

$$[f(\mathbf{y}) - f(\mathbf{x})] - \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) \leq \frac{\beta}{2}\|\mathbf{x} - \mathbf{y}\|_2^2.$$

For a twice-differentiable scalar function f, equivalent to $f''(x) \geq \alpha$.

When f is convex, we always have that $f''(x) \geq 0$, so larger values of α correspond to a “stronger” condition.

Gradient descent for strongly convex functions:

- Choose number of steps T.
- For $i = 0, \ldots, T$:
  - $\eta = \frac{2}{\alpha \cdot (i+1)}$
  - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f(\mathbf{x}^{(i)})$
- Return $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}^{(i)}} f(\mathbf{x}^{(i)})$.

Theorem (GD convergence for α-strongly convex functions.)

Let f be an α-strongly convex function and assume we have that, for all $\mathbf{x}$, $\|\nabla f(\mathbf{x})\|_2 \leq G$. If we run GD for T steps (with adaptive step sizes) we have:

$$f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \frac{2G^2}{\alpha T}$$

Corollary: If $T = O\left(\frac{G^2}{\alpha\epsilon}\right)$ we have $f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \epsilon$

$f(\hat{x}) - f(x^*) \leq \varepsilon$

We could also have that f is both β-smooth and α-strongly convex.

Theorem (GD for β-smooth, α-strongly convex.)

Let f be a β-smooth and α-strongly convex function. If we run GD for T steps (with step size $\eta = \frac{1}{\beta}$) we have:

$$\frac{2}{\beta}\left[f(x^{(T)}) - f(x^*)\right] \leq \|\mathbf{x}^{(T)} - \mathbf{x}^*\|_2^2 \leq e^{-T\frac{\alpha}{\beta}} \|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2^2 \leq e^{-T\frac{\alpha}{\beta}} R^2$$

$\kappa = \frac{\beta}{\alpha}$ is called the "condition number" of f.

Is it better if κ is large or small?

$e^{-T\frac{\alpha}{\beta}} = \varepsilon$ $\quad\quad$ $T = \frac{\beta}{\alpha} \log(1/\varepsilon)$

Converting to more familiar form: Using that fact the $\nabla f(\mathbf{x}^*) = \mathbf{0}$ along with $x = x^*$ $y = x^{(T)}$

$$\frac{\alpha}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 \leq [f(\mathbf{y}) - f(\mathbf{x})] - \cancel{\nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x})} \leq \frac{\beta}{2}\|\mathbf{x}-\mathbf{y}\|_2^2,$$

we have:

$$\|\mathbf{x}^{(T)} - \mathbf{x}^*\|_2^2 \geq \frac{2}{\beta}\left[f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*)\right].$$

We also assume

$$\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2^2 \leq R^2.$$

CONVERGENCE GUARANTEE

Corollary (GD for β-smooth, α-strongly convex.)

Let f be a β-smooth and α-strongly convex function. If we run GD for T steps (with step size $\eta = \frac{1}{\beta}$) we have:

$$f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \frac{\cancel{\beta}}{2} e^{-T\frac{\alpha}{\beta}} \cdot \cancel{R^2}$$

Corollary: If $T = O\left(\frac{\beta}{\alpha}\log(R\beta/\epsilon)\right)$ we have:

$$f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) \leq \epsilon$$

Only depend on $\log(1/\epsilon)$ instead of on $1/\epsilon$ or $1/\epsilon^2$!

x^2 $2x$ 2

$A \in \mathbb{R}^{n \times d}$

$x \in \mathbb{R}^{d}$

$b \in \mathbb{R}^{n}$

We are going to prove the guarantee on the previous page for the special case of:

$$f(\mathbf{x}) = \frac{1}{2}\|\underline{\mathbf{A}\mathbf{x} - \mathbf{b}}\|_2^2$$

Goal: Get some of the key ideas across, introduces important concepts like the Hessian, and show the connection between conditioning and linear algebra.

Let f be a twice differentiable function from $\mathbb{R}^d \to \mathbb{R}$. Let the **Hessian** $\nabla^2 f(\mathbf{x})$ contain all of its second derivatives at a point $\mathbf{x}$. So the Hessian is a $d \times d$ matrix and we have:

$$\left[\nabla^2 f(\mathbf{x})\right]_{j,k} = \frac{\partial^2 f}{\partial x_j x_k}.$$

$f(x+tv)$
$\approx f(x) + t\langle v, \nabla f(x)\rangle$

For vector $\mathbf{x}, \mathbf{v}$:

$$\nabla f(\mathbf{x} + t\mathbf{v}) \approx \nabla f(\mathbf{x}) + t\underbrace{\left[\nabla^2 f(\mathbf{x})\right]}_{d \times d} \mathbf{v}.$$

$\mathbb{R}^d$

$$\lim_{t \to 0} \frac{\nabla f(x+tv) - \nabla f(x)}{t} = \left[\nabla^2 f(x)\right] v.$$

Let f be a twice differentiable function from $\mathbb{R}^d \to \mathbb{R}$. Let the **Hessian** $\nabla^2 f(\mathbf{x})$ contain all of its second derivatives at a point $\mathbf{x}$. So the Hessian is a $d \times d$ matrix and we have:

x^2

$$\left[\nabla^2 f(\mathbf{x})\right]_{j,k} = \frac{\partial^2 f}{\partial x_j x_k}.$$

ith row of a.

Example: $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 = \frac{1}{2}\sum_{i=1}^n \left(\mathbf{x}^T\mathbf{a}^{(i)} - b^{(i)}\right)^2$

$$\frac{\partial f}{\partial x_k} = \cancel{\frac{1}{2}}\sum_{i=1}^n \cancel{2}\left(\mathbf{x}^T\mathbf{a}^{(i)} - b^{(i)}\right) \cdot a_k^{(i)}$$

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \left(\sum_{i=1}^n a_j^{(i)} a_k^{(i)}\right) = a_j^T a_k$$

where a_j, a_k are jth and kth columns of A.

$$\nabla^2 f(\mathbf{x}) = A^T A$$

ALTERNATIVE DERIVATION

$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$. Recall that $\nabla f(\mathbf{x}) = \frac{1}{2} \cdot 2\mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{b})$. $= A^T(Ax - b)$

$$\lim_{t \to 0} \frac{\nabla f(x + tv) - \nabla f(x)}{t} = \underbrace{\left[\nabla^2 f(x)\right]}\, v.$$

$$\frac{A^T(A(x+tv) - b) - A^T(Ax - b)}{t} = \frac{\cancel{A^TAx} + A^TAtv - \cancel{A^Tb} - \cancel{A^TAx} + \cancel{A^Tb}}{t}$$

$$\frac{A^TA\cancel{t}v}{\cancel{t}} = A^TAv$$

$$\downarrow$$

$$= \nabla^2 f(x)$$

IMPORTANT NOTE

The Hessian matrix is symmetric if for all j, k,

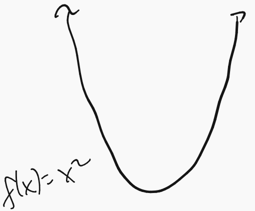
$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \frac{\partial^2 f}{\partial x_k \partial x_j}$$

I.e. the order of differentiation does not matter. This is true whenever the second derivatives are continuous, which we will assume is the case.

CONVEXITY IN ONE DIMENSION

A twice-differentiable function $f : \mathbb{R} \rightarrow R$ is :

- convex if and only if $f''(x) \geq 0$ for all x.
- β-smooth if $f''(x) \leq \beta$.
- α-strongly convex if $f''(x) \geq \alpha$.



2

How do these statements generalize to the case when f has a vector input, so the second derivative is a matrix $\nabla^2 f(\mathbf{x})$?

Claim: If f is twice differentiable, then it is convex if and only if the matrix $\nabla^2 f(\mathbf{x})$ is positive semidefinite for all $\mathbf{x}$.

Definition (Positive Semidefinite (PSD))

A square, symmetric matrix $\mathbf{H} \in \mathbb{R}^{d \times d}$ is positive semidefinite (PSD) for any vector $\mathbf{y} \in \mathbb{R}^d$, $\mathbf{y}^T \mathbf{H} \mathbf{y} \geq 0$.

This is a natural notion of "positivity" for symmetric matrices. To denote that $\mathbf{H}$ is PSD we will typically use "Loewner order" notation (`\succeq` in LaTex):

$$\mathbf{H} \succeq 0.$$

We write $\mathbf{B} \succeq \mathbf{A}$ or equivalently $\mathbf{A} \preceq \mathbf{B}$ to denote that $(\mathbf{B} - \mathbf{A})$ is positive semidefinite. This gives a partial ordering on matrices.

$B - A \succeq 0$ $\quad$ $A \npreceq B$ $\quad$ $B - A \nsucceq 0$

Claim: If f is twice differentiable, then it is convex if and only if the matrix $\nabla^2 f(\mathbf{x})$ is positive semidefinite for all $\mathbf{x}$.

Definition (Positive Semidefinite (PSD))

A square, symmetric matrix $\mathbf{H} \in \mathbb{R}^{d \times d}$ is positive semidefinite (PSD) for any vector $\mathbf{y} \in \mathbb{R}^d$, $\mathbf{y}^T \mathbf{H} \mathbf{y} \geq 0$.

For the least squares regression loss function:
$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$, $\nabla^2 f(\mathbf{x}) = \mathbf{A}^T\mathbf{A}$ for all $\mathbf{x}$. Is $\nabla^2 f(\mathbf{x})$ PSD?

For any y, $y^T A^T A y \geq 0$.

$\langle Ay, Ay \rangle = \|Ay\|_2^2 \geq 0$.

If f is β-smooth and α-strongly convex then at any point $\mathbf{x}$, the Hessian $\nabla^2 f(\mathbf{x})$ satisfies:

$$\alpha \mathbf{I} \preceq \nabla^2 f(\mathbf{x}) \preceq \beta \mathbf{I},$$

where $\mathbf{I}$ is a $d \times d$ identity matrix.

This is the natural matrix generalization of the statement for scalar valued functions:

$$\alpha \leq f''(x) \leq \beta.$$

$$\alpha \mathbf{I}_{d\times d} \preceq \nabla^2 f(\mathbf{x}) \preceq \beta \mathbf{I}_{d\times d}.$$

Equivalently for any $\mathbf{z}$,

$$\alpha\|\mathbf{z}\|_2^2 \leq \mathbf{z}^T[\nabla^2 f(\mathbf{x})]\mathbf{z} \leq \beta\|\mathbf{z}\|_2^2.$$

$\underbrace{\beta I - \nabla^2 f(x)}_{\text{Is PSD.}} \succeq 0$

For all z, $z^T(\beta I - \nabla^2 f(x))z \geq 0$

$\beta \underbrace{z^T z}_{\|z\|_2^2} - z^T \nabla^2 f(x) z \geq 0$

Let $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2$ where $\mathbf{D}$ is a diagonal matrix. For now imagine we're in two dimensions: $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, $\mathbf{D} = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}$.

What are α, β for this problem?

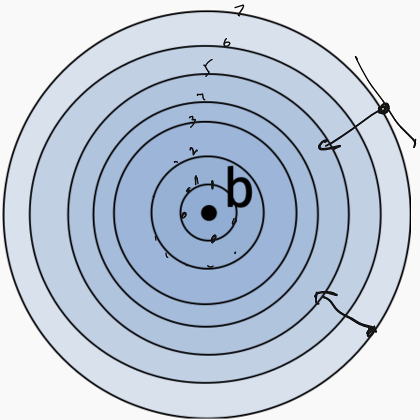
$D^T D = D^2 = \begin{bmatrix} d_1^2 & 0 \\ 0 & d_2^2 \end{bmatrix}$

$$\alpha\|\mathbf{z}\|_2^2 \leq \mathbf{z}^T[\nabla^2 f(\mathbf{x})]\mathbf{z} \leq \beta\|\mathbf{z}\|_2^2$$

$\min(d_1^2, d_2^2)$

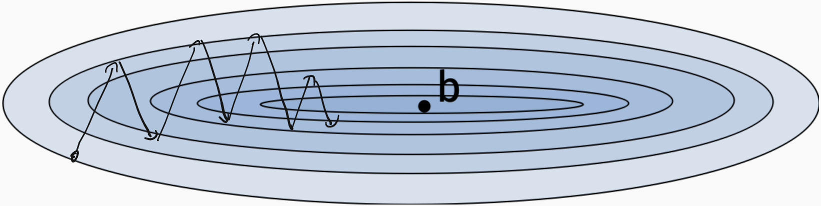
$\max(d_1^2, d_2^2)$

$z = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ $z = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$



Level sets of $\frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2$ when $\underline{d_1^2} = 1, \underline{d_2^2 =} 1$.

$= \frac{1}{2} \|x - b\|_2^2$



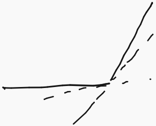
Level sets of $\frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2$ when $d_1^2 = \frac{1}{3}, d_2^2 = 2$.

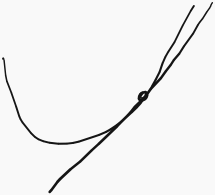
$\beta = 2$

$\alpha = 1/3$

$\kappa \overset{\text{def}}{=} \frac{\beta}{\alpha} = 6$

condition #

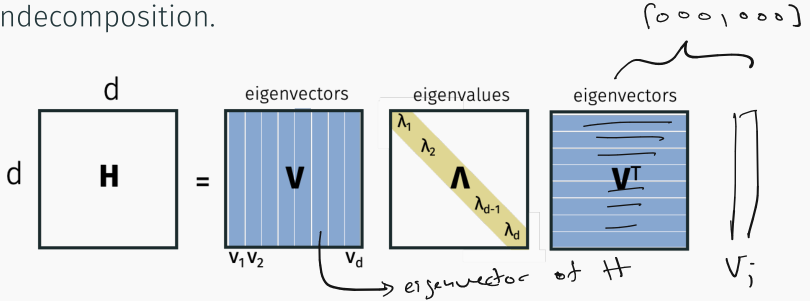




What about non-diagonal $\mathbf{D}$?

EIGENDECOMPOSITION VIEW

Any symmetric matrix $\mathbf{H}$ has an orthogonal, real valued eigendecomposition.



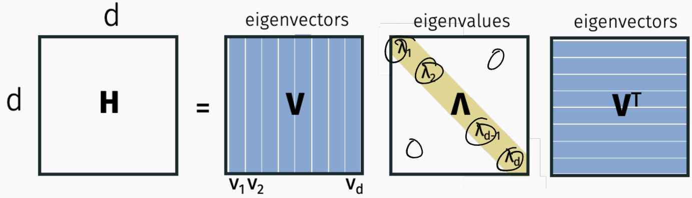
Here $\mathbf{V}$ is square and orthogonal, so $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}$. And for each $\mathbf{v}_i$, we have:

$$\mathbf{H}\mathbf{v}_i = \lambda_i \mathbf{v}_i.$$

By definition, that's what makes $\mathbf{v}_1, \ldots, \mathbf{v}_d$ eigenvectors.

EIGENDECOMPOSITION VIEW

Recall $\mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}$.



Claim: H is PSD $\Leftrightarrow \lambda_1, \ldots, \lambda_d \geq 0$. **Proof for** $\Leftarrow$:

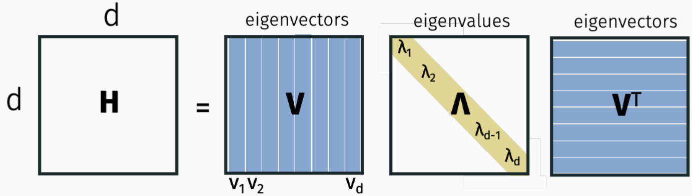
$$H = V\sqrt{\Lambda}\sqrt{\Lambda}V^T = (\sqrt{\Lambda}V^T)^T(\sqrt{\Lambda}V^T) = C^T C$$

$$z^T H z = \|Cz\|_2^2 \geq 0.$$

EIGENDECOMPOSITION VIEW

Recall $\mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}$.

$\lambda_1 \geq \lambda_2 \ldots \geq \lambda_d$

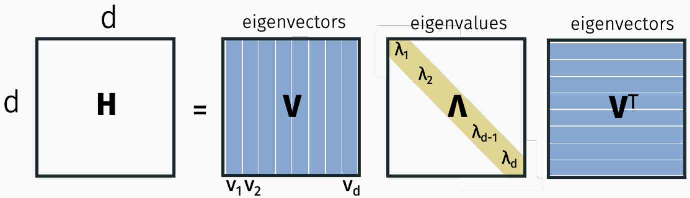


Claim: $\alpha \mathbf{I} \preceq \mathbf{H} \preceq \beta \mathbf{I} \Leftrightarrow \alpha \leq \lambda_d \leq \ldots \leq \lambda_1 \leq \beta$.

$H \preceq \beta . I \qquad (\beta \cdot I - H) \succeq 0$

$\beta \cdot VV^T - V\Lambda V^T = V(\beta . I - \Lambda)V^T$

$(\beta \cdot I - \Lambda)_{ii} \geq 0$ for all i.



Recall that if $\lambda_{\text{max}}(\mathbf{H})$ and $\lambda_{\text{min}}(\mathbf{H})$ be the smallest and largest eigenvalues of $\mathbf{H}$, then for all $\mathbf{z}$ we have:

$$\mathbf{z}^T\mathbf{H}\mathbf{z} \leq \lambda_{\text{max}}(\mathbf{H}) \cdot \|\mathbf{z}\|^2$$
$$\mathbf{z}^T\mathbf{H}\mathbf{z} \geq \lambda_{\text{min}}(\mathbf{H}) \cdot \|\mathbf{z}\|^2$$

If for all $\mathbf{x}$ the maximum eigenvalue of $\nabla^2 f(\mathbf{x})$ is $\leq \beta$ and the minimum eigenvalue of $\nabla^2 f(\mathbf{x})$ is $\geq \alpha$ then $f(\mathbf{x})$ is β-smooth and α-strongly convex.

Note that for $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$, we have that, for all $\mathbf{x}$, $\nabla^2 f(\mathbf{x}) = \mathbf{A}^T\mathbf{A}$. So, we can take $\alpha = \lambda_{\min}(\mathbf{A}^T\mathbf{A})$ and $\beta = \lambda_{\max}(\mathbf{A}^T\mathbf{A})$.

POLYNOMIAL VIEW POINT

Theorem (GD for β-smooth, α-strongly convex.)

Let f be a β-smooth and α-strongly convex function. If we run GD for S steps (with step size $\eta = \frac{1}{\beta}$) we have:

$$\|\mathbf{x}^{(S)} - \mathbf{x}^*\|_2^2 \leq e^{-S/\kappa}\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2^2$$

Goal: Prove for $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$.

Let $\lambda_{\max} = \lambda_{\max}(\mathbf{A}^T\mathbf{A})$ and set step size $\eta = \frac{1}{\lambda_{max}}$. Gradient descent update is:

$$\left(\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \frac{1}{\lambda_{\max}}\mathbf{A}^T(\mathbf{A}\mathbf{x}^{(t)} - \mathbf{b}) \right)$$

μ

$\nabla f(x^*) = 0$ $\quad A^T(Ax^* - b) = 0$

$A^TAx^* = A^Tb$

Richardson Iteration view:

$$\underline{(x^{(t+1)} - x^*)} = \underbrace{\left(I - \frac{1}{\lambda_{\max}} A^T A\right) \underline{(x^{(t)} - x^*)}}$$

$$\underbrace{x^{(t)} - x^* - \frac{1}{\lambda_{max}} A^TA\,x^{(t)} + \frac{1}{\lambda_{max}} A^TAx^*}$$

$$= x^{(t)} - x^* \underbrace{- \frac{1}{\lambda_{max}} A^TA\,x^{(t)} + \frac{1}{\lambda_{max}} A^Tb}_{\frac{1}{\lambda_{max}} A^T(Ax^{(t)} - b)}$$

$$= \underbrace{x^{(t)} - \frac{1}{\lambda_{max}} A^T(Ax^{(t)} - b)}_{x^{(t+1)}} - x^*$$

$$(x^{(S)} - x^*) = \left(I - \frac{1}{\lambda_{max}} A^T A\right)^S (x^{(0)} - x^*)$$

$$x^{(1)} - x^* = \left(I - \frac{1}{\lambda_{max}} A^T A\right)(x^0 - x^*)$$

$$(x^{(2)} - x^*) = \left(I - \frac{1}{\lambda_{max}} A^T A\right)(x^{(1)} - x^*)$$

$$\vdots$$

$$(x^{(2)} - x^*) = \left(I - \frac{1}{\lambda_{max}} A^T A\right)^2 (x^0 - x^*)$$

$$(\mathbf{x}^{(S)} - \mathbf{x}^*) = \left(\overbrace{\left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{A}^T\mathbf{A}\right)}^{B}{}^{S}(\mathbf{x}^{(0)} - \mathbf{x}^*)\right)$$

Conclusion: $\|\mathbf{x}^{(S)} - \mathbf{x}^*\|_2^2 \leq (x^{(0)} - x^*)^T B^2 (x^{(0)} - x^*)$

$$\leq \|x^{(0)} - x^*\|_2 \cdot \lambda_{max}(B^2).$$

H

$$|z^T H z| \leq \|z\|_2^2 \, |\lambda_{max}(H)|$$

Approach: Show that the maximum eigenvalue of $\left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{A}^T\mathbf{A}\right)^{2S}$ is small – i.e., bounded by $e^{-S/\kappa} = \epsilon$.

$$(\mathrm{x}^{(S)} - \mathrm{x}^*) = \left(\mathrm{I} - \frac{1}{\lambda_{\max}} \mathrm{A}^T\mathrm{A}\right)^{S} (\mathrm{x}^{(0)} - \mathrm{x}^*)$$

What is the maximum eigenvalue of the symmetric matrix $\left(\mathrm{I} - \frac{1}{\lambda_{\max}} \mathrm{A}^T\mathrm{A}\right)$ in terms of the eigenvalues of $\mathrm{A}^T\mathrm{A}$?

$$VV^T - \frac{1}{\lambda_{max}} V \Lambda V^T = V\left(I - \frac{1}{\lambda_{max}}\Lambda\right)V^T = I - \frac{1}{\lambda_{max}}(A^TA)$$

$$1 - \frac{\lambda_1}{\lambda_{max}}$$

$$\lambda_{max}\left(I - \frac{1}{\lambda_{max}} A^TA\right) = 1 - \frac{\lambda_{min}(A^TA)}{\lambda_{max}(A^TA)}$$

$$1 - \frac{\lambda_2}{\lambda_{max}}$$

$$\vdots$$

$$1 - \frac{\lambda_d}{\lambda_{max}} \quad \lambda_d = \lambda_{min}$$

$(1-\frac{1}{\kappa})^{\kappa} \leq 1/e$ when $\kappa \geq 2$

$$(\mathbf{x}^{(S)} - \mathbf{x}^*) = \left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{A}^T\mathbf{A}\right)^S (\mathbf{x}^{(0)} - \mathbf{x}^*)$$

What is the maximum eigenvalue of $\underbrace{\left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{A}^T\mathbf{A}\right)}_{B}^{2S}$?

$B = V \Lambda V^T$

$B^s = V \Lambda \cancel{V^T V} \Lambda \cancel{V^T} \ldots V \Lambda V^T = V \Lambda^s V^T$

$\lambda_{\max}(B) = 1 - \frac{\lambda_{\min}}{\lambda_{\max}}$ $\quad \lambda_{\max}(B^{2s}) = \left(1 - \frac{\lambda_{\min}}{\lambda_{\max}}\right)^{2s}$

$= (1 - 1/\kappa)^{2s}$

$= \left[\left(1 - \frac{1}{\kappa}\right)^{2s\kappa}\right]^{1/\kappa} \leq \left(\frac{1}{e}\right)^{s/\kappa} = e^{-s/\kappa}.$

ACCELERATION

ACCELERATED GRADIENT DESCENT

Nesterov's accelerated gradient descent:

$e^{-s/\kappa}$

$s = \kappa \log(1/\epsilon)$

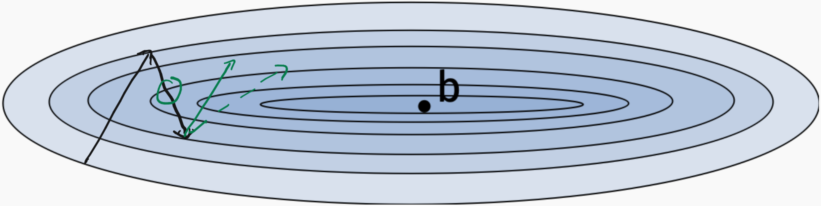
- $\mathbf{x}^{(0)} = \mathbf{y}^{(1)} = \mathbf{z}^{(1)}$
- For $t = 1, \ldots, T$
  - $\mathbf{y}^{(t+1)} = \mathbf{x}^{(t)} - \frac{1}{\beta}\nabla f(\mathbf{x}^{(t)})$
  - $\mathbf{x}^{(t+1)} = \left(1 + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)\mathbf{y}^{(t+1)} + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\left(\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)}\right)$

Theorem (AGD for β-smooth, α-strongly convex.)

Let f be a β-smooth and α-strongly convex function. If we run AGD for S steps we have:

$$\|\mathbf{x}^{(S)} - \mathbf{x}^*\|_2 \leq e^{-S/\sqrt{\kappa}}\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2$$

Corollary: If $T = O\left(\sqrt{\kappa}\log(\beta R/\epsilon)\right)$ achieve error ϵ.



Level sets of $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$.

Other terms for similar ideas:

- Momentum
- Heavy-ball methods

What if we look back beyond two iterates?

BREAK

Second part of class:

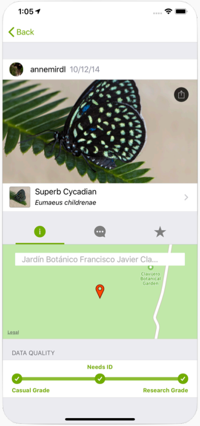
- Basics of Online Learning + Optimization.
- Introduction to Regret Analysis.
- Application to analyzing Stochastic Gradient Descent.

Many machine learning problems are solved in an online setting with constantly changing data.

- Spam filters are incrementally updated and adapt as they see more examples of spam over time.
- Image classification systems learn from mistakes over time (often based on user feedback).
- Content recommendation systems adapt to user behavior and clicks (which may not be a good thing...)

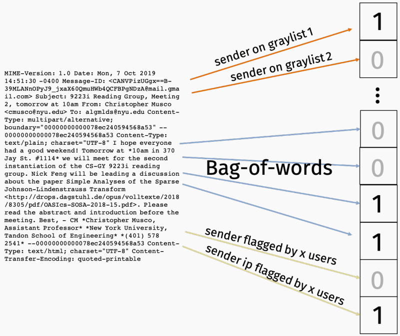
Plant identification via iNaturalist app.

(California Academy of Science + National Geographic)



- When the app fails, image is classified via crowdsourcing (backed by huge network of amateurs and experts).
- Single model that is updated constantly, not retrained in batches.

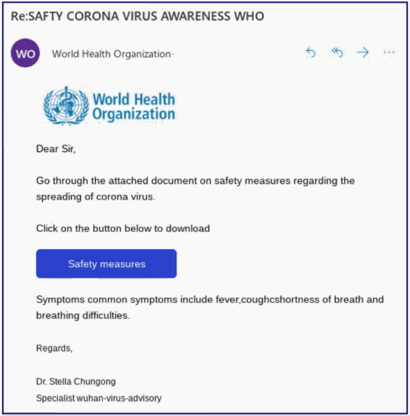
Machine learning based email spam filtering.



Markers for spam change overtime, so model might change.

EXAMPLE

Machine learning based email spam filtering.


Re:SAFTY CORONA VIRUS AWARENESS WHO

WO World Health Organization

World Health Organization

Dear Sir,

Go through the attached document on safety measures regarding the spreading of corona virus.

Click on the button below to download

Safety measures

Symptoms common symptoms include fever,coughcshortness of breath and breathing difficulties.

Regards,

Dr. Stella Chungong
Specialist wuhan-virus-advisory

Markers for spam change overtime, so model might change.

Choose some model $M_{\mathbf{x}}$ parameterized by parameters $\mathbf{x}$ and some loss function ℓ. At time steps $1, \ldots, T$, receive data vectors $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(T)}$.

- At each time step, we pick ("play") a parameter vector $\mathbf{x}^{(i)}$.
- Make prediction $\tilde{y}^{(i)} = M_{\mathbf{x}^{(i)}}(\mathbf{a}_i)$.
- Then told true value or label $y^{(i)}$. Possibly use this information to choose a new $\mathbf{x}^{(i+1)}$.
- Goal is to minimize cumulative loss:

$$L = \sum_{i=1}^{n} \ell(\mathbf{x}^{(i)}, \mathbf{a}^{(i)}, y^{(i)})$$

For example, for a regression problem we might use the ℓ_2 loss:

$$\ell(\mathbf{x}^{(i)}, \mathbf{a}^{(i)}, y^{(i)}) = \left| \langle \mathbf{x}^{(i)}, \mathbf{a}^{(i)} \rangle - y^{(i)} \right|^2 .$$

For classification, we could use logistic/cross-entropy loss.

Abstraction as optimization problem: Instead of a single objective function f, we have a single (initially unknown) function $f_1, \ldots, f_T : \mathbb{R}^d \to \mathbb{R}$ for each time step.

- For time step $i \in 1, \ldots, T$, select vector $\mathbf{x}^{(i)}$.
- Observe f_i and pay cost $f_i(\mathbf{x}^{(i)})$
- Goal is to minimize $\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)})$.

We make <u>no assumptions</u> that $f_1, \ldots, f_T$ are related to each other at all!

In offline optimization, we wanted to find $\hat{\mathbf{x}}$ satisfying $f(\hat{\mathbf{x}}) \leq \min_{\mathbf{x}} f(\mathbf{x})$. Ask for a similar thing here.

Objective: Choose $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)}$ so that:

$$\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)}) \leq \left[\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})\right] + \epsilon.$$

Here ϵ is called the regret of our solution sequence $\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(T)}$.

We typically ϵ to be growing <u>sublinearly</u> in T.

Regret compares to the best fixed solution in hindsight.

$$\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)}) \leq \left[\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})\right] + \epsilon.$$

It's very possible that $\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)}) < \left[\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})\right]$. Could we hope for something stronger?

Exercise: Argue that the following is impossible to achieve:

$$\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)}) \leq \left[\sum_{i=1}^{T} \min_{\mathbf{x}} f_i(\mathbf{x})\right] + \epsilon.$$

Convex functions:

$$f_1(x) = |x - h_1|$$
$$\vdots$$
$$f_n(x) = |x - h_T|$$

where $h_1, \ldots, h_T$ are i.i.d. uniform $\{0, 1\}$.

$$\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)}) \leq \left[\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})\right] + \epsilon.$$

Beautiful balance:

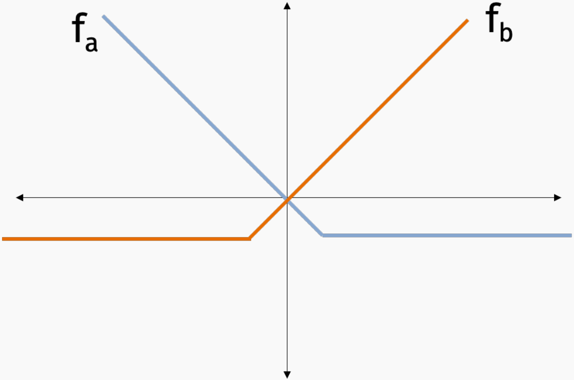
- Either $f_1, \ldots, f_T$ are similar or changing slowly, so we can learn predict f_i from earlier functions.
- Or $f_1, \ldots, f_T$ are very different, in which case $\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})$ is large, so regret bound is easy to achieve.
- Or we live somewhere in the middle.

Follow-the-leader algorithm:

- Choose $\mathbf{x}^{(0)}$.
- For $i = 1, \ldots, T$:
  - Let $\mathbf{x}^{(i)} = \arg\min_{\mathbf{x}} \sum_{j=1}^{i-1} f_j(\mathbf{x})$.
  - Play $\mathbf{x}^{(i)}$.
  - Observe f_i and incur cost $f_i(\mathbf{x}^{(i)})$.

Simple and intuitive, but there are <u>two</u> issues with this approach. One is computational, one is related to the accuracy.

Hard case:



Online Gradient descent:

- Choose $\mathbf{x}^{(1)}$ and $\eta = \frac{R}{G\sqrt{T}}$.
- For $i = 1, \ldots, T$:
    - Play $\mathbf{x}^{(i)}$.
    - Observe f_i and incur cost $f_i(\mathbf{x}^{(i)})$.
    - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f_i(\mathbf{x}^{(i)})$

If $f_1, \ldots, f_T = f$ are all the same, this looks a lot like regular gradient descent. We update parameters using the gradient ∇f at each step.

$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})$ (the offline optimum)

Assume:

- $f_1, \ldots, f_T$ are all convex.
- Each is G-Lipschitz: for all $\mathbf{x}$, i, $\|\nabla f_i(\mathbf{x})\|_2 \leq G$.
- Starting radius: $\|\mathbf{x}^* - \mathbf{x}^{(1)}\|_2 \leq R$.

Online Gradient descent:

- Choose $\mathbf{x}^{(1)}$ and $\eta = \frac{R}{G\sqrt{T}}$.
- For $i = 1, \ldots, T$:
  - Play $\mathbf{x}^{(i)}$.
  - Observe f_i and incur cost $f_i(\mathbf{x}^{(i)})$.
  - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f_i(\mathbf{x}^{(i)})$

Let $\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i=1}^{T} f_i(\mathbf{x})$ (the offline optimum)

Theorem (OGD Regret Bound)

After T steps, $\epsilon = \left[\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)})\right] - \left[\sum_{i=1}^{T} f_i(\mathbf{x}^*)\right] \leq RG\sqrt{T}$.

Average regret overtime is bounded by $\frac{\epsilon}{T} \leq \frac{RG}{\sqrt{T}}$.

Goes $\to 0$ as $T \to \infty$.

All this with no assumptions on how $f_1, \ldots, f_T$ relate to each other! They could have even been chosen adversarially – e.g. with f_i depending on our choice of $\mathbf{x}_i$ and all previous choices.

Theorem (OGD Regret Bound)

After T steps, $\epsilon = \left[\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)})\right] - \left[\sum_{i=1}^{T} f_i(\mathbf{x}^*)\right] \leq RG\sqrt{T}$.

Claim 1: For all $i = 1, \ldots, T$,

$$f_i(\mathbf{x}^{(i)}) - f_i(\mathbf{x}^*) \leq \frac{\|\mathbf{x}^{(i)} - \mathbf{x}^*\|_2^2 - \|\mathbf{x}^{(i+1)} - \mathbf{x}^*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

(Same proof for standard GD. Only uses convexity of f_i.)

Theorem (OGD Regret Bound)

After T steps, $\epsilon = \left[\sum_{i=1}^{T} f_i(\mathbf{x}^{(i)})\right] - \left[\sum_{i=1}^{T} f_i(\mathbf{x}^*)\right] \leq RG\sqrt{T}$.

Claim 1: For all $i = 1, \ldots, T$,

$$f_i(\mathbf{x}^{(i)}) - f_i(\mathbf{x}^*) \leq \frac{\|\mathbf{x}^{(i)} - \mathbf{x}^*\|_2^2 - \|\mathbf{x}^{(i+1)} - \mathbf{x}^*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

Telescoping Sum:

$$\sum_{i=1}^{T}\left[f_i(\mathbf{x}^{(i)}) - f_i(\mathbf{x}^*)\right] \leq \frac{\|\mathbf{x}^{(1)} - \mathbf{x}^*\|_2^2 - \|\mathbf{x}^{(T)} - \mathbf{x}^*\|_2^2}{2\eta} + \frac{T\eta G^2}{2}$$

$$\leq \frac{R^2}{2\eta} + \frac{T\eta G^2}{2}$$

Efficient offline optimization method for functions f with finite sum structure:

$$f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x}).$$

Goal is to find $\hat{\mathbf{x}}$ such that $f(\hat{\mathbf{x}}) \leq f(\mathbf{x}^*) + \epsilon$.

- The most widely use optimization algorithm in modern machine learning.
- Easily analyzed as a special case of online gradient descent!

Recall the machine learning setup. In empirical risk minimization, we can typically write:

$$f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$$

where f_i is the loss function for a particular data example $(\mathbf{a}^{(i)}, y^{(i)})$.

Example: least squares linear regression.

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\mathbf{x}^T \mathbf{a}^{(i)} - y^{(i)})^2$$

Note that by linearity, $\nabla f(\mathbf{x}) = \sum_{i=1}^{n} \nabla f_i(\mathbf{x})$.

Main idea: Use random approximate gradient in place of actual gradient.

Pick random $j \in 1, \ldots, n$ and update $\mathbf{x}$ using $\nabla f_j(\mathbf{x})$.

$$\mathbb{E}\left[\nabla f_j(\mathbf{x})\right] = \frac{1}{n}\nabla f(\mathbf{x}).$$

$n\nabla f_j(\mathbf{x})$ is an unbiased estimate for the true gradient $\nabla f(\mathbf{x})$, but can often be computed in a $1/n$ fraction of the time!

Trade slower convergence for cheaper iterations.

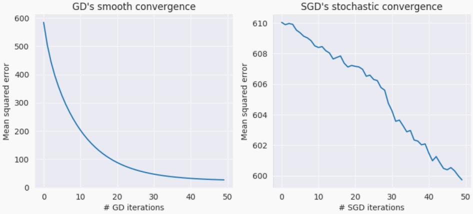
Stochastic first-order oracle for $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$.

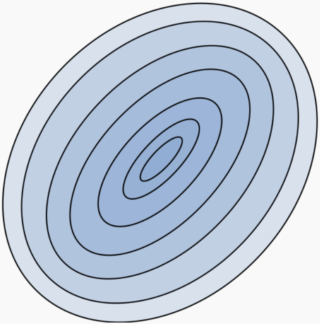
- **Function Query:** For any chosen $j, \mathbf{x}$, return $f_j(\mathbf{x})$
- **Gradient Query:** For any chosen $j, \mathbf{x}$, return $\nabla f_j(\mathbf{x})$

Stochastic Gradient descent:

- Choose starting vector $\mathbf{x}^{(1)}$, step size η
- For $i = 1, \ldots, T$:
  - Pick random $j_i \in 1, \ldots, n$.
  - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f_{j_i}(\mathbf{x}^{(i)})$
- Return $\hat{\mathbf{x}} = \frac{1}{T} \sum_{i=1}^{T} \mathbf{x}^{(i)}$

VISUALIZING SGD





Assume:

- Finite sum structure: $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$, with $f_1, \ldots, f_n$ all convex.
- Lipschitz functions: for all $\mathbf{x}$, j, $\|\nabla f_j(\mathbf{x})\|_2 \leq \frac{G'}{n}$.
  - What does this imply about Lipschitz constant of f?
- Starting radius: $\|\mathbf{x}^* - \mathbf{x}^{(1)}\|_2 \leq R$.

Stochastic Gradient descent:

- Choose $\mathbf{x}^{(1)}$, steps T, step size $\eta = \frac{R}{G'\sqrt{T}}$.
- For $i = 1, \ldots, T$:
  - Pick random $j_i \in 1, \ldots, n$.
  - $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \eta \nabla f_{j_i}(\mathbf{x}^{(i)})$
- Return $\hat{\mathbf{x}} = \frac{1}{T} \sum_{i=1}^{T} \mathbf{x}^{(i)}$

Approach: View as online gradient descent run on function sequence $f_{j_1}, \ldots, f_{j_T}$.

Only use the fact that step equals gradient in expectation.

For a convex function f and points $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(t)}$

$$f\left(\frac{1}{t} \cdot \mathbf{x}^{(1)} + \ldots + \frac{1}{t} \cdot \mathbf{x}^{(t)}\right) \leq \frac{1}{t} \cdot f(\mathbf{x}^{(1)}) + \ldots + \frac{1}{t} \cdot f(\mathbf{x}^{(t)})$$

Claim (SGD Convergence)

After $T = \frac{R^2 G'^2}{\epsilon^2}$ iterations:

$$\mathbb{E}\left[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)\right] \leq \epsilon.$$

Claim 1:

$$f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \frac{1}{T}\sum_{i=1}^{T}\left[f(\mathbf{x}^{(i)}) - f(\mathbf{x}^*)\right]$$

Prove using Jensen's Inequality:

Claim (SGD Convergence)

After $T = \frac{R^2 G'^2}{\epsilon^2}$ iterations:

$$\mathbb{E}\left[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)\right] \leq \epsilon.$$

$$\begin{aligned}\mathbb{E}[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)] &\leq \frac{1}{T}\sum_{i=1}^{T} \mathbb{E}\left[f(\mathbf{x}^{(i)}) - f(\mathbf{x}^*)\right] \\ &= \frac{1}{T}\sum_{i=1}^{T} n\mathbb{E}\left[f_{j_i}(\mathbf{x}^{(i)}) - f_{j_i}(\mathbf{x}^*)\right]\end{aligned}$$

Claim (SGD Convergence)

After $T = \frac{R^2 G'^2}{\epsilon^2}$ iterations:

$$\mathbb{E}\left[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)\right] \leq \epsilon.$$

$$\begin{aligned}
\mathbb{E}[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)] &\leq \frac{1}{T}\sum_{i=1}^{T} \mathbb{E}\left[f(\mathbf{x}^{(i)}) - f(\mathbf{x}^*)\right] \\
&= \frac{1}{T}\sum_{i=1}^{T} n\mathbb{E}\left[f_{j_i}(\mathbf{x}^{(i)}) - f_{j_i}(\mathbf{x}^*)\right] \\
&\leq \frac{n}{T} \cdot \mathbb{E}\left[\sum_{i=1}^{T} f_{j_i}(\mathbf{x}^{(i)}) - f_{j_i}(\mathbf{x}^{offline})\right],
\end{aligned}$$

where $\mathbf{x}^{offline} = \arg\min_{\mathbf{x}} \sum_{i=1}^{T} f_{j_i}(\mathbf{x})$.

Claim (SGD Convergence)

After $T = \frac{R^2 G'^2}{\epsilon^2}$ iterations:

$$\mathbb{E}\left[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)\right] \leq \epsilon.$$

$$\begin{aligned}
\mathbb{E}[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)] &\leq \frac{1}{T}\sum_{i=1}^{T} \mathbb{E}\left[f(\mathbf{x}^{(i)}) - f(\mathbf{x}^*)\right] \\
&= \frac{1}{T}\sum_{i=1}^{T} n\mathbb{E}\left[f_{j_i}(\mathbf{x}^{(i)}) - f_{j_i}(\mathbf{x}^*)\right] \\
&\leq= \frac{n}{T}\mathbb{E}\left[\sum_{i=1}^{T} f_{j_i}(\mathbf{x}^{(i)}) - f_{j_i}(\mathbf{x}^{offline})\right] \\
&\leq \frac{n}{T}\cdot\left(R\cdot\frac{G'}{n}\cdot\sqrt{T}\right) \qquad \text{(by OGD guarantee.)}
\end{aligned}$$

Number of iterations for error ϵ:

- **Gradient Descent**: $T = \frac{R^2G^2}{\epsilon^2}$.
- **Stochastic Gradient Descent**: $T = \frac{R^2G'^2}{\epsilon^2}$.

Always have $G \leq G'$:

$$\begin{aligned}
\max_{\mathbf{x}} \|\nabla f(\mathbf{x})\|_2 &\leq \max_{\mathbf{x}} \left(\|\nabla f_1(\mathbf{x})\|_2 + \ldots + \|\nabla f_n(\mathbf{x})\|_2\right) \\
&\leq \max_{\mathbf{x}} \left(\|\nabla f_1(\mathbf{x})\|_2\right) + \ldots + \max_{\mathbf{x}} \left(\|\nabla f_n(\mathbf{x})\|_2\right) \\
&\leq n \cdot \frac{G'}{n} = G'.
\end{aligned}$$

So GD converges strictly faster than *SGD*.

But for a fair comparison:

- SGD cost = (# of iterations) $\cdot\, O(1)$
- GD cost = (# of iterations) $\cdot\, O(n)$

We always have $G \leq G'$. When it is much smaller then GD will perform better. When it is closer to this upper bound, SGD will perform better.

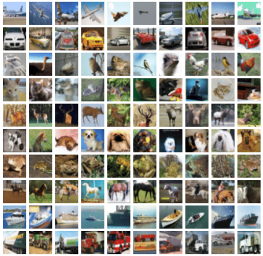
What is an extreme case where $G = G'$?

What if each gradient $\nabla f_i(\mathbf{x})$ looks like random vectors in $\mathbb{R}^d$? E.g. with $\mathcal{N}(0,1)$ entries?

$$\mathbb{E}\left[\|\nabla f_i(\mathbf{x})\|_2^2\right] =$$

$$\mathbb{E}\left[\|\nabla f(\mathbf{x})\|_2^2\right] = \mathbb{E}\left[\|\sum_{i=1}^{n} \nabla f_i(\mathbf{x})\|_2^2\right] =$$

Takeaway: SGD performs better when there is more structure or repetition in the data set.



PRECONDITIONING

Main idea: Instead of minimizing $f(\mathbf{x})$, find another function $g(\mathbf{x})$ with the same minimum but which is better suited for first order optimization (e.g., has a smaller conditioner number).

Claim: Let $h(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ be an invertible function. Let $g(\mathbf{x}) = f(h(\mathbf{x}))$. Then

$$\min_{\mathbf{x}} f(\mathbf{x}) = \min_{\mathbf{x}} g(\mathbf{x}) \quad \text{and} \quad \arg\min_{\mathbf{x}} f(\mathbf{x}) = h\left(\arg\min_{\mathbf{x}} g(\mathbf{x})\right).$$

First Goal: We need $g(\mathbf{x})$ to still be convex.

Claim: Let $\mathbf{P}$ be an invertible $d \times d$ matrix and let $g(\mathbf{x}) = f(\mathbf{P}\mathbf{x})$.

$g(\mathbf{x})$ is always convex.

Second Goal:

$g(\mathbf{x})$ should have better condition number κ than $f(\mathbf{x})$.

Example:

- $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$. $\kappa_f = \frac{\lambda_1(\mathbf{A}^T\mathbf{A})}{\lambda_d(\mathbf{A}^T\mathbf{A})}$.
- $g(\mathbf{x}) = \|\mathbf{A}\mathbf{P}\mathbf{x} - \mathbf{b}\|_2^2$. $\kappa_g = \frac{\lambda_1(\mathbf{P}^T\mathbf{A}^T\mathbf{A}\mathbf{P})}{\lambda_d(\mathbf{P}^T\mathbf{A}^T\mathbf{A}\mathbf{P})}$.

Third Goal: $\mathbf{P}$ should be easy to compute.

Many, many problem specific preconditioners are used in practice. There design is usually a heuristic process.

Example: Diagonal preconditioner.

- Let $\mathbf{D} = \text{diag}(\mathbf{A}^T\mathbf{A})$
- Intuitively, we roughly have that $\mathbf{D} \approx \mathbf{A}^T\mathbf{A}$.
- Let $\mathbf{P} = \sqrt{\mathbf{D}^{-1}}$

$\mathbf{P}$ is often called a **Jacobi preconditioner**. Often works very well in practice!

DIAGONAL PRECONDITIONER

```
A =

    -734       1      33     9111       0
     -31      -2     108     5946     -19
     232      -1     101     3502      10
     426       0     -65    12503       9
    -373       0      26     9298       0
    -236      -2     -94     2398      -1
    2024       0    -132    -6904     -25
   -2258      -1      92    -6516       6
    2229       0       0    11921     -22
     338       1      -5   -16118     -23
```

```
>> cond(A'*A)

ans =

   8.4145e+07
```

```
>> P = sqrt(inv(diag(diag(A'*A))));
>> cond(P*A'*A*P)

ans =

   10.3878
```

ADAPTIVE STEPSIZES

Another view: If $g(\mathbf{x}) = f(\mathbf{P}\mathbf{x})$ then $\nabla g(\mathbf{x}) = \mathbf{P}^T \nabla f(\mathbf{P}\mathbf{x})$.

$\nabla g(\mathbf{x}) = \mathbf{P} \nabla f(\mathbf{P}\mathbf{x})$ when $\mathbf{P}$ is symmetric.

Gradient descent on g:

- For $t = 1, \ldots, T$,
  - $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \eta \mathbf{P} \left[\nabla f(\mathbf{P}\mathbf{x}^{(t)}) \right]$

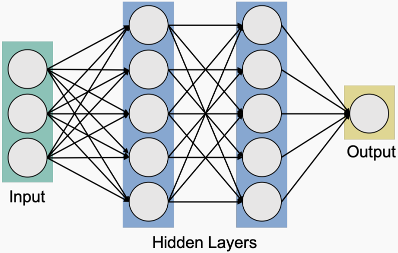
Gradient descent on g:

- For $t = 1, \ldots, T$,
  - $\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \eta \mathbf{P}^2 \left[\nabla f(\mathbf{y}^{(t)}) \right]$

When $\mathbf{P}$ is diagonal, this is just gradient descent with a different step size for each parameter!

Algorithms based on this idea:

- AdaGrad
- RMSprop
- Adam optimizer



(Pretty much all of the most widely used optimization methods for training neural networks.)

COORDINATE DESCENT

Main idea: Trade slower convergence (more iterations) for cheaper iterations.

Stochastic Gradient Descent: When $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$, approximate $\nabla f(\mathbf{x})$ with $\nabla f_i(\mathbf{x})$ for randomly chosen i.

Main idea: Trade slower convergence (more iterations) for cheaper iterations.

Stochastic Coordinate Descent: Only compute a single random entry of $\nabla f(\mathbf{x})$ on each iteration:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\mathbf{x}) \\ \frac{\partial f}{\partial x_2}(\mathbf{x}) \\ \vdots \\ \frac{\partial f}{\partial x_d}(\mathbf{x}) \end{bmatrix} \qquad \nabla_i f(\mathbf{x}) = \begin{bmatrix} 0 \\ \frac{\partial f}{\partial x_i}(\mathbf{x}) \\ \vdots \\ 0 \end{bmatrix}$$

Update: $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} + \eta \nabla_i f(\mathbf{x}^{(t)})$.