CS 760: Machine Learning
Optimization

University of Wisconsin-Madison

Outline

- **Optimization problems in machine learning**

- **Gradient descent**
  - idea, algorithm, convexity, convergence, nonconvexity

- **Drawbacks and other optimizers**
  - stochastic gradient descent, alternative optimizers

Outline

- **Optimization problems in machine learning**

- **Gradient descent**
  - idea, algorithm, convexity, convergence, nonconvexity

- **Drawbacks and other optimizers**
  - stochastic gradient descent, alternative optimizers

Optimization in ML

- in supervised learning, we
- have a training dataset of $\left(x^{(i)}, y^{(i)}\right)$ pairs for $i = 1, \dots, n$
- search a hypothesis space H for a function h that
  - predicts well, i.e. $h\left(x^{(i)}\right) = y^{(i)}$ on most of the training data
  - satisfies other constraints, e.g. simplicity so as not to overfit

Optimization in ML

- often **searching the hypothesis space** is an **optimization problem**:

- decision trees

$$\min_{\substack{h(x^{(i)})=y^{(i)} \\ \forall i\in[n]}} \quad depth(h)$$

predicts well **is simple**

- parametric models

$$\min_{\theta\in\mathbb{R}^d} \sum_{i=1}^{n} loss\left(h_\theta\left(x^{(i)}\right), y^{(i)}\right) + penalty(\theta)$$

Optimization in ML

- often **searching the hypothesis** space is an **optimization problem**:

- decision trees

$$\min_{\substack{h(x^{(i)})=y^{(i)} \\ \forall i\in[n]}} depth(h)$$

- parametric models

$$\min_{\theta\in\mathbb{R}^d} \sum_{i=1}^{n} \underbrace{loss}_{\text{squared error}}\left(h_\theta\left(x^{(i)}\right), y^{(i)}\right) + \underbrace{penalty}_{\text{squared } \ell_2\text{-norm}}(\theta)$$

h_θ is a neural net

linear / Ridge regression
$\theta = (X^T X + \lambda I)^{-1} X^T y$

sometimes optimization is (NP) hard and we must use a heuristic

sometimes we have a closed-form solution

This lecture: sometimes there is an **efficient optimization algorithm** to get the solution

Outline

- **Optimization problems in machine learning**

- **Gradient descent**
  - idea, algorithm, convexity, convergence, nonconvexity

- **Drawbacks and other optimizers**
  - stochastic gradient descent, alternative optimizers

Iterative Methods: Gradient Descent

- What if there's no closed-form solution?
- Use an iterative approach to gradually get closer to the solution.

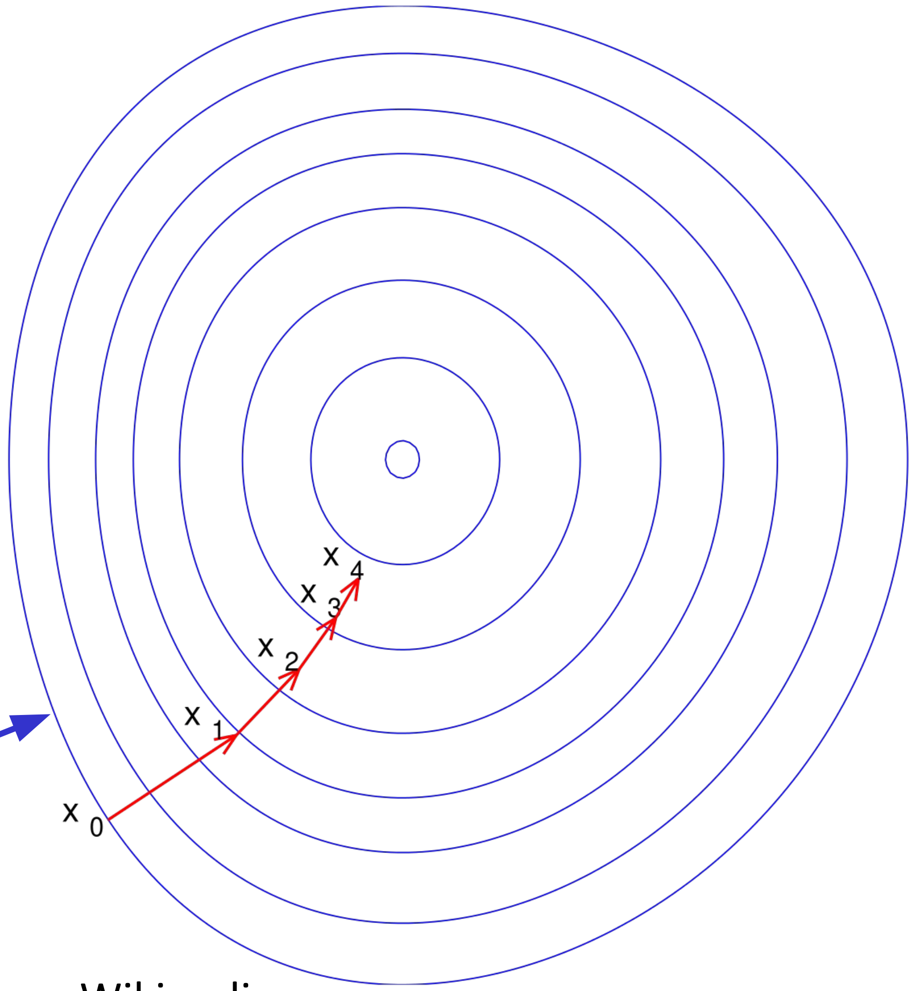
- Gradient descent:
  - Suppose we're computing $\min_{\theta} g(\theta)$
  - Start at some θ_0
  - Iteratively compute $\theta_{t+1} = \theta_t - \underset{\text{learning rate/step size}}{\alpha} \nabla g(\theta_t)$
  - Stop after some # of steps

Gradient Descent: Illustration

- **Goal**: steps get closer to minimizer

- Some notes:
  - Step size can be fixed or varying
  - Under certain conditions, will converge to global minimum

level sets



Wikipedia

Gradient Descent: Efficiency

- Back to our linear regression problem

- Want to find $\min_{\theta} \ell(f_\theta) = \min_{\theta} \frac{1}{n} \|X\theta - y\|_2^2$
  - What's our gradient? $\nabla \ell(f_\theta) = \frac{1}{n}(2X^T X\theta - 2X^T y)$
  - So, plugging in , we get

$$\theta_{t+1} = \theta_t - \alpha \frac{1}{n}(2X^T X\theta_t - 2X^T y)$$

Linear Regression: Normal Equations vs GD

- Why do this for a setting with a closed-form solution?
- **Consider its computational cost:**

$$\theta = (\underbrace{X^T X}_{d \times d})^{-1} \underset{\overset{\uparrow}{d \times n}}{X^T} \underset{\overset{\uparrow}{n \times 1}}{y}$$

- Cost: (i) invert matrix, $\Theta(d^3)$. (ii) multiplication, $\Theta(d^2 n)$.
- **Total**: $\Theta(d^2 n + d^3)$.

Recall: by standard methods, inverting an *m* x *m* matrix is $\Theta(m^3)$.

Multiplying a *m* x *p* with a *p* x *q* matrix is $\Theta(mpq)$

Linear Regression: Normal Equations vs GD

Now let's compare to the cost of gradient descent

- Normal Equations $\theta = (X^T X)^{-1} X^T y$
  - **Total Cost**: $\Theta(d^2 n + d^3)$.

- Gradient Descent: t iterations

$$\theta_{t+1} = \theta_t - \alpha \frac{1}{n} (2X^T X \theta_t - 2X^T y)$$

  - Cost: $\Theta(dn)$ at each step.
  - **Total Cost**: $\Theta(dnt)$.

If we do "few" steps t, then **GD is cheaper**: $t < \max\{d, d^2/n\}$

Gradient Descent: Convergence

- Even if GD is cheaper, what does it give us?
- Let's analyze it. We'll need some **assumptions**
  - convex and differentiable objective f
  - f has L-Lipschitz-continuous gradients

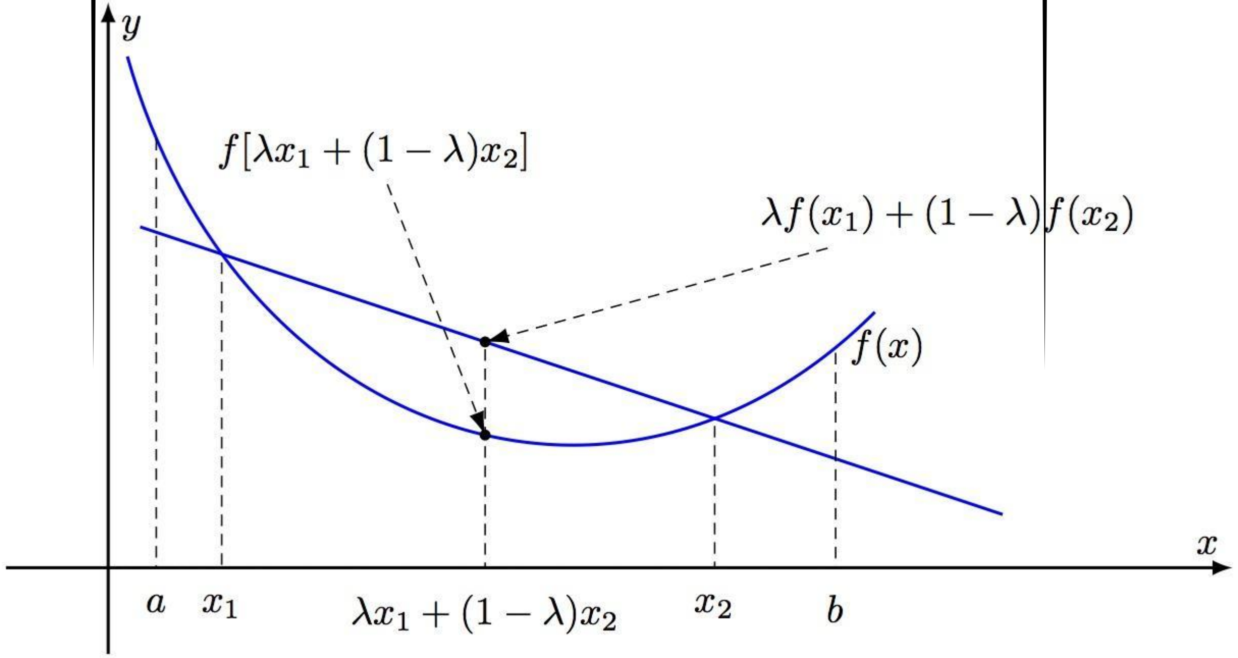
- Under these assumptions, we have the following guarantee:
  - if we run T steps of GD with fixed step size $\alpha \leq 1/L$ starting at x_0, then the Tth iterate x_T satisfies

$$f(x_T) - f(\underset{\uparrow\atop\text{minimizer}}{x^*}) \leq \frac{||x_0 - x^*||_2^2}{2T\alpha}$$

Gradient Descent Analysis : Convexity

A function f with convex domain X is called convex if for all x_1, x_2 in the domain and all $\lambda \in [0, 1]$ we have

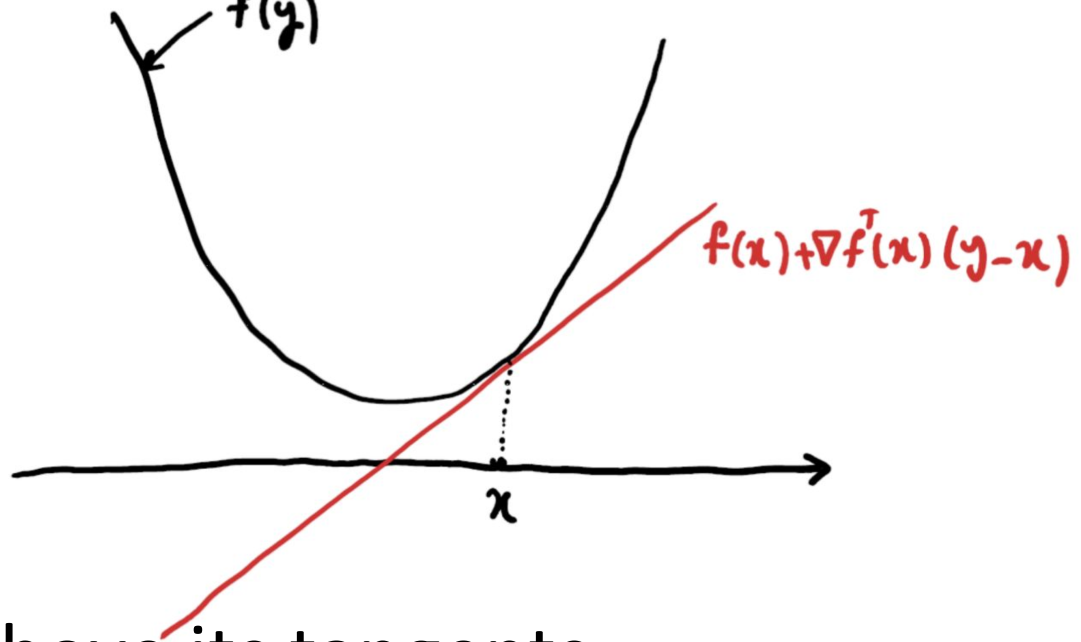
$$f(\underbrace{\lambda x_1 + (1-\lambda)x_2}_{\text{Convex combination}}) \le \underbrace{\lambda f(x_1) + (1-\lambda) f(x_2)}_{\text{Line segment joining } f(x_1) \text{ and } f(x_2)}$$


y
$f[\lambda x_1 + (1-\lambda)x_2]$
$\lambda f(x_1) + (1-\lambda) f(x_2)$
$f(x)$
x
a
x_1
$\lambda x_1 + (1-\lambda)x_2$
x_2
b

Gradient Descent Analysis : Convexity

- An equivalent definition:

$$f(x_2) \geq f(x_1) + \nabla f(x_1)^T (x_2 - x_1)$$



- Function sits above its tangents

Gradient Descent Analysis : Lipschitzness

- A function has L-Lipschitz gradients if

$$\|\nabla f(x_1) - \nabla f(x_2)\|_2 \leq L\|x_1 - x_2\|_2$$

- Equivalent to $\nabla^2 f(x) \preceq LI$

- Recall: $A \preceq B$ means that $B - A$ is positive semidefinite

- Recall some more: C is positive semidefinite if $x^T C x \geq 0$

Gradient Descent: Convergence Proof p. 1

- We'll use our **two assumptions**.
- Let's start with a Taylor expansion:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + 1/2 (y - x)^T \nabla^2 f(z) (y - x)$$

- Next, our gradient Lipschitz condition means $\nabla^2 f(x) \preceq LI$

$$\boxed{\implies f(y) \leq \underbrace{f(x) + \nabla f(x)^T (y - x)}_{\text{Linear Approximation}} + \underbrace{1/2 L \|y - x\|^2}_{\text{Remainder: at most a quadratic}}}$$

Gradient Descent: Convergence Proof p. 2

- Let’s plug in our GD relationship $y \leftarrow x_{t+1} = x_t - \alpha \nabla f(x_t)$

$$\boxed{\implies f(y) \leq f(x) + \nabla f(x)^T (y - x) + 1/2L \|y - x\|^2}$$

- Start with some algebra

$$f(x_{t+1}) \leq f(x_t) + \nabla f(x_t)^T (x_{t+1} - x_t) + 1/2L \|x_{t+1} - x_t\|_2^2$$

$$= f(x_t) - \nabla f(x_t)^T \alpha \nabla f(x_t) + 1/2L \|\alpha \nabla f(x_t)\|_2^2$$

$$= f(x_t) - \alpha \|\nabla f(x_t)\|_2^2 + 1/2L\alpha^2 \|\nabla f(x_t)\|_2^2$$

$$\boxed{= f(x_t) - \alpha(1 - 1/2L\alpha) \|\nabla f(x_t)\|_2^2}$$

Gradient Descent: Convergence Proof p. 3

- So, we now have

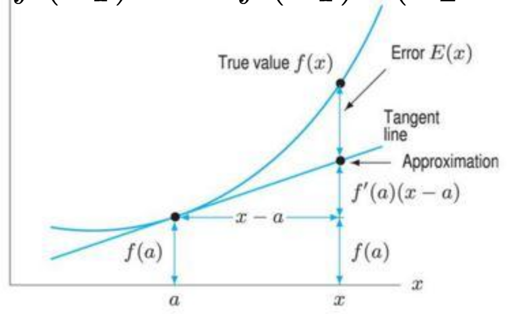
$$f(x_{t+1}) \leq f(x_t) - \underbrace{1/2\alpha\|\nabla f(x_t)\|_2^2}_{\text{Positive except at minimum (where it's 0)}}$$

- Promising! Our estimates are getting better.

- Still need how big these gradient magnitudes are

Gradient Descent: Convergence Proof p. 4

$$f(x_2) \geq f(x_1) + \nabla f(x_1)^T (x_2 - x_1)$$



- Haven't used convexity yet, so let's:

$$f(x_t) \leq f(x^*) + \nabla f(x_t)^T (x_t - x^*)$$

- Combine with $\boxed{f(x_{t+1}) \leq f(x_t) - 1/2\alpha \|\nabla f(x_t)\|_2^2}$

$$f(x_{t+1}) \leq f(x^*) + \nabla f(x_t)^T (x_t - x^*) - \alpha/2 \|\nabla f(x_t)\|_2^2$$

$$f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(2\alpha \nabla f(x_t)^T (x_t - x^*) - \alpha^2 \|\nabla f(x_t)\|_2^2)$$

$$\|b-a\|_2^2 = \|a\|_2^2 - 2a^\top b + \|b\|_2^2 \;\Rightarrow\; 2a^\top b - \|a\|_2^2 = \|b\|_2^2 - \|b-a\|_2^2$$

$$\boxed{f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(\|x_t - x^*\|_2^2 - \|x_t - \alpha \nabla f(x_t) - x^*\|_2^2)}$$

Gradient Descent: Convergence Proof p. 5

- Now, simplify

$$f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}\left(\|x_t - x^*\|_2^2 - \|\underbrace{x_t - \alpha\nabla f(x_t)}_{\text{This part is just } x_{t+1}} - x^*\|_2^2\right)$$

$$f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}\left(\|x_t - x^*\|_2^2 - \|x_{t+1} - x^*\|_2^2\right)$$

Gradient Descent: Convergence Proof p. 6

- So, we have something familiar…

$$\boxed{f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}\underbrace{(\|x_t - x^*\|_2^2 - \|x_{t+1} - x^*\|_2^2)}_{\text{Can telescope!}}}$$

$$\sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*) \leq \sum_{t=0}^{T-1} \frac{1}{2\alpha}(\|x_t - x^*\|_2^2 - \|x_{t+1} - x^*\|_2^2)$$

$$\boxed{\sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(\|x_0 - x^*\|_2^2 - \|x_T - x^*\|_2^2)}$$

Gradient Descent: Convergence Proof p. 7

- Now we have

$$\boxed{\sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(\|x_0 - x^*\|_2^2 - \|x_T - x^*\|_2^2)}$$

- Can ignore the rightmost term (we're just making the RHS same or bigger)

$$\boxed{\sum_{t=0}^{T-1} \underbrace{f(x_{t+1}) - f(x^*)}_{\text{Value gap for all steps}} \leq \frac{1}{2\alpha}(\underbrace{\|x_0 - x^*\|_2^2}_{\text{Initial guess gap to minimizer}})}$$

Gradient Descent: Convergence Proof p. 7

- Continue,

$$\sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(\|x_0 - x^*\|_2^2)$$

- But, recall that each iterate has a smaller value, i.e.,

$$f(x_{t+1}) \leq f(x_t) - 1/2\alpha\|\nabla f(x_t)\|_2^2$$

- So

$$\sum_{t=0}^{T-1} f(x_T) \leq \sum_{t=0}^{T-1} f(x_{t+1})$$

Gradient Descent: Convergence Proof p. 8

- Almost there! We have $$\sum_{t=0}^{T-1} f(x_T) \leq \sum_{t=0}^{T-1} f(x_{t+1})$$
- Divide by T,

$$\implies f(x_T) - f(x^*) \leq \frac{1}{T} \sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*)$$

- Combine with $$\boxed{\sum_{t=0}^{T-1} f(x_{t+1}) - f(x^*) \leq \frac{1}{2\alpha}(\|x_0 - x^*\|_2^2)}$$

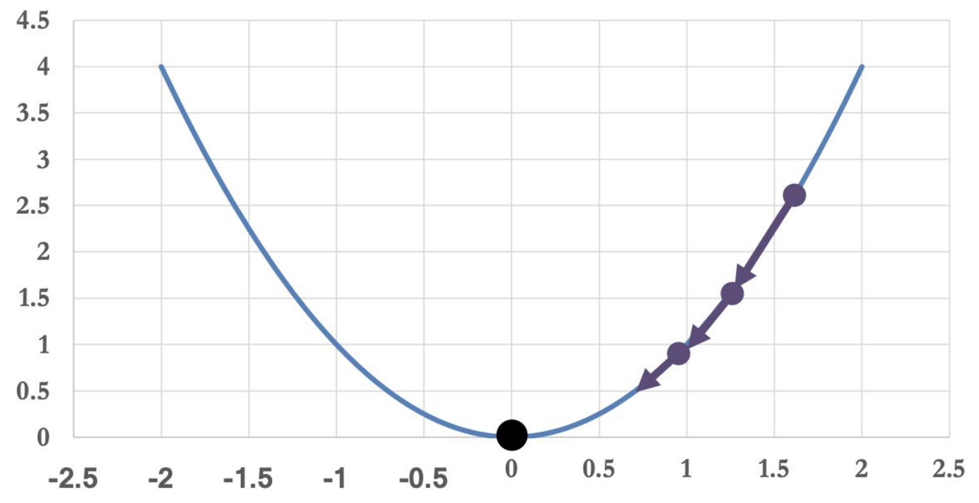
$$\boxed{\implies f(x_T) - f(x^*) \leq \frac{\|x_0 - x^*\|_2^2}{2T\alpha}}$$

Done!

Gradient Descent: Convergence Proof Recap

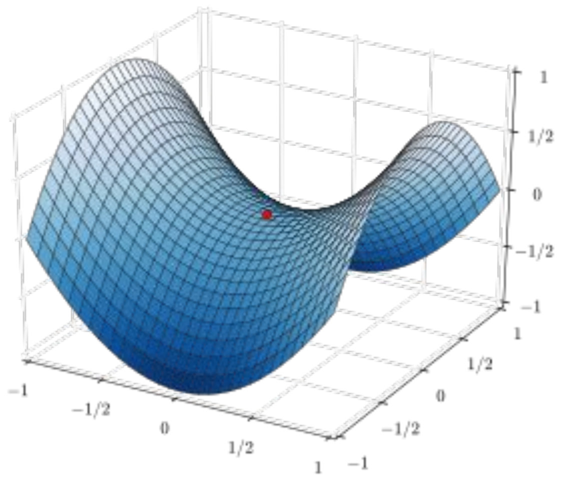
- **Note**: used all conditions in one or more places in the proof.
  - If you don't use an assumption, either your result is stronger than you thought or (more likely) you are making a mistake

- Other assumptions that lead to varying proofs/rates:
  - **Strong convexity**
  - **Non-convexity**
  - **Non-differentiability**



Gradient Descent as a heuristic

- If a function is non-convex, gradient descent
  - can still be applied so long as it is differentiable
  - is only guaranteed to reach a **stationary point**, not necessarily a global minimum



Wikipedia

- Nevertheless, neural networks are commonly fit using (extensions of) gradient descent:
  - objectives are non-convex AND non-smooth
  - often get parameters that **are optimal** (low training loss) AND **generalize well** (high test accuracy)
  - required decades hacking and experimentation
  - should be viewed as a poorly understood heuristic like information gain for decision trees

Break & quiz

Q: True or False: gradient descent is always a more efficient way of computing linear regression than the normal equations.

Q: True or False: gradient descent is always a more efficient way of computing linear regression than the normal equations.

False: depending on the number of data points, feature dimension, and number of steps taken, gradient descent can be more expensive.

Outline

- **Optimization problems in machine learning**
- **Gradient descent**
  - idea, algorithm, convexity, convergence, nonconvexity
- **Drawbacks and other optimizers**
  - stochastic gradient descent, alternative optimizers

Gradient Descent: Drawbacks

- Why would we use anything but GD?
- Let's go back to ERM. $$\arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x^{(i)}, y^{(i)})$$
- For GD, need to compute $\nabla \ell(h(x^{(i)}, y^{(i)})$
    - Each step: n gradient computations
    - ImageNet: 10^6 samples… so for 100 iterations, **10^8 gradients**

Solution: **Stochastic** Gradient Descent

- Simple modification to GD.
- Let’s use some notation: ERM:

$$\arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f(\theta; x^{(i)}), y^{(i)})$$

Note: this is what we’re optimizing over!
x’s are fixed samples.

- GD:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{n} \sum_{i=1}^{n} \nabla \ell(f(\theta_t; x^{(i)}), y^{(i)})$$

Solution: **Stochastic** Gradient Descent

- Simple modification to GD:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{n}\sum_{i=1}^{n} \nabla \ell(f(\theta_t; x^{(i)}), y^{(i)})$$

- SGD:

$$\theta_{t+1} = \theta_t - \alpha \nabla \ell(f(\theta_t; x^{(a)}), y^{(a)})$$

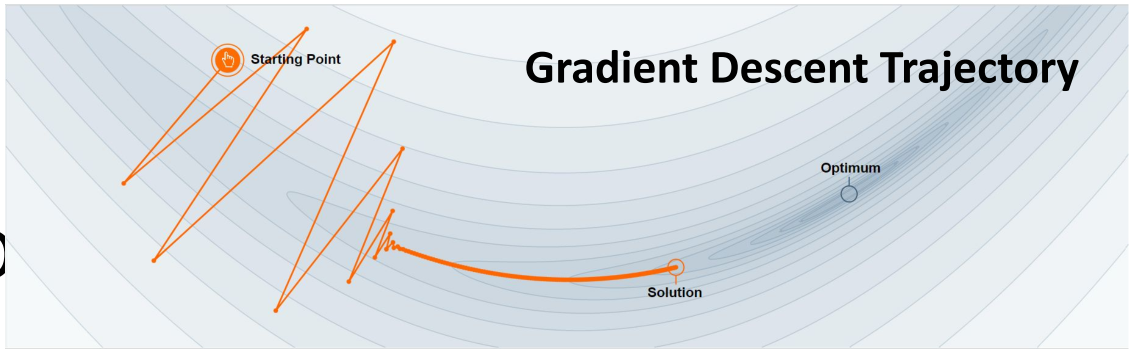
  - Here a is selected uniformly from 1,…,n ("**stochastic**" bit)
  - Note: **no sum**!
  - In expectation, same gradient as GD.
  - In practice we often update using **minibatches** of data to take advantage of (GPU) parallelism

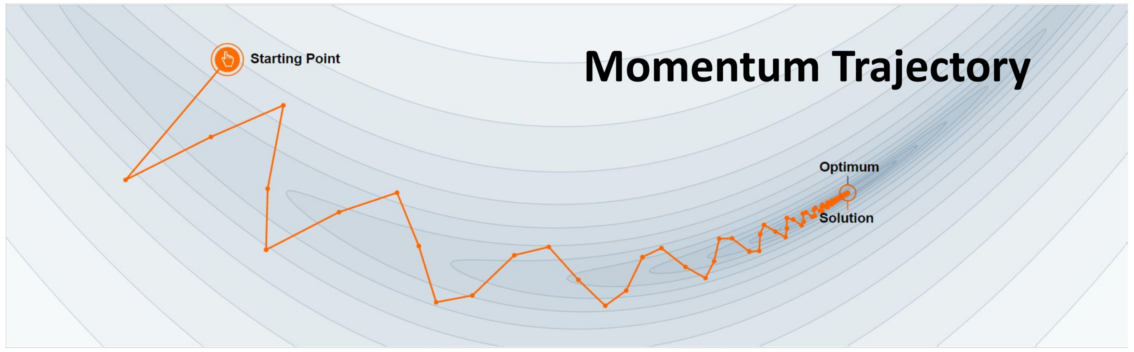
Other drawbacks of gradient descent

Behaves poorly on many important functions:

- e.g. LASSO is convex but non-differentiable, so we use coordinate descent or proximal methods

- on poorly conditioned problems, GD struggles to make progress down narrow valleys. Alternatives:
  - momentum methods
  - second-order methods (Newton)
  - approximate second-order methods (includes widely used deep net optimizers such as Adam)





Source: Goh. *Why Momentum Really Works*. Distill 2017.

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Misha Khodak, Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Ryan Tibshirani