CPSC 340: Machine Learning and Data Mining

Stochastic Gradient

Fall 2015

Admin

- Assignment 3 due now.
  - Solutions posted Monday after class.
- Practice midterm will be posted after class.
  - Monday tutorials will go through it.
- Midterm next Friday, October 30.
  - In class, 55 minutes, closed-book, cheat sheet: 2-pages each double-sided.

Last time: Kernel Trick

- Given test data $\hat{X}$, predict $\hat{y}$ using:

$$\hat{y} = \hat{X}w$$
$$= \hat{X}X^T(XX^T + \lambda I)^{-1}y$$
$$= \hat{K}(K + \lambda I)^{-1}y$$

where $K = XX^T$ and $\hat{K} = \hat{X}X^T$

- Key observation behind kernel trick:
  - If we have K and $\widehat{K}$, we don't need the features.
  - We can train regression models based on similarities rather than features.

Today: Problems with a Huge Number of Examples

- With L2-regularized least squares, can compute 'w' in $O(nd^2 + d^3)$.
- What if 'd' is huge?
  - With kernel trick, cost is $O(n^2d + n^3)$.
  - With gradient descent, cost is $O(nd)$ per iteration.
  - Gradient descent applies to any differentiable loss and regularizer.
- What if 'n' is huge?
  - For example, every e-mail in g-mail.
  - If 'n' is too large, even $O(nd)$ becomes too expensive.

Minimizing Sums with Gradient Descent

- Consider minimizing average of differentiable functions:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} f(w) \quad \text{where} \quad f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$

- Includes all our differentiable losses as special cases.
- Gradient descent for this problem: $w^{t+1} = w^t - \alpha_t \nabla f(w^t)$

$$= w^t - \alpha_t \left(\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w^t)\right)$$

- Nice properties, but iterations require gradients of all 'n' examples.
- Key idea behind stochastic gradient methods:
  - On average, we can decrease 'f' using the gradient of a random example.

Stochastic Gradient Method

- Stochastic gradient method:
  1. Pick a random example i_t.
  2. Perform a gradient descent step based only on this example.

$$w^{t+1} = w^t - \alpha_t \nabla f_{i_t}(w^t)$$

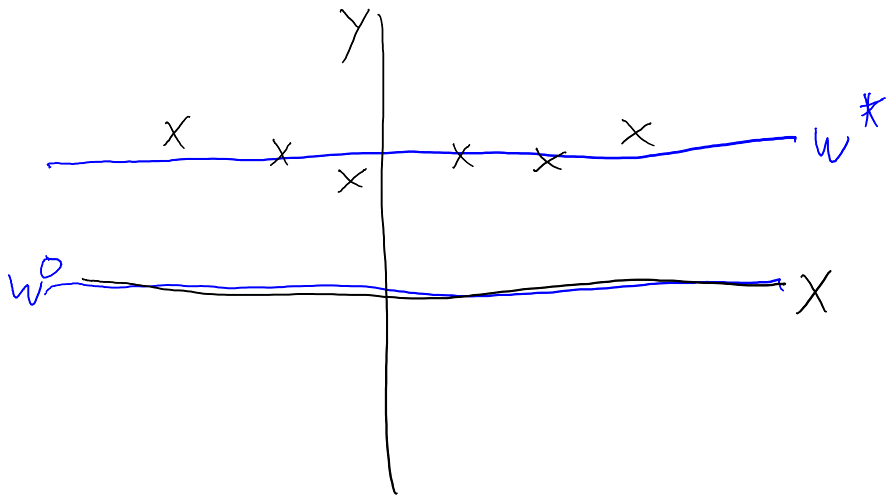
- Intuition: unbiased estimate of full gradient:

$$E_{i_t}[\nabla f_i(w^t)] = \sum_{i=1}^{n} \left(\frac{1}{n}\right) \nabla f_i(w^t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w^t) = \nabla f(w^t)$$

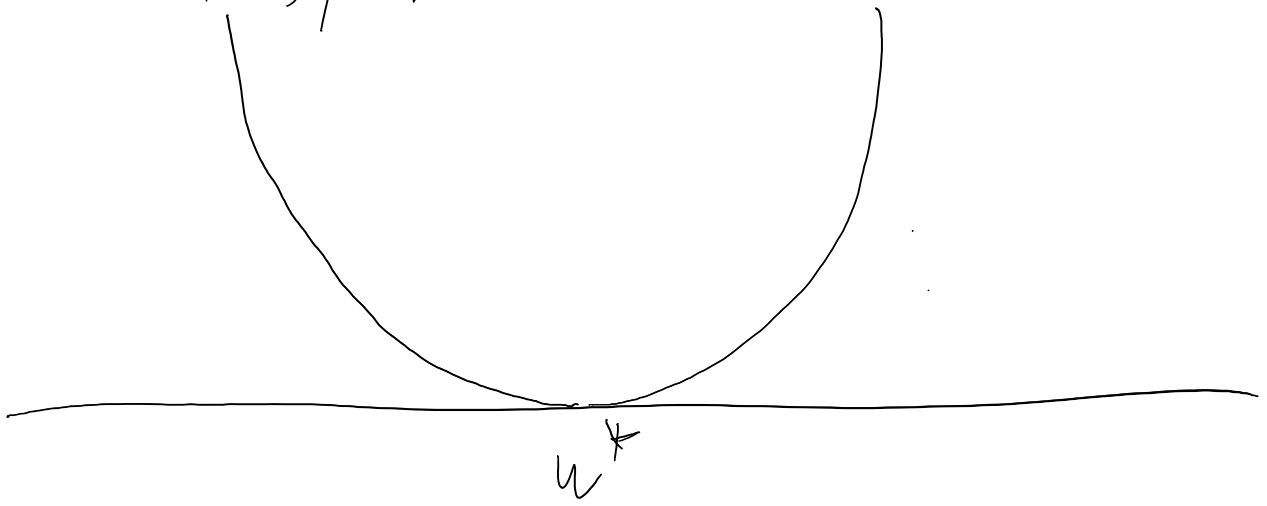
- Key advantage:
  - Iteration cost is O(d), it does not depend on ‘n’.
  - If ‘n’ is 1 billion, it is 1 billion times faster than gradient descent.
- But does this actually work?

Deterministic Gradient Method in Action

Consider just estimating bias:

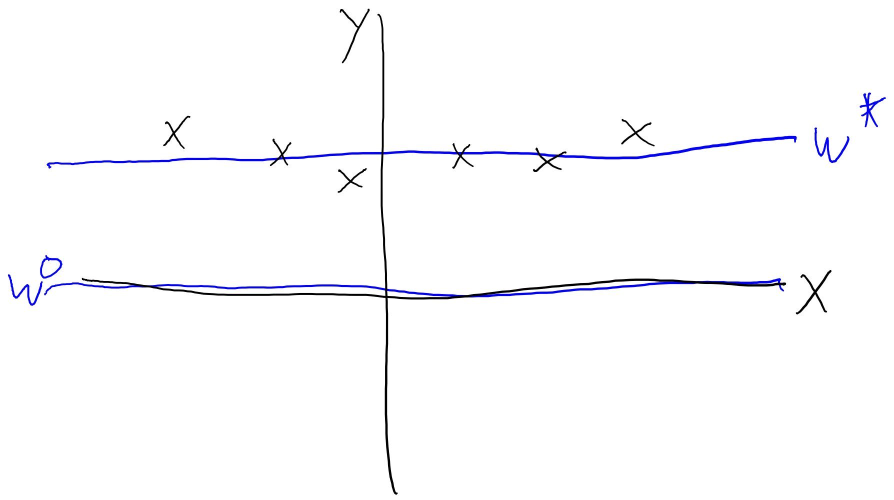


Overall squared error:

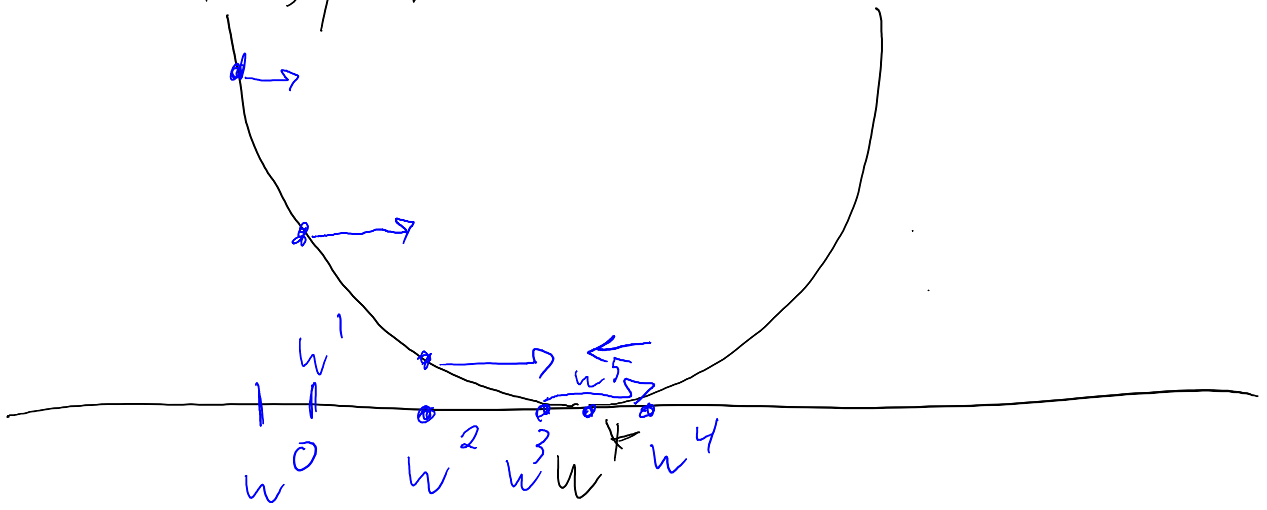


Deterministic Gradient Method in Action

Consider just estimating bias:

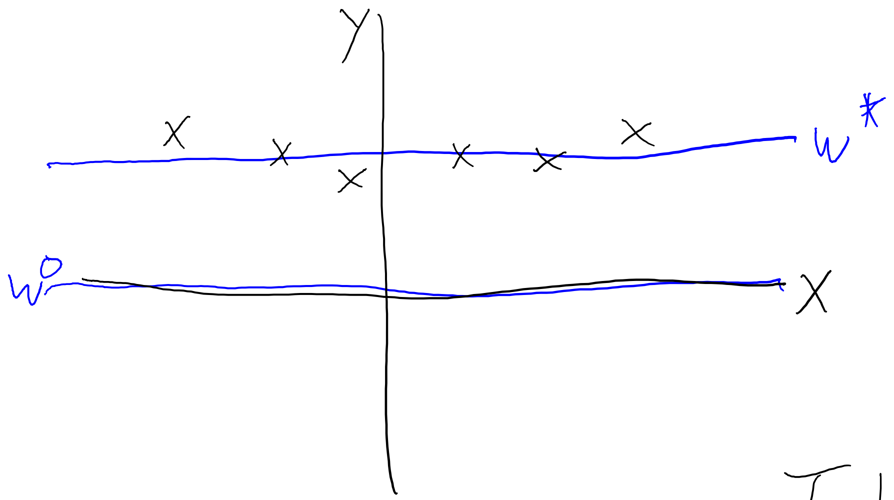


Overall squared error:

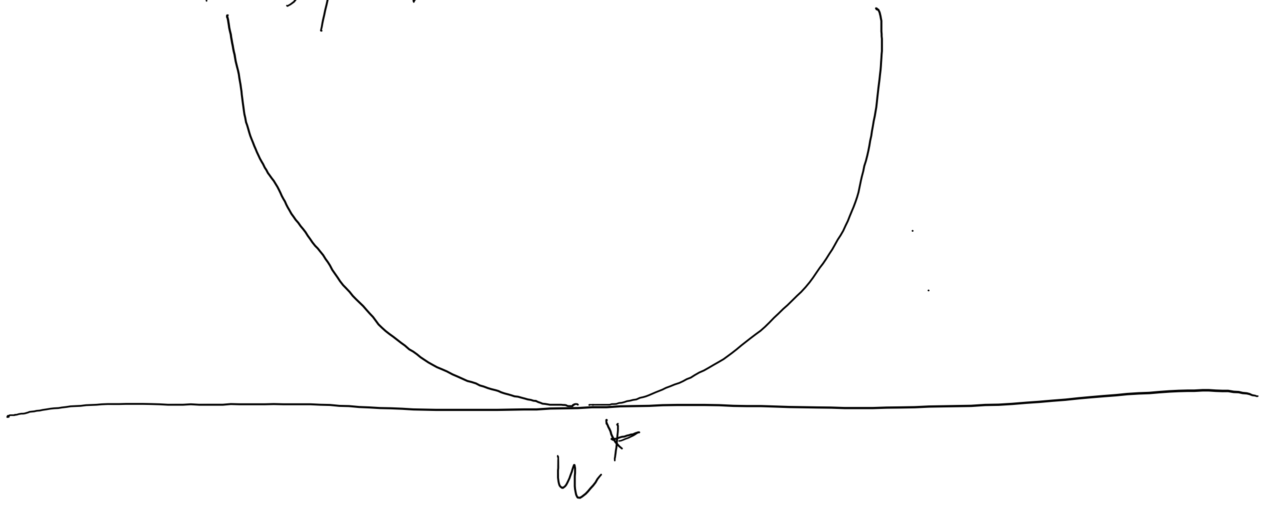


Stochastic Gradient Method in Action

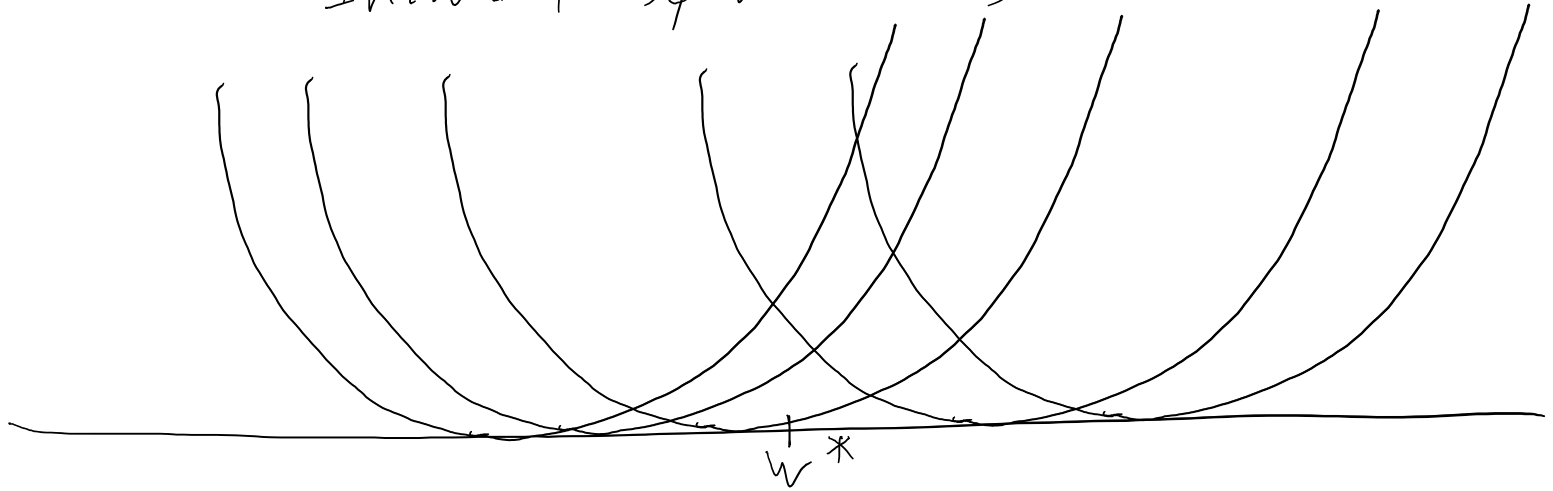
Consider just estimating bias:



Overall squared error:

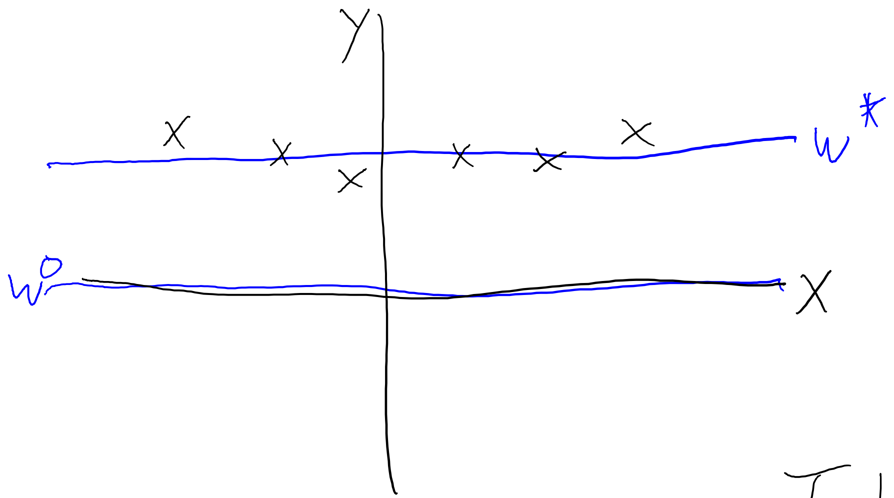


Individual Squared Errors

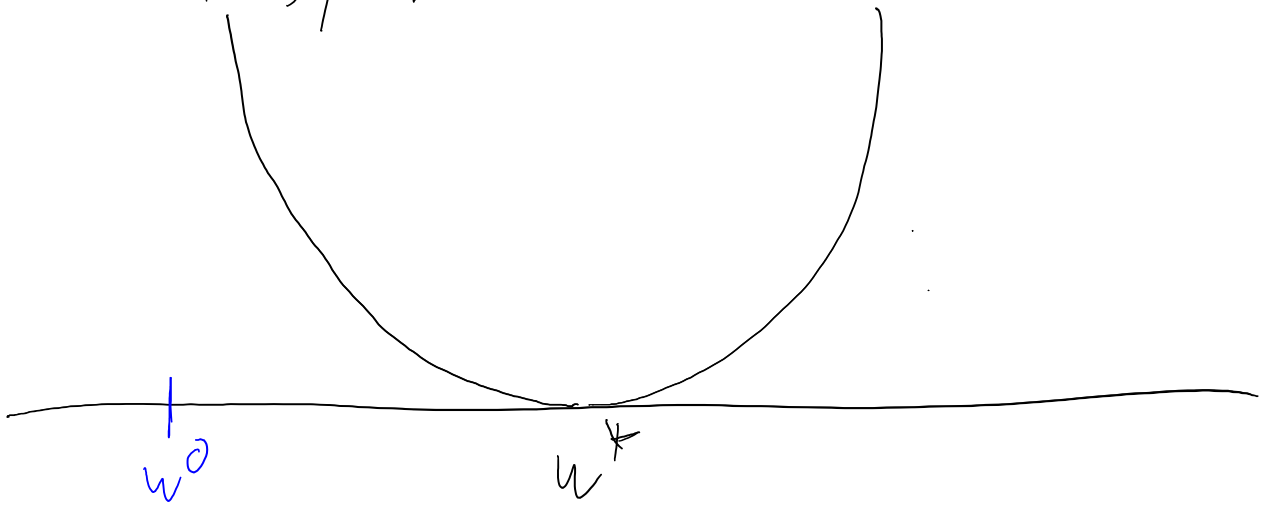


Stochastic Gradient Method in Action

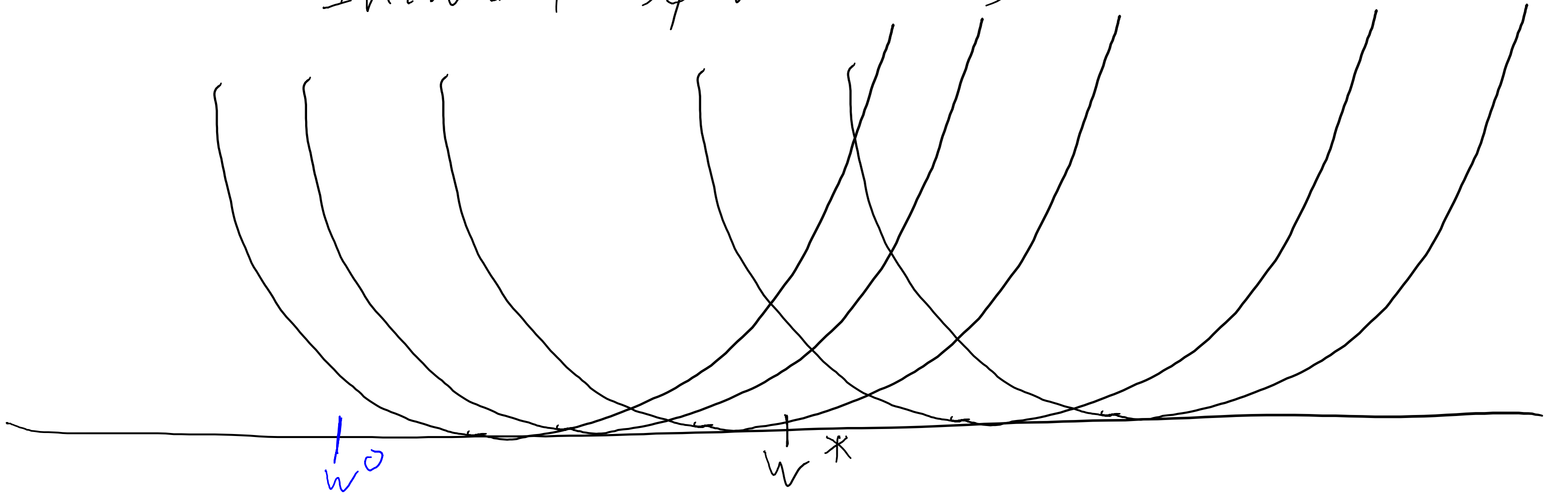
Consider just estimating bias:



Overall squared error:

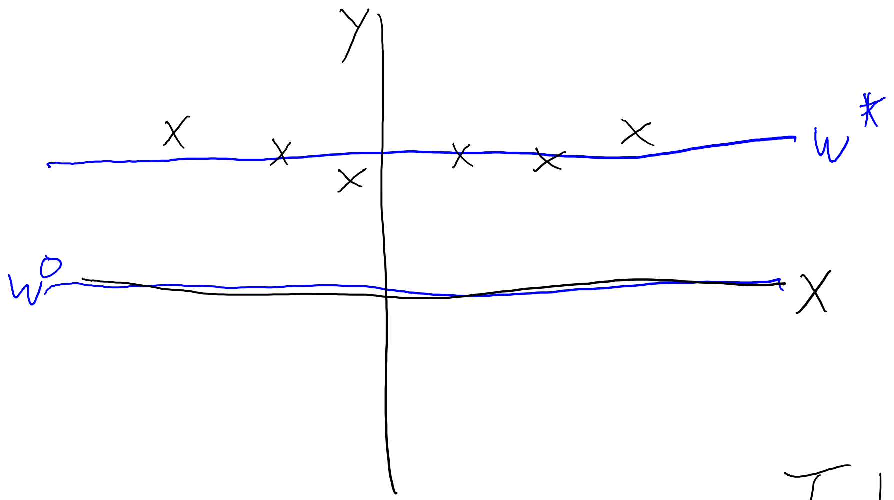


Individual Squared Errors

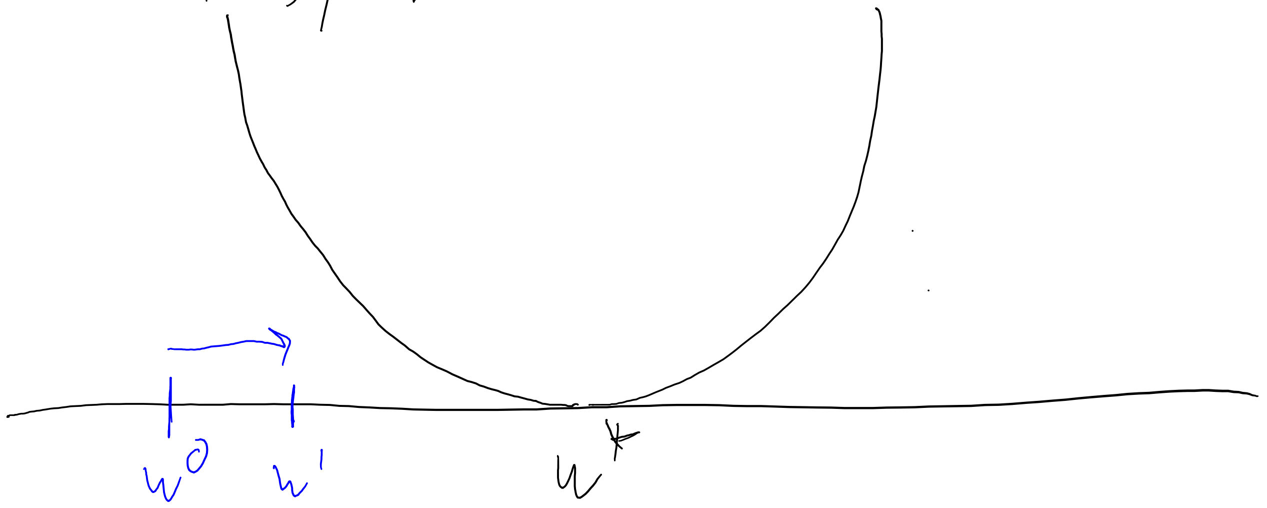


Stochastic Gradient Method in Action

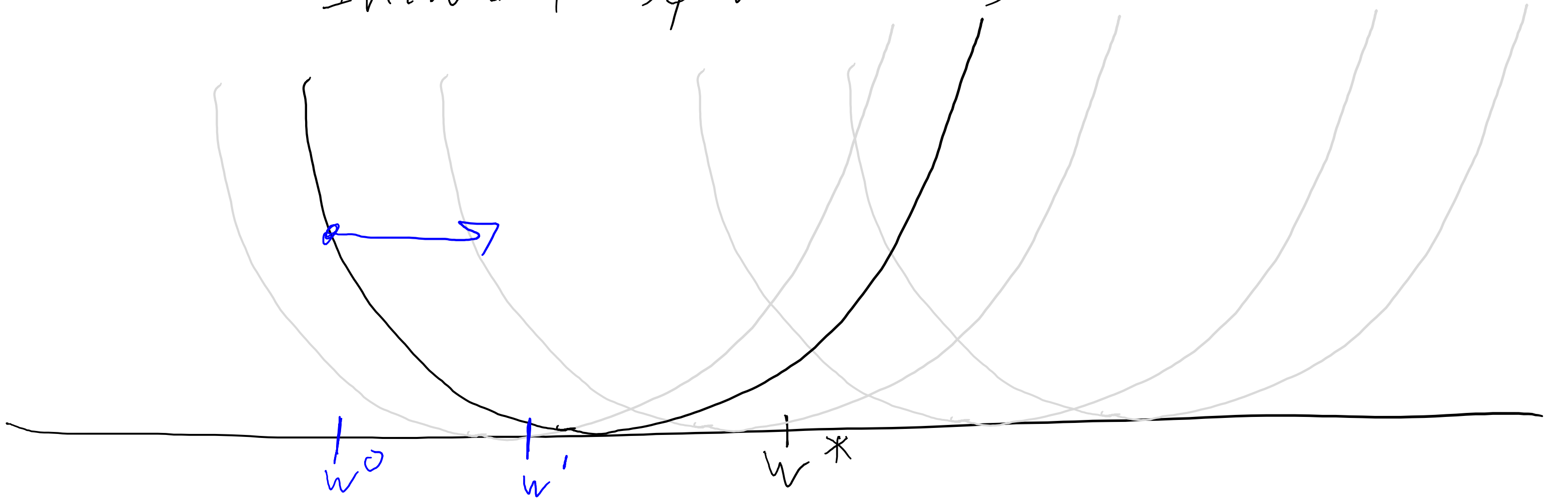
Consider just estimating bias:



Overall squared error:

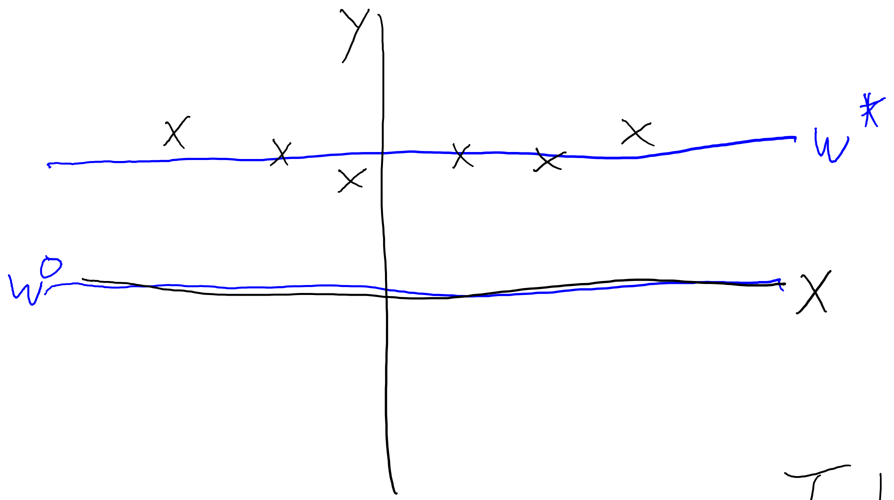


Individual Squared Errors

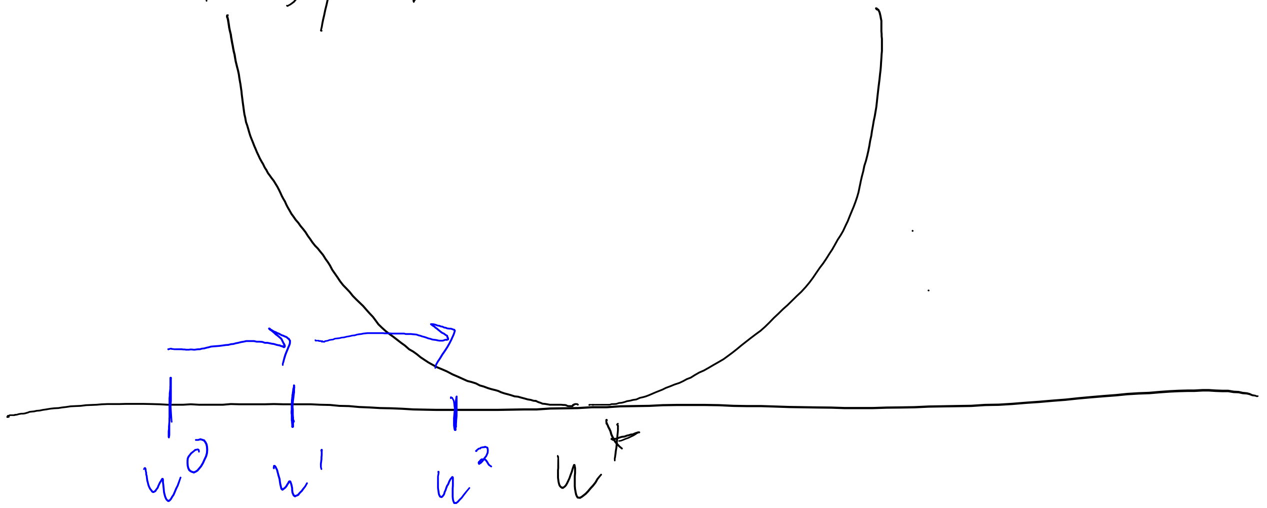


Stochastic Gradient Method in Action

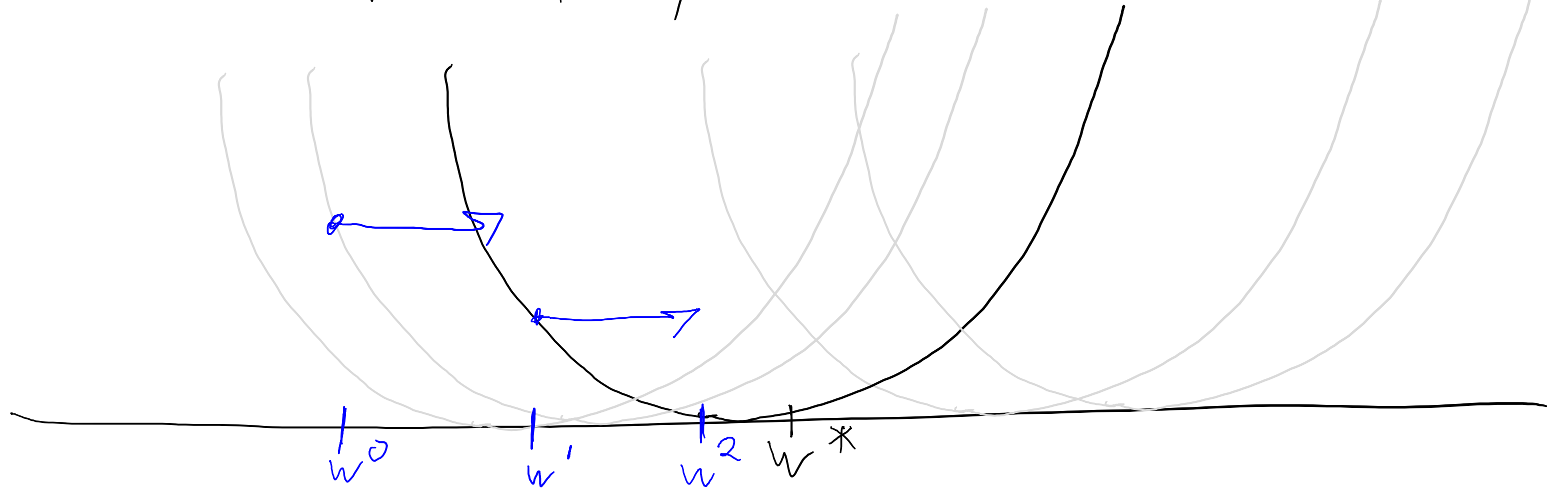
Consider just estimating bias:



Overall squared error:

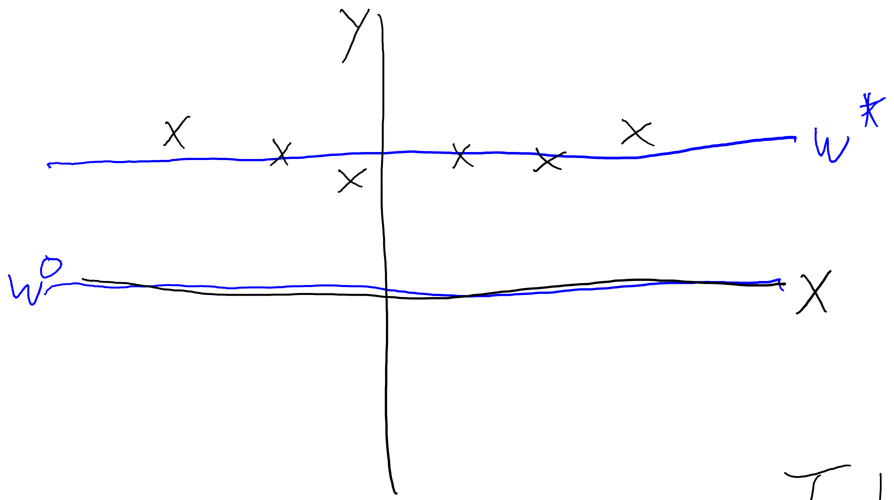


Individual Squared Errors

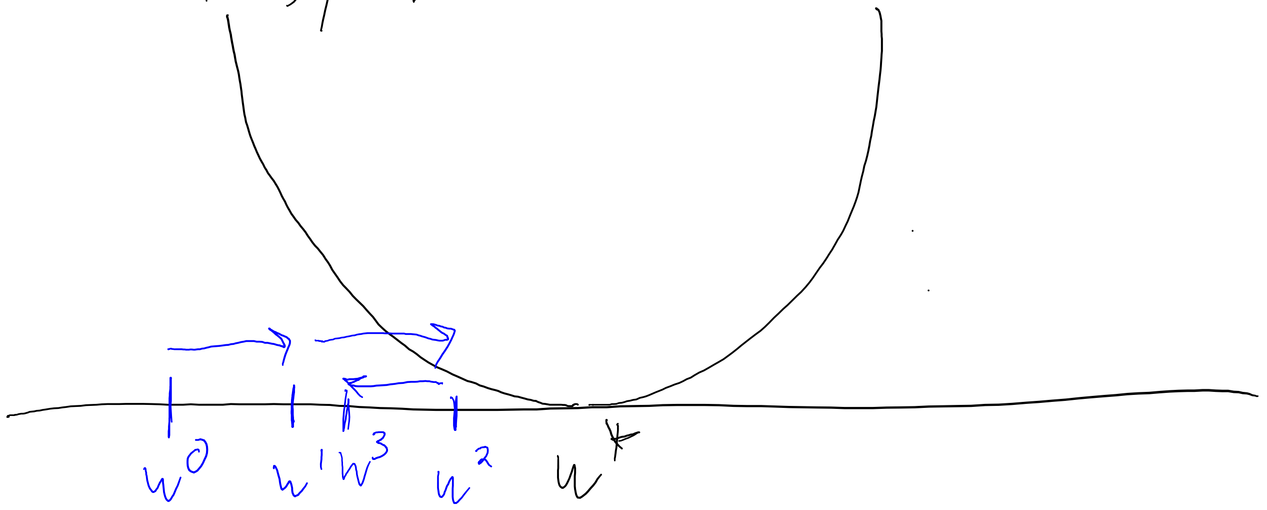


Stochastic Gradient Method in Action

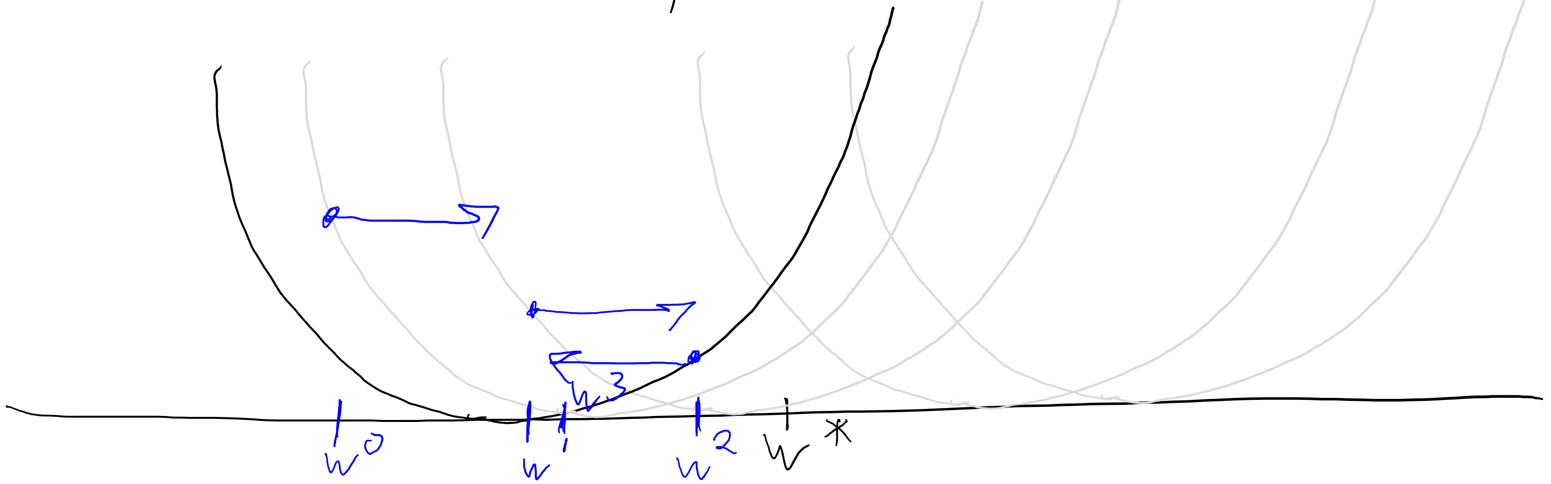
Consider just estimating bias:



Overall squared error:

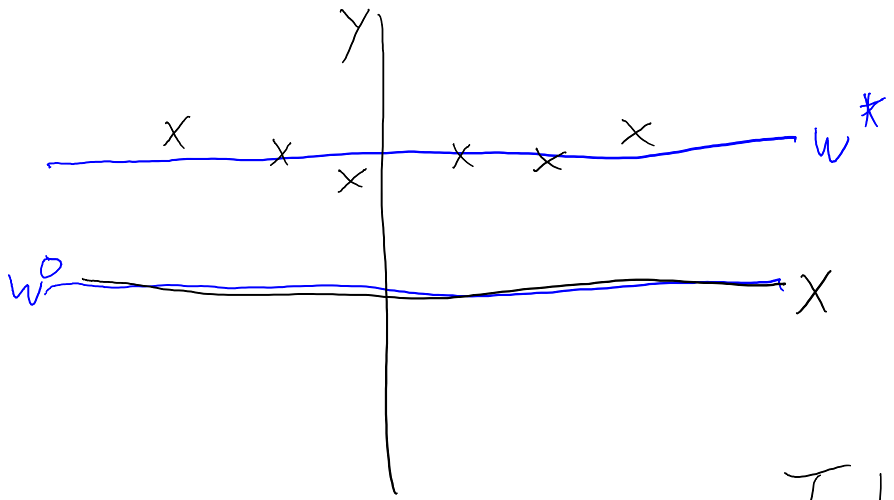


Individual Squared Errors

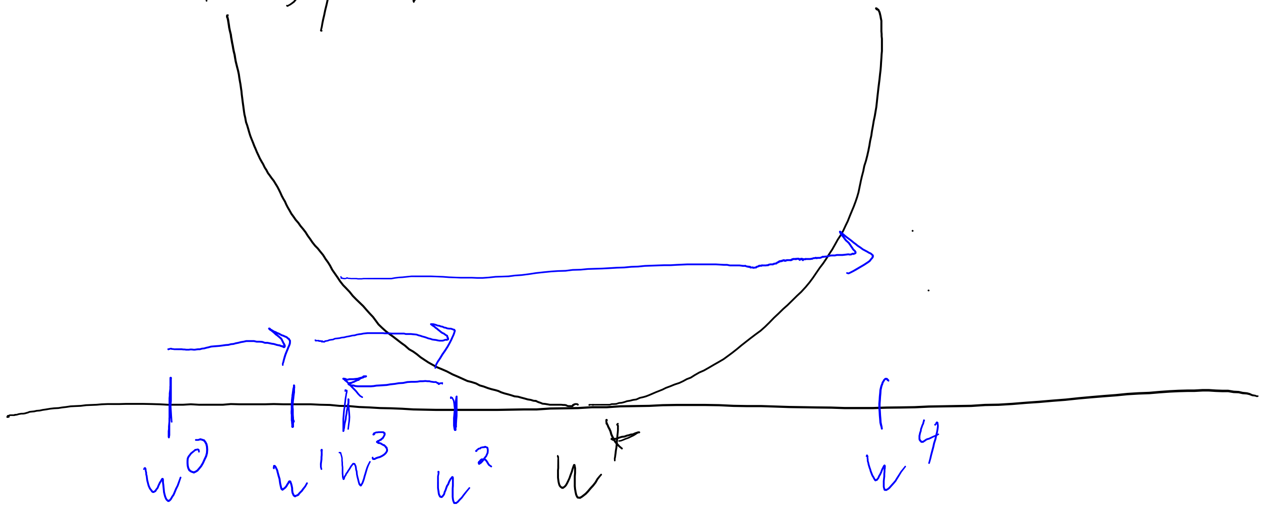


Stochastic Gradient Method in Action

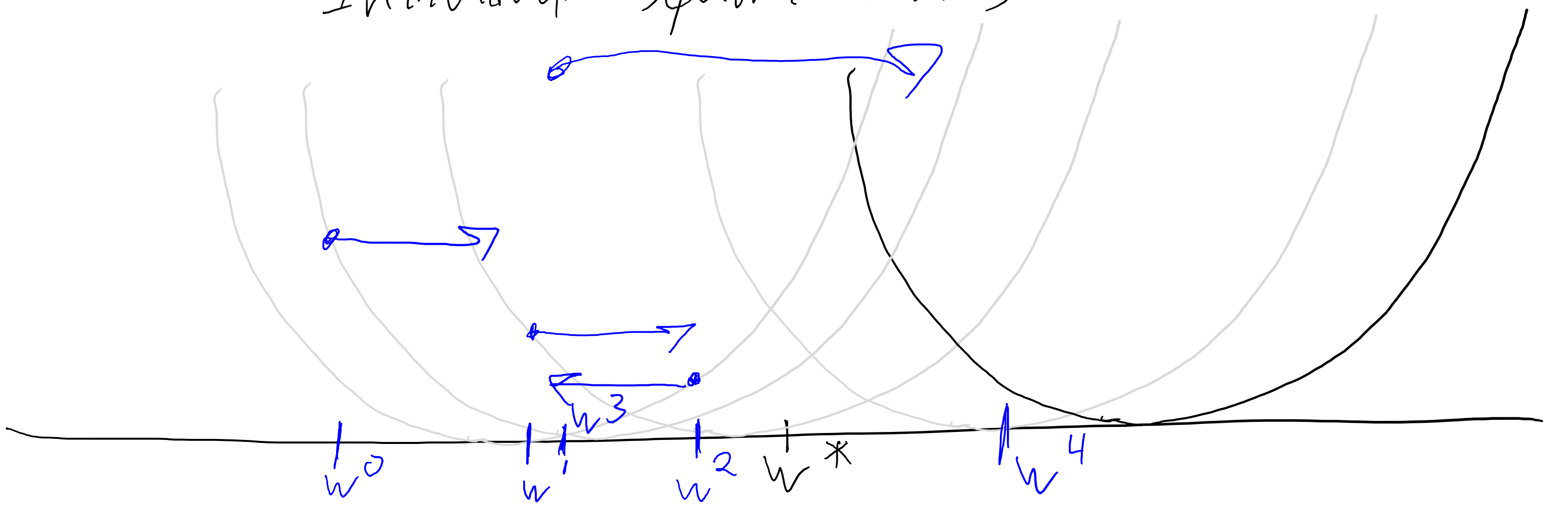
Consider just estimating bias:



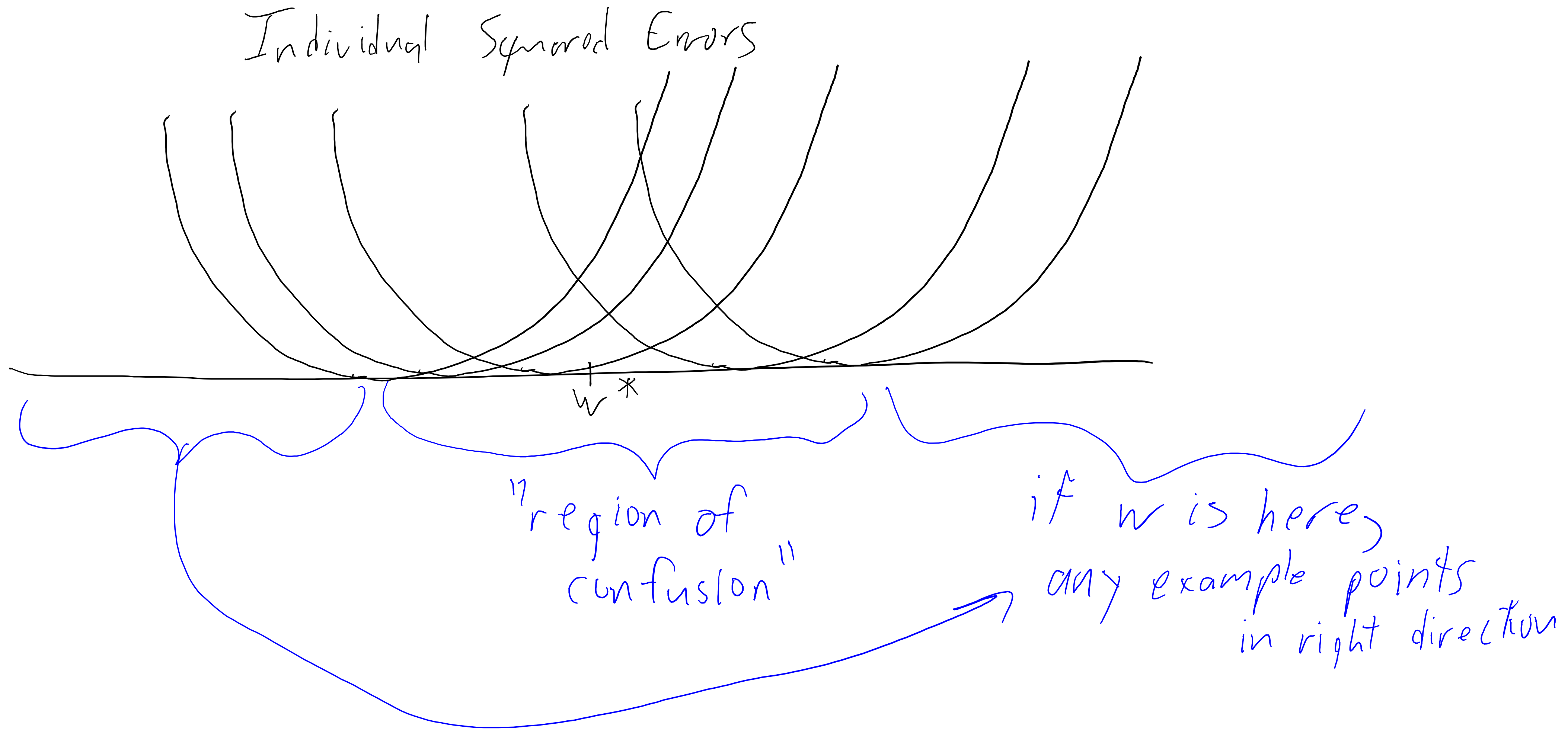
Overall squared error:



Individual Squared Errors



Stochastic Gradient Method in Action

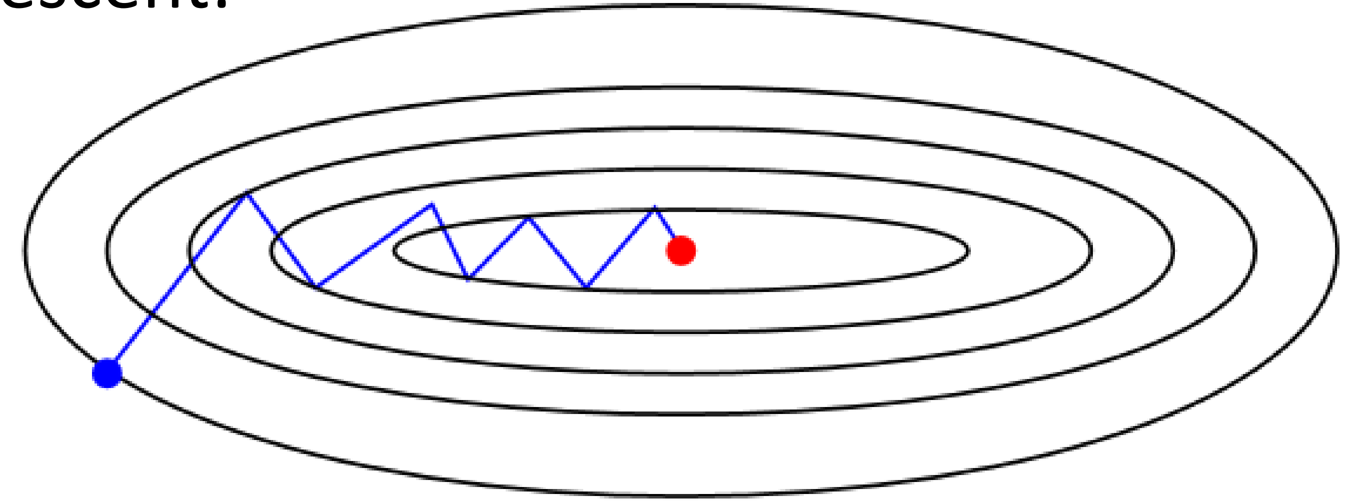

Individual Squared Errors
w^*
"region of confusion"
if w is here, any example points in right direction

Convergence of Stochastic Gradient

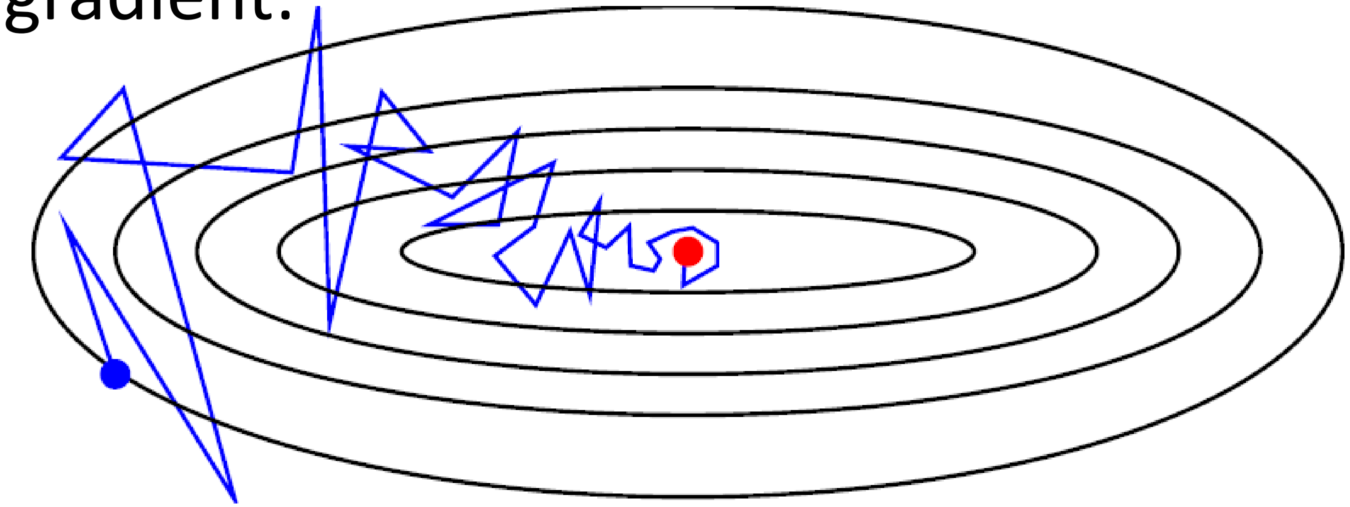
- Problem is that stochastic gradient step might increase error 'f':
  - Since you only look at one example, you can't just check 'f'.
- Key property used for convergence:
  - If the sequence of w^t are sufficiently 'close', we decrease 'f' on average.
  - How 'close' they need to be depends on how close we are to minimum.
- To get convergence, we need a decreasing sequence of step sizes:
  - Need to converge to zero fast enough (makes variance go to 0).
  - Can't converge to zero too quickly (need to be able to get anywhere).
- For example: $\alpha_t = O(\frac{1}{t})$ implies that $\underbrace{\sum_{t=1}^{\infty} \alpha_t = \infty}_{\text{not too small}}$, $\underbrace{\sum_{t=1}^{\infty} \alpha_t^2 < \infty}_{\text{not too big}}$

Deterministic vs. Stochastic Gradient

- Gradient descent:



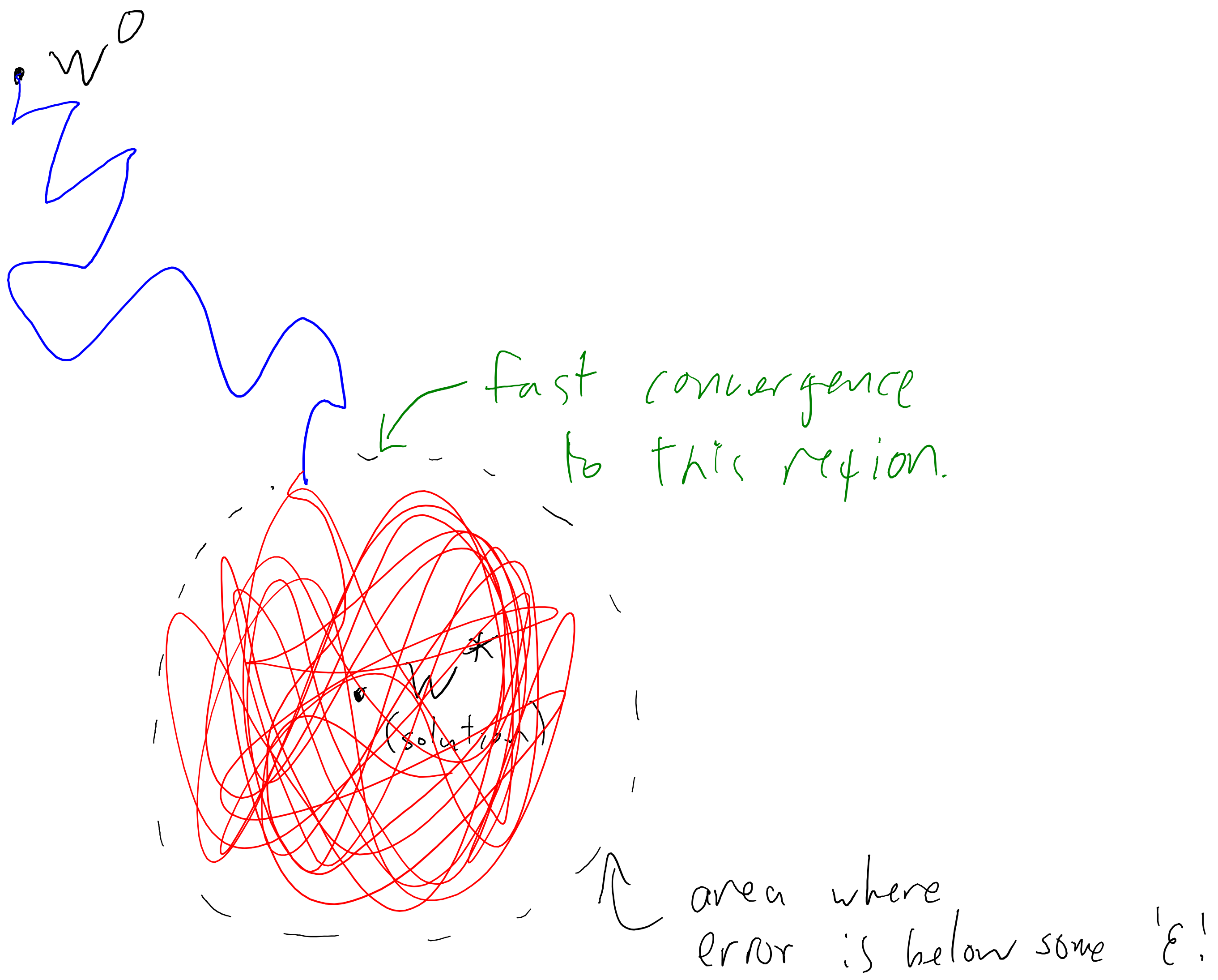
- Stochastic gradient:



Decreasing vs. Constant Step Size

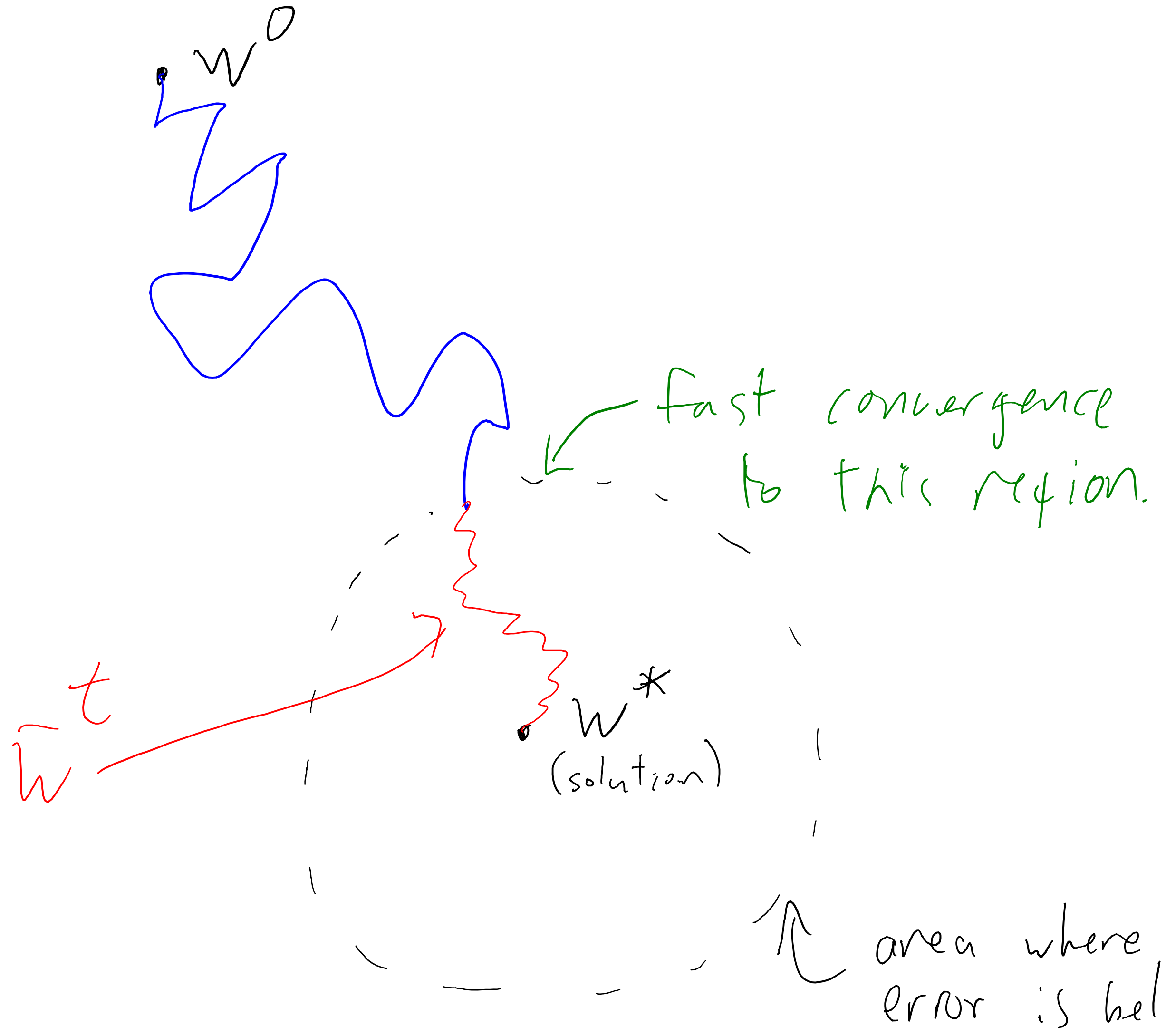
- Stochastic gradient needs decreasing step-sizes for convergence:
  - But with this strategy, convergence rate is very slow.
- Practical alternative: constant step-size.
  - When outside zone of confusion, convergence is very fast.
  - Convergence stays fast until a fixed error level.
    - But random behaviour after this point.
- Another practical alternative:
  - Use bigger step sizes like O(1/$\sqrt{t}$), but average later iterations.
  - Averages out random behaviour.

Stochastic Gradient with Constant Step Size


w^0
fast convergence to this region.
w^* (solution)
area where error is below some 'ε'!

Stochastic Gradient with Averaging

$$\bar{w}^t = \frac{1}{t}\sum_{i=1}^{t} w^i$$

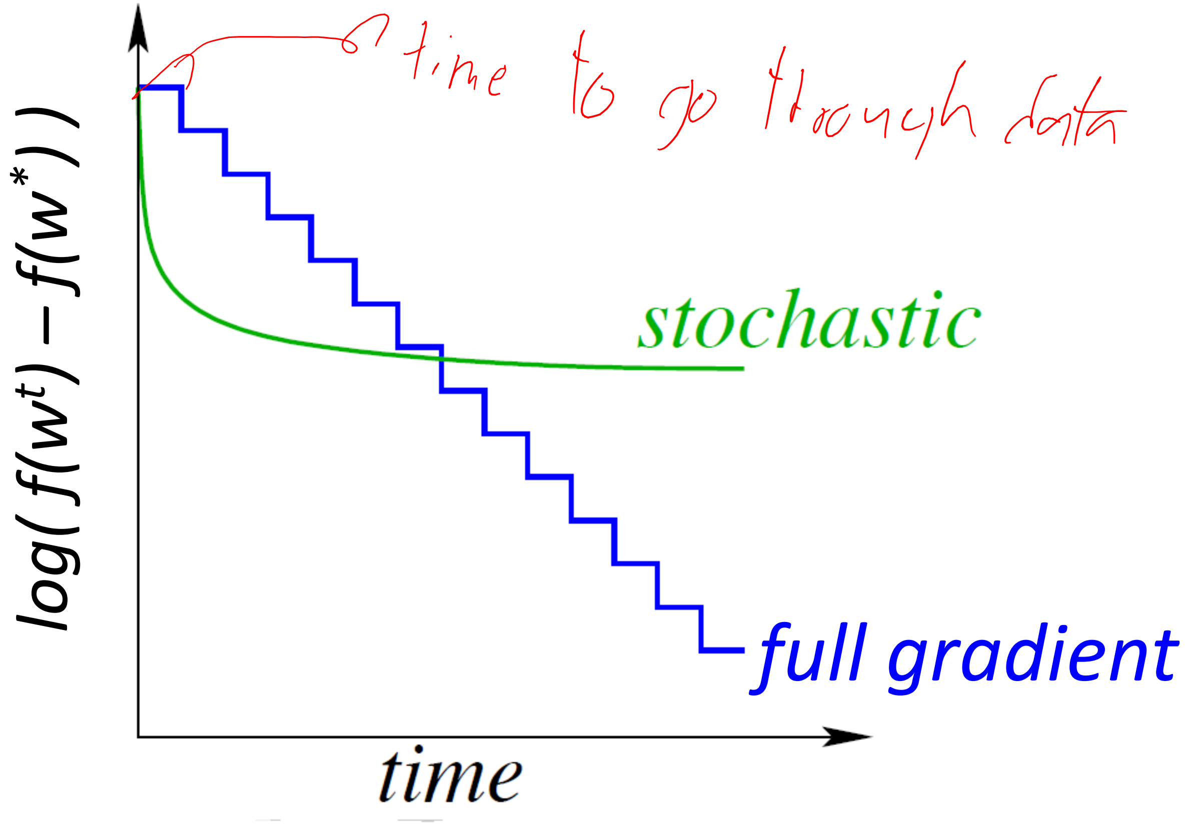


w^0
fast convergence to this region.
$\bar{w}^t$
w^*
(solution)
area where error is below some 'ε'!

Gradient Descent or Stochastic Gradient?

- Assume you want solution with fixed number of digits of accuracy.
- Gradient descent:
  - Iterations are expensive, O(nd).
  - But number of iterations ‘t’ is polynomial in digits of accuracy.
- Stochastic gradient:
  - Iterations are cheap, O(d).
  - But number of iterations ‘t’ is exponential in digits of accuracy.
- In computer science, polynomial usually means ‘efficient’:
  - But stochastic gradient faster for sufficiently large ‘n’ or low time/accuracy.

Gradient Descent vs. Stochastic Gradient



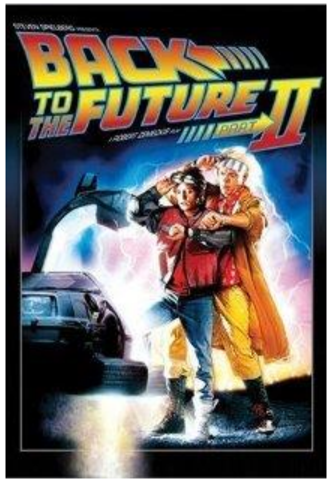
- Since 2012: methods with O(d) cost and polynomial in number of digits.
  - Key idea: if 'n' is finite, you can use a memory instead of having α_t go to zero.
  - First was stochastic average gradient (SAG).

Stochastic Gradient with Infinite Data

- Magical property of stochastic gradient:
  - The classic convergence analysis does not rely on ‘n’ being finite.
- Consider an infinite sequence of IID samples.
  - Or any dataset that is so large we cannot even go through it once.
- Approach 1 (gradient descent):
  - Stop collecting data once you have a very large ‘n’.
  - Fit a model on this fixed dataset (our standard approach).
- Approach 2 (stochastic gradient):
  - Perform a stochastic gradient iteration on each example as we see it.
  - Never re-visit any example, always take a new one.
- Current theory:
  - Approach 2 is doing stochastic gradient on test error, it cannot overfit.
  - Approach 2 achieves test error of Approach 1 (up to constant) after ‘n’ steps.
  - In practice, Approach 1 usually gives lower test error but we don’t know why.

Back to the Future Part 2

- What if our infinite data is coming in over time?
  - Stochastic gradient directly drives down the test error and cannot overfit.
- So can we learn to optimally predict the future?



- No, stochastic gradient requires that the examples are IID.
- But, we can predict the future in some limited ways…

http://www.imdb.com/title/tt0096874/
http://www.october212015.com/

Online Learning

- Online learning starts with a limited set of possible models:
  - For example, the set of all possible linear models $w^T x_i$.
  - Another example would be 50 different global climate models.
- Assumes we get a sequence of samples, but no IID assumption.
  - We could be collecting data over time.
- Framework of online learning:
  1. At each time 't', we receive features of new example x_i.
  2. We predict $\hat{y}_i$ using 'best' model (or weighted combination of models).
  3. We see the true y_i and suffer a loss (such as squared error: $(\hat{y}_i - y_i)^2$).
  4. Update estimate of best model (or model weights) based on true y_i.
- Update:
  - Stochastic gradient step (linear models).
  - More weight on models that predict well (weighted combination).

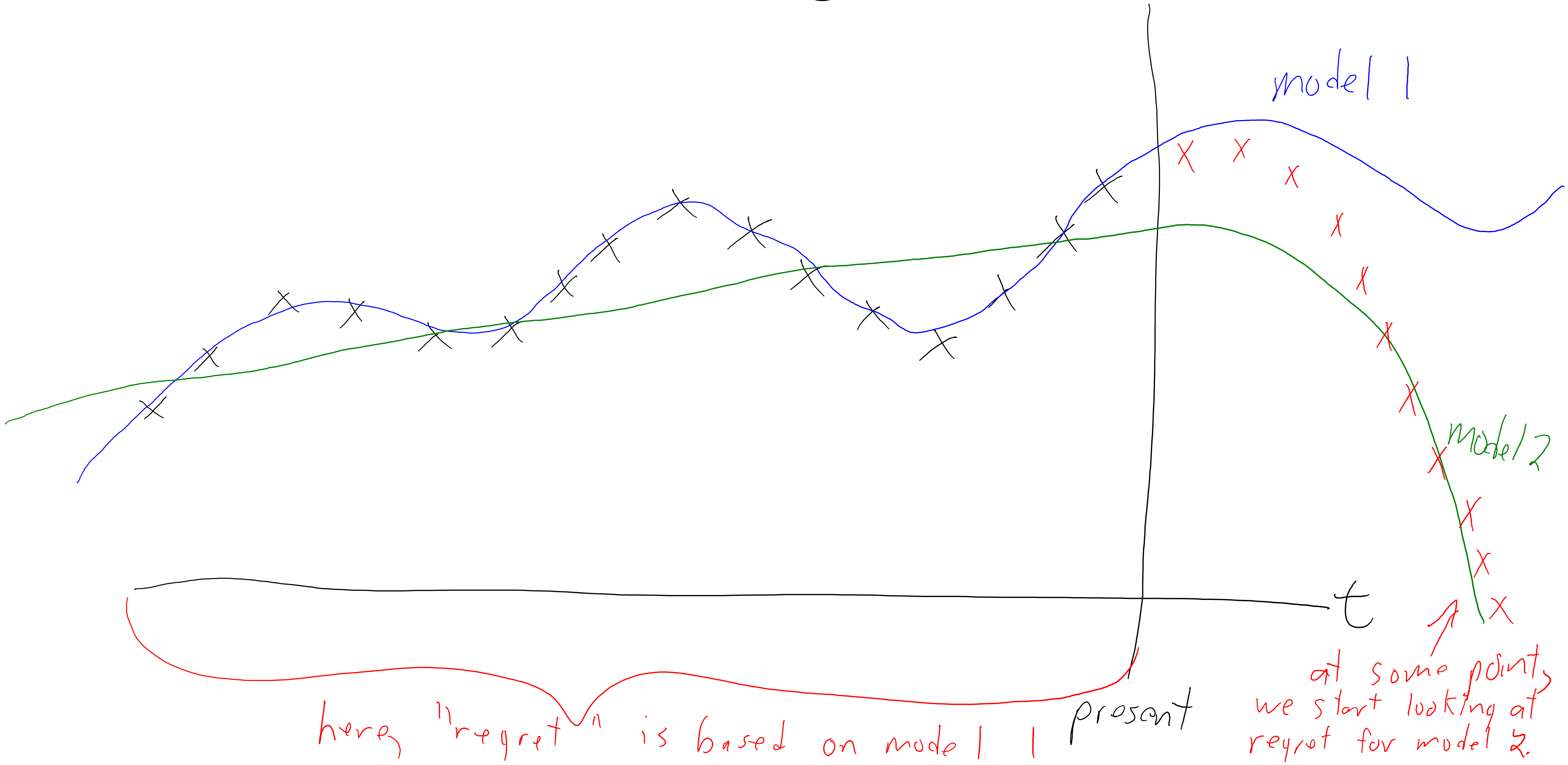
Regret in Online Learning

- Without IID assumption, performance could be arbitrarily bad:
  - Data could be in future "there really is no free lunch" zone.
- But we can still say something about the 'regret':

$$\text{regret}(w_1, w_2, \ldots, w_t) = \sum_{i=1}^{t}\left[g(y_i, \underbrace{w_i}_{\text{the model you \underline{chose} on iteration } i}^{T} x_i) - g(y_i, \underbrace{w_*}_{\text{the best single model from your set, up to iteration } t.}^{T} x_i)\right]$$

- The average regret converges to zero:
  - Does not mean that any of the original models was good.
  - Just means that difference in performance compared to best goes down.

Online Learning in Action



Digression: should ensembles treat models equally?

- Recall the key observation regarding ensemble methods:
  - If models overfit in "different" ways, averaging gives better performance.
- But should all models get equal weight?
  - E.g., decision trees of different depths, when lower depths have low training error.
  - E.g., a random forest where one tree does very well (on validation error) and others do horribly.
  - In online learning, give weight to models that perform well on test data.
  - In science, research may be fraudulent or not based on evidence.
- In these cases, naïve averaging may do worse.

Bayesian Model Averaging

- Suppose we have a set of ‘m’ probabilistic binary classifiers w_j.
- If each one gets equal weight, then we predict using:

$$p(y_i \mid x_i) = \frac{1}{m} p(y_i \mid w_1, x_i) + \frac{1}{m} p(y_i \mid w_2, x_i) + \dots + \left(\frac{1}{m}\right) p(y_i \mid w_m, x_i)$$

- Bayesian model averaging treats model as a random variable:

Assume $w_j \perp x_i$

$$p(y_i \mid x_i) = \sum_{j=1}^{m} p(y_i, w_j \mid x_i) = \sum_{j=1}^{m} p(y_i \mid w_j, x_j) p(w_j \mid x_j) \overset{\uparrow}{=} \sum_{j=1}^{m} p(y_i \mid w_j, x_j) p(w_j)$$

- So we should weight by probability that w_j is the correct model:
  - Equal weights assume all models are equally probable.

Bayesian Model Averaging

Again, assuming $w_j \mid X$

- Can get better weights by conditioning on training set:

$$p(w_j \mid X, y) \propto p(y \mid w_j, X)\, p(w_j \mid X) = p(y \mid w_j, X)\, p(w_j)$$

- The 'likelihood' $p(y \mid w_j, X)$ makes sense:
  - We should give more weight to models that predict 'y' well.
  - Note that hidden denominator penalizes complex models.
- The 'prior' $p(w_j)$ is our 'belief' that w_j is the correct model.
- This is how rules of probability say we should weigh models.
  - The 'correct' way to predict the future given what we know.
  - But it makes people uncomfortable because it is subjective.

Conditioning by Observation vs. by Intervention

- Conditioning by observation:
  - If I see my watch says 3:50, the weekend is almost here.
- Conditioning by intervention:
  - If I set my watch to say 3:50, it doesn't help.
- If we plan to take actions, we need to model effects of the actions:
  - Otherwise, predictions could be meaningless.
- Leads us into causality, planning, and reinforcement learning.

  (but not in this course)

Summary

- Stochastic gradient methods let us use huge datasets.
- Convergence of stochastic gradient requires decreasing step sizes.
- Stochastic gradient with infinite data has nearly-optimal test error.
- Online learning can minimize ‘regret’ for non-IID data.
- Bayesian model averaging give coherent way to combine models.

- Next time:
  - What ‘parts’ are my personality made of?