CPSC 340: Machine Learning and Data Mining

Stochastic Gradient

Fall 2016

Admin

- Assignment 3:
  - 3 late days before class Wednesday.
  - Solutions will be posted after class Wednesday.
- Extra office hours Thursday:
  - 10:30-12 and 4:30-6 in X836.
- Midterm Friday:
  - Midterm from last year and list of topics posted (covers Assignments 1-3).
    - Tutorials this week will cover practice midterm (and non-1D version of Q5).
  - In class, 55 minutes, closed-book, cheat sheet: 2-pages each double-sided.

Big-N Problems

- Consider fitting a least squares model:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2$$

- Gradient methods are effective when 'd' is very large.
  - O(nd) per iteration instead of $O(nd^2 + d^3)$ to solve as linear system.
- What if number of training examples 'n' is very large?
  - All Gmails, all products on Amazon, all homepages, all images, etc.

Gradient Descent vs. Stochastic Gradient

- Recall the gradient descent algorithm:

$$w^{t+1} = w^t - \alpha^t \nabla f(w^t)$$

- For least squares, our gradient has the form:

$$\nabla f(w) = \sum_{i=1}^{n} (w^T x_i - y_i) x_i$$

- The cost of computing the gradient is linear in 'n'.
  - As 'n' gets large, gradient descent iterations become expensive.

Gradient Descent vs. Stochastic Gradient

- Common solution to this problem is stochastic gradient algorithm:

$$w^{t+1} = w^t - \alpha^t \nabla f_i(w^t)$$

- Uses gradient of randomly-chosen training example:

$$\nabla f_i(w) = (w^T x_i - y_i) x_i$$

- Cost of computing this gradient is independent of 'n'.
  - Iterations are 'n' times faster than gradient descent iterations.

Stochastic Gradient (SG)

- Stochastic gradient is an algorithm for minimizing averages:

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}(w^Tx_i - y_i)^2 \quad \text{(squared error)}$$

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}h(w^Tx_i - y_i)^2 \quad \text{(Huber loss)}$$

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_iw^Tx_i)) \quad \text{(logistic regression)}$$

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}f_i(w) \quad \text{(our } \underline{\text{notation}} \text{ for the general case)}$$

- Key advantage: iterations cost doesn't depend on 'n'.

Stochastic Gradient (SG)

- Stochastic gradient is an iterative optimization algorithm:
  - We start with some initial guess, w^0.
  - Generate new guess by moving in the negative gradient direction:

$$w^1 = w^0 - \alpha^0 \nabla f_i(w^0)$$

    - For a random training example 'i'.
  - Repeat to successively refine the guess:

$$w^{t+1} = w^t - \alpha^t \nabla f_i(w^t) \quad \text{for } t = 1, 2, 3, \ldots$$

    - For a random training example 'i'.

Why Does Stochastic Gradient Work / Not Work?

- Main problem with stochastic gradient:
  - Gradient of random example might point in the wrong direction.

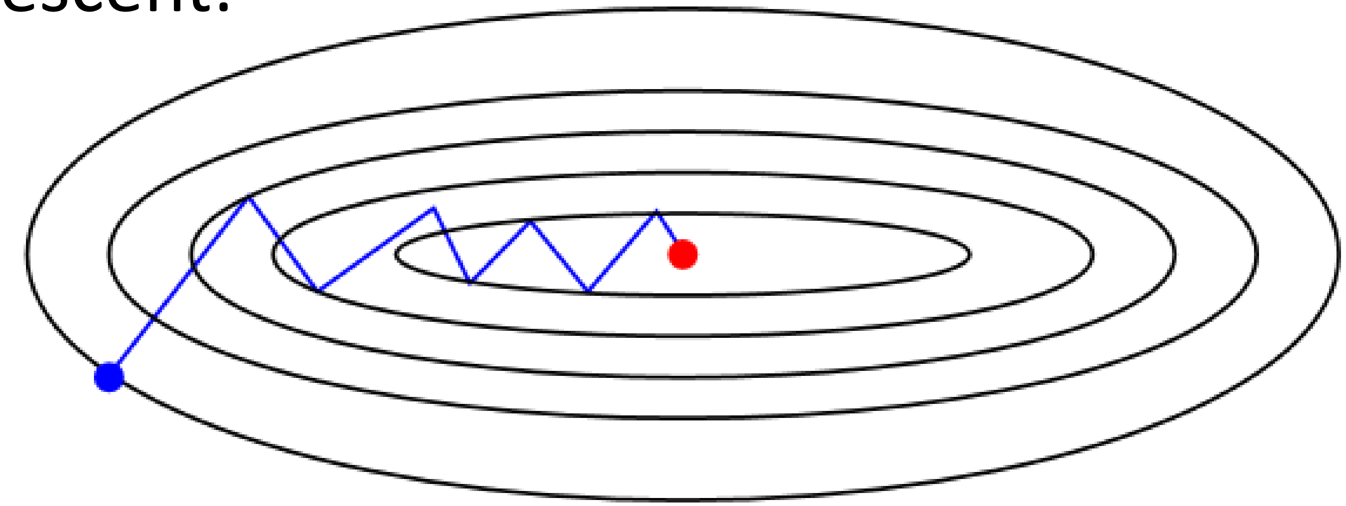
- Does this have any hope of working?
  - The average of the random gradients is the full gradient.

Mean over $\nabla f_i(w^t)$ is $\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w^t)$ which is $\nabla f(w^t)$

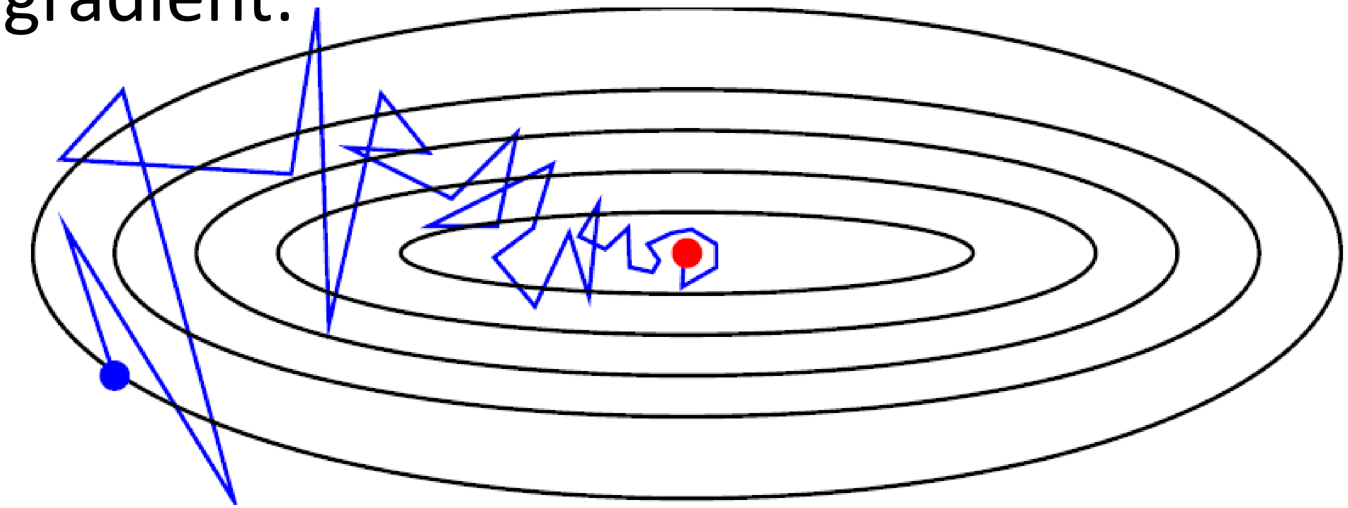
  - The algorithm is going in the right direction on average.

Gradient Descent vs. Stochastic Gradient (SG)

- Gradient descent:

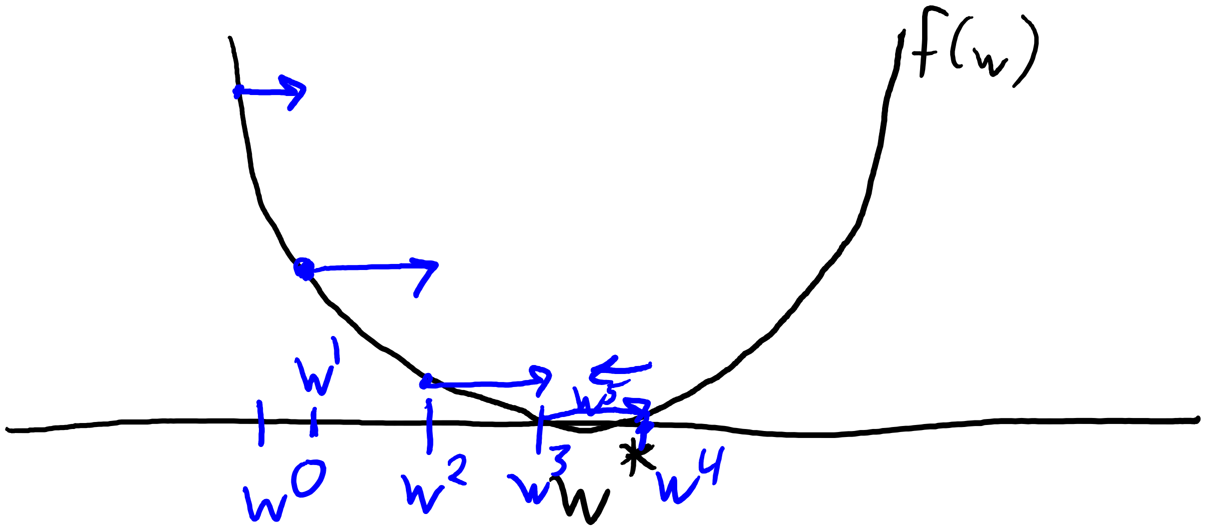


- Stochastic gradient:



Gradient Descent in Action

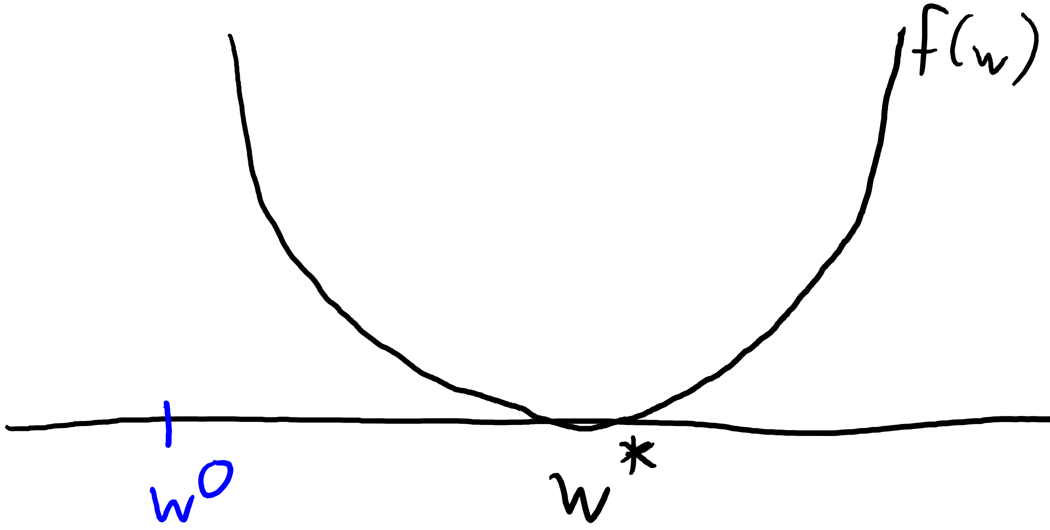
$$f(w) = \frac{1}{5}\sum_{i=1}^{5} (w^T x_i - y_i)^2$$

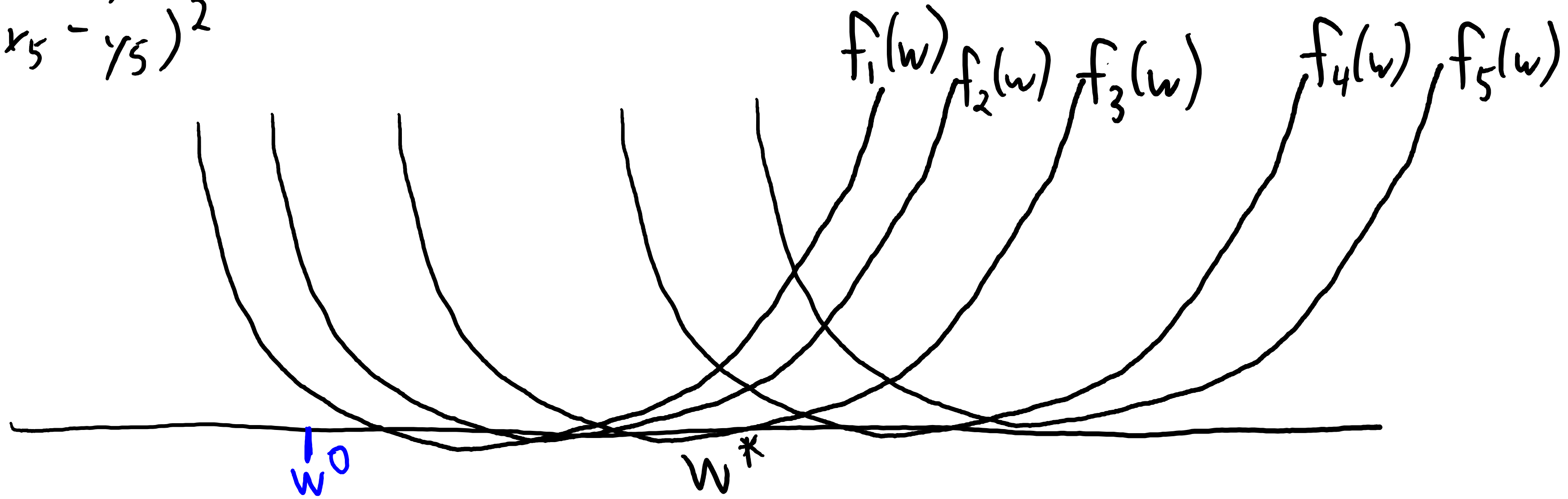


Stochastic Gradient in Action

$$f(w) = \frac{1}{5}\sum_{i=1}^{5}(w^T x_i - y_i)^2$$

$$f_1(w) = (w^T x_1 - y_1)^2$$
$$f_2(w) = (w^T x_2 - y_2)^2$$
$$f_3(w) = (w^T x_3 - y_3)^2$$
$$f_4(w) = (w^T x_4 - y_4)^2$$
$$f_5(w) = (w^T x_5 - y_5)^2$$





Stochastic gradient minimizes <u>average</u> value.

Stochastic Gradient in Action

$$f(w) = \frac{1}{5}\sum_{i=1}^{5} (w^T x_i - y_i)^2$$

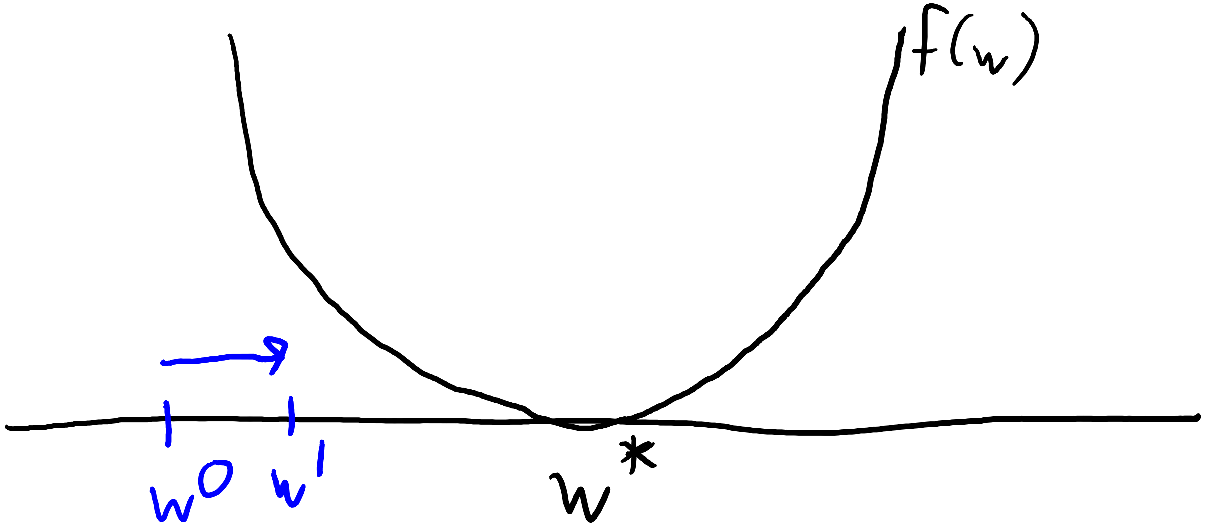
$$f_1(w) = (w^T x_1 - y_1)^2$$

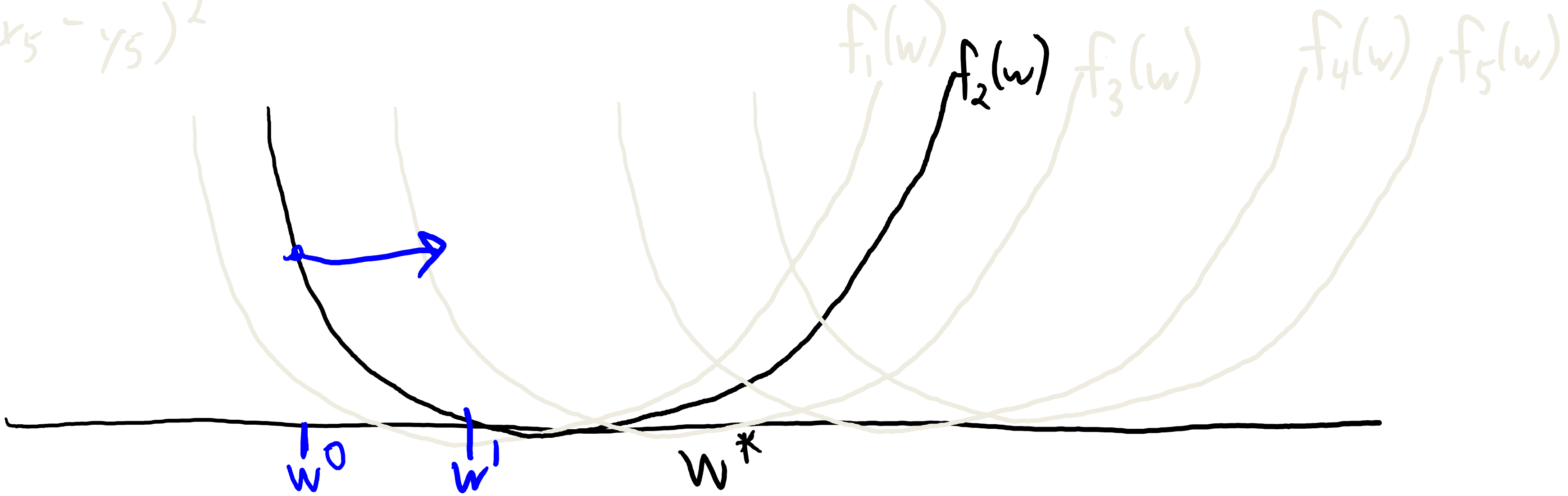
$$f_2(w) = (w^T x_2 - y_2)^2$$

$$f_3(w) = (w^T x_3 - y_3)^2$$

$$f_4(w) = (w^T x_4 - y_4)^2$$

$$f_5(w) = (w^T x_5 - y_5)^2$$





Stochastic gradient minimizes <u>average</u> value.

Stochastic Gradient in Action

$$f(w) = \frac{1}{5}\sum_{i=1}^{5}(w^T x_i - y_i)^2$$

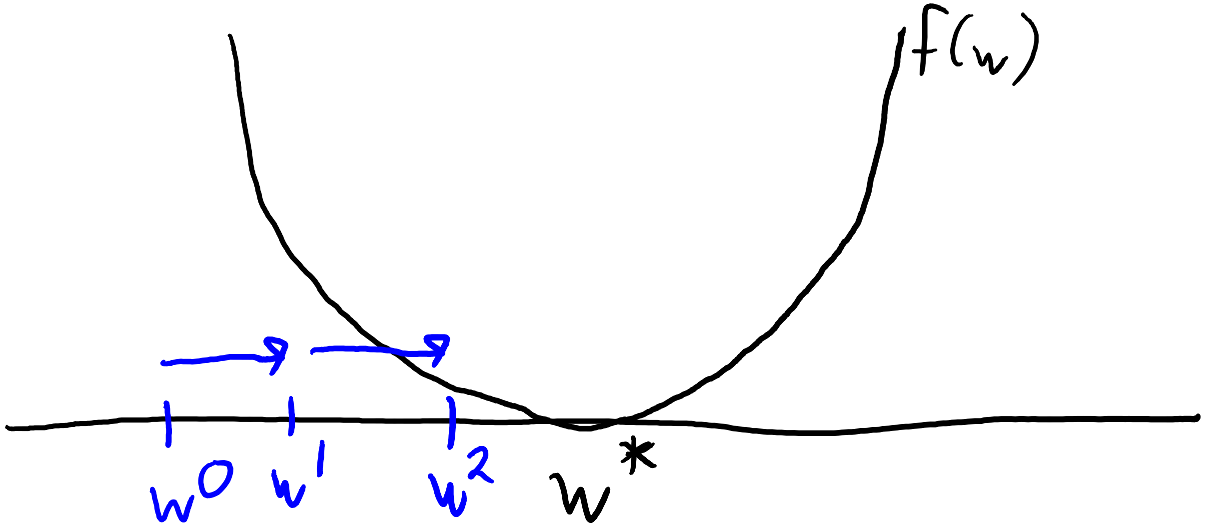
$$f_1(w) = (w^T x_1 - y_1)^2$$

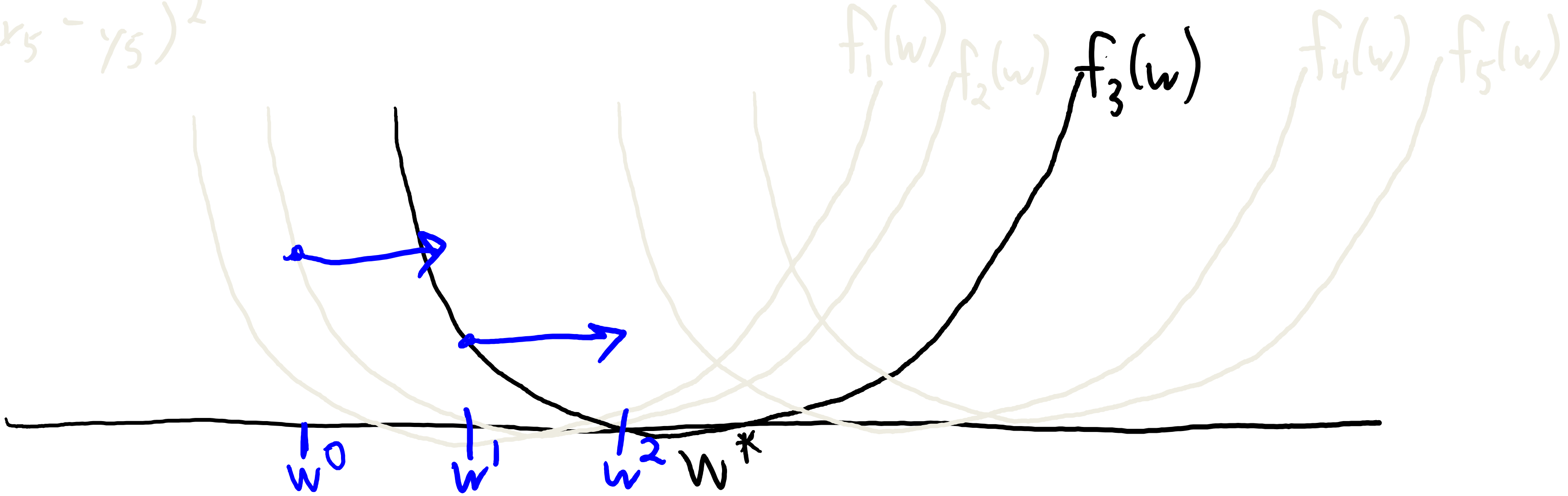
$$f_2(w) = (w^T x_2 - y_2)^2$$

$$f_3(w) = (w^T x_3 - y_3)^2$$

$$f_4(w) = (w^T x_4 - y_4)^2$$

$$f_5(w) = (w^T x_5 - y_5)^2$$





Stochastic gradient minimizes average value.

Stochastic Gradient in Action

$$f(w) = \frac{1}{5}\sum_{i=1}^{5}(w^T x_i - y_i)^2$$

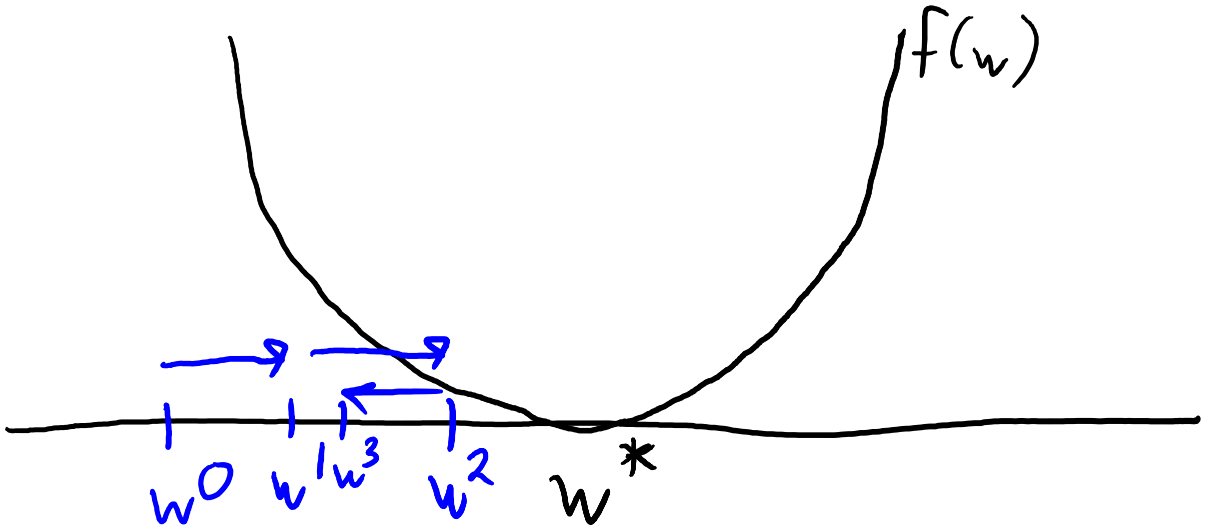
$$f_1(w) = (w^T x_1 - y_1)^2$$

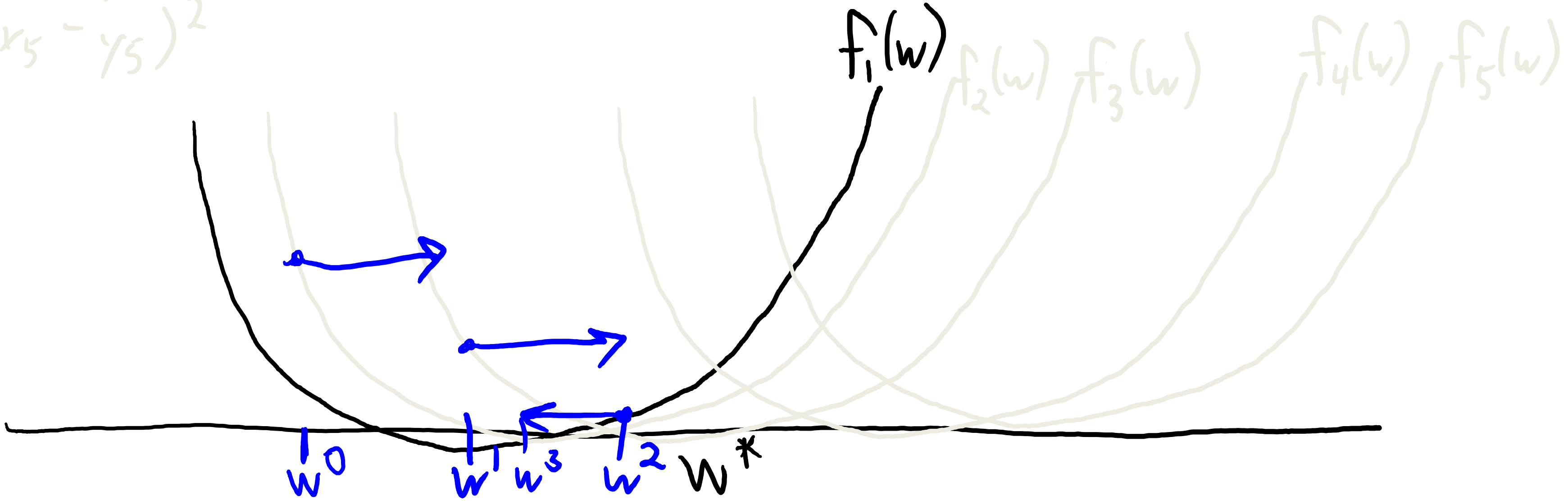
$$f_2(w) = (w^T x_2 - y_2)^2$$

$$f_3(w) = (w^T x_3 - y_3)^2$$

$$f_4(w) = (w^T x_4 - y_4)^2$$

$$f_5(w) = (w^T x_5 - y_5)^2$$





Stochastic gradient minimizes <u>average</u> value.

Stochastic Gradient in Action

$$f(w) = \frac{1}{5}\sum_{i=1}^{5}(w^T x_i - y_i)^2$$

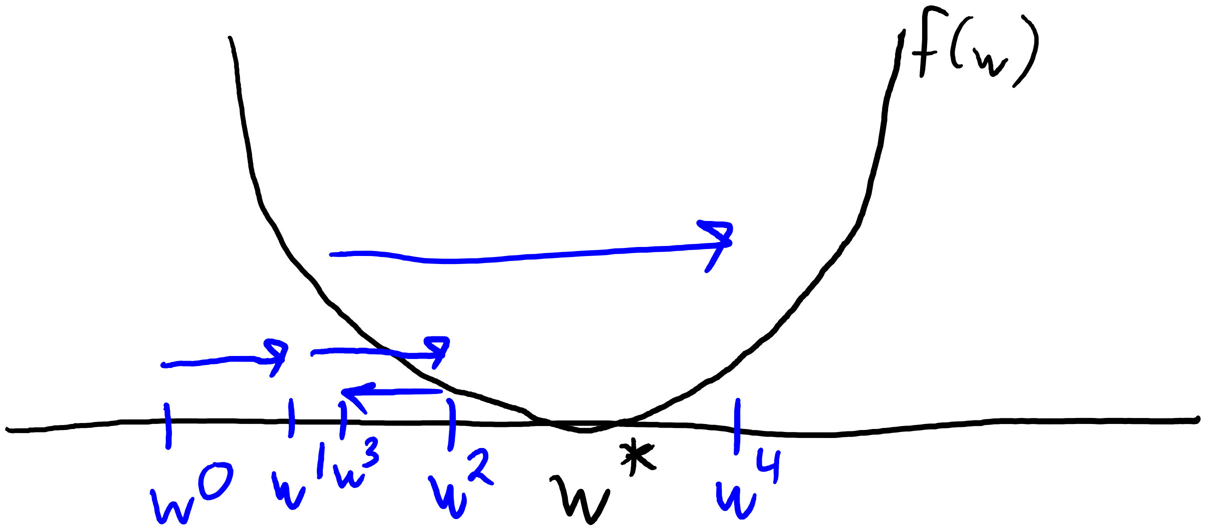
$$f_1(w) = (w^T x_1 - y_1)^2$$

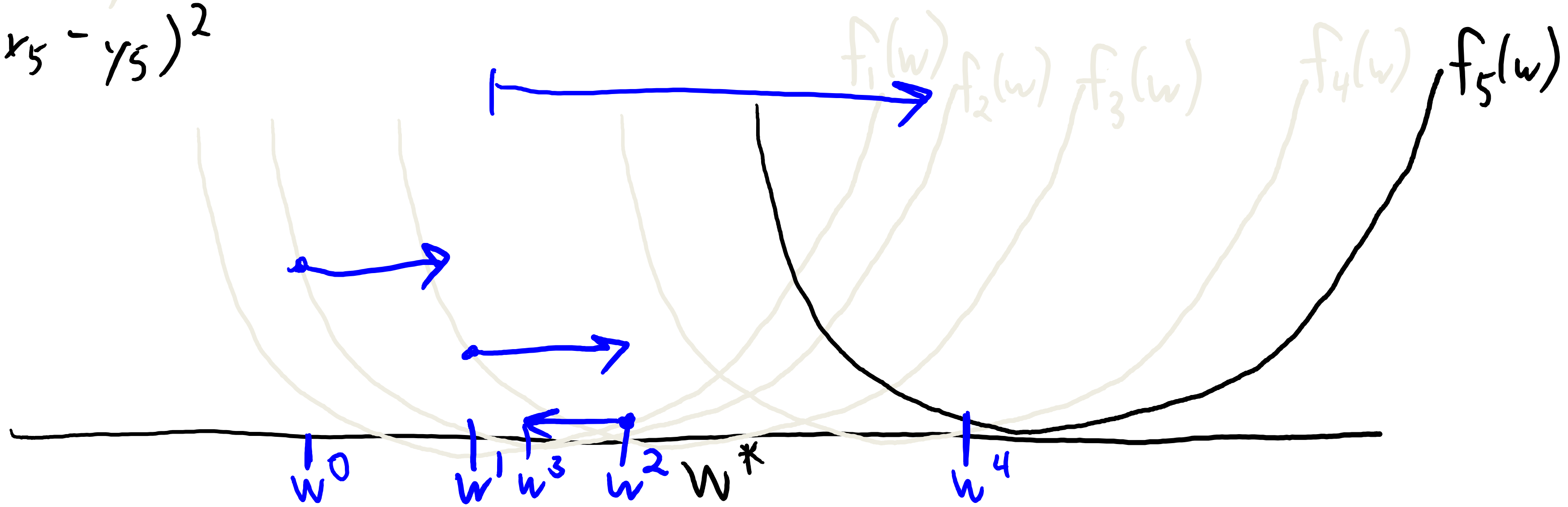
$$f_2(w) = (w^T x_2 - y_2)^2$$

$$f_3(w) = (w^T x_3 - y_3)^2$$

$$f_4(w) = (w^T x_4 - y_4)^2$$

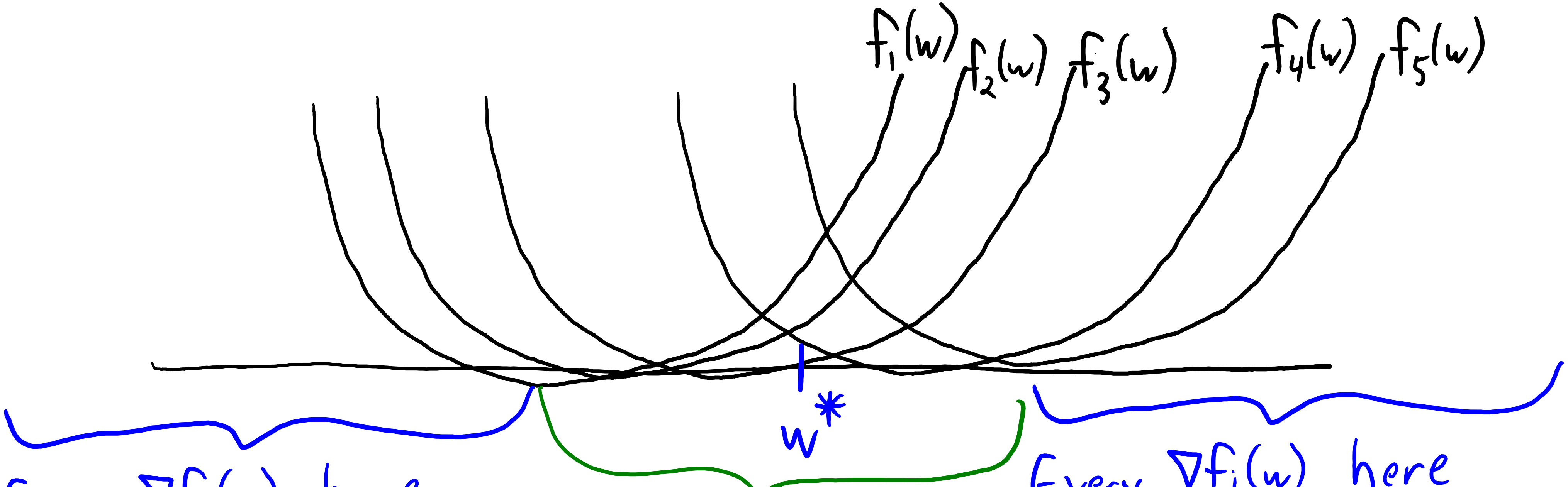
$$f_5(w) = (w^T x_5 - y_5)^2$$





Stochastic gradient minimizes average value.

Effect of ‘w’ Location on Progress



Every $\nabla f_i(w)$ <u>here</u> points towards w^*

"Region of confusion": <u>some</u> $\nabla f_i(w)$ point towards w^* and <u>some don't</u>

Every $\nabla f_i(w)$ <u>here</u> points towards w^*

Variance of the Random Gradients

- The "confusion" is captured by a kind of variance of the gradients:

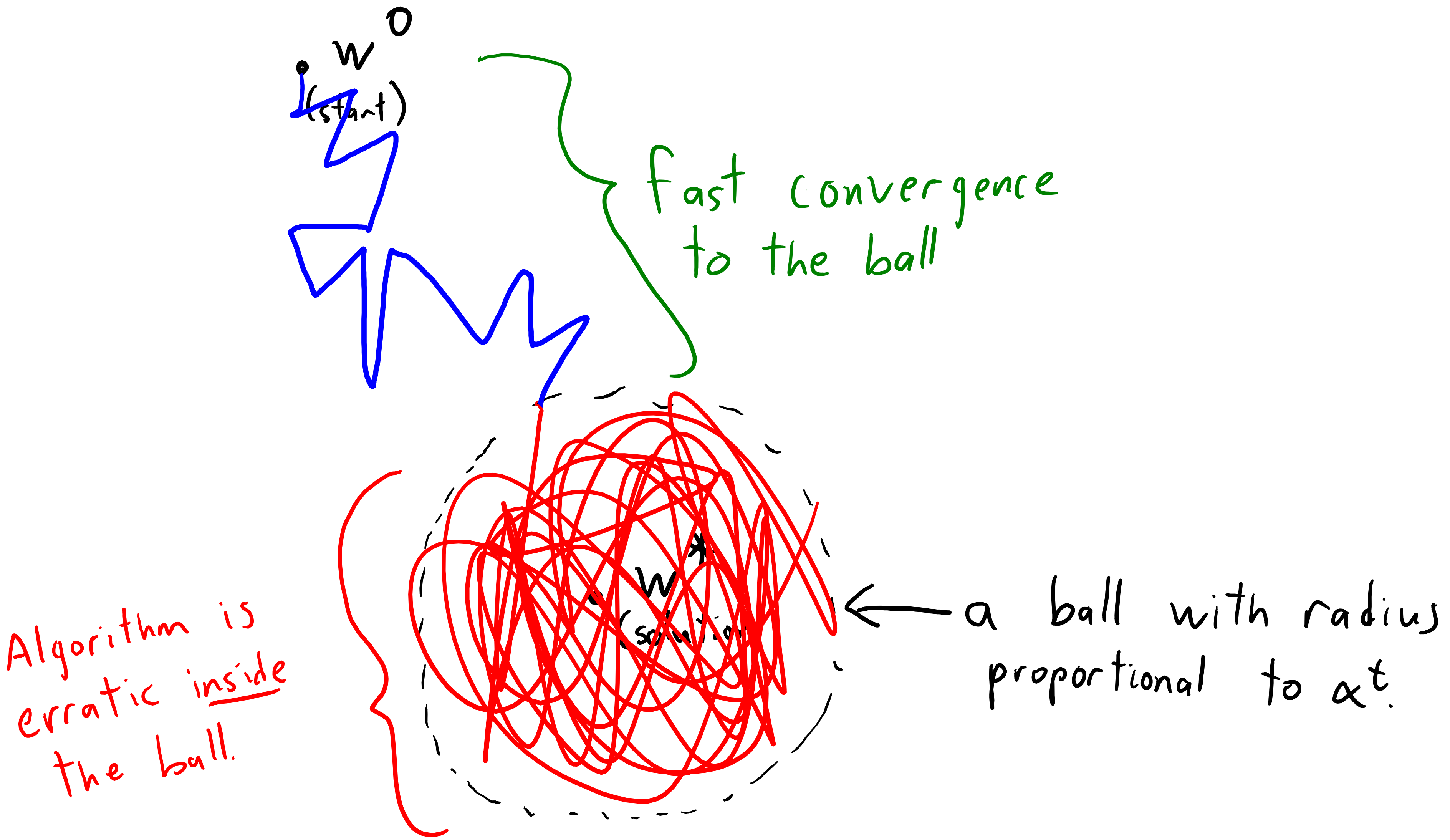
$$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(w^t) - \nabla f(w^t)\|^2$$

- If the variance is 0, every step goes in the right direction.
  - We're outside of region of confusion.
- If the variance is small, most steps point in the direction.
  - We're just inside region of confusion.
- If the variance is large, many steps will point in the wrong direction.
  - Middle of region of confusion, where w^* lives.

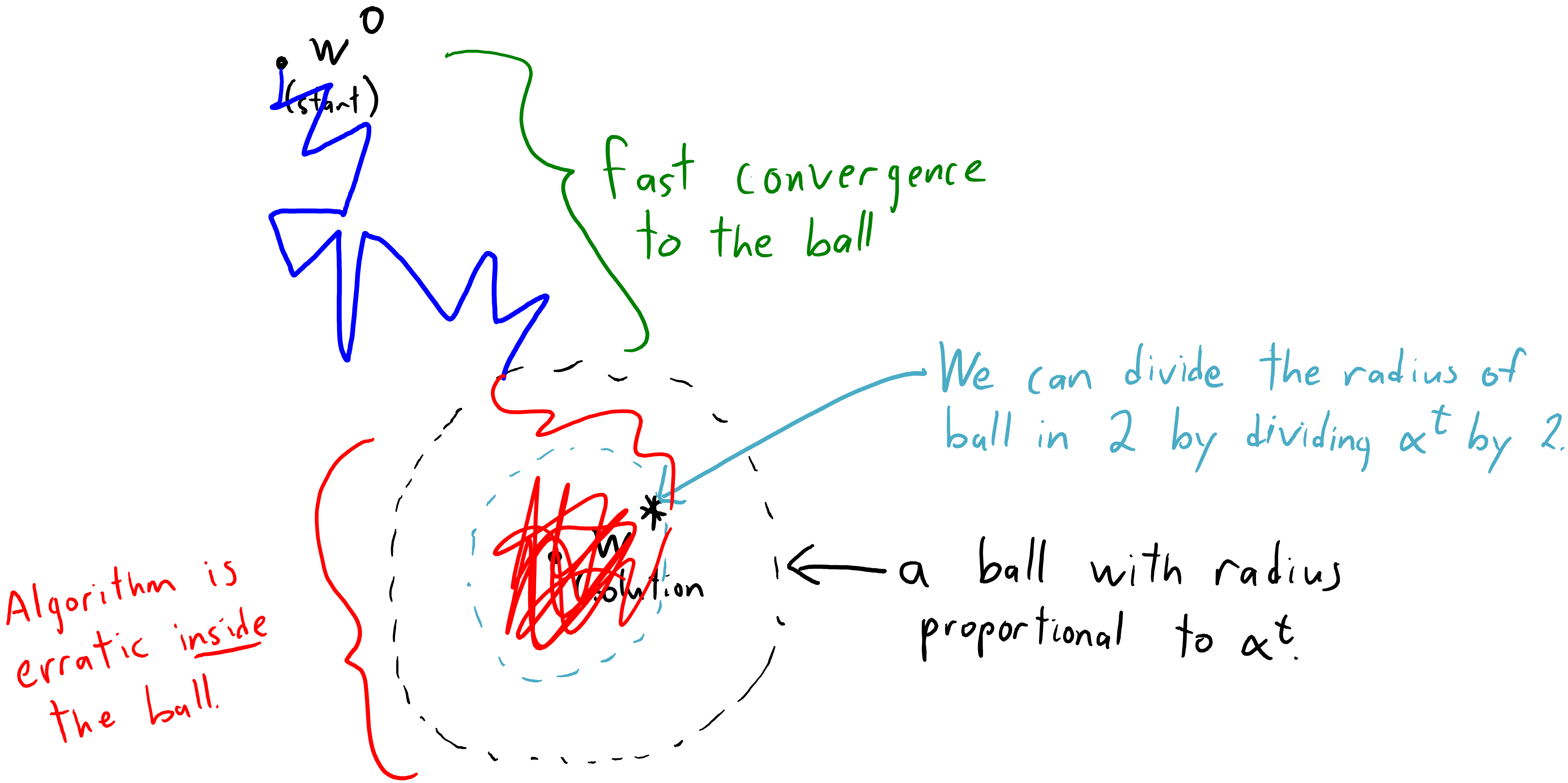
Effect of the Step-Size

- We can control the variance with the step size:
  - Variance slows progress by amount proportional to square of step-size.
- For a fixed step-size, SG makes progress until variance is too big.
- This leads to 2 phases when we use a constant step-size:
  1. Rapid progress when we are far from the solution.
  2. Erratic behaviour within a “ball” around solutions.
     (Radius of ball is proportional to the step-size.)

Stochastic Gradient with Constant Step Size



Stochastic Gradient with Constant Step Size



Stochastic Gradient with Decreasing Step Sizes

- To get convergence, we need a decreasing step size.
  - Shrinks size of ball to zero so we converge to w^*.
- But it can't shrink too quickly:
  - Otherwise, we don't move fast enough to reach ball.
- Classic solution to this problem is set step-sizes α^t so that:

$$\underbrace{\sum_{t=1}^{\infty} \alpha^t = \infty}_{\text{"we can get anywhere"}} \qquad \underbrace{\sum_{t=1}^{\infty} (\alpha^t)^2 < \infty}_{\text{"effect of variance goes to zero"}}$$

- We can achieve this by using sure $\alpha^t = O(1/t)$.

Stochastic Gradient Methods in Practice

- Unfortunately, setting $\alpha^t = O(1/t)$ works badly in practice:
  - Initial steps can be very large.
  - Later steps get very tiny.
- Practical tricks:
  - Some authors propose add extra parameters like $\alpha^t = \beta/(t + \gamma)$.
  - Theory and practice support using steps that go to zero more slowly:

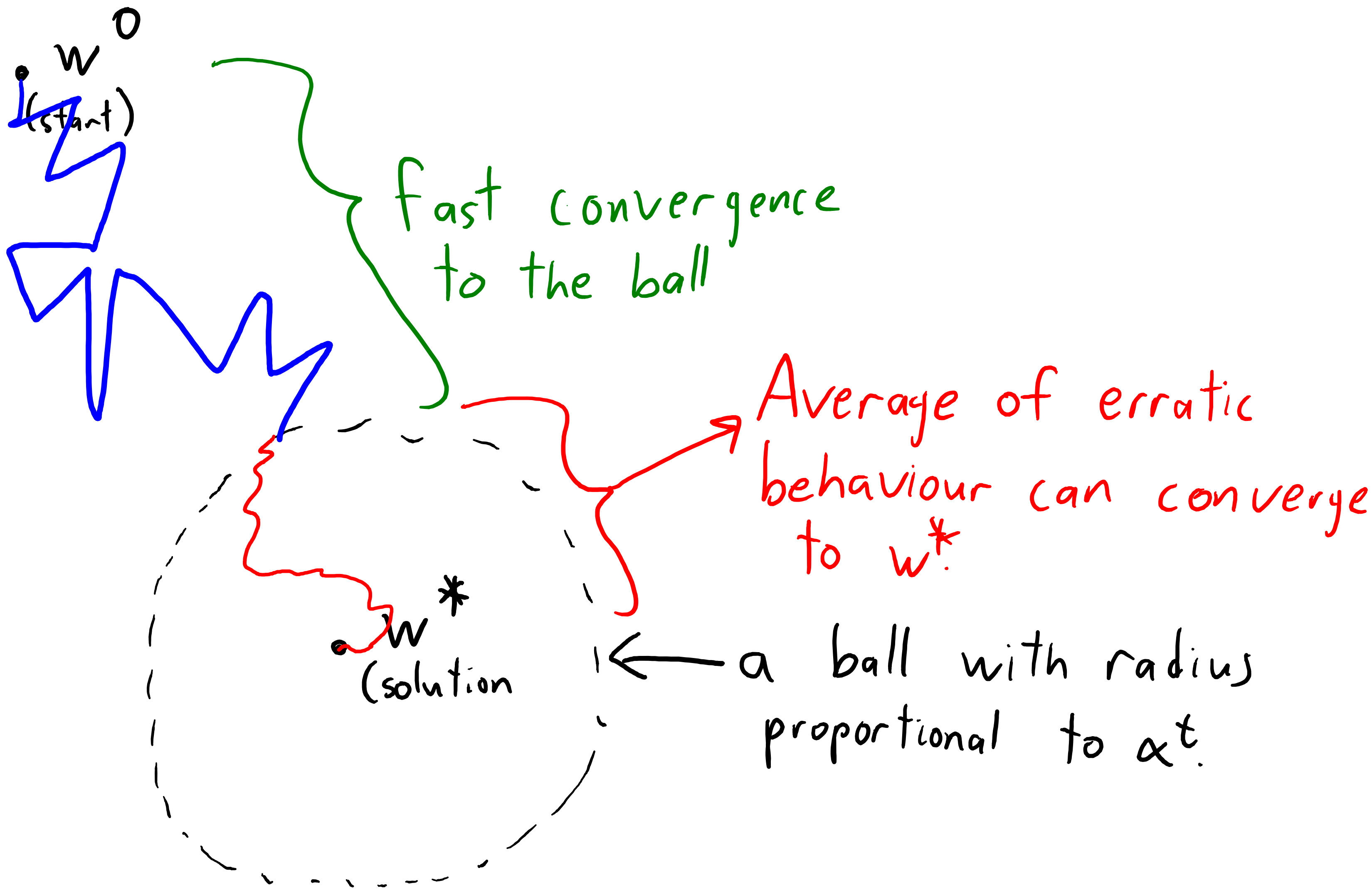
$$\alpha^t = O(1/\sqrt{t}) \quad \text{or} \quad \alpha^t = O(1) \quad (\text{constant})$$

    - But using a weighted average of the iterations:

$$\bar{w}^t = \sum_{k=1} \beta^k w^k$$

"weight" of iteration 'k'

Stochastic Gradient with Averaging



Often, you average the second half of the iterations.

A Practical Strategy For Choosing the Step-Size

- All these step-sizes have a constant factor in the "O" notation.
  - E.g., $\alpha^t = \frac{\gamma}{\sqrt{t}}$ ← How do we choose this constant?

- We don't know how to do line-searches in the stochastic case.
  - And choosing wrong γ can destroy performance.

- Common practical trick:
  - Take a small amount of data (maybe 5% of the original data).
  - Do a binary search for γ that most improves objective on this subset.

A Practical Strategy for Deciding When to Stop

- In gradient descent, we can stop when gradient is close to zero.
- In stochastic gradient:
  - Individual gradients don't necessarily go to zero.
  - We can't see full gradient, so we don't know when to stop.

- Practical trick:
  - Every 'k' iterations (for some large 'k'), measure validation set error.
  - Stop if the validation set error isn't improving.

More Practical Issues

- Does it make sense to use 2 random examples?
  - Yes, you can use a "mini-batch" of examples.

$$w^{t+1} = w^t - \alpha^t \frac{1}{|B^t|} \sum_{i \in B^t} \nabla f_i(w^t)$$

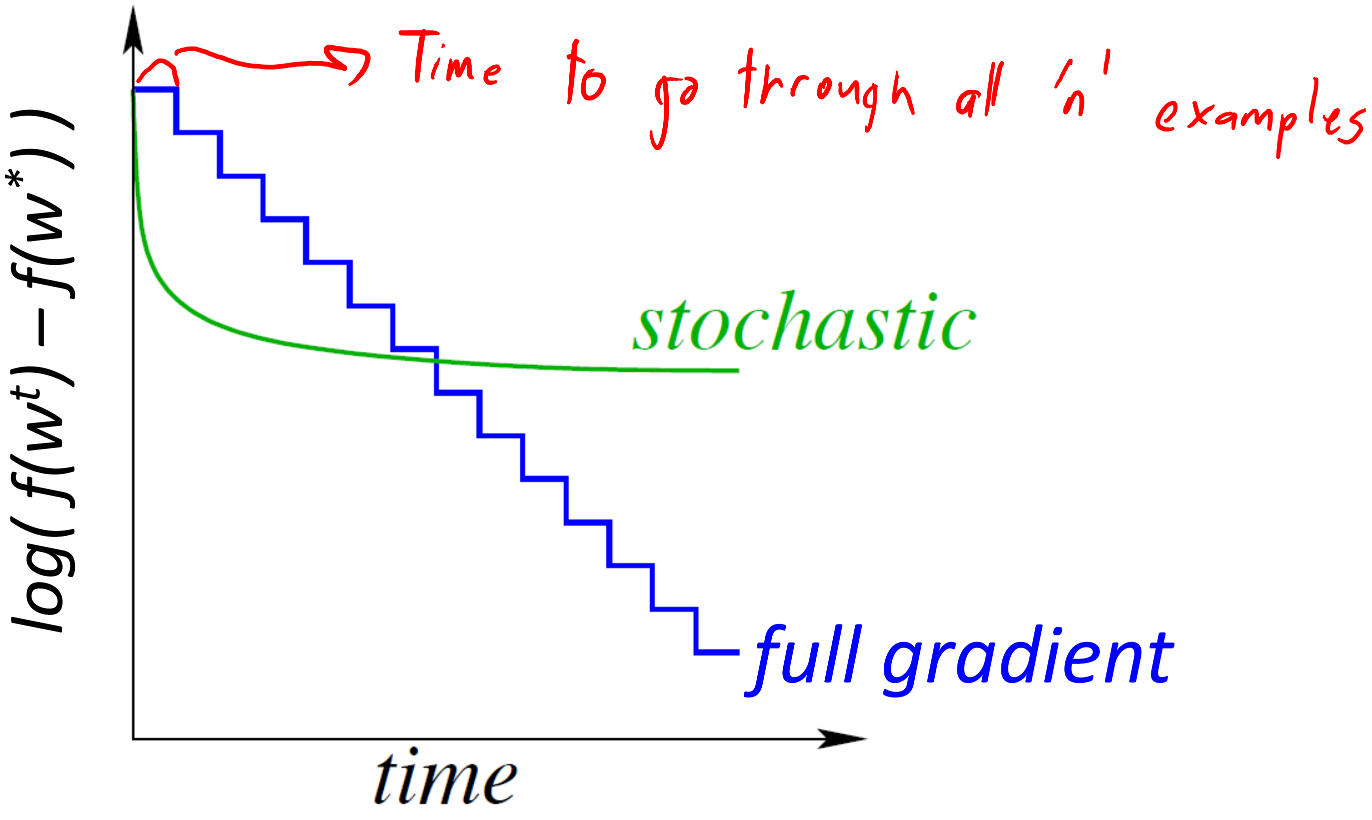
Random "batch" of examples.

  - The variance is inversely proportional to the mini-batch size.
    - You can use a bigger step size.
    - Big gains for going from 1 to 2, less big gains from going from 100 to 101.
  - Useful for vectorizing/parallelizing code.
- Can we use regularization?

If $f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w) + \frac{\lambda}{2}\|w\|^2$

then SG update is $w^{t+1} = w^t - \alpha^t(\nabla f_i(w^t) + \lambda w^t)$

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 regularized model on this fixed dataset.
- 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.

Stochastic Gradient with Infinite Data

- Approach 2 only looks at data point once:
  - Each example is an unbiased approximation of test data.

- So Approach 2 is doing stochastic gradient on test error:
  - It cannot overfit.

- Up to a constant, Approach 2 achieves test error of Approach 1.
  - This is sometimes used to justify SG as the "ultimate" learning algorithm.
  - In practice, Approach 1 usually gives lower test error (we don't know why).

Summary

- Stochastic gradient methods let us use huge datasets.
- Step-size in stochastic gradient is a huge pain:
  - Needs to go to zero to get convergence, but this works badly.
  - Constant step-size works well, but only up to a certain point.
- SAG and other newer methods fix convergence for finite datasets.
- Infinite datasets can be used with SG and do not overfit.

- Next time:
  - Feature selection?