CPSC 340:
Machine Learning and Data Mining

Stochastic Gradient

Fall 2018

Admin

- Assignment 4:
  - Due Friday.
- Midterm:
  - Can view exam during Mike or my office hours this week.
- 532M Projects:
  - “No news is good news”.
- Kernel trick for Fourier transform:
  - Not needed, as Fourier transform has same size as input.
  - But, sampling Fourier basis is common to reduce cost of using kernel trick.
    - Especially elegant for Gaussian RBFs (won “test of time award” last year at NIPS).

Motivation: 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.

- But 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:

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

(scalar; $d \times 1$)

- So the cost of computing this 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 the gradient of a randomly-chosen training example:

$$\nabla f_i(w) = (\underbrace{w^T x_i - y_i}_{\text{scalar}})\underbrace{x_i}_{d \times 1}$$

- Cost of computing this one gradient is independent of 'n'.
  - Iterations are 'n' times faster than gradient descent iterations.
  - With 1 billion training examples, this iteration is 1 billion times faster.

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'.

Problem where we can use Stochastic Gradient

- Stochastic gradient applies when minimizing averages:

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

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

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}\left[\log(1+\exp(-y_i w^T x_i)) + \frac{\lambda}{2}\|w\|^2\right] \quad (L_2\text{-regularized logistic})$$

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

- Basically, all our regression losses except "brittle" regression.
  - Recall: multiplying by positive constant doesn't change location of optimal 'w'.

Why Does Stochastic Gradient Work / Not Work?

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

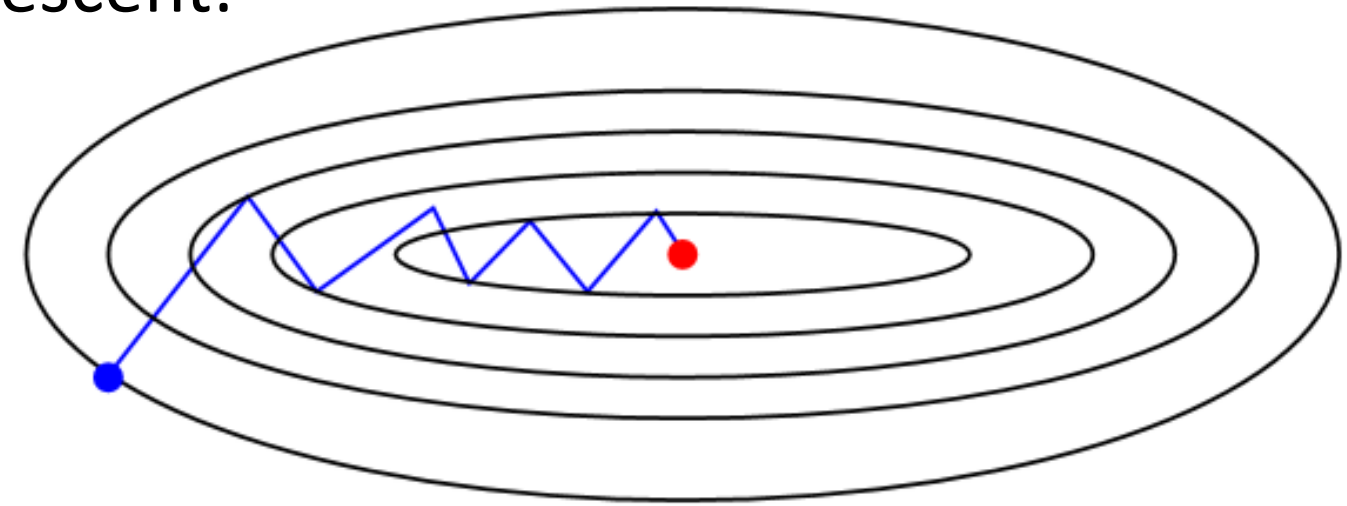
- Does this have any hope of working?
  - The average of the random gradients is the full gradient.

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

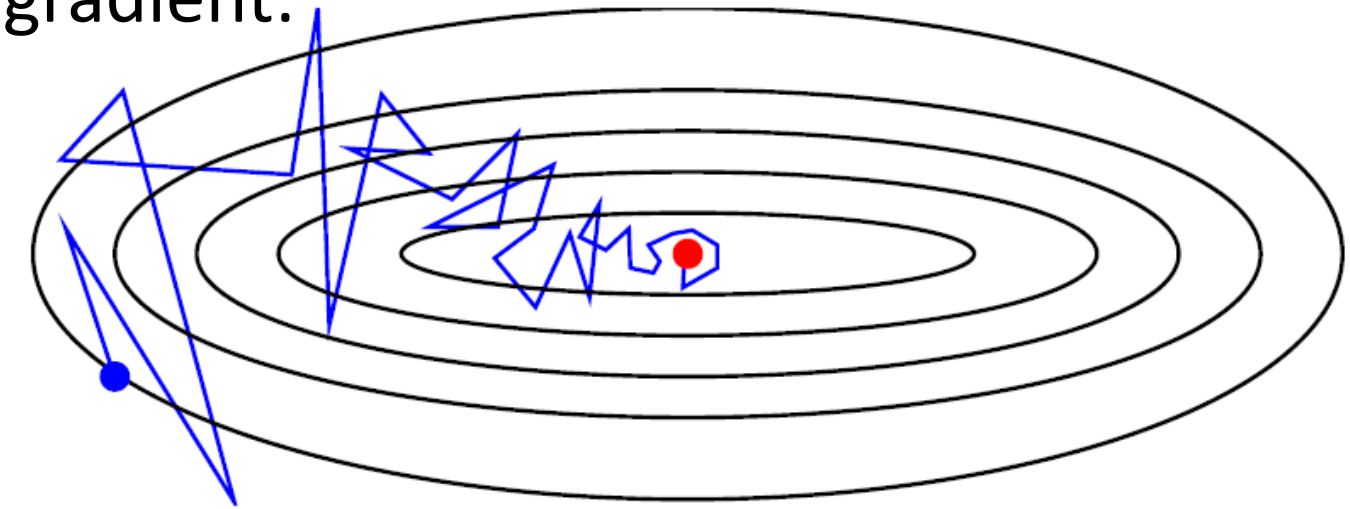
  - The algorithm is going in the right direction on average.

Gradient Descent vs. Stochastic Gradient (SG)

- Gradient descent:

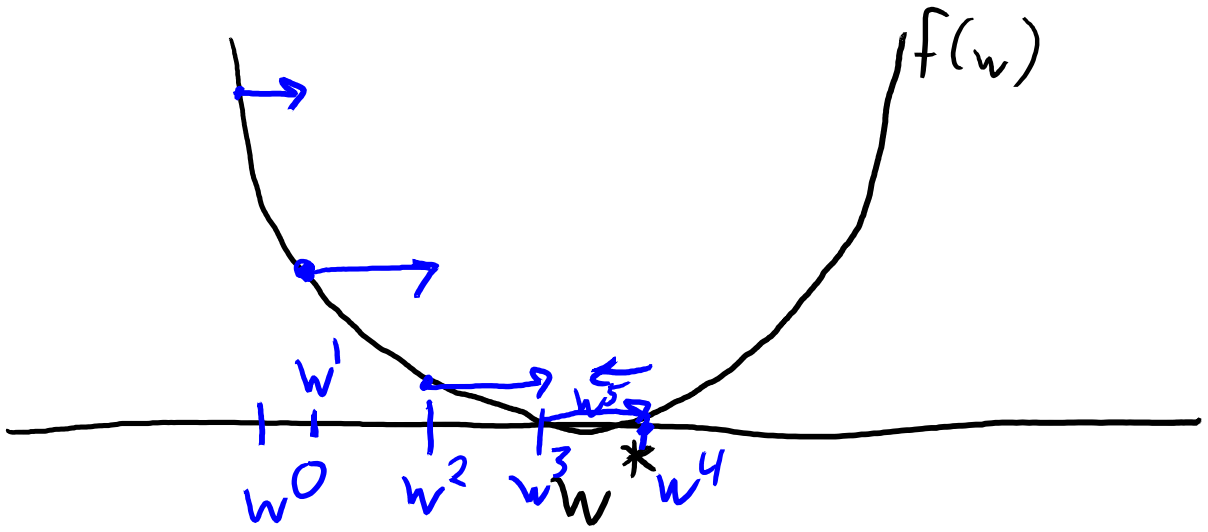


- Stochastic gradient:



Gradient Descent in Action

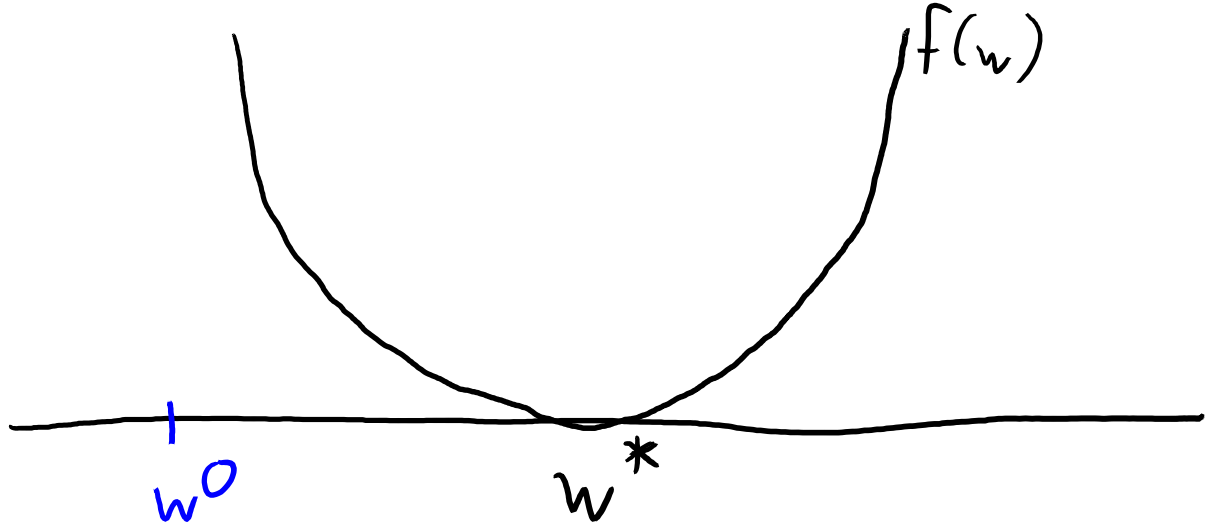
$$f(w) = \frac{1}{5}\sum_{i=1}^{5} (w^T x_i - y_i)^2$$

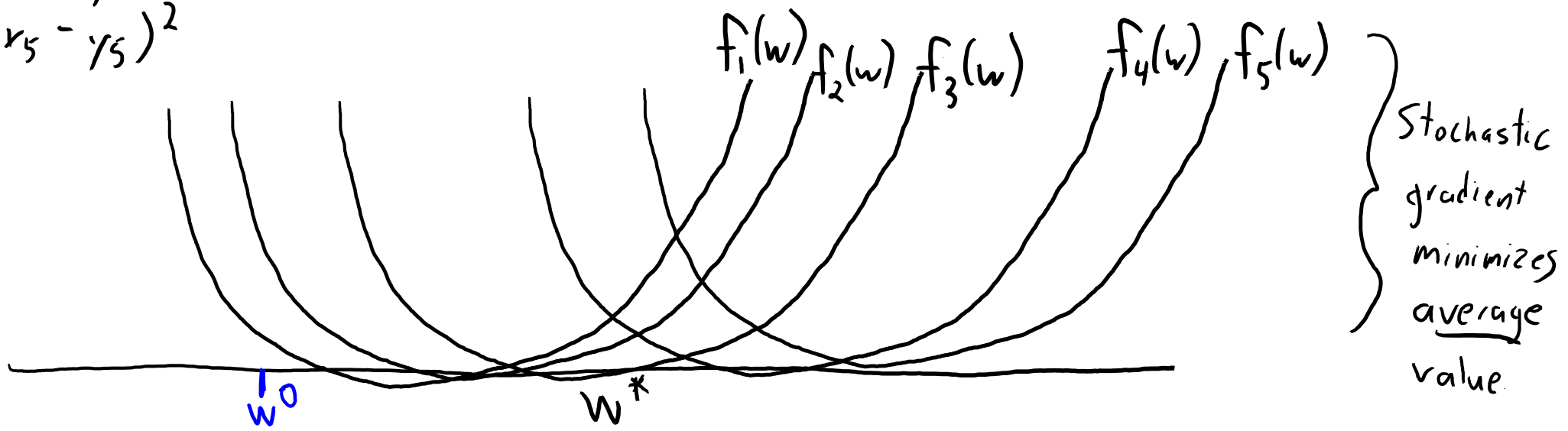


Stochastic Gradient in Action

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

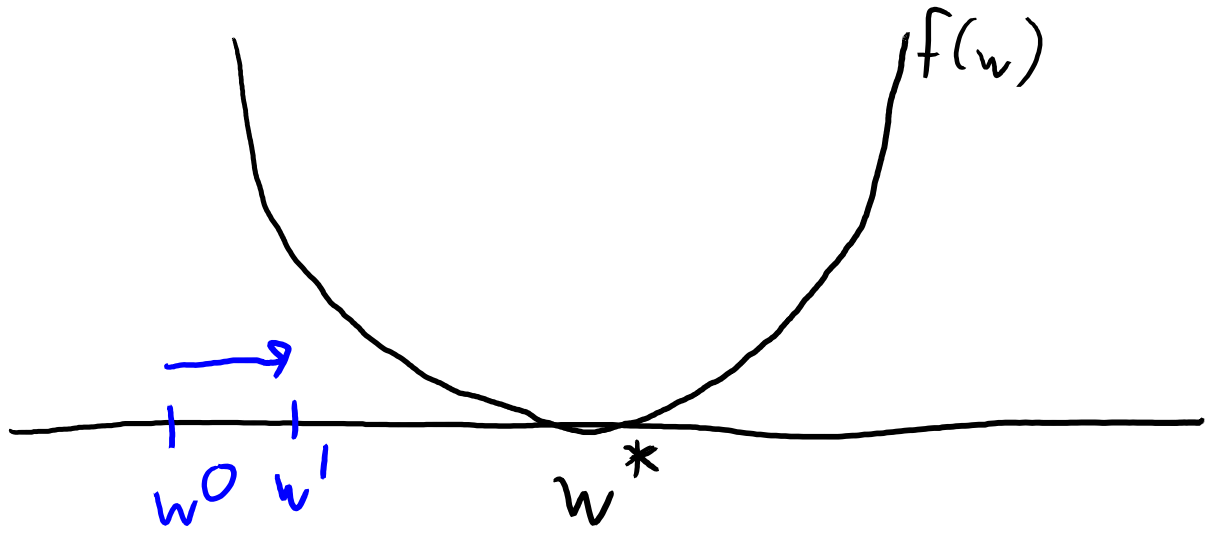
$$f_1(w) = (w^T x_1 - y_1)^2$$
$$f_2(w) = (w^T y_2 - y_2)^2$$
$$f_3(w) = (w^T x_3 - y_3)^2$$
$$f_4(w) = (w^T x_4 - y_4)^2$$
$$f_5(w) = (w^T x_5 - y_5)^2$$

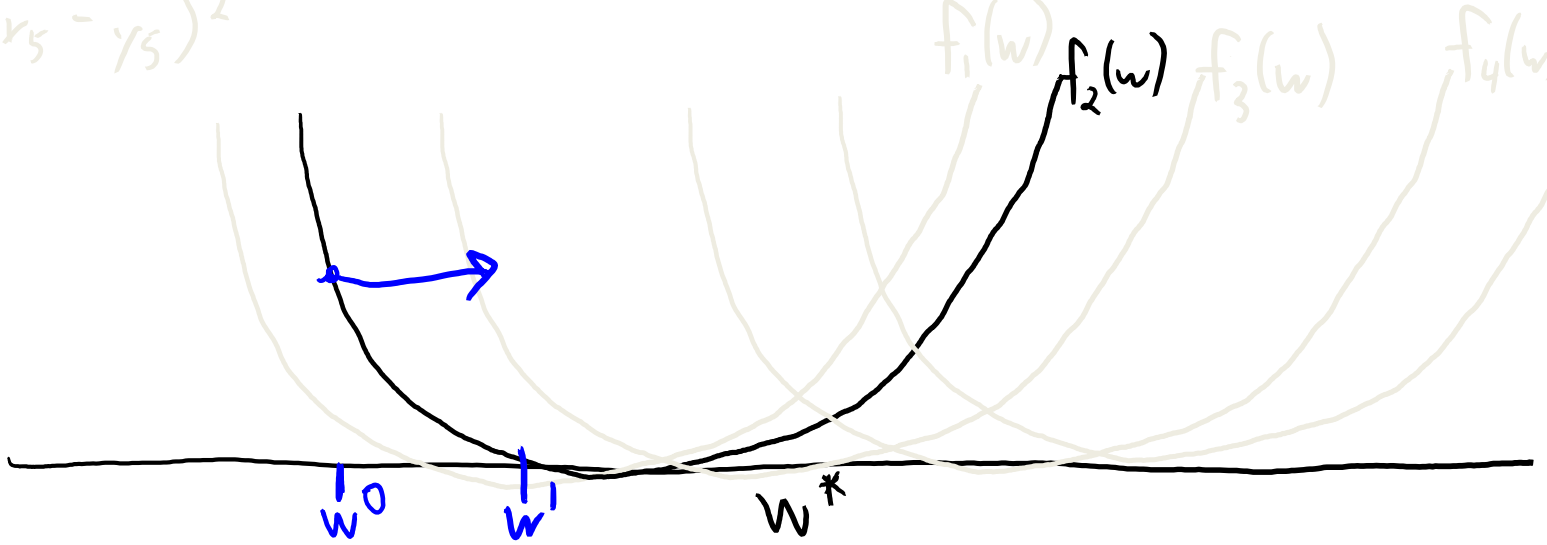




Stochastic Gradient in Action

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





Stochastic gradient minimizes average value

Stochastic Gradient in Action

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

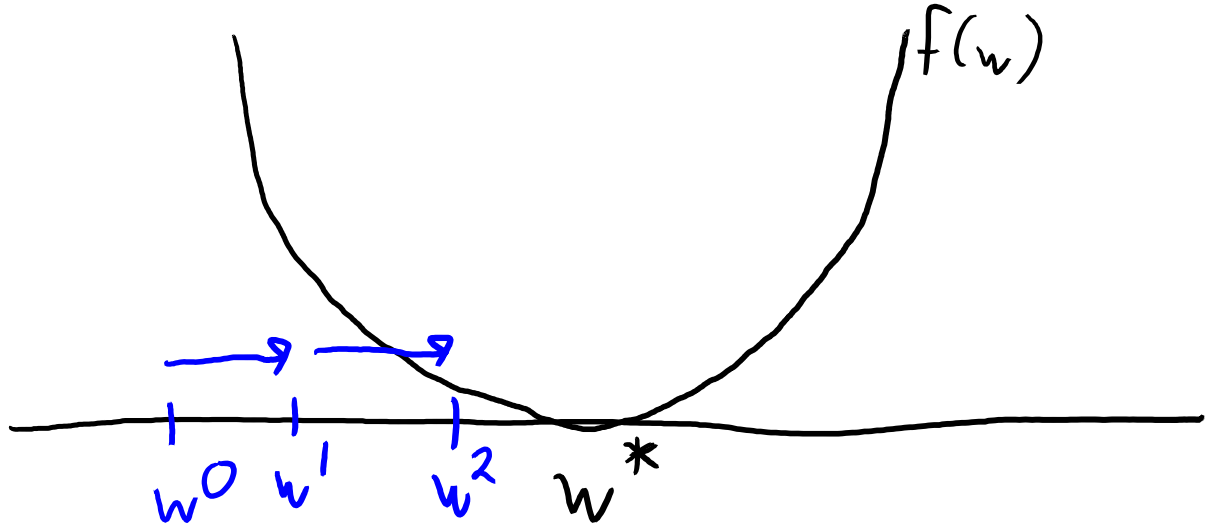
$$f_1(w) = (w^T x_1 - y_1)^2$$

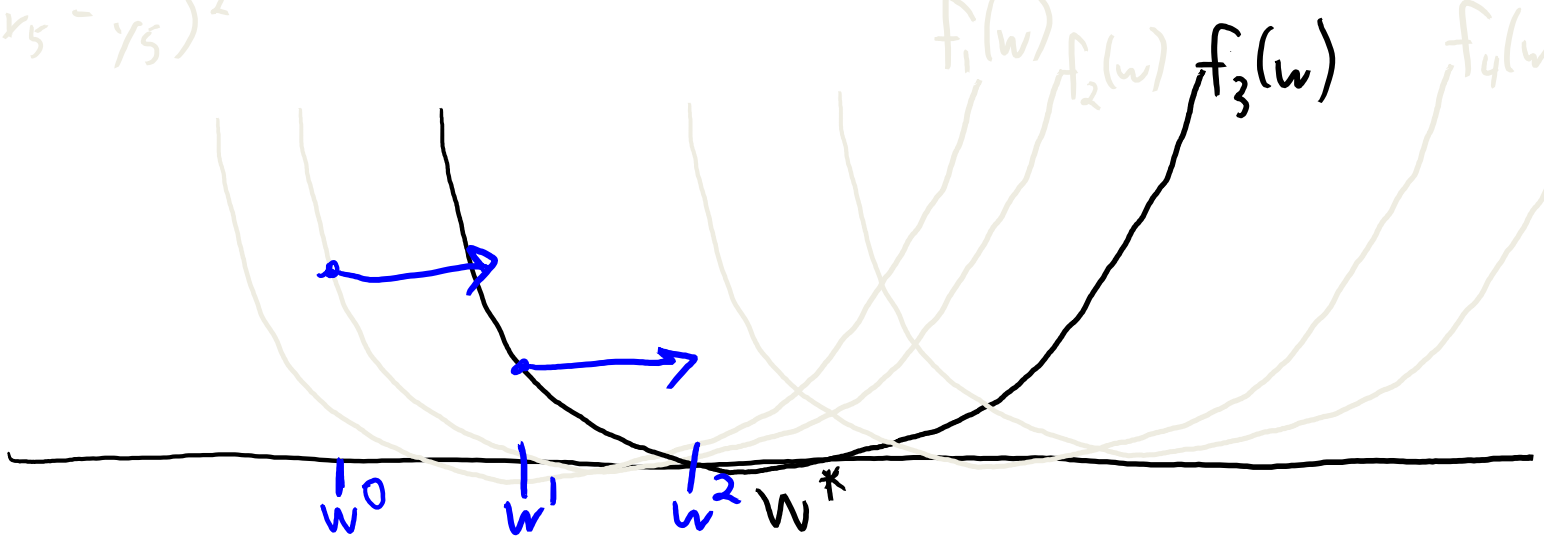
$$f_2(w) = (w^T x_2 - y_2)^2$$

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

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

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





Stochastic gradient minimizes average value

Stochastic Gradient in Action

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

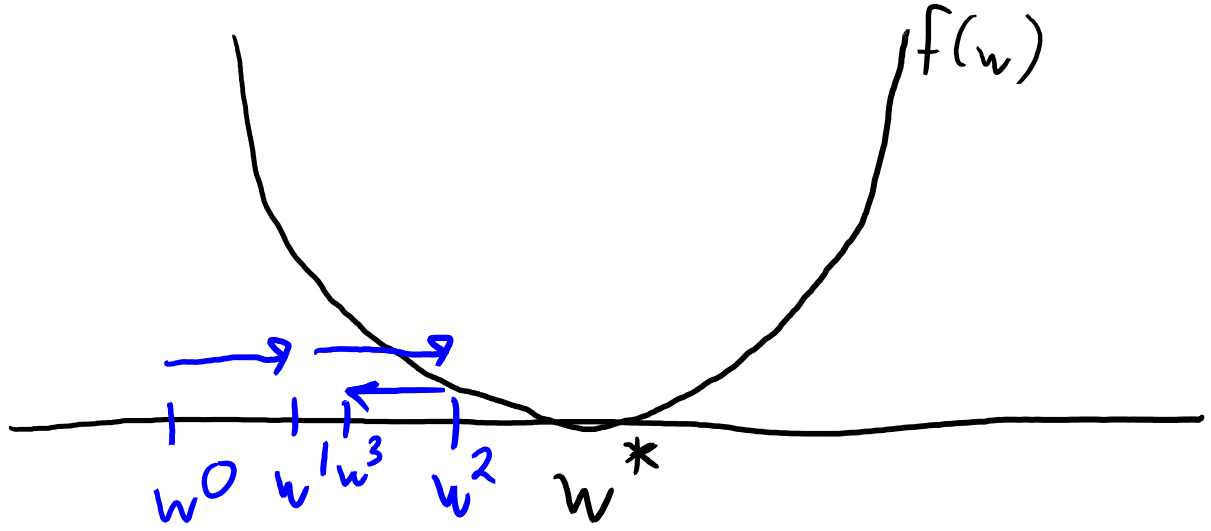
$$f_1(w) = (w^T x_1 - y_1)^2$$

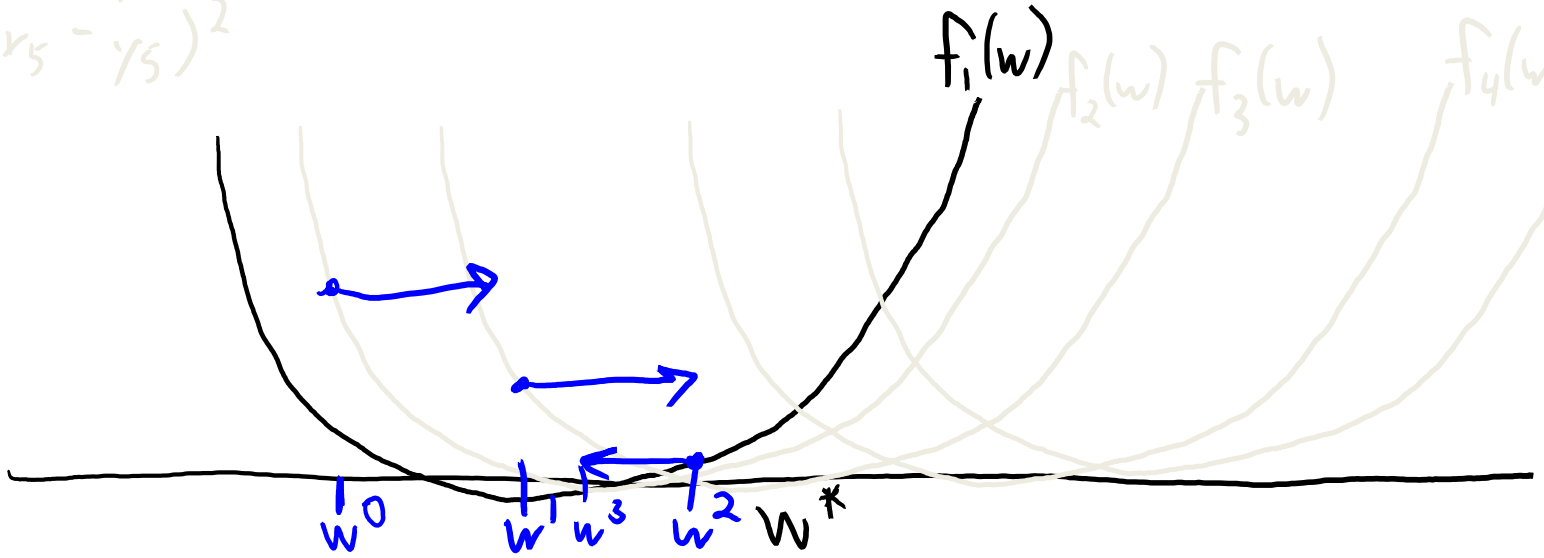
$$f_2(w) = (w^T x_2 - y_2)^2$$

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

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

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

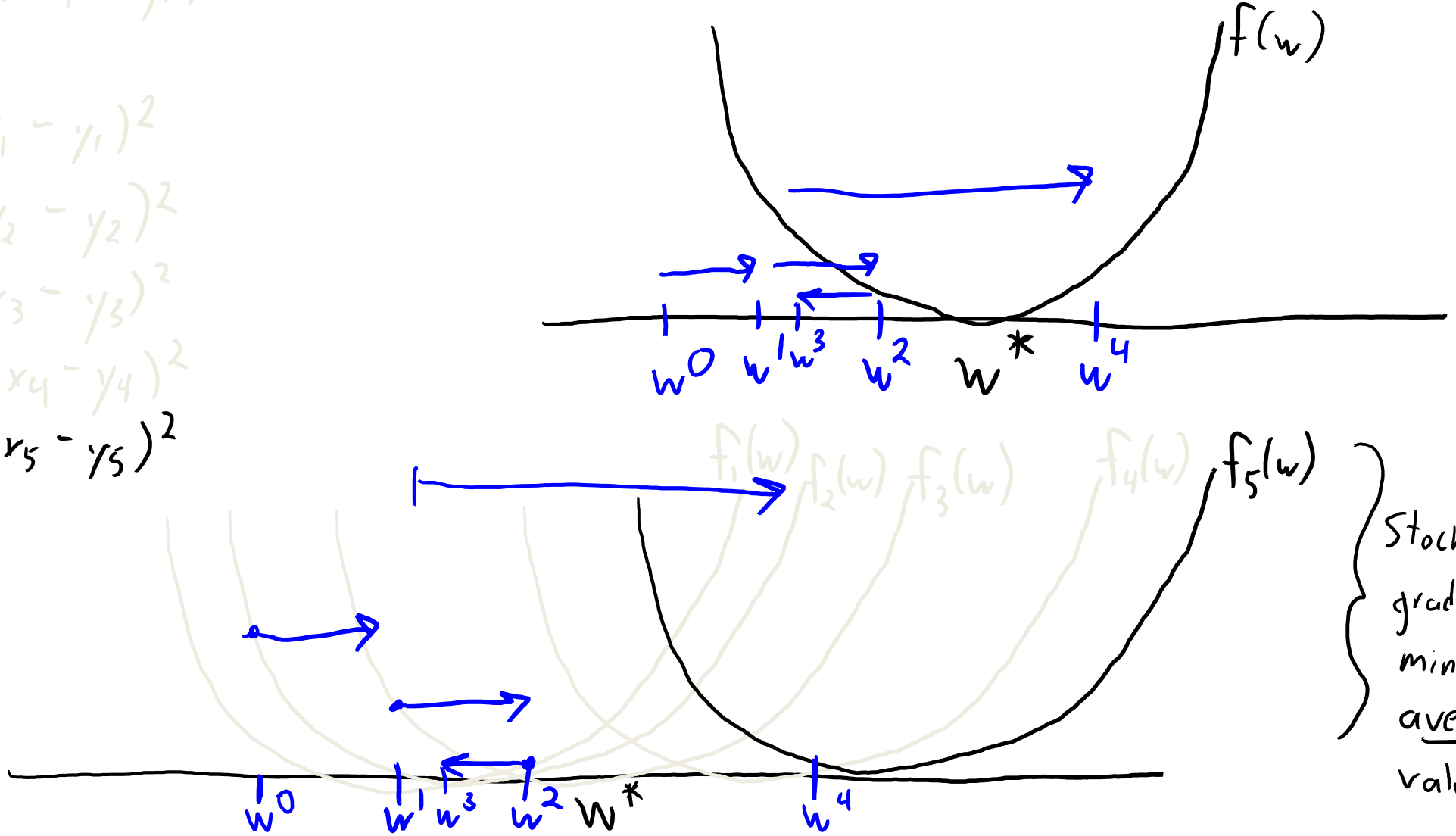




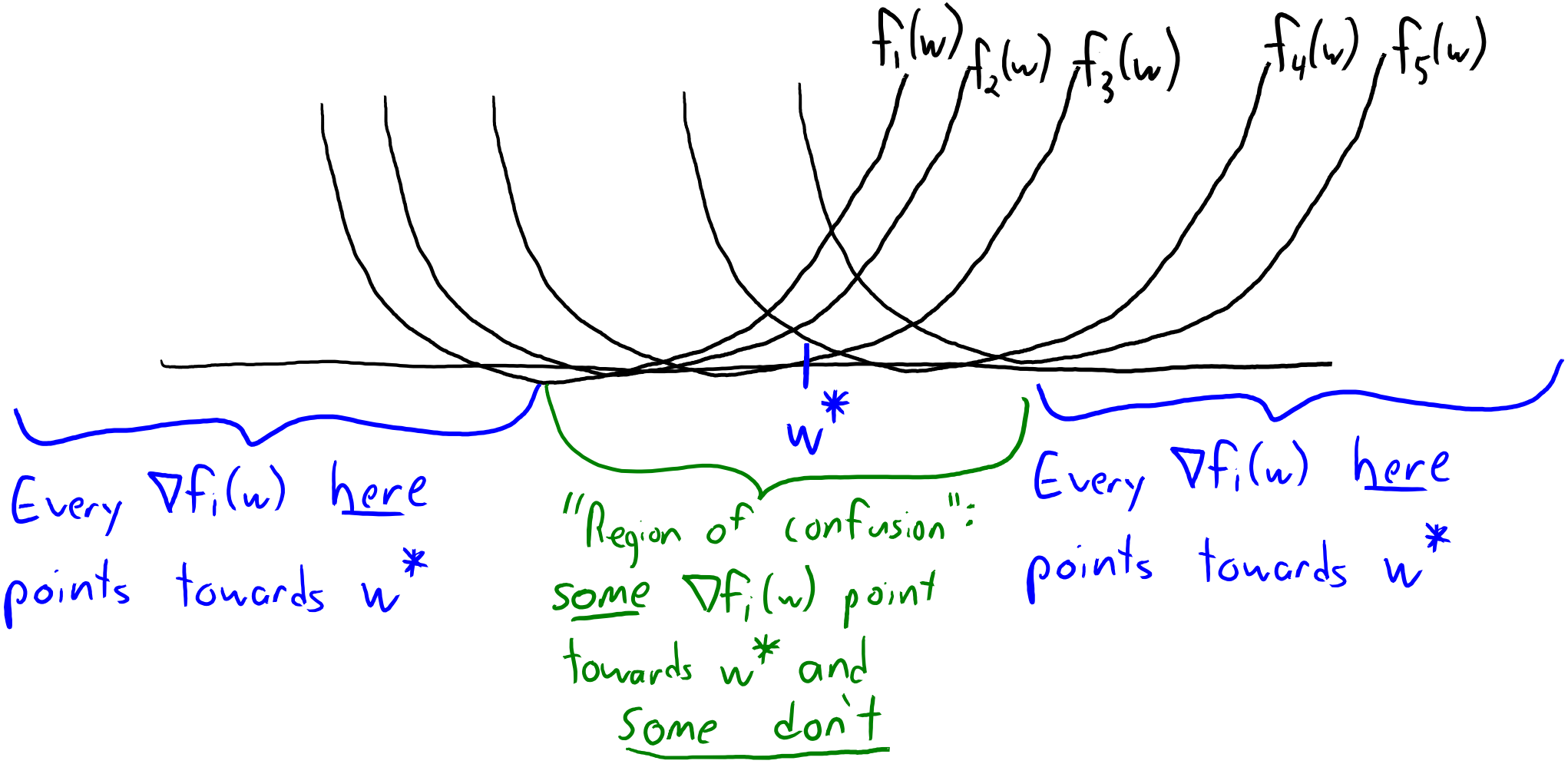
Stochastic gradient minimizes average value

Stochastic Gradient in Action

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



Effect of 'w' Location on Progress



- We'll still make good progress if most gradients points in right direction.

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 the 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.

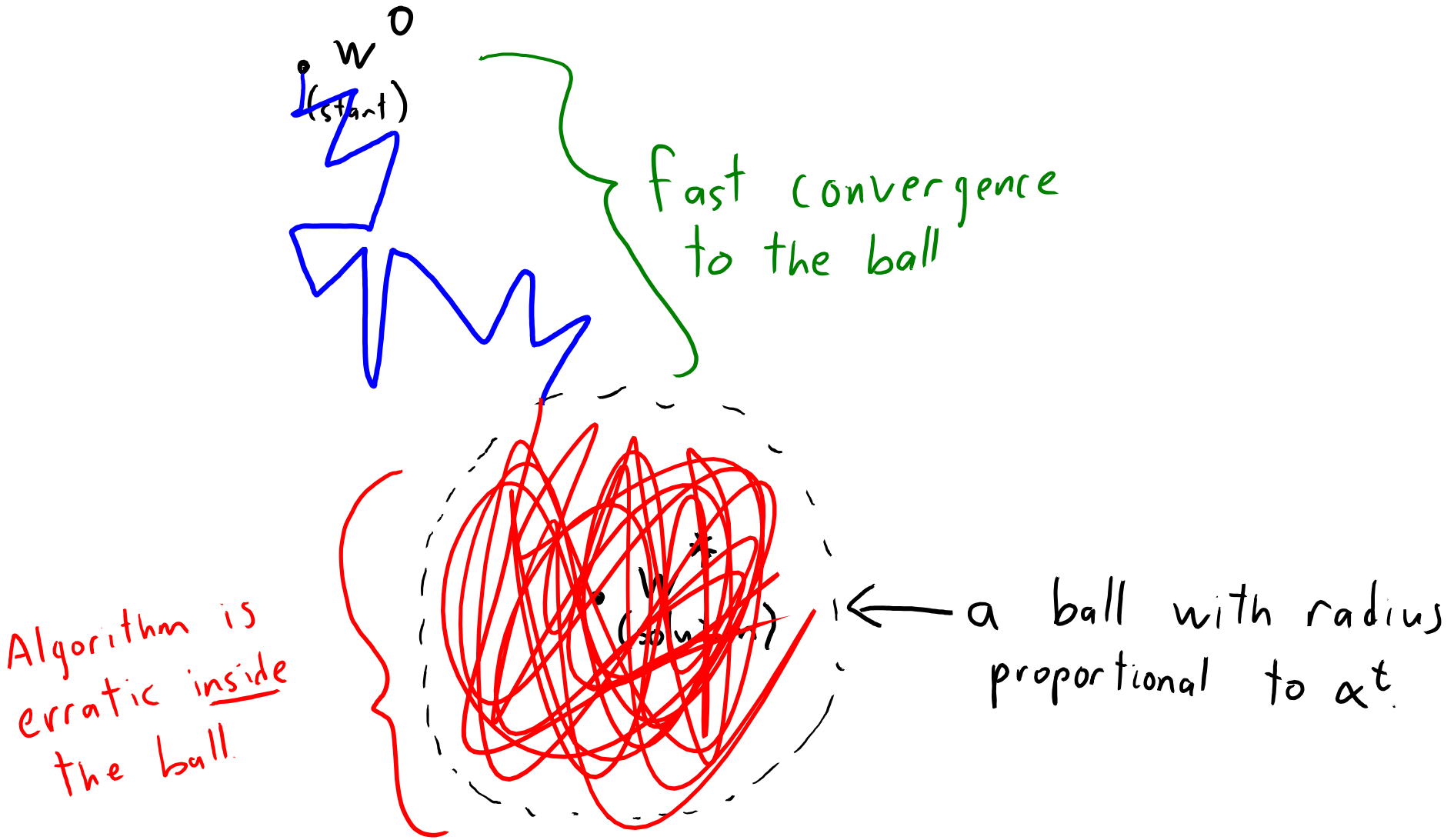
Effect of the Step-Size

- We can reduce the effect of the variance with the step size.
  - Variance slows progress by amount proportional to square of step-size.
  - So as the step size gets smaller, the variance has less of an effect.

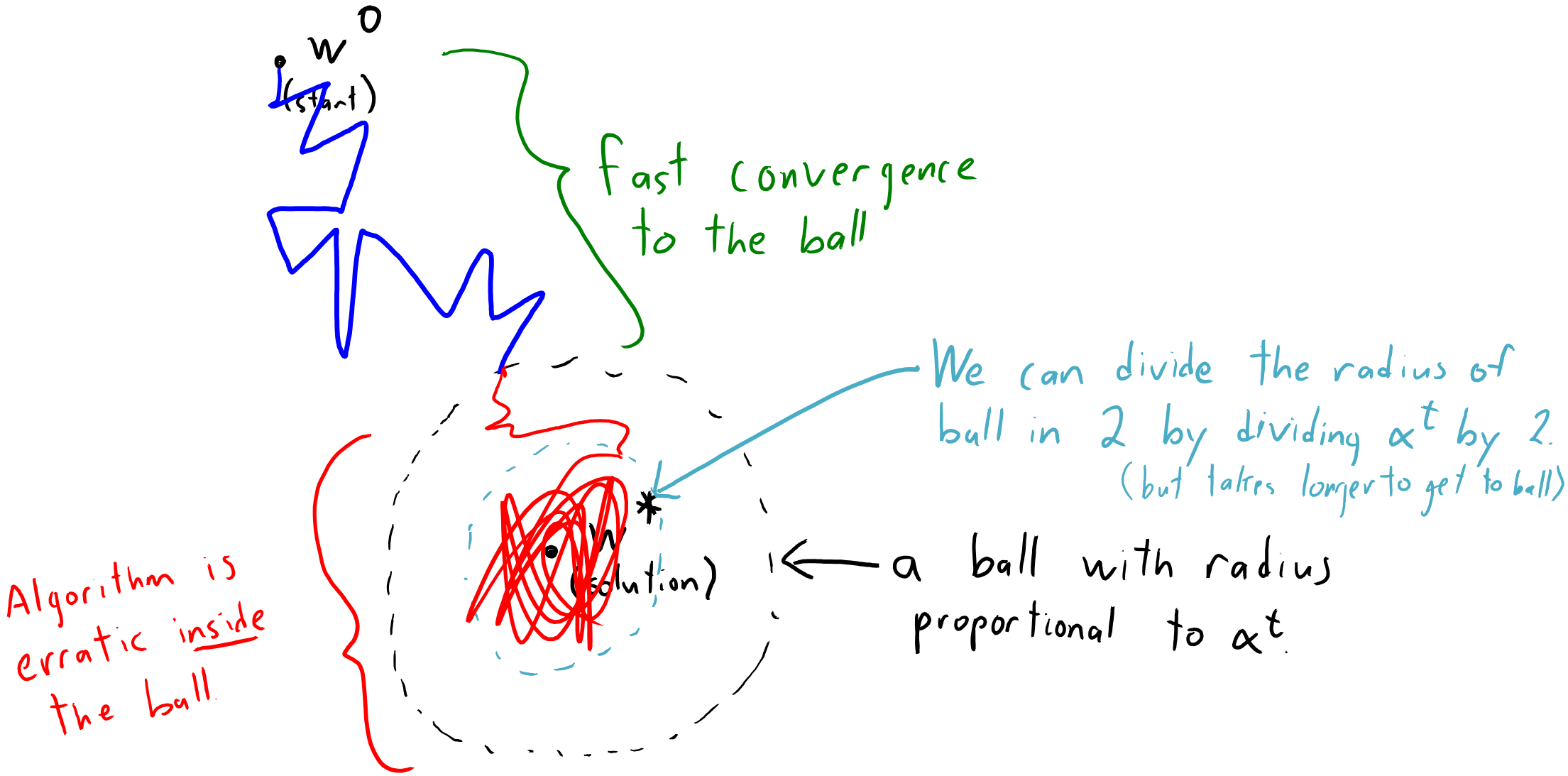
- For a fixed step-size, SG makes progress until variance is too big.

- This leads to two "phases" when we use a constant step-size:
  1. Rapid progress when we are far from the solution.
  2. Erratic behaviour confined to a "ball" around solution.
     (Radius of ball is proportional to the step-size.)

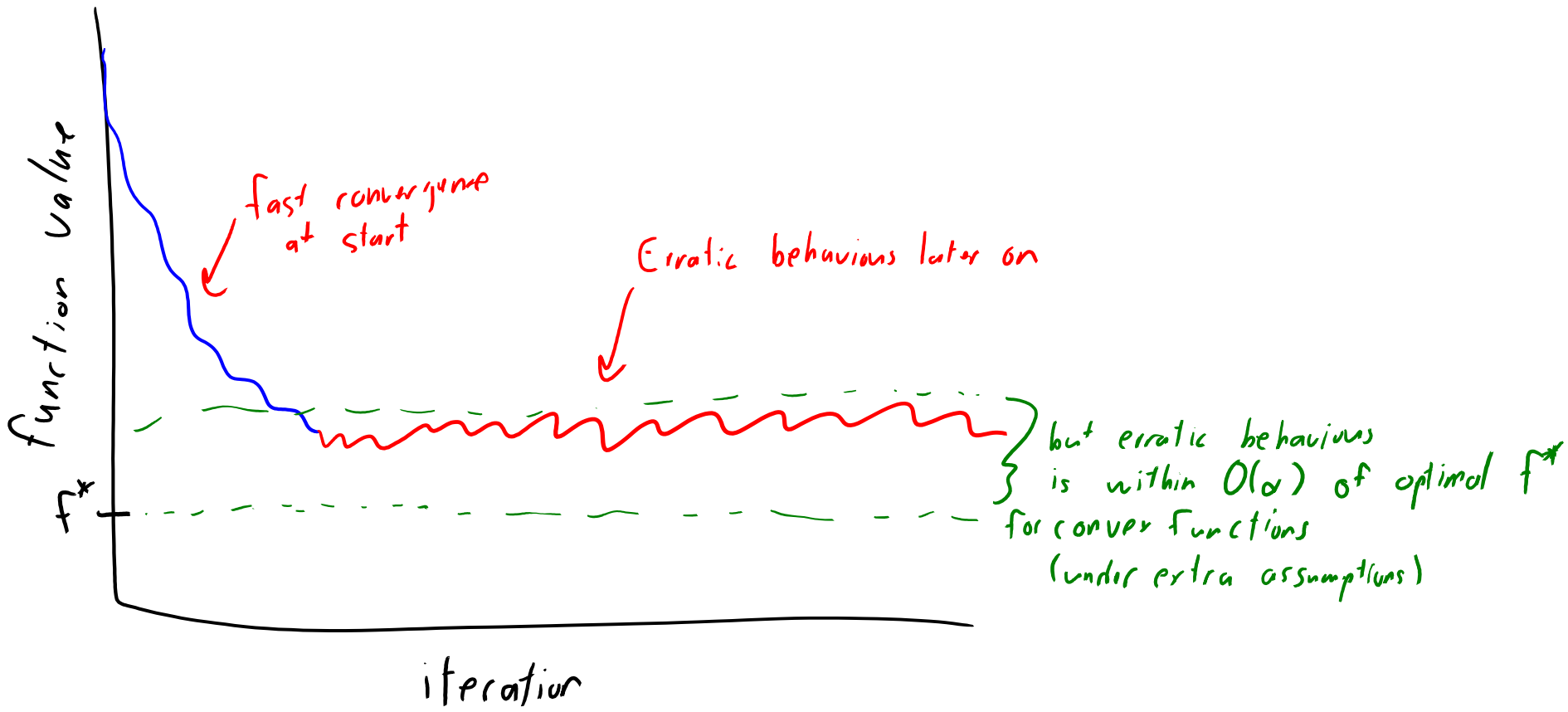
Stochastic Gradient with Constant Step Size



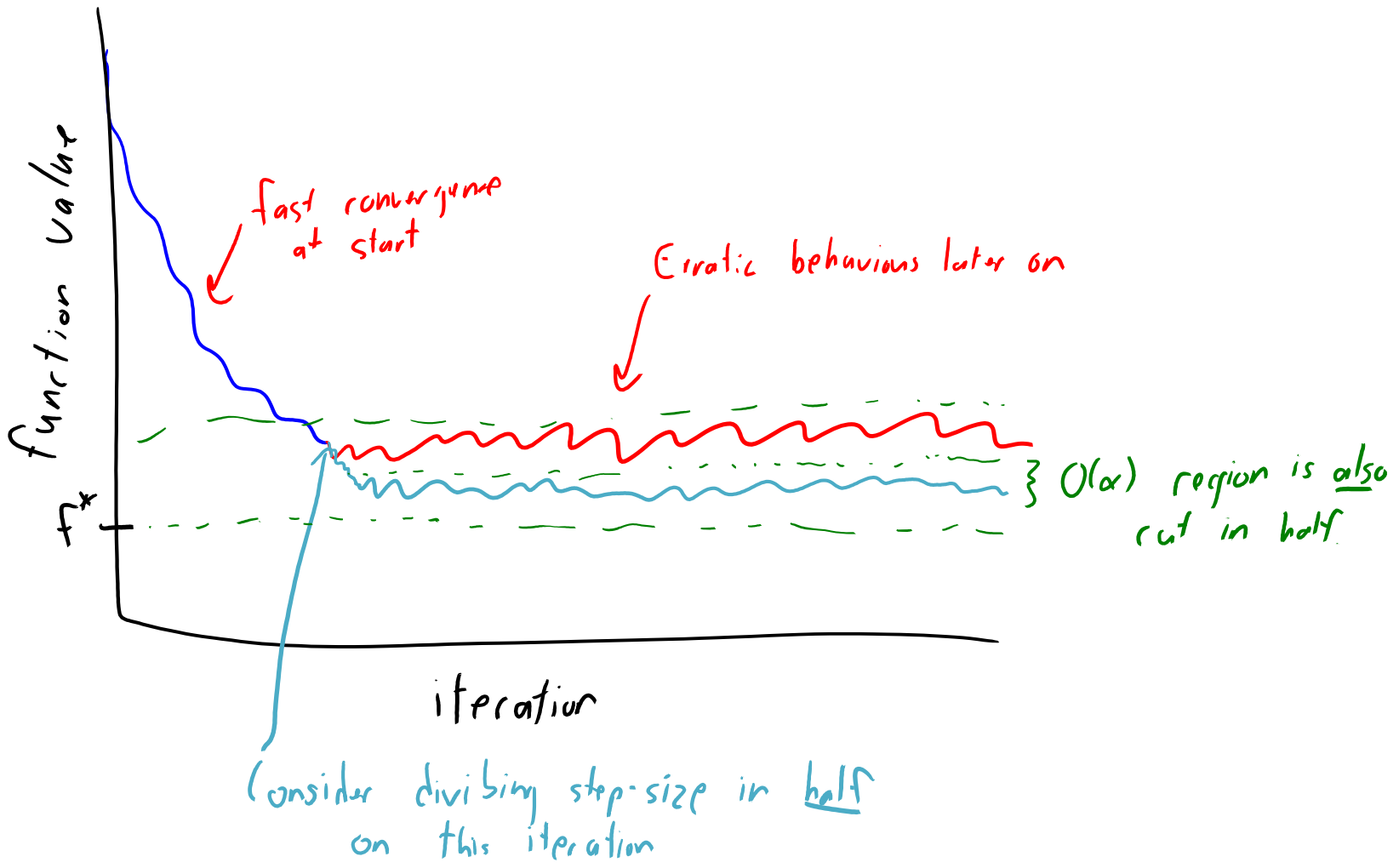
Stochastic Gradient with Constant Step Size



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 the ball.

- Stochastic gradient converges to a stationary point if:
  - Ratio of sum of squared step-sizes over sum of step-sizes converges to 0.

"how much noise affects you" $\rightarrow \sum_{t=1}^{\infty} (\alpha^t)^2$

"how far you can get" $\rightarrow \sum_{t=1}^{\infty} \alpha^t$

$$\frac{\sum_{t=1}^{\infty} (\alpha^t)^2}{\sum_{t=1}^{\infty} \alpha^t} = 0$$

Stochastic Gradient with Decreasing Step Sizes

- For convergence step-sizes need to satisfy: $\sum_{t=1}^{\infty} (\alpha^t)^2 / \sum_{t=1}^{\infty} \alpha^t = 0$

- Classic solution is to use a step-size sequence like $\alpha^t = O(1/t)$.

$$\sum_{t=1}^{\infty} \alpha^t = \sum_{t=1}^{\infty} \frac{1}{t} = \infty$$

"we can get everywhere"

$$\sum_{t=1}^{\infty} (\alpha^t)^2 = \sum_{t=1}^{\infty} \frac{1}{t^2} < \infty$$

"effect of variance goes to zero"

  - E.g., $\alpha^t = .001/t$.
- Unfortunately, this often works badly in practice:
  - Steps get really small really fast.
  - Some authors add extra parameters like $\alpha^t = \gamma/(t + \Delta)$, which helps a bit.
  - One of the only cases where this works well: binary SVMs with $\alpha^t = 1/\lambda t$.

Stochastic Gradient with Decreasing Step Sizes

- How do we pick step-sizes satisfying $\sum_{t=1}^{\infty} (\alpha^t)^2 / \sum_{t=1}^{\infty} \alpha^t = 0$

- Better solution is to use a step-size sequence like $\alpha^t = O(1/\sqrt{t})$.

$$\sum_{t=1}^{\infty} \alpha^t = \sum_{t=1}^{\infty} \frac{1}{\sqrt{t}} = O(\sqrt{t}) \qquad \sum_{t=1}^{\infty} (\alpha^t)^2 = \sum_{t=1}^{\infty} \frac{1}{t} = O(\log t)$$

  - E.g., use $\alpha^t = .001/\sqrt{t}$
  - Both sequences diverge, but denominator diverges faster.

- This approach (roughly) optimizes rate that it goes to zero.
  - Better worst-case theoretical properties (and more robust to step-size).
  - Often better in practice too.

Stochastic Gradient with Constant Step Sizes?

- Alternately, could we just use a constant step-size.
  - E.g., use $\alpha^t = .001$ for all 't'.
- This will not converge to a stationary point in general.
  - However, do we need it to converge?
- What if you only care about the first 2-3 digits of the test error?
  - Who cares if you aren't able to get 10 digits of optimization accuracy?
- There is a step-size small enough to achieve any fixed accuracy.
  - Just need radius of "ball" to be small enough.

Mini-batches: Using more than 1 example

- Does it make sense to use more than 1 random example?
  - Yes, you can use a "mini-batch" B^t 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

  - Radius of ball is inversely proportional to the mini-batch size.
    - If you double the batch size, you half the radius of the ball.
      - Big gains for going from 1 to 2, less big gains from going from 100 to 101.
    - You can use a bigger step size as the batch size increases.
      - Gets you to the ball faster (though diverges if step-size is too big).
  - Useful for vectorizing/parallelizing code.
    - Evaluate one gradient on each core.

Polyak-Ruppert Iterate Averaging

- Another practical/theoretical trick is averaging of the iterations.
  1. Run the stochastic gradient algorithm with $\alpha^t = O(1/\sqrt{t})$ or α^t constant.
  2. Take some weighted average of the w^t values.

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

where v^k is a scalar "weight" of iteration 'k'

Uniform average: $v^k = \frac{1}{n}$

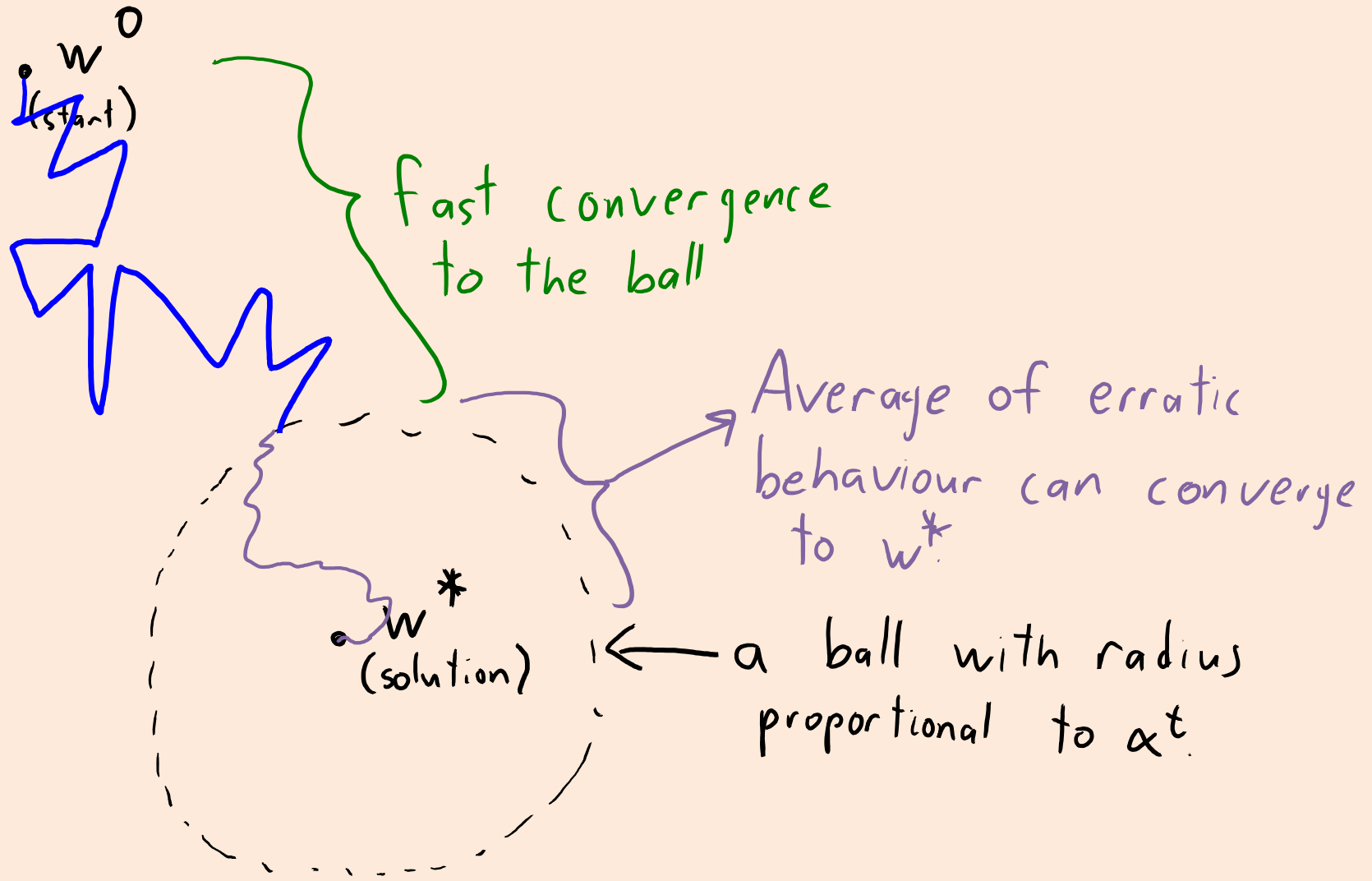
$$\bar{w}^t = \frac{1}{n} \sum_{k=1}^{t} w^k$$

- Average does not affect the algorithm, it's just "watching".
- Surprising result shown by Polyak and by Ruppert in the 1980s:
  - Asymptotically achieves the same rate as stochastic Newton's method.

Stochastic Gradient with Averaging

Often, you average the second half of the iterations

Set $\bar{w}^t = \frac{1}{t/2} \sum_{k=t/2}^{t} w^k$



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 classic O(1/t) steps are bad.
  - $O(1/\sqrt{t})$ works better, but still pretty slow.
  - Constant step-size is fast, but only up to a certain point.

- Next time:
  - What do regression and regularization have to do with probabilities?

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 set step size as we go 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.