$$\ell(w) = \sum_{i=1}^{n} (y_i - f_w(x_i))^2$$

"least squares"

$$w^* \in \arg\min_w \ell(w)$$

$$w^* \in \arg\min_w \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{m} w_j x_{i,j}\right)^2$$

Lecture 4
- BBO
- Gradient Descent

Black box optimization (BBO): Optimization when we know little about ℓ.

- Given w, can query $\ell(w)$.
- Many BBO algorithms.
  - Genetic algs.
  - CEM
  - CMA-ES
  - * Hill-climbing / simulated annealing.
  - Finite difference methods.
    - SPSA
  - Bayesian optimization.

Hill Climbing

1) Pick initial point w_0 arbitrarily

2) Randomly select a nearby point w'

Example: $\forall_j \in \{1, \ldots, n\}$ $w_j' = w_j + \eta_j$ (eta)

$\eta_j \sim N(0, \sigma^2)$ (mean)

- η: noise
- $\sim$: is sampled from.
- N: normal distribution hyperparameters.
- σ^2: variance. (small) → hyperparameter

alternative: $\eta_j \sim U(c, -c)$ (c is a hyperparameter.)

- U: uniform distribution.

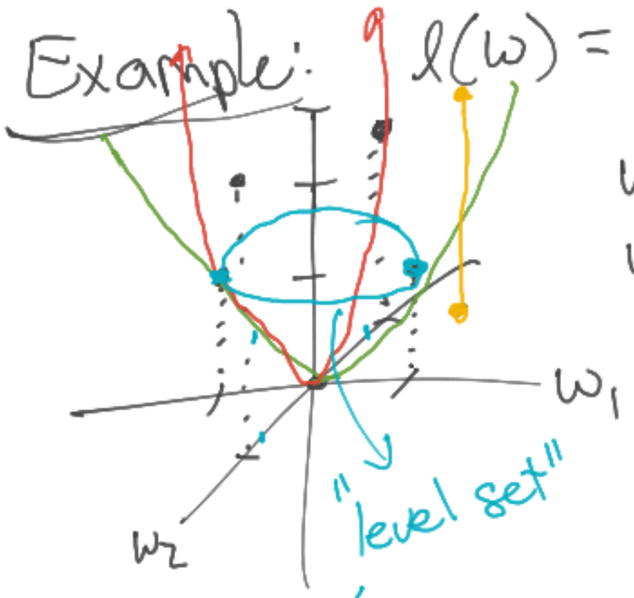
3) IF $\ell(w') < \ell(w)$

$w \leftarrow w'$

4) Go to #2

Stop when tired of waiting (duration is HP), when decrease in ℓ is very small (over last few improvements), or when no improvement has been found after trying many w'.

Example: $f(w) = w_1^2 + 2w_2^2$



all points where $f(w)$ takes the same value.

$w = (1,0)$, $f(w) = 1$

$w = (-1,0)$, $f(w) = 1$

$w = (0,1)$, $f(w) = 2$

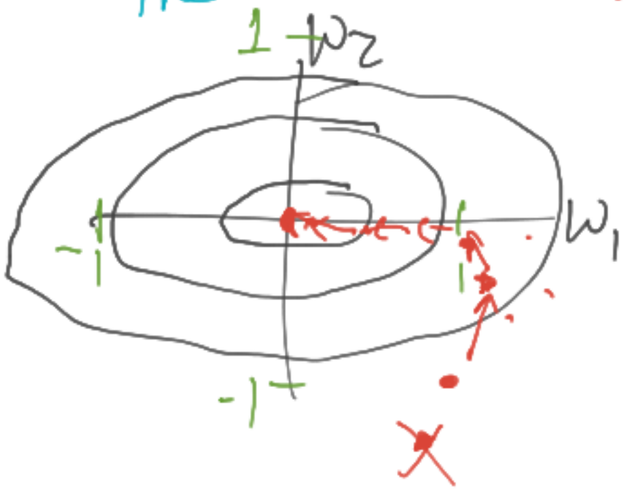
$w = (0,-1)$, $f(w) = 2$

$w = (0,0)$, $f(w) = 0$

Init $w = (1,-1)$

$w' = (.9, -1.2)$ ✗

$w' = (1.1, -0.5)$ (new w)



Pros

- Sometimes simple
- Don't require much knowkge about f.
  → Some BBO algs use more knowledge.

Cons:

- Sophisticated BBO algs quite complex. (CMA-ES)
- Don't leverage additional knowledge about f.

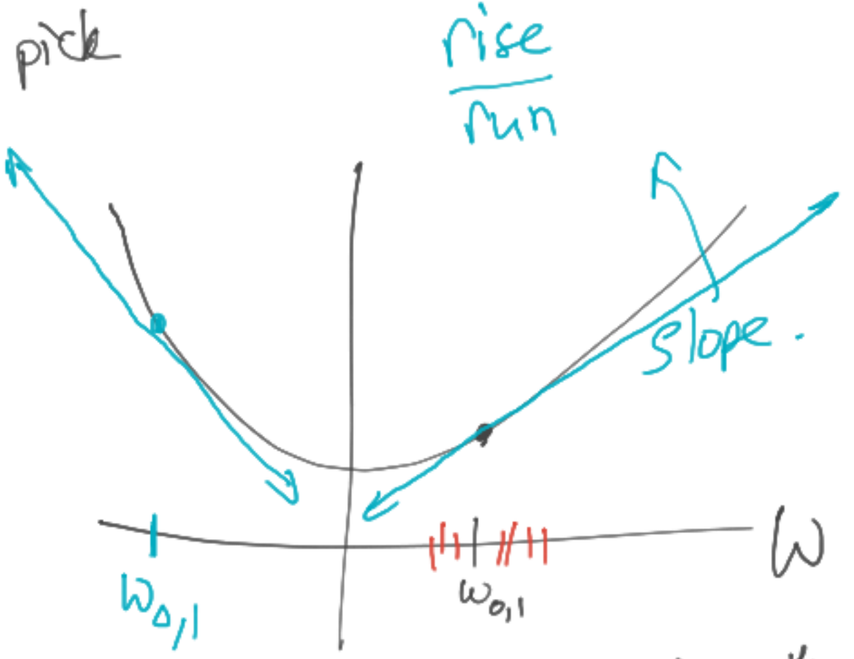
Idea: What if we could use knowledge about l to pick a good direction?

$$l(w) = \sum_{i=1}^{n} (y_i - w_1 x_{i,1})^2$$

$$= \sum_{i=1}^{n} (y_i^2 - 2w_1 x_{i,1} + w_1^2 x_{i,1}^2)$$

quadratic function of w_1

- Least squares loss, l, is always quadratic for linear models $\Rightarrow$ one global minimum.



$w_{k,j}$ = j^{th} weight of k^{th} w.

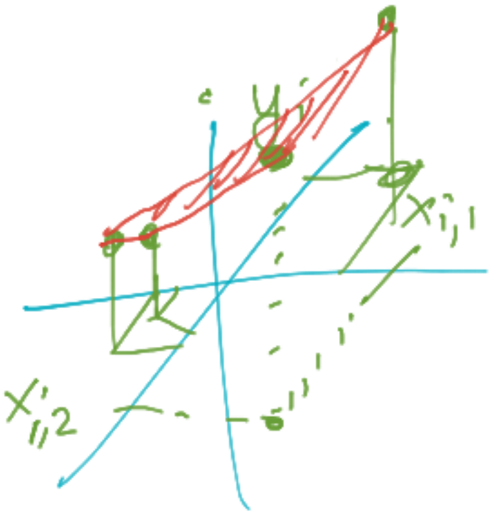
$x_{i,j}$ = j^{th} feature of i^{th} input

w_j = j^{th} weight

x_i is a vector.

$y_i \in \mathbb{R}$

(x_i, y_i) = i^{th} data point

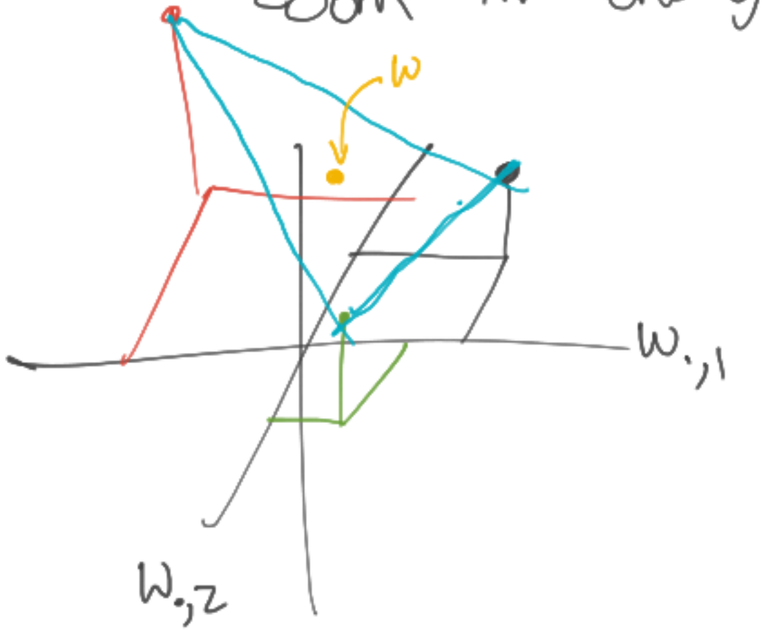


$M = 2$

- Idea: use negative of the slope as our downwards direction "descent direction"

  - Descent direction is a local property
    - depends only on immediate surroundings, of current position.

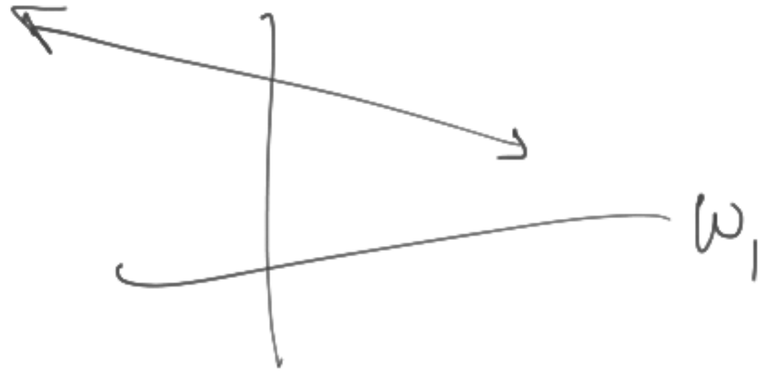
- "zoom in" enough, surface is flat.



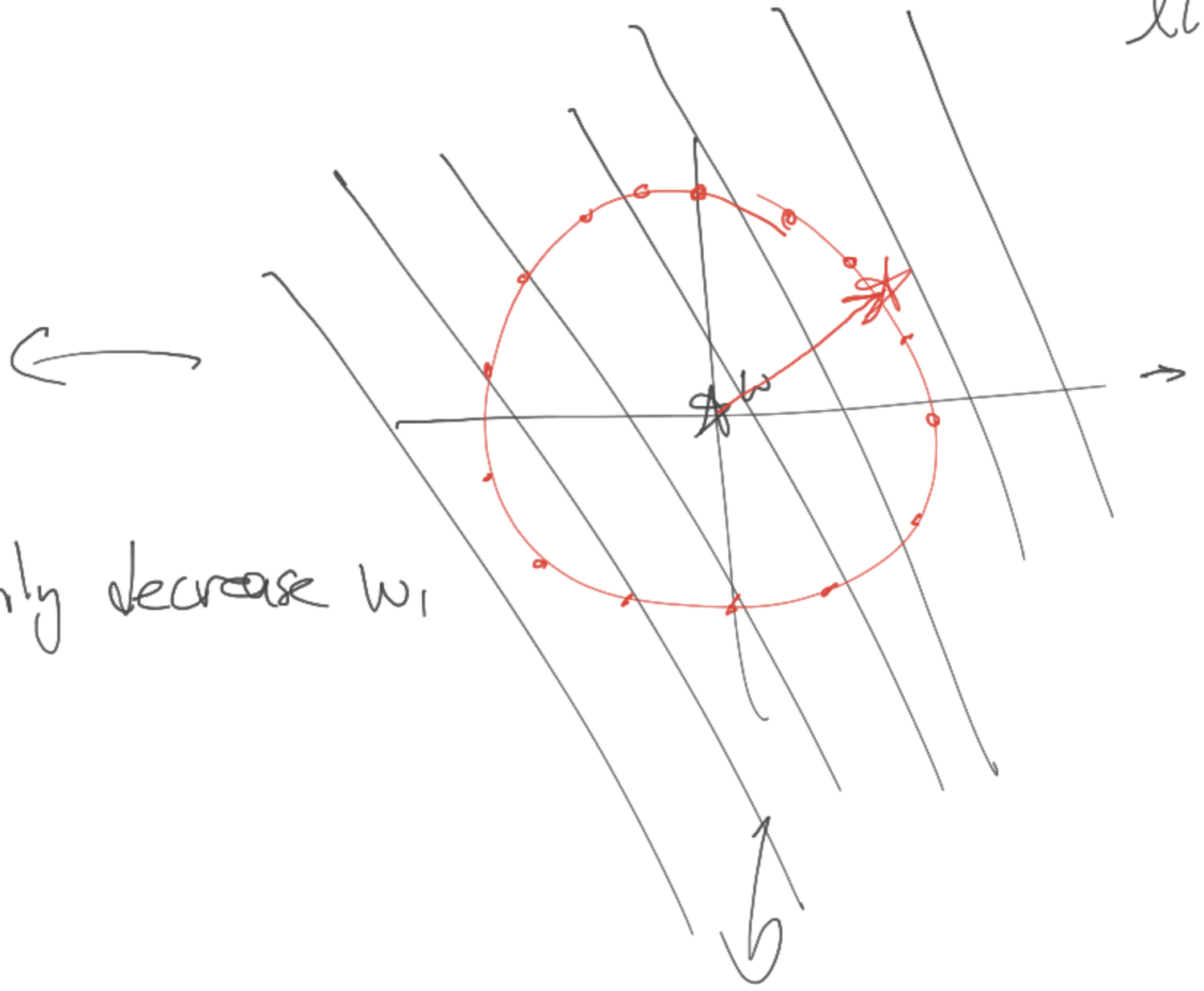
$\frac{\partial l(w)}{\partial w_1}$ = slope along $w_{,1}$ axis

↳ $g(w_1) = l(w_1, w_2)$

$\frac{\partial g(w_1)}{\partial w_1}$



$l(w)$ level sets. (locally)



$$\frac{\partial l(w)}{\partial w_1} = -10$$

$$\frac{\partial l(w)}{\partial w_2} = -1$$

AG: Only decrease w_1

The direction of steepest ascent is the gradient

$$\nabla l(w) = \left(\frac{\partial l(w)}{\partial w_1}, \frac{\partial l(w)}{\partial w_2}, \dots, \frac{\partial l(w)}{\partial w_m} \right).$$

∇ nabla

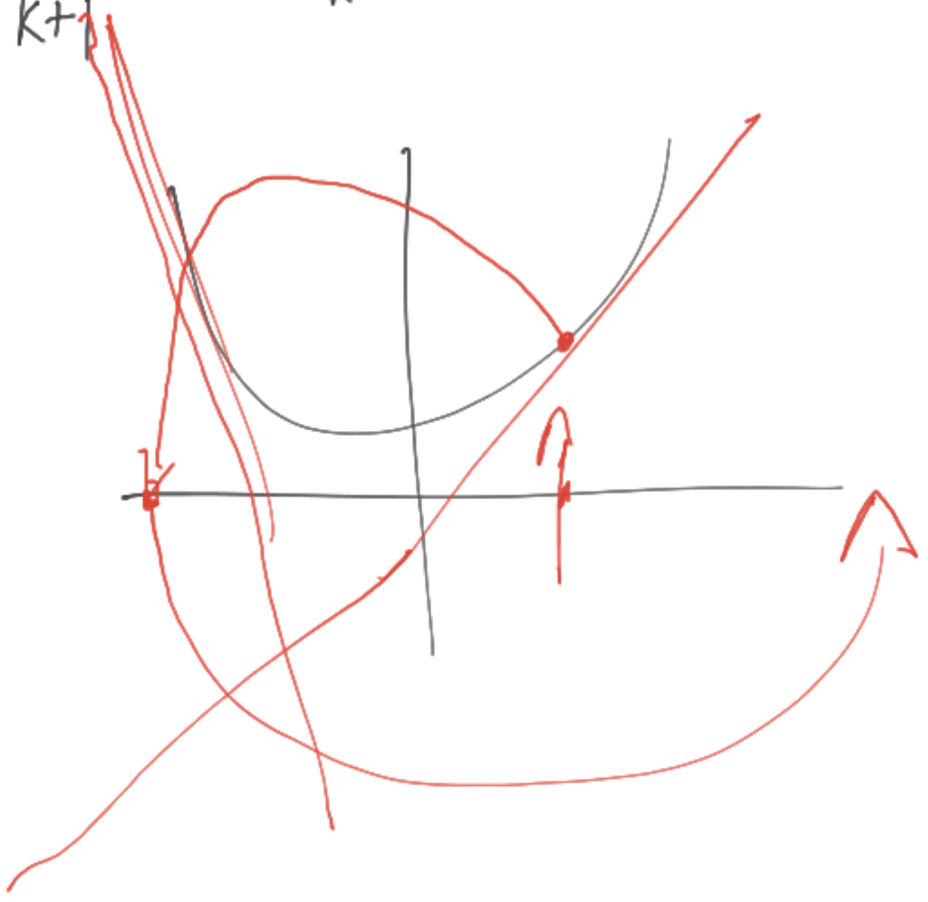
The direction of steepest descent is negative the gradient:

$$-\nabla l(w).$$

sequence of w's

$$w' = w - \nabla l(w)$$

$$w_{k+1} = w_k - \nabla l(w_k) \rightarrow \forall j \quad w_{k+1,j} = w_{k,j} - \frac{\partial l(w_k)}{\partial w_{k,j}}$$



$$w_{k+1} = w_k - \alpha \nabla l(w_k)$$

$$\forall j \in \{1, \ldots, m\} \quad w_{k+1,j} = w_{k,j} - \alpha \frac{\partial l(w_k)}{\partial w_{k,j}}$$

Gradient descent.

"small"

Hyperparameter.

0.1

0.01

0.001