Unconstrained optimization I

Topics we'll cover

1. Optimization by local search
2. The problem of multiple local optima
3. Gradient descent
4. Taking the derivative of a function of many variables

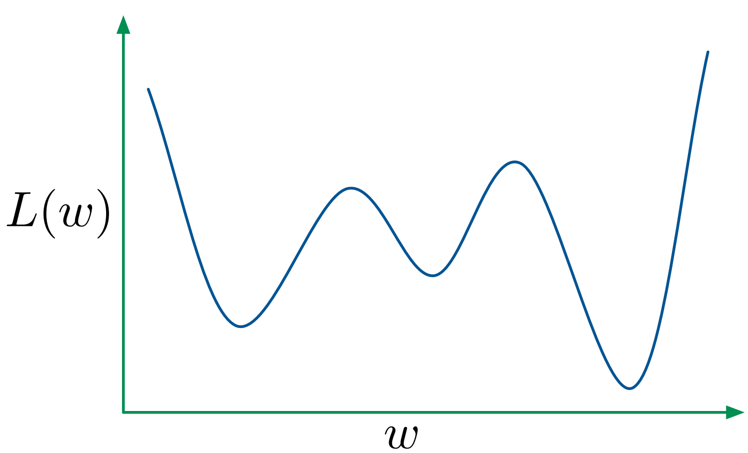
Minimizing a loss function

Usual setup in machine learning: choose a model w by minimizing a loss function $L(w)$ that depends on the data.

- Linear regression: $L(w) = \sum_i (y^{(i)} - (w \cdot x^{(i)}))^2$
- Logistic regression: $L(w) = \sum_i \ln(1 + e^{-y^{(i)}(w \cdot x^{(i)})})$

Default way to solve this minimization: **local search**.

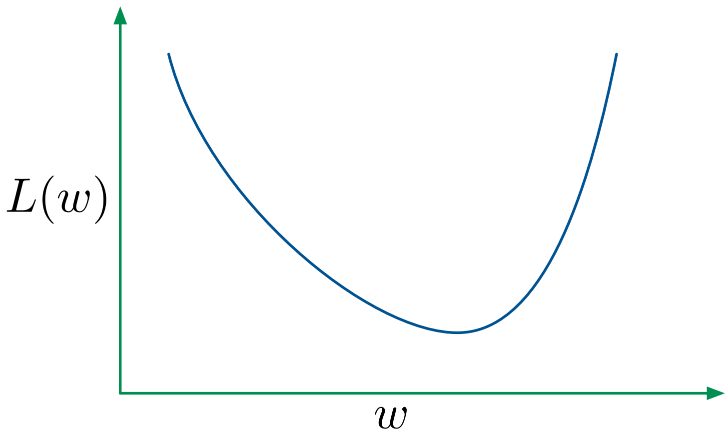
Local search



- Initialize w arbitrarily
- Repeat until w converges:
  - Find some w' close to w with $L(w') < L(w)$.
  - Move w to w'.

A good situation for local search

When the loss function is **convex**:



Idea for picking search direction:
Look at the **derivative** of $L(w)$ at the current point w.

Gradient descent

For minimizing a function $L(w)$:

- $w_o = 0, t = 0$
- while $\nabla L(w_t) \not\approx 0$:
  - $w_{t+1} = w_t - \eta_t \nabla L(w_t)$
  - $t = t + 1$

Here η_t is the *step size* at time t.

Multivariate differentiation

Example: $w \in \mathbb{R}^3$ and $F(w) = 3w_1 w_2 + w_3$.

Example: $w \in \mathbb{R}^d$ and $F(w) = w \cdot x$.

Example: $w \in \mathbb{R}^d$ and $F(w) = \|w\|^2$.

Gradient descent

For minimizing a function $L(w)$:

- $w_o = 0, t = 0$
- while $\nabla L(w_t) \not\approx 0$:
  - $w_{t+1} = w_t - \eta_t \nabla L(w_t)$
  - $t = t + 1$

Here η_t is the *step size* at time t.