03. Gradient Descent

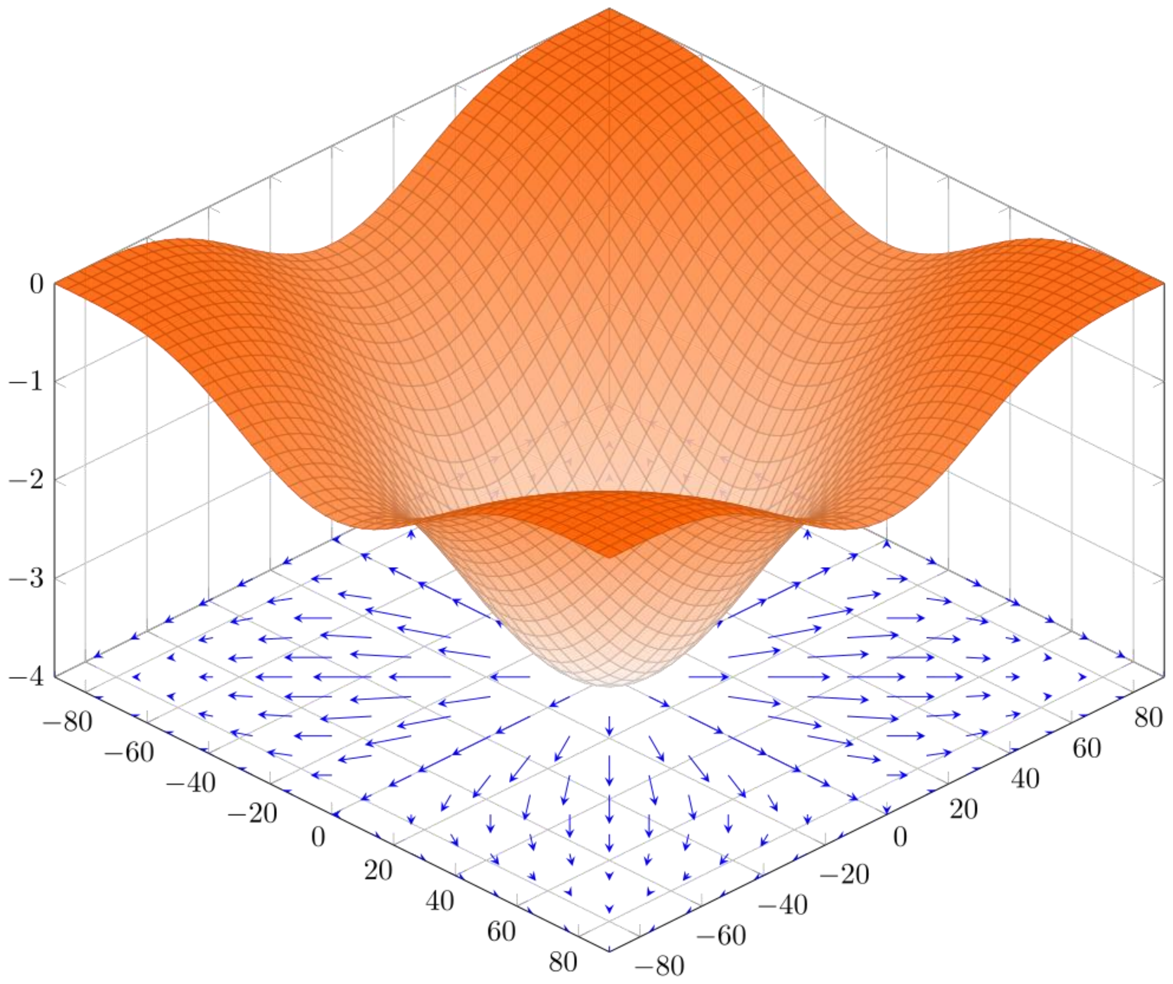
Общая идея

- Часто в машинном обучении стоит задача найти минимум какой-то «функции»
- Часто эти «функции» - модели машинного обучения, нейросети и т.д.

Производная

- Градиент – вектор, составленный из частных производных функции
- Градиент направлен в сторону роста функции

$$\nabla f(p) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(p) \\ \vdots \\ \frac{\partial f}{\partial x_n}(p) \end{bmatrix}$$



Как искать минимум?

- Известно, что в точке экстремума производная равна 0
- Но большая часть функций, которые мы хотим оптимизировать либо не задаются аналитически, либо решения приравнивание из производных к 0 и решение уравнения даст нам неустойчивое решение

Градиентный спуск

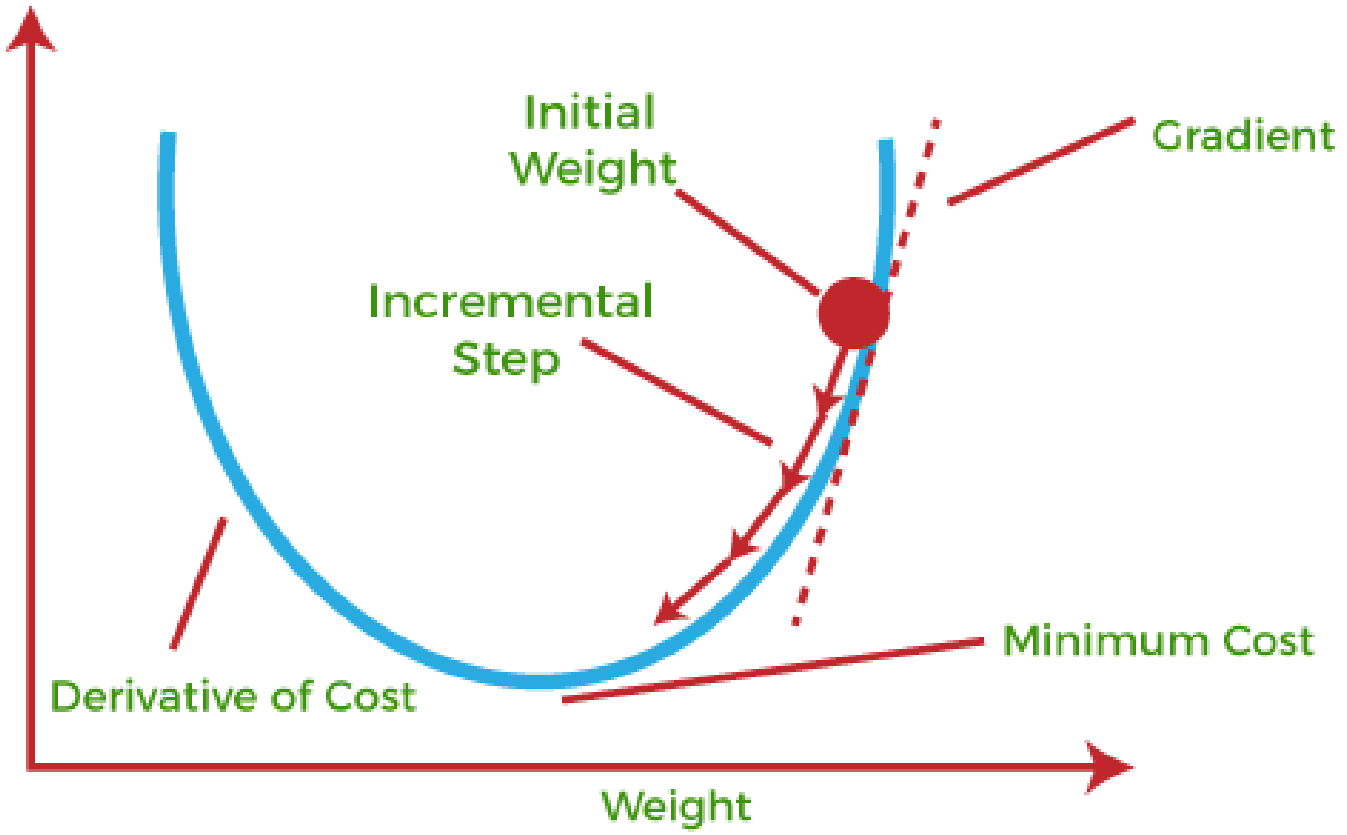
Gradient Descent Algorithm:

- Pick an initial point x_0
- Iterate until convergence

$$x_{t+1} = x_t - \gamma_t \nabla f(x_t)$$

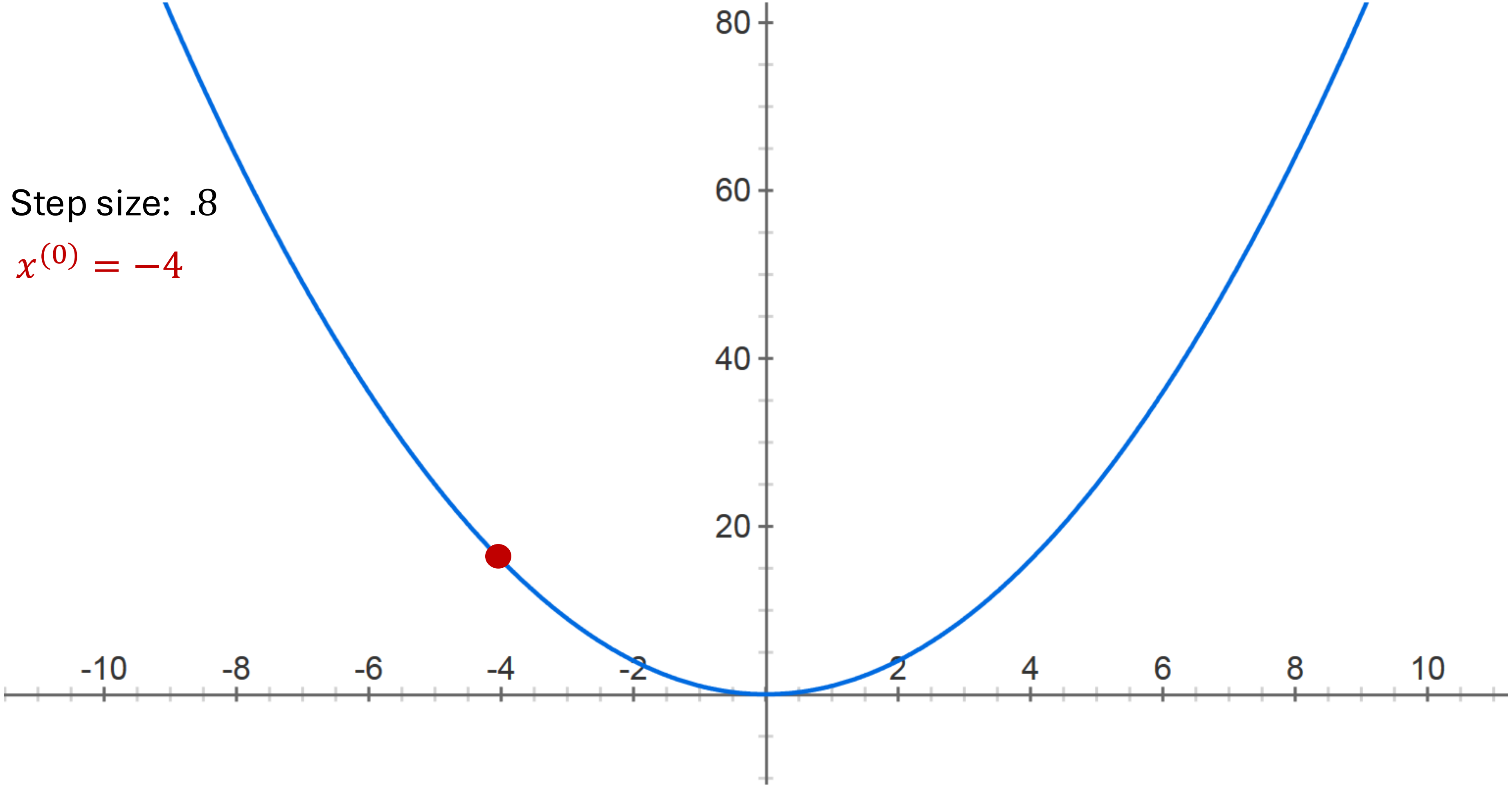
where γ_t is the t^{th} step size (sometimes called *learning rate*)

Градиентный спуск



Gradient Descent

$f(x) = x^2$



Step size: .8

$x^{(0)} = -4$

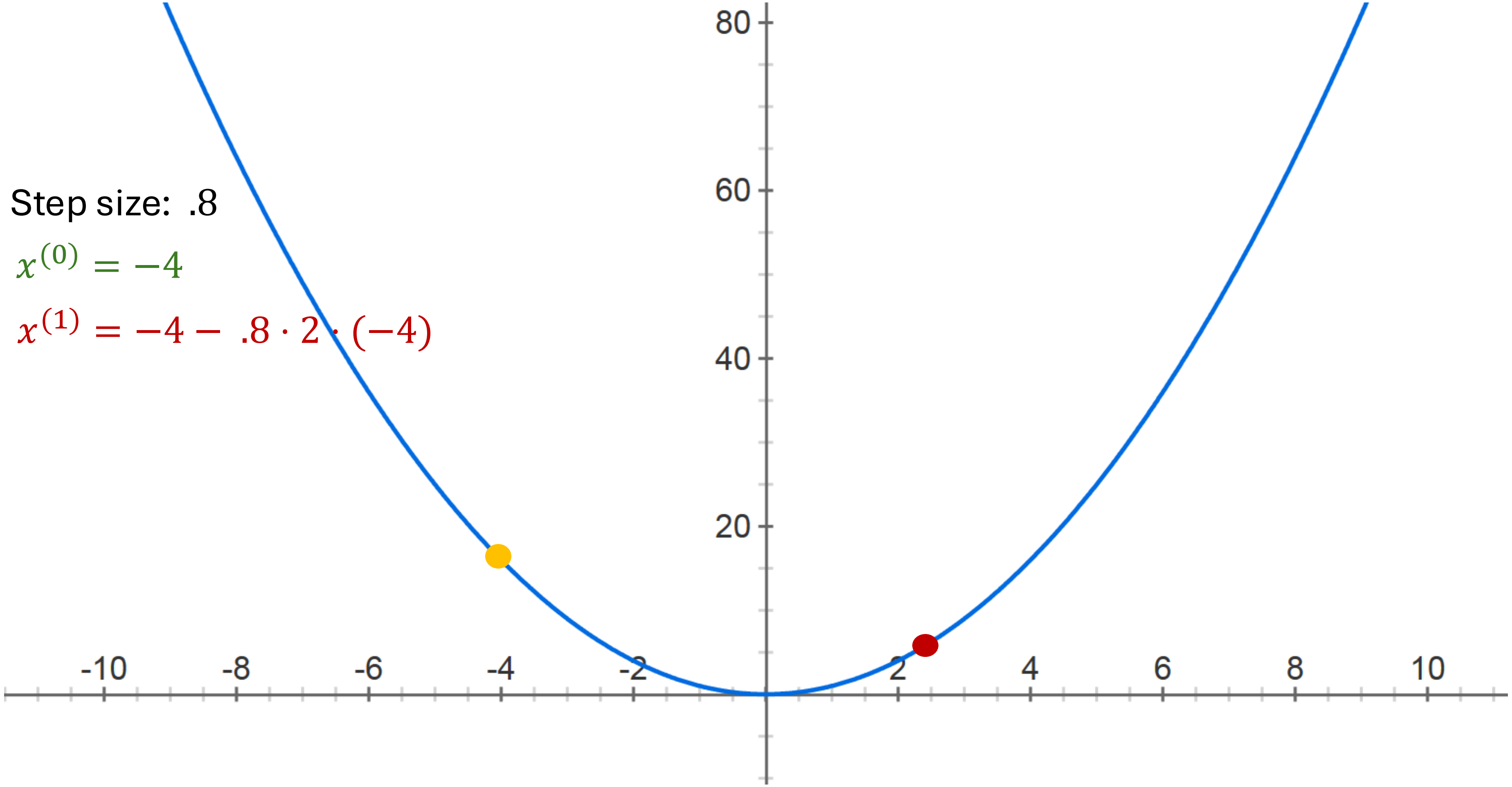
Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = -4 - .8 \cdot 2 \cdot (-4)$



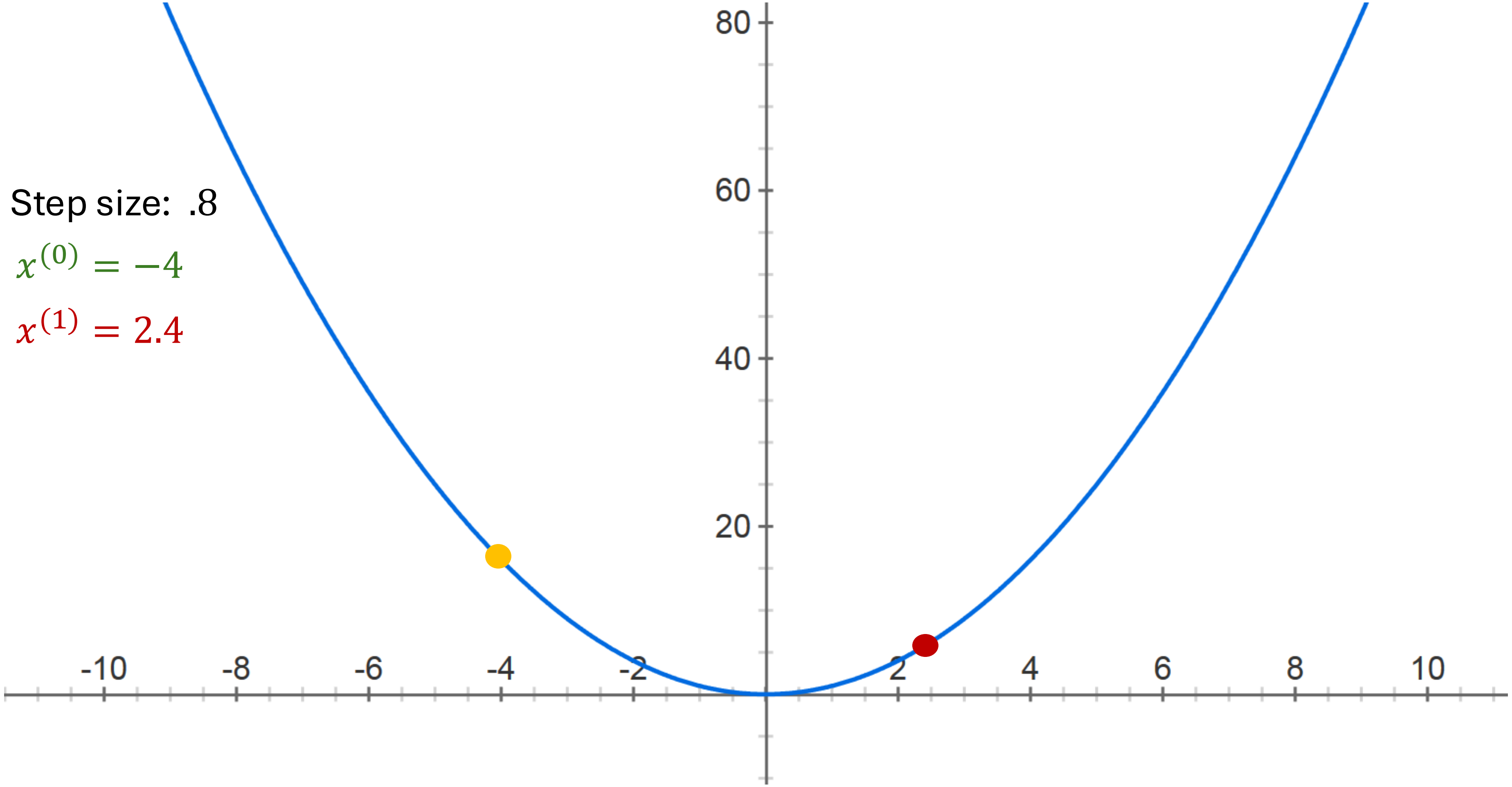
Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$



Gradient Descent

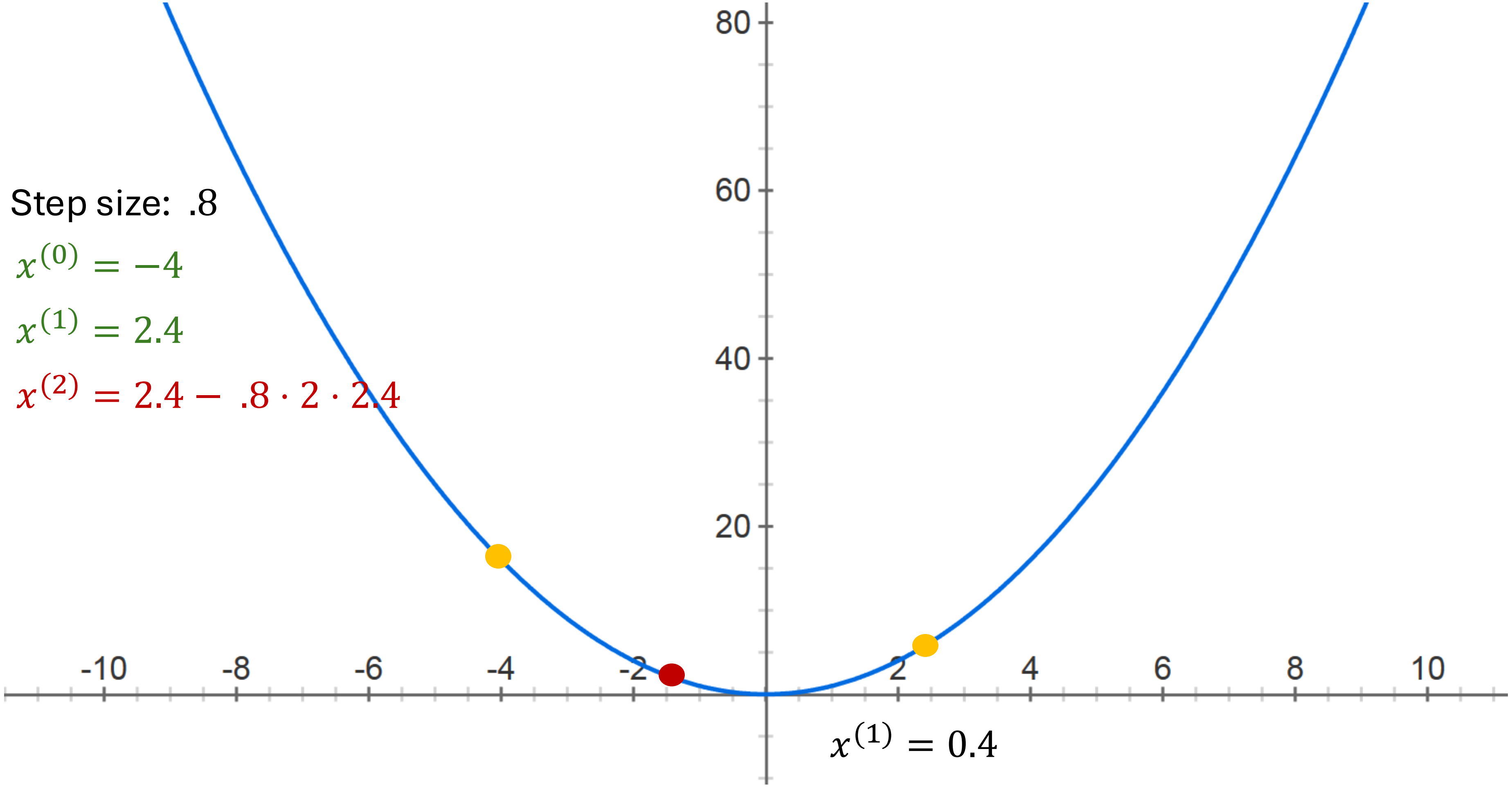
$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$

$x^{(2)} = 2.4 - .8 \cdot 2 \cdot 2.4$



$x^{(1)} = 0.4$

Gradient Descent

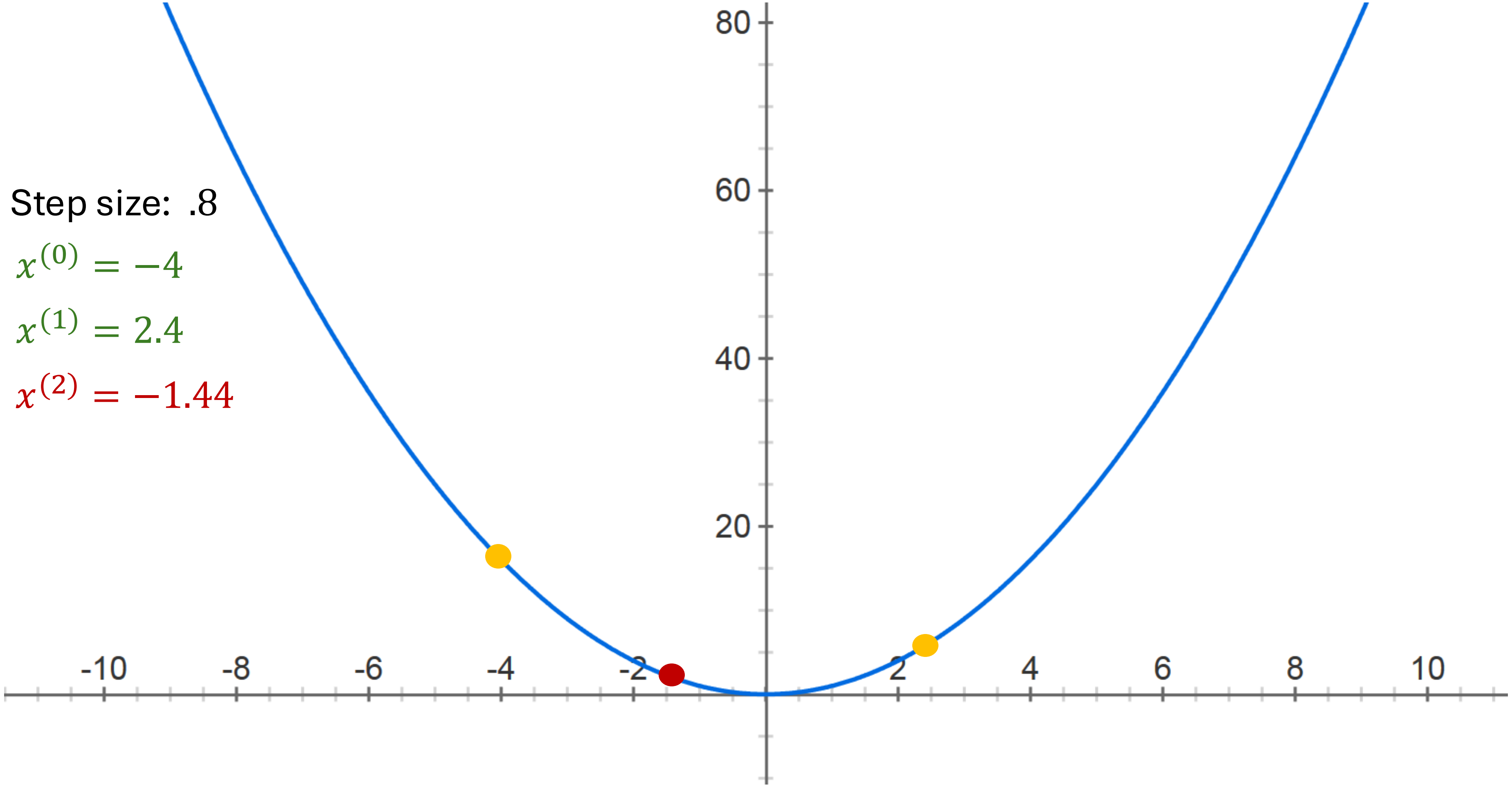
$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$

$x^{(2)} = -1.44$



Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

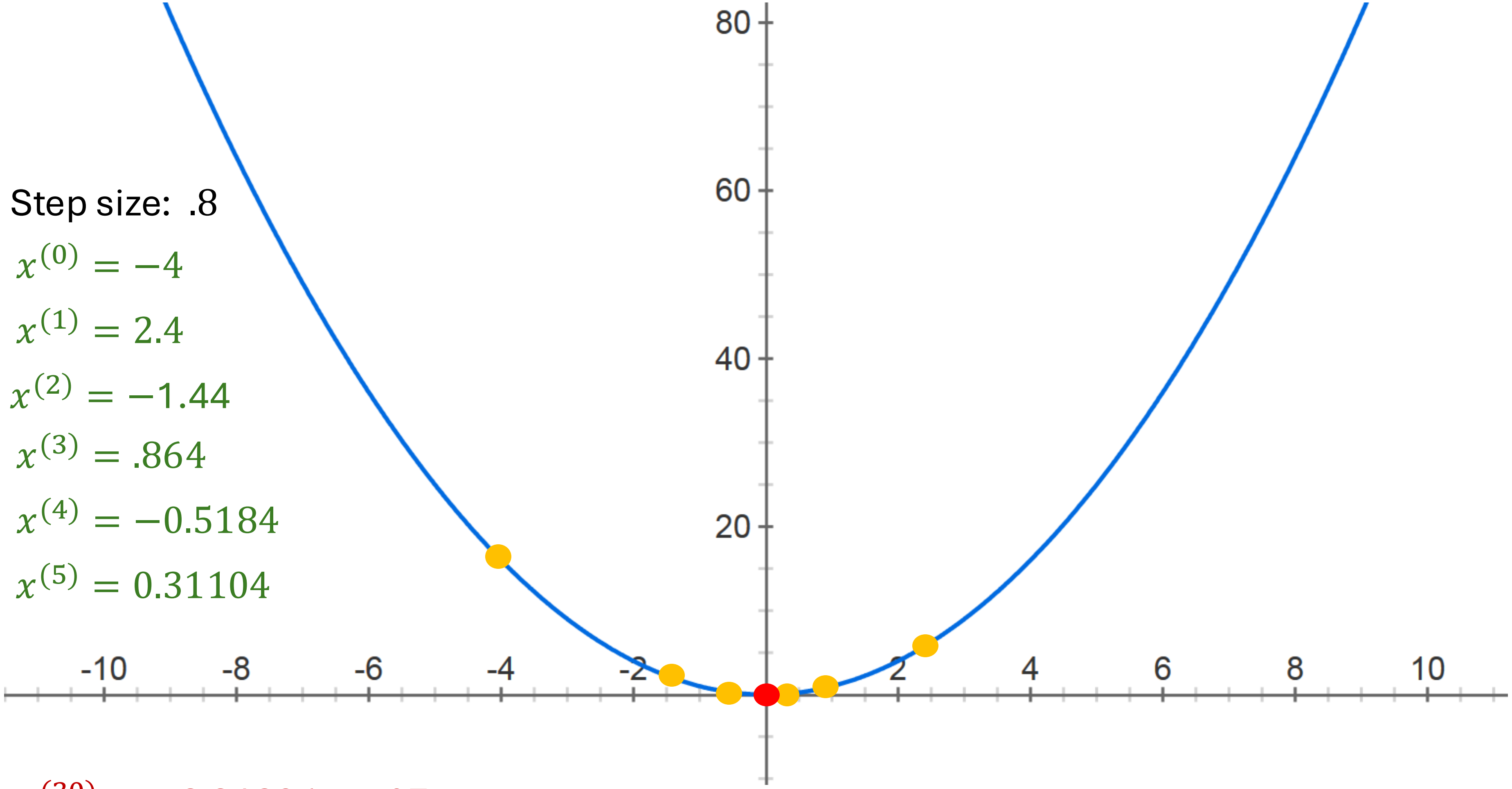
$x^{(1)} = 2.4$

$x^{(2)} = -1.44$

$x^{(3)} = .864$

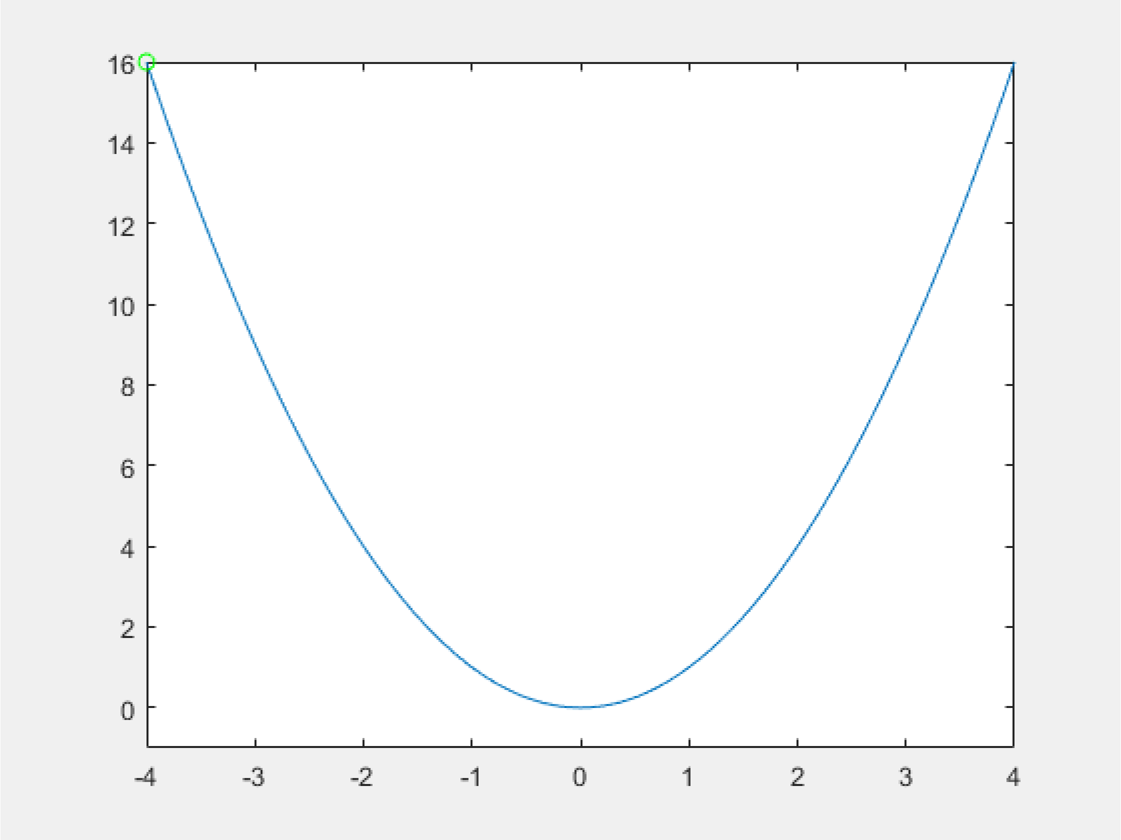
$x^{(4)} = -0.5184$

$x^{(5)} = 0.31104$



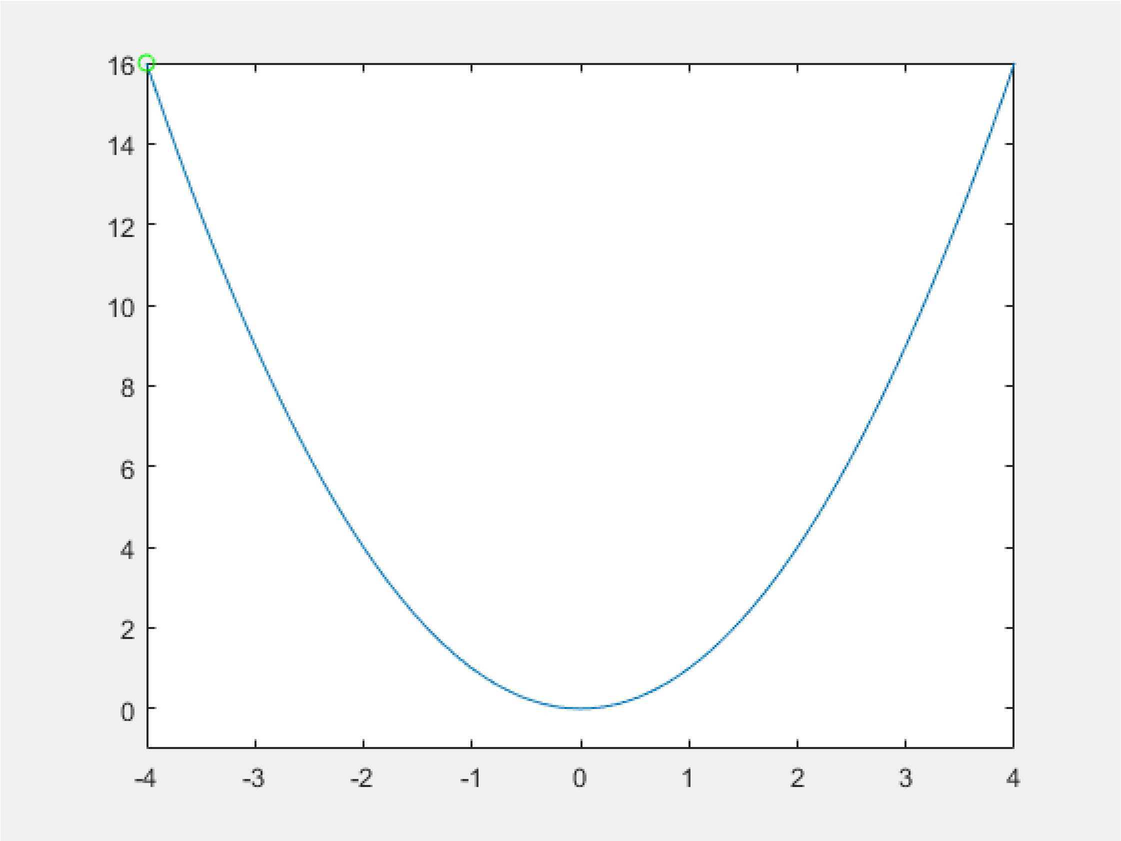
$x^{(30)} = -8.84296e - 07$

Gradient Descent



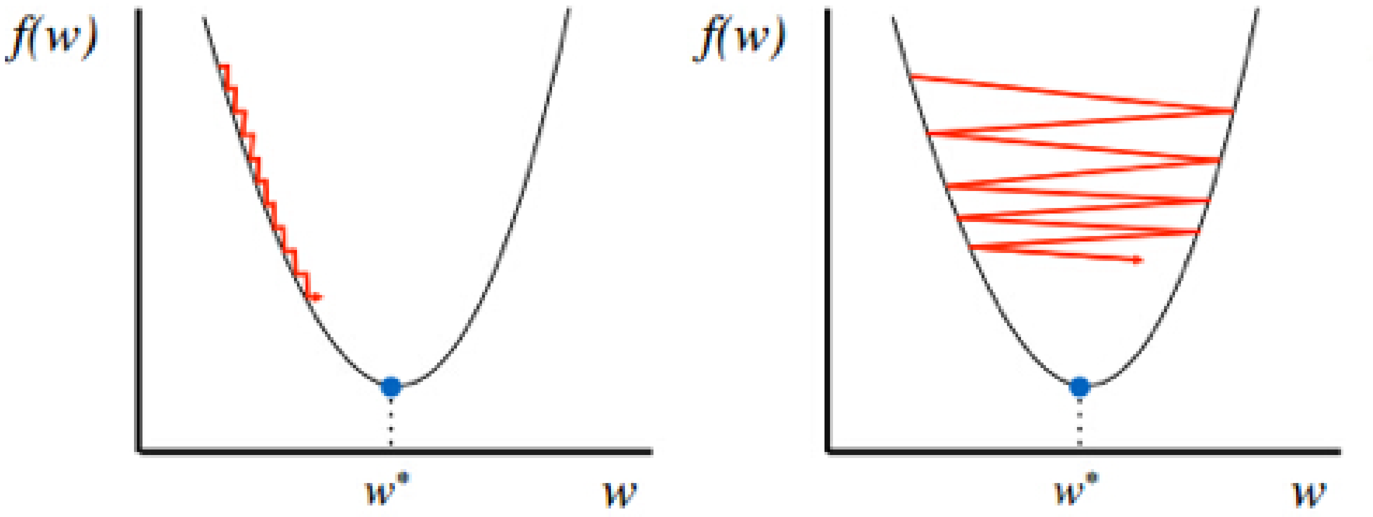
Step size: .9

Gradient Descent



Step size: .2

Learning rate



Too small: converge very slowly

Too big: overshoot and even diverge

Задача регрессии

- Пусть у нас есть набор данных $D = \{(x_i, y_i)\}$ и задача предсказывать значения y_i по x_i.
- Для этого мы строим функцию $f(x, w)$, у которой есть параметры w.
- Мы хотим подобрать параметры w так, чтобы ошибка (квадратичная ошибка) была минимальна

Задача регрессии

- Более формально:
- $\widehat{w} = \arg\min \sum L(x_i, y_i) = \arg\min \sum \big(y_i - f(x_i, w)\big)^2$
- Далее будем итеративно искать $\widehat{w}$:
- $w_{t+1} = w_t - \eta \frac{\partial}{\partial w}\left(\sum \big(y_i - f(x_i, w_t)\big)^2\right) = w_t - \sum \frac{\partial}{\partial w}\big(y_i - f(x_i, w_t)\big)^2 = w_t - \sum 2(y_i - f(x_i, w_t)) \frac{\partial}{\partial w} f(x_i, w_t)$

Batch Gradient Descent

- Пусть у нас есть набор данных $D = \{(x_i, y_i)\}$
- Давайте будем искать минимум $w_{t+1} = w_t \ - \eta \sum_{\mathrm{i}} \nabla L(x_i, y_i)$

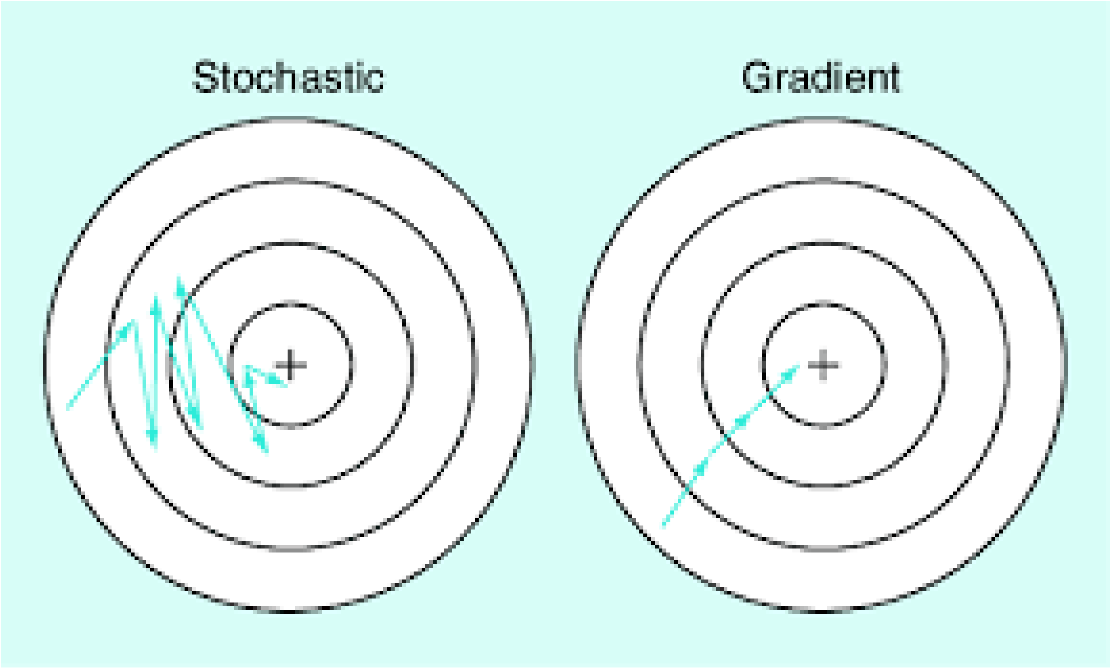
- Вычислительно сложно
- Требует много памяти
- Может застрять в локальном минимуме
- Чувствителен к выбросам

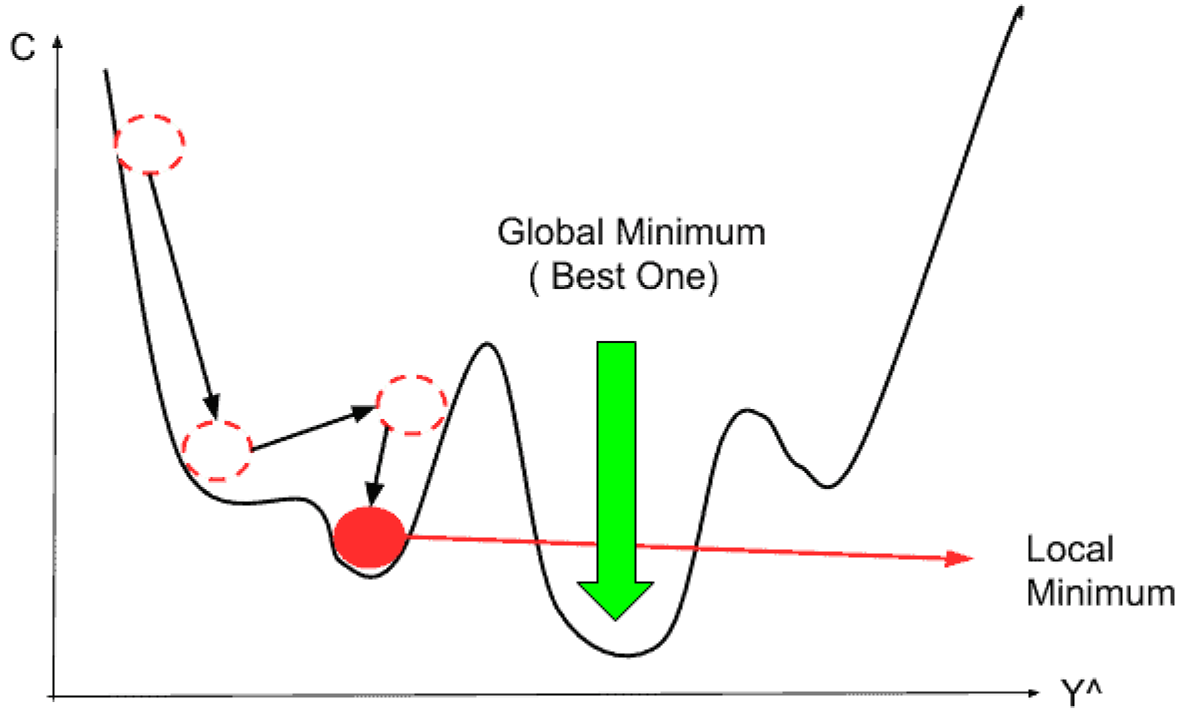
Stochastic Gradient Descent

- Давайте считать производные не по всем точкам датасета, а по случайному поднабору

- Может быть шумным
- Может застрять в локальном минимуме
- Чувствителен к гиперпараметрам

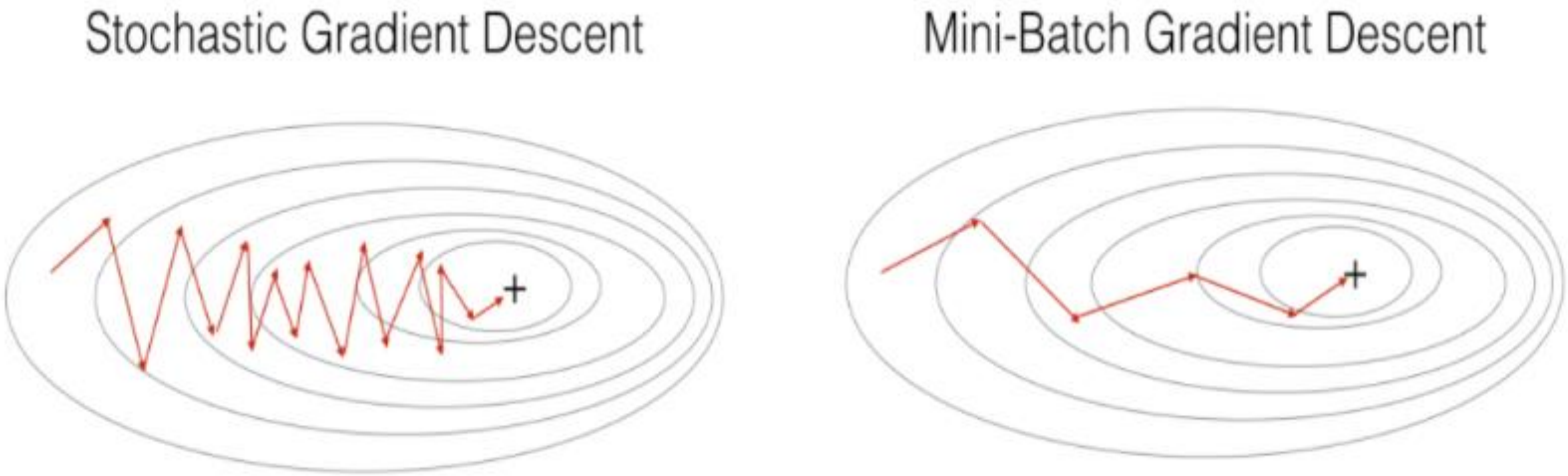
Stochastic Gradient Descent





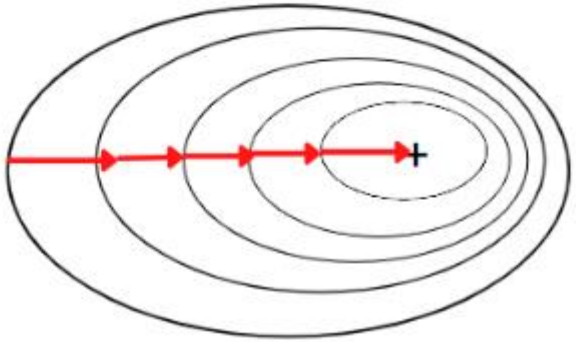
Mini-batch gradient descent

- Компромисс между двумя описанными методами
- Давайте разделим данные заранее на поднаборы и будем считать градиенты по ним

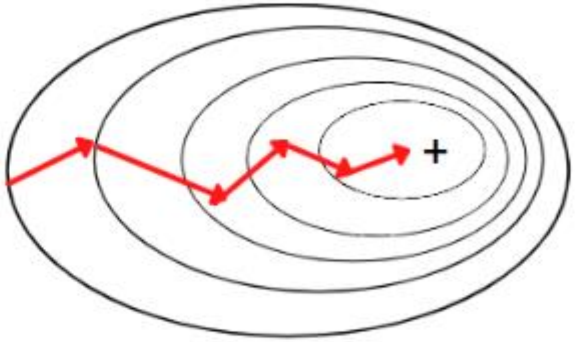


Сравнение

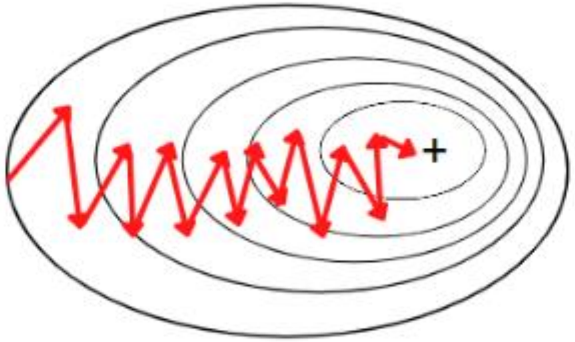
Batch Gradient Descent



Mini-Batch Gradient Descent



Stochastic Gradient Descent



Практика

- $f(x, w) = wx^2$
- $w_{t+1} = w_t - \eta \frac{\partial}{\partial w}\left(\sum\left(y_i - f(x_i, w_t)\right)^2\right) = w_t - \sum \frac{\partial}{\partial w}\left(y_i - f(x_i, w_t)\right)^2 = w_t - \sum 2\left(y_i - f(x_i, w_t)\right)\frac{\partial}{\partial w} f(x_i, w_t) = w_t - \sum 2\left(y_i - f(x_i, w_t)\right) x\_i\hat{}2$