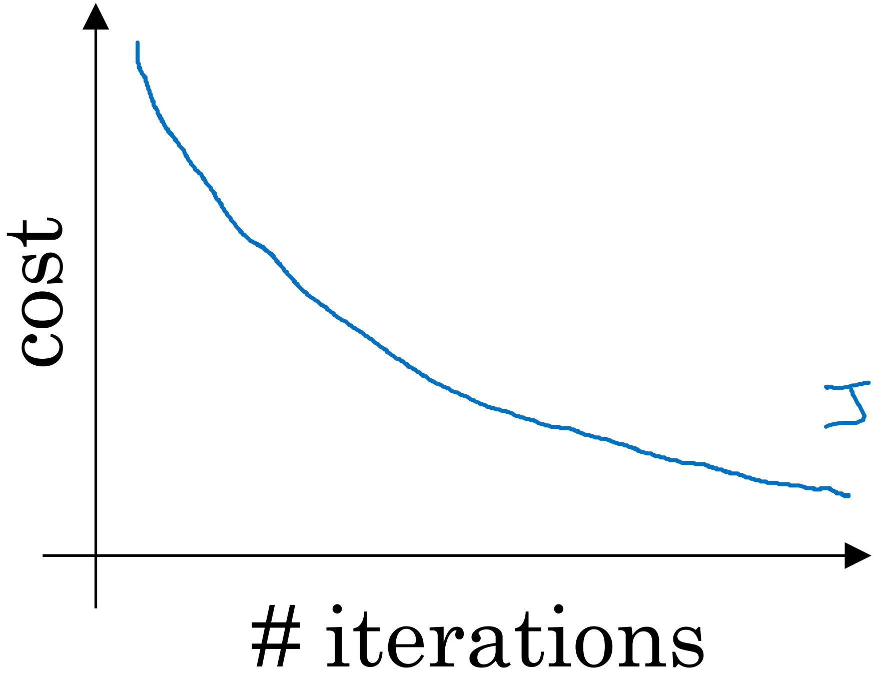
deeplearning.ai

Optimization Algorithms

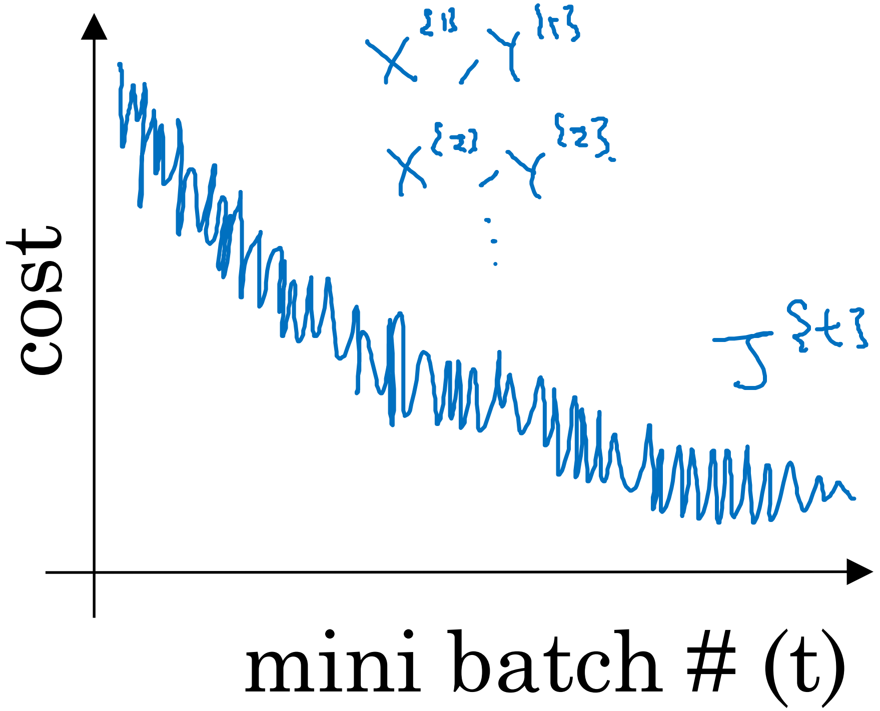
Understanding mini-batch gradient descent

Training with mini batch gradient descent

Batch gradient descent



Mini-batch gradient descent



Plot $J^{\{t\}}$ computed using $\underline{X^{\{t\}}, Y^{\{t\}}}$

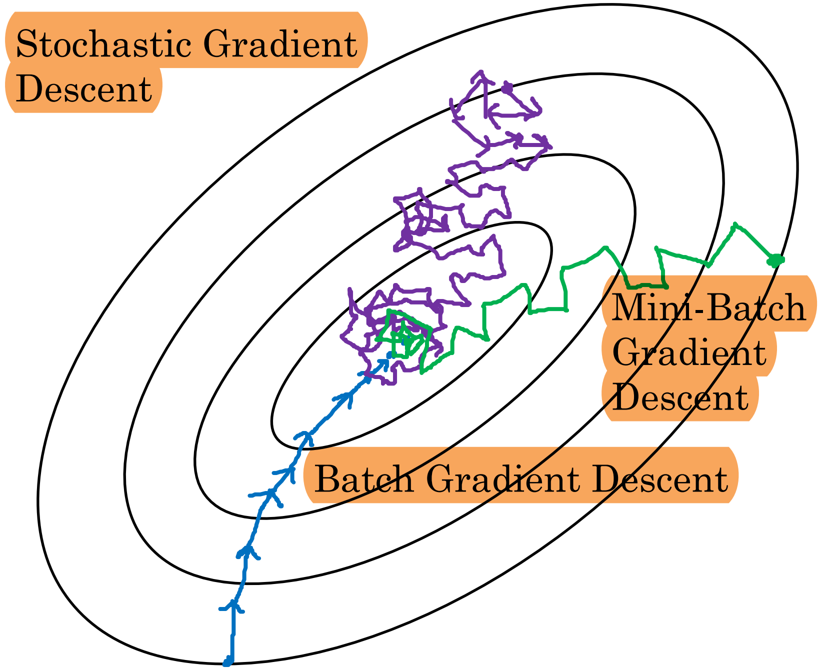
Andrew Ng

Choosing your mini-batch size

→ If mini-batch size = m : Batch gradient descent. $(X^{\{1\}}, Y^{\{1\}}) = (X, Y)$.

→ If mini-batch size = 1 : Stochastic gradient descent. Every example is it own mini-batch.
$(X^{\{1\}}, Y^{\{1\}}) = (x^{(1)}, y^{(1)})$ … $(x^{(2)}, y^{(1)})$ mini-batch.

In practice: Somewh in-between 1 and m



Stochastic Gradient Descent

Mini-Batch Gradient Descent

Batch Gradient Descent

Stochastic gradient descent

↓

Lose speedup from vectorization

In-between (mini-batch size not too big/small)

↓

Fastest learning.

- Vectorization. (~1000)
- Make progress without processing entire training set.

Batch gradient descent (mini-batch size = m)

↓

Too long per iteration

Choosing your mini-batch size

If small training set: Use batch gradient descent.
$(m \le 2000)$

Typical mini-batch sizes:

$$\longrightarrow \underbrace{\underset{2^6}{64}, \underset{2^7}{128}, \underset{2^8}{256}, \underset{2^9}{512}} \qquad \underset{2^{10}}{\underline{1024}}$$

Make sure mini-batch fit in CPU/GPU memory.
$X^{\{t\}}, Y^{\{t\}}$