Mini-Batch Stochastic Gradient Descent

1. Preparing dataset
   - proxy → quick test dataset for debugging
     - remove bugs
     - small, run fast
   - split into : training / validation / evaluation

     held out for you

2. Setting initial hyperparameters

   Not network parameters
   Used to train network
   Not "learned"

   Learned parameters are things like $W^{[l]}$ + $b^{[l]}$

3. Create the NN (model)

   instance

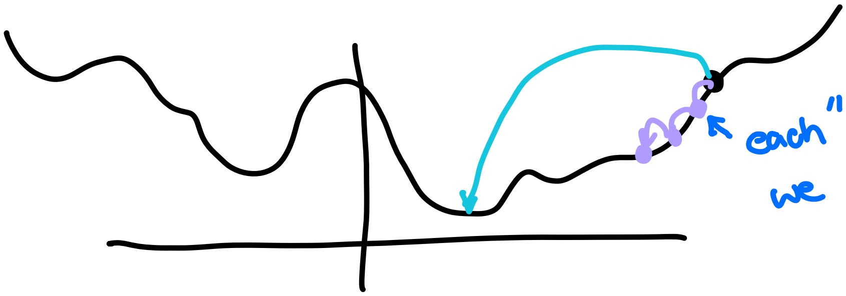
4. Train model

Gradient Descent (Batch Gradient Descent)

We average gradients across all training examples.

```
for each epoch
    1. compute all N gradients
    2. average all N gradients
    3. update parameters using average
       gradients
```

Compute gradient w.r.t. every training example



each "jump" is when we update parameters

+ very stable (loss nearly always goes down)

- very slow

Stochastic Gradient Descent

```
for each epoch
    shuffle the examples
    randomly / stochasticly
    for each example
        1. compute gradients
        2. update parameters
```

How many times do we update parameter per epoch for $\underbrace{\text{BGD}}_{1}$ + $\underbrace{\text{SGD.}}_{N \leftarrow \text{\# of training examples}}$

- + much faster convergence
- − susceptible outliers
  - ↳ less general

Mini-Batch SGD

```
for each epoch
    randomly create batches
    for each batch
        1. compute gradients
        2. average gradients
        3. update parameters
```

BGD

↓

[1, N]

↑

SGD