Lecture 9 (Stochastic Gradient Descent)

1 SGD Intuition

The complexity of GD is $O(mn)$ for each iteration which is not time efficient when m is large, so SGD aims at improving the time complexity in this aspect. The input (x_i, y_i) given to the model is assumed to be **i.i.d (identically independently distributed)**. So the gradient can be computed over a *batch* on the examples.

2 SGD Algorithm

```
t = 0
theta = init()
do {
    Bt = RandomSample(data, r) # sampling
    updateTheta(theta, f, Bt)
    t += 1
} while (not converged)
```

Time complexity for each iteration: $O(rn)$

Instead of `sampling`, we can do the following:

1. Randomly shuffle the data
2. Divide the dataset into m/r batches
3. Cyclically move through the batches for each iteration

3 Fact (Prove as HW)

$$E_{B(t)}[\nabla_\theta(J_{B(t)}(\theta))] = \nabla_\theta(J(\theta))$$

(my idea: can be proven by showing contribution of each input i is same on both sides)