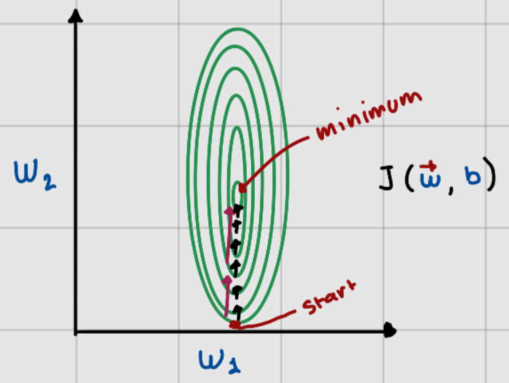
Advanced Optimization

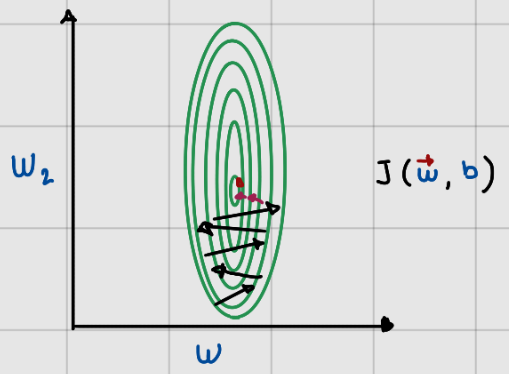
Gradient descent

$$\vec{w}_j = \vec{w}_j - \alpha \frac{\partial}{\partial \vec{w}_j} J(\vec{w}, b)$$



Go faster - increase α

"ADAM Algorithm"



Go slower - decrease α

Adam Algorithm Intuition

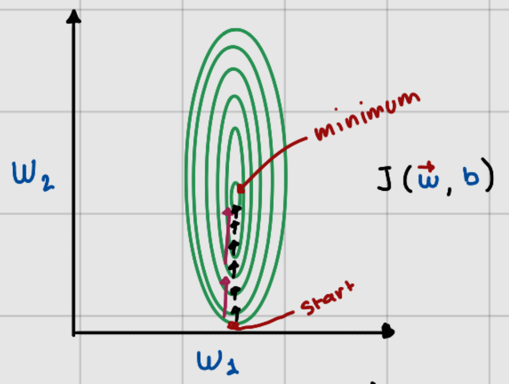
Adam: Adaptative Moment estimation not just one α

$$w_1 = w_1 - \alpha_1 \frac{\partial}{\partial w_1} J(\vec{w}, b)$$

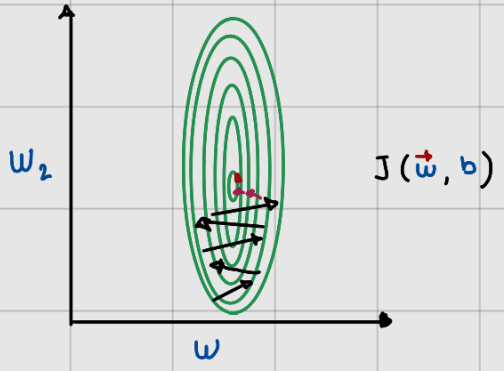
$$\vdots$$

$$w_{10} = w_{10} - \alpha_{10} \frac{\partial}{\partial w_{10}} J(\vec{w}, b)$$

$$b = b - \alpha_{11} \frac{\partial}{\partial b} J(\vec{w}, b)$$



IF w_j (or b) Keeps moving in same direction, increase α_j



if w_j (or b) keeps oscillating, reduce α_j

In code

Model

```
model = Sequential([
            Dense(units = 25, activation = 'relu'),   # layer 1
            Dense(units = 15, activation = 'relu'),   # layer 2
            Dense(units = 25, activation = 'linear')  # layer 3
            ])
```

Compile

```
model.compile(opmizer = tf.keras.optimizers.Adam(learning_rate = 1e-3),
              loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True))
```

Fit

```
model.fit(X, Y, epochs= 100)
```