

# FLAVORS OF GRADIENT DESCENT

---

Heuristic Multi-Modal and Plateau Optimization

# GRADIENT DESCENT METHODS

---

All the current optimization algorithms are based on a variant of gradient descent.

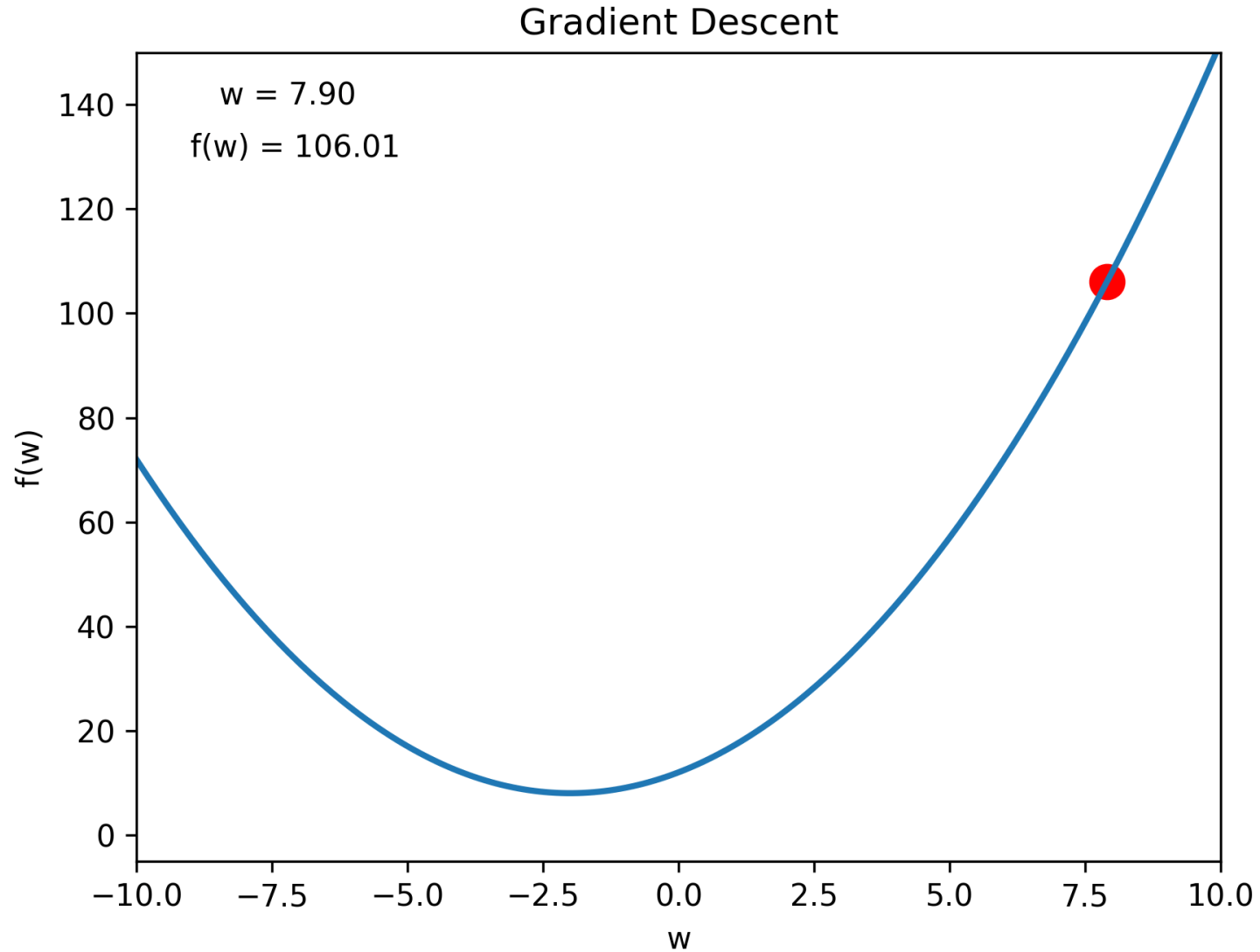
Let's denote the vector of weights at step  $t$  by  $w_t$  and the gradient of the objective function with respect to the weights by  $g_t$ . The idea is that the gradient descent algorithm updates the weights under the following principle:

$$w_t = w_{t-1} - \eta \cdot g_{t,t-1}$$

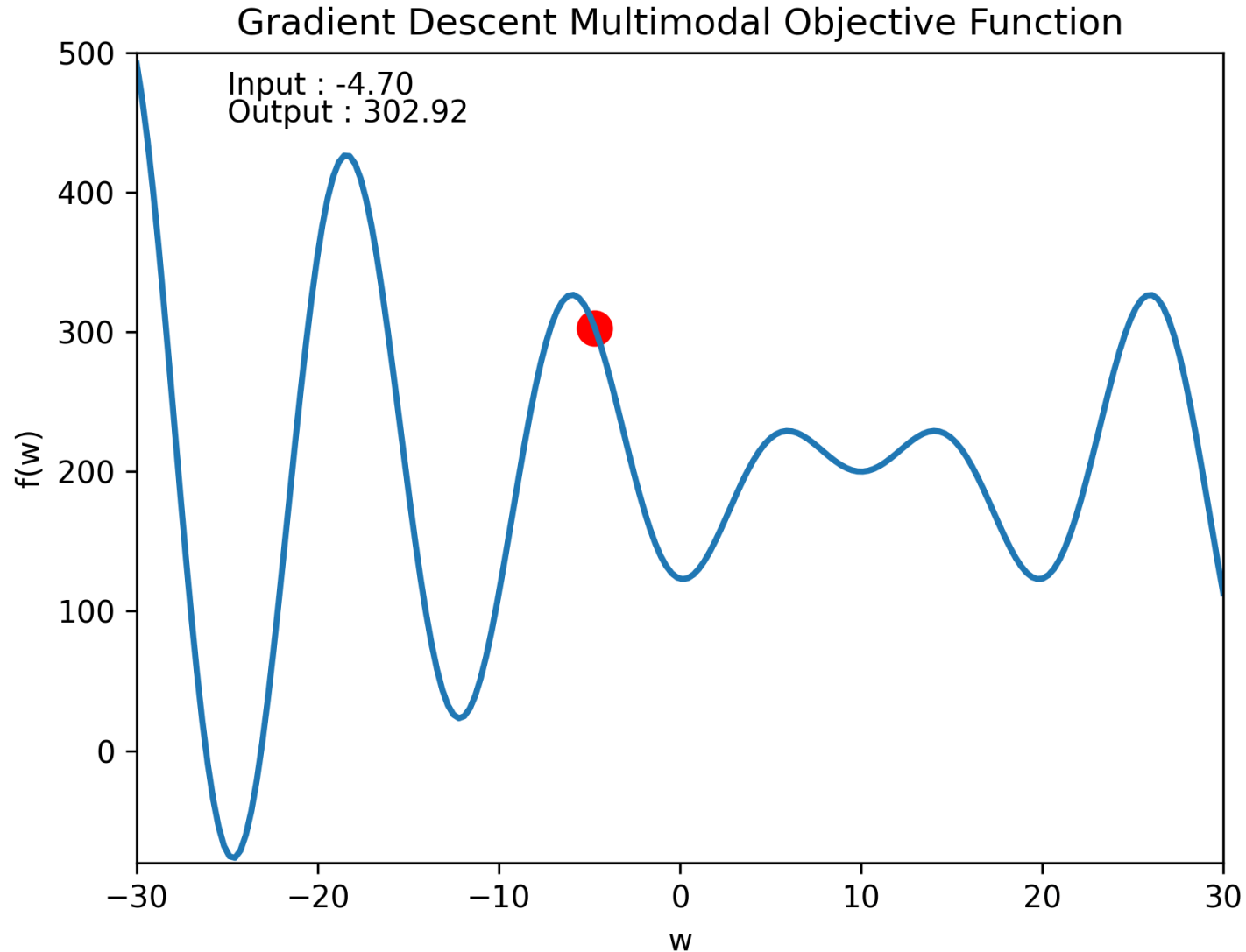
When the objective function (whose gradient with respect to the weights) is represented by  $g_t$  and has multiple local minima or a very shallow region containing the minima, the plain gradient descent algorithm may not converge to the position sought. To remediate this deficiency, research proposed alternatives by varying the way we evaluate the learning rate at each step or how we compute the "velocity" for updating the weights:

$$w_t = w_{t-1} - \eta_t \cdot v_t$$

# GRADIENT DESCENT



# WHY FLAVORS OF GRADIENT DESCENT?



# DYNAMIC LEARNING RATES AND MINI BATCHES

---

The big idea is the same:

$$w_t = w_{t-1} - \eta_t \cdot v_t$$

In the equation above,  $\eta_t$  is an adaptive learning rate and  $v_t$  a modified gradient.

We can consider an exponential decay, such as

$$\eta_t = \eta_0 e^{-\lambda \cdot t}$$

or a polynomial decay

$$\eta_t = \eta_0 (\beta t + 1)^{-\alpha}$$

A mini-batch is a random subset of the observations, denoted by  $B_t$  and the loss function by  $f$  (thus,  $f$  can be understood as SSE or MSE).

$$g_{t,t-1} = \partial_w \frac{1}{|B_t|} \sum_{i \in B_t} f(x_i, w_{t-1}) = \frac{1}{|B_t|} \sum_{i \in B_t} h_{i,t-1}$$

## (MINI-BATCH) MOMENTUM GRADIENT DESCENT

---

A mini-batch is a random subset of the observations, denoted by  $B_t$  and the loss function by  $f$ .

$$g_{t,t-1} = \partial_w \frac{1}{|B_t|} \sum_{i \in B_t} f(x_i, w_{t-1}) = \frac{1}{|B_t|} \sum_{i \in B_t} h_{i,t-1}$$

The momentum gradient descent computes the velocity as follows

$$v_t = \beta v_{t-1} + g_{t,t-1}$$

and  $\beta \in (0, 1)$ .

For an explicit formula, we have

$$v_t = \sum_{\tau=0}^{t-1} \beta^\tau g_{t-\tau,t-\tau-1}$$

and the weights are updated as follows:

$$w_t = w_{t-1} - \alpha v_t$$

where


$$\alpha = \frac{\eta}{1 - \beta}$$

# ADAPTIVE GRADIENT DESCENT (ADAGRAD)

---

The main idea is to create adaptive learning rates to the gradient:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{s_t + \epsilon}} \cdot g_t$$

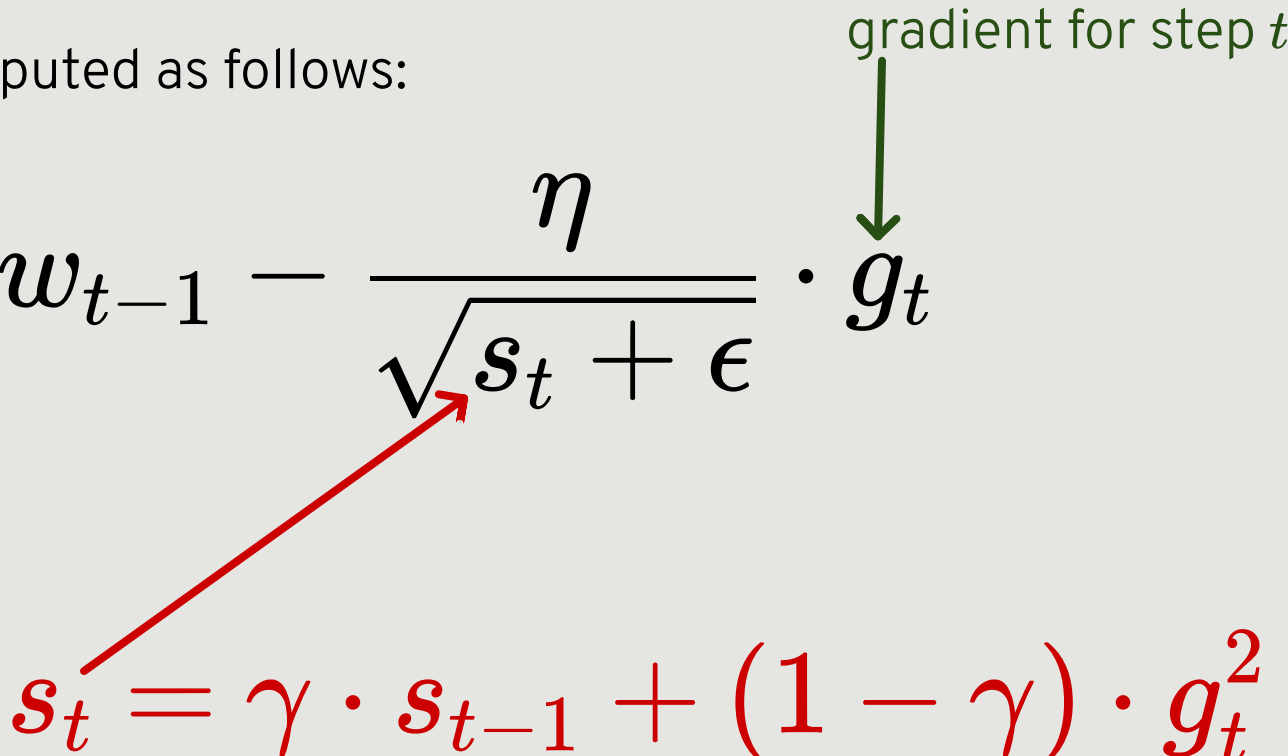
$$s_t = s_{t-1} + g_t^2$$


**Critical Thinking:** What happens in this case with the learning rate if the magnitude of the gradient increases?

# ROOT MEAN SQUARE PROPAGATION (RMSPROP)

---

The updates are computed as follows:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{s_t + \epsilon}} \cdot g_t$$


A green arrow points from the text 'gradient for step t' to the term  $g_t$  in the denominator of the update formula. A red arrow points from the term  $s_t$  in the denominator to the equation  $s_t = \gamma \cdot s_{t-1} + (1 - \gamma) \cdot g_t^2$  below it.

$$s_t = \gamma \cdot s_{t-1} + (1 - \gamma) \cdot g_t^2$$

This approach leads to the following decoupling:

$$s_t = (1 - \gamma) \cdot (g_t^2 + \gamma g_{t-1}^2 + \gamma^2 g_{t-2}^2 + \gamma^3 g_{t-3}^2 + \dots + \gamma^t g_0^2)$$



# ADAPTIVE MOMENTUM GRADIENT DESCENT

---

Consider

$$v_t = \beta_1 v_{t-1} + (1 - \beta_1) g_t$$

and

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2) g_t^2$$

We further consider

$$\hat{v}_t = \frac{v_t}{1 - \beta_1^{t+1}}, \text{ and } \hat{s}_t = \frac{s_t}{1 - \beta_2^{t+1}}$$

and

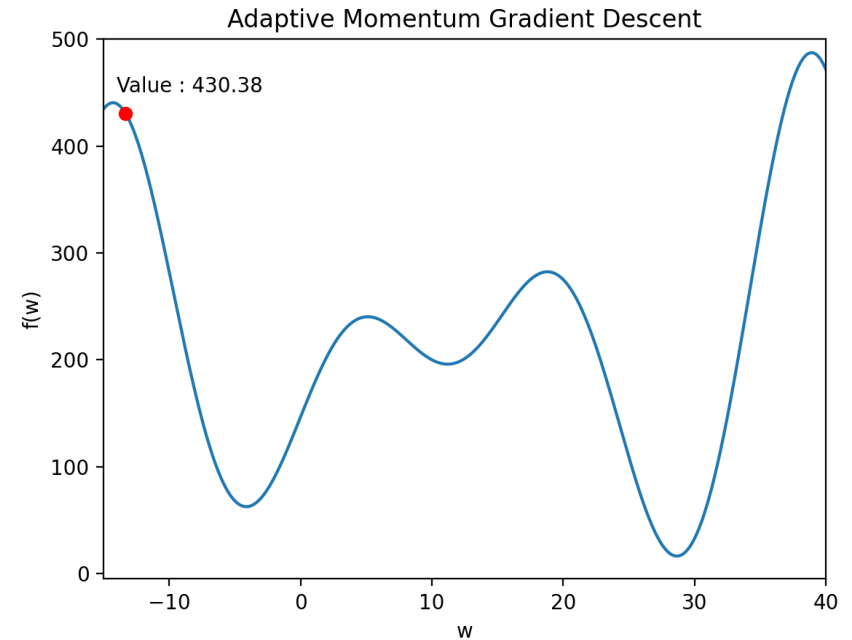
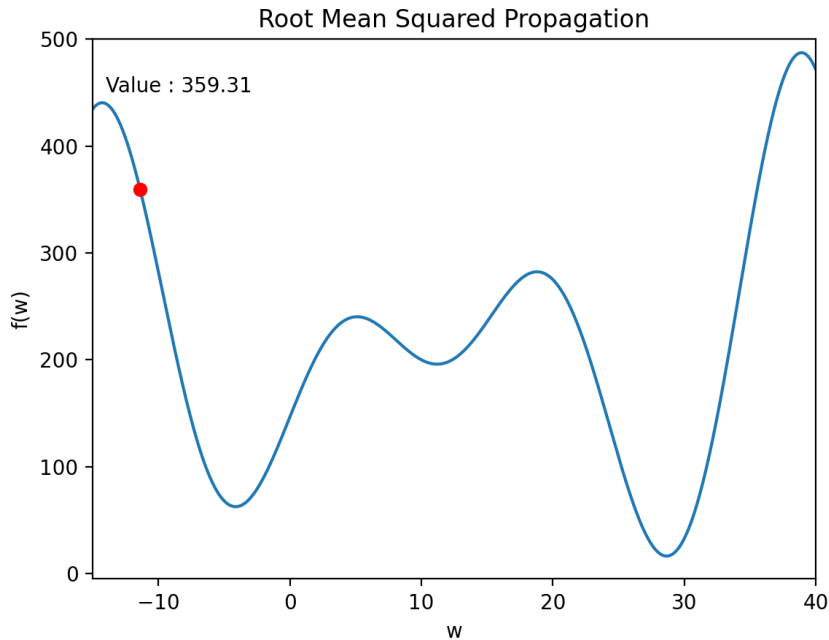
$$\hat{g}_t = \frac{\eta \cdot \hat{v}_t}{\sqrt{\hat{s}_t} + \epsilon}$$

The updates to the weights are implemented as follows

$$w_t = w_{t-1} - \hat{g}_t$$


# RMSPROP VS ADAM

To clarify, this situation is difficult because the objective function is not U-shaped; it is rather multimodal.



Do you see the difference?