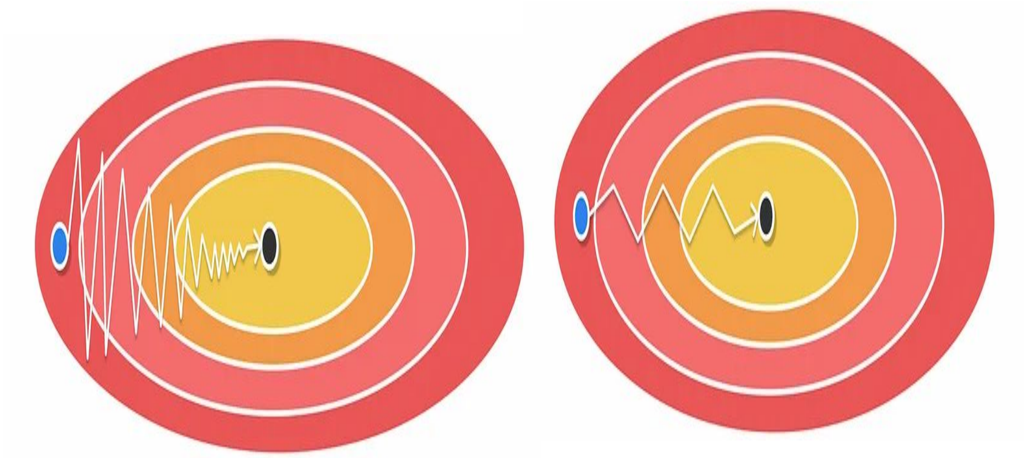
Scope, challenges, limitations and comparative Analysis of momentum, adagrad, RMSProp, Adam

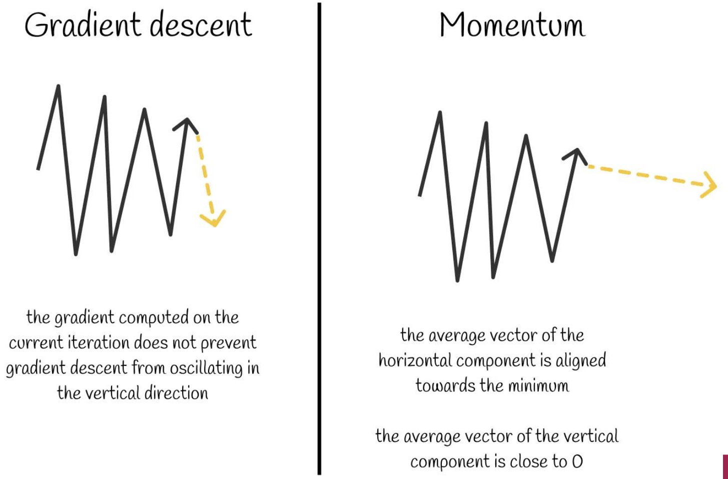
Presented By Suraj Bhattarai

Contents

Momentum

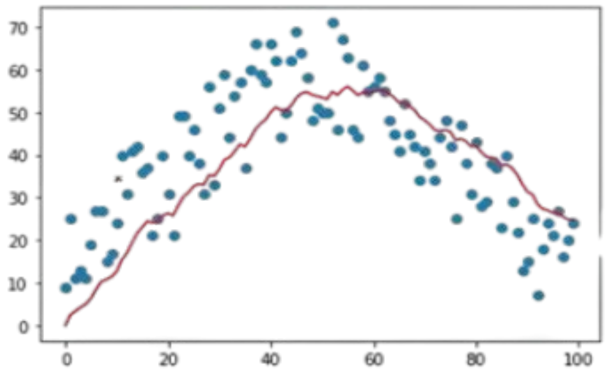
Momentum usually converges much faster than gradient descent.





continue....

$$V_0 = 0$$
$$V_1 = \beta * V_0 + (1-\beta) * \theta_1$$
$$V_2 = \beta * V_1 + (1-\beta) * \theta_2$$
$$V_3 = \beta * V_2 + (1-\beta) * \theta_3$$
$$V_4 = \beta * V_3 + (1-\beta) * \theta_4$$
$$V_5 = \beta * V_4 + (1-\beta) * \theta_5$$
$$V_6 = \beta * V_5 + (1-\beta) * \theta_6$$
$$V_7 = \beta * V_6 + (1-\beta) * \theta_7$$
$$V_8 = \beta * V_7 + (1-\beta) * \theta_8$$
$$V_9 = \beta * V_8 + (1-\beta) * \theta_9$$



Momentum continue….

Scope:

- **Convergence:** Momentum reduces oscillations and helps the optimizer converge faster towards the minimum
- **Application**: Momentum is widely used in training deep neural networks, particularly in scenarios where standard gradient descent might be slow.

Challenges:

- **Parameter Tuning** (usually denoted as **β or μ**) : Recommended value is 0.9, but the optimal value might differ depending on the problem. (Canno use **β=0 and β=1**)
- **Learning rate Sensitivity**: While momentum helps with convergence, it still requires a well-tuned learning rate.

Limitations:

- Momentum does not adapt the learning rate; hence it might still struggle with sparse data or saddle points.

2. Adagrad

The adaptive gradient descent algorithm is slightly different from other gradient descent algorithms.

This is because it uses different learning rates for each iteration.

Advantages of Adagrad:

- Adaptive learning rate
- Faster convergence
- Handling sparse data efficiently

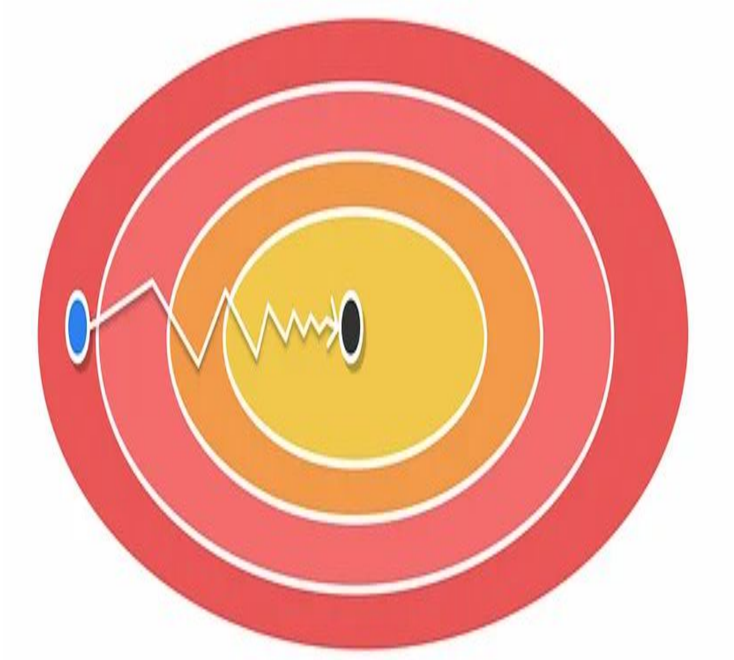
Challenges:

- Tends to perform well in the early stages of learning but might lead to too small learning rates over time, which can stall training

Challenges:

- The learning rate decays(reduction) too aggressively, leading to convergence issues in the long run.

Continue....



$$v_t = v_{t-1} + dw_t^2$$

$$w_t = w_{t-1} - \frac{\alpha}{\sqrt{v_t + \varepsilon}} dw_t$$

3. RMS Prop

RMSProp (Root Mean Square Propagation) is designed to resolve the diminishing learning rate problem of Adagrad.

Challenges:

- Requires careful tuning of hyperparameters, particularly the learning rate and the decay rate (usually denoted as β, typically set to 0.9).

Limitations:

- It is not a universal solution and might still face issues with complex, non-convex loss.

In RMSProp, it is recommended to choose β close to 1.

4 Adam Optimizer

Scope:

Combining Momentum and RMSProp: Adam incorporates the advantages of both Momentum and RMSProp by computing adaptive learning rates.

Versatility: Adam is highly versatile and is often the default choice for many deep learning tasks due to its robustness and adaptability.

Challenges

- **Hyperparameter Sensitivity**: Adam introduces additional hyperparameters (β1, β2, and ε) that need careful tuning, though default values generally work well.

Limitations

- **Complexity**: Adam is more complex to implement and tune than simpler methods like SGD or Momentum.
- **Overfitting**: Adam's adaptability can lead to overfitting if not carefully monitored, particularly on small datasets.

According to the Adam paper, good default values for hyperparameters are $\boldsymbol{\beta_1 = 0.9}$ *,* $\boldsymbol{\beta_2 = 0.999}$*,* $\boldsymbol{\varepsilon = 1e\text{-}8}$*.*

Implementation using Pythons

Thank you