# Gradient Descent Visualization and Analysis

This code provides visualizations for gradient descent optimization algorithms applied to various functions in one and two dimensions. Here's a breakdown of the implemented functions and their visualizations:

## 1. *One-Dimensional Functions:*

**Quadratic Function:**

- *Function:* $f(x) = x^2 + 3x + 8$
- *Gradient:* $f'(x) = 2x + 3$

**Observations:**

- The animation demonstrates the iterative process of gradient descent.
- The algorithm converges to the minimum of the quadratic function.
- Learning rate significantly influences convergence speed.

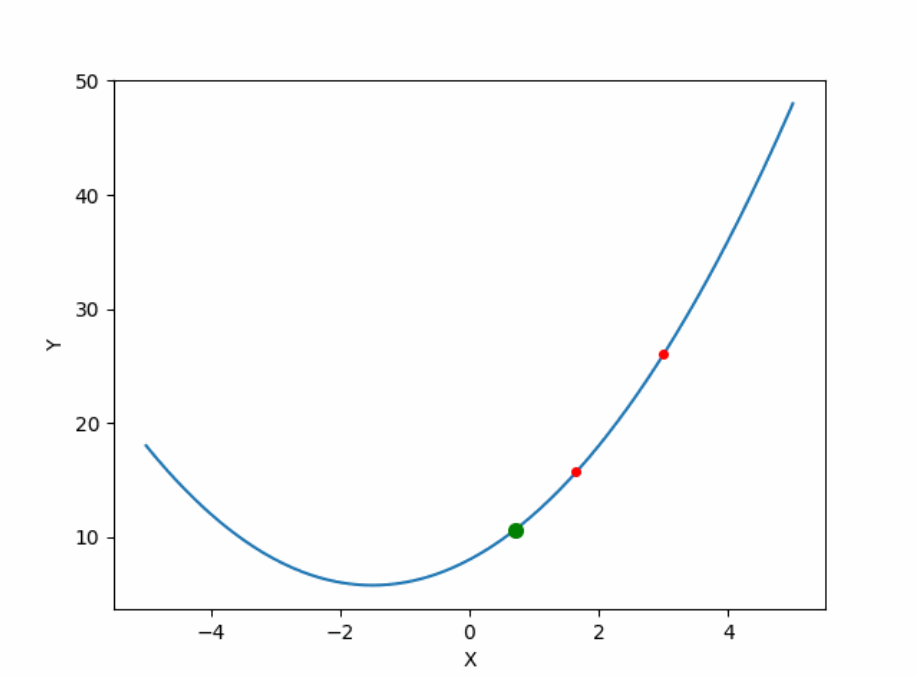


Figure 1: a1.gif

**Trigonometric Function:**

- *Function:* $f(x) = \cos^4(x) - \sin^3(x) - 4\sin^2(x) + \cos(x) + 1$

- *Gradient:* $f'(x) = -4\sin(x)(\cos^3(x)) - 3(\sin^2(x))\cos(x) - 8\sin(x)\cos(x) - \sin(x)$

**Observations:**

- The algorithm navigates through the complex landscape of the trigonometric function.
- Convergence depends on the function's oscillatory behavior and the chosen learning rate.

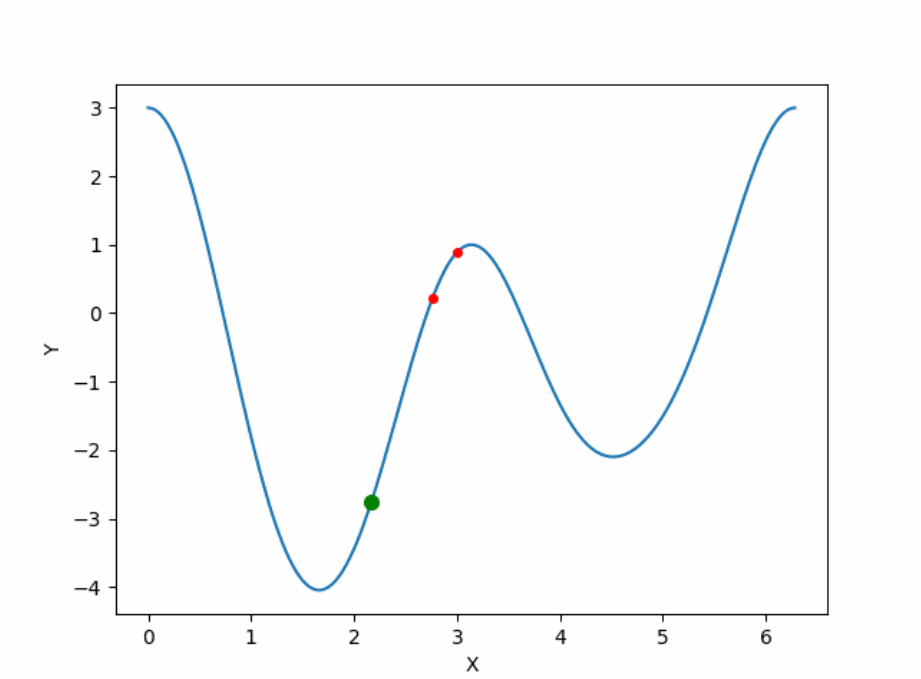


Figure 2: a4.gif

## 2. *Two-Dimensional Functions:*

**Cubic Function:**

- *Function:* $f(x, y) = x^4 - 16x^3 + 96x^2 - 256x + y^2 - 4y + 262$
- *Gradients:*
  - $f'_x(x, y) = 4x^3 - 48x^2 + 192x - 256$
  - $f'_y(x, y) = 2y - 4$

**Observations:**

- The 3D surface illustrates the function's curvature.

- The algorithm finds the minimum by adjusting both $x$ and $y$ simultaneously.
- Gradient magnitudes guide the algorithm toward optimal values.

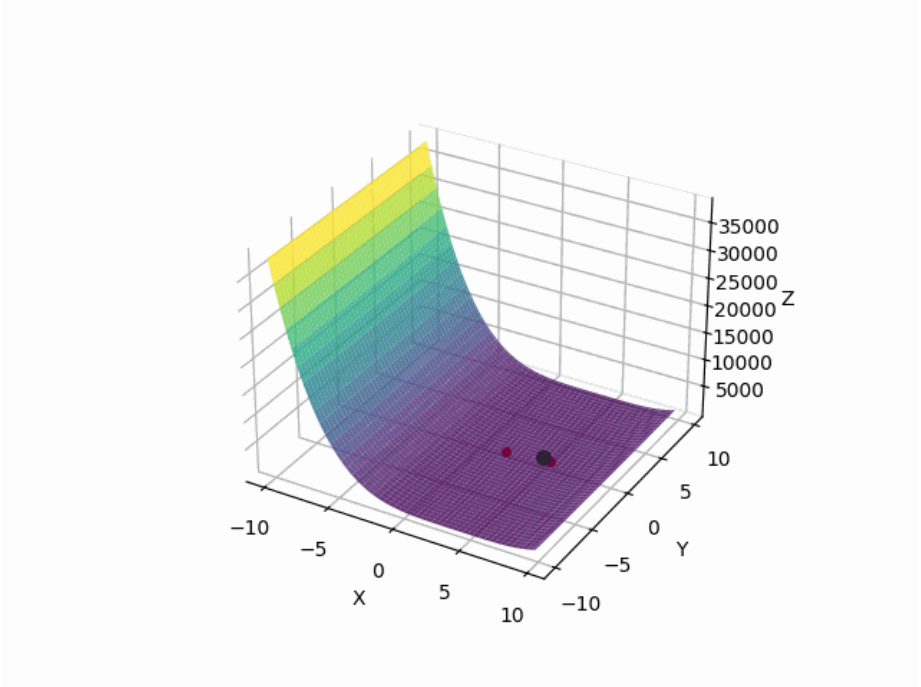


Figure 3: a2.gif

**Gaussian Function:**

- *Function:* $f(x, y) = e^{-(x-y)^2} \sin(y)$
- *Gradients:*
  - $f'_x(x, y) = -2e^{-(x-y)^2} \sin(y)(x - y)$
  - $f'_y(x, y) = e^{-(x-y)^2} \cos(y) + 2e^{-(x-y)^2} \sin(y)(x - y)$

**Observations:**

- The function's behavior is influenced by both the exponential and sinusoidal components.
- Gradient directions guide the search towards regions of lower magnitude.

---

## *General Observations:*

- *Learning Rate Impact:* The choice of learning rate significantly affects convergence. Too large a rate might cause overshooting, whereas too small

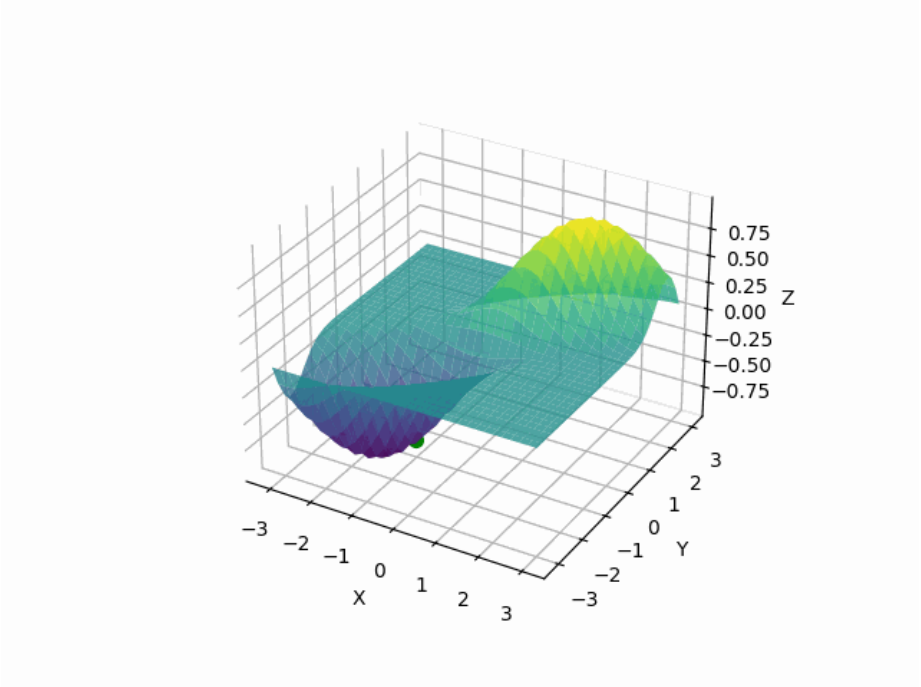


Figure 4: a3.gif

a rate leads to slow convergence.
- *Multimodal Functions:* Functions with multiple peaks and valleys present challenges for gradient descent. Convergence might depend on the algorithm's initialization.
- *Function Complexity:* The complexity of the function (in terms of oscillations, peaks, and valleys) impacts convergence speed and the likelihood of finding the global minimum.

## *Inferences:*

- *Algorithm Robustness:* Gradient descent is a versatile optimization algorithm applicable to various functions.
- *Hyperparameter Tuning:* Adjusting the learning rate is crucial. An optimal learning rate ensures faster convergence without overshooting.
- *Function Landscape:* Understanding the function's behavior aids in choosing suitable optimization algorithms. Complex functions require careful initialization and smaller learning rates.

---

This code provides a practical visualization of gradient descent algorithms, offering insights into their behavior across different functions. Understanding these visualizations is valuable for tuning hyperparameters and selecting appropriate optimization methods in real-world applications.