DL HW

Arman Khoshnevis

February 2025

1 Analytical Derivatives

The required derivatives are computed analytically in the following steps.
Step 1: Compute partial derivatives of $\theta = \tan^{-1}\left(\frac{x_2}{x_1}\right)$ and $r = \sqrt{x_1^2 + x_2^2}$

- Partial derivatives of θ:
  - $\frac{\partial \theta}{\partial x_1} = \frac{-x_2}{x_1^2+x_2^2}$
  - $\frac{\partial \theta}{\partial x_2} = \frac{x_1}{x_1^2+x_2^2}$
- Partial derivatives of r:
  - $\frac{\partial r}{\partial x_1} = \frac{x_1}{\sqrt{x_1^2+x_2^2}} = \frac{x_1}{r}$
  - $\frac{\partial r}{\partial x_2} = \frac{x_2}{\sqrt{x_1^2+x_2^2}} = \frac{x_2}{r}$

Step 2: Compute partial derivatives of $y = r^2\left(\sin^2(6\theta + 2r) + 1\right)$ using the chain rule:

- w.r.t. θ:
  - $\frac{\partial y}{\partial \theta} = 12r^2 \sin(6\theta + 2r) \cdot \cos(6\theta + 2r)$
- w.r.t. r:
  - $\frac{\partial y}{\partial r} = 2r\left(\sin^2(6\theta + 2r) + 1\right) + 4r^2 \sin(6\theta + 2r)\cos(6\theta + 2r)$

Step 3: Compute partial derivatives of y w.r.t. x_1 and x_2 using the chain rule (each of the following terms has be calculated already.):

- w.r.t. x_1:
  - $\frac{\partial y}{\partial x_1} = \frac{\partial y}{\partial \theta}\frac{\partial \theta}{\partial x_1} + \frac{\partial y}{\partial r}\frac{\partial r}{\partial x_1}$
- w.r.t. x_2:
  - $\frac{\partial y}{\partial x_2} = \frac{\partial y}{\partial \theta}\frac{\partial \theta}{\partial x_2} + \frac{\partial y}{\partial r}\frac{\partial r}{\partial x_2}$

2 Results and Discussion

Next, we present a brief discussion of the results, focusing on the impact of the learning rate and initialization.

Cases with $\lambda = 1, 0.1$

A learning rate that is too large (i.e., step size for updating x_1 and x_2) led to divergence, regardless of the initial seed values. This phenomenon is illustrated in Figure 1.

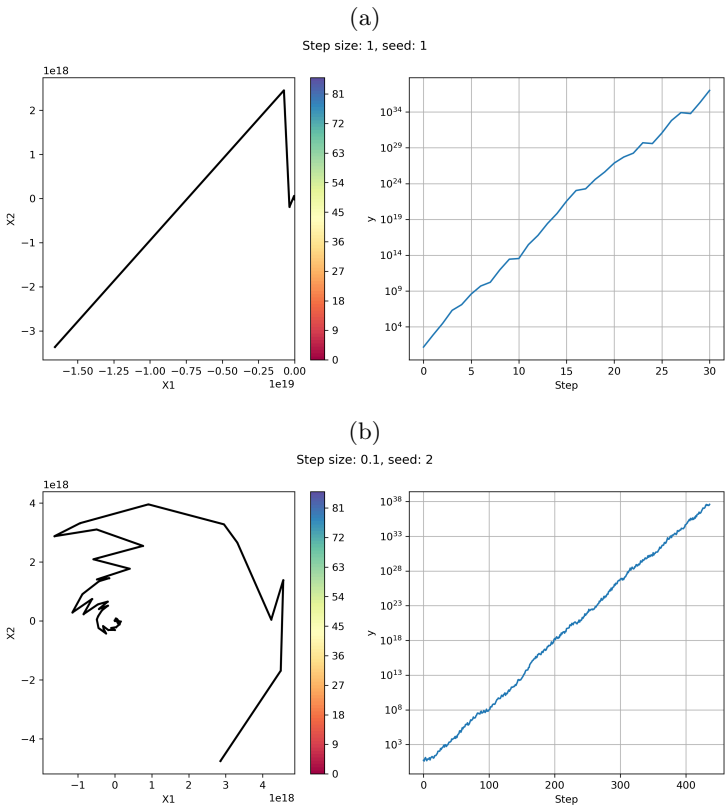

Figure 1: Divergence in the optimization results caused by excessively large step sizes: (a) $\lambda = 1$ and (b) $\lambda = 0.1$.

Cases with $\lambda = 0.01$

This learning rate appears to be well-balanced, as it is neither too large nor too small for effectively updating the parameter values. By comparing the

convergence results across different initializations, a clear dependence on the initial parameter values emerges, which can be summarized as follows:

- In some cases, the algorithm converges successfully. Depending on the initialization, the number of steps required to reach the global minimum may vary.
- In other cases, the algorithm gets trapped in a local minimum.

These two scenarios are illustrated in Figure 2.

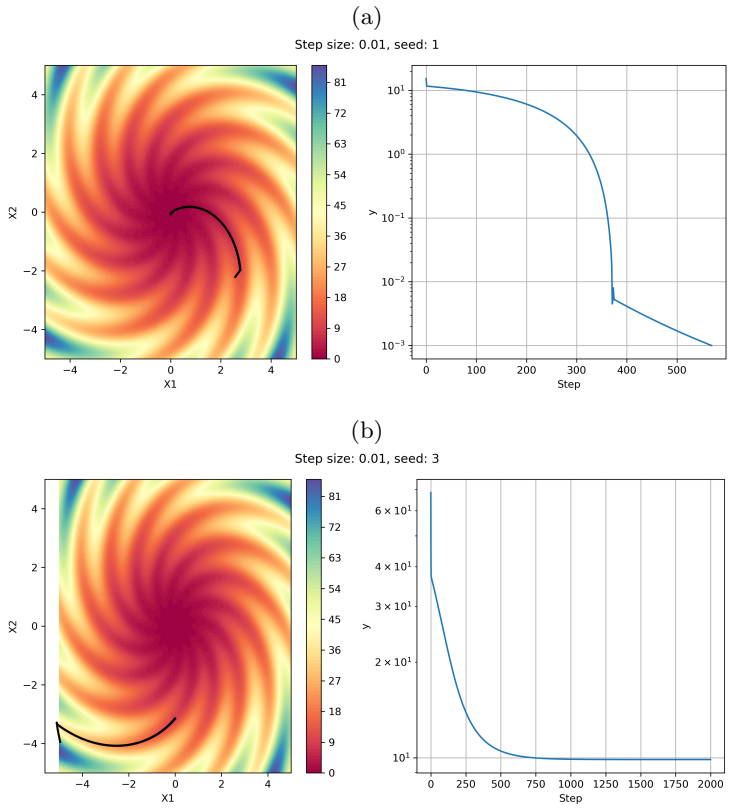

Figure 2: Two possible outcomes with a suitable learning rate: (a) successful convergence to the global minimum, and (b) getting trapped in a local minimum.

Cases with $\lambda = 0.001, 0.0001$

This learning rate appears to be too small, resulting in a slow convergence rate. Unless the initialization is sufficiently close to the global minimum, the algorithm requires significantly more steps to converge. Figure 3 illustrates this slow convergence in several cases. Notably, a similar trend was observed for

$\lambda = 0.0001$, where none of the optimizations successfully reached the global minimum.

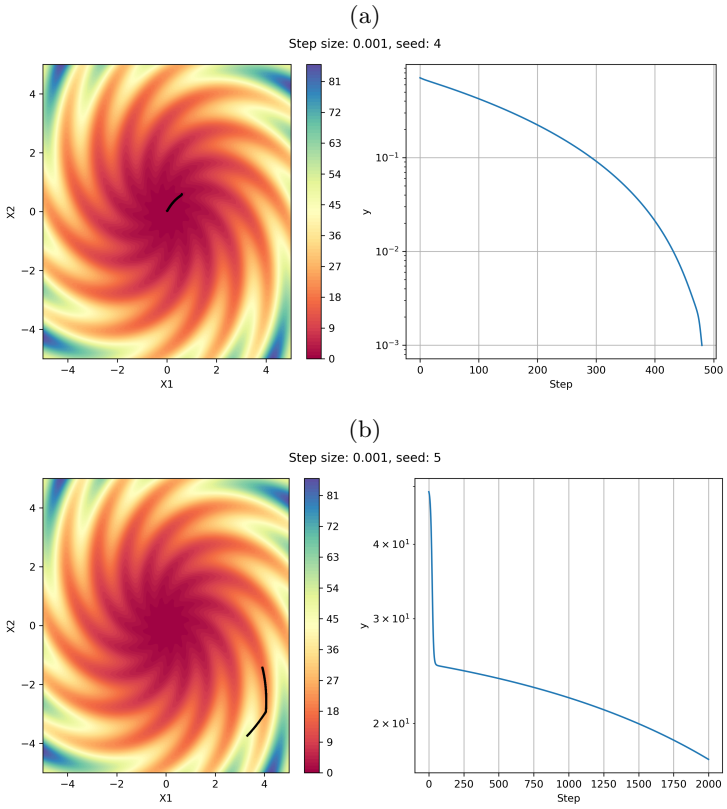

Figure 3: Two scenarios with a suitable learning rate: (a) Successful convergence to the global minimum when the initialization is close to it, and (b) requiring more steps or converging to a local minimum when the initialization is farther away.