Gradient Descent for Linear Regression with MSE and Huber Loss

Teamwork Assignment (Learning Curves and Stop Criteria)

Druv Jayantilal Patel (ASU ID: 1236080380)
Jay Sanjay Sonawane (ASU ID: 1233750832)
Jaishva Baijubhai Patel (ASU ID: 1231581457)

October 3, 2025

Abstract

We implement gradient descent (GD) for univariate linear regression under two loss functions: mean squared error (MSE) and Huber. We compare three learning rates $\eta \in \{0.01, 0.05, 0.2\}$, apply two stopping criteria (parameter-change tolerance and loss-change tolerance), and visualize efficient learning curves (one figure per loss with all η overlaid). We validate GD (MSE) against the least-squares closed form and discuss convergence behavior and robustness.

1 Dataset and Design Matrix

The provided Excel sheet (*Teamwork-regress w 2 losses (dataset)-1.xlsx*) supplies pairs (x_i, y_i) *(the same values are also provided as a CSV for convenience).* We fit a line $\hat{y} = wx + b$ using the design matrix

$$X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}, \qquad \theta = \begin{bmatrix} w \\ b \end{bmatrix}.$$

2 Gradient Descent Derivations

Notation. For MSE we set $e_i = \hat{y}_i - y_i$; for Huber we set $r_i = y_i - \hat{y}_i$. Note $e_i = -r_i$.

2.1 MSE Loss (matches code)

Using $e_i = \hat{y}_i - y_i = (wx_i + b) - y_i$, our code defines

$$L_{\text{MSE}}(\theta) = \frac{1}{n} \sum_{i=1}^{n} e_i^2,$$

so the gradients are

$$\nabla_w L_{\text{MSE}} = \frac{2}{n} \sum_{i=1}^{n} e_i x_i, \qquad \nabla_b L_{\text{MSE}} = \frac{2}{n} \sum_{i=1}^{n} e_i,$$

and GD updates are

$$w \leftarrow w - \eta \frac{2}{n} \sum_i e_i x_i, \qquad b \leftarrow b - \eta \frac{2}{n} \sum_i e_i.$$

2.2 Huber Loss

Let $r_i = y_i - \hat{y}_i$ and choose $\delta > 0$ (we use $\delta = 0.5$). The per-sample Huber loss is

$$\ell_\delta(r) = \begin{cases} \frac{1}{2}r^2, & |r| \le \delta, \\ \delta\left(|r| - \frac{1}{2}\delta\right), & |r| > \delta. \end{cases}$$

Its derivative w.r.t. r is

$$\frac{\partial \ell_\delta}{\partial r} = \begin{cases} r, & |r| \le \delta, \\ \delta\,\mathrm{sign}(r), & |r| > \delta. \end{cases}$$

With $r_i = y_i - (wx_i + b)$, by chain rule $\frac{\partial r_i}{\partial w} = -x_i$ and $\frac{\partial r_i}{\partial b} = -1$, hence

$$\nabla_w L_{\text{Huber}} = -\frac{1}{n}\sum_{i=1}^{n} g_i x_i, \qquad \nabla_b L_{\text{Huber}} = -\frac{1}{n}\sum_{i=1}^{n} g_i,$$

where $g_i = \frac{\partial \ell_\delta}{\partial r}\big|_{r=r_i}$. The GD update (subtracting the gradient) becomes

$$w \leftarrow w + \eta\,\frac{1}{n}\sum_i g_i x_i, \qquad b \leftarrow b + \eta\,\frac{1}{n}\sum_i g_i.$$

3 Implementation Details

- Environment: `pandas 2.3.2`, `openpyxl 3.1.5`, `numpy`, `matplotlib`.
- Learning rates: $\eta \in \{0.01, 0.05, 0.2\}$.
- Stopping criteria:
  1. **Parameter-change tolerance** $\|\theta_t - \theta_{t-1}\|_2 \le$ `tol_param` with `tol_param` $= 10^{-6}$.
  2. **Loss-change tolerance** $|L_t - L_{t-1}| \le$ `tol_loss` with `tol_loss` $= 10^{-8}$.
- Maximum iterations: `max_iters` $= 3000$.
- Huber parameter: $\delta = 0.5$.
- Validation: Compare GD(MSE) θ with least-squares solution from `np.linalg.lstsq`.

Termination criteria and rationale

Table 1: Termination criteria and thresholds

Criterion	Definition	Value / Rationale
Loss change	$\|L_t - L_{t-1}\| < \varepsilon_L$	$\varepsilon_L = 10^{-8}$ (objective is flat)
Param change	$\|\theta_t - \theta_{t-1}\|_2 < \varepsilon_P$	$\varepsilon_P = 10^{-6}$ (weights stable)

4 Results

Tables 2 and 3 summarize the printed training outputs.[1]

[1] "stop(param)" or "stop(loss)" show which criterion triggered; "None" means the run ended via the max-iteration cap.

Table 2: MSE training results (updated)

Loss	η	$\theta = [w, b]$	final_loss	stop(param)	stop(loss)	iters
MSE	0.01	[0.7266, 0.4607]	0.011105	None	None	3000
MSE	0.05	[0.7333, 0.4575]	0.011101	**1172**	**721**	1173
MSE	0.20	[0.7334, 0.4575]	0.011101	**340**	**204**	341

Table 3: Huber training results ($\delta = 0.5$; updated)

Loss	η	$\theta = [w, b]$	final_loss	stop(param)	stop(loss)	iters
Huber	0.01	[0.6768, 0.4844]	0.005686	None	None	3000
Huber	0.05	[0.7331, 0.4576]	0.005551	**2162**	**1259**	2163
Huber	0.20	[0.7333, 0.4575]	0.005551	**639**	**365**	640

Least-squares check. Closed-form (LS) parameters:

$$\theta_{\text{LS}} = [0.73340398,\ 0.45748571]^\top$$

GD(MSE) with $\eta = 0.05$ returned

$$\theta_{\text{GD}} = [0.73327417,\ 0.45754744]^\top,$$

and the norm difference was

$$\|\theta_{\text{GD}} - \theta_{\text{LS}}\|_2 \approx 1.44 \times 10^{-4},$$

confirming correctness.

5 Learning Curves

To present *efficient* learning curves, we overlay the three learning-rate traces in one figure per loss and mark the stop events ("o" for parameter tolerance, "x" for loss tolerance).

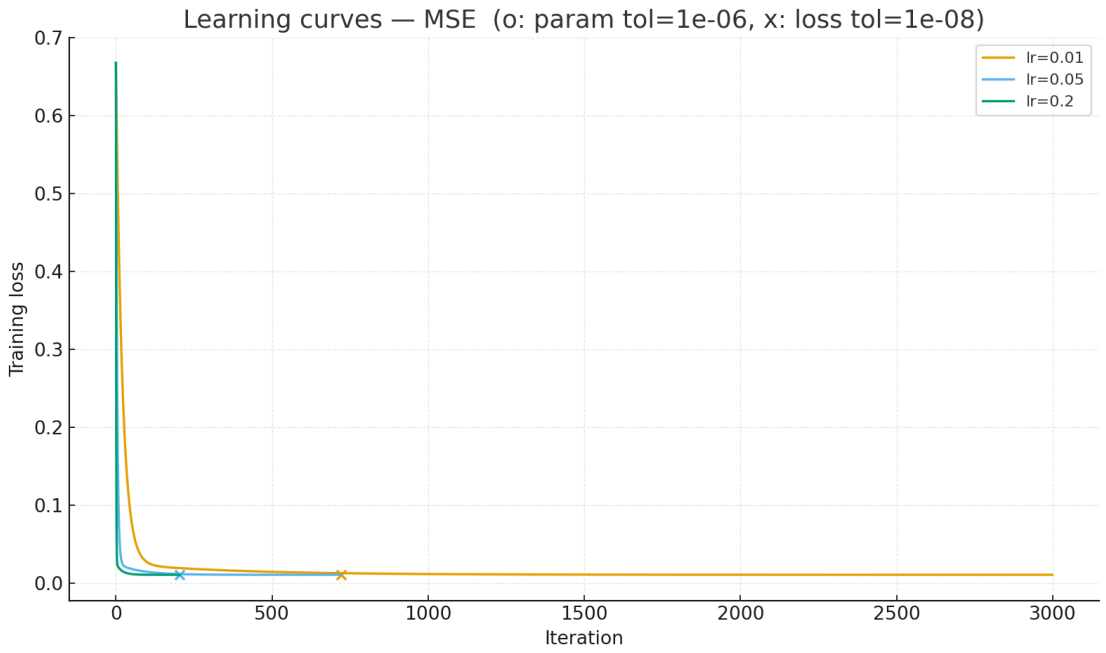

Figure 1: Learning curves—MSE (o: param tol, x: loss tol). For η=0.2, loss-tol (×) occurs before param-tol ().

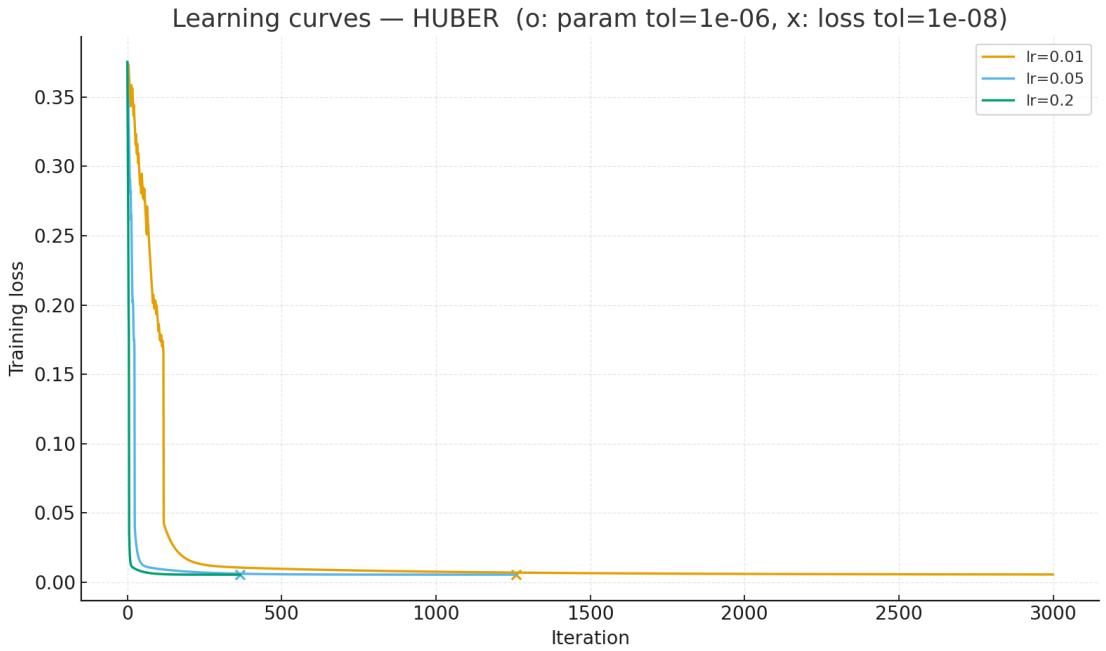

Figure 2: Learning curves—Huber (o: param tol, x: loss tol). For η=0.2, loss-tol (×) occurs before param-tol ().

6 Discussion

- **Effect of learning rate.** Larger η substantially reduces iteration count. For both losses, $\eta = 0.2$ converged fastest; $\eta = 0.01$ was slow and hit the max-iteration cap.
- **Stop criteria.** In this run, *both* criteria were reached for $\eta \in \{0.05, 0.2\}$. The *loss* tolerance triggered earlier (721/204 iterations for MSE; 1259/365 for Huber), and the *parameter* tolerance triggered later (1172/340 for MSE; 2162/639 for Huber), indicating the objective flattened before the parameters fully stabilized.
- **MSE vs. Huber.** Huber achieved a slightly lower training loss and shows smooth early-iteration behavior at larger η, consistent with robustness to outliers.
- **Correctness.** GD(MSE) parameters closely match least squares, with $\|\theta_{\text{GD}} - \theta_{\text{LS}}\|_2 \approx 1.44 \times 10^{-4}$.

Learning-rate taxonomy: η=0.01 (conservative/slow), η=0.05 (efficient), η=0.2 (aggressive but stable on this dataset).

7 Reproducibility and Code

Code, notebook, and figures are available at:

GitHub: https://github.com/Jay-Sonawane2712000/EEE_511_1233750832_HW_1

To run: `pip install -r requirements.txt` and execute the notebook or script to produce Figures 1–2 and the tables above.

Conclusion

GD converges reliably for both MSE and Huber on the provided dataset. Higher learning rates reduce iteration counts without overshooting here, while the Huber loss yields slightly better fit under potential outliers. The agreement with least-squares verifies the implementation.