HW-1

Q.1 Polynomial Regression as linear least squares.

(1)

$$\hat{b} = \arg_b \min \|Xb - y\|_2^2$$

$X \rightarrow$ design matrix of size $N \times (d+1)$

$y \rightarrow$ response vector

$b \rightarrow$ parameter vector.

~~Loo~~ Loss Function -

$$\|Xb - y\|_2^2 = (Xb-y)^T(Xb-y).$$

$$= (Xb)^T Xb - 2y^T Xb + y^T y.$$

To minimizer, taking derivative of the loss function wrt b.

$$\frac{\partial}{\partial b} = \left((Xb)^T Xb - 2y^T Xb + y^T y\right).$$

$$\therefore \quad \frac{\partial}{\partial b} = \cancel{2x} + \boxed{2X^T Xb - 2X^T y.}$$

To minimize the loss fn,

Let $2X^T Xb - 2X^T y = 0$

$$\therefore \quad \boxed{2X^T Xb = 2X^T y}$$

To solve for $\hat{b}$,
Let us assume that X^TX is invertible.

$$\boxed{\hat{b} = (X^TX)^{-1}X^Ty.}$$

Therefore, empirical risk minimizer $\hat{b}$ is given by

$$\boxed{\hat{b} = (X^TX)^{-1}X^Ty}$$

---

(2)

Given Loss Function -

$$L(b) = \|Xb - y\|_2^2 = (Xb-y)^T(Xb-y).$$

$$\therefore \quad L(b) = (Xb)^TXb - 2y^TXb + y^Ty.$$

$$\frac{\partial L(b)}{\partial b} = 2X^TXb - 2X^Ty$$

Simplifying,

$$\boxed{X^TXb = X^Ty}$$

Now, if X^TX is invertible,

$$\hat{b} = (X^TX)^{-1}X^Ty$$

Need for cond^n $N > d$ & X being full rank.

condition 1 : $(N > d)$ (more samples less features).

If $N > d$, it means that there are more samples a than features. This ensures that matrix X^TX is at least square $[(d+1) \times (d+1)]$ & hence potentially invertible.

If $N \leq d$, X^TX would not be invertible because it would be singular and we wouldn't be able to compute the inverse.

condition 2 - X being full rank.

X being full rank means that the columns of X are ~~really~~ linearly dependent. which causes X^TX to be singular.

In this case, normal equation $X^TXb = X^Ty$ would not have a unique solution.

Ridge Regression -

Q. 11]

Objective Function -

$$J_\lambda\theta = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x_i) - y_i)^2 + \lambda\theta^T\theta$$

$h_\theta(x_i) = \theta^T x$ (hypothesis Function)

$m \rightarrow$ number of training Examples.

$\lambda \rightarrow$ Regularization Parameter

Converting it in a matrix form,

$$J_\lambda\theta = \frac{1}{2m}\|X\theta - y\|_2^2 + \frac{\lambda}{2m}\|\theta\|_2^2$$

where,

X is the design matrix $(m \times n)$

y is the target vector $(m \times 1)$

θ is the parameter vector $(n \times 1)$.

Gradient of the loss term,

$$\nabla\theta\left[\left(\frac{1}{2m}\right)\|X\theta - y\|_2^2\right] = \left(\frac{1}{m}\right)X^T(X\theta - y)$$

Gradient of regularization form,

$$\nabla\theta\left[\left(\frac{\lambda}{2m}\right)\|\theta\|_2^2\right] = \left(\frac{\lambda}{2m}\right)\theta$$

Full gradient,

$$\nabla\theta J\lambda(\theta) = \frac{1}{m}X^T(X\theta - y) + \frac{\lambda}{m}\theta.$$

Gradient descent update rule,

$$\theta = \theta - \alpha\nabla\theta J_\lambda(\theta).$$

sub. the value of gradient,

$$\theta = \theta - \alpha\left[\left(\frac{1}{m}\right)X^T(X\theta - y) + \left(\frac{\lambda}{m}\right)\theta\right].$$

$$= \theta - \frac{\alpha}{m}\left[X^T(X\theta - y) + \left(\frac{\theta}{m}\right)\right]$$

Thus,

Gradient $\rightarrow$ $\nabla\theta J_\lambda(\theta) = \left(\frac{1}{m}\right)X^T(X\theta - y) + \left(\frac{\lambda}{m}\right)\theta.$

Update Rule $\rightarrow$ $\theta - \left(\frac{\alpha}{m}\right)\left[X^T(X\theta - y) + \lambda\theta\right].$

3. Image classification with regularized logistic regression.

(16)

→ Given function for Logistic Loss for Binary classification:

$$L(\theta, b) = \frac{1}{m} \sum_{i=1}^{m} \log\left(1 + e^{-y_i h_{\theta,b}(x_i)}\right).$$

where,

$y_i \in \{-1, 1\}$, $\hat{y}_i = h_{\theta,b}(x_i) = \theta^T x_i + b$

The margin m_i for a given sample i is

$$m_i = y_i h_{\theta,b}(x_i) = y_i(\theta^T x_i + b).$$

Now,

Logistic loss in terms of margin

$$L(\theta, b) = \frac{1}{m} \sum_{i=1}^{m} \log\left(1 + e^{-m_i}\right).$$

where $m_i = y_i h_{\theta,b}(x_i)$.

We can write the objective f^ in terms of $y_i \in \{-1, 1\}$ as

$$L(\theta, b) = \frac{1}{m} \sum_{i=1}^{m} [y_i \log(1 + e^{-h_{\theta,b}(x_i)}) + (1 - y_i) \log(1 + e^{-h_{\theta,b}(x_i)})$$

Rewriting this using $y_i \in \{-1, 1\}$:

$(1 + y_i)/2 = 1$ if $y_i = 1$ and

if $y_i = 0$

Also,

$(1 - y_i)/2 = 1$ if $y_i = -1$ and $y_i = 1$.

Using this, we can write the final objective function as

$$L(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [(1 + y_i) \log(1 + e^{-h_{\theta,b}(x_i)}) + (1 - y_i) \log(1 + e^{-h_{\theta,b}(x_i)})].$$

(17)

→ We know that,

The logistic loss function for binary classification

$$L(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left[(1+y_i) \log\left(1+e^{-h_{\theta,b}(x_i)}\right) + (1-y_i) \log\left(1+e^{-h_{\theta,b}(x_i)}\right) \right].$$

for labels $y_i \in \{-1, 1\}$.

With l_1 regularization, it becomes:

$$L_{regularized}(\theta) = L(\theta) + \alpha \|\theta\|_1$$

where,

$\alpha \rightarrow$ Regularization parameter. A higher α increases the regularization effect.

$\|\theta\|_1 \rightarrow$ sum of the absolute values of co-efficients of θ.

After regularizing the coefficients of θ with l_1 penalty using reg. param α:

$$L(\theta)_{reg.} = \frac{1}{2m}\left[(1+y_i)\log\left(1+e^{-h_{\theta,b}(x_i)}\right)+(1-y_i)\log\left(1+e^{-h_{\theta,b}(x_i)}\right)\right]$$

$$+ \alpha\|\theta\|_1 .$$

After L1 regularization,

① Some coefficients in θ may be driven exactly to zero.

② The loss function effectively performs feature selection by eliminating less important features.

③ Increasing α can potentially reduce model accuracy.

(20)

→ Sources of Randomness -

(i) Sampling of data -

Each time the experiment is repeated, a different random subset of training data is selected. Since we are sub-sampling the dataset to a smaller number of experiments, selected data point in each repetition vary. This causes randomness because different subsets of the data might represent slightly different distributions, affecting the model's performance.

(ii) Random Initialization -

The SGDC classifier uses scholastic gradient descent to optimize model parameters. During training process, there is inherent randomness in how model weights are initialized. & how the algorithm proceeds with updates. Repeating the experiment multiple times & averaging over results helps mitigate the effect of this randomness.

(21)

→ The optimal value of the regularization parameter α is the one that minimizes the mean test classification error.

In our experiment, $\boxed{0.001025}$ would ~~bet~~ be the optimal value for α after conducting for the values ranging from 10^{-4} to 10^{-1}.

This value produced the lowest mean test classification error across 10 repeated experiments.

---

(23)

Pattern in θ -

(i) Sparsity - Higher α leads to sparser θ, with many co-efficients near zero due to L1 regularization.

(ii) Feature selection - Lower α uses more features, while higher α keeps only the most important ones.

(iii) Smaller α allows larger coefficients, with larger α shrinks them.

(iv) With low α, areas corresponding to digits shapes have higher coefficients.

Regularization Effects -

(i) Increasing α produces simpler, sparser models to prevent overfitting.

(ii) Bias - Variance Relation -

Smaller α results in models with lower bias but higher variance, leading to overfitting.

Larger α results in higher bias but lower variance. which can cause in underfitting if the α is too large.

(iii) If the α is smaller, the plot results in a smoother pattern.
But if the α is larger, the plot results in a blocked pattern.

Linear Regression -

Q.2.

→ Given -

Hypothesis of linear function,

$h_\theta : R^d \rightarrow R$

where

$$\boxed{h_\theta(x) = \theta^T x.}$$

for $\theta, x \in R^d$

Avg. square loss function,

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

· Now,

$X \in R^{m \times (d+1)}$ be design matrix

$y = (y_1, \cdots y_m)^T \in R^{m \times 1}$ be response vector.

Let $\theta \in R^{(d+1) \times 1}$ be parameter vector.

The objective function in matrix/vector:

$$J(\theta) = \frac{1}{m} \| X\theta - y \|_2^2$$

$X\theta$ → vector of predictions for all training examples

$||\ ||_2^2$ → sq. Euclidean form which sums the sq. of elements of vector

y → vector of actual outputs.

$||X\theta - y||_2^2$ → it represents the sum. of sq. differences, hence we don't need an explicit summation.

Thus, final matrix form is

$$\boxed{J(\theta) = \frac{1}{m}(X\theta - y)^T(X\theta - y)}$$

Q. 3

→ Matrix form of $J(\theta)$:

$$J(\theta) = \frac{1}{m}(X\theta - y)^T(X\theta - y).$$

Expanding,

$$J(\theta) = \frac{1}{m}(X^T\theta^T - y^T)(X\theta - y)$$

$$= \frac{1}{m}(\theta^T X^T X\theta - \theta^T X^T y - y^T X\theta + y^T y)$$

Since, $\theta^T X^T y$ and $y^T X \theta$ are scalars & equal to their own transpose, they are equal.

By the rules of matrix calculus,

$$\nabla\theta(\theta^T A\theta) = (A + A^T)\theta$$
$$\nabla\theta(B^T\theta) = B$$
$$\nabla\theta(\theta^T B) = B$$

Applying these & chain rule,

$$\nabla\theta J(\theta) = \frac{1}{m}[2X^T X\theta - 2X^T y]$$

$$\boxed{\nabla\theta J(\theta) = \frac{2}{m} X^T[X\theta - y]}$$

Q. 4.

→ The general update rule for gradient descent :

$$\theta : \theta - \eta \nabla_\theta J(\theta)$$

where,

θ → gleneraah current parameter vector.

η → learning rate

$\nabla_\theta J(\theta)$ → gradient wrt θ

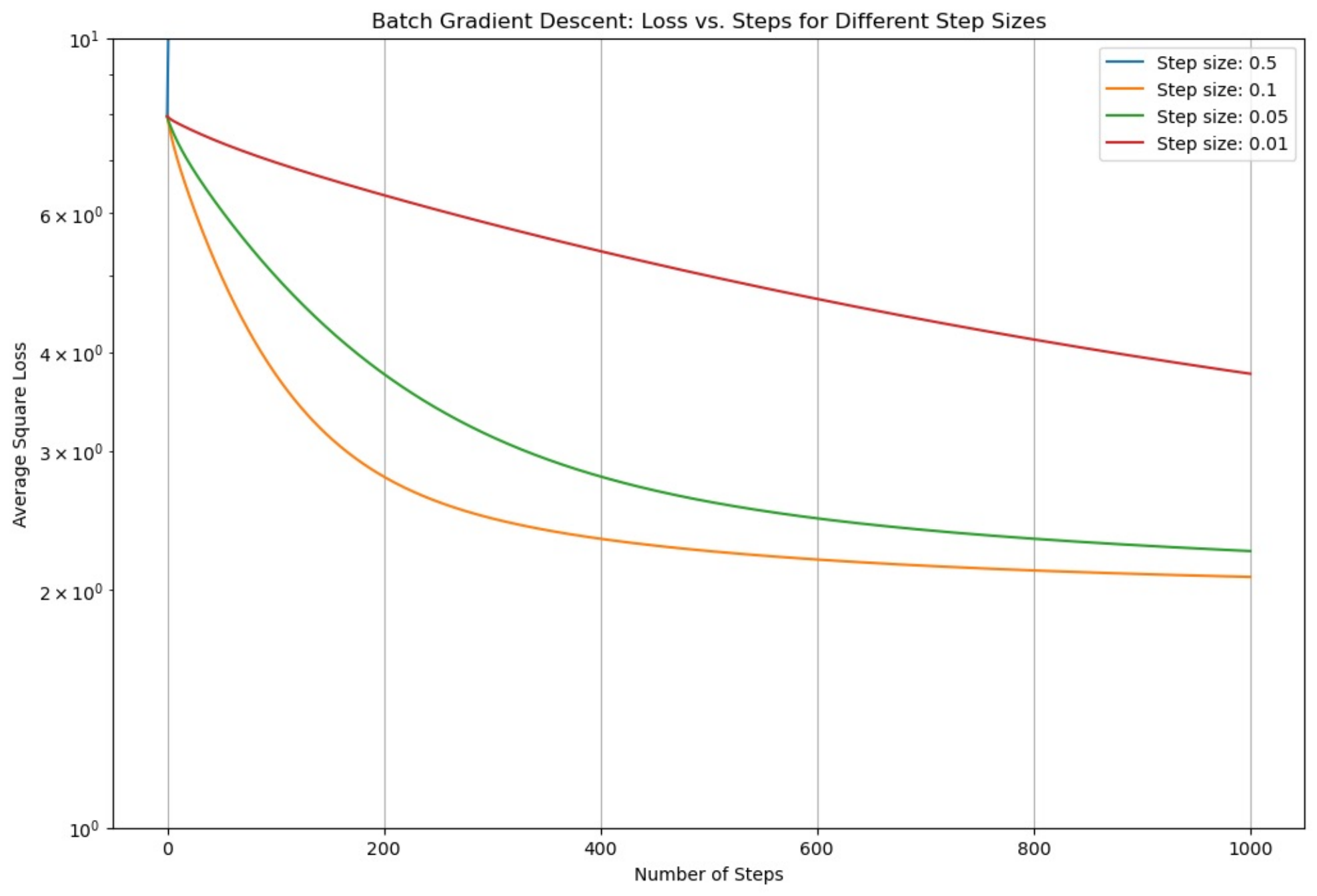
We have calculated the expression of gradient,

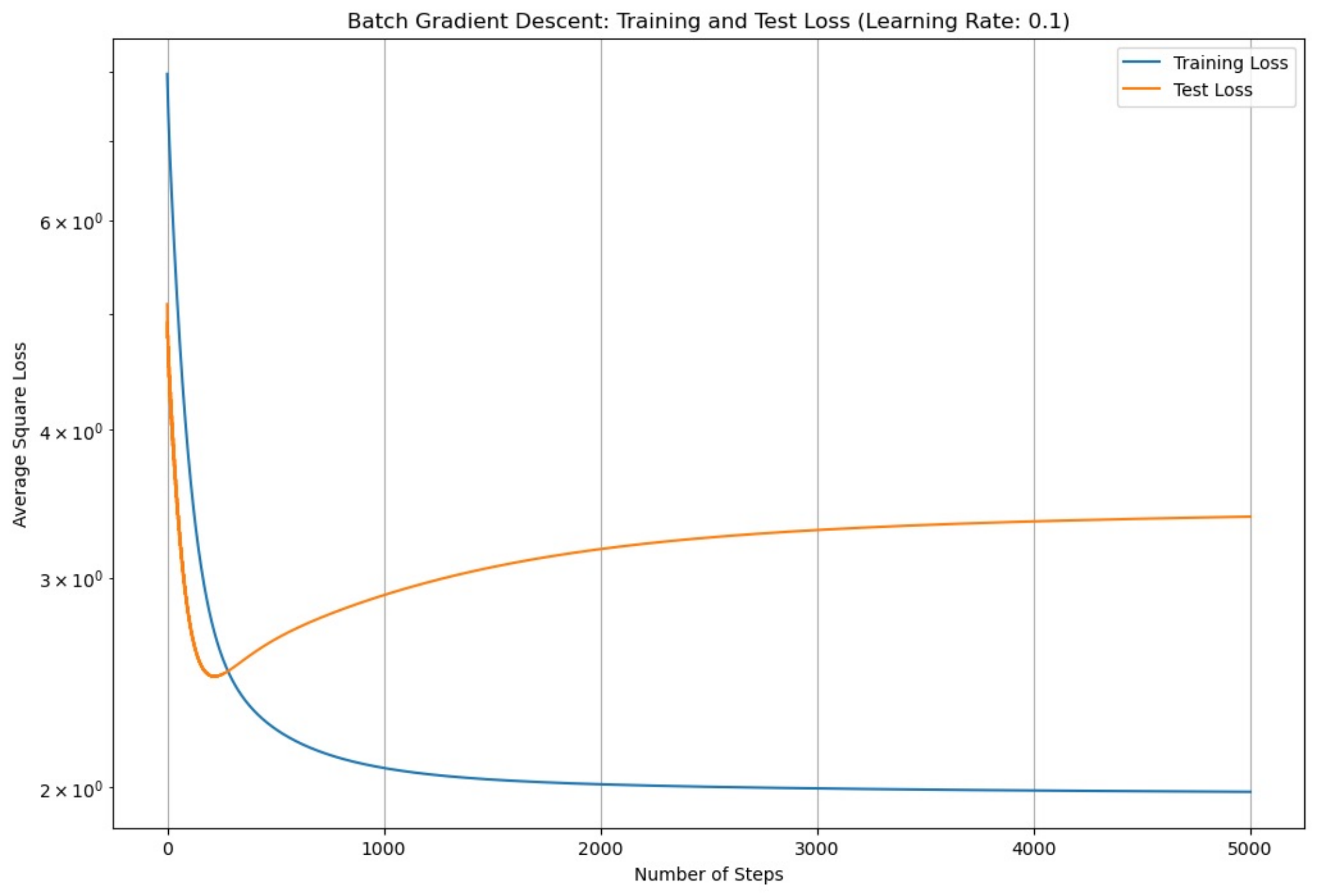
$$\nabla_\theta J(\theta) = \frac{2}{m} X^T(X\theta - y)$$

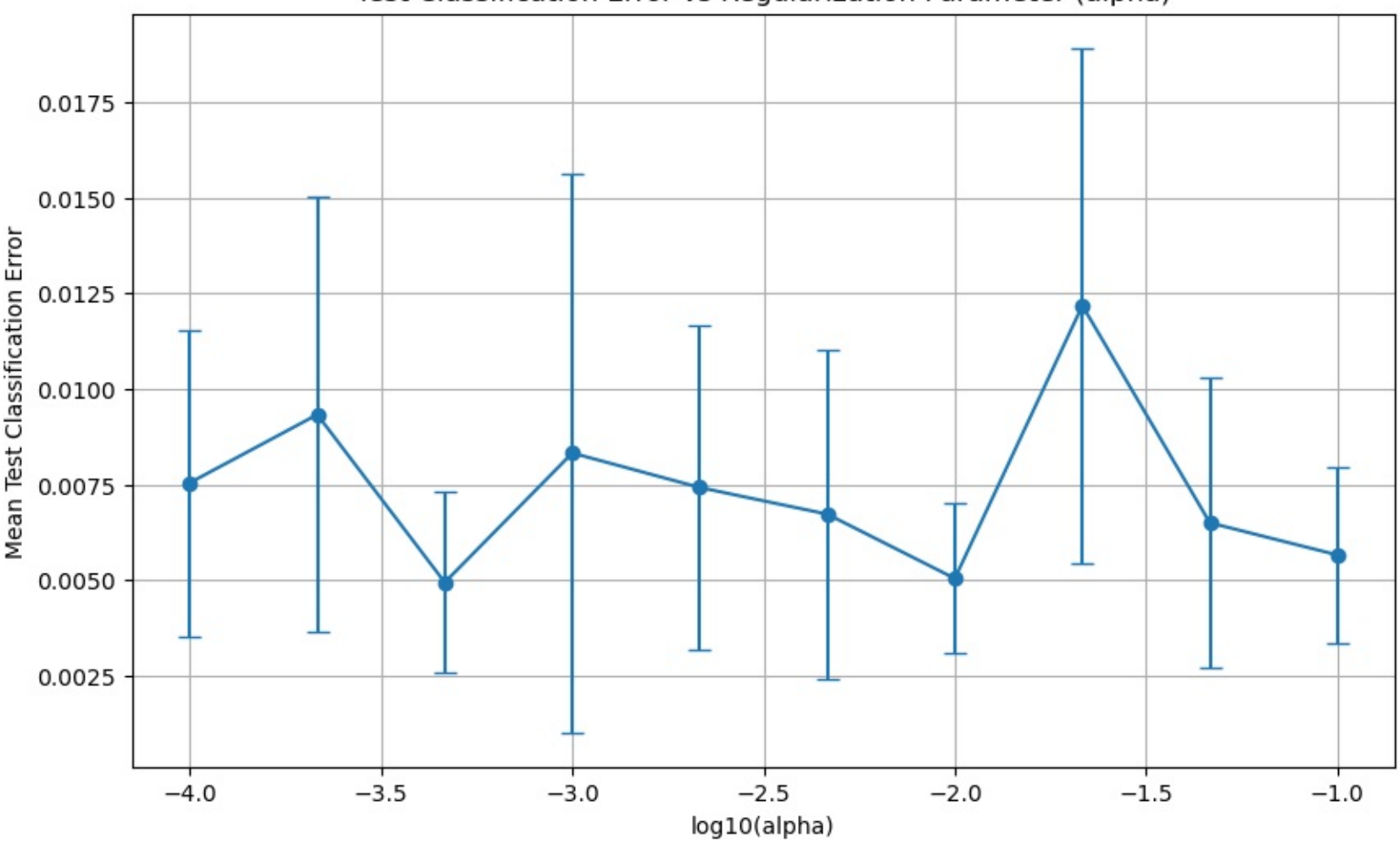
substituting,

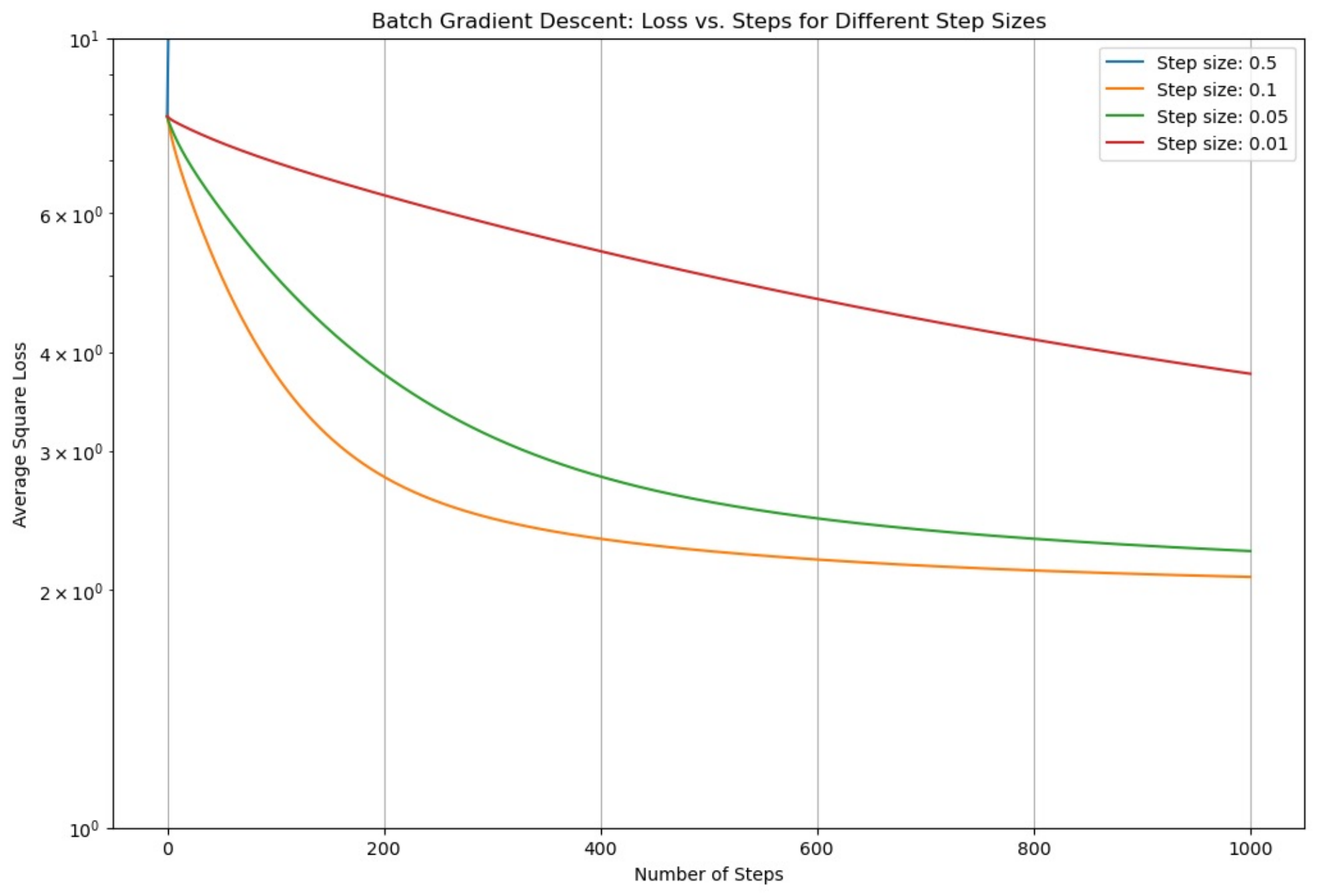
$$\theta\_new = \theta\_old - \eta^* \left(\frac{2}{m}\right) X^T(X\theta\_old - y).$$

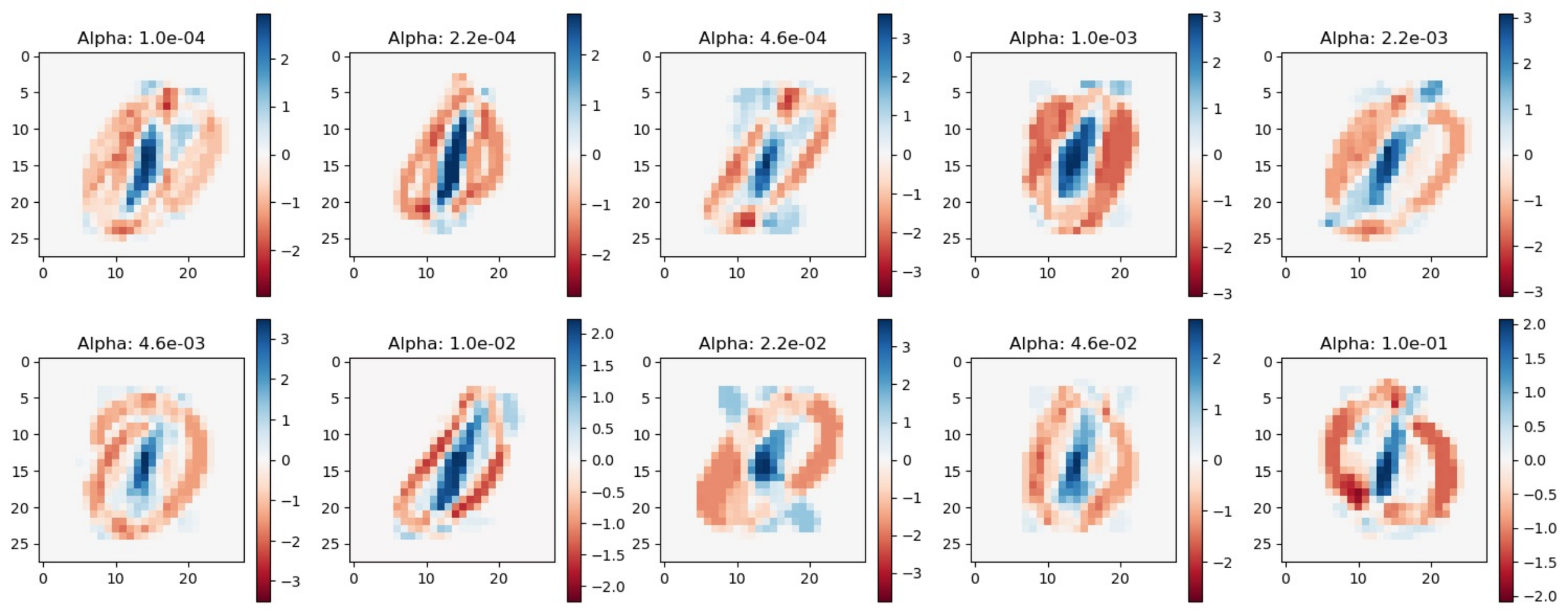
$$\therefore \quad \theta\_new = \theta\_old - \left(\frac{2\eta}{m}\right) X^T(X\theta\_old - y).$$


Batch Gradient Descent: Loss vs. Steps for Different Step Sizes
Step size: 0.5
Step size: 0.1
Step size: 0.05
Step size: 0.01
Average Square Loss
Number of Steps
10^1
6×10^0
4×10^0
3×10^0
2×10^0
10^0
0
200
400
600
800
1000


Batch Gradient Descent: Training and Test Loss (Learning Rate: 0.1)
Training Loss
Test Loss
Average Square Loss
6×10^0
4×10^0
3×10^0
2×10^0
0
1000
2000
3000
4000
5000
Number of Steps


Test Classification Error vs Regularization Parameter (alpha)
Mean Test Classification Error
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
−4.0
−3.5
−3.0
−2.5
−2.0
−1.5
−1.0
log10(alpha)


Batch Gradient Descent: Loss vs. Steps for Different Step Sizes
Step size: 0.5
Step size: 0.1
Step size: 0.05
Step size: 0.01
Average Square Loss
Number of Steps
10^1
6×10^0
4×10^0
3×10^0
2×10^0
10^0
0
200
400
600
800
1000


Alpha: 1.0e-04
Alpha: 2.2e-04
Alpha: 4.6e-04
Alpha: 1.0e-03
Alpha: 2.2e-03
Alpha: 4.6e-03
Alpha: 1.0e-02
Alpha: 2.2e-02
Alpha: 4.6e-02
Alpha: 1.0e-01