ECE421: Introduction to Machine Learning — Fall 2024

Assignment 2: Gradient Descent, Multiclass Logistic Regression, and K-Means

Due Date: Friday, October 18, 11:59 PM

General Notes

1. Programming assignments can be done in groups of up to 2 students. Students can be in different sections.
2. Only one submission from a group member is required.
3. Group members will receive the same grade.
4. Please post assignment-related questions on Piazza.

Turning It In

You need to submit your version of the following files:

- `myTorch.py`
- `PA2_qa.pdf` that answer questions related to the implementations.
- The cover file with your name and student ID filled (it can be as the first page of your `PA2_qa.pdf` or as a separate PDF file.)

Please pack them into a single folder, compress into a zip file and name it as PA2.zip. Please submit the zip file to Quercus.

Group Members

Name (and Name on Quercus)	UTORid
Edwin Chacko	chackoed
Bala Kannan Murali	muralib1

1 Gradient Descent

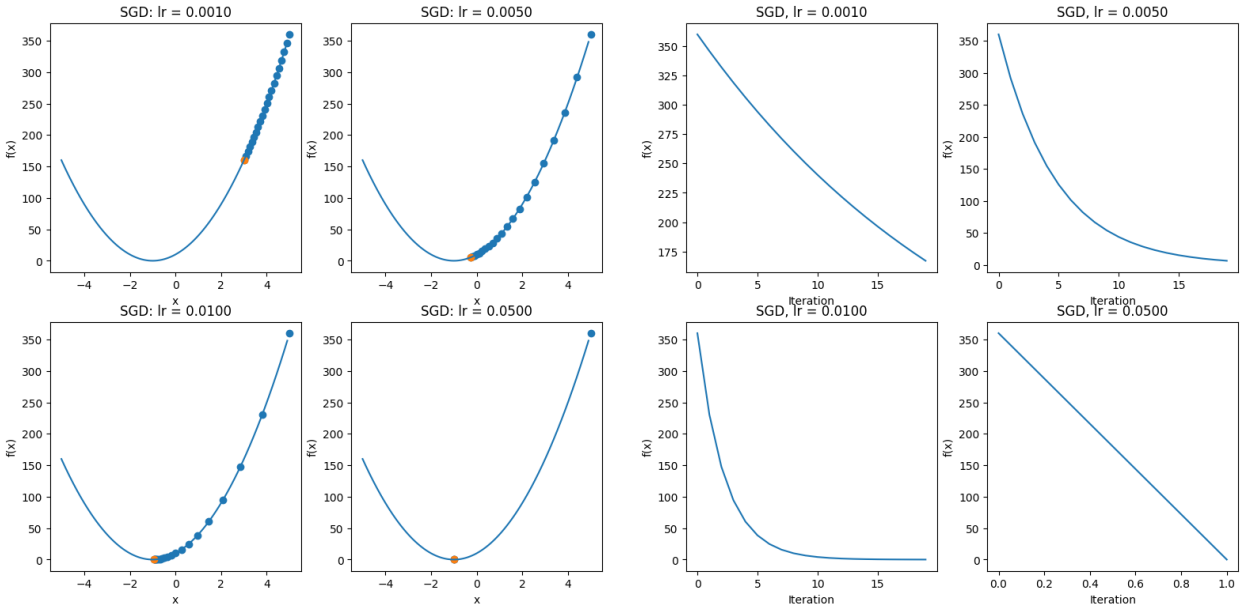
1.1 `Optimizer.sgd` method

1.1.a Test function `q1()`.

1.1.a.i Describe the termination criteria used in the `test_sgd` function in the `tests_A2.py` file. (1 mark)

Answer. There are 2 termination criterion used in `test_sgd`. The first is `max_iters` which places a constraint on how many iterations the algorithm can run. This is used to prevent the training from running forever and is set to 20 here – as per the handout. The second is `update_thres` which ensures that further iterations are meaningful. `update_thres` is set to $1 \cdot 10^{-6}$, and we break when `np.abs(update).max()` $<$ `update_thres`. `np.abs(update).max()` extracts the greatest magnitude in the update vector, so if the biggest change is less than `update_thres` then our update is negligible and we decide to stop. This is likely due to $v_t \to 0$ as $\nabla f \to 0$, meaning we are close enough and to save computation, we can end here.

1.1.a.ii Include the figures generated by `q1()` in your `PA2_qa.pdf` file. (1 mark)



(a) SGD decision parameter trace.

(b) SGD Loss vs. iteration.

Figure 1: Figures generated by `q1()`.

1.1.a.iii With learning rate $\eta = 0.05$, what would be the value of w_1, *i.e.*, after one iteration of SGD update. Show your mathematical process. If you implemented SGD correctly, the figures generated by `q1()` should verify your w_1. (1 mark)

Answer.

$$w^* = \arg\min_w f(w) = 10w^2 + 20w + 10 \quad \text{starting at} \quad w_o = 5 \qquad \nabla f(w) = 20w + 20$$

$$\nabla f(w_0) = 20(5) + 20 \to \nabla f(w_0) = 120$$

$$w_1 = w_0 - \eta \nabla f(w_0)$$

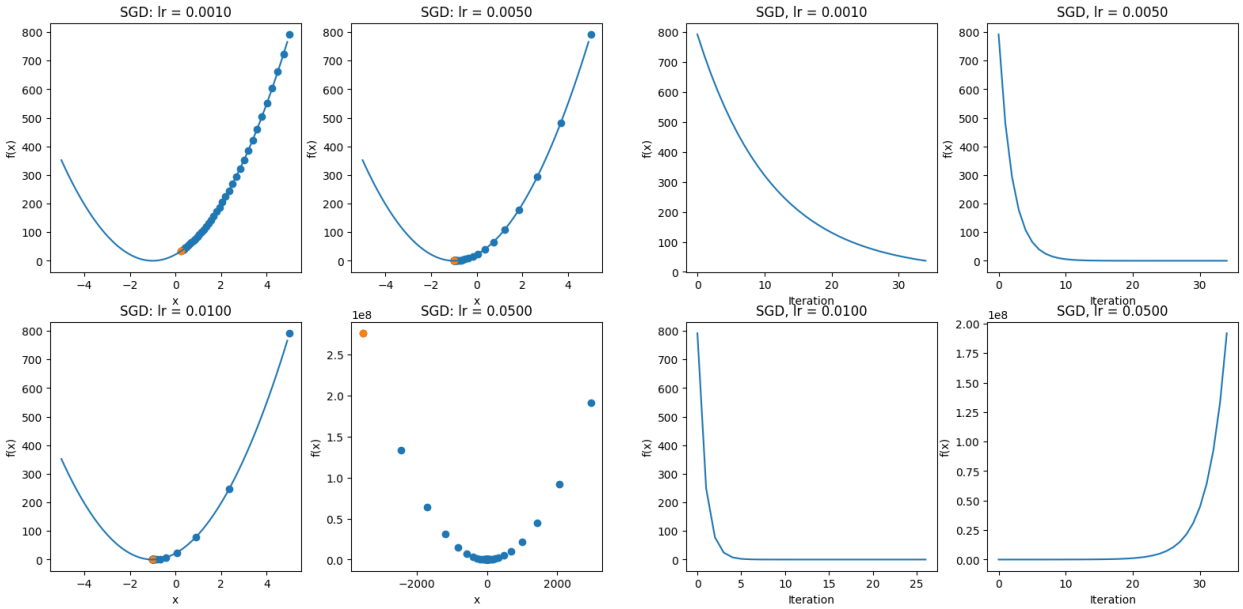
$$w_1 = 5 - (0.05)(120)$$

$$w_1 = -1$$

Consistent with the figure and output as we converge to $w_1 = -1$ in 1 iteration.

1.1.b Test function `q2()`.

1.1.b.i Include the figures generated by `q2()` in your `PA2_qa.pdf` file. (1 mark)



(a) SGD decision parameter trace.

(b) SGD Loss vs. iteration.

Figure 2: Figures generated by `q2()`.

1.1.b.ii When $\eta = 0.05$, SGD would fail to converge to the optimal solution. What causes such behavior? (1 mark)

Answer.

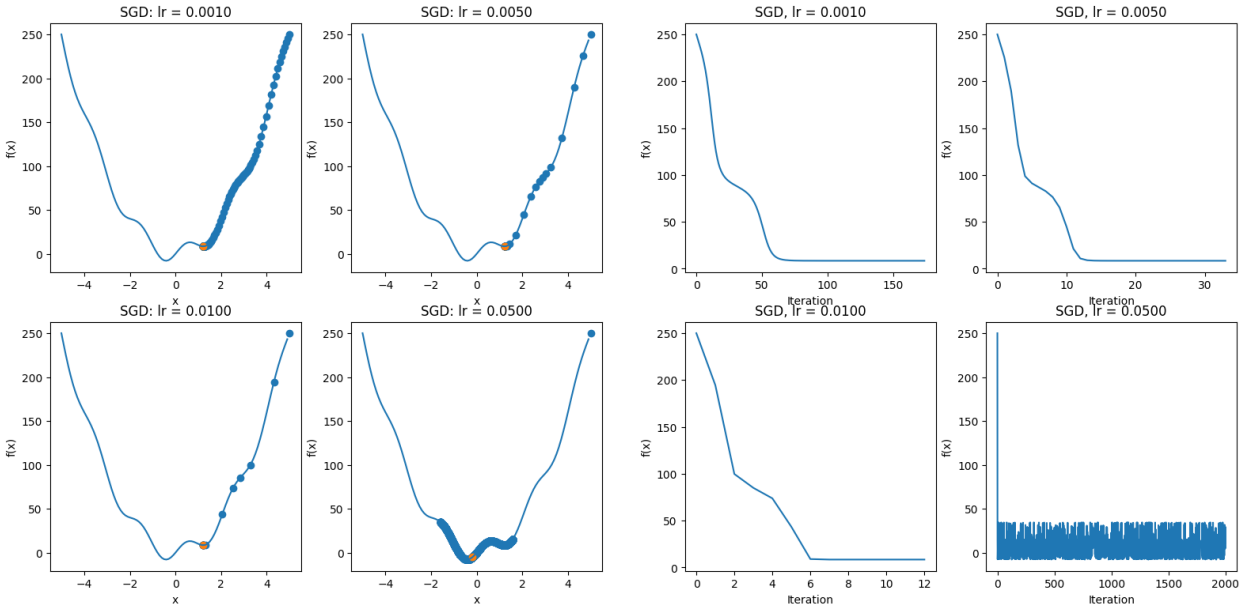
$$w^* = \arg\min_w f(w) = 22^2 + 44w + 22 \quad \text{starting at} \quad w_o = 5$$

$$\nabla f(w) = 44w + 44$$
$$\nabla f(w_0) = 44(5) + 44 \rightarrow \nabla f(w_0) = 264$$
$$w_1 = w_0 - \eta \nabla f(w_0)$$
$$w_1 = 5 - (0.05)(264)$$
$$w_1 = -8.2$$
$$w_2 = 20.84$$

This shows that choosing $\eta = 0.05$, we get the ping-pong explosion since the learning rate is too high. This causes the minima to be overshot and we land at a place where the slope is higher, leading to the same issue and divergence away from the minima and toward $\pm\infty$. The loss plot also confirms this as it increases with iterations.

1.1.c Test function `q3()`.

1.1.c.i Include the figures generated by `q3()` in your `PA2_qa.pdf` file. (1 mark)



(a) SGD decision parameter trace.

(b) SGD Loss vs. iteration.

Figure 3: Figures generated by `q3()`.

1.1.c.ii In 1-2 sentences describe the behavior of SGD in `q3()` when $\eta = 0.001, 0.005$, and 0.01. Explain why SGD fails to find the global optimum point? (1 mark)

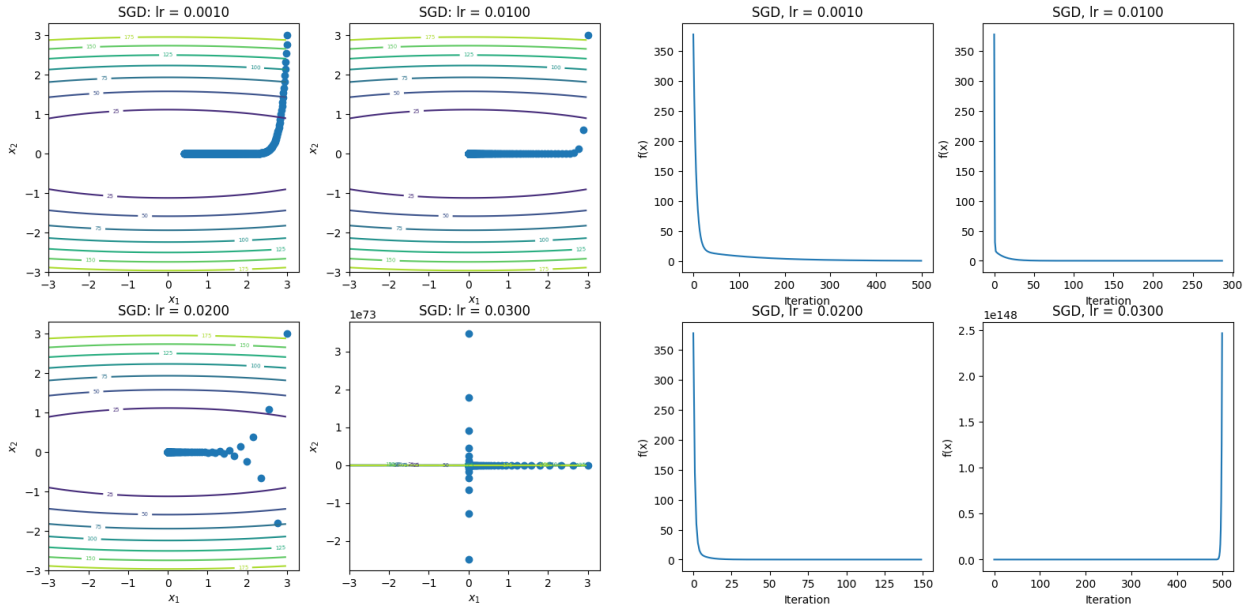
Answer. In each of these cases, SGD converges to the same point, $w^* = 1.22$, where the gradient is 0. The cases of greater learning rate converge faster. SGD fails to find the global optimum since it reaches a point where $\nabla f = 0$ which makes the update vector $v_t = -0$ and thus $w_{t+1} = w_t$. Since we use the stopping criteria `np.abs(update).max()` $<$ `update_thres`, this results in early stopping once the gradient of 0 is found. Note that this is not because of the code, rather the code is because the update of 0 will not change w but will still require needless computation.

1.1.c.iii In 1-2 sentences describe the behavior of SGD in `q3()` when $\eta = 0.05$. (1 mark)

Answer. In this case, the high learning rate leads to highly oscillatory updates and ultimately, it fails to converge to any minima and instead is stopped by the max iterations stopping criteria. It seems that ∇f is never 0 leading to bouncing around. The spurious optimia in this region keeps it trapped and prevents explosion like in the previous section, but here we still bounce around and do not converge.

1.1.d Test function `q4()`.

1.1.d.i Include the figures generated by `q4()` in your `PA2_qa.pdf` file. (1 mark)



(a) SGD decision parameter trace.

(b) SGD Loss vs. iteration.

Figure 4: Figures generated by `q4()`.

1.1.d.ii In 1-2 sentences describe the behavior of SGD in `q4()` when $\eta = 0.001$ and 0.01. How is this behavior related to the stretched nature of the function $f(\underline{w})$? (1 mark)

Answer. In these cases, the function manages to converge to the minima. The stretched nature of the function explains the path taken to converge. Initially, the change is almost entirely in the direction of w_2 and minimally in w_1. This is because the w_2 terms is stretched by a factor of 40 leading to a higher impact on the gradient. $\nabla f(\underline{w}) = 4w_1 + 80w_2$. Here we see that the w_2 term contributes 20 times more to the gradient, so the update will be $20x$ more in the w_2 direction when $w_1 = w_2$. This leads to finding the w_2 coordinate for $\nabla f = 0$ faster, as seen in the plots where we quickly reach $w_2 = 0$, and then gradually approach the w_1 coordinate for $\nabla f = 0$. This approach is slower since for $w_2 \approx 0$, $\nabla f(\underline{w}) = 4w_1$, which is much smaller in magnitude.

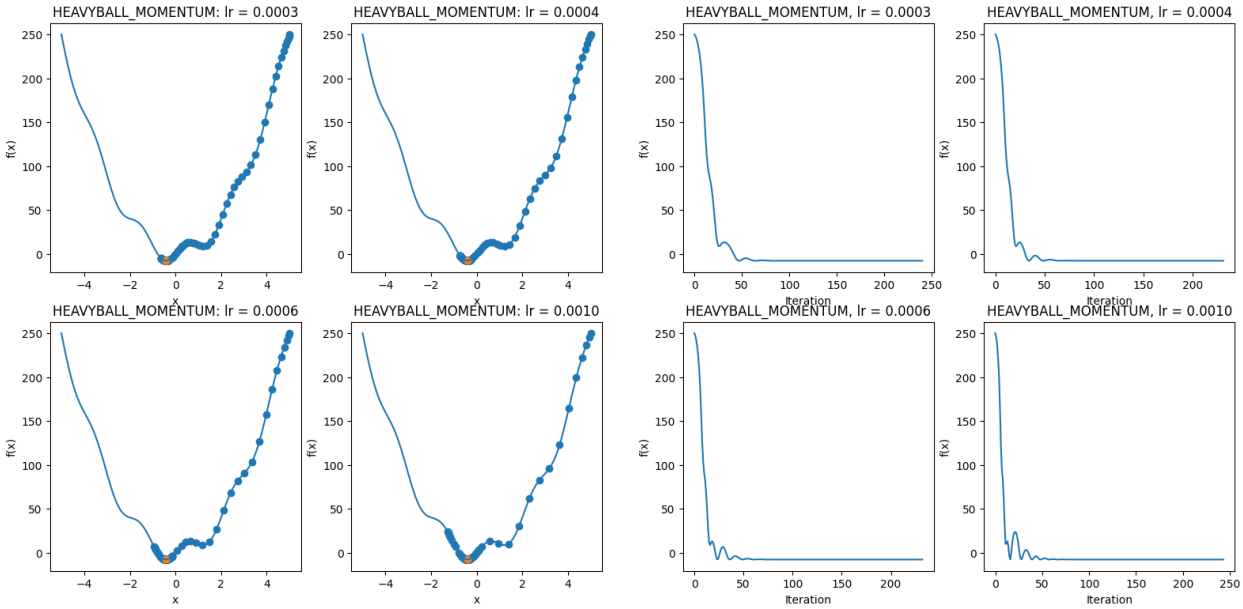
1.1.d.iii In 1-2 sentences describe the behavior of SGD in `q4()` when $\eta = 0.03$. (1 mark)

Answer. This is the case where the learning rate is too big and the gradient explodes. In this case, we are dealing with a multivariate function and the plot indicates that only the w_2 variable explodes and that w_1 actually converges. This is confirmed by the output which shows that the first entry in $\nabla f(\underline{w} \approx 0$ and $w_1 \approx 0$ indicating that in the w_1 direction, we are able to converge. This makes sense as w_1 does not have such a huge coefficient so we can increase η more forgivingly compared to w_2.

1.2 `Optimizer.heavyball_momentum` and `Optimizer.nestrov_momentum` methods

1.2.a Test function `q5()`.

1.2.a.i Include the figures generated by `q5()` in your `PA2_qa.pdf` file. (1 mark) use proper address to your png files



(a) Heavy-ball momentum decision parameter trace.

(b) Heavy-ball momentum Loss vs. iteration.

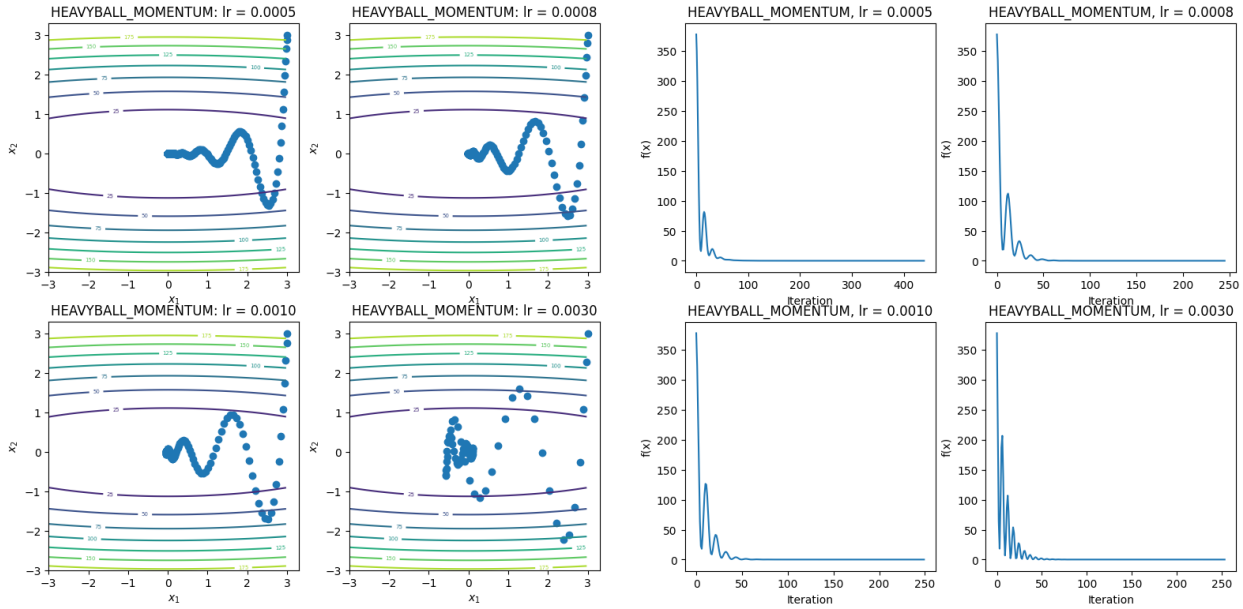
Figure 5: Figures generated by q5().

1.2.a.ii In 1-2 sentences, compare the performance of SGD with and without heavy-ball momentum by comparing the outcome of tests `q3()` and `q5()` (2 marks)

Answer. Your answer ...

1.2.b Test function `q6()`.

1.2.b.i Include the figures generated by `q6()` in your `PA2_qa.pdf` file. (1 mark)



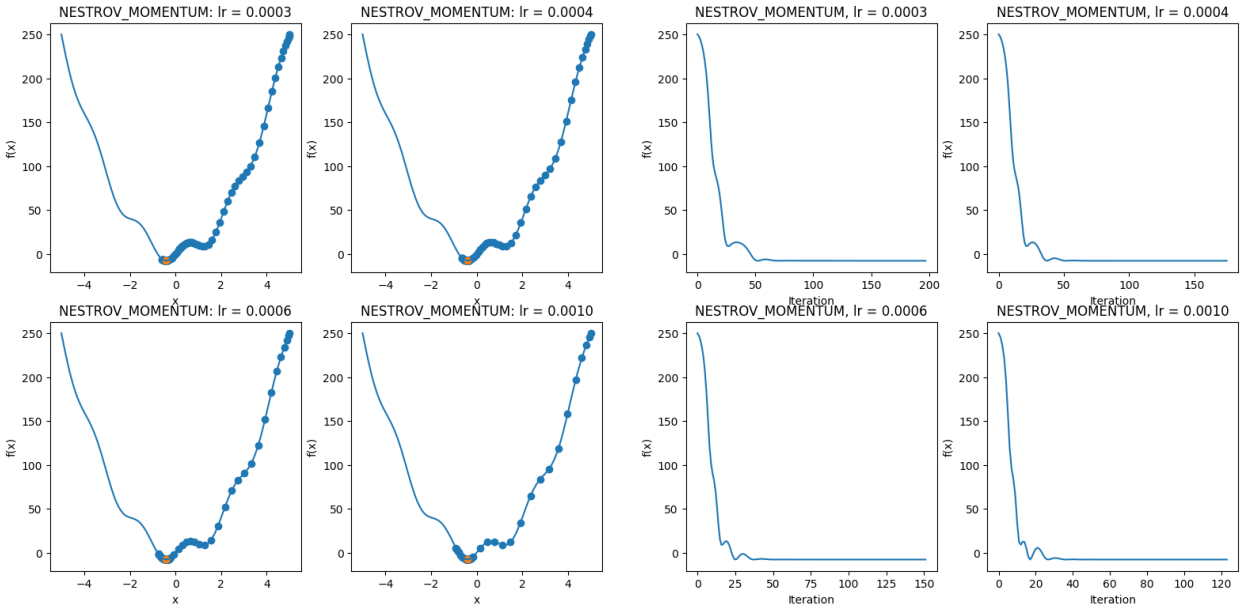
(a) Heavy-ball momentum decision parameter trace.

(b) Heavy-ball momentum Loss vs. iteration.

Figure 6: Figures generated by q6().

1.2.c Test function `q7()`.

1.2.c.i Include the figures generated by `q7()` in your `PA2_qa.pdf` file. (1 mark)



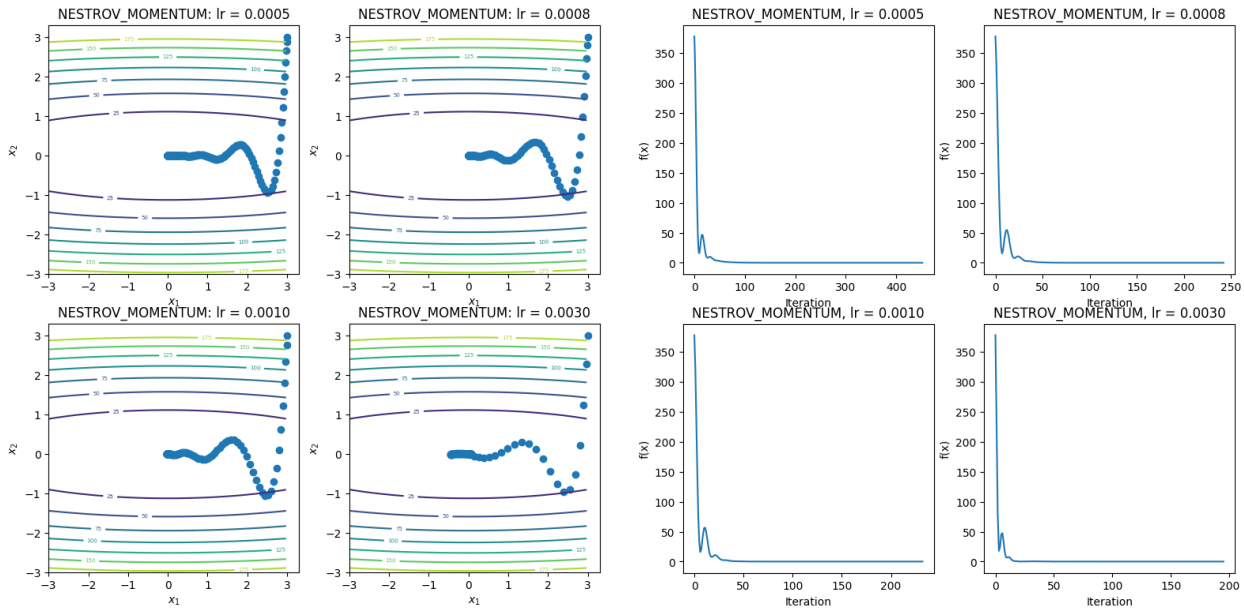
(a) Nestrov momentum decision parameter trace.

(b) Nestrov momentum Loss vs. iteration.

Figure 7: Figures generated by q7().

1.2.d Test function `q8()`.

1.2.d.i Include the figures generated by `q8()` in your `PA2_qa.pdf` file. (1 mark)



(a) Nestrov momentum decision parameter trace.

(b) Nestrov momentum Loss vs. iteration.

Figure 8: Figures generated by q8().

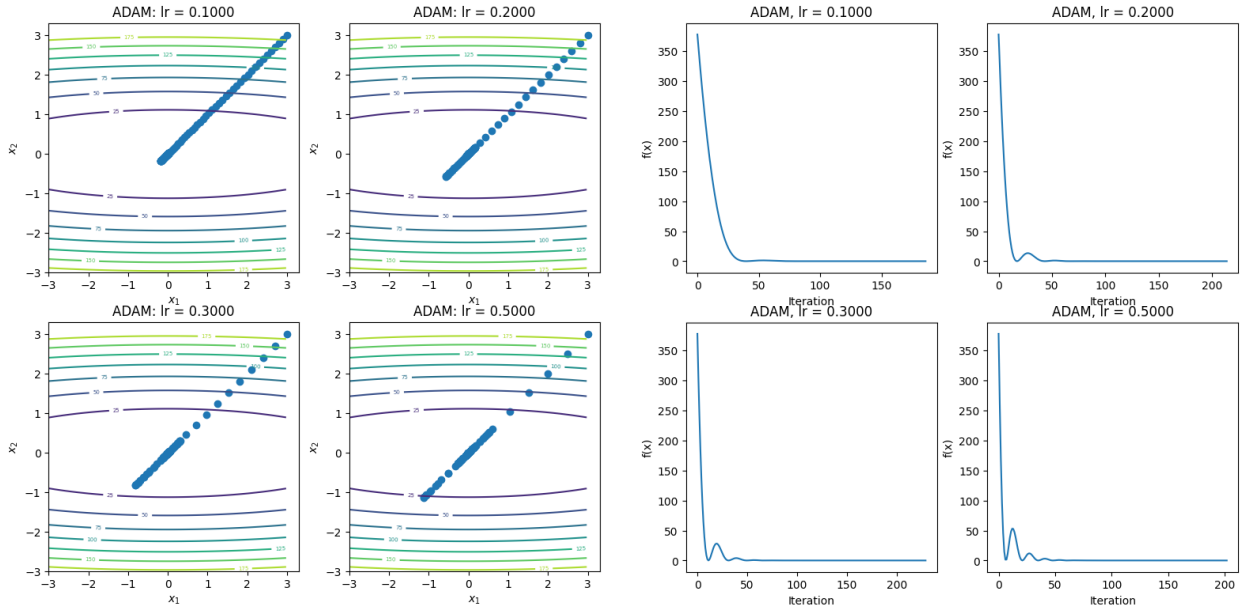
1.2.d.ii In 1-2 sentences, compare the performance of Nestrov Momentum with the heavy-ball momentum by comparing the outcome of tests `q5()` and `q6()` with that of `q7()` and `q8()`. (1 mark)

Answer. Your answer ...

1.3 `Optimizer.adam` method

1.3.a Test function `q9()`

1.3.a.i Include the figures generated by `q9()` in your `PA2_qa.pdf` file. (1 mark)



(a) Adam momentum decision parameter trace.

(b) Adam momentum Loss vs. iteration.

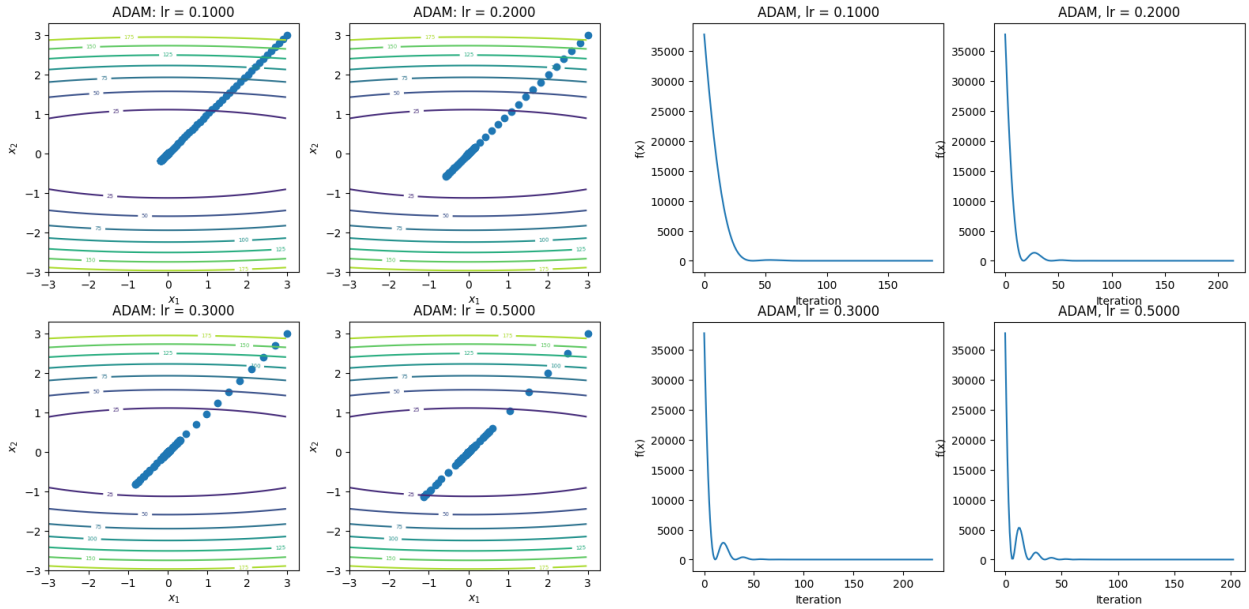
Figure 9: Figures generated by q9().

1.3.a.ii In 1-2 sentences, compare the performance of adam with momentum method (heavy-ball or Nestrov) (2 marks)

Answer. The most obvious difference is the path to convergence, Nestrov takes a path aligning with the magnitude of the gradient while adam takes a straight line from the point to the minimum. This is due to the use of two stored parameters m_t and v_t. These parameters store the direction and magnitude separately allowing more nuanced exploration. This is in contrast with the v_t in Nestrov momentum, which holds both the magnitude and direction, which is more restrictive.
When comparing the iterations to convergence, adam is better, especially for the low learning rate and manages to converge in less than 200. The others are similar.
It is also worth noting that the learning rates used here are significantly higher than those using Nestrov, suggesting that adam can handle higher learning rates better.

1.3.b Test function `q10()`.

1.3.b.i Include the figures generated by `q10()` in your `PA2_qa.pdf` file. (1 mark)



(a) Adam momentum decision parameter trace.

(b) Adam momentum Loss vs. iteration.

Figure 10: Figures generated by q10().

1.3.b.ii Based on the outcome of `q9()` and `q10()`, describe the advantage of Adam in 1-2 sentence. (2 marks) [**HINT:** run `q11()` to see what could be the impact of scaling the function (or gradients) on the other optimization method such as gradient descent with Nestrov Momentum. You don't need to report the output of `q11()` in your report. Also, note that `q11()` would most often result in error. Don't worry. That is intentional. Try to understand why this happens.]

Answer. Based on the outputs of `q9()` and `q10()`, we see that the convergence occurs in the exact same number of iterations and that the values are also the same. This suggests that adam is scale-invariant and only depends on the shape. adam uses an adaptive learning rate to adjust each parameter according to the size of the gradient at each step by keeping track of the m_1 the mean and v_1 the variance of the gradients. The variance makes adam adaptive and adapts the learning rate according to the magnitude of the gradients, ensuring that updates are appropriately sized for both large and small gradients. This adaptability leads to greater stability. `q11()` demonstrates that Nestrov is not scale invariant is it no blows up when we take the scaled function. This is advantageous as adam can be more versatile and used for scaled functions which is often the case in ML.

2 Multiclass Logistic Regression

2.1 Implementing the Learning Model

No written part.

2.2 Implementing the Learning Algorithm

2.2.a The test function `q22()` runs your implementation on the Iris dataset.

2.2.a.i Include the figures generated by `q22()` in your `PA2_qa.pdf` file. (2 marks)

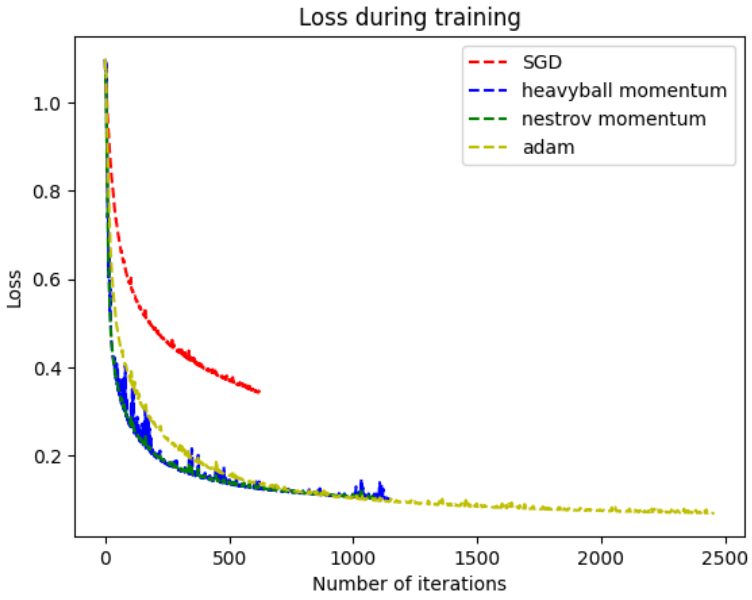


Figure 11: Figures generated by q22()

2.2.a.ii In 1-2 sentences, compare the performance of the four variants of gradient descent on this dataset (2 marks)

Answer. Regular SGD converges first, however the loss is significantly higher indicating that it reached a local minimum. This is characteristic of SGD. Nestrov and heavyball performed similar to one another, however it is evident that heavyball is far "bouncier" while Nestrov is smooth. This is because Nestrov looks ahead and computes the gradient from its future trajectory. Nestrov's forsight allows it to use knowledge of its trajectory to adjust its trajectory and prevent bouncing. adam learns third slowest initially, however after the others converge, adam continues to learn and achieves a lower loss. This is a result of adam being able to scale low updates up, bypassing the stopping criterion.

2.2.a.iii In 1-2 sentences, explain how is it possible that the loss derived by the Adam optimizer is smaller than that of Heavy-ball Momentum, but the evaluation score of Adam is equal to the evaluation score of the heavy-ball momentum. (2 marks)

Answer. Loss is in-sample error and the evaluation score is taken from the validation set. This means that adam was able to learn more from the train set and became a better classifier on the training set, however when it came to validation, the two scored the same, perfect. On the one hand, this suggests that whatever adam learned turned out to be negligible and possibly noise. On the other hand, it can also mean that our validation set wasn't large enough and didn't have any of the cases where the additional learning can be used. For a simple task like this, it can be assumed that it was noise. It is worth noting that if there was a decrease in the validation accuracy of adam, it would be a clear indication of overfitting.

2.2.b The test function `q23()` runs your implementation on the digits dataset.

2.2.b.i Include the figures generated by `q23()` in your `PA2_qa.pdf` file. (2 marks)

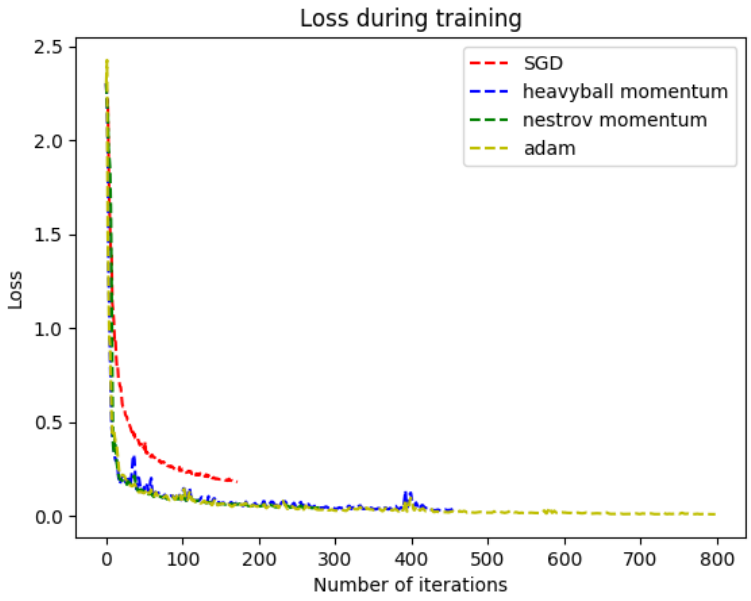


Figure 12: Figures generated by q23().

3 K-Means Clustering (Bonus)

No Written part.

4 Discussion

4.a How much time did you spend on each part of this assignment? (1 mark)

Answer. Rough logs:

1. 30 mins writing code for section 1
2. 2 hrs working on the answers for 1.1
3. **TIME** working on the answers for 1.2
4. 1 hrs working on the answers for 1.3
5. 2 hrs writing code for section 2.1
6. 30 mins working on the answers for 2.2
7. 10 mins contemplating doing kmeans
8. *doing k means*
9. *revising and submitting*
10. Total of 5 mins doing this

4.b Any additional feedback? (optional)

Answer. This was a good assignment that connected well with what we learned in class. Shame it couldn't contain more midterm material but I guess that's in A3.