MA20278 Machine Learning 1

Machine Learning 1 Coursework

Callum Gregory

July 16, 2023

Question 4

Which value of t gives the fastest convergence? What makes smaller and larger t not optimal?

The value of t which gives the fastest convergence is $t = 32$ since the test loss converges to the desired accuracy $\left(\epsilon = 10^{-8}\right)$ in the smallest number of iterations.

Smaller t values are not optimal as they converge to the desired test loss accuracy slower. This is because if the learning rate t is small, tG^u and tG^w will be matrices consisting of small elements so U and W won't change as fast and thus the test loss will change slower so more iterations will be needed for convergence.

Large values of t are not optimal as they cause the U and W to change more drastically. This means that the minima can be "overshot". The test loss can then get larger and larger each iteration until it can't be stored in Python. This is what happened when $t = 64$, leading to overflow.

At which iteration k does the algorithm stops with this optimal t?

The Gradient Descent method for Matrix Completion for the optimal t which is $t = 32$ stops on average at iteration $k \sim 36$. This can be seen from taking the average from the initial run and the 3 re-runs:

$$k = \frac{36 + 32 + 41 + 34}{4} = 35.75 \sim 36$$

What types of convergence can be seen in the loss plot?

For $t = 8, 16, 32$, it appears that the test loss converges linearly for Algorithm 3. This is because the test loss plotted in the logarithmic scale appears as basically a straight line with negative gradient. Furthermore, as t gets smaller, the general trend is that the test loss converges slower for longer before beginning to converge in a linear manner.

What is the relative error $\frac{\|UW^T - Y\|_F}{\| Y\|_F}$ of the entire result? How does it relate to the final test loss?

From the relative errors (see Figure (a) at the bottom of the document), we can see that when $t = 1, 2, 4, 8, 16, 32$ the test loss converges and all these values of t have similar low relative error - and as such Y is predicted well. Whereas when $t = 64$ and the test loss doesn't converge the relative error is very large and so, Y is not predicted well.

Question 6

Which value of t gives the fastest convergence? How different is the optimal t for sag compared to that of gd?

On average, $t = \frac{1}{16}$ gives the fastest convergence rate for SAG since the test loss converges to the desired accuracy in the smallest number of iterations. We see that the optimal learning rate $t = 32$ for Gradient Descent is 512 times bigger than the optimal $t = \frac{1}{16}$ for the Stochastic Average Gradient Descent Method.

At which iteration k does sag stops with this optimal t? Does it reach convergence within K iterations?

The Stochastic Average Gradient Descent method for Matrix Completion for the optimal t - which on average is $t = \frac{1}{16}$ - stops on average at iteration $k \sim 32319 < K = 50000$. So it does reach convergence within $K = 50000$ iterations. This can be seen from taking the average from the initial run and the 3 re-runs:

$$k = \frac{31073 + 33230 + 34018 + 31982}{4} = 32318.5 \sim 32319$$

What types of convergence can be seen in the loss plot?

From the loss plots we can see that due to the stochastic nature of the algorithm we have jagged lines so the loss fluctuates as it attempts to manoeuvre to the minimum. Furthermore, the jaggedness of the the loss plot could also be down to the fact that the algorithm is converging to the minimum in an oscillatory manner.

Learning rates that are large tend to oscillate towards the minimum (e.g.$t = \frac{1}{4}$) or diverge away from the minimum like $t = 1, \frac{1}{2}$.

Unsurprisingly, the Stochastic Average Gradient Descent algorithm converges much slower than the Gradient Descent due to the fact that in SAG the full gradient is not used (although this does make it a lot quicker to run).

What is the relative error $\frac{\|UW^T - Y\|_F}{\| Y\|_F}$ of the entire result?

Upon examining the relative errors (see Figure (b)) at the different values of t we see that for $t = \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ they all give relative errors of less than 0.01. Furthermore, upon observing the average number of iterations for $t = \frac{1}{8}$ we get $k = 33735.75 \sim 33736$. Since $t = \frac{1}{8}$ has a smaller relative error than the optimal $t = \frac{1}{16}$ it might be worth performing the extra ~ 1500 iterations, so $t = \frac{1}{8}$ can be used to get a slightly better prediction for Y.

Question 7

What is L_{low}? (note that it's independent of a particular algorithm)

From the initial run and the 2 re-runs, it can be seen that the loss doesn't seem to go below $L_{\mathcal{I}_{Test},Y}(U, W) = 10^{-3}$

Which value of t gives the fastest convergence, for each algorithm?

For the Gradient Descent algorithm, on average $t = 16$ gives the fastest convergence, whereas for the Stochastic Average Gradient Descent algorithm $t = \frac{1}{16}$ gives the fastest convergence.

At which iteration k does each algorithm stop with this t?

On average, for the Gradient Descent algorithm with learning rate $t = 16$ the loss converges to the desired accuracy within $k = 363$ iterations. This can be seen from taking the average of the initial run and the two re-runs:

$$k = \frac{459 + 432 + 199}{3} = 363.33 \sim 363$$

On average, for the Stochastic Average Gradient Descent algorithm with learning rate $t = \frac{1}{16}$ the loss converges to the desired accuracy within $k = 71331$ iterations. This can be seen from taking the average of the initial run and the two re-runs:

$$k = \frac{67328 + 124536 + 23708}{3} = 71857.33 \sim 71857$$

What types of convergence can be seen in the loss plots?

In the Gradient Descent algorithm we see that when $t = 32$ the loss seems to fluctuate up and down in a jagged horizontal pattern after roughly 2000 iterations. This seems to incline that effectively we are finding pairs of U and W around the desired U and W which give the minimum loss, and jumping around these pairs, without getting much closer to the desired U and W which give minimum loss. For $t = 1, 2, 4, 8, 16$ we see that within the first roughly 100 iterations the test loss decreases much faster than during the remaining iterations. For $t = 64$ the test loss increases very fast within less than 10 iterations and hence it doesn't converge.

In the Stochastic Average Gradient Descent algorithm, in general the learning rate t need to be less than $t = \frac{1}{16}$ for the algorithm to converge. This algorithm also seems to take many more iterations to converge in comparison to the Gradient Descent algorithm.

What is the relative error $\frac{\|UW^T - Y\|_F}{\| \ Y\|_F}$ of the entire result?

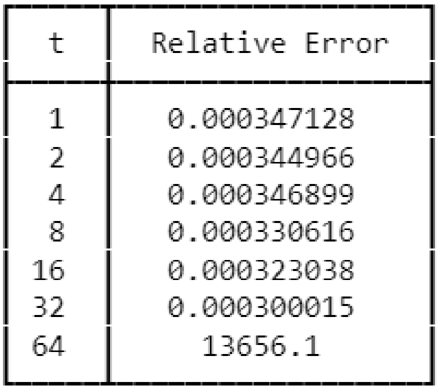
From Figure (c) we can see that the relative errors for $t = 2, 4, 8, 16$ are similar. Therefore, it makes sense to take the t that converges the quickest and use that to calculate U and W. This is because if all the relative errors for these t values are similar, then the image they will produce will be similar.

From Figure (d) when $t = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ the loss doesn't converge and we have very high relative errors and so the corresponding U and W will give a very poor prediction of the image Y. For smaller learning rates like $t = \frac{1}{16}, \frac{1}{32}, \frac{1}{64}$ the relative error is small but not quite as small as the relative errors given for the best values of t in the GD algorithm.

How does the behaviour of the algorithms compare overall to the full data tests in Q4 and Q6?

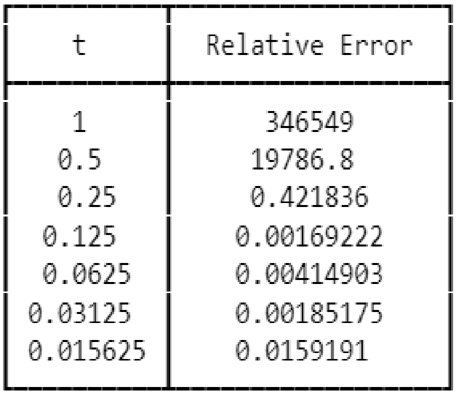
For the Stochastic Average Gradient Descent algorithm, there is a smaller learning rate required in general for convergence of the matrix Y in Q7 compared to the matrix Y used in Q6.

For the Gradient Descent algorithm, in Q7 the loss converges much faster within the first roughly 100 iterations before decreasing at a much slower rate. Whereas in Q4 the test loss seemed to decrease much quicker and to a lower value. Furthermore, When $t = 32$ in Q7 there appears to be some form of oscillatory behaviour which prevents the loss decreasing any further after roughly 2000 iterations, whereas in Q4 $t = 32$ converges fastest.



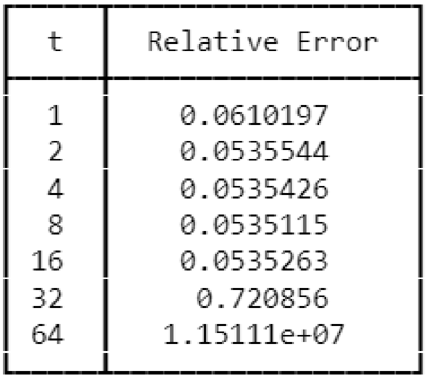
t	Relative Error
1	0.000347128
2	0.000344966
4	0.000346899
8	0.000330616
16	0.000323038
32	0.000300015
64	13656.1

(a) **Q4**: Relative Error for GD



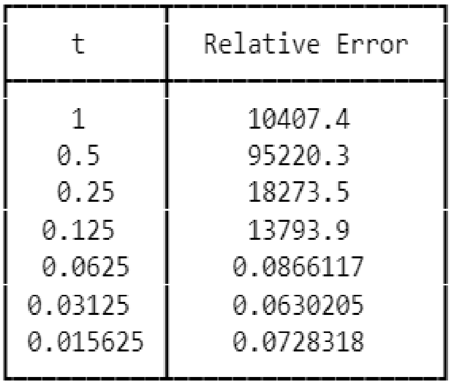
t	Relative Error
1	346549
0.5	19786.8
0.25	0.421836
0.125	0.00169222
0.0625	0.00414903
0.03125	0.00185175
0.015625	0.0159191

(b) **Q6**: Relative Error for SAG



t	Relative Error
1	0.0610197
2	0.0535544
4	0.0535426
8	0.0535115
16	0.0535263
32	0.720856
64	1.15111e+07

(c) **Q7**: Relative Error for GD



t	Relative Error
1	10407.4
0.5	95220.3
0.25	18273.5
0.125	13793.9
0.0625	0.0866117
0.03125	0.0630205
0.015625	0.0728318

(d) **Q7**: Relative Error for SAG