Q4: testing GD on full data

After running tests of Algorithm 3 a few times on synthetic data and varying the step size t, we see that $t = 32$ gives the fastest convergence. With this value convergence took around 35 to 45 iterations. It was seen throughout repeats that doubling the value of t tended to double the rate of convergence.

The smaller values of t are not as optimal as their step size is too small: they moved slowly down the function and therefore took longer to converge. $t = 64$ showed an interesting behaviour - each time it looked to oscillate around the solution before its loss diverged. The algorithm is overshooting the local minimum each time due to the large step size; eventually it seems to diverge from our desired solution to a different local minimum with an extremely high loss value.

Each plot (except $t = 64$ previously discussed) showed linear convergence on the logged-loss scale, i.e. the exponential convergence discussed in Theorem 4.27.

Plotting the logged relative error against iteration number, it follows an extremely similar trajectory as the logged test loss. Their logs look like they are almost exactly proportional. In actual value, the relative error begins close to 1 and converges to 0.

Q6: testing SAG on full data

After running tests of Algorithm 4 a few times on synthetic data and varying the step size t, we see that $t = \frac{1}{16}$ tended to give fastest convergence in around 30,000 - 35,000 iterations (within our K = 50,000 limit). This step size is far smaller than the optimal step size for GD - the update in SAG does not fully compute the gradient, and is therefore much more prone to noise and variance. This means a much lower step size needs to be used to ensure the algorithm does not keep bouncing around the optimal solution.

Values of t up to $t = \frac{1}{4}$ showed exponential convergence with some noise due to stochasticity. Values of $t = \frac{1}{2}$ and higher did not converge. Their behaviour looks very similar to $t = 64$ for GD suggesting the step size was too large for the algorithm to converge, and it eventually diverged to a different local minima with much larger loss.

Again the logged relative error followed a similar trajectory as the logged test loss, but looks to be a lot less affected by the stochasticity than the test loss - its fluctuations are a lot less pronounced (this is especially apparent in $t = \frac{1}{2}$, $t = 1$).

Q7: tests with real data

After some experimenting with both algorithms on the data, $L_{low} = e^{-5.8}$ is about the minimum that we can realistically achieve with both algorithms. Since the real data is not actually representable as a rank 4 matrix, there is no solution with 0 loss. L_{low} is the loss value of the optimal solution, i.e. the infimum of the loss function.

It was found that $t = 24$ gave the fastest convergence for GD and $t = \frac{1}{10}$ was the fastest for SAG. GD took far less iterations, with only just only 500 typically required for convergence to $2L_{low}$. SAG required far more - typically 100,000 - 150,000+ to converge to the same threshold.

In the loss plots, both algorithms had fast initial convergence (with quite a bit of stochasticity in SAG), followed by a small 'bump' beyond which the algorithms showed slow convergence that looked like $O(k^{-1/2})$. From theory this suggests that the loss function is both β-smooth and λ-strongly convex on the initial domain, however is only β-smooth past a certain point.

The logged relative error again followed a similar pattern to the logged test loss, starting near 1 and converging to 0. It was again less sensitive to the noise in the convergence, following a smoother line.

Compared to Q4, GD behaved quite differently: in Q4 it tended to have a period of uncertainty at the beginning before quickly converging exponentially to an extremely accurate solution. On the real data, it exhibited exponential convergence followed by $O(k^{-1/2})$ to a solution with far less accuracy. The difference in accuracy is explained by the fact that the synthetic data had a solution with 0 loss (Y was in fact a rank 2 matrix), whilst the real data has only an approximation. The same can be said of SAG, although when applied to the real data it took a very large number of iterations to converge to solutions of comparable accuracy to GD.

The results are shown below and are also included in the jupyter notebook. Both algorithms are able to produce a result indistinguishable from the original by eye.