## Stochastic Gradient Descent

Before presenting performance of SGD and its variants with respect to parameters of interest, I would like to talk about convergence criteria.

At first, I tested convergence based on coefficient. That leads to unsatisfactory results. The reason is that coefficients update is very sensitive to random samples each iteration. The adjacent two randomly sampled data points could be very similar, giving pretty close coefficients.

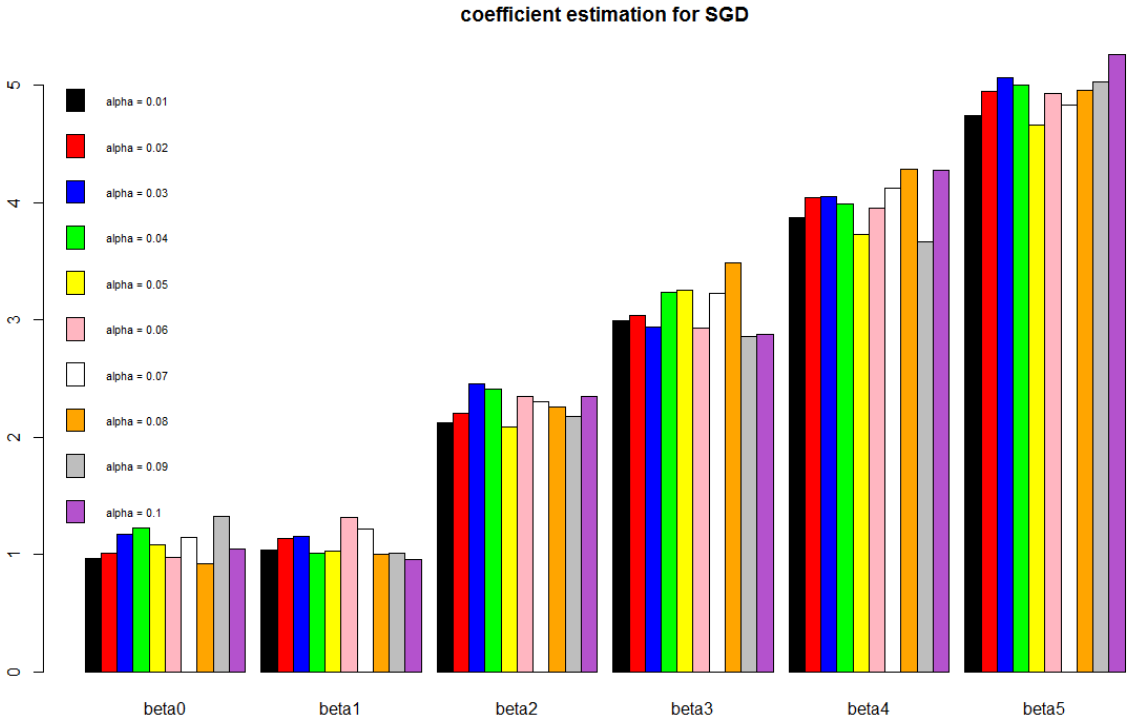
Then, I turned to test convergence based on log likelihood. I tried 3 criteria: (1) actual log likelihood; (2) simple moving average log likelihood from samples; (3) both actual log likelihood and simple moving average log likelihood from samples. The last one works best in terms of giving the right estimation. Therefore, in the R code for SGD, I implement convergence test based on the last criterion.

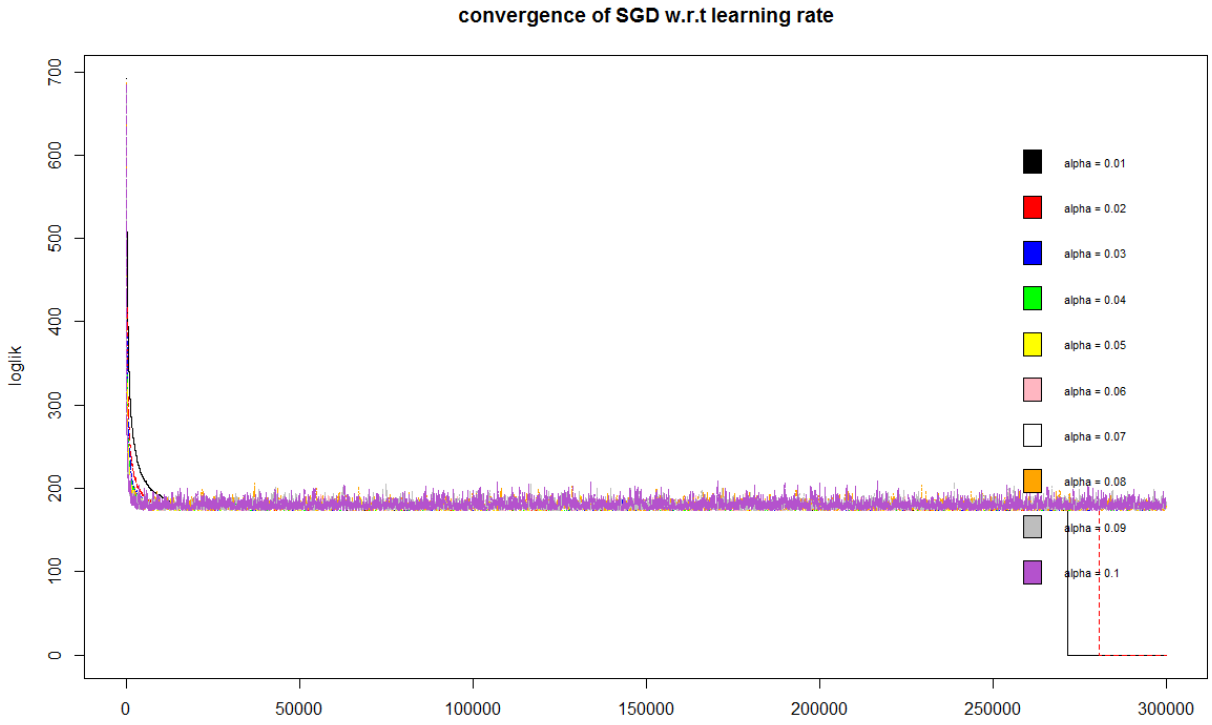
Tradeoff between Gradient Descent and Stochastic Gradient Descent. GD takes fewer iterations to converge, but each iteration requires more time since the computation of the gradient is based on the entire data. SGD takes more iterations to converge, but each iteration requires less time since the computation of the gradient is based on one sampled data point. The intuition is that in SGD, we are not sure about the direction, so we take more steps to get to where we want. Also, the step size should be smaller in each iteration under SGD than under GD.

***Performance of SGD w.r.t. learning rate***

*Take away*:

The algorithm gives reasonable estimates. The log likelihood value fluctuates, making it hard to converge. As learning rate grows, the log likelihood value falls faster but takes more time to converge.



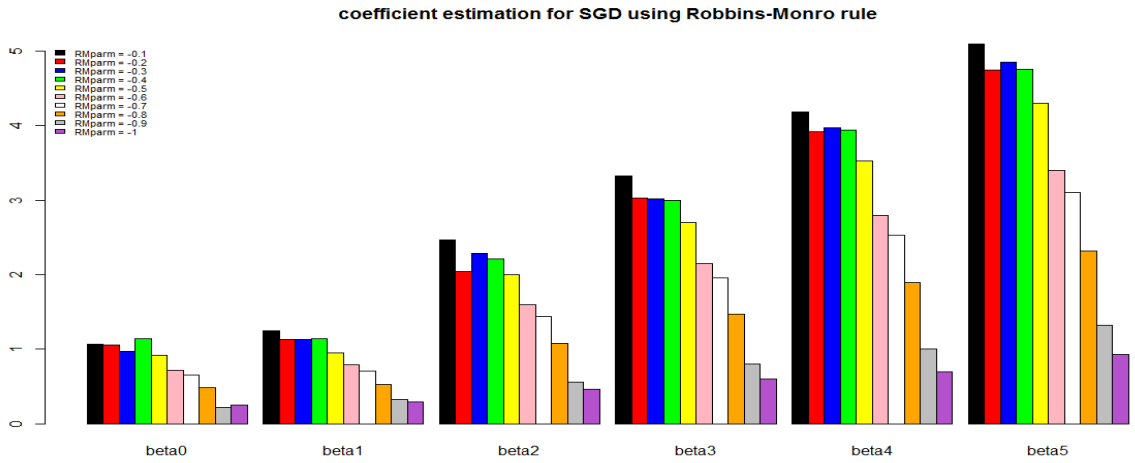

convergence of SGD w.r.t learning rate
loglik
0
100
200
300
400
500
600
700
0
50000
100000
150000
200000
250000
300000
alpha = 0.01
alpha = 0.02
alpha = 0.03
alpha = 0.04
alpha = 0.05
alpha = 0.06
alpha = 0.07
alpha = 0.08
alpha = 0.09
alpha = 0.1

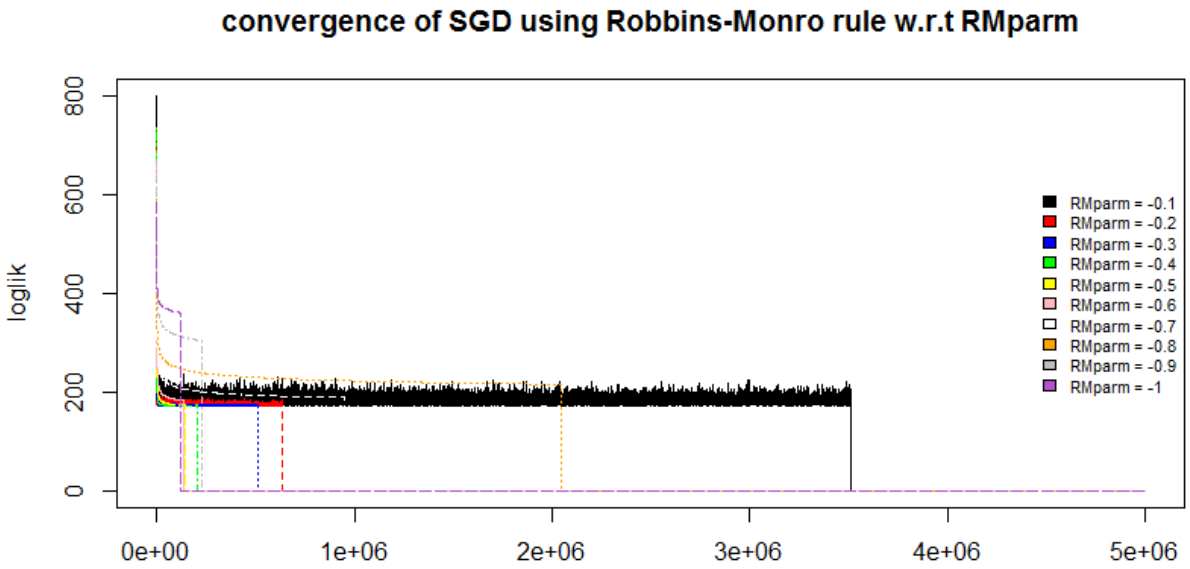
## Stochastic Gradient Descent using Robbins-Monro rules

The RM rules implement a decreased dynamic learning rate over time. I will investigate performance of this variant of SGD with respect to the parameters that determine how learning rate change over time.

### *Performance of SGD using RM rules w.r.t RM parameter*

*Take away*:
As RM parameter (in absolute value) grows, the algorithm converges faster, but does not return precise estimates. Bigger RM parameter leads to greater decrease in learning rate over time. The intuition is that under SGD, we should changes step size more cautiously since the direction is unreliable at each iteration.



***Performance of SGD using RM rules w.r.t C***

*Take away*:
As C grows, the algorithm returns more accurate estimates, but converges more slowly.