Big Data Analytics - GUID: 2383746W

R–Project: *Implementation of Stochastic Gradient Descent*

This project aims at the development of several stochastic optimization algorithms in R, including;

1. *Gradient Descent (GD),*
2. *Stochastic Gradient Descent (SGD),*
3. *Stochastic Gradient Descent with Momentum (MSGD),*
4. *Stochastic Gradient Descent with Nesterov Accelerated Gradient (NAGSGD),*
5. *Adaptive Gradient Algorithm (AdaGrad),*
6. *Root Mean Square Propagation (RMSProp) and,*
7. *Adaptive Moment Estimation (ADAM).*

The above optimizers are *trained* upon a small *simulated dataset* ($n = 1000$, one covariate) and a large *weather dataset* ($n = 96\,453$, six covariates) in order to conduct inference for linear regression coefficients via stochastic optimization. In a further step, dataset-dependant *hyperparameter tuning* is carried out for each algorithm in order to converge to the coefficient values suggested by linear regression (baseline). Moreover, *computational efficiency* is based on successive iterations of the cost function values and *convergence performance* is determined by the trajectories from a common initial value to the endpoint representing the parameter estimate obtained via linear regression. Since this work is code-based only, main results will be discussed in more detail.

Parameter	Interpretation	Dataset / initial values
θ	Coefficient (including intercept term)	simulated: $\theta^{(0)} = (7, -8)^T$ weather: $\theta^{(0)} = (-5, -3,\ 4, 1, 10, -9)^T$
a, v	Learning rate	None for a, to be tuned to converge towards the regression coefficient estimates. simulated: $v^{(0)} = (0, 0)^T$ weather: $v^{(0)} = (0, 0, 0, 0, 0, 0)^T$
m	Momentum / moving average	simulated: $m^{(0)} = (0, 0)^T$ weather: $m^{(0)} = (0, 0, 0, 0, 0, 0)^T$
g, G	Gradient and squared gradient	simulated: $diag\ (G^{(0)}) = (0, 0)^T$ weather: $diag\ (G^{(0)}) = (0, 0, 0, 0, 0, 0)^T$
b, c	Memory factor	None, to be tuned to converge towards the regression coefficient estimates
ϵ	Error tolerance	$\epsilon = 1e - 8$

There were in total of two assessed questions, question 1 is based on the *simulated dataset* where 100 iterations of each algorithm are run by sampling a subset of $s = 100$ data points per iteration. The same procedure is repeated for the *weather dataset* in question 2, this time with 200 iterations and $s = 1000$.

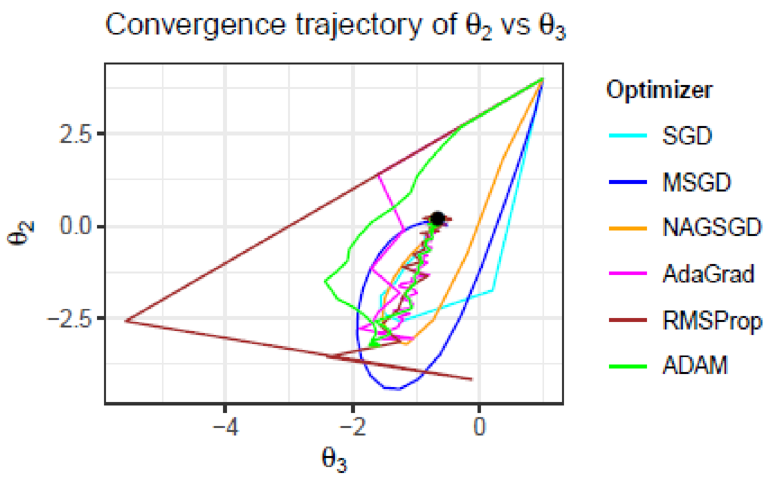
Hyperparameter tuning and parametric convergence

It was found that with increasing data size and a constant sampling subset of roughly 10 per cent of the original data points, the learning rate a tends to require significantly higher values in order to converge towards the regression coefficient estimates. As the weather dataset is larger by a factor of ten, a higher learning rate implies faster training, which seems required under the consideration that *six* rather than just *one covariate* is required to be optimized. As for the convergence, the latter three optimizers behaved very sensitive under hyperparameter tuning with minor adjustments resulting in erratic changes of values for the estimates. Furthermore, optimizers 1 to 7 were increasingly difficult to be tuned, while sensible solutions for *GD*, *SGD* and *MSGD* could be found in a comparatively short time. After parameter tuning, the optimal choice of hyperparameter combinations for each stochastic algorithm and for each question are summarised in the table below.

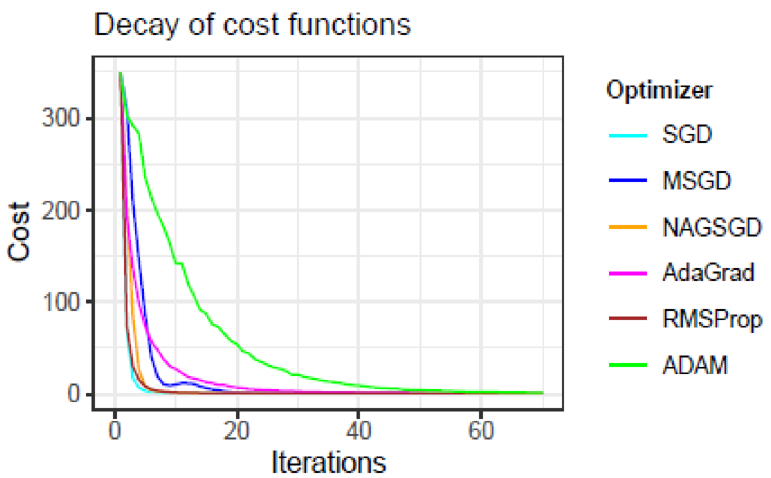
Optimizer	**Question 1** (simulated $n = 1{,}000$)	**Question 2** (weather $n = 96{,}453$)
1. GD	$a = 0.28$	$a = 0.6$
2. SGD	$a = 0.085$	$a = 0.5$
3. MSGD	$a = 0.085$, $b = 0.27$	$a = 0.26$, $b = 0.75$
4. NAGSGD	$a = 0.155$, $b = 0.58$	$a = 0.5$, $b = 0.6$
5. AdaGrad	$a = 1.5$	$a = 2.6$
6. RMSProp	$a = 0.054$, $c = 0.999$	$a = 0.208$, $c = 0.999$
7. ADAM	$a = 0.45$, $b = 0.554$, $c = 0.999$	$a = 0.45$, $b = 0.6$, $c = 0.999$

Since parametric convergence towards the ML estimate is harder to achieve for the larger weather dataset with six covariates, final results will only be represented for this case (question 2). Gradient Descent achieves identical results to *LM*, this is because *GD* uses the real data instead of a sampling subset.

Coeff	LM	GD	SGD	MSGD	NAGSGD	AdaGrad	RMSProp	ADAM
θ_0	10.8550	10.8550	10.8344	10.8544	10.8517	10.8508	10.8450	10.8490
θ_1	10.7502	10.7502	10.7315	10.7495	10.7001	10.7480	10.7231	10.7487
θ_2	0.2023	0.2023	0.2224	0.2048	0.2344	0.1977	0.2045	0.1971
θ_3	-0.6615	-0.6615	-0.6704	-0.6759	-0.6695	-0.6354	-0.6662	-0.6801
θ_4	0.0569	0.0569	0.0615	0.0546	0.0419	0.0426	0.1050	0.0480
θ_5	0.0231	0.0231	0.0321	0.01594	-0.0096	0.0367	0.0296	-0.0049



Based on the convergence trajectories it is evident that *SGD*, *MSGD* and *NAGSGD* not only converge much faster but also behave significantly less erratic if compared with all other optimizers. *RMSProp* is very unstable for initial iterations but tends to stabilize while approaching the LM estimates.



As can be seen from the left figure, *SGD*, *RMSProp* and *NAGSGD* have the steepest decay in cost values over successive iterations, whereas *ADAM* decays multiple times slower than the rest.

Thus from a computational view, SGD is the most efficient and also reaches the lowest cost at the last iteration ($i = 200$).

The table below quantifies the final cost values for all optimizers.

Cost	GD	SGD	MSGD	NAGSGD	AdaGrad	RMSProp	ADAM
$C_{n=200}$	0.5809	0.5612	0.5667	0.6362	0.6026	0.6049	0.6060