Gradient Flow Optimizers

Raghav K Singhal

New York University

Abstract

In recent years there has been a great interest in using Differential Equations to model and understand the behaviour of Optimization Algorithms, as a closer look at Gradient Descent Algorithm shows that it is an Euler-Approximation to a certain Ordinary Differential Equation. Our work here shows that we can take the Differential Equation approach further, building new classes of optimizers based on integration techniques for Differential Equations.

Gradient Flow Equation

Gradient Descent has a very simple interpretation as a First-Order Integration scheme for the gradient flow equation,

$$\dot{x}_t = -\nabla f(x_t)$$

And then based on the following result for strongly convex functions we explore using other integration schemes for the optimization task,

Proposition 1. *Let $f \in \mathcal{S}^{2,1}_{\mu,\beta}$, and suppose x^* is the global minimum, then the solution trajectory to the gradient flow equation satisfies the following:*

$$f(x_t) - f(x^*) \leq e^{-2\mu t}\big(f(x_0) - f(x^*)\big)$$
$$\|x_t - x^*\|^2 \leq e^{-2\mu t}\|x_0 - x^*\|^2$$

And also a similar but weaker result holds for general f, as the set of critical points of f are in the ω-limit set for $\dot{x}_t = -\nabla f(x_t)$. Where the ω-limit set is defined as:

$$\omega(x_0) = \{x : \forall T \text{ and } \epsilon > 0, \exists t > T, |\phi(x_t, x_0) - x| < \epsilon\}$$

where $\phi(x_t, x_0)$ is the flow of the gradient flow equation and x_0 is the initial condition.

Runge-Kutta Methods

Runge-Kutta Methods refer to a family of Integration Schemes consisting of explicit and implicit methods. There are several advantages to using Runge-Kutta methods, which can be found in the Numerical Analysis Literature.

Runge-Kutta 4th Order

This is the most well known of the Runge-Kutta Families, commonly known as RK4. This Scheme is given by:

$$\begin{aligned} k_1 &= \nabla f(x_k) \\ k_2 &= \nabla f(x_k - \frac{\alpha}{2}\nabla f(x_k)) \\ k_3 &= \nabla f(x_k - \frac{\alpha}{2}\nabla k_2) \\ k_4 &= \nabla f(x_k - \alpha k_3) \\ x_{k+1} &= x_k + \frac{\alpha}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

Runge-Kutta 2nd order - Ralston Method

$$\begin{aligned} k_1 &= \nabla f(x_k) \\ k_2 &= \nabla f(x_k - \frac{2\alpha}{3}\nabla f(x_k)) \\ x_{k+1} &= x_k + \frac{\alpha}{4}(k_1 + 3k_2) \end{aligned}$$

Runge-Kutta 2nd order - Heun's Method

$$\begin{aligned} k_1 &= \nabla f(x_k) \\ k_2 &= \nabla f(x_k - \alpha\nabla f(x_k)) \\ x_{k+1} &= x_k + \frac{\alpha}{2}(k_1 + k_2) \end{aligned}$$

Experiments

Model	RK2-Test Accuracy	RK2-Test Loss
WideResNet	93.07%	0.286
ResNet18	92.5%	0.212
Logistic Regression	92.14%	0.0015
Lasso	92.65%	0.0021

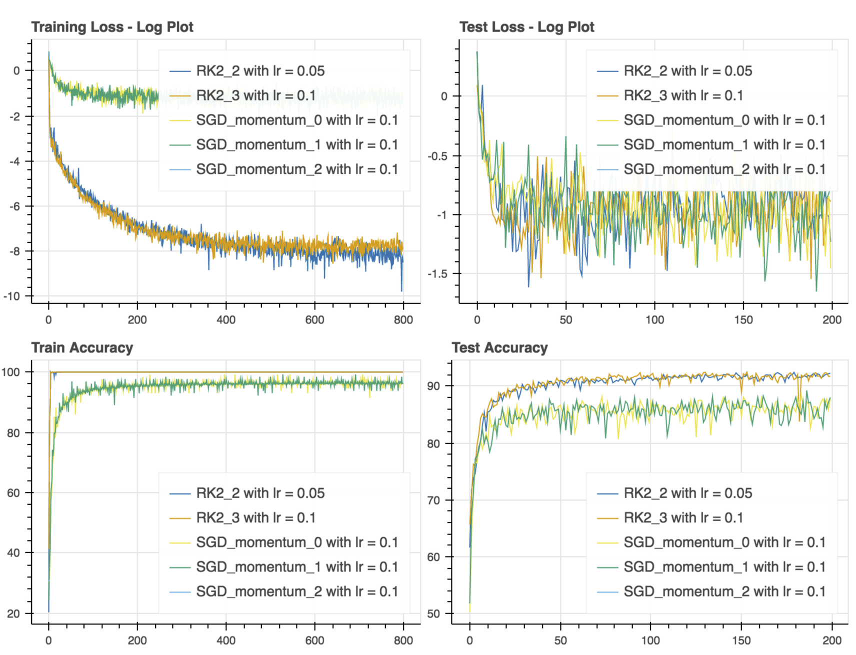
Model	SGD-Test Accuracy	SGD-Test Loss
WideResNet	92.95%	0.249
ResNet18	89.27%	0.1914
Logistic Regression	91.95%	0.00101
Lasso	92.52%	0.00210

Neural Networks

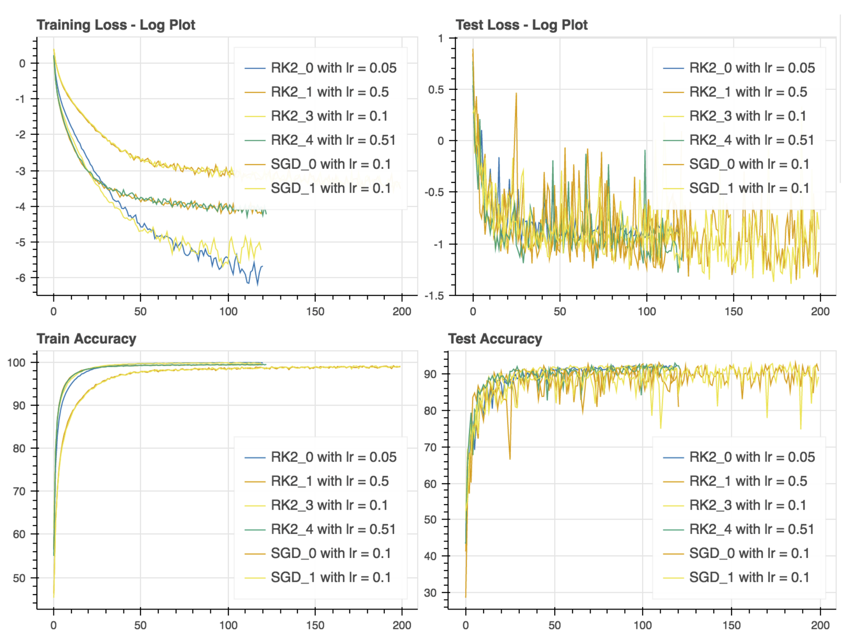
We do experiments with three main models:

- Resnet18 on CIAFR10
- WideResnet on CIAFR10
- Logistic Regression with Regularization on MNIST

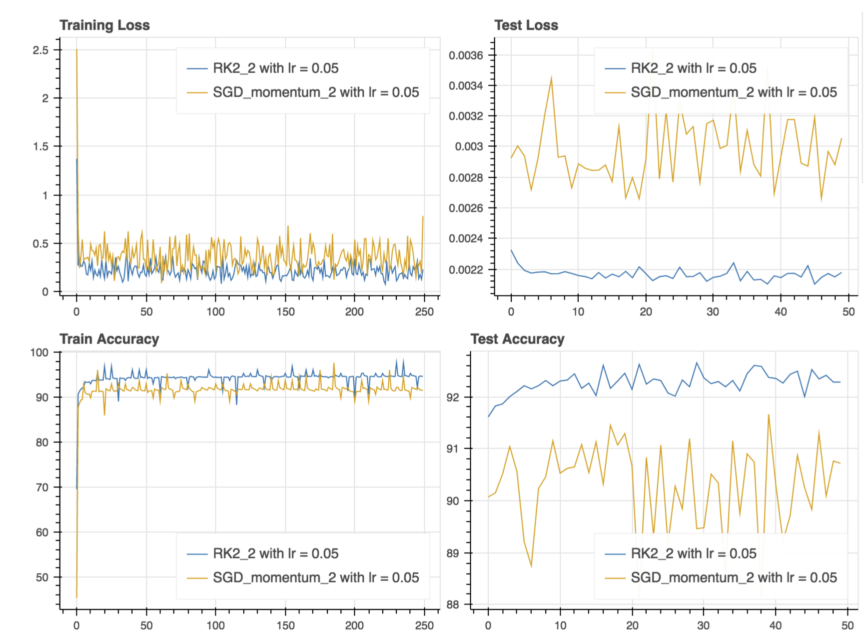
ResNet18



WideResNet



Regression



Conclusions

We aim to explore these methods further and see what concepts and methods from Numerical Analysis can shed light on the performance of Optimization Scheme, for instance stability, stiff equations, etc.

References

[1] Su, Weijie, Stephen Boyd, and Emmanuel Candes. *A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights*. Advances in Neural Information Processing Systems. 2014.

[2] Scieur, Damien, et al. *Integration Methods and Accelerated Optimization Algorithms* arXiv preprint arXiv:1702.06751 (2017).

[3] Nesterov, Yurii. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science and Business Media, 2013.