International Program on Machine Learning

Paper review
Direct Runge-Kutta Discretization Achieves Acceleration

Our team:
Elfimov Nikita
Moscow State University,
Moscow, 119234,
terilat34634@yandex.ru

Artamonov Sergey
Moscow State University,
Moscow, 119234,
serega200010@mail.ru

Bitri Rea
Epoka University
Tirana, Albania
rbitri20@epoka.edu.al

IPML, 2022-2023 years.

1 Abstract

Although the authors in the experiments generate accelerated gradient methods by setting the (s+2) times derivable ODEs, those methods are unable to fully model acceleration. Runge-Kutta integrator is one of the most popular and effective methods of solving differential equations. This method is specified with its order of accuracy $\mathcal{S}$, and it builds additional nodes between two integration nodes. The most popular R-K method is method of 4^{th} order. But we realized Runge-Kutta Dormand–Prince method with adaptive stepsize and the 5^{th} order. We see that method Runge-Kutta with different value parameter q has bad convergence. We considered different origin point and other parameter of method, but we didn't see convergence.

2 Introduction

We consider main problem in the theory of the convex optimization:

$$\min_{x\in\mathbb{R}^d} f(x), \tag{1}$$

where we suggest, that f is convex and sufficiently smooth. The classical method for this problem is gradient descent (GD). It shows a sub-optimal converge rate of $\mathcal{O}(N^{-1})$. There are Nesterov's seminal accelerated gradient method, which has converge rate $\mathcal{O}(N^{-2})$. It's central result in the theory of convex optimization. It has next view:

$$\begin{aligned} x_k &= y_{k-1} - h\nabla f(y_{k-1}),\\ y_k &= x_k + \frac{k-1}{k+2}(x_k - x_{k-1}). \end{aligned} \tag{2}$$

In paper [1] authors use idea continuous-time. They use Nesterov's formulas for creating second order ODE, which describes physical system with vanishing friction.

There are many works about considering continuous-time for (2). But generally problem for them is trouble with describing of convergence and choice of discretization for showing convergence. But method Runge-Kutta hasn't this problem, because authors consider second order equation and use integrator with a suitable step size.

In paper was considered two main result:

- Sequence of iterates generated by discretizing using a Runge-Kutta integrator converges to the optimal solution at the rate $\mathcal{O}(N^{\frac{-2s}{s+1}})$, where s is the order of the integrator. If s have big value, then rate approaches the optimal rate $\mathcal{O}(N^{-2})$.
- Classic problem in the theory of convex optimization is flatness extremum points. But in the paper was considered conditions, when give convergence rates even faster than $\mathcal{O}(N^{-2})$. In particular, in the paper was proofed, that if point have p quantities the degree of local flatness, then convergence rate is equal $\mathcal{O}(N^{-p})$.

Authors state, that their work presents the first direct discretization of an ODE that yields accelerated gradient methods. They argue that the stability inherent to the ODE and order conditions on the integrators suffice to achieve acceleration. Other work focus on variational integrators or order of integrators.

In the paper was considered numerical experiments. But these experiments was built on a data, which distribution describe how $Ax+c$, where A is linear operator. We want check theory in the paper on different distribution of data.

3 Math theory

Before we consider main results of the paper, we must define some assumptions and definitions.

Firstly, we define sublevel set:

$$\mathcal{S} := \{x \in \mathbb{R}^d | f(x) \leq \exp(1)((f(x_0) - f(x^*)) + ||x_0 - x^*||^2) + 1\}, \tag{3}$$

where x_0 is the initial point, x^* is a minimum point of (1). This subset must to hold only within a subset of $\mathcal{R}^d$, which be defined as:

$$\mathcal{A} := \{x \in \mathbb{R}^d \colon \exists x' \in \mathcal{S}, ||x - x'|| \leq 1\} \tag{4}$$

This set contains points, which have unit distance to the initial set. Unit distance is arbitrary.

So we must consider some assumption for function f.

Assumption 1. *There exists an integer $p \geq 2$ and a positive constant Lsuch that for any point $x \in \mathcal{A}$, and for all indices $i \in \{1, \ldots, p-1\}$, we have the lower-bound*

$$f(x) - f(x^*) \geq \frac{1}{L}||\nabla^{(i)} f(x)||^{\frac{p}{p-i}}, \tag{5}$$

where x^* minimizes f and $||\nabla^{(i)} f(x)||$ denotes the operator norm of the tensor $\nabla^{(i)} f(x)$.

Assumption 2. *There exists an integer $s \geq p$ and a constant $M \geq 0$, such that $f(x)$ is order $(s+2)$ differentiable. Furthermore, for any $x \in \mathcal{A}$, the following operator norm bounds holds:*

$$||\nabla^{(i)} f(x)|| \leq M, \quad \text{for } i = p, p+1, \ldots, s, s+1, s+2. \tag{6}$$

Important **Example** for the assumption 1 is considering l_p-norm regression, when $f(x) = ||Ax - b||_p^p$, where $Ax^* = b$. Because x^* have degree p of flatness. It's important for numerical experiments in the paper.

4 Replacement standard NAG with ODE

The main idea of the article [1] is to apply R-K numerical method to solve ODE system. This system is related to continuous version of NAG which is a dynamic system. The best way to describe dynamic systems is to present them as ODE system. In order to do it we get recurrent formulas (2) and combine them:

$$x_{k+1} - x_k = \frac{k-1}{k+2}(x_k - x_{k-1}) - s \nabla f(y_k) \tag{7}$$

The next step is to scale this equation and divide it by $\sqrt{s}$

$$\frac{x_{k+1} - x_k}{\sqrt{s}} = \frac{1}{\sqrt{s}} \frac{k-1}{k+2}(x_k - x_{k-1}) - \sqrt{s} \nabla f(y_k) \tag{8}$$

In continuous space X which is a source of x_k we can note $X(t)$ as some parameterized curve. Trajectories of x in the NAG process are supposed to be samples from curves X which satisfy the following condition: $x_k \approx X(k \cdot \sqrt{s})$. Our goal is to find limits when $s \to 0$, so we put $k = \frac{t}{\sqrt{s}}$. This allow us to say, that $\lim\limits_{s \to 0} X(t) = x_k$. Using second order Taylor expansions with Peano additional term we get:

$$\frac{x_{k+1} - x_k}{\sqrt{s}} = \dot{X}(t) + \frac{1}{2}\ddot{X}(t)\sqrt{s} + o(\sqrt{s}) \tag{9}$$

$$\frac{x_k - x_{k-1}}{\sqrt{s}} = \dot{X}(t) - \frac{1}{2}\ddot{X}(t)\sqrt{s} + o(\sqrt{s}) \tag{10}$$

$$\sqrt{s}\nabla f(y_k) = \sqrt{s}\nabla f(X(t)) + o(\sqrt{s}) \tag{11}$$

Include (9), (10), (11) into (8) and get the following equation:

$$\dot{X}(t) + \frac{1}{2}\ddot{X}(t)\sqrt{s} + o(\sqrt{s}) = (1 - \frac{3\sqrt{s}}{t})(\dot{X}(t) - \frac{1}{2}\ddot{X}(t)\sqrt{s} + o(\sqrt{s})) - \sqrt{s}\nabla f(X(t)) + o(\sqrt{s}) \tag{12}$$

Let's compare the coefficients of $\sqrt{s}$ in (12):

$$\frac{1}{2}\ddot{X}(t) = -\frac{3}{t}\dot{X}(t) - \nabla f(X(t))$$

in other words

$$\frac{1}{2}\ddot{X}(t) + \frac{3}{t}\dot{X}(t) + \nabla f(X(t)) = 0 \tag{13}$$

Equation (13) is the main result of the article [2]. In addition, this article provides several interesting theorems and properties of (12).

Authors use Runge-Kutta integrator to converge to the optimal solution faster than the derivative of $\mathcal{O}(l = t)$. Results of the outcome are not achieved by considering continuous functions which are smooth enough, but by using the more comprehensive Lyapunov function, which consists of achieving both stability and acceleration with respect to time for continuous functions bounded on a given interval.

Taking into account Nesterov's NAG, defined by (2).

For a fixed step size $h = \frac{1}{L}$, L: Lipschitz constant of ∇f, convergence rate equals

$$f(x_k) - f^* \leq \mathcal{O}\Big(\frac{L||x_0 - x^*||^2}{k^2}\Big),$$

where x^* being minimizer of f, for any $f^* = f(x^*)$, $\frac{k-1}{k+2} \approx 1 - \frac{3}{k}$.

Rate is known to be optimal, based on experiments performed by Y. Nesterov [6, 7]. The function achieves a rate of $\mathcal{O}(\frac{1}{k})$, per experiments done by R. T. Rockafellar [8].

In this work, we derive a second-order ordinary differential equation, which is the exact limit of Nesterov's scheme.

$$\ddot{X}(t) + \frac{3}{t}\dot{X}(t) + \nabla f(X(t)) = 0, \tag{14}$$

for $t > 0$ with initial conditions $X(0) = x_0$, $\dot{X}(0) = 0$, here x_0 is the starting point in Nesterov's scheme, $\dot{X}$ denotes the time derivative of velocity $\frac{dX}{dt}$ and similarly $\ddot{X} = \frac{d^2X}{dt^2}$ denotes the accelaration.

5 Runge-Kutta method

Runge-Kutta integrator is one of the most popular and effective methods of differential equations numerical solving. Runge-Kutta method is specified with it's order of accuracy S. This method build additional nodes between two integration nodes. The most popular R-K method is method of 4^{th} order. But we realized Runge-Kutta Dormand–Prince method with adaptive stepsize and the 5^{th} order.

Consider ODE $\dot{y} = f(y, t)$ with start condition $y(0) = y_0$. This method evaluates value in one point with 4^{th} and 5^{th} order. General formulae for evaluating is:

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i,$$

where

$$k_i = f(t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} k_j).$$

Let y_k is value, which evaluating with the 5^{th} order and $\tilde{y}_k$ is value, which evaluating with the 4^{th} order. Then we evaluate error on the ntext formulae:

$$err = \text{dist}(y_k, \tilde{y}_k).$$

If error is less than some parameter, then method does step. Otherwise method doesn't step. But in any case we must evaluate new stepsize:

$$h_{\text{new}} = h \cdot \min(facmax, \max(facmin, fac \cdot (tol/err)^{\frac{1}{6}})),$$

where $facmin$, $facmax$, fac and tol are some parameters.

In the article [1] was considered only 1^{th}, 2^{th} and 4^{th}. We use their result, that better convergence matches bigger order.

6 Experiments

Authors of the main article [1] provides several numerical experiments in order to show efficiency of proposed method and compare different ways of there realisation.

By focusing only on ODEs with $q \geqslant 2$, we receive a convergence rate of $\mathcal{O}(t^p)$ for the

$$\ddot{X}(t) + \frac{2q+1}{t}\dot{X}(t) + q^2 t^{p-2}\nabla f(X(t)) = 0 \tag{15}$$

Solution to the above ODE exists only when $t > 0$, as proven by A.Wibisono, A. C.Wilson, and M. I. Jordan [9]. The above ODE can also be written as a vector field for:

$$\dot{y} = F(y) = \begin{bmatrix} -\frac{2q+1}{t}\dot{X}(t) - q^2 t^{p-2}\nabla f(X(t)) \\ v \\ 1 \end{bmatrix}$$

where $y = [v; x; t]$ with origin condition: $y_0 = [0; x_0; t_0]$.

As the iterates approach a minimum, the high order derivatives of the function f, in addition to the gradient, also converge to zero. Consequently, the trajectory slows down around the optimum and we can stably discrete the process with a large enough step size.

6.1 Choosing parameters

For finding good values for parameters we used synthetic data. This process explains in 6.2.1.

6.1.1 About choosing stepsize

Value h is stepsize for Runge-Kutta. We analyze different values for h and we got, that $h = 0.001$. For proof we show plots for different value of h in application (8.1) under $t_0 = 0.001$.

6.1.2 About choosing value for t_0

We analyze different value for t_0 and got, that $t_0 = 0.001$ is enough good point. Because if value is less, then we have problem with overflow. If value is more than 1, then we have problem with converge. Values between 0.001 and 1 are good, but we think, that 0.001 the best. For proof we show plots for different value of t_0 in application (8.2) under $h = 0.001$.

6.2 Numerical experiments

We defined good parameters for Runge-Kutta method. So we can apply our method for different data.

Such way requires to find good values of q and some of the experiments were made in order to do it. We considered values $q = 1, 2, 4, 8, 10, 12$. We consider (15) as in [1] $f(x) = ||Ax - b||^2$, where A is data, b is result of data. We consider only one order of the integrator, because our realization have more order, that in the article.

Our group have provided some same experiments in order to check results of article [1] and expand them. We tried find sources with code in the github pages of relevant people and organizations but found nothing. Then we decided to reproduce it by ourselves. We wrote python code of Runge-Kutta integrator with $s = 5$ (one of our expansions). We took such value of s because considered it more interesting, informative and unusual and at the same time potentially better variant of s. We realised simple NAG method, which described in [1] and compared them. We made the same experiments as were described in article [1].

For researching we considered 2 case, when we generated data and we took real data.

6.2.1 Synthetic data

We generated data with help next commands on python:

```
data = np.random.uniform(data_min, data_max, size=(n, m))
delta = np.random.uniform(delta_min, delta_max, size=(n,))
y = data.dot(k) + delta
```

where $(\text{data\_min}, \text{data\_max}) = (-100, 100)$ – range for generating data, $(\text{delta\_min}, \text{delta\_max}) = (-2, 2)$ – range for noise. Data and noise have uniform distribution. k – set of coefficients for generating data, y – getting data, m – number of coefficients, n – dimension of getting data.

We used for this data all considering methods: classical Gradient Descent with adaptive stepsize, our Runge-Kutta method with adaptive stepsize and Nesterov method. We got next result:

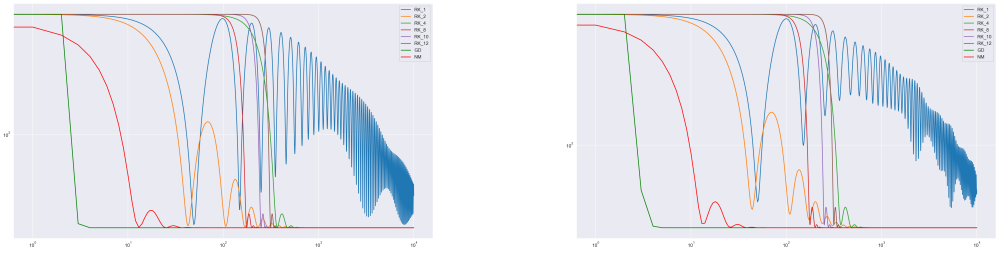

Figure 1: The abscissa axis is number of iteration, the ordinate axis is value $||Ax_k - b||$

On the left plot we consider case, when $k = (1, -1, 1, -1)$ and $x_0 = (0.9, -0.9, 0.9, -0.9)$. On the right plot $k = (1, -1, 1, -1, 1, -1, 1, -1)$ and $x_0 = (0.9, -0.9, 0.9, -0.9, 0.9, -0.9, 0.9, -0.9)$. We considered other cases with different values of k and x_0, but we didn't get qualitatively different results.

Also we think, that $q = 2$ is the best value, because under this method usually convergence and have good rating.

We may see, that Runge-Kutta method with different value parameter q has not bad convergence. But Gradient Descent is some better for synthetic data.

6.2.2 Real data

One of the goals of our work is to show that the proposed optimization method can be successfully applied to solve real problems. To show this, we are going to compare standard machine learning models with the proposed method in various settings. We will compare a simple linear regression model studied using our method and one of the most popular SGDRegressor models from sklearn, which is also a simple linear model optimized using stochastic gradient descent. There are several reasons why we decided to compare our optimization with sklearn's SGD: we want to get one of the most popular models that thousands of people use every day. Sklearn is the best choice for us because of its popularity, ease of use and accessibility. Sklearn does not use Nesterov gradient descent to optimize linear models, but this is not a problem until our theoretical studies have shown that SGD is also an excellent optimizer that can compete with NAG in many tasks.

In this section, we get several popular regression datasets from Kaggle. In this article we want to discuss two of them. We tried to find such data sets that, firstly, can be modeled quite easily by the objective function even with the help of linear models, and secondly, are included in standard benchmarks. We decided to take "diabetes"[10] and "California housing"[11]. Both of them are available for download from the sklearn.

For both of the datasets two types of experiments were provided:

Experiment 1 . We took the standard SGDRegressor and our RK model, fitted them with standard training parameters several times, varying the maximum number of iterations. As expected, starting from a certain value of this parameter, the quality of training of both models stopped changing. We compared the behavior of both models and drew conclusions.

Experiment 2 . From experiment 1, we found out that a certain number of iterations are suitable for both models. Then we used the learning rate parameter and compared how both algorithms work with its different values.

6.2.3 Discussion

Experiment 1. We fixed learning rate is equiale to 1e-2. Than we fitted both models with max_iter in range from 500 to 7000 with step=200. In both situations RK in the end overcomed SGDRegressor and showed batter value of MSE.

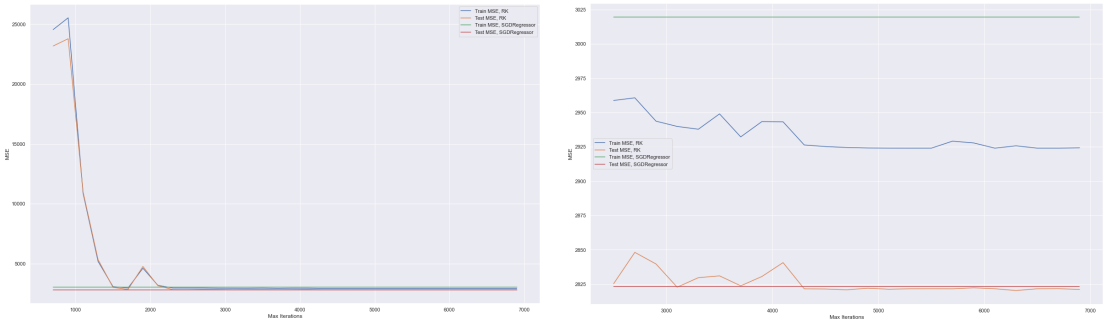

Figure 2: Mean squared error measured on "Diabetes" database.

For the "Diabetes" dataset, the best test MSE RK is 2820.3, which corresponds to $r^2 \approx 0.477$, while the best test MSE of SGDRegressor is 2823.3 and $r^2 \approx 0.476$. The results in this case are about the same, the improvement from using RK is very insignificant.

For the "California Housing" dataset, the RK optimizer was able to find the parameters of models for which MSE is equal to 0.529, while the SGDRegressor was able to reach only 0.53. In terms of r^2, this is 0.596 and 0.595, respectively.

The results in both cases are about the same, but in both cases RK was better. The question here is the ability to find solutions better than standard ones. RK succeeded in this.

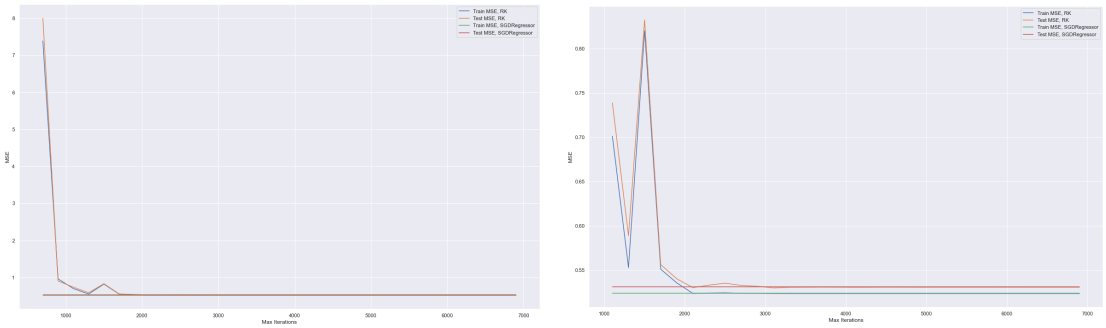

Figure 3: Mean squared error measured on "California Housing" database.

Experiment 2. Now we fix the maximum number of iterations for both optimizers at 5000 and try 5 different values of the learning rate parameter: 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}.

We find out that for both datasets, RK is better at high learning rates, and SGD is better at low values. In the Diabetes dataset, the best quality of all studies was achieved by SGD (MSE = 2813.6 for SGD with learning rate = 1e-2 versus MSE = 2821.6 for RK with learning rate = 1e-3). You can note an interesting detail: while the best qualities of all models with different learning rates are about the same, the SGD error is very large for the learning rate = 1e-1, and the RK error is very large for the learning rate = 1e-5.

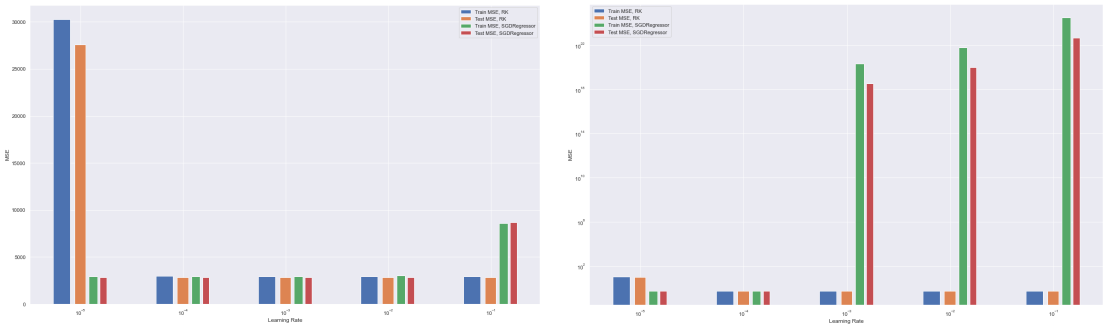

Figure 4: Mean squared error measured on "Diabetes"(left) and "CH"(right) databases.

The same situation is for "California Housing". This time, RK turned out to be the best, reaching a quality of 0.53 at a learning rate = 1e-4 while SGD could reached in best case 0.52 at a learning rate = 1e-5. You also can note that SGD doesn't converge at all at learning rates = 1e-1, 1e-2 and 1e-3 while RK is still very good for it.

7 Conclusion

We repeated numerical experiments from [1], because they have interesting result. We can see, that Runge-Kutta method converge in case with synthetic data. But rate is less than Gradient Descent method and Nesterov method. But Runge-Kutta method good converge on real data. And it have good stability on h by comparison SGDRegressor.

We found, that the initial time $t_0 = 1$ from origin paper [1] have bad influence for convergence. And we found, that $t_0 = 0.001$ is better, method have better stability and have more rate of convergence. So maybe it explains bad convergence in [1] in some considering cases.

We think the algorithm can be improved. It is obtained using the transformation of the Nesterov method. The main problem of Runge-Kutta method is that often the method quickly converges to a minimum, but it has a high velocity (if consider equation as equation of motion ball) and it needs time to converge to the minimum. By analogy with the Nesterov method, it is probably possible to add some feature that would reduce the velocity near the minimum point. But so far we have not come up with an implementation for this idea.

8 Application

8.1 Finding optimum value for h

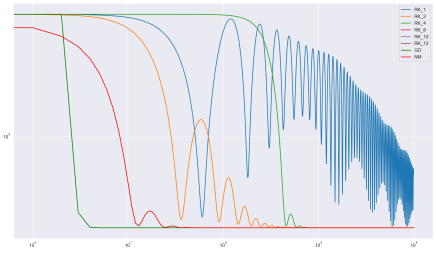

Figure 5: $h = 1$

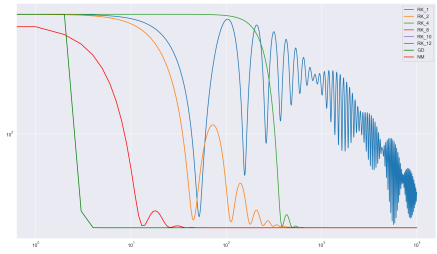

Figure 6: $h = 10^{-1}$

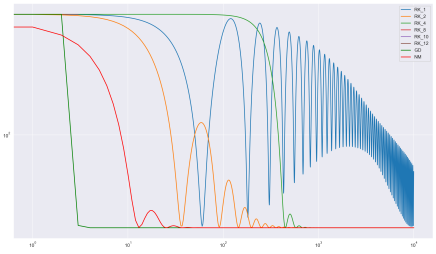

Figure 7: $h = 10^{-2}$

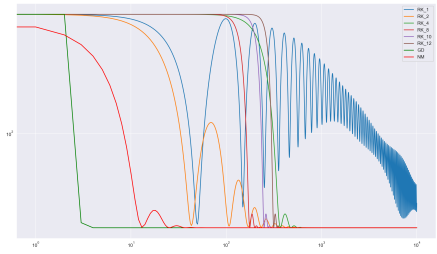

Figure 8: $h = 10^{-3}$

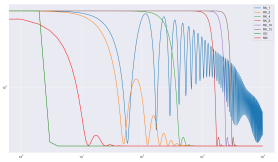

Figure 9: $h = 10^{-4}$

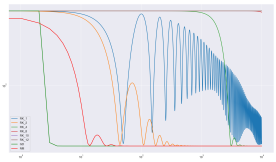

Figure 10: $h = 10^{-5}$

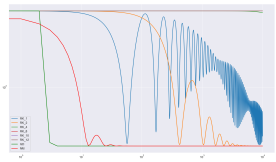

Figure 11: $h = 10^{-6}$

8.2 Finding optimum t_0

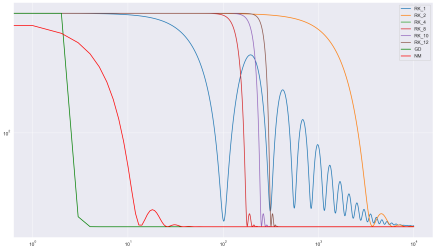

Figure 12: $t_0 = 0.00001$

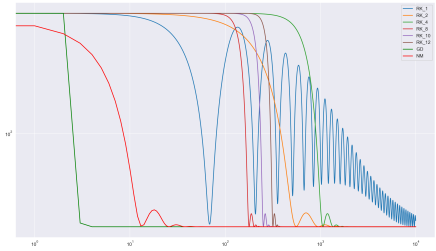

Figure 13: $t_0 = 0.0001$

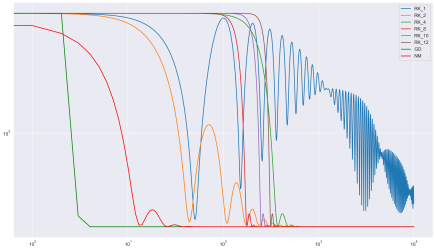

Figure 14: $t_0 = 0.001$

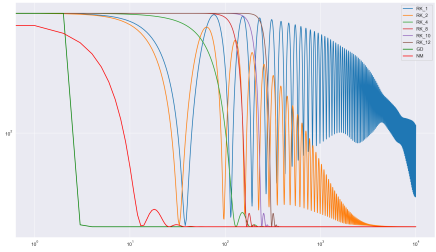

Figure 15: $t_0 = 0.01$

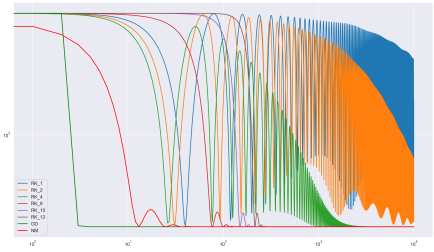

Figure 16: $t_0 = 0.1$

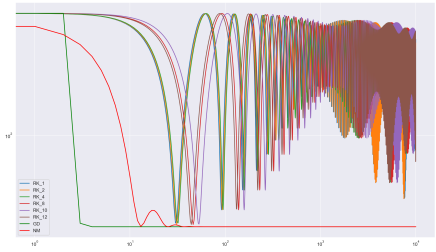

Figure 17: $t_0 = 1$

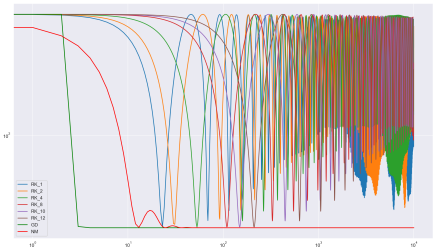

Figure 18: $t_0 = 5$

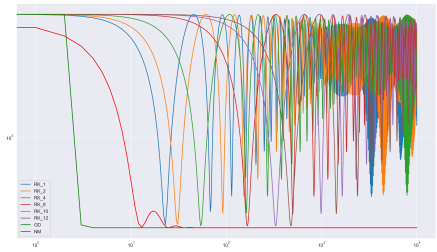

Figure 19: $t_0 = 10$

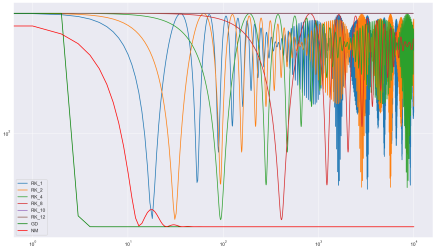

Figure 20: $t_0 = 50$

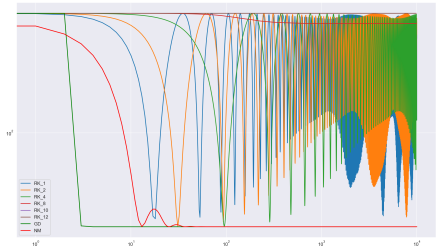

Figure 21: $t_0 = 100$

References

[1] Jingzhao Zhang, Aryan Mokhtari, Suvrit Sra, Ali Jadbabaie, Direct Runge-Kutta Discretization Achieves Acceleration

[2] . Su, S. Boyd, and E. Candes. A differential equation for modeling nesterovs accelerated gradient method: Theory and insights. In Advances in Neural Information Processing Systems, pages 2510–2518, 2014

[3] A Variational Perspective on Accelerated Methods in Optimization, Wibisono et al., 2016

[4] `https://www.kaggle.com/code/sudhirnl7/linear-regression-tutorial`

[5] `https://github.com/terilat/Runge-Kutta-in-Gradient-descent.git`

[6] Kluwer Academic Publishers, Boston, MA, 2004.

[7] Introductory lectures on convex optimization: A basic course, volume 87 of Applied Optimization.

[8] Convex analysis. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 1997. Reprint of the 1970 original, Princeton Paperbacks.

[9] A variational perspective on accelerated methods in optimization. Proceedings of the National Academy of Sciences, 113(47):E7351–E7358, 2016.

[10] Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., & Johannes, R. S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Symposium on Computer Applications and Medical Care*(pp. 261–265). IEEE Computer Society Press.

[11] Pace, R. Kelley, and Ronald Barry. "Sparse spatial autoregressions." Statistics & Probability Letters 33.3 (1997): 291-297.