On the connection between stochastic optimization and splitting scheme for ODE.

Anonymous Authors[1]

Abstract

We present different view on stochastic optimization, which goes back to the splitting schemes for approximate solutions of ODE. In this work we provide a connection between stochastic gradient descent approach and first order splitting scheme for ODE. We present, that the Kaczmarz method is the limit case of the splitting scheme for unitary batch SGD linear least squares approach. We support our findings with empirical tests.

1. Introduction

A lot of practical problems arising in machine learning require minimization of a finite sample average which can be written in the form

$$f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\theta}) \to \min_{\boldsymbol{\theta}\in\mathbb{R}^p}, \tag{1}$$

where the sum goes over the *minibatches* of the original dataset. Vanilla stochastic gradient descent (SGD) method (Robbins & Monro, 1951) consists sequential steps in the direction of the gradient of $f_i(\boldsymbol{\theta})$, where i is to be chosen randomly from 1 to n without replacement.

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \nabla f_i. \tag{2}$$

Gradient descent method can be considered as an Euler discretization of the ordinary differential equation (ODE) of the form of the gradient flow

$$\frac{d\boldsymbol{\theta}}{dt} = -\nabla f(\boldsymbol{\theta}). \tag{3}$$

In continuous time, SGD if often analyzed by introducing a noise into the right-hand side of (3). However, for real dataset the distribution of the noise obtained by replacing the full gradient by its minibatch variant is not known and can be different for different problems. Instead, we propose a new view on the SGD as a *first-order splitting scheme* for (3), thus shedding a new light on SGD-type algorithms. This representation allows to use more efficient splitting schemes for the approximation of the full gradient flow. We show, that second-order Marchuk/Strang splitting scheme ((Marchuk, 1968), (Strang, 1968)) provides faster convergence

Contributions

- We show, that vanilla SGD could be considered as a splitting scheme for a full gradient flow.
- We demonstrate the connection between rebalancing splitting and stochastic average gradient method.
- We propose new optimization method, SAG2 based on second order splitting scheme and show that it gives better convergence, than the standard SAG method.

2. SGD as a splitting scheme

We want to establish the connection between splitting scheme for ODE and stochastic optimization. In this section we firstly consider simple ODE, where we can apply splitting idea and corresponding minimization problem.

2.1. Splitting schemes for ODEs

The best example to start from is simple ODE with right-hand-side, consisting of two summands:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\left(g_1(\boldsymbol{\theta}) + g_2(\boldsymbol{\theta})\right) \tag{4}$$

Suppose, we want to find the solution $\boldsymbol{\theta}(h)$ of (4) via integrating it on the small timestep h. The first order splitting scheme defined by solving first:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2} g_1(\boldsymbol{\theta}), \quad \boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$$

with exact solution $\boldsymbol{\theta}_1(h)$ at the moment h , followed by

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2} g_2(\boldsymbol{\theta}), \quad \boldsymbol{\theta}(0) = \boldsymbol{\theta}_1(h)$$

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

with exact solution $\boldsymbol{\theta}_2(h)$ at the moment h. Thus, the first order approximation could be written as a combinations of both solutions:

$$\boldsymbol{\theta}^I(h) = \boldsymbol{\theta}_2(h) \circ \boldsymbol{\theta}_1(h) \circ \boldsymbol{\theta}_0,$$

while the second order scheme takes 3 substeps:

$$\boldsymbol{\theta}^{II}(h) = \boldsymbol{\theta}_1\left(\frac{h}{2}\right) \circ \boldsymbol{\theta}_2(h) \circ \boldsymbol{\theta}_1\left(\frac{h}{2}\right) \circ \boldsymbol{\theta}_0$$

Order of scheme defines the degree of polynomial of h, up to which the true solution and approximation are coincide. The local error of both schemes is given by Baker - Campbell - Hausdorff formula ((Baker, 1901), (Campbell, 1896), (Hausdorff, 1906)) (see (MacNamara & Strang, 2016) for the details)

$$\boldsymbol{\theta}^I(h) - \boldsymbol{\theta}(h) = \frac{h^2}{2}[g_1, g_2] + \mathcal{O}(h^3), \tag{5}$$

$$\begin{aligned}&\boldsymbol{\theta}^{II}(h) - \boldsymbol{\theta}(h) = \\ &= \frac{h^3}{24}\left([[g_1, g_2], g_1] + 2[[g_1, g_2], g_2]\right) + \mathcal{O}\left(h^4\right)\end{aligned} \tag{6}$$

where $[g_1, g_2] = \frac{dg_1}{d\boldsymbol{\theta}}g_2 - \frac{dg_2}{d\boldsymbol{\theta}}g_1$ stands for commutator of the vector fields g_1 and g_2. The $\boldsymbol{\theta}_0 = \boldsymbol{\theta}(0)$ for initial condition of original ODE.

Note, that the basic idea of splitting could be also applied, when the number of terms in the right-hand side of ODE is greater, than two. In this case splitting scheme will take the following form:

$$\boldsymbol{\theta}^I(h) = \boldsymbol{\theta}_m(h) \circ \boldsymbol{\theta}_{m-1} \circ \ldots \circ \boldsymbol{\theta}_2(h) \circ \boldsymbol{\theta}_1(h) \circ \boldsymbol{\theta}_0 \tag{7}$$

$$\begin{aligned}\boldsymbol{\theta}^{II}(h) = \boldsymbol{\theta}_1\left(\frac{h}{2}\right) \circ \boldsymbol{\theta}_2\left(\frac{h}{2}\right) \circ \ldots \boldsymbol{\theta}_m(h) \circ \ldots \\ \ldots \circ \boldsymbol{\theta}_2\left(\frac{h}{2}\right) \circ \boldsymbol{\theta}_1\left(\frac{h}{2}\right) \circ \boldsymbol{\theta}_0\end{aligned} \tag{8}$$

2.2. SGD as approximation for the Gradient Flow equation

Now we consider classical SGD method as a splitting scheme for the full gradient descent ((Cauchy, 1847)). Suppose, we have the simplest ($m = 2$) finite sum minimization problem:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^n} \frac{1}{m}\sum_{i=1}^m f_i(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta} \in \mathbb{R}^n} \frac{1}{2}\left(f_1(\boldsymbol{\theta}) + f_2(\boldsymbol{\theta})\right)$$

Let us denote by $g_i^k = \nabla f_i(\boldsymbol{\theta}_k)$, than, the vanilla gradient descent will be written as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h \cdot \frac{1}{2}\left(g_1^k + g_2^k\right),$$

while SGD version will take steps iteratively over minibatch gradient directions:

$$\begin{aligned}\boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k - h \cdot g_1^k \\ \boldsymbol{\theta}_{k+2} &= \boldsymbol{\theta}_{k+1} - h \cdot g_2^{k+1}\end{aligned}$$

These two iterations forms an epoch in SGD approach. Each of the substeps can be considered as a forward Euler method for the discretization of the ODE for a timestep h

$$\begin{aligned}\frac{d\boldsymbol{\theta}^I}{dt} &= -g_1, \quad \boldsymbol{\theta}^I(0) = \boldsymbol{\theta}_k, \\ \frac{d\boldsymbol{\theta}^{II}}{dt} &= -g_2, \quad \boldsymbol{\theta}^{II}(0) = \boldsymbol{\theta}^I(h),\end{aligned}$$

therefore the final result for a sufficiently small h approximates the gradient flow for at time $t+h$. The vanilla gradient descent, however, is only approximating the gradient flow at time $t + h/2$. For larger number of minibatches, rather then the GD flow. One can notice, that in SGD we use a very simple time integration inside the substep. In some cases, we can integrate the subproblem exactly, without using forward Euler scheme. Generalized linear models are among such problems, but we will first study the linear least squares case in more details, since in this case we can obtain non-trivial error bounds.

2.3. Splitting approximation for the Gradient Flow equation

It is interesting to look how the pure splitting scheme corresponds to the SGD approach. For this purpose we consider illustrative example of Gradient Flow equation 9, where the right-hand side of ODE is just the sum of operators acting on $\boldsymbol{\theta}$, which allows us to apply splitting scheme approximation directly.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\sum_{i=1}^2 \nabla f_i(\boldsymbol{\theta}) = -\frac{1}{2}\nabla f_1(\boldsymbol{\theta}) - \frac{1}{2}\nabla f_2(\boldsymbol{\theta}) \tag{9}$$

In order to establish the connection between splitting scheme and SGD we use the Euler discretization below:

Splitting step	Euler discretization
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\nabla f_1(\boldsymbol{\theta})$	$\tilde{\boldsymbol{\theta}}_I = \boldsymbol{\theta}_0 - \frac{h}{2}\nabla f_1(\boldsymbol{\theta}_0)$
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\nabla f_2(\boldsymbol{\theta})$	$\boldsymbol{\theta}_I = \tilde{\boldsymbol{\theta}}_I - \frac{h}{2}\nabla f_2(\tilde{\boldsymbol{\theta}}_I)$

SGD epoch	First order splitting
$\tilde{\boldsymbol{\theta}}_{SGD} = \boldsymbol{\theta}_0 - h\nabla f_1(\boldsymbol{\theta}_0)$	$\tilde{\boldsymbol{\theta}}_I = \boldsymbol{\theta}_0 - \frac{h}{2}\nabla f_1(\boldsymbol{\theta}_0)$
$\boldsymbol{\theta}_{SGD} = \tilde{\boldsymbol{\theta}}_{SGD} - h\nabla f_2(\tilde{\boldsymbol{\theta}}_{SGD})$	$\boldsymbol{\theta}_I = \tilde{\boldsymbol{\theta}}_I - \frac{h}{2}\nabla f_2(\tilde{\boldsymbol{\theta}}_I)$

Thus, we can conclude, that *one epoch of SGD is just the splitting scheme for the discretized Gradient Flow ODE with $2 \cdot h$ step size ($m \cdot h$ in case of m batches)*

This idea gives additional intuition on the method. Both approaches are solving local problems through the Euler discretization. Given an information about the Euler scheme limitation, why not solve these local problems more accurate? While Euler's scheme is first order accurate, there is no point in using higher order splitting scheme's in such setting. Therefore, total error is defined by the accuracy of splitting and local problem integration. We should improve them both to achieve gain.

3. Applications

3.1. Linear least squares

3.1.1. Problem

Let $f_i(\boldsymbol{\theta}) = \|\mathbf{x_i}^\top \boldsymbol{\theta} - y_i\|^2$, then problem (1) is the linear least squares problem, which can be written as

$$f(\boldsymbol{\theta}) = \frac{1}{n}\|X\boldsymbol{\theta} - \mathbf{y}\|_2^2 = \frac{1}{n}\sum_{i=1}^{s}\|X_i\boldsymbol{\theta} - \mathbf{y_i}\|_2^2 \to \min_{\boldsymbol{\theta}\in\mathbb{R}^p}, \tag{10}$$

where $X \in \mathbb{R}^{n\times p}$ and $\mathbf{y} \in \mathbb{R}^p$ and the second part of the equation stands for s mini-batches with size b regrouping ($b \cdot s = n$): $X_i \in \mathbb{R}^{b\times p}, \mathbf{y_i} \in \mathbb{R}^b$

$$\nabla_\theta f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{s} X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) \tag{11}$$

The gradient flow equation will be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{s} X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) \tag{12}$$

3.1.2. Exact solution of the local problem

On each splitting approximation step we need to solve the local problem:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) \tag{13}$$

Theorem 1. *For any matrix $\mathbf{x_i} \in \mathbb{R}^{b\times p}$, any vector of right-hand side $\mathbf{y_i} \in \mathbb{R}^b$ and initial vector of parameters $\boldsymbol{\theta}_0$, there is a solution of the ODE in (13), given by formula:*

$$\begin{aligned} \boldsymbol{\theta}(h) = Q_i e^{-\frac{1}{n}R_i R_i^\top h}\left(Q_i^\top \boldsymbol{\theta}_0 - R_i^{-\top}\mathbf{y_i}\right) + \\ + Q_i R_i^{-\top}\mathbf{y_i} + (I - Q_i Q_i^\top)\boldsymbol{\theta}_0, \end{aligned} \tag{14}$$

where $Q_i \in \mathbb{R}^{p\times b}$ and $R_i \in \mathbb{R}^{b\times b}$ stands for the QR decomposition of the matrix $\mathbf{x_i}^\top$, $\mathbf{x_i}^\top = Q_i R_i$.

Proof. Given $X_i^\top = Q_i R_i$, we have $(I - Q_i Q_i^\top)X_i^\top = 0$. Note, that Q_i is left unitary matrix, i.e. $Q_i^\top Q_i = I$.

$$\begin{aligned} \frac{d\boldsymbol{\theta}}{dt} &= -\frac{1}{n}X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) \\ (I - Q_i Q_i^\top)\frac{d\boldsymbol{\theta}}{dt} &= 0 \\ \frac{d\boldsymbol{\theta}}{dt} &= Q_i \frac{d(Q_i^\top \boldsymbol{\theta})}{dt} \qquad Q_i^\top \boldsymbol{\theta} = \boldsymbol{\eta}_\mathbf{i} \\ \frac{d\boldsymbol{\theta}}{dt} &= Q_i \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} \quad \text{integrate from 0 to } h \\ \boldsymbol{\theta}(h) &= Q_i\left(\boldsymbol{\eta}_\mathbf{i}(h) - \boldsymbol{\eta}_\mathbf{i}(0)\right) + \boldsymbol{\theta}_0 \end{aligned} \tag{15}$$

On the other hand:

$$\begin{aligned} \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = Q_i^\top \frac{d\boldsymbol{\theta}}{dt} &= -\frac{1}{n}Q_i^\top X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) = \\ &= -\frac{1}{n}Q_i^\top Q_i R_i (R_i^\top Q_i^\top \boldsymbol{\theta} - \mathbf{y_i}) = \\ &= -\frac{1}{n}\left(R_i R_i^\top \boldsymbol{\eta}_\mathbf{i} - R_i \mathbf{y_i}\right) \end{aligned} \tag{16}$$

Consider the moment of time $t = \infty$. $\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = 0$, since $\exists \boldsymbol{\theta}^*, Q_i^\top \boldsymbol{\theta}^* = \boldsymbol{\eta}_\mathbf{i}^*$. Also consider (16):

$$\begin{aligned} \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = 0 &= -\frac{1}{n}\left(R_i R_i^\top \boldsymbol{\eta}_\mathbf{i}^* - R_i \mathbf{y_i}\right) \\ R_i \mathbf{y_i} &= R_i R_i^\top \boldsymbol{\eta}_\mathbf{i}^* \end{aligned} \tag{17}$$

Now we look at the (16) with the replacement, given in (17):

$$\begin{aligned} \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} &= -\frac{1}{n}\left(R_i R_i^\top \boldsymbol{\eta}_\mathbf{i} - R_i R_i^\top \boldsymbol{\eta}_\mathbf{i}^*\right) \\ \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} &= -\frac{1}{n}R_i R_i^\top\left(\boldsymbol{\eta}_\mathbf{i} - \boldsymbol{\eta}_\mathbf{i}^*\right) \qquad \text{integrate from 0 to } h \\ \boldsymbol{\eta}_\mathbf{i}(h) - \boldsymbol{\eta}_\mathbf{i}^* &= e^{-\frac{1}{n}R_i R_i^\top h}(\boldsymbol{\eta}_\mathbf{i}(0) - \boldsymbol{\eta}_\mathbf{i}^*) \\ \text{while } \boldsymbol{\eta}_\mathbf{i}^* &= R_i^{-\top}\mathbf{y_i}, \boldsymbol{\eta}_\mathbf{i}(0) = Q_i^\top \boldsymbol{\theta}_0 \\ \boldsymbol{\eta}_\mathbf{i}(h) &= e^{-\frac{1}{n}R_i R_i^\top h}(Q_i^\top \boldsymbol{\theta}_0 - R_i^{-\top}\mathbf{y_i}) + R_i^{-\top}\mathbf{y_i} \end{aligned}$$

Using (15) we obtain the target formula

$$\begin{aligned} \boldsymbol{\theta}(h) = Q_i e^{-\frac{1}{n}R_i R_i^\top h}\left(Q_i^\top \boldsymbol{\theta}_0 - R_i^{-\top}\mathbf{y_i}\right) + \\ + Q_i R_i^{-\top}\mathbf{y_i} + (I - Q_i Q_i^\top)\boldsymbol{\theta}_0, \end{aligned}$$

□

3.1.3. Kaczmarz as the limit case of splitting

Kaczmarz method (Kaczmarz., 1937), (Strohmer & Vershynin, 2009), (Gower & Richtárik, 2015) is a well-known iterative algorithm for solving linear systems It is interesting to mention, that splitting approach immediately leads to the Kaczmarz method for solving linear system in the same setting with unit batch size.

When the batch size is equal to one, we need to do n QR decompositions for each transposed batch matrix, which is just column vector $\mathbf{x_i}$ in our case:

$$\mathbf{x_i} = \mathbf{q_i r_i} = \underbrace{\frac{\mathbf{x_i}}{\|\mathbf{x_i}\|}}_{\mathbf{q_i}} \underbrace{\|\mathbf{x_i}\|}_{\mathbf{r_i}} \tag{18}$$

Now, we need to use (14) to derive analytic local solution in that case:

$$\begin{aligned}\boldsymbol{\theta}(h) &= \frac{\mathbf{x_i}}{\|\mathbf{x_i}\|} e^{-\frac{\|\mathbf{x_i}\|^2 h}{n}} \left(\frac{\mathbf{x_i}^\top}{\|\mathbf{x_i}\|}\boldsymbol{\theta}_0 - \frac{y_i}{\|\mathbf{x_i}\|} \right) + \\ &+ \frac{\mathbf{x_i}}{\|\mathbf{x_i}\|^2} y_i + \left(I - \frac{\mathbf{x_i}\mathbf{x_i}^\top}{\|\mathbf{x_i}\|^2} \right) \boldsymbol{\theta}_0 = \\ &= \frac{\left(y_i - \mathbf{x_i}^\top \boldsymbol{\theta}_0\right)}{\|\mathbf{x_i}\|^2} \left(1 - e^{-\frac{\|\mathbf{x_i}\|^2 h}{n}} \right) \mathbf{x_i} + \boldsymbol{\theta}_0\end{aligned}$$

It can be easily seen, that:

$$\lim_{h \to \infty} \boldsymbol{\theta}(h) = \frac{\left(y_i - \mathbf{x_i}^\top \boldsymbol{\theta}_0\right)}{\|\mathbf{x_i}\|^2} \mathbf{x_i} + \boldsymbol{\theta}_0, \tag{19}$$

which is exact formula for Kaczmarz method for solving linear system. This result correlates with the statements of (Needell et al., 2014), but provides us with a new sense of similarity between SGD and Kaczmarz method.

3.1.4. Upper bound on the global splitting error

Suppose, that we have only two batches, and the problem (10) is consistent, i.e. there exists an exact solution $\boldsymbol{\theta}_*$ such as $X\boldsymbol{\theta}_* = \mathbf{y}$. The GD flow has the form

$$\begin{aligned}\frac{d\boldsymbol{\theta}}{dt} &= -X^\top(X\boldsymbol{\theta} - \mathbf{y}) = -X^\top X(\boldsymbol{\theta} - \boldsymbol{\theta}_*) = \\ &= -(X_1^\top X_1 + X_2^\top X_2)(\boldsymbol{\theta} - \boldsymbol{\theta}_*),\end{aligned} \tag{20}$$

i.e. the splitting scheme corresponds to a linear operator splitting

$$A = A_1 + A_2, \; A = -X^\top X, \; A_i = -X_i^\top X_i, \; i = 1, 2.$$

Both A_1 and A_2 are symmetric non-negative definite matrices. Without loss of generality, we can assume that $\boldsymbol{\theta}_* = 0$, Suppose that the rank of A is r_1 and the rank of A_2 is r_2. Then, we can write them as

$$A_i = Q_i B_i Q_i^*,$$

where Q_i is an $N \times r_i$ matrix with orthonormal columns. The following Lemma gives the representation of the matrix exponents of such matrices.

Lemma 1. *Let $A = QBQ^*$, where Q is an $N \times r$ matrix with orthonormal columns, and B is an $r \times r$ matrix. Then,*

$$e^{tA} = (I - QQ^*) + Qe^{tB}Q^*. \tag{21}$$

To prove (21) we note that

$$\begin{aligned}e^{tA} &= \sum_{k=0}^{\infty} \frac{t^k A^k}{k!} = \sum_{k=0}^{\infty} \frac{t^k Q B^k Q^*}{k!} = \\ &= I - QQ^* + QQ^* + Q\sum_{k=1}^{\infty} \frac{t^k B^k}{k!} Q^* = \\ &= (I - QQ^*) + Qe^{tB}Q^*.\end{aligned}$$

Lemma 2. *Let $A_1, A_2 \in \mathbb{S}_+^p$ be the square negative semidefinite matrices, that don't have full rank, i.e.* $\operatorname{rank} A_1 \leq p$ *and* $\operatorname{rank} A_2 \leq p$*. While the sum of those matrices has full rank, i.e. $A = A_1 + A_2, \operatorname{rank} A = p$. Then, the global upper bound error will be written as follows:*

$$\lim_{t \to \infty} \|e^{A_2 t} e^{A_1 t} - e^{At}\| = \|(I - Q_2 Q_2^*)(I - Q_1 Q_1^*)\| \tag{22}$$

Proof. The proof is straightforward. We will use the low rank matrix exponential decomposition from the Lemma 1

$$e^{A_i t} = \Pi_i + Q_i e^{B_i t} Q_i^*, \text{where } \Pi_i = I - Q_i Q_i^*; i = 1, 2$$

$$\begin{aligned}&\lim_{t \to \infty} \|e^{A_2 t} e^{A_1 t} - e^{At}\| = \\ &= \lim_{t \to \infty} \|(\Pi_2 + Q_2 e^{B_2 t} Q_2^*)(\Pi_1 + Q_1 e^{B_1 t} Q_1^*) - e^{At}\| = \\ &= \lim_{t \to \infty} \|\Pi_2 \Pi_1 + Q_1 e^{B_1 t} Q_1^* \Pi_2 + \Pi_1 Q_2 e^{B_2 t} Q_2^* + \\ &+ Q_1 e^{B_1 t} Q_1^* Q_2 e^{B_2 t} Q_2^* - e^{At}\| = \\ &= \Pi_2 \Pi_1\end{aligned}$$

Since all matrices B_1, B_2, A are negative all the matrix exponentials are decaying: $\|e^{At}\| \leq e^{t\mu(A)} \, \forall t \geq 0$, where $\mu(A) = \lambda_{max}\left(\frac{A + A^\top}{2}\right)$ - the logarithmic norm. □

The graph presented on the Figure 1 describes . One can easily see significant difference between existing global upper bounds for that case (Sheng, 1994) and derived upper bound.

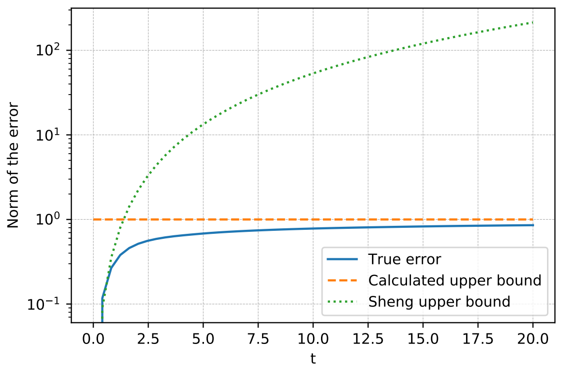


Figure 1. Global error of the splitting scheme. Initial random full rank matrix $X \in \mathbb{R}^{100\times100}$ was splitted by rows. $X_1, X_2 \in \mathbb{R}^{50\times100}$. Target matrices were obtained the following way: $A_1 = -X_1^*X_1, A_2 = -X_2^*X_2, A = -X^*X$. So A_1, A_2 are negative and lacking full rank, while $A = A_1 + A_2$ has full rank.

Theorem 2. *Let $A_1, A_2, \dots, A_b \in \mathbb{S}_+^p$ be the square negative semidefinite matrices, that don't have full rank, i.e.* $\operatorname{rank} A_i \le p,\ \forall i = 1, \dots, b$. *While the sum of those matrices has full rank, i.e. $A = \sum\limits_{i=1}^{b} A_i, \operatorname{rank} A = p$. Then, the global upper bound error will be written as follows:*

$$\lim_{t\to\infty} \|e^{A_bt}\cdot\ldots\cdot e^{A_1t} - e^{At}\| = \left\| \prod_{i=1}^{b} \Pi_{b-i+1} \right\|, \tag{23}$$

where $\Pi_i = I - Q_iQ_i^*$ and $A_i = Q_iB_iQ_i^*$ and Q_i is a matrix with orthonormal columns.

The graph on the Figure 2 shows empirical validity of the presented upper bound.

3.2. Binary logistic regression

3.2.1. Problem

In this classification task then problem (1) takes the following form:

$$-\frac{1}{n}\sum_{i=1}^{n}\Big(y_i \ln\sigma(\boldsymbol{\theta}^\top \mathbf{x_i}) + (1-y_i)\ln(1-\sigma(\boldsymbol{\theta}^\top \mathbf{x_i}))\Big) \to \min_{\boldsymbol{\theta}\in\mathbb{R}^p}, \tag{24}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, while $y_i \in \{0, 1\}$ stands for the label of the object class.

$$\nabla_\theta f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}(\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}) - y_i) \tag{25}$$

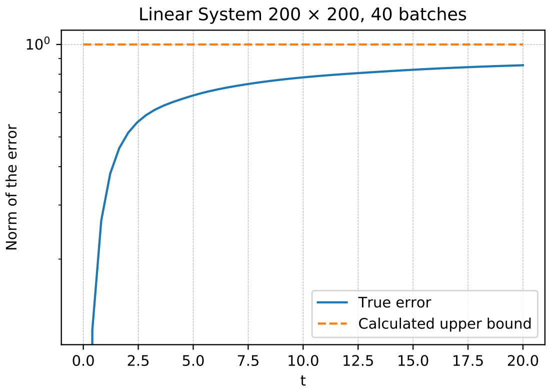


Figure 2. Global upper bound on the splitting scheme in case of 40 summands in the right-hand side.

The gradient flow equation will be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}(\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}) - y_i) \tag{26}$$

Our particular interest lies in mini-batch reformulation of the given problem. We consider s mini-batches with size b regrouping ($b \cdot s = n$): $X_i \in \mathbb{R}^{b\times p}, \mathbf{y_i} \in \mathbb{R}^b$ and $\sigma(\mathbf{x})$ stands for the element-wise sigmoid function.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{s} X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right) \tag{27}$$

3.2.2. Splitting scheme and local problem

Since we are applying splitting scheme to find the approximate solution of the (27), each local problem should be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right) \tag{28}$$

Note, that this is not linear equation and cannot be solved as easy as in Theorem 1. However, we can apply the same technique to reduce the dimension of ODE, which is needed to be solved numerically.

Suppose, we have QR decomposition of each batch data matrix $X_i^\top = Q_iR_i$, then we can multiply both sides of (28) on the $(I - Q_iQ_i^\top)$ on the left.

$$\begin{aligned}
(I - Q_i Q_i^\top)\frac{d\boldsymbol{\theta}}{dt} &= (I - Q_i Q_i^\top)\frac{1}{n}X_i^\top(\mathbf{y_i} - \sigma\left(X_i\boldsymbol{\theta}\right)) \\
\frac{d\boldsymbol{\theta}}{dt} &= Q_i\frac{d(Q_i^\top\boldsymbol{\theta})}{dt} \qquad Q_i^\top\boldsymbol{\theta} = \boldsymbol{\eta}_\mathbf{i} \\
\frac{d\boldsymbol{\theta}}{dt} &= Q_i\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} \quad \text{integrate from 0 to } h \\
\boldsymbol{\theta}(h) &= Q_i\left(\boldsymbol{\eta}_\mathbf{i}(h) - \boldsymbol{\eta}_\mathbf{i}(0)\right) + \boldsymbol{\theta}_0 \qquad (29)
\end{aligned}$$

On the other hand:

$$\begin{aligned}
\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} &= Q_i^\top\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}Q_i^\top X_i^\top(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}) = \\
&= -\frac{1}{n}Q_i^\top Q_i R_i(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}) = \\
&= -\frac{1}{n}R_i(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i})
\end{aligned}$$

Recall, that each hypothesis function depends on linear function $\mathbf{x_i}^\top\boldsymbol{\theta}$, which means, that in batch reformulation it is just entries of the vector $X_i\boldsymbol{\theta}$. Since we have QR decomposition of $X_i^\top$, we can write: $X_i\boldsymbol{\theta} = R_i^\top Q_i^\top\boldsymbol{\theta} = R_i^\top\boldsymbol{\eta}_\mathbf{i}$. In other words:

$$\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = -\frac{1}{n}R_i\left(\sigma\left(R_i\boldsymbol{\eta}_\mathbf{i}\right) - \mathbf{y_i}\right), \qquad (30)$$

To sum it up, we need to solve (30) (which is much simpler, than original differential equation (28)), than substitute it to the (29) with $\boldsymbol{\eta}_\mathbf{i}(0) = Q_i^\top\boldsymbol{\theta}_0$. Note, that matrices Q_i and R_i can be computed only once before the training.

3.3. Softmax Regression

3.3.1. Problem

In this classification task then problem (1) takes the following form:

$$-\frac{1}{n}\sum_{i=1}^{n}\log\left(\frac{\mathbf{y_i}^\top e^{\Theta^\top\mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top\mathbf{x_i}}}\right) \to \min_{\Theta\in\mathbb{R}^{p\times K}}, \qquad (31)$$

where $e^{\mathbf{x}}$ is element-wise exponential function, while $\mathbf{y_i} \in \mathbb{R}^K$ stands for the one-hot encoding of the i-th object label.

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}\left(\mathbf{y_i} - \frac{e^{\Theta^\top\mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top\mathbf{x_i}}}\right)^\top \qquad (32)$$

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}\left(\mathbf{y_i} - s\left(\Theta^\top\mathbf{x_i}\right)\right)^\top \qquad (33)$$

Here we use $s(\mathbf{x})$ as a softmax function of a vector $\mathbf{x}$, i.e. $s(\mathbf{x}) = \frac{e^{\mathbf{x}}}{\mathbf{1}^\top e^{\mathbf{x}}}$ .While mini-batch reformulation will take the following form:

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{s}X_i\left(Y_i - s(\Theta^\top X_i^\top)\right)^\top, \qquad (34)$$

where $s(X) = \begin{bmatrix} | & | & & | \\ s(\mathbf{x}_{(1)}) & s(\mathbf{x}_{(2)}) & \cdots & s(\mathbf{x}_{(b)}) \\ | & | & & | \end{bmatrix}$ is a column-wise softmax function

4. Results

4.1. Iteration comparison

4.1.1. Linear Least Squares

4.1.2. Binary Logistic Regression

4.1.3. Softmax Regression

4.2. Time comparison

4.2.1. Linear Least Squares

4.2.2. Binary Logistic Regression

4.2.3. Softmax Regression

4.3. Robustness to stepsize choosing

4.3.1. Linear Least Squares

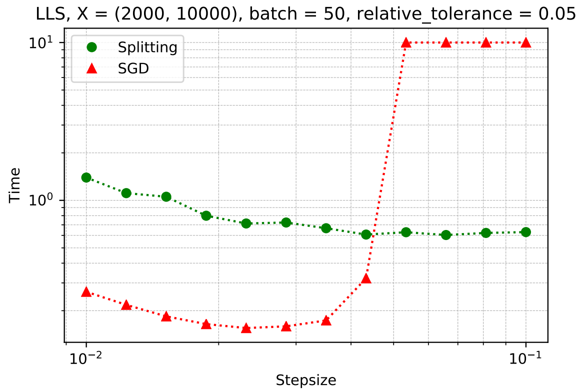


Figure 3. Random linear system. Averaging on 30 runs.

4.3.2. Binary Logistic Regression

4.3.3. Softmax Regression

5. Related work

In (Su et al., 2014) authors introduced second order ODE, which is equivalent (in the limit sense) to the gradient de-

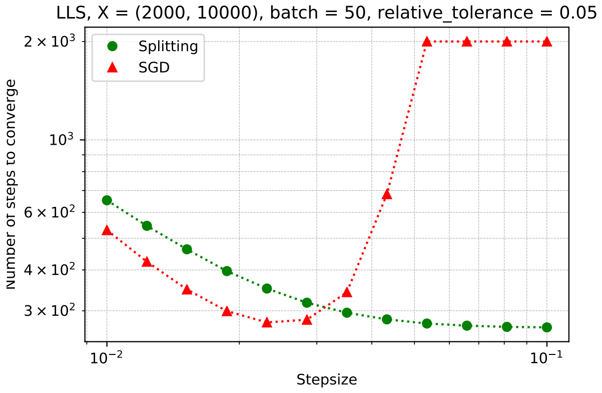


Figure 4. Random linear system. Averaging on 30 runs.

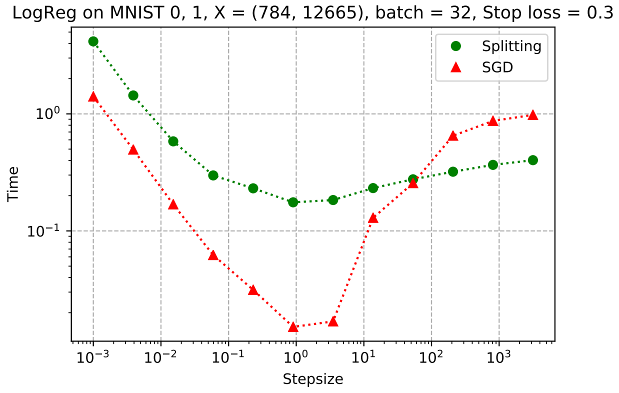


Figure 5. MNIST 0,1. Binary logistic regression. Averaging on 30 runs

scent with Nesterov momentum (Nesterov, 1983). The paper contains both formal and intuitive derivation of the proposed ODE from the iterative method itself with analogous upper bounds for general convex optimization setting and closed form solution for quadratic function. Strongly convex and composite optimizations are also covered. Theoretical conclusions are supported by strong empirical results on the variety of test functions.

Generalization of these ideas were presented in (Wibisono et al., 2016) with an arbitrary polynomial acceleration using the same parameter in ODE.

Solution dynamics of the linear least squares problem was also studied in (Osher et al., 2016) based on the linearized Bregman iteration.

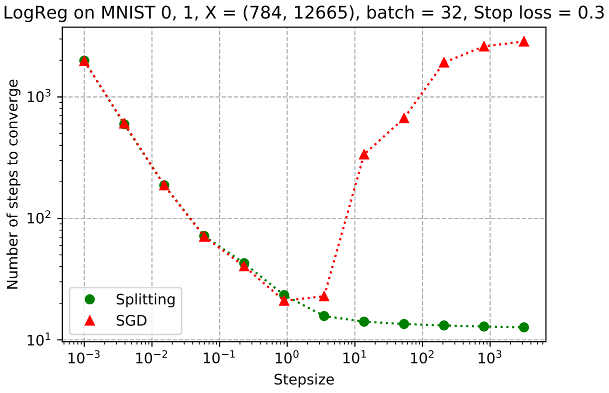


Figure 6. MNIST 0,1. Binary logistic regression. Averaging on 30 runs

General overview of interplay between continuous-time and discrete-time point of views on dynamical systems and iterative optimization methods is covered in (Helmke & Moore, 2012), (Evtushenko & Zhadan, 1994)

References

Baker, H. F. Further applications of metrix notation to integration problems. *Proceedings of the London Mathematical Society*, 1(1):347–360, 1901.

Campbell, J. On a law of combination of operators bearing on the theory of continuous transformation groups. *Proceedings of the London Mathematical Society*, 1(1): 381–390, 1896.

Cauchy, A. M'ethode g'en'erale pour la r'esolution des systemes d''equations simultan'ees. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

Evtushenko, Y. G. and Zhadan, V. G. Stable barrier-projection and barrier-newton methods in linear programming. *Computational Optimization and Applications*, 3 (4):289–303, 1994.

Gower, R. M. and Richtárik, P. Randomized iterative methods for linear systems. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1660–1690, 2015.

Hausdorff, F. Die symbolische exponentialformel in der gruppentheorie. *Ber. Verh. Kgl. SÃ chs. Ges. Wiss. Leipzig., Math.-phys. Kl.*, 58:19–48, 1906.

Helmke, U. and Moore, J. B. *Optimization and dynamical systems*. Springer Science & Business Media, 2012.

Kaczmarz., S. Angenäherte auflösung von systemen linearer gleichungen. *Bull. Internat. Acad. Polon.Sci. Lettres A*, pp. 335–357, 1937.

MacNamara, S. and Strang, G. Operator splitting. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 95–114. Springer, 2016.

Marchuk, G. I. Some application of splitting-up methods to the solution of mathematical physics problems. *Aplikace matematiky*, 13(2):103–132, 1968.

Needell, D., Ward, R., and Srebro, N. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in neural information processing systems*, pp. 1017–1025, 2014.

Nesterov, Y. E. A method of solving a convex programming problem with convergence rate $o(k^2)$. In *Doklady Akademii Nauk*, volume 269, pp. 543–547. Russian Academy of Sciences, 1983.

Osher, S., Ruan, F., Xiong, J., Yao, Y., and Yin, W. Sparse recovery via differential inclusions. *Applied and Computational Harmonic Analysis*, 41(2):436–469, 2016.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Sheng, Q. Global error estimates for exponential splitting. *IMA Journal of Numerical Analysis*, 14(1):27–56, 1994.

Strang, G. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

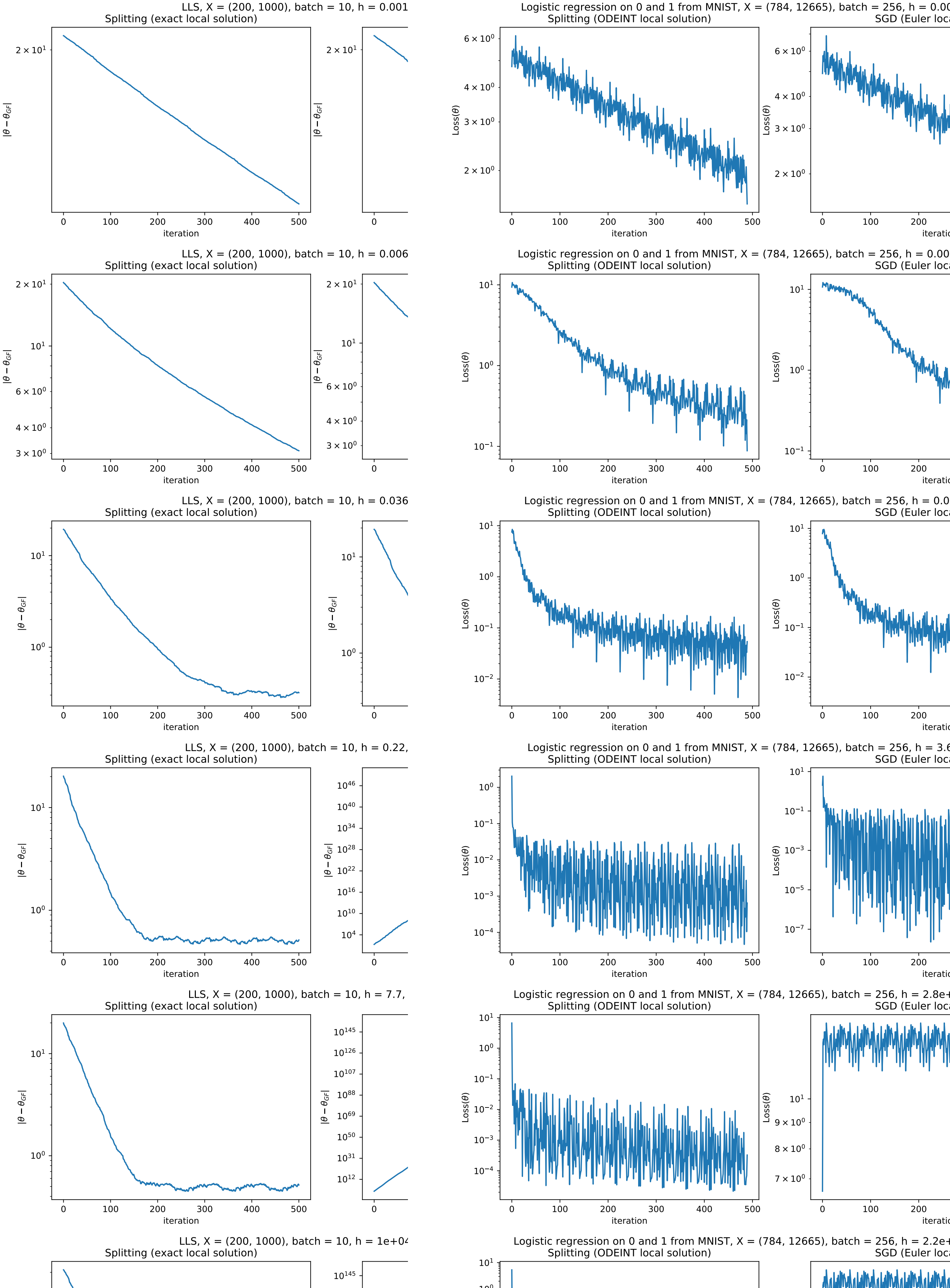
Strohmer, T. and Vershynin, R. A randomized kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(2):262, 2009.

Su, W., Boyd, S., and Candes, E. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems*, pp. 2510–2518, 2014.

Wibisono, A., Wilson, A. C., and Jordan, M. I. A variational perspective on accelerated methods in optimization. *proceedings of the National Academy of Sciences*, 113(47): E7351–E7358, 2016.

A. Proofs

B. Additional graphs


LLS, X = (200, 1000), batch = 10, h = 0.001
Splitting (exact local solution)
$|\theta - \theta_{GF}|$
2×10^1
iteration
0
100
200
300
400
500
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 0.00
Splitting (ODEINT local solution)
SGD (Euler loca
Loss(θ)
6×10^0
4×10^0
3×10^0
2×10^0
LLS, X = (200, 1000), batch = 10, h = 0.006
Splitting (exact local solution)
2×10^1
10^1
6×10^0
4×10^0
3×10^0
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 0.00
10^1
10^0
10^{-1}
LLS, X = (200, 1000), batch = 10, h = 0.036
10^1
10^0
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 0.0
10^{-2}
LLS, X = (200, 1000), batch = 10, h = 0.22,
10^{46}
10^{40}
10^{34}
10^{28}
10^{22}
10^{16}
10^{10}
10^4
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 3.6
10^{-3}
10^{-4}
10^{-5}
10^{-7}
LLS, X = (200, 1000), batch = 10, h = 7.7,
10^{145}
10^{126}
10^{107}
10^{88}
10^{69}
10^{50}
10^{31}
10^{12}
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 2.8e+
9×10^0
8×10^0
7×10^0
LLS, X = (200, 1000), batch = 10, h = 1e+04
Logistic regression on 0 and 1 from MNIST, X = (784, 12665), batch = 256, h = 2.2e+