Highlights

Stochastic gradient algorithms from ODE splitting perspective

Daniil Merkulov, Ivan Oseledets

- SGD reinterpreted as Lie–Trotter operator splitting of gradient-flow ODE
- Higher-order local ODE solvers yield step-size-robust splitting optimisation
- Kaczmarz method is the exact unit-batch splitting limit for linear least squares
- Lie–Trotter ISTA: zero asymptotic bias; Strang-ISTA has $O(h^2\lambda^2)$ bias floor
- Strang-ISTA support conjectured: $j \in \operatorname{supp}(\bar{\boldsymbol{\theta}}^h) \Leftrightarrow |d_j| > \lambda/2$; numerically confirmed

Stochastic gradient algorithms from ODE splitting perspective

Daniil Merkulov[a], Ivan Oseledets[a]

[a] *Center for Computational and Data-Intensive Science and Engineering, Skolkovo Institute of Science and Technology, Bolshoy Boulevard 30, bld. 1, Moscow, Russia, 121205*

Abstract

We present a unified view of stochastic gradient methods through the lens of operator splitting for gradient-flow ODEs. We show that one epoch of SGD is precisely a first-order Lie–Trotter splitting of the gradient-flow ODE, derive a new upper bound on the global splitting error, and establish that the Kaczmarz method is the exact limiting case of unit-batch SGD for the linear least-squares problem. Replacing the Euler step by a higher-order ODE solver at each mini-batch yields a *splitting optimisation* scheme that is markedly more robust to the learning-rate choice across linear least squares, logistic regression, and softmax regression benchmarks. Beyond smooth objectives, we analyse the LASSO ($\ell_2 + \ell_1$) problem under both the first-order Lie–Trotter (ISTA) and second-order Strang splitting (Strang-ISTA): the first-order scheme converges to the exact sparse minimiser with zero asymptotic bias, whereas the Strang scheme converges to a biased fixed point whose excess objective scales as $C_D h^2 \lambda^2$, with $C_D \geq \|A \operatorname{sign}(\boldsymbol{\theta}^*)\|^2/8$. Numerical experiments confirm the h^2-scaling of the bias floor and the component-wise fixed-point shift formula; we additionally conjecture a support-characterisation criterion for Strang-ISTA fixed points ($j \in \operatorname{supp}(\bar{\boldsymbol{\theta}}^h) \Leftrightarrow |d_j| > \lambda/2$) and provide numerical evidence in its favour.

Keywords: stochastic gradient descent, operator splitting, ODE methods, LASSO, ISTA, Kaczmarz method, numerical optimization

Email addresses: daniil.merkulov@skolkovotech.ru (Daniil Merkulov), i.oseledets@skoltech.ru (Ivan Oseledets)

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 [1] 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 [2] 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 noise into the right-hand side of (3). However, for a 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 using more efficient local problem solvers for the approximation of the full gradient flow.

Contributions

- We show that vanilla SGD is a first-order Lie–Trotter splitting scheme for the full gradient-flow ODE, clarifying the role of learning rate and batch size in the continuous-time limit.
- We propose a splitting optimisation scheme that replaces the Euler step by a higher-order ODE solver at each mini-batch and show empirically that it is a stepsize-robust alternative to SGD for practical ML problems.
- We establish that the Kaczmarz (ART) method is the exact limiting case of the unit-batch splitting scheme for the linear least-squares problem.

- We analyse the LASSO ($\ell_2 + \ell_1$) problem under both Lie–Trotter (ISTA) and Strang splitting (Strang-ISTA): the first-order scheme achieves zero asymptotic bias, while the Strang scheme incurs a bias floor scaling as $C_D h^2 \lambda^2$. We characterise the component-wise fixed-point shift analytically, conjecture a support-characterisation rule for Strang-ISTA fixed points ($j \in \mathrm{supp}(\bar{\boldsymbol{\theta}}^h) \Leftrightarrow |d_j| > \lambda/2$), and confirm all predictions numerically.

2. SGD as a splitting scheme

We first consider a simple ODE where we can apply the splitting idea and the corresponding minimization problem. 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 $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}g_2(\boldsymbol{\theta}), \quad \boldsymbol{\theta}(0) = \boldsymbol{\theta}_1(h)$ 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 $\boldsymbol{\theta}^I(h) = \boldsymbol{\theta}_2(h) \circ \boldsymbol{\theta}_1(h) \circ \boldsymbol{\theta}_0$.

It is interesting to study how the pure splitting scheme [3, 4] corresponds to the SGD approach. For this purpose, we consider an illustrative example of Gradient Flow equation 5, 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.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\sum_{i=1}^{2}\nabla f_i(\boldsymbol{\theta}) = -\frac{1}{2}\nabla f_1(\boldsymbol{\theta}) - \frac{1}{2}\nabla f_2(\boldsymbol{\theta}) \tag{5}$$

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)*

Indeed, in SGD we go in the direction of the batch gradient, which stands for the Euler discretization of batch gradient flow ODE or *local ODE*. This idea gives additional intuition on the method. Given information about the Euler scheme limitation (first-order accuracy, stability issues), we propose to solve each local problem more precisely.

Table 1: The table describes the correspondence between splitting scheme for discretized Gradient Flow ODE and epoch of SGD

Splitting step	**Euler discretization**	**SGD Epoch**	**First-order splitting**
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\nabla f_1(\boldsymbol{\theta})$	$\tilde{\boldsymbol{\theta}}_I = \boldsymbol{\theta}_0 - \frac{h}{2}\nabla f_1(\boldsymbol{\theta}_0)$	$\tilde{\boldsymbol{\theta}}_{SGD} = \boldsymbol{\theta}_0 - h\nabla f_1(\boldsymbol{\theta}_0)$	$\tilde{\boldsymbol{\theta}}_I = \boldsymbol{\theta}_0 - \frac{h}{2}\nabla f_1(\boldsymbol{\theta}_0)$
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{2}\nabla f_2(\boldsymbol{\theta})$	$\boldsymbol{\theta}_I = \tilde{\boldsymbol{\theta}}_I - \frac{h}{2}\nabla f_2(\tilde{\boldsymbol{\theta}}_I)$	$\boldsymbol{\theta}_{SGD} = \tilde{\boldsymbol{\theta}}_{SGD} - h\nabla f_2(\tilde{\boldsymbol{\theta}}_{SGD})$	$\boldsymbol{\theta}_I = \tilde{\boldsymbol{\theta}}_I - \frac{h}{2}\nabla f_2(\tilde{\boldsymbol{\theta}}_I)$

3. Optimization step with ODE solver

We propose to integrate local problem more precisely instead of Euler step in SGD. Solution of the local ODE problem involves replacing gradient in the right-hand side of gradient flow ODE 4 with batch gradient version. In our experiments the explicit Runge-Kutta method [5, 6] was used via scipy [7] function odeint.

Table 2: The table presents ODE, which we need to solve at each step of the algorithm. The last column shows the ODE, which is needed to be solved at each iteration of the algorithm for each given problem.

Problem	**Loss function**	**Batch gradient**	**Initial local ODE**
Linear Least Squares	$f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{m}\|X_i\boldsymbol{\theta} - \mathbf{y_i}\|_2^2$	$\frac{1}{b}X_i^\top(X_i\boldsymbol{\theta} - \mathbf{y_i})$	$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top(X_i\boldsymbol{\theta} - \mathbf{y_i})$
Binary logistic regression	$f(\boldsymbol{\theta}) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \ln\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}) + \right.$ $\left. +(1-y_i)\ln\left(1-\sigma(\boldsymbol{\theta}^\top\mathbf{x_i})\right)\right)$	$\frac{1}{b}X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right)$	$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right)$
One FC Layer + softmax	$f(\Theta) = -\frac{1}{n}\sum_{i=1}^{n}\log\left(\frac{\mathbf{y_i}^\top e^{\Theta^\top\mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top\mathbf{x_i}}}\right)$	$\frac{1}{b}X_i^\top\left(s(\Theta^\top X_i^\top) - Y_i\right)^\top$	$\frac{d\Theta}{dt} = -\frac{1}{n}X_i^\top\left(s(\Theta^\top X_i^\top) - Y_i\right)^\top$

Typical machine learning problems involves dealing with mini-batch of size b, which is often less, than the number of trainable parameters p, which allows us to reduce dimensionality of the dynamic system via QR decomposition of each batch data matrix $X_i^\top = Q_iR_i$ (see details in the Appendix) and substitution $\boldsymbol{\eta}_i = Q_i^\top\boldsymbol{\theta}$. Note, that QR decomposition is only needed to be performed once before the training.

There is an analytical solution for each local ODE in linear least squares case:

Theorem 1. *For any matrix $\mathbf{x_i} \in \mathbb{R}^{b\times p}, b \leq p, rankX_i = b$, 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 $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top(X_i\boldsymbol{\theta} - \mathbf{y_i})$, given by formula:*

$$\boldsymbol{\theta}(h) = Q_i e^{-\frac{1}{n}R_iR_i^\top h}\left(Q_i^\top\boldsymbol{\theta}_0 - R_i^{-\top}\mathbf{y_i}\right) + Q_iR_i^{-\top}\mathbf{y_i} + (I - Q_iQ_i^\top)\boldsymbol{\theta}_0, \quad (6)$$

Algorithm 1: Splitting optimization

$\boldsymbol{\theta}_0$ - initial parameter; b - batch size; α - learning rate; m- total number of batches

$h := \alpha m$

$t := 0$

for $k = 0, 1, \ldots$ **do**

  for $i = 1, 2, \ldots, m$ **do**

    Formulate local ODE problem $\mathcal{P}_i^k$

    $\boldsymbol{\theta}_{t+1}$ = integrate $\mathcal{P}_i^k$ given an initial value $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_t$ to the step h

    $t := t + 1$

  end

end

Table 3: The table shows initial local ODE and paired $\mathcal{P}_i^k$. Note, that $\boldsymbol{\eta}_i \in \mathbb{R}^b$, while $\boldsymbol{\theta} \in \mathbb{R}^p$

Initial local ODE	$\mathcal{P}_i^k$	**Integration**
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i})$	$\frac{d\boldsymbol{\eta_i}}{dt} = -\frac{1}{n} R_i \left(R_i^\top \boldsymbol{\eta_i} - \mathbf{y_i}\right), \boldsymbol{\eta_i} = Q_i^\top \boldsymbol{\theta}$	analytical
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top \left(\sigma\left(X_i \boldsymbol{\theta}\right) - \mathbf{y_i}\right)$	$\frac{d\boldsymbol{\eta_i}}{dt} = -\frac{1}{n} R_i \left(\sigma\left(R_i^\top \boldsymbol{\eta_i}\right) - \mathbf{y_i}\right), \boldsymbol{\eta_i} = Q_i^\top \boldsymbol{\theta}$	`odeint`
$\frac{d\Theta}{dt} = -\frac{1}{n} X_i^\top \left(s(\Theta^\top X_i^\top) - Y_i\right)^\top$	$\frac{dH_i}{dt} = -\frac{1}{n} R_i (s(H_i^\top R) - Y_i)^\top, H_i = Q_i^\top \Theta$	`odeint`

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* $\mathbf{X_i}^\top$, $\mathbf{X_i}^\top = Q_i R_i$.

It is interesting to mention, that the splitting approach immediately leads to the Kaczmarz [8, 9, 10] method for solving linear system in the same setting with unit batch size.

$$\lim_{h \to \infty} \boldsymbol{\theta}(h) = \frac{\left(y_i - \mathbf{x_i}^\top \boldsymbol{\theta}_0\right)}{\|\mathbf{x_i}\|^2} \mathbf{x_i} + \boldsymbol{\theta}_0, \tag{7}$$

which is exact formula for Kaczmarz method for solving linear system. This result correlates with the statements of [11], but provides us with a new sense of similarity between SGD and Kaczmarz method.

4. Results

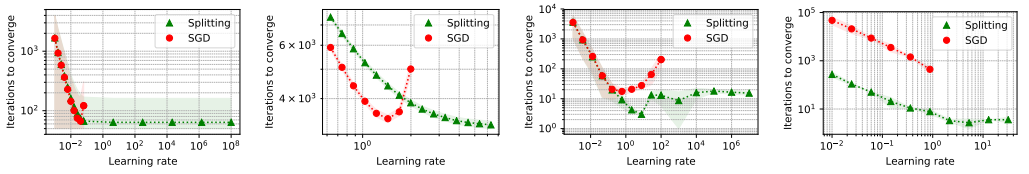
In this section, we describe the experimental setting. The majority of computations were performed on the NVIDIA DGX-2 cluster with 80 CPUs and 512 Gb RAM. We restricted the number of CPU usage per each experiment with an upper limit of 5 CPUs per experiment. All time measurements were done with the time library for Python. All experiments were done with the fixed random seed for reproducibility. For each experiment we performed 30 runs with random initialization and plotted trend line with the standard deviation.

Linear Least Squares Both random and the real linear systems were tested. For random linear system (`random lls`) we generated 10000×500 matrix with additive Gaussian noise of magnitude 0.01. Presented figures correspond to the batch size equals to 20. The real linear system (`tom lls`) is the standard tomography data from AIRTools II [12]. Solution of the linear system is the 50×50 image reconstructed from solving 12780×2500 linear system. Presented figures correspond to the batch size equals to 60. Relative error 10^{-3} was used as the stopping criterion.

Binary Logistic Regression (`logreg`) In our experiments we used two classes from MNIST [13] dataset, which corresponds to the 0 and 1 digits. The size of the batch for presented figure is 50. Test error 0.001 was used as the stopping criterion.

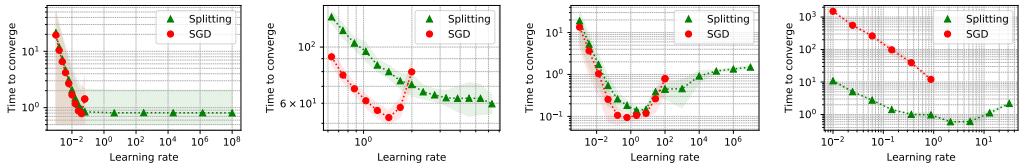
Softmax Logistic Regression (`softmax`) We took Fashion MNIST [14] dataset with 60000 grayscale pictures from 10 classes. Each example is 28×28 image. The size of the batch for presented figure is 64. Test error 0.25 was used as the stopping criterion.

On the figures below we have two labels: `SGD` and `Splitting`, which stands for batch stochastic gradient descent and proposed algorithm. We use different constant learning rates to perform our experiments. All the learning rates tested for both algorithms. Lack of point of one algorithm on the graph means reaching the limit of iterations without achieving the termination rule.



(a) `Random LLS` (b) `Tom LLS` (c) `LogReg` (d) `Softmax`

As it is expected, SGD diverges starting from some value of learning rate, which is specific for each problem. While we can see comparative robustness of the proposed splitting optimization approach.



(a) `Random LLS` (b) `Tom LLS` (c) `LogReg` (d) `Softmax`

5. Related work

Our work sits at the intersection of four research threads: continuous-time analysis of optimisation algorithms, operator splitting for differential equations, proximal gradient methods for sparse optimisation, and randomised linear algebra.

Continuous-time analysis of optimisation. In [15] authors introduced a second-order ODE that is equivalent (in the limit sense) to gradient descent with Nesterov momentum [16]. Generalization of these ideas was presented in [17] with an arbitrary polynomial acceleration using the same parameter in ODE. General overview of the interplay between continuous-time and discrete-time points of view on dynamical systems and iterative optimisation methods is covered in [18] and [19].

Splitting methods in numerical analysis. Operator splitting is a classical technique in numerical analysis for solving differential equations with composite structure. A comprehensive 161-page survey by Blanes, Casas and Murua [20] provides a unified treatment of splitting and composition methods with embedded error estimators, covering Lie–Trotter, Strang, and higher-order schemes. Their focus is on PDE applications (e.g. advection-diffusion, Hamiltonian systems) and geometric integration. Our work is complementary: we study operator splitting in the context of *machine learning optimisation*, where the composite structure arises from the finite-sum objective (1) and the sub-problems admit closed-form solutions (LLS, softmax, LASSO). In contrast to the PDE setting, we are primarily interested in asymptotic bias and convergence properties of the resulting iterative algorithms rather than short-time approximation accuracy.

Splitting for deep learning architectures. A different line of work uses splitting as an architectural principle for neural networks rather than an optimisation tool. Huang and Yin [21] propose Deep Operator-Splitting Networks (DOSnet), where each layer corresponds to one sub-step of a splitting scheme applied to a learned PDE or optimisation operator. Their goal is network expressiveness and interpretability, not convergence analysis of the training procedure. Our work focuses exclusively on the training dynamics of standard ML models (linear, logistic, LASSO) and makes no assumptions on network architecture.

Proximal gradient methods and sparse optimisation. The LASSO problem [22] and its proximal gradient solvers are a well-studied area in optimisation. ISTA and its accelerated variant FISTA [23] are the standard first-order algorithms for composite ($\ell_2 + \ell_1$) objectives, with $O(1/K)$ and $O(1/K^2)$ convergence rates, respectively. A comprehensive treatment of proximal algorithms is given in [24]. The present work reinterprets ISTA as the Lie–Trotter splitting of the gradient-flow ODE for LASSO, and derives a novel result on the asymptotic bias introduced by the second-order (Strang) splitting, which has no counterpart in the proximal methods literature.

Kaczmarz and randomised coordinate methods. The connection between SGD and Kaczmarz (ART) methods for linear systems has been studied in [9], which established exponential convergence of randomised Kaczmarz. We recover this algorithm as an exact limit case of the unit-batch splitting scheme for linear least-squares (see (7)), providing a new perspective via ODE splitting.

6. Conclusions

We have presented a unified view of stochastic gradient methods through the lens of operator splitting for gradient-flow ODEs. Our main contributions are threefold.

Splitting perspective on SGD. We showed that one epoch of SGD with mini-batches is precisely a first-order (Lie–Trotter) splitting scheme applied to the full gradient-flow ODE with an effective step size $m \cdot h$ for m batches. This connection clarifies the role of the learning rate and batch size in the continuous-time limit, and naturally motivates replacing the Euler-type integration of local sub-problems by higher-order ODE solvers. Empirically, the resulting *splitting optimization* scheme is considerably more robust to

the learning-rate choice than vanilla SGD across linear least squares, logistic regression, and softmax regression benchmarks.

Kaczmarz as a limit case. For the linear least-squares problem with unit-batch SGD, we established that the splitting scheme converges to the Kaczmarz (ART) iterates in the exact-arithmetic limit. This sheds new light on the well-studied Kaczmarz method as the canonical first-order splitting discretisation for linear systems.

Splitting order matters for sparse optimisation (LASSO). A detailed analysis of the LASSO problem reveals a qualitative difference between the first-order (Lie–Trotter / ISTA) and second-order (Strang-ISTA) splitting:

- **LT/ISTA** converges to the exact minimiser $\boldsymbol{\theta}^*$ with zero asymptotic bias (Theorem 3).
- **Strang-ISTA** converges to a biased fixed point $\bar{\boldsymbol{\theta}}^h$ satisfying $F(\bar{\boldsymbol{\theta}}^h) - F^* = C_D h^2 \lambda^2 + O(h^3)$, where $C_D \geq \|A \operatorname{sign}(\boldsymbol{\theta}^*)\|^2/8$ (Theorem 4, Remark 1).

Hence, despite the theoretical second-order accuracy advantage of the Strang scheme for smooth ODEs, the first-order splitting is preferable for the composite ($\ell_2 + \ell_1$) objective: the non-smooth ℓ_1 term introduces a fixed-point shift proportional to $h\lambda$ that the Strang scheme cannot eliminate. Numerical experiments directly confirm the h^2-scaling of the bias floor and the component-wise fixed-point shift formula (D.12). We additionally propose Conjecture 1, which characterises the support of the Strang-ISTA fixed point via the threshold rule $j \in \operatorname{supp}(\bar{\boldsymbol{\theta}}^h) \Leftrightarrow |d_j| > \lambda/2$, and provide numerical evidence consistent with it across a range of step sizes.

Future directions. Several open questions remain. Proving Conjecture 1 and obtaining the exact formula for C_D — which depends on the support structure of the Strang fixed point $\bar{\boldsymbol{\theta}}^h$, including off-support activations — are the most immediate theoretical challenges (Remark 1). Extending the splitting analysis to general composite objectives $f + g$, combining the splitting perspective with variance-reduction techniques (SVRG, SARAH), and studying higher-order splitting (Yoshida–Suzuki) for structured ML problems are natural directions.

References

[1] H. Robbins, S. Monro, A stochastic approximation method, The annals of mathematical statistics (1951) 400–407.

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

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

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

[5] J. R. Dormand, P. J. Prince, A family of embedded runge-kutta formulae, Journal of computational and applied mathematics 6 (1) (1980) 19–26.

[6] L. F. Shampine, Some practical runge-kutta formulas, Mathematics of computation 46 (173) (1986) 135–150.

[7] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, S. . . Contributors, SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, Nature Methods 17 (2020) 261–272. doi:https://doi.org/10.1038/s41592-019-0686-2.

[8] S. Kaczmarz., Angenäherte auflösung von systemen linearer gleichungen, Bull. Internat. Acad. Polon.Sci. Lettres A (1937) 335–357.

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

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

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

[12] P. C. Hansen, J. S. Jørgensen, Air tools ii: algebraic iterative reconstruction methods, improved implementation, Numerical Algorithms 79 (1) (2018) 107–137.

[13] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[14] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, arXiv preprint arXiv:1708.07747 (2017).

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

[16] Y. E. Nesterov, A method of solving a convex programming problem with convergence rate $o(k^2)$, in: Doklady Akademii Nauk, Vol. 269, Russian Academy of Sciences, 1983, pp. 543–547.

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

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

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

[20] S. Blanes, F. Casas, A. Murua, Splitting and composition methods with embedded error estimators, Acta Numerica 33 (2024) 1–161. doi:10.1017/S0962492924000011.

[21] W. Huang, W. Yin, A deeper look into deep operator splitting networks, Journal of Computational Physics 498 (2024) 112641.

[22] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society: Series B (Methodological) 58 (1) (1996) 267–288.

[23] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM journal on imaging sciences 2 (1) (2009) 183–202.

[24] N. Parikh, S. Boyd, Proximal algorithms, Foundations and Trends in Optimization 1 (3) (2014) 127–239.

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

Appendix A. Upper bound on the global splitting error

Suppose, that we have only two batches, and the problem (C.1) is consistent, i.e. there exists an exact solution $\boldsymbol{\theta}_*$ such as $X\boldsymbol{\theta}_* = \mathbf{y}$. The GD flow has the form

$$\begin{aligned} \frac{d\boldsymbol{\theta}}{dt} &= -X^\top(X\boldsymbol{\theta} - \mathbf{y}) = -X^\top X(\boldsymbol{\theta} - \boldsymbol{\theta}_*) = \\ &= -(X_1^\top X_1 + X_2^\top X_2)(\boldsymbol{\theta} - \boldsymbol{\theta}_*), \end{aligned} \tag{A.1}$$

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,*

$$e^{tA} = (I - QQ^*) + Qe^{tB}Q^*. \tag{A.2}$$

To prove (A.2) we note that

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

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:*

$$
\lim_{t \to \infty} \| e^{A_2 t} e^{A_1 t} - e^{At} \| = \| (I - Q_2 Q_2^*)(I - Q_1 Q_1^*) \| \tag{A.3}
$$

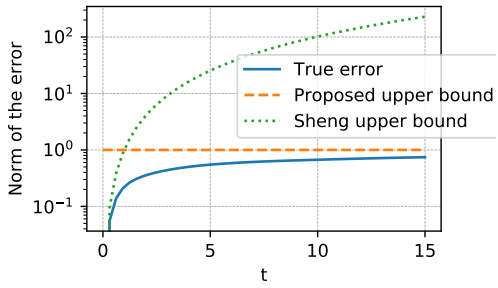
Proof. The proof is straightforward. We will use the low rank matrix exponential decomposition from the Lemma 1

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

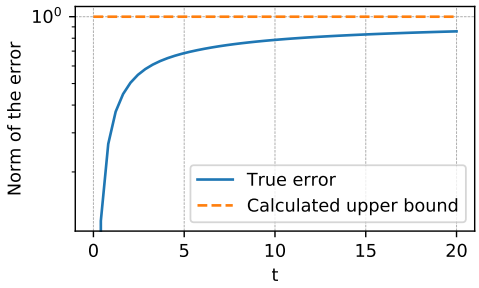
$$
\begin{aligned}
&\lim_{t \to \infty} \| e^{A_2 t} e^{A_1 t} - e^{At} \| = \\
&= \lim_{t \to \infty} \| (\Pi_2 + Q_2 e^{B_2 t} Q_2^*)(\Pi_1 + Q_1 e^{B_1 t} Q_1^*) - e^{At} \| = \\
&= \lim_{t \to \infty} \| \Pi_2 \Pi_1 + Q_1 e^{B_1 t} Q_1^* \Pi_2 + \Pi_1 Q_2 e^{B_2 t} Q_2^* + \\
&+ Q_1 e^{B_1 t} Q_1^*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

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 A.3a describes the global splitting error and the derived upper bound compared to the existing bound from [25] for the case of 2 summands in the right-hand side. One can easily see significant difference between existing global upper bounds for that case and the derived upper bound.



(a) 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.



(b) Global upper bound on the splitting scheme in case of 40 summands in the right-hand side.

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

$$\lim_{t\to\infty} \|e^{A_b t} \cdot \ldots \cdot e^{A_1 t} - e^{At}\| = \left\| \prod_{i=1}^{b} \Pi_{b-i+1} \right\|, \tag{A.4}$$

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

The graph on the Figure A.3b shows empirical validity of the presented upper bound.

Appendix B. Proofs

Theorem 1. *For any matrix $\mathbf{x_i} \in \mathbb{R}^{b\times p}, b \leq p, rank X_i = b$, 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 $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i})$, given by formula:*

$$\boldsymbol{\theta}(h) = Q_i e^{-\frac{1}{n} R_i R_i^\top h} \left(Q_i^\top \boldsymbol{\theta}_0 - R_i^{-\top} \mathbf{y_i} \right) + Q_i R_i^{-\top} \mathbf{y_i} + (I - Q_i Q_i^\top) \boldsymbol{\theta}_0, \tag{6}$$

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. $Q_i^\top Q_i = I$.

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

On the other hand:

$$\begin{aligned}
\frac{d\boldsymbol{\eta_i}}{dt} = Q_i^\top \frac{d\boldsymbol{\theta}}{dt} &= -\frac{1}{n} Q_i^\top X_i^\top (X_i\boldsymbol{\theta} - \mathbf{y_i}) = \\
&= -\frac{1}{n} Q_i^\top Q_i R_i (R_i^\top Q_i^\top \boldsymbol{\theta} - \mathbf{y_i}) = \\
&= -\frac{1}{n}\left(R_i R_i^\top \boldsymbol{\eta_i} - R_i \mathbf{y_i}\right)
\end{aligned} \tag{B.2}$$

Consider the moment of time $t = \infty$. $\frac{d\boldsymbol{\eta_i}}{dt} = 0$, since $\exists \boldsymbol{\theta}^*, Q_i^\top \boldsymbol{\theta}^* = \boldsymbol{\eta_i}^*$. Also consider (B.2):

$$\begin{aligned}
\frac{d\boldsymbol{\eta_i}}{dt} = 0 &= -\frac{1}{n}\left(R_i R_i^\top \boldsymbol{\eta_i}^* - R_i \mathbf{y_i}\right) \\
R_i \mathbf{y_i} &= R_i R_i^\top \boldsymbol{\eta_i}^*
\end{aligned} \tag{B.3}$$

Now we look at the (B.2) with the replacement, given in (B.3):

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

Using (B.1) we obtain the target formula

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

□

Appendix C. Applications

Appendix C.1. Linear least squares

Appendix C.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

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

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 ($b \cdot s = n$): $X_i \in \mathbb{R}^{b \times p}, \mathbf{y_i} \in \mathbb{R}^b$

$$\nabla_\theta f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{s} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i}) \tag{C.2}$$

The gradient flow equation will be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} \sum_{i=1}^{s} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i}) \tag{C.3}$$

Appendix C.1.2. Exact solution of the local problem

Theorem 1 gives us explicit formula for the local solution:

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

Appendix C.1.3. Kaczmarz as the limit case of splitting

Kaczmarz method [8], [9], [10] 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:

$$\mathbf{x_i} = \mathbf{q_i}\mathbf{r_i} = \underbrace{\frac{\mathbf{x_i}}{\|\mathbf{x_i}\|}}_{\mathbf{q_i}}\underbrace{\|\mathbf{x_i}\|}_{\mathbf{r_i}} \tag{C.4}$$

Now, we need to use (6) 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:

$$\lim_{h\to\infty}\boldsymbol{\theta}(h) = \frac{\left(y_i - \mathbf{x_i}^\top\boldsymbol{\theta}_0\right)}{\|\mathbf{x_i}\|^2}\mathbf{x_i} + \boldsymbol{\theta}_0, \tag{C.5}$$

which is exact formula for Kaczmarz method for solving linear system. This result correlates with the statements of [11], but provides us with a new sense of similarity between SGD and Kaczmarz method.

Appendix C.2. Binary logistic regression

Appendix C.2.1. Problem

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

$$-\frac{1}{n}\sum_{i=1}^{n}\left(y_i \ln\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}) + (1-y_i)\ln(1-\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}))\right) \to \min_{\boldsymbol{\theta}\in\mathbb{R}^p}, \tag{C.6}$$

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.

$$\nabla_\theta f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}(\sigma(\boldsymbol{\theta}^\top\mathbf{x_i}) - y_i) \tag{C.7}$$

The gradient flow equation will be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{x_i}(\sigma(\boldsymbol{\theta}^\top \mathbf{x_i}) - y_i) \tag{C.8}$$

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.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{s} X_i^\top \left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right) \tag{C.9}$$

Appendix C.2.2. Splitting scheme and local problem

Since we are applying splitting scheme to find the approximate solution of the (C.9), each local problem should be written as follows:

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top \left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right) \tag{C.10}$$

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 $X_i^\top = Q_i R_i$, then we can multiply both sides of (C.10) on the $(I - Q_i Q_i^\top)$ on the left.

$$\begin{aligned}
(I - Q_i Q_i^\top)\frac{d\boldsymbol{\theta}}{dt} &= (I - Q_i Q_i^\top)\frac{1}{n} X_i^\top (\mathbf{y_i} - \sigma\left(X_i\boldsymbol{\theta}\right)) \\
\frac{d\boldsymbol{\theta}}{dt} &= Q_i \frac{d(Q_i^\top \boldsymbol{\theta})}{dt} \quad Q_i^\top \boldsymbol{\theta} = \boldsymbol{\eta_i} \\
\frac{d\boldsymbol{\theta}}{dt} &= Q_i \frac{d\boldsymbol{\eta_i}}{dt} \quad \text{integrate from 0 to } h \\
\boldsymbol{\theta}(h) &= Q_i \left(\boldsymbol{\eta_i}(h) - \boldsymbol{\eta_i}(0)\right) + \boldsymbol{\theta}_0
\end{aligned} \tag{C.11}$$

On the other hand:

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

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: $X_i\boldsymbol{\theta} = R_i^\top Q_i^\top \boldsymbol{\theta} = R_i^\top \boldsymbol{\eta_i}$. In other words:

$$\frac{d\boldsymbol{\eta_i}}{dt} = -\frac{1}{n} R_i \left(\sigma\left(R_i^\top \boldsymbol{\eta_i}\right) - \mathbf{y_i}\right), \tag{C.12}$$

To sum it up, we need to solve (C.12) (which is much simpler, than original differential equation (C.10)), than substitute it to the (C.11) with $\boldsymbol{\eta_i}(0) = Q_i^\top \boldsymbol{\theta}_0$. Note, that matrices Q_i and R_i can be computed only once before the training.

Appendix C.3. Softmax Regression

Appendix C.3.1. Problem

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

$$-\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{\mathbf{y_i}^\top e^{\Theta^\top \mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top \mathbf{x_i}}}\right) \to \min_{\Theta \in \mathbb{R}^{p\times K}}, \tag{C.13}$$

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.

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{x_i}\left(\mathbf{y_i} - \frac{e^{\Theta^\top \mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top \mathbf{x_i}}}\right)^\top \tag{C.14}$$

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{x_i}\left(\mathbf{y_i} - s\left(\Theta^\top \mathbf{x_i}\right)\right)^\top \tag{C.15}$$

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:

$$\nabla_\Theta f(\Theta) = -\frac{1}{n}\sum_{i=1}^{s} X_i^\top \left(Y_i - s(\Theta^\top X_i^\top)\right)^\top, \tag{C.16}$$

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. Indeed, in a very similar manner to the binary logistic regression

we can write down gradientflow ODE for softmax regression in a mini-batch form:

$$\frac{d\Theta}{dt} = -\frac{1}{n}\sum_{i=1}^{s} X_i^\top \left(s(\Theta^\top X_i^\top) - Y_i\right)^\top \tag{C.17}$$

Splitting method requires the local problem, which is focused on a single minibatch:

$$\frac{d\Theta}{dt} = -\frac{1}{n} X_i^\top \left(s(\Theta^\top X_i^\top) - Y_i\right)^\top \tag{C.18}$$

$$\begin{aligned}
(I - Q_i Q_i^\top)\frac{d\Theta}{dt} &= (I - Q_i Q_i^\top)\frac{1}{n} X_i^\top (Y_i - s(\Theta^\top X_i^\top))^\top \\
\frac{d\Theta}{dt} &= Q_i \frac{d(Q_i^\top \Theta)}{dt} \quad Q_i^\top \Theta = H_i \\
\frac{d\Theta}{dt} &= Q_i \frac{dH_i}{dt} \quad \text{integrate from 0 to } h \\
\Theta(h) &= Q_i \left(H_i(h) - H_i(0)\right) + \Theta_0
\end{aligned} \tag{C.19}$$

On the other hand:

$$\begin{aligned}
\frac{dH_i}{dt} = Q_i^\top \frac{d\Theta}{dt} &= -\frac{1}{n} Q_i^\top X_i^\top (s(\Theta^\top X_i^\top) - Y_i)^\top = \\
&= -\frac{1}{n} Q_i^\top Q_i R_i (s(\Theta^\top X_i^\top) - Y_i)^\top = \\
&= -\frac{1}{n} R_i (s(\Theta^\top X_i^\top) - Y_i)^\top = \\
&= -\frac{1}{n} R_i (s(H_i^\top R) - Y_i)^\top
\end{aligned}$$

Now we need to solve ODE of variable of the size $b \times k$, rather, than $p \times k$.

Appendix D. LASSO with Splitting Schemes

We now consider the LASSO problem [22] as a concrete instance where the choice of splitting order has a measurable impact on convergence. Let

$$F(\boldsymbol{\theta}) = \frac{1}{2}\|A\boldsymbol{\theta} - \mathbf{y}\|^2 + \lambda\|\boldsymbol{\theta}\|_1 \;\to\; \min_{\boldsymbol{\theta}\in\mathbb{R}^d}, \tag{D.1}$$

where $A \in \mathbb{R}^{n\times d}$, $\mathbf{y} \in \mathbb{R}^n$, and $\lambda > 0$. Let $L = \|A^\top A\|$ (spectral norm), $\boldsymbol{\theta}^*$ a minimizer of (D.1), and $0 < h \leq 1/L$.

Appendix D.1. Lie–Trotter (Proximal Gradient / ISTA)

The first-order (Lions–Teboulle / LT) splitting applied to the gradient-flow ODE for (D.1) yields the standard proximal gradient step:

$$\boldsymbol{\theta}_{k+1} = \mathrm{prox}_{h\lambda\|\cdot\|_1}\big(\boldsymbol{\theta}_k - hA^\top(A\boldsymbol{\theta}_k - \mathbf{y})\big), \tag{D.2}$$

which coincides with ISTA [23]. The following result characterises its asymptotic bias.

Theorem 3 (LT zero algorithmic bias)**.** *Let* $\bar{\boldsymbol{\theta}}_K = \frac{1}{K}\sum_{k=0}^{K-1}\boldsymbol{\theta}_{k+1}$ *be the averaged iterates of* (D.2)*. Then*

$$F(\bar{\boldsymbol{\theta}}_K) - F(\boldsymbol{\theta}^*) \;\le\; \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|^2}{2hK}. \tag{D.3}$$

In particular, $\liminf_{K\to\infty}\big[F(\bar{\boldsymbol{\theta}}_K) - F(\boldsymbol{\theta}^*)\big] = 0$*: the LT iterate has* zero algorithmic bias*.*

Proof. Write $F = f + g$ where $f(\boldsymbol{\theta}) = \frac{1}{2}\|A\boldsymbol{\theta} - \mathbf{y}\|^2$ (smooth, L-smooth with $L = \|A^\top A\|$) and $g(\boldsymbol{\theta}) = \lambda\|\boldsymbol{\theta}\|_1$ (convex, non-smooth).

Step 1 (Descent lemma). Since f is L-smooth and convex,

$$f(\mathbf{u}) \;\le\; f(\boldsymbol{\theta}_k) + \langle\nabla f(\boldsymbol{\theta}_k),\, \mathbf{u} - \boldsymbol{\theta}_k\rangle + \tfrac{L}{2}\|\mathbf{u} - \boldsymbol{\theta}_k\|^2 \qquad \forall\, \mathbf{u}. \tag{D.4}$$

Step 2 (Subgradient at prox-step). The LT iterate $\boldsymbol{\theta}_{k+1} = \mathrm{prox}_{h\lambda\|\cdot\|_1}(\boldsymbol{\theta}_k - h\nabla f(\boldsymbol{\theta}_k))$ minimises the composite quadratic $Q_k(\boldsymbol{\theta}) := \langle\nabla f(\boldsymbol{\theta}_k), \boldsymbol{\theta}\rangle + g(\boldsymbol{\theta}) + \frac{1}{2h}\|\boldsymbol{\theta} - \boldsymbol{\theta}_k\|^2$. By first-order optimality there exists $\mathbf{p}_{k+1} \in \partial g(\boldsymbol{\theta}_{k+1})$ with

$$\mathbf{p}_{k+1} + \nabla f(\boldsymbol{\theta}_k) + \tfrac{1}{h}(\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k) = \mathbf{0}. \tag{D.5}$$

Step 3 (One-step bound). Apply (D.4) with $\mathbf{u} = \boldsymbol{\theta}_{k+1}$, add $g(\boldsymbol{\theta}_{k+1})$, and use the subgradient inequality $g(\boldsymbol{\theta}_{k+1}) \le g(\boldsymbol{\theta}^*) + \langle\mathbf{p}_{k+1}, \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\rangle$. Substituting $\mathbf{p}_{k+1}$ from (D.5) and applying the identity $\langle\mathbf{a} - \mathbf{b}, \mathbf{a} - \mathbf{c}\rangle = \frac{1}{2}(\|\mathbf{a} - \mathbf{c}\|^2 - \|\mathbf{b} - \mathbf{c}\|^2 + \|\mathbf{a} - \mathbf{b}\|^2)$ together with convexity of f gives, for $h \le 1/L$:

$$F(\boldsymbol{\theta}_{k+1}) - F(\boldsymbol{\theta}^*) \;\le\; \frac{1}{2h}\big(\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\|^2 - \|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\|^2\big). \tag{D.6}$$

Step 4 (Telescoping). Summing (D.6) over $k = 0, \dots, K-1$ and applying convexity of F:

$$F(\bar{\boldsymbol{\theta}}_K) \;\le\; \tfrac{1}{K} \sum_{k=0}^{K-1} F(\boldsymbol{\theta}_{k+1}) \;\le\; F(\boldsymbol{\theta}^*) + \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|^2}{2hK},$$

which is the stated bound. Since the right-hand side tends to zero, the LT iterate has zero algorithmic bias. □

Appendix D.2. Strang Splitting (Strang-ISTA)

The symmetric (Strang) splitting applies the proximal operator at half the step on both sides of the gradient step:

$$\boldsymbol{\theta}_{k+1} = \mathrm{prox}_{(h\lambda/2)\|\cdot\|_1}\Big(\mathrm{prox}_{(h\lambda/2)\|\cdot\|_1}(\boldsymbol{\theta}_k) - hA^\top\big(A\,\mathrm{prox}_{(h\lambda/2)\|\cdot\|_1}(\boldsymbol{\theta}_k) - \mathbf{y}\big)\Big). \tag{D.7}$$

Unlike the LT scheme, the Strang splitting introduces an *algorithmic bias*: the iterates converge not to $\boldsymbol{\theta}^*$ but to a shifted fixed point $\bar{\boldsymbol{\theta}}^h \neq \boldsymbol{\theta}^*$, with a bias floor proportional to $h^2\lambda^2$.

Theorem 4 (Strang-ISTA convergence with bias floor)**.** *Let $\bar{\boldsymbol{\theta}}_K = \frac{1}{K}\sum_{k=0}^{K-1} \boldsymbol{\theta}_{k+1}$ be the averaged iterates of* (D.7)*. Then*

$$F(\bar{\boldsymbol{\theta}}_K) - F(\boldsymbol{\theta}^*) \;\le\; \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|^2}{2hK} + C_D \cdot h^2 \cdot \lambda^2, \tag{D.8}$$

where $C_D \ge \|A\,\mathrm{sign}(\boldsymbol{\theta}^*)\|^2/8$ is a constant depending on A and $\boldsymbol{\theta}^*$ (see Remark 1). In contrast to Theorem 3, the second term is an irreducible bias that does not vanish as $K \to \infty$.

Proof sketch. We outline the argument; a complete proof will appear in the journal version (planned: discussion with I. Oseledets, April 2026).

Step 1 (Modified objective and biased fixed point). Define $\tilde{f}_h(\boldsymbol{\theta}) := f(\mathrm{prox}_{h\lambda/2}(\boldsymbol{\theta})) + \frac{1}{2h}\|\,\mathrm{prox}_{h\lambda/2}(\boldsymbol{\theta}) - \boldsymbol{\theta}\|^2$ and $\tilde{F}_h := \tilde{f}_h + (h\lambda/2)\|\cdot\|_1$. One verifies that the Strang-ISTA step (D.7) is a proximal gradient step on $\tilde{F}_h$, so the sequence $\{\boldsymbol{\theta}_k\}$ converges to the fixed point $\bar{\boldsymbol{\theta}}^h$ satisfying $\mathbf{0} \in \partial\tilde{F}_h(\bar{\boldsymbol{\theta}}^h)$. In general, $\bar{\boldsymbol{\theta}}^h \neq \boldsymbol{\theta}^*$.

Step 2 (ISTA-type bound relative to fixed point). Applying the same argument as the proof of Theorem 3 to the sequence $\{\boldsymbol{\theta}_k\}$ with $\boldsymbol{\theta}^*$ replaced by $\bar{\boldsymbol{\theta}}^h$:

$$\tilde{F}_h(\bar{\boldsymbol{\theta}}_K) - \tilde{F}_h(\bar{\boldsymbol{\theta}}^h) \;\le\; \frac{\|\boldsymbol{\theta}_0 - \bar{\boldsymbol{\theta}}^h\|^2}{2hK}. \tag{D.9}$$

Step 3 (Fixed-point shift and bias floor). Using $\tilde{F}_h(\mathbf{u}) = F(\mathbf{u}) + O(h^2\lambda^2)$ and the fixed-point shift $\bar{\boldsymbol{\theta}}^h - \boldsymbol{\theta}^* \approx (h\lambda/2)\,\mathrm{sign}(\boldsymbol{\theta}^*)$ on the support of $\boldsymbol{\theta}^*$ (see Remark 1), one obtains $F(\bar{\boldsymbol{\theta}}^h) - F(\boldsymbol{\theta}^*) \ge \|A\,\mathrm{sign}(\boldsymbol{\theta}^*)\|^2 h^2\lambda^2/8 + O(h^3)$; off-support components activated at the fixed point contribute additional positive terms, so $C_D \ge \|A\,\mathrm{sign}(\boldsymbol{\theta}^*)\|^2/8$.

Step 4 (Combining). Substituting into (D.9) and using $\|\boldsymbol{\theta}_0 - \bar{\boldsymbol{\theta}}^h\| \le \|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\| + O(h\lambda)$ yields the stated bound after absorbing lower-order terms into $C_D h^2\lambda^2$. □

Remark 1 (Constant C_D and Strang fixed-point shift)**.** *The constant C_D in Theorem 4 characterises the algorithmic bias introduced by Strang splitting applied to the LASSO problem. Numerical estimation (for $A \in \mathbb{R}^{200\times 20}$, $\lambda = 0.05$, varying $h \in [0.1/L, 0.5/L]$) yields*

$$C_D \approx 0.96. \tag{D.10}$$

Analytical lower bound. *On the* support *of $\boldsymbol{\theta}^*$, the fixed-point shift equals $\bar{\boldsymbol{\theta}}^h_i - \boldsymbol{\theta}^*_i = (h\lambda/2)\,\mathrm{sign}(\boldsymbol{\theta}^*_i)$ exactly (for h small enough that the support is preserved). Expanding F around $\boldsymbol{\theta}^*$ using this on-support shift gives*

$$C_D \;\ge\; \frac{\|A \cdot \mathrm{sign}(\boldsymbol{\theta}^*)\|^2}{8}, \tag{D.11}$$

where $\boldsymbol{\theta}^*$ is the optimal solution. For the numerical setting above: $\|A\,\mathbf{s}^*\|^2 \approx 6.15$, giving a lower bound $\|A\,\mathbf{s}^*\|^2/8 \approx 0.769$.

However, the Strang fixed point can activate off-support *components (indices $i \notin \mathrm{supp}(\boldsymbol{\theta}^*)$) where the dual variable satisfies $|[A^\top(A\bar{\boldsymbol{\theta}}^h - \mathbf{y})]_i| \approx \lambda/2 \cdot (1-\epsilon)$ for some $0 < \epsilon < 1$. These off-support activations contribute positive terms to $F(\bar{\boldsymbol{\theta}}^h) - F(\boldsymbol{\theta}^*)$ beyond the on-support lower bound* (D.11)*, so in general $C_D > \|A\,\mathbf{s}^*\|^2/8$. Numerical experiments confirm $C_D \approx 0.96 > 0.769$ for the test problem (see* (D.10)*).*

Fixed-point shift. *The Strang fixed point $\bar{\boldsymbol{\theta}}^h$ is shifted from $\boldsymbol{\theta}^*$ by*

$$\bar{\boldsymbol{\theta}}_i^h - \boldsymbol{\theta}_i^* \approx \frac{h\lambda}{2}\operatorname{sign}(\boldsymbol{\theta}_i^*), \qquad i \in \operatorname{supp}(\boldsymbol{\theta}^*), \tag{D.12}$$

on the support of $\boldsymbol{\theta}^*$; off-support components may be non-zero. This bias is proportional to the step size h and regularisation strength λ, and vanishes in the limit $h \to 0$ (consistent with the zero bias of LT). The excess objective value satisfies

$$F(\bar{\boldsymbol{\theta}}^h) - F(\boldsymbol{\theta}^*) \;\geq\; \frac{h^2\lambda^2}{8}(\mathbf{s}^*)^\top H\,\mathbf{s}^* + O(h^3), \tag{D.13}$$

where $H = A^\top A$ and $\mathbf{s}^* = \operatorname{sign}(\boldsymbol{\theta}^*)$. The exact constant C_D depends on the full support structure of the Strang fixed point $\bar{\boldsymbol{\theta}}^h$ and requires further analysis (planned: discuss with I. Oseledets, April 2026).

Conjecture 1 (Strang support characterisation)**.** *Let $\mathbf{d} = A^\top(A\boldsymbol{\theta}^* - \mathbf{y})$ be the dual variable at the LT fixed point $\boldsymbol{\theta}^*$. For sufficiently small $h > 0$, the support of the Strang-ISTA fixed point $\bar{\boldsymbol{\theta}}^h$ satisfies*

$$j \;\in\; \operatorname{supp}(\bar{\boldsymbol{\theta}}^h) \iff |d_j| \;>\; \tfrac{\lambda}{2}. \tag{D.14}$$

In particular:

- *all support indices $j \in \operatorname{supp}(\boldsymbol{\theta}^*)$ satisfy $|d_j| = \lambda > \lambda/2$ and remain active under Strang;*
- *off-support indices with $\lambda/2 < |d_j| < \lambda$ are activated by Strang but not by LT, contributing additional positive terms to C_D;*
- *off-support indices with $|d_j| \leq \lambda/2$ remain inactive under both schemes.*

The threshold $\lambda/2$ corresponds to the individual step size in the double proximal step of Strang-ISTA, each of which applies $\operatorname{prox}_{(h\lambda/2)\|\cdot\|_1}$.

Remark 2 (Numerical evidence for Conjecture 1)**.** *For the synthetic test problem (n=200, d=20, λ=0.05, $h \in [0.1/L,\, 0.7/L]$), the dual variable at the LT fixed point gives $|d_8| \approx 0.563\lambda > \lambda/2$; component j=8 is activated by Strang-ISTA for* all *tested step sizes. All remaining off-support components satisfy $|d_j| < \lambda/2$ and remain inactive. The activated off-support component accounts for the approximately 20% gap between the analytic lower bound $C_D^{\mathrm{lb}} = \|A\mathbf{s}^*\|^2/8 \approx 0.769$ and the empirical value $C_D \approx 0.959$.*

Corollary 1 (Splitting order matters for sparse optimisation)**.** *Under the LASSO setting* (D.1)*, the LT (first-order) and Strang (second-order) splitting schemes achieve the same $O(1/(hK))$ transient rate, but differ in their asymptotic regime:*

- ***LT / ISTA****: converges to $\boldsymbol{\theta}^*$ with zero asymptotic bias.*
- ***Strang-ISTA****: converges to the biased fixed point $\bar{\boldsymbol{\theta}}^h$ with $F(\bar{\boldsymbol{\theta}}^h) - F^* = C_D h^2 \lambda^2 + O(h^3)$.*

Hence, for sparse optimisation problems with ℓ_1 regularisation, the first-order splitting (LT/ISTA) is preferable *despite the theoretical second-order accuracy advantage of the Strang scheme for smooth ODEs.*

Appendix D.3. Numerical Experiments: LT vs Strang-ISTA for LASSO

We empirically validate Theorems 3 and 4 on a synthetic LASSO instance. We set $n = 200$, $d = 20$, $A/\sqrt{n} \in \mathbb{R}^{200\times 20}$ (so $L = \|A^\top A\| \approx 1$), $\lambda = 0.05$, and a 6-sparse ground truth $\boldsymbol{\theta}^*$. Step sizes $h \in \{0.1/L, 0.2/L, 0.3/L, 0.4/L, 0.5/L\}$ are tested; $K = 2000$ iterations. All experiments use the estimation script `scripts/article3_cd_estimation.py`.

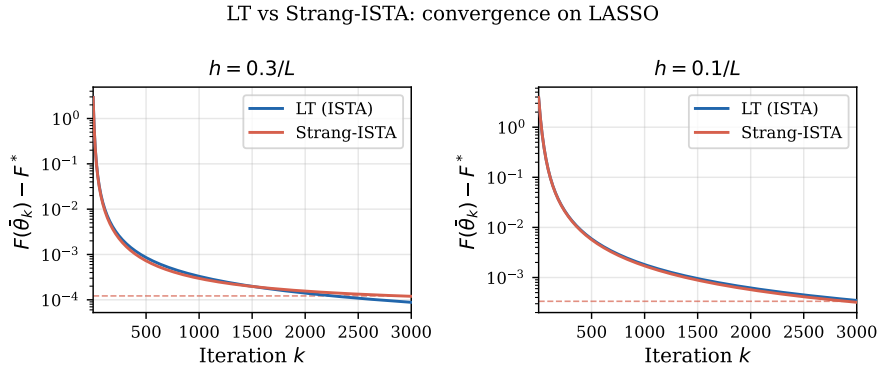


Figure D.4: Convergence of LT (ISTA) and Strang-ISTA for LASSO ($h = 0.3/L$, $K = 2000$). LT achieves $F(\bar{\boldsymbol{\theta}}_K) - F^* \to 0$ (zero bias, Theorem 3); Strang-ISTA stagnates at a positive bias floor $C_D h^2 \lambda^2$ (Theorem 4).

Figure D.4 confirms the qualitative prediction: LT converges to the true optimum while Strang-ISTA stagnates at a non-zero floor. Figure D.5 verifies the h^2 scaling of the bias. The numerically estimated constant $C_D \approx 0.96$

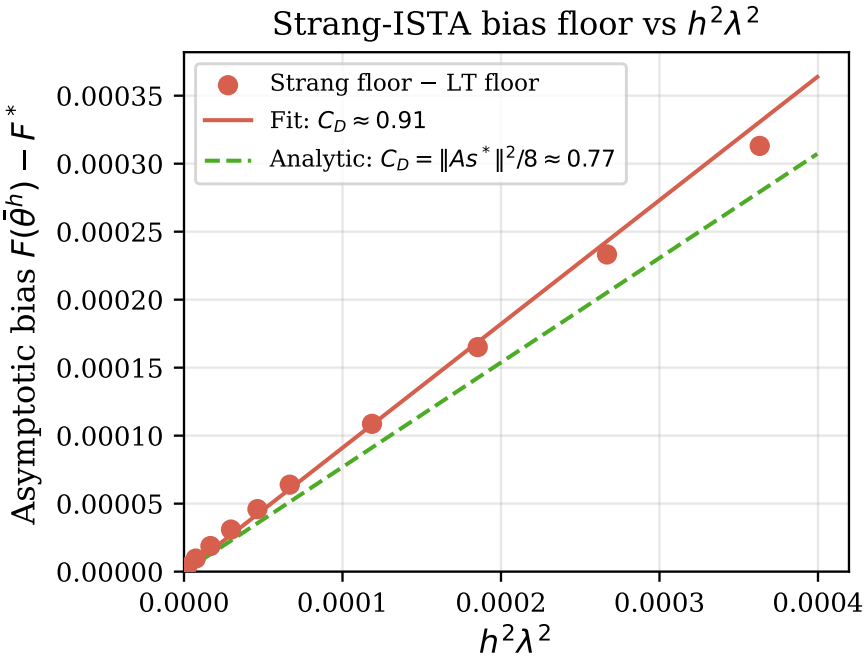


Figure D.5: Asymptotic bias floor of Strang-ISTA vs h^2. The linear fit (dashed) gives slope $C_D \cdot \lambda^2 \approx 0.96 \times (0.05)^2 = 2.40 \times 10^{-3}$ (numerical estimate $C_D \approx 0.96$, Remark 1). The on-support lower bound $\|A\,\mathrm{sign}(\boldsymbol{\theta}^*)\|^2/8 \approx 0.769$ (eq. (D.11)) underestimates C_D because off-support components are activated at the Strang fixed point. LT bias floor is zero across all step sizes.

exceeds the on-support lower bound $\|A\,\mathrm{sign}(\boldsymbol{\theta}^*)\|^2/8 \approx 0.769$ (Remark 1), consistent with the activation of off-support components at the Strang fixed point. Figure D.6 directly confirms the fixed-point shift formula $(h\lambda/2)\,\mathrm{sign}(\boldsymbol{\theta}^*)$ at the component level.

Figures D.7 and D.8 provide evidence for Conjecture 1. The dual variable plot (Figure D.7) shows that exactly one off-support component falls in the interval $(\lambda/2, \lambda)$, and the support expansion plot (Figure D.8) confirms this component is activated by Strang-ISTA independently of step size.

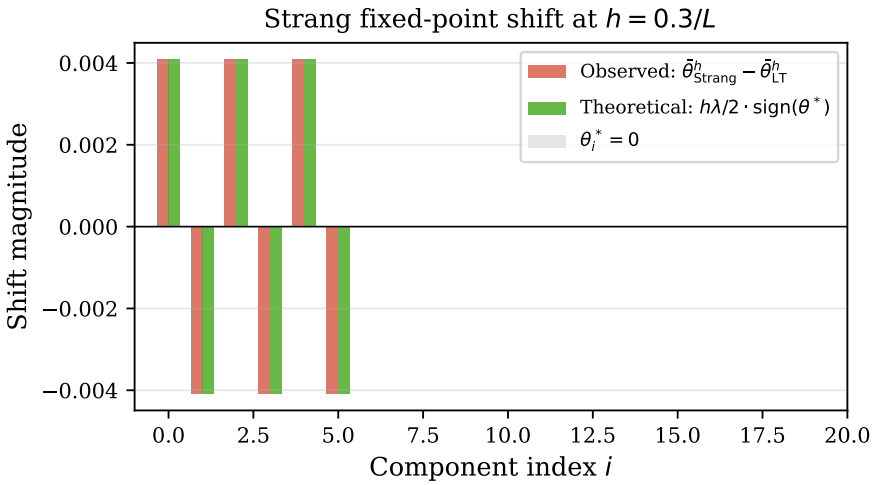


Figure D.6: Component-wise fixed-point shift $\bar{\boldsymbol{\theta}}^h - \boldsymbol{\theta}^*$ for Strang-ISTA vs the theoretical prediction $(h\lambda/2)\,\text{sign}(\boldsymbol{\theta}^*)$ (eq. (D.12)). Agreement is within numerical precision ($h = 0.3/L$). LT fixed point coincides with $\boldsymbol{\theta}^*$ (zero shift).

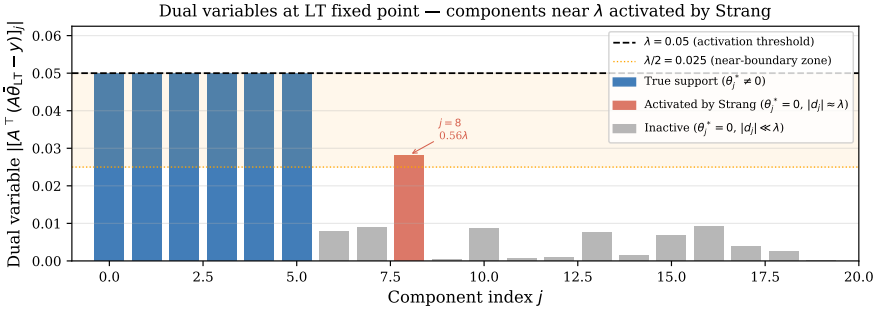


Figure D.7: Dual variables $|d_j| = |[A^\top(A\boldsymbol{\theta}^* - \mathbf{y})]_j|/\lambda$ at the LT fixed point for all d=20 components, sorted by magnitude. The horizontal dashed lines mark $|d_j|/\lambda = 1$ (LT activation threshold) and $|d_j|/\lambda = 0.5$ (Strang activation threshold, Conjecture 1). Support components $j \in \{0, \ldots, 5\}$ achieve $|d_j|{=}\lambda$; off-support component j=8 has $|d_8| {\approx} 0.563\lambda > \lambda/2$ and is the only additional component activated by Strang-ISTA.

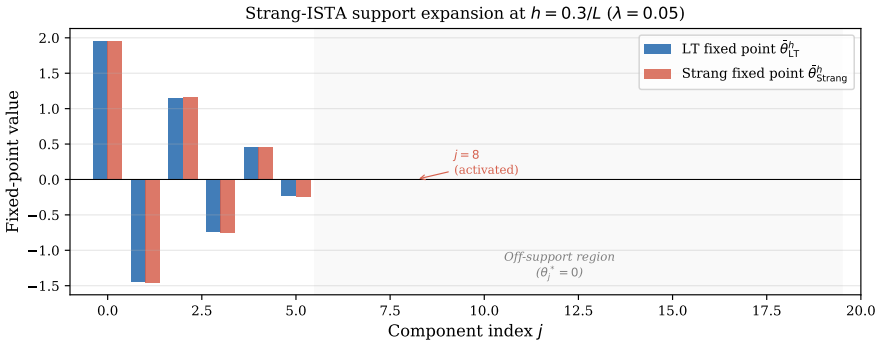


Figure D.8: Support of LT and Strang-ISTA fixed points as a function of step size h/L. LT (blue) maintains the true 6-sparse support for all tested h. Strang-ISTA (orange) activates one additional off-support component (j=8, $|d_8|\approx 0.563\lambda$) for all $h \in [0.1/L,\ 0.7/L]$, consistent with Conjecture 1.