Stochastic gradient algorithms from ODE splitting perspective

ICLR 2020 DeepDiffEq workshop

Daniil Merkulov, Ivan Oseledets

daniil.merkulov@skolkovotech.ru, i.oseledets@skoltech.ru

Skolkovo Institute of Science and Technology

Finite sum problems

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

Finite sum problems

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

Discrete time

Vanilla: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

Finite sum problems

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

Discrete time

Vanilla: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

SGD: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{b}\sum_{j=1}^{b} \nabla f_{i_j}(\boldsymbol{\theta})$

Finite sum problems

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

Discrete time

Vanilla: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

SGD: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{b}\sum_{j=1}^{b} \nabla f_{i_j}(\boldsymbol{\theta})$

Continuous time

Vanilla: $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

Finite sum problems

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

Discrete time

Vanilla: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

SGD: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{b}\sum_{j=1}^{b} \nabla f_{i_j}(\boldsymbol{\theta})$

Continuous time

Vanilla: $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$

SGD(?): $\frac{d\boldsymbol{\theta}}{dt} = -\left(\nabla f(\boldsymbol{\theta}) + \xi\right)$

Finite sum problems

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

Discrete time	Continuous time
Vanilla: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$	Vanilla: $\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta})$
SGD: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - h_k \frac{1}{b}\sum_{j=1}^{b} \nabla f_{i_j}(\boldsymbol{\theta})$	SGD(?): $\frac{d\boldsymbol{\theta}}{dt} = -\left(\nabla f(\boldsymbol{\theta}) + \xi\right)$

We propose a new view on the continuous time SGD as a first-order splitting scheme.

Splitting scheme for ODE

Simplest example of initial value problem. We have $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ and $\boldsymbol{\theta}(h) = ?$:

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

Splitting scheme for ODE

Simplest example of initial value problem. We have $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ and $\boldsymbol{\theta}(h) = ?$:

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

We split right-hand side into primitive summands and solve each problem separately with reinitialization.

$$\frac{d\boldsymbol{\theta_1}}{dt} = -\frac{1}{2} g_1(\boldsymbol{\theta_1}), \boldsymbol{\theta_1}(0) = \boldsymbol{\theta}_0 \;\rightarrow\; \boldsymbol{\theta}_1(h);$$

Splitting scheme for ODE

Simplest example of initial value problem. We have $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ and $\boldsymbol{\theta}(h) = ?$:

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

We split right-hand side into primitive summands and solve each problem separately with reinitialization.

$$\frac{d\boldsymbol{\theta_1}}{dt} = -\frac{1}{2}g_1(\boldsymbol{\theta_1}), \boldsymbol{\theta_1}(0) = \boldsymbol{\theta}_0 \;\rightarrow\; \boldsymbol{\theta}_1(h);$$

$$\frac{d\boldsymbol{\theta_2}}{dt} = -\frac{1}{2}g_2(\boldsymbol{\theta_2}), \boldsymbol{\theta_2}(0) = \boldsymbol{\theta_1}(h) \;\rightarrow\; \boldsymbol{\theta}_2(h)$$

Splitting scheme for ODE

Simplest example of initial value problem. We have $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ and $\boldsymbol{\theta}(h) =?$:

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

We split right-hand side into primitive summands and solve each problem separately with reinitialization.

$$\frac{d\boldsymbol{\theta_1}}{dt} = -\frac{1}{2}g_1(\boldsymbol{\theta_1}), \boldsymbol{\theta_1}(0) = \boldsymbol{\theta}_0 \ \rightarrow \ \boldsymbol{\theta}_1(h);$$

$$\frac{d\boldsymbol{\theta_2}}{dt} = -\frac{1}{2}g_2(\boldsymbol{\theta_2}), \boldsymbol{\theta_2}(0) = \boldsymbol{\theta_1}(h) \ \rightarrow \ \boldsymbol{\theta}_2(h)$$

First order splitting scheme: $\boldsymbol{\theta}^I(h) = \boldsymbol{\theta}_n(h) \circ \cdots \circ \boldsymbol{\theta}_1(h) \circ \boldsymbol{\theta}_0$

SGD as a splitting scheme

What if $g_1 = \nabla f_1(\boldsymbol{\theta}), g_2 = \nabla f_2(\boldsymbol{\theta})$?

SGD as a splitting scheme

What if $g_1 = \nabla f_1(\boldsymbol{\theta}), g_2 = \nabla f_2(\boldsymbol{\theta})$?

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

SGD as a splitting scheme

What if $g_1 = \nabla f_1(\boldsymbol{\theta}), g_2 = \nabla f_2(\boldsymbol{\theta})$?

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

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)

Idea

1. Consider some finite-sum problem

Idea

1. Consider some finite-sum problem
2. Formulate corresponding ODE

Idea

1. Consider some finite-sum problem
2. Formulate corresponding ODE
3. Apply first order splitting and solve local problems more precisely, than Euler (as it is done in SGD).

Idea

1. Consider some finite-sum problem
2. Formulate corresponding ODE
3. Apply first order splitting and solve local problems more precisely, than Euler (as it is done in SGD).
4. Compare with SGD approximation in the same time.

Idea

1. Consider some finite-sum problem
2. Formulate corresponding ODE
3. Apply first order splitting and solve local problems more precisely, than Euler (as it is done in SGD).
4. Compare with SGD approximation in the same time.

Problem	Loss function	Batch gradient	Initial local ODE
Linear Least Squares	$f(\boldsymbol{\theta}) = \frac{1}{n}\sum\limits_{i=1}^{m} \lVert X_i\boldsymbol{\theta} - \mathbf{y_i}\rVert_2^2$	$\frac{1}{b}X_i^\top(X_i\boldsymbol{\theta} - \mathbf{y_i})$	$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top(X_i\boldsymbol{\theta} - \mathbf{y_i})$
Binary logistic regression	$f(\boldsymbol{\theta}) = -\dfrac{1}{n}\sum\limits_{i=1}^{n}\Big(y_i \ln \sigma(\boldsymbol{\theta}^\top \mathbf{x_i}) + (1-y_i)\ln\big(1-\sigma(\boldsymbol{\theta}^\top\mathbf{x_i})\big)\Big)$	$\frac{1}{b}X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right)$	$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n}X_i^\top\left(\sigma\left(X_i\boldsymbol{\theta}\right) - \mathbf{y_i}\right)$
One FC Layer + softmax	$f(\Theta) = -\frac{1}{n}\sum\limits_{i=1}^{n}\log\left(\frac{\mathbf{y_i}^\top e^{\Theta^\top \mathbf{x_i}}}{\mathbf{1}^\top e^{\Theta^\top \mathbf{x_i}}}\right)$	$\frac{1}{b}X_i^\top\left(s(\Theta^\top X_i^\top) - Y_i\right)^\top$	$\frac{d\Theta}{dt} = -\frac{1}{n}X_i^\top\left(s(\Theta^\top X_i^\top) - Y_i\right)^\top$

Integration of local problems

It is important, that we can reduce dimensionality of the dynamic system via QR decomposition of each batch data matrix $X_i^\top = Q_i R_i$ and substitution $\boldsymbol{\eta}_i = Q_i^\top \boldsymbol{\theta}$.

Integration of local problems

It is important, that we can reduce dimensionality of the dynamic system via QR decomposition of each batch data matrix $X_i^\top = Q_i R_i$ and substitution $\boldsymbol{\eta}_i = Q_i^\top \boldsymbol{\theta}$.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i}), \boldsymbol{\theta} \in \mathbb{R}^p$$

Integration of local problems

It is important, that we can reduce dimensionality of the dynamic system via QR decomposition of each batch data matrix $X_i^\top = Q_i R_i$ and substitution $\boldsymbol{\eta}_i = Q_i^\top \boldsymbol{\theta}$.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i}), \boldsymbol{\theta} \in \mathbb{R}^p$$

$$\begin{cases} \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = -\frac{1}{n} R_i \left(R_i^\top \boldsymbol{\eta}_\mathbf{i} - \mathbf{y_i} \right), \boldsymbol{\eta}_\mathbf{i} = Q_i^\top \boldsymbol{\theta}, \boldsymbol{\eta}_\mathbf{i} \in \mathbb{R}^b \\ \boldsymbol{\theta}(h) = Q_i \left(\boldsymbol{\eta}_\mathbf{i}(h) - \boldsymbol{\eta}_\mathbf{i}(0) \right) + \boldsymbol{\theta}_0 \end{cases}$$

Integration of local problems

It is important, that we can reduce dimensionality of the dynamic system via QR decomposition of each batch data matrix $X_i^\top = Q_i R_i$ and substitution $\boldsymbol{\eta}_i = Q_i^\top \boldsymbol{\theta}$.

$$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i}), \boldsymbol{\theta} \in \mathbb{R}^p$$

$$\begin{cases} \frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = -\frac{1}{n} R_i \left(R_i^\top \boldsymbol{\eta}_\mathbf{i} - \mathbf{y_i} \right), \boldsymbol{\eta}_\mathbf{i} = Q_i^\top \boldsymbol{\theta}, \boldsymbol{\eta}_\mathbf{i} \in \mathbb{R}^b \\ \boldsymbol{\theta}(h) = Q_i \left(\boldsymbol{\eta}_\mathbf{i}(h) - \boldsymbol{\eta}_\mathbf{i}(0) \right) + \boldsymbol{\theta}_0 \end{cases}$$

Initial local ODE	$\mathcal{P}_i^k$	**Integration**
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top (X_i \boldsymbol{\theta} - \mathbf{y_i})$	$\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = -\frac{1}{n} R_i \left(R_i^\top \boldsymbol{\eta}_\mathbf{i} - \mathbf{y_i} \right), \boldsymbol{\eta}_\mathbf{i} = Q_i^\top \boldsymbol{\theta}$	analytical
$\frac{d\boldsymbol{\theta}}{dt} = -\frac{1}{n} X_i^\top \left(\sigma \left(X_i \boldsymbol{\theta} \right) - \mathbf{y_i} \right)$	$\frac{d\boldsymbol{\eta}_\mathbf{i}}{dt} = -\frac{1}{n} R_i \left(\sigma \left(R_i^\top \boldsymbol{\eta}_\mathbf{i} \right) - \mathbf{y_i} \right), \boldsymbol{\eta}_\mathbf{i} = Q_i^\top \boldsymbol{\theta}$	`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`

Algorithm

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

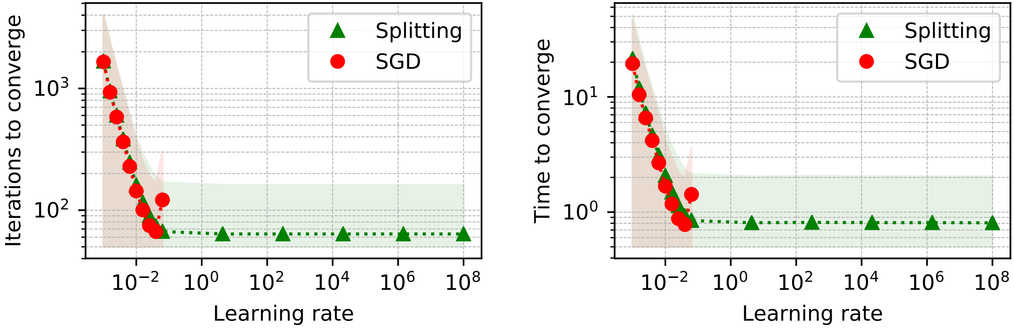
    $t := t + 1$

  end

end

Random linear system

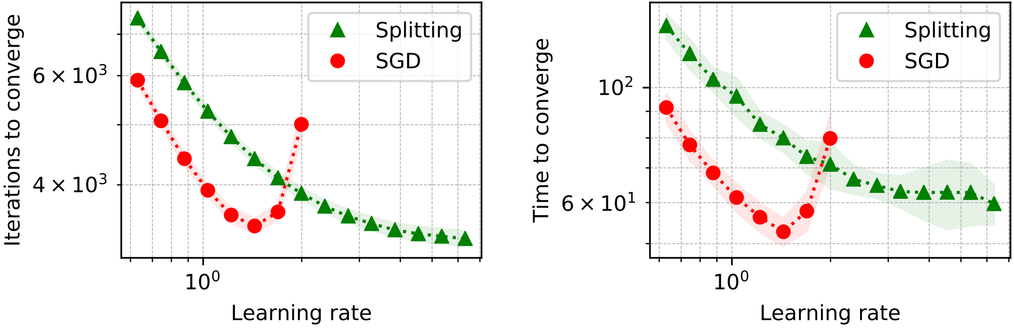
10000×500. $b = 20$. Relative error 10^{-3}



(a) Iterations (b) Time

Real linear system

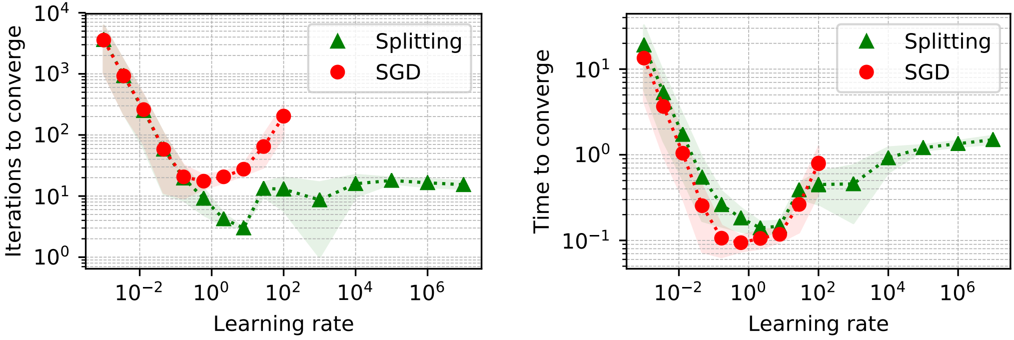
Tomogropy data from AIRTools II 12780×2500. $b = 60$. Relative error 10^{-3}



(a) `Iterations` (b) `Time`

Binary logistic regression

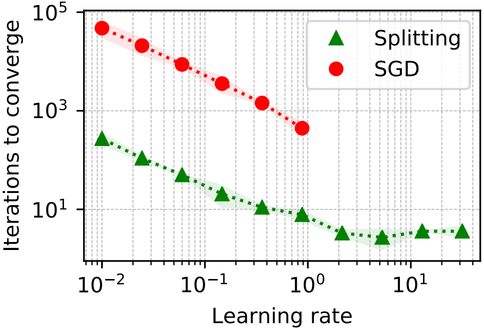
MNIST 0,1 dataset. $b = 50$. Test error 10^{-3}



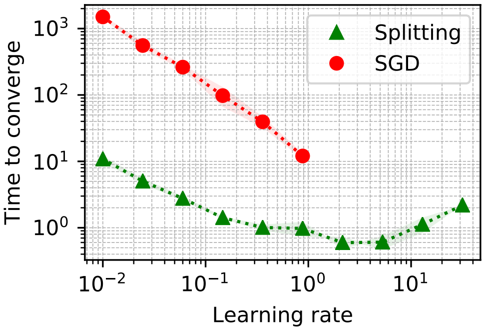
(a) Iterations (b) Time

Softmax regression

Fashion MNIST dataset. 10 classes. 28×28 images, $b = 64$. Test error 0.25

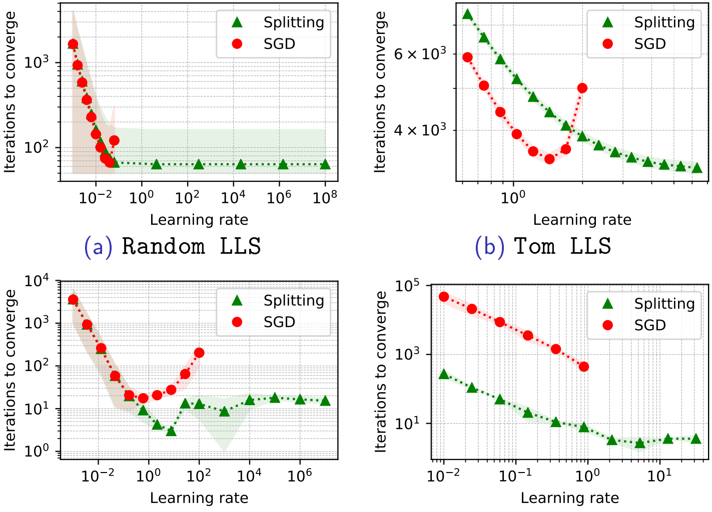


(a) Iterations



(b) Time

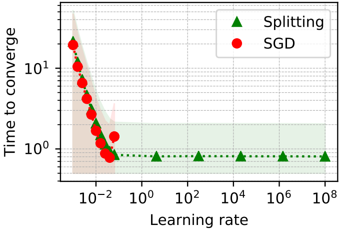
SGD vs Splitting. Iteration comparison



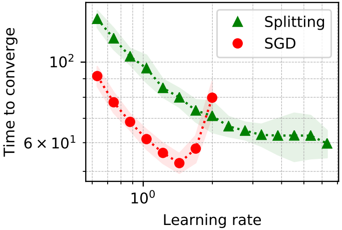
(a) Random LLS (b) Tom LLS

(c) LogReg (d) Softmax

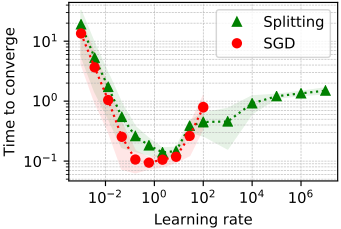
SGD vs Splitting. Time comparison



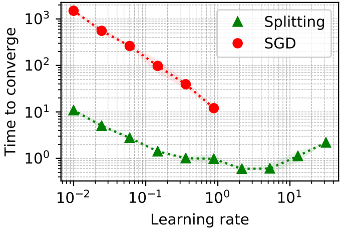
(a) Random LLS



(b) Tom LLS



(c) LogReg



(d) Softmax

Thank you for your attention!

Contact:

daniil.merkulov@skolkovotech.ru

Paper and code:

merkulov.top/sgd_splitting