A Subsampling Line-Search Method with Second-Order Results: Online Companion

E. Bergou * Y. Diouane† V. Kunc ‡ V. Kungurtsev § C. W. Royer ¶

December 31, 2021

Abstract

This document is the online companion to the manuscript *A Subsampling Line-Search Method with Second-Order Results*, accepted to INFORMS Journal on Optimization. It provides detailed proofs of technical results, as well as more information on the experimental setup and additional experiments using an implementation of the proposed method.

1 Supplementary proofs

1.1 Proof of [1, Lemma 3]:

We consider in turn the three possible steps that can be taken at iteration k, and obtain a lower bound on the amount $\alpha_k\|d_k\|$ for each of those.

Case 1: $\lambda_k < -\epsilon^{1/2}$ (negative curvature step). In that case, we apply the same reasoning than in [4, Proof of Lemma 1] with the model $\hat{f}_k$ playing the role of the objective, the backtracking line search terminates with the step length $\alpha_k = \theta^{j_k}$, with $j_k \leq \bar{j}_{nc} + 1$ and $\alpha_k \geq \frac{3\theta}{L_H+\eta}$. When d_k is computed as a negative curvature direction, one has $\|d_k\| = -\lambda_k > 0$. Hence,

$$\alpha_k\|d_k\| \;\geq\; \frac{3\theta}{L_H+\eta}[-\lambda_k] \;=\; c_{nc}[-\lambda_k] \geq c\epsilon^{1/2}.$$

Case 2: $\lambda_k > \|g_k\|^{1/2}$ (Newton step). Because the stationarity is not achieved and $\lambda_k > 0$ in this case, we necessarily have $\|\tilde{g}_k\| = \min\{\|g_k\|, \|g_k^+\|\} > \epsilon$. From [1, Algorithm 1], we know that d_k is chosen as the Newton step. Hence, using the argument of [4, Proof of Lemma 3] with $\epsilon_H = \epsilon^{1/2}$, the backtracking line search terminates with the step length $\alpha_k = \theta^{j_k}$, where

$$j_k \leq \left[\log_\theta\left(\sqrt{\frac{3}{L+\eta}}\frac{\epsilon^{1/2}}{\sqrt{U_g}}\right)\right]_+ + 1 = \bar{j}_n + 1,$$

*Mohammed VI Polytechnic University, Ben Guerir, Morocco (elhoucine.bergou@um6p.ma).

†ISAE-SUPAERO, Université de Toulouse, 31055 Toulouse Cedex 4, France (youssef.diouane@isae.fr).

‡Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague (kuncvlad@fel.cvut.cz).

§Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague (vyacheslav.kungurtsev@fel.cvut.cz).

¶LAMSADE, CNRS, Université Paris Dauphine-PSL, 75016 PARIS, FRANCE (clement.royer@dauphine.psl.eu).

thus the first part of the result holds. If the unit step size is chosen, we have by [4, Relation (23)] that

$$\alpha_k\|d_k\| = \|d_k\| \geq \left[\frac{2}{L_H}\right]^{1/2} \|g(x_k + \alpha_k d_k; \mathcal{S}_k)\|^{1/2} \geq c\epsilon^{1/2}. \tag{1}$$

Consider now the case $\alpha_k < 1$. Using [4, Relations (25) and (26), p. 1457], we have:

$$\|d_k\| \geq \frac{3}{L_H + \eta}\epsilon_H = \frac{3}{L_H + \eta}\epsilon^{1/2} \text{ and}$$

$$\alpha_k \geq \theta\sqrt{\frac{3}{L_H + \eta}}\epsilon_H^{1/2}\|d_k\|^{-1/2} = \theta\sqrt{\frac{3}{L_H + \eta}}\epsilon^{1/4}\|d_k\|^{-1/2}.$$

As a result,

$$\alpha_k\|d_k\| \quad \geq \quad \theta\left[\frac{3}{L_H + \eta}\right]^{1/2} \epsilon^{1/4}\|d_k\|^{-1/2}\|d_k\| \geq \left[\frac{3\theta}{L_H + \eta}\right]\epsilon^{1/2} \geq c\epsilon^{1/2}. \tag{2}$$

Case 3: (Regularized Newton step) This case occurs when the conditions for the other two cases fail, that is, when $-\epsilon^{1/2} \leq \lambda_k \leq \|g_k\|^{1/2}$. We again exploit the fact that the stationarity is not achieved to deduce that we necessarily have $\|\tilde{g}_k\| = \min\{\|g_k\|, \|g_k^+\|\} > \epsilon$. This in turn implies that $\min\{\|\tilde{g}_k\|\epsilon^{-1/2}, \epsilon^{1/2}\} \geq \epsilon^{1/2}$. As in the proof of the previous lemma, we apply the theory of [4, Proof of Lemma 4] using $\epsilon_H = \epsilon^{1/2}$. We then know that the backtracking line search terminates with the step length $\alpha_k = \theta^{j_k}$, with

$$j_k \leq \left[\log_\theta\left(\frac{6}{L_H + \eta}\frac{\epsilon}{U_g}\right)\right]_+ + 1 = \bar{j}_{rn} + 1.$$

We now distinguish between the cases $\alpha_k = 1$ and $\alpha_k < 1$. If the unit step size is chosen, we can use [4, relations 30 and 31], where $\nabla f(x_k + d_k)$ and ϵ_H are replaced by g_k^+ and $\epsilon^{1/2}$, respectively. This gives

$$\alpha_k\|d_k\| = \|d_k\| \quad \geq \quad \frac{1}{1 + \sqrt{1 + L_H/2}} \min\left\{\|g_k^+\|/\epsilon^{1/2}, \epsilon^{1/2}\right\}.$$

Therefore, if the unit step is accepted, one has by [4, equation 31]

$$\alpha_k\|d_k\| \quad \geq \quad \frac{1}{1 + \sqrt{1 + L_H/2}} \min\left\{\|\tilde{g}_k\|\epsilon^{-1/2}, \epsilon^{1/2}\right\}. \tag{3}$$

Considering the case $\alpha_k < 1$ and using [4, equation 32, p. 1459], we have:

$$\alpha_k \geq \theta\frac{6}{L_H + \eta}\epsilon_H\|d_k\|^{-1} = \frac{6\theta}{L_H + \eta}\epsilon^{1/2}\|d_k\|^{-1},$$

which leads to

$$\alpha_k\|d_k\| \quad \geq \quad \frac{6\theta}{L_H + \eta}\epsilon^{1/2}. \tag{4}$$

Putting (3) and (4) together, we obtain

$$\alpha_k\|d_k\| \quad \geq \quad \min\left\{\frac{1}{(1 + \sqrt{1 + L_H/2})^3}, \left[\frac{6\theta}{L_H + \eta}\right]^3\right\} \min\{\|\tilde{g}_k\|\epsilon^{-1/2}, \epsilon^{1/2}\} = c_{rn}\min\{\|\tilde{g}_k\|\epsilon^{-1/2}, \epsilon^{1/2}\} \geq c\epsilon^{1/2}.$$

By putting the three cases together, we arrive at the desired conclusion.□

1.2 Proof of [1, Lemma 4]:

Indeed, since the lemma trivially holds if $\|d_k\| = 0$, we only need to prove that it holds for $\|d_k\| > 0$. We consider three disjoint cases:

Case 1: $\lambda_k < -\epsilon^{1/2}$. Then the negative curvature step is taken and $\|d_k\| = |\lambda_k| \leq U_H$.

Case 2: $\lambda_k > \|g_k\|^{1/2}$. We can suppose that $\|g_k\| > 0$ because otherwise $\|d_k\| = 0$. Then, d_k is a Newton step with

$$\|d_k\| \leq \|H_k^{-1}\|\|g_k\| \leq \|g_k\|^{-1/2}\|g_k\| \leq \|g_k\|^{1/2} \leq U_g^{1/2}.$$

Case 3: $-\epsilon^{1/2} \leq \lambda_k \leq \|g_k\|^{1/2}$. As in Case 2, we suppose that $\|g_k\| > 0$ as $\|d_k\| = 0$ if this does not hold. Then, d_k is a regularized Newton step with

$$\|d_k\| = \|(H_k + (\|g_k\|^{1/2} + \epsilon^{1/2})\mathbb{I}_n)^{-1}g_k\| \leq \frac{\|g_k\|}{\lambda_k + \|g_k\|^{1/2} + \epsilon^{1/2}} \leq \|g_k\|^{1/2} \leq U_g^{1/2}.$$

where the last inequality uses $\lambda_k + \epsilon^{1/2} \geq 0$ and $\|g_k\| > 0$. □

1.3 Proof of [1, Theorem 5]:

To prove [1, Theorem 5.], we will combine the following three standard lemmas.

Lemma 5 *Under Assumption 2, consider an iterate x_k of Algorithm 1. For any $p \in (0,1)$, if the sample set $\mathcal{S}_k$ is chosen to be of size*

$$\bar{\pi} \geq \frac{1}{N}\frac{16L^2}{\delta_H^2}\ln\left(\tfrac{2N}{1-p}\right), \tag{5}$$

then

$$\mathbb{P}\left(\left\|H(x_k;\mathcal{S}_k) - \nabla^2 f(x_k)\right\| \leq \delta_H \middle| \mathcal{F}_{k-1}\right) \geq p.$$

Proof. Proof of Lemma 5. See [6, Lemma 16]; note that here we are using L_i (Lipschitz constant of ∇f_i) as a bound on $\|\nabla^2 f_i(x_k)\|$, and that we are providing a bound on the sampling fraction $\bar{\pi} = \frac{|\mathcal{S}_k|}{N}$. See also [5, Theorem 1.1], considering the norm as related to the maximum singular vector. □ ■

By the same reasoning as for Lemma 5, but in one dimension, we can readily provide a sample size bound for obtaining accurate function values. To this end, we define

$$f_{\text{up}} \geq \max_k \max_{i=1,..,N} f_i(x_k). \tag{6}$$

Note that such a bound necessarily exists when the iterates are contained in a compact set. Specific structure of the problem can also guarantee such a bound, even tough it may exhibit dependencies on the problem's dimension. For instance, in the case of classification and logistic regression, one has $f_{\text{up}} = 1$, while in the case of (general) regression, one has $f_{\text{up}} \leq C_1 + C_2\|x\|^2 = \mathcal{O}(n)$. We emphasize that both of these bounds can be very pessimistic.

Lemma 6 *Under Assumption 1, consider an iterate x_k of Algorithm 1. For any $p \in (0,1)$, if the sample set $\mathcal{S}_k$ is chosen to be of size*

$$\bar{\pi} \geq \frac{1}{N}\frac{16f_{\text{up}}^2}{\delta_f^2}\ln\left(\tfrac{2}{1-p}\right), \tag{7}$$

then

$$\mathbb{P}\left(\left|\hat{f}(x_k;\mathcal{S}_k) - f(x_k)\right| \leq \delta_f \middle| \mathcal{F}_{k-1}\right) \geq p.$$

Proof. Proof of Lemma 6. The proof follows that of [6, Lemma 4.1] by considering $\hat{f}(x_k;\mathcal{S}_k)$ and $f(x_k)$ as one-dimensional matrices. □ ■

Lemma 7 *Under Assumption 2, consider an iterate x_k of Algorithm 1. For any $p \in (0,1)$, if the sample set $\mathcal{S}_k$ is chosen to be of size*

$$\bar{\pi} \geq \frac{1}{N}\frac{U_g^2}{\delta_g^2}\left[1 + \sqrt{8\ln\left(\tfrac{1}{1-p}\right)}\right]^2, \tag{8}$$

then

$$\mathbb{P}\left(\|g(x_k;\mathcal{S}_k) - \nabla f(x_k)\| \leq \delta_g | \mathcal{F}_{k-1}\right) \geq p.$$

Proof. Proof of Lemma 7. See [3, Lemma 2]. □ ■

We now combine the three previous lemmas to obtain an overall result indicating the required $\bar{\pi}$ such that Lemmas 5, 6 and 7 simultaneously hold, i.e., the event I_k holds.

Indeed, let $\delta_f = \frac{\eta}{24}c^3\epsilon^{3/2}$, $\delta_g = \kappa_g\epsilon$, $\delta_H = \kappa_H\epsilon^{1/2}$ and note that,

$$I_k \equiv I_k^h \cap I_k^g \cap I_k^f,$$

where

$$\begin{aligned} I_k^h &:= \{\left\|H(x_k;\mathcal{S}_k) - \nabla^2 f(x_k)\right\| \leq \delta_H\} \\ I_k^g &:= \{\|g(x_k;\mathcal{S}_k) - \nabla f(x_k)\| \leq \delta_g\} \\ I_k^f &:= \{\left|\hat{f}(x_k;\mathcal{S}_k) - f(x_k)\right| \leq \delta_f\}. \end{aligned}$$

Using the required conditions on π_n, Lemmas 5, 6 and 7 imply

$$\mathbb{P}\left((I_k^f)^c \middle| \mathcal{F}_{k-1}\right) \leq 1-\hat{p}\ , \quad \mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) \leq 1-\hat{p}, \quad \text{and} \quad \mathbb{P}\left((I_k^h)^c \middle| \mathcal{F}_{k-1}\right) \leq 1-\hat{p}.$$

In the other hand, one has

$$\begin{aligned} \mathbb{P}\left((I_k)^c | \mathcal{F}_{k-1}\right) &= \mathbb{P}\left((I_k^f)^c \middle| \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) - \mathbb{P}\left((I_k^f)^c \cap (I_k^g)^c \middle| \mathcal{F}_{k-1}\right) \\ &\quad -\mathbb{P}\left((I_k^g)^c \cap (I_k^h)^c | \mathcal{F}_{k-1}\right) - \mathbb{P}\left((I_k^f)^c \cap (I_k^h)^c \middle| \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^f)^c \cap (I_k^g)^c \cap (I_k^h)^c \middle| \mathcal{F}_{k-1}\right) \\ &\leq \mathbb{P}\left((I_k^f)^c \middle| \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^h)^c | \mathcal{F}_{k-1}\right) \\ &\quad +\mathbb{P}\left((I_k^f)^c \middle| (I_k^h)^c, (I_k^g)^c, \mathcal{F}_{k-1}\right) \mathbb{P}\left((I_k^h)^c | (I_k^g)^c, \mathcal{F}_{k-1}\right) \mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) \\ &\leq \mathbb{P}\left((I_k^f)^c \middle| \mathcal{F}_{k-1}\right) + 2\mathbb{P}\left((I_k^g)^c | \mathcal{F}_{k-1}\right) + \mathbb{P}\left((I_k^h)^c | \mathcal{F}_{k-1}\right) \leq 4(1-\hat{p}). \end{aligned}$$

Hence,

$$\mathbb{P}(I_k | \mathcal{F}_{k-1}) \geq 1 - 4(1-\hat{p}) = p,$$

meaning that the model sequence is p-probabilistically $(\delta_f, \delta_g, \delta_H)$-accurate, thus results from Section 4.1 (of the main paper) hold. □

1.4 Proof of [1, Proposition 1]

In this proof, we will use the notation $\mathbb{P}_k(\dots) = \mathbb{P}(\cdot|\mathcal{F}_{k-1})$, as well as the random events

$$E = \left\{\text{One of the iterates in } \{x_{k+j}\}_{j=0..J} \text{ is } ((1+\kappa_g)\epsilon, (1+\kappa_H)\epsilon^{1/2})\text{-function stationary}\right\},$$

$$E_j = \left\{\text{The iterate } x_{k+j} \text{ is } (\epsilon, \epsilon^{1/2})\text{-model stationary}\right\} \quad \forall j = 0, \dots, J.$$

For every $j = 0, \dots, J$, we have $E_j \in \mathcal{F}_{k+j}$ and $I_{k+j} \in \mathcal{F}_{k+j}$ (where I_j is the event introduced in [1, Definition 2]). Moreover, the events E_j and I_{k+j} are conditionally independent:

$$\forall j = 0, \dots, J, \quad \mathbb{P}_k\left(E_j \cap I_{k+j} | \mathcal{F}_{k+j-1}\right) = \mathbb{P}_k\left(E_j | \mathcal{F}_{k+j-1}\right) \mathbb{P}_k\left(I_{k+j} | \mathcal{F}_{k+j-1}\right). \tag{9}$$

This conditional independence holds because $x_{k+j} \in \mathcal{F}_{k+j-1}$, and the model $\hat{f}_{k+J}$ is constructed independently of x_{k+j} by assumption. Using these events, we can reformulate the statement of the theorem as

$$\mathbb{P}_k\left(E | E_0, \dots, E_J\right) \geq 1 - (1-p)^{J+1}.$$

Now, by [1, Lemma 1],

$$\mathbb{P}_k\left(E | E_0, \dots, E_J\right) \geq \mathbb{P}_k\left(\bigcup_{0 \leq j \leq J} I_{k+j} \middle| E_0, \dots, E_J\right) = 1 - \mathbb{P}_k\left(\bigcap_{0 \leq j \leq J} \bar{I}_{k+j} \middle| E_0, \dots, E_J\right).$$

Thus,to obtain the desired result, it suffices to prove that

$$\mathbb{P}_k\left(\bigcap_{0 \leq j \leq J} \bar{I}_{k+j} \middle| E_0, \dots, E_J\right) \leq (1-p)^{J+1}. \tag{10}$$

We now make use of the probabilistically accuracy property. For every $j = 0, \dots, J$, we have

$$\mathbb{P}_k\left(I_{k+j} | A\right) \geq p, \tag{11}$$

for any set of events A belonging to the σ-algebra $\mathcal{F}_{k+j-1}$ [2, Chapter 5]. In particular, for any $j \geq 1$, $\mathbb{P}_k\left(I_{k+j} | I_k, \dots, I_{k+j-1}, E_0, \dots, E_j\right) \geq p$. Returning to our target probability, we have:

$$\begin{aligned}
\mathbb{P}_k\left(\bigcap_{0 \leq j \leq J} \bar{I}_{k+j} \middle| E_0, \dots, E_J\right) &= \frac{\mathbb{P}_k\left(\left\{\bigcap_{0 \leq j \leq J} \bar{I}_{k+j}\right\} \cap E_J \middle| E_0, \dots, E_{J-1}\right)}{\mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}\right)} \\
&= \frac{\mathbb{P}_k\left(\bar{I}_J \cap E_J \middle| E_0, \dots, E_{J-1}, I_k, \dots, I_{k+J-1}\right) \mathbb{P}_k\left(\cap_{0 \leq j \leq J-1} \bar{I}_{k+j} \middle|\ E_0, \dots, E_{J-1}\right)}{\mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}\right)} \\
&= \frac{\mathbb{P}_k\left(\bar{I}_J \middle| E_0, \dots, E_{J-1}, I_k, \dots, I_{k+J-1}\right) \mathbb{P}_k\left(\cap_{0 \leq j \leq J-1} \bar{I}_{k+j} \middle|\ E_0, \dots, E_{J-1}\right) \mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}, I_k, \dots, I_{k+J-1}\right)}{\mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}\right)} \\
&\leq \mathbb{P}_k\left(\bar{I}_J \middle| E_0, \dots, E_{J-1}, I_k, \dots, I_{k+J-1}\right) \mathbb{P}_k\left(\cap_{0 \leq j \leq J-1} \bar{I}_{k+j} \middle|\ E_0, \dots, E_{J-1}\right),
\end{aligned}$$

where the last equality comes from (9), and the final inequality uses the fact that the events $E_0, \dots, E_{J-1}$ and $I_k, \dots, I_{k+J-1}$ are pairwise independent, thus

$$\mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}, I_k, \dots, I_{k+J-1}\right) = \mathbb{P}_k\left(E_J | E_0, \dots, E_{J-1}\right).$$

Using (11), we then have that

$$\mathbb{P}_k\left(\bar{I}_J \middle| E_0, \ldots, E_{J-1}, I_k, \ldots, I_{k+J-1}\right) = 1 - \mathbb{P}_k\left(I_J | E_0, \ldots, E_{J-1}, I_k, \ldots, I_{k+J-1}\right) \leq 1 - p.$$

Thus,

$$\mathbb{P}_k\left(\bigcap_{0\leq j\leq J} \bar{I}_{k+j} \middle| E_0, \ldots, E_J\right) \leq (1-p)\mathbb{P}_k\left(\bigcap_{0\leq j\leq J-1} \bar{I}_{k+j} \middle| E_0, \ldots, E_{J-1}\right). \tag{12}$$

By a recursive argument on the right-hand side of (12), we thus arrive at (10), which yields the desired conclusion.□

1.5 Proof of [1, Proposition 2]

As in [1, Theorem 4], $T^m_{\epsilon,J}$ clearly is a stopping time. Moreover, if $\pi_k = 1$ for all k, then $T^m_{\epsilon,J} = T_\epsilon$ for every J, where T_ϵ is the stopping time defined in [1, Theorem 4], and therefore the result holds. In what follows, we thus focus on the remaining case.

Consider an iterate k such that x_k is $(\hat{\epsilon}, \hat{\epsilon}^{1/2})$-function stationary and the model $\hat{f}_k$ is accurate. From the definition of $\hat{\epsilon}$, such an iterate is also $((1-\kappa_g)\epsilon, (1-\kappa_H)\epsilon^{1/2})$-function stationary and the model $\hat{f}_k$ is accurate. Then, by a reasoning similar to that of the proof of [1, Lemma 1], we can show that x_k is $(\epsilon, \epsilon^{1/2})$-model stationary. As a result, if $T^m_{\epsilon,J} > k$, one of the models $\hat{f}_k, \hat{f}_{k+1}, \ldots, \hat{f}_{k+J}$ must be inaccurate, which happens with probability $1 - p^{J+1}$.

Let $T_{\epsilon,J}$ be the first iteration index for which the iterate is a $(\hat{\epsilon}, \hat{\epsilon}^{1/2})$ function stationary point and satisfies [1, Eq. (33)], i. e.

$$\min\{\|g_k\|, \|g_k^+\|\} < \epsilon \quad \text{and} \quad \lambda_k > -\epsilon^{1/2}, \quad \forall k \in \{T^m_{\epsilon,J}, T^m_{\epsilon,J}+1, ..., T^m_{\epsilon,J}+J\}.$$

Clearly $T^m_{\epsilon,J} \leq T_{\epsilon,J}$ (for all realizations of these two stopping times), and it thus suffices to bound $T_{\epsilon,J}$ in expectation. By applying [1, Theorem 4] (with ϵ in the theorem's statement replaced by $\hat{\epsilon}$), one can see that there must exist an infinite number of $(\hat{\epsilon}, \hat{\epsilon}^{1/2})$-function stationary points in expectation. More precisely, letting $\{T^{(i)}_{\hat{\epsilon}}\}_{i=1,\ldots}$ be the corresponding stopping times indicating the iteration indexes of these points and using [1, Theorem 4], we have

$$\begin{aligned} \mathbb{E}\left[T^{(1)}_{\hat{\epsilon}}\right] = \mathbb{E}\left[T_{\hat{\epsilon}}\right] &\leq \frac{(f(x_0) - f_{\text{low}})}{p\hat{c}}\,\hat{\epsilon}^{-3/2} + 1, \\ \forall i \geq 1, \quad \mathbb{E}\left[T^{(i+1)}_{\hat{\epsilon}} - T^{(i)}_{\hat{\epsilon}}\right] &\leq \frac{(f(x_0) - f_{\text{low}})}{p\hat{c}}\,\hat{\epsilon}^{-3/2} + 1. \end{aligned}$$

Consider now the subsequence $\{T^{(i_\ell)}_{\hat{\epsilon}}\}_{\ell=1,\ldots}$ such that all stopping times are at least J iterations from each other, i.e., for every $\ell \geq 1$, we have $T^{(i_{\ell+1})}_{\hat{\epsilon}} - T^{(i_\ell)}_{\hat{\epsilon}} \geq J$. For such a sequence, we get

$$\begin{aligned} \forall \ell \geq 1, \quad \mathbb{E}\left[T^{(i_{\ell+1})}_{\hat{\epsilon}} - T^{(i_\ell)}_{\hat{\epsilon}}\right] &\leq \frac{(f(x_0) - f_{\text{low}})}{p\hat{c}}\,\hat{\epsilon}^{-3/2} + J + 1 \triangleq K(\epsilon, J) \\ \mathbb{E}\left[T^{(i_1)}_{\hat{\epsilon}}\right] = \mathbb{E}\left[T_{\hat{\epsilon}}\right] &\leq \frac{(f(x_0) - f_{\text{low}})}{p\hat{c}}\,\hat{\epsilon}^{-3/2} + 1 \leq K(\epsilon, J). \end{aligned}$$

For every $\ell \geq 1$, we define the event

$$B_\ell \;=\; \bigcap_{j=0}^{J} I_{T^{(i_\ell)}_{\hat{\epsilon}}+j} \;=\; \bigcap_{j=0}^{J} \left\{ m_{T^{(i_\ell)}_{\hat{\epsilon}}+j} \text{ is accurate} \right\}.$$

By [1, Assumption 5], the samples are generated independently of the current iterate, and for every k, $\mathbb{P}(I_k|\mathcal{F}_{k-1}) = p$. By the same recursive reasoning as in the proof of [1, Proposition 1], we have that $\mathbb{P}\left(B_\ell|\mathcal{F}_{T_{\hat{\epsilon}}^{(i_\ell)}-1}\right) = p^{J+1}$. Moreover, by definition of the sequence $\{T_{\hat{\epsilon}}^{(i_\ell)}\}$, two stopping times in that sequence correspond to two iteration indexes distant of at least $J+1$. Therefore, they also correspond to two separate sequences of $(J+1)$ models that are generated in an independent fashion. We can thus consider $\{B_\ell\}$ to be an independent sequence of Bernoulli trials. Therefore, the variable G representing the number of runs of B_ℓ until success follows a geometric distribution with an expectation less than $\frac{1}{p^{J+1}} = p^{-(J+1)} < \infty$. On the other hand, $T_{\epsilon,J}$ is less than the first element of $\{T_{\hat{\epsilon}}^{(i_\ell)}\}$ for which B_ℓ happens, and thus $T_{\epsilon,J} \leq T_{\hat{\epsilon}}^{(i_G)}$. To conclude the proof, we define

$$S_G = T_{\hat{\epsilon}}^{(i_G)}, \qquad X_1 = T_{\hat{\epsilon}}^{(i_1)} = T_{\hat{\epsilon}}^{1}, \quad X_\ell = T_{\hat{\epsilon}}^{(i_\ell)} - T_{\hat{\epsilon}}^{(i_{\ell-1})} \quad \forall \ell \geq 2.$$

From the proof of Wald's equation [2, Theorem 4.1.5] (more precisely, from the third equation appearing in that proof), one has

$$\mathbb{E}\left[S_G\right] = \sum_{\ell=1}^{\infty} \mathbb{E}\left[X_\ell\right] \mathbb{P}(G \geq \ell).$$

Since $\mathbb{E}\left[X_\ell\right] \leq K(\epsilon, J)$, one arrives at

$$\mathbb{E}\left[T_{\epsilon,J}^m\right] \leq \mathbb{E}\left[T_{\epsilon,J}\right] \leq \mathbb{E}\left[T_{\hat{\epsilon}}^{(i_G)}\right] \leq K(\epsilon, J) \sum_{\ell=1}^{\infty} \mathbb{P}(G \geq \ell) = K(\epsilon, J)\mathbb{E}\left[G\right],$$

which is the desired result.□

2 More details on numerical results

2.1 ALAS algorithm as implemented

Our implementation of the ALAS algorithm is described in Algorithm 2. The main differences between Algorithm 2 and Algorithm 1 are described in the main paper.

2.2 Distribution of Steps

IJCNN Dataset We have included most of the visual information for the runs on the IJCNN in the main part of the paper. One additional consideration to verify the stability of the algorithm is to perform a sensitivity analysis of the hyperparameters of ALAS. There are two hyperparameters in the ALAS algorithm, η and ϵ. In deterministic contexts, η is typically kept small to allow for fairly liberal step acceptance. We studied the impact of varying this hyperparameter on the performance of ALAS, reporting our results in Figure 1. We confirm that indeed for reasonably small values the precise value has limited impact on the convergence. This is by contrast with SGD wherein the learning rate has a very significant impact on the performance and typically has to be tuned for each problem. The sensitivity of the Algorithm with respect to ϵ is shown in Figure 2. It can be seen that here, as well, the Algorithm is not particularly sensitive. Although the Regularized Newton step, which depends on ϵ explicitly, is chosen the most, it appears that it is modified in an appropriate space so as to mitigate any degradation of performance.

Algorithm 2: ALAS, as implemented.

Initialization: Choose $x_0 \in \mathbb{R}^n$, $\theta \in (0,1), \eta > 0$, $\epsilon > 0$.
for $k = 0, 1, ...$ **do**

1. Draw a random sample set $\mathcal{S}_k \subset \{1, \ldots, N\}$, and compute the associated quantities $g_k := g(x_k; \mathcal{S}_k), H_k := H(x_k; \mathcal{S}_k)$. Form the model:
$$\hat{f}_k(x_k + s) := \hat{f}(x_k + s; \mathcal{S}_k). \tag{13}$$
2. Compute $R_k = \frac{g_k^T H_k g_k}{\|g_k\|^2}$.
3. If $R_k < -\|g_k\|^{1/2}$ then set $d_k = \frac{R_k}{\|g_k\|} g_k$ and go to the Line-search step.
4. Else if $R_k < \|g_k\|^{1/2}$ and $\|g_k\| \geq \epsilon$ then set $d_k = -\frac{g_k}{\|g_k\|^{1/2}}$ and go to the Line-search step.
5. Compute λ_k as the minimum eigenvalue of the Hessian estimate H_k.
If $\lambda_k \geq -\epsilon^{1/2}$ and $\|g_k\| = 0$ set $\alpha_k = 0,\ d_k = 0$ and go to Step 10.
6. If $\lambda_k < -\|g_k\|^{1/2}$, Compute a negative eigenvector v_k such that
$$H_k v_k = \lambda_k v_k,\ \|v_k\| = -\lambda_k,\ v_k^\top g_k \leq 0, \tag{14}$$
set $d_k = v_k$ and go to the line-search step.
7. If $\lambda_k > \|g_k\|^{1/2}$, compute a Newton direction d_k solution of
$$H_k d_k = -g_k, \tag{15}$$
go to the line-search step.
8. If d_k has not yet been chosen, compute it as a regularized Newton direction, solution of
$$\left(H_k + (\|g_k\|^{1/2} + \epsilon^{1/2})\mathbb{I}_n\right) d_k = -g_k, \tag{16}$$
and go to the line-search step.
9. **Line-search step** Compute the minimum index j_k such that the step length $\alpha_k := \theta^{j_k}$ satisfies the decrease condition:
$$\hat{f}_k(x_k + \alpha_k d_k) - \hat{f}_k(x_k) \ \leq\ -\frac{\eta}{2}\alpha_k^2 \|d_k\|^2. \tag{17}$$
10. Set $x_{k+1} = x_k + \alpha_k d_k$.
11. Set $k = k + 1$.

end

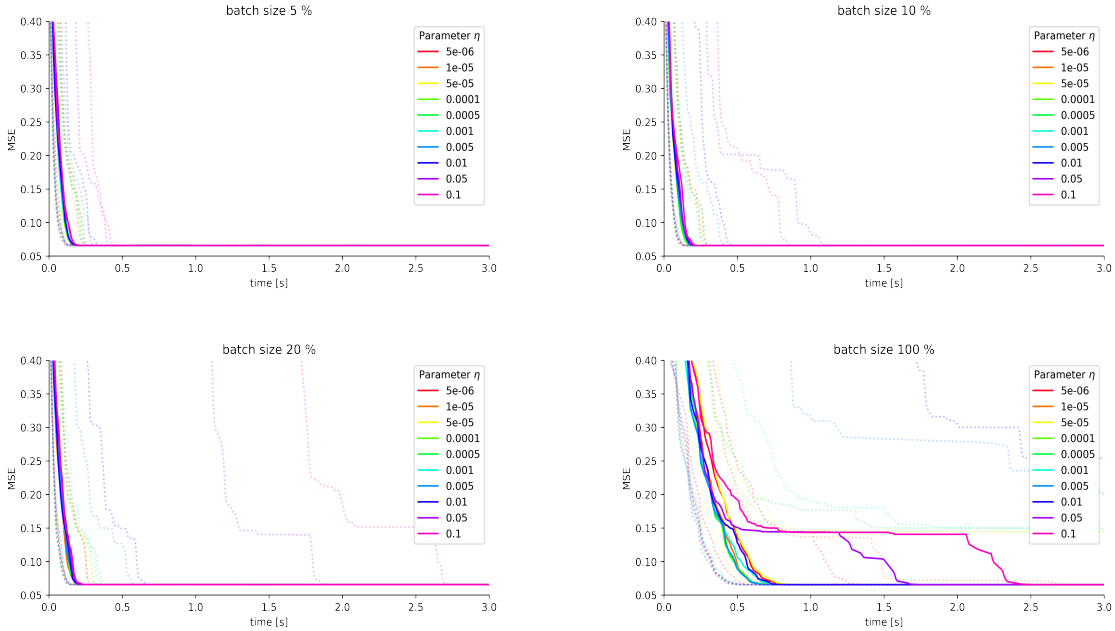


Figure 1: The sensitivity of the performance of the ALAS algorithm on the IJCNN1 task as depending on the choice of η.

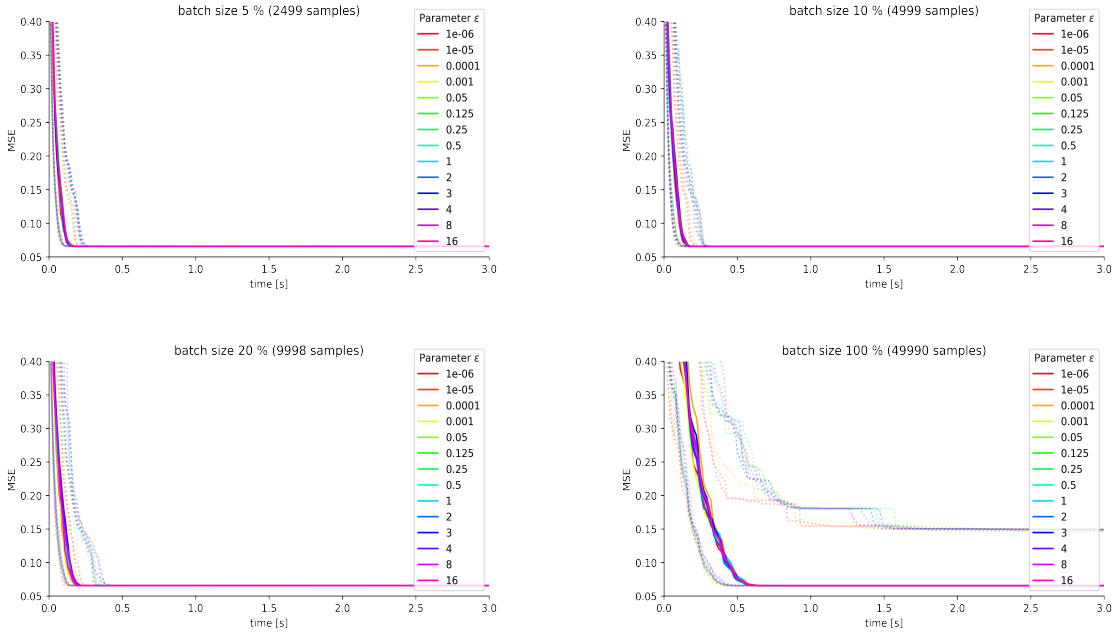
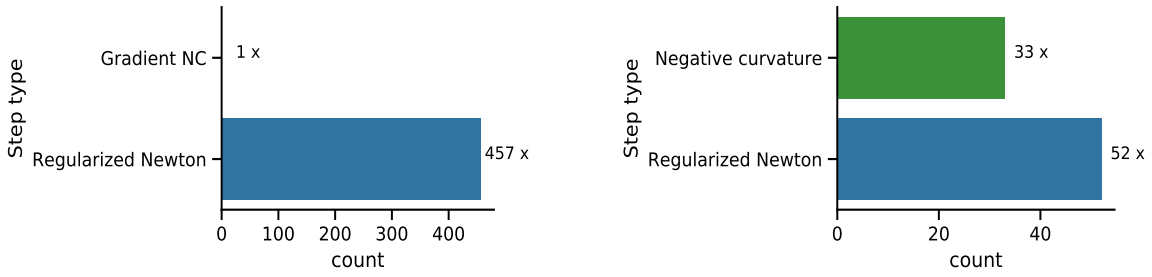


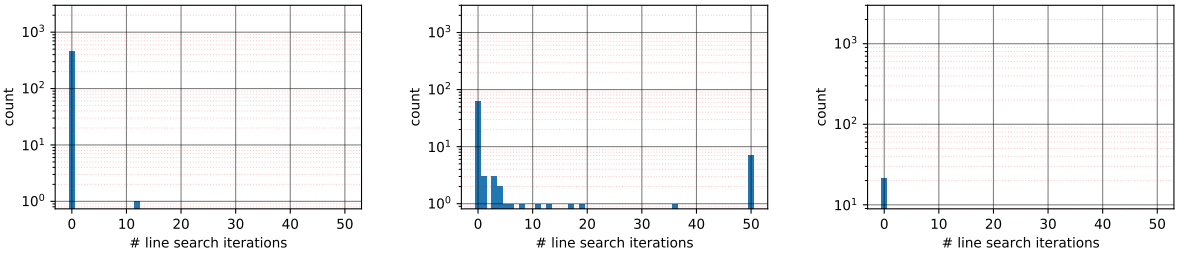
Figure 2: The sensitivity of the performance of the ALAS algorithm on the IJCNN1 task as depending on the choice of ϵ.



(a) 123-1 5%

(b) 123-1 100%

Figure 3: The step type distribution of a single run of ALAS algorithm on the A9A task for the 123-1 architecture and two sampling sizes.



(a) 123-1 5%

(b) 123-1 100%

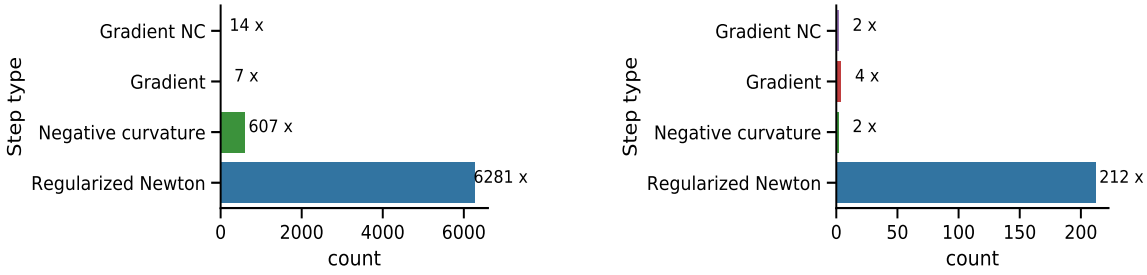
(c) 123-2-1 100%

Figure 4: Plots of the number of line-search iterations during each update for the A9A task for selected runs of the ALAS algorithm with different architectures and sampling sizes. The maximum number of line-search iterations was set to 50.

A9A Dataset The distribution of the individual type of steps as described in Algorithm 2 for selected runs for the A9A task is shown in Figure 3. As in the case of IJCNN1 task, the most frequently used step type is the *Regularized Newton*. We note that when a full sample is used, *Negative curvature* steps are more common, suggesting that the problem is quite nonconvex. Note that in all of our runs (including those not reported here), other step choices were used less than 10 times. The distributions of number of line-search iterations for selected runs of the ALAS algorithm are shown in Figure 4.

MNIST Transfer Learning The distribution of the individual type of steps as described in Algorithm 2 for selected runs for the transfer learning task is shown in Figure 5. Again, the most frequently used step type was the *Regularized Newton* and the others occurred very rarely. The distribution of the number of line-search iterations for selected runs of the ALAS algorithm are shown in Figure 6; the line search was usually rather short (0 iterations of were dominating for most runs) but it was still used quite often.

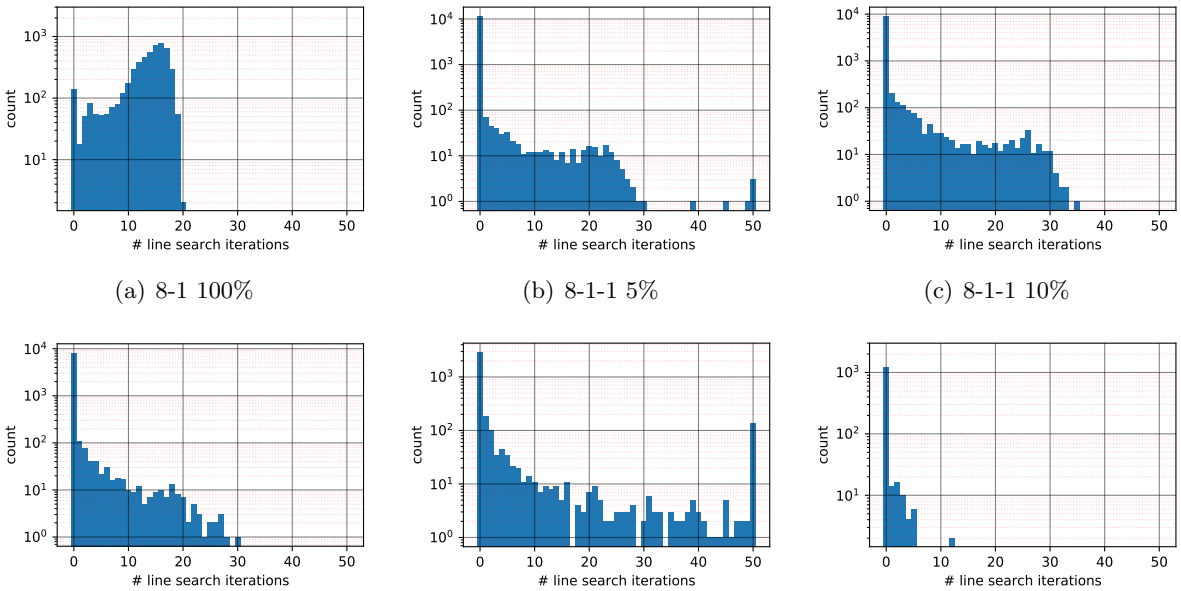
Artificial NN The step type distribution for different runs is shown in Figure 7 — the most frequent steps in general for this task were the *Regularized Newton* (strongly dominating) and the *Negative curvature* step.



(a) 8-2-1 5%

(b) 8-4-1 100%

Figure 5: The step type distribution of a single run of ALAS algorithm on the transfer learning task.



(a) 8-1 100%

(b) 8-1-1 5%

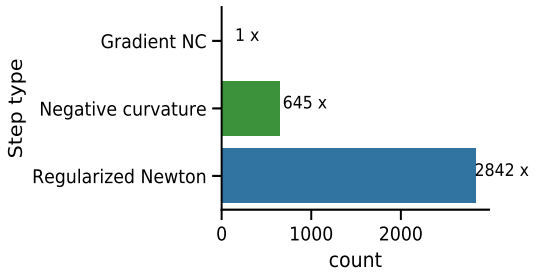
(c) 8-1-1 10%

(d) 8-1-1 20%

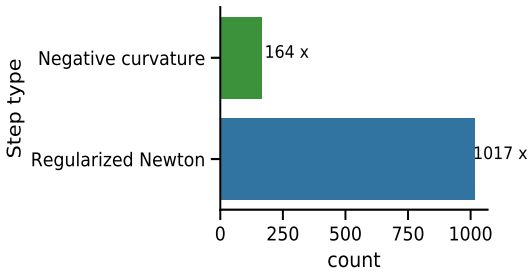
(e) 8-1-1 100%

(f) 8-2-1 100%

Figure 6: Plots of the number of line-search iterations during each update for the transfer learning task for selected runs of the ALAS algorithm with different architectures and sampling sizes. The maximum number of line-search iterations was set to 50.



(a) 2-1-1 20%



(b) 2-4-2-1 5%

Figure 7: The step type distribution of a single run of ALAS algorithm on the NN1 task.

2.3 Additional plots for IJCNN1

We tested SGD with several possible values for the learning rate, namely 1, 0.6, 0.3, 0.1, 0.01, 0.001. Figure 8 shows the performance of a subset of those values: as one may expect, SGD is quite sensitive to the choice of the step size.

3 Additional numerical experiments

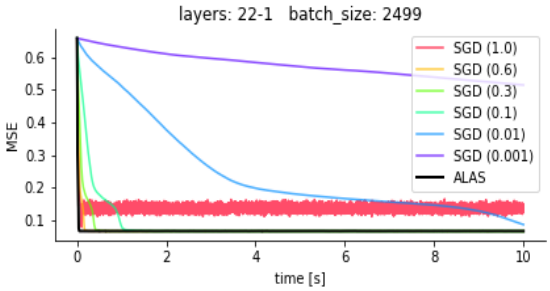
3.1 Experiment using the A9A dataset

The second set of experiments was run on the A9A training dataset, also part of the LIBSVM library [**?**]: for this classification dataset, we have $N = 32,561$ samples, each of them possessing $n = 123$ features. For this task, we trained two neural networks without hidden layers and with one hidden layer having a single neuron; the optimized function was the MSE as in the first experiment. Our batch sizes consisted of 100%, 20%, 10% and 5% of the dataset size, and the mean-squared error loss was used. We tested four variants of SGD corresponding to the values 1, 0.6, 0.3, 0.1 for the learning rate, and selected the best variant for comparison with ALAS. The results are shown in Figures 9 and 10, as well as Table 1. Our method, ALAS, was significantly better for all sampling sizes on training the first network. However, for the second network with more optimization variables (learning parameters), ALAS appears to slow down relative to SGD. This appears due to our use of exact linear algebra techniques: as explained in the introduction of this section, the results could be better for an inexact variant of ALAS.

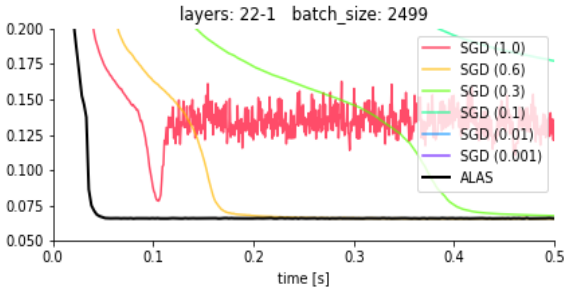
3.2 An artificial dataset

To illustrate the performance of ALAS on highly non-linear problems, we have created two artificial datasets, the first of which (thereafter called NN1) was generated using a neural network with random weights sampled from a normal distribution ($w_i \in \mathcal{N}(0, 3)$) and two input neurons, two hidden layers with four and two neurons, respectively. We use hyperbolic tangent activation functions in all layers. To produce the actual training data, 50,000 points were sampled from a uniform distribution ($\vec{x_i} \in \mathcal{R}^2, x_{ij} \in \mathcal{U}(0, 1)$) and passed through the generated network to produce target values $\{y_i\}_i$.

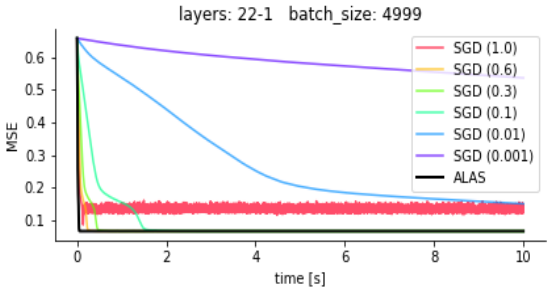
The optimization task was then to reconstruct the target values $\{y_i\}_i$ from the generated points $\{\vec{x_i}\}_i$ using different neural networks using the MSE as the loss function. We tried the



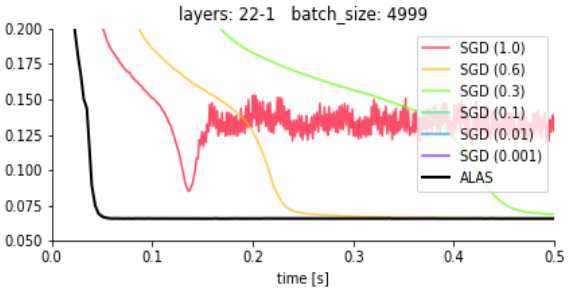
(a) 22-1 5%



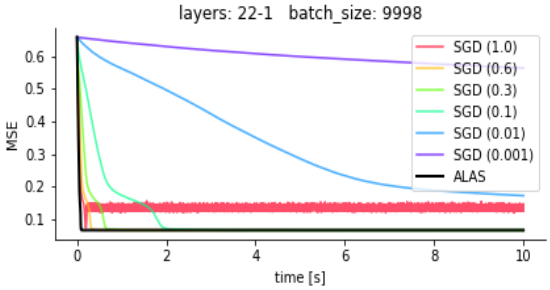
(b) 22-1 5% (zoom)



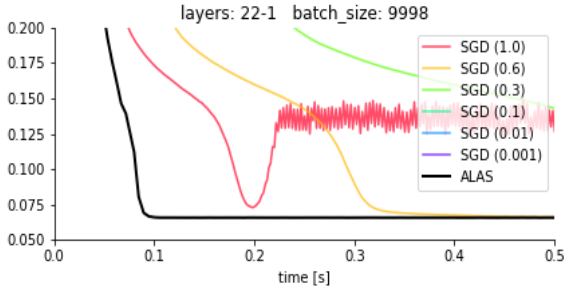
(c) 22-1 10%



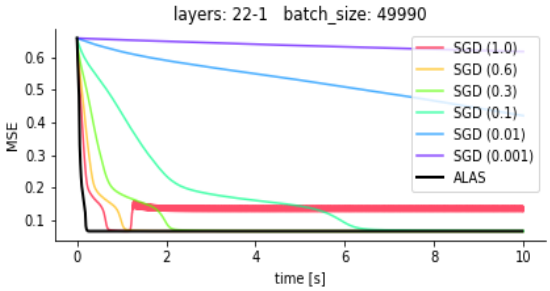
(d) 22-1 10% (zoom)



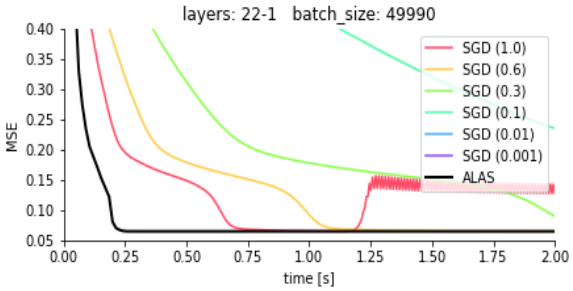
(e) 22-1 20%



(f) 22-1 20% (zoom)

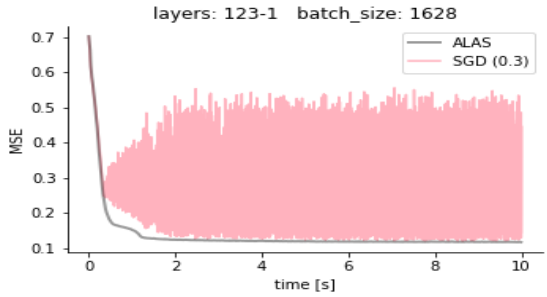


(g) 22-1 100%

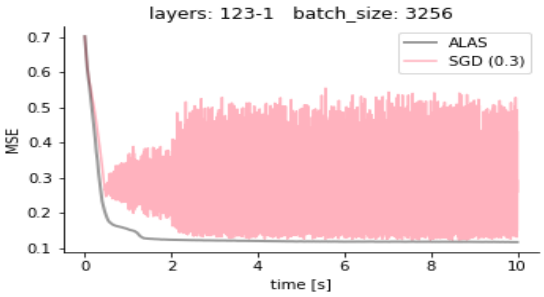


(h) 22-1 100% (zoom)

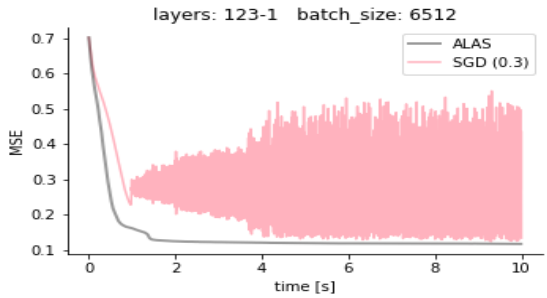
Figure 8: Comparison of ALAS and SGD (with various learning rates) on the IJCNN1 dataset.



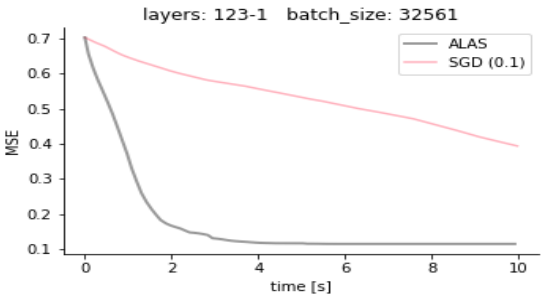
(a) 123-1 5%



(b) 123-1 10%

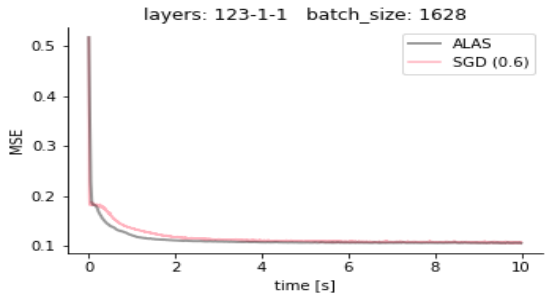


(c) 123-1 20%

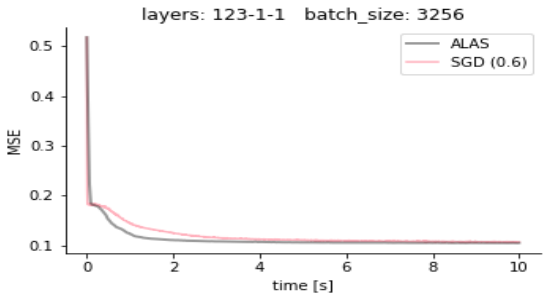


(d) 123-1 100%

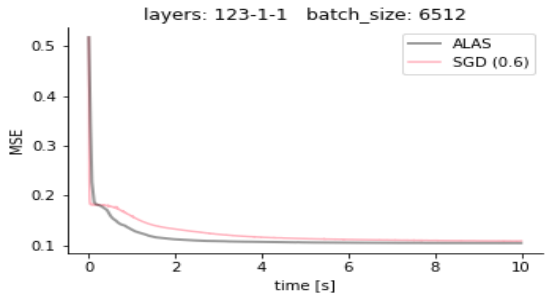
Figure 9: Comparison of ALAS and SGD (with best performing learning rate) on the A9A dataset with a simple neural network with 123 input neurons, no hidden layer and an output neuron.



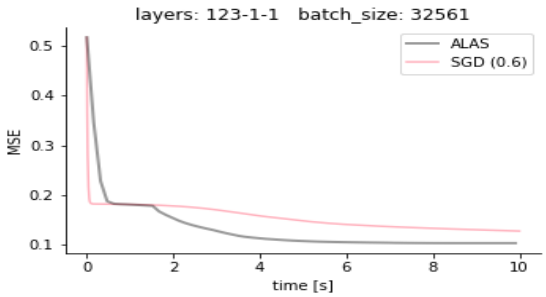
(a) 123-1-1 5%



(b) 123-1-1 10%



(c) 123-1-1 20%



(d) 123-1-1 100%

Figure 10: Comparison of ALAS and SGD (with best performing learning rate) on the A9A dataset with a simple neural network with 123 input neurons, hidden layer with a single neuron and an output neuron.

Layers: 123-1					Layers: 123-2-1				
alg.	π_k	min loss	loss [8-10]s	iter.	alg.	π_k	min loss	loss [8-10]s	iter.
ALAS	5%	**0.1176**	**0.1182**	458	ALAS	5 %	0.1428	0.1437	20
SGD (0.1)	5%	0.1224	0.1231	8260	SGD (0.6)	5 %	**0.1066**	**0.1085**	5992
ALAS	10 %	**0.1168**	**0.1176**	360	ALAS	10 %	0.1464	0.1482	13
SGD (0.1)	10 %	0.1245	0.1255	6168	SGD (0.6)	10 %	**0.1094**	**0.1129**	3948
ALAS	20 %	**0.1163**	**0.1171**	255	ALAS	20 %	0.1436	0.1452	14
SGD (0.1)	20 %	0.1396	0.1482	4152	SGD (0.6)	20%	**0.1163**	**0.1213**	2449
ALAS	100 %	**0.1151**	**0.1151**	85	ALAS	100 %	0.1550	0.1579	6
SGD (0.3)	100 %	0.2157	0.2786	645	SGD (0.6)	100 %	**0.1353**	**0.1365**	819

Table 1: Results reached over the given time period $t = 10$ s on the A9A task.

values 1, 0.6, 0.3, 0.1 for the learning rate of SGD, and we compared these variants with ALAS, using 100%, 20 %, 10 %, and 5% of the sample sizes. The results are shown in Table 2: the algorithm ALAS performed best with a single exception when it got stuck in a worse local optimum than the SGD variant.

Layers: 2-1-1					Layers: 2-2-1				
alg.	π_k	min loss	loss [8-10]s	iter.	alg.	π_k	min loss	loss [8-10]s	iter.
ALAS	5%	**9.08e−10**	**1.12e−9**	7471	ALAS	5%	**6.59e−10**	4.000	5736
SGD (1.0)	5%	$2.49e{-6}$	$2.76e{-6}$	15432	SGD (1.0)	5%	$1.60e{-6}$	**1.78e−6**	14105
ALAS	10%	**8.94e−10**	**1.01e−9**	5595	ALAS	10%	**1.59e−10**	**2.18e−10**	2352
SGD (1.0)	10%	$2.77e{-6}$	$3.09e{-6}$	14015	SGD (1.0)	10%	$1.86e{-6}$	$2.07e{-6}$	12134
ALAS	20%	**8.91e−10**	**9.39e−10**	3353	ALAS	20%	**1.96e−10**	**2.23e−10**	1267
SGD (1.0)	20%	$3.41e{-6}$	$3.80e{-6}$	11533	SGD (1.0)	20%	$2.20e{-6}$	$2.44e{-6}$	10283
ALAS	100%	**1.15e−9**	**1.211e−9**	538	ALAS	100%	**9.62e−10**	**9.99e−10**	304
SGD (1.0)	100%	$7.51e{-6}$	$8.42e{-6}$	5503	SGD (1.0)	100%	$5.99e{-6}$	$6.68e{-6}$	3833
Layers: 2-4-1-1					Layers: 2-4-2-1				
ALAS	5%	**9.21e−10**	**9.23e−10**	1529	ALAS	5%	**9.27e−10**	**9.34e−10**	922
SGD (1.0)	5%	$5.47e{-7}$	$5.60e{-7}$	13224	SGD (1.0)	5%	$1.68e{-6}$	$1.86e{-6}$	11909
ALAS	10%	**9.94e−10**	**1.03e−9**	818	ALAS	10%	**1.03e−9**	**1.05e−9**	493
SGD (1.0)	10%	$5.75e{-7}$	$5.88e{-7}$	10873	SGD (1.0)	10%	$2.17e{-6}$	$2.39e{-6}$	9192
ALAS	20%	**1.26e−9**	**1.34e−9**	485	ALAS	20%	**1.25e−9**	**1.30e−9**	262
SGD (1.0)	20%	$6.03e{-7}$	$6.12e{-7}$	8223	SGD (1.0)	20%	$2.91e{-6}$	$3.20e{-6}$	6850
ALAS	100%	**4.05e−9**	**4.43e−9**	94	ALAS	100%	**3.59e−9**	**3.96e−9**	55
SGD (1.0)	100%	$6.76e{-7}$	$6.80e{-7}$	2819	SGD (1.0)	100%	$9.76e{-6}$	$1.09e{-5}$	2008

Table 2: Results reached over the given time period $t = 10$ s on the artificial dataset NN1.

3.3 A second artificial dataset

The second artificial dataset, called NN2, was created using the same process as described in the main paper for the first. For this second dataset, we used a deeper neural network in order to introduce more nonlinearities: this network consisted of four input neurons, three hidden layers with up to four neurons, and one output layer producing the targets $\{y_i\}_i$. The networks used for experiments had one or two hidden layers — first hidden layer had always four neurons and the second had one or two neurons if present. All neurons used a hyperbolic tangent as their

activation function, the optimized function was the MSE. Few runs of the ALAS and the SGD algorithms are shown in Figures 11 and 12. The step type distribution for different runs is shown in Figure 13 — the *Regularized Newton* step type was the most frequent. Our implementation of ALAS performed generally better than the best SGD variant, sometimes by a significant margin.

Layers: 4-4-1					Layers: 4-4-2-1				
alg.	π_k	min loss	loss [8-10]s	iter.	alg.	π_k	min loss	loss [8-10]s	iter.
ALAS	5%	**0.0153**	**0.0163**	1654	ALAS	5%	**0.0150**	**0.0154**	1046
SGD (1.0)	5%	0.0411	0.0466	13185	SGD (1.0)	5%	0.0167	0.0222	11814
ALAS	10%	**0.0145**	**0.0156**	1218	ALAS	10%	**0.0160**	**0.0161**	650
SGD (1.0)	10%	0.0466	0.0513	10548	SGD (1.0)	10%	0.0185	0.0260	8764
ALAS	20%	**0.0158**	**0.0168**	727	ALAS	20%	**0.0159**	**0.0160**	320
SGD (1.0)	20%	0.0527	0.0562	7617	SGD (1.0)	20%	0.0242	0.0317	6123
ALAS	100%	**0.0122**	**0.0133**	126	ALAS	100%	**0.0148**	**0.0153**	69
SGD (1.0)	100%	0.0646	0.0651	3068	SGD (1.0)	100%	0.0384	0.0489	2222

Table 3: The comparison of the minimal (full) losses reached over the given time period $t = 10$ s on the artificial dataset NN2. The number in the parenthesis for SGD entries is the step size. The column *loss [8-10]s* shows the median loss over the last two seconds.

References

[1] El Houcine Bergou, Youssef Diouane, Vladimir Kunc, Vyacheslav Kungurtsev, and Clément W. Royer. A subsampling line-search method with second-order results. INFORMS Journal on Optimization, 2022 (to appear).

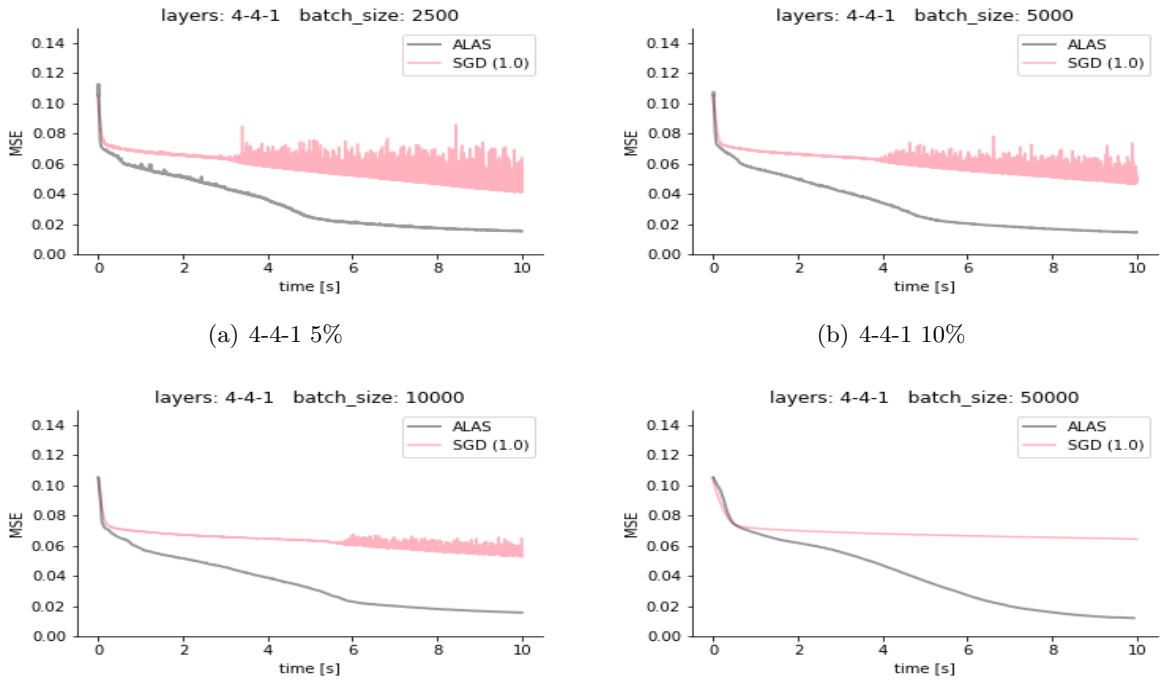
[2] Rick Durrett. *Probability: Theory and Examples*. Camb. Ser. Stat. Prob. Math. Cambridge University Press, Cambridge, fourth edition, 2010.

[3] Farbod Roosta-Khorasani and Michael W. Mahoney. Sub-sampled Newton methods I: Globally convergent algorithms. *arXiv preprint arXiv:1601.04737*, 2016.

[4] Clément W. Royer and Stephen J. Wright. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM J. Optim.*, 28:1448–1477, 2018.

[5] Joel A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8:1–230, 2015.

[6] Peng Xu, Farbod Roosta-Khorasani, and Michael W. Mahoney. Newton-type methods for non-convex optimization under inexact hessian information. *Math. Program.*, 2019.



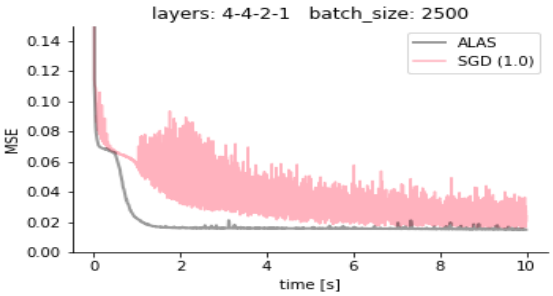
(a) 4-4-1 5%

(b) 4-4-1 10%

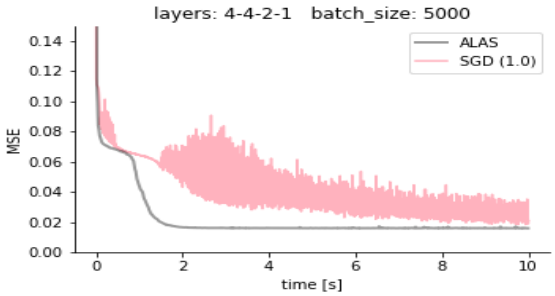
(c) 4-4-1 20%

(d) 4-4-1 100%

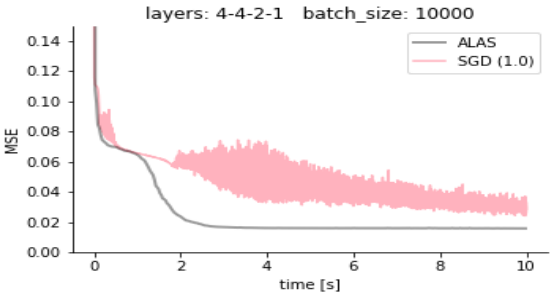
Figure 11: Evaluation of the ALAS algorithm on artifical task NN2 compared to the SGD with best performing learning rate. Full losses are depicted.



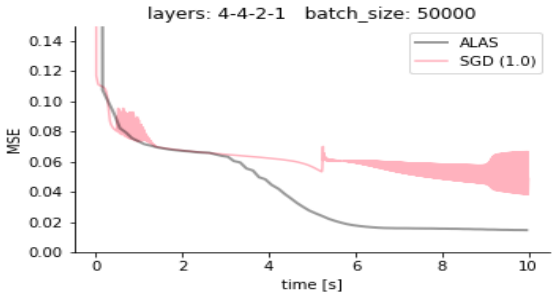
(a) 4-4-2-1 5%



(b) 4-4-2-1 10%

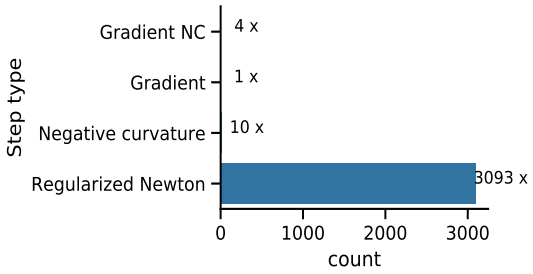


(c) 4-4-2-1 20%

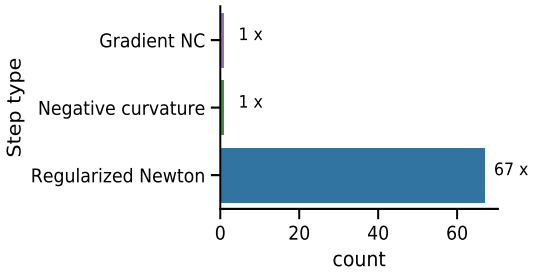


(d) 4-4-2-1 100%

Figure 12: Evaluation of the ALAS algorithm on artificial task NN2 compared to the SGD with best performing learning rate. Full losses are depicted.



(a) 4-2-1 10%



(b) 4-4-2-1 100%

Figure 13: The step type distribution of a single run of ALAS algorithm on the NN2 task.