Convergence and Accuracy Trade-Offs in Federated Learning and Meta-Learning

Zachary Charles
Google Research

Jakub Konečný
Google Research

Abstract

We study a family of algorithms, which we refer to as *local update methods*, generalizing many federated and meta-learning algorithms. We prove that for quadratic models, local update methods are equivalent to first-order optimization on a surrogate loss we exactly characterize. Moreover, fundamental algorithmic choices (such as learning rates) explicitly govern a trade-off between the condition number of the surrogate loss and its alignment with the true loss. We derive novel convergence rates showcasing these trade-offs and highlight their importance in communication-limited settings. Using these insights, we are able to compare local update methods based on their convergence/accuracy trade-off, not just their convergence to critical points of the empirical loss. Our results shed new light on a broad range of phenomena, including the efficacy of server momentum in federated learning and the impact of proximal client updates.

1 Introduction

Federated learning (McMahan et al., 2017) is a distributed framework for learning models without directly sharing data. In this framework, clients perform *local updates* (typically using first-order optimization) on their own data. In the popular FedAvg algorithm (McMahan et al., 2017), the client models are then averaged at a central server. Since the proposal of FedAvg, many new federated optimization algorithms have been developed (Li et al., 2020a; Reddi et al., 2020; Hsu et al., 2019; Xie et al., 2019; Basu et al., 2019; Li et al., 2020b; Karimireddy et al., 2019). These methods typically employ multiple local client epochs in order to improve communication-efficiency. We defer to Kairouz et al. (2019) and Li et al. (2019) for more detailed summaries of federated learning.

Local updates have also been used extensively in meta-learning. The celebrated MAML algorithm (Finn et al., 2017) employs multiple local model updates on a set of tasks in order to learn a model that quickly adapt to new tasks. MAML has inspired a number of model-agnostic meta-learning methods that also employ first-order local updates (Balcan et al., 2019; Fallah et al., 2020a; Nichol et al., 2018; Zhou et al., 2019). There are strong connections between federated learning and meta-learning, despite differences in practical concerns. Formal connections between the two were shown by Khodak et al. (2019) and have since been explored in many other works (Jiang et al., 2019; Fallah et al., 2020b).

We refer to methods that utilize multiple local updates across clients (or in the language of meta-learning, tasks) as *local update methods* (see Section 2.1 for a formal characterization). In practice, local update methods frequently outperform "centralized" methods such as SGD (McMahan et al., 2017; Finn et al., 2017; Hard et al., 2018; Yang et al., 2018; Hard et al., 2020). However, the empirical benefits of local update methods are not fully explained by existing theoretical analyses. For example, Woodworth et al. (2020) show that FedAvg often obtains convergence rates comparable to or worse than those of mini-batch SGD.

We focus on two difficulties that arise when analyzing local update methods. First, analyses must account for *client drift* (Karimireddy et al., 2019). As clients perform local updates on heterogeneous datasets, their local models drift apart. This hinders convergence to globally optimal models, and makes theoretical analyses more challenging. Similar phenomena were examined by Li et al. (2020a); Malinovsky et al. (2020); Pathak and Wainwright (2020) and Fallah et al. (2020b), who show that various local update methods do not converge to critical points of

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

the empirical loss.

Second, local update methods are difficult to compare. Analyses of different methods may use different hyperparameters regimes, or make different assumptions. Even comparing seemingly similar methods can require significant theoretical insight (Karimireddy et al., 2019; Fallah et al., 2020b). Moreover, comparisons can be made in fundamentally different ways. One may wish to maximize the final accuracy, or minimize the number of communication rounds needed to attain a given accuracy. Thus, it is not even clear *how* local update methods should be compared.

Contributions In this work, we invert the conventional narrative that issues such as client drift harm convergence. Instead, we view such phenomena as improving convergence, but to sub-optimal points.

More generally, we show that local update methods face a fundamental trade-off between convergence and accuracy that is explicitly governed by algorithmic hyperparameters. Perceived failures of methods such as FedAvg actually correspond to operating points prioritizing convergence over accuracy. We use this trade-off to develop a novel framework for comparing local update methods. We compare methods based on their entire convergence-accuracy trade-off, not just their convergence to optimal points. In more detail:

1. We show that for quadratic models, local update methods are equivalent to optimizing a single *surrogate* loss function. The condition number of the surrogate is controlled by algorithmic choices. Popular local update methods, including FedAvg and MAML, reduce the surrogate's condition number, but increase the discrepancy between the empirical and surrogate losses. Our results also encompass *proximal* local update methods (Li et al., 2020a; Zhou et al., 2019).

2. We derive novel convergence rates that showcasing this trade-off between convergence and accuracy. Our bounds demonstrate the benefit of local update methods over methods such as mini-batch SGD in communication-limited settings.

3. We use this theory to develop a framework for comparing local update methods through a novel *Pareto frontier*, which compares convergence-accuracy trade-offs of classes of algorithms. We use this to derive novel comparisons of many popular local update methods.

4. We use this technique to shed light on a broad range of phenomena, including the benefit of server momentum, the effect of proximal local updates, and differences between the dynamics of FedAvg and MAML.

5. While our theoretical results are restricted to quadratic models, we show that such convergence-accuracy trade-offs occur empirically in non-convex settings. We also validate our theoretical observations regarding server momentum and proximal updates on a non-convex task.

We view our work as a step towards holistic understandings of local update methods. Using the aforementioned Pareto frontiers, we highlight a number of new phenomena and open problems. One particularly intriguing observation is that the convergence-accuracy trade-off for FedAvg with heavy-ball server momentum appears to be completely symmetric. For more details, see Section 5. Our proof techniques may be of independent interest. We derive a novel analog of the Bhatia-Davis inequality (Bhatia and Davis, 2000) for mean absolute deviations, and use this to understand the accuracy of local update methods.

Notation We let $\|\cdot\|$ denote the ℓ_2 norm for vectors and the spectral norm for matrices. For a symmetric positive semi-definite matrix A, we let $A^{1/2}$ denote its matrix square root. We let $\preceq$ be the *Loewner order* on positive semi-definite matrices. For a real symmetric matrix A, we let $\lambda_{\max}(A)$, $\lambda_{\min}(A)$ denote its largest and smallest eigenvalues, and let $\text{cond}(A)$ denote their ratio. In a slight abuse of notation, if f is a L-smooth, μ-strongly convex function, we say $\text{cond}(f) \leq L/\mu$.

Accuracy and Meta-Learning We study the accuracy of local update methods on the training population. However, meta-learning algorithms are designed to learn a model that adapts well to new tasks; The empirical loss is not necessarily indicative of the "post-adaptation" accuracy of such methods (Finn et al., 2017). Despite this our focus still yields novel insights into qualitative differences between the training dynamics of federated learning and meta-learning methods. Perhaps surprisingly, we show that in certain hyperparameter regions, these methods exhibit identical trade-offs between convergence and pre-adaptation accuracy (see Figures 4 and 5). While we believe our results can be adapted to post-adaptation accuracy via techniques developed by Fallah et al. (2020a), we leave the analysis to future work.

2 Problem Setup

Let $\mathcal{I}$ denote some collection of clients, and let $\mathcal{P}$ be a distribution over $\mathcal{I}$. For each $i \in \mathcal{I}$, there is an associated distribution $\mathcal{D}_i$ over the space $\mathcal{Z}$ of examples.

For any $z \in \mathcal{Z}$, we assume there is symmetric matrix $B_z \in \mathbb{R}^{d \times d}$ and vector $c_z \in \mathbb{R}^d$ such that the loss of a model $x \in \mathbb{R}^d$ at z is given by

$$f(x; z) := \tfrac{1}{2}\|B_z^{1/2}(x - c_z)\|^2. \quad (1)$$

For $i \in \mathcal{I}$, we define the client loss function f_i and the overall loss function f as follows:

$$f_i(x) := \mathop{\mathbb{E}}_{z \sim \mathcal{D}_i}[f(x; z)], \quad f(x) := \mathop{\mathbb{E}}_{i \sim \mathcal{P}}[f_i(x)]. \quad (2)$$

The joint distribution $(\mathcal{I}, \mathcal{Z})$ defines a distribution over $\mathcal{Z}$, recovering standard risk minimization, as well as distributed risk minimization in which $\mathcal{P}$ and all $\mathcal{D}_i$ are uniform over finite sets. For $i \in \mathcal{I}$, define:

$$A_i := \mathop{\mathbb{E}}_{z \sim \mathcal{D}_i}[B_z], \quad c_i := A_i^{-1} \mathop{\mathbb{E}}_{z \sim \mathcal{D}_i}[B_z c_z]. \quad (3)$$

We assume these expectations exist and are finite. One can show that up to some additive constant,

$$f_i(x) = \tfrac{1}{2}\|A_i^{1/2}(x - c_i)\|^2.$$

We make the following assumptions throughout.

Assumption 1. *There are $\mu, L > 0$ such that for all i, $\mu I \preceq A_i \preceq LI$.*

Assumption 2. *There is some $C > 0$ such that for all i, $\|c_i\| \leq C$.*

Assumption 1 bounds the Lipschitz and strong convexity parameters of the f_i, and holds if the matrices B_z satisfy bounded eigenvalue conditions. Assumption 2 states the c_i are bounded. Intuitively, local update methods provide larger benefit for smaller values of C, as the clients progress towards similar optima. While our analysis can be directly generalized to the case where the c_i are contained in a ball of radius C about $p \in \mathbb{R}^d$, we assume $p = 0$ for simplicity. Assumptions 1 and 2 can be relaxed to only hold in expectation, though this complicates the analysis.

2.1 Local Update Methods

We consider a class of algorithms we refer to as *local update* methods. In these methods, at each round t the server samples a set I_t of M clients (in the language of meta-learning, tasks) from $\mathcal{P}$, and broadcasts its model x_t to all clients in I_t. Each client $i \in I_t$ optimizes its loss function f_i (starting at x_t) by applying K iterations of mini-batch SGD with batch size B and client learning rate γ. As proposed by Li et al. (2020a) and Zhou et al. (2019), clients also add ℓ_2 regularization with parameter $\alpha \geq 0$ towards the broadcast model x_t.

The client sends a linear combination of the gradients it computes to the server. The coefficients of the linear combination are given by $\Theta = (\theta_1, \theta_2, \ldots, \theta_k, \ldots)$ for $\theta_i \in \mathbb{R}_{\geq 0}$, where Θ has finite and non-zero support. For such Θ, we define

$$K(\Theta) := \max\{k \mid \theta_k > 0\}, \quad w(\Theta) = \sum_{k=1}^{K(\Theta)} \theta_k. \quad (4)$$

After receiving all client updates, the server treats their average q_t as an estimate of the gradient of the loss function f, and applies q_t to a first-order optimization algorithm ServerOpt. For example, the server could perform a gradient descent step using the "pseudo-gradient" q_t. We refer to this process (parameterized by α, γ, Θ and ServerOpt) as LocalUpdate and give pseudo-code in Algorithms 1 and 2.

Algorithm 1 LocalUpdate: ServerUpdate

ServerUpdate$(x, \text{ServerOpt}, \alpha, \gamma, \Theta)$**:**
  $x_0 = x$
  for each round $t = 0, 1, \ldots, T-1$ **do**
    sample a set I_t of size M from $\mathcal{P}$
    for each client $i \in I_t$ **in parallel do**
      $q_t^i = \text{ClientUpdate}(i, x_t, \alpha, \gamma, \Theta)$
    $q_t = (1/M) \sum_{i \in I_t} q_t^i$
    $x_{t+1} = \text{ServerOpt}(x_t, q_t)$
  return x_{T+1}

Algorithm 2 LocalUpdate: ClientUpdate

ClientUpdate$(i, x, \alpha, \gamma, \Theta)$**:**
  $x_1 = x$
  for $k = 1, 2, \ldots, K(\Theta)$ **do**
    sample a set S_k of size B from $\mathcal{D}_i$
    $g_k = (1/B) \sum_{z \in S_k} \nabla_{x_k} \left(f_i(x_k; z) + \frac{\alpha}{2} \|x_k - x\|^2 \right)$
    $x_{k+1} = x_k - \gamma g_k$
  return $\sum_{k=1}^{K(\Theta)} \theta_k g_k$

LocalUpdate recovers many well-known algorithms for various choices Θ. For convenience, define

$$\Theta_K := (\underbrace{0, \ldots, 0}_{\text{K-1 times}}, 1), \quad \Theta_{1:K} := (\underbrace{1, \ldots, 1}_{\text{K times}}). \quad (5)$$

Special cases of LocalUpdate when ServerOpt is gradient descent are given in Table 1. For details on the relation between FedAvg and LocalUpdate, see Appendix A. By changing ServerOpt, we can recover methods such as FedAvgM (Hsu et al., 2019) (server gradient descent with momentum), and FedAdam (Reddi et al., 2020) (server Adam (Kingma and Ba, 2014)).

Table 1: Special cases of LocalUpdate when ServerOpt is gradient descent.

Algorithm	Θ	Conditions
Mini-batch SGD	Θ_1	$\lvert\mathcal{P}\rvert = 1, \alpha = 0, \gamma = 0$
Lookahead (Zhang et al., 2019)	$\Theta_{1:K}$	$\lvert\mathcal{P}\rvert = 1, \alpha = 0, \gamma > 0$
FedSGD (McMahan et al., 2017)	$\Theta_{1:K}$	$\alpha = 0, \gamma = 0$
FedAvg (McMahan et al., 2017), Reptile (Nichol et al., 2018)	$\Theta_{1:K}$	$\alpha = 0, \gamma > 0$
FedProx (Li et al., 2020a), MetaMinibatchProx (Zhou et al., 2019)	$\Theta_{1:K}$	$\alpha > 0, \gamma > 0$
FOMAML (Finn et al., 2017)	Θ_K	$\alpha = 0, \gamma > 0$
MAML (Finn et al., 2017)	Θ_{2K+1}	$\alpha = 0$, quadratics (Theorem 2)

3 Local Update Methods as First-Order Methods

LocalUpdate can vary drastically from first-order optimization methods on the empirical loss. Despite this, we will show that Algorithm 2 is equivalent in expectation to ServerOpt applied to a single *surrogate loss*. This surrogate loss is determined by the inputs α, γ and Θ to Algorithm 2. For each client $i \in \mathcal{I}$, we define its *distortion matrix* $Q_i(\alpha, \gamma, \Theta)$ as

$$Q_i(\alpha, \gamma, \Theta) := \sum_{k=1}^{K(\Theta)} \theta_k (I - \gamma(A_i + \alpha I))^{k-1}. \quad (6)$$

We define the surrogate loss function of client i as

$$\tilde{f}_i(x, \alpha, \gamma, \Theta) := \frac{1}{2} \|(Q_i(\alpha, \gamma, \Theta) A_i)^{1/2} (x - c_i)\|^2 \quad (7)$$

and the overall surrogate loss function as

$$\tilde{f}(x, \alpha, \gamma, \Theta) := \mathop{\mathbb{E}}_{i \sim \mathcal{P}} [\tilde{f}_i(x, \alpha, \gamma, \Theta)]. \quad (8)$$

When $\Theta = \Theta_1$, $Q_i(\alpha, \gamma, \Theta) = I$, in which case there is no distortion. In general, $Q_i(\alpha, \gamma, \Theta)$ can amplify the heterogeneity of the A_i. We derive the following theorem linking the surrogate losses to Algorithm 2.

Theorem 1. *For all $i \in \mathcal{P}$,*

$$\mathbb{E}[\text{ClientUpdate}(i, x, \alpha, \gamma, \Theta)] = \nabla \tilde{f}_i(x, \alpha, \gamma, \Theta).$$

For q_t as in Algorithm 1, Theorem 1 implies $\mathbb{E}[q_t] = \nabla \tilde{f}(x, \alpha, \gamma, \Theta)$. Thus, one round of LocalUpdate is equivalent in expectation to one step of ServerOpt on the surrogate loss $\tilde{f}(x, \alpha, \gamma, \Theta)$. As $\gamma \to 0$ or $K \to 1$, $\tilde{f}(x, \alpha, \gamma, \Theta) \to w(\Theta) f(x)$, so as γ gets smaller, the "pseudo-gradients" q_t more closely resemble stochastic gradients of the empirical loss function.

A version of Theorem 1 was shown for $\alpha = 0, \Theta = \Theta_2$ by Fallah et al. (2020b). We take this a step further and show that in certain settings, MAML is equivalent in expectation to ServerOpt on a surrogate loss.

MAML MAML with K local steps can be viewed as a modification of LocalUpdate. Algorithm 1 remains the same, and in Algorithm 2, each client executes K mini-batch SGD steps. However, the client's message to the server is different. Let $X_K^i(x)$ be the function that runs K steps of mini-batch SGD, starting from x, for fixed mini-batches $S_1, \ldots, S_K$ of size B drawn independently from $\mathcal{D}_i$. Define

$$m_K^i(x; z) = f(X_K^i(x); z), \; m_K^i(x) = \mathop{\mathbb{E}}_{z \sim \mathcal{D}_i} [m_K^i(x; z)].$$

Each client i sends a stochastic estimate of $\nabla m_K^i(x)$ to the server. The rest is identical to LocalUpdate; The server averages the client outputs and uses this as a gradient estimate for ServerOpt. While MAML is not a special case of LocalUpdate, we show that if the clients use gradient descent, MAML is equivalent in expectation to LocalUpdate with $\Theta = \Theta_{2K+1}$.

Theorem 2. *If $X_K^i(x)$ is the function that runs K steps of gradient descent on $\mathcal{D}_i$ with learning rate γ starting at x, then*

$$\nabla m_K^i(x) = \nabla_x \tilde{f}_i(x, 0, \gamma, \Theta_{2K+1}).$$

An analogous result holds if the clients perform proximal updates ($\alpha > 0$ in ClientUpdate). Thus, to understand LocalUpdate and MAML on quadratic models, it suffices to analyze the optimization dynamics of $\tilde{f}(x, \alpha, \gamma, \Theta)$. We use this viewpoint to study the convergence and accuracy of these methods.

4 Convergence and Accuracy of Local Update Methods

Comparing (2) and (8), we see that $\tilde{f}(x, \alpha, \gamma, \Theta)$ and $f(x)$ need not share critical points. Special cases of this fact were noted by Malinovsky et al. (2020), Fallah et al. (2020b), and Pathak and Wainwright (2020). We will show that this is not a failure of local update methods. Rather, by altering the loss function being optimized, LocalUpdate can greatly improve convergence, but to a less accurate point. More generally, the choice of α, γ and Θ dictates a trade-off between

convergence and accuracy. Intuitively, the larger γ and $K(\Theta)$ are, the faster LocalUpdate will converge, and the less accurate the resulting model may be.

To show this formally, we restrict to $Q_i(\alpha, \gamma, \Theta) \succ 0$, as then $\tilde{f}(x, \gamma, \Theta)$ is strongly convex with a unique minimizer. This is ensured by the following.

Lemma 1. *Suppose that $\gamma < (L + \alpha)^{-1}$. Then for all i, $Q_i(\alpha, \gamma, \Theta)$ is positive definite and $\tilde{f}(x, \alpha, \gamma, \Theta)$ is strongly convex.*

4.1 Condition Numbers

Under the conditions of Lemma 1, $\tilde{f}_i(x, \alpha, \gamma, \Theta)$ has a well-defined condition number which we bound.

Lemma 2. *Suppose $\gamma < (L + \alpha)^{-1}$. Define*

$$\kappa(\alpha, \gamma, \Theta) := \frac{\mathbb{E}_i[\lambda_{\max}(Q_i(\alpha, \gamma, \Theta)A_i]}{\mathbb{E}_i[\lambda_{\min}(Q_i(\alpha, \gamma, \Theta)A_i]}. \tag{9}$$

Then $\text{cond}(\tilde{f}) \leq \kappa(\alpha, \gamma, \Theta)$.

We wish to better understand (9) in cases of interest. We first consider $\Theta = \Theta_{1:K}$, as in FedAvg. Define

$$\phi(\lambda, \alpha, \gamma, K) := \sum_{k=1}^{K} (1 - \gamma(\lambda + \alpha))^{k-1}\lambda. \tag{10}$$

We now derive a bound on $\text{cond}(\tilde{f})$ for $\Theta = \Theta_{1:K}$.

Lemma 3. *If $\gamma < (L + \alpha)^{-1}$, $\tilde{f}(x, \alpha, \gamma, \Theta_{1:K})$ is $\phi(L, \alpha, \gamma, K)$-smooth, $\phi(\mu, \alpha, \gamma, K)$-strongly convex, and $\text{cond}(\tilde{f}) \leq \kappa(\alpha, \gamma, \Theta_K)$ where*

$$\kappa(\alpha, \gamma, \Theta_{1:K}) \leq \frac{\phi(L, \alpha, \gamma, K)}{\phi(\mu, \alpha, \gamma, K)}. \tag{11}$$

When $\gamma = 0, \alpha = 0$, we recover the condition number L/μ of the empirical loss f. We next consider $\Theta = \Theta_K$, as in MAML-style algorithms. Define

$$\psi(\lambda, \alpha, \gamma, K) := (1 - \gamma(\lambda + \alpha))^{K-1}\lambda. \tag{12}$$

We now derive a bound on $\text{cond}(\tilde{f})$ for $\Theta = \Theta_K$.

Lemma 4. *If $\gamma < (KL + \alpha)^{-1}$, $\tilde{f}(x, \alpha, \gamma, \Theta_K)$ is $\psi(L, \alpha, \gamma, K)$-smooth, $\psi(\mu, \alpha, \gamma, K)$-strongly convex, and $\text{cond}(\tilde{f}) \leq \kappa(\alpha, \gamma, \Theta_K)$ where*

$$\kappa(\alpha, \gamma, \Theta_K) \leq \left(\frac{1 - \gamma(L + \alpha)}{1 - \gamma(\mu + \alpha)}\right)^{K-1} \frac{L}{\mu}. \tag{13}$$

We show in Appendix B.3 that Lemmas 3 and 4 are tight. The extra condition that $\gamma < (KL + \alpha)^{-1}$ for $\Theta = \Theta_K$ is due to the fact that when $\gamma \geq (KL+\alpha)^{-1}$, $\text{cond}(\tilde{f}(x, \alpha, \gamma, \Theta_K))$ depends on intermediate eigenvalues of the A_i, and exhibits more nuanced behavior. We explore this further in Section 5.

As $K \to 1$ or $\gamma \to 0$, $\kappa(\alpha, \gamma, \Theta_K) \to L/\mu$, which bounds the condition number of the empirical loss f. If γ is not close to 0, we get an exponential reduction (in terms of K) of the condition number. While the analysis is not as clear for $\Theta_{1:K}$, one can show that $\kappa(\alpha, \gamma, \Theta_{1:K}) \leq L/\mu$, with equality if and only if $\alpha = 0$, and either $\gamma = 0$ or $K = 1$. Moreover, $\kappa(\alpha, \gamma, \Theta_{1:K})$ decreases as $K \to \infty$ or $\gamma \to (L+\alpha)^{-1}$. For both $\Theta_{1:K}$ and Θ_K, increasing α decreases κ.

Here we see the impact of local update methods on convergence: Popular methods such as FedAvg, FedProx, MAML, and Reptile reduce the condition number of the surrogate loss function they are actually optimizing. In the next section, we translate this into concrete convergence rates for LocalUpdate.

4.2 Convergence Rates

We now focus on a *deterministic* version of LocalUpdate in which all clients participate at each round and perform K steps of gradient descent. The server updates its model using $q_t = \mathbb{E}_{i\sim\mathcal{P}}[q_t^i]$, where q_t^i is the output of ClientUpdate for client i. In particular, $\mathcal{P}$ must be known to the server. In this case, Theorem 1 implies that $q_t = \nabla\tilde{f}(x_t, \alpha, \gamma, \Theta)$, so LocalUpdate is equivalent to applying ServerOpt to the true gradients of $\tilde{f}(x, \alpha, \gamma, \Theta)$.

We specialize to the setting where ServerOpt is gradient descent, with or without momentum (though our analysis can be directly extended to other optimizers). Thus, LocalUpdate is equivalent to gradient descent on $\tilde{f}(x, \alpha, \gamma, \Theta)$. Using the bound on $\text{cond}(\tilde{f})$ in Lemma 2, we can directly apply classical convergence theory gradient descent (Lessard et al. (2016, Proposition 1) give a useful summary) to derive convergence rates for LocalUpdate. Similar analyses can be done in the stochastic setting.

Theorem 3. *Suppose $\gamma < (L+\alpha)^{-1}$ and ServerOpt is gradient descent with Nesterov, heavy-ball, or no momentum. Then for some hyperparameter setting of ServerOpt, and ρ as in Table 2, the iterates $\{x_t\}_{t\geq 1}$ of LocalUpdate satisfy*

$$\|x_T - x^*(\alpha, \gamma, \Theta)\| \leq \rho^T \|x_0 - x^*(\alpha, \gamma, \Theta)\|. \tag{14}$$

Thus, (properly tuned) server momentum improves the convergence of LocalUpdate, giving theoretical grounding[1] to the improved convergence of FedAvgM shown by Hsu et al. (2019) and Reddi et al. (2020). Since ServerOpt does not change the surrogate loss,

[1] Yuan and Ma (2020) first showed that momentum can accelerate FedAvg, though they use a different momentum scheme with extra per-round communication.

Table 2: Convergence rates of LocalUpdate when ServerOpt is gradient descent (with or without momentum), and $\kappa = \kappa(\alpha, \gamma, \Theta)$ is as in (9).

Momentum	Rate
None	$\rho = \frac{\kappa-1}{\kappa+1}$
Nesterov	$\rho = 1 - \frac{2}{\sqrt{3\kappa+1}}$
Heavy-ball	$\rho = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$

this improvement in convergence does not degrade the accuracy of the learned model.

Given the bounds on $\text{cond}(\tilde{f})$ in (11) and (13), we obtain explicit convergence rates for FedAvg- and MAML-style algorithms as well. In particular, one can show that increasing γ or K decreases κ (and therefore ρ). We show in the next section that this comes at the expense of increasing the empirical loss.

4.3 Distance Between Global Minimizers

We now turn our attention towards the discrepancy between the surrogate loss $\tilde{f}$ and the empirical loss f. We assume $\gamma < (L+\alpha)^{-1}$. By Lemma 1, $\tilde{f}$ and f are strongly convex with global minimizers we denote by

$$x^*(\alpha, \gamma, \Theta) := \operatorname*{argmin}_x \tilde{f}(x, \alpha, \gamma, \Theta),$$

$$x^* := \operatorname*{argmin}_x f(x).$$

We are interested in $\|x^*(\alpha, \gamma, \Theta) - x^*\|$. While we focus on the setting where $\mathcal{P}$ is a discrete distribution over some finite $\mathcal{I}$, our analysis can be generalized to arbitrary probability spaces $(\mathcal{I}, \mathcal{F}, \mathcal{P})$. We derive the following bound.

Lemma 5. *Let* $b = \max_{i\in\mathcal{I}} \lambda_{\max}(Q_i(\alpha, \gamma, \Theta))$ *and* $a = \min_{i\in\mathcal{I}} \lambda_{\min}(Q_i(\alpha, \gamma, \Theta))$. *Then*

$$\|x^*(\alpha, \gamma, \Theta) - x^*\| \leq 8C \frac{\sqrt{b} - \sqrt{a}}{\sqrt{b} + \sqrt{a}}. \tag{15}$$

When $d = 1$, we can reduce the constant factor to $2C$, which we show is tight (see Appendix C.2). While we conjecture that this bound holds with a constant of $2C$ for all d, we leave this to future work.

Our proof technique for Lemma 5 may be of independent interest. We derive this result by first proving an analog of the Bhatia-Davis inequality (Bhatia and Davis, 2000) for mean absolute deviations of bounded random variables (Theorem 5 in Appendix C).

Let $\kappa_0 := L/\mu$. Specializing to $\Theta = \Theta_{1:K}$ or Θ_K, we derive a link between $\kappa(\alpha, \gamma, \Theta)$ in (11) and (13) and the distance between optimizers.

Lemma 6. *Suppose that either (I)* $\gamma < (L+\alpha)^{-1}$ *and* $\Theta = \Theta_{1:K}$ *or (II)* $\gamma < (KL+\alpha)^{-1}$ *and* $\Theta = \Theta_K$. *Then for all* $i \in \mathcal{P}$, $\text{cond}(Q_i(\alpha, \gamma, \Theta)) \leq \kappa_0 \kappa(\alpha, \gamma, \Theta)^{-1}$.

Combining this with Lemma 5, we get:

Theorem 4. *Under the same settings as Lemma 6,*

$$\|x^*(\alpha, \gamma, \Theta) - x^*\| \leq 8C \frac{\sqrt{\kappa_0} - \sqrt{\kappa(\alpha, \gamma, \Theta)}}{\sqrt{\kappa_0} + \sqrt{\kappa(\alpha, \gamma, \Theta)}}. \tag{16}$$

Applying Theorem 3, we bound the convergence of LocalUpdate to the empirical minimizer x^*.

Corollary 1. *Under the same settings as Theorem 3, for some hyperparameter setting of* ServerOpt, *the iterates* $\{x_t\}_{t\geq 1}$ *of* LocalUpdate *satisfy*

$$\|x_T - x^*\| \leq \rho^T \|x_0 - x^*(\alpha, \gamma, \Theta)\| + 8C \frac{\sqrt{\kappa_0} - \sqrt{\kappa}}{\sqrt{\kappa_0} + \sqrt{\kappa}}$$

where $\kappa = \kappa(\alpha, \gamma, \Theta)$ *is given in* (11) *and* (13), *and* ρ *is given in Table 2.*

Here we see the benefit of local update methods in *communication-limited* settings. When T is small and $\|x_0 - x^*\|$ is large, we can achieve better convergence by decreasing ρ and leaving the second term fixed. In such settings, FedAvg can arrive at a neighborhood of a critical point in fewer communication rounds than mini-batch SGD, but may not ever actually reach the critical point. If $\|x_0 - x^*\|$ is small, we may be better served by using mini-batch SGD instead.

5 Comparing Local Update Methods

Comparing optimization algorithms is a fundamental theoretical effort. Many past works compare local update methods based on their convergence to critical points of the empirical loss. By Theorem 1, LocalUpdate is only guaranteed to converge to critical points of f if $\gamma = 0$ or $K(\Theta) = 1$. Thus, existing analyses ignore many useful cases of LocalUpdate.

To remedy this, we compare local update algorithms on the basis of both convergence and accuracy. Instead of fixing γ and Θ, we analyze LocalUpdate as γ and $K(\Theta)$ vary. To do so, we use our theory from Section 4. Given α, γ and Θ, we define the **convergence rate** $\rho(\alpha, \gamma, \Theta)$ as the infimum over all ρ such that for all $T \geq 1$, (14) holds. Values of ρ when ServerOpt is gradient descent are given in Table 2. For $\Theta = \Theta_{1:K}$ or Θ_K, we define the **suboptimality** $\Delta(\alpha, \gamma, \Theta)$ by

$$\Delta(\alpha, \gamma, \Theta) := \frac{\sqrt{\kappa_0} - \sqrt{\kappa(\alpha, \gamma, \Theta)}}{\sqrt{\kappa_0} + \sqrt{\kappa(\alpha, \gamma, \Theta}}. \tag{17}$$

By Theorem 4, this captures the asymptotic worst-case suboptimality of LocalUpdate.

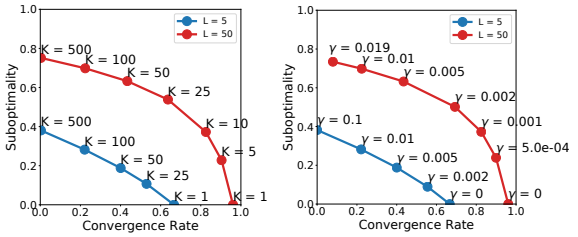

Figure 1: Pareto frontiers for $\mu = 1, \alpha = 0, \Theta = \Theta_{1:K}$ and $L \in \{5, 50\}$. We fix $\gamma = 0.01$ and vary K (left), and fix $K = 100$ and vary γ (right).

Note that $\rho, \Delta \in [0, 1]$. Therefore, fixing μ, L and ServerOpt, we obtain a **Pareto frontier** in $[0, 1]^2$ by plotting (ρ, Δ) for various γ and $K(\Theta)$. This curve represents the worst-case convergence/accuracy trade-off of a class of local update methods. We generally want the curve to be as close to $(0, 0)$ as possible.

For example, in Figure 1 we let ServerOpt be gradient descent and set $\alpha = 0, \Theta = \Theta_{1:K}$. We plot (ρ, Δ) as we vary K and fix γ, and vice-versa. When $L = 5$, we obtain nearly identical curves. The curves for $L = 50$ are similar, except that when we fix K and vary γ, we do not reach $\rho \approx 0$. While γ and K have similar impacts on convergence-accuracy trade-offs, varying K leads a larger set of attainable (ρ, Δ). Formally, this is because in (11), $\lim_{\gamma \to L^{-1}}(\kappa) \neq 0$. Intuitively, $K \to \infty$ recovers one-shot averaging while $\gamma \to L^{-1}$ does not. Notably, the convergence-accuracy trade-off becomes closer to a linear trade-off as L/μ decreases.

The Pareto frontiers contain more information than just the convergence rate to a critical point (the curve's intersection with the x-axis). This information is useful in communication-limited regimes, where we wish to minimize the number of rounds needed to attain a given accuracy. The curves also help visualize various hyperparameter settings of an algorithms simultaneously. To illustrate this, we use the Pareto frontiers to derive novel findings regarding server momentum, proximal client updates, and qualitative differences between FedAvg and MAML. The results are all given below. For more results, see Appendix D.

Impact of Server Momentum As shown empirically by Hsu et al. (2019) and as reflected in Table 2, server momentum can improve convergence. To understand this, in Figure 2 we compare Pareto frontiers where $\Theta = \Theta_{1:K}$ and ServerOpt is gradient descent with various types of momentum (Nesterov, heavy-ball, or no momentum). We see a strict ordering of the server optimization methods. Heavy-ball momentum is better than Nesterov momentum, which is better than no momentum.

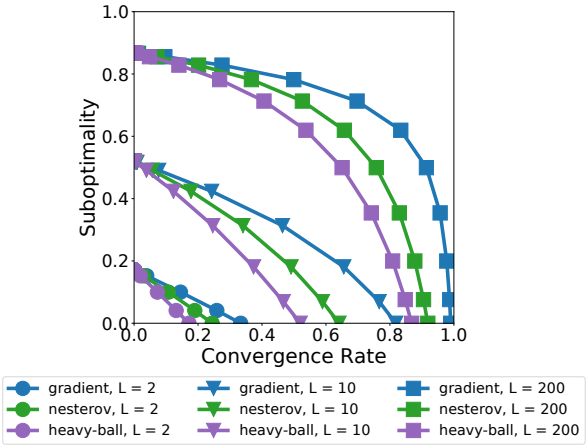

Figure 2: Pareto frontiers for $\mu = 1$, varying L, and where ServerOpt is gradient descent with different types of momentum. We let $\alpha = 0, \gamma = (2L)^{-1}$ and $\Theta = \Theta_{1:K}$ for varying $K \in [1, 10^6]$.

One important finding is that the benefit of momentum is more pronounced as L/μ increases. On the other hand, the benefit of server momentum diminishes for sufficiently large K: In Figure 4, the various types of momentum lead to similar suboptimality when the convergence rate is close to 0. Intuitively, as $K \to \infty$, we recover one-shot averaging, which converges in a single communication round with or without momentum.

Another intriguing observation: The Pareto frontiers for heavy-ball momentum appear to be symmetric about the line $\rho = \Delta$. We conjecture this is true for any μ, L. While we believe that this may be provable by careful algebraic manipulation of our results above, ideally a proof would explain the root causes of this symmetry. Thus, we leave a proof to future work.

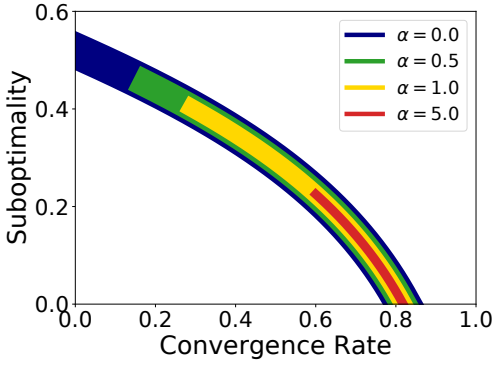

Figure 3: Pareto frontiers for $\mu = 1, L = 10, \Theta = \Theta_{1:K}$. We set $\gamma = 1/2(L + \alpha)^{-1}$, ServerOpt as gradient descent, vary α and K over $\{0, 0.5, 1.0, 5.0\}$ and $[1, 10^6]$.

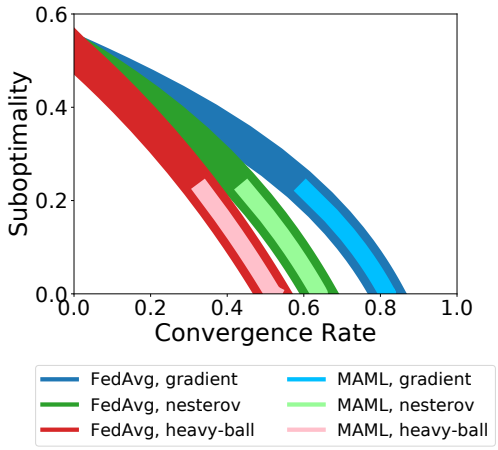

Figure 4: Pareto frontiers for $\mu = 1$, $L = 10$, $\alpha = 0$, $\Theta = \Theta_{1:K}$ (FedAvg) and Θ_K (MAML) for varying $K \in [0, 10^6]$. ServerOpt is gradient descent, with Nesterov, heavy-ball, or no momentum. For $\Theta_{1:K}$, we use $\gamma = 0.001$, and for Θ_K, we use $\gamma = (2LK)^{-1}$.

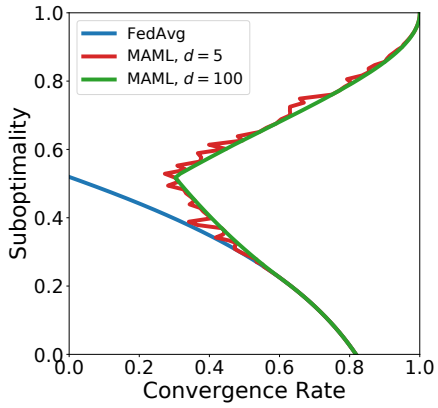

Figure 5: Simulated Pareto frontiers for $\mu = 1$, $L = 10$, $\alpha = 0$, $\gamma = 0.001$, $\Theta = \Theta_K$ (MAML). ServerOpt is gradient descent. We randomly sample $A \in \mathbb{R}^{d \times d}$ with $\mu \preceq A \preceq L$, and compute (ρ, Δ) for various $K \in [1, 10^6]$ and $d \in \{5, 100\}$. We compare to the Pareto frontier for $\Theta_{1:K}$ (FedAvg).

Proximal Client Updates So far we have only considered $\alpha = 0$. One might posit that as α varies, the Pareto frontier moves closer to the origin. This appears to not be the case. In all settings we examined, changing α did not bring the Pareto frontier closer to 0. Instead, the frontier for $\alpha > 0$ was simply a subset of the frontier for $\alpha = 0$.

To illustrate this, we plot Pareto frontiers for varying α in Figure 3. As α increases, the frontier becomes a smaller subset of the frontier for $\alpha = 0$. Thus, proximal client updates may not enable faster convergence. Rather, their benefit may be in guarding against setting γ too small or K too large. Figure 3 shows that FedAvg can always attain the same (ρ, Δ) as FedProx, but it may require different hyperparameters. The reverse is not true, as FedProx cannot recover one-shot averaging. Our findings are consistent with work by Wang et al. (2020), who show that FedProx can reduce the "objective inconsistency" of FedAvg, at the expense of increasing convergence time.

Comparing MAML to FedAvg We now turn our attention to comparing FedAvg-style algorithms ($\Theta = \Theta_{1:K}$) to MAML-style algorithms ($\Theta = \Theta_K$). We plot Pareto frontiers for the ρ, Δ guaranteed by Theorems 3 and 4. The results are in Figure 4.

For each ServerOpt, the MAML frontier is a subset of the FedAvg frontier. Recall that in Theorem 3, we require $\gamma < (L + \alpha)^{-1}$ for FedAvg, but $\gamma < (KL + \alpha)^{-1}$ for MAML. In Figure 4 this causes the frontier for Θ_K to be more restrictive than for $\Theta_{1:K}$. However, it is still notable that these two fundamentally different methods, attain the same frontier when ρ is large.

By Lemma 1, ρ and Δ are still well-defined for Θ_K when $(KL + \alpha)^{-1} \leq \gamma < (L + \alpha)^{-1}$. To understand what happens in this regime, we generate random symmetric $A \in \mathbb{R}^{d \times d}$ satisfying $\text{cond}(A) = L/\mu$ and compute $Q(\alpha, \gamma, \Theta_K)$ as in (6). We then compute κ via (9), and plug this into Table 2 and (17) to get ρ, Δ. This gives us a simulated Pareto frontier for Θ_K, which we compare to $\Theta_{1:K}$ in Figure 5. For details and additional experiments, see Appendix D.1.

The Pareto frontiers are identical for small K (mirroring Figure 4), but diverge when $\gamma \geq (KL + \alpha)^{-1}$. The frontier for MAML then moves further from 0. Intuitively, FedAvg tries to learn a global model, while MAML tries to learn a model that adapts quickly to new tasks (Finn et al., 2017); MAML need not minimize (ρ, Δ). The MAML frontier is noisy for $d = 5$ (as ρ, Δ depend on random eigenvalues of A), but stabilizes for $d = 100$. While we posit that this reflects a semi-circle law for eigenvalues of random matrices (Alon et al., 2002), we leave an analysis to future work.

One final observation that highlights the similarities and differences of FedAvg- and MAML-style methods: In Figure 4, the curve for MAML when $d = 100$ has a clear cusp. This seems to occur at the same suboptimality (ie. y-value) as the intersection of the FedAvg curve with the y-axis. In other words, the behavior of MAML diverges substantially from FedAvg, but only after it reaches the same suboptimality as FedAvg for $K \to \infty$ (which corresponds to one-shot averaging). We are unsure why the suboptimality of one-shot averaging corresponds to a cuspidal operating point of MAML, but this observation highlights significant nuance in the behavior of these methods.

6 Limitations and Discussion

Our convergence-accuracy framework and the resulting Pareto frontiers can be useful tools in understanding how algorithmic choices impact local update methods. The obvious limitation is that they only apply to quadratic models. While this is restrictive, we show empirically in Appendix F that even for non-convex functions, the client learning rate governs a convergence-accuracy trade-off for FedAvg.

Our framework may also be useful in identifying important phenomena underlying LocalUpdate, even in non-quadratic settings. To demonstrate this, we show that many of the observations in Section 5 hold in non-convex settings. We train a CNN on the FEMNIST dataset (Caldas et al., 2018) using LocalUpdate where $\Theta = \Theta_{1:50}$. We tune client and server learning rates. See Appendix E for full details. In Figure 6, we illustrate how server momentum and α change convergence. Our results match the Pareto frontiers in Figures 2 and 3: Server momentum improves convergence, while α has little to no effect, provided we tune learning rates.

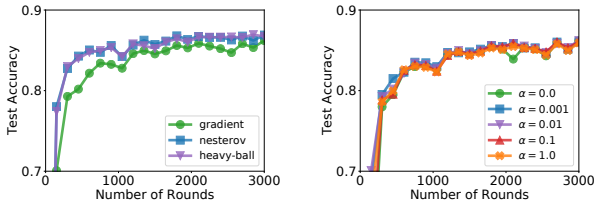

Figure 6: Test accuracy of LocalUpdate with $\Theta = \Theta_{1:50}$ on FEMNIST with tuned learning rates. (Left) Varying types of server momentum, $\alpha = 0$. (Right) No momentum and varying α.

This brief example illustrates that our framework can identify crucial facets of local update methods. While our framework may not capture all relevant details of such methods, we believe it greatly simplifies their analysis, comparison, and design. In the future, we hope to extend this framework to more general loss functions. Other important extensions include stochastic settings with partial client participation, as well as trade-offs between convergence and post-adaptation accuracy of local update methods.

References

Noga Alon, Michael Krivelevich, and Van H Vu. On the concentration of eigenvalues of random symmetric matrices. *Israel Journal of Mathematics*, 131(1): 259–267, 2002.

Maria-Florina Balcan, Mikhail Khodak, and Ameet Talwalkar. Provable guarantees for gradient-based meta-learning. In *International Conference on Machine Learning*, pages 424–433. PMLR, 2019.

Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-SGD: Distributed SGD with quantization, sparsification and local computations. In *Advances in Neural Information Processing Systems*, pages 14668–14679, 2019.

Rajendra Bhatia and Chandler Davis. A better bound on the variance. *The American Mathematical Monthly*, 107(4):353–357, 2000.

Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1082–1092. PMLR, 26–28 Aug 2020a. URL http://proceedings.mlr.press/v108/fallah20a.html.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020b.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR, 2017.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

Andrew Hard, Kurt Partridge, Cameron Nguyen, Niranjan Subrahmanya, Aishanee Shah, Pai Zhu, Ignacio Lopez Moreno, and Rajiv Mathews. Training keyword spotting models on non-IID data with federated learning. *arXiv preprint arXiv:2005.10406*, 2020.

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B Gibbons. The non-IID data quagmire of decentralized machine learning. *arXiv preprint arXiv:1910.00189*, 2019.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Alex Ingerman and Krzys Ostrowski. Introducing TensorFlow Federated, 2019. URL https://medium.com/tensorflow/introducing-tensorflow-federated-a4147aa20041.

Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, pages 5915–5926, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *arXiv preprint arXiv:1908.07873*, 2019.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems 2020*, pages 429–450, 2020a.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=ByexElSYDr.

Grigory Malinovsky, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local SGD to local fixed point methods for federated learning. *arXiv preprint arXiv:2004.01442*, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, pages 1273–1282, 2017.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Reese Pathak and Martin J Wainwright. FedSplit: An algorithmic framework for fast federated optimization. *arXiv preprint arXiv:2005.05238*, 2020.

Tiberiu Popoviciu. Sur les équations algébriques ayant toutes leurs racines réelles. *Mathematica*, 9:129–145, 1935.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Sebastian U. Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1g2JnRcFX.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*, 2020.

Blake Woodworth, Kumar Kshitij Patel, Sebastian U Stich, Zhen Dai, Brian Bullins, H Brendan McMahan, Ohad Shamir, and Nathan Srebro. Is local SGD better than minibatch SGD? *arXiv preprint arXiv:2002.07839*, 2020.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

Cong Xie, Oluwasanmi Koyejo, Indranil Gupta, and Haibin Lin. Local AdaAlter: Communication-efficient stochastic gradient descent with adaptive learning rates. *arXiv preprint arXiv:1911.09030*, 2019.

Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. *Advances in Neural Information Processing Systems*, 33, 2020.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps for-

ward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9593–9604, 2019.

Pan Zhou, Xiaotong Yuan, Huan Xu, Shuicheng Yan, and Jiashi Feng. Efficient meta learning via minibatch proximal update. In *Advances in Neural Information Processing Systems*, pages 1534–1544, 2019.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010.