We would like to thank the reviewers for their helpful feedback, including the minor concerns that will be clarified but which we do not have space to discuss here.

Significance of the graph reformulation: The augmented graph allows to cast the stochastic decentralized problem as a batch decentralized problem with a more complex structure, captured by the matrix A. In our case, the augmented graph contains all the intuition behind the dual formulation (what should the consensus constraints be). More generally, we believe that it is a powerful way of reasoning to obtain finite sum algorithms from decentralized algorithms that work with subgraphs of communication. Algorithms and rates can then directly be obtained by studying the Laplacian of the augmented graph, as done in Appendix D. Yet, it is only needed to obtain the dual formulation.

Extension of past work: We mainly refer to the extension of the APCG algorithm to arbitrary sampling and to strong convexity in subspaces only. This extension is very important for our work because it allows us to choose different probabilities for communications and for local computations.

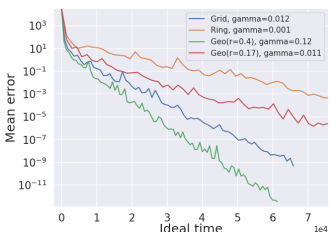
How do we obtain $W_{k\ell}\Sigma^{-1}y_t$: This comes from the coordinate update $A\nabla_{k\ell}q_A(y_t) = Ae_{k\ell}e_{k\ell}^TA\Sigma^{-1}y_t = \mu_{k\ell}^2W_{k\ell}\Sigma^{-1}y_t$. The $\mu_{k\ell}$ are indeed related to the communication matrix since the Laplacian matrix of the communication graph is $L = \sum_{k\ell}\mu_{k\ell}^2W_{k\ell}$. There is a typo and the $\mu_{k\ell}^2$ of line 188 should not appear.

Hyper parameters: The only degrees of freedom of ADFS are $p_{k\ell}$ and $\mu_{k\ell}$. The other parameters (such as $R_{k\ell}$, ρ, η, σ_A) directly depend on them. For **communication edges**, choosing values of μ amounts to choosing a gossip matrix, and choosing $p_{k\ell}$ amounts to tuning how frequently edge (k,ℓ) is sampled. Therefore, choosing uniform p and μ (as in the experiments) is efficient as long as the graph is not too heterogeneous. For graphs that have non-regular topologies, choosing μ better than uniform can be challenging but this is also the case for DSBA and MSDA which consider a fixed gossip matrix. Besides, MSDA and DSBA are synchronous algorithms that enforce uniform probabilities for all edges (all edges are activated at each step). Note that we would like to choose $\mu_{k\ell}^2 = p_{k\ell}^2/(R_{k\ell}[\sigma_k^{-1} + \sigma_\ell^{-1}])$ but this is not possible because $R_{k\ell}$ depends on $\mu_{k\ell}$. For **computation edges**, the theory tells us how to set $p_{k\ell}$ and $\mu_{k\ell}$ using an importance sampling scheme. This only requires knowing σ_k, the strong convexity of the problem, $L_{k\ell}$, the smoothness of individual training examples, and γ, the eigengap of the gossip matrix, which are standard constants of decentralized optimization problems. In the end, ADFS can be used with parameters given by theory (as given by Theorem 3), **without any extra tuning**. ADFS may perform better with other choices of p and μ in some situations but its main competitors are either forced to use uniform values or require the same tuning.

Probability $p_{k\ell}$: The choice of $p_{k\ell}$ impacts both the convergence rate (precisely captured by Inequality (6)) and the average time per iteration (cruder bound of Theorem 2). Contrary to MSDA, the communication $p_{k\ell}$ can be tuned to adapt to the topology of the graph, delays and local condition numbers. Although the optimal choice is only clear for regular graphs, heuristics can be designed using Inequality (6). For example, the communication probabilities can be chosen as inversely proportional to the edge delay or to the degree of their incident nodes, but this choice is highly dependent on the setting. Computing probabilities are chosen optimally in the homogeneous setting but could be set differently as well to reduce waiting time, e.g., to reduce the computational burden of a busy node.

Datasets used for experiments: We favored large networks and datasets with many samples over high-dimensional datasets for our experiments. Yet, we believe that the scale of the experiments is already quite significant. As a comparison, experiments in the MSDA paper use synthetic datasets of dimension 10 and experiments in the DSBA paper use a network of size 10. We use 100 computing nodes and 10^6 samples in total, which is significantly larger. Finally, serial APCG performs well on RCV1 and ADFS is a decentralized version of APCG, so we expect it to have good performances on RCV1 as well. We will add a full set of experiments on RCV1 to a revised version of the paper.

Comparison with Point-SAGA: As written in the introduction, Point-SAGA is a serial algorithm indeed and we apologize if this was not clear enough. Yet, we thought that it was interesting to have it as a competitor because it beats many distributed algorithms when the number of machines is not too big, as shown in Figure 3 (a). Our point was that ADFS compares nicely with optimal single machine algorithms on one machine (which is not the case of many other algorithms) while being able to take advantage of using more machines (Point-SAGA is only designed for one machine, as Reviewer 2 pointed out). We will make the distinction clearer by adding "serial" to the legend.



(a) ADFS on several graphs

Influence of the network topology: As long as all nodes have roughly the same probability of starting an update (to avoid waiting times), the topology of the network mainly influences the iteration complexity of ADFS through the constant γ. Figure 1a shows experiments with 100 nodes and 500 samples per node on the covtype dataset. For the geographic graph, each node is randomly placed in the unit square and nodes that are at distance less than r are neighbours. As expected, the convergence rate improves with γ. Yet, ADFS is slower on $Geo(0.17)$ than on the grid because nodes have uneven degrees so waiting time increases (because all edges have equal probabilities). A non-uniform probability distribution could speed-up convergence for $Geo(0.17)$.