Simulation Update:

- **Environment:** Our experiments are performed over a machine with 27 cores (Intel Xeon Processor E5-2682 2.5GHz) in Python 3.7. Each parallel node is an independent process/core and inter-process communication uses MPI4PY.
- **per round communication time vs per round computation time**: The exact time depends on the number of nodes/processes and variable dimensions. In the experiment of Sec 4.1 , each computation round takes $0.3ms$ and each communication round takes $43.7ms$. (Communication is 110 times more expensive in this case.)
- **Large scale real data set and more baselines:** We further perform the multi-class (10 classes) classification task over MNIST data set, which contains 60000 training images and each image can be considered as a $784 + 1$ dimensional feature vector. Since the number of classes is 10, the classification is a convex optimization with a 7850 dimensional variable. Besides our method, RPDBUS ADMM, and DCS, we further test the deterministic ADMM and the stochastic ADMM in Pu&Nedic 18 (suggested by Rev5). We partition the training set into 4 disjoint subsets and solve the multi-class classification problem with 4 parallel processes. The wall-clock time (including both computation and communication) to converge to the optimal with $\|\mathbf{x}_i - \mathbf{x}_j\|_\infty \leq 10^{-4}, \forall i, j$ for each method is: our method (28.49sec), PRDBUS (1837sec), DCS (684sec), deterministic ADMM (12hour+), Pu&Nedic (3591sec). Note that our method is significantly faster than others when measured by wall-clock time.

R2Q1: Elaborate more and discuss tolerance on failure of communication

A: Our method is robust to failure of communication. If communication fails, we can skip (4-5) and let each local node continue to run its sub-procedure STO-LOCAL for one more time. Mathematically, this is equivalent to a normal Algorithm 1 implementation where one particular STO-LOCAL step runs more iterations. Our convergence analysis only requires a minimum number of iterations is executed in each STO-LOCAL sub-procedure. So the convergence is guaranteed by our theory. Both theoretical elaboration and extra experiment results will be reported in the final version.

R2Q2: Decomposable property and L_{21} regularization.

A: This paper assumes the original problem has been **reformulated** into (1), which has a decomposable structure. For problems with L_{21} regularization, the applicability of our method depends on whether they can be reformulated into (1). For example, consider a robust L_{21} feature selection given by $\min_{\mathbf{W}} \|\mathbf{W}^T\mathbf{X} - \mathbf{Y}\|_{2,1} + \gamma\|\mathbf{W}\|_{2,1}$. It can be reformulated as $\min_{\mathbf{W},\mathbf{V}} \|\mathbf{V}\|_{2,1} + \gamma\|\mathbf{W}\|_{2,1} \;\; s.t. \;\; \mathbf{W}^T\mathbf{X} - \mathbf{Y} - \mathbf{V} = \mathbf{0}$. Since L_{21} norm is separable w.r.t. each row and linear constraints are separable w.r.t. each entry, it is decomposable w.r.t. each row of $\mathbf{W}$ and $\mathbf{V}$ and can be solved in a distributed way with our method.

R3Q1: Strong duality in Assumption 1

A: Assumption 1 is mild for convex programs with linear constraints. For problems with linear constraints, Proposition 6.4.2 in "D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, Convex Analysis and Optimization." ensures Assumption 1 as long as the feasible set is non-empty and the domain of the objective function satisfies any of the following 3 conditions: (1) contains the feasible set (2) open or (3) can be convexly extended to open sets. In particular, all linear programs with non-empty feasible sets satisfy Assumption 1.

R3Q2: stochastic objective function and related papers

A: The stochastic objective fun in Sec 4.1 is a pure stochastic function where the randomness is $\mathbf{c}_i$. The stochastic fun in Sec 4.2 is a finite sum that is expectation involving uniform distribution of the samples. Stochastic opt methods for Sec 4.2 allow us to evaluate a single sample rather than all samples for each iteration and yield low complexity. All your suggested papers on ADMM are discussed and cited in the revision.

R5: dependence on network topology and references on "local averaging" methods.

A: Yes, the dependence on network topology is hidden in $\|\mathbf{A}\|$. By our Remark 3, if we choose ρ to balance the dependence, both objective and constraint violations linearly depends on $\|\mathbf{A}\|$.

Compared with Nedic et al. 2018, Scaman et al. 17, Uribe et al. 17, and Pu&Nedic 18, all of which use a doubly stochastic or symmetric PSD matrix for local averaging, our ADMM method has the following advantages:

- Our inter-node communication pattern is more flexible and is not restricted to a (symmetric) pattern such as the (doubly) stochastic or symmetric PSD matrix. Of course, we can choose $\mathbf{A} = \mathbf{I} - \mathbf{W}$ where $\mathbf{W}$ is a stochastic matrix used in your suggested works since it ensures the consensus of local solutions. However, in general, we can use any $\mathbf{A}$ to ensure consistence as long as Null$\{\mathbf{A}\}$ =Span$\{\mathbf{1}\}$.
- While the dynamics of ADMM is different from mixing (local averaging) based method, our Theorem 1 and Remark 3 suggest our method can have better dependence on network topology. Our convergence only depends on $\|\mathbf{A}\|$. By choosing $\mathbf{A} = \mathbf{I} - \mathbf{W}$, we know $\|\mathbf{A}\| \leq 2$. The convergence in suggested works (using a doubly stochastic or a symmetric PSD $\mathbf{W}$ for mixing) further depends on $1/(1 - \{|\lambda_2(\mathbf{W})|, |\lambda_N(\mathbf{W})|\})$ or eigengap $\lambda_1(\mathbf{W})/\lambda_{N-1}(\mathbf{W})$, which can be much larger than constant 2 if some eigenvalues are extreme.

Nevertheless, the above suggested papers are related and complement ADMM methods. They are discussed and cited in the revision.