We thank all reviewers for their time and effort in reviewing our paper.

– Reviewer 1 –

We set up experiments on PyTorch with ResNet18 (He et al., 2016) on CIFAR10 (Krizhevsky, 2009). The model size is about 44MB. We use 50k training samples and 10k evaluation samples. For direct comparison, no data augmentation is used. The batch size is 128. The learning rate starts from 0.1, and is divided by 10 at 150 and 250 epochs. We set $b_x = 8$, $b = 4$ for low-precision algorithms. The sparsity budget $\varphi_t = ||\alpha_t||_1/||\alpha_t||_\infty$ and $\theta_s = 2/(s+2)$, $\eta_s = lr/\theta_s$ (parameters in Acc-AsyLPG). In Figure 1, we plot the training loss and test accuracy w.r.t. epochs, and provide the total transmitted bits until the training loss first gets below 0.17. It shows that our algorithms achieve similar accuracy and effectively reduce the communication cost compared to benchmarks.

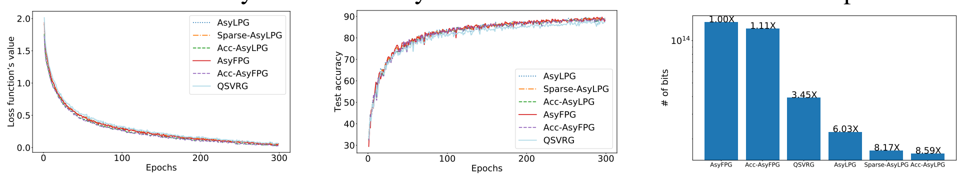

Figure 1: Evaluations on CIFAR10: training loss (1st column), test accuracy (2nd column) and total number of transmitted bits.

– Reviewer 2 –

(Running time) The statistics of running time in Figure $3(b)$ in our paper are obtained by averaging results of 5 runs in order to make the evaluations accurate. In Figure 2 here, we provide the total running time of logistic regression on rcv1 and a 3-layer fully connected neural network on MNIST. The experimental settings are the same as Section 5. The statistics in both graphs are recorded until the training loss first gets below 0.5. The results show that our algorithms can effectively reduce the total running time.

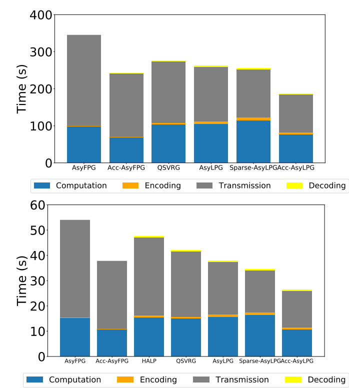

Figure 2: Decomposition of time consumption. Top: rcv1. Bottom: MNIST.

(Scalability) We present the running time on MNIST using 4, 8, 12 workers in Figure 3 here. The experimental settings are the same as Section 5.2. Each bar represents the total running time which is decomposed into communication (top, light, include transmission, encoding and decoding) and computation (bottom, dark), and is recorded until the training loss first gets below 0.1. The results show that algorithm speedup increases in the number of workers. More evaluations of training ResNet18 (model size 44MB) on CIFAR10 are shown in Figure 1 above. We will release our code on GitHub in the final version.

– Reviewer 3 –

(Comparison with existing results) (Bernstein et al., 2018) studies bi-direction 1-bit compression between master and workers. In their case, the master and workers exchange quantized gradients, whereas in our case, the master receives quantized gradients from workers and sends quantized model vectors to them. This difference leads to a very different analysis. The key novel components of our work, compared to existing results (including signSGD), include the following. (i) We propose the new double quantization scheme (DB). The gradient quantizer, though unbiased, is nontrivial to analyze, because it is evaluated on quantized model vectors. Since the function f is nonlinear, the stochastic gradients are biased. As a result, our algorithms cannot be analyzed with arguments used in full-precision distributed SGD analysis, and require new proofs. (ii) We further integrate sparsification and momentum into DB, and establish convergence rates under asynchrony. We will be sure to include more related references in the final version, e.g., (Wang et al., 2017), (Jiang & Agrawal, 2018), (Chen et al., 2018) and (Tang et al., 2019).

(Quantizer) Note that with our selections of δ_x, δ_{α_t} and δ_{β_t}, the unquantized coordinates of $x_{D(t)}$, α_t, and β_t all lie in the convex hull of the corresponding domains $\mathrm{dom}(\delta, b)$. In this case, the quantizer is unbiased. Such a quantizer is equivalent to that in QSVRG (Alistarh, 2018; Yu et al., AISTATS, 2019: Section 4.1) (also see Lemma 1 in Supplementary for details). Note that we can also adopt other biased **model quantizer** such as clipping, as long as the precision loss satisfies Eq. (2) or (4). Similar results can be proven with minor modifications of our analysis. Also note that although a 1-to-N broadcast is cheaper than N 1-to-1 unicasts, broadcasting a quantized vector is still much more communication efficient than broadcasting a full-precision vector.

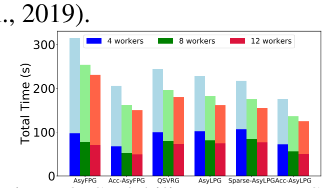

Figure 3: Scalability test on MNIST.

(Scalability) See Fig. 1,2,3 here for more evaluations on scalability and other datasets/models.

(Accuracy of model quantizer) μ is a hyperparameter to control the precision loss. When μ is fixed, we choose b_x to satisfy Eq. (2). In Figure 4 here, we set $\mu = 0.5$ and study how the accuracy of model quantizer improves with iterations when running AsyLPG on MNIST. We see that the quantization error diminishes. Thus, the number of transmitted bits increases as the number of iteration grows. Table 1 in manuscript records the total number of bits for reaching the desired accuracy, which validates the communication efficiency of our algorithms compared to benchmarks. Moreover, Table 1 evaluates the total transmitted bits under different μ for attaining the same accuracy.

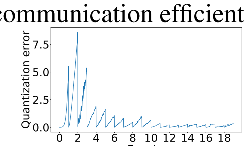

Figure 4: The accuracy of model quantizer.