GradiVeQ: Vector Quantization for Bandwidth-Efficient Gradient Aggregation in Distributed CNN Training

M. Yu, Z. Lin, K. Narra, S. Li, Y. Li, N. S. Kim, A. Schwing, M. Annavaram, S. Avestimehr

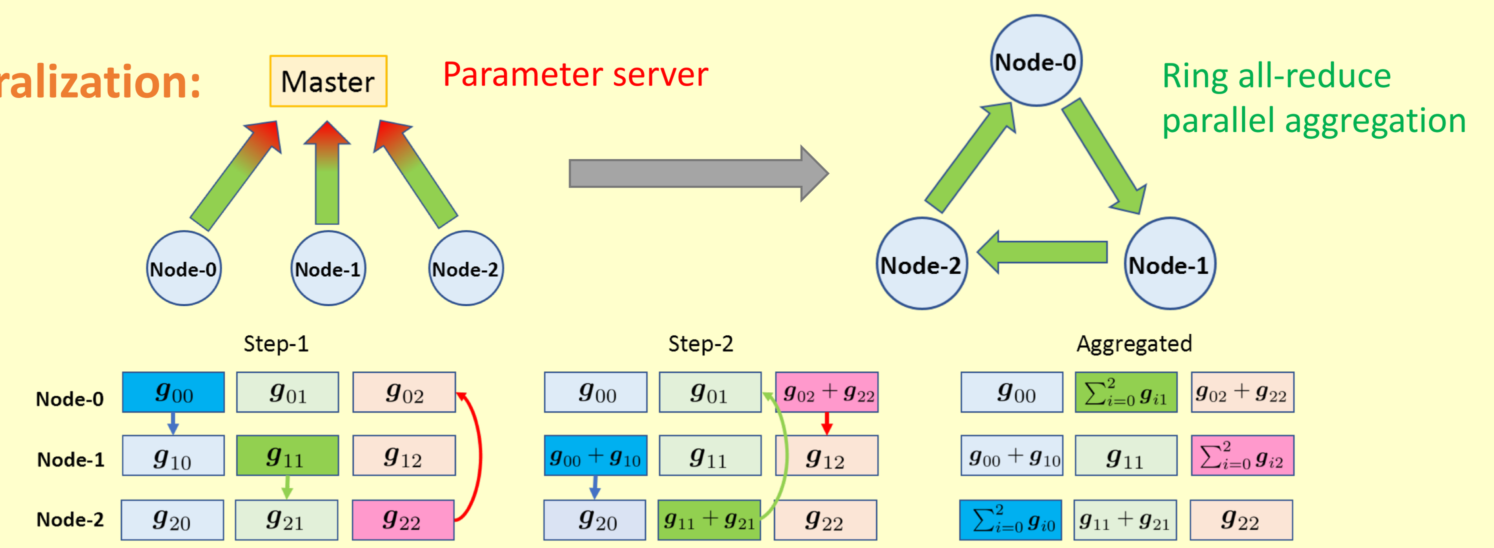
University of Southern California

University of Illinois at Urbana–Champaign

Why Linear Quantization?

Mitigating the communication bottleneck in distributed CNN training

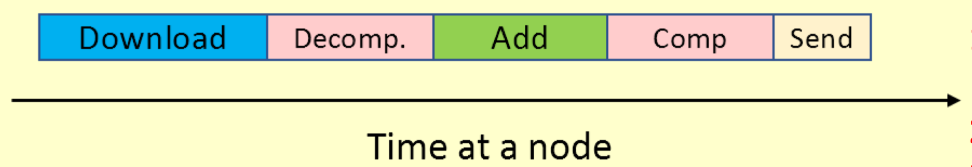
Decentralization:



Quantization: sacrifice precision for bandwidth

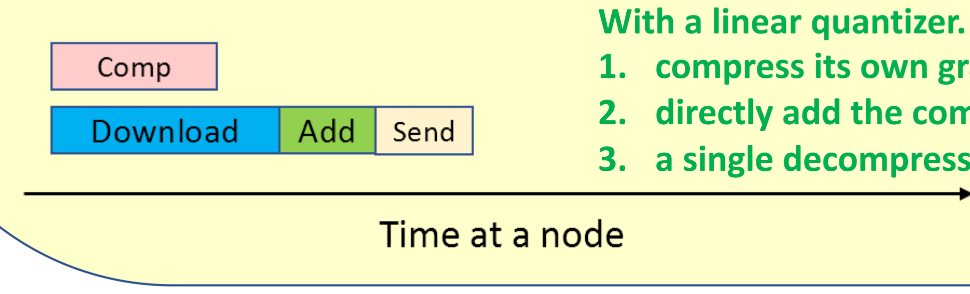
- Limited to non-linear scalar quantizer [1,2]

Only *linear quantizer* can be hidden behind parallel aggregation!



With a non-linear quantizer, each node must:

1. decompress the gradient vector it has received;
2. add its own gradient vector to it;
3. compress the result



With a linear quantizer. each node only needs to:

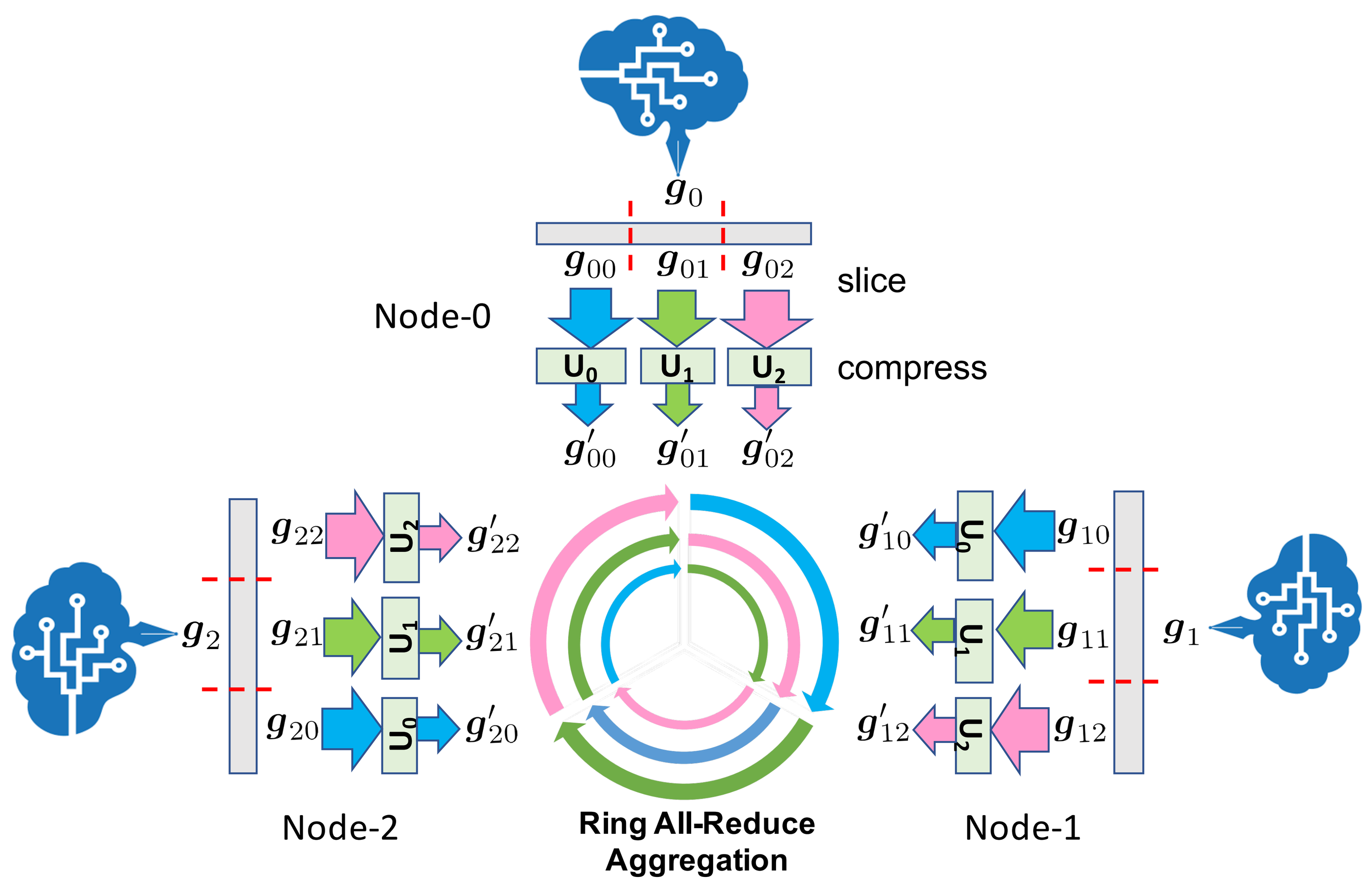
1. compress its own gradient vector *WHILE* downloading;
2. directly add the compressed gradient vectors;
3. a single decompression after aggregation finishes

GradiVeQ

The First Linear Vector Quantizer

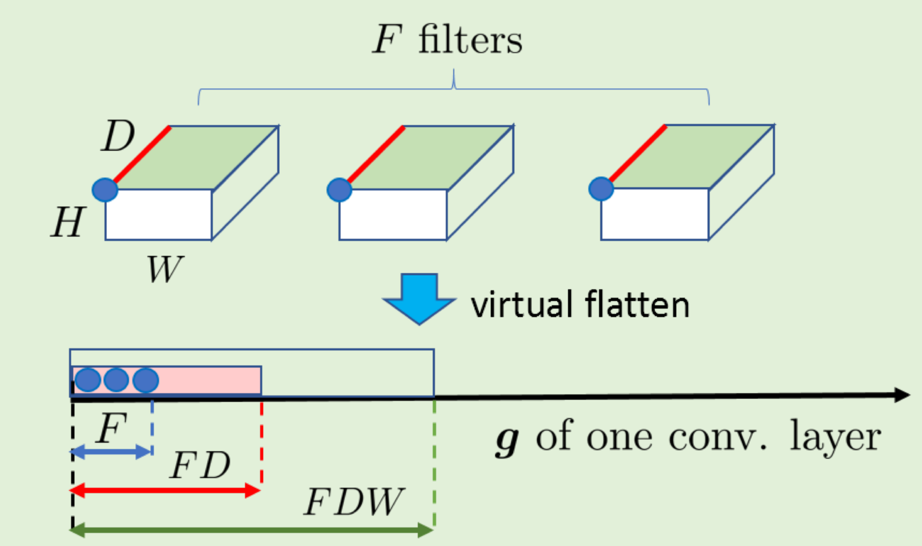
for CNN Gradients!

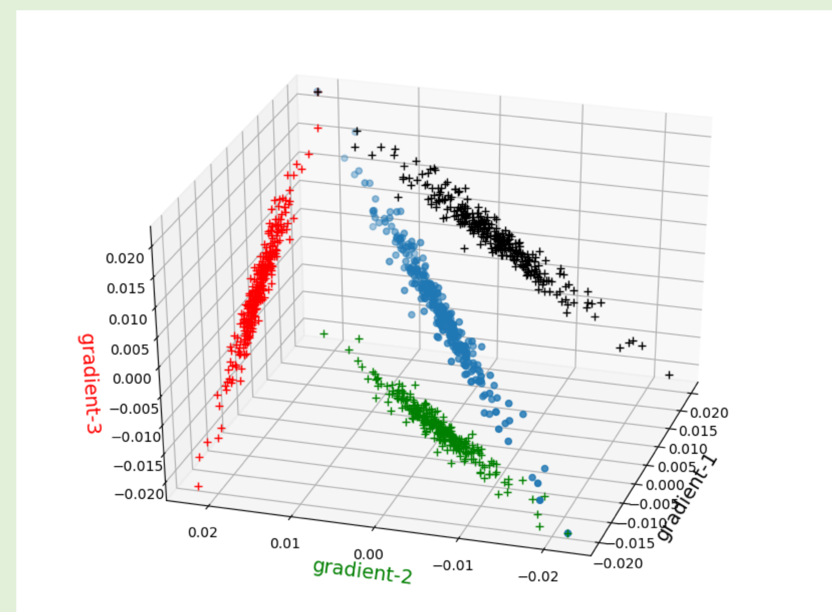
$$\sum Q(g_i) = Q\left(\sum g_i\right)$$



- $U_i \in \mathbb{R}^{d\times K}$ is the PCA matrix for slice i
- $g'_{ji} = U_i g_{ji}$ with compression ratio K/d
- In GradiVeQ, only U_0 is computed and re-used to compress all slices in a conv. layer
- After aggregation, multiply by $U_i^\top$ to decompress

How could linearity be possible?

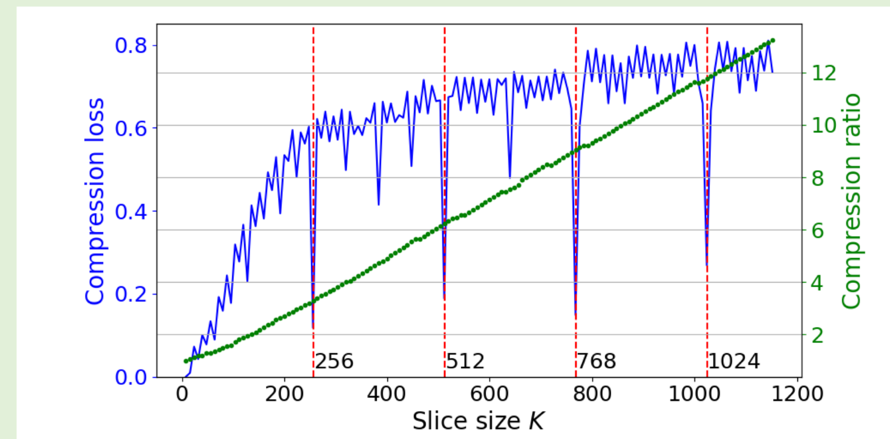




The values of 3 adjacent gradients over 150 iterations.

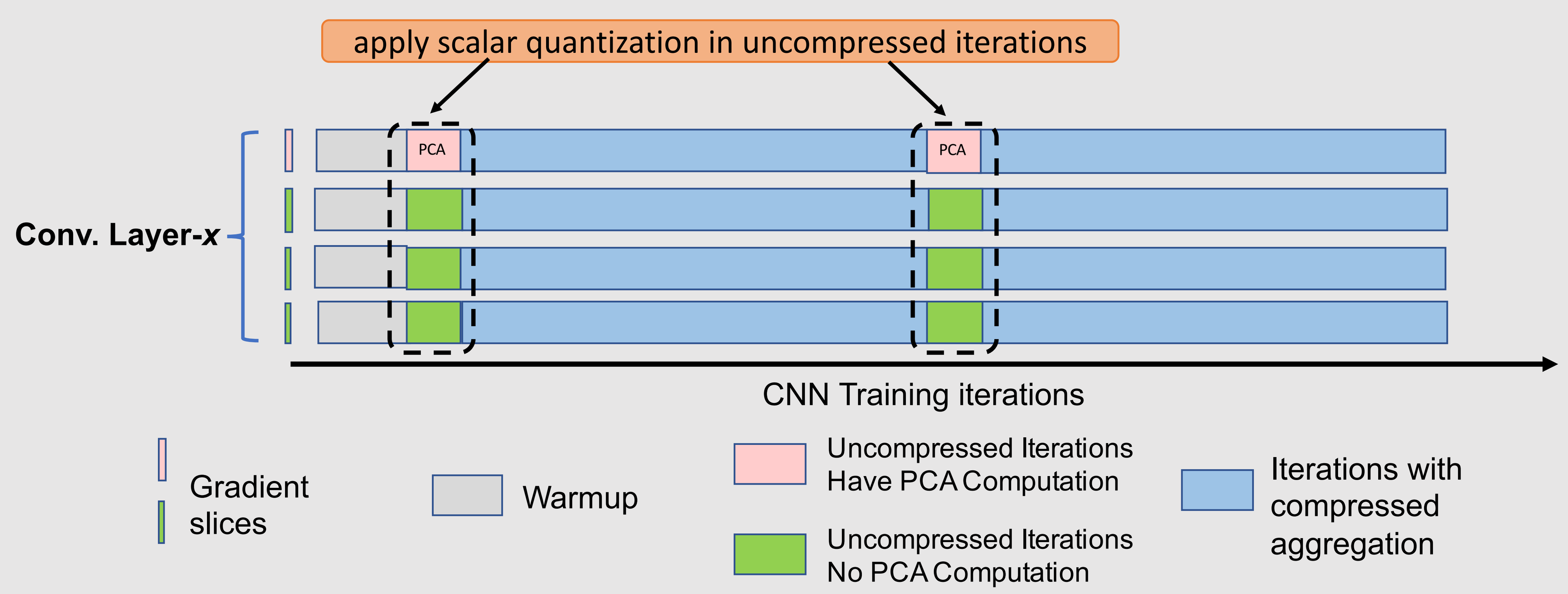
Linearity with excellent features:

- Strong linear correlation
- Temporal persistency
- Spatial consistency



The loss of using the compressor of the first K gradient to compress the remaining gradients

How to capture the linearity?

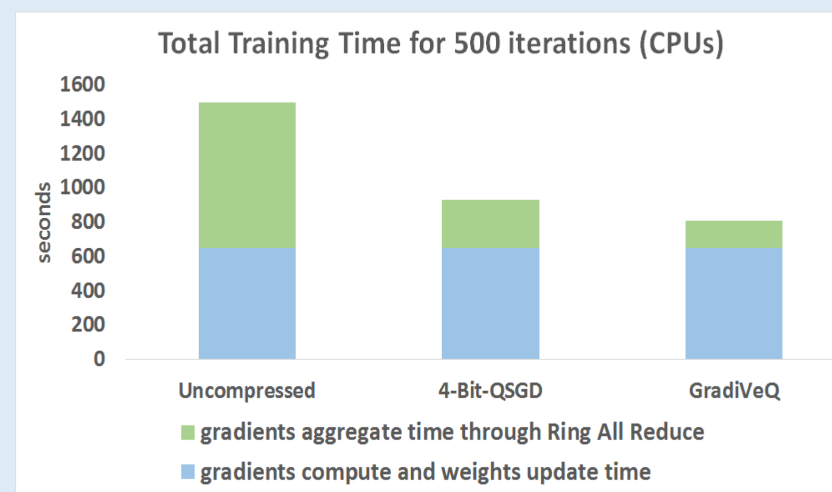


- Thanks to temporal persistency, we can invest time on PCA training;
- Thanks to spatial consistency, only need one PCA per layer;
- Low complexity;
- Compression is fully hidden behind RAR

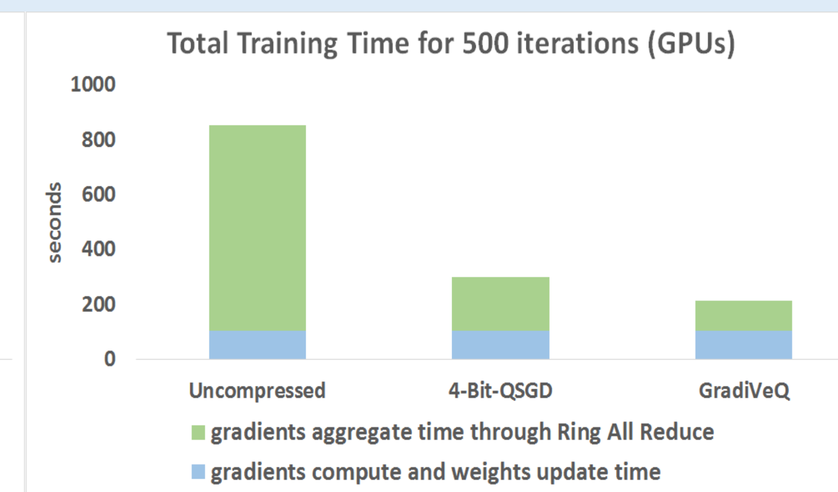
How do we do wall-clock wise?

Training ResNet-32 using CIFAR-100

	Training time (CPUs)	Training time (GPUs)	Top-1 accuracy
Baseline RAR	135,000 s	75,000 s	67.6%
4-bit QSGD	90,000 s	30,000 s	66.7%
GradiVeQ	**76,000 s**	**24,000 s**	66.6%



- 8x compression ratio
- 1.5x faster than baseline
- 1.2x faster than 4-bit-QSGD



- 8x compression ratio
- 4x faster than baseline
- 1.6x faster than 4-bit-QSGD

[1] F.Seide,H.Fu,J.Droppo,G.Li,andD.Yu,"1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs," *INTERSPEECH*, 2014

[2] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," *NIPS*, 2017.

USC

ILLINOIS

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN