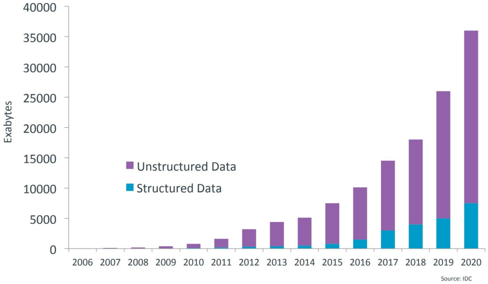
Computational Polarization: An Information-theoretic Method for Resilient Computing

Mert Pilanci

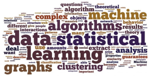
March 27, 2020

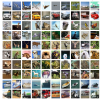
Electrical Engineering
Stanford University

Scale of data

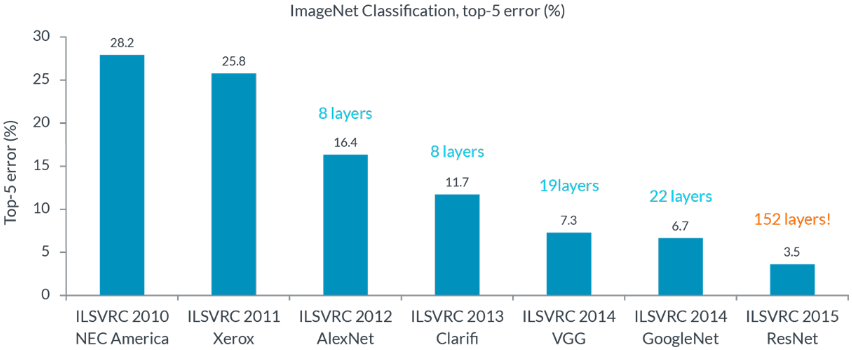


- Every day, we create 2.5 billion gigabytes of data
- Data stored grows 4x faster than world economy (Mayer-Schonberger)





Deep learning revolution



in machine learning and data science

- more data results in better and accurate models
  $\rightarrow$ **large scale distributed computing problems**

in machine learning and data science

- more data results in better and accurate models
  $\rightarrow$ **large scale distributed computing problems**

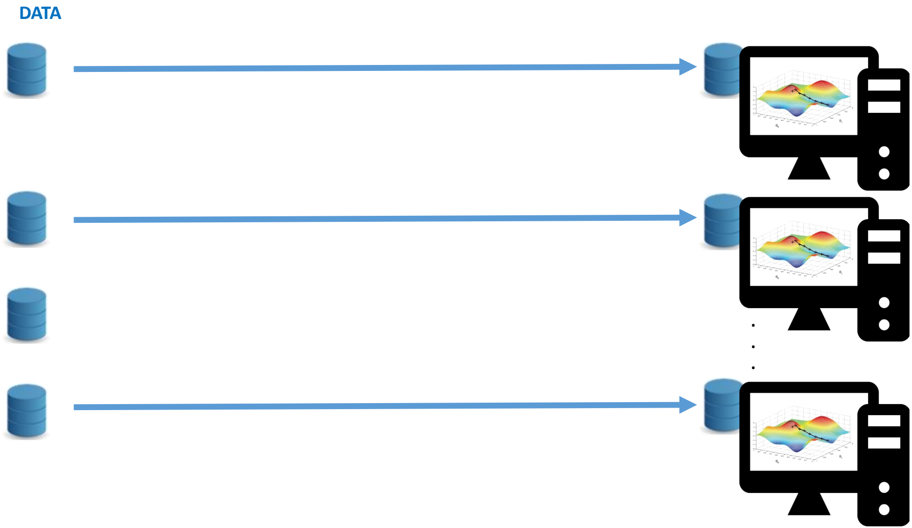
Can we scale computation inexpensively ?

Our approach: **serverless systems with error correction to auto-scale computation**

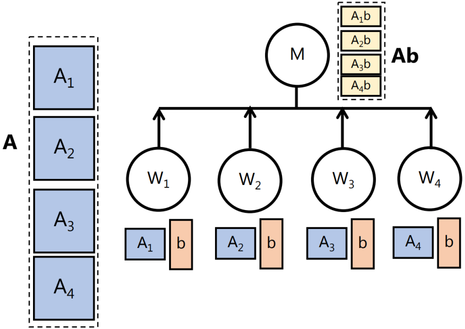
M. Pilanci, **Computational Polarization: An Information-theoretic Method for Resilient Computing**, accepted to IEEE Transactions on Information Theory, 2021

B. Bartan and M. Pilanci **Straggler Resilient Serverless Computing Based on Polar Codes**, Allerton 2019

Distributed computation

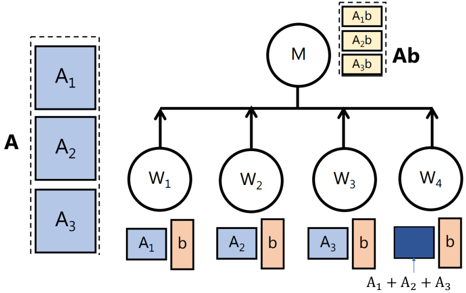


Error Resilient Matrix Multiplication



Speeding Up Distributed Machine Learning Using Codes. Lee et al., 2017

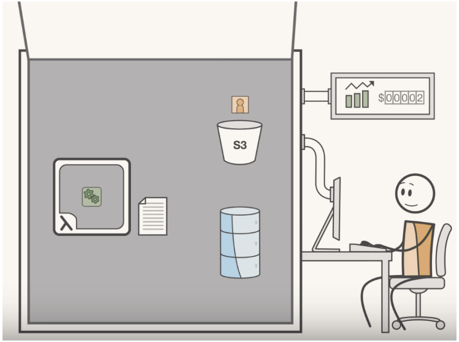
Error Resilient Matrix Multiplication



Speeding Up Distributed Machine Learning Using Codes. Lee et al., 2017

Serverless computing: AWS Lambda

- low cost, no upfront investment
- 900 seconds single-core, 3GB RAM
- Python, Java, C#

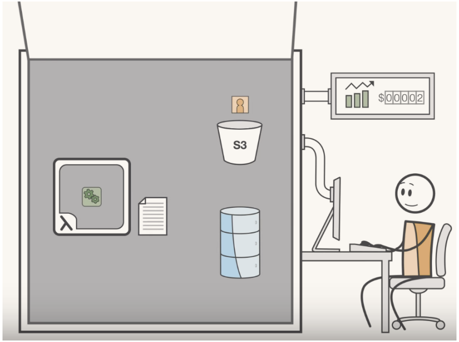


Serverless computing: AWS Lambda

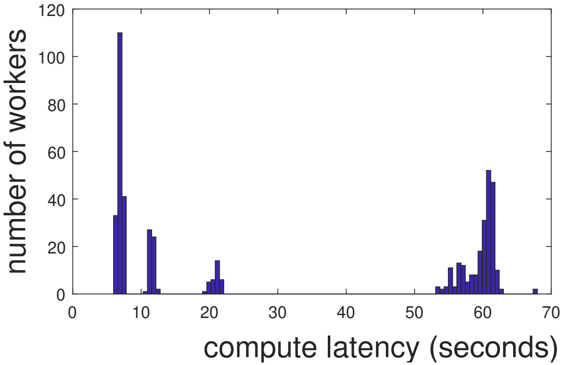
- Lambda functions are *stateless*

  Local file system access and child processes may not extend beyond the lifetime of the request

  Persistent state should be stored in a storage service (e.g., S3)
- Pywren (E. Jonas et al., 2017)
- Google Cloud and Microsoft Azure offer similar services

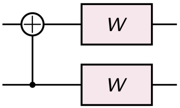


- return times

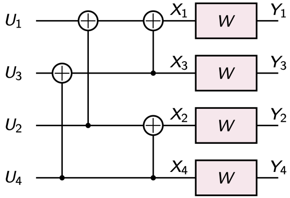


Polar Codes

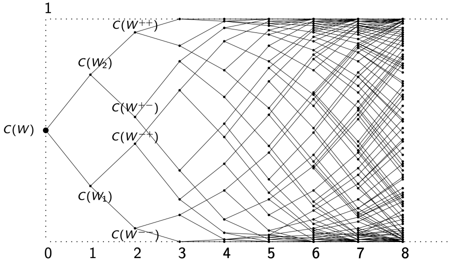
- Polar Codes were invented by Arikan in 2009
- Combines communication channels recursively to obtain better/worse channels
- It is the first code with an explicit construction to provably achieve the channel capacity for all symmetric discrete memoryless channels
- 3rd Generation Partnership Project (3GPP) adopted polar codes as the official coding scheme for the control channels of the 5G New Radio interface.



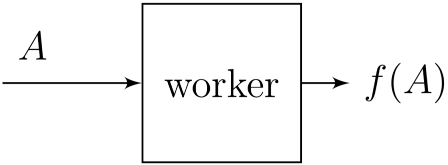
2× 2 construction



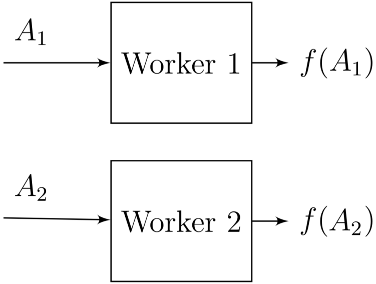
4×4 construction



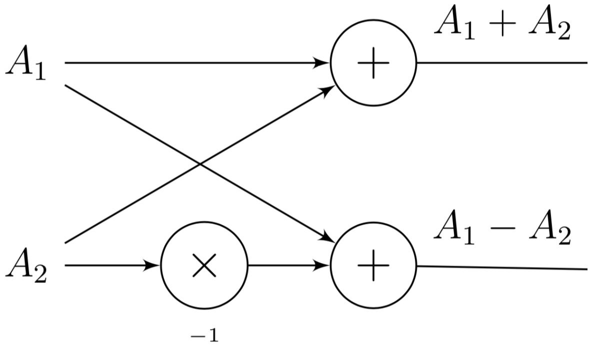
Computational Polar Codes



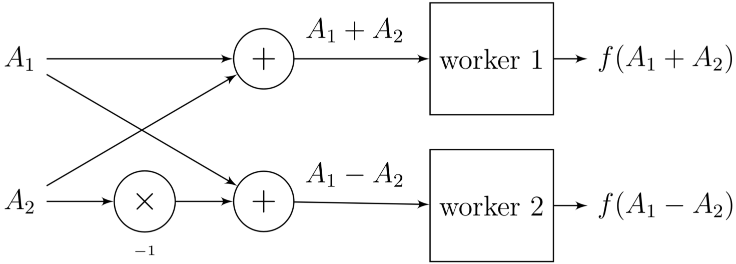
Computational Polar Codes



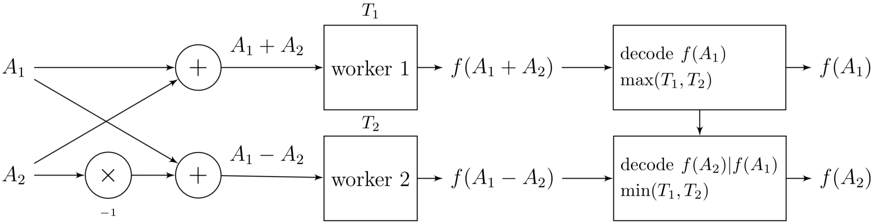
Hadamard transform



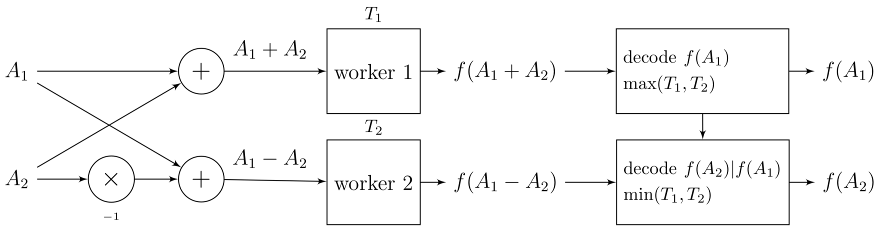
Butterfly coded computation

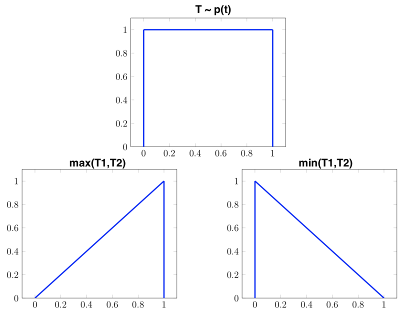


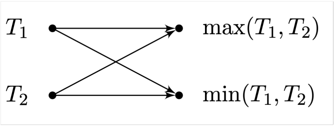
Decoding the original computation $f(A_1)$ and $f(A_2)$ for linear functions

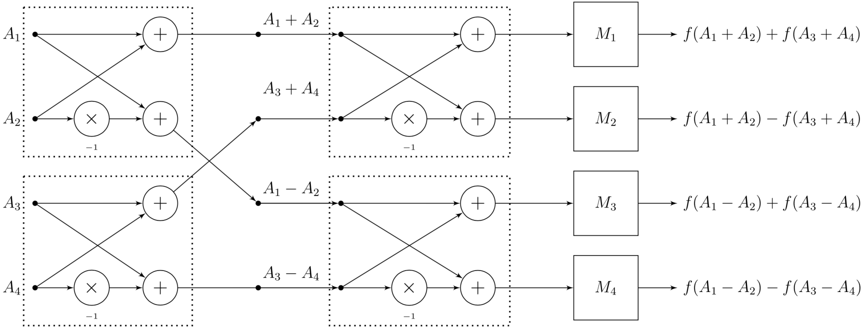


Runtime distribution

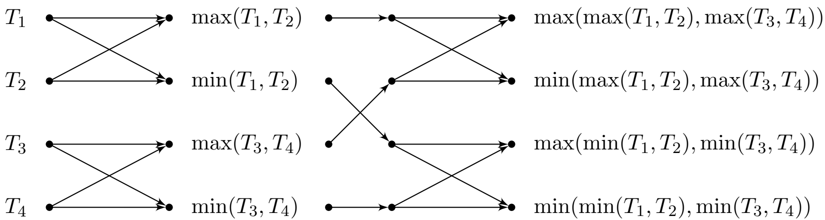




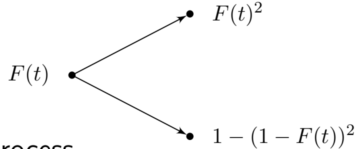



A_1
A_2
A_3
A_4
-1
-1
$A_1 + A_2$
$A_3 + A_4$
$A_1 - A_2$
$A_3 - A_4$
-1
-1
M_1
M_2
M_3
M_4
$f(A_1 + A_2) + f(A_3 + A_4)$
$f(A_1 + A_2) - f(A_3 + A_4)$
$f(A_1 - A_2) + f(A_3 - A_4)$
$f(A_1 - A_2) - f(A_3 - A_4)$

Computational Polarization Process



Computational Polarization Process



- Functional Martingale process
- $F(t)$ is the cumulative density function of the i.i.d. run-times

$$F_{n+1}(t) = \begin{cases} 1-(1-F_n(t))^2 & \text{with probability } \frac{1}{2} \\ F_n(t)^2 & \text{with probability } \frac{1}{2} \end{cases}$$

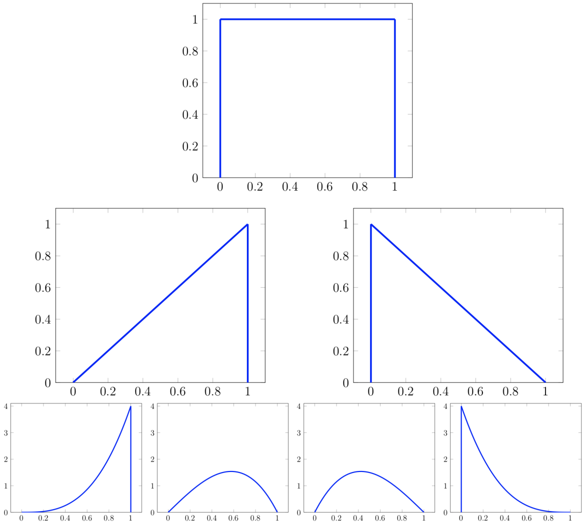
- $\mathbb{E}[F_{n+1}(t)|F_n] = F_n(t)$
- **Theorem:** $\left\|F_{n+1}(t) - F_n(t)\right\|_{L_2} \to 0$ as $n \to \infty$ with rate $O(2^{-2^{\sqrt{n}}})$

  $F_n(t)$ converges to unit step functions

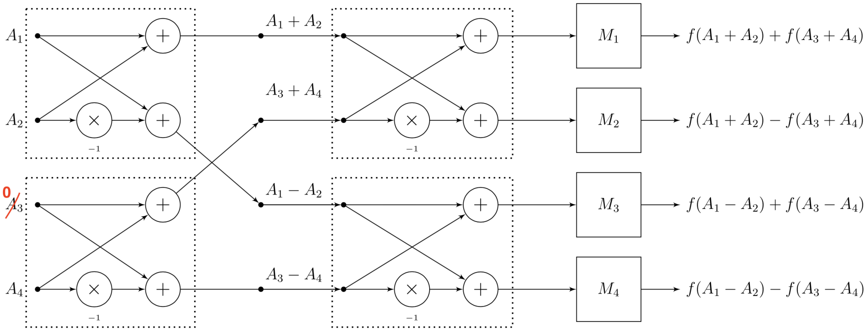
  run-time distributions converge to the Dirac measure

  M. Pilanci, **Computational Polarization: An Information-theoretic Method for Resilient Computing**, arXiv preprint 2021

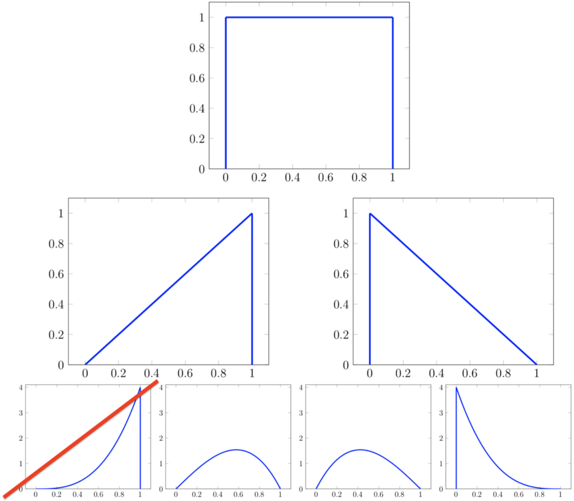
Run-time distributions



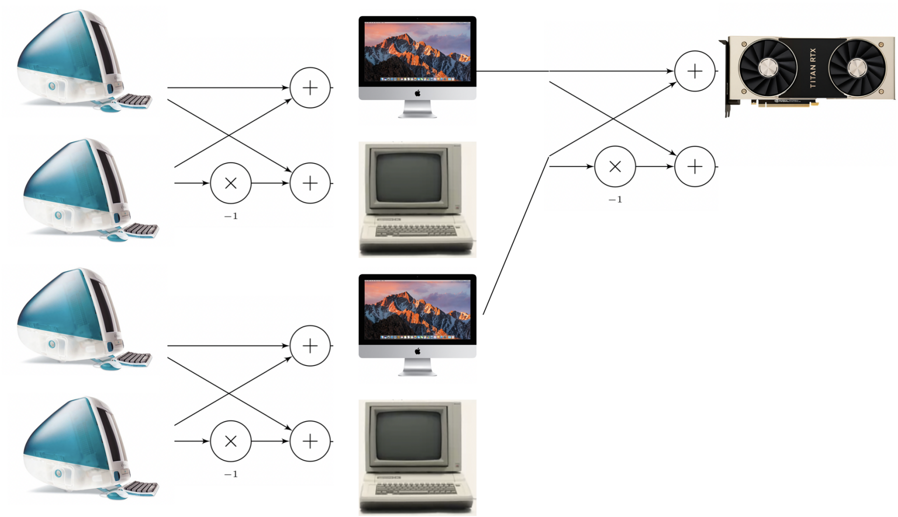
Fixing certain inputs to zero



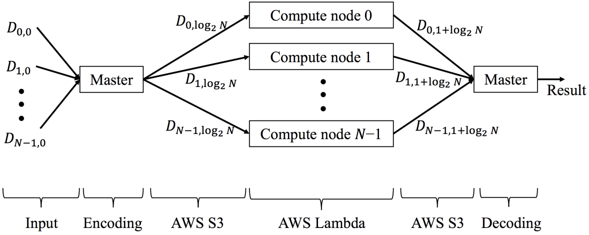
Fixing certain inputs to zero



Computational Polarization



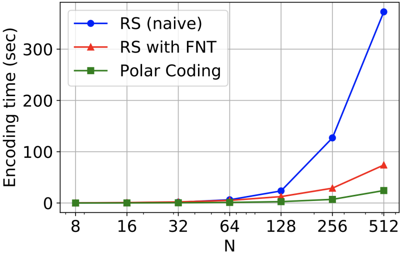
Computational Polarization



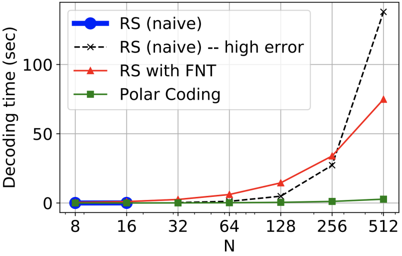
(B. Bartan and M. Pilanci, Straggler Resilient Serverless Computing Based on Polar Codes, 2019)

Comparison with other coding methods

- Reed-Solomon codes, LT codes, LDPC codes, Fermat Number Transform (FNT) based codes
- Computational Polar codes have $O(n \log n)$ encoding and decoding complexity
- only addition and subtraction operations in encoding and decoding
- can scale to 10,000 workers

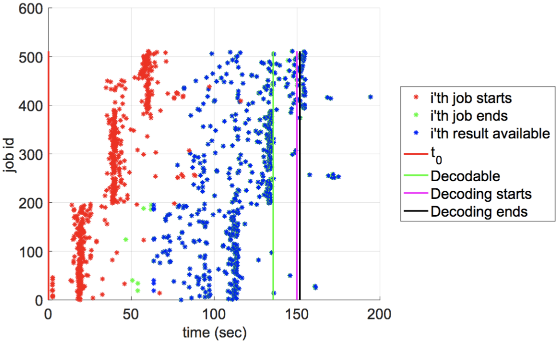


(a) Encoding

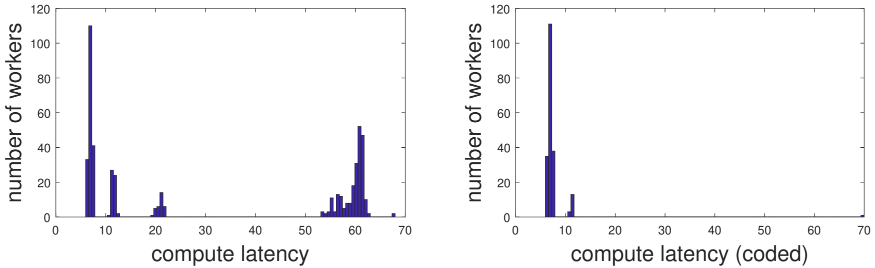


(b) Decoding

Compute jobs



Elastic computing



- AWS Lambda serverless compute jobs 1.5 GB memory each

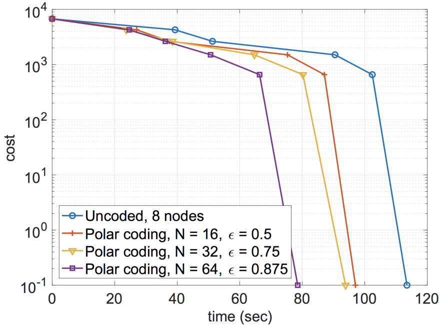
(a) uncoded: 500 workers

(b) coded: 1500 workers (1000 redundant parity)

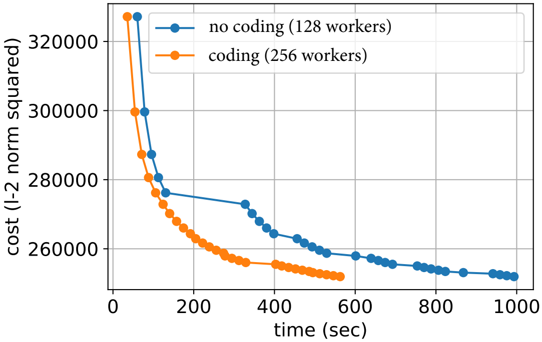
Computational Polarization for optimization on AWS Lambda

- encode data matrix A for gradient calculation, e.g., Ax and $A^T y$ for Least Squares and Generalized Linear Models

random data (20000×4800)



Imagenet (2013526×196608 ∼ 1.2 TB)



Computing Nonlinear Functions

- linear functions of data $f(A)$
- polynomial functions of data $f(A)$
- gradient and Hessian calculations involving data A

Computational Polarization for gradient estimation

- gradient estimator

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x+he_i) - f(x-he_i)}{2h}$$

- coded gradient estimator

$$\frac{f(x+hz_i) - f(x-hz_i)}{2h}$$

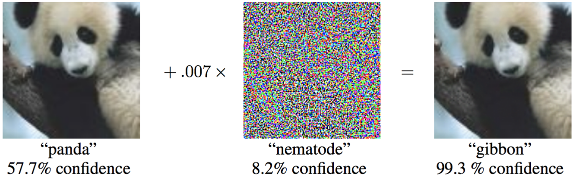
- z_i: redundant function evaluation directions =
- decode the gradient $\frac{\partial f(x)}{\partial x}$ from $\langle \frac{\partial f(x)}{\partial x}, z_i \rangle$

B. Bartan, M. Pilanci, Distributed Black-Box Optimization via Error Correcting Codes, 2019

Adversarial Examples

- given a trained neural network
- constrained optimization problem

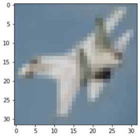
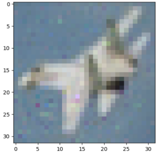
$\min_x ||x - x_0||$ subject to $\text{probability}_j(x) > \text{probability}_i(x)$

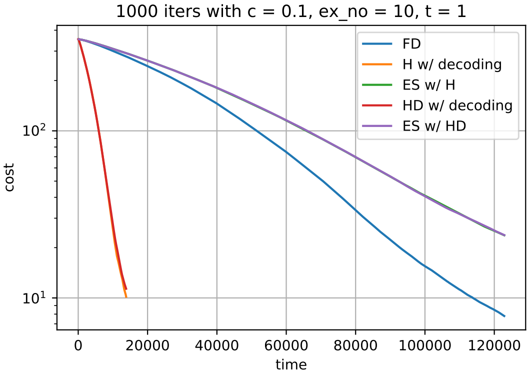


(Szegedy et al., 2014, Goodfellow et al., 2015)

Comparison with finite differences and random search

- plane classified as truck in CIFAR10

	
-	1.157
Pr(class = 0) = 0.9984509 Pr(class = 1) = 2.97521e-05 Pr(class = 2) = 8.516136e-05 Pr(class = 3) = 0.0006085799 Pr(class = 4) = 1.0096494e-05 Pr(class = 5) = 0.0007527866 Pr(class = 6) = 2.6582893e-05 Pr(class = 7) = 1.3017156e-05 Pr(class = 8) = 1.4263238e-05 Pr(class = 9) = 8.826052e-06	Pr(class = 0) = 3.2234791e-01 Pr(class = 1) = 1.7196946e-03 Pr(class = 2) = 2.8547004e-04 Pr(class = 3) = 1.1163305e-02 Pr(class = 4) = 5.5930251e-04 Pr(class = 5) = 9.8933965e-02 Pr(class = 6) = 7.9891388e-04 Pr(class = 7) = 6.8563002e-04 Pr(class = 8) = 3.3265535e-02 **Pr(class = 9) = 5.3024024e-01**



Conclusions and future work

✓ Scalable and error resilient distributed computing system

✓ cheap encoding and decoding

✓ distributed Least Squares and GLMS

→ privacy and encryption

→ more general convex optimization problems with constraints, e.g.,

convex optimization for neural networks

M. Pilanci, T. Ergen

Neural Networks are Convex Regularizers: Exact Polynomial-time Convex Optimization Formulations for Two-Layer Networks, arXiv:2002.10553

`stanford.edu/~pilanci`

- M. Pilanci, Computational Polarization: An Information-theoretic Method for Resilient Computing arXiv Preprint, 2021. https://arxiv.org/pdf/2109.03877.pdf

  B. Bartan, M. Pilanci, Straggler Resilient Serverless Computing Based on Polar Codes. 57th Annual Allerton Conference on Communication, Control, and Computing 2019, https://arxiv.org/pdf/1901.06811.pdf

  B. Bartan, M. Pilanci, Distributed Black-Box Optimization via Error Correcting Codes. 57th Annual Allerton Conference on Communication, Control, and Computing 2019, https://arxiv.org/pdf/1907.05984.pdf

  M. Pilanci, T. Ergen, Neural Networks are Convex Regularizers: Exact Polynomial-time Convex Optimization Formulations for Two-Layer Networks, ICML 2020