Are generative models the new sparsity*?

Bruno Loureiro
IdePHICS, EPFL

*from https://solevillar.github.io/2018/03/28/SUNLayer.html

Simons Institute for the Theory of Computing
Rigorous Evidence for Information-Computation Trade-offs workshop

Joint work with

Benjamin Aubin
(ClipDrop)

Lenka Zdeborová
(EPFL)

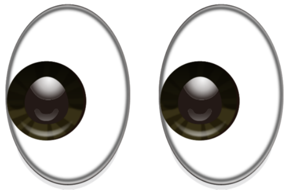
Antoine Maillard
(ENS Paris)

Antoine Baker
(ENS Paris)

Florent Krzakala
(EPFL)

Inverse problems

Signal	Channel	Observation
$\mathbf{x} \in \mathbb{R}^d$ $\longrightarrow$	f	$\longrightarrow$ $\mathbf{y} \in \mathbb{R}^n$



Inverse problems

Signal — Channel — Observation

$$\mathbf{x} \in \mathbb{R}^d \longrightarrow \boxed{f} \longrightarrow \mathbf{y} \in \mathbb{R}^n$$

Questions: How many samples?

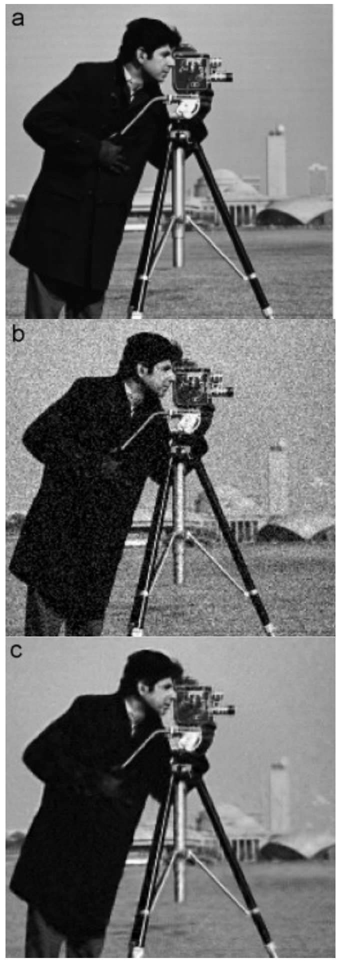
How to do it efficiently?

What is the role of structure?

Examples

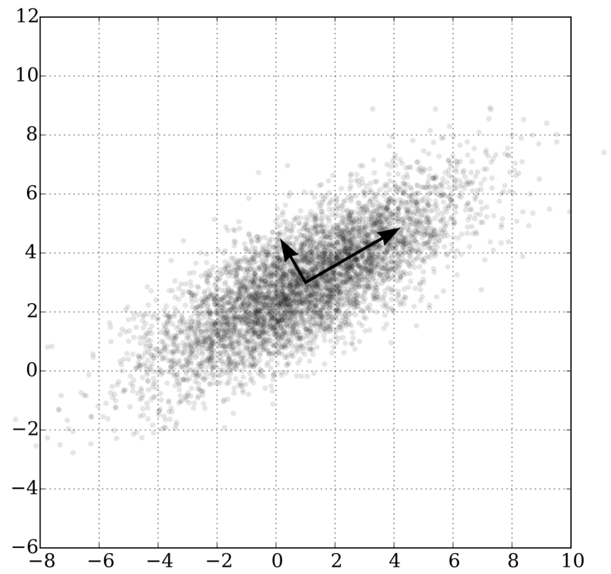
Denoising

$$\mathbf{y} = \mathbf{x} + \xi$$



Matrix factorisation

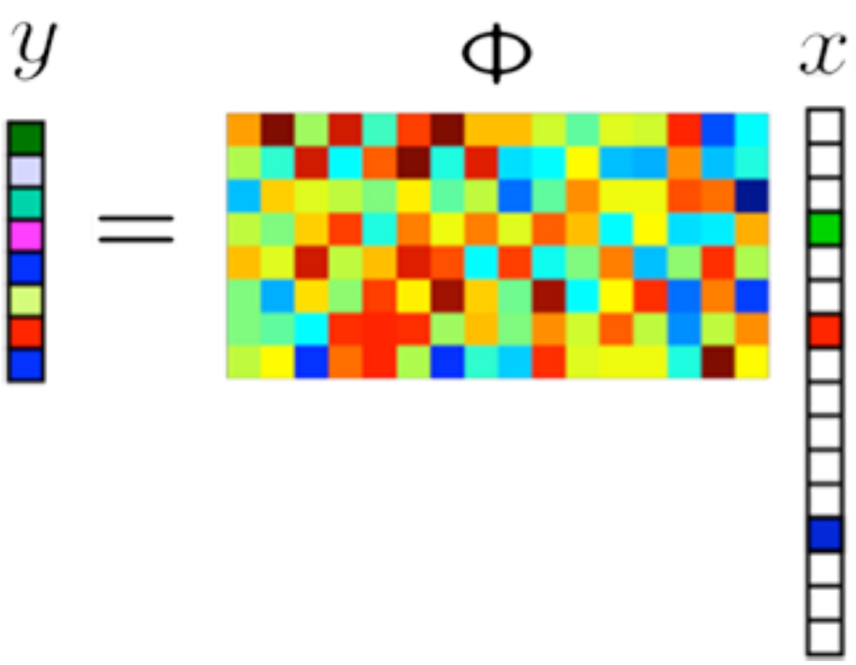
$$\mathbf{Y} = \mathbf{x}\mathbf{x}^\intercal + \mathbf{Z}$$



Compressed sensing
Phase retrieval

$$\mathbf{y} = \Phi\mathbf{x} + \xi$$

$$\mathbf{y} = |\mathbf{A}\mathbf{x}| + \xi$$

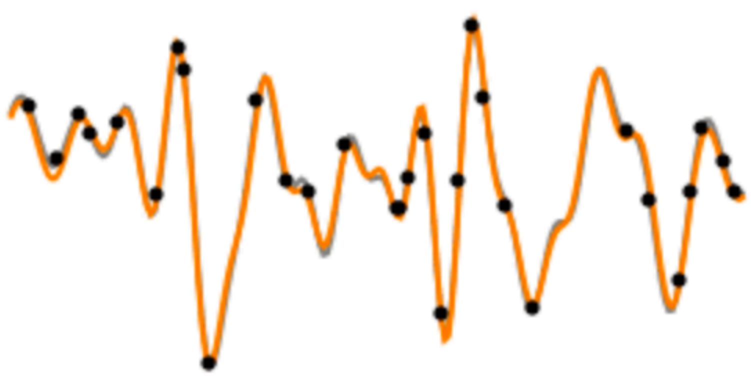


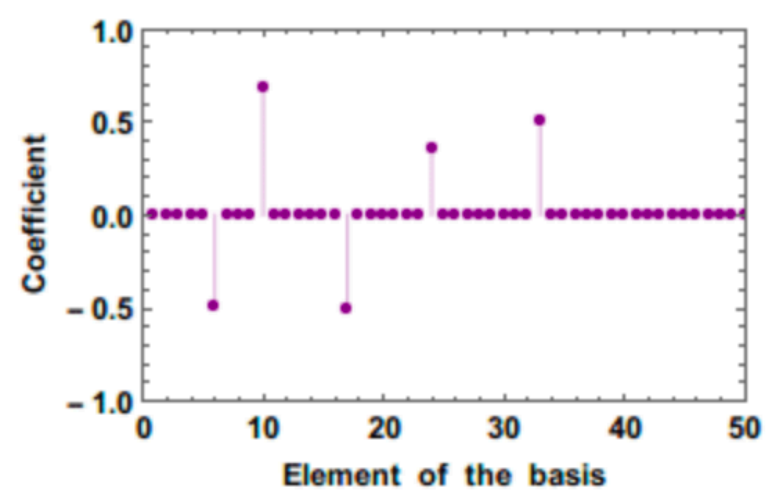
Sparsity

Structure helps!

If there is a basis (possibly learned) for which signal is sparse:

$$\mathbf{x} = (-\mathbf{z} - \underbrace{0\cdots0}_{d-k})$$



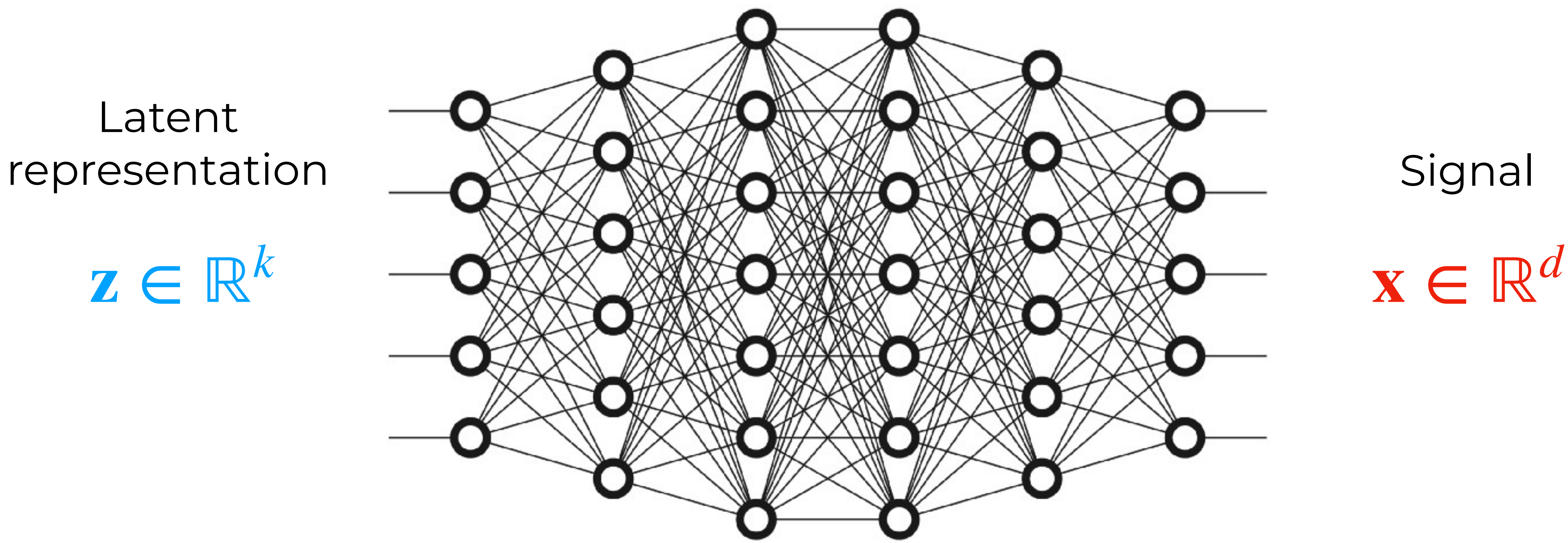


Efficient reconstruction might be possible both statistically (NP-hard?) and algorithmically, e.g. L1-minimisation in CS.

Learning a basis

[Ulyanov et al. 17,
Bora et al. 17',
Hecklel, Hand 18']

Deep generative networks: VAEs, GANs, etc.



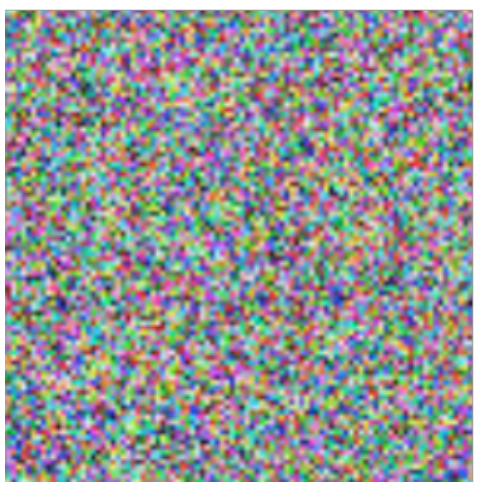
$$\mathbf{x} = \sigma^{(L)}\left(\mathrm{W}^{(\mathrm{L})}\sigma^{(\mathrm{L}-1)}\left(\mathrm{W}^{(\mathrm{L}-1)}\cdots\sigma^{(1)}\left(\mathrm{W}^{(1)}\mathbf{z}\right)\cdots\right)\right) \in \mathbb{R}^d$$

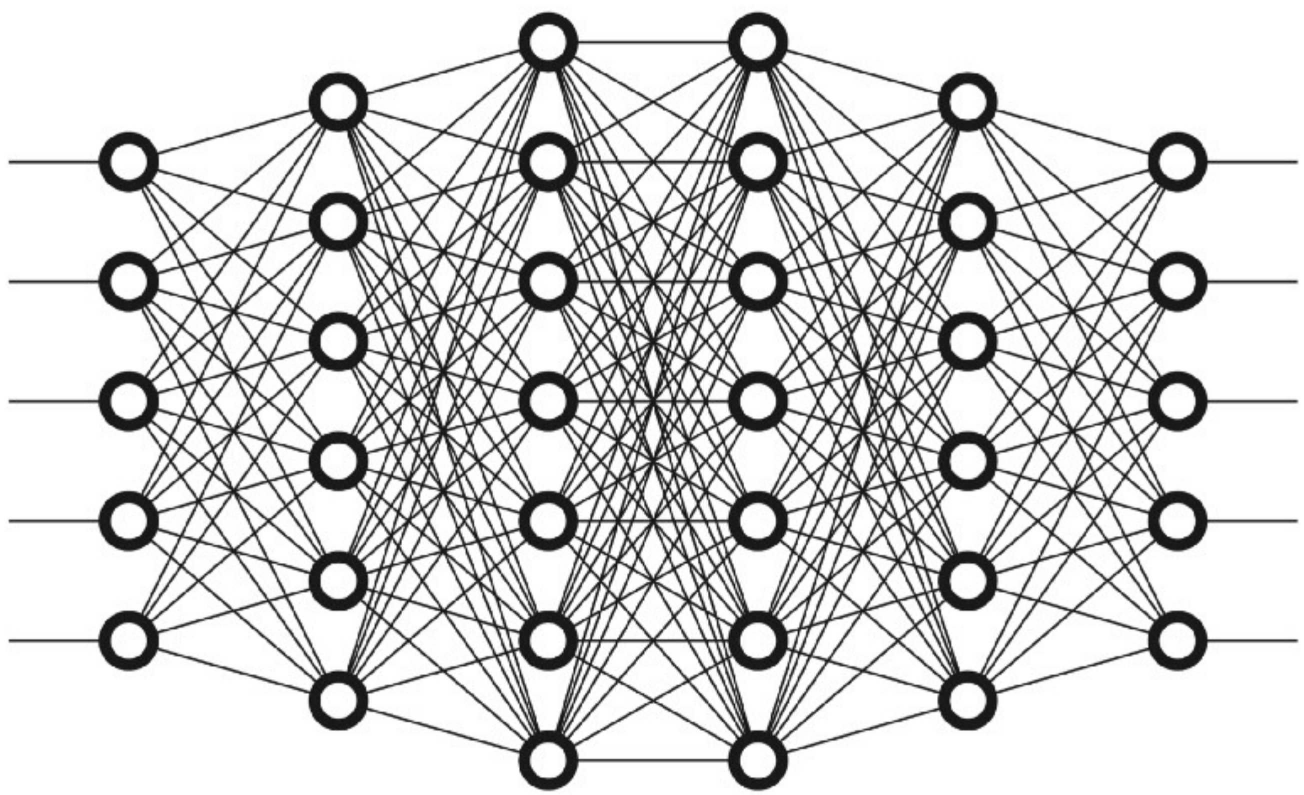
Learning a basis

[Ulyanov et al. 17,
Bora et al. 17',
Hecklel, Hand 18']

Deep generative networks: VAEs, **GANs**, etc.

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathrm{I}_k)$$

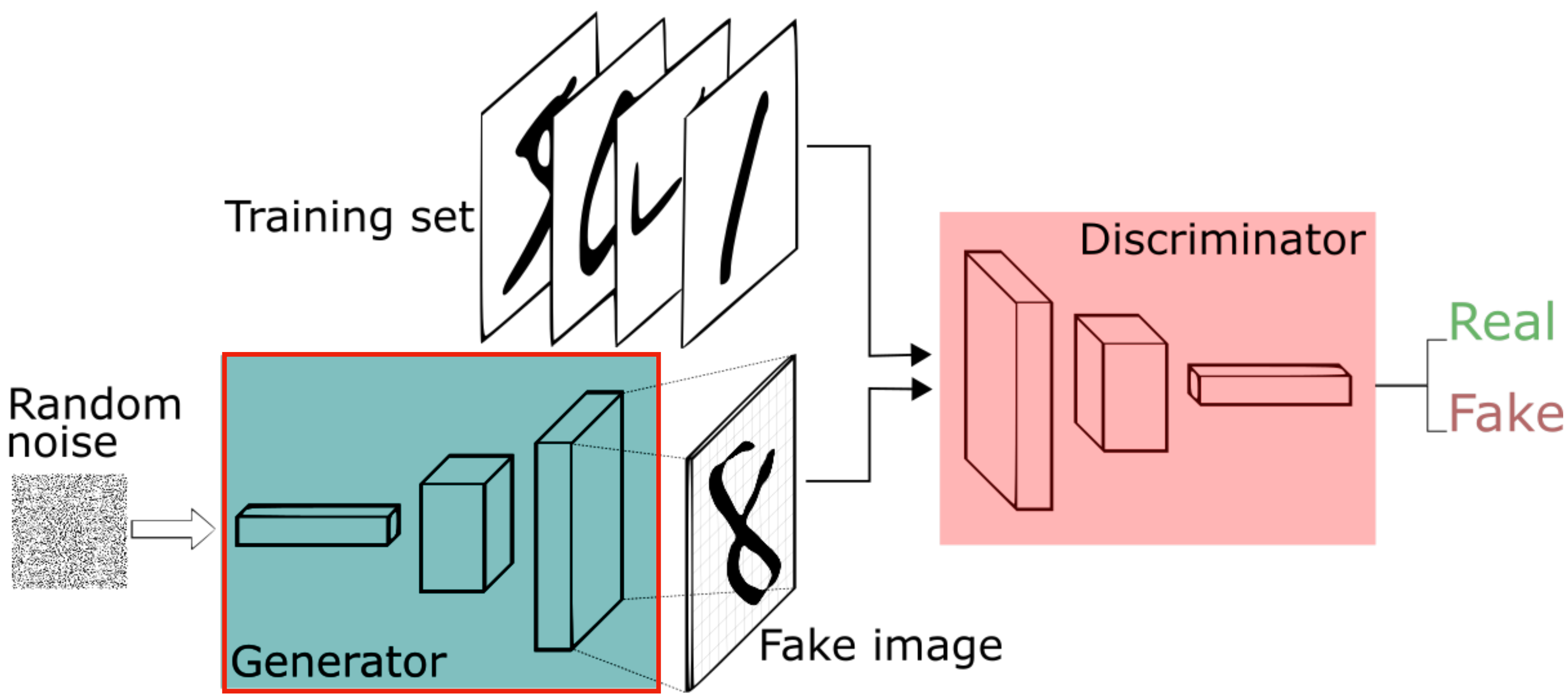




https://www.thispersondoesnotexist.com/

Setting

Someone, somewhere



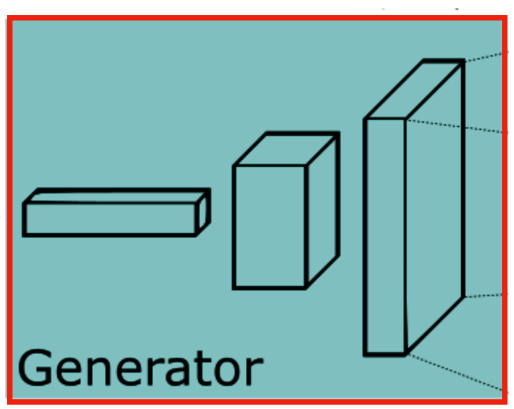
Setting

Signal

Channel

Observation

$$\mathbf{x} \in \mathbb{R}^d \longrightarrow f \longrightarrow \mathbf{y} \in \mathbb{R}^n$$



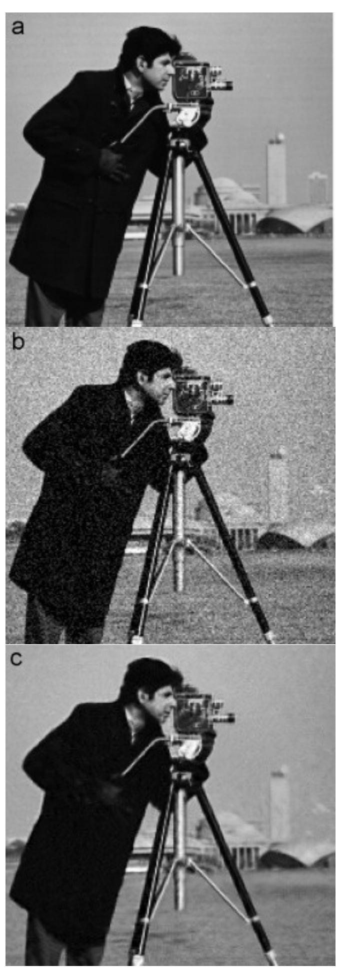
Prior on signal

$$P\left(\mathbf{x} \mid \left\{\mathbf{W}^{(l)}\right\}_{l=1}^{L}\right)$$

Matrix factorisation

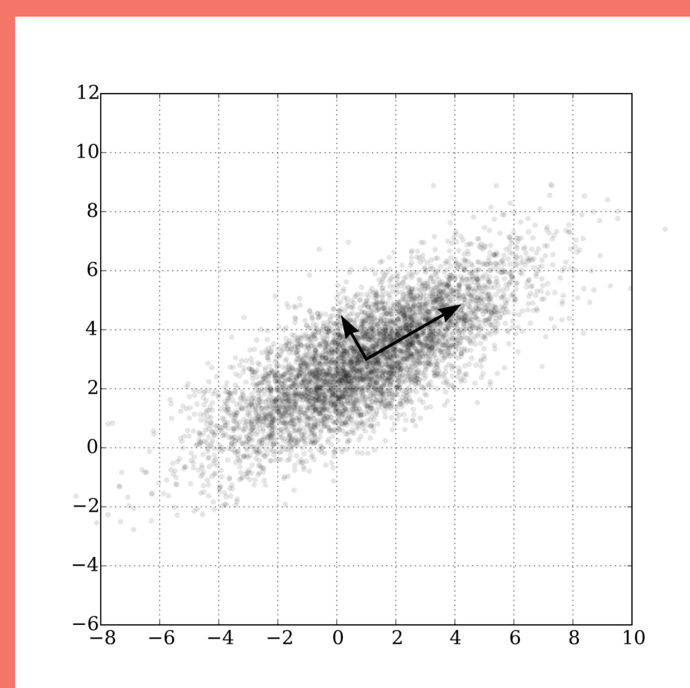
Denoising

$$\mathbf{y} = \mathbf{x} + \xi$$



Matrix factorisation

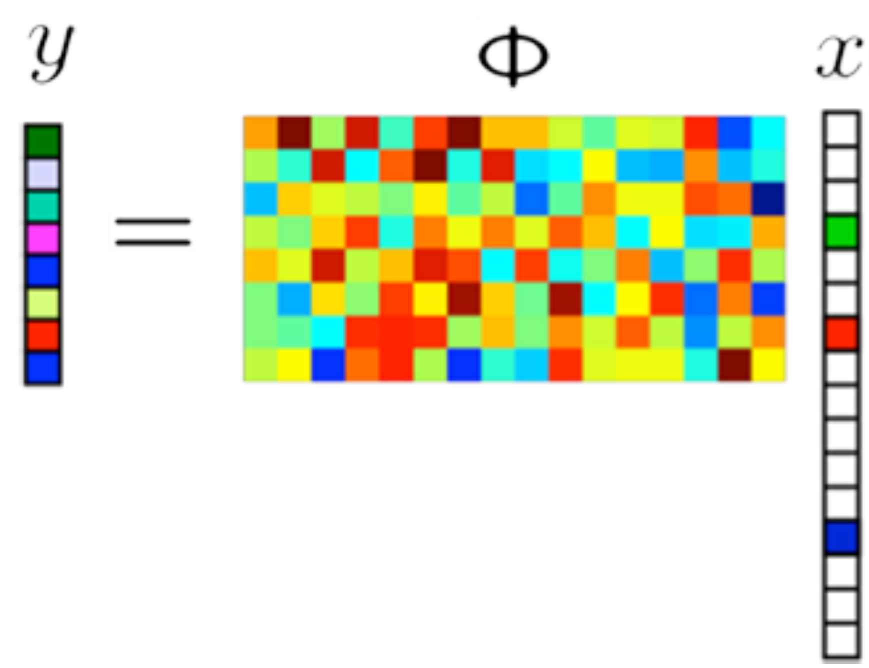
$$\mathbf{Y} = \mathbf{x}\mathbf{x}^\intercal + \mathbf{Z}$$



Compressed sensing
Phase retrieval

$$\mathbf{y} = \Phi\mathbf{x} + \xi$$

$$\mathbf{y} = |\mathbf{A}\mathbf{x}| + \xi$$



Matrix factorisation

Signal
“Spike”

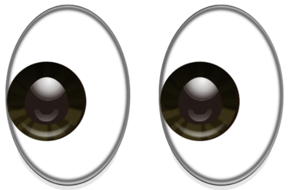
Channel

AWGN

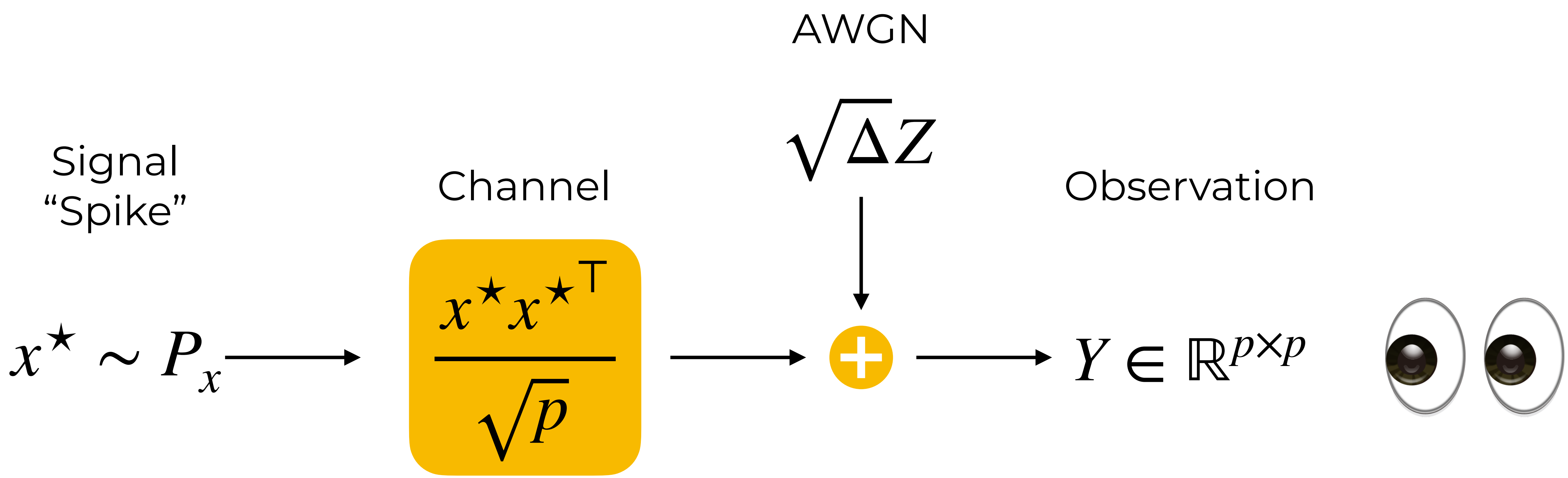
$$\sqrt{\Delta}Z$$

Observation

$$x^\star \sim P_x \longrightarrow \frac{x^\star {x^\star}^\top}{\sqrt{p}} \longrightarrow \oplus \longrightarrow Y \in \mathbb{R}^{p\times p}$$



Matrix factorisation

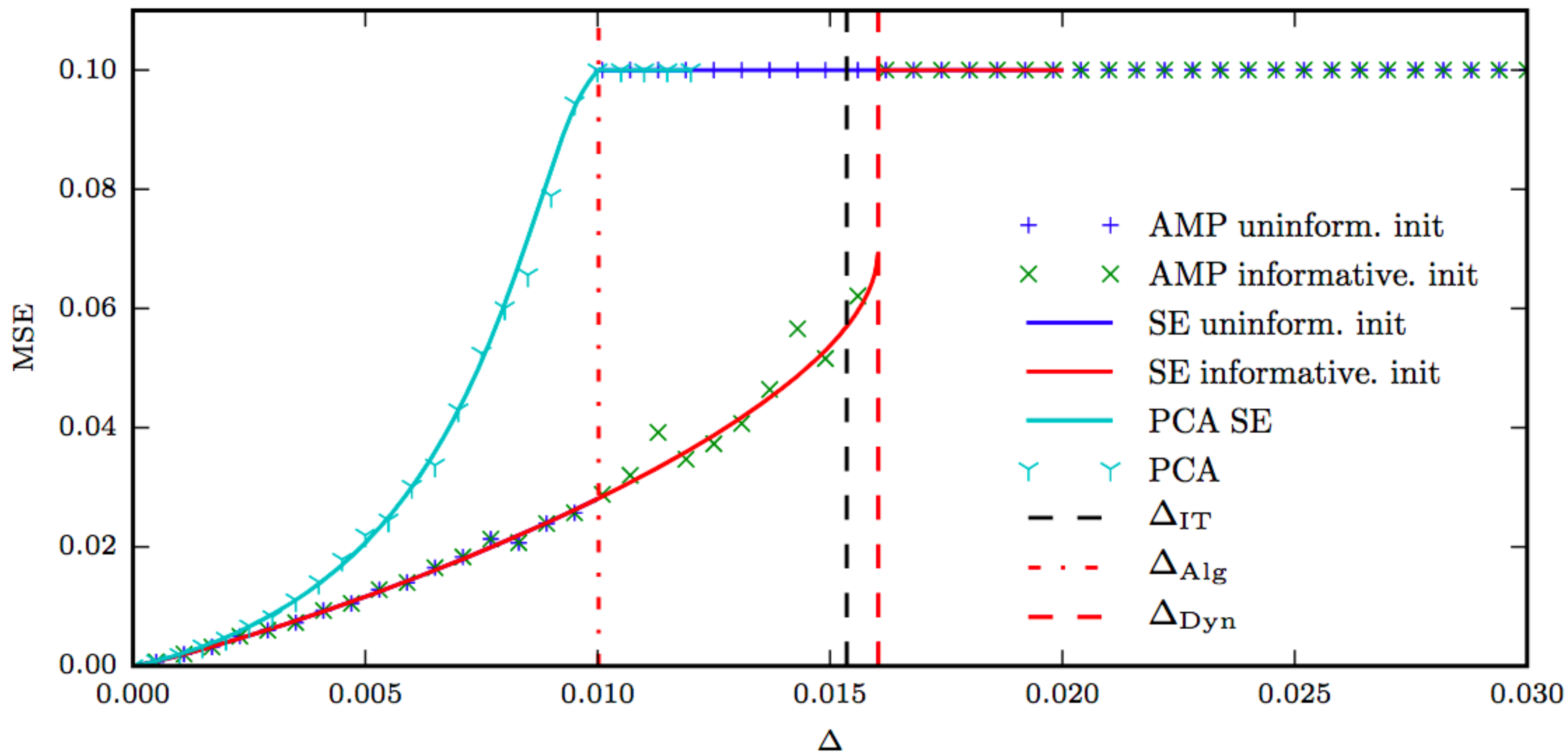


Goal: reconstruct $x^\star$ from knowledge of P_x, Δ
in the high-dimensional regime
$p \to \infty$ and $\Delta = O(1)$

Sparse case

Consider a sparse spike

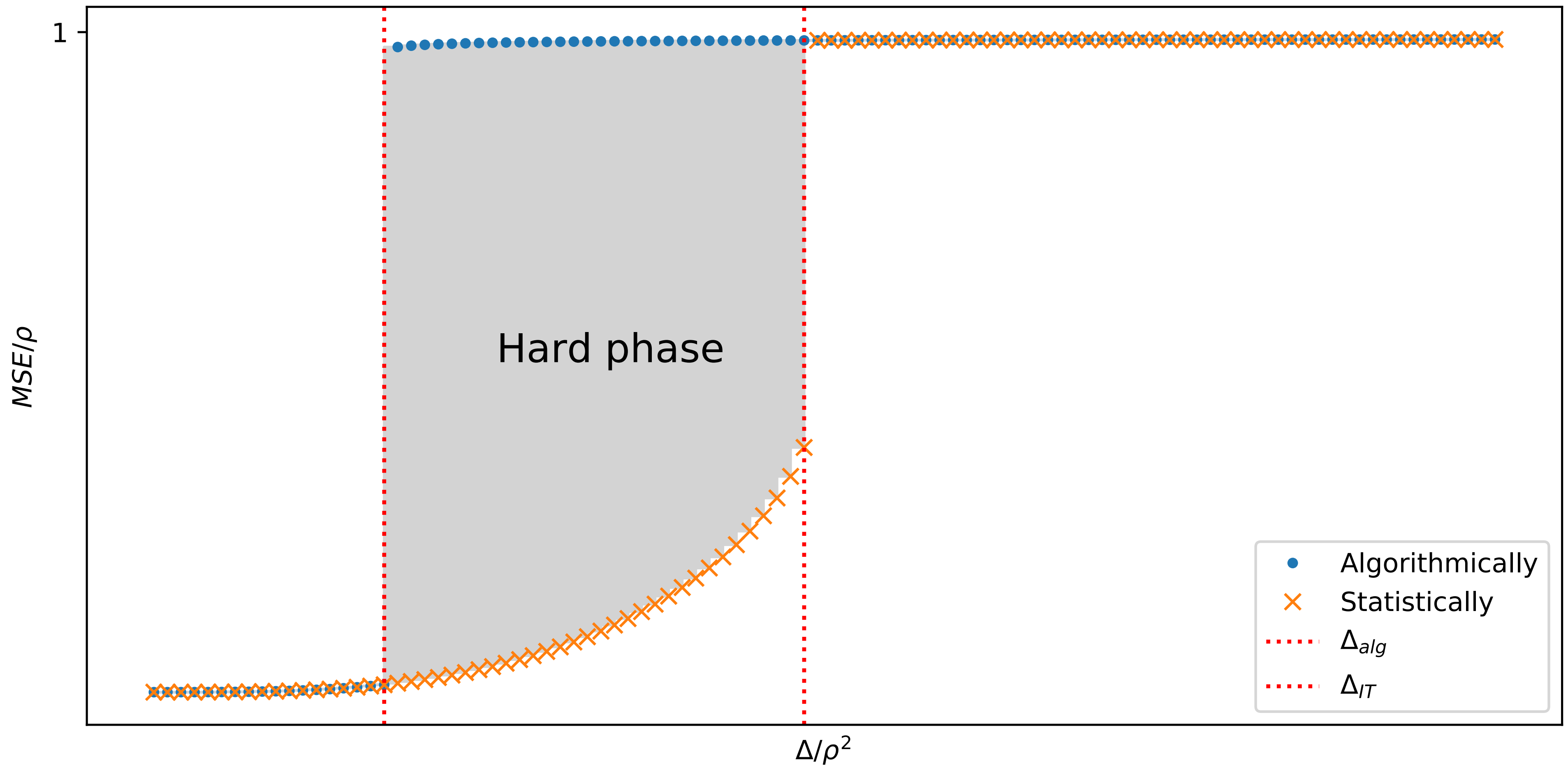
$$x^\star \sim \prod_{i=1}^{p} \left[(1-\rho)\delta_0 + \rho\mathcal{N}(0,1)\right] \qquad \rho = 0.1$$



[Lesieur et al. 17']

Sparse case

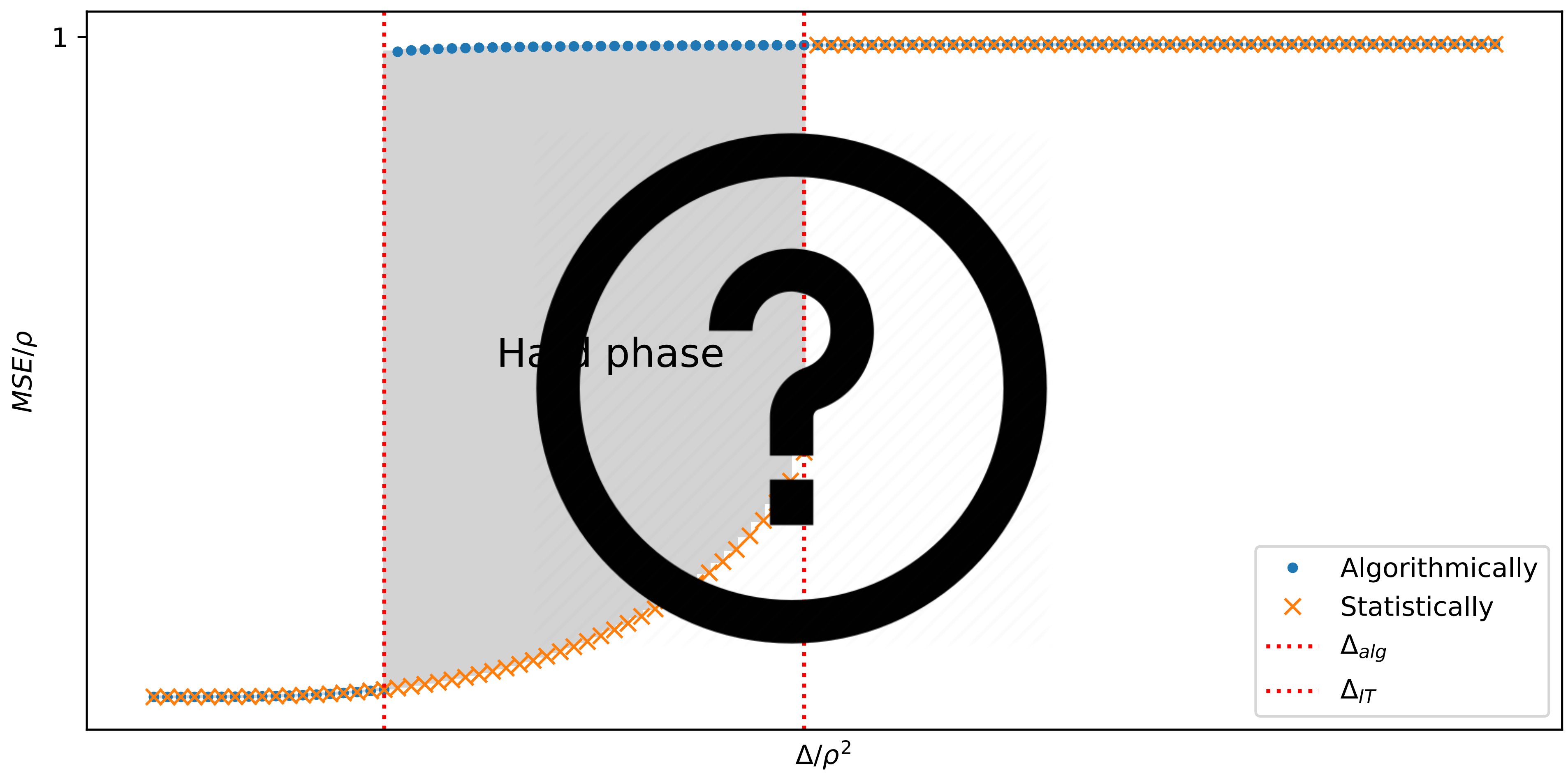
Large sparsity leads to
statistical-to-algorithmic gap



Generative networks

$$\mathbf{x} = \sigma^{(L)}\left(\mathrm{W}^{(\mathrm{L})}\sigma^{(\mathrm{L}-1)}\left(\mathrm{W}^{(\mathrm{L}-1)}\dots\sigma^{(1)}\left(\mathrm{W}^{(1)}\mathbf{z}\right)\cdots\right)\right)$$

$W^{(l)} \in \mathbb{R}^{k_{l+1}\times k_l}$ i.i.d. Gaussian , $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathrm{I}_{\mathrm{k}})$



Statistical guarantee

Theorem (informal) In the high-dimensional limit,

$$\mathrm{MMSE} = \rho_x - q_x^\star$$

where: $q_x^\star = \underset{0 \le q_x \le \rho}{\mathrm{arginf}}\ i_{\mathrm{RS}}(\Delta, q_x)$

$$i_{\mathrm{RS}}(\Delta, q_v) = \frac{(\rho_x - q_x)^2}{4\Delta} + \lim_{p \to \infty} \frac{I\left(x; x + \sqrt{\frac{\Delta}{q_x}}\xi\right)}{p}$$

$$\rho_x = \lim_{p \to \infty} \mathbb{E}_{P_x}\left[\frac{x \cdot x}{p}\right], \qquad \xi \sim \mathcal{N}(0,1)$$

Statistical guarantee

Remark:

$$\frac{1}{p} I \left(x; x + \sqrt{\frac{\Delta}{q_x}} \xi \right)$$

Mutual information density for a denonising problem.

Has been rigorously computed for

$$P_x(x) = \prod_{i=1}^{p} P_x(x_i)$$

Uncorrelated signal

[Lesieur et al. 15-17']

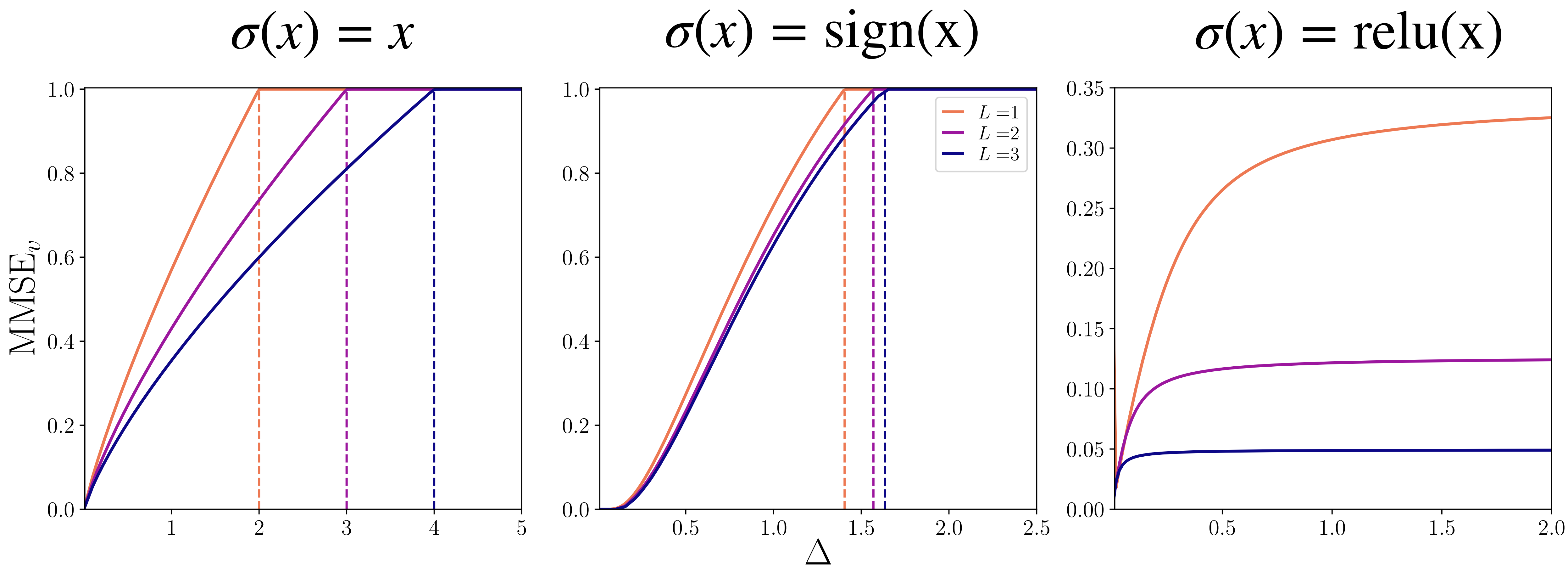
$$x = \varphi^{(2)} \left(W^2 \varphi^{(1)} \left(W^{(1)} z \right) \right)$$

2-layer NN with random weights

[Krzakala et al. 16']
[Barbier et al. 16']

Role of depth

$$\mathbf{x}^\star = \sigma^{(L)}\left(\mathrm{W}^{(\mathrm{L})}\sigma^{(\mathrm{L}-1)}\left(\mathrm{W}^{(\mathrm{L}-1)}\ldots\sigma^{(1)}\left(\mathrm{W}^{(1)}\mathbf{z}\right)\cdots\right)\right)$$



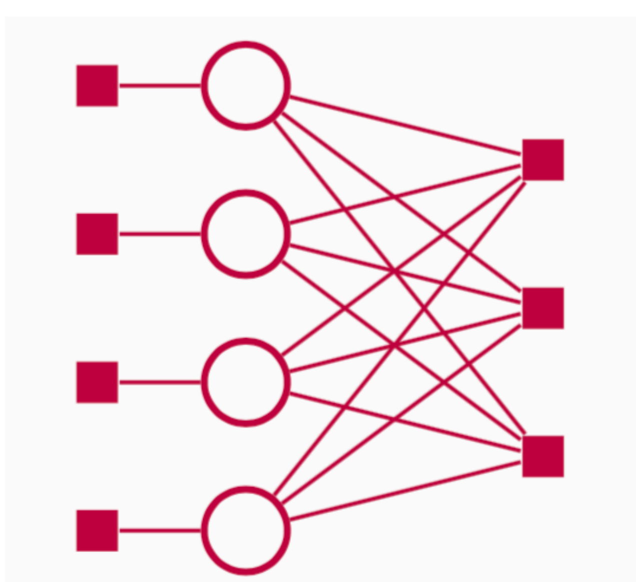
Compositional Principle

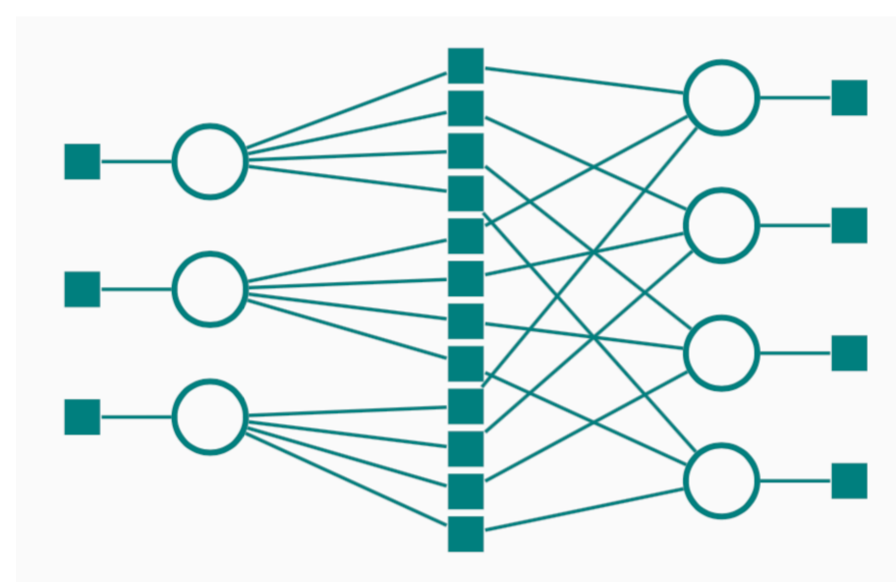
A compositional principle for AMP

$$z \to h^{(1)} \to \cdots \to h^{(L)} \to x \to Y$$

Generative layers

estimation layer





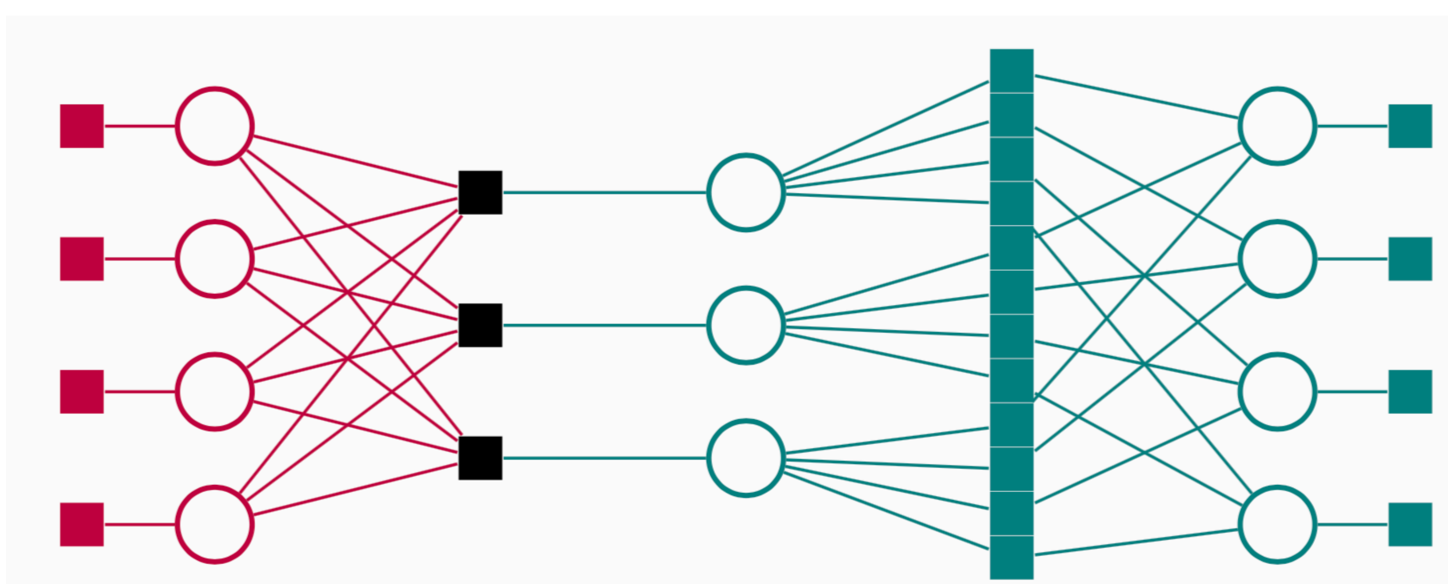
Compositional Principle

A compositional principle for AMP

$$z \to h^{(1)} \to \cdots \to h^{(L)} \to x \to Y$$

Generative layers

estimation layer



AMP for one-layer prior

Input: $Y \in \mathbb{R}^{p \times p}$ and $W \in \mathbb{R}^{p \times k}$:

Initialize with: $\hat{\mathbf{v}}^{t=1} = \mathcal{N}(\mathbf{0}, \sigma^2 I_p)$, $\hat{\mathbf{z}}^{t=1} = \mathcal{N}(\mathbf{0}, \sigma^2 I_k)$, and $\hat{\mathbf{c}}_v^{t=1} = I_p$, $\hat{\mathbf{c}}_z^{t=1} = I_k$, $t = 1$.

repeat

  Spiked layer denoising:

  $\mathbf{B}_v^t = \frac{1}{\Delta} \frac{Y}{\sqrt{p}} \hat{\mathbf{v}}^t - \frac{1}{\Delta} \frac{\left(I_p^\intercal \hat{\mathbf{c}}_v^t\right)}{p} \hat{\mathbf{v}}^{t-1}$ and $A_v^t = \frac{1}{\Delta p} (\|\hat{\mathbf{v}}^t\|_2)^2 I_p$.

  Generative layer denoising:

  $V^t = \frac{1}{k} \left(I_k^\intercal \hat{\mathbf{c}}_z^t\right) I_p$, $\boldsymbol{\omega}^t = \frac{1}{\sqrt{k}} W \hat{\mathbf{z}}^t - V^t \mathbf{g}^{t-1}$ and $\mathbf{g}^t = f_{\text{out}}\left(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t\right)$,

  $\Lambda^t = \frac{1}{k} \|\mathbf{g}^t\|_2^2 t I_k$ and $\gamma^t = \frac{1}{\sqrt{k}} W^\intercal \mathbf{g}^t + \Lambda^t \hat{\mathbf{z}}^t$.

  Marginals estimation:

  $\hat{\mathbf{v}}^{t+1} = f_v(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t)$ and $\hat{\mathbf{c}}_v^{t+1} = \partial_B f_v(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t)$,

  $\hat{\mathbf{z}}^{t+1} = f_z(\gamma^t, \Lambda^t)$ and $\hat{\mathbf{c}}_z^{t+1} = \partial_\gamma f_z(\gamma^t, \Lambda^t)$,

  $t = t + 1$.

until Convergence.

Output: $\hat{\mathbf{v}}, \hat{\mathbf{z}}$.

TRAMP

Compositional Inference with Tree Approximate Message Passing

Antoine Baker ANTOINE.BAKER@ENS.FR
Florent Krzakala FLORENT.KRZAKALA@ENS.FR
Laboratoire de Physique
CNRS, École Normale Supérieure, PSL University
Paris, France

Benjamin Aubin BENJAMIN.AUBIN@CEA.FR
Lenka Zdeborová LENKA.ZDEBOROVA@CEA.FR
Institut de Physique Théorique
CNRS, CEA, Université Paris-Saclay
Saclay, France

Abstract

We introduce tramp, standing for *TRee Approximate Message Passing*, a python package for compositional inference that runs Expectation Propagation on high-dimensional tree-structured models. The package provides an unifying framework to study several approximate message passing algorithms previously derived for a variety of machine learning tasks such as generalized linear models, inference in multi-layer networks, matrix factorization, and reconstruction using non-separable penalties. For some models, the performance of the algorithm can be theoretically predicted by the State Evolution, and the measurements entropy estimated by the free entropy formalism. The implementation is modular by design: each module, which implements a factor, can be composed at will with other modules to solve complex inference tasks. The user only needs to declare the factor graph of the model: the Expectation Propagation, State Evolution and entropy estimation are fully automated. The source code is publicly available at https://github.com/sphinxteam/tramp.

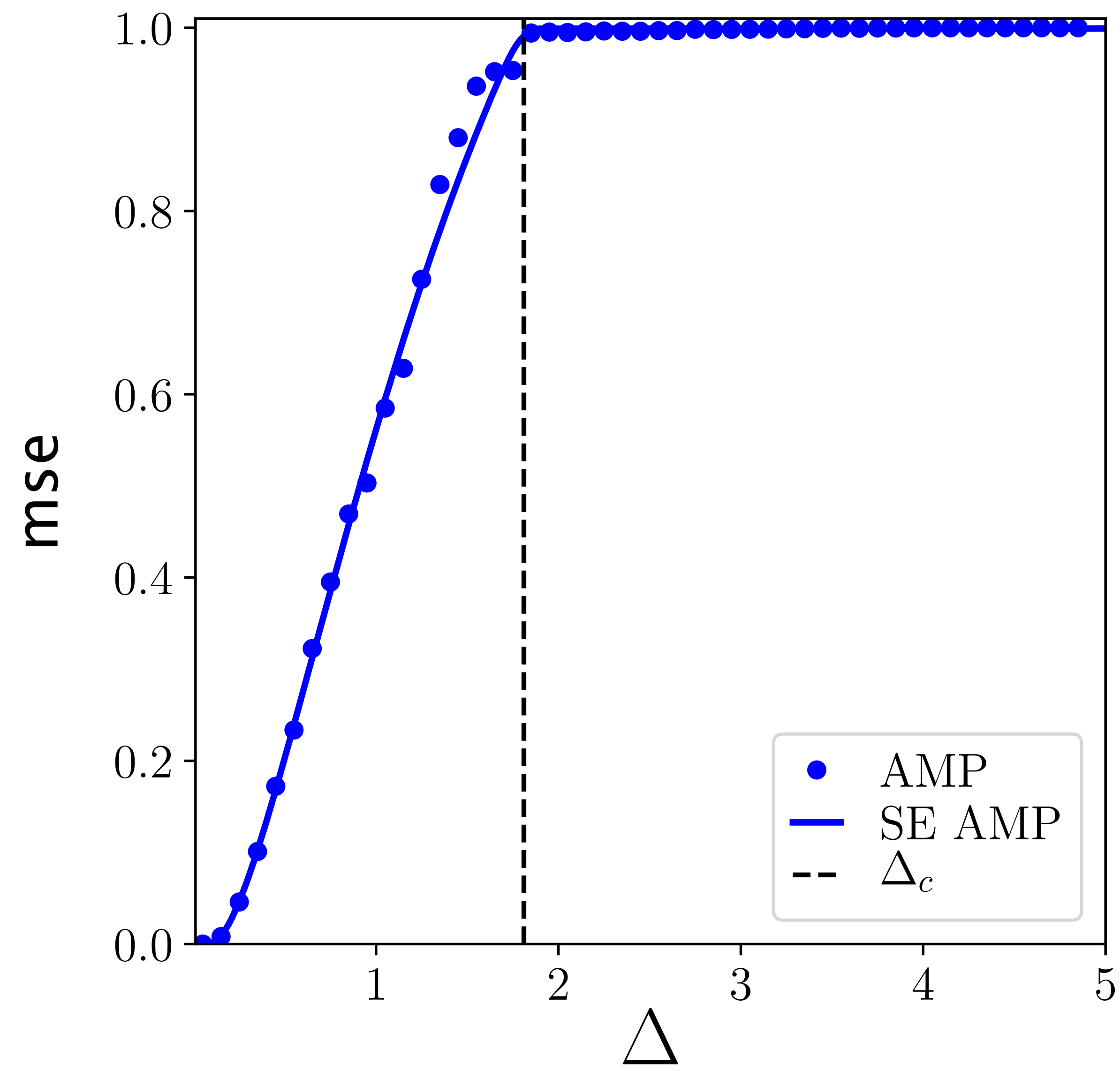
Keywords: Probabilistic programming, Bethe entropy, State Evolution, Expectation Propagation

https://sphinxteam.github.io/tramp.docs

arXiv2004.01571

Surprise

$$\mathbf{x} = \mathrm{sign}\left(\mathbf{W}\mathbf{z}\right)$$



No statistical-to-algorithmic gap!

Non-asymptotic: [Cocola, Hand, Voroninski 21']

Spectral method

PCA

Input: $\mathbf{Y} \in \mathbb{R}^{\mathrm{p} \times \mathrm{p}}$

Output: $\hat{\mathbf{v}} = \sigma_{\max}(\mathbf{Y})$

Spectral method

PCA

Input: $\mathrm{Y} \in \mathbb{R}^{\mathrm{p}\times\mathrm{p}}$

Output: $\hat{\mathbf{v}} = \sigma_{\max}(\mathrm{Y})$

L-AMP

Input: $\mathrm{Y} \in \mathbb{R}^{\mathrm{p}\times\mathrm{p}}$, prior P_v

Output: $\hat{\mathbf{v}} = \sigma_{\max}\left(K_p\left(\mathrm{Y} - \mathrm{I}_{\mathrm{p}}\right)\right)$

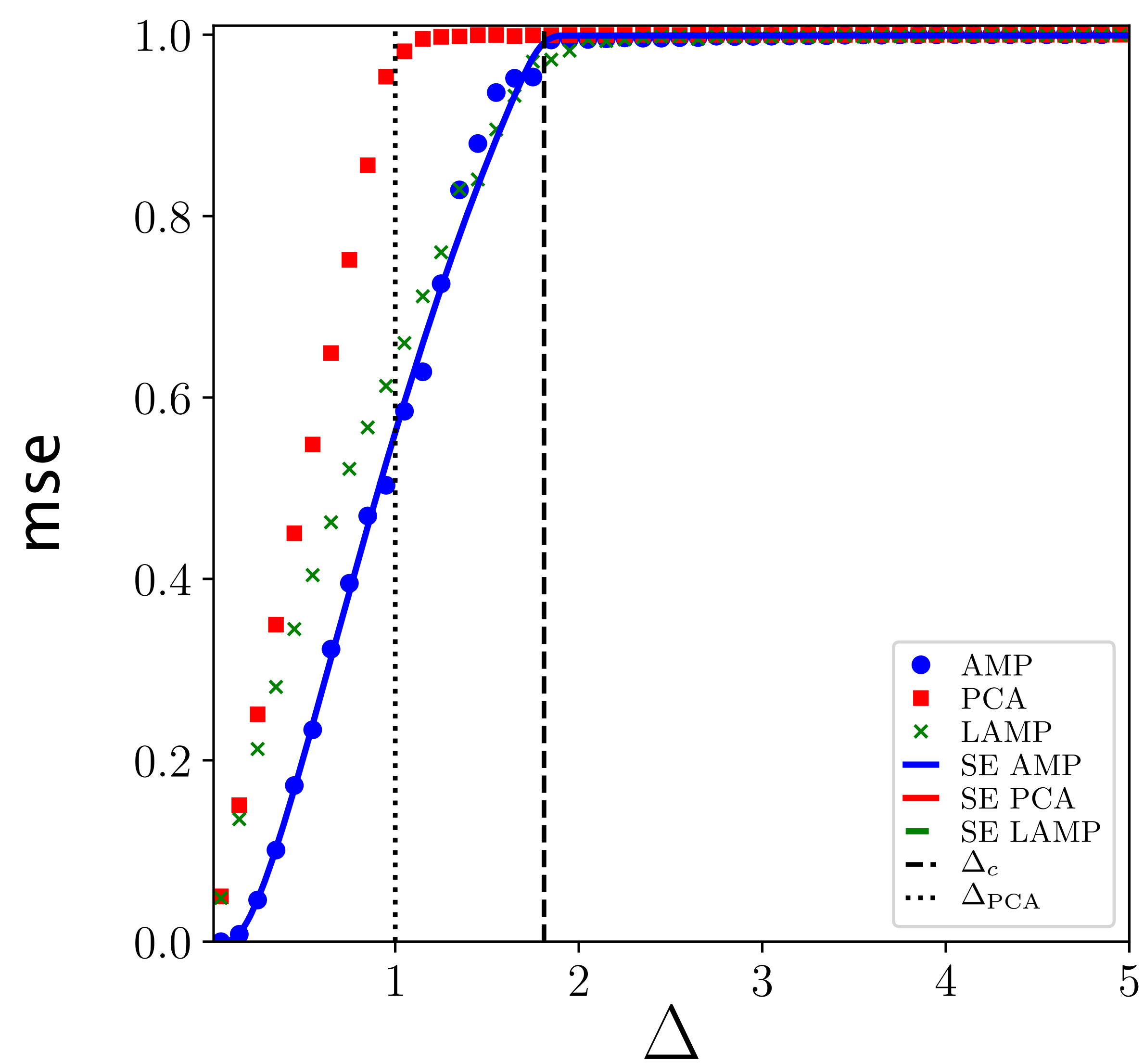
$$K_p = \mathbb{E}_{P_v}\left[\mathbf{v}\mathbf{v}^\top\right]$$

Example: for a linear layer $K_p = \dfrac{WW^\top}{p}$

for a sign layer $K_p = \left(1 - \dfrac{2}{\pi}\right) I_p + \dfrac{2}{\pi}\dfrac{WW^\top}{p}$

L-AMP

$$\mathbf{x} = \text{sign}\left(\mathbf{W}\mathbf{z}\right)$$

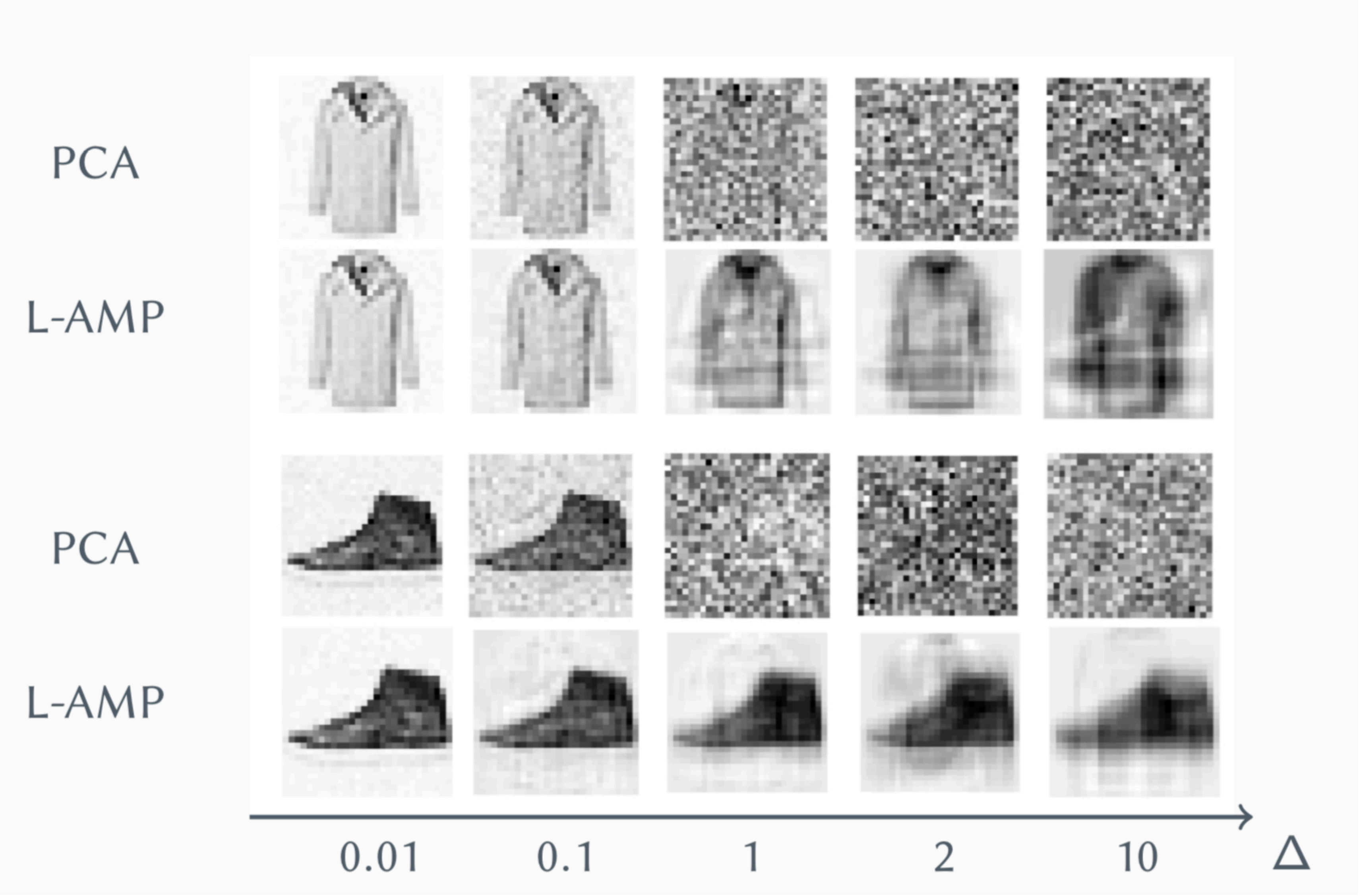


Same recovery threshold!

LAMP on Fashion MNIST

Spikes $\{x^\mu\}$ drawn from fashion MNIST

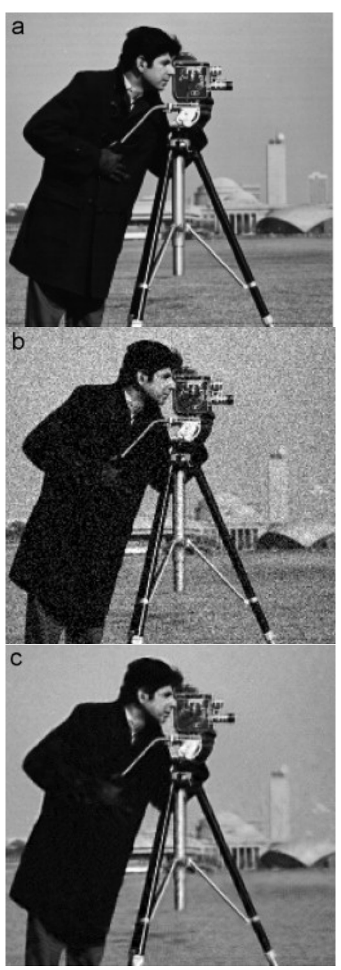
Use empirical covariance: $K_p = \mathbb{E}\left[xx^\top\right] \approx \frac{1}{m}\sum_{\mu=1}^{m} x^\mu {x^\mu}^\top$



Other problems

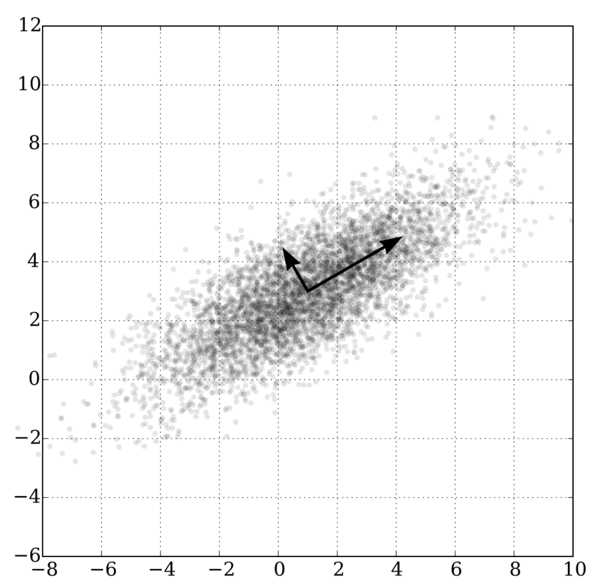
Denoising

$$\mathbf{y} = \mathbf{x} + \xi$$



Matrix factorisation

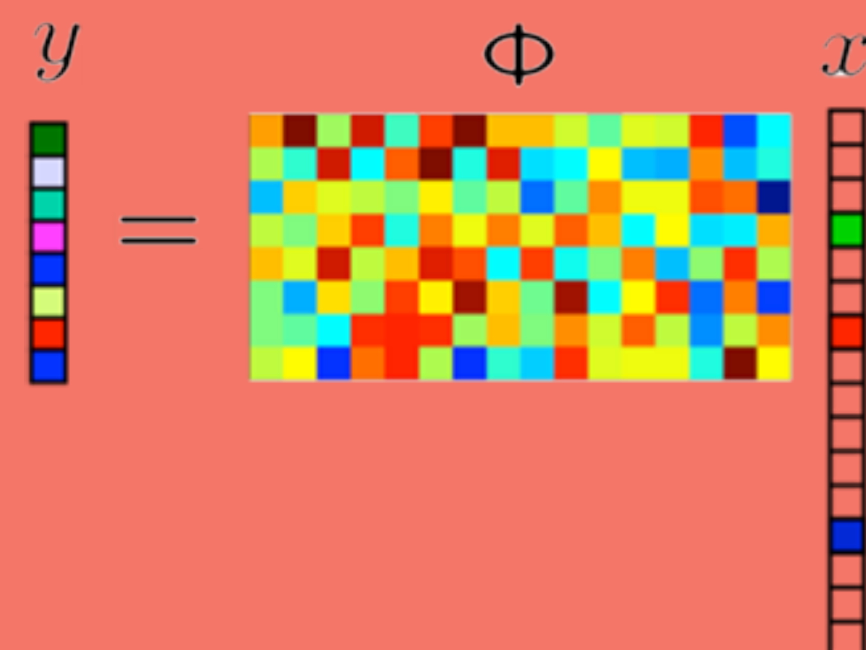
$$\mathbf{Y} = \mathbf{x}\mathbf{x}^\intercal + \mathbf{Z}$$



Compressed sensing
Phase retrieval

$$\mathbf{y} = \Phi\mathbf{x} + \xi$$

$$\mathbf{y} = |\mathbf{A}\mathbf{x}| + \xi$$



Generalised linear estimation

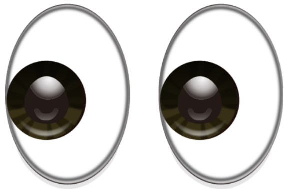
Signal — Linear channel — Activation

$$x^\star \sim P_x \longrightarrow \boxed{\mathrm{A}x^\star} \longrightarrow \boxed{\varphi(\mathrm{A}x^\star)} \longrightarrow y \in \mathbb{R}^n$$ 👀

Generalised linear estimation

Signal | Linear channel | Activation

$$x^\star \sim P_x \longrightarrow \boxed{\mathrm{A}x^\star} \longrightarrow \boxed{\varphi(\mathrm{A}x^\star)} \longrightarrow y \in \mathbb{R}^n$$

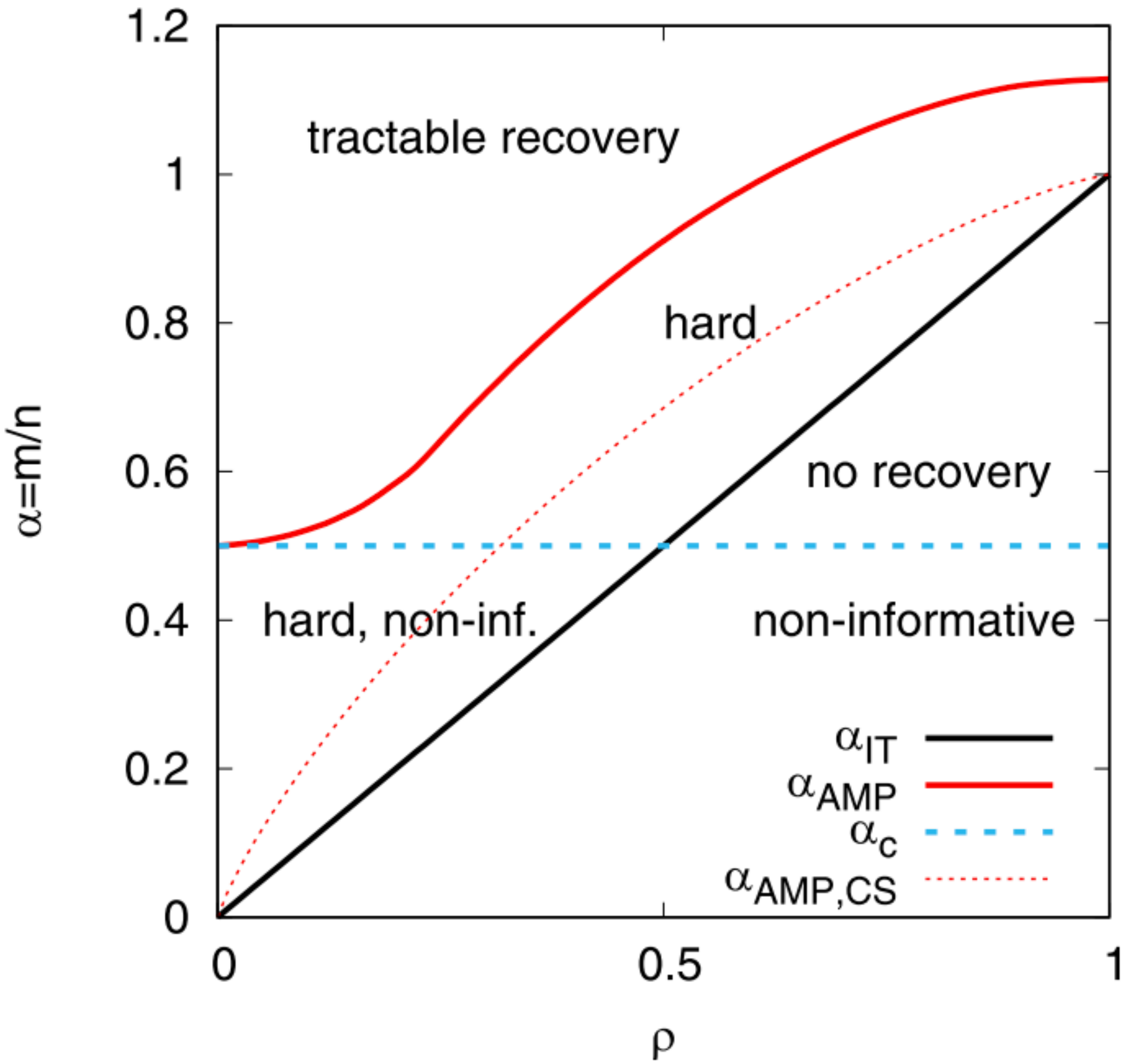


Goal: reconstruct $x^\star$ from knowledge of P_x, A, σ
in the high-dimensional regime
$n, p \to \infty$ and $n/p = O(1)$

Sparse case

$$x^\star \sim \prod_{i=1}^{p} [(1-\rho)\delta_0 + \rho\mathcal{N}(0,1)]$$

$$y = |\mathrm{A}x^\star|$$

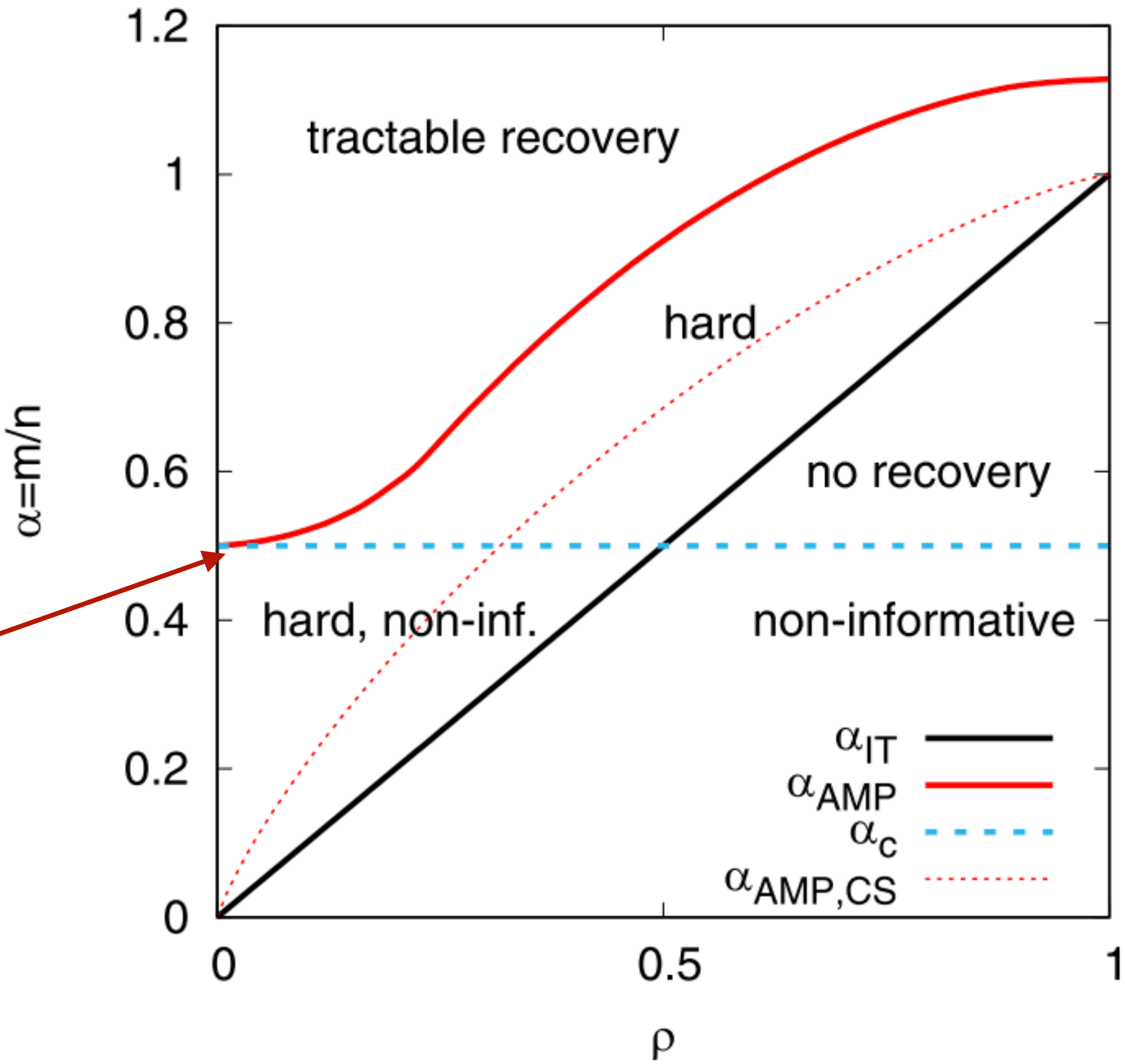


[Barbier et al. 16']

Sparse case

$$x^\star \sim \prod_{i=1}^{p} [(1-\rho)\delta_0 + \rho\mathcal{N}(0,1)]$$

$$y = |Ax^\star|$$



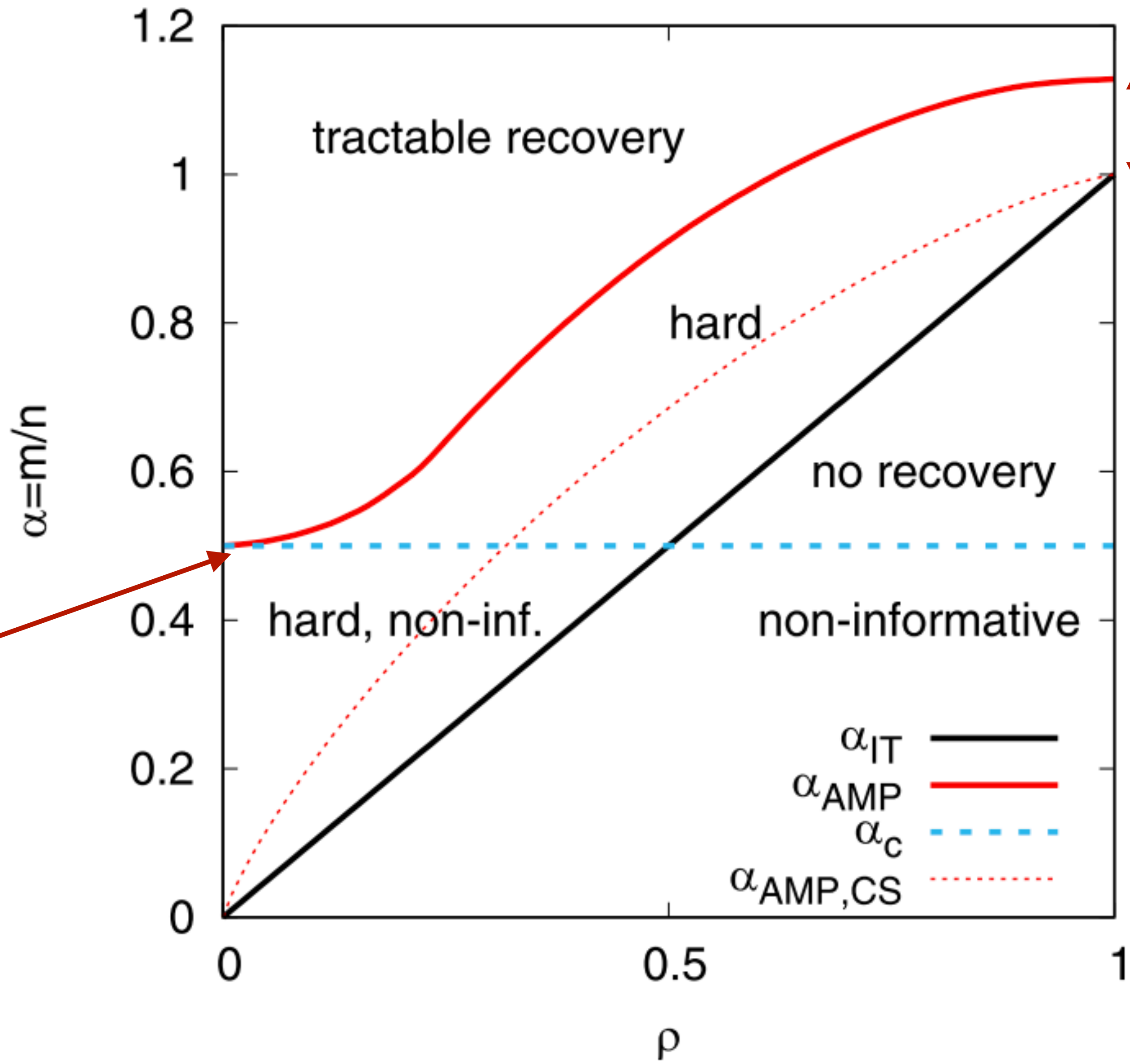
Hard even for
$\rho \to 0^+$

[Barbier et al. 16']

Sparse case

$$x^\star \sim \prod_{i=1}^{p} [(1-\rho)\delta_0 + \rho\mathcal{N}(0,1)]$$

$$y = |Ax^\star|$$



[Song, Zadik, Bruna 21']

Hard even for
$\rho \to 0^+$

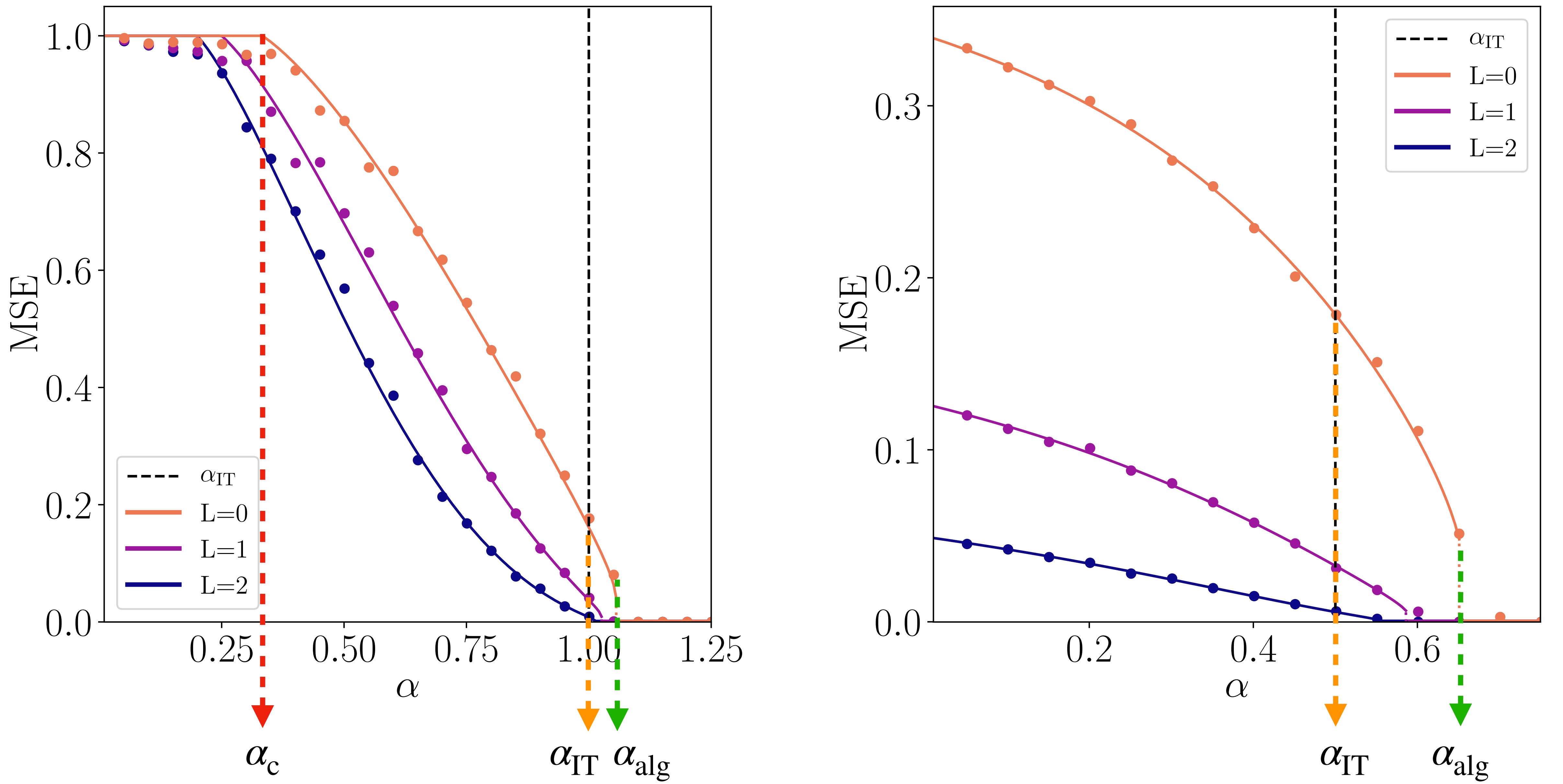
[Barbier et al. 16']

Generative network

$$\mathbf{x}^\star = \sigma^{(L)}\left(\mathrm{W}^{(\mathrm{L})}\sigma^{(\mathrm{L}-1)}\left(\mathrm{W}^{(\mathrm{L}-1)}\dots\sigma^{(1)}\left(\mathrm{W}^{(1)}\mathbf{z}\right)\cdots\right)\right)$$

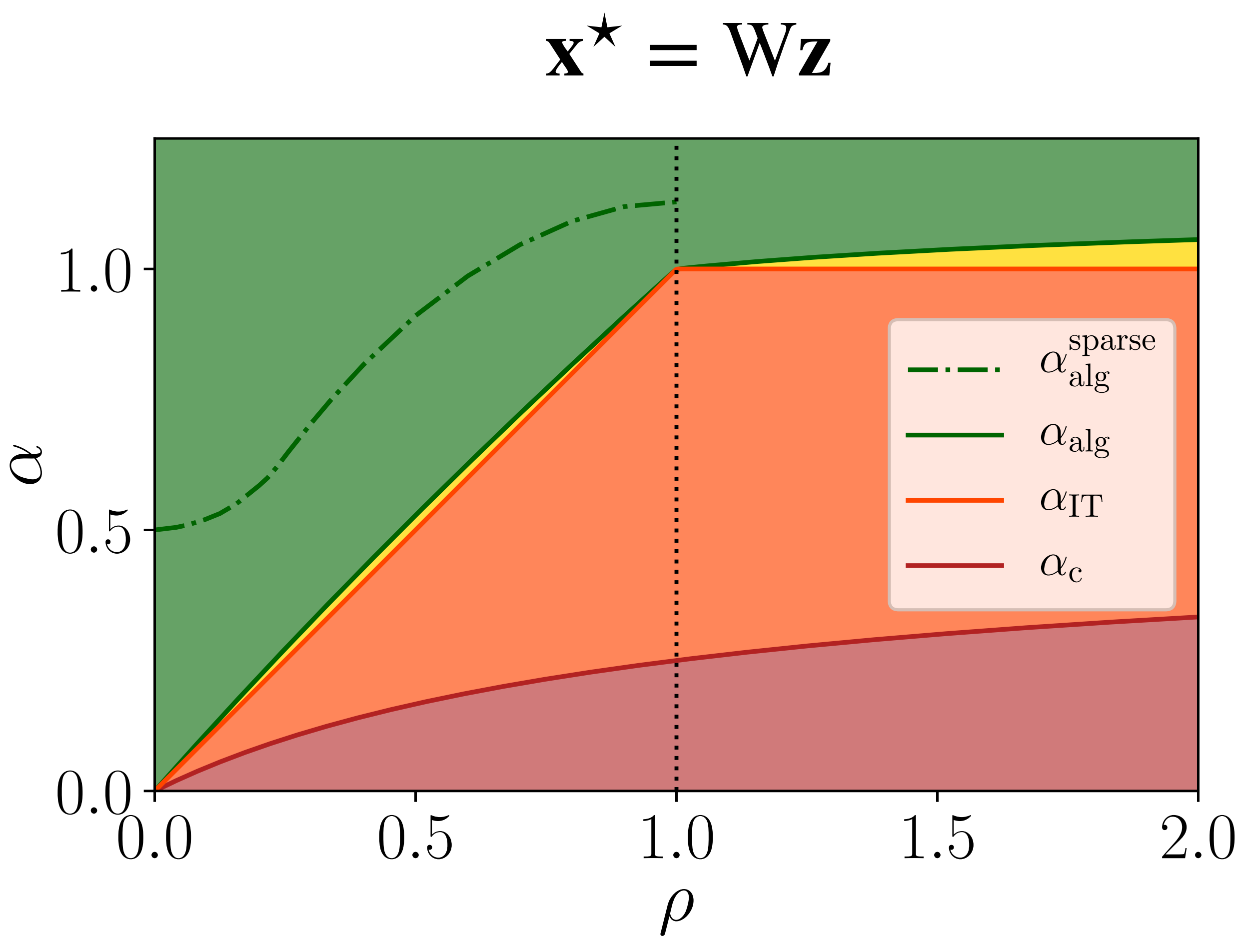
$$y = |\mathrm{A}\mathrm{x}^\star|$$

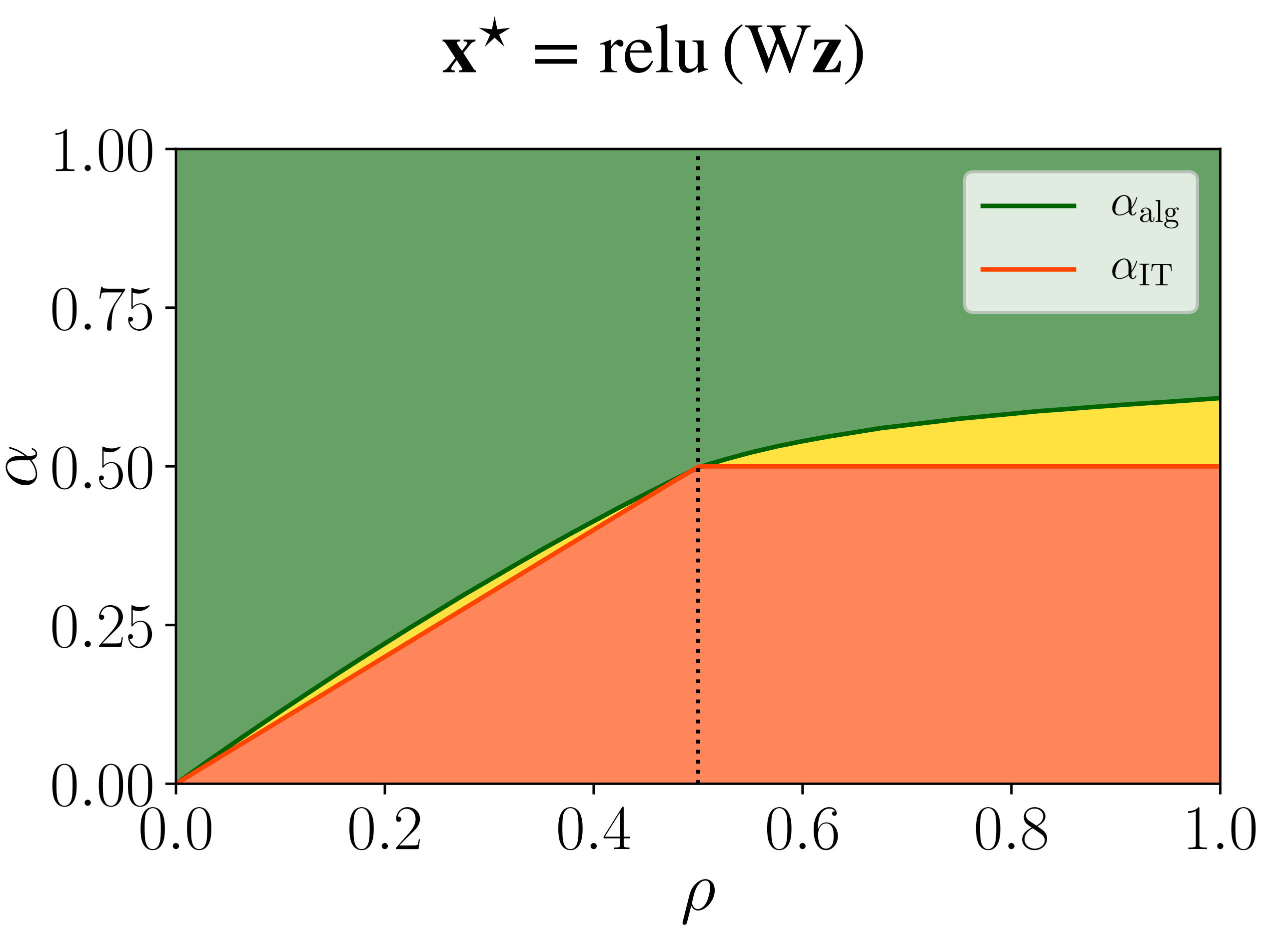
$\sigma(x) = x$ $\qquad$ $\sigma(x) = \mathrm{relu(x)}$



Generative network

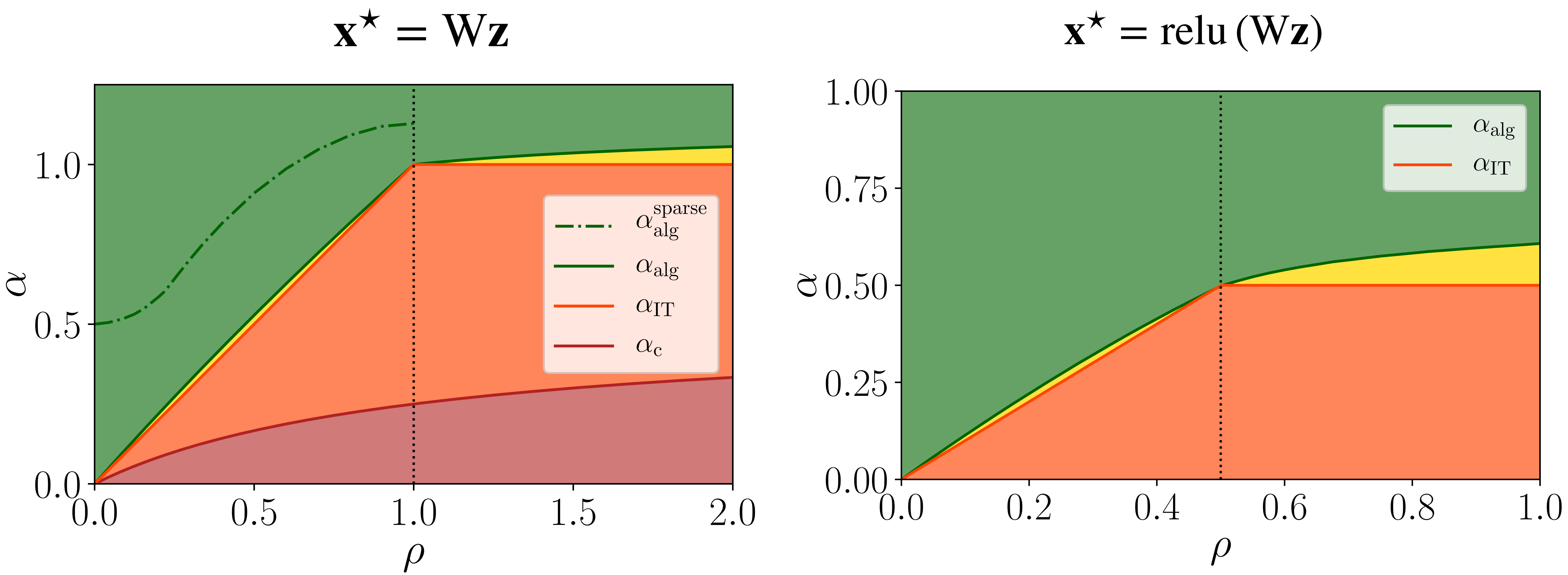
$$y = |\mathrm{A}\mathbf{x}^\star|$$





Generative network

$$y = |\mathrm{A}x^\star|$$



Generative priors have a smaller algorithmic gap!

c.f. [Hand, Leong, Voroninski 18']

Conclusions

Generative priors have comparatively smaller algorithmic gaps than their sparse counterparts

Algorithmic gap can be reduced with both depth and compression

Conclusions

Generative priors have comparatively smaller algorithmic gaps than their sparse counterparts

Algorithmic gap can be reduced with both depth and compression

Are generative priors the new sparsity?

or

Generative priors are better than sparsity?

Thank you

Based on:
arXiv: 1905.12385,
1912.02008

Selected references

Eric W Tramel, Andre Manoel, Francesco Caltagirone, Marylou Gabrié, and Florent Krzakala.
Inferring sparsity: *Compressed sensing using generalized restricted Boltzmann machines*.
In 2016 IEEE InformationTheory Workshop (ITW), pages 265–269. IEEE, 2016

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis.
Compressed sensing using generative models.
In Proceedings of the 34th International Conference on Machine Learning-Volume 70,
pages 537–546.JMLR. org, 2017

Soledad Villar.
Generative models are the new sparsity?
https://solevillar.github.io/2018/03/28/SUNLayer.html, 2018.

Jean Barbier, Mohamad Dia, Nicolas Macris, Florent Krzakala, Thibault Lesieur, and Lenka Zdeborová.
Mutual information for symmetric rank-one matrix estimation: A proof of the replica formula.
In Advancesin Neural Information Processing Systems, pages 424–432, 2016

Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová.
Constrained low-rank matrix estimation: phase transitions, approximate message passing and applications.
Journal of Statistical Mechanics: Theory and Experiment, 2017(7):073403, 2017.

Selected references

Jean Barbier and Nicolas Macris.
The adaptive interpolation method: a simple scheme to prove replica formulas in bayesian inference.
Probability Theory and Related Fields, pages 1–53, 2018.

Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová.
Optimal errors and phase transitions in high-dimensional generalized linear models.
Proceedings of the National Academy of Sciences, 116(12):5451–5460, 2019.

M. Gabrié, A. Manoel, C. Luneau, J. Barbier, N. Macris, F. Krzakala, and L. Zdeborová.
Entropy and mutual information in models of deep neural networks.
In Advances in Neural Information Processing Systems 31

Jinho Baik, Gérard Ben Arous, Sandrine Péché, et al.
Phase transition of the largest eigenvalue for non null complex sample covariance matrices.
The Annals of Probability, 33(5):1643–1697, 2005.

Andre Manoel, Florent Krzakala, Marc Mézard, and Lenka Zdeborová.
Multi-layer generalized linear estimation.
In 2017 IEEE International Symposium on Information Theory (ISIT),
pages 2098–2102. IEEE,2017.

Recent development

On the Cryptographic Hardness of Learning Single Periodic Neurons

Min Jae Song*[a], Ilias Zadik*[b], and Joan Bruna[a,b]

[a]Courant Institute of Mathematical Sciences, New York University, New York
[b]Center for Data Science, New York University, New York

June 22, 2021

Moreover, we demonstrate the *necessity of noise* in the hardness result by designing a polynomial-time algorithm for learning certain families of such functions under exponentially small adversarial noise. Our proposed algorithm is not a gradient-based or an SQ algorithm, but is rather based on the celebrated Lenstra-Lenstra-Lovász (LLL) lattice basis reduction algorithm. Furthermore, in the absence of noise, this algorithm can be directly applied to solve CLWE detection (Bruna et al.'21) and phase retrieval with an optimal sample complexity of $d + 1$ samples. In the former case, this improves upon the quadratic-in-d sample complexity required in (Bruna et al.'21). In the latter case, this improves upon the state-of-the-art AMP-based algorithm, which requires approximately $1.128d$ samples (Barbier et al.'19).

See also [Andoni, Hsu, Shi, Sun 17']

AMP for one-layer prior

Input: $Y \in \mathbb{R}^{p\times p}$ and $W \in \mathbb{R}^{p\times k}$:

Initialize with: $\hat{\mathbf{v}}^{t=1} = \mathcal{N}(\mathbf{0}, \sigma^2 I_p)$, $\hat{\mathbf{z}}^{t=1} = \mathcal{N}(\mathbf{0}, \sigma^2 I_k)$, and $\hat{\mathbf{c}}_v^{t=1} = I_p$, $\hat{\mathbf{c}}_z^{t=1} = I_k$, $t = 1$.

repeat

Spiked layer ***denoising***:

$$\mathbf{B}_v^t = \frac{1}{\Delta}\frac{Y}{\sqrt{p}}\hat{\mathbf{v}}^t - \frac{1}{\Delta}\frac{\left(I_p^\top \hat{\mathbf{c}}_v^t\right)}{p}\hat{\mathbf{v}}^{t-1} \quad \text{and} \quad A_v^t = \frac{1}{\Delta p}(\|\hat{\mathbf{v}}^t\|_2)^2 I_p.$$

Generative layer ***denoising***:

$$V^t = \frac{1}{k}\left(I_k^\top \hat{\mathbf{c}}_z^t\right) I_p, \quad \boldsymbol{\omega}^t = \frac{1}{\sqrt{k}} W\hat{\mathbf{z}}^t - V^t \mathbf{g}^{t-1} \quad \text{and} \quad \mathbf{g}^t = f_{\text{out}}\left(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t\right),$$

$$\Lambda^t = \frac{1}{k}\|\mathbf{g}^t\|_2^2 t I_k \quad \text{and} \quad \gamma^t = \frac{1}{\sqrt{k}} W^\top \mathbf{g}^t + \Lambda^t \hat{\mathbf{z}}^t.$$

Marginals estimation:

$$\hat{\mathbf{v}}^{t+1} = f_v(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t) \quad \text{and} \quad \hat{\mathbf{c}}_v^{t+1} = \partial_B f_v(\mathbf{B}_v^t, A_v^t, \boldsymbol{\omega}^t, V^t),$$

$$\hat{\mathbf{z}}^{t+1} = f_z(\gamma^t, \Lambda^t) \quad \text{and} \quad \hat{\mathbf{c}}_z^{t+1} = \partial_\gamma f_z(\gamma^t, \Lambda^t),$$

$t = t + 1$.

until Convergence.

Output: $\hat{\mathbf{v}}$, $\hat{\mathbf{z}}$.

Aubin, BL, Maillard, Krzakala, Zdeborova, arXiv:1905.12385

Replica symmetric formula

For a L-layer deep, feed-forward, random generative prior, the *free entropy*

$$\Phi = \operatorname*{extr}_{q_x, \hat{q}_x, \{q_l, \hat{q}_l\}} \left\{ -\frac{1}{2}\hat{q}_x q_x - \frac{\rho}{2}\sum_{l=1}^{L} \beta_l q_l \hat{q}_l + \alpha \Psi_y\left(q_x\right) + \rho \sum_{l=2}^{L} \beta_l \Psi_{\text{out}}^{(l)}\left(\hat{q}_l, q_{l-1}\right) + \Psi_{\text{out}}^{(L+1)}(\hat{q}_x, q_L) + \rho \Psi_z\left(\hat{q}_z\right) \right\}$$

$$\Psi_{\text{out}}^{(l)}(r,s) = \mathbb{E}_{\xi,\eta}\left[\mathcal{Z}_{\text{out}}^{(l)}(\sqrt{r}\xi, r, \sqrt{s}\xi, \rho_{l-1} - s)\log \mathcal{Z}_{\text{out}}^{(l)}(\sqrt{r}\xi, r, \sqrt{s}\xi, \rho_{l-1} - s)\right]$$

$$\Psi_z(t) = \mathbb{E}_{\xi}\left[\mathcal{Z}_z(\sqrt{t}\xi, t)\log \mathcal{Z}_z(\sqrt{t}\xi, t)\right]$$

$$Q_{\text{out}}^{(l)}(x, z; B, A, \omega, V) = \frac{e^{-\frac{A}{2}x^2 + Bx}}{\mathcal{Z}_{\text{out}}(B, A, \omega, V)} \frac{e^{-\frac{1}{2V}(z-\omega)^2}}{\sqrt{2\pi V}} P_{\text{out}}^{(l)}(x \mid z)$$

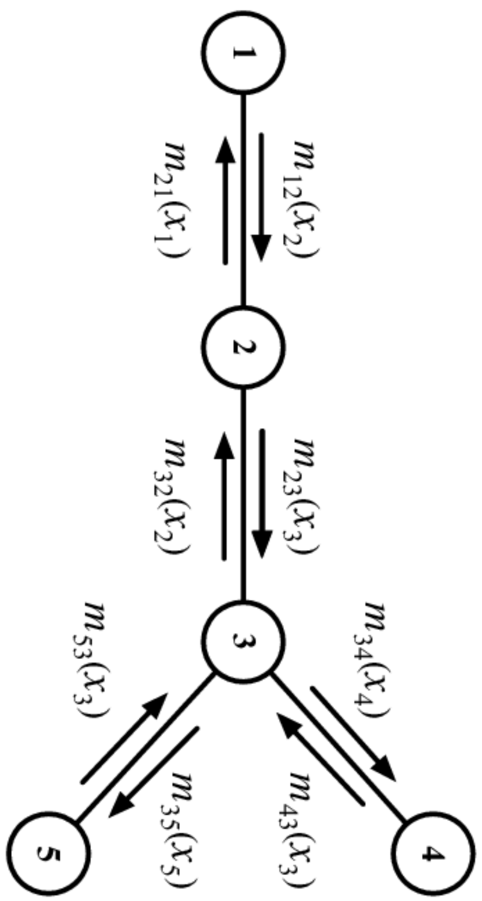
$$Q_z(z; B, A) = \frac{e^{-\frac{A}{2}z^2 + Bz}}{\mathcal{Z}_z(B, A)} P_z(z)$$

Message Passing

This problem screams for a Approximated Message Passing algorithm!

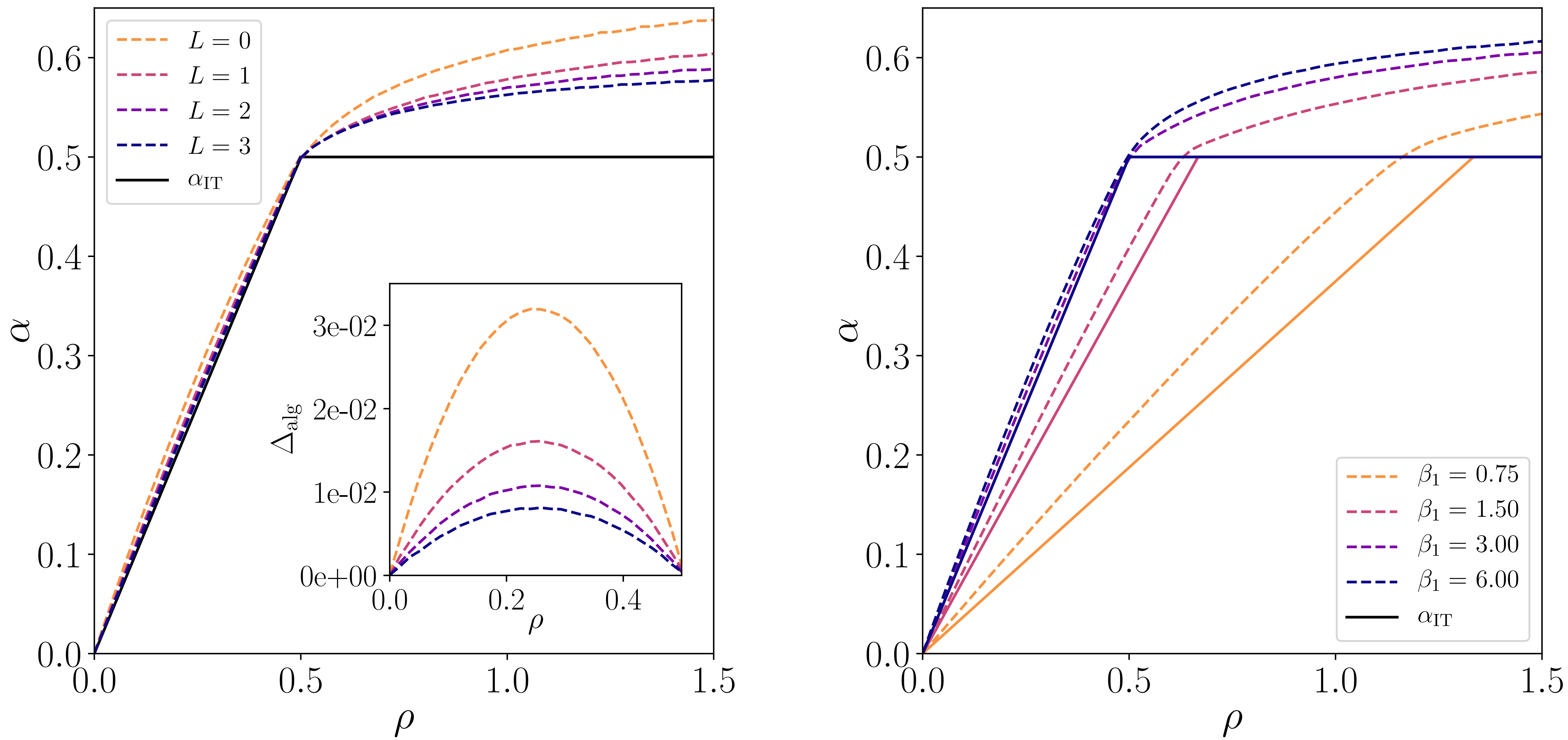
Key facts about AMP:

- Polynomial time iterative algorithm that approximate marginals from the posterior
- Can be derived systematically from Belief propagation.
- Its performance can be tracked analytically.
- Best well-known polynomial time algorithm for a large class of random problems.



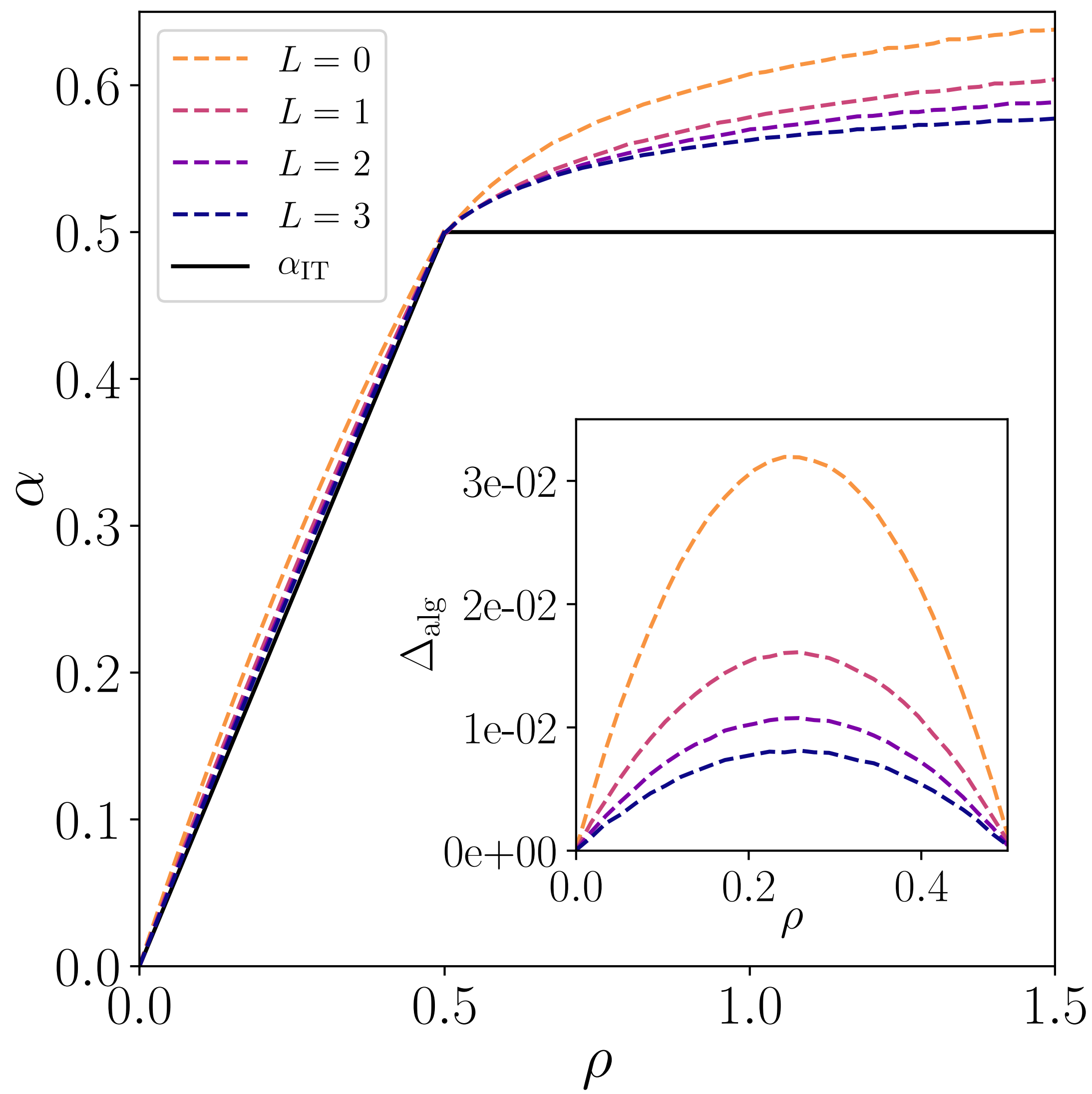
Role of depth / width

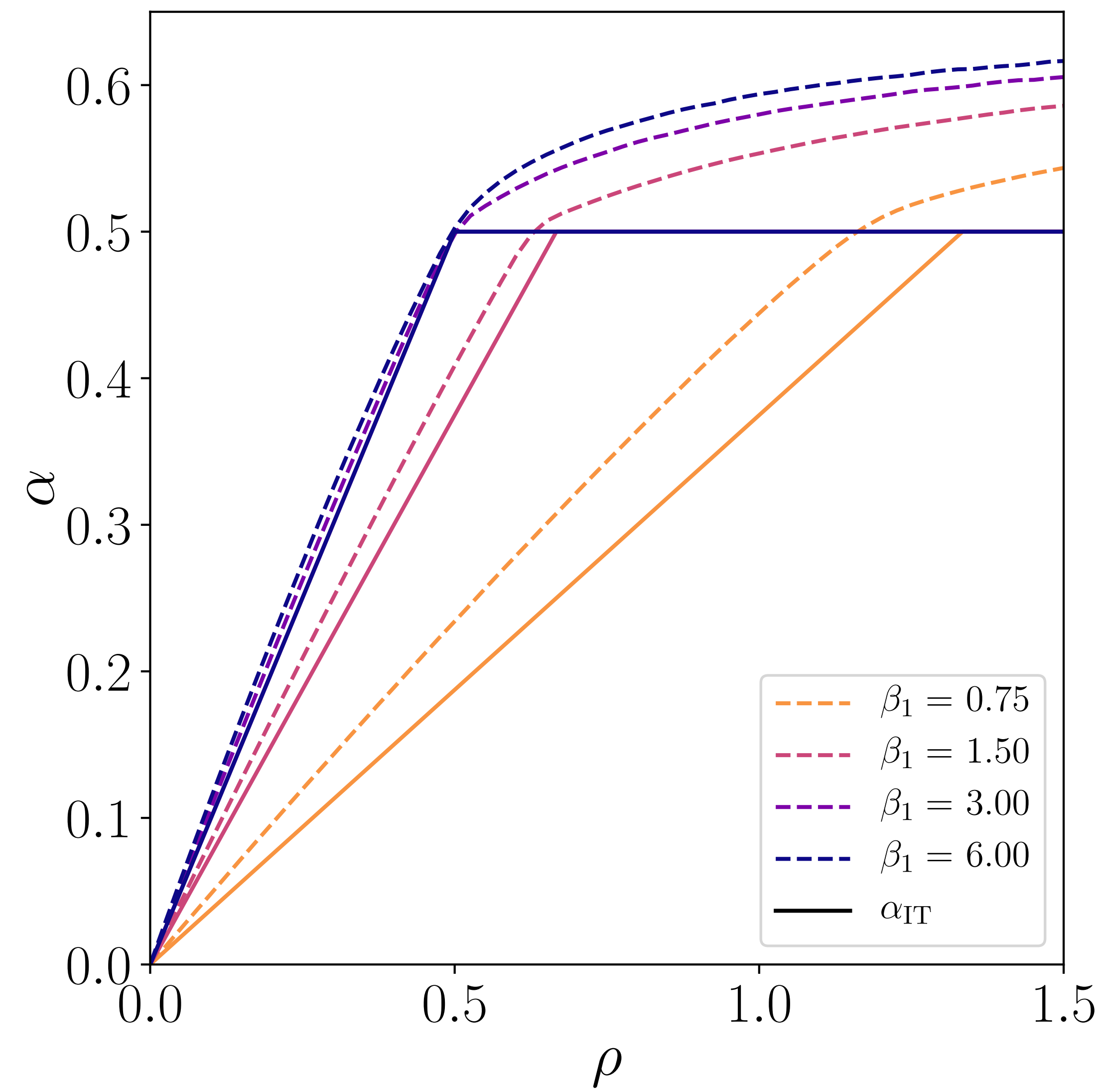
$$y = |A\mathbf{x}^\star| \qquad \mathbf{x}^\star = \mathrm{relu}(\mathbf{W}\mathbf{z})$$



Role of depth / width

$$y = |Ax^\star| \qquad \mathbf{x}^\star = \mathrm{relu}(\mathbf{W}\mathbf{z})$$





Gap can be made smaller with increasing depth and compression

[Bora et al 17'] [Hand et al 18']