Partially Linearized Update for Generative Inversion in Compressive Sensing

PLUGIn-CS: A simple algorithm for compressive sensing with generative prior

THE UNIVERSITY OF BRITISH COLUMBIA

Babhru Joshi, Xiaowei Li, Yaniv Plan, Ozgur Yilmaz

Overview

We extend PLUGIn (a novel algorithm which can invert deep generative models) to compressive sensing and introduce the algorithm PLUGIn-CS, which can invert deep generative models from compressive observations.

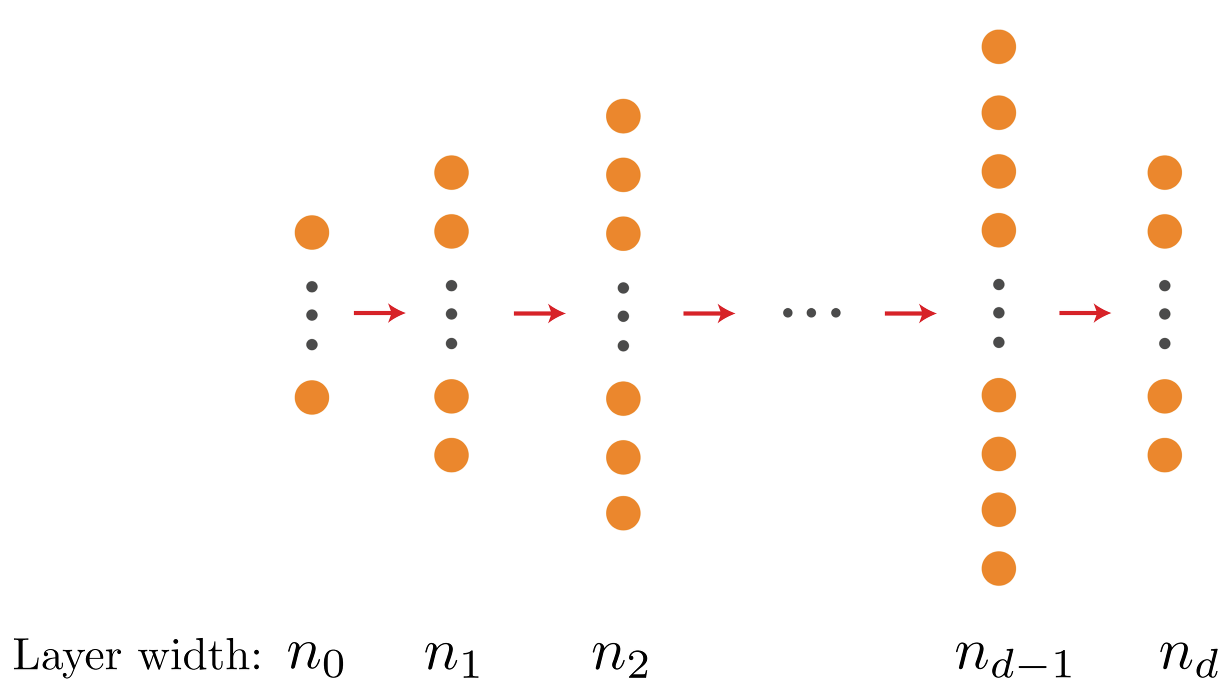
- PLUGIn-CS has a simple form and does not require gradient calculation.
- PLUGIn-CS has provable global geometric convergence[1].
- Our analysis for PLUGIn-CS allows some contractive layers in network.
- Numerical results show that PLUGIn-CS can effectively recover images from compressive measurements.

Problem formulation

- A deep generative model with ReLU activation functions:

$$\mathcal{G}(x) = \sigma(A_d\sigma(A_{d-1}\ldots\sigma(A_1 x)\ldots))$$

where $A_i \in \mathbb{R}^{n_i \times n_{i-1}}$.



- Given a noisy compressive observation:

$$y = \Phi\mathcal{G}(x^*) + \epsilon, \quad \Phi \in \mathbb{R}^{m \times n_d}.$$

Goal: Recover latent code x^* and/or signal $\mathcal{G}(x^*)$.

The algorithm

A natural way for recovery is to solve the following non-convex optimization problem using gradient descent

$$\min_{x\in\mathbb{R}^{n_0}} \|\Phi\mathcal{G}(x) - y\|^2.$$

The gradient descent update (with step size η) is given by

$$x^{k+1} = x^k - \eta(D_1A_1)^\intercal \cdots (D_dA_d)^\intercal \Phi^\intercal \left(\Phi\mathcal{G}(x^k) - y\right)$$

where each D_i is a diagonal 0 or 1 matrix *depending* on x^k and A_j $(j \le i)$.

Similar to PLUGIn, we linearize $(D_iA_i)^\intercal$ to $A_i^\intercal$ and obtain

$$\textbf{PLUGIn-CS:} \quad x^{k+1} = x^k - \eta A_1^\intercal A_2^\intercal \cdots A_d^\intercal \Phi^\intercal \left(\Phi\mathcal{G}(x^k) - y\right)$$

- PLUGIn-CS becomes PLUGIn if Φ is identity matrix.
- $A_1^\intercal A_2^\intercal \cdots A_d^\intercal$ is *static* (does not depend on x^k). No gradient calculation required.
- If Φ and A_i are i.i.d. Gaussian (properly normalized), first iterate of PLUGIn-CS can provide an unbiased estimate of x^* with step size $\eta = 2^d$.

Convergence results

We assume:

- Each A_i has i.i.d. $\mathcal{N}(0, 1/n_i)$ entries and are independent.
- Layer width satisfy

$$n_i \gtrsim 5^i n_0 \log\left(\prod_{j=0}^{i-1} \frac{en_j}{n_0}\right), \quad 1 \le i \le d.$$

- Φ has i.i.d. $\mathcal{N}(0, 1/m)$ entries (independent from weight matrices) with

$$m \gtrsim 2^d n_0 \log\left(\prod_{j=0}^{d} \frac{en_j}{n_0}\right).$$

- The noise ϵ does not depend on $\{A_i\}_{i\le d}$ or Φ.

Also let R be a positive number such that $\|x^0 - x^*\| \le R$.

We prove that, the k-th estimate given by PLUGIn-CS algorithm with constant step size $\eta = 2^d$ satisfies

$$\|x^k - x^*\| \le 2^{-k}R + 30 \cdot 2^d\sqrt{n_0/m}\|\epsilon\|, \text{ and}$$
$$\|\mathcal{G}(x^k) - \mathcal{G}(x^*)\| \le 2^{-k}(3R) + 90 \cdot 2^d\sqrt{n_0/m}\|\epsilon\|$$

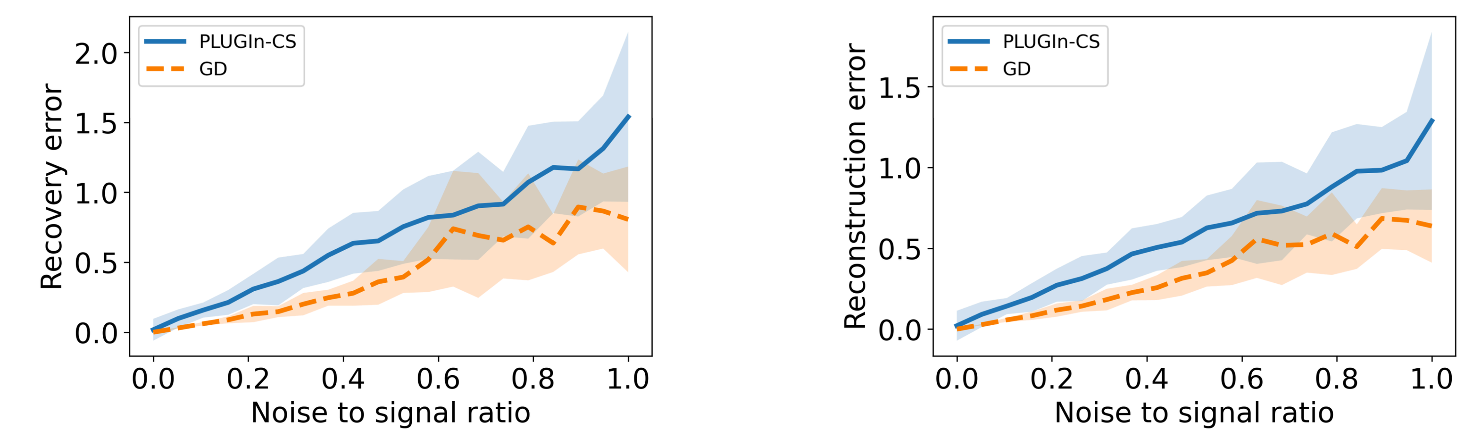
with probability at least $1 - 2(k+4)e^{-10n_0}$.

- Global and geometrical convergence for estimation errors.
- Network is expansive on "average", thus can have some contractive layers.
- PLUGIn-CS can recover latent code and signal when m/n_0 is large.
- Exact step size can be relaxed to a constant step size $\eta \le 2^d$.

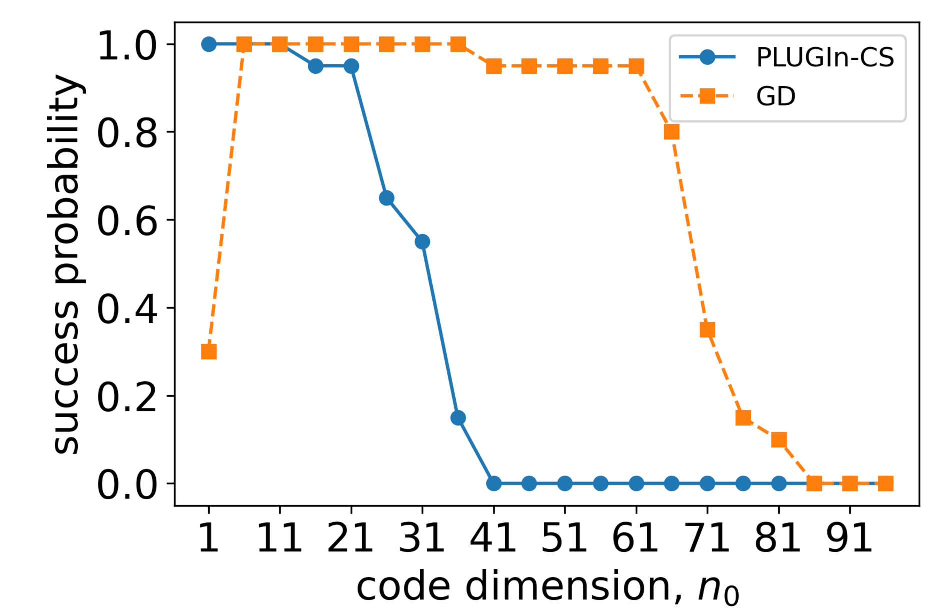
Numerical results

1) Performance of PLUGIn-CS and gradient descent (GD) on a 2-layer synthetic networks with random Gaussian weights.

(a) We compare their relative recovery error $\|\hat{x} - x^*\|/\|x^*\|$ and relative reconstruction error $\|\mathcal{G}(\hat{x}) - \mathcal{G}(x^*)\|/\|\mathcal{G}(x^*)\|$ w.r.t. noise level.

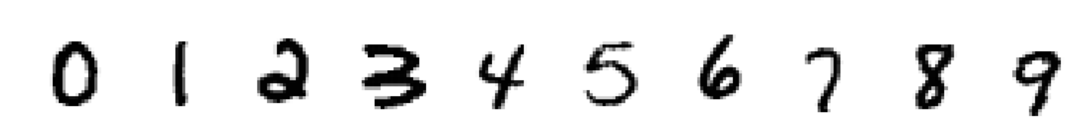


(b) We compare their success probability from 20 independent trials w.r.t. different code dimension n_0 (with $n_1 = 250$, $n_2 = 700$, $m = 150$ fixed).



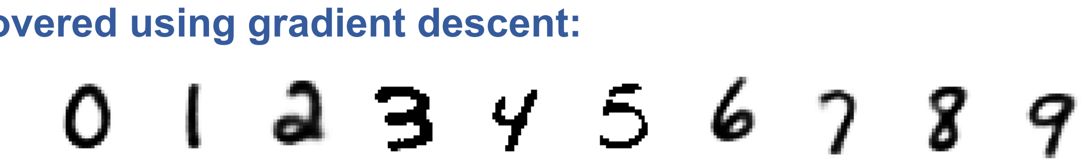
2) We trained a generative model on MNIST dataset, then took compressive measurements (with $m = 150$ and $n_d = 784$) of some images and tried PLUGIn-CS as well as gradient descent for recovery.

Original images:



Recovered using PLUGIn-CS:

Recovered using gradient descent:



1. Under assumptions listed in convergence results section.