Learning to Solve Inverse Problems in Imaging

Rebecca Willett, University of Chicago

Davis Gilton, UW-Madison

Greg Ongie, UChicago

Inverse problems in imaging

Observe: $y = X\beta + \varepsilon$

Goal: Recover β from y

- Inpainting
- Deblurring
- Superresolution
- Compressed Sensing
- MRI
- Radar

Classical approach: Tikhonov regularization (1943)

- Example: deblurring

- Least squares solution:

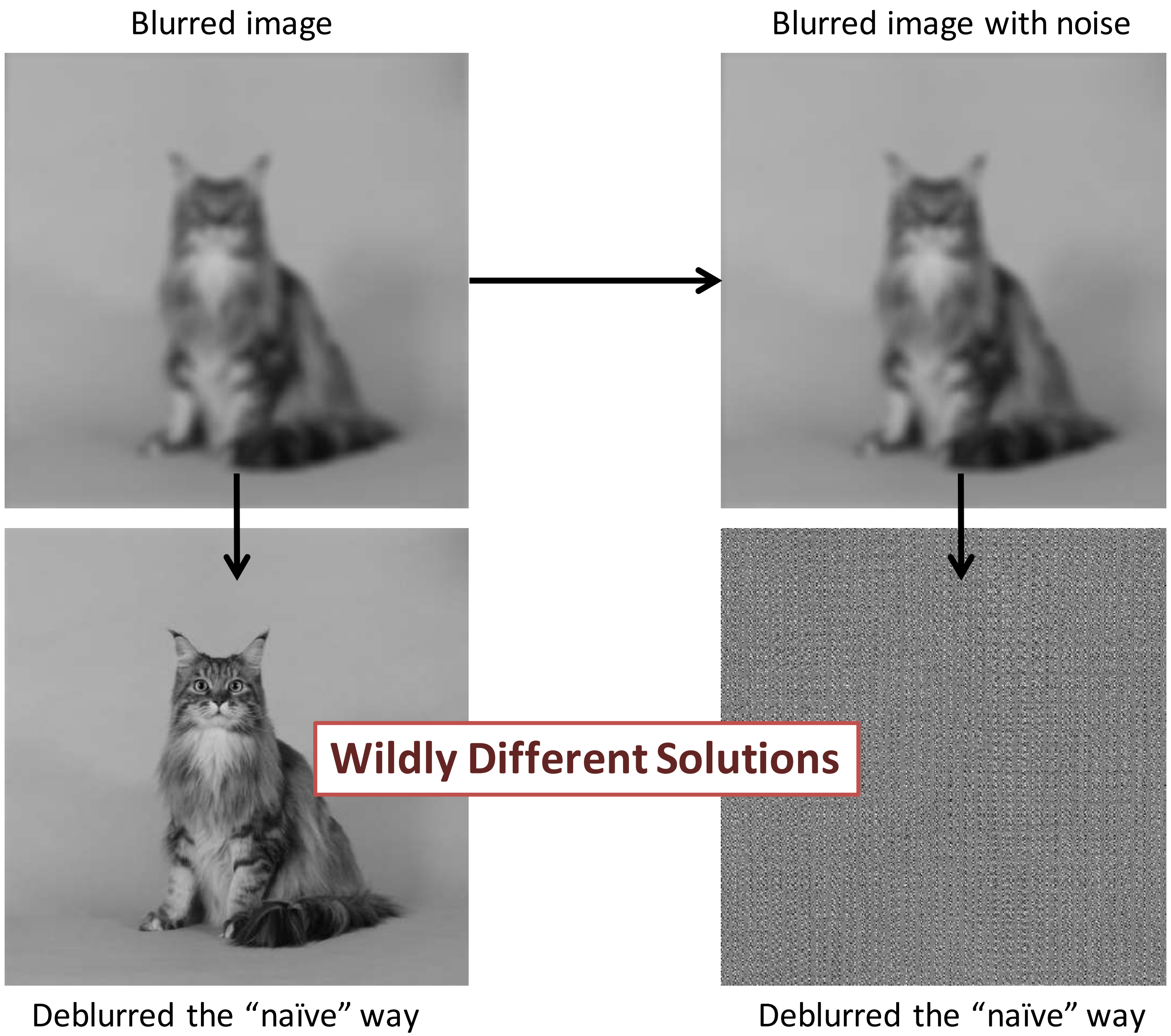
$$\widehat{\beta} = (X^\top X)^{-1} X^\top y$$

Classical approach: Tikhonov regularization (1943)

- Example: deblurring

- Least squares solution:

$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$



Classical approach: Tikhonov regularization (1943)

- Example: deblurring

- Least squares solution:

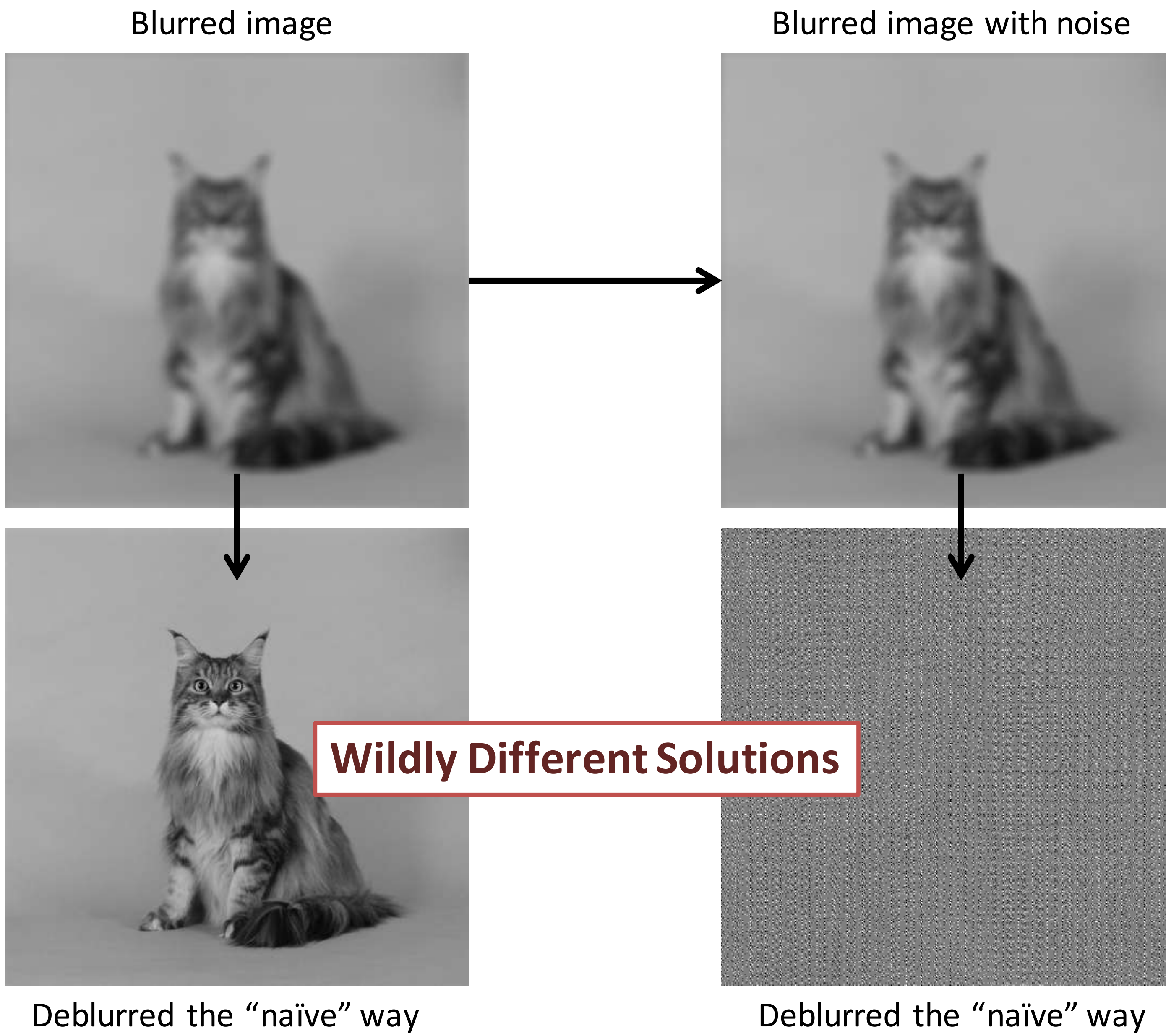
$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$

- Tikhonov regularization (aka “ridge regression”)

$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2$$

$$= (X^\top X + \lambda I)^{-1} X^\top y$$

better conditioned; suppresses noise



Classical approach: Tikhonov regularization (1943)

- Example: deblurring

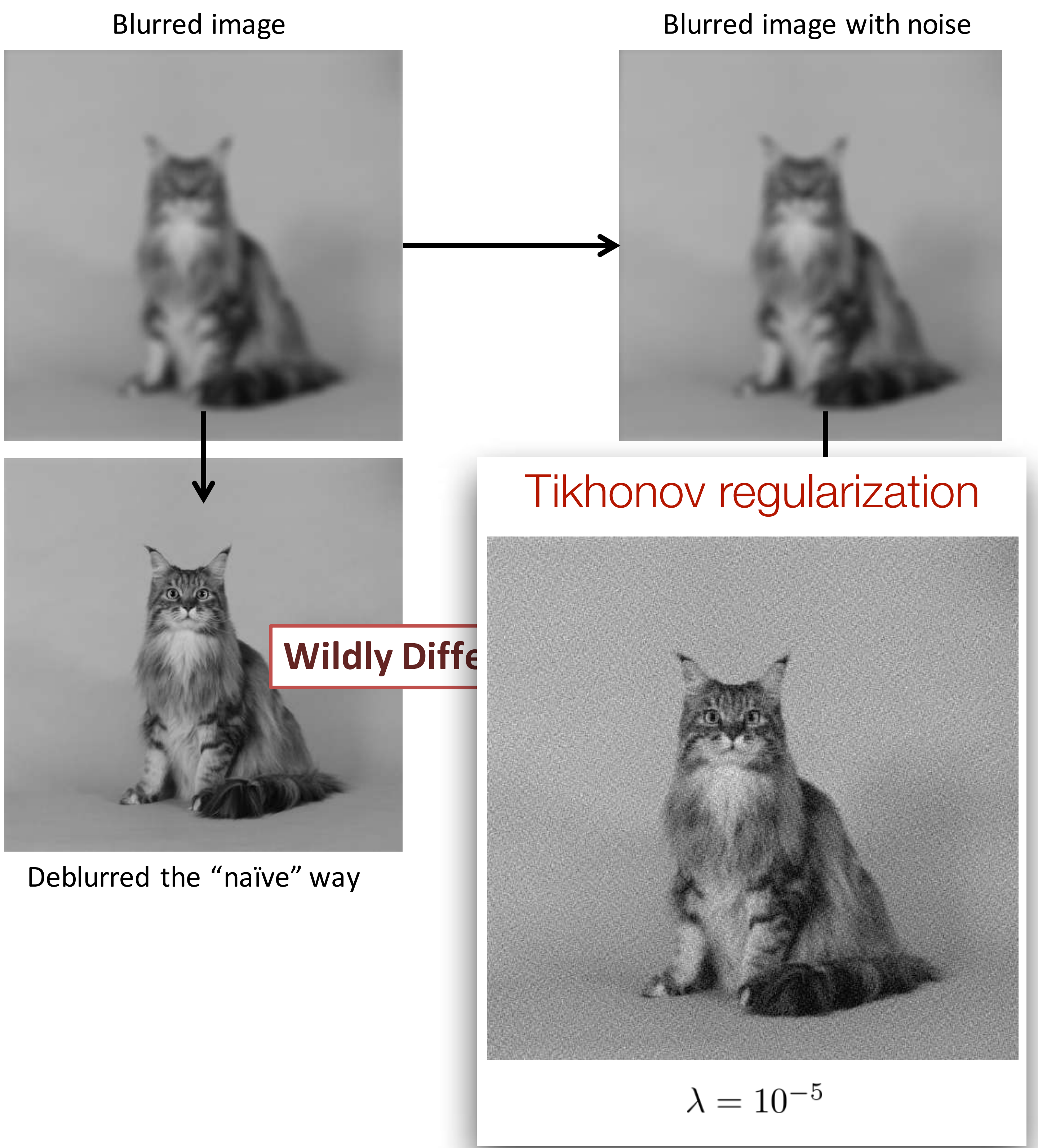
- Least squares solution:

$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$

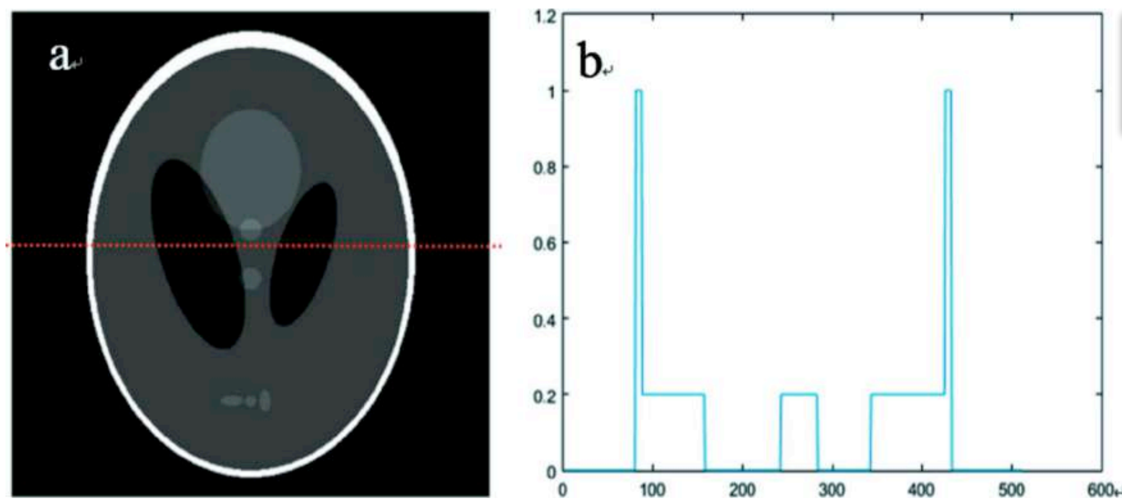
- Tikhonov regularization (aka "ridge regression")

$$\hat{\beta} = \arg\min_{\beta} ||y - X\beta||_2^2 + \lambda ||\beta||_2^2$$
$$= (X^\top X + \lambda I)^{-1} X^\top y$$

better conditioned; suppresses noise

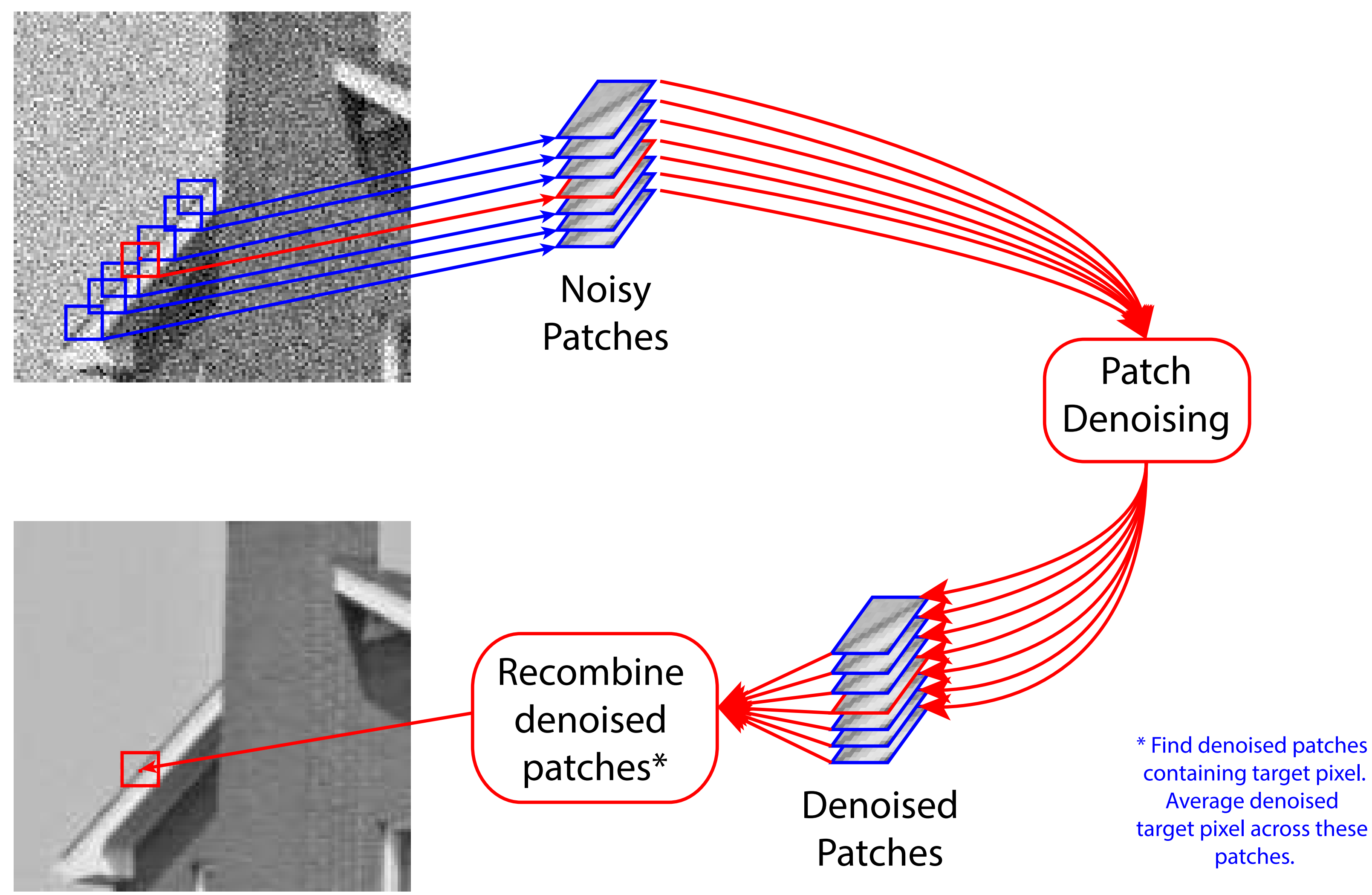


Geometric models of images

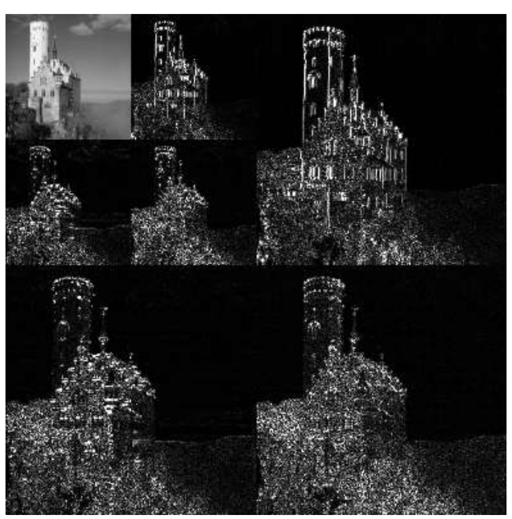


Total variation

Patch subspaces and manifolds



(Wavelet) sparsity



Regularization in inverse problems

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

Regularization in inverse problems

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

Classical: r(β) is a pre-defined smoothness-promoting regularizer (e.g. Tikhinov or ridge estimation)

Bayesian: r(β) = -log p(β)
Uses a prior distribution over space of β's
(e.g. sparsity, patch redundancy, total variation)

Learned: use training data to learn r(β)

Limitations of classical regularizers

Original

Input

CS-style recovery

Limitations of classical regularizers

Original

Input

CS-style recovery

Learned

Examples in recent literature

- Deep CNN's for signal recovery

  Dong, Loy, He, Tang, 2014
  Mousavi and Baraniuk, 2017
  Jin, McCann, Froustey, Unser, 2017
  Ye, Han, Cha, 2018

- Compressed sensing with GANs

  Bora, Jalal, Price, Dimakis, 2017

- Unrolled algorithms for solving inverse problems

  - Deep proximal gradient descent nets

    Chen, Yu, Pock, 2015
    Mardani et al, 2018

  - Deep ADMM nets

    Sun, Li, Xu, 2016
    Chang, Li, Poczos, Kumar, Sankaranarayanan, 2017

  - Deep half-quadratic splitting

    Zhang, Zuo, Gu, Zhang, 2017

  - Deep primal-dual nets

    Adler and Öktem, 2018

Classes of methods

Model Agnostic
(Ignore X)

Decoupled
(First learn, then reconstruct)

Unrolled Optimization

Neumann Networks
(this talk!)

Deep proximal gradient

$$y \longrightarrow \boxed{\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \widehat{\beta}$$

set $\widehat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$z^{(k)} = \widehat{\beta}^{(k)} + \eta X^\top (y - X\widehat{\beta}^{(k)})$$ gradient descent

$$\widehat{\beta}^{(k+1)} = \arg\min_{\beta} \|z^{(k)} - \beta\|_2^2 + \eta r(\beta)$$ denoising

Deep proximal gradient

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

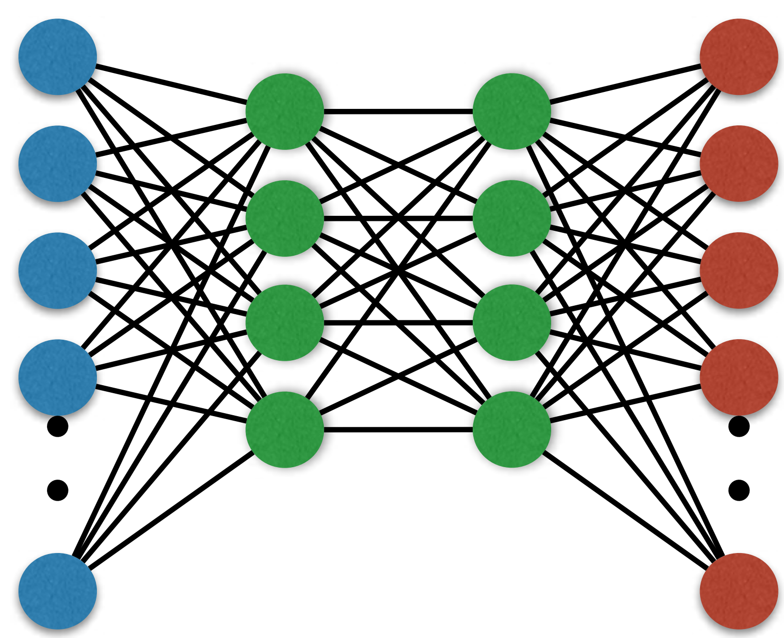
set $\hat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$z^{(k)} = \hat{\beta}^{(k)} + \eta X^\top (y - X\hat{\beta}^{(k)})$$ gradient descent

$$\hat{\beta}^{(k+1)} = \boxed{\arg\min_{\beta} \|z^{(k)} - \beta\|_2^2 + \eta r(\beta)}$$ denoising

Replace with learned neural network



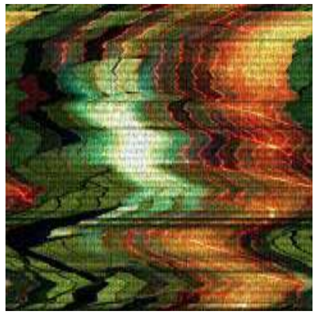
Chang, Li, Poczos, Kumar, & Sankaranarayanan 2017

GANs for inverse problems

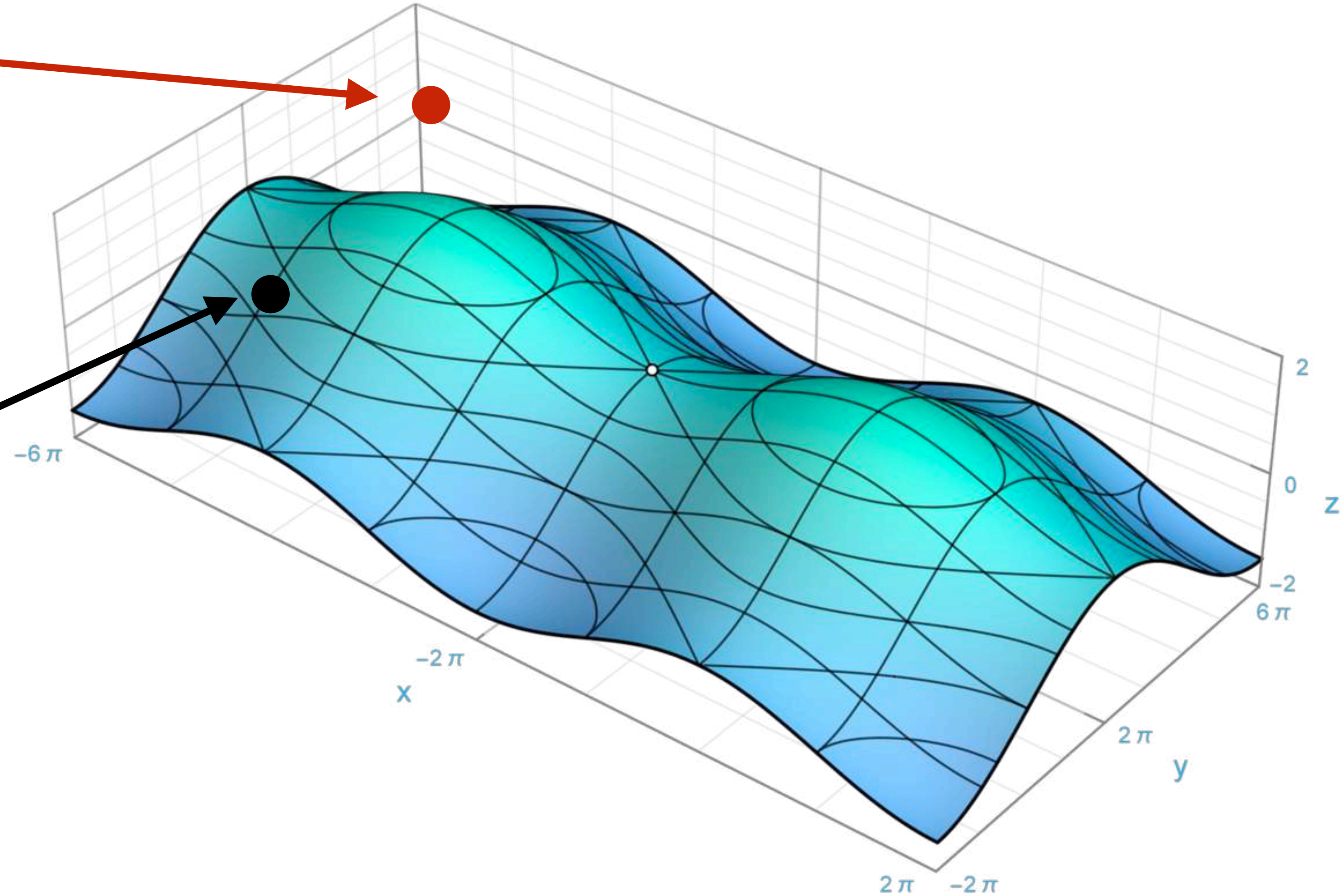
$$y \longrightarrow \boxed{\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \widehat{\beta}$$

$$r(\beta) = \begin{cases} 0, & \beta \text{ on image manifold} \\ \infty, & \text{otherwise} \end{cases}$$

"Bad" image off manifold



"Good" image on manifold



GANs for inverse problems

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

$$r(\beta) = \begin{cases} 0, & \beta \text{ on image manifold} \\ \infty, & \text{otherwise} \end{cases}$$

GANs for inverse problems

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

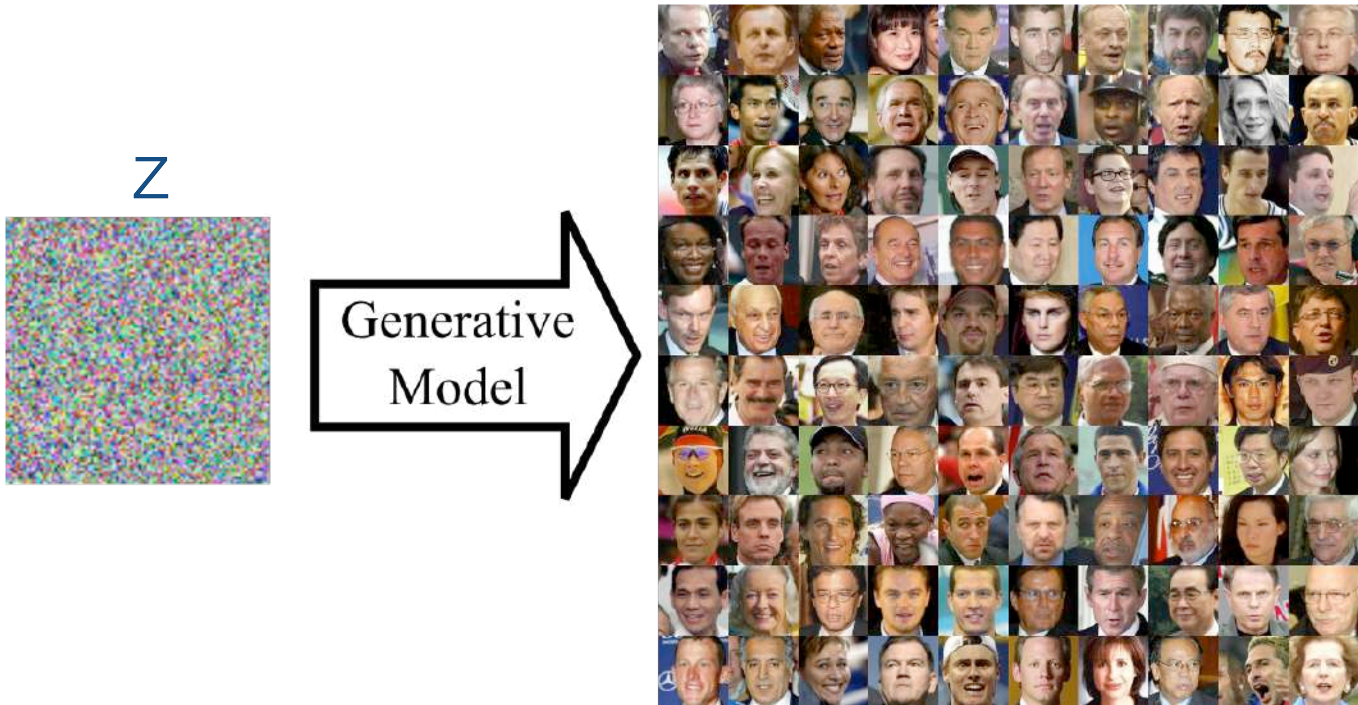
$$r(\beta) = \begin{cases} 0, & \beta \text{ on image manifold} \\ \infty, & \text{otherwise} \end{cases}$$

Learn generator G that outputs $\beta \in \mathbb{R}^d$ given $z \in \mathbb{R}^{d'}$ for $d' < d$

$$r(\beta) = \begin{cases} 0, & \beta \in \text{range}(G) \\ \infty, & \text{otherwise} \end{cases}$$

G(z)

z



GANs for inverse problems

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

$$r(\beta) = \begin{cases} 0, & \beta \text{ on image manifold} \\ \infty, & \text{otherwise} \end{cases}$$

Learn generator G that outputs $\beta \in \mathbb{R}^d$ given $z \in \mathbb{R}^{d'}$ for $d' < d$

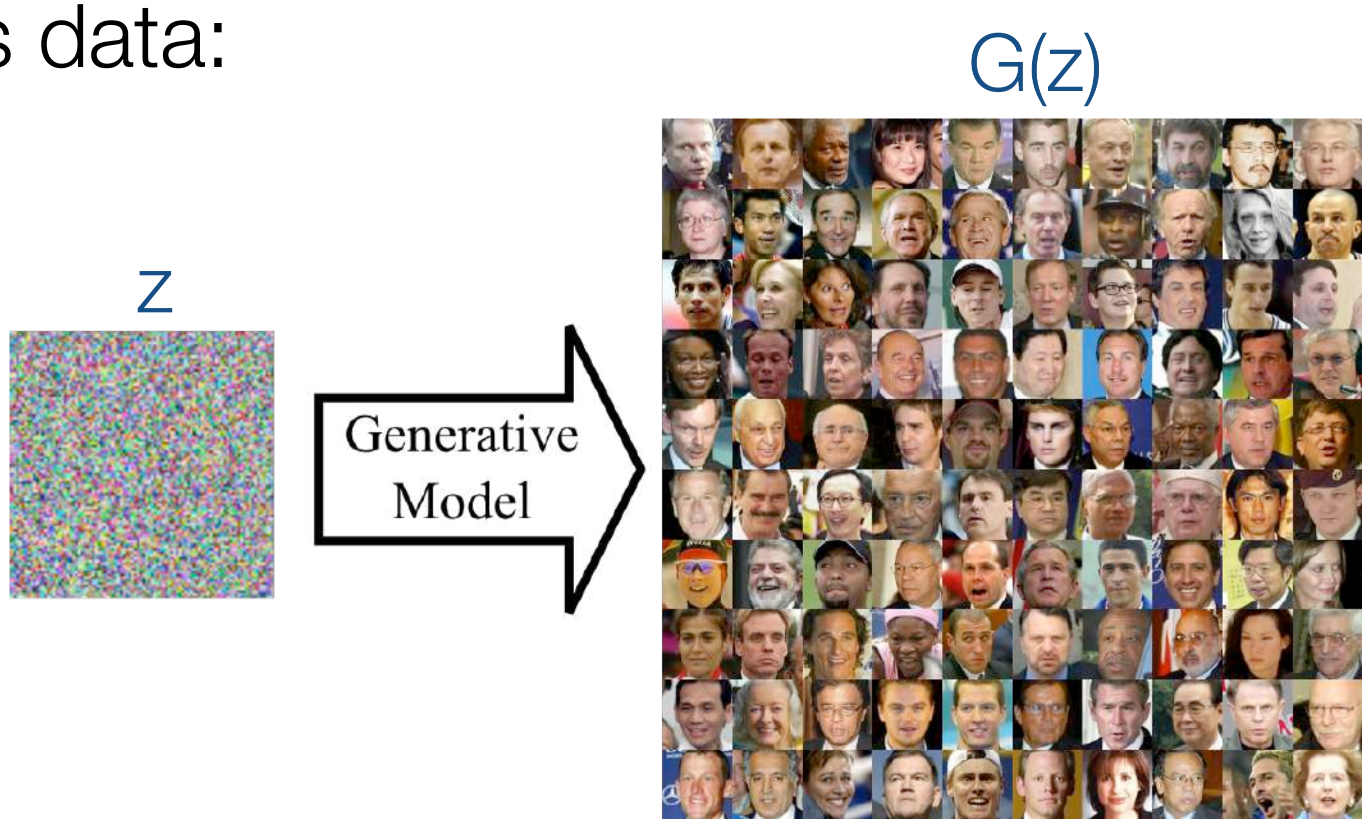
$$r(\beta) = \begin{cases} 0, & \beta \in \text{range}(G) \\ \infty, & \text{otherwise} \end{cases}$$

Choose $\beta \in$ range(G) that best fits data:

$$\hat{\beta} = \arg\min_{\beta \in \text{range}(G)} \|y - X\beta\|_2^2$$

$$= G(\hat{z})$$

$$\hat{z} = \arg\min_{z} \|y - XG(z)\|_2^2$$



Bora, Jalal, Price, Dimakis, 2017

How much training data?

Original
β

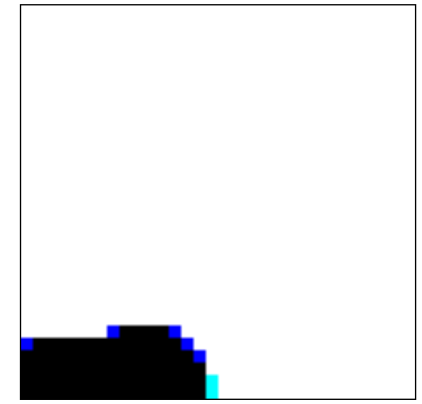
Observed
y

Reconstruction with convolutional neural network (CNN) trained with 80k samples

How much training data?

Original
β

Observed
y

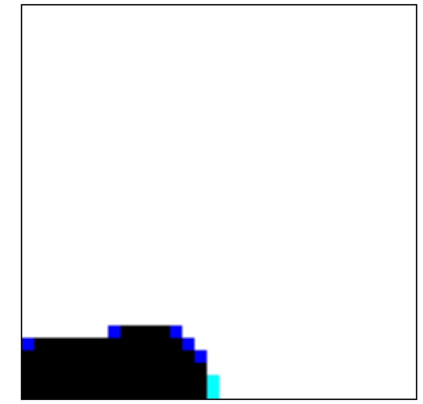


Reconstruction with convolutional neural network (CNN) trained with 2k samples

How much training data?

Original
β

Observed
y



Reconstruction with convolutional neural network (CNN) trained with 2k samples

Donald J. Trump
@realDonaldTrump

You cannot trust CNN! They are FAKE!!!

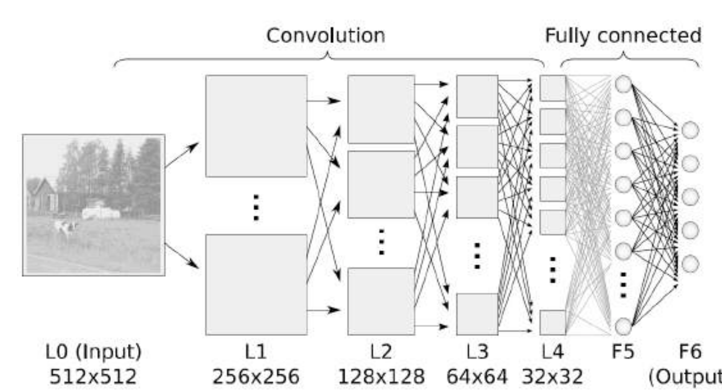
RETWEETS 7,771 LIKES 2,094

11:07 AM - 21 Nov 2017

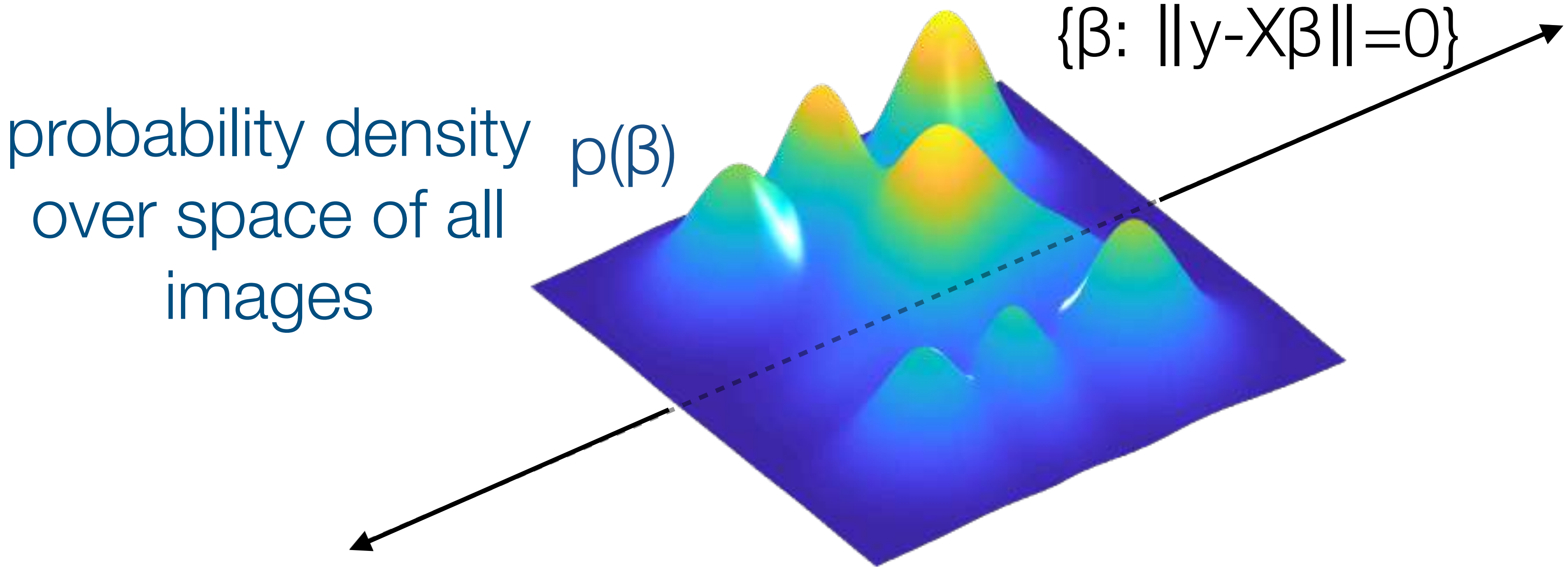
364 8K 2K

What people think he's referring to:

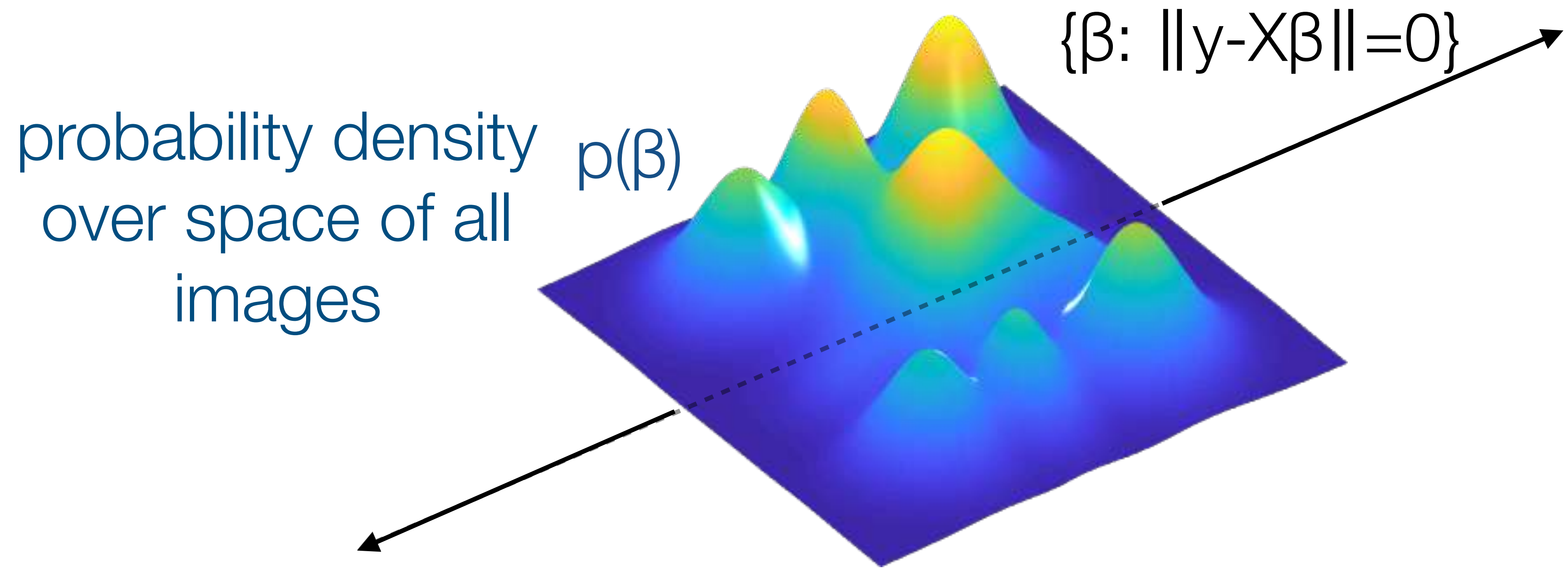
What he's actually referring to:



Learning a proximal operator or learning a generative model both implicitly require estimating p(β)



Learning a proximal operator or learning a generative model both implicitly require estimating p(β)



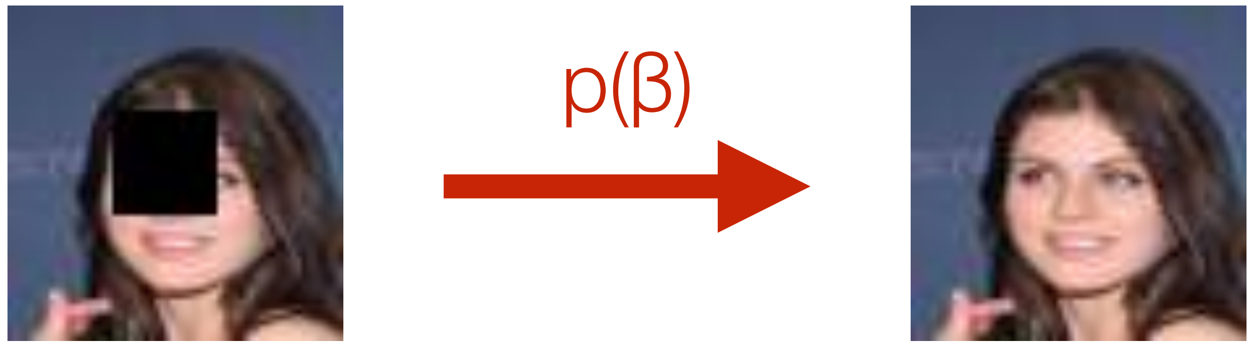
If $\beta \in \mathbb{R}^d$ and $p(\beta) \in \mathcal{B}_\alpha$ (Besov-α smooth functions), then the minimax rate for learning p(β)

$$\min_{\hat{p}} \max_{p \in \mathcal{B}_\alpha} \mathbb{E}\|\hat{p}(\beta) - p(\beta)\|_2 = \mathcal{O}\left(n^{-\frac{\alpha}{2\alpha+d}}\right)$$

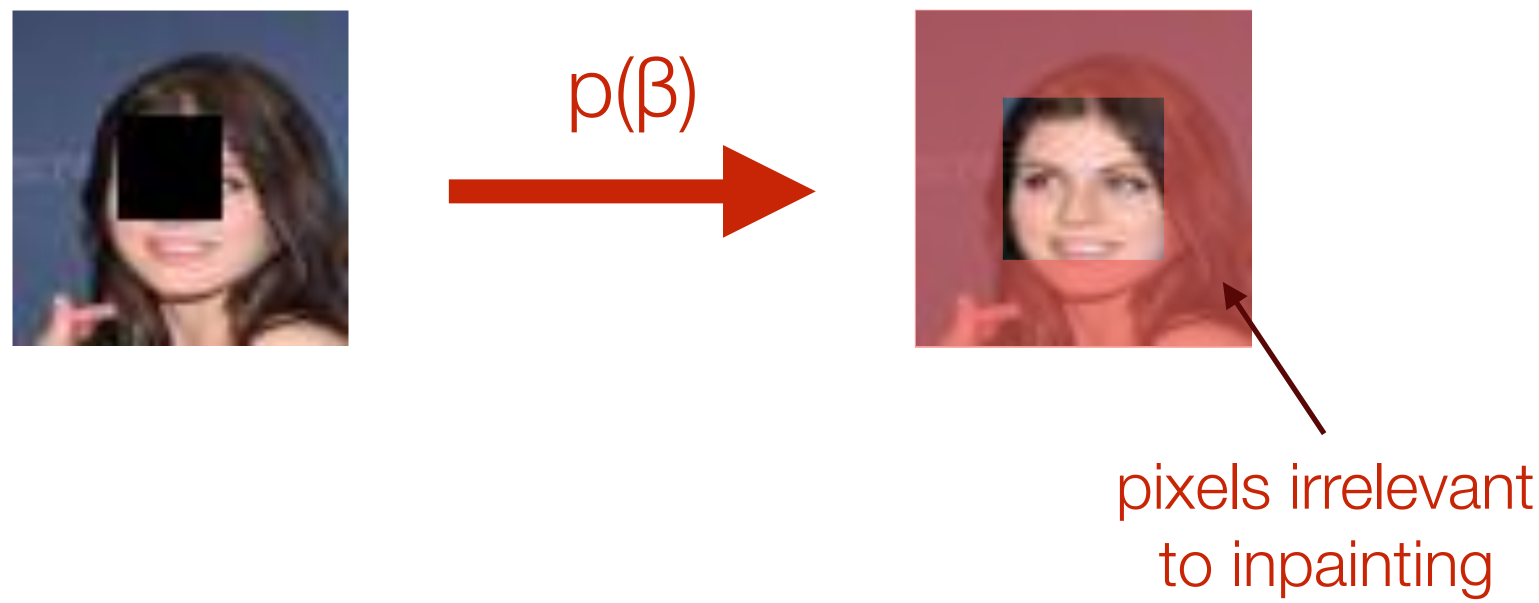
No neural network can beat this rate!

Donoho, Johnstone, Kerkyacharian, and Picard 1996

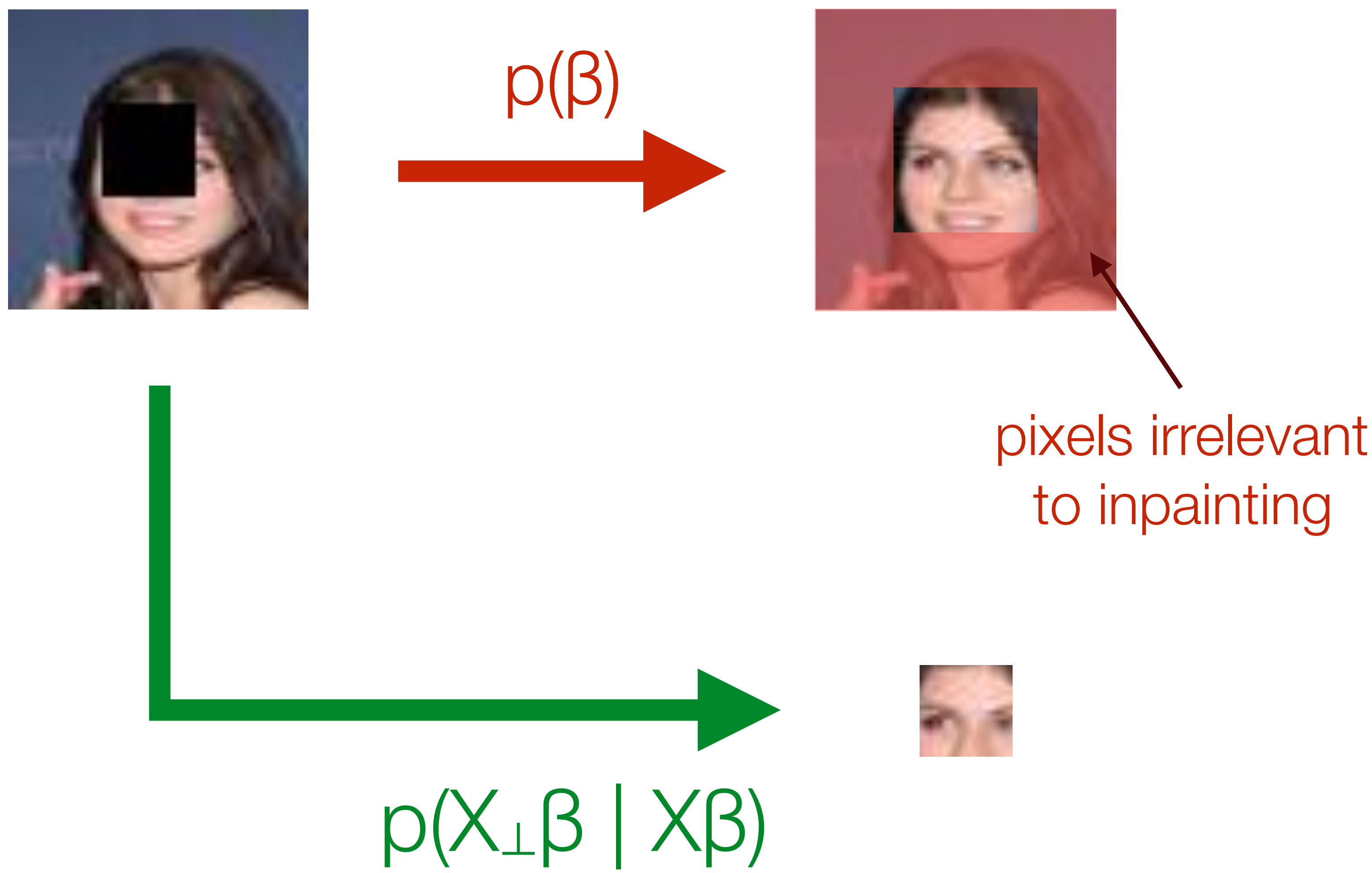
Prior vs. conditional density estimation



Prior vs. conditional density estimation



Prior vs. conditional density estimation



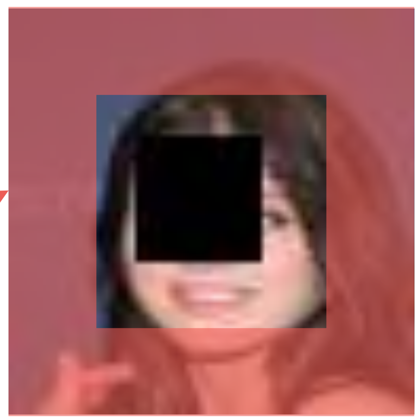
We need conditional density $p(X_{\perp}\beta \mid X\beta)$

Conditional density estimation

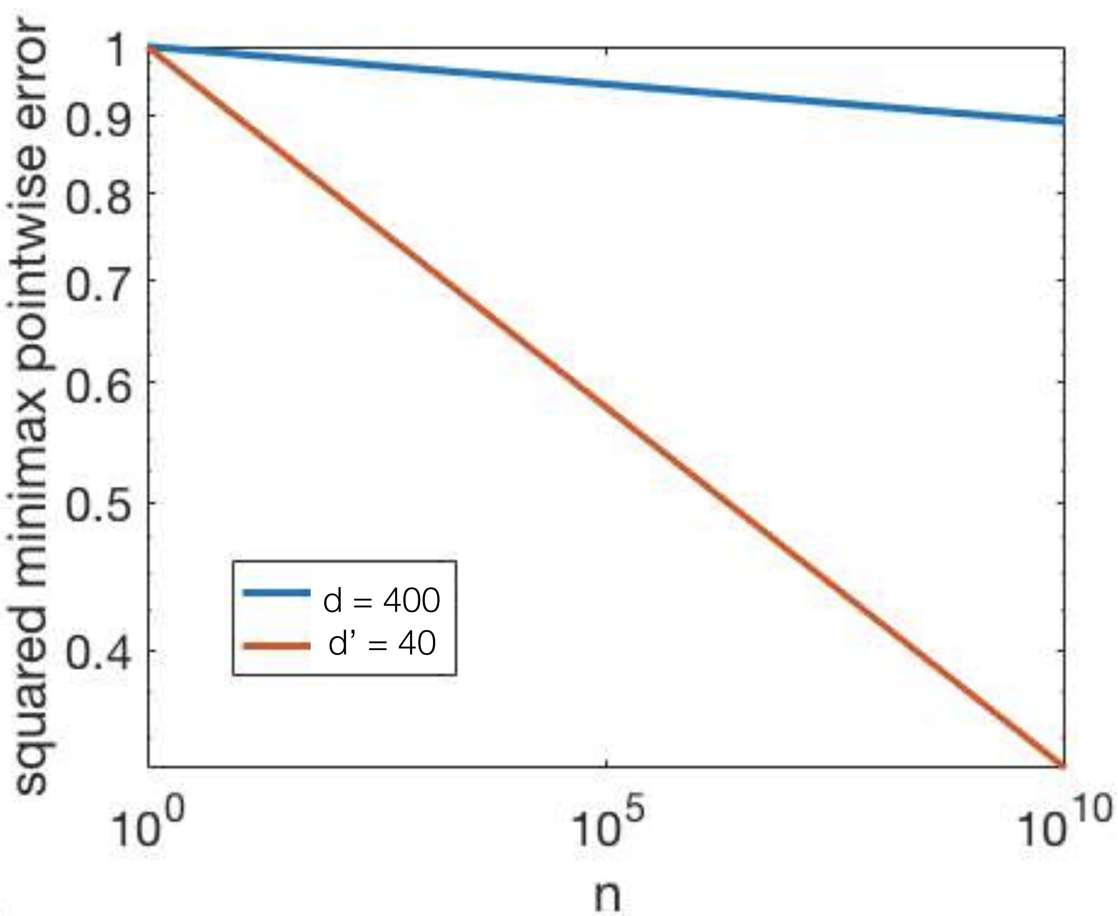
Conditional density [$p(X_\perp\beta \mid X\beta)$] estimation can be much easier than density [$p(\beta)$] estimation

If $X_\perp\beta$ only depends on d' elements in $X\beta$, then the minimax rate is

$$\min_{\hat{p}} \max_{p\in\mathcal{B}_\alpha} \mathbb{E}\|\hat{p}(X_\perp\beta|X\beta) - p(X_\perp\beta|X\beta)\|_2 = \mathcal{O}\left(n^{-\frac{\alpha}{2\alpha+d'}}\right)$$



Pixels irrelevant for inpainting



To reach a target squared pointwise error of ½:

- estimating $p(\beta)$ requires $n \approx 10^{60}$
- estimating $p(X_\perp\beta \mid X\beta)$ requires $n \approx 10^6$

Efromovich 2007
Bertin, Lacour, Rivoirard 2016
Nguyen 2018

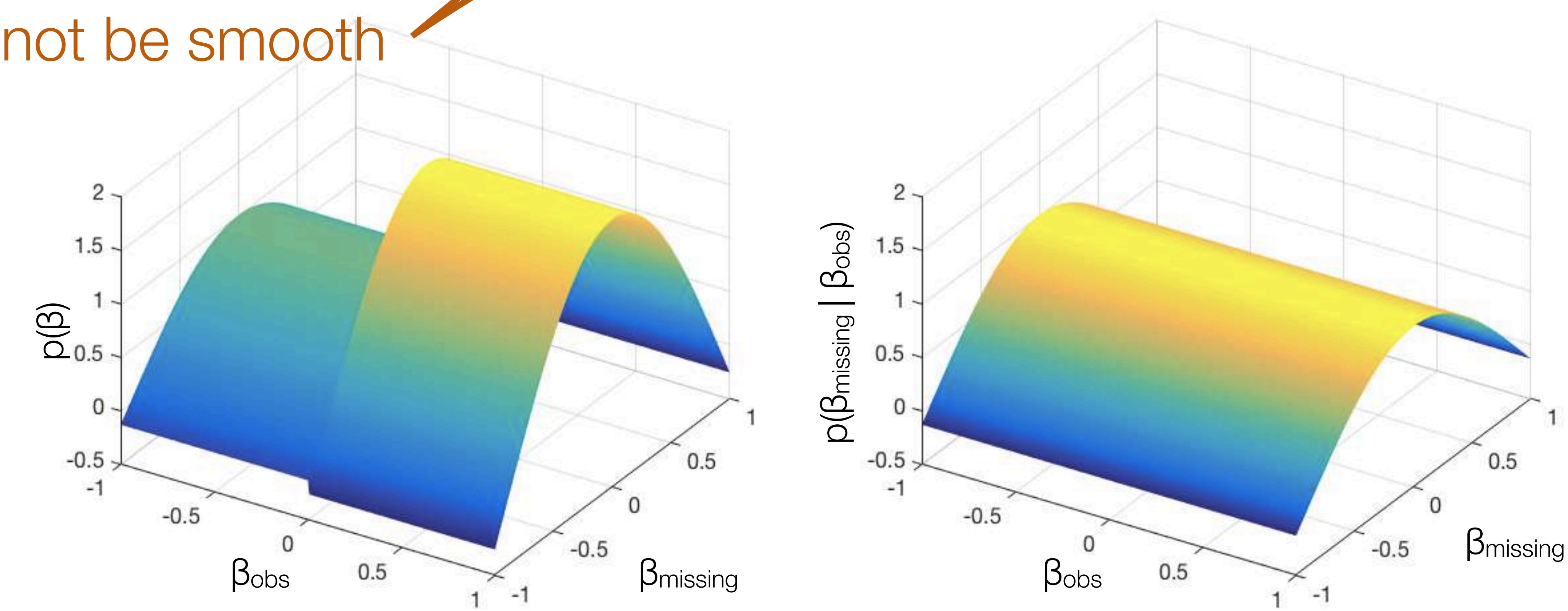
Conditional density estimation

Conditional density [p(X⊥β | Xβ)] estimation can be much easier than density [p(β)] estimation

$$p(\beta_{\text{missing}}|\beta_{\text{obs}}) = \frac{p(\beta_{\text{missing}}, \beta_{\text{obs}})}{p(\beta_{\text{obs}})} = \frac{p(\beta)}{p(\beta_{\text{obs}})}$$

Can be smooth

Either may not be smooth



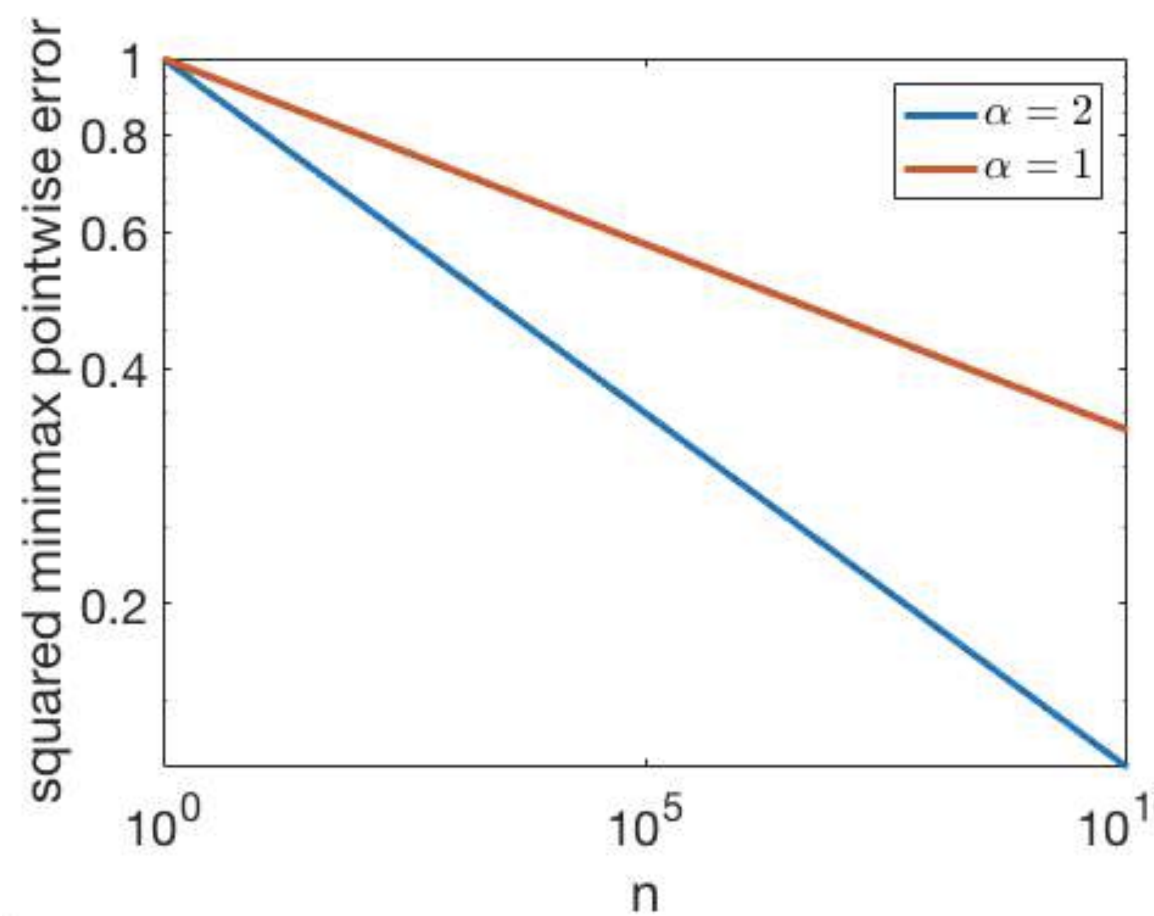
Conditional density estimation

Conditional density [p(X⊥β | Xβ)] estimation can be much easier than density [p(β)] estimation

$$p(\beta_{\text{missing}}|\beta_{\text{obs}}) = \frac{p(\beta_{\text{missing}}, \beta_{\text{obs}})}{p(\beta_{\text{obs}})} = \frac{p(\beta)}{p(\beta_{\text{obs}})}$$

Can be smooth

Either may not be smooth



Implications for learning to regularize

Estimating conditional density $p(X_{\perp}\beta \mid X\beta)$ can require far fewer samples than estimating full density $p(\beta)$

$\Downarrow$

X should be fully utilized in learning process

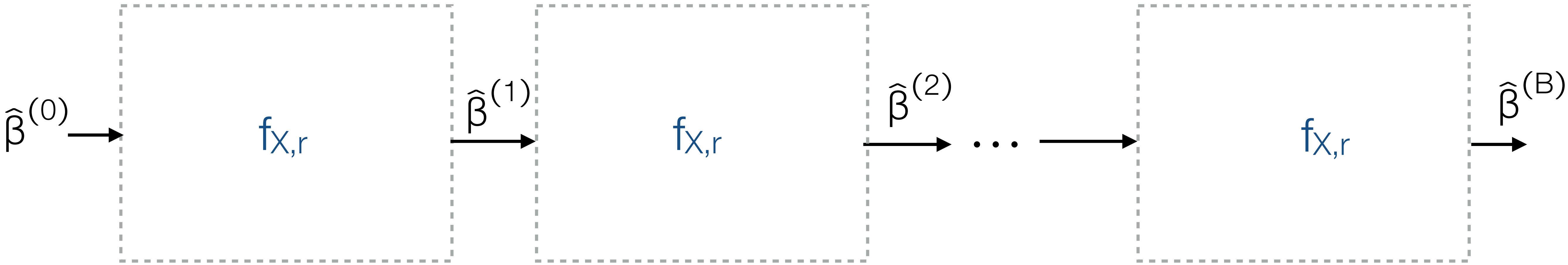
Unrolled optimization methods

$$y \longrightarrow \boxed{\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

Initialize $\hat{\beta}^{(0)}$

$$\hat{\beta}^{(B)} = f_{X,r}(\hat{\beta}^{(B-1)})$$ iteration map parameterized by X,r

$$= f_{X,r}(f_{X,r}(f_{X,r}(\cdots f_{X,r}(\hat{\beta}^{(0)}) \cdots)))$$ recurrent network



Unrolled optimization methods

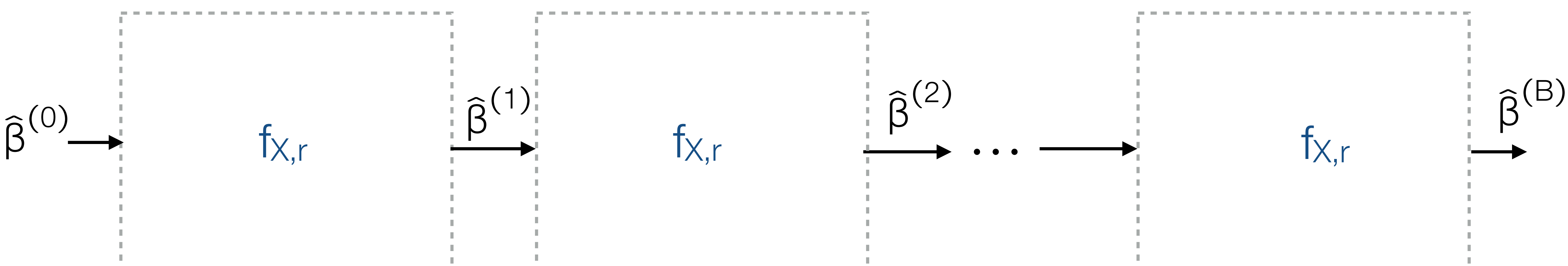
$$y \longrightarrow \boxed{\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \widehat{\beta}$$

Initialize $\widehat{\beta}^{(0)}$

$$\begin{aligned} \widehat{\beta}^{(B)} &= f_{X,r}(\widehat{\beta}^{(B-1)}) \\ &= f_{X,r}(f_{X,r}(f_{X,r}(\cdots f_{X,r}(\widehat{\beta}^{(0)})\cdots))) \end{aligned}$$

iteration map parameterized by X,r

recurrent network



learn r from training data

Gregor and LeCun, 2010

“Unrolled” gradient descent

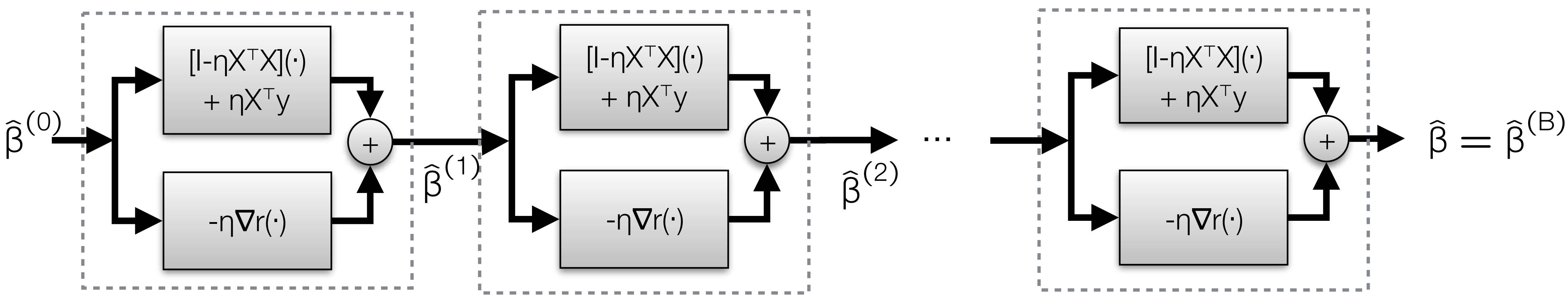
Assume $r(\beta)$ differentiable.

$$\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

set $\widehat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$\widehat{\beta}^{(k+1)} = \widehat{\beta}^{(k)} + \eta X^\top (y - X\widehat{\beta}^{(k)}) + \eta \nabla r(\widehat{\beta}^{(k)})$$



“Unrolled” gradient descent

Assume $r(\beta)$ differentiable.

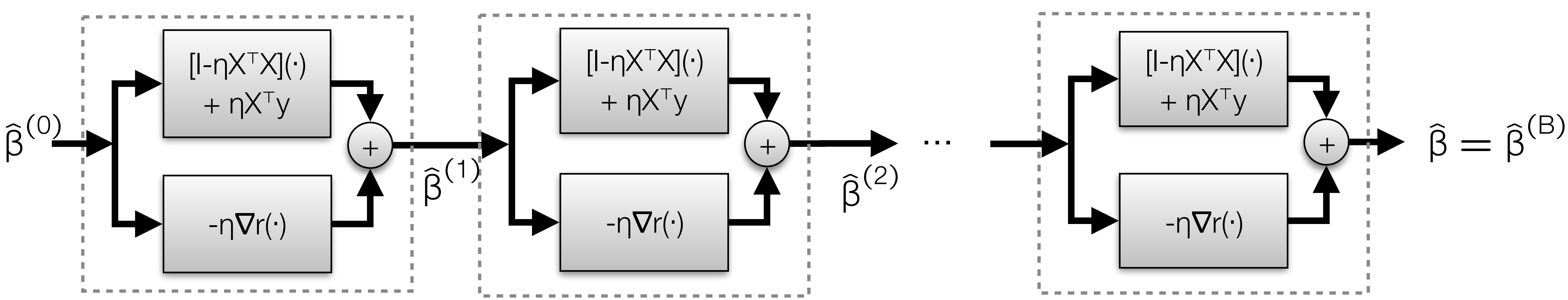
$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

set $\hat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$\hat{\beta}^{(k+1)} = \hat{\beta}^{(k)} + \eta X^\top (y - X\hat{\beta}^{(k)}) + \boxed{\eta \nabla r(\hat{\beta}^{(k)})}$$

Replace with learned neural network



"Unrolled" gradient descent

Assume $r(\beta)$ differentiable.

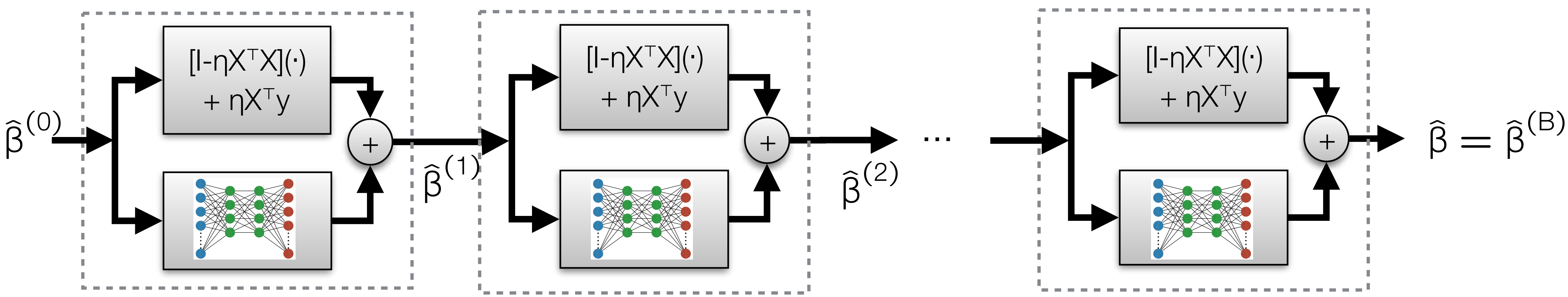
$$\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

set $\widehat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$\widehat{\beta}^{(k+1)} = \widehat{\beta}^{(k)} + \eta X^\top (y - X\widehat{\beta}^{(k)}) + \boxed{\eta \nabla r(\widehat{\beta}^{(k)})}$$

Replace with learned neural network



"Unrolled" gradient descent

Assume $r(\beta)$ differentiable.

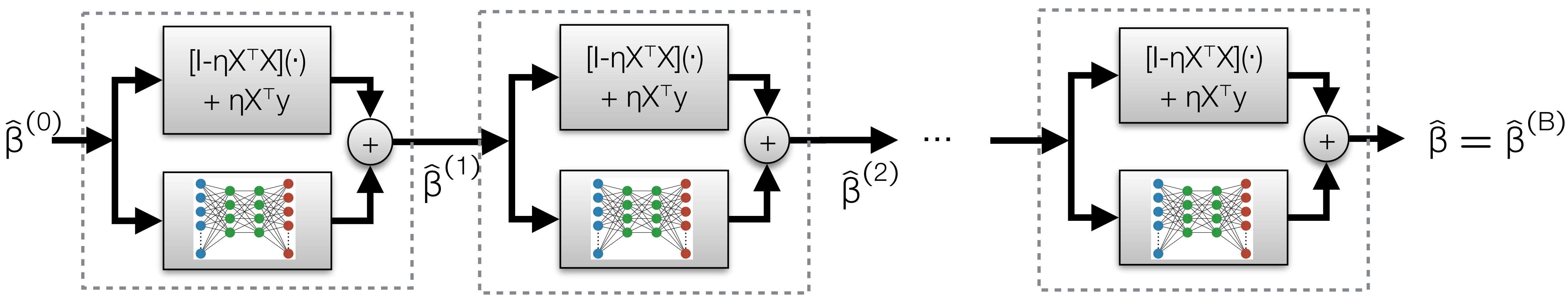
$$\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

set $\widehat{\beta}^{(1)}$ and stepsize $\eta > 0$

for $k = 1, 2, \ldots$

$$\widehat{\beta}^{(k+1)} = \widehat{\beta}^{(k)} + \eta X^\top (y - X\widehat{\beta}^{(k)}) + \boxed{\eta \nabla r(\widehat{\beta}^{(k)})}$$

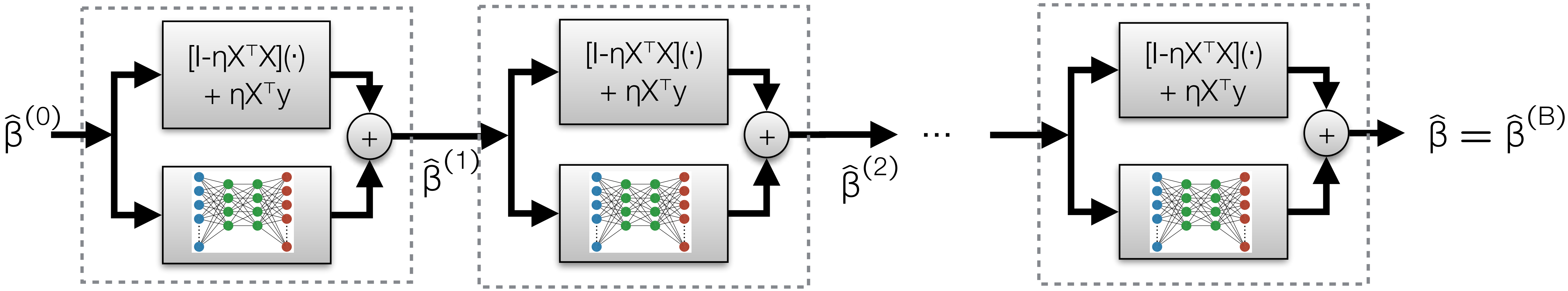
Replace with learned neural network



"Unrolled" optimization framework **trained end-to-end**

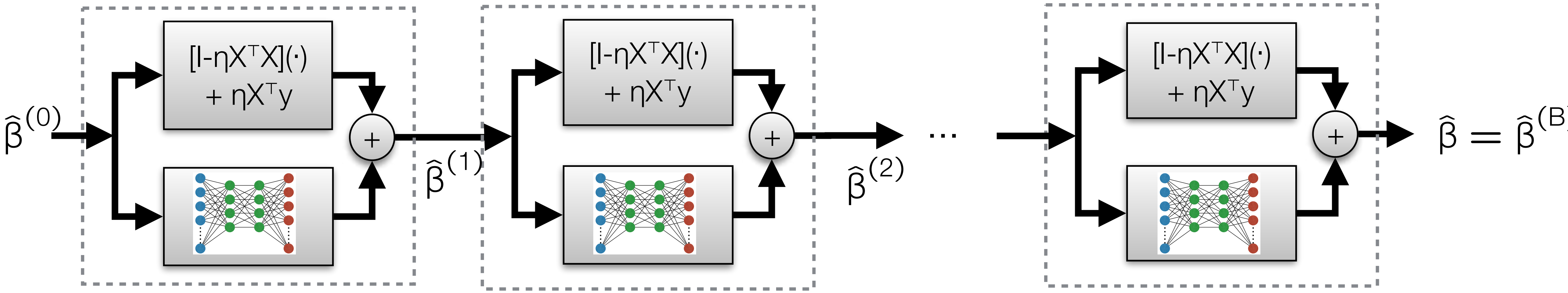
Beyond optimization

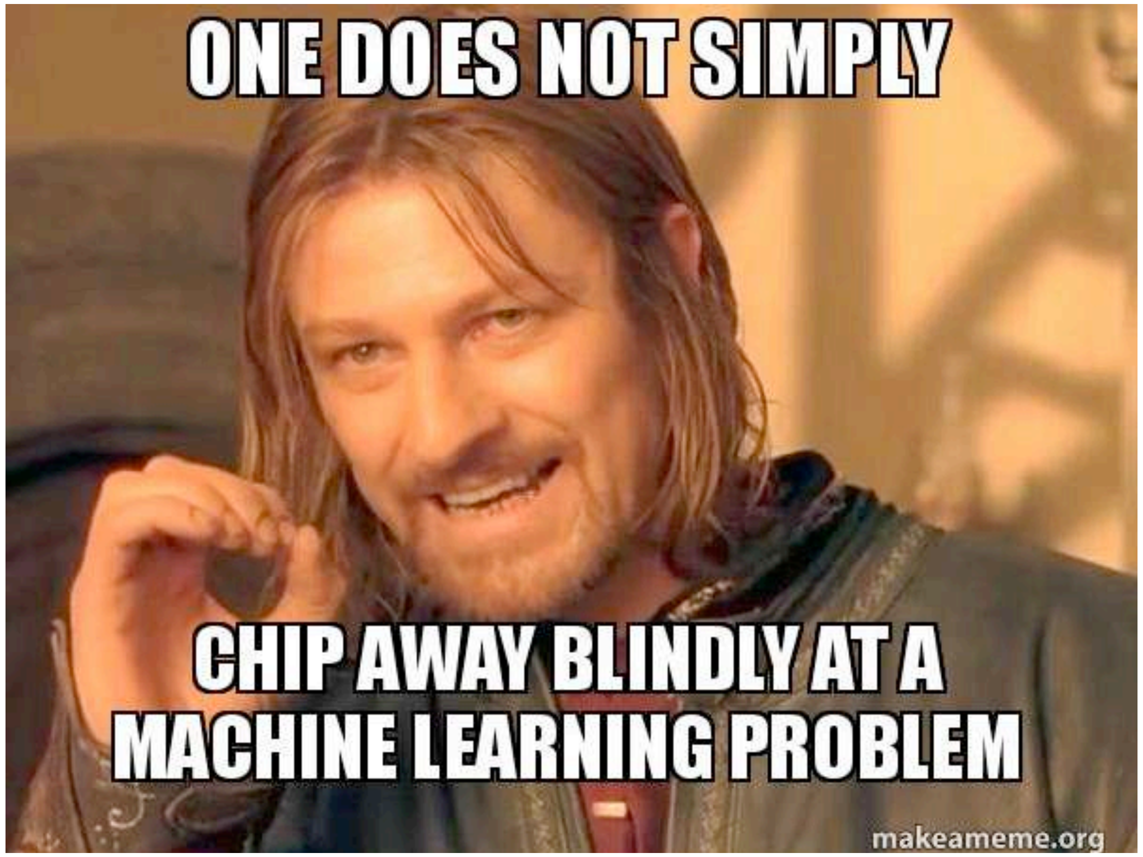
Unrolled methods so far originated in optimization — underlying theory does not apply here!



Beyond optimization

Unrolled methods so far originated in optimization — underlying theory does not apply here!





Neumann series

Assume $r(\beta)$ differentiable.

$$\begin{aligned}\hat{\beta} &= \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta) \\ &= (X^\top X + \nabla r)^{-1} X^\top y \end{aligned} \qquad (1)$$

Let A be a linear operator. Then the Neumann series is

$$(I - A)^{-1} = \sum_{k=0}^{\infty} A^k = I + A + A^2 + A^3 + \cdots \qquad (2)$$

If A is contractive, we know higher-order terms are smaller.

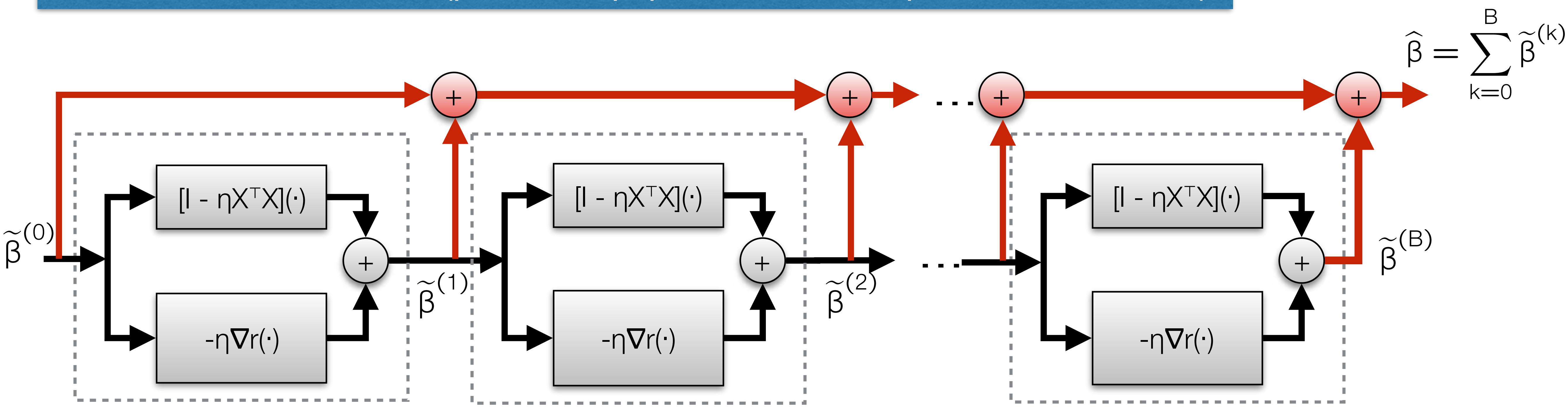
Can we estimate β by approximating (1) using (2)?
(e.g. $A = I - X^\top X + \nabla r$ if r is linear)

Neumann networks

Assume $r(\beta)$ differentiable.

$$\begin{aligned}\widehat{\beta} &= \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)\\ &= (X^\top X + \nabla r)^{-1} X^\top y\\ &\approx \sum_{k=1}^{B} (I - \eta X^\top X - \eta \nabla r)^k \eta X^\top y\end{aligned}$$

Neumann network (parallel pipelines + skip connections):



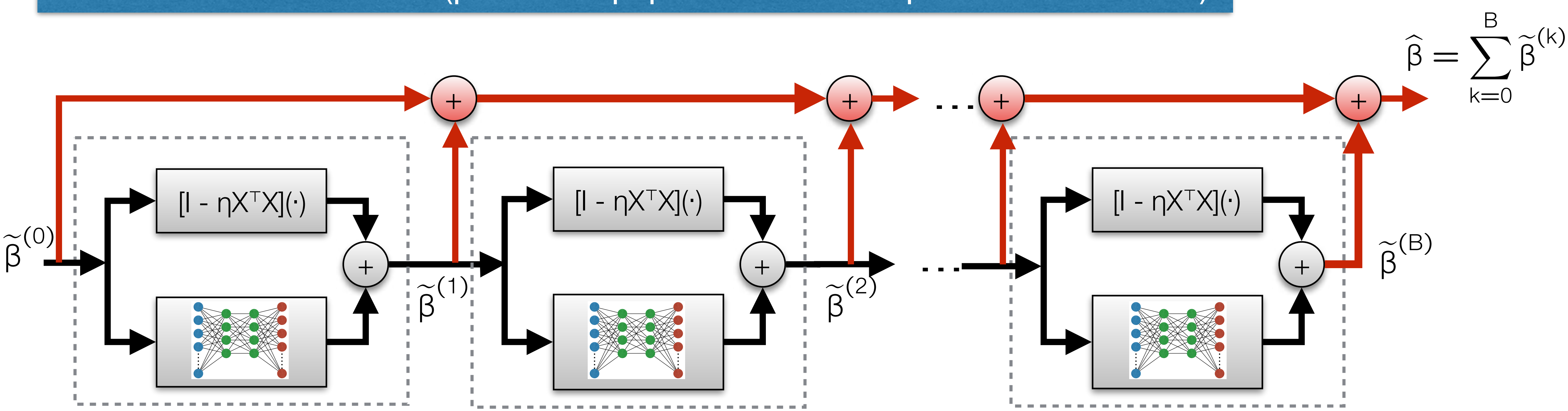
Neumann networks

Assume $r(\beta)$ differentiable.

$$\begin{aligned}\widehat{\beta} &= \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)\\ &= (X^\top X + \nabla r)^{-1} X^\top y\\ &\approx \sum_{k=1}^{B} (I - \eta X^\top X - \boxed{\eta \nabla r})^k \eta X^\top y\end{aligned}$$

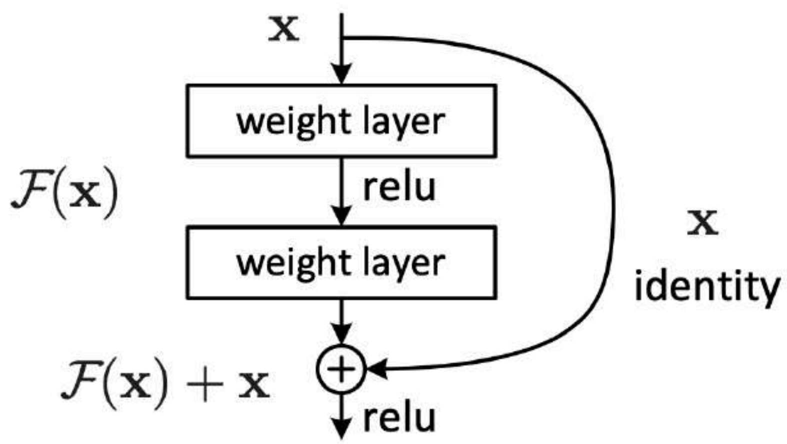
Replace with learned neural network

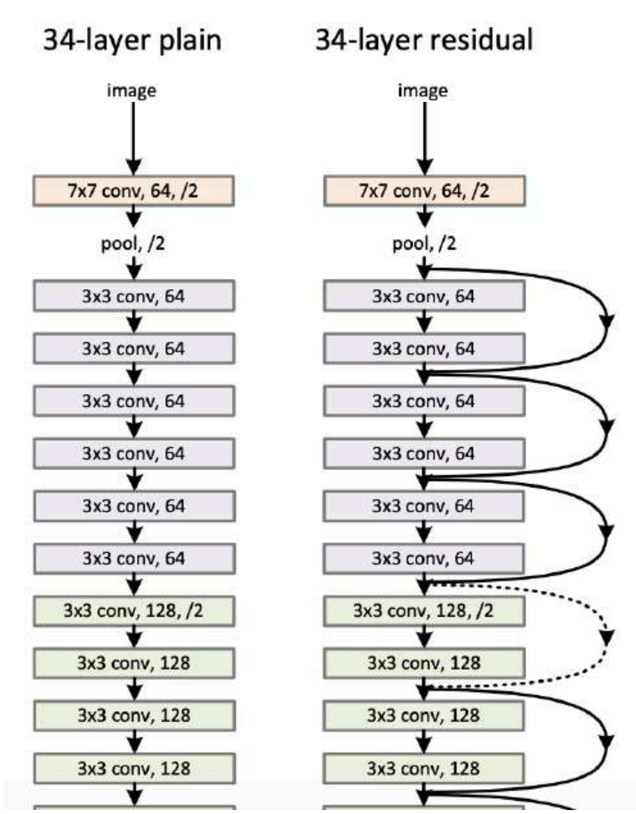
Neumann network (parallel pipelines + skip connections):



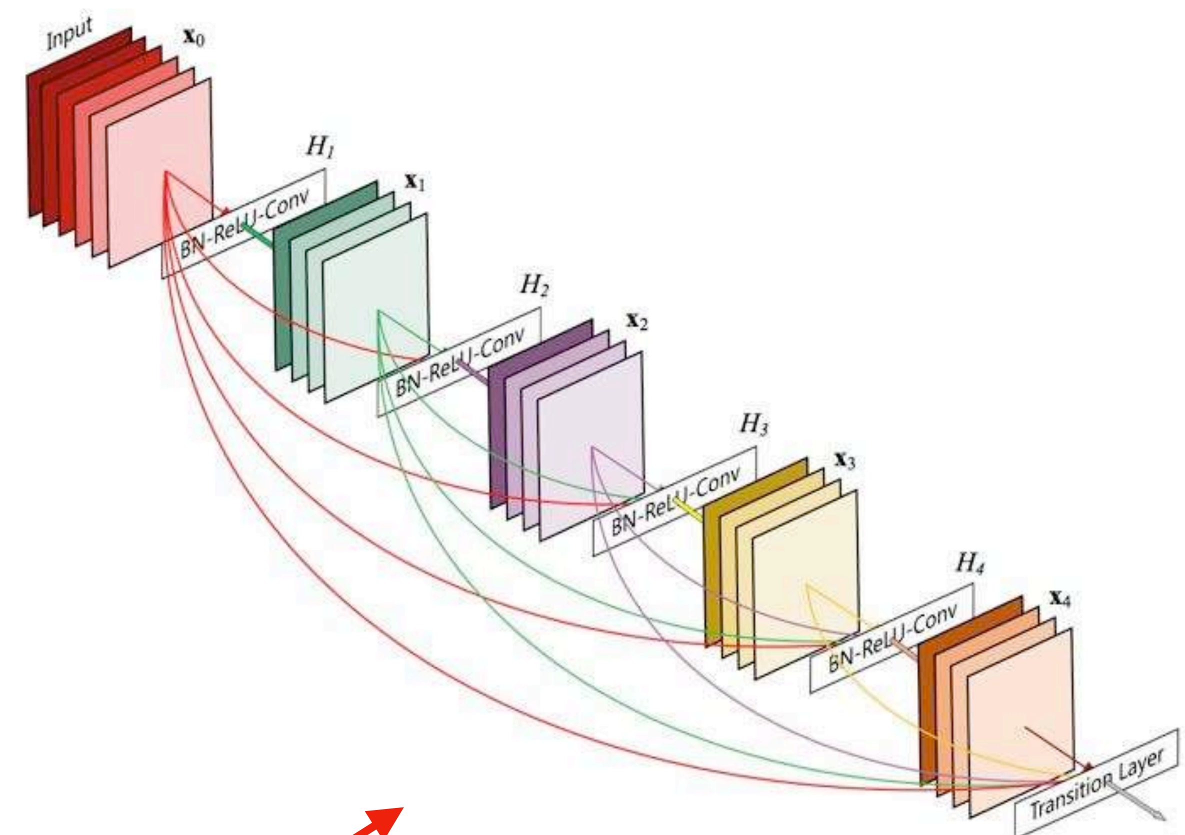
Skip connections in ResNets and DenseNets

Residual Networks (ResNets)





Dense Convolutional Networks (DenseNets)



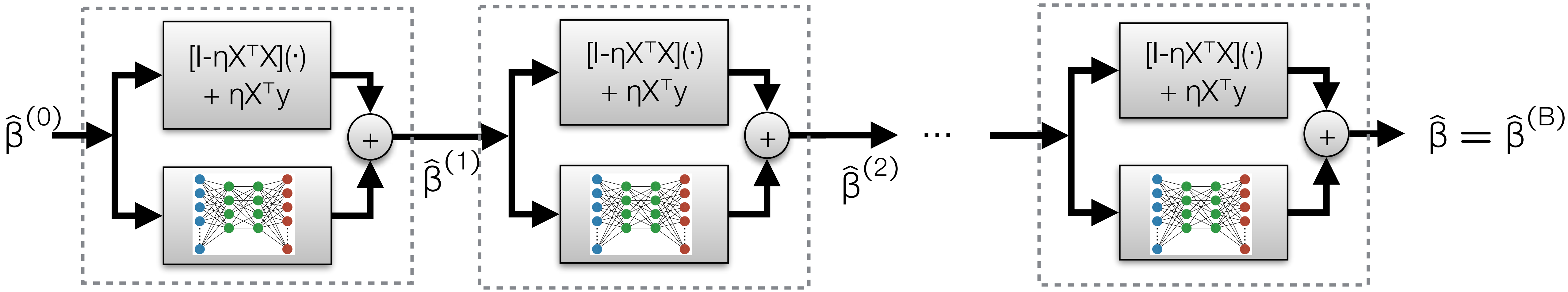
skip connections

He, Zhang, Ren, Sun 2015

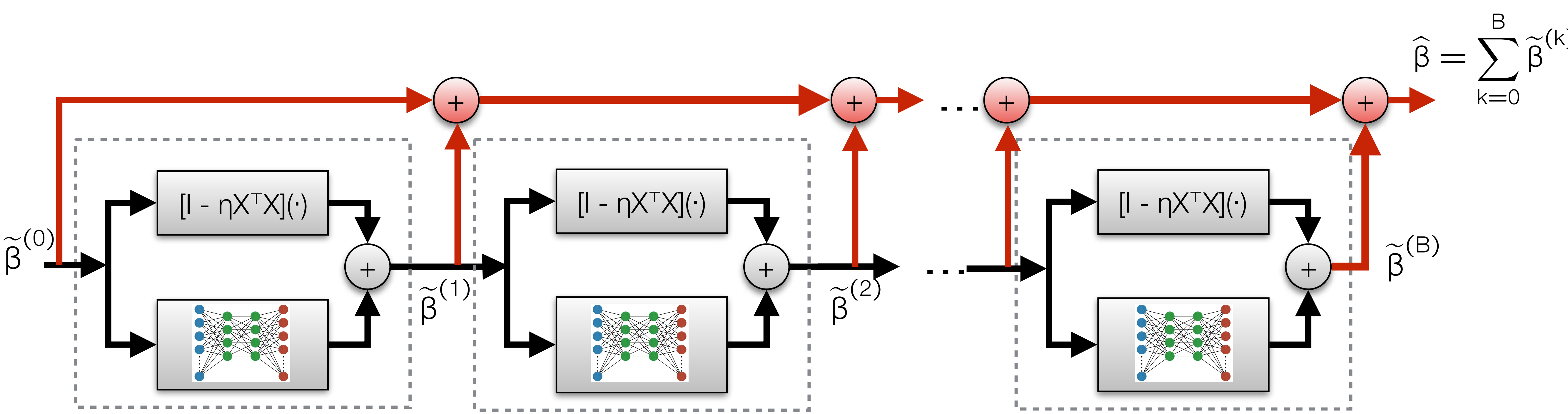
Huang, Liu, Van Der Maaten, & Weinberger 2017

Comparison

Gradient descent network



Neumann network (parallel pipelines + skip connections):



Experiments

Comparison Methods

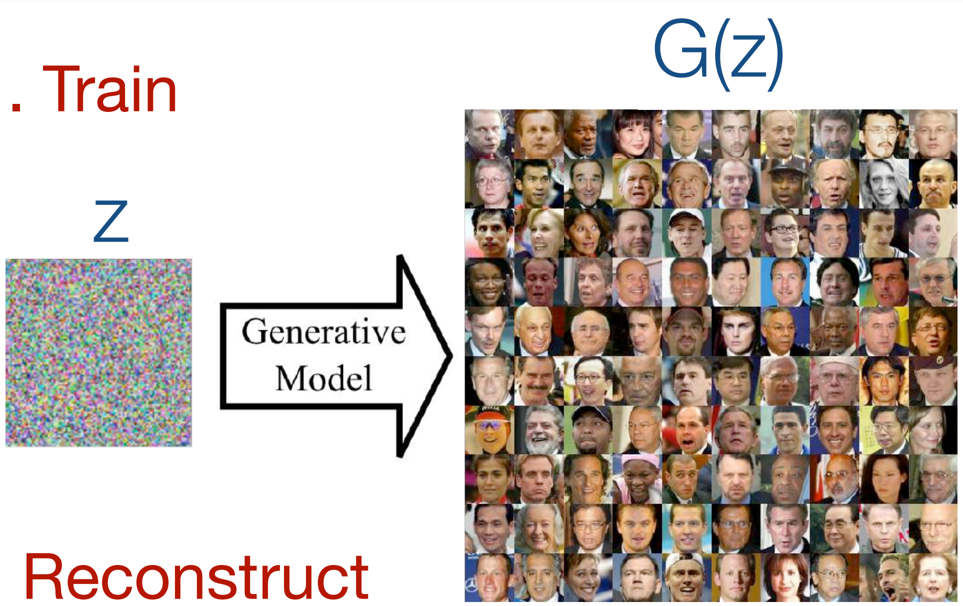
LASSO

$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\mathrm{DCT}(\beta)\|_1$$

discrete cosine transform
on 16x16 blocks

Design-agnostic GAN

1. Train

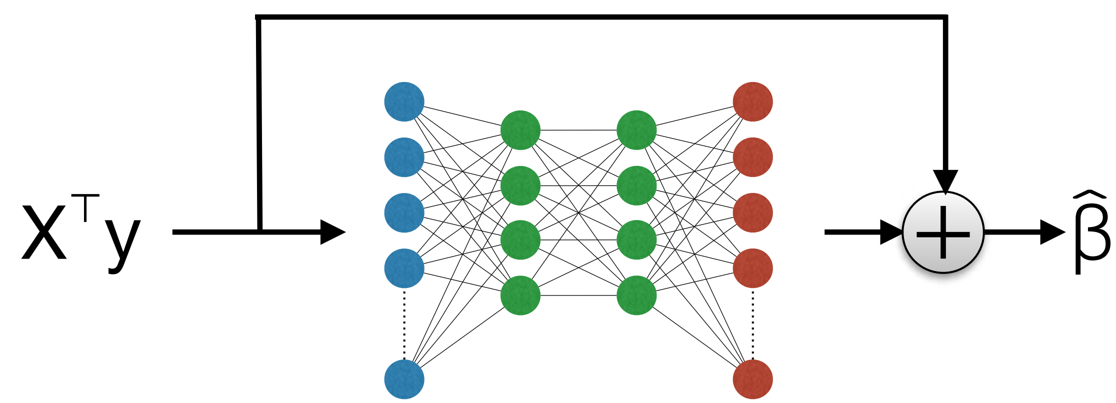


2. Reconstruct

$$\hat{\beta} = \arg\min_{\beta \in \mathrm{range}(G)} \|y - X\beta\|_2^2$$

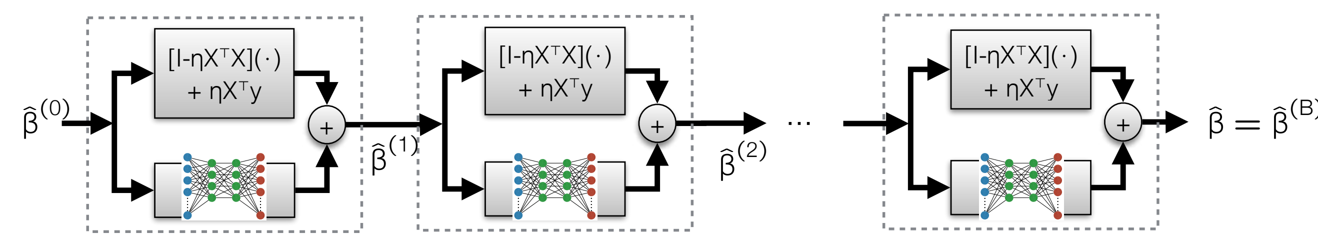
Bora, Jalal, Price, Dimakis, 2017

Residual Autoencoder

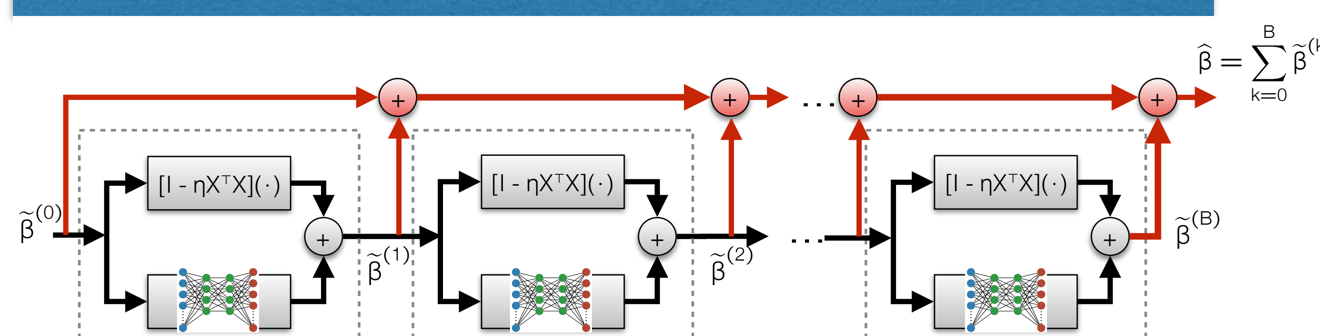


Mao, Shen, Yang, 2016

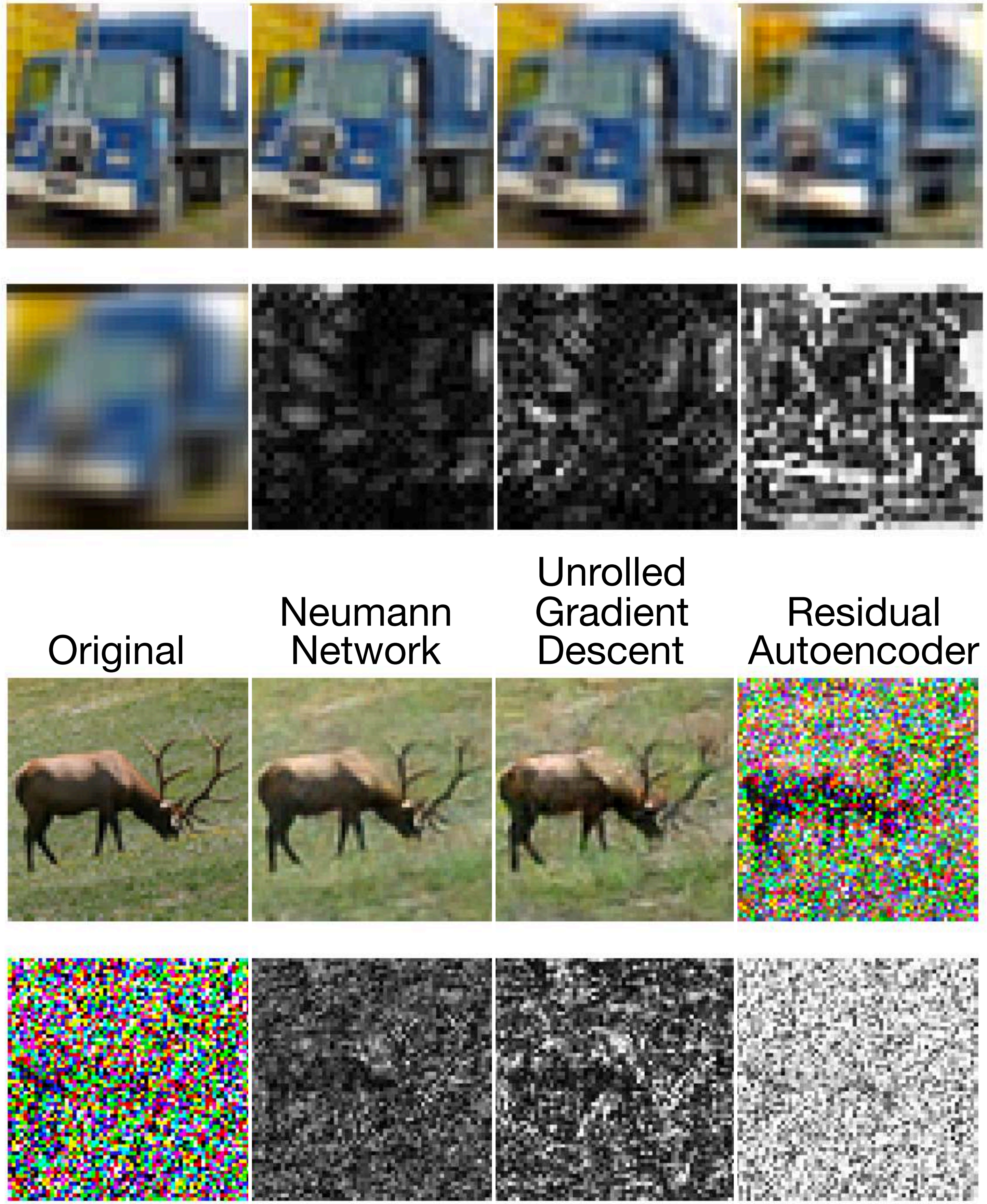
Unrolled Gradient Descent



Neumann Network



Examples



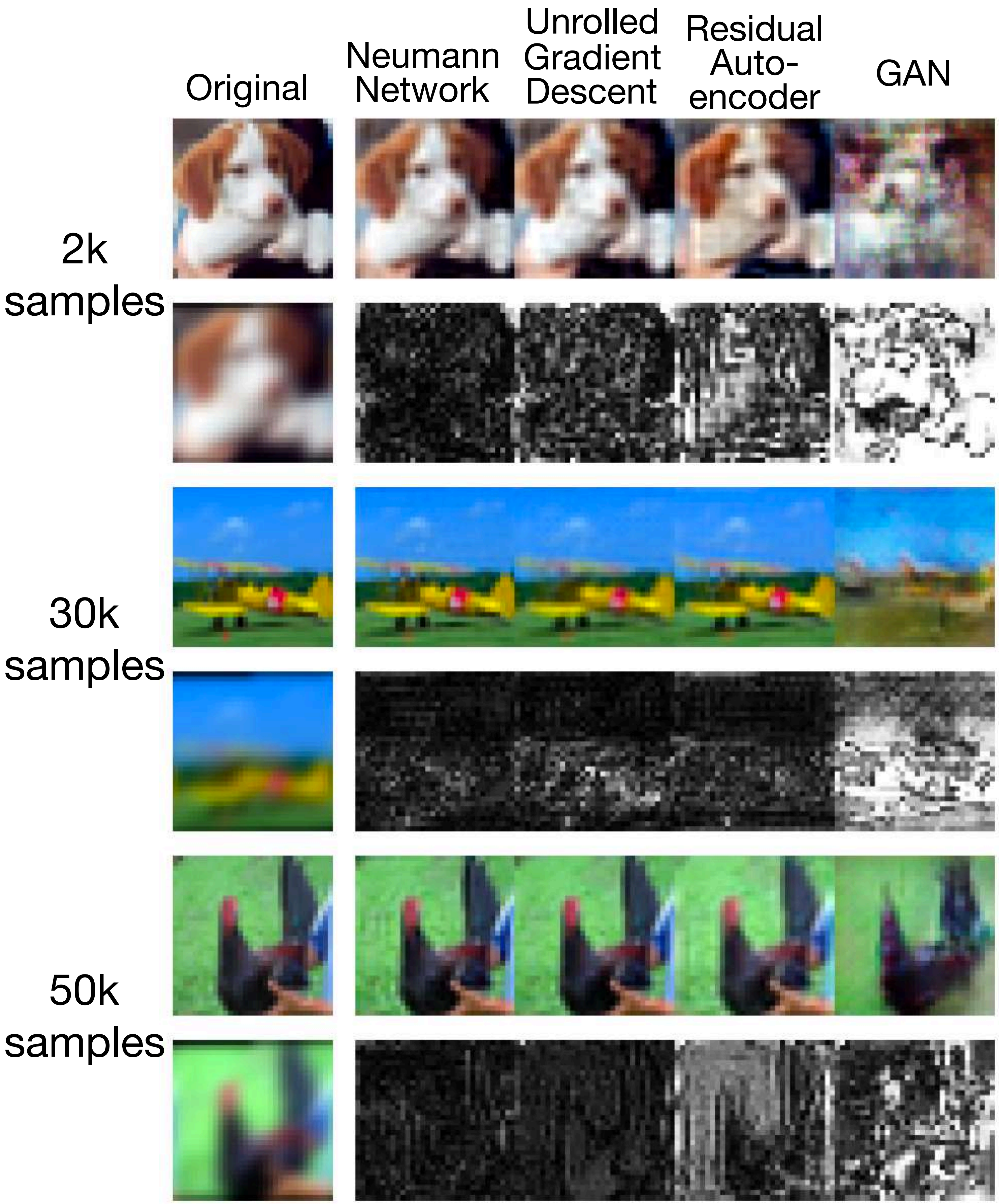
Deblurring
on CIFAR-10
Original
Neumann Network
Unrolled Gradient Descent
Residual Autoencoder
$X^\top y$
Error images (scaled x6)
Compressed Sensing (x8) on STL-10
Original
Neumann Network
Unrolled Gradient Descent
Residual Autoencoder
$X^\top y$
Error images (scaled x6)

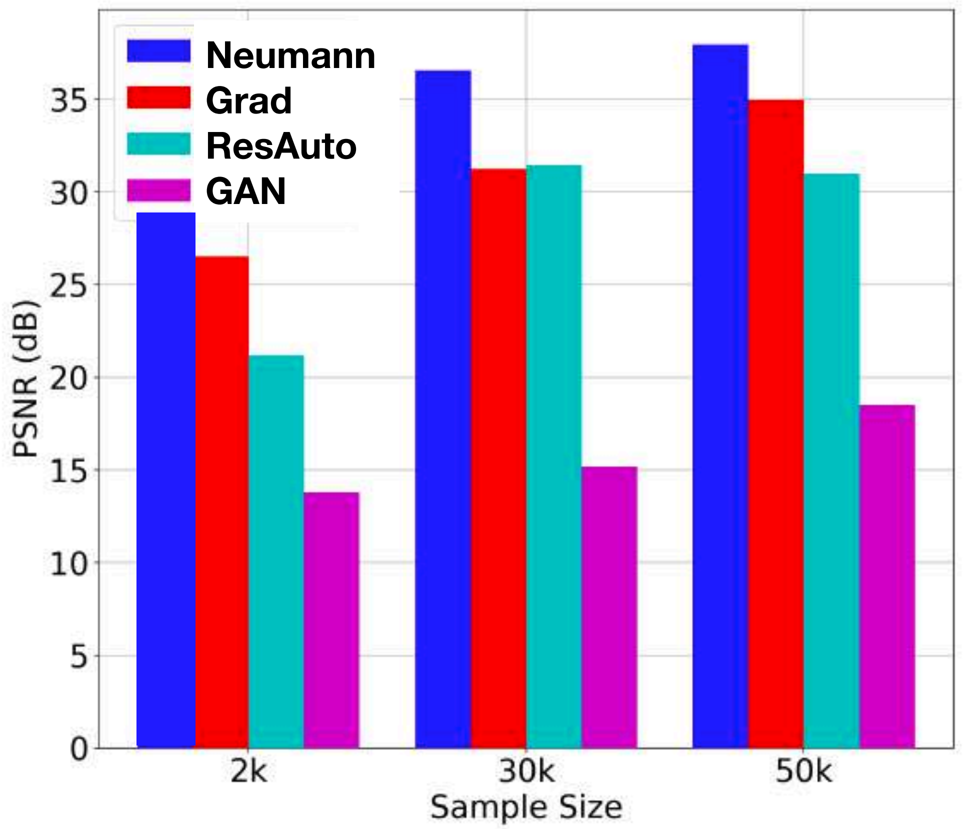
Summary of Results

		Inpaint	Deblur	CS2	CS8	SR4	SR10
CIFAR10	NN	28.20	**36.55**	33.83	**25.15**	24.48	**23.09**
	GDN	27.76	31.25	**34.99**	25.00	24.49	20.47
	ResAuto	**29.05**	31.04	18.51	9.29	**24.84**	21.92
	CSGM	17.88	15.20	17.99	19.33	16.87	16.66
	LASSO	19.34	23.70	22.74	16.37	20.03	19.93
CelebA	NN	**31.06**	**31.01**	**35.12**	**28.38**	**27.31**	23.57
	GDN	30.99	30.19	34.93	28.33	27.14	23.46
	ResAuto	29.66	25.65	19.41	9.16	25.62	**24.92**
	CSGM	17.75	15.68	17.99	18.21	18.11	17.88
	LASSO	15.99	14.82	24.37	17.61	16.56	22.74
STL10	NN	27.47	29.43	**31.98**	**26.65**	**24.88**	**21.80**
	GDN	**28.07**	**30.19**	31.11	26.19	**24.88**	21.46
	ResAuto	27.28	25.42	19.48	9.30	24.12	21.13
	CSGM	16.50	14.04	16.67	16.39	16.58	16.47
	LASSO	18.70	16.54	23.14	17.46	18.79	21.36

Table 1: PSNR comparison for the CIFAR, CelebA, and STL10 datasets respectively. Values reported are the median across a test set of size 256.

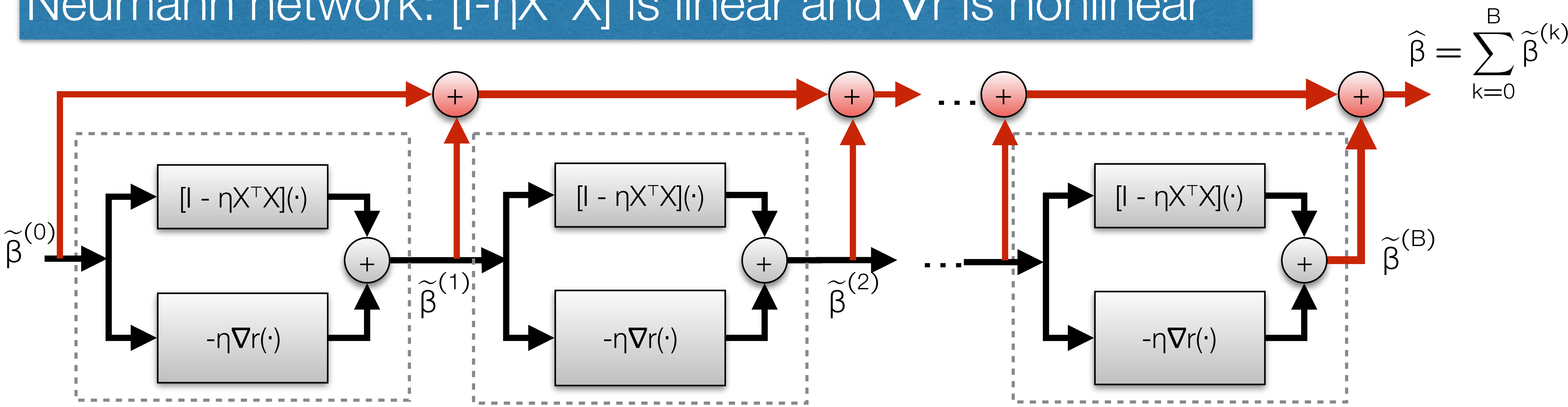
Sample Complexity





Preconditioning

Neumann network: $[I-\eta X^T X]$ is linear and ∇r is nonlinear

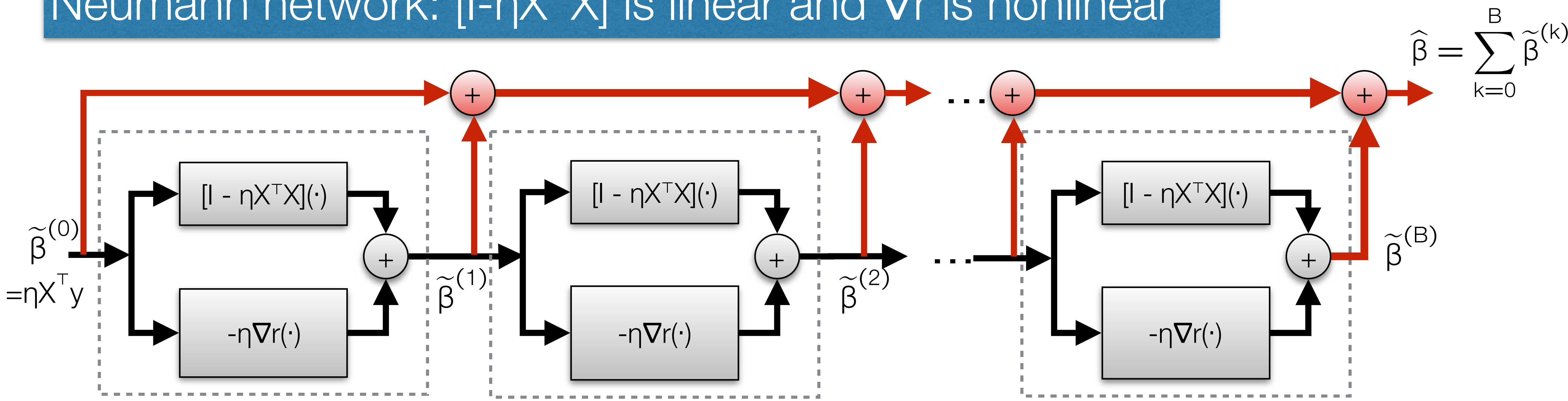


Preconditioned Neumann network:

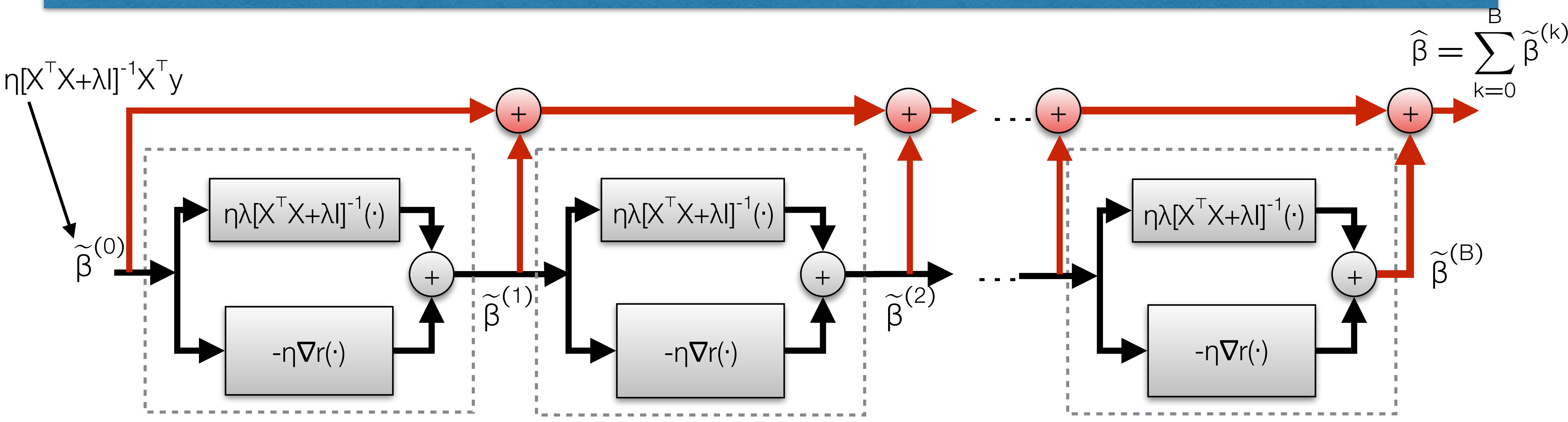
Instead of inputting $X^T y$, input Tikhinov estimate $(X^T X+\lambda I)^{-1} X^T y$ and adjust top blocks based on Neumann series expansion.

Preconditioning

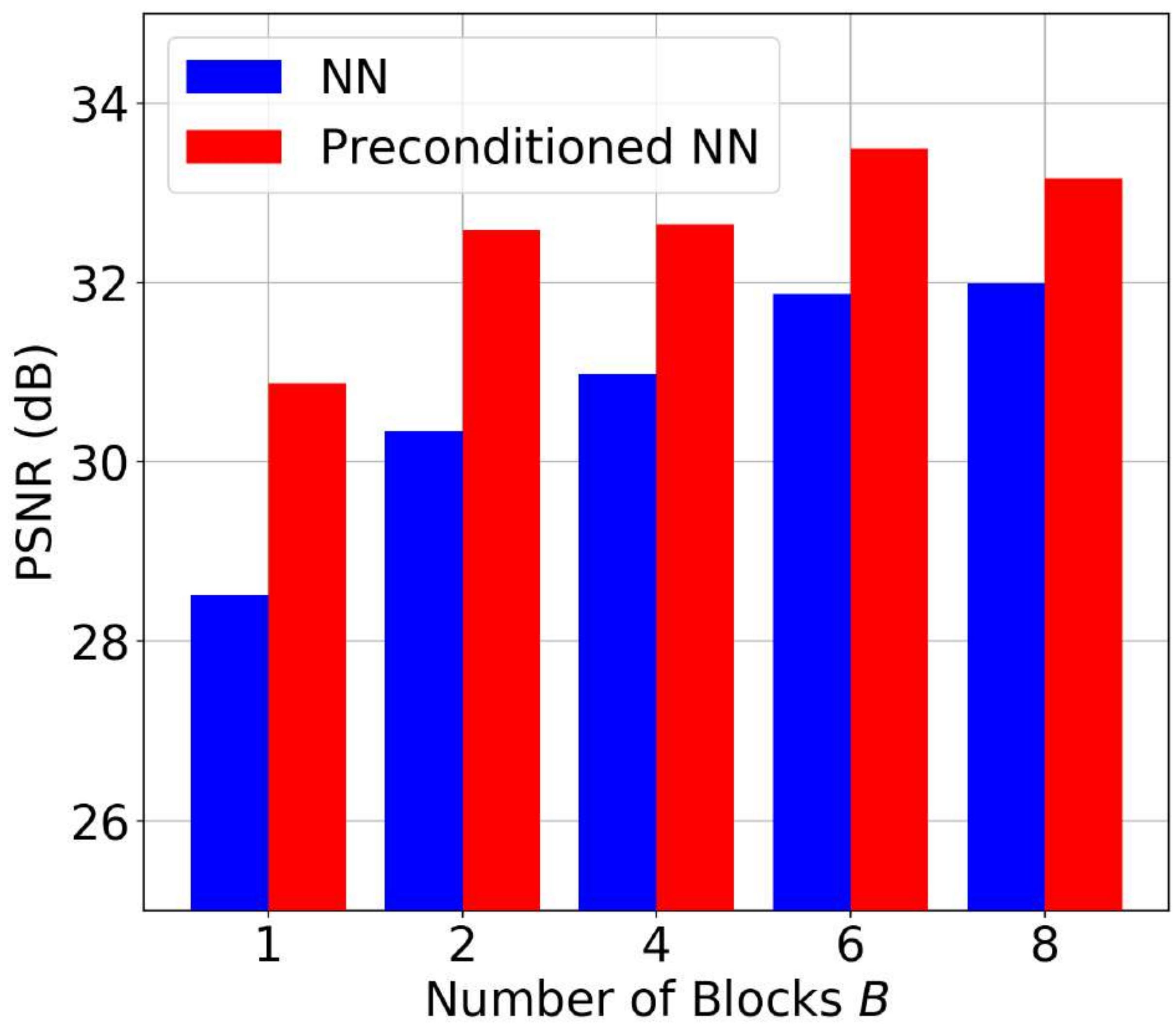
Neumann network: $[I-\eta X^T X]$ is linear and ∇r is nonlinear

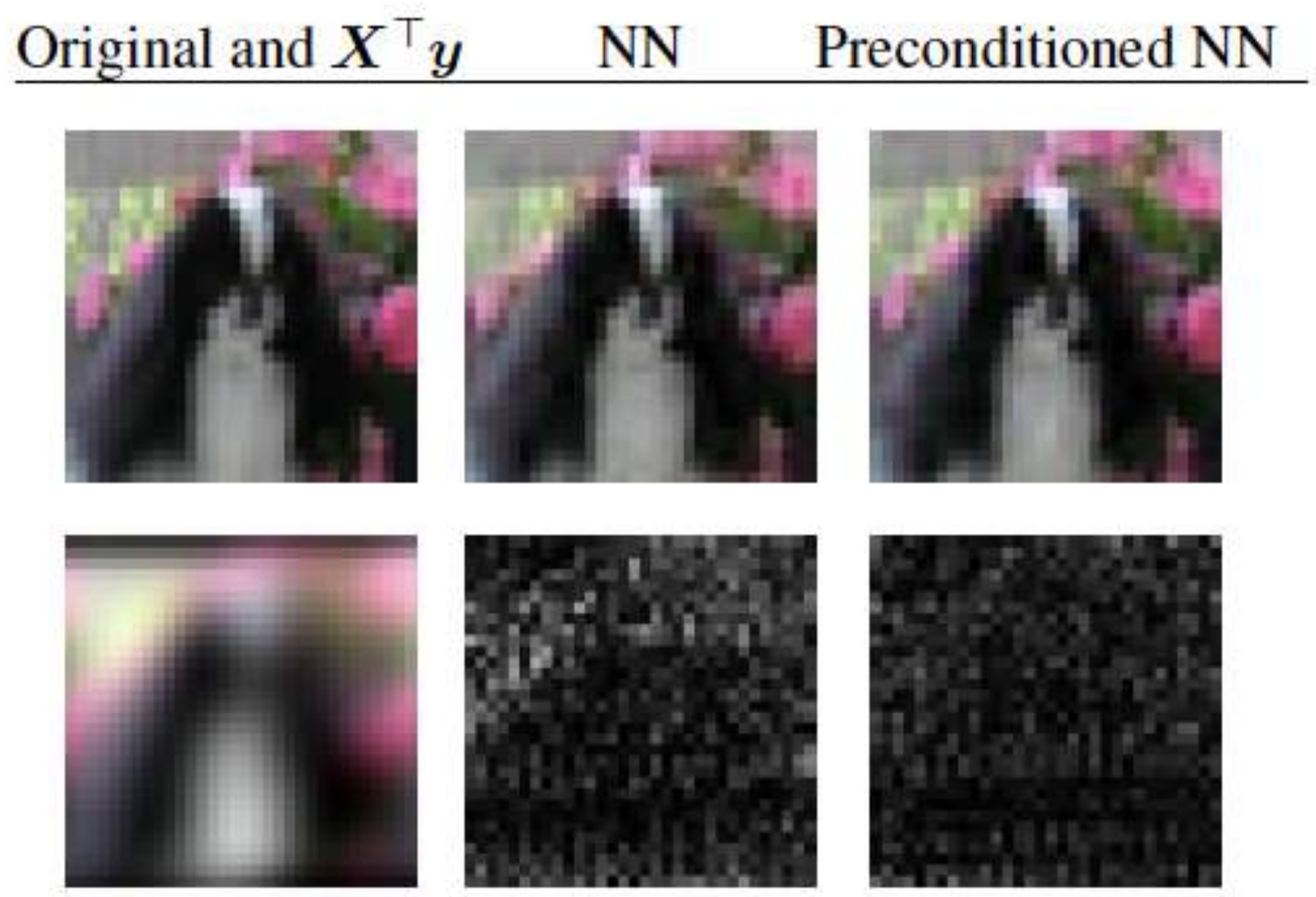


Preconditioned Neumann net: $\eta\lambda[I+\lambda X^T X]^{-1}$ is linear and ∇r nonlinear



Preconditioning





Theory

Neumann series for nonlinear operators?

If A is a *nonlinear* operator, Neumann series identity does not hold:

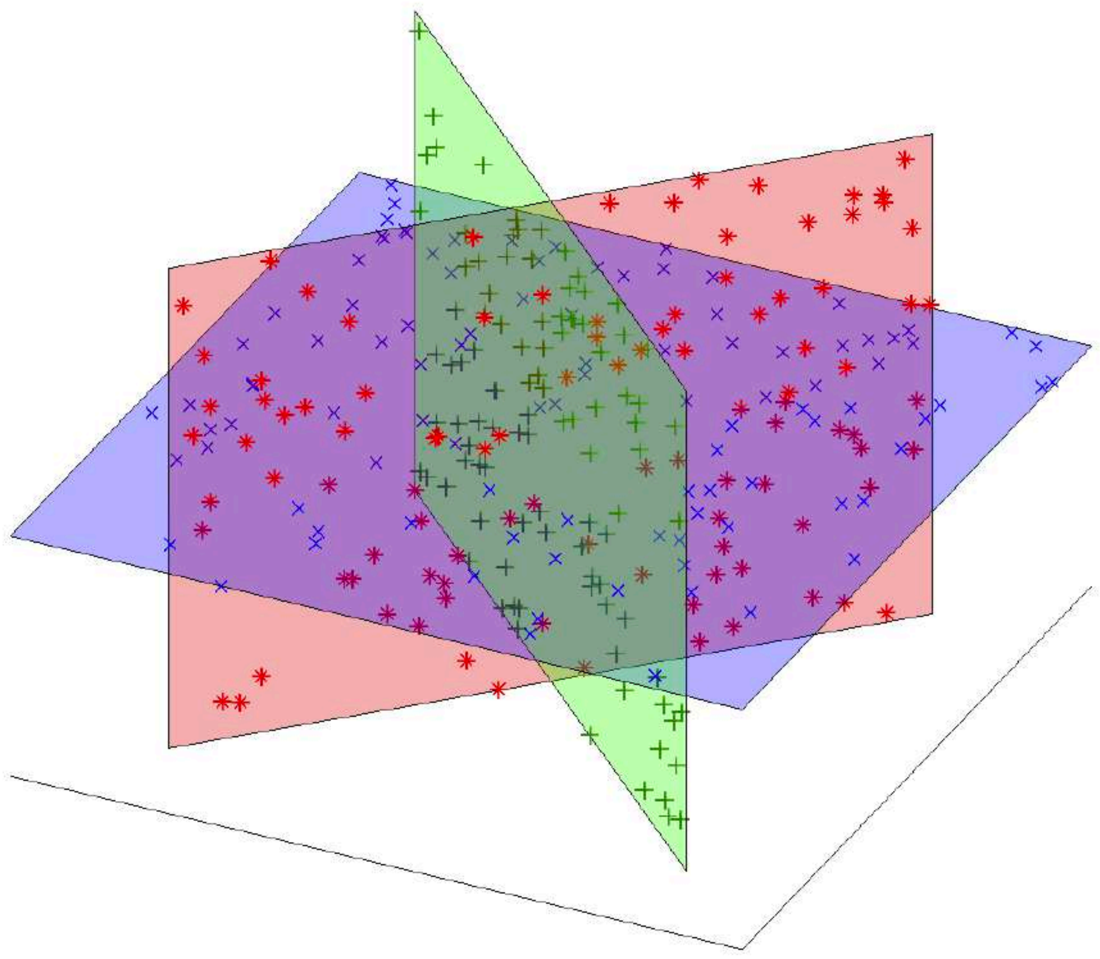
$$(I - A)^{-1} \neq \sum_{k=0}^{\infty} A^k$$

In our case, $A = I - \eta X^T X - \eta R$, where $R = \nabla r$ may be nonlinear

Can we justify Neumann net as an estimator beyond the linear setting?

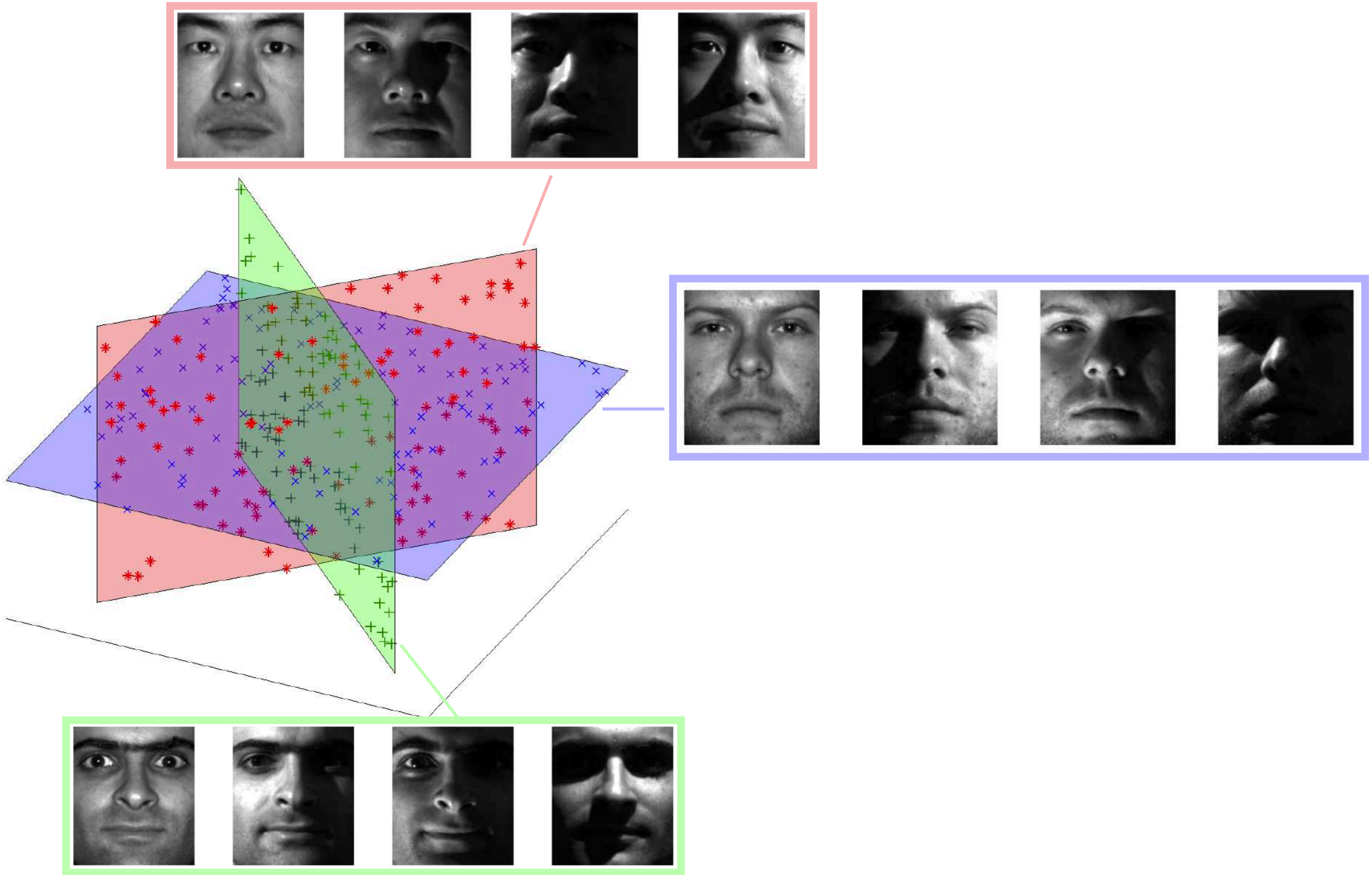
Case Study: Union of Subspaces Models

Model images as belonging to a union of low-dimensional subspaces



Case Study: Union of Subspaces Models

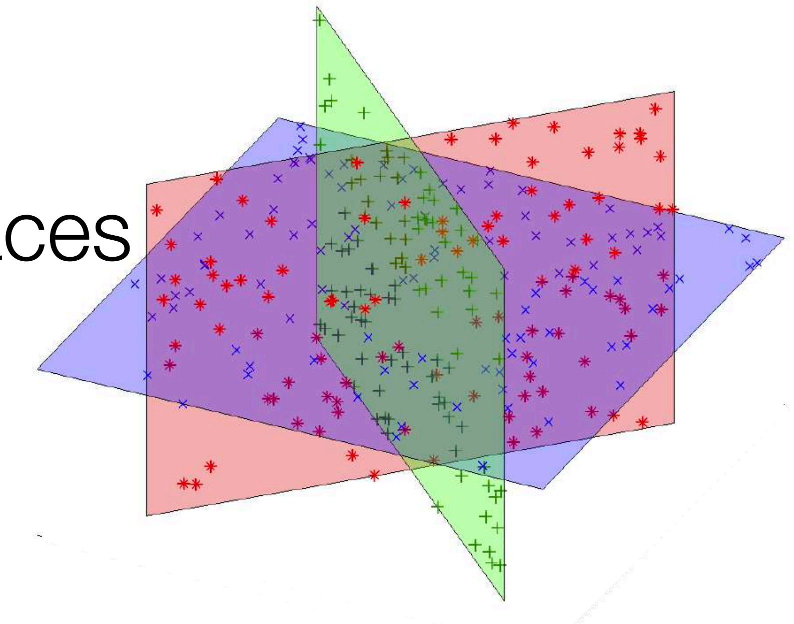
Model images as belonging to a union of low-dimensional subspaces



Images from: Extended Yale B dataset
& http://dhpark22.github.io/greedysc.html

Neumann network estimator

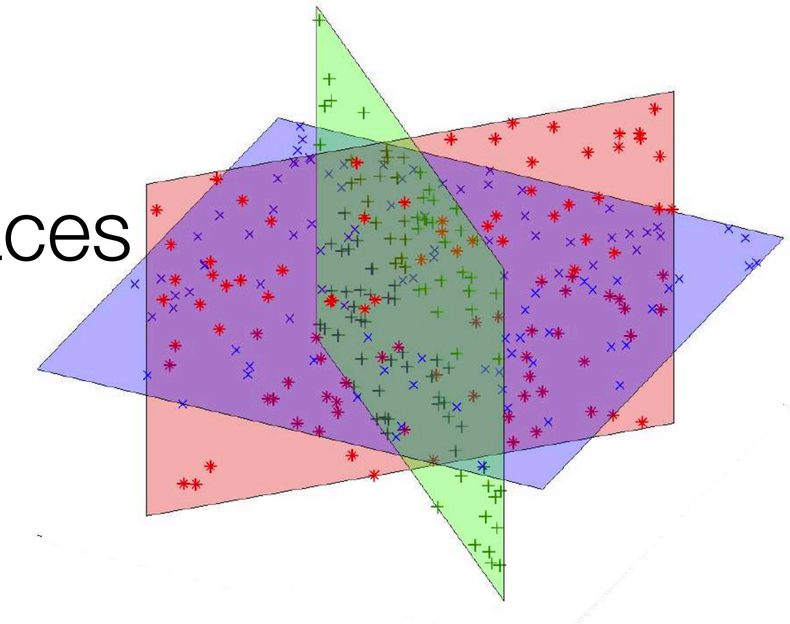
Neumann network estimator



Observe: $y = X\beta + \varepsilon$, β in a union of subspaces

Goal: Recover β from y

Neumann network estimator



Observe: $y = X\beta + \varepsilon$, β in a union of subspaces

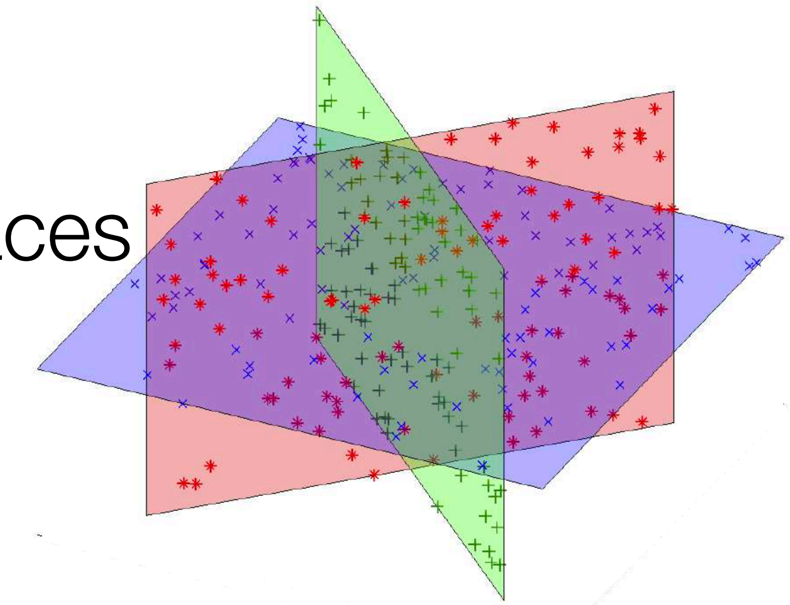
Goal: Recover β from y

Consider the *Neumann network estimator*

$$\hat{\beta}(y) := \sum_{j=0}^{B} (I - \eta X^\top X - \eta R)^j (\eta X^\top y)$$

where	$\eta > 0$	"step size"	
	$R: \mathbb{R}^p \to \mathbb{R}^p$	"learned component"	are "parameters".
	B	"number of blocks"	

Neumann network estimator



Observe: $y = X\beta + \varepsilon$, β in a union of subspaces

Goal: Recover β from y

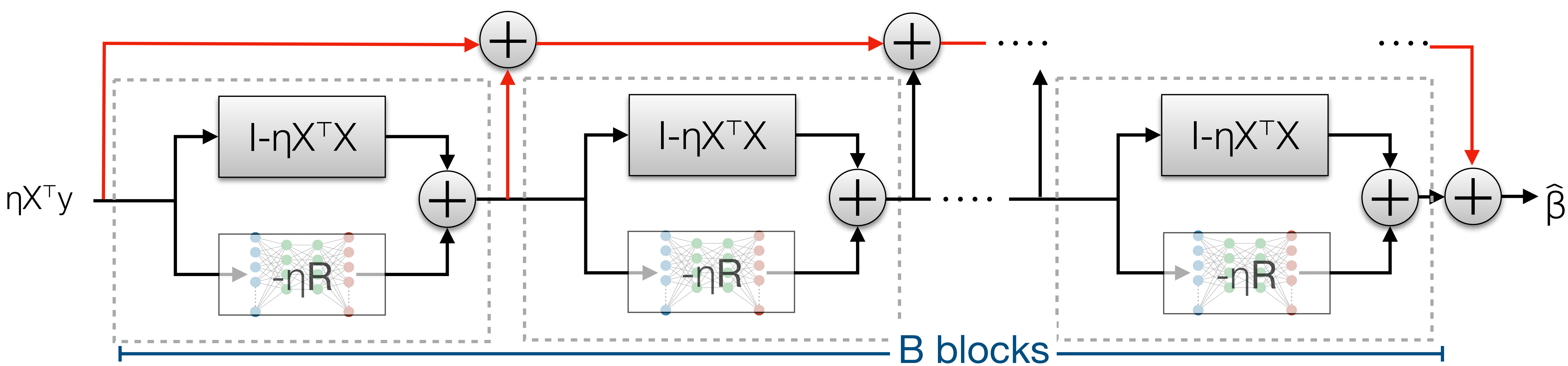
Consider the *Neumann network estimator*

$$\hat{\beta}(y) := \sum_{j=0}^{B} (I - \eta X^{\top} X - \eta R)^{j} (\eta X^{\top} y)$$

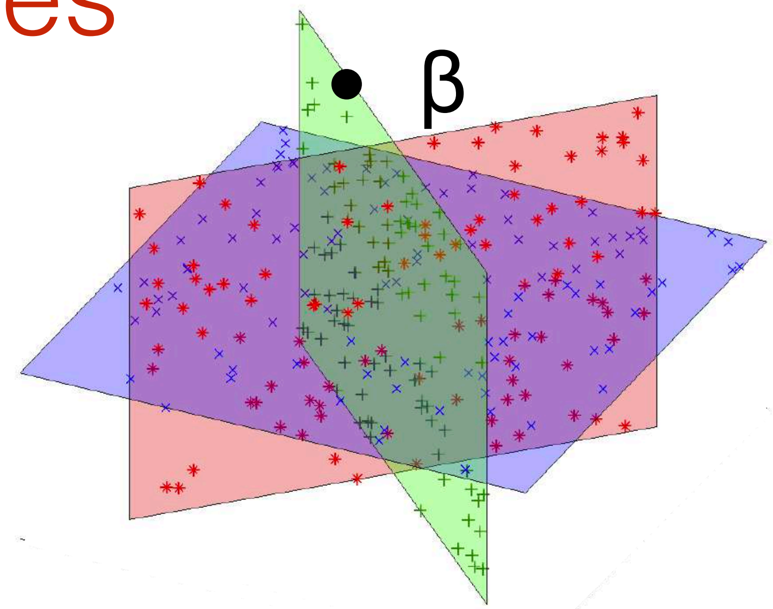
where $\eta > 0$ "step size"

$R: \mathbb{R}^p \to \mathbb{R}^p$ "learned component" are "parameters".

B "number of blocks"



Neumann nets and union of subspaces



For simplicity, assume:

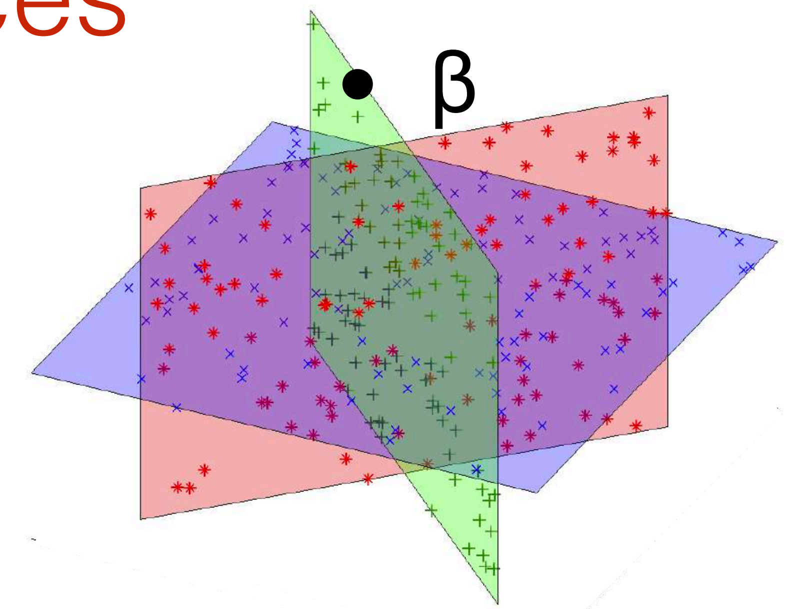
- X has orthonormal rows
- measurements are noise-free: $y = X\beta \in \mathbb{R}^m$
- maximum subspace dimension < m/2
- the union of subspaces is “generic”

Lemma:

- Optimal “oracle” regularizer gradient R is piecewise linear in β
- Neumann network with ReLU activations can closely approximate this oracle
- The output of each block is closest to the same subspace
  $\Rightarrow$ for a fixed input, R behaves linearly

  $\Rightarrow$ Neumann series foundation is justifiable and accurate

Neumann nets and union of subspaces



For simplicity, assume:

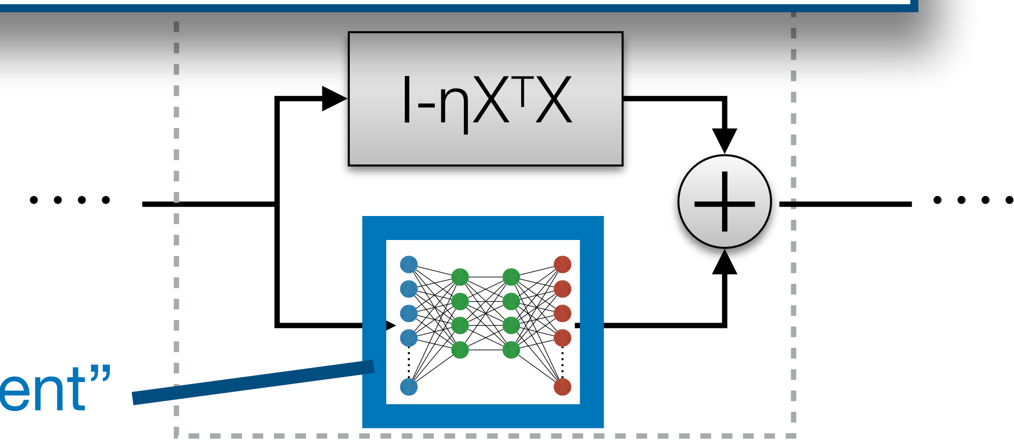
- X has orthonormal rows
- measurements are noise-free: $y = X\beta \in \mathbb{R}^m$
- maximum subspace dimension < m/2
- the union of subspaces is "generic"

Theorem (informal):

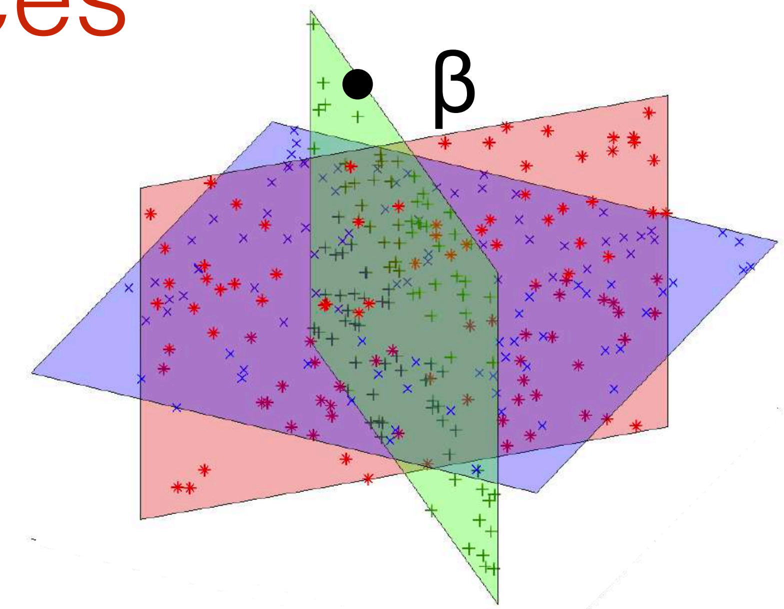
For a given step size $0 < \eta < 1$ and number of blocks B there exists a Neumann network estimator $\hat{\beta}(X\beta)$ with a piecewise linear learned component such that

$$\|\hat{\beta}(X\beta) - \beta\| \leq (1-\eta)^{B+1}\|X\beta\|$$

for all β in the union of subspaces.



Neumann nets and union of subspaces



For simplicity, assume:

- X has orthonormal rows
- measurements are noise-free: $y = X\beta \in \mathbb{R}^m$
- maximum subspace dimension < m/2
- the union of subspaces is “generic”

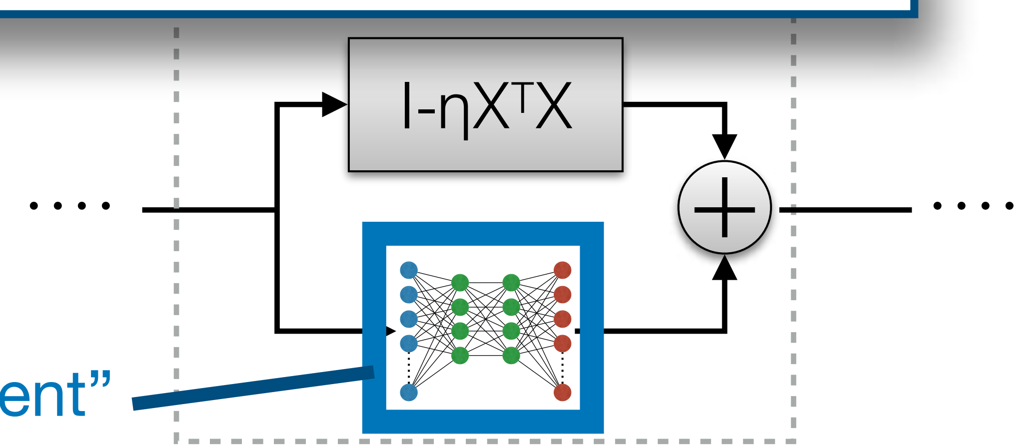
Theorem (informal):

For a given step size $0 < \eta < 1$ and number of blocks B there exists a Neumann network estimator $\hat{\beta}(X\beta)$ with a piecewise linear learned component such that

$$\|\hat{\beta}(X\beta) - \beta\| \leq (1-\eta)^{B+1}\|X\beta\|$$

for all β in the union of subspaces.

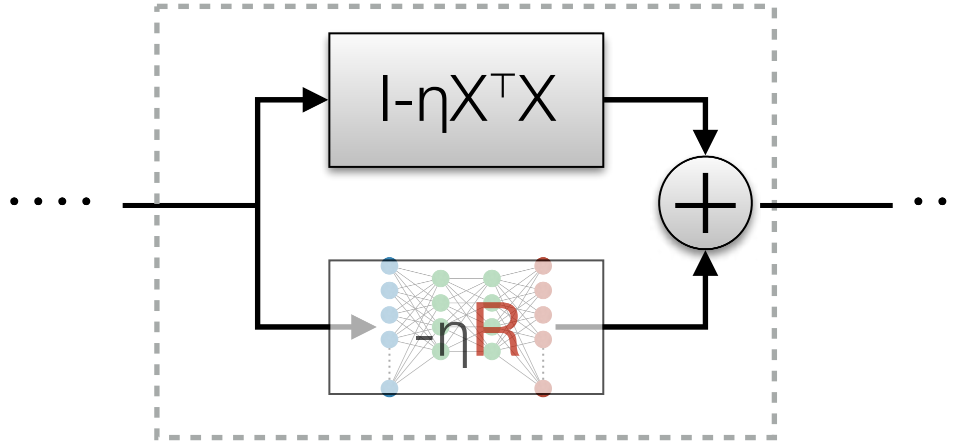
arbitrarily small reconstruction error



“learned component”

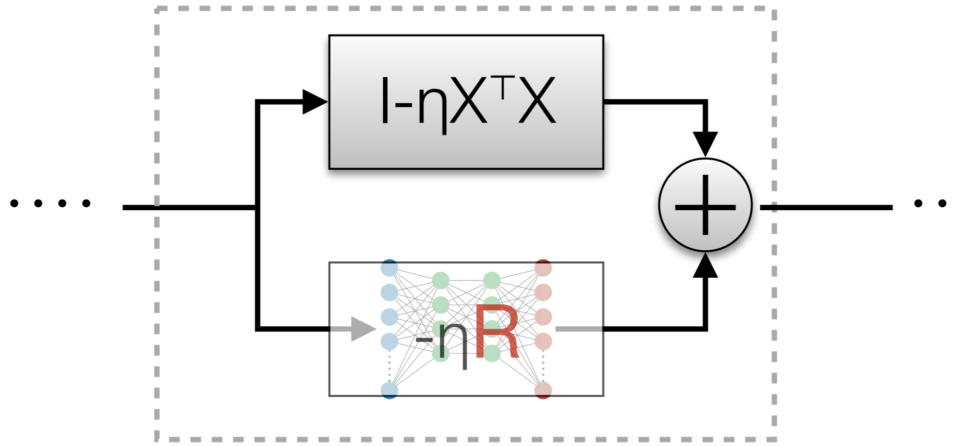
Empirical support for theory

Theorem predicts a specific form R* of learned component R in a Neumann network when trained on vectors in a union of subspaces



Empirical support for theory

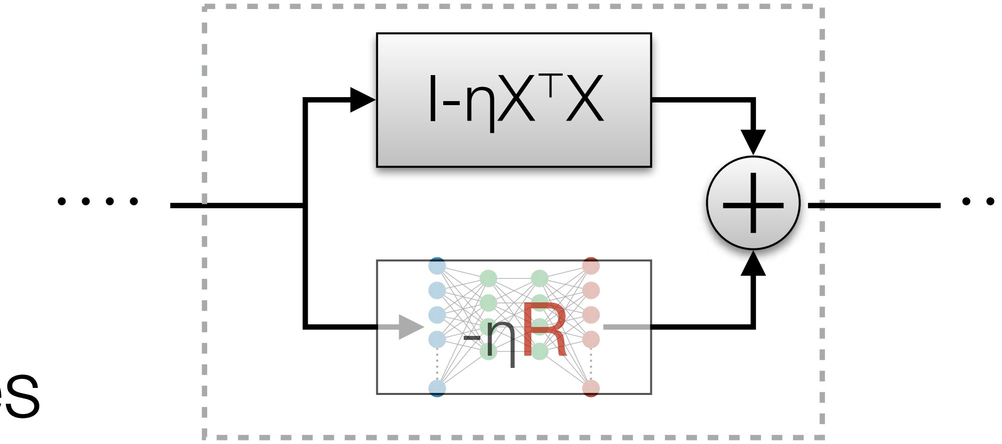
Theorem predicts a specific form R^* of learned component R in a Neumann network when trained on vectors in a union of subspaces



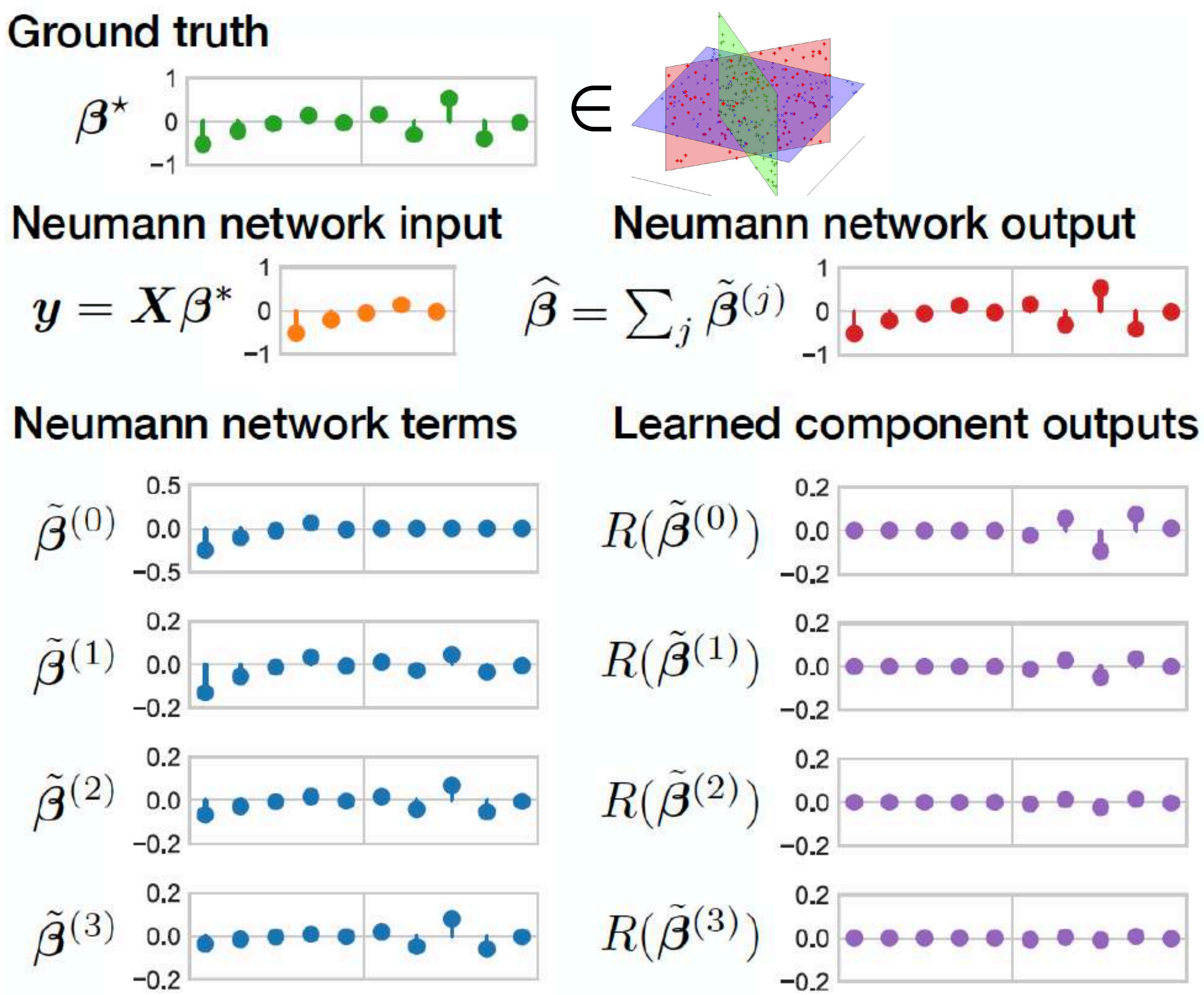
Experiments on synthetic data show that when R is a deep ReLU network, the trained R behaves as the predicted R^*

Empirical support for theory

Theorem predicts a specific form R* of learned component R in a Neumann network when trained on vectors in a union of subspaces

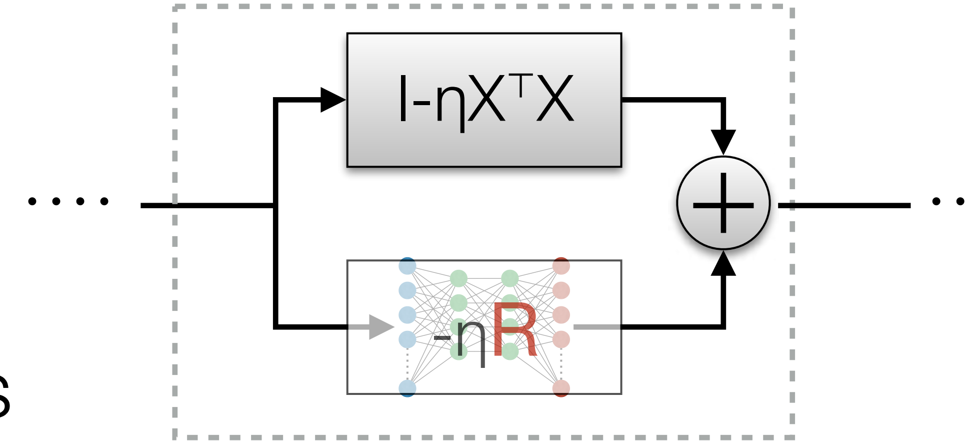


Experiments on synthetic data show that when R is a deep ReLU network, the trained R behaves as the predicted R*

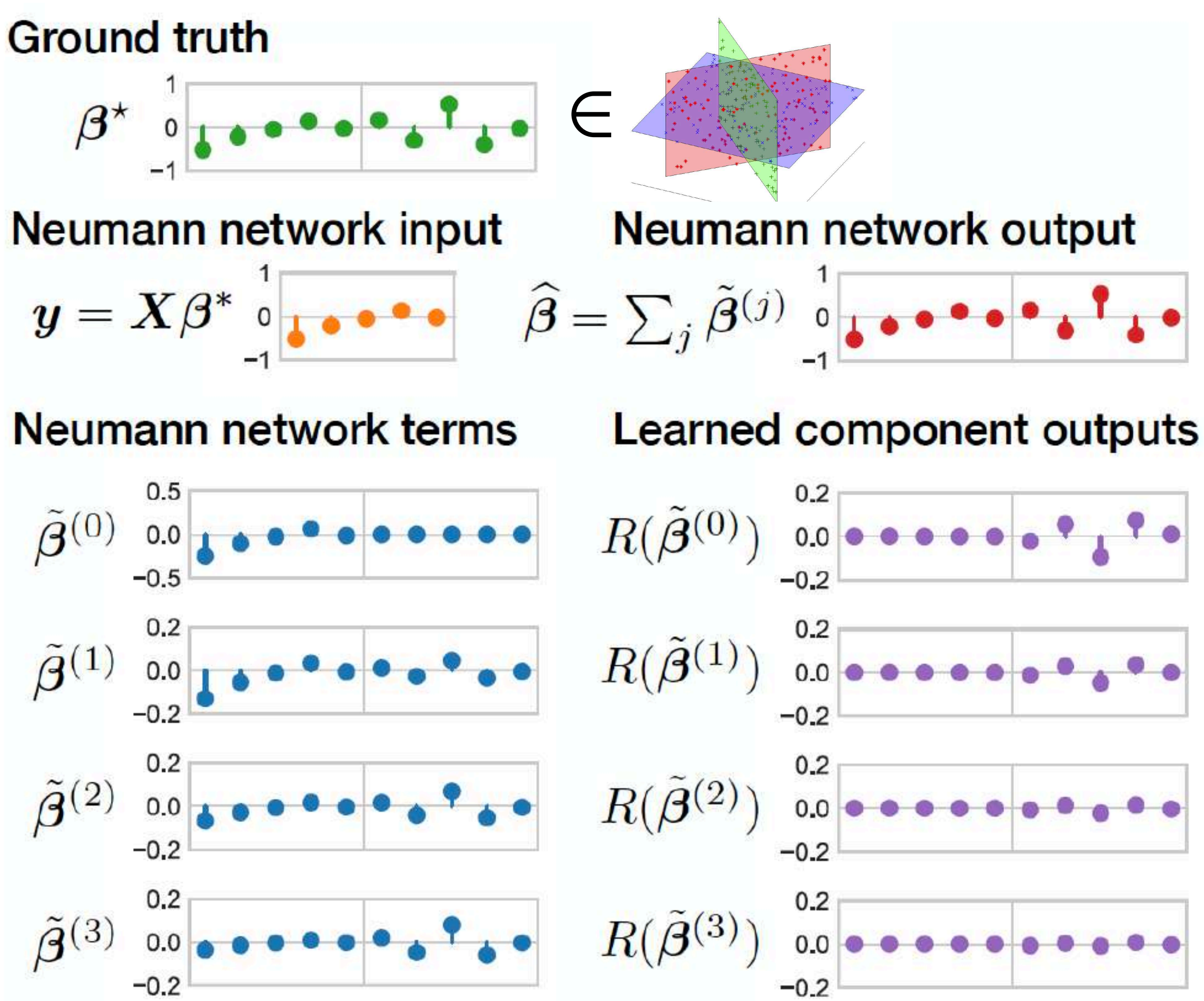


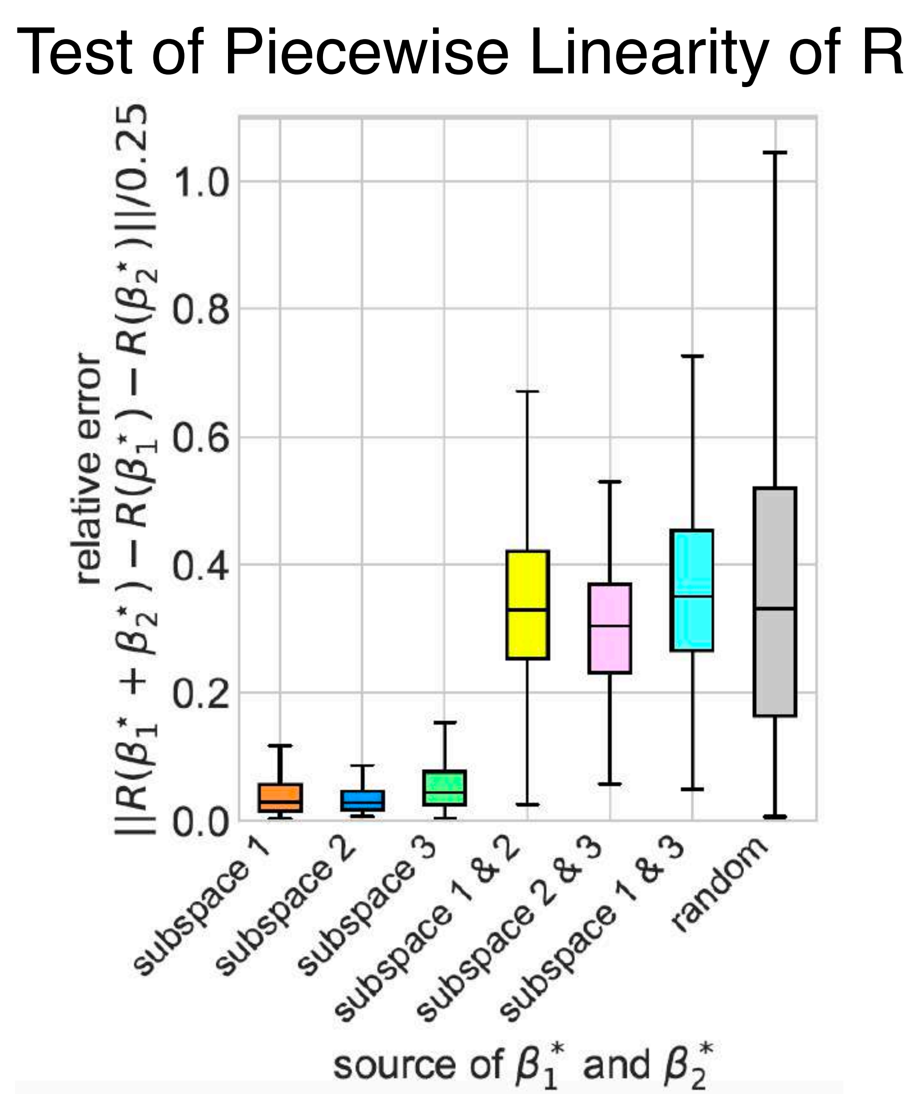
Empirical support for theory

Theorem predicts a specific form R* of learned component R in a Neumann network when trained on vectors in a union of subspaces



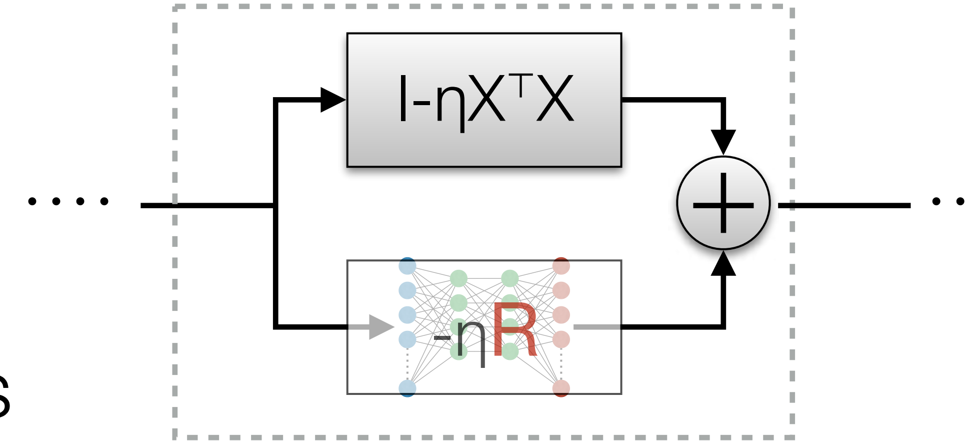
Experiments on synthetic data show that when R is a deep ReLU network, the trained R behaves as the predicted R*



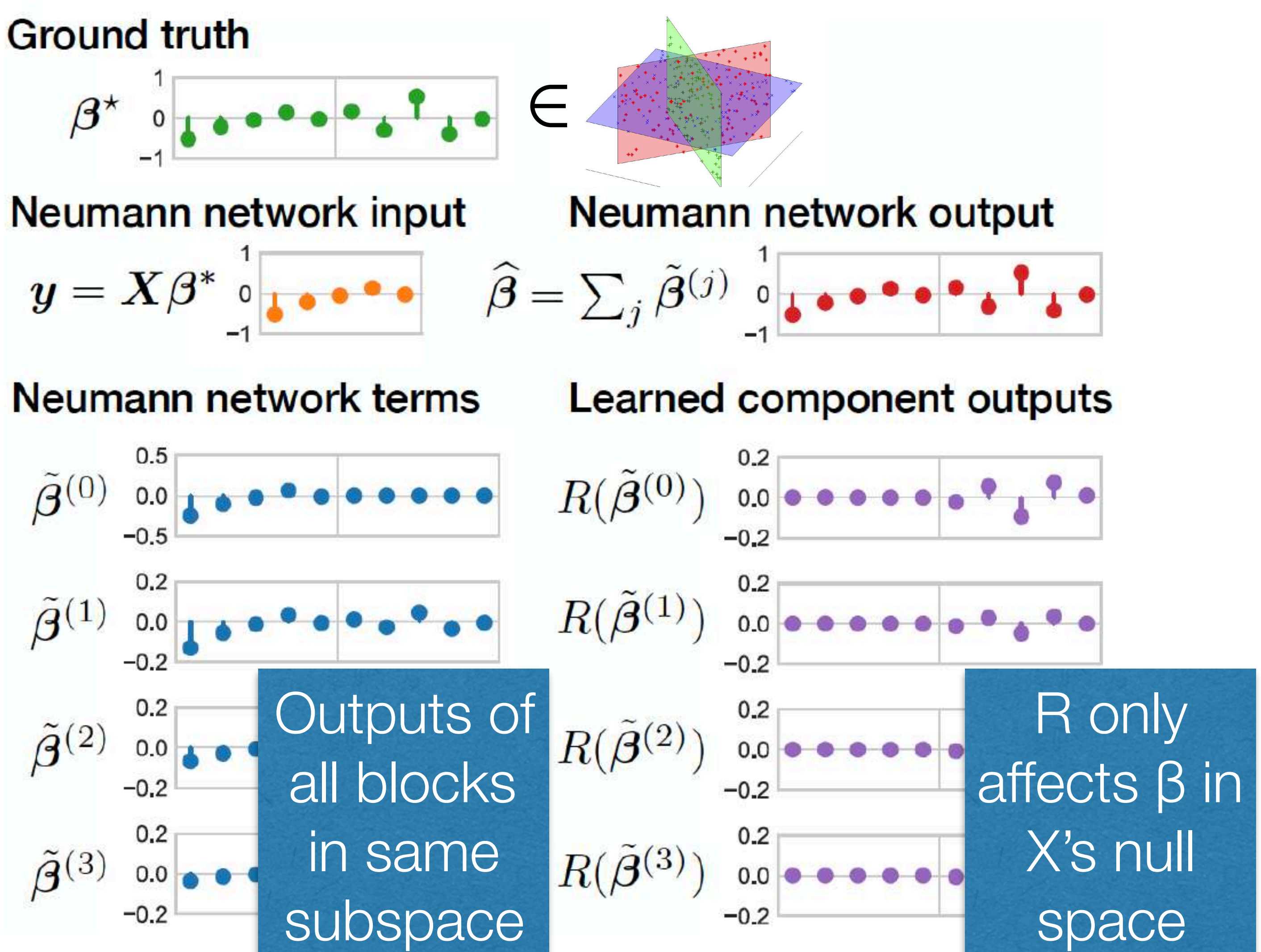


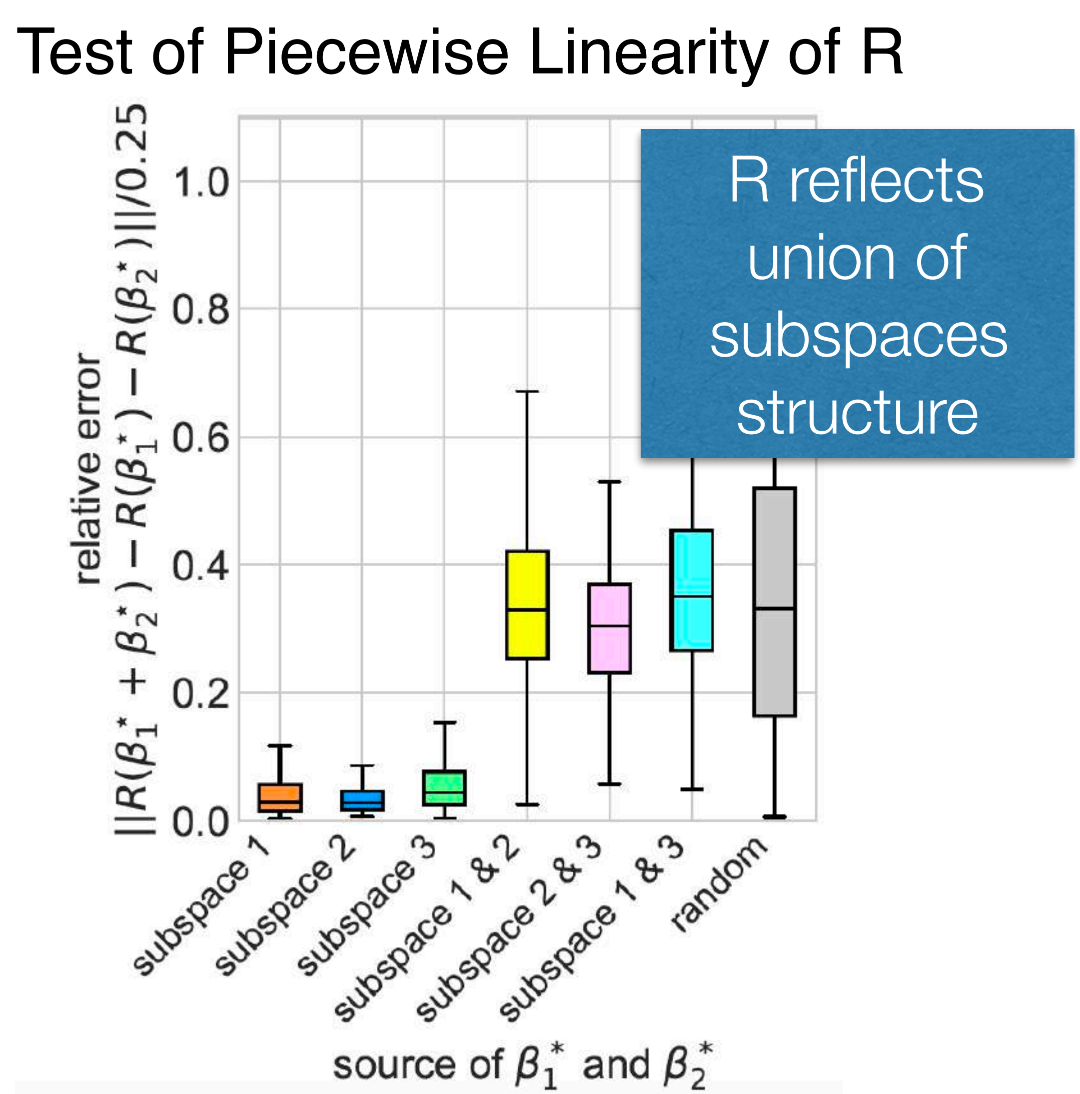
Empirical support for theory

Theorem predicts a specific form R* of learned component R in a Neumann network when trained on vectors in a union of subspaces



Experiments on synthetic data show that when R is a deep ReLU network, the trained R behaves as the predicted R*

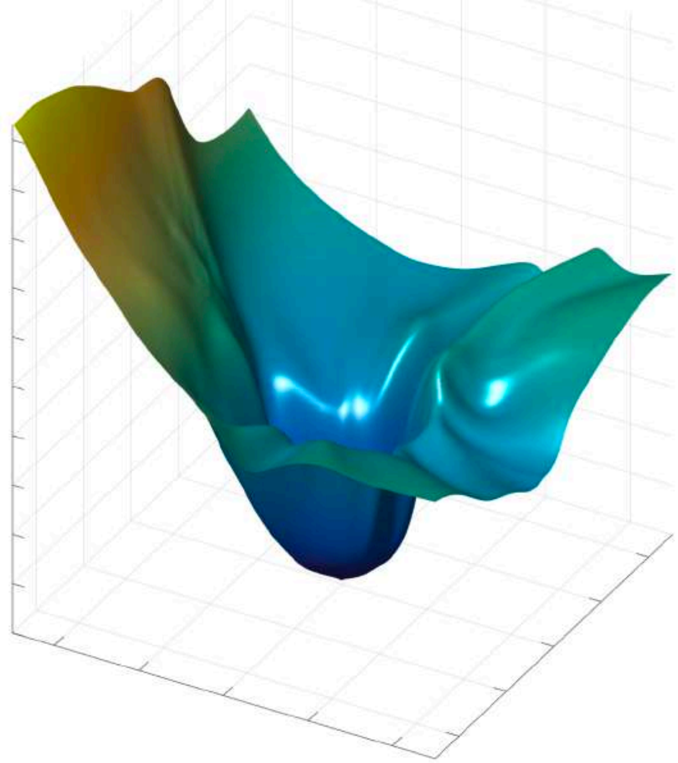




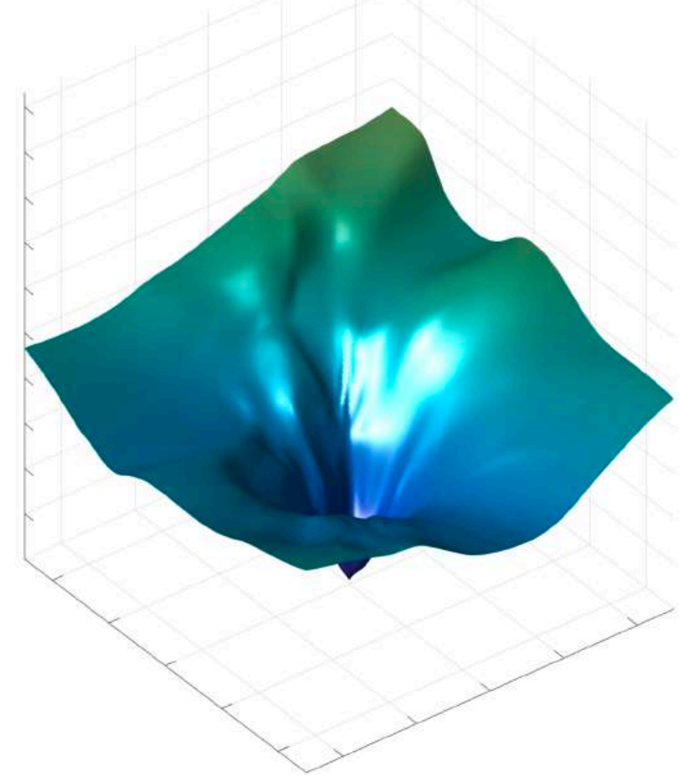
Why do Neumann nets give a performance boost?

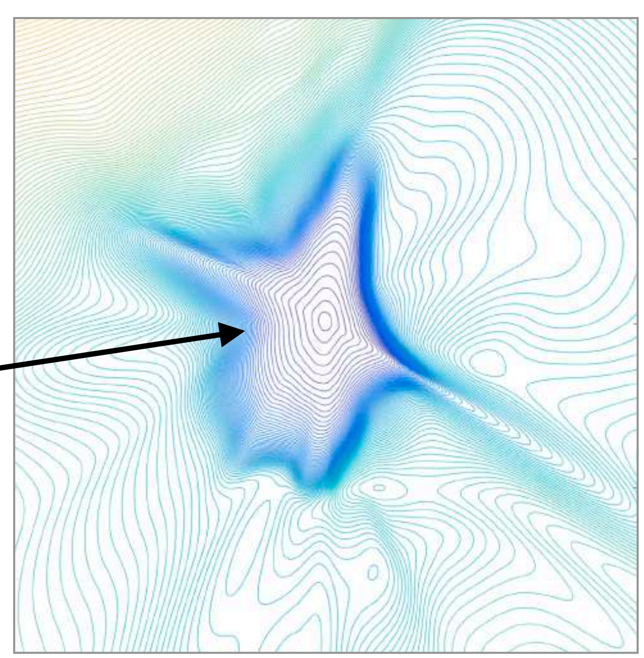
Hypothesis: friendlier optimization landscape

Neumann Network

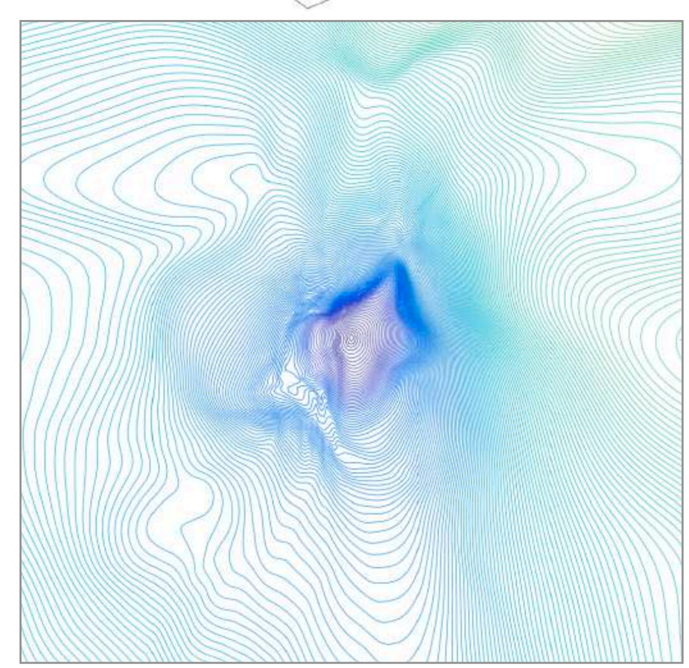


Gradient Descent Network





“Wider” local minimum



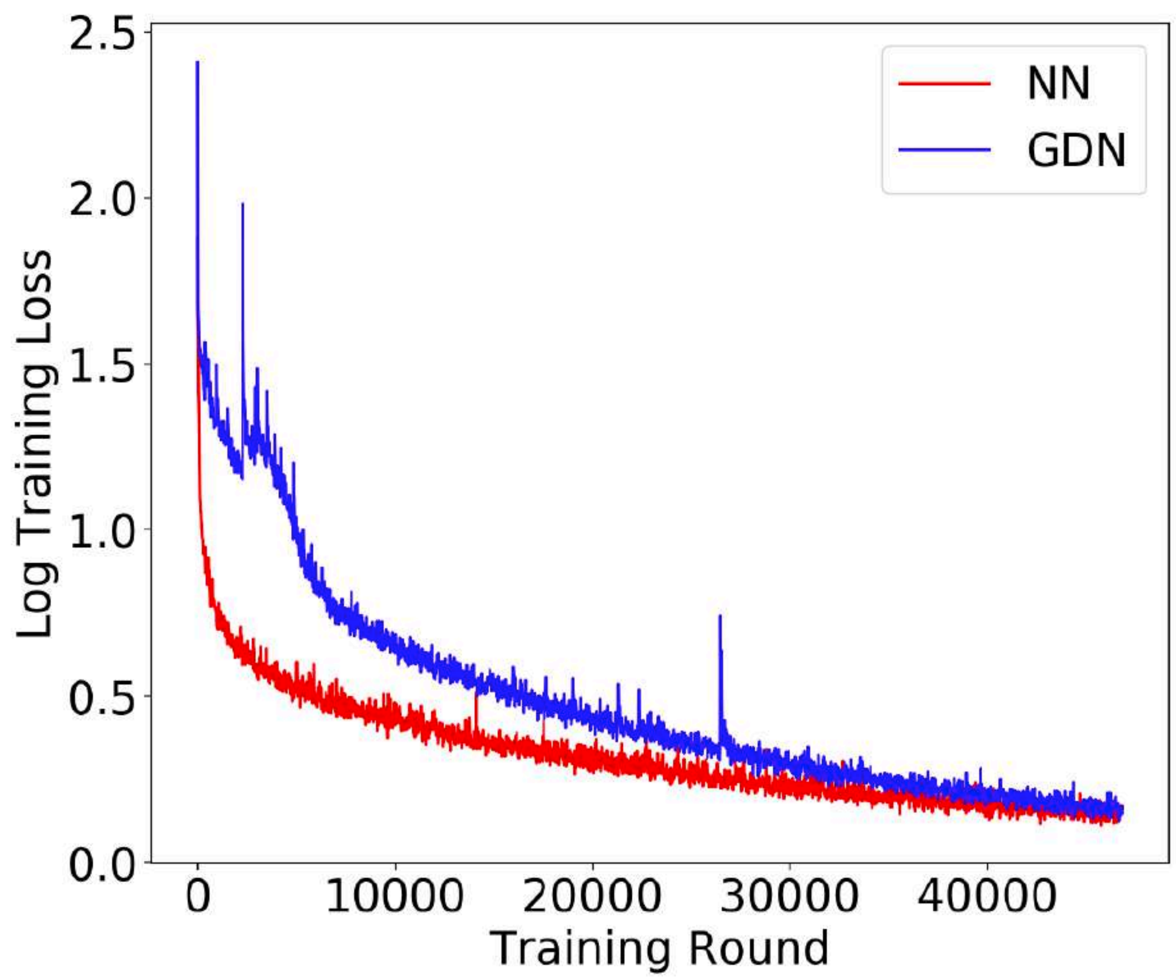
Li, Xu, Taylor, Studer, & Goldstein, 2017

Why do Neumann nets give a performance boost?

Hypothesis: friendlier optimization landscape

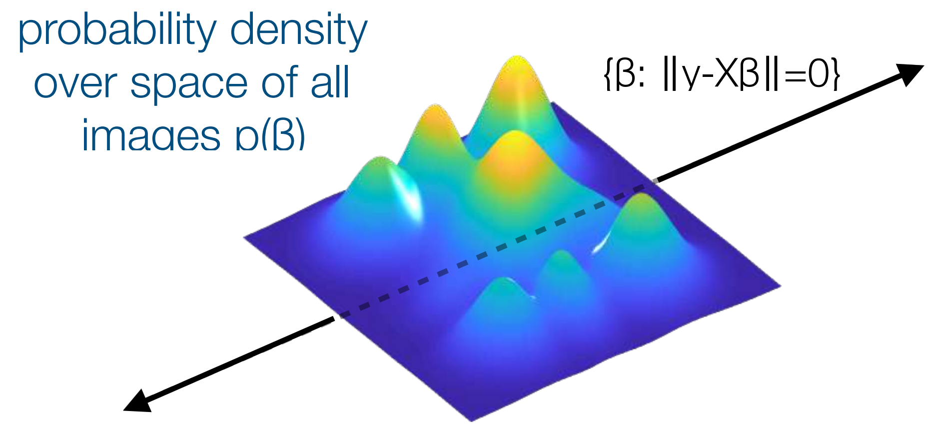
In our experience, gradient descent networks tended to be more sensitive to initialization and step size tuning.

Training curves for Neumann Network (NN)
and Unrolled Gradient Descent (GDN)
on CIFAR10 Deblurring

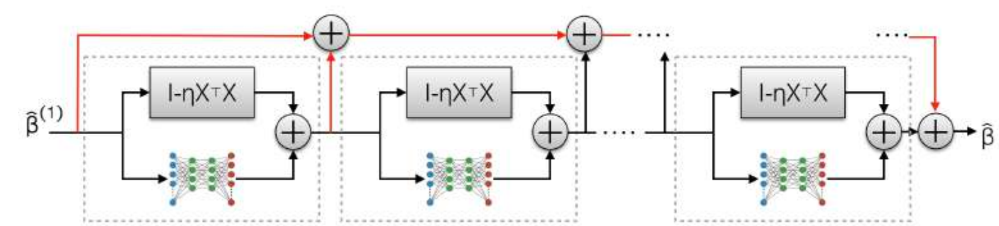


Conclusions

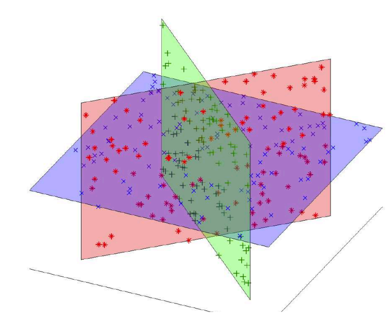
- Explicitly accounting for design (X) during training can dramatically reduce sample complexity.



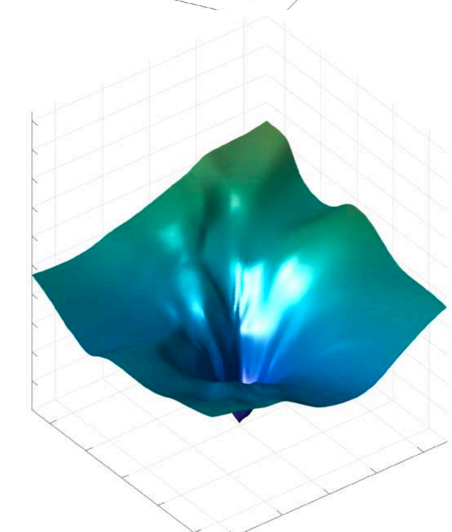
- Networks that include X in training, such as unrolling approaches and Neumann networks, perform well in the low-sample regime.



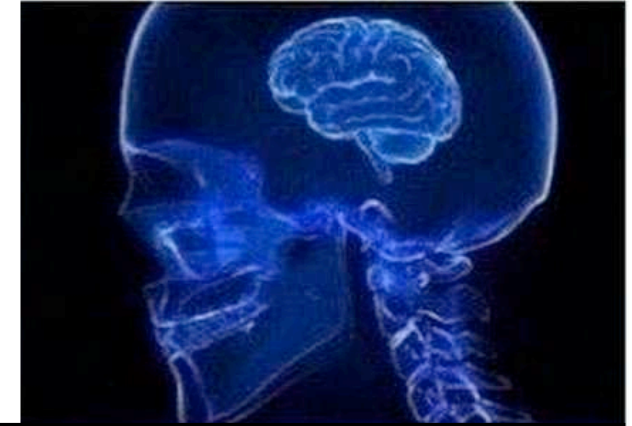
- Neumann networks (and unrolled gradient descent)
  are mathematically justified for union of subspaces.



- Further benefits from Neumann networks, likely due to friendlier optimization landscape.

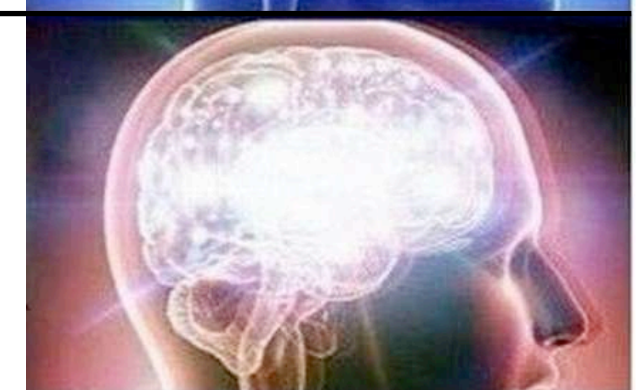


Classical: r(β) is a pre-defined smoothness-promoting regularizer (e.g. Tikhinov or ridge estimation)



Bayesian: r(β) = -log p(β)
Uses a prior distribution over space of β's
(e.g. sparsity, patch redundancy, total variation)

Learned: use training data to learn r(β)



Next: using theory to guide network architecture design

arXiv:1901.03707 [pdf, other] cs.CV cs.LG stat.ML

Neumann Networks for Inverse Problems in Imaging

Authors: Davis Gilton, Greg Ongie, Rebecca Willett

References

- Gregor, K, and LeCun, Y. (2010). ”Learning fast approximations of sparse coding." ICML 2010.
- Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434.
- Mousavi, A., & Baraniuk, R. G. (2017). Learning to invert: Signal recovery via deep convolutional networks. ICASSP 2017.
- Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a deep convolutional network for image super-resolution. ECCV 2014.
- Y. Chen, W. Yu, & T. Pock. (2015). On learning optimized reaction diffusion processes for effective image restoration. CVPR 2015.
- Jin, K. H., McCann, M. T., Froustey, E., & Unser, M. (2017). Deep convolutional neural network for inverse problems in imaging. IEEE TIP, 26(9), 4509-4522.
- Ye, J. C., Han, Y., & Cha, E. (2018). Deep convolutional framelets: A general deep learning framework for inverse problems. SIAM Journal on Imaging Sciences, 11(2), 991-1048.
- Mardani, M., Sun, Q., Vasawanala, S., Papyan, V., Monajemi, H., Pauly, J., & Donoho, D. (2018). Neural Proximal Gradient Descent for Compressive Imaging. arXiv:1806.03963.
- Sun, J., Li, H., & Xu, Z. (2016). Deep ADMM-Net for compressive sensing MRI. NIPS 2016.
- Chang, J. H. R., Li, C. L., Poczos, B., Kumar, B. V., & Sankaranarayanan, A. C. (2016). One Network to Solve Them All-Solving Linear Inverse Problems using Deep Projection Models. ICCV 2016.
- Adler, J., & Öktem, O. (2018). Learned primal-dual reconstruction. IEEE TMI, 37(6), 1322-1332.
- Zhang, K., Zuo, W., Gu, S., & Zhang, L. (2017). Learning deep CNN denoiser prior for image restoration. CVPR 2017.
- Bora, A., Jalal, A., Price, E., & Dimakis, A. G. (2017). Compressed Sensing using Generative Models. ICML 2017.
- Donoho, D. L., & Low, M. G. (1992). Renormalization exponents and optimal pointwise rates of convergence. The Annals of Statistics, 944-970.
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. CVPR 2017.