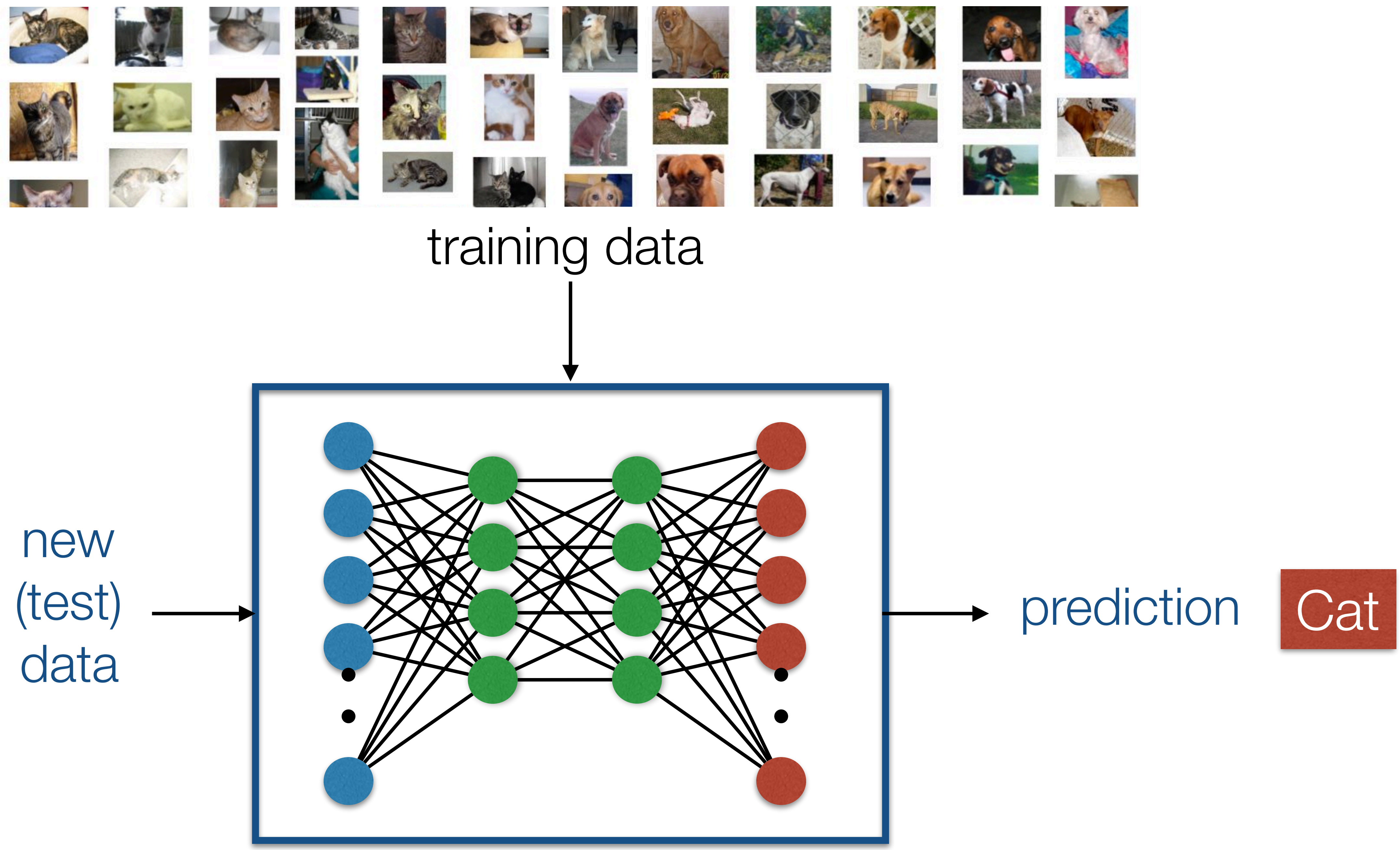
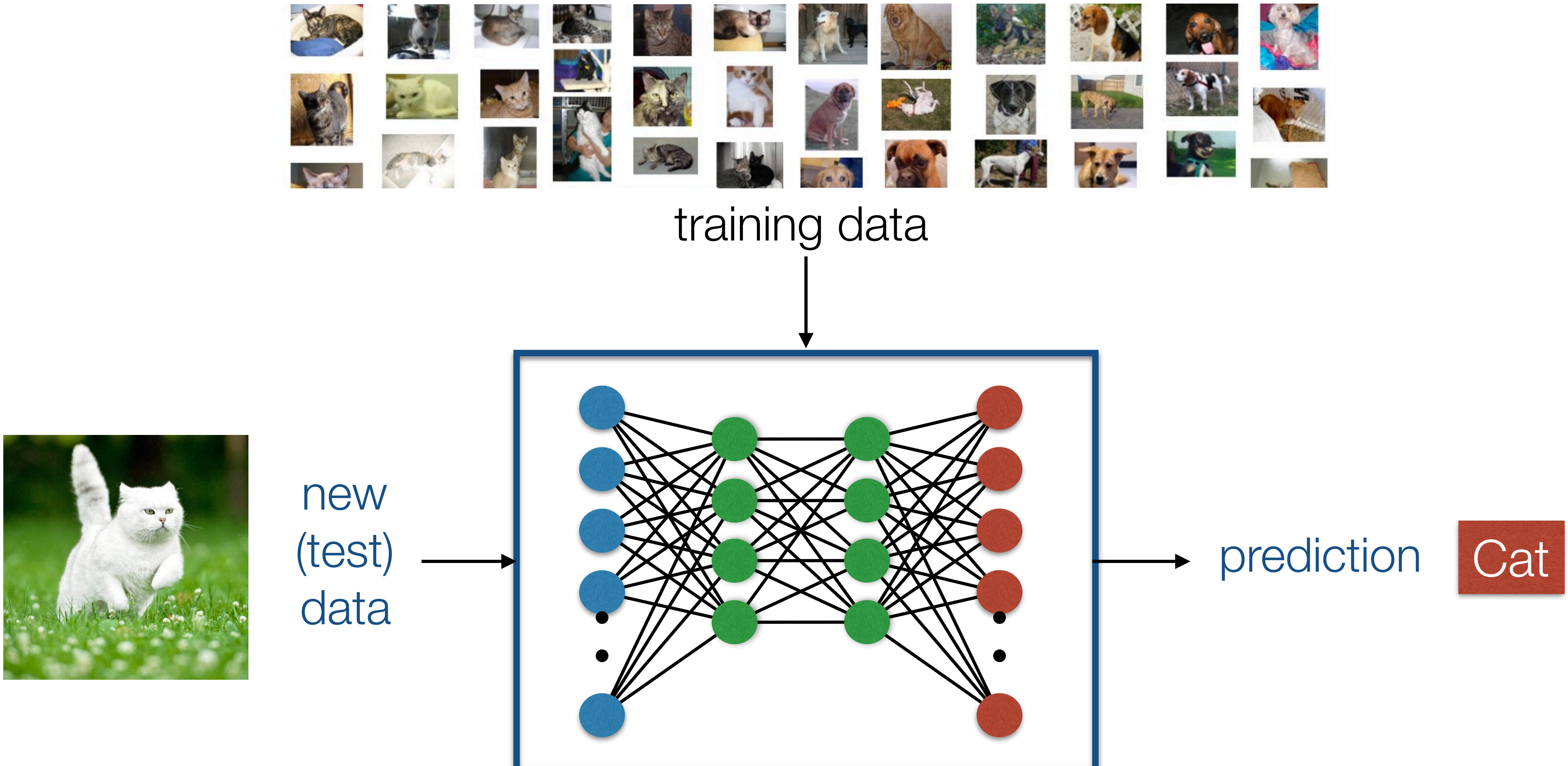
Leveraging Physical Models in Machine Learning

Rebecca Willett
Statistics, Applied Math, and Computer Science

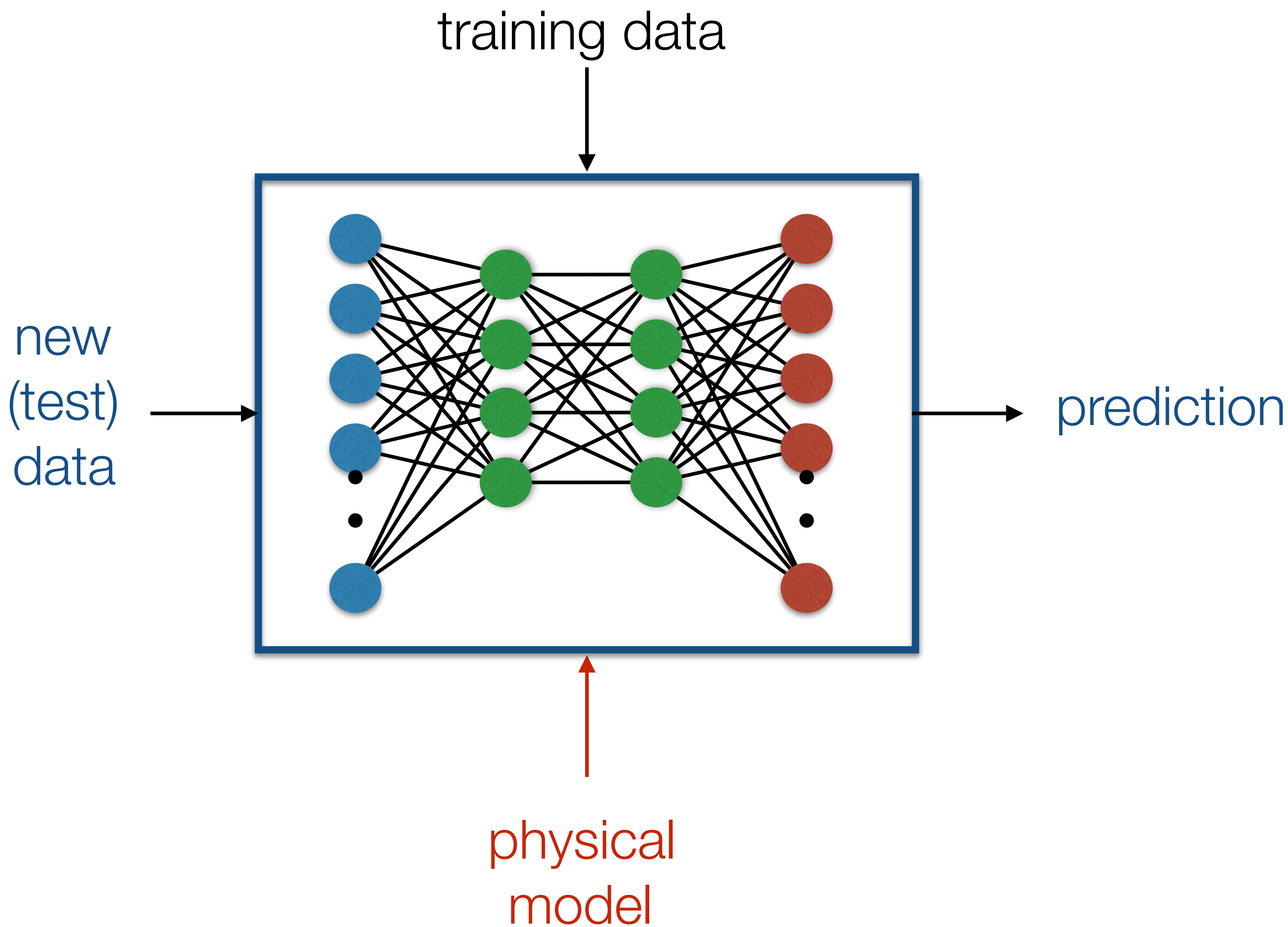
Machine learning



Machine learning



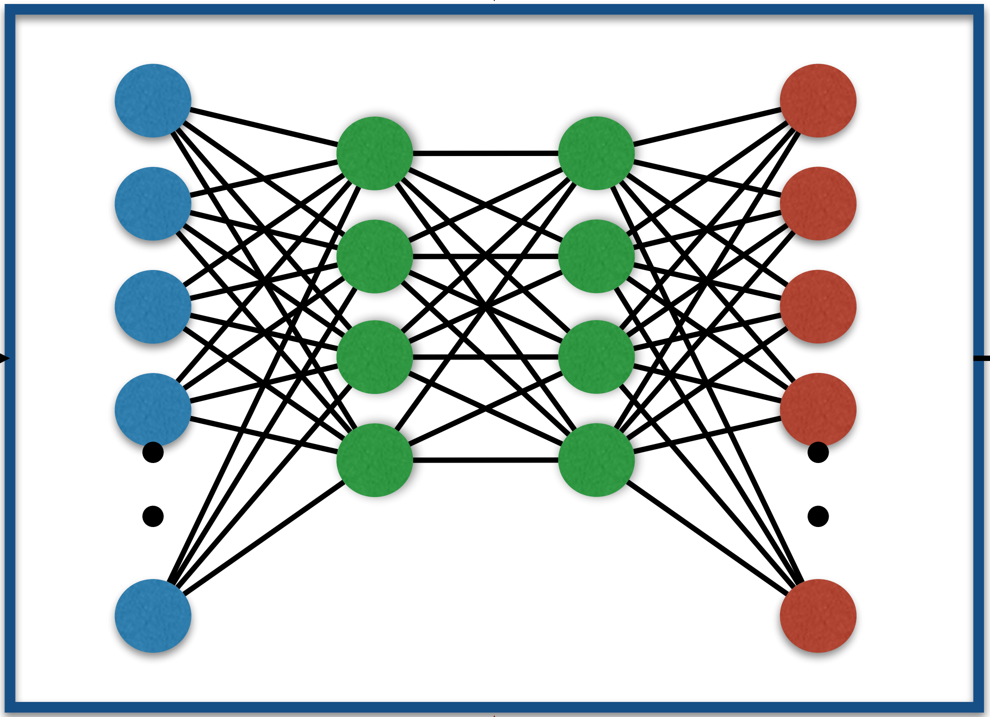
How do we leverage a combination of training data and physical models?



How do we leverage a combination of training data and physical models?

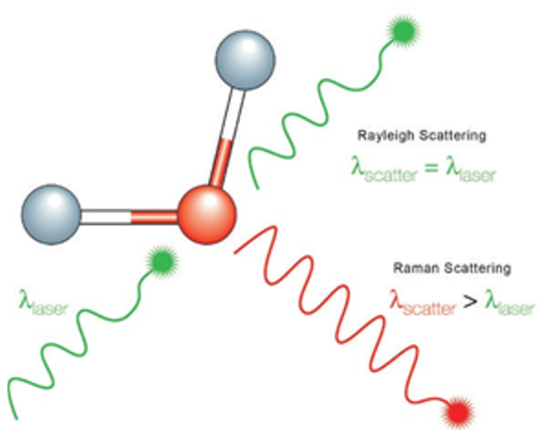
training data

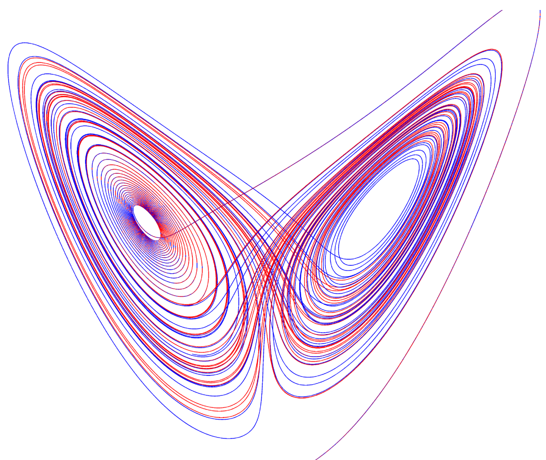
new (test) data

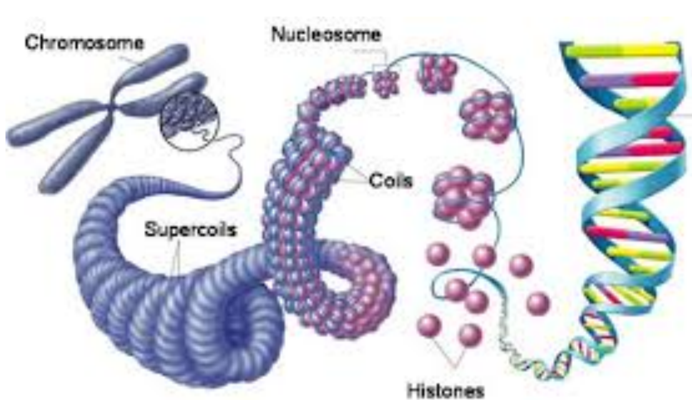


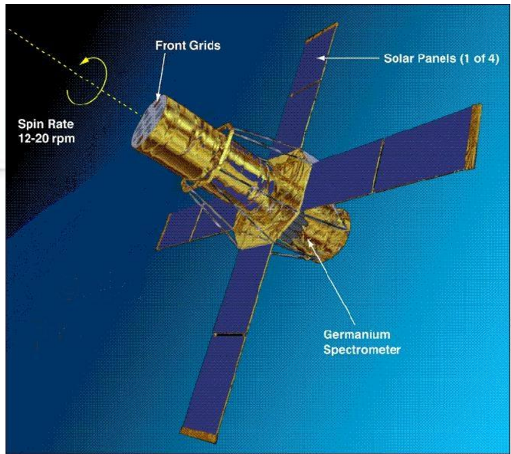
prediction

physical model

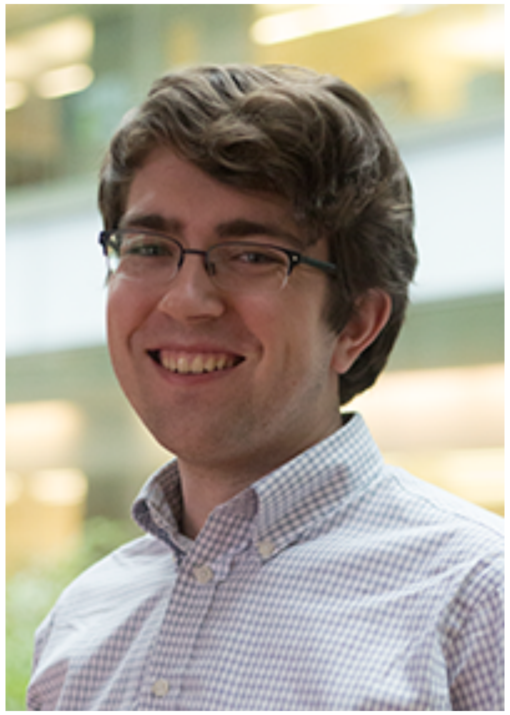








Learning to Solve Inverse Problems in Imaging



Davis Gilton,
UW-Madison

Greg Ongie,
UChicago

Inverse problems in imaging

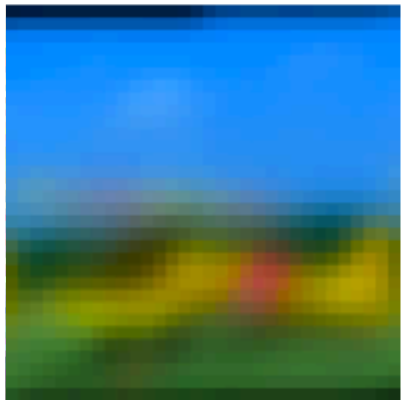
Observe: $y = X\beta + \varepsilon$

Goal: Recover β from y

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

β

y

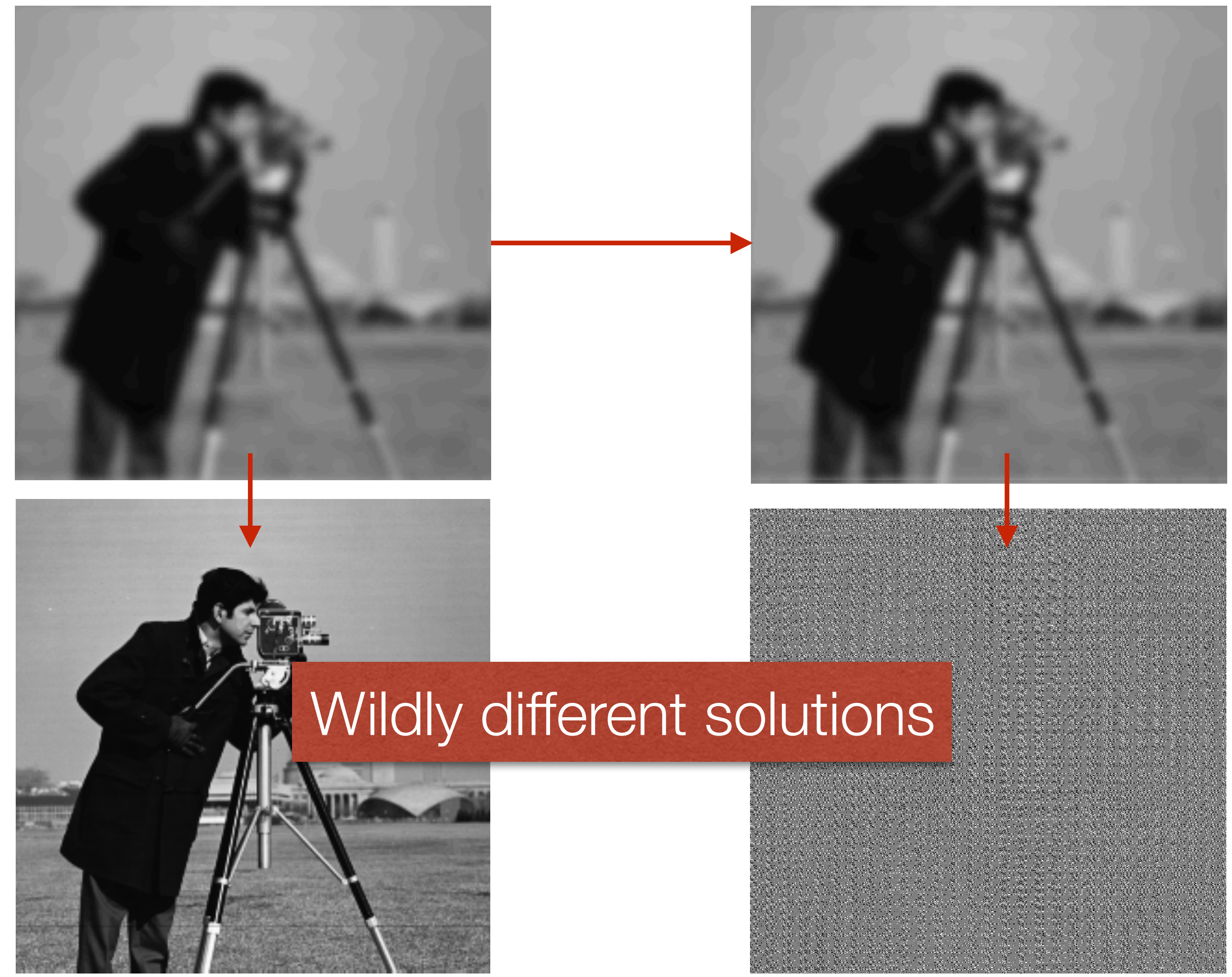


Classical approach: Tikhonov regularization (1943)

- Example: deblurring

- Least squares solution:

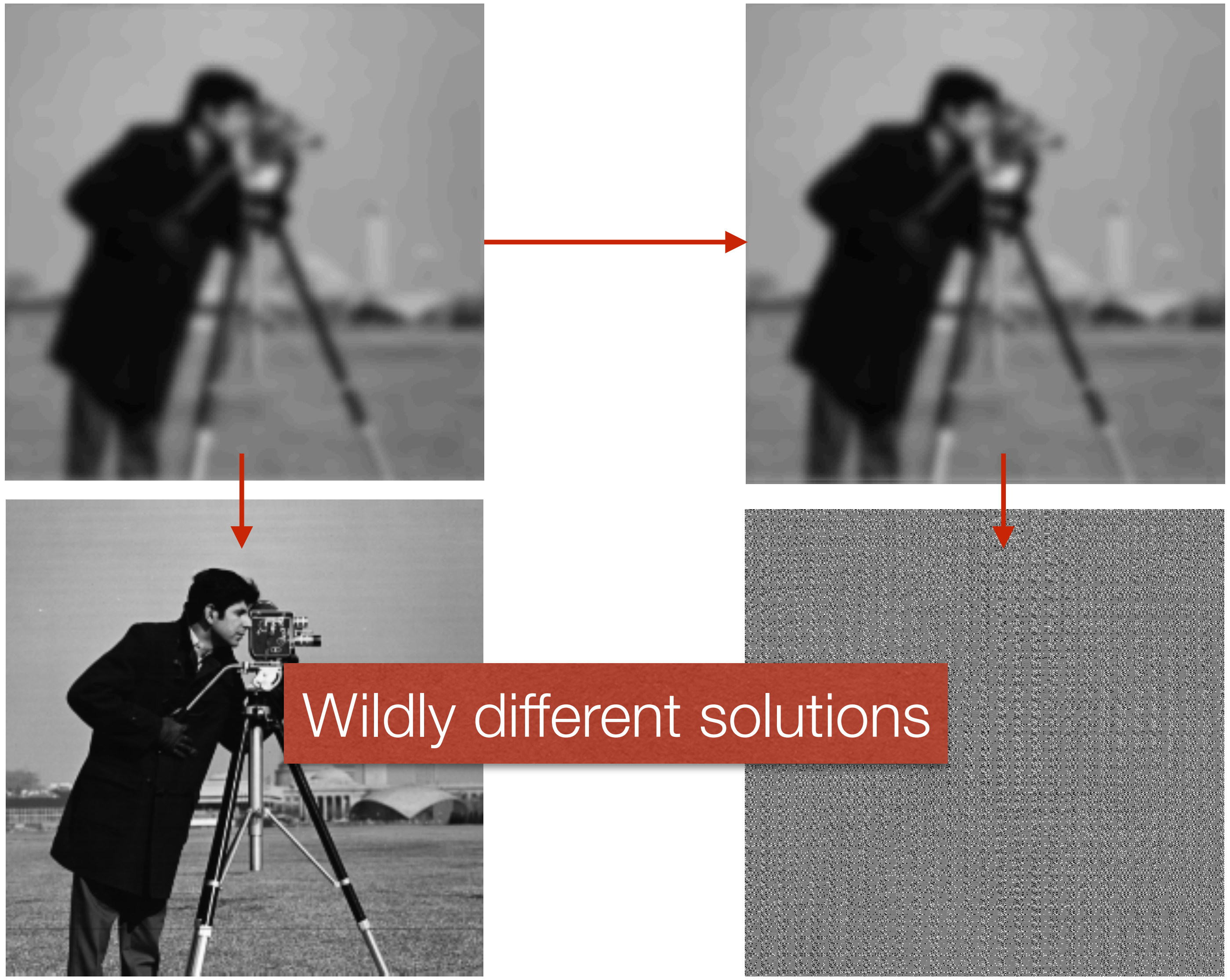
$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$



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

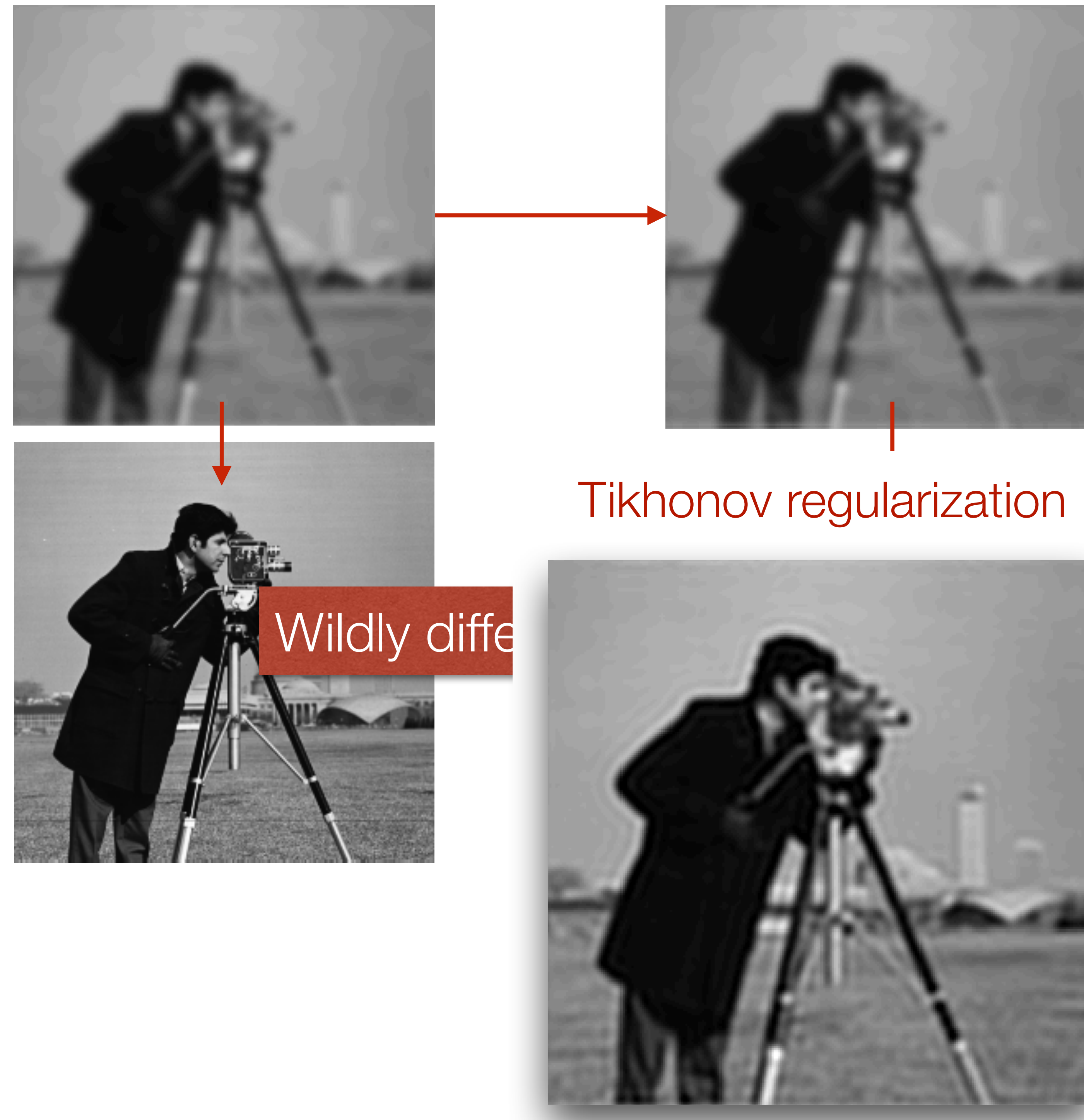
- Least squares solution:

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

- Tikhonov regularization (aka "ridge regression")

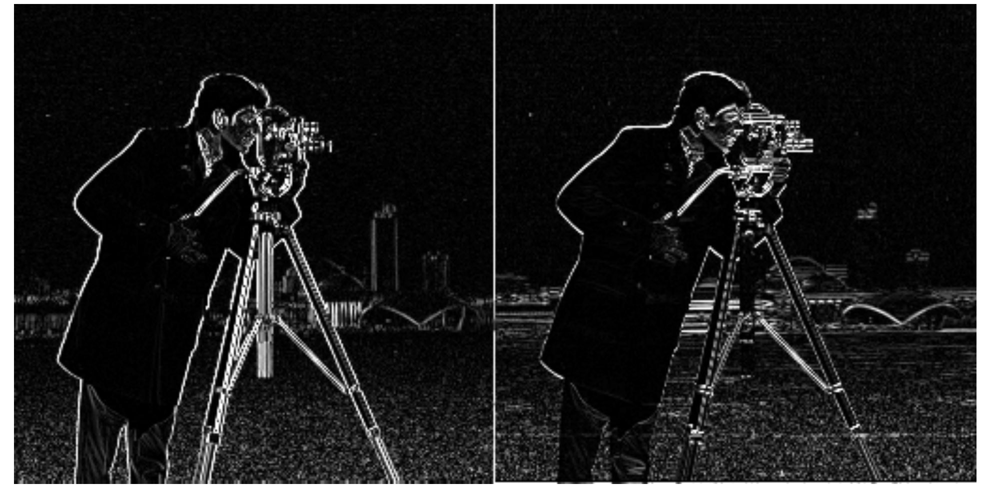
$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2$$
$$= (X^\top X + \lambda I)^{-1} X^\top y$$

better conditioned; suppresses noise



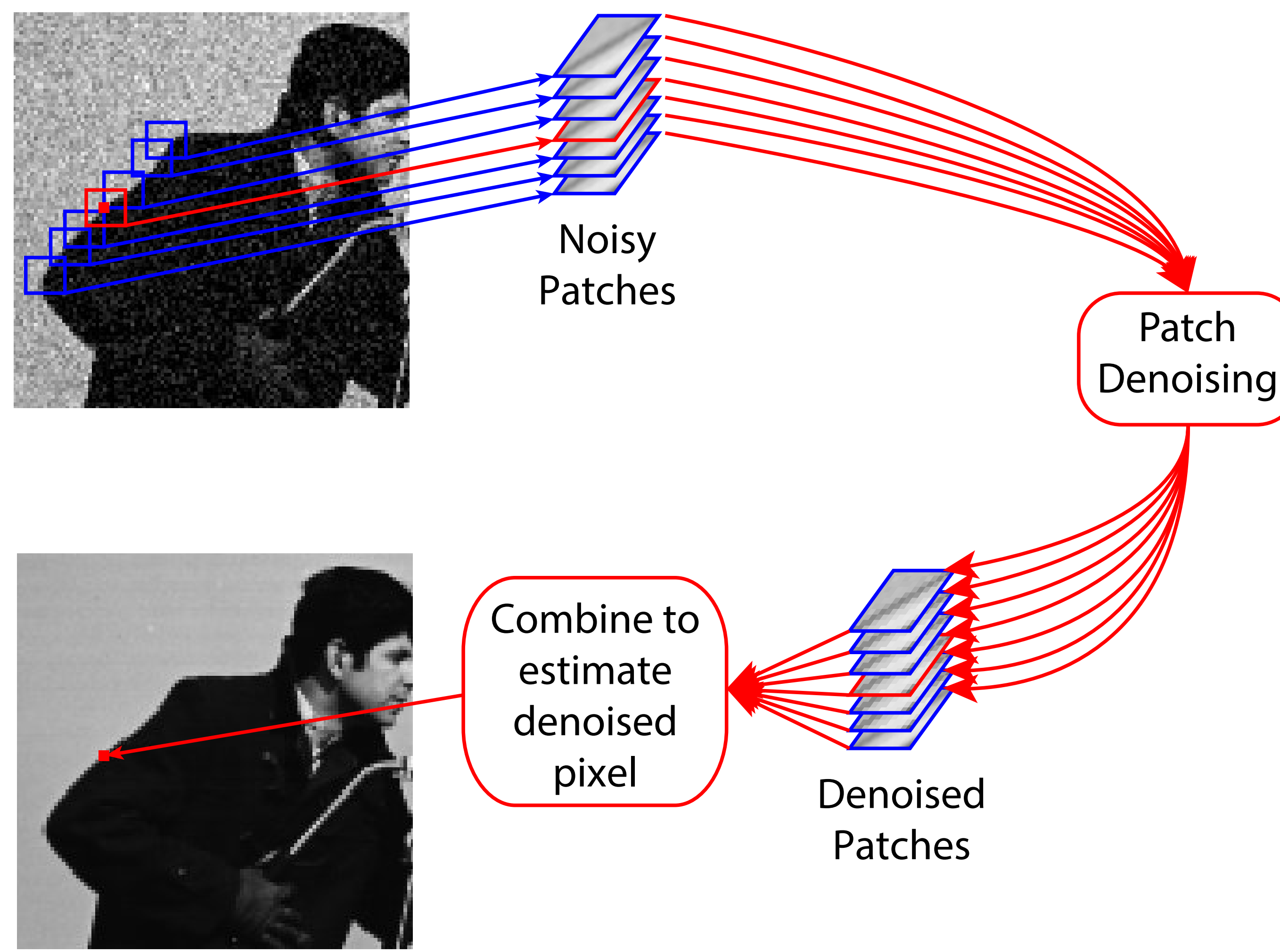
Geometric models of images

Total variation



(Wavelet) sparsity

Patch subspaces and manifolds



Regularization in inverse problems

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

Regularization in inverse problems

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

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

Geometric: $r(\beta)$ reflects image geometry (e.g. sparsity, patch redundancy, total variation)

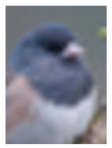
Learned: use training data to learn $r(\beta)$

Classes of methods

Model Agnostic (Ignore X)	**Decoupled** (First learn, then reconstruct)
Unrolled Optimization	**Neumann Networks** (this talk!)

Super-resolution with CNNs

Model Agnostic
(Ignore X)



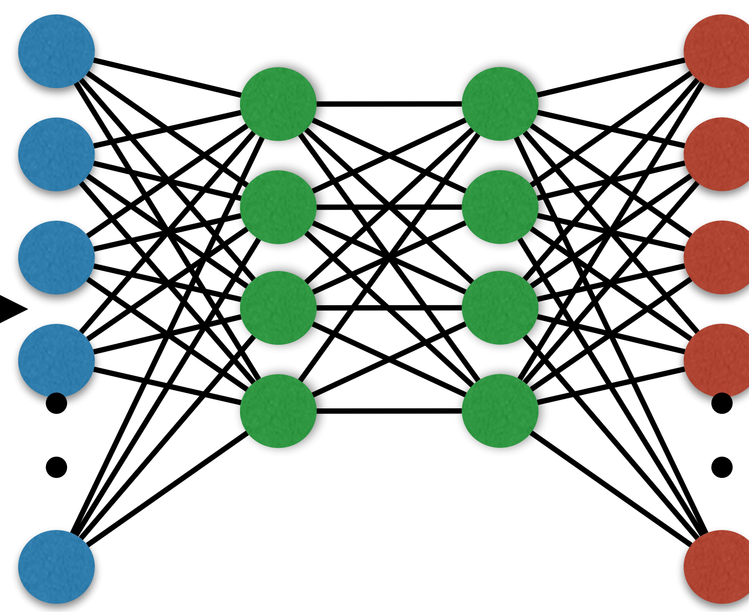
raw data
(low resolution image)

$\tilde{X}^{-1}$

bicubic interpolation

approximate high-resolution image

Train deep CNN to remove artifacts



blurry/blocky artifacts due to re-scaling

reconstruction

Pictures from: http://webdav.tuebingen.mpg.de/pixel/enhancenet/

Classes of methods

Model Agnostic
(Ignore X)

Decoupled
(First learn, then reconstruct)

Unrolled Optimization

Neumann Networks
(this talk!)

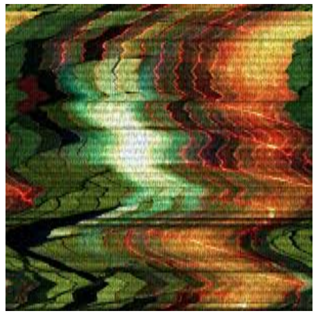
GANs for inverse problems

Decoupled
(First learn, then reconstruct)

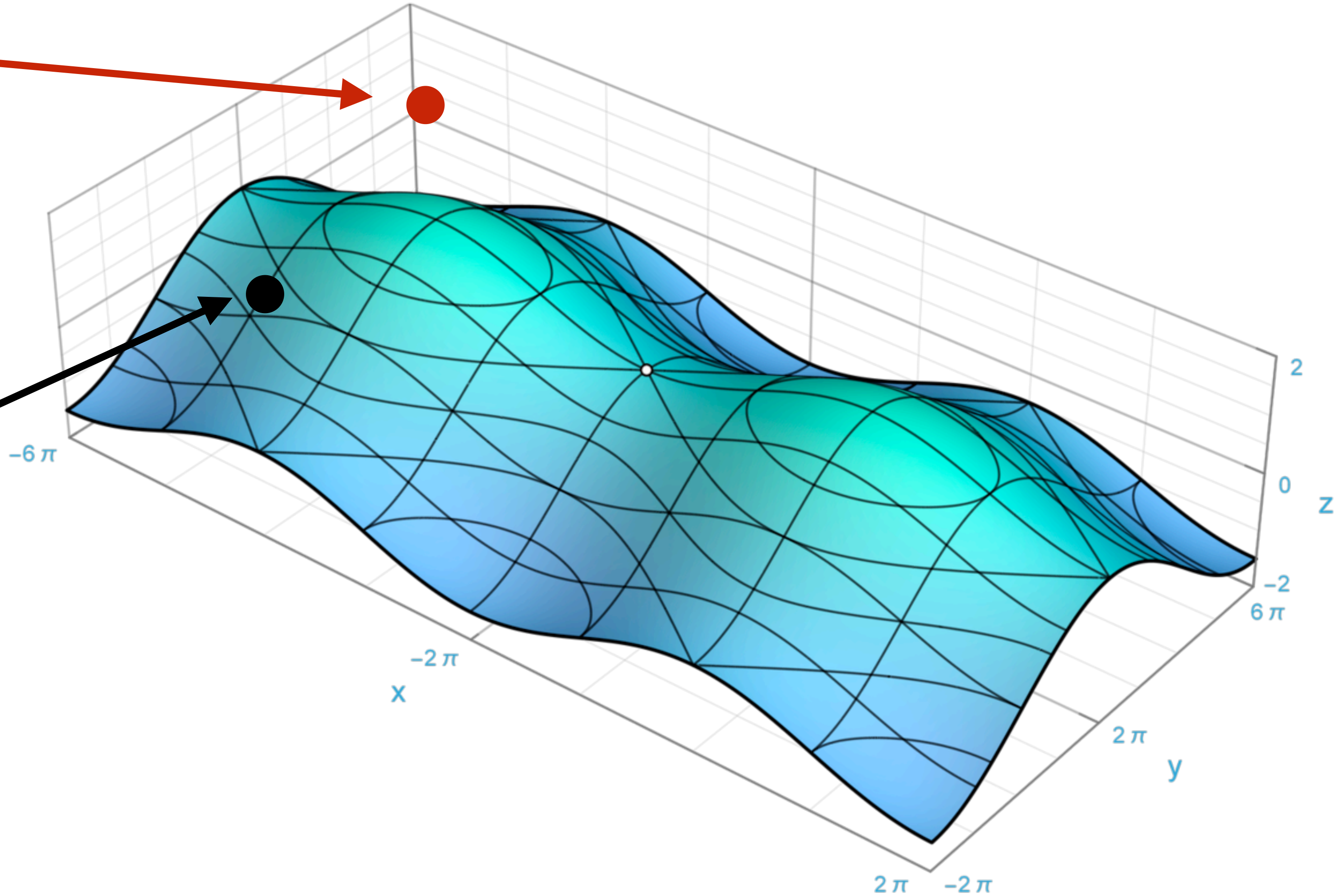
$$y \longrightarrow \boxed{\arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)} \longrightarrow \hat{\beta}$$

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

“Bad” image off manifold



“Good” image on manifold



GANs for inverse problems

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

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

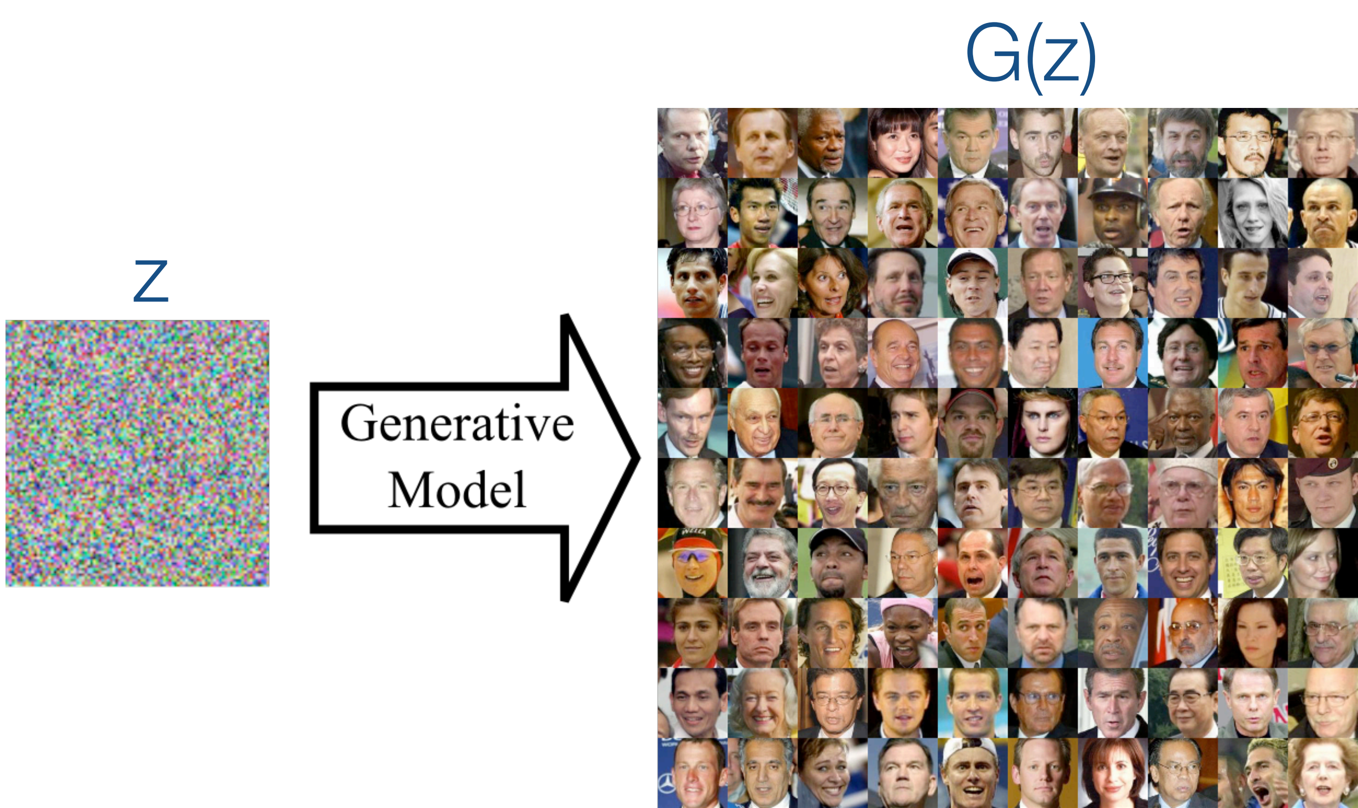
GANs for inverse problems

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

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

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

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



GANs for inverse problems

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

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

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

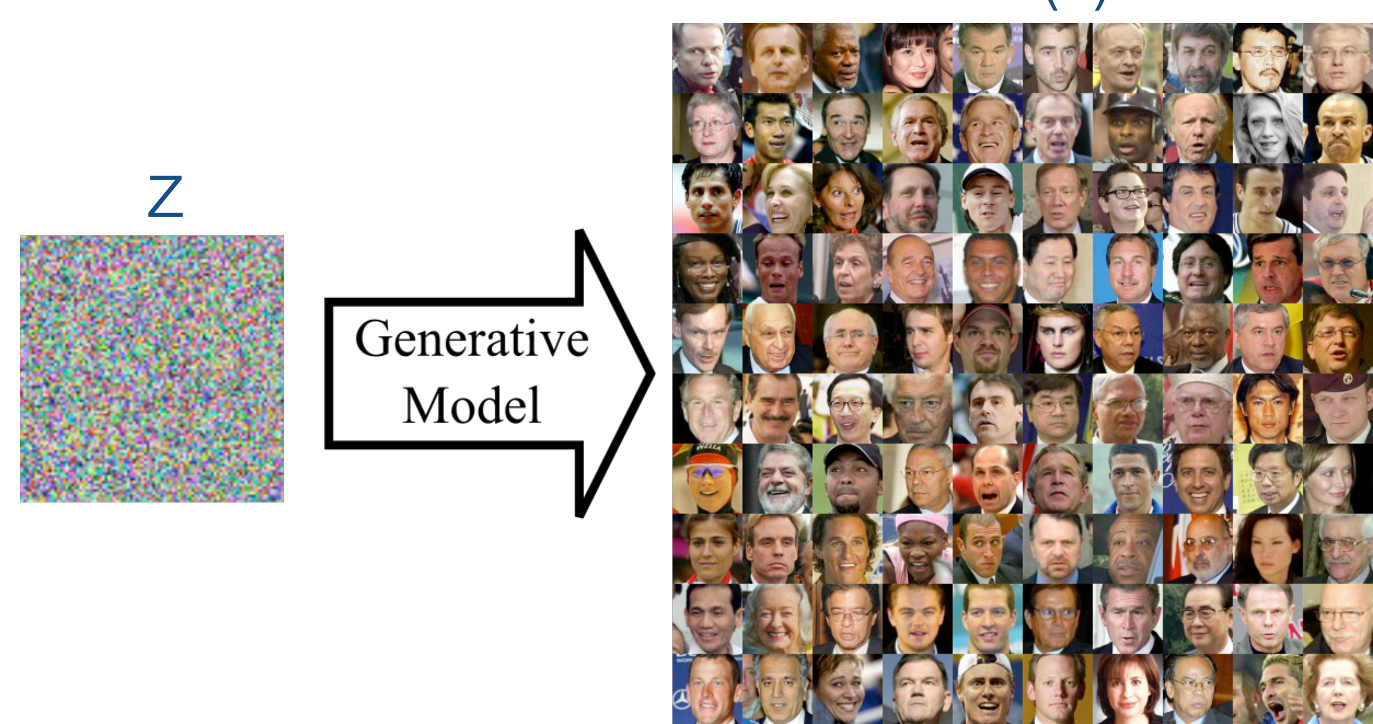
$$r(\beta) = \begin{cases} 0, & \beta \in \text{range}(G) \\ \infty, & \text{otherwise} \end{cases}$$

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

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

z

G(z)



Bora, Jalal, Price, Dimakis, 2017

How much training data?

Original
β

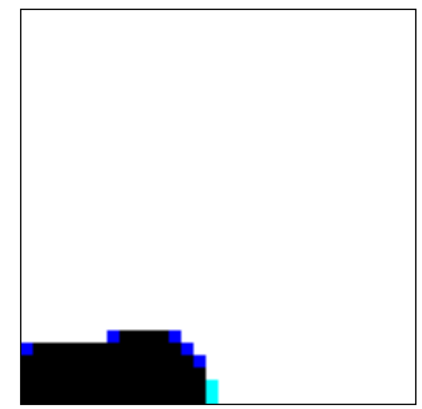
Observed
y

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

How much training data?

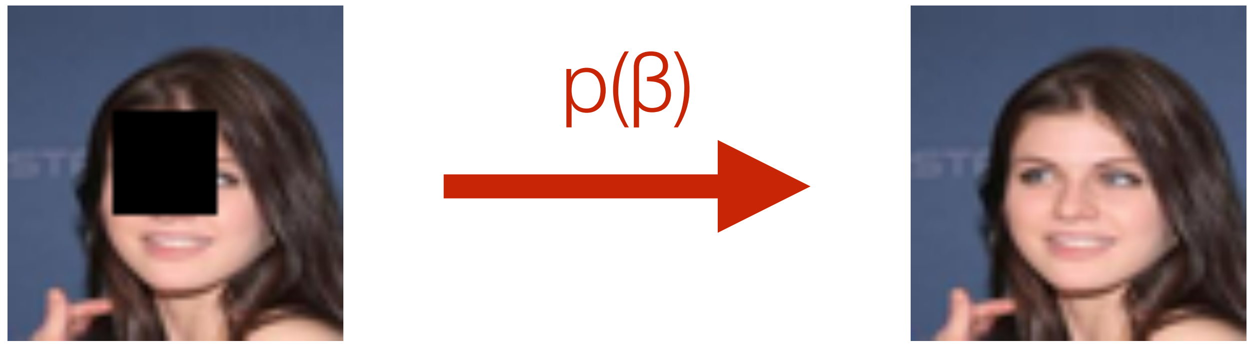
Original
β

Observed
y

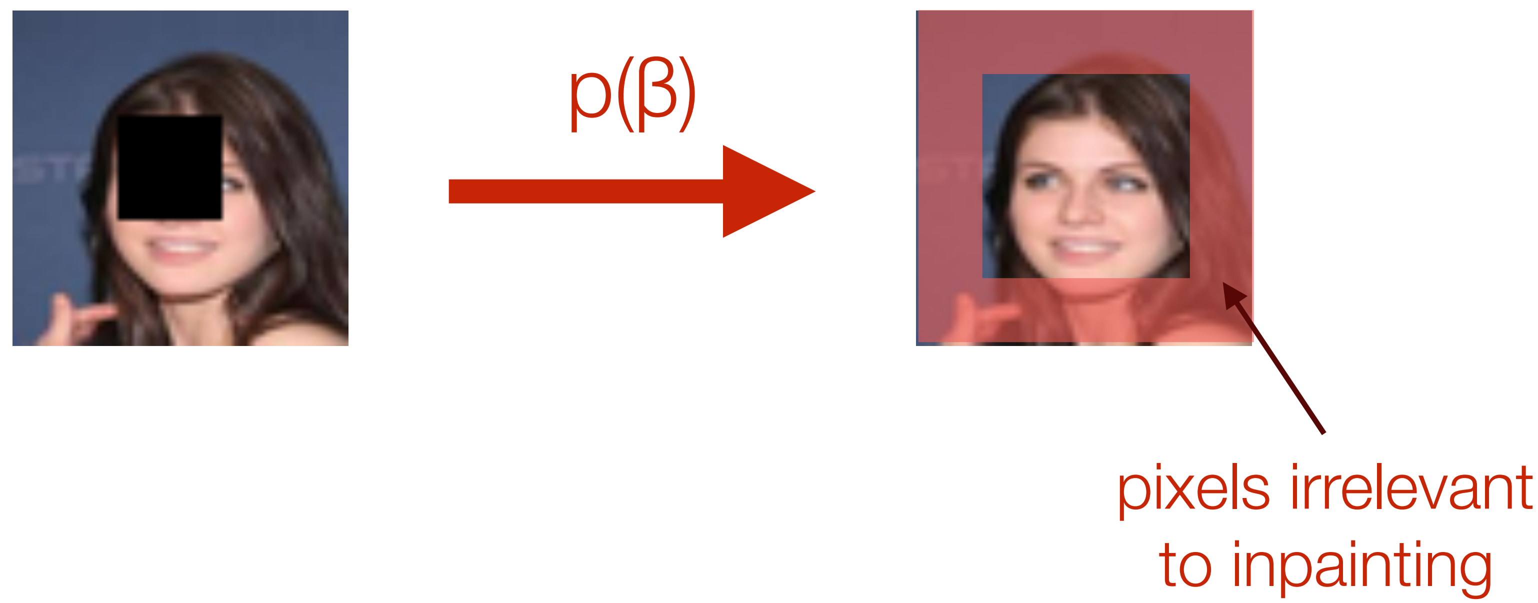


Reconstruction with convolutional neural network (CNN) trained with 2k samples

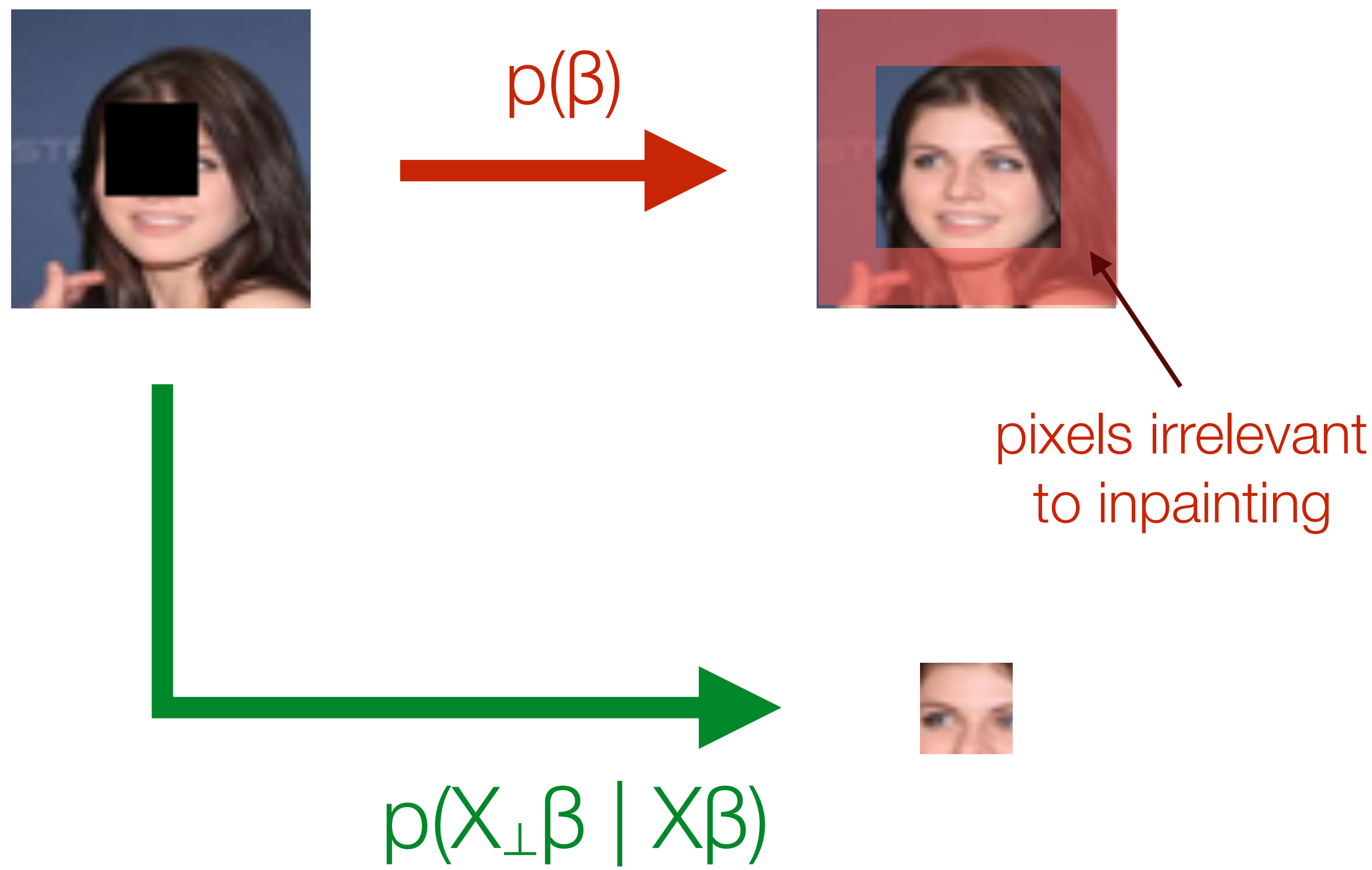
Prior vs. conditional density estimation



Prior vs. conditional density estimation



Prior vs. conditional density estimation



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

Implications for learning to regularize

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

⇓

X should be fully utilized in learning process

Classes of methods

Model Agnostic
(Ignore X)

Decoupled
(First learn, then reconstruct)

Unrolled Optimization

Neumann Networks
(this talk!)

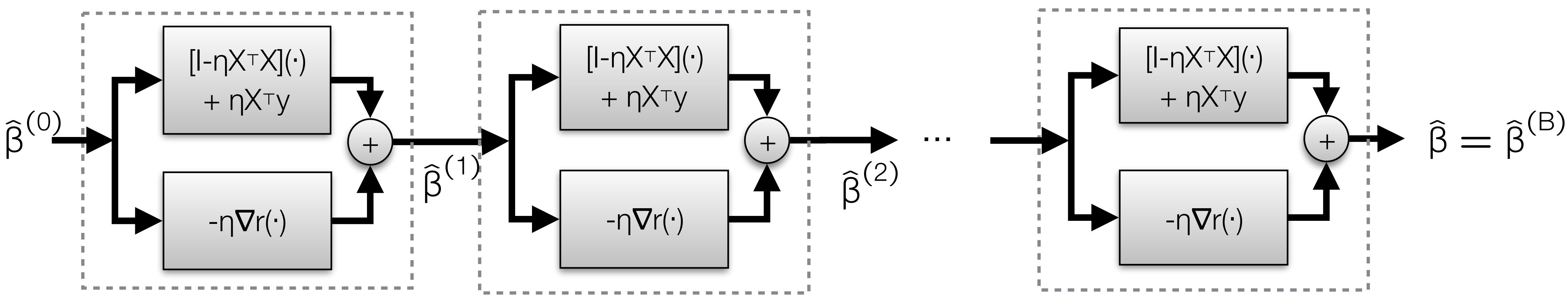
Assume $r(\beta)$ differentiable.

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

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

for $k = 1, 2, \ldots$

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



Unrolled Optimization

Assume $r(\beta)$ differentiable.

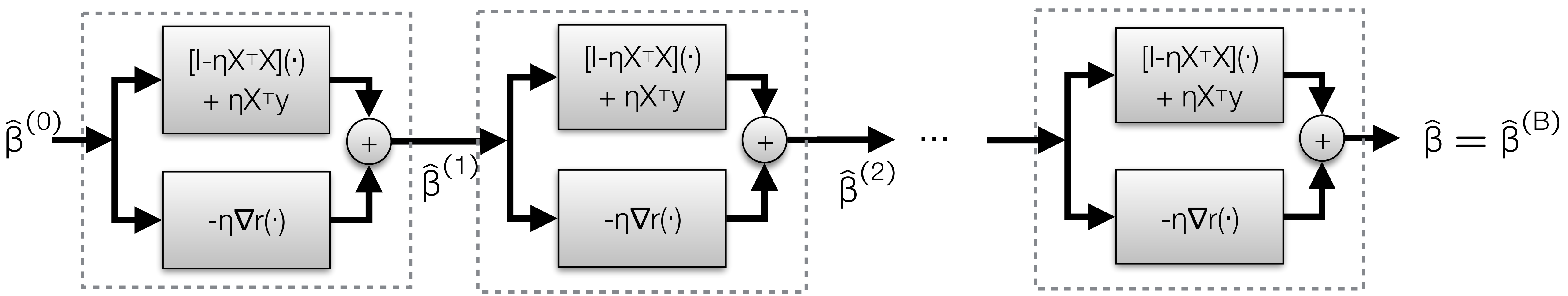
$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

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 Optimization

Assume $r(\beta)$ differentiable.

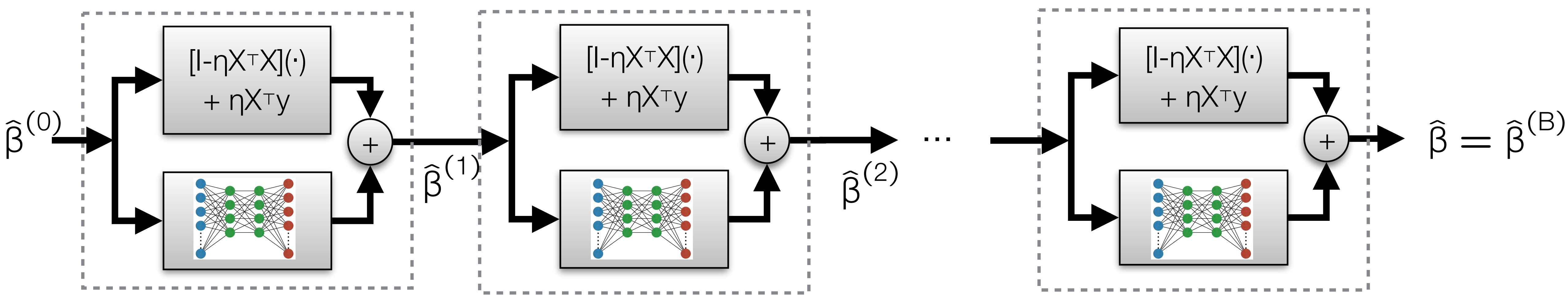
$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

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 Optimization

Assume $r(\beta)$ differentiable.

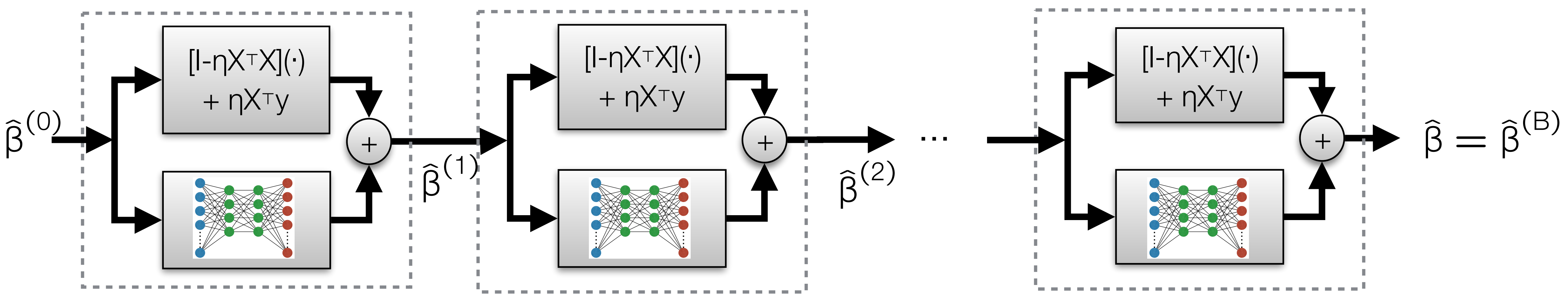
$$\hat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

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

Neumann series

Assume $r(\beta)$ differentiable.

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

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

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

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

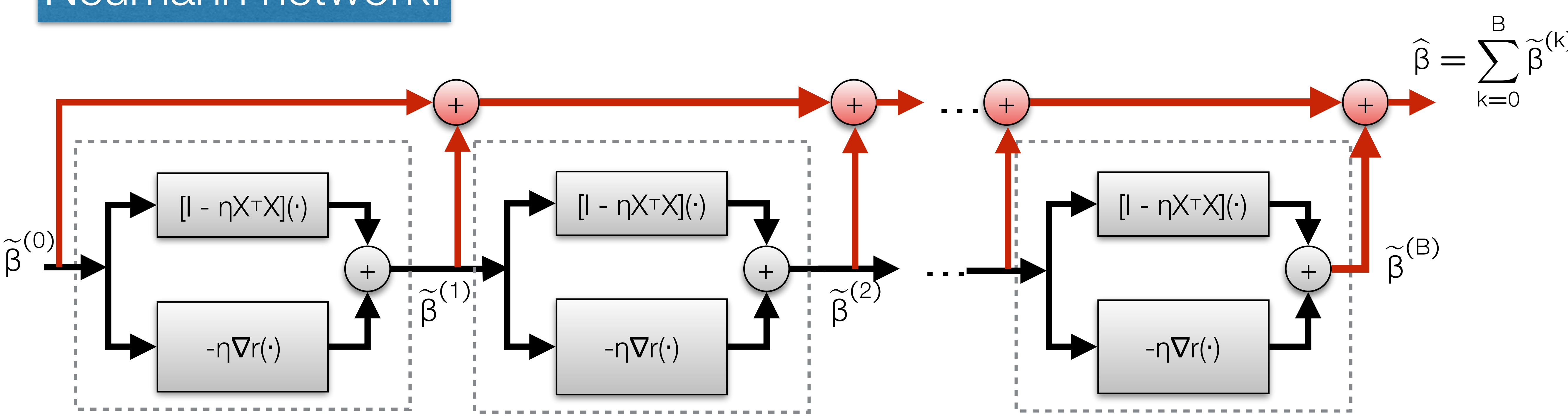
Can we estimate β by approximating (1) using (2)?
(e.g. $A = I - X^\top X + \nabla r$ if ∇r is linear)

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:



Neumann networks

Assume $r(\beta)$ differentiable.

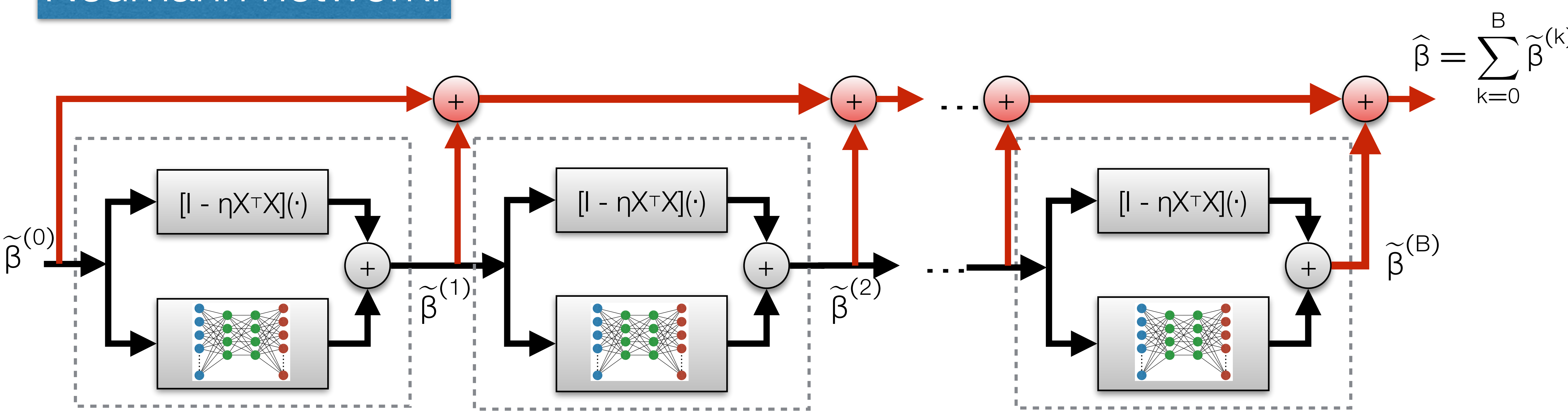
$$\widehat{\beta} = \arg\min_{\beta} \|y - X\beta\|_2^2 + r(\beta)$$

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

$$\approx \sum_{k=1}^{B} (I - \eta X^\top X - \boxed{\eta \nabla r})^k \eta X^\top y$$

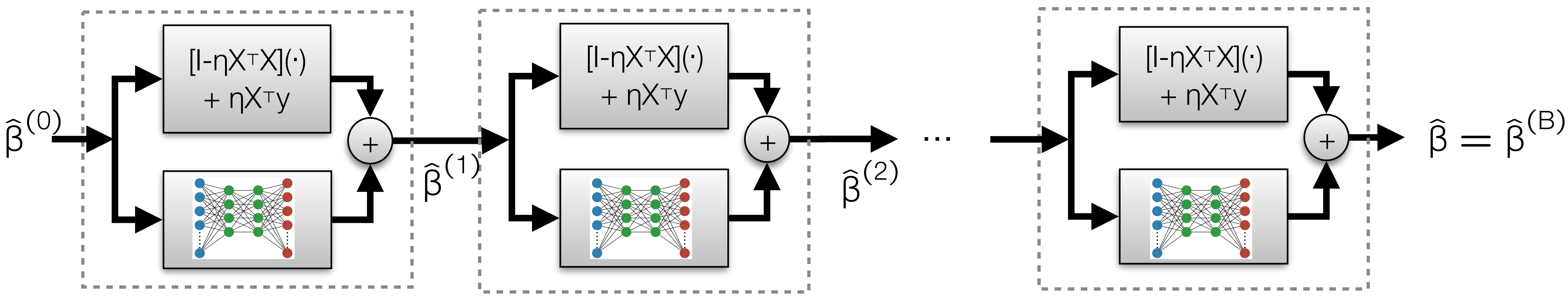
Replace with learned neural network

Neumann network:

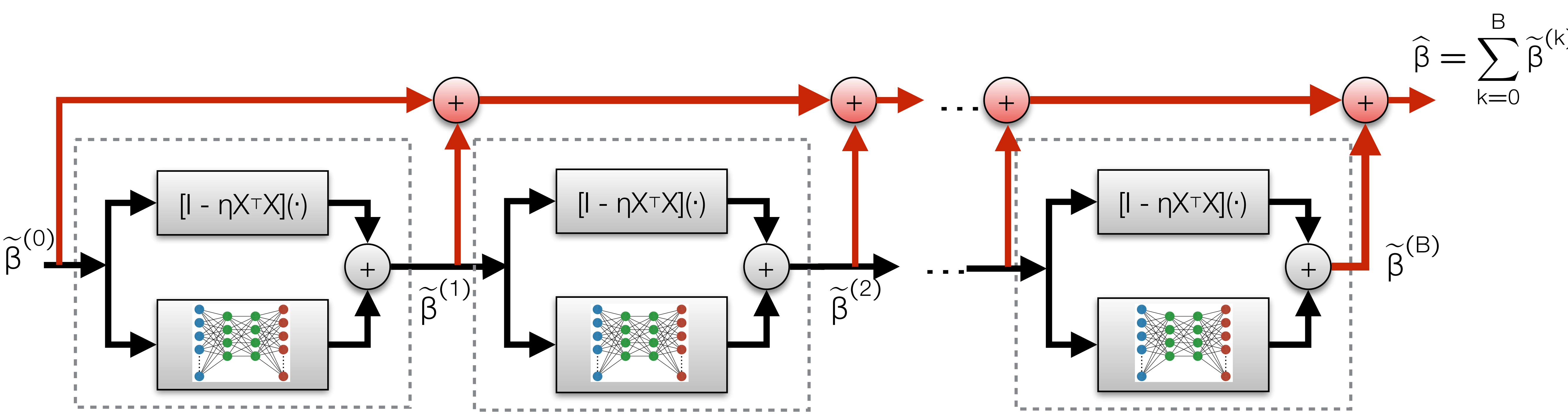


Comparison

Gradient descent network

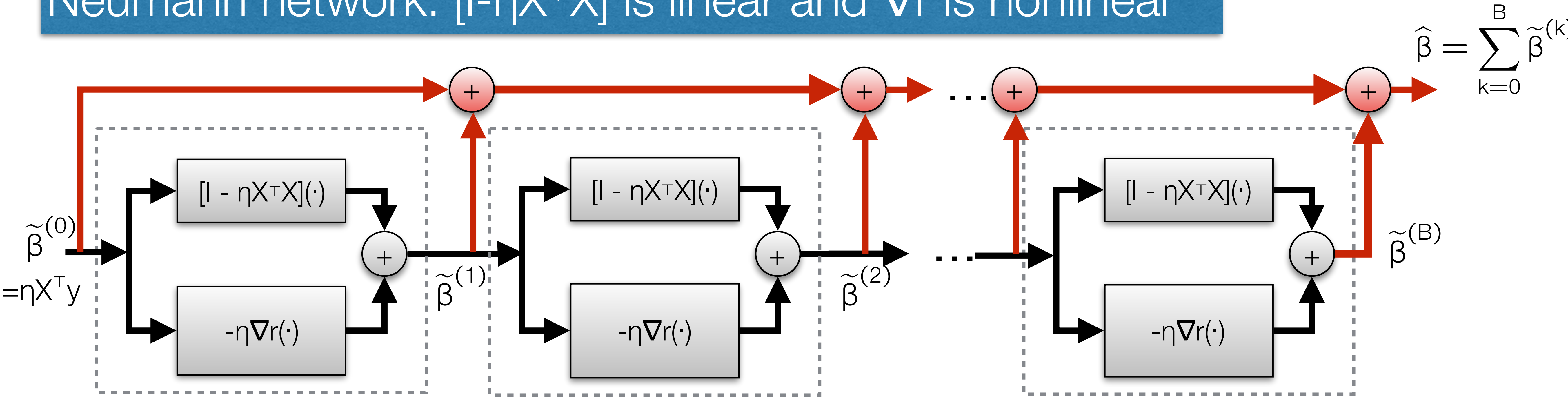


Neumann network

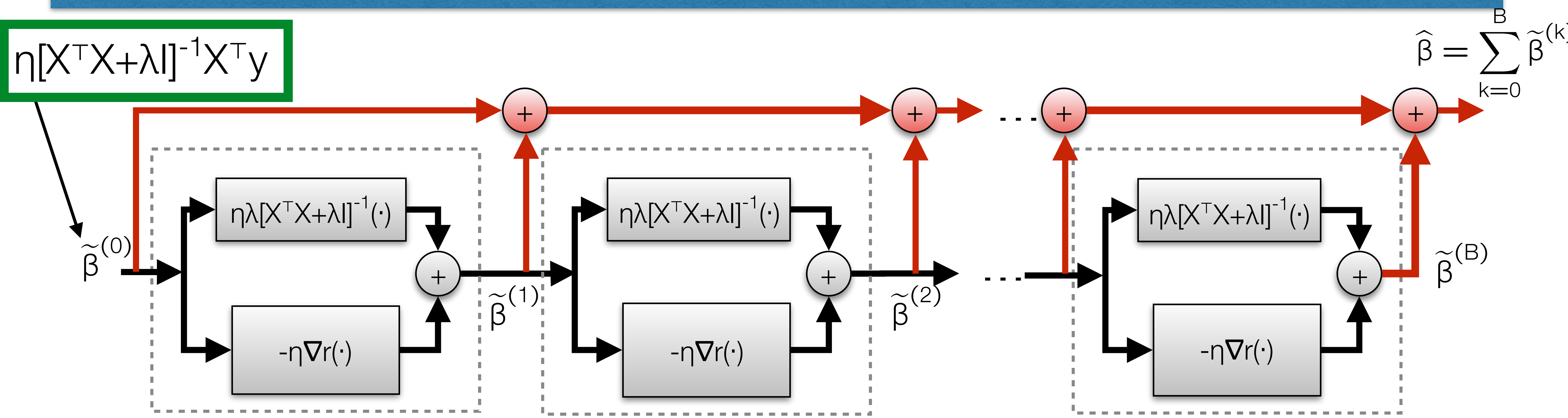


Preconditioning

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



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



Classes of methods

Model Agnostic
(Ignore X)

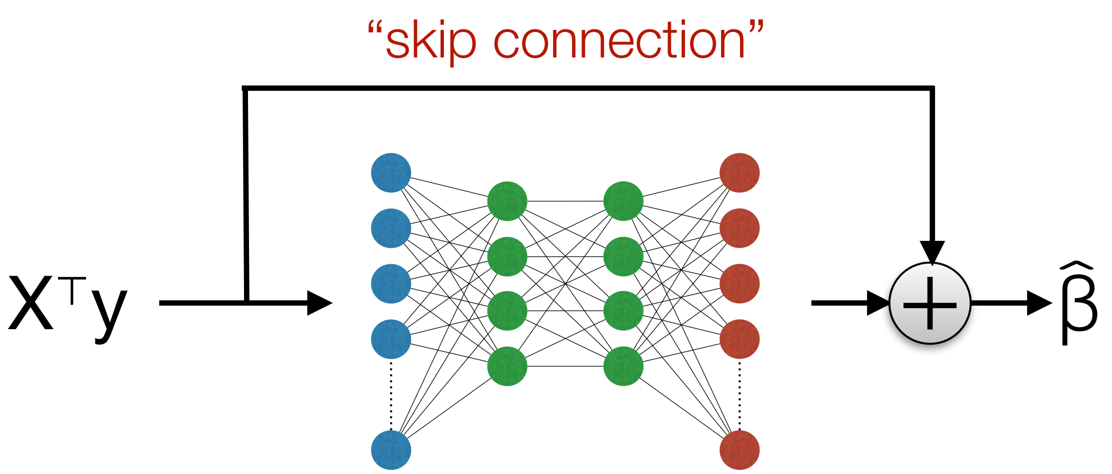
Decoupled
(First learn, then reconstruct)

Unrolled Optimization

Neumann Networks
(this talk!)

Comparison Methods

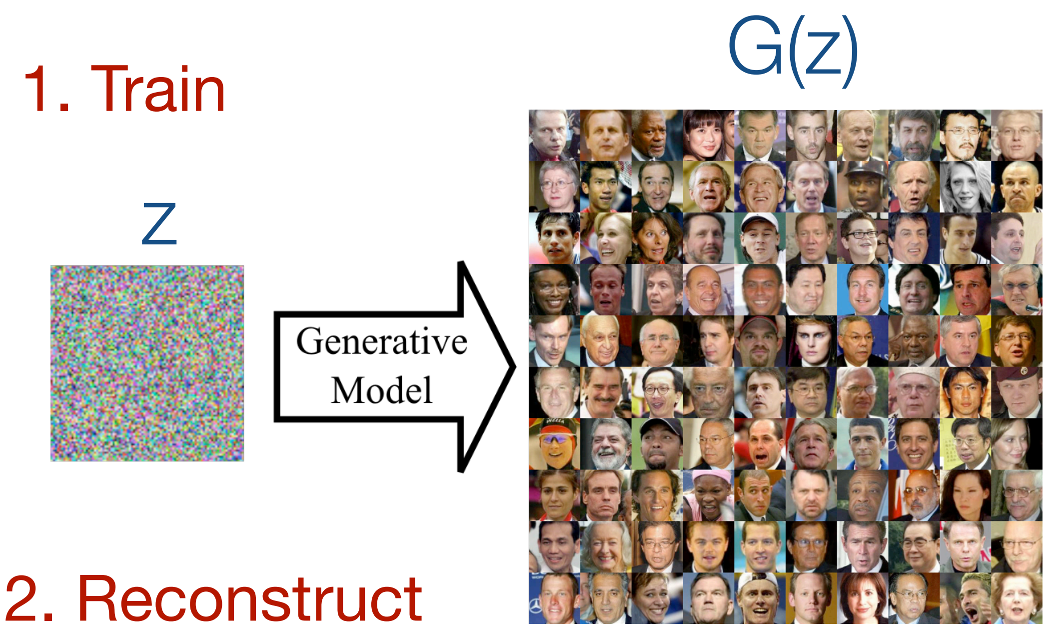
Residual Autoencoder



Mao, Shen, Yang, 2016

Design-agnostic GAN

1. Train

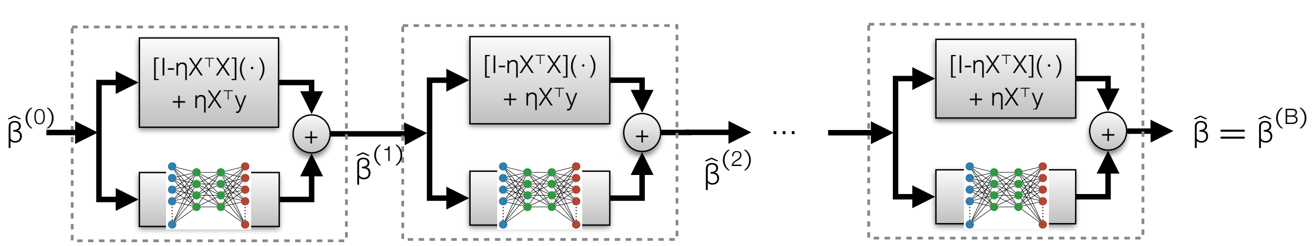


2. Reconstruct

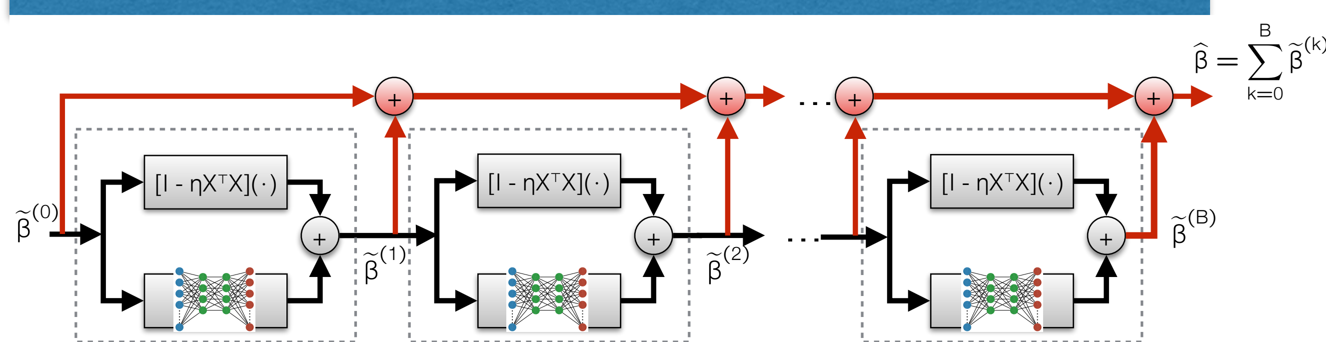
$$\hat{\beta} = \underset{\beta \in \mathrm{range}(G)}{\arg\min} \ \|y - X\beta\|_2^2$$

Bora, Jalal, Price, Dimakis, 2017

Unrolled Gradient Descent



Neumann Network

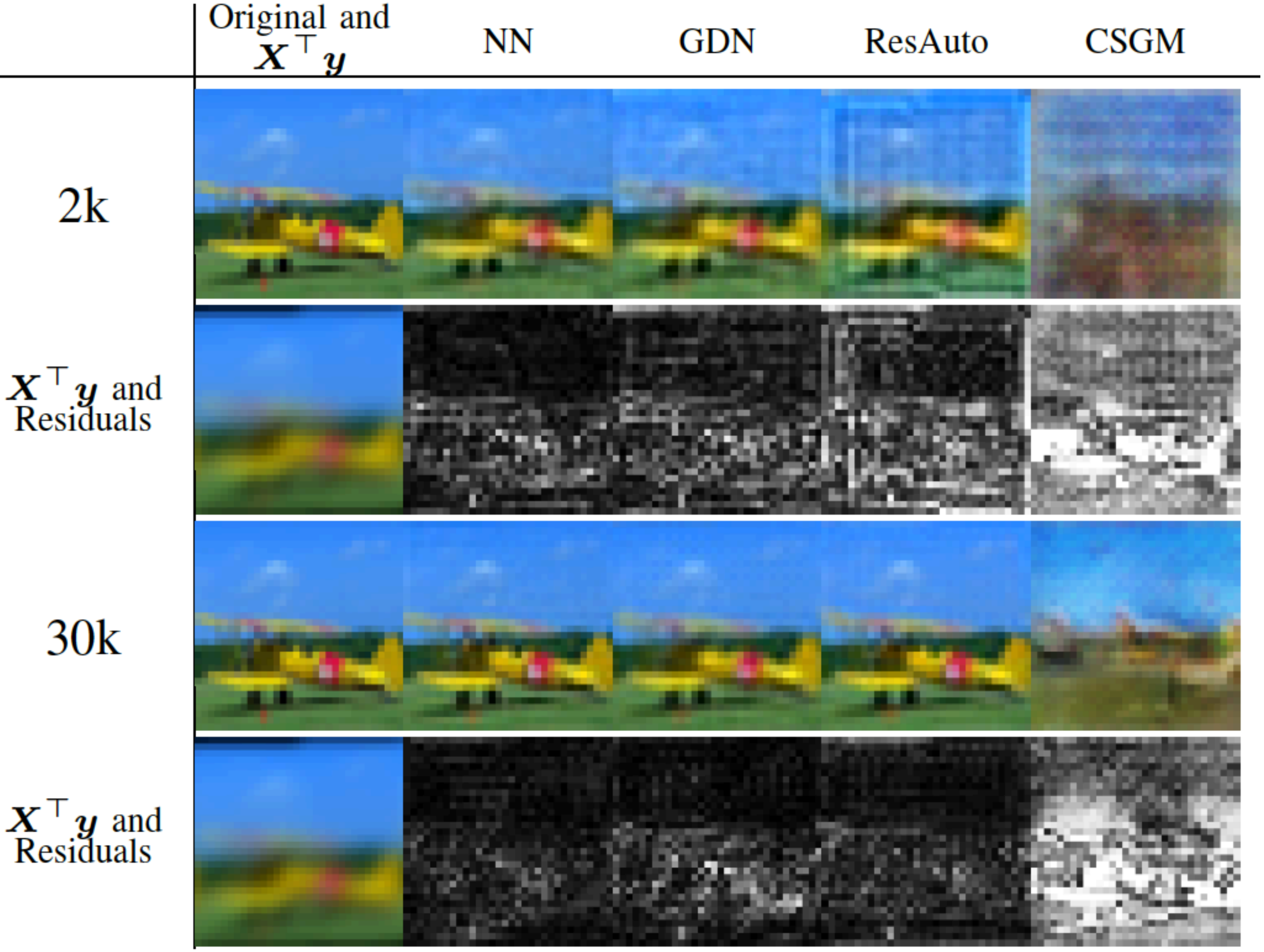


Summary of Results

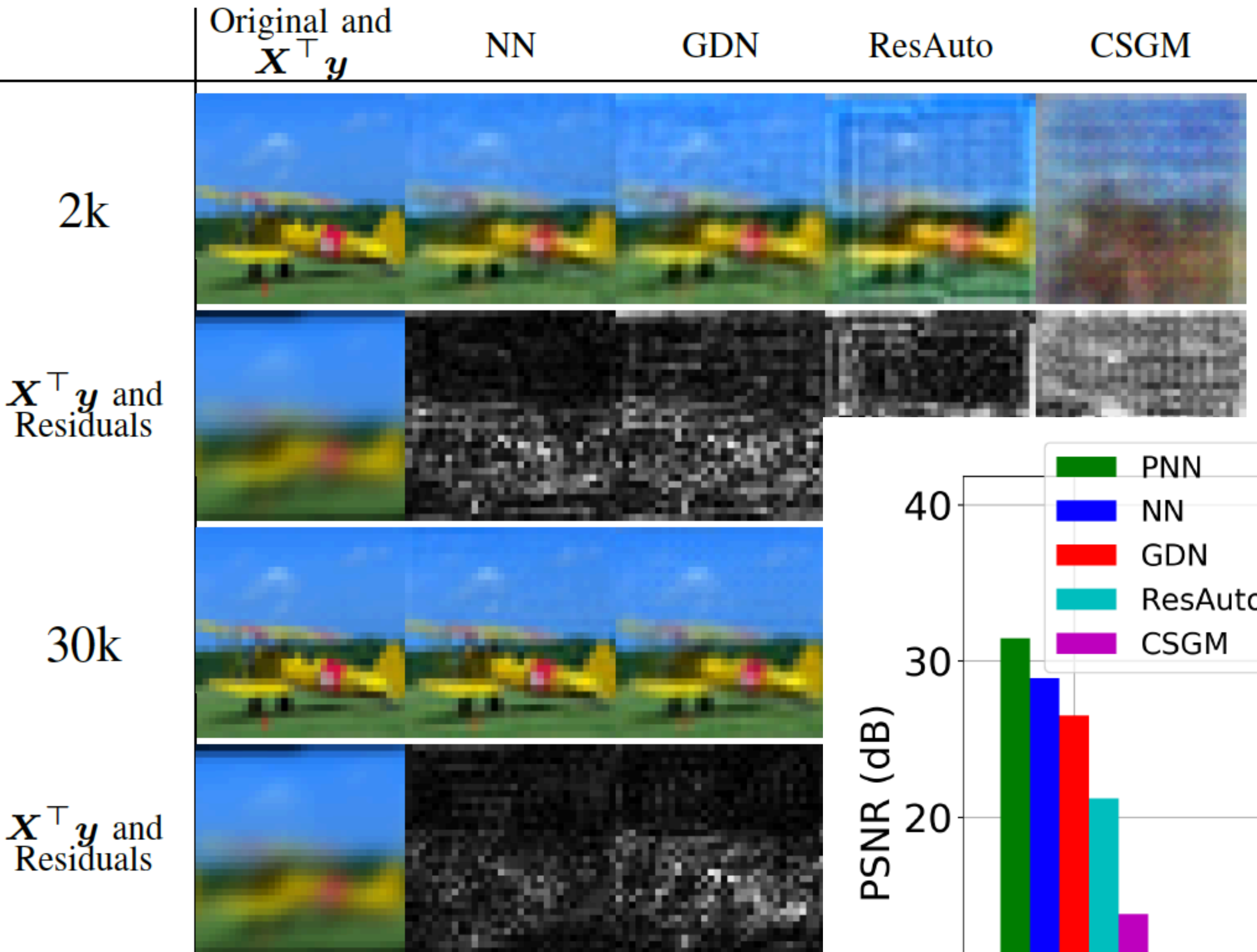
		Inpaint	Deblur	Deblur+ϵ	CS2	CS8	SR4	SR10
CIFAR10	NN	28.20	36.55	29.43	33.83	**25.15**	24.48	**23.09**
	PNN	28.40	**37.83**	**30.47**	33.75	23.43	**26.06**	21.79
	GDN	27.76	31.25	29.02	**34.99**	25.00	24.49	20.47
	MoDL	28.18	34.89	29.72	33.47	23.72	24.54	21.90
	TNRD	27.87	34.84	29.70	32.74	25.11	23.84	21.99
	ResAuto	**29.05**	31.04	25.24	18.51	9.29	24.84	21.92
	CSGM	17.88	15.20	14.61	17.99	19.33	16.87	16.66
	TV	25.90	27.57	26.64	25.41	20.68	24.71	20.68
CelebA	NN	**31.06**	31.01	30.43	**35.12**	**28.38**	27.31	23.57
	PNN	30.45	**33.79**	**30.89**	32.61	26.41	**28.70**	23.74
	GDN	30.99	30.19	29.27	34.93	28.33	27.14	23.46
	MoDL	30.75	30.80	29.59	30.22	25.84	26.42	24.12
	TNRD	30.21	29.92	29.79	33.89	28.19	25.75	22.73
	ResAuto	29.66	25.65	25.29	19.41	9.16	25.62	**24.92**
	CSGM	17.75	15.68	15.30	17.99	18.21	18.11	17.88
	TV	24.07	30.96	26.24	25.91	23.01	26.83	20.70
STL10	NN	27.47	29.43	26.12	**31.98**	**26.65**	24.88	21.80
	PNN	28.00	**30.66**	**27.21**	31.40	23.43	**25.95**	**22.19**
	GDN	**28.07**	30.19	25.61	31.11	26.19	24.88	21.46
	MoDL	28.03	29.42	26.06	27.29	23.16	24.67	16.88
	TNRD	27.88	29.33	26.32	31.05	25.38	24.55	21.21
	ResAuto	27.28	25.42	25.13	19.48	9.30	24.12	21.13
	CSGM	16.50	14.04	15.59	16.67	16.39	16.58	16.47
	TV	26.29	29.96	26.85	24.82	22.04	26.37	20.12

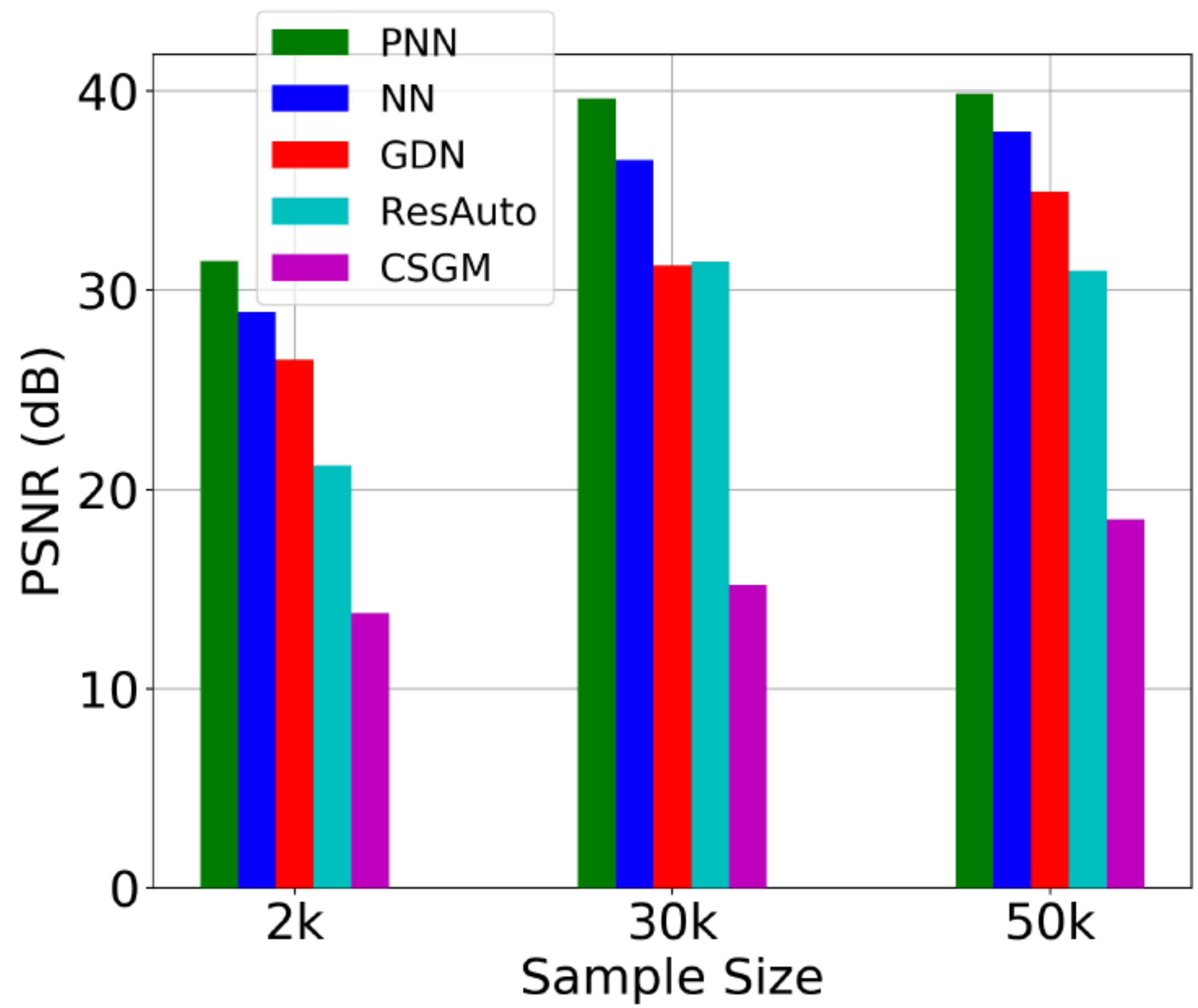
Table 1: PSNR comparison for the CIFAR, CelebA, and STL10 datasets respectively. Values reported are the median across a test set of size 256.

Sample Complexity

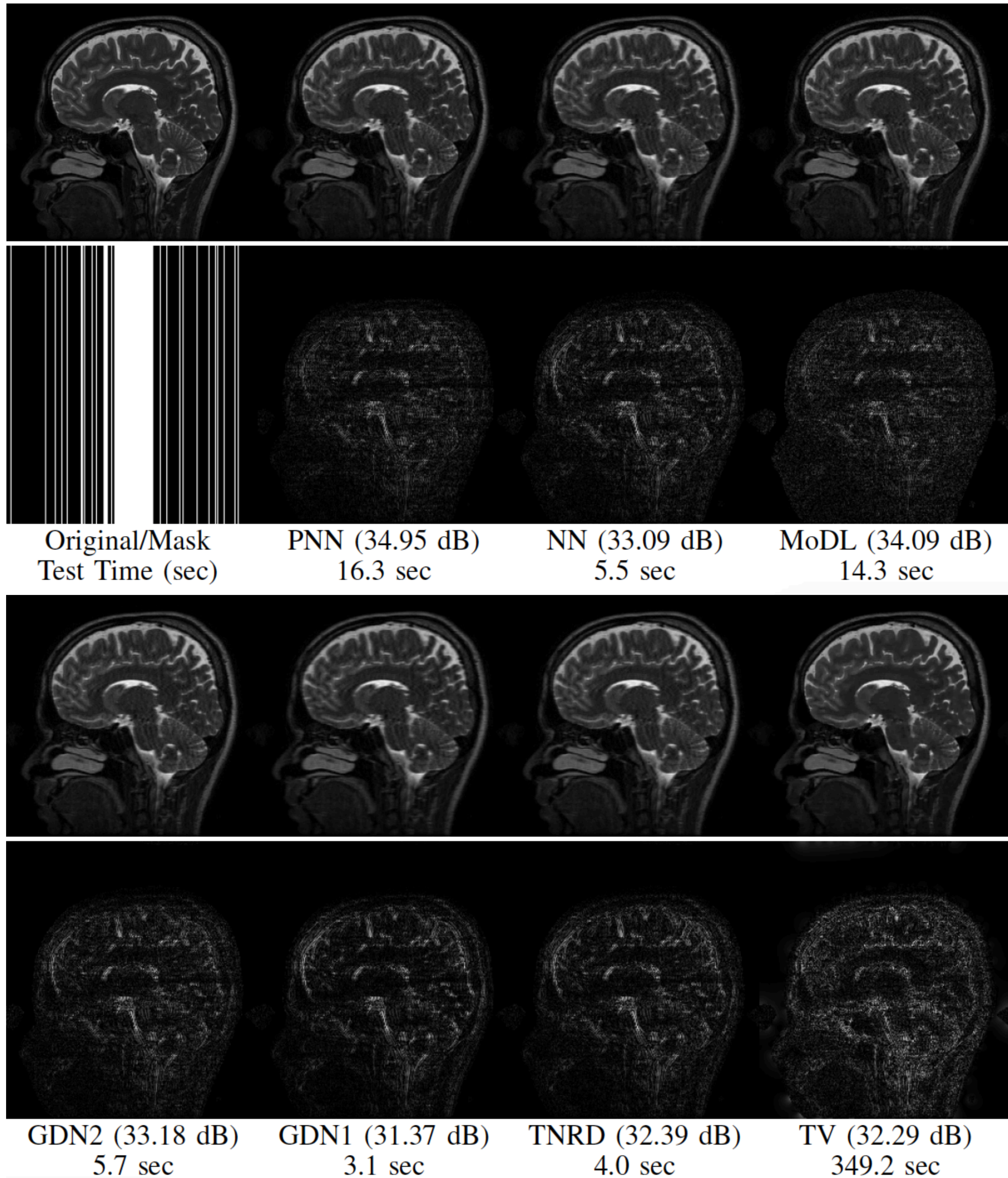


Sample Complexity





Application: MRI reconstruction



Neumann series for nonlinear operators?

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

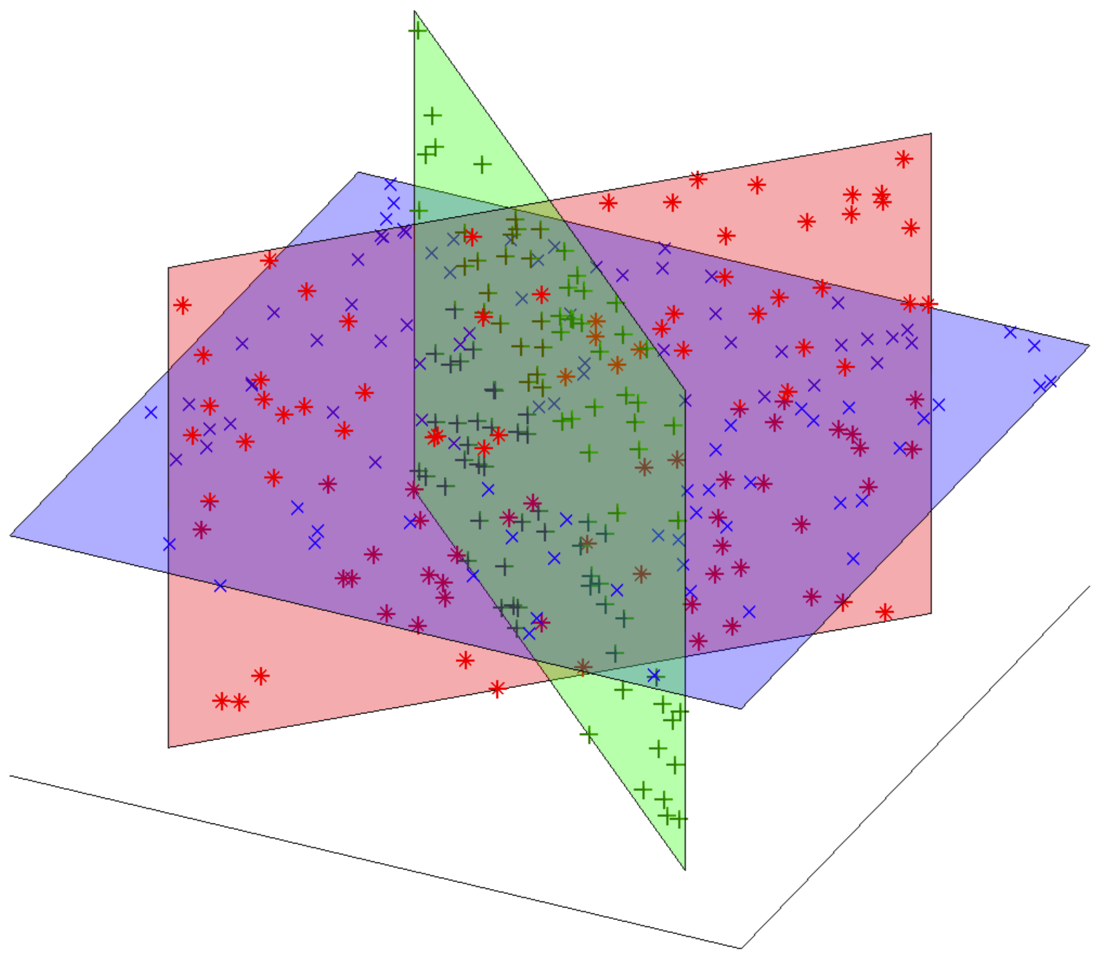
$$(\mathrm{I} - \mathrm{A})^{-1} \neq \sum_{k=0}^{\infty} \mathrm{A}^k$$

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

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

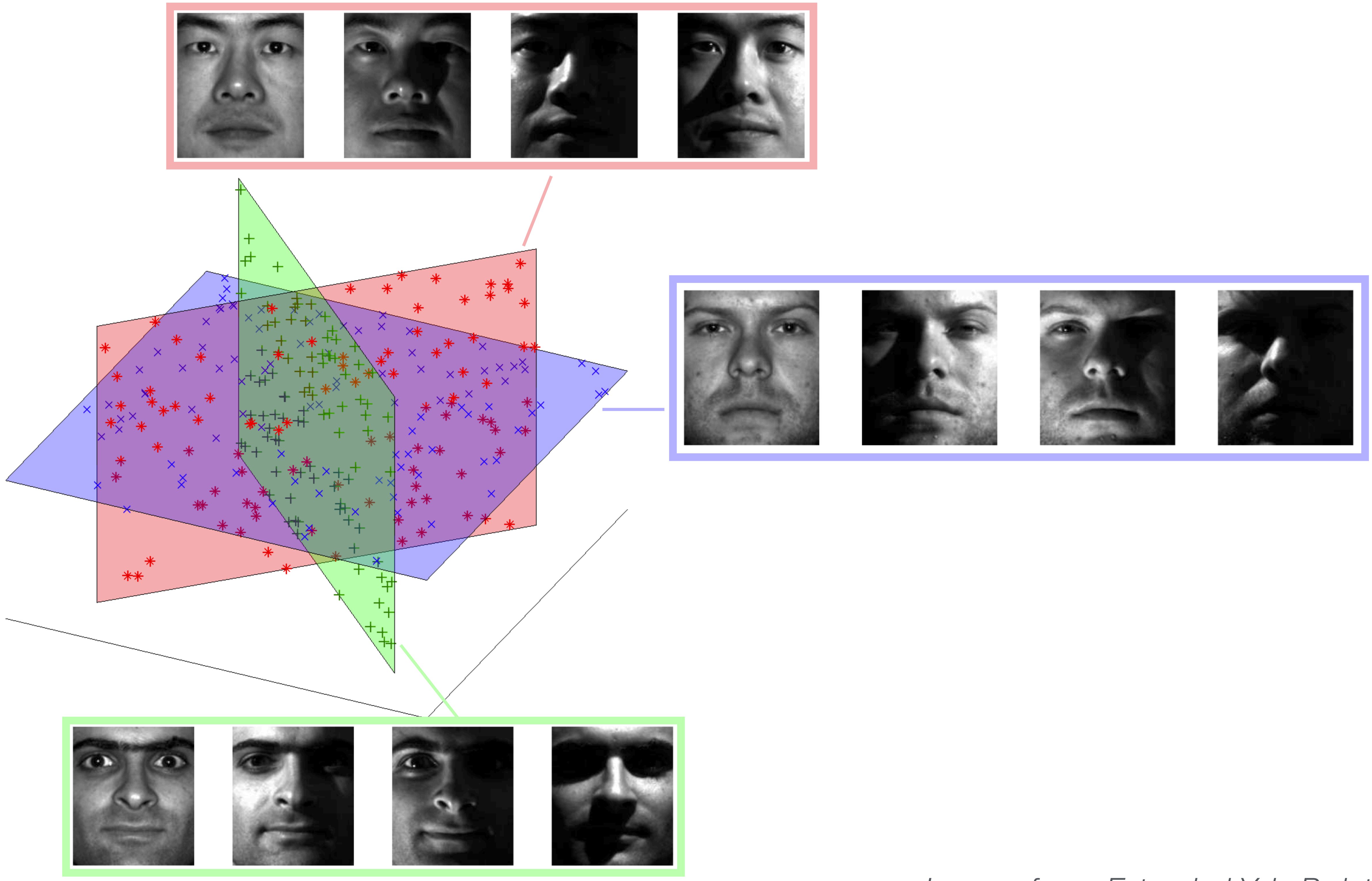
Case Study: Union of Subspaces Models

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



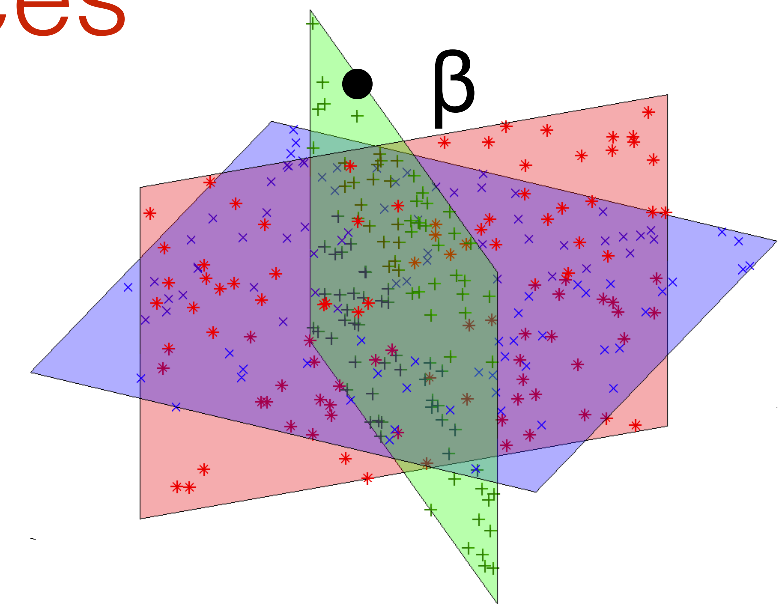
Case Study: Union of Subspaces Models

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



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

Neumann 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 ∇r is piecewise linear in β

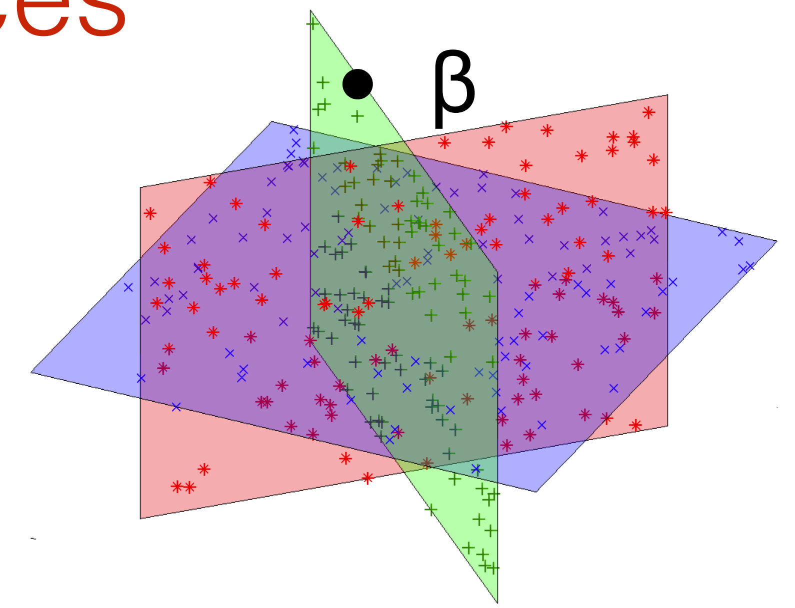
$$\nabla r^*(\beta) = \begin{cases} R_1\beta & \text{if } \beta \in S_1 \\ \vdots & \vdots \\ R_K\beta & \text{if } \beta \in S_K \end{cases}$$

S_k = set of points closer to subspace k than any other subspace

- Neumann network with ReLU activations can closely approximate this
- Outputs of all Neumann net blocks are in the same S_k for some k
  ⇒ for a fixed input, ∇r behaves linearly

  ⇒ Neumann series foundation is justifiable and accurate

Neumann nets and union of subspaces



For simplicity, assume:

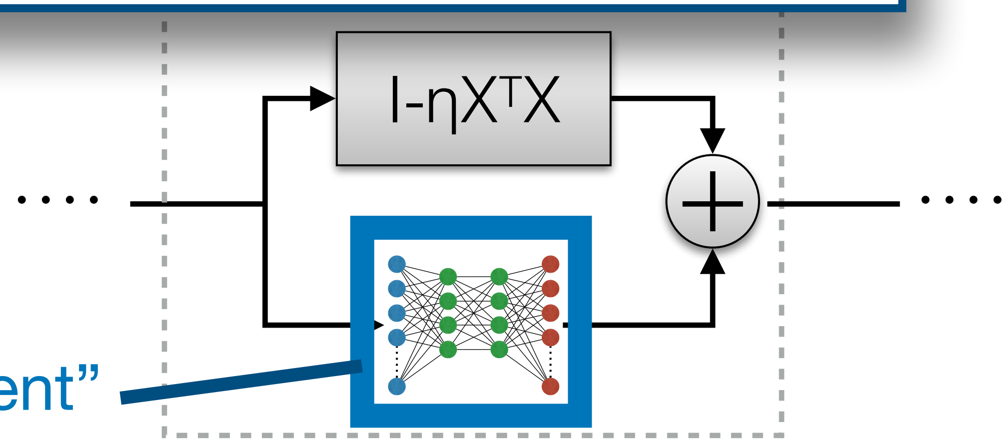
- X has orthonormal rows
- measurements are noise-free: $y = X\beta \in \mathbb{R}^m$
- maximum subspace dimension < m/2
- the union of subspaces is "generic"

Theorem (informal):

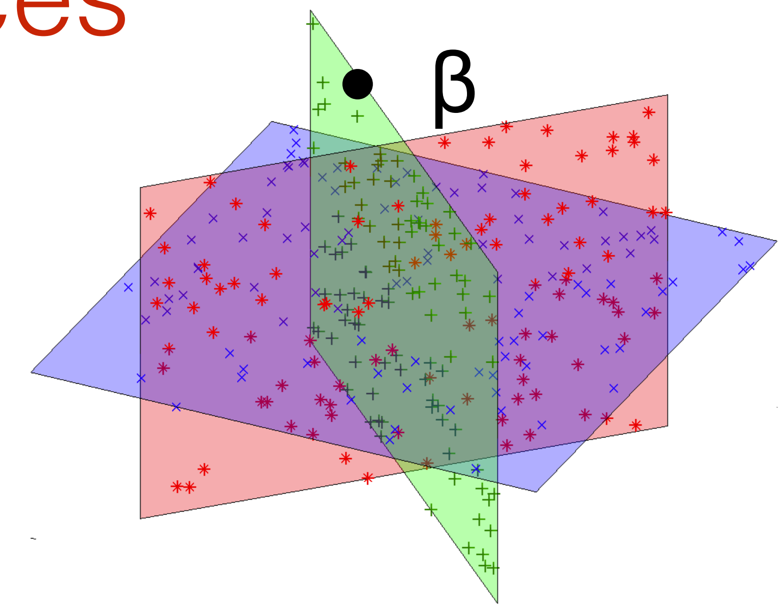
For a given step size $0 < \eta < 1$ and number of blocks B there exists a Neumann network estimator $\hat{\beta}(X\beta)$ with a piecewise linear learned component such that

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

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"

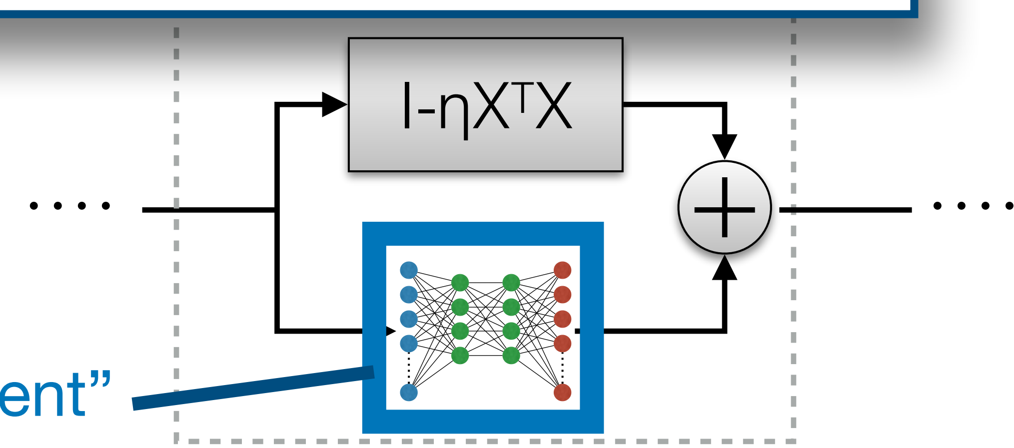
Theorem (informal):

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

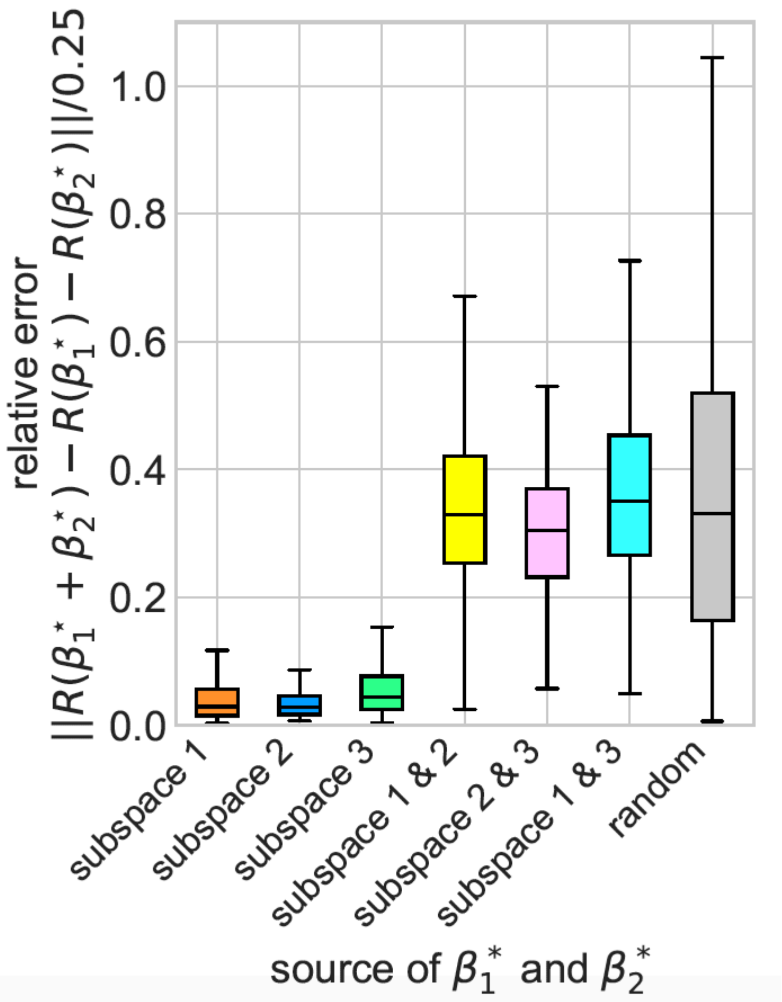


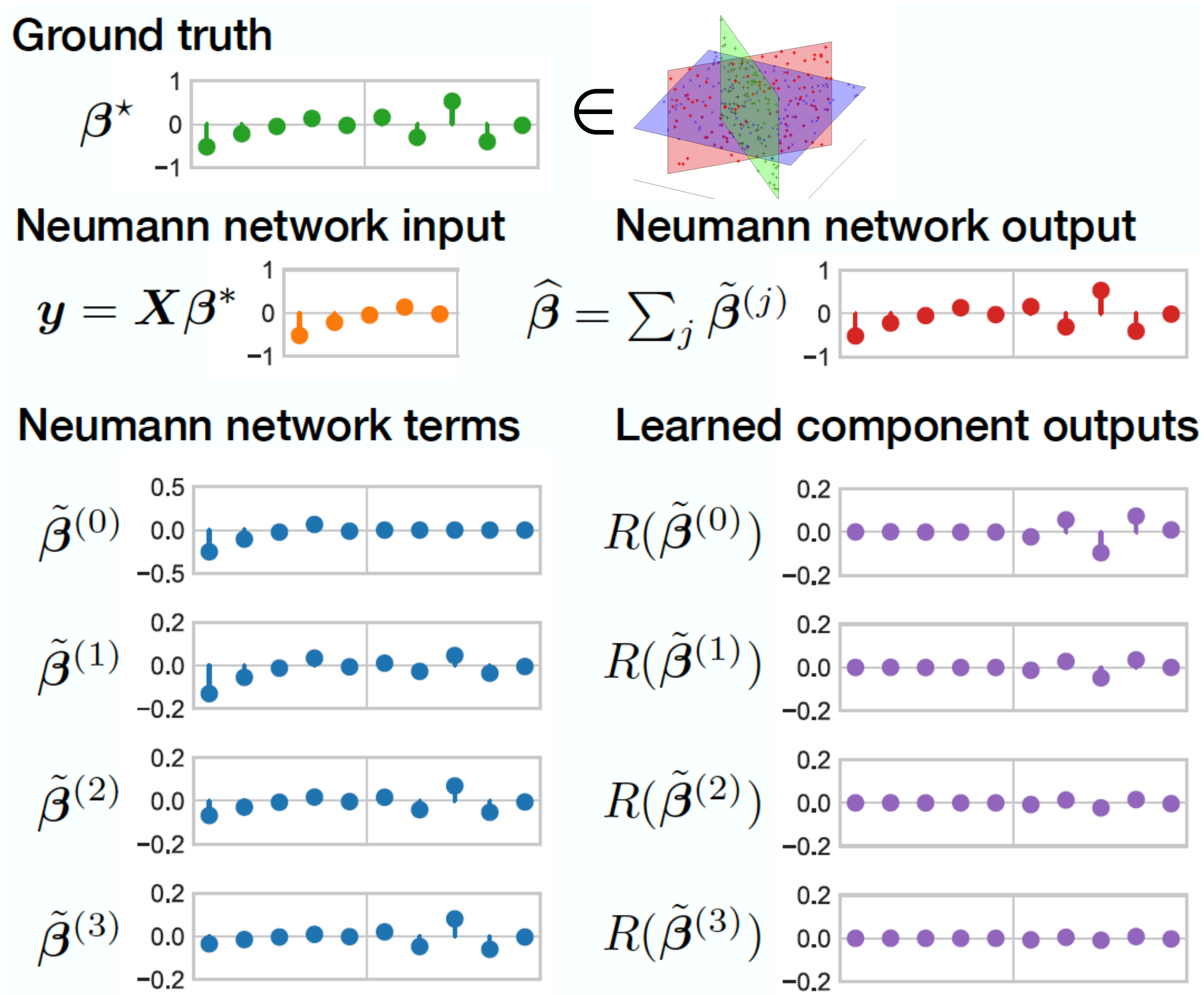
"learned component"

Empirical support for theory

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

Test of Piecewise Linearity of ∇r





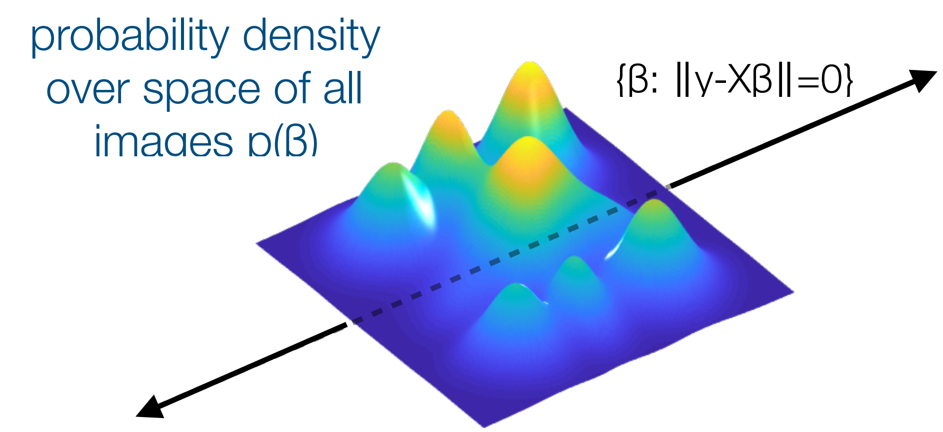
R = ∇r reflects union of subspaces structure

Outputs of all blocks in same subspace

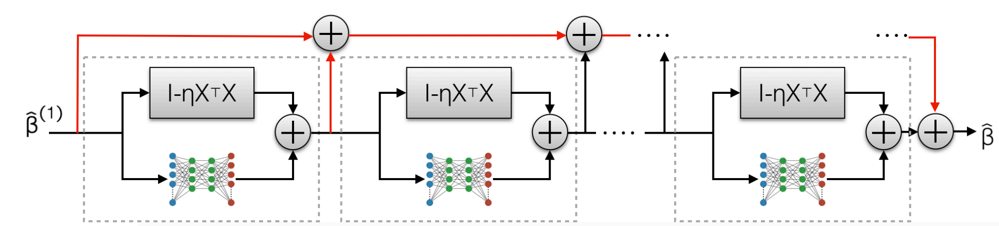
R = ∇r only affects β in X's null space

Conclusions

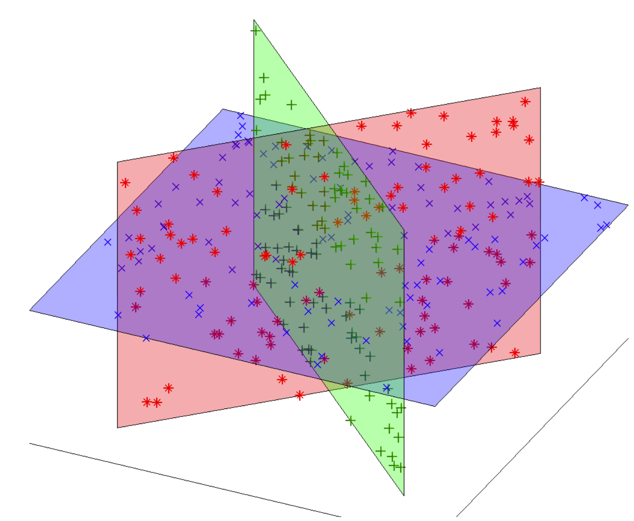
- Explicitly accounting for design (X) during training can dramatically reduce sample complexity.



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



- Neumann networks are mathematically justified for union of subspaces.



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

Learning from Highly Correlated Features using Graph Total Variation

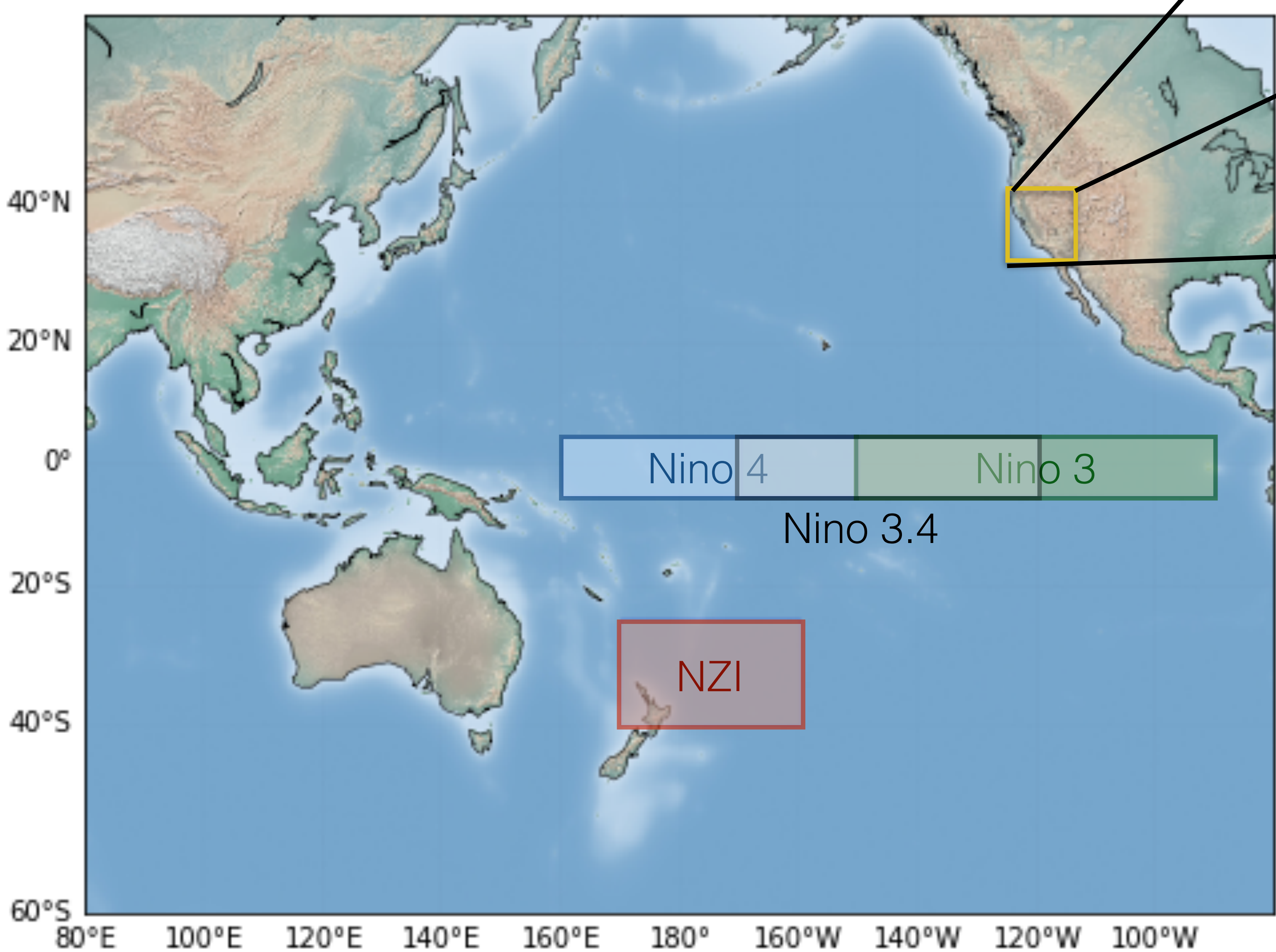
Abby Stevens, UChicago

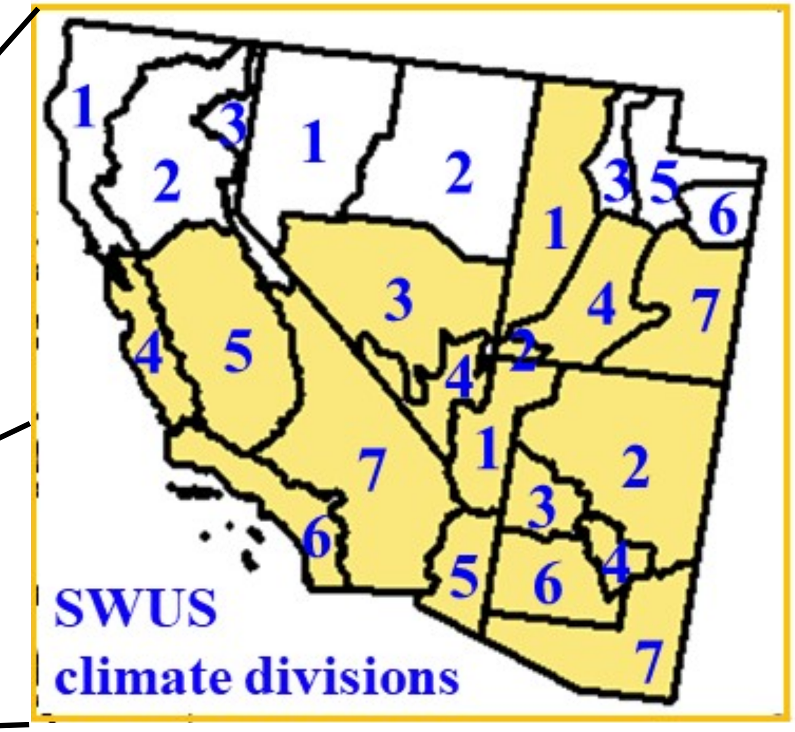
Ben Mark, UW-Madison

Yuan Li, UW-Madison

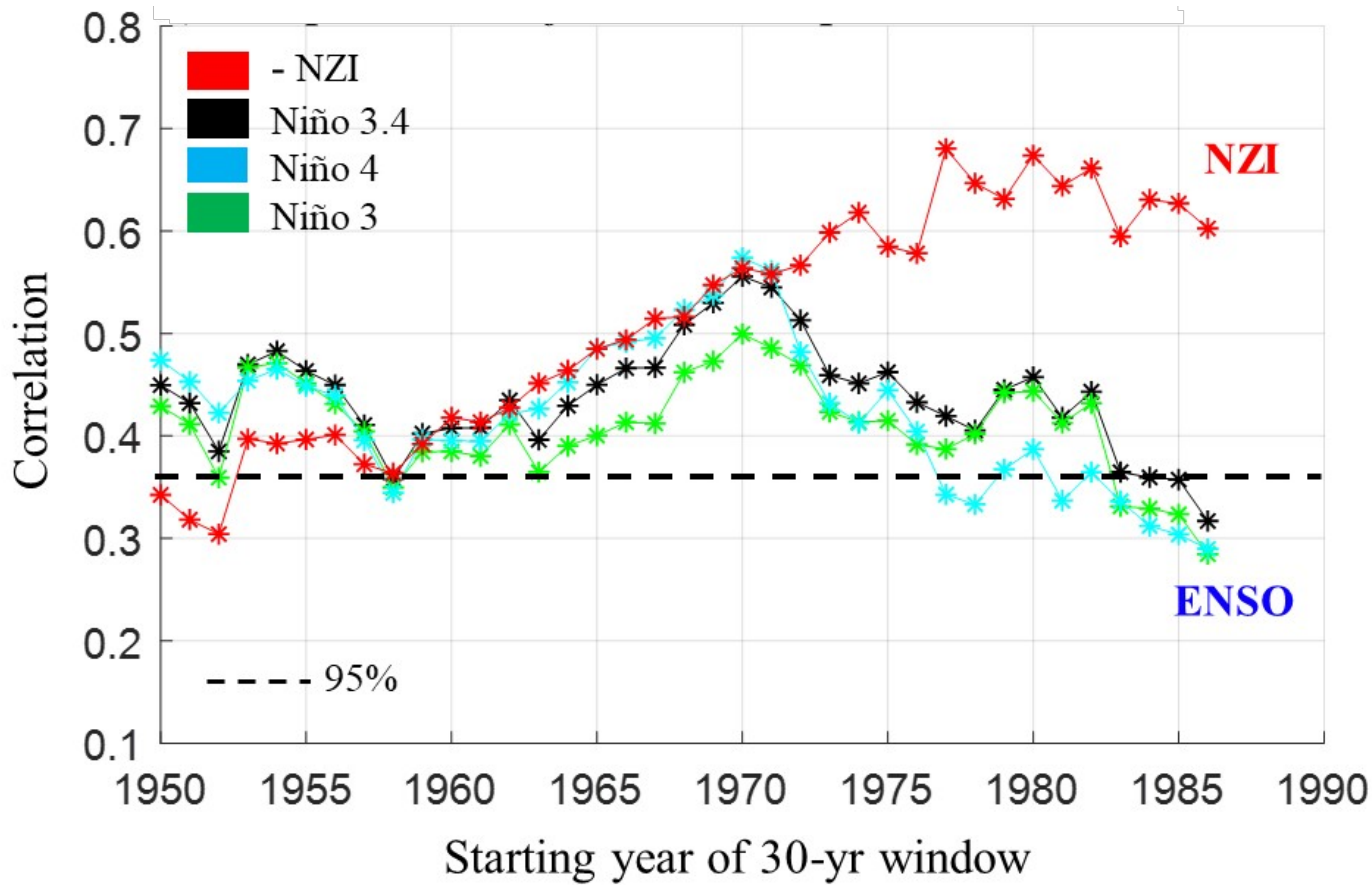
Garvesh Raskutti, UW-Madison

Predicting precipitation in southwest US

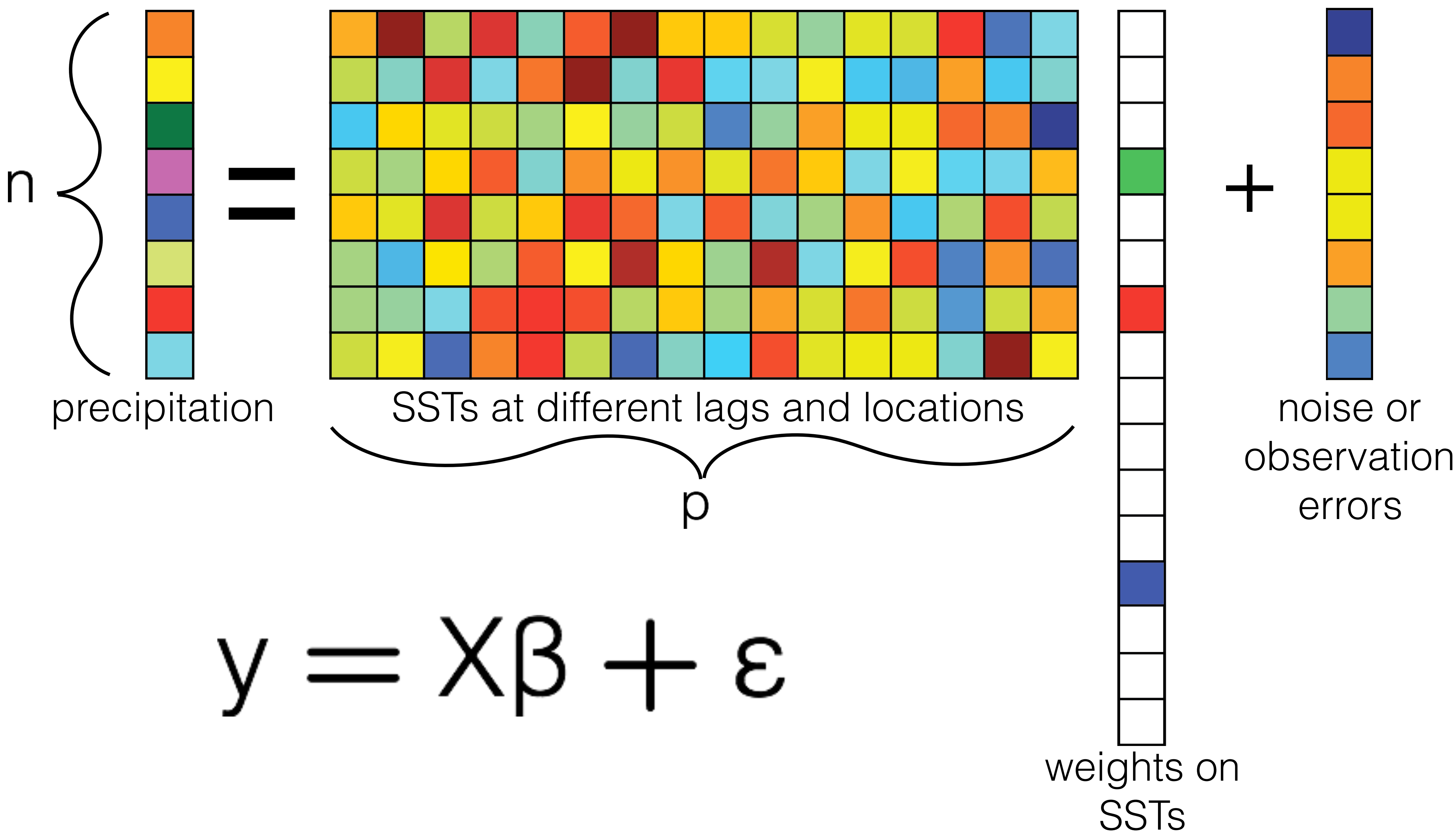




Predicting precipitation in southwest US

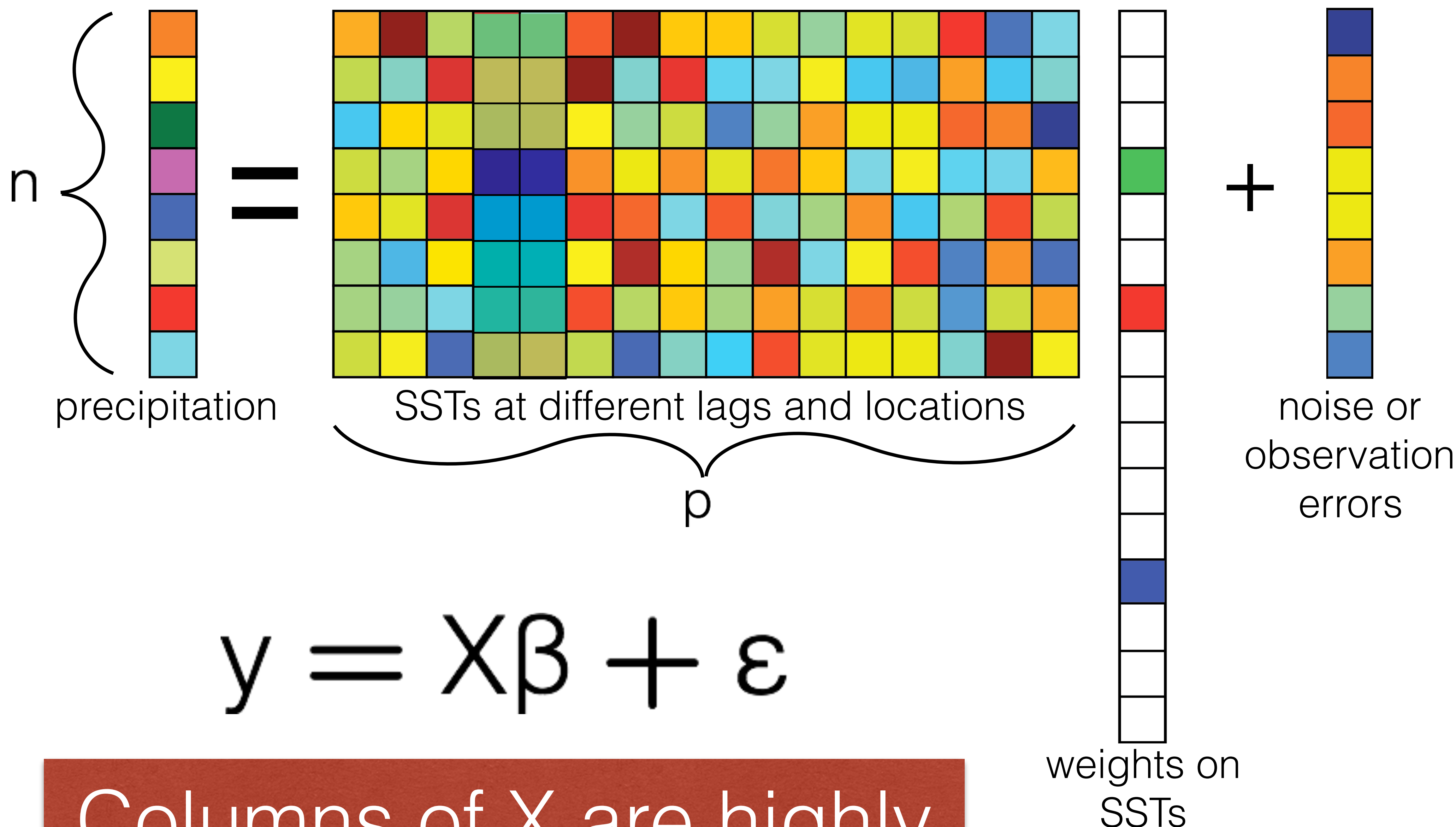


Sparse inverse problems



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

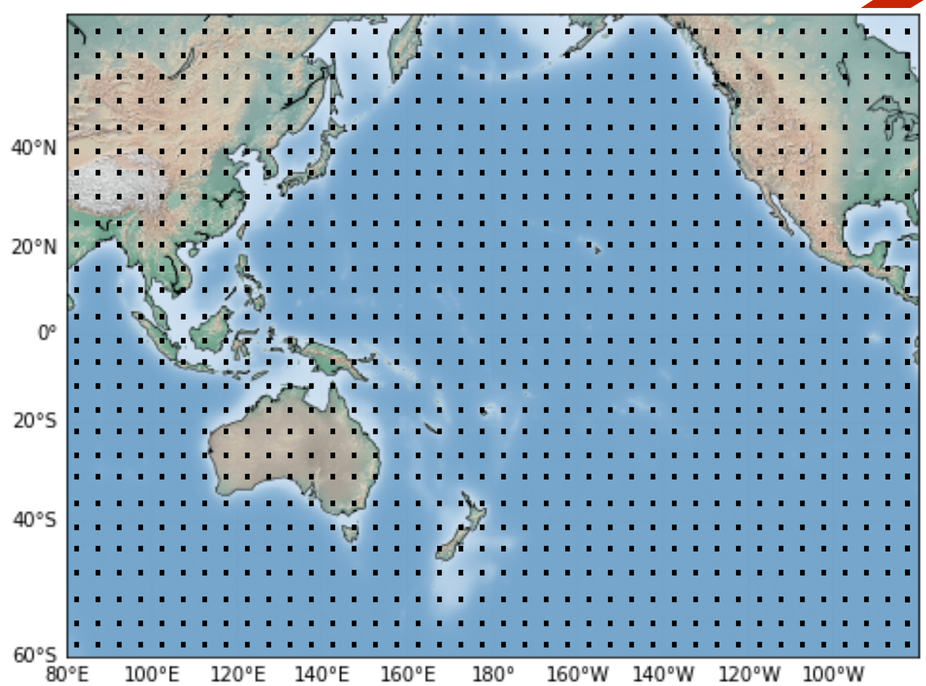
Sparse inverse problems

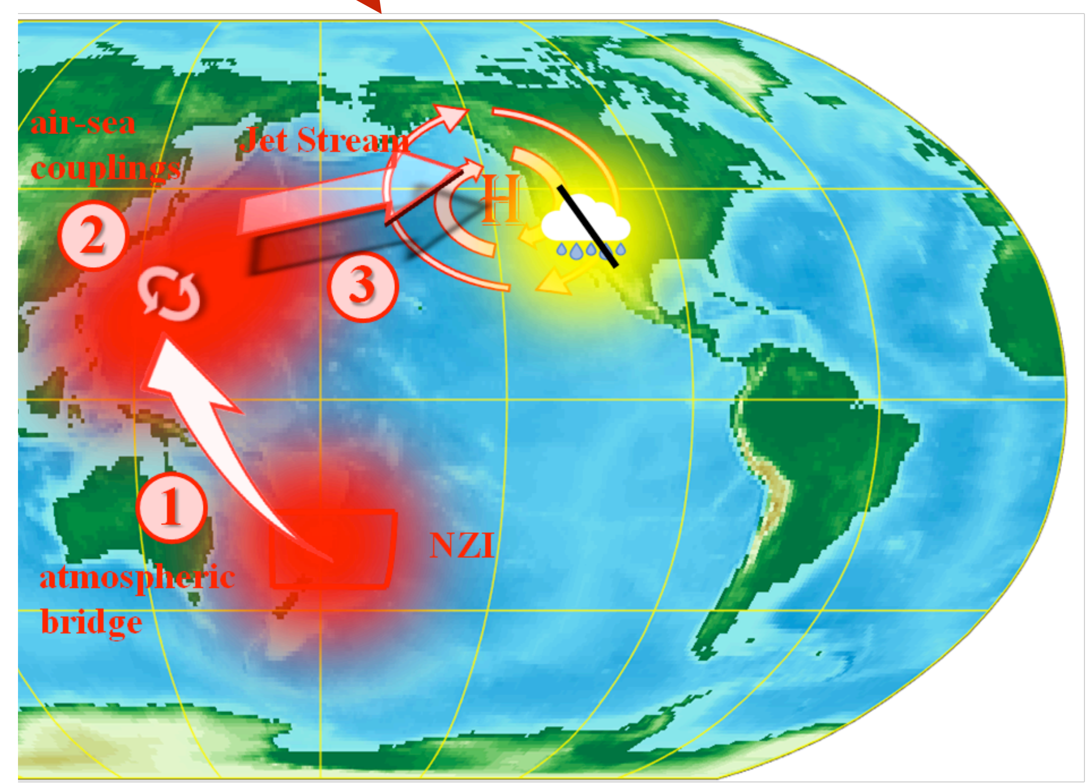


$$y = X\beta + \varepsilon$$

Columns of X are highly correlated

Climate forecasting



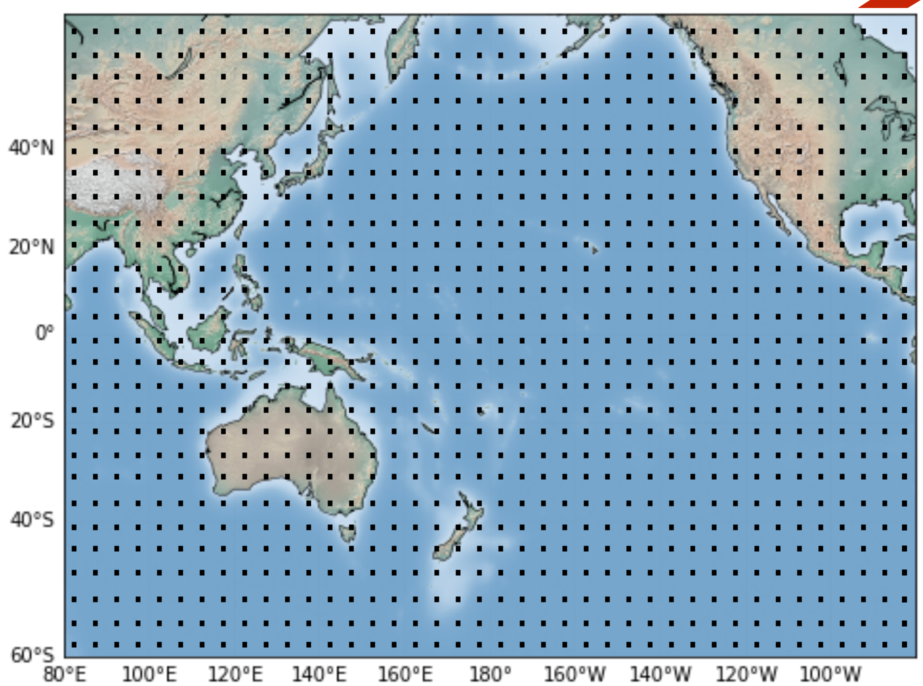


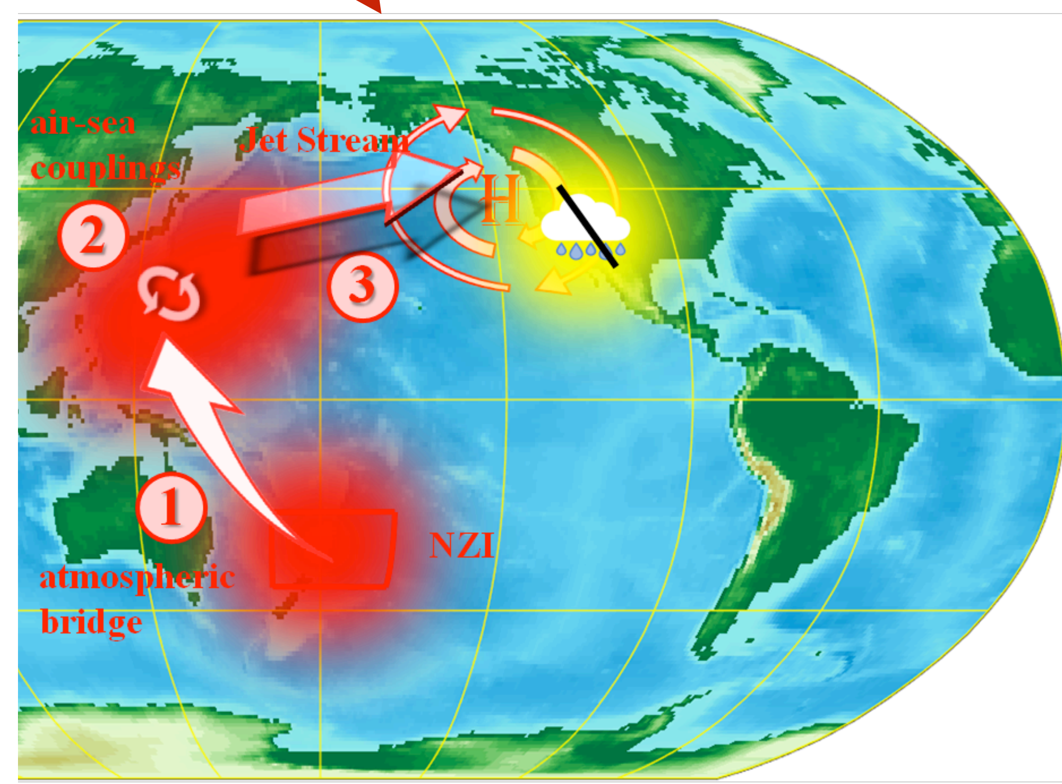
900 spatio-temporal sea-surface temperatures each year

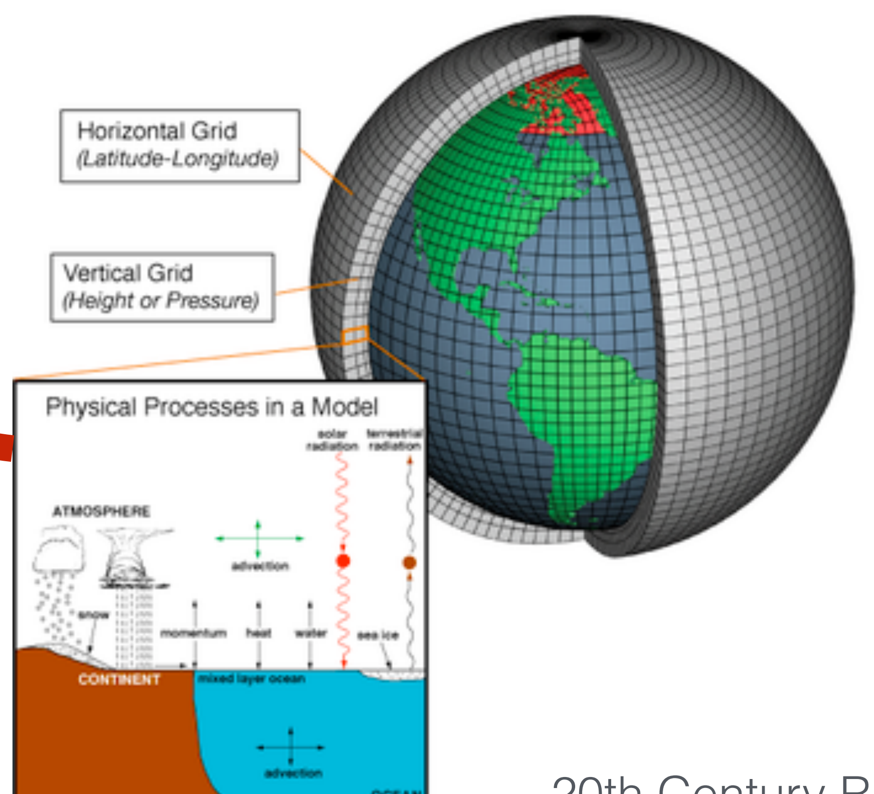
75 years of data

20th Century Reanalysis https://www.esrl.noaa.gov/psd/data/20thC_Rean/

Climate forecasting



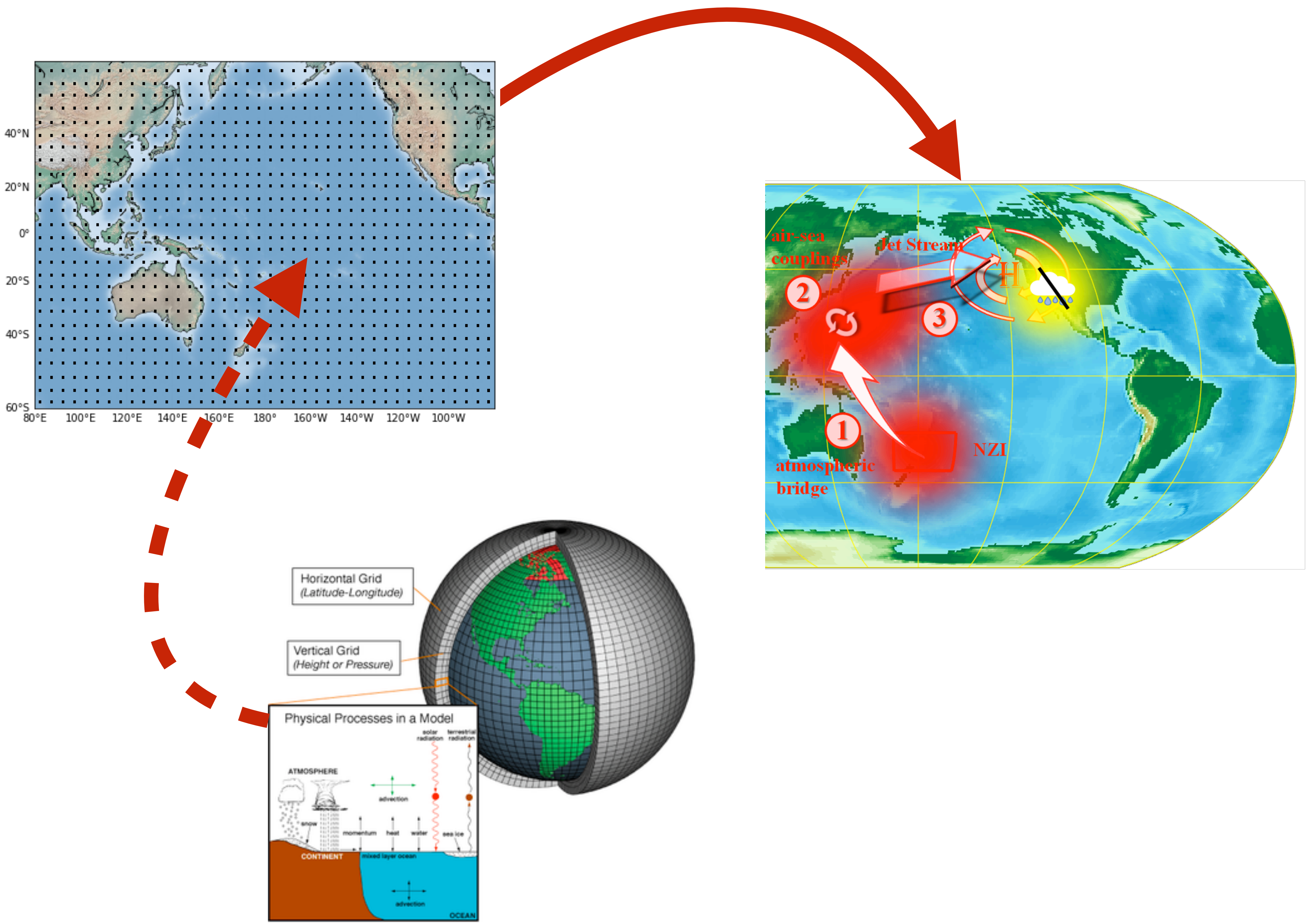




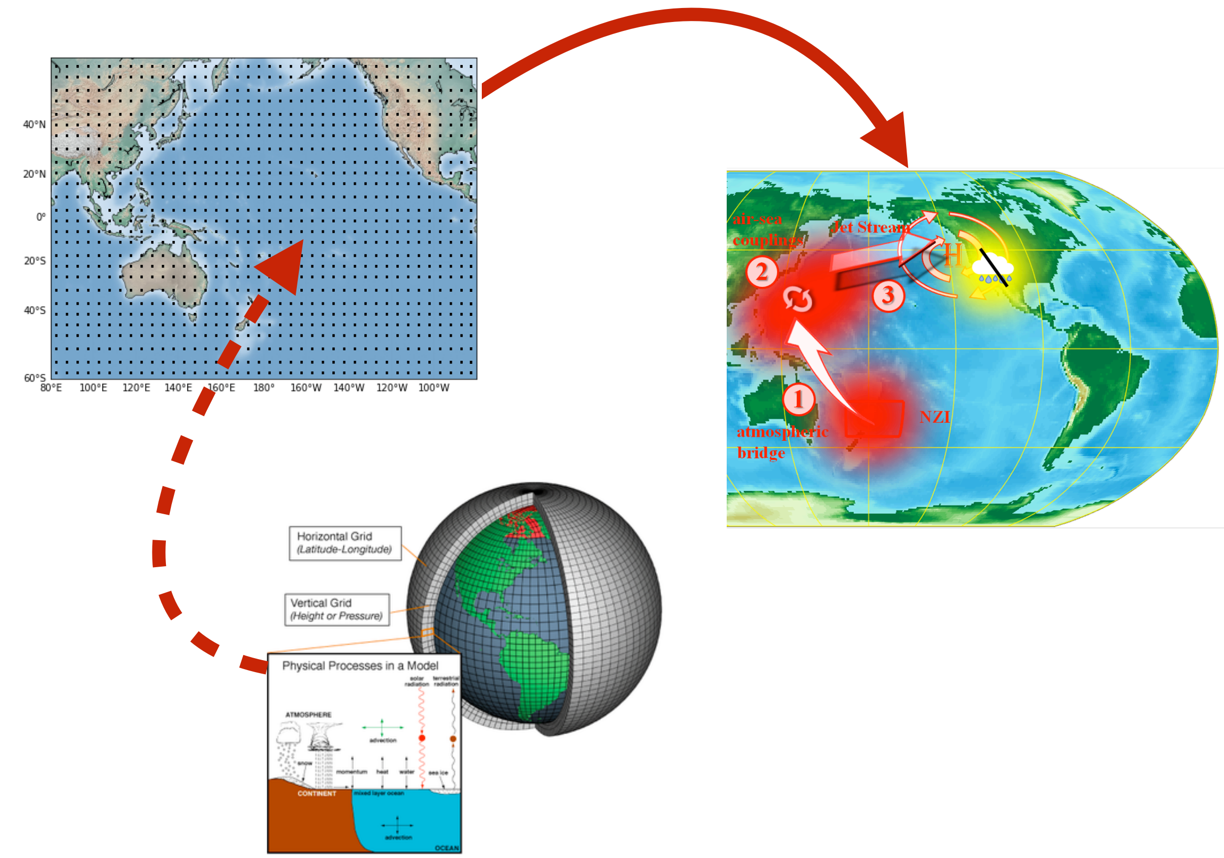
900 spatio-temporal sea-surface temperatures each year

75 years of data

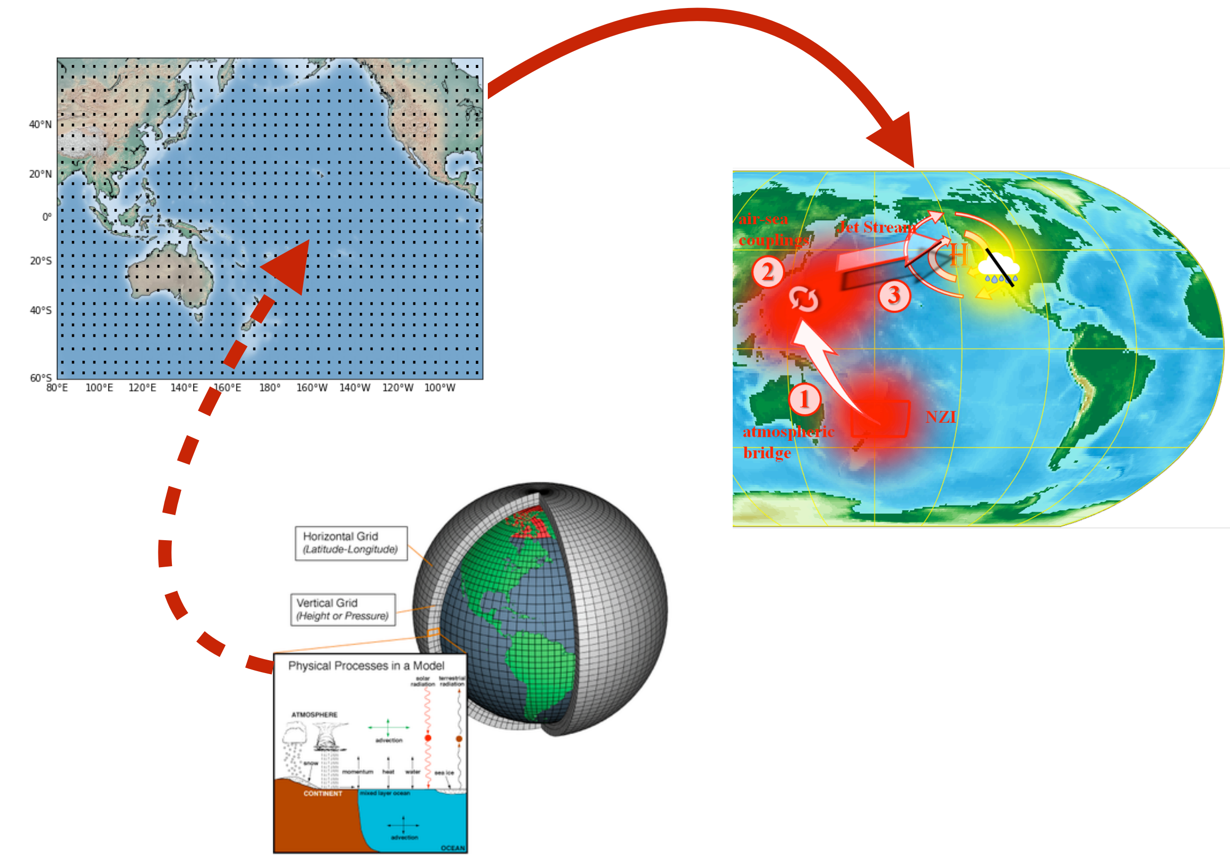
20th Century Reanalysis https://www.esrl.noaa.gov/psd/data/20thC_Rean/


40°N
20°N
0°
20°S
40°S
60°S
80°E
100°E
120°E
140°E
160°E
180°
160°W
140°W
120°W
100°W
air-sea
couplings
Jet Stream
2
3
1
H
NZI
atmospheric
bridge
Horizontal Grid
(Latitude-Longitude)
Vertical Grid
(Height or Pressure)
Physical Processes in a Model
solar radiation
terrestrial radiation
ATMOSPHERE
advection
snow
momentum
heat
water
sea ice
CONTINENT
mixed layer ocean
advection
OCEAN

What is the best way to combine simulated data with observational / experimental data?

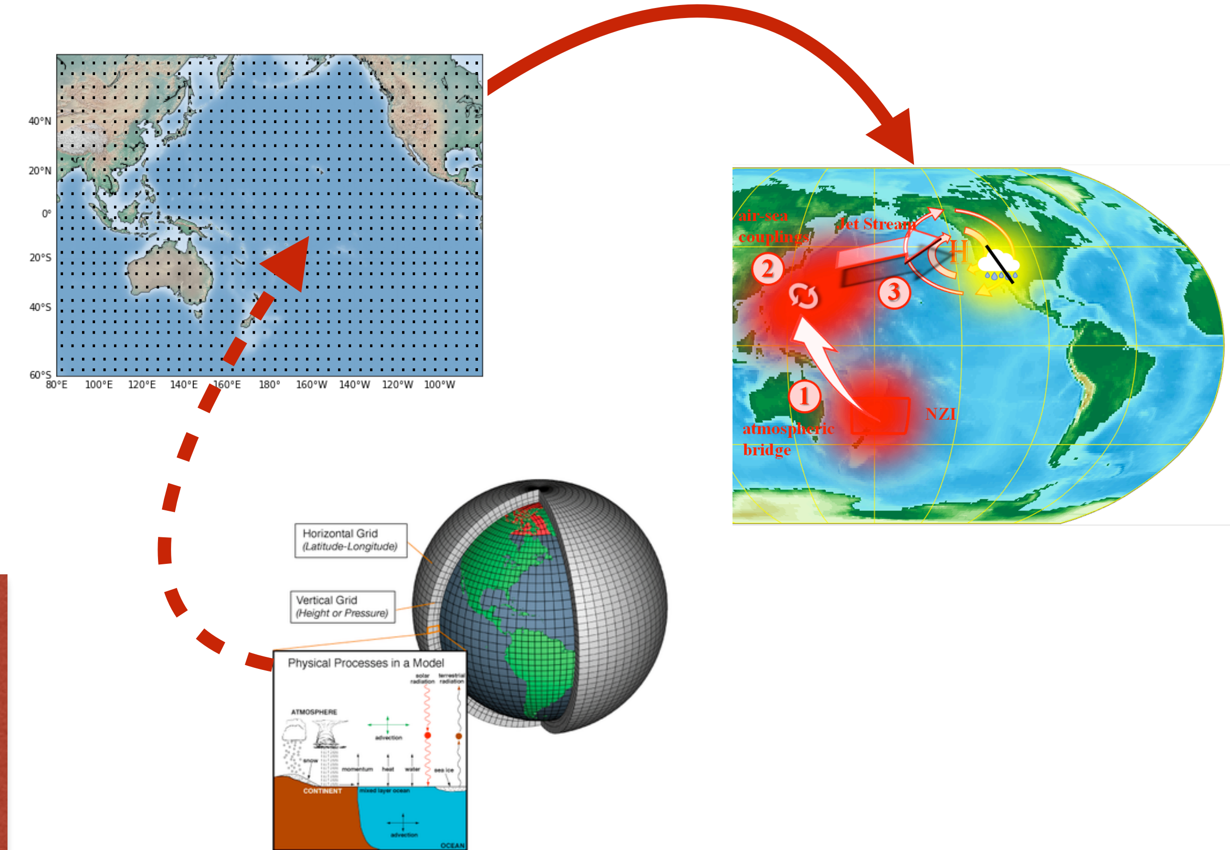


What is the best way to combine simulated data with observational / experimental data?



- Data augmentation (treat simulated data as extra samples from same distribution at experimental data) — poorly understood biases
- Transfer learning (train on simulated data, then tweak learned model using experimental data) — active area of ML
- Prior selection (use simulated data to choose a prior distribution) — GTV is special case of this

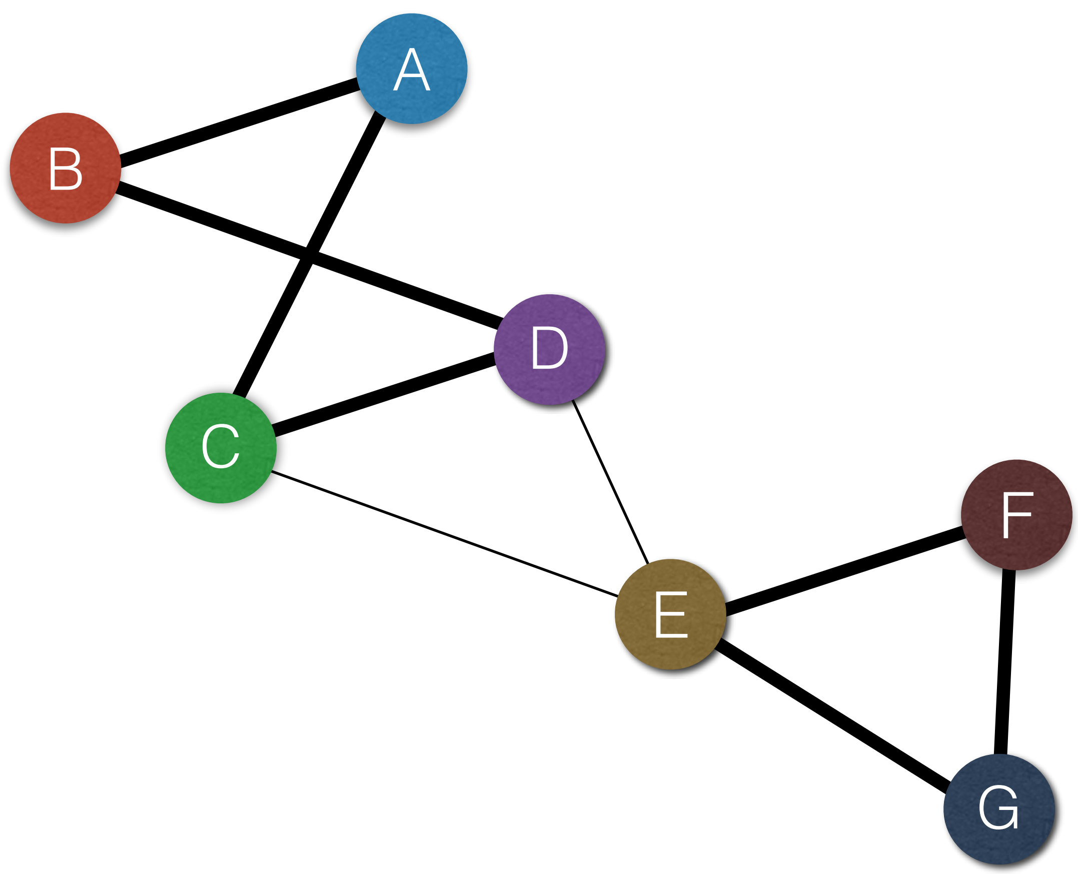
What is the best way to combine simulated data with observational / experimental data?



Depends on physical model accuracy, computational complexity of simulations, scale (mis)match between simulations and experiments, etc.

- Data augmentation (treat simulated data as extra samples from same distribution at experimental data) — poorly understood biases
- Transfer learning (train on simulated data, then tweak learned model using experimental data) — active area of ML
- Prior selection (use simulated data to choose a prior distribution) — GTV is special case of this

Model

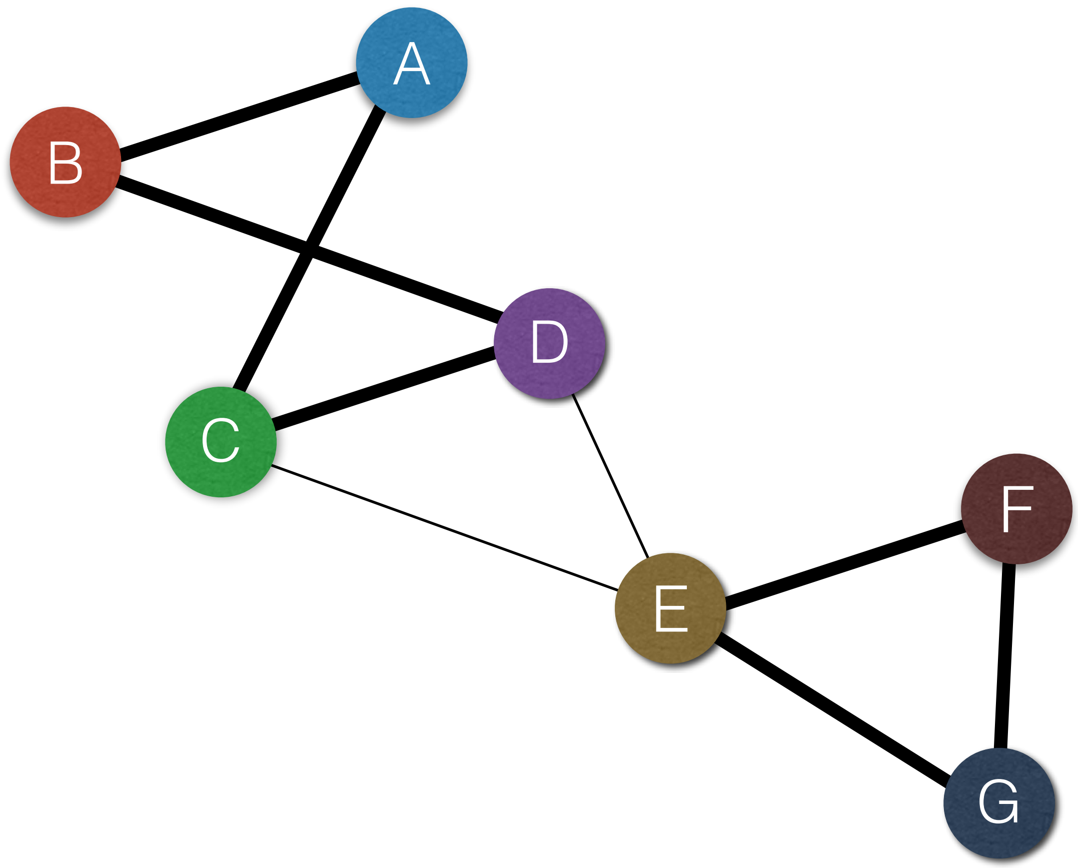


Weighted graph G = (V,E,W)

V = covariates; (E,W) influences:

1. Correlations among covariates (columns of X)
2. Similarity among covariate weights (β's)

Model



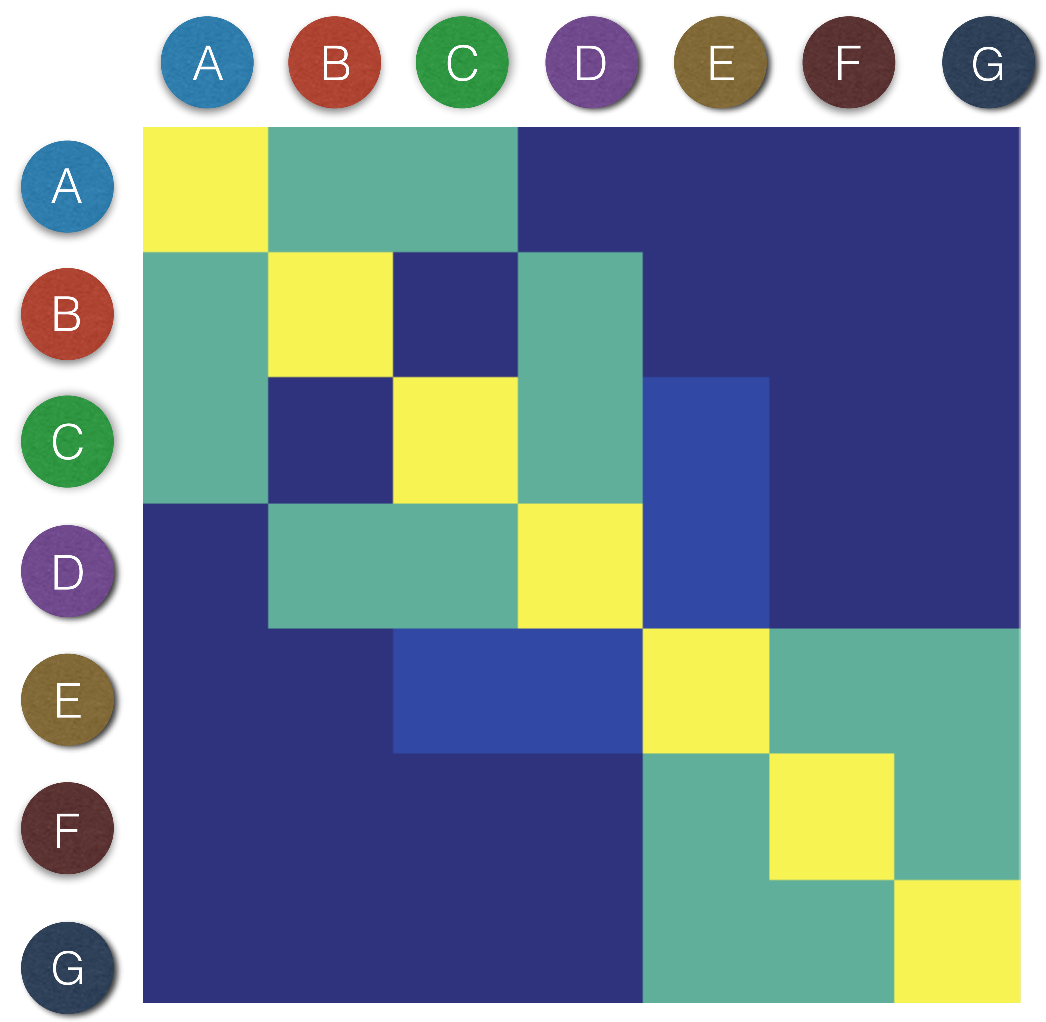
Weighted graph $G = (V,E,W)$

V = covariates; (E,W) influences:

1. Correlations among covariates (columns of X)
2. Similarity among covariate weights (β's)

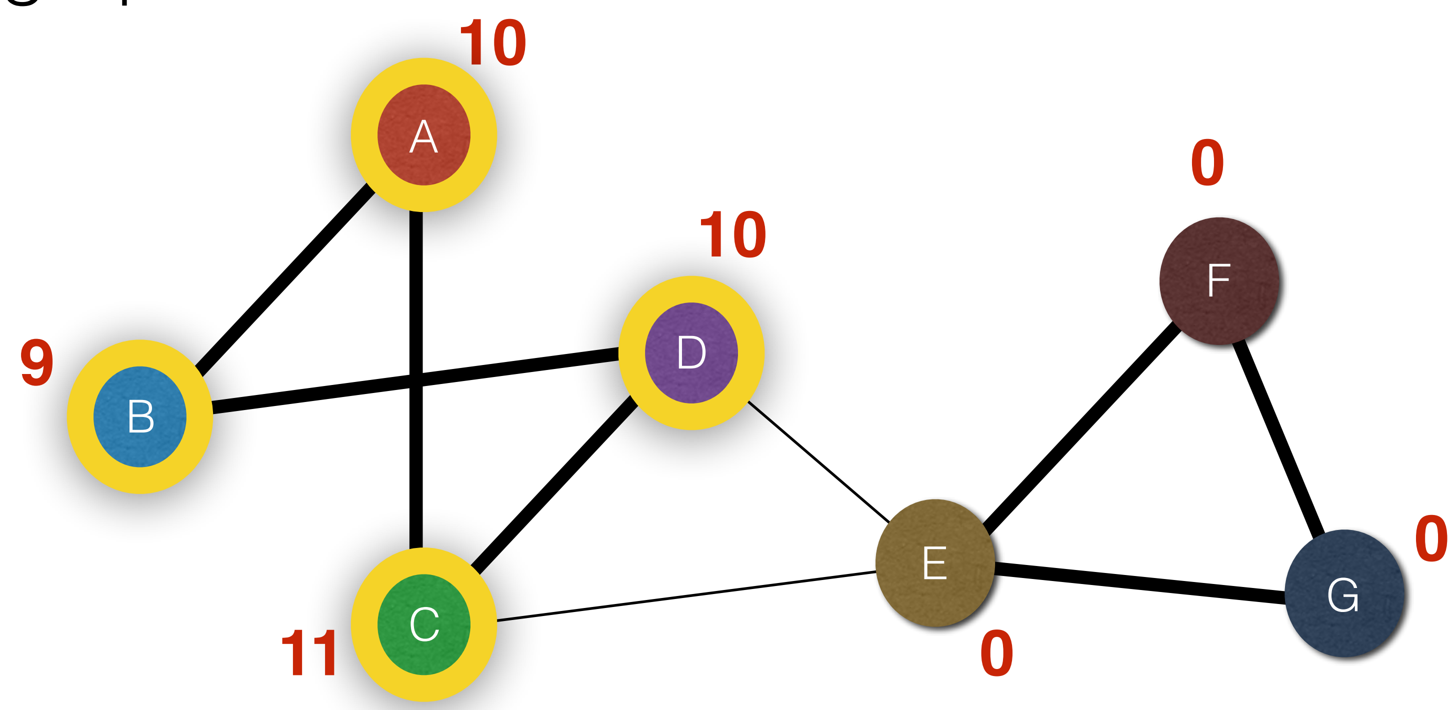
Assume i-th row of X is distributed $X_i \sim \mathcal{N}(0,\Sigma)$

$\sigma_{j,k}$ gives covariance of columns j and k

$\Sigma =$ 

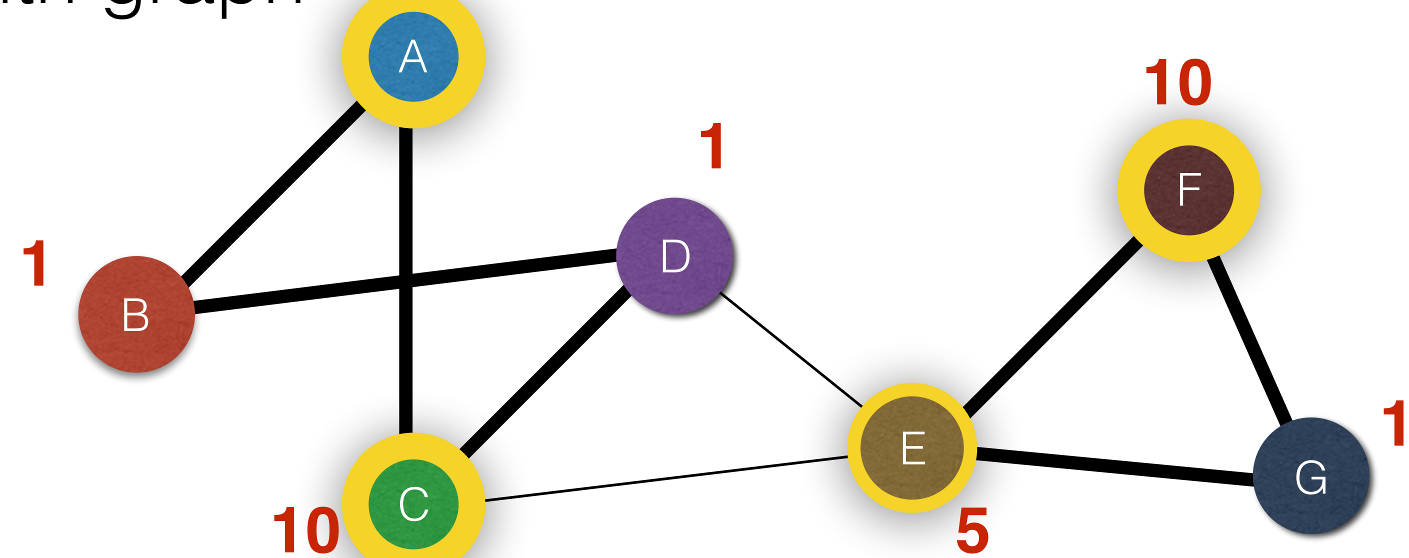
$$\beta = \begin{bmatrix} 10 \\ 9 \\ 11 \\ 10 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

β well-aligned with graph



$$\beta = \begin{bmatrix} 10 \\ 1 \\ 10 \\ 1 \\ 5 \\ 10 \\ 1 \end{bmatrix}$$

β not well-aligned with graph



Graph total variation estimation

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

Data fit

$$+ \lambda_S \sum_{j,k=1}^{p} \sigma_{j,k}(\beta_j - \beta_k)^2$$

Laplacian smoothness

reduces the ill-conditionedness of X when columns are highly correlated

$$+ \lambda_1 \lambda_{TV} \sum_{j,k=1}^{p} \sigma_{j,k}^{1/2} |\beta_j - \beta_k|$$

Graph total variation

promotes estimates that are well-aligned with graph structure

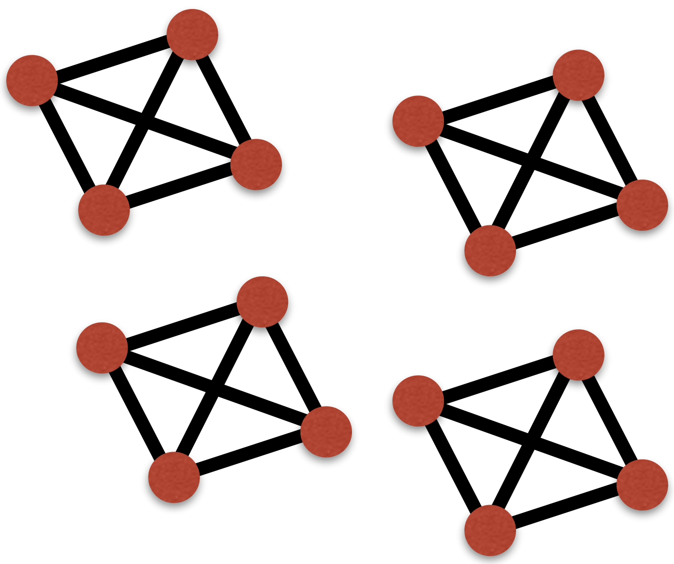
$$+ \lambda_1 ||\beta||_1$$

LASSO

promotes sparsity

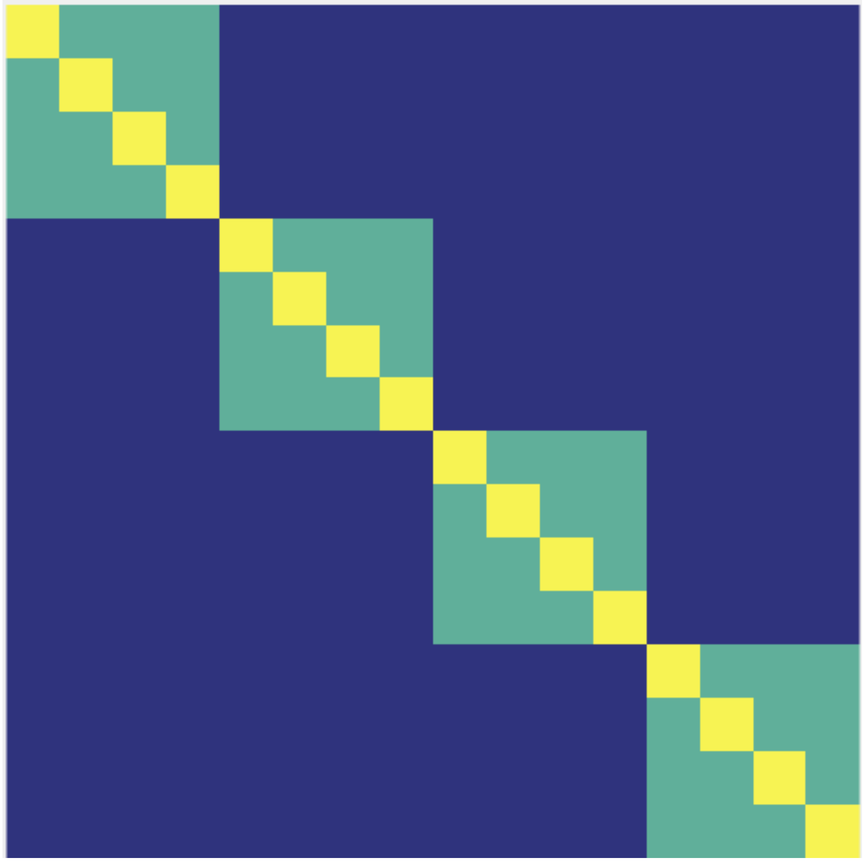
Method finds a ***sparse set of covariate clusters*** that encode information on response

Example 1: Highly correlated clusters

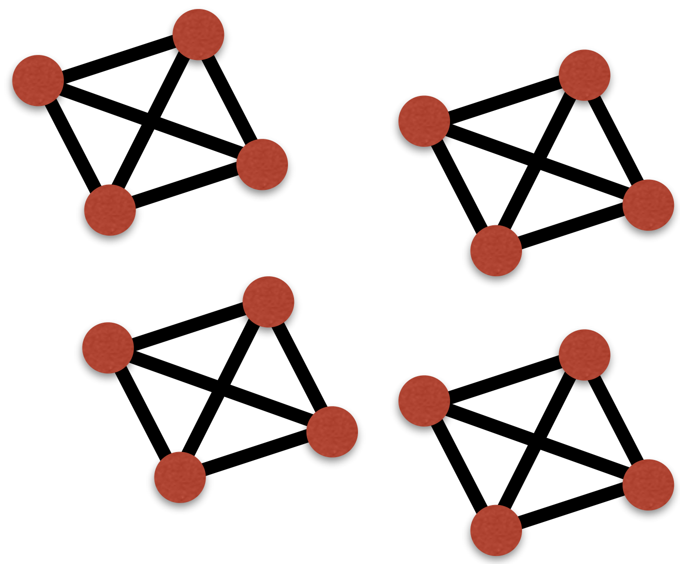


Columns of X in well-separated clusters

$$\Sigma =$$

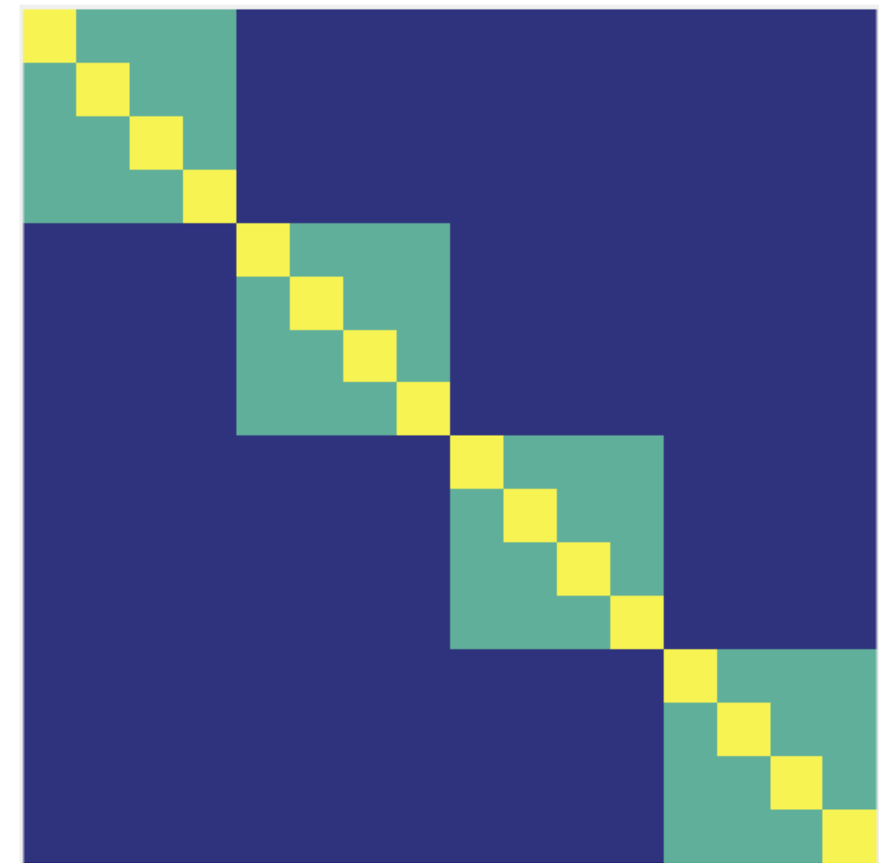


Example 1: Highly correlated clusters



Columns of X in well-separated clusters

$$\Sigma =$$



B = # blocks containing nonzero elements of β

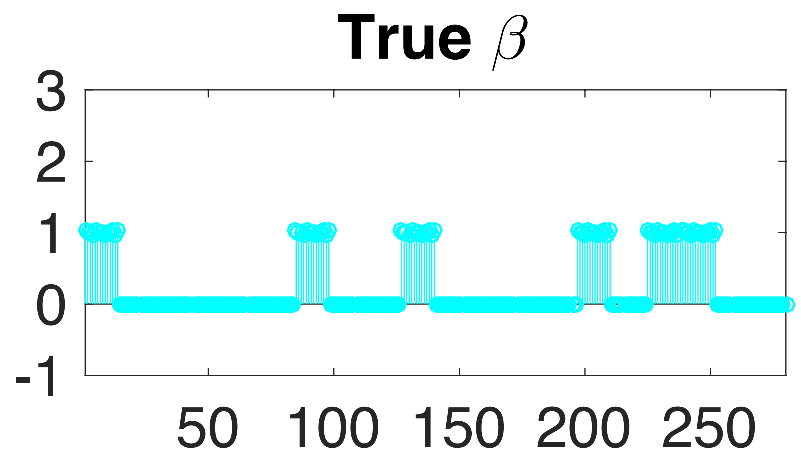
$$\|\beta - \hat{\beta}_{GTV}\|_2^2 \preceq \frac{B \log p}{n}$$

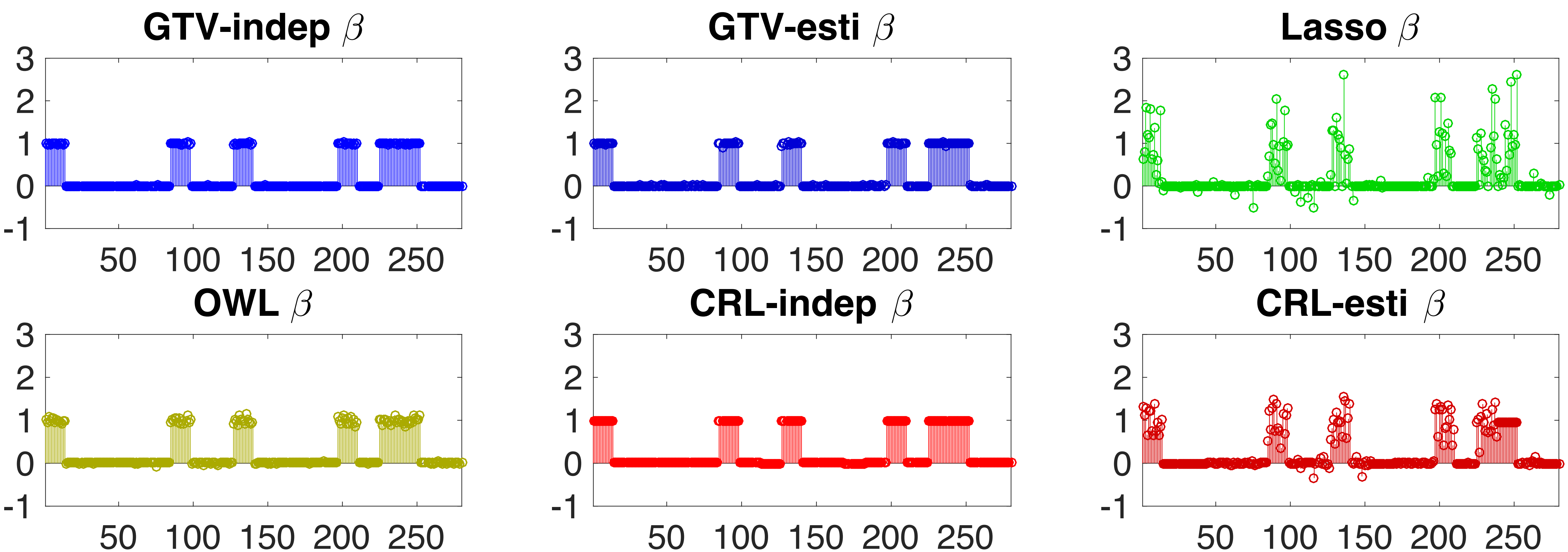
$$\|\beta - \hat{\beta}_{LASSO}\|_2^2 \preceq \frac{\|\beta\|_0 \log p}{n}$$

Much bigger than B!

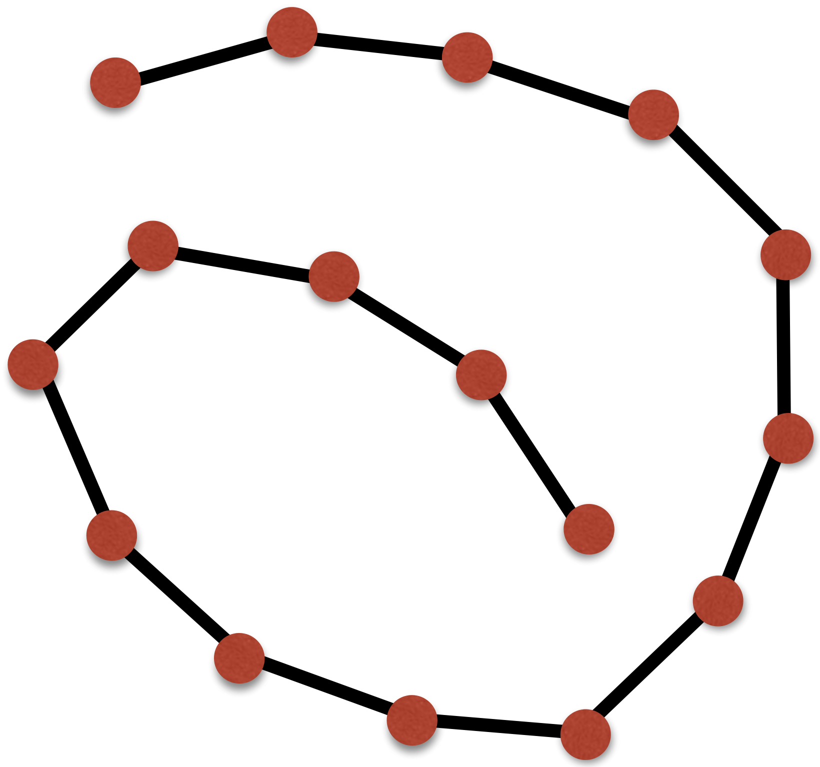
Highly correlated clusters: Estimates

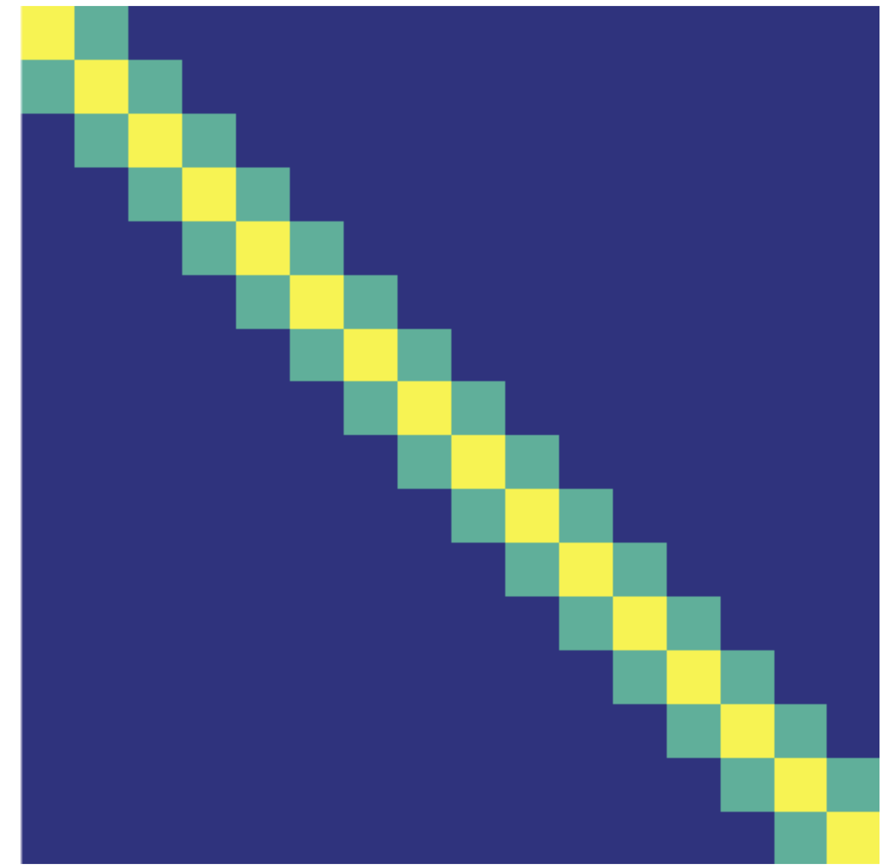
p = 280 = number of covariates
n = 100 = number of responses
r = .6 = off-diagonal correlation strength
a = 2 = diagonal variance
|Supp(β)| = 84
6/20 active blocks





Example 2: Chain graph



$\Sigma =$ 

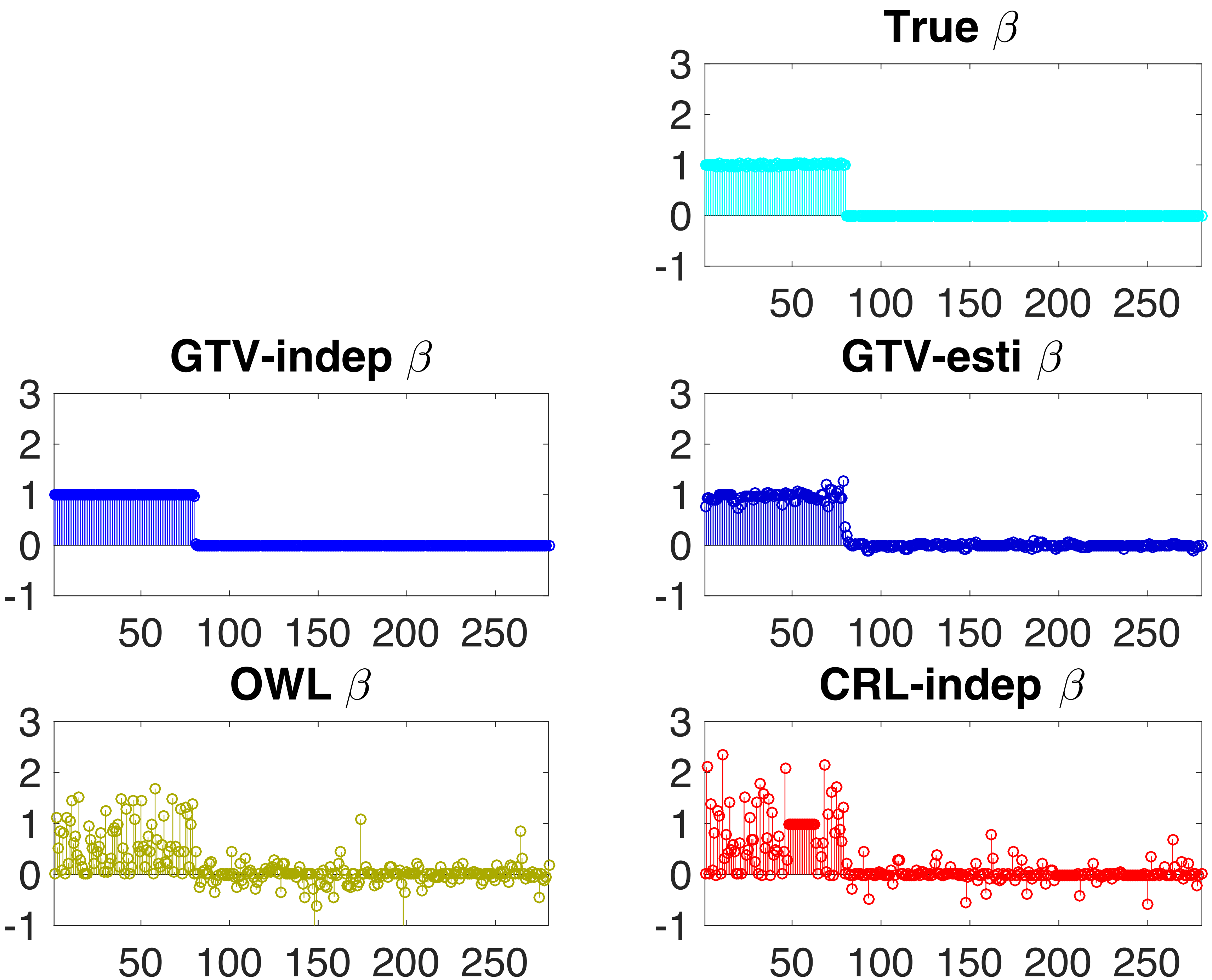
Columns of X **not** in well-separated clusters

$$\|\beta - \hat{\beta}_{GTV}\|_2^2 \preceq \frac{\sqrt{\|\beta\|_0 \log p}}{n}$$

$$\|\beta - \hat{\beta}_{LASSO}\|_2^2 \preceq \frac{\|\beta\|_0 \log p}{n}$$

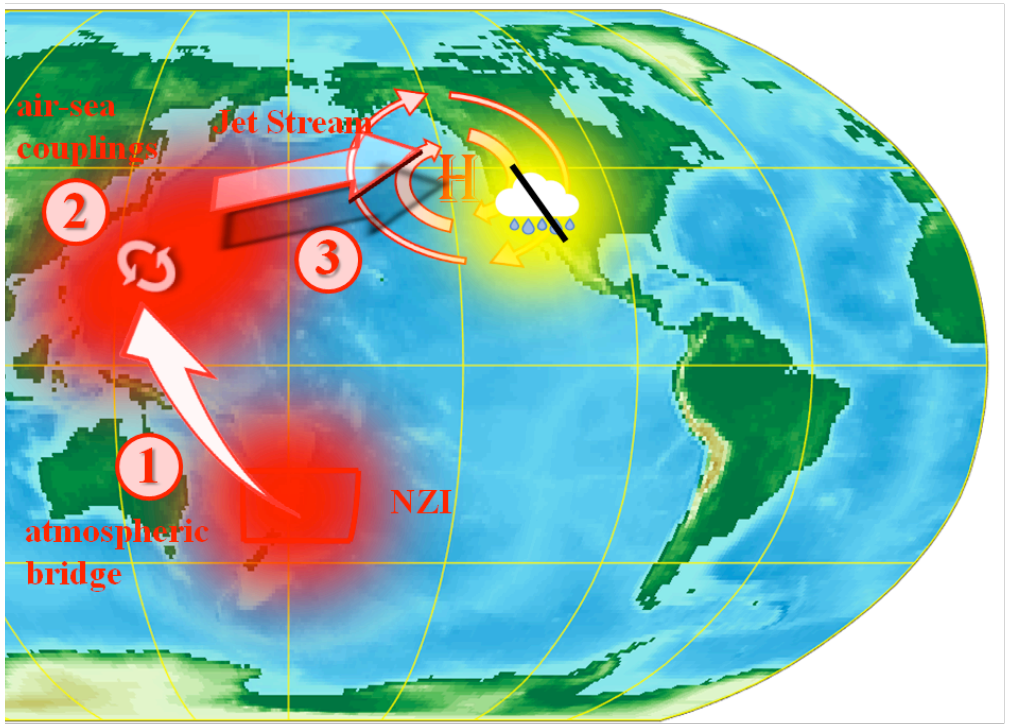
Chain graph: Estimates

p = 280 = number of covariates
n = 100 = number of responses
r = .45 = off-diagonal correlation strength
a = 2 = diagonal variance
|Supp(β)| = 80



GTV in climate forecasting

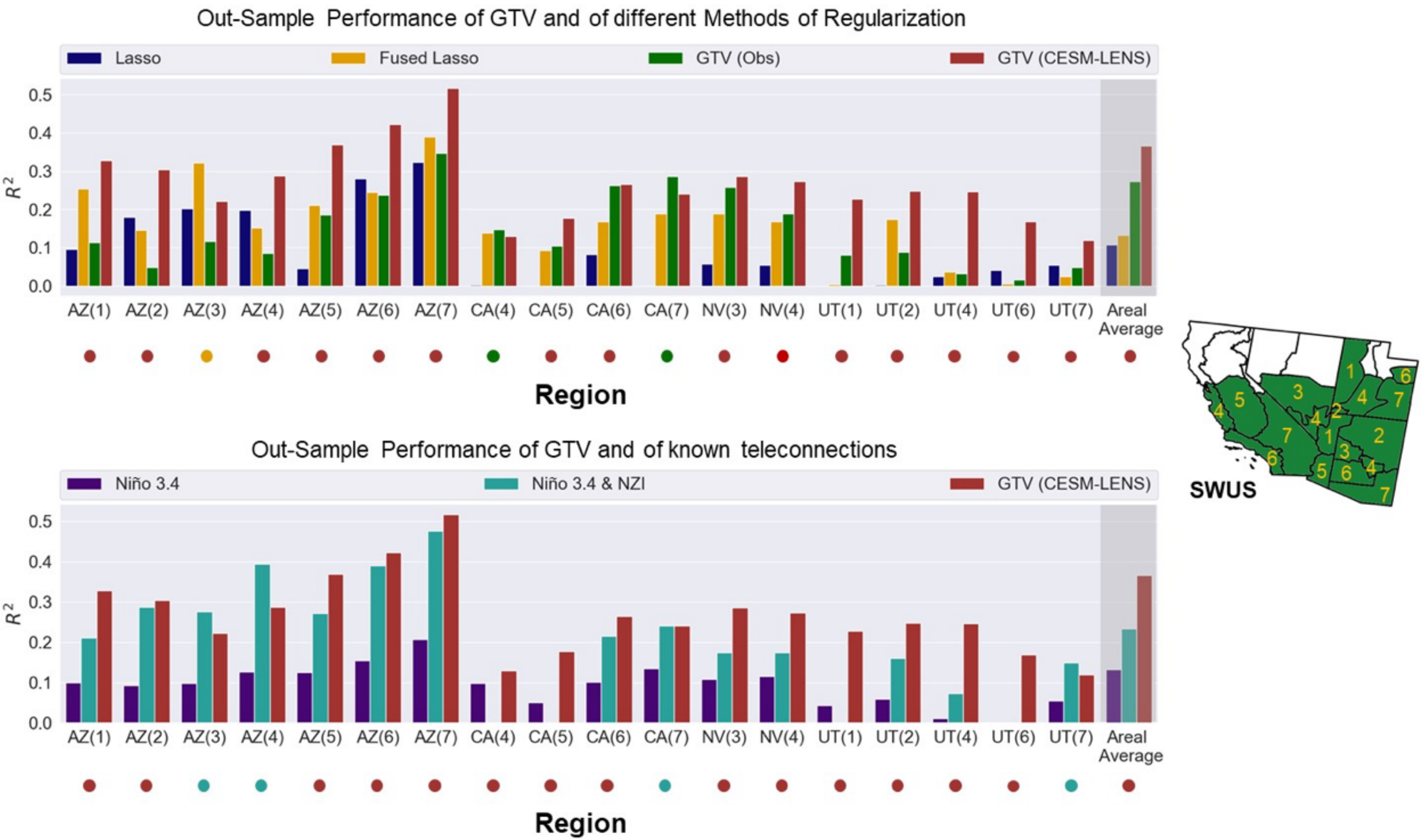
- We have 75 years of observational data
- We also have physical models we can use to generate simulated data:
  - Large Ensemble Community Project (LENS)
  - 40 independent 75-year simulations of SSTs and precipitation
- How can we best leverage this?



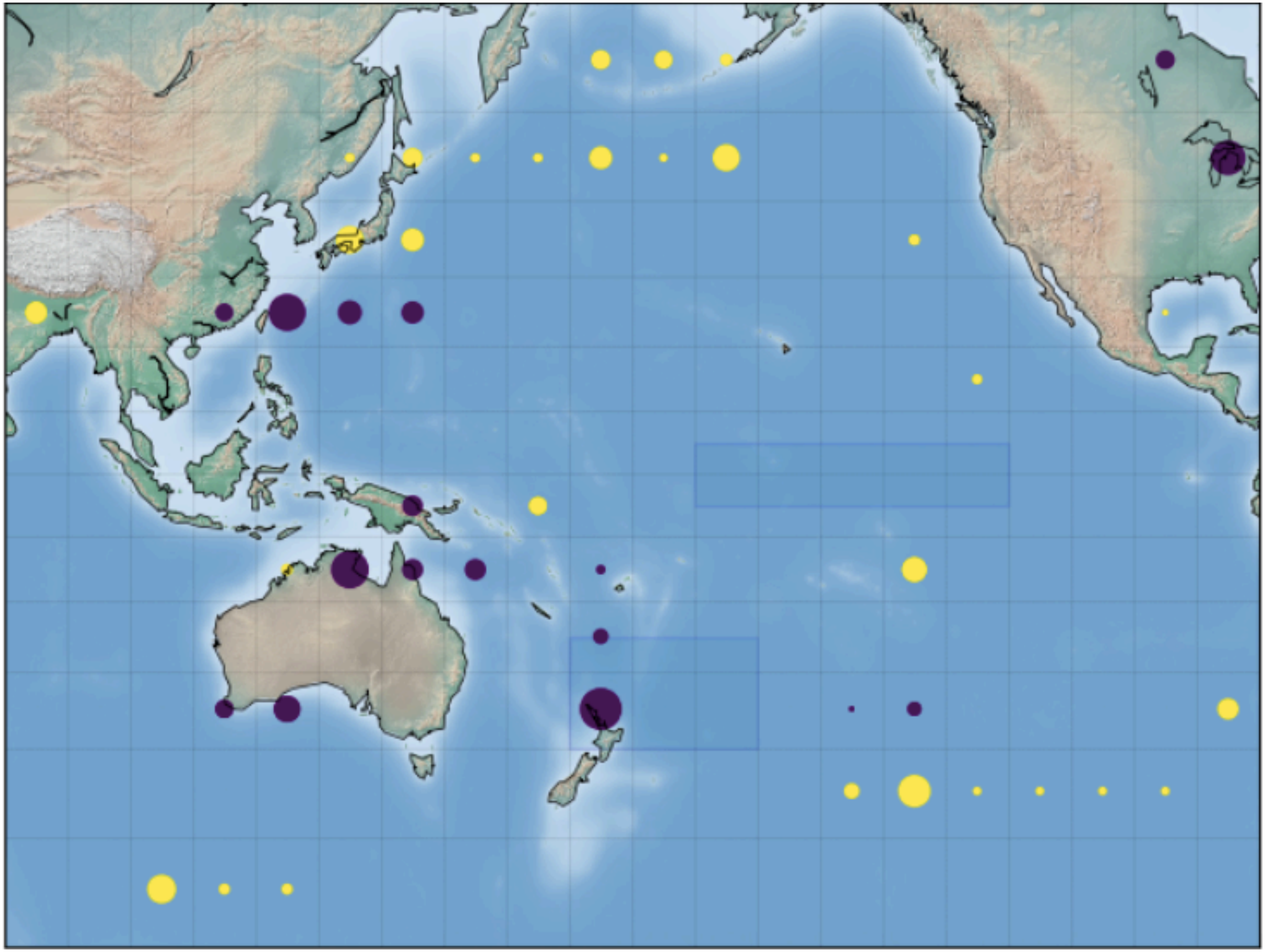
Efi Foufoula-Georgiou, UCI

Jim Randerson, UCI

https://www.esrl.noaa.gov/psd/data/20thC_Rean/



Out-Sample Performance of GTV and of different Methods of Regularization
Lasso
Fused Lasso
GTV (Obs)
GTV (CESM-LENS)
R^2
0.0
0.1
0.2
0.3
0.4
0.5
AZ(1) AZ(2) AZ(3) AZ(4) AZ(5) AZ(6) AZ(7) CA(4) CA(5) CA(6) CA(7) NV(3) NV(4) UT(1) UT(2) UT(4) UT(6) UT(7) Areal Average
Region
Out-Sample Performance of GTV and of known teleconnections
Niño 3.4
Niño 3.4 & NZI
GTV (CESM-LENS)
R^2
0.0
0.1
0.2
0.3
0.4
0.5
AZ(1) AZ(2) AZ(3) AZ(4) AZ(5) AZ(6) AZ(7) CA(4) CA(5) CA(6) CA(7) NV(3) NV(4) UT(1) UT(2) UT(4) UT(6) UT(7) Areal Average
Region
SWUS

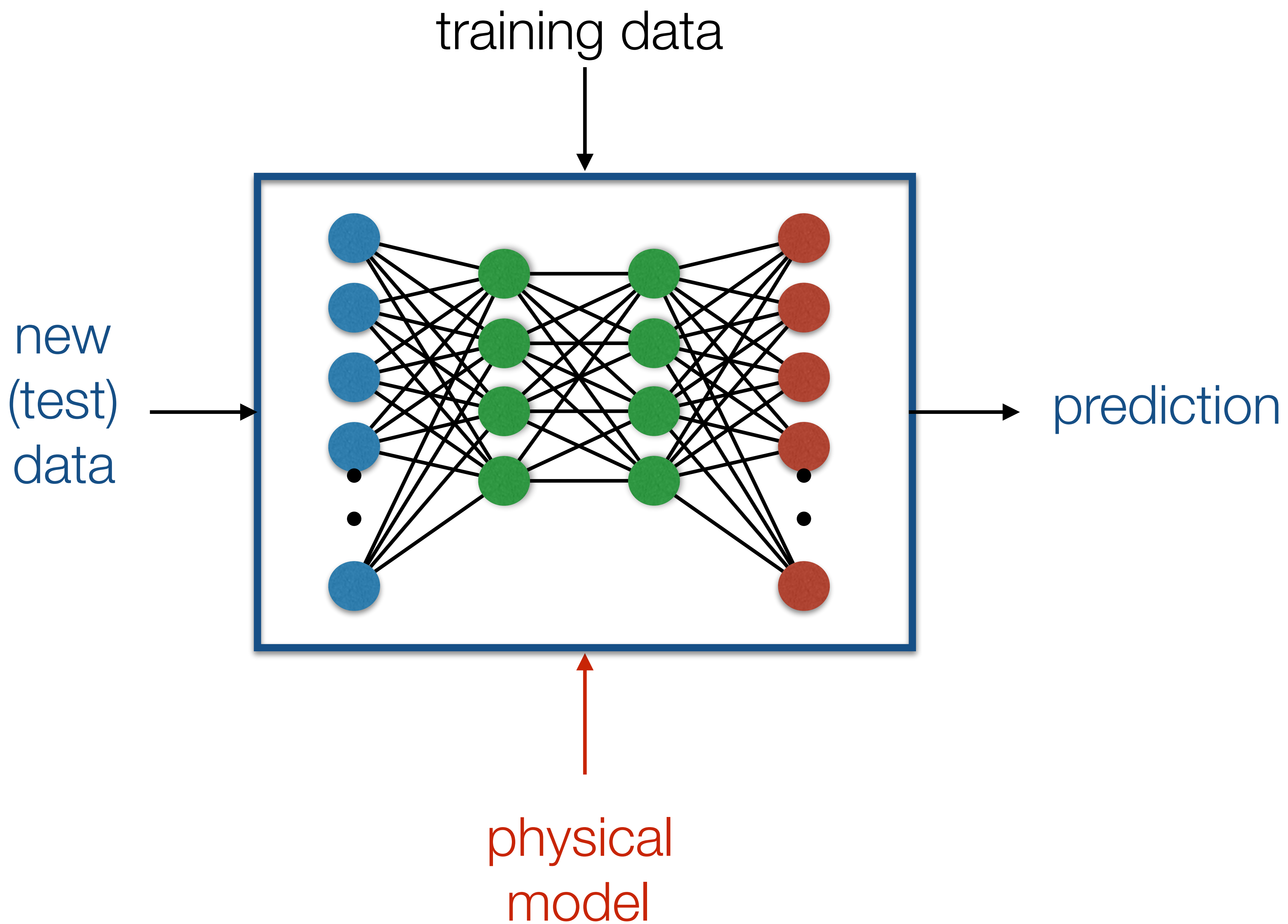


arXiv:1803.07658 [pdf, other] stat.ML

Graph-based regularization for regression problems with highly-correlated designs

Authors: Yuan Li, Benjamin Mark, Garvesh Raskutti, Rebecca Willett

How do we leverage a combination of training data and physical models?



Physical models and training data

- Training data can be limited in volume, expensive to collect → we may learn **over-simplified predictors**

- Physical models can be inaccurate or biased → we may end up with a **biased predictor**

- If we think of machine learning as using training data to search over a family of predictors, then **physical models help constrain the set of viable predictors**

- Fundamental tradeoffs among volumes of training data, manifestation of physical models, and risk minimization present **significant open challenges**

Thank you!