Seismic data interpolation with Generative Adversarial Networks

Ali Siahkoohi and Felix Herrmann - Fall 2017

SLIM
University of British Columbia

Spitz, S., 1991. Seismic trace interpolation in the FX domain. *Geophysics*, *56*(6), pp.785-794.
Herrmann, F.J. and Hennenfent, G., 2008. Non-parametric seismic data recovery with curvelet frames. *Geophysical Journal International*, *173*(1), pp. 233-248.
Kumar, R., Mansour, H., Herrmann, F.J. and Aravkin, A.Y., 2013. Reconstruction of seismic wavefields via low-rank matrix factorization in the hierarchical-separable matrix representation. In *SEG Technical Program Expanded Abstracts 2013* (pp. 3628-3633). Society of Exploration Geophysicists.

Interpolation

Interpolation schemes rely on prior information on the data to fill in missing values:

- seismic data consists of limited number of events.
- sparsity in transform domain.
- low-rank structure of seismic data in coordinate transformed domain.

Can we use probabilistic information?

Why probabilistic information?

In probabilistic methods, if we have a **precise model** for our data:

- We don’t need to make any additional assumptions about model

Assumptions not always work for us:

- As frequency increases, frequency slices become less low rank
- If we miss a big chunk of data, interpolating using Curvelets becomes less efficient

Probabilistic point of view

Images can be thought as samples from a complex high-dimensional probability distribution

Maximum likelihood estimation for finding the probability distribution (if we **assume** we have an probability function)

$$\theta^* = \max_{\theta} \; \frac{1}{m} \sum_{i=1}^{m} \log p(\underbrace{x^{(i)}}_{i^{th}\text{ data sample}}; \underbrace{\theta}_{\text{Model parameters}})$$

Maximum likelihood estimation

If data samples are IID, then we look for a set of parameters which maximize the probability of data being observed:

$$\max_{\theta} p(X;\theta) = \max_{\theta} \prod_{i=1}^{m} p(x^{(i)};\theta) = \max_{\theta} \ \frac{1}{m}\sum_{i=1}^{m} \log p(x^{(i)};\theta)$$

θ : Model parameters

m : Number of data samples

p : Explicit probability function for model

X : All the data samples

$x^{(i)}$: i^{th} data sample

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

SLIM

Generative models

Instead of writing out a function $p(x;\theta)$, learn to draw samples from the distribution directly.

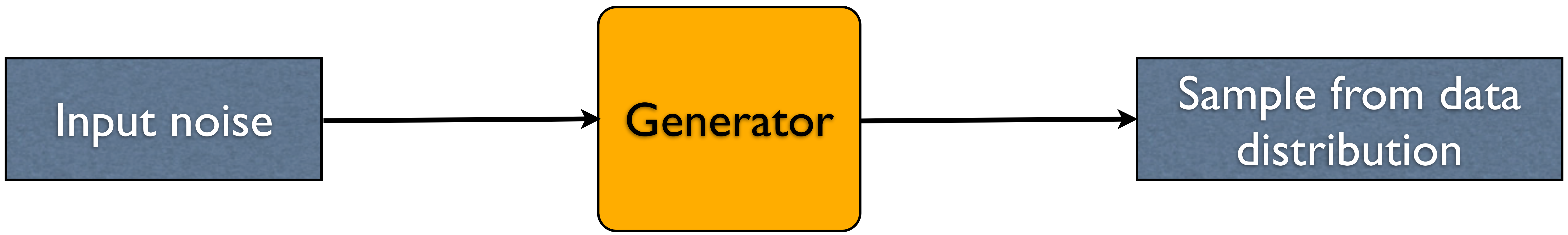
Generative Adversarial Network is a way to learn to sample from complex, high-dimensional training set that comes from a distribution.

How? By playing a game between two players:

- Discriminator (D)
- Generator (G)

Goal of the game

Learn a transformation mapping of noise into data distribution



The game

Player D's task: discrimination between:

- a sample from the data distribution
- and a sample from the generator

Player G's task: try to "fool" D by generating samples that are hard for D to discriminate from data.

Competition drives both players to improve their methods.

SLIM

Players in our game

Two players in the game are represented by two **differentiable** functions.

- Discriminator:
  - Input: x from data space
  - Output: probability that $x \sim p_{\mathrm{data}}$
- Generator:
  - Input: $z \sim \mathcal{U}(-1, 1)$
  - Output: a mapping to data space

Goodfellow, I., 2016. NIPS 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160.

The game played in GAN

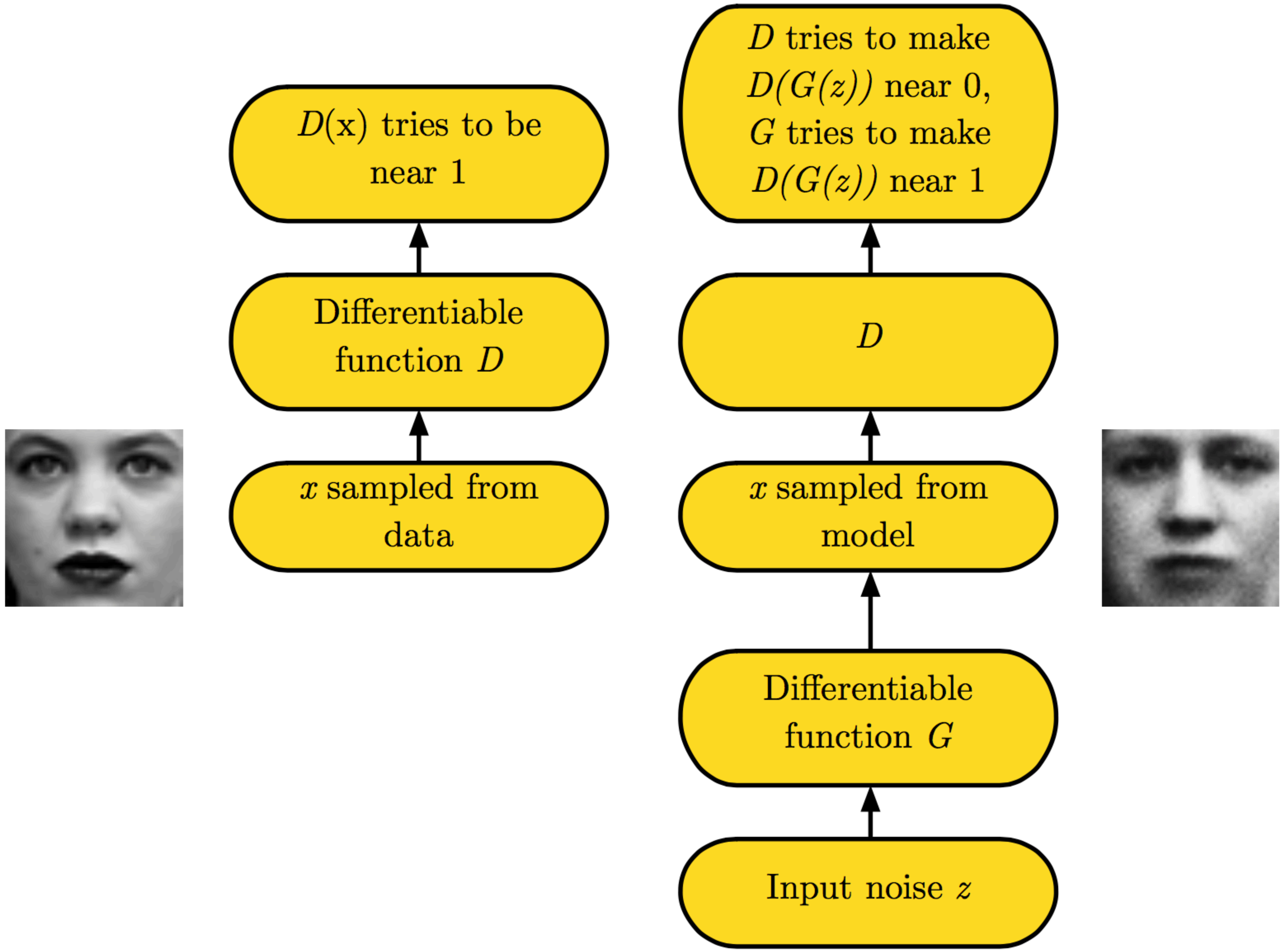


Figure 1 GANs framework

Notation

Generator

$$z \sim \mathcal{U}(-1,1) \longrightarrow \boxed{\theta^{(G)}} \longrightarrow \begin{array}{r} G_{\theta^{(G)}}(z) \sim p_{\text{model}} \\ G_{\theta^{(G)}}(z) \in dom(\text{data}) \end{array}$$

Discriminator

$$x \in dom(\text{data}) \longrightarrow \boxed{\theta^{(D)}} \longrightarrow D_{\theta^{(D)}}(x) = prob(x \sim p_{\text{data}})$$

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

Game in equations

The simplest version of the game is **zero-sum** game:

$$\min_{\theta^{(G)}} \max_{\theta^{(D)}} \left\{ \mathop{\mathbb{E}}_{x \sim p_{\text{data}}(x)} \underbrace{[\log D_{\theta^{(D)}}(x)]}_{\text{Discriminator's ability to recognize data as being real}} + \mathop{\mathbb{E}}_{z \sim p_z(z)} \underbrace{[\log(1 - D_{\theta^{(D)}}(G_{\theta^{(G)}}(z)))]}_{\text{Discriminator's ability to recognize output of generator as being fake}} \right\}$$

Generator parameters

Discriminator parameters

Samples drawn from data distribution

Samples drawn from latent space

Nowozin, S., Cseke, B. and Tomioka, R., 2016. f-gan: Training generative neural samplers using variational divergence minimization. In Advances in Neural Information Processing Systems (pp. 271-279).

SLIM

Dealing with $\mathop{\mathbb{E}}_{x \sim p_{\text{data}}(x)}(.)$

Given finite training dataset, X, we approximate the expectation using batch of samples:

- sample m data points, **without replacement** from X

$$\mathop{\mathbb{E}}_{x \sim p_{\text{data}}(x)} [\log D(x)] \simeq \frac{1}{m} \sum_{j=1}^{m} \log(D(x_{i(j)}))$$

Radford, A., Metz, L. and Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

GANs: Convolutional Architectures (DCGAN)

Stable set of neural network architectures for training generative adversarial networks.

Using **convolutional neural networks** as generator and discriminator functions.

GAN trained on seismic frequency slices

We trained a DCGAN on seismic frequency slices for a specific frequency.

The size of each frequency slice is 68x68.

The images are normalized so that details can be visible.

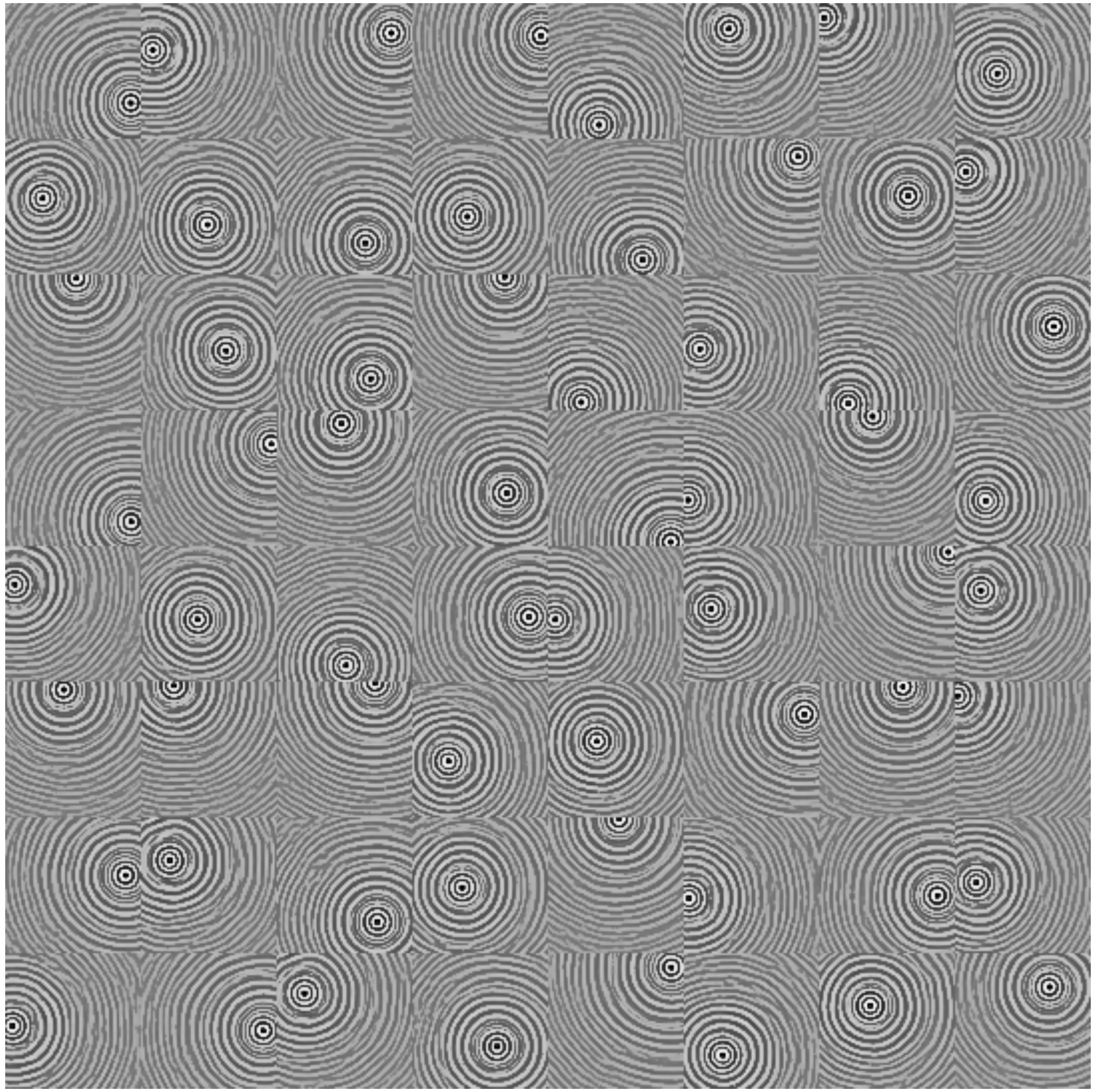


Figure 3 Original images in dataset

Dataset

Normalized seismic frequency slices in data set

Synthetic 3D BG model

68 x 68 sources

401 x 401 receivers

Data at 7.43 Hz

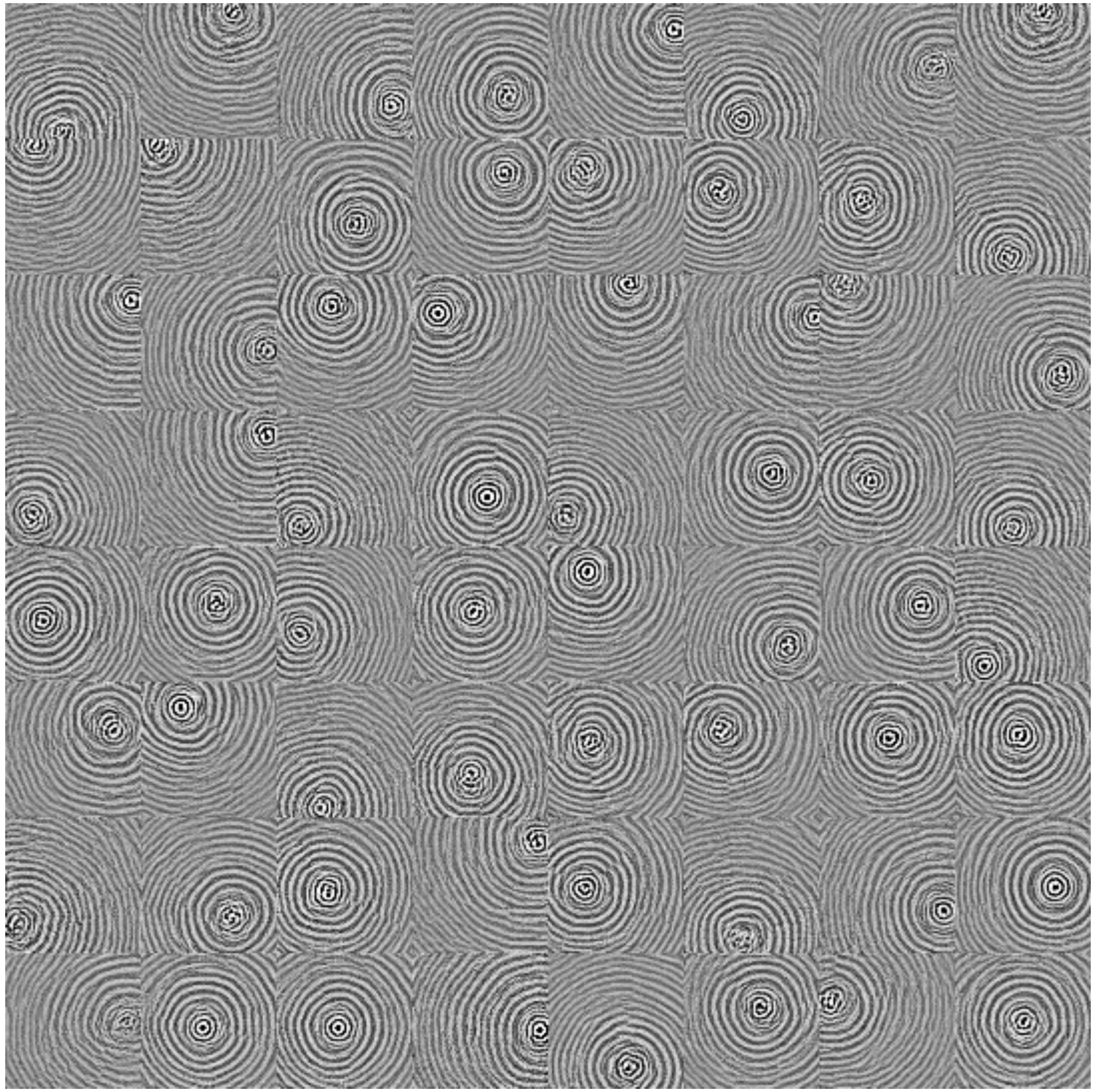


Figure 4 Output of generator for several random inputs

Fake slices

Random outputs of the trained generator on seismic frequency slices

What can we do with GANs?

Now that we can sample from the probability distribution of interest, we can do the following:

- find missing values in images
- map between two different image domains.
- and much more...

Yeh, R., Chen, C., Lim, T.Y., Hasegawa-Johnson, M. and Do, M.N., 2016. Semantic image inpainting with perceptual and contextual losses. arXiv preprint arXiv:1607.07539.

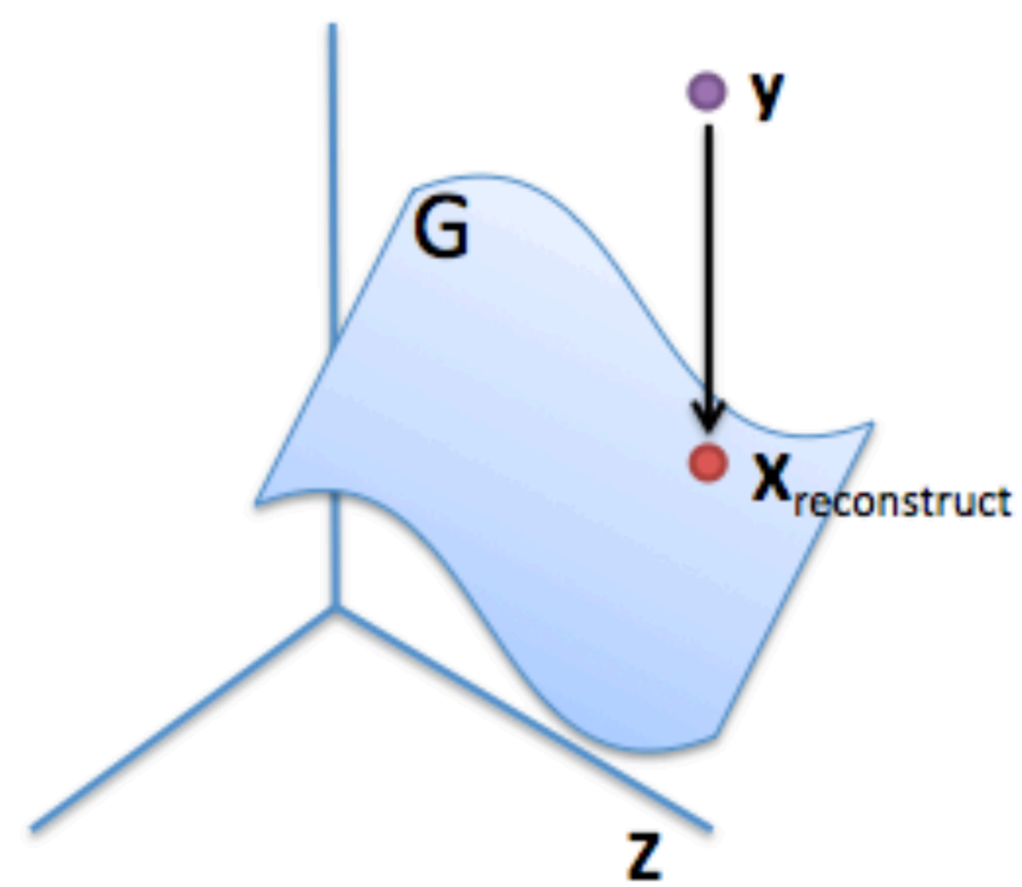
SLIM

Image reconstruction

Find the closest image on the range of generator to the corrupted image.

Looking for z such that the mapping $G(z)$ is close to corrupted image where we have data

$$L(z) = \|M(G(z) - y)\|$$



Mask for existing data.

Yeh, R., Chen, C., Lim, T.Y., Hasegawa-Johnson, M. and Do, M.N., 2016. Semantic image inpainting with perceptual and contextual losses. arXiv preprint arXiv:1607.07539.

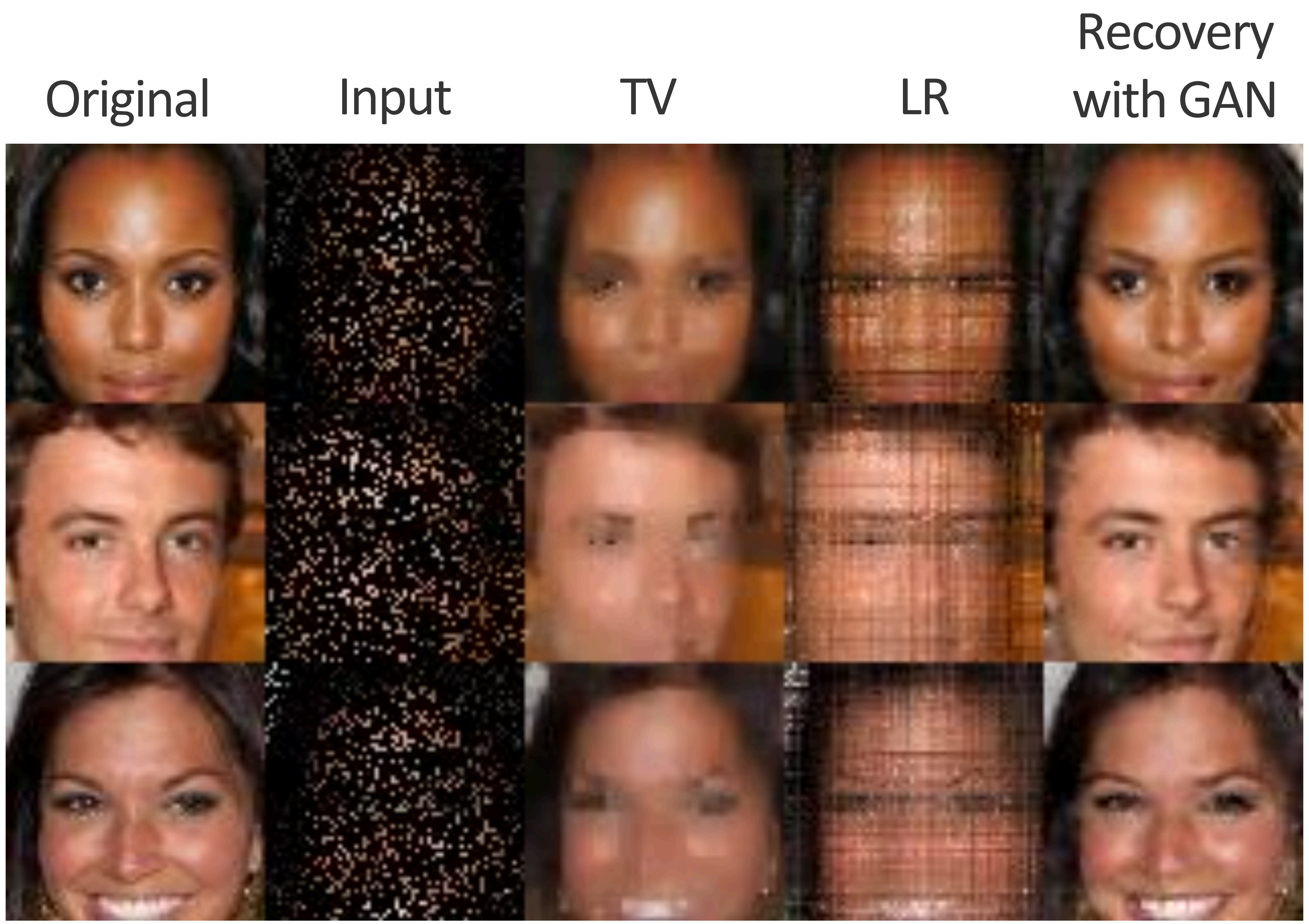


Figure 2 Comparisons with total variation (TV) and low rank (LR) based methods on input with random 80% missing.

Application
Missing data recovery

Isola, P., Zhu, J.Y., Zhou, T. and Efros, A.A., 2016. Image-to-image translation with conditional adversarial networks. arXiv preprint arXiv:1611.07004.
Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

SLIM

Image reconstruction

Why not leave the image reconstruction job to GAN?

- Less problems to deal with: finding the "closest" mapping

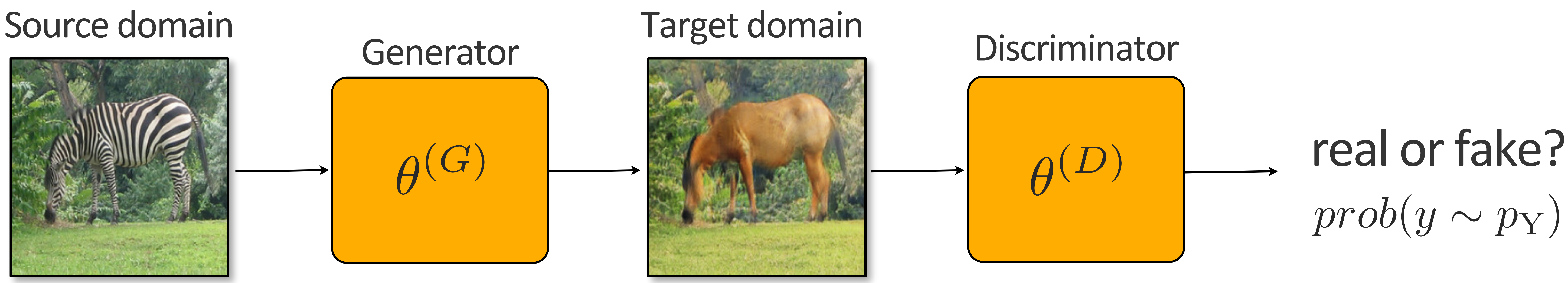
Idea: use a generator which gets an **image as input** instead of random vector

$$\min_{\theta^{(G)}} \max_{\theta^{(D)}} \left\{ \mathop{\mathbb{E}}_{y \sim p_{\mathrm{Y}}(y)} [\log D(y)] + \mathop{\mathbb{E}}_{x \sim p_X(x)} [\log(1 - D(G(x)))] \right\}$$

Target **image** domain

Source **image** domain

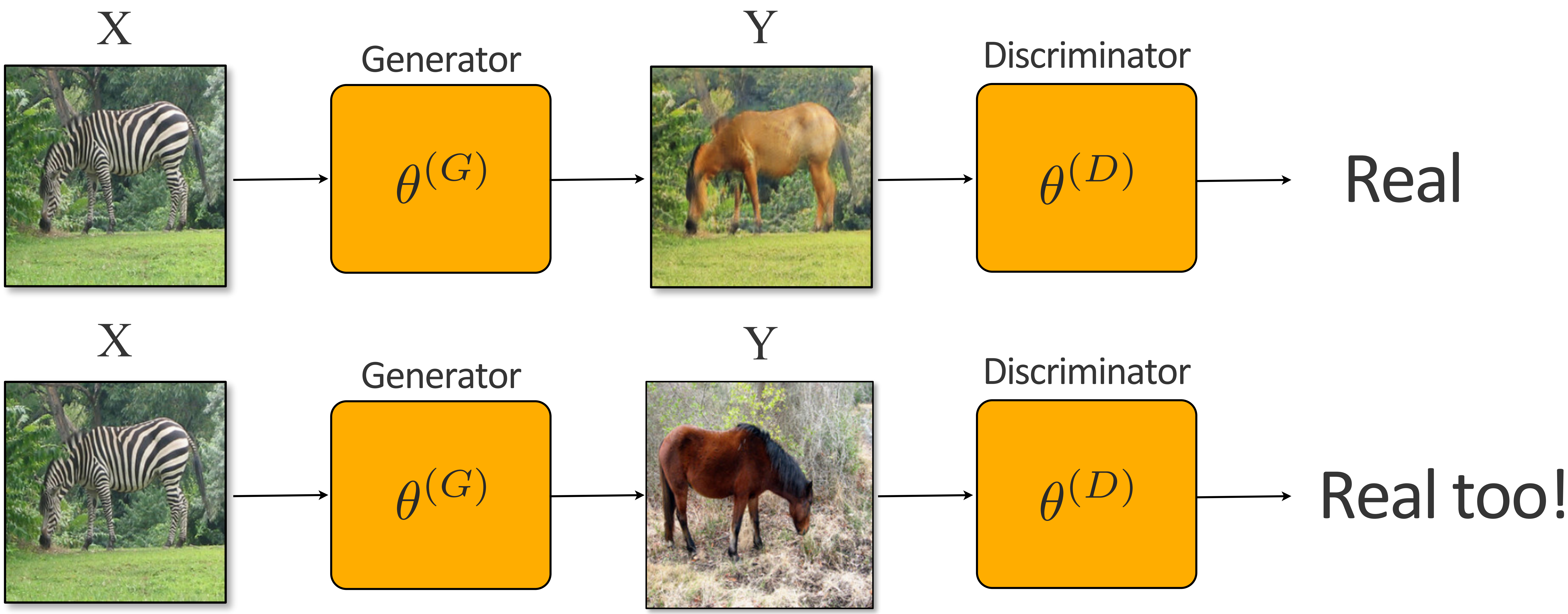
mapping from X to Y



Generator: generate fake images that can fool discriminator

Discriminator: classify fake samples vs. real samples

There are infinitely many mappings

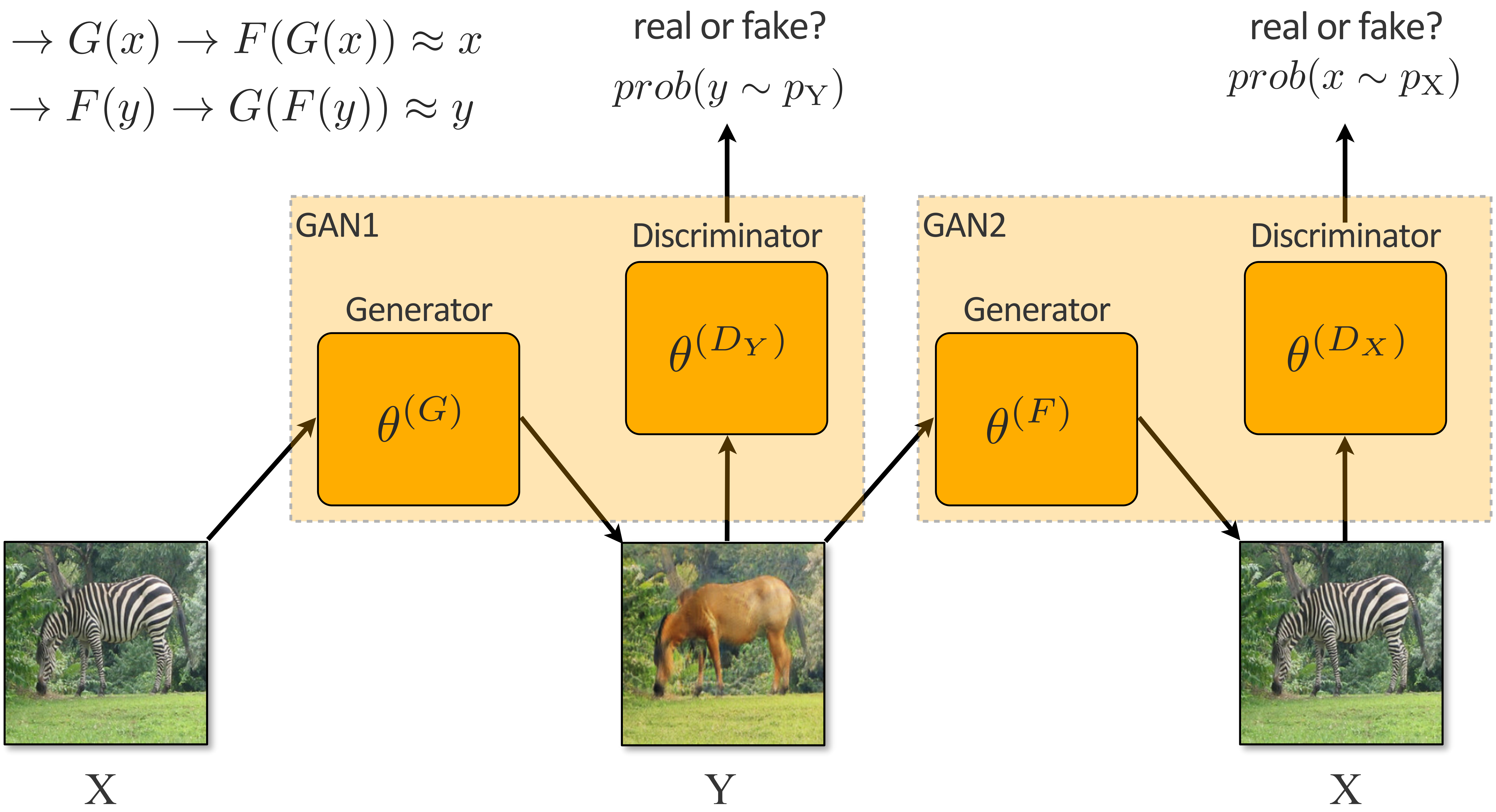


But we need a meaningful mapping! How?

Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593.

Cycle consistency

$$x \rightarrow G(x) \rightarrow F(G(x)) \approx x$$
$$y \rightarrow F(y) \rightarrow G(F(y)) \approx y$$

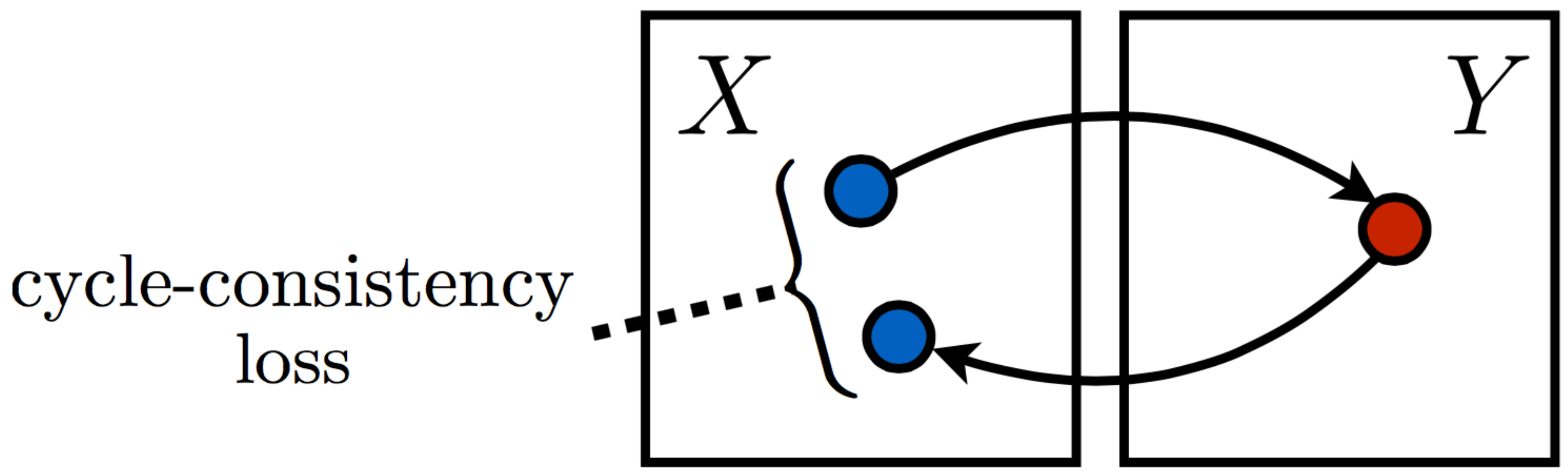


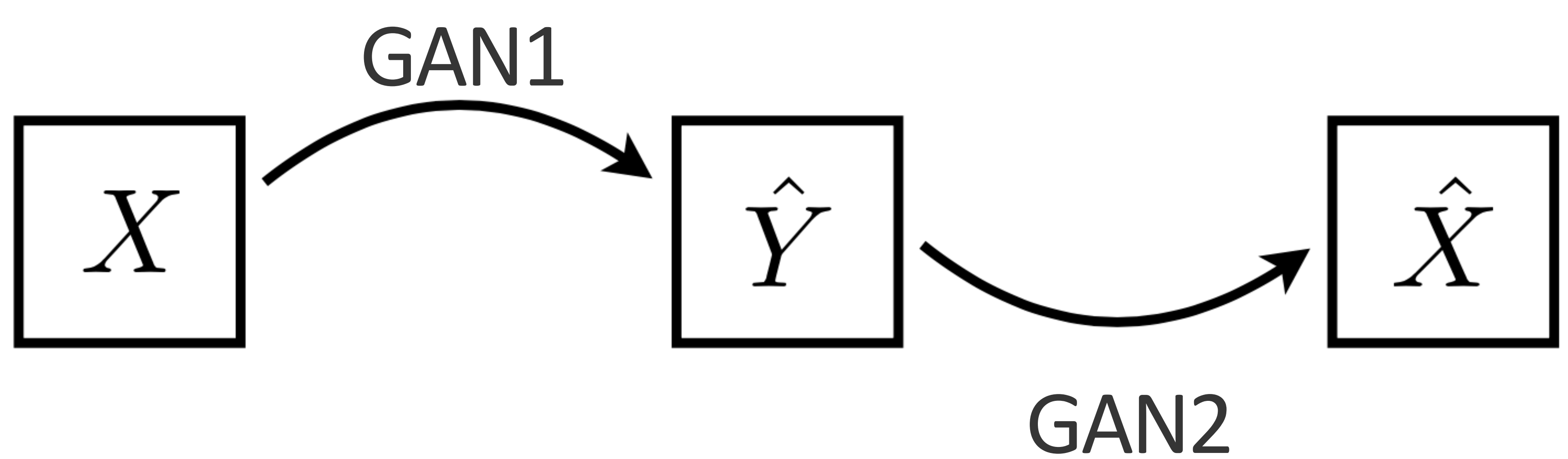
Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

Getting a meaningful mapping

Cycle consistency: using two GANs, map from source domain to target domain and back again and arrive where started

- i.e. make sure we can invert the transform.





Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

SLIM

CycleGAN math

GAN1 Loss: $$L_G(\theta^{(G)}, \theta^{(D_Y)}) = \underset{y \sim p_Y(y)}{\mathbb{E}} [\log D_Y(y)] + \underset{x \sim p_X(x)}{\mathbb{E}} [\log(1 - D_Y(G(x)))]$$

Mapping X to Y

Target domain probability distribution

Source domain probability distribution

GAN2 Loss: $$L_F(\theta^{(F)}, \theta^{(D_X)}) = \underset{x \sim p_X(x)}{\mathbb{E}} [\log D_X(x)] + \underset{y \sim p_Y(y)}{\mathbb{E}} [\log(1 - D_X(G(y)))]$$

Mapping Y to X

Target domain probability distribution

Source domain probability distribution

Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

SLIM

CycleGAN math

Cycle consistency Loss:

$$L_{cycle}(\theta^{(G)}, \theta^{(F)}) = \underset{x \sim p_X(x)}{\mathbb{E}} \underbrace{[\|F(G(x)) - x\|]}_{\text{forcing } F(G(x)) \approx x} + \underset{y \sim p_Y(y)}{\mathbb{E}} \underbrace{[\|G(F(y)) - y\|]}_{\text{forcing } G(F(y)) \approx y}$$

CycleGAN game:

$$\min_{\theta^{(G)}, \theta^{(F)}} \max_{\theta^{(D_X)}, \theta^{(D_Y)}} \{L_G + L_F + \lambda L_{cycle}\}$$

$\lambda = 10$: a hyper-parameter

Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

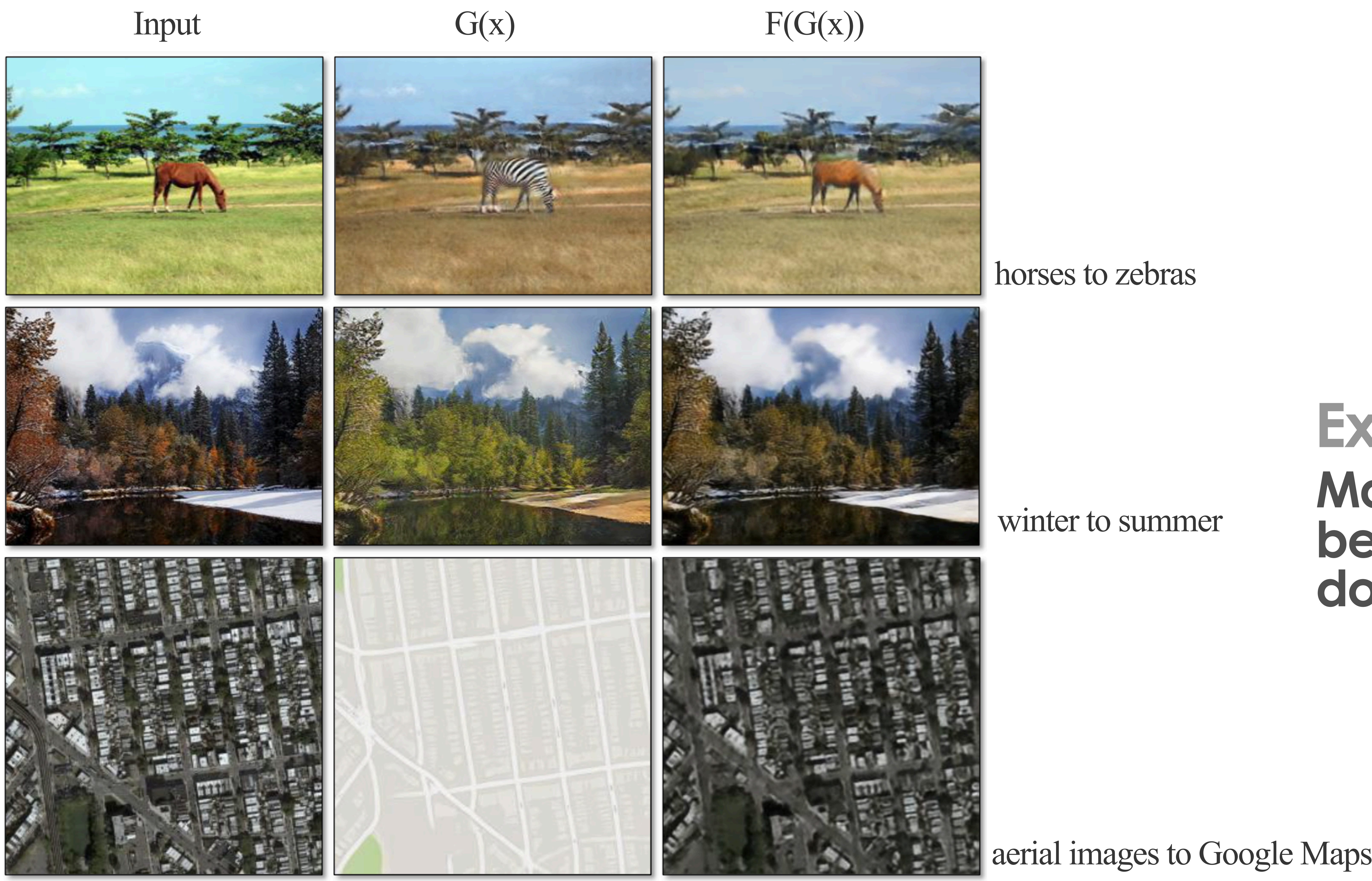


Figure 7 Mapping between two image spaces

Example

Mapping done between image domains by CycleGAN

Domain definition

We investigate three cases and train a separate network for each task:

1. **Source domain (X)**: freq. slices with missing data in a box
2. **Source domain (X)**: freq. slices with missing columns (receiver x location)
3. **Source domain (X)**: freq. slices with randomly missing pixels (random receivers)

Target domain (Y): complete freq. slices with no missing data

Training

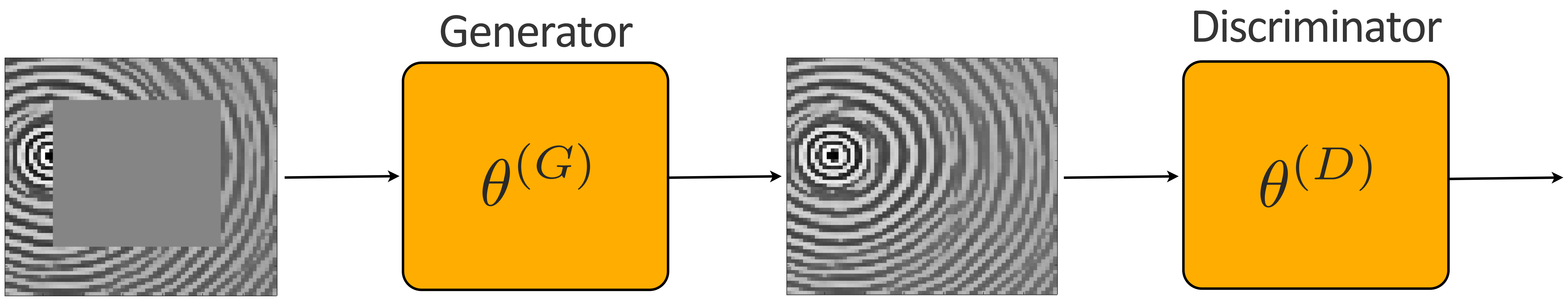
The CycleGAN is trained on freq. slices with no missing entries and set of corrupted images:

- Number of training freq. slices in each set: 5000
- Total number of freq. slices in data set: 160801
- Algorithm is tested on not used freq. slices (randomly picked).

Training is done using one node and 20 threads for 24 hours.

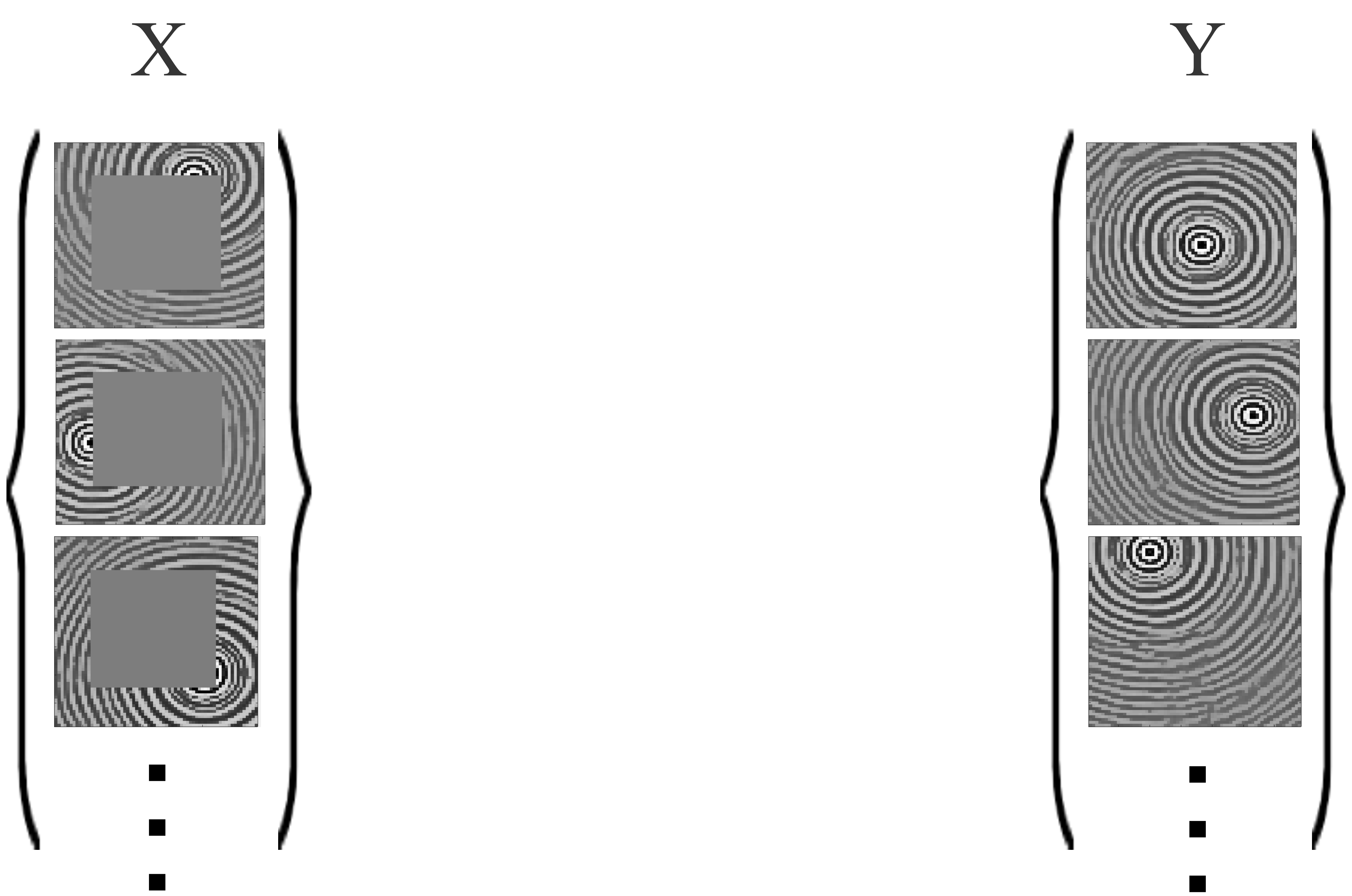
In test time, we fill in the missing values using pixels in the mapped freq. slice.

Seismic application - Testing stage



real or fake?

$prob(y \sim p_{\mathrm{Y}})$



- This example is showing the X domain for case which we miss a box in data

Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv: 1703.10593.

SLIM

Scalability

The are 68 sources in x and y direction

- size of freq. slices is 68x68

Is it scalable?

Based on the reference paper of CycleGAN:

- “patch-level discriminator architecture has fewer parameters than a full-image discriminator, and can be applied to arbitrarily-sized images”

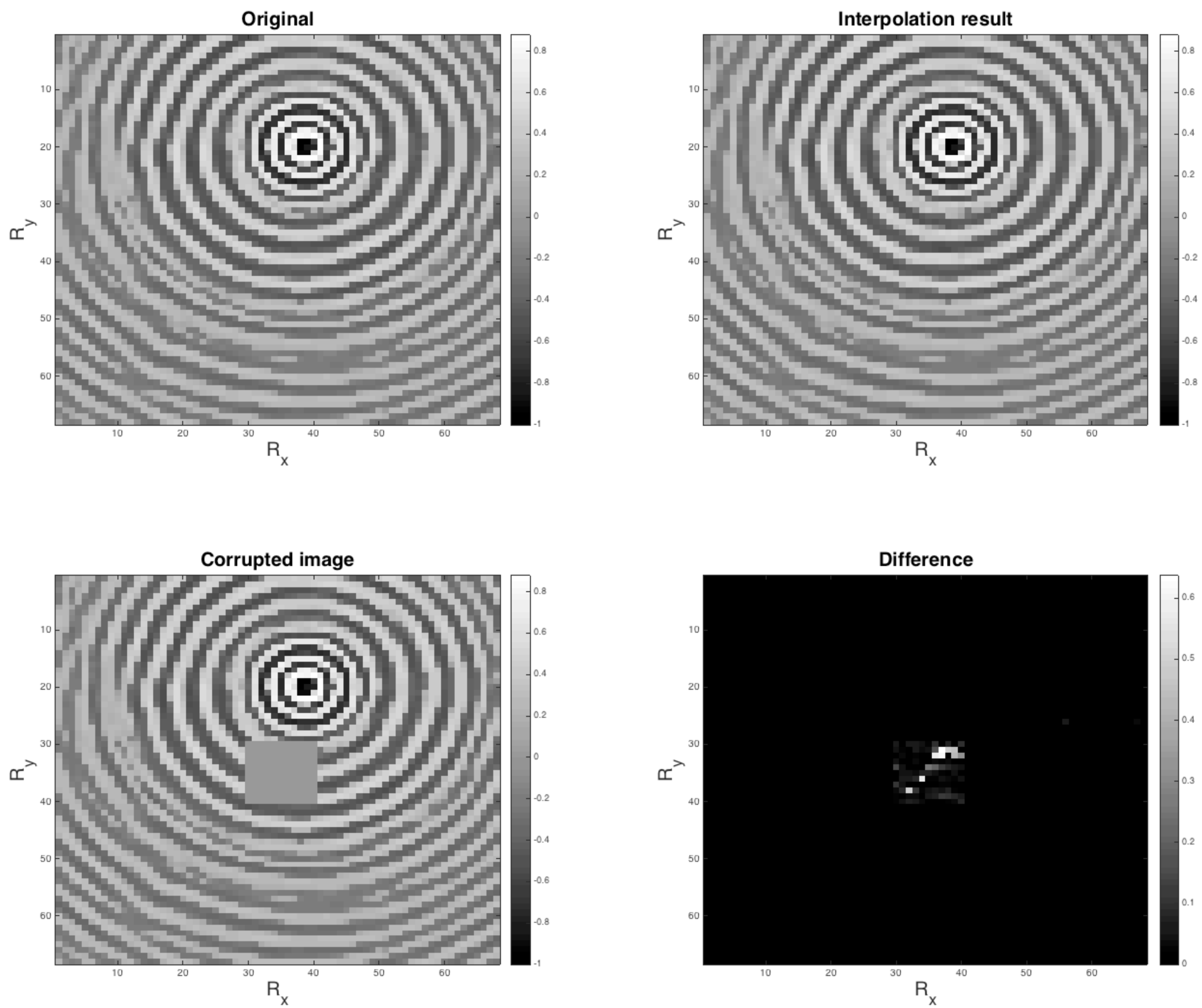


Figure 8 Result

Results

Values missing in a box

missing box: 14x14

Synthetic 3D BG model
68 x 68 sources
401 x 401 receivers
Data at 7.43 Hz

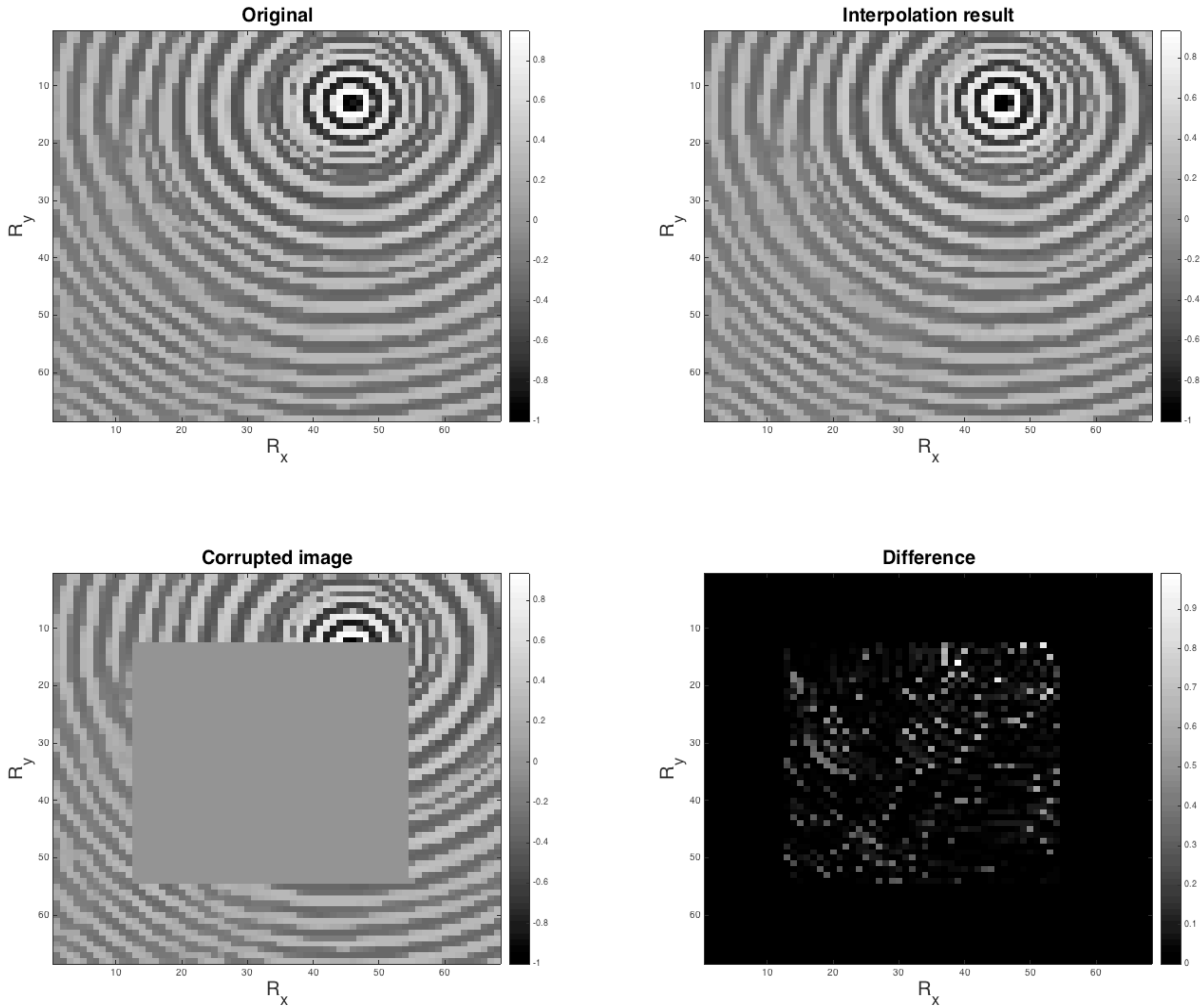


Figure 9 Result

Results
Values missing in a box

missing box: 42x42

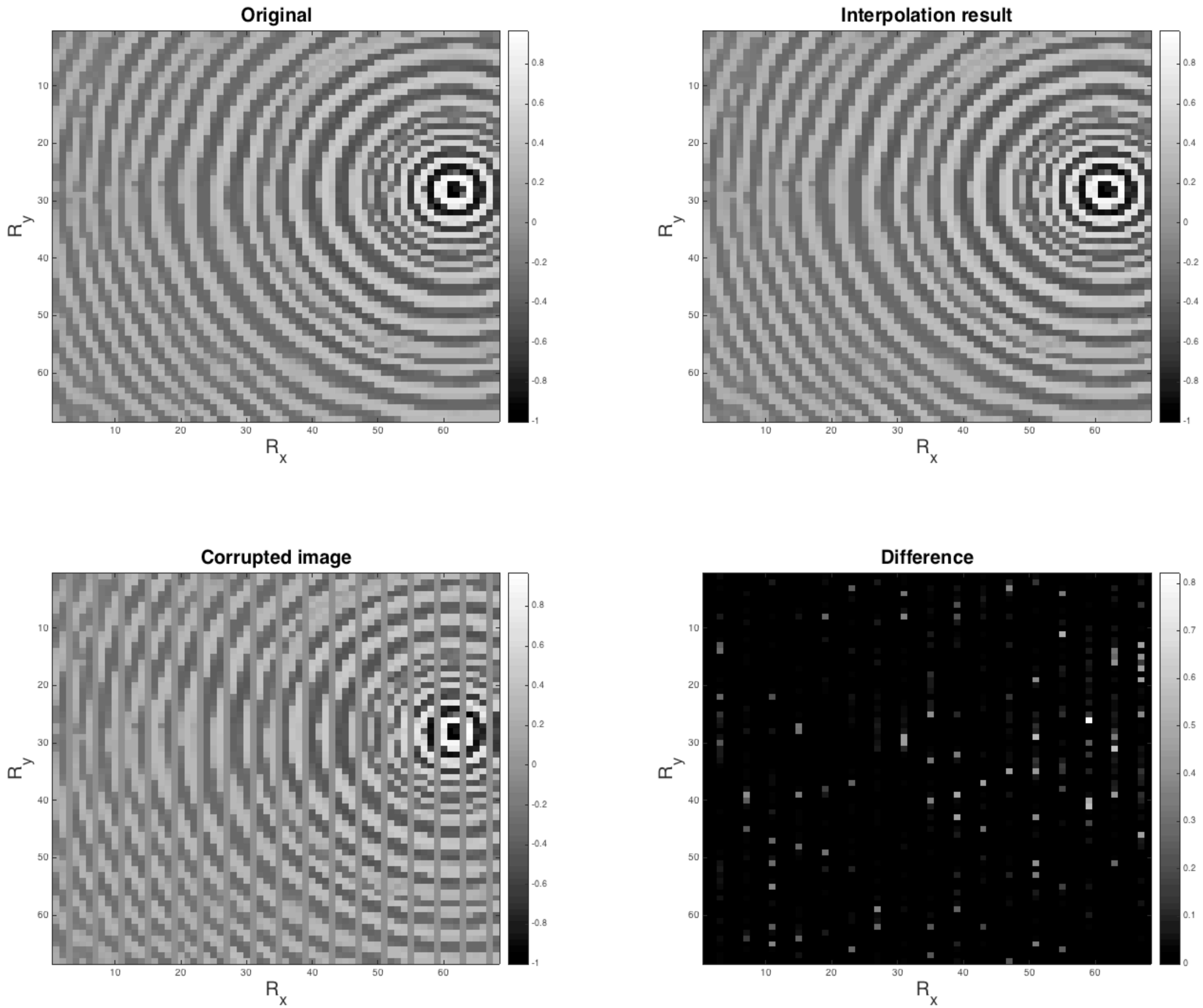


Figure 10 Result

Results

Values missing in columns

missing every 4th column

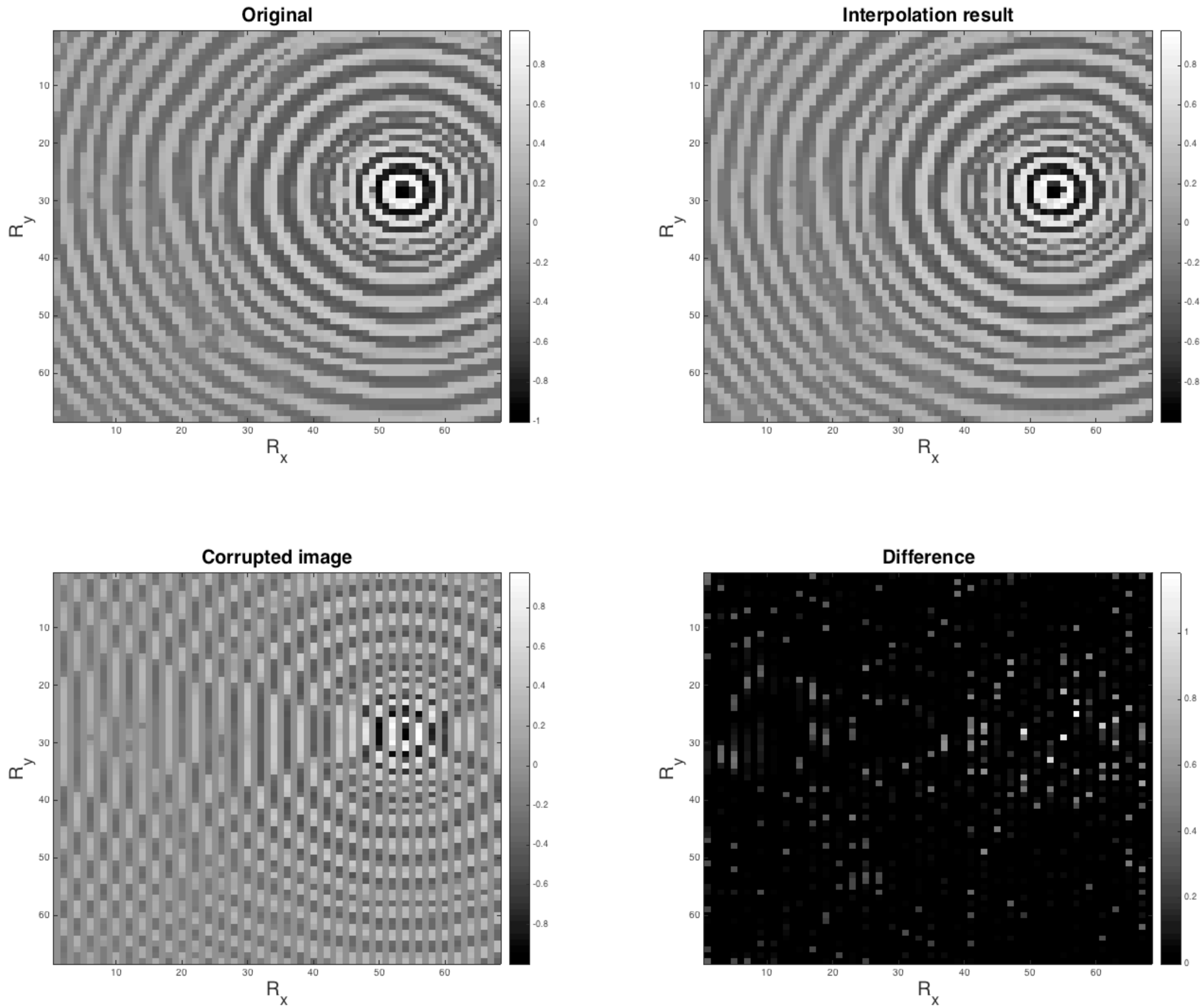


Figure 11 Result

Results

Values missing in columns

missing half of columns

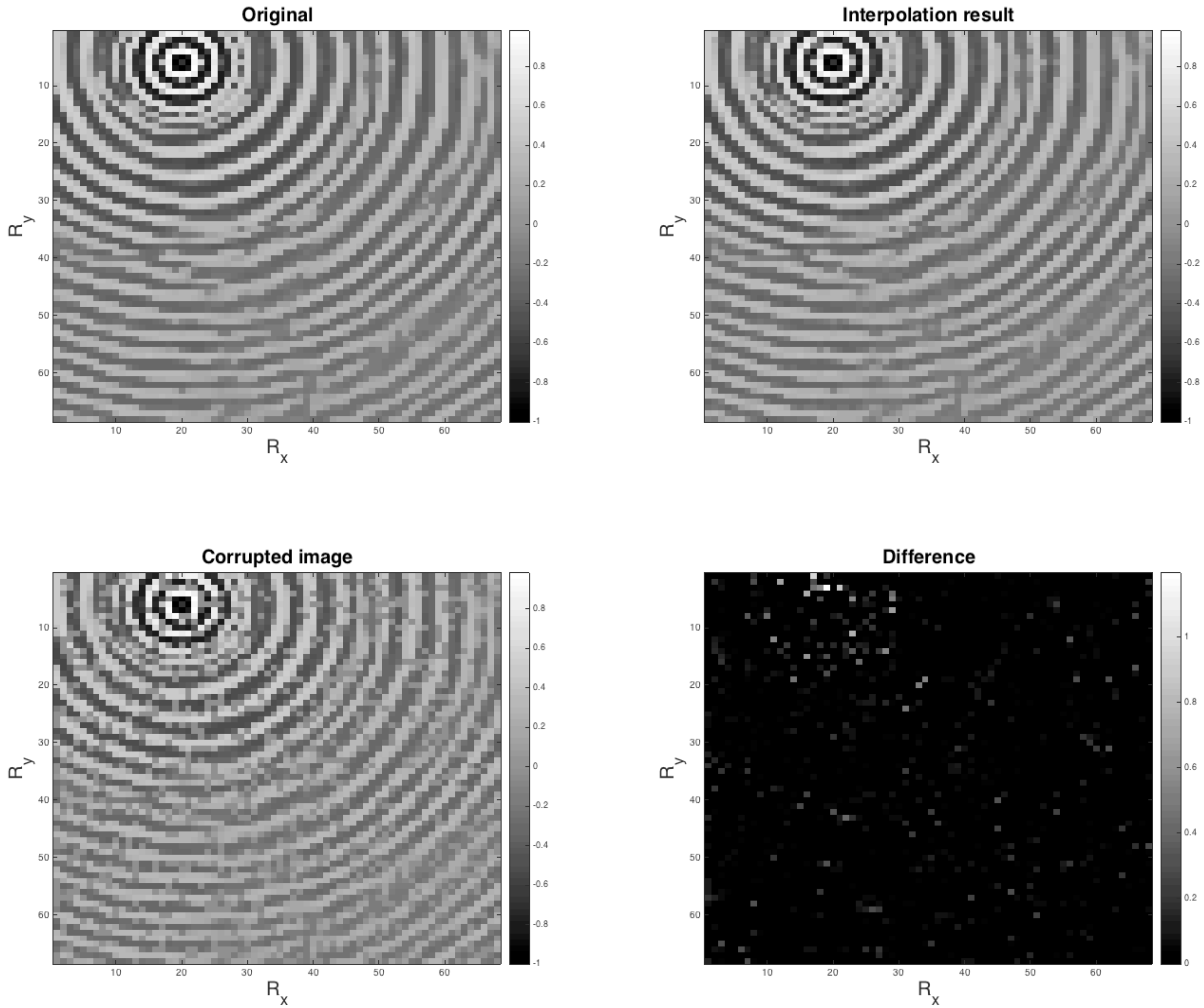


Figure 12 Result

Results

Values missing randomly

20% missing samples

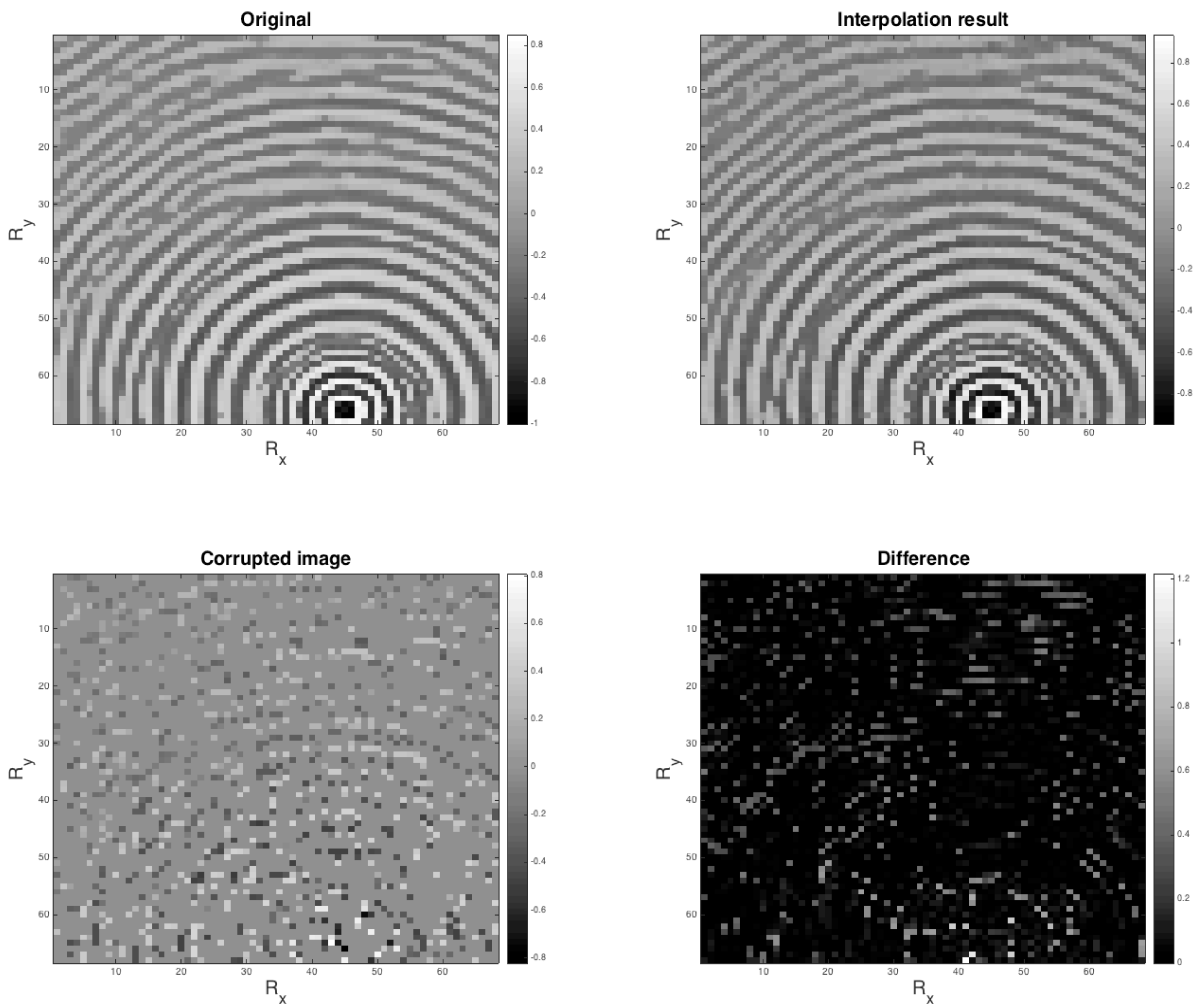


Figure 13 Result

Results

Values missing randomly

80% missing samples

SLIM

Future work

- Train a single network which can take care of all sort of different missing values
- map from domain of data without multiples to domain of data with multiples, i.e. **multiple prediction**
- map from domain of acoustic data to domain of elastic data, i.e. **elastic forward modeling**

Acknowledgements

This research was carried out as part of the SINBAD project with the support of the member organizations of the SINBAD Consortium.

Thank you for your attention.