MULTI-RESOLUTION DATA FUSION FOR SUPER RESOLUTION OF MICROSCOPY IMAGES

by

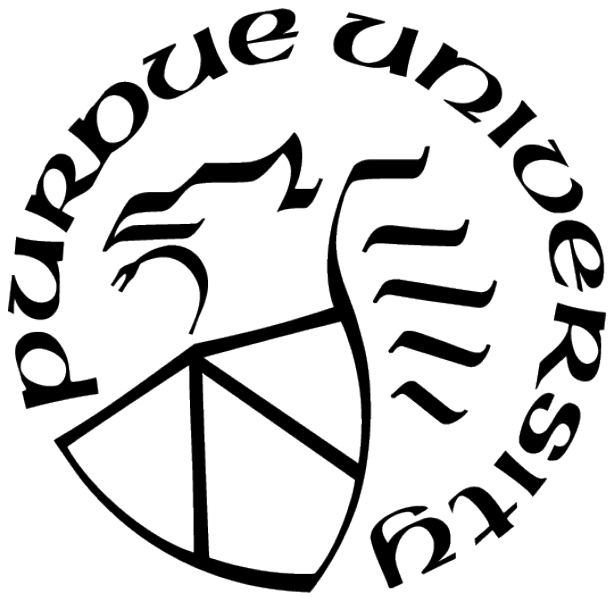
Emma Reid

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy



Department of Mathematics

West Lafayette, Indiana

August 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Gregery Buzzard, Co-Chair

College of Science

Dr. Charles Bouman, Co-Chair

School of Electrical and Computer Engineering

Dr. Aaron Yip

College of Science

Dr. Lawrence Drummy

Air Force Research Laboratory

Approved by:

Dr. Plamen Stefanov

Thank you to my parents, friends, and advisors who always believed in me, even when I didn't believe in myself.

"We seek, not to know the answers, but to understand the questions." - Kung Fu

ACKNOWLEDGMENTS

Thank you to Dr. Gregery Buzzard, Dr. Charles Bouman, and Dr. Lawrence Drummy for their mentorship throughout my PhD process. Your contributions were invaluable and this work wouldn't exist without you.

Additionally I'd like to acknowledge Dr. Cheri Hampton for collecting the data used for our reconstructions and Dr. Asif Mehmood for his guidance on neural networks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ADMM	Alternating Direction Method of Multipliers
CNN	Convolutional Neural Network
DPSR	Deep Plug and Play Super Resolution
DPSRGAN	Deep Plug and Play Super Resolution Generative Adversarial Network
EM	Electron Microscopy
FoV	Field of View
GAN	Generative Adversarial Network
HR	High Resolution
iid	Independent and Identically Distributed
LR	Low Resolution
MACE	Multi-Agent Consensus Equilibrium
MBIR	Model-Based Image Reconstruction
MDF	Multi-resolution Data Fusion
NRMSE	Normalized Root Mean Square Error
PnP	Plug and Play
PSF	Point Spread Function
RAP	Relaxed Adjoint Projection
SEM	Scanning Electron Microscopy
SR	Super Resolution
TEM	Transmission Electron Microscopy

ABSTRACT

Applications in materials and biological imaging are currently limited by the ability to collect high-resolution data over large areas in practical amounts of time. One possible solution to this problem is to collect low-resolution data and apply a super-resolution interpolation algorithm to produce a high-resolution image. However, state-of-the-art super-resolution algorithms are typically designed for natural images, require aligned pairing of high and low-resolution training data for optimal performance, and do not directly incorporate a data-fidelity mechanism.

We present a Multi-Resolution Data Fusion (MDF) algorithm for accurate interpolation of low-resolution SEM and TEM data by factors of 4x and 8x. This MDF interpolation algorithm achieves these high rates of interpolation by first learning an accurate prior model denoiser for the TEM sample from small quantities of unpaired high-resolution data and then balancing this learned denoiser with a novel mismatched proximal map that maintains fidelity to measured data. The method is based on Multi-Agent Consensus Equilibrium (MACE), a generalization of the Plug-and-Play method, and allows for interpolation at arbitrary resolutions without retraining. We present electron microscopy results at 4x and 8x super resolution that exhibit reduced artifacts relative to existing methods while maintaining fidelity to acquired data and accurately resolving sub-pixel-scale features.

1. INTRODUCTION

The contents of this chapter appear in [1].

Many important material science problems require the collection of high resolution (HR) data over large fields of view (FoV). For example, high resolution images are needed to extract detailed features, such as the 4 nanometer curli fibers structures in *E. coli*, which are fundamental in the formation of bacterial biofilms, or the 10-20 nm structures in gold nanorods materials, which are of interest due to their near-infrared light tunability and biological inertness [2]. Also, a large FoV is typically required to collect the representative volumes (RV) of materials that are needed to determine macroscopic properties such as material toughness and fracture strength [3].

However, imaging multiple, large FoVs at high resolution is difficult under realistic constraints. For example, raster scanning a 1mm $\times$ 1mm FoV at a resolution of 10nm requires the acquisition of approximately 10 Giga-pixels of data, which requires roughly 17 hours under conditions described in [4]. In fact, the total of all electron microscopy (EM) data collected was estimated in 1996 as less than a cubic millimeter of actual material [5].

One approach to overcoming this barrier is to acquire a large FoV at low resolution (LR) and then to interpolate it to obtain a higher resolution image of sufficient quality. Ideally, 4x interpolation in each direction leads to a 16x decrease in acquisition time, while 8x interpolation leads to a 64x decrease.

Traditional interpolation methods such as splines [6] do not offer sufficient quality, but recent advances using deep neural networks (DNNs) have produced a number of methods for high quality interpolation of natural images. For example, [7] and [8] use end-to-end DNNs trained on HR/LR paired images. This approach is improved in SRGAN [9] through adversarial training and perceptual loss. Another approach is EnhanceNet [10], which uses automated texture synthesis and perceptual loss but focuses on creating realistic textures rather than reproducing ground truth images. ESRGAN [11] introduces architectural improvements to SRGAN, and ESRGAN+ [12] adds further refinements. DPSR [13] uses a form of Plug-and-Play as described later, but still requires a CNN super-resolver trained on

HR/LR pairs. And finally, DPSRGAN [13] attempts to fuse these Plug-and-Play methods with the realistic textures generated by GANs.

An impediment to applying these DNN methods to material imaging problems is that they are typically trained using a large set of aligned HR/LR patch pairs. However, in practice it is difficult to acquire large quantities of accurately aligned HR/LR data [14]. More recently, zero-shot learning has been proposed as a method that does not require aligned HR/LR data for training. In zero-shot learning, the LR image is further downsampled to produce paired data to train a small, image-specific CNN that is used to upsample the original [15]. While this allows for quick and accurate reconstructions, it does not make use of high-resolution data in its training.

Plug-and-play (PnP) [16] and Multi-Agent Consensus Equilibrium (MACE) [17] methods provide a framework in which the forward and prior models can be designed separately, so they do not require aligned HR/LR training data. In [18], the multi-resolution data fusion (MDF) architecture was proposed to integrate large quantities of LR image data with a relatively small amount of HR data. They used PnP with a library-based Non-Local Means (NLM) prior model to incorporate HR data, but this led to slow reconstruction times due to the computational cost of NLM [19]. A similar application of Plug-and-Play in [20] used the NCSR algorithm combined with sparse coding and dictionary learning to turn a denoiser into a super-resolver. However this method begins to break down in the presence of noise, which is quite prevalent in our low-resolution images. A related approach to data fusion in the MACE framework was recently used in [21] to combine CT and MRI modalities, but was applied at only a single resolution.

Here, we present a Multi-Resolution Data Fusion (MDF) algorithm for accurate 4x and 8x interpolation of large FoV low-resolution EM images using selected unpaired patches of HR data. The algorithm includes a forward model that promotes agreement with the acquired LR data along with a prior model that encourages similarity to the HR training data. Since the forward model has an analytical form, an advantage of our method is that it can perform interpolation by any factor without retraining of the prior model.

The novel contributions of this work include:

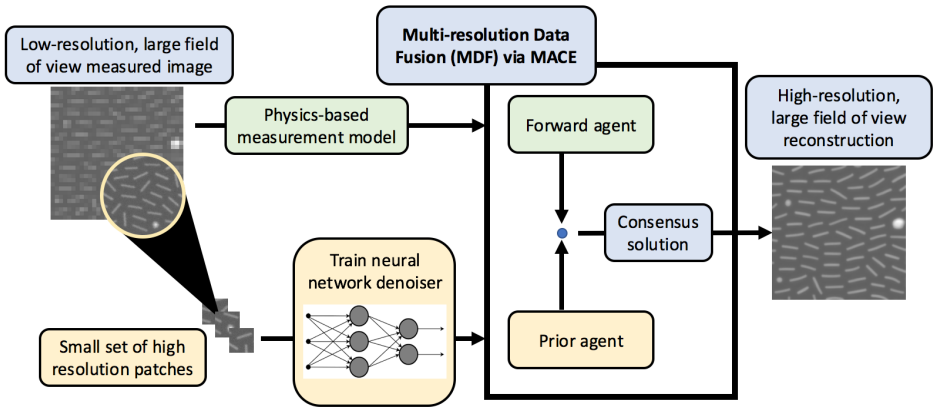


Figure 1.1. Overview of the Multi-resolution Data Fusion (MDF) pipeline. A small set of high-resolution data (unpaired with low-resolution data) is used to tune a CNN denoiser. This tuned denoiser is used in the Plug-and-Play algorithm with a microscopy-based forward model to produce high-resolution images from low-resolution data.

- An MDF framework for accurate interpolation of low-resolution data at multiple scales using a limited amount of unpaired HR data.
- The introduction of a relaxed adjoint projector (RAP), which can improve interpolation results by using a mis-matched back projector and is provably equivalent to using PnP with a modified prior.
- Experimental results indicating that interpolation factors of 4x to 8x are possible with realistic TEM and SEM data sets.

In Figure 1.1, we provide a visual representation of the MDF framework. Using a LR base image, we train a denoiser on a sparse set of HR patches of the same (or similar) specimen. This allows the denoiser to learn the underlying manifold of the HR data while simultaneously being trained to remove additive white Gaussian noise (AWGN). This denoiser is then applied in the MACE framework to achieve a super-resolution reconstruction of the low-resolution base image. This approach does not require registered pairs of HR and LR data, allowing for flexible levels of super resolution and simple generalization to other problems. Additionally it allows for use of known forward models and has a single parameter that can be used to control the weight of data fidelity relative to strength of regularization.

We structure the rest of this document as follows. In Chapter 2, we give background as to Model-Based Image Processing, the microscopy image acquisition process, and its application to our super-resolution problem and the MACE framework. In Chapter 3, we discuss initial experiments with reconstruction texture and quality. In Chapter 4, we detail the process of accomplishing Multi-Resolution Data Fusion and the theory behind Relaxed Adjoint Projection. In Chapter 5, we discuss comparison methods, our experimental framework, and present reconstructions at a variety of resolutions. We discuss these results further in Chapter 6 and detail future work. We additionally include appendices containing proofs and relevant code.

2. BACKGROUND

2.1 Model Based Image Processing

Given a physical system outputting measurements Y, we wish to reconstruct the original image X. The physical system has calibration parameters Φ as well as intrinsic characteristics that may be unknown. Using a chosen inversion method and smoothing parameters Θ, we can generate a reconstruction $\hat{X}$. As X is unknown, reconstructing $\hat{X}$ is clearly an inverse problem. This framework is shown in Figure 2.1.

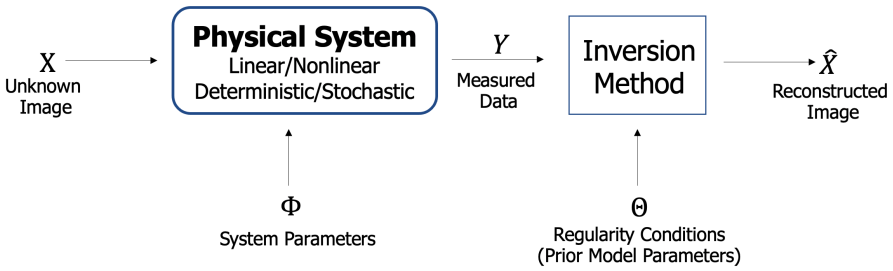


Figure 2.1. A diagram representing the framework of an inverse problem adapted from [22]

We may express the physical system and our regularity conditions as probability distributions. Here we denote $p(y|x)$ as our forward model and $p(x)$ as our prior model. These models are based on our assumptions of the physical system and the unknown image respectively. An example algorithm for reconstructing X is shown in Figure 2.2.

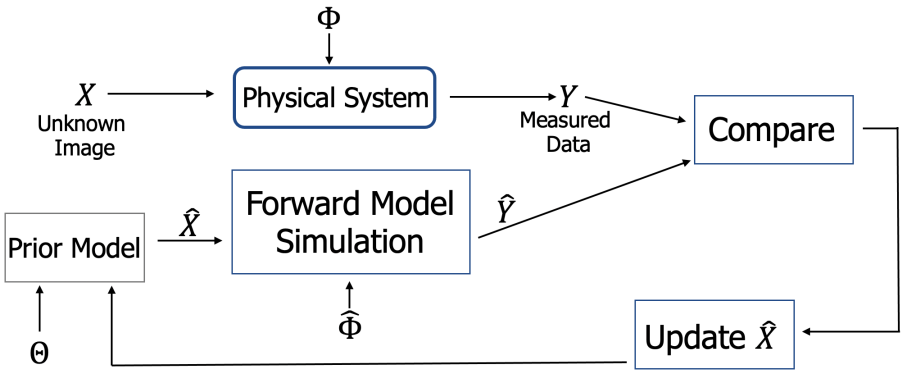


Figure 2.2. A diagram showing the process for reconstructing $\hat{X}$ adapted from [22]

Note that this algorithm is constantly balancing the weight of the forward and prior models, as we need a $\hat{X}$ that matches the measured data Y but also meets our assumptions on X [22]. Through iterative methods, we hope to obtain a solution $\hat{X}$ that meets both conditions.

2.1.1 Maximum a Posteriori Estimate

The Maximum a Posteriori estimate is defined as

$$\hat{X}_{MAP} = \text{argmax}_{x\in\Omega} p_{x|y}(x|Y) \tag{2.1}$$

Noting that the natural logarithm is an increasing function, the argmax will be equivalent for each. This gives

$$\hat{X}_{MAP} = \text{argmax}_{x\in\Omega} \log(p_{x|y}(x|Y)) \tag{2.2}$$

By Bayes Rule, we know that $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$. Substitution of this into 2.2 gives

$$\hat{X}_{MAP} = \text{argmax}_{x\in\Omega} \log(\frac{p_{y|x}(y|x)p_x(x)}{p_y(y)}) = \text{argmax}_{x\in\Omega} \log(p_{y|x}(y|x)) + log(p_x(x)) - log(p_y(y)) \tag{2.3}$$

We may then drop the $\log(p(y))$ term as it doesn't depend on x and multiply by a negative to convert the problem from an argmax to that of an argmin, or

$$\hat{X}_{MAP} = \text{argmin}_{x\in\Omega} - \log(p_{y|x}(y|x)) - \log(p_x(x)) \tag{2.4}$$

We may define $l(x) = -\log(p_{y|x}(y|x))$ and $\beta s(x) = -\log(p_x(x))$ for β a regularizing parameter. This then gives that

$$\hat{X}_{MAP} = \text{argmin}_{x\in\Omega} l(x) + \beta s(x)) \tag{2.5}$$

The equation in 2.5 is simpler but still difficult to minimize in its current form.

2.1.2 Alternating Directions Method of Multipliers (ADMM)

The Alternating Directions Method of Multipliers takes a problem like 2.5 and simplifies it further by employing variable splitting. Namely we consider the following constrained optimization problem

$$\hat{X} = \text{argmin}_{x,v\in\Omega} l(x) + \beta s(v)) s.t.\ x = v \tag{2.6}$$

and construct the scaled augmented Lagrangian below to enforce that $x = v$ and that our solution $\hat{X}$ matches both the forward and prior models [23], [16].

$$L_\lambda(x, v; u) = l(x) + \beta s(v) + \frac{1}{2\sigma_\lambda^2}\|x - v + u\|_2^2 - \frac{\|u\|_2^2}{2\sigma_\lambda^2} \tag{2.7}$$

Here u represents the distance between x and v. Iterations through ADMM are of the form

$$\hat{x} \leftarrow \text{argmin}_x L_\lambda(x, \hat{v}; u) \tag{2.8}$$

$$\hat{v} \leftarrow \text{argmin}_v L_\lambda(\hat{x}, v; u) \tag{2.9}$$

$$u \leftarrow u + \hat{x} - \hat{v} \tag{2.10}$$

where $\hat{v}$ can be initialized as a baseline reconstruction and u is initialized to be 0.

The ADMM iterates converge if the functions l and s are closed, proper, and convex on Ω and Φ respectively and there exists a saddle point (x^*, v^*, λ^*) such that for all $x \in \Omega, v \in \Phi$, $\lambda \in \mathbb{R}^K$,

$$L(x^*, v^*, \lambda) \leq L(x^*, v^*, \lambda^*) \leq L(x, v; \lambda^*) \tag{2.11}$$

as shown in [23] [22].

2.1.3 Plug & Play

The Plug & Play framework takes ADMM one step further by constructing explicit functions F and H where F is an inversion operator and H is a denoiser. Let $\tilde{x} = v - u$ and $\tilde{v} = x + u$. Define the following operators

$$F(\tilde{x}; \sigma_\lambda) = argmin_{x \in \mathbb{R}^N} \{l(x) + \frac{\|x - \tilde{x}\|_2^2}{2\sigma_\lambda^2}\} \tag{2.12}$$

$$H(\tilde{v}; \sigma_n) = argmin_{v \in \mathbb{R}^N} \{s(v) + \frac{\|v - \tilde{v}\|_2^2}{2\sigma_n^2}\} \tag{2.13}$$

where $\sigma_n = \sqrt{\beta}\sigma_\lambda$ is the assumed standard deviation of noise for the denoiser H [18]. This gives rise to the Plug & Play algorithm below.

Algorithm 1: Plug & Play Algorithm

Input: Initial Reconstruction $\hat{v}$
Output: Final Reconstruction x^*
$u \leftarrow 0$
while *unconverged* **do**
  $\tilde{x} \leftarrow \hat{v} - u$
  $\hat{x} \leftarrow F(\tilde{x}; \sigma_\lambda)$
  $\tilde{v} \leftarrow \hat{x} + u$
  $\hat{v} \leftarrow H(\tilde{v}; \sigma_n)$
  $u \leftarrow u + (\hat{x} - \hat{v})$
end

Note at convergence that $\hat{x} = \hat{v}$, so it doesn't ultimately matter which is returned. The Plug & Play framework is especially powerful here as it begins with an algorithm for function minimization and replaces components with direct algorithmic input-output maps. However by replacing those components, we eliminate the minimization problem and have no clear replacement problem that's being solved by the new algorithm. This is solved by the generalization to 2.3.

2.2 Problem Formulation

The contents of this section appear in [1].

Our goal is to interpolate a rasterized image $y \in \mathbb{R}^M$ to a HR version $x \in \mathbb{R}^N$. Super-resolution by a factor of L implies scaling by L in each direction, so that $N = L^2M$. For such a problem, the forward model is typically

$$y = \Psi x + \epsilon, \tag{2.14}$$

where $\Psi \in \mathbb{R}^{M \times N}$ represents the point spread function of the microscope and $\epsilon \sim N(0, \sigma_w^2)$ is an M dimensional vector of AWGN. The data-fitting cost function is then $\frac{1}{2}\|y - \Psi x\|^2$, which will be embedded in a proximal map and balanced by the action of a prior agent in the MACE formulation described below.

2.2.1 Microscopy Forward Model

In bright-field transmission electron microscopy, a parallel beam of electrons illuminates a thin material sample, and the resulting transmitted beam is formed into an image using the microscope's objective lens. This is then further magnified using the microscope's projection lens system. Using this configuration, the TEM can yield a magnification ranging from 1,000X to over 1,000,000X. The magnified beam is sampled at the image plane using a pixel array detector such as a direct electron detector or CCD camera. Since the electron-optical magnification can be controlled and the image is detected using a pixel array detector, a block-averaging forward model is a good approximation for this acquisition modality. In this forward model, a square region in the high-resolution image is averaged to produce a single pixel value in the low-resolution image.

For scanning electron microscopy, an electron beam is focused to a small diameter probe which is then raster scanned across the surface of the sample using beam deflectors. As the probe strikes the sample, secondary electrons are ejected from the sample and collected with an integrating detector, which sums the total number of electrons scattered at each point on the surface of the sample. The raster array dimensions can be controlled to give LR and HR data, so as in the TEM case a block-averaging forward model is a good approximation to the imaging system. A diagram representing the image acquisition process for TEM and SEM is shown in Figure 2.3.

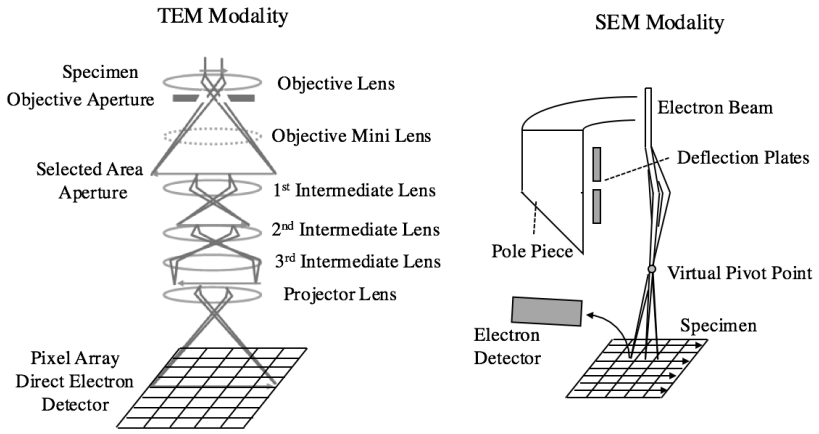


Figure 2.3. Illustration of the TEM and SEM image acquisition process.

With this in mind, we can approximate Ψ for super resolution by a factor of L by block-averaging every non-overlapping neighborhood of $L \times L$ pixels in x [18]. For notational convenience, we let A represent summation over $L \times L$ blocks, in which case $\Psi = A/L^2$ in (2.14). Also, A^t is given by replicating each pixel into an $L \times L$ block, so $AA^t = L^2 I$. The negative log-likelihood is then

$$l(x) = \frac{1}{2\sigma_w^2} \left\| y - \frac{1}{L^2} Ax \right\|^2 + \frac{M}{2} \log(2\pi\sigma_w^2). \tag{2.15}$$

As part of the Plug-and-Play algorithm, Sreehari et al. used the proximal map of $l(x)$ with an additional constraint to ensure nonnegativity:

$$F(x; \sigma_\lambda) = \underset{\hat{x} \geq 0}{\operatorname{argmin}} \left[\frac{1}{2\sigma_w^2} \left\| y - \frac{1}{L^2} A\hat{x} \right\|_2^2 + \frac{1}{2\sigma_\lambda^2} ||x - \hat{x}||_2^2 \right] \tag{2.16}$$

They took $\sigma_w^2 \to 0$ to obtain the proximal map for the noiseless case as

$$F(x; \sigma_\lambda) = \left[x + A^t \left(y - \frac{1}{L^2} Ax \right) \right]_+ \tag{2.17}$$

where $[\,\cdot\,]_+$ enforces positivity [18] and A^t is the adjoint or back projection operator.

When σ_w^2 is positive, this proximal map is

$$F(x;\sigma_\lambda) = \left[x + \frac{\sigma_\lambda^2}{\sigma_\lambda^2 + L^2\sigma_w^2} A^t \left(y - \frac{1}{L^2} Ax\right)\right]_+ \tag{2.18}$$

The block replication inherent in A^t can lead to blocky artifacts in the final reconstruction, which motivates our introduction of the Relaxed Adjoint Projector.

2.3 Multi-Agent Consensus Equilibrium Reconstruction Framework

The contents of this section appear in [1].

Previous related work used the forward model of Section 2.2.1 in the Plug & Play algorithm with a standard Gaussian denoiser as the prior model [16] [18]. Plug & Play has also been used with deep neural network denoisers serving as prior models [24] [25] [26]. However, limitations of this previous work include little ability to control regularization strength and the use of generically trained neural networks as opposed to domain-specific denoisers.

This leads to the use of the Multi-Agent Consensus Equilibrium (MACE) framework [17]. The MACE framework provides a problem formulation that is consistent with Plug & Play but that extends it by allowing for multiple forward and prior models and by providing parametric control of regularization. The MACE framework also gives us the means to formulate and understand the Relaxed Adjoint Projection as either a modification to the update operators in Plug & Play or a modification to the averaging operator.

The motivating problem for MACE is to minimize the function

$$f(x) = \sum_{\mathrm{i}=1}^{K} \mu_\mathrm{i} f_\mathrm{i}(x_\mathrm{i}) \text{ s.t. } x_\mathrm{i} = x,\ \mathrm{i} = 1, \ldots, K, \tag{2.19}$$

with $x, x_\mathrm{i} \in \mathbb{R}^N$ and weights $\mu_\mathrm{i} > 0$ with $\sum_{\mathrm{i}=1}^{K} \mu_\mathrm{i} = 1$.

By analyzing the solution to this problem, Buzzard et al. [17] proposed a framework that encompasses the problem in (2.19) and that generalizes to include algorithmic priors and forward maps. For K vector valued maps, $F_\mathrm{i} : \mathbb{R}^N \to \mathbb{R}^N$, $\mathrm{i} = 1, \ldots, K$, define the consensus equilibrium for these maps to be any solution $(x^*, \mathbf{u}^*) \in \mathbb{R}^N \times \mathbb{R}^{NK}$ such that

$$F_\mathrm{i}(x^* + \mathbf{u}_\mathrm{i}^*) = x^*,\ \mathrm{i} = 1, \ldots K \tag{2.20}$$

$$\overline{\mathbf{u}}_\mu^* = 0 \tag{2.21}$$

where $\mathbf{u}$ is a vector in $\mathbb{R}^{NK}$ constructed by stacking vectors $u_1, \dots u_K$, and $\overline{\mathbf{u}}_\mu$ is the weighted average $\overline{\mathbf{u}}_\mu = \sum_{\mathrm{i}=1}^{K} \mu_\mathrm{i} u_\mathrm{i}$. In the case of (2.19), the maps F_i are chosen to be proximal maps associated with the f_i, but the MACE framework extends this to more general operators.

The conditions in (2.20) and (2.21) are equivalent to a related system of equations. Namely for $\mathbf{v} \in \mathbb{R}^{NK}$ with $\mathbf{v} = (v_1^T, \dots, v_K^T)$, $v_\mathrm{i} \in \mathbb{R}^N \ \forall \mathrm{i}$, define $\mathbf{F}$, $\mathbf{G}_\mu : \mathbb{R}^{NK} \to \mathbb{R}^{NK}$ by

$$\mathbf{F}(\mathbf{v}) = \begin{pmatrix} F_1(v_1) \\ \vdots \\ F_K(v_K) \end{pmatrix} \text{ and } \mathbf{G}_\mu(\mathbf{v}) = \begin{pmatrix} \bar{\mathbf{v}}_\mu \\ \vdots \\ \bar{\mathbf{v}}_\mu \end{pmatrix}. \tag{2.22}$$

Here $\overline{\mathbf{v}}_\mu = \sum_{\mathrm{i}=1}^{K} \mu_\mathrm{i} v_\mathrm{i}$ and $\mathbf{G}_\mu$ copies this weighted average into each entry of the vector. Then $(x^*, \mathbf{u}^*)$ solves (2.20) and (2.21) if and only if $\mathbf{v}^* = \hat{x}^* + \mathbf{u}^*$ satisfies $\overline{\mathbf{v}}_\mu^* = x^*$ and

$$\mathbf{F}(\mathbf{v}^*) = \mathbf{G}_\mu(\mathbf{v}^*). \tag{2.23}$$

As in Plug & Play, the F_i may be replaced by more general operators such as denoisers, in which case the solution of (2.23) is the fixed point of a generalized Plug & Play algorithm. Majee et al. [27] provide a reformulation of the algorithm used to find MACE solutions; we use this approach here and describe it in Algorithm 1 below.

Here, we use two types of agents. One type is a map incorporating the forward model of the imaging system and is designed to promote fidelity to data. The other type is a set of denoisers trained to remove additive white Gaussian noise (AWGN) of various standard deviations and structure from the image. We used the DnCNN archictecture [28] trained to remove 10% AWGN as our denoising prior. This prior is further described in 4.1.3. We chose the weights associated with the averaging operator in (2.22) to satisfy

$$\bar{x} = \mu x_{K+1} + \frac{1-\mu}{K} \sum_{k=1}^{K} x_k, \tag{2.24}$$

where $\mu \in (0, 1)$ represents the weighting of the forward model and can be used to adjust the relative importance of data versus regularization, with larger μ giving more weight to data.

This leads to the following algorithmic framework, adapted from [17] and [27], shown in Algorithm 1.

Algorithm 2: MACE Framework for Data Fusion

1 **Input:** Initial Reconstruction $x^{(0)} \in \mathbb{R}^N$
2 **Output:** Final Reconstruction x^*
3 $\mathbf{x} \leftarrow \mathbf{v} \leftarrow \begin{pmatrix} x^{(0)} \\ \vdots \\ x^{(0)} \end{pmatrix}$
4 **while** *unconverged* **do**
5 $\quad \mathbf{x} \leftarrow \mathbf{F}(\mathbf{v})$
6 $\quad \mathbf{z} \leftarrow \mathbf{G}(2\mathbf{x} - \mathbf{v})$
7 $\quad \mathbf{v} \leftarrow \mathbf{v} + 2\rho(\mathbf{z} - \mathbf{x})$
8 **end**
9 $x^* \leftarrow \overline{\mathbf{v}}_\mu$

Here ρ is a user parameter used to control the speed of convergence, which we set to be $\rho = 0.5$. While our method differs from Majee et al. in Line 9, at convergence, x_1 and $\overline{\mathbf{v}}_\mu$ are equal. We chose to adopt $\overline{\mathbf{v}}_\mu$ in cases of stopping pre-convergence.

Using the MACE framework, the goal now is to reconstruct a HR, wide-field image given a LR, wide-field image of the same region and a limited set of HR data patches (which may be disjoint from the area in the LR image).

3. FORWARD MODEL COMPENSATION

Initially we noticed texture artifacts in MACE reconstructions and began to brainstorm ideas to improve our methods. We applied 2 ideas in parallel, that of a perceptual loss function and the idea of Forward Model Compensation (FMC).

Perceptual loss is based on the idea that small differences in pixel value can create a large difference from a visual perspective. Comparing high level features allows for higher quality reconstructions. Typically perceptual loss is implemented by passing 2 images through the VGG network, then taking their L2 difference.

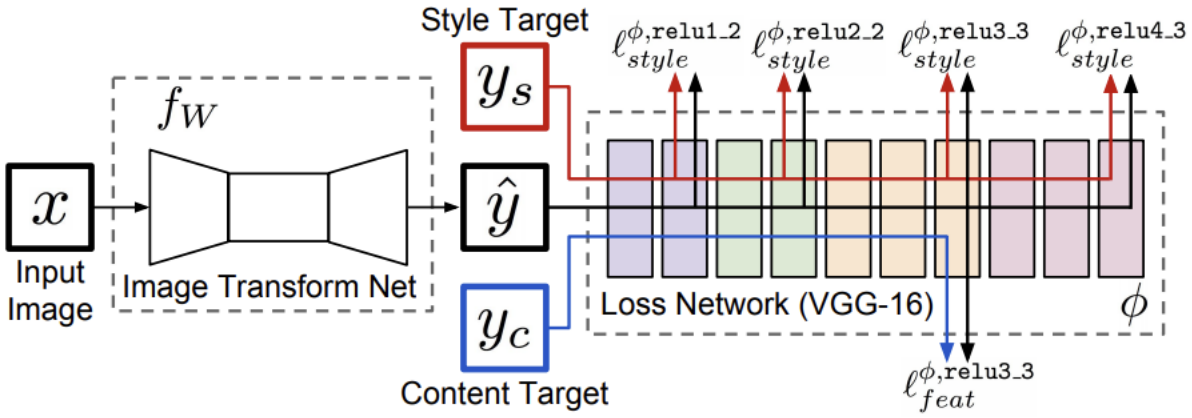


Figure 3.1. Neural network architecture for VGG as in [29]

We incorporated this architecture into our neural network framework.

FMC is rooted in the idea that we can account for shortcomings in our forward model by compensating with our prior model. Based in the P&P structure, we often don't have control over the model of the physical system. However we are able to choose our denoiser. Our choice of denoiser was the following: for a given noisy patch x_k with noise assumed to be AWGN with standard deviation σ,

$$z_k = (1-\lambda)x_k + \frac{(\lambda)}{L^2}(A^T A x_k) \tag{3.1}$$

where A and A^T are as defined in Chapter 2 and $\lambda \in [0,1]$ quantifies the blockiness added to the image. The application of A^T induces the blocky artifacts that we'd like to correct.

Incorporating the artifacts into the training of the denoiser teaches it to remove them. An example of this is shown in Figure A.1. The blocky artifacts are particularly prevalent on the diagonal line segments.

In Figure A.2, we show reconstructions within the MACE framework on *E. coli* fibers. All methods contain artifacts, though MDF is able to mitigate some of the blocky artifacts in the MACE reconstruction. However it induces its own diagonal artifacts likely learned in the prior training process.

This approach suffered due to poor control over the training process and neural network instabilities. For our work to come, we made the following adjustments. First, we switched our training code to the DnCNN code from Zhang [28] which has been shown to be a stable training process. Additionally we were careful to separate our training, validation, and testing datasets to reduce overfitting throughout.

4. METHODS

4.1 RAP and MDF

The contents of this chapter appear in [1].

A key strength of the MACE framework is the ability to incorporate operators that are not proximal maps of negative log likelihoods or other cost functions. This allows for the use of algorithmic denoisers and other operators. In this section, we leverage this observation in two ways.

First, we replace the forward operator in (2.18) with a related update operator by using a Relaxed Adjoint Projector (RAP) B in place of A^t. This operator is sometimes known as a mismatched backprojector. This change means that the forward operator is no longer the proximal map for $l(x)$. However, we show using the MACE framework that this formulation is equivalent to a formulation using the original forward operator in (2.18) but with an alternative prior that depends on B. This provides important intuition for the use of RAP and allows for the application of existing convergence results.

Second, we describe our method of Multi-Resolution Data Fusion (MDF), in which HR representative patches are collected independently of the LR scan. These patches are then used to train a denoising prior model, which intrinsically learns the underlying distribution of the high-resolution modality. We then perform LR scans over a large area and fuse the two modalities using the Multi-Agent Consensus Equilibrium framework to produce a HR image encompassing the full FoV.

4.1.1 Relaxed Adjoint Projection

Motivated by earlier work on iterative filtered backprojection in [30] [31] [32], we consider an alternative update map given by

$$\tilde{F}(x;\sigma_\lambda) = \left[x + \frac{\sigma_\lambda^2}{\sigma_\lambda^2 + L^2\sigma_w^2} B\left(y - \frac{1}{L^2}Ax\right)\right]_+ \tag{4.1}$$

where the B operator represents a bicubic upsampling by a factor of L and replaces the block replication operator A^t. We call this Relaxed Adjoint Projection (RAP) as the backprojection

operator is no longer required to be an exact adjoint. The update is driven by $y - \frac{1}{L^2}Ax$, the error between the low-resolution data and the current reconstruction, but the block-replication inherent in A^t can lead to blocky artifacts in the reconstruction. The bicubic backprojection is used to provide a smoother update to the HR image, but other interpolants may be used.

To incorporate this adjoint mismatch into the MACE framework. we first formulate the equilibrium problem associated with RAP, then prove that the solution of this problem arises from 3 different formulations:

- **RAP forward update**, standard prior, equal weight averaging
- standard forward update, standard prior, **modified averaging**
- standard forward update, **modified prior**, equal weight averaging.

Thus, the RAP formulation is equivalent to a standard inverse problem using a modified prior that can be described in terms of the mismatched backprojector.

In general, the update in (4.1) is not a proximal map for any function since the Jacobian of this map is not symmetric, which is a property of all proximal maps. By converting the mismatched backprojector with standard averaging into a standard backprojector with alternative averaging, we recover the ability to use standard proximal maps while maintaining the benefits associated with mismatched backprojection. We note that results in [33] prove convergence for an adjoint mismatch in the Proximal Gradient Algorithm but do not address the equivalence described here.

To motivate this further, note that the first-order optimality condition for a solution of (2.19) when all μ_j are equal is

$$\nabla f_1(x^*) + \cdots + \nabla f_K(x^*) = 0. \tag{4.2}$$

Also, the proximal map for a convex and differentiable f_j is given by $F_j(v_j) = x_j = v_j - \nabla f_j(x_j)$; i.e, the update can be regarded as an implicit gradient descent step, with the gradient evaluated at the output point $F_j(v_j) = x_j$.

In the case of a mismatched backprojector, we assume for the moment that $F_{\mathrm{j}}^{R_{\mathrm{j}}}$ is given by $F_{\mathrm{j}}^{R_{\mathrm{j}}}(v_{\mathrm{j}}) = x_{\mathrm{j}} = v_{\mathrm{j}} - R_{\mathrm{j}}\nabla f_{\mathrm{j}}(x_{\mathrm{j}})$ for some matrix R_{j}, which we think of as close to the identity. Using this F^R and all $\mu = 1/K$ in the equilibrium condition (2.23), we have

$$v_{\mathrm{j}}^* - R_{\mathrm{j}}\nabla f_{\mathrm{j}}(x_{\mathrm{j}}^*) = \frac{1}{K}\sum_k v_k^*. \tag{4.3}$$

Since the left hand side is x_{j}^* for each j and the right hand side is independent of j, we have $x_{\mathrm{j}}^* = x^*$ is independent of j. Summing these equations over j and subtracting the sum of the v_{j}^* from both sides and taking the negative gives

$$R_1\nabla f_1(x^*) + \cdots + R_K\nabla f_K(x^*) = 0. \tag{4.4}$$

This is the corresponding equilibrium condition for the operators $F_{\mathrm{j}}^{R_{\mathrm{j}}}$ and equal weight averaging.

When the F_{j} are algorithmic operators without a gradient formulation, we get an analogous result by using $v_{\mathrm{j}}^* - F_{\mathrm{j}}^{R_{\mathrm{j}}}(v_{\mathrm{j}}^*)$ in place of $\nabla f_{\mathrm{j}}(x_{\mathrm{j}}^*)$.

The theorem below states that the set of solutions of the equilibrium condition with mismatched backprojection are the same as those obtained using standard back projections but an alternative averaging operator $\mathbf{G}^R$, given by a matrix-weighted average using the R_{j} as matrix weights. As before, we stack the operators $\mathbf{F}_{\mathrm{j}}^{R_{\mathrm{j}}}$ to obtain $\mathbf{F}^R$. The details of the notation, hypotheses, and the proof are given in the appendix.

***Theorem** 1:* Under appropriate hypthotheses on $\mathbf{F}$, and R, there is a map from solutions $\mathbf{v}^*$ of

$$\mathbf{F}^R(\mathbf{v}^*) = \mathbf{G}(\mathbf{v}^*)$$

to solutions $\hat{\mathbf{v}}^*$ of

$$\mathbf{F}(\hat{\mathbf{v}}^*) = \mathbf{G}^R(\hat{\mathbf{v}}^*)$$

such that for each such pair, $\mathbf{G}(\mathbf{v}^*) = \mathbf{G}^R(\hat{\mathbf{v}}^*)$. There is also such a map from $\hat{\mathbf{v}}^*$ to $\mathbf{v}^*$.

Note that $\mathbf{G}(\mathbf{v}^*)$ is obtained by stacking the solution x^*, so this theorem says that these two formulations have the same set of possible reconstructions. The following theorem applies

this to give the equivalence between the use of mismatched backprojection in the data-fitting operator and the use of standard backprojection with an alternative prior.

***Theorem** 2:* With appropriate assumptions on $B = RA^T$ and the denoiser H and with equal weighting $\mu_j = 1/2$, the following two choices lead to the same MACE solution in (2.23):

- $F_1 = \tilde{F}$ is the RAP update in (4.1) and $F_2 = H$ is a given prior operator;
- $F_1 = F$ is the standard update in (2.18) and $F_2 = H \circ \Phi_R$, where Φ_R is a function dependent on the matrix R.

Hence we see that the mismatch in RAP is equivalent to a corresponding modification to the the update step of the prior operator. By using the mismatch, which is often more efficiently implemented in the form of RAP than with R^{-1}, we gain the ability to more closely match the prior to the observed structure of the data without changing the algorithmic nature of the prior.

In Figure A.4, we show the differences in image quality arising from the use of the block upsampling interpolant A^t versus the bicubic upsampling interpolant B. The use of RAP removes the repetitive texture visible in the MACE reconstruction using A^t while still preserving the fine detail seen in the ground truth.

4.1.2 Convergence of Relaxed Adjoint Projection (RAP)

From [17], Algorithm 1 is known to converge when the map $v \mapsto 2F(v) - v$ is nonexpansive, and this condition is satisfied when each F_j is the proximal map for a convex function. When the RAP update is used, then F_j is no longer a proximal map in general.

In Proposition 3 we give conditions under which F_j using RAP is a proximal map after an appropriate change of coordinates. The key idea, related to work in [34], is to consider operators of the form $F(x) = Wx + q$. When F is a proximal map, then W is symmetric and positive-definite. A change of variables allows us to recover this property even for some non-symmetric W. This allows us to prove convergence of the PnP algorithm with a RAP

forward model in Theorem 2. Since the PnP algorithm is equivalent to Algorithm 1 in the case of two operators [17], this also applies to MDF with 2 operators.

Theorem 3: Let $F(x) = Wx + q$ where $W = V\Lambda V^{-1}$ is an $N \times N$ matrix, Λ is diagonal with eigenvalues in $(0, 1]$, and $q \in \mathbb{R}^N$, and let H be a denoiser such that $V^{-1}HV$ is nonexpansive. Then the PnP algorithm converges using the operators F and H.

The conditions for guaranteed convergence of RAP allow for this method to be generally applied.

4.1.3 Multi-Resolution Data Fusion

The MACE framework with the standard or RAP forward operator provides a natural way to incorporate low-resolution data and maintain fidelity of the reconstruction to this data. For MDF, we need to incorporate selected high-resolution data, either from the image under reconstruction or from related images. For this we use a neural network denoiser as our prior agent.

The theoretical foundation of Plug-and-Play implies that the prior operator should be a denoiser for images in the target distribution perturbed by AWGN, in principle independent of the noise present in the data itself [35]. However, as seen in [16] the prior operator can play a large role in the quality of the final reconstruction. In the context of learned denoisers such as CNNs, this means that the CNN must denoise well on the images in the distribution under consideration. Since PnP is an iterative algorithm, the CNN must also denoise well on neighboring images in order to converge to a high-quality reconstruction. Ideally, the denoiser should be able to take any image in the reconstruction space and move it closer to an image in the target distribution, but in practice we must settle for an approximation based on a sparsely sampled set of images near the distribution.

Since the prior agent operates in the reconstruction space of HR images, we use a CNN denoiser trained to remove AWGN from high-resolution target images, first using a network with natural images (DnCNN), then using this same network structure trained on a small set of domain-specific HR images (MDF). We used code adapted from https://github.com/cszn/KAIR to implement DnCNN. The network architecture consists of 17 total layers with the following

structure: (i) Conv+ReLU for the first layer with 64 filters of size 3x3x1. (ii) Conv+Batch-Norm+ReLU for layers 2-16 with 64 filters of size 3x3x64. (iii) Conv for the last layer with 1 filter of size 3x3x64. The network uses a residual mapping R to learn the structure of the noise in its training pairs $(x_{\text{clean}}, x_{\text{noisy}})$. A forward pass through the model is thus given by $\hat{x}_{\text{clean}} = x_{\text{noisy}} - R(x_{\text{noisy}})$.

To train each image-tuned prior for MDF, we randomly extracted 400 180x180 patches from a high-resolution training image. This represents a naive, sparse sampling of the high-resolution data. Using these training patches, we generate corresponding noisy patches by adding AWGN with standard deviation $\sigma = 0.1$. These patch pairs are then passed through the network for training (note that there is no pairing of high-resolution images with low-resolution images). We used an increase in validation loss as a stopping criterion to avoid overfitting. Each of our MDF networks trained for 1-2 hours using 1 Nvidia V100 GPU.

We note here that increasing the super-resolution factor L necessarily increases the set of data-consistent reconstructions – increasing L increases the dimension of the kernel of A. In particular, given two reconstructions that both fit the data equally well and that are both equally well-denoised by the denoiser (more precisely, both are equilibrium solutions), there is no reason to favor one over the other. This means first that the importance of the prior operator increases with L and second that larger L gives any reconstruction algorithm more opportunity to "hallucinate" detail that may or may not be present in the true image. In the context of scientific and medical applications where the reconstruction can influence important decisions, it can be detrimental to push the limits of super-resolution and/or regularization beyond reasonable expectations [36].

5. PRESENTATION, ANALYSIS, AND INTERPRETATION

Some of the contents of this chapter appear in [1].

We apply the MDF method on 3 microscopy datasets with pronounced differences in data distribution: pentacene crystals, gold nanorods, and an *E. coli* biofilm. The pentacene crystal images are typically composed of large regions of relatively constant intensity with sharply defined edges. The gold nanorod images are composed of non-overlapping, nearly linear segments at various angles with nearly circular impurities. The *E. coli* biofilm images contain a wide variety of shapes and textures as well as large regions of nearly-empty space. The pentacene and nanorod images were obtained using SEM, while the *E. coli* images were obtained using TEM.

For the maps F_{i}, we use the data-fidelity agent in (2.18) and the DnCNN archictecture [28] trained to remove 10% AWGN as a denoising prior. We applied Algorithm 1 to determine the corresponding reconstructions.

We present comparisons of our MDF method with a variety of alternatives for 4x, 8x, and 16x interpolation. At 4x and 16x, we compare our MDF algorithm, with bicubic interpolation, MACE interpolation [27], DPSR, and DPSRGAN [13]. However, at 8x, we do not compare with DPSRGAN since it is not available for this interpolation rate.

5.0.1 Experimental Methods

Our experiments were run for both partially simulated and fully real data. In the partially simulated case, we use actual HR data and then apply the forward model to obtain the LR data but do not use the resulting aligned pairs for CNN training. In the fully real data, both the HR and the LR data are obtained experimentally. The partially simulated data has the advantage of providing ground truth for more quantitative measures of accuracy, while the fully real data results allow for qualitative assessment under more realistic conditions. Table 5.1 lists the three data sets and the experimental parameters used for data acquisition.

Transmission electron microscopy was performed on a 60-300 Thermo Fisher Titan operating at 300 kV in bright field mode. Images were collected on a 4k by 4k Gatan K2 Direct Electron Detector (DED) using Serial EM at various electron optical magnifications. LR

Table 5.1. Acquisition parameters for experimental datasets. We omit LR pixel spacing for *E. coli* as we do not perform super resolution on measured data in this case. The 2.2 nm nanorods sample was used for the results shown in Figures A.8,A.9,A.11, and A.12.

Material	HR Pixel Spacing	LR Pixel Spacing	Data Modality
E. coli	0.98 nm	N/A	TEM
Nanorods	1.1, 2.2 nm	5.5 nm	SEM
Pentacene	41.7 nm	168.9 nm	SEM

overview images were first collected, followed by automated aligned HR image montages. The bright field imaging modality uses a wide illumination that covers the entire imaging array (DED). SEM images were obtained on a FEI XL30 at 5 kV with a secondary electron detector and a Zeiss Gemini at 5 kV using an in-lens secondary electron detector. The interaction volume of the focused electron beam was on the order of the size of the resulting pixel size in the image. The Scanning Electron Microscopy modality raster scanned a focused beam across the sample with a pixel dwell time of 50 nanoseconds. These TEM and SEM images form our HR datasets.

From this HR data, we create synthetic LR data by applying the A/L^2 operator, which mimics the acquisition of LR data. These LR images were then passed to each method to generate a super-resolution image. This allows for a ground truth comparison. All data used in testing the algorithm was separate from data used to train it. This holds for both LR and HR data used in testing.

Based on the MACE equation $\mathbf{F}(\mathbf{v}) = \mathbf{G}(\mathbf{v})$, we define a measure of convergence error as

$$\text{Convergence Error} = \frac{||\mathbf{G}(\mathbf{v}) - \mathbf{F}(\mathbf{v})||_2}{\sigma_n ||\mathbf{G}(\mathbf{v})||_2}, \tag{5.1}$$

where σ_n is the noise level used to train the prior model.

5.0.2 Results on Synthetic Data

In Figures A.6–A.11, we display a HR ground-truth image, the corresponding simulated LR data, the output of bicubic upsampling (as a baseline), DPSRGAN (when possible), DPSR, MACE using a CNN trained on natural images as the prior agent, and MDF.

In Figure A.6 at 4x super-resolution, MDF removes noise while maintaining data fidelity and reproducing realistic texture, while the other reconstructions contain blocky artifacts or texture inaccuracies. We display a zoomed region in Figure A.7 to further show the disparities.

In Figure A.8 at 4x super-resolution, the images are quite similar. This is due to the rigidity and repetition of the structure. At this level, DPSR produces a high-quality reconstruction but tends to generate overly-thick nanorods. In comparison, MDF is able to produce a high-quality and high-fidelity reconstruction.

Figure A.9 illustrates an example of 8x super-resolution on more highly structured data. In this case the problem is significantly underconstrained, but MDF is able to leverage the data-fidelity operator and a domain-specific CNN to produce a high-quality reconstruction. In this case the competing reconstructions each have significant shortcomings.

Table 5.2. Average PSNRs over 100 images for *E. coli*, Pentacene, and Nanorods datasets at 4x interpolation. On average, MDF outperforms all other methods.

Material	MDF	MACE	DPSR	DPSRGAN	Bicubic
E. coli	**20.02**	19.95	19.69	14.27	19.98
Pentacene	**34.09**	33.78	32.65	28.84	30.00
Nanorods	**34.89**	34.21	34.55	29.87	32.43

In Figure A.10, with 4x super-resolution on data with much less regularity and significant high-frequency components, MDF produces the highest PSNR, although not by a significant margin. However, MDF arguably provides the best visual compromise between clarity and fit to data among the competing methods.

In Figure A.11, we present 16x super-resolution results. At this level, there is not enough data present for any method to generate an accurate reconstruction. Moreover MDF begins to 'hallucinate' and generate shapes that are both not present in the image nor in its training data.

In Table 5.2, we show average results for reconstructions of synthetic LR data at 4x interpolation. For each dataset, we extracted 100 256x256 images and created corresponding simulated LR data. These images were then passed into each interpolation method for recon-

struction. Finally we collected the PSNRs relative to the original high-resolution image and averaged these across the 100 images to generate the values shown. As shown in Table 5.2, MDF outperforms all other interpolation methods tested.

In Figure A.3, we plot the average convergence error in (5.1) as a function of iteration over these 100 256x256 images. For each dataset, MDF converged with under 5% error. The gold nanorods dataset at 4x super-resolution leads to the highest convergence error, which is likely due to the existence of multiple data-consistent reconstructions at this level of super-resolution.

5.0.3 Results on Measured Data

In Figures A.12–A.16, we show results using measured LR data as input. In this case there is no paired HR data for quantitative comparison, so we provide a measured HR image of a similar specimen for visual comparison.

In Figure A.12, with 4x super-resolution of gold nanorod images, all methods are able to reconstruct the data. However in comparison, MDF produces a sharper image that is more faithful to the LR data.

In Figure A.13, with 4x super-resolution of gold nanorod images, the data is not severely undersampled, so each method is able to reconstruct the shape of the nanorods. However, relative to the other methods, MDF provides a more faithful reconstruction of the nanorod interiors and ends while removing background noise. We provide a zoomed image in A.14 that shows the difference in the individual nanorod reconstructions.

In Figure A.15, the majority of the methods produce aliasing artifacts along the edge of the crystals and texture issues on the crystal itself. MDF minimizes these artifacts relative to the other methods and again provides a good balance between clarity and realistic texture. These issues are illustrated further in Figure A.16.

5.0.4 RAP in Practice

It's worth noting that the assumptions made in Theorem 2 may not always be met. However we performed experiments across our pentacene, gold nanorods, and *E. coli* datasets to show that the error between the methods was acceptable.

This was performed by defining the following:

$$F_1 = (2.18), G(w) = (I + BA)^{-1}[I(\sum_{\mathrm{i}=1}^{K-1} w_{\mathrm{i}}) + \frac{1}{L^2}(BA)w_K] \tag{5.2}$$

$$\tilde{F}_1 = (4.1), \tilde{G}(w) = \frac{1}{K}\sum_k w_k \tag{5.3}$$

where F_2 is the base DnCNN denoiser described in Section 4.1.3 for both F and $\tilde{F}$. We then alternated application of the forward and averaging operators in both cases for 200 iterations. Finally, we compiled the error in the final reconstructions and averaged them across 100 images for each dataset. Here our error is calculated as

$$Error = \frac{\|w - \tilde{w}\|}{\|w\|} \tag{5.4}$$

where w is the reconstruction using the operators in (5.2) and $\tilde{w}$ is the reconstruction using the operators in (5.3).

Table 5.3. Error calculation for RAP in the MACE framework. All datasets have under 4×10^{-3} error in practice.

Material	SR Factor	Error
Pentacene	4x	3.902×10^{-3}
E. coli	4x	2.175×10^{-4}
Nanorods	4x	1.1891×10^{-7}

As shown in Table 5.3, all methods have error less than 4×10^{-3}. For practical purposes, this is sufficient for use.

Table 5.4. Data acquisition speedup with MDF framework. The speedup factor is calculated by taking the ratio of the number of HR reconstructed pixels to the sum of the acquired LR pixels and HR training pixels

Material	SR Factor	LR data	HR training data	Reconstructed HR data	Speed-Up
E. coli	4x	7404 x 7666	5049 x 9827	29616 x 30664	8.54x
Pentacene	4x	1280 x 755	1280 x 755	5120 x 3020	10.05x
Nanorods	5x	2048 x 1388	1232 x 1367	10240 x 6940	15.70x
Nanorods	8x	2048 x 1388	1232 x 1367	16384 x 11104	40.19x

In Table 5.4, we examine the speedup in acquisition time due to the MDF framework. The speed-up is calculated by taking the ratio of the pixels necessary for a HR FoV to the sum of the pixels in the LR FoV and the pixels in the HR training data.

$$\text{Speed-Up} = \frac{\text{HR Reconstruction Pixels}}{\text{Acquired LR pixels} + \text{HR Training Pixels}}$$

In the ideal case, in which a domain-specific CNN denoiser is already trained, the acquisition speed-up for Lx interpolation is L^2. In the cases shown in Table 5.4 we include the HR data acquisition needed for CNN training, so the actual speed-up ranges from roughly 50% to 62% of the ideal.

6. SUMMARY

We introduced a Multi-Resolution Data Fusion framework that incorporates a Relaxed Adjoint Projection and a domain-specific neural network prior operator. The Relaxed Adjoint Projection leads to improved final reconstruction quality, is straightforward to implement, and is provably equivalent to using the standard data fitting operator with a modified prior. The domain-specific neural network prior operator is trained on a limited set of high-resolution images that are not paired with low-resolution images.

In our set of experiments, MDF reduces artifacts relative to existing methods while maintaining fidelity to acquired data and accurately resolving sub-pixel-scale features. Moreover, since MDF relies on a denoiser for HR images, it can be used at multiple super-resolution factors without additional training. By changing the forward model, it can be used for multiple image acquisition models, again without retraining. This modularity is an important strength in that each component can be used for multiple applications.

For future work, we'd like to extend this method to alternative forward models and generative denoisers. Additionally we plan to work on improving the speed-up further by parallelizing the code and applying it across multiple GPUs.

REFERENCES

[1] E. J. Reid, G. T. Buzzard, L. F. Drummy, and C. A. Bouman, "Multi-resolution data fusion for super resolution imaging," 2021. arXiv: 2105.06533 [eess.IV].

[2] K. Park, M.-s. Hsiao, Y.-J. Yi, S. Izor, H. Koerner, A. Jawaid, and R. A. Vaia, "Highly concentrated seed-mediated synthesis of monodispersed gold nanorods," *ACS Applied Materials & Interfaces*, vol. 9, no. 31, pp. 26 363–26 371, 2017. DOI: 10.1021/acsami.7b08003.

[3] Y. He, J. Wu, D. Qiu, and Z. Yu, "Construction of 3-D realistic representative volume element failure prediction model of high density rigid polyurethane foam treated under complex thermal-vibration conditions," *International Journal of Mechanical Sciences*, vol. 193, p. 106 164, Mar. 2021. DOI: 10.1016/j.ijmecsci.2020.106164.

[4] H. S. Anderson, J. Ilic-Helms, B. Rohrer, J. Wheeler, and K. Larson, "Sparse imaging for fast electron microscopy," in *Computational Imaging XI*, vol. 8657, International Society for Optics and Photonics, Feb. 2013, p. 86570C. DOI: 10.1117/12.2008313.

[5] D. B. Williams and C. B. Carter, "The transmission electron microscope," in *Transmission electron microscopy*. Springer, 1996, pp. 3–17.

[6] S. A. Dyer and J. S. Dyer, "Cubic-spline interpolation. 1," *IEEE Instrumentation Measurement Magazine*, vol. 4, no. 1, pp. 44–46, 2001. DOI: 10.1109/5289.911175.

[7] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a Deep Convolutional Network for Image Super-Resolution," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2014, pp. 184–199. DOI: 10.1007/978-3-319-10593-2_13.

[8] J. Yamanaka, S. Kuwashima, and T. Kurita, "Fast and accurate image super resolution by deep cnn with skip connection and network in network," Oct. 2017, pp. 217–225. DOI: 10.1007/978-3-319-70096-0_23.

[9] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 105–114. DOI: 10.1109/CVPR.2017.19.

[10] M. S. M. Sajjadi, B. Schölkopf, and M. Hirsch, "EnhanceNet: Single Image Super-Resolution Through Automated Texture Synthesis," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 4501–4510. DOI: 10.1109/ICCV.2017.481.

[11] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *The European Conference on Computer Vision Workshops (ECCVW)*, Sep. 2018.

[12] N. C. Rakotonirina and A. Rasoanaivo, "ESRGAN+ : Further Improving Enhanced Super-Resolution Generative Adversarial Network," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 3637–3641. DOI: 10.1109/ICASSP40776.2020.9054071.

[13] K. Zhang, W. Zuo, and L. Zhang, "Deep Plug-And-Play Super-Resolution for Arbitrary Blur Kernels," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019, pp. 1671–1681. DOI: 10.1109/CVPR.2019.00177.

[14] Y. Qian, J. Xu, L. F. Drummy, and Y. Ding, "Effective super-resolution methods for paired electron microscopic images," *IEEE Transactions on Image Processing*, vol. 29, pp. 7317–7330, 2020. DOI: 10.1109/tip.2020.3000964.

[15] A. Shocher, N. Cohen, and M. Irani, "Zero-Shot Super-Resolution Using Deep Internal Learning," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 3118–3126. DOI: 10.1109/CVPR.2018.00329.

[16] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-Play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing*, Dec. 2013, pp. 945–948. DOI: 10.1109/GlobalSIP.2013.6737048.

[17] G. T. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, "Plug-and-Play Unplugged: Optimization-Free Reconstruction Using Consensus Equilibrium," *SIAM Journal on Imaging Sciences*, vol. 11, no. 3, Jan. 2018, Publisher: Society for Industrial and Applied Mathematics. DOI: 10.1137/17M1122451.

[18] S. Sreehari, S. V. Venkatakrishnan, K. L. Bouman, J. P. Simmons, L. F. Drummy, and C. A. Bouman, "Multi-Resolution Data Fusion for Super-Resolution Electron Microscopy," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jul. 2017, pp. 1084–1092. DOI: 10.1109/CVPRW.2017.146.

[19] B. Liu and J. Liu, "Overview of image noise reduction based on non-local mean algorithm," *MATEC Web of Conferences*, vol. 232, p. 03 029, Jan. 2018. DOI: 10.1051/matecconf/201823203029.

[20] A. Brifman, Y. Romano, and M. Elad, "Turning a denoiser into a super-resolver using plug and play priors," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 1404–1408.

[21] M. U. Ghani and W. C. Karl, "Data and image prior integration for image reconstruction using consensus equilibrium," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 297–308, 2021. DOI: 10.1109/TCI.2021.3062986.

[22] C. A. Bouman, "Model based imaging," in. 2013.

[23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, Jan. 2011. DOI: 10.1561/2200000016.

[24] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3929–3938.

[25] K. Zhang, Y. Li, W. Zuo, L. Zhang, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *arXiv preprint arXiv:2008.13751*, 2020.

[26] B. Shi, Q. Lian, and H. Chang, "Deep prior-based sparse representation model for diffraction imaging: A plug-and-play method," *Signal Processing*, vol. 168, p. 107 350, 2020.

[27] S. Majee, T. Balke, C. A. J. Kemp, G. T. Buzzard, and C. A. Bouman, "4D X-Ray CT Reconstruction using Multi-Slice Fusion," in *2019 IEEE International Conference on Computational Photography (ICCP)*, May 2019, pp. 1–8. DOI: 10.1109/ICCPHOT.2019.8747328.

[28] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017. DOI: 10.1109/TIP.2017.2662206.

[29] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2016, pp. 694–711. DOI: 10.1007/978-3-319-46475-6_43.

[30] D. S. Lalush and B. M. W. Tsui, "Improving the convergence of iterative filtered backprojection algorithms," *Medical Physics*, vol. 21, no. 8, pp. 1283–1286, 1994. DOI: 10.1118/1.597210.

[31] A. Ziabari, D. H. Ye, L. Fu, S. Srivastava, K. Sauer, J.-B. Thibault, and C. Bouman, "Model based iterative reconstruction with spatially adaptive sinogram weights for wide-cone cardiac ct," in *The 5th International Conference on Image Formation in X-Ray Computed Tomography*, Dec. 2018, pp. 15–18.

[32] R. Schofield, L. King, U. Tayal, I. Castellano, J. Stirrup, F. Pontana, J. Earls, and E. Nicol, "Image reconstruction: Part 1 – understanding filtered back projection, noise and image acquisition," *Journal of Cardiovascular Computed Tomography*, vol. 14, no. 3, pp. 219–225, 2020. DOI: https://doi.org/10.1016/j.jcct.2019.04.008.

[33] E. Chouzenoux, J.-C. Pesquet, C. Riddell, M. Savanier, and Y. Trousset, "Convergence of Proximal Gradient Algorithm in the Presence of Adjoint Mismatch," CentraleSupélec, Research Report, Oct. 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02961431.

[34] P. Nair, R. G. Gavaskar, and K. N. Chaudhury, "Fixed-point and objective convergence of plug-and-play algorithms," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 337–348, 2021. DOI: 10.1109/TCI.2021.3066053.

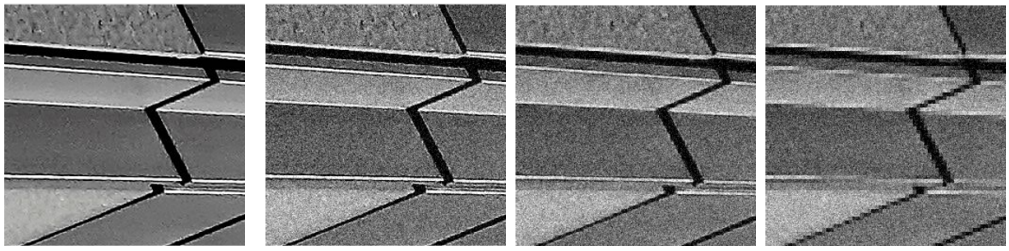
[35] C. A. Bouman, G. T. Buzzard, and B. Wohlberg, "Plug-and-Play: A General Approach for the Fusion of Sensor and Machine Learning Models," *SIAM News*, vol. 54, no. 2, pp. 1, 4, Mar. 2021.

[36] V. Antun, F. Renna, C. Poon, B. Adcock, and A. C. Hansen, "On instabilities of deep learning in image reconstruction and the potential costs of AI," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30 088–30 095, Dec. 2020, Publisher: National Academy of Sciences Section: Colloquium on the Science of Deep Learning. DOI: 10.1073/pnas.1907377117.

[37] E. K. Ryu and S. Boyd, "Primer on monotone operator methods," *Applied and Computational Mathematics*, vol. 15, no. 1, pp. 3–43, 2016.

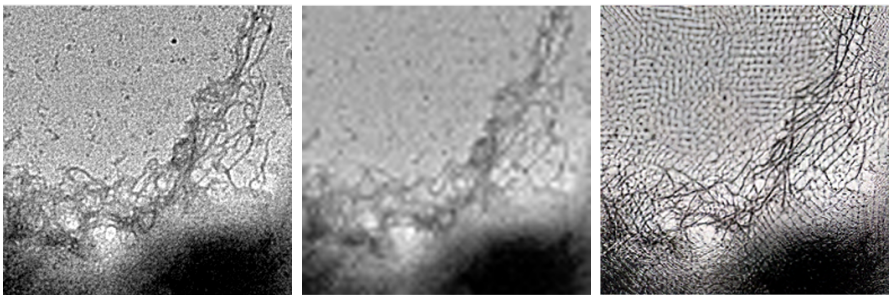
[38] F. H. Clarke, "On the inverse function theorem.," *Pacific Journal of Mathematics*, vol. 64, no. 1, pp. 97–102, 1976. DOI: pjm/1102867214.

A. FIGURES



(a) Ground Truth (b) $\lambda = 0$ (c) $\lambda = 0.5$ (d) $\lambda = 1$

Figure A.1. Example of the blockiness added in Forward Model Compensation for various λ values.



(a) HR GT (b) Bicubic (c) DPSRGAN

(d) DPSR (e) MACE (f) MDF

Figure A.2. 4x super-resolution reconstructions of a simulated LR EM image of *E. coli* fibers. MDF is able to most accurately reconstruct the fibers, but still suffers from false textures.

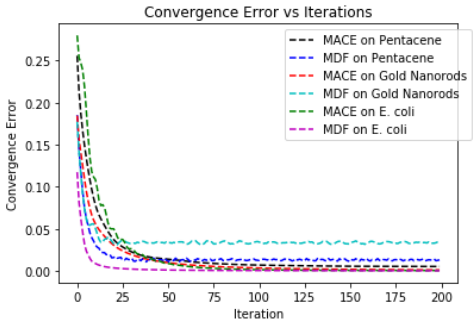
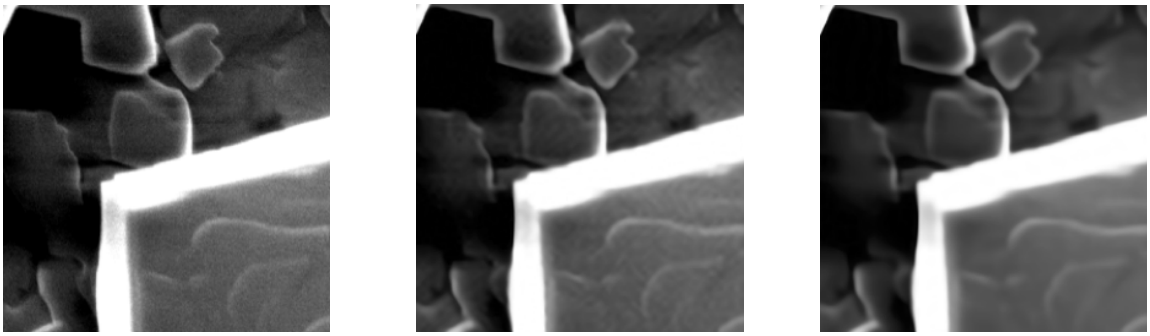
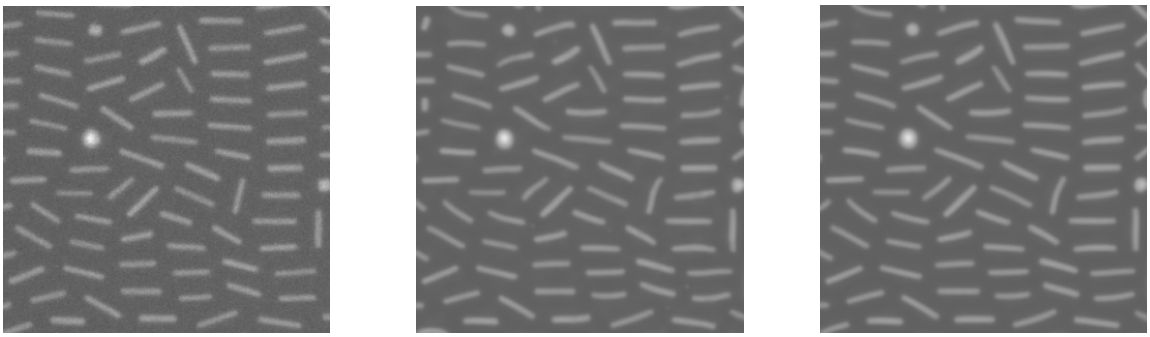


Figure A.3. Convergence Error using MACE and MDF on pentacene, gold nanorods, and *E. coli* datasets. All methods converge with under 5% error.



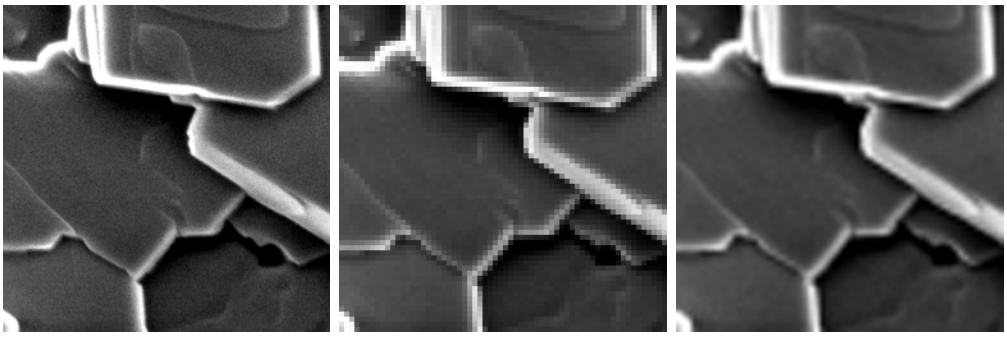
(a) Ground Truth (b) MACE (c) MACE+RAP

Figure A.4. 4x super-resolution reconstructions of a simulated LR EM image of pentacene crystals. Each shows a field of view 10.68μ wide. Note the textural differences between the MACE and MACE + RAP reconstructions.

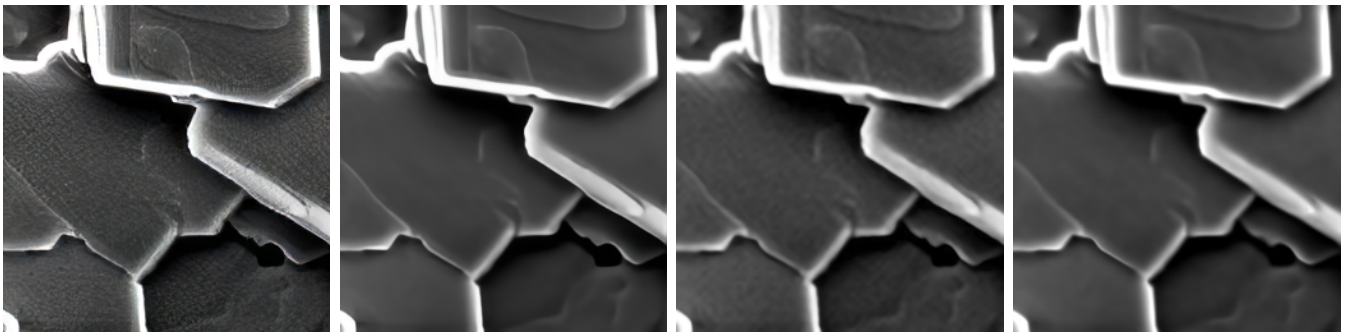


(a) Ground Truth (b) MACE (c) MACE+RAP

Figure A.5. 8x super-resolution reconstructions of a simulated LR EM image of gold nanorods. Each shows a field of view 568.32 nm wide. Note the pixel-sized artifacts removed by MACE + RAP



(a) HR GT (b) Simulated LR (c) Bicubic, 28.94 dB

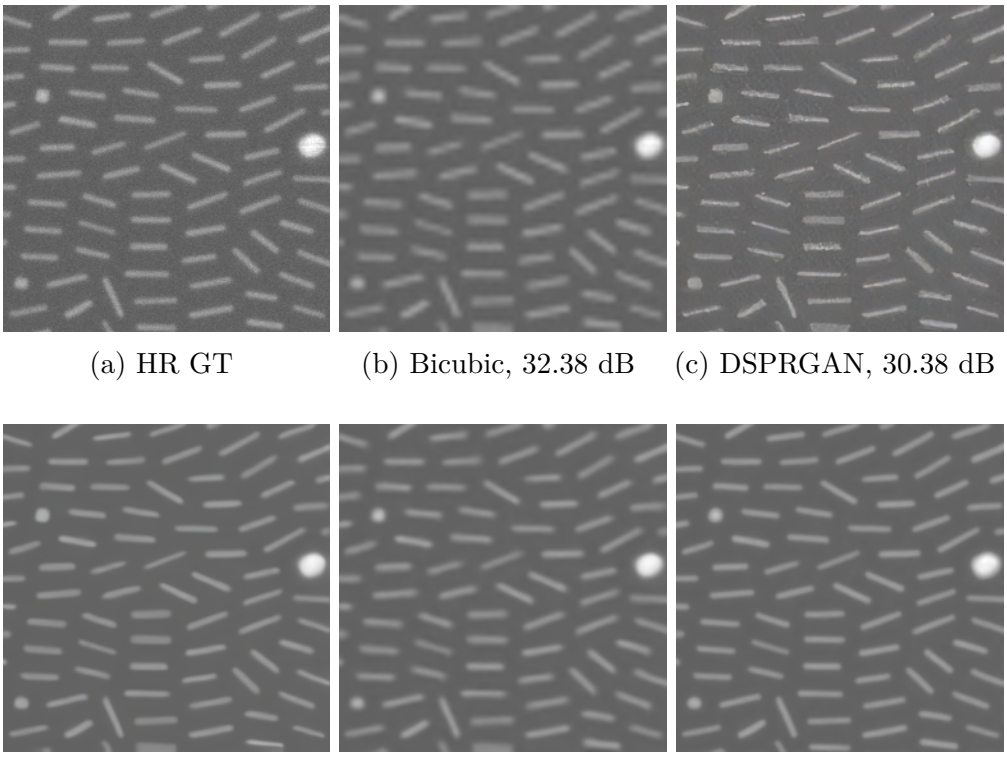


(d) DSPRGAN, 27.03 dB (e) DPSR, 30.79 dB (f) MACE, 32.18 dB (g) MDF 32.36 dB

Figure A.6. 4x super-resolution reconstructions of a simulated LR EM image of pentacene crystals. Each shows a field of view 10.68 μ wide. MDF is able to most accurately reconstruct the crystal without inducing false textures.

(a) HR GT (b) Simulated LR (c) Bicubic (d) DSPRGAN (e) DPSR (f) MACE (g) MDF

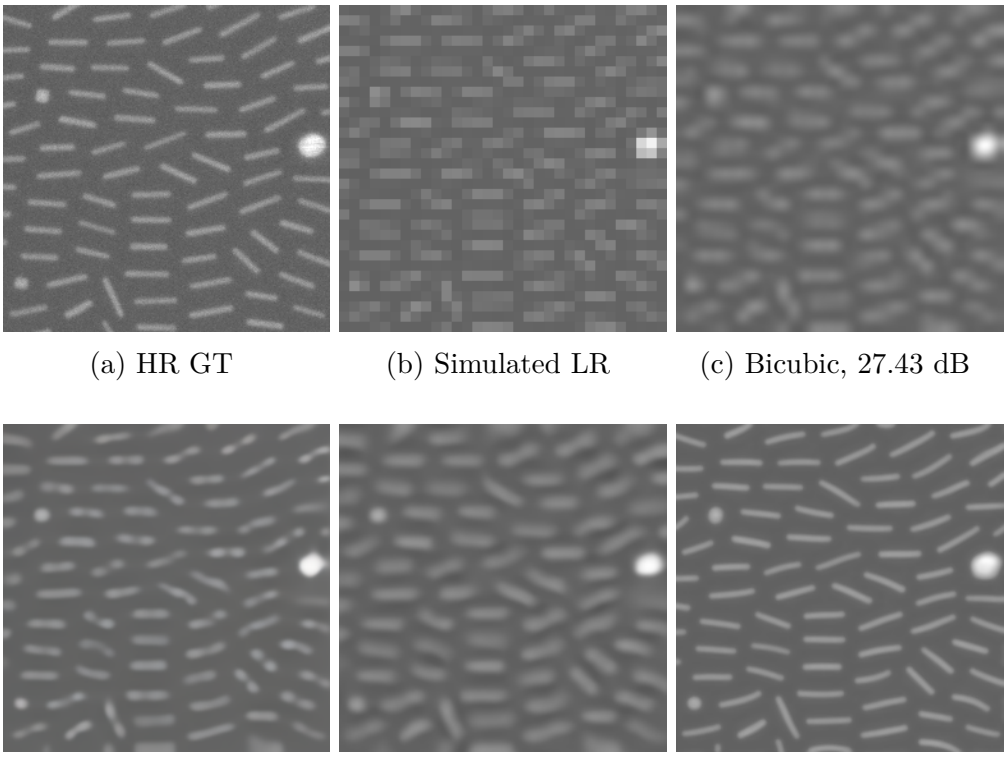
Figure A.7. Zoomed field of view of the 4x super-resolution reconstructions in Figure A.6. Each shows a zoomed field of view 5.34μ wide.



(a) HR GT (b) Bicubic, 32.38 dB (c) DSPRGAN, 30.38 dB

(d) DPSR, 34.51 dB (e) MACE, 33.87 dB (f) MDF 35.00 dB

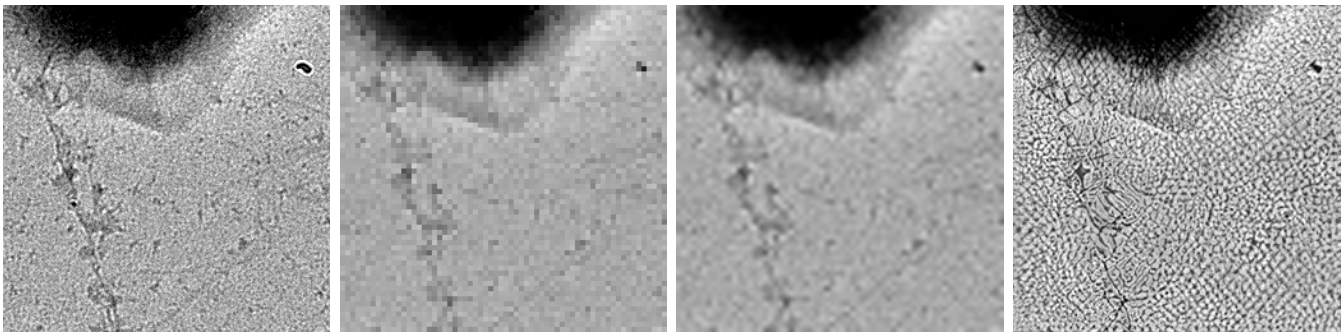
Figure A.8. 4x Super Resolution reconstructions of a simulated LR EM image of gold nanorods. Reconstructions show a field of view 569 nm wide.



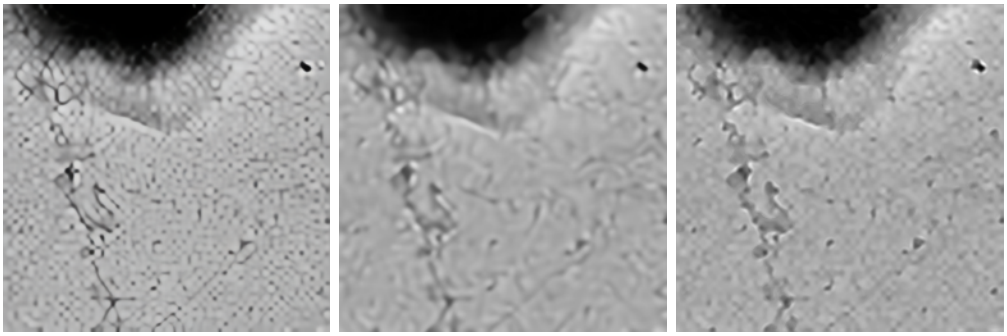
(a) HR GT (b) Simulated LR (c) Bicubic, 27.43 dB

(d) DPSR, 30.09 dB (e) MACE, 28.36 dB (f) MDF 32.43 dB

Figure A.9. 8x super-resolution reconstructions of a simulated LR EM image of gold nanorods. Each shows a field of view 569 nm wide. Only MDF is able to reconstruct defined nanorods with the proper shape.

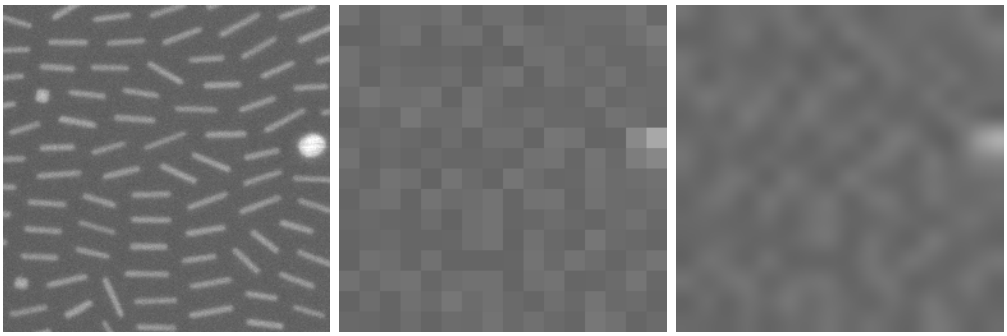


(a) HR GT (b) Simulated LR (c) Bicubic, 19.90 dB (d) DPSRGAN, 14.54 dB

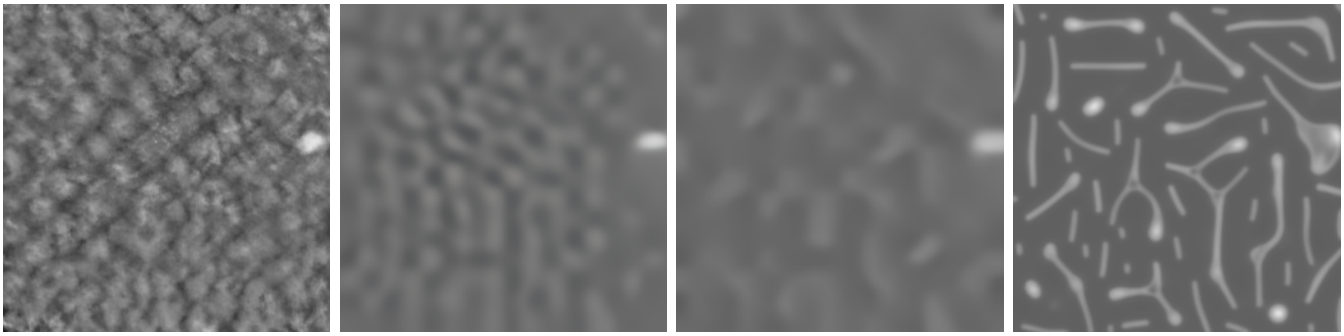


(e) DPSR, 19.69 dB (f) MACE, 19.84 dB (g) MDF 19.96 dB

Figure A.10. 4x super-resolution reconstructions of a simulated LR EM image of E coli. Each shows a field of view 251 nm wide. MACE and MDF are the only methods able to reconstruct the background textures.

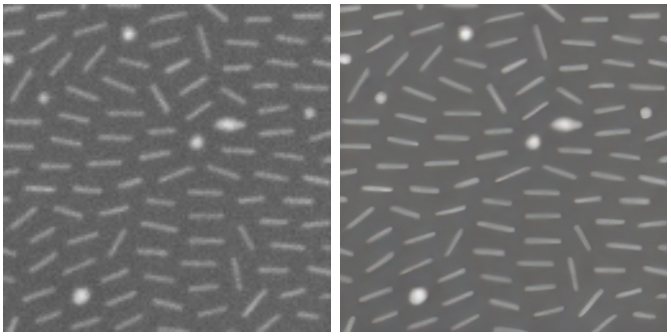


(a) HR GT (b) Simulated LR (c) Bicubic, 23.80 dB



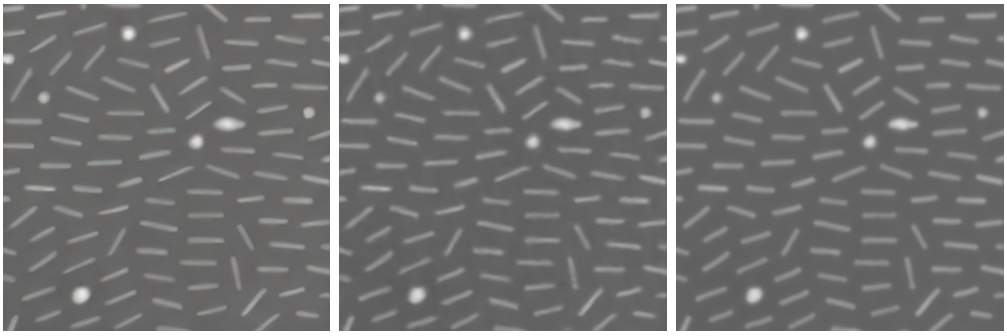
(d) DPSRGAN, 21.10 dB (e) DPSR, 24.04 dB (f) MACE, 23.96 dB (g) MDF 20.32dB

Figure A.11. 16x super-resolution reconstructions of a simulated LR EM image of gold nanorods. Each shows a field of view 569 nm wide. Note the artifacts created by the prior in this case of severe ill-conditioning.



(a) Given LR Image, Bicubically Upsampled

(b) DPSRGAN

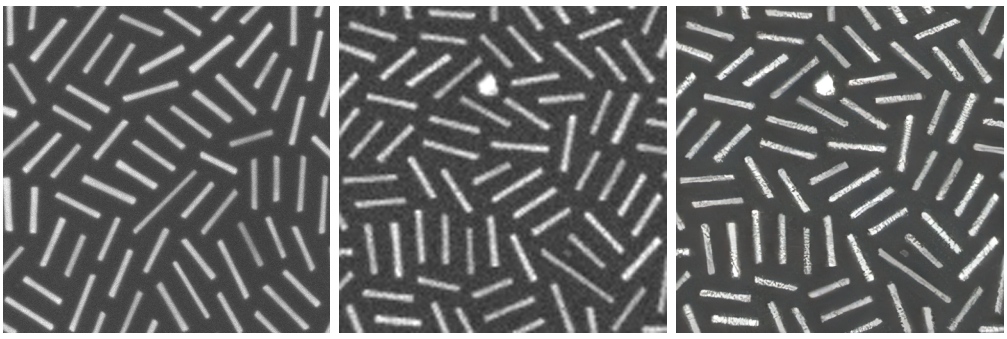


(c) DPSR

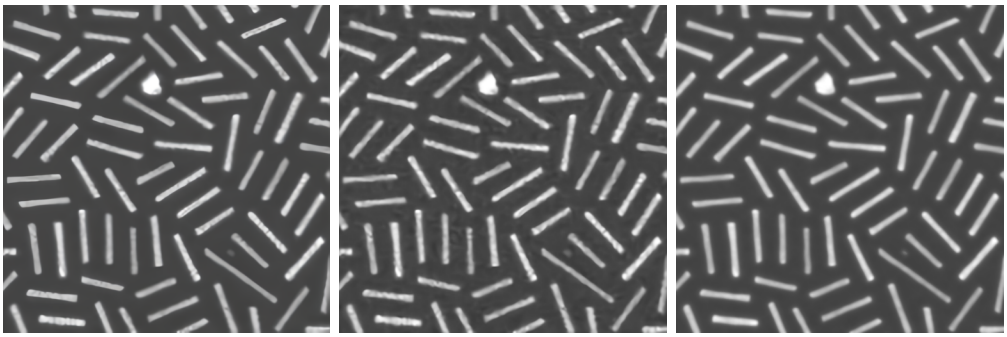
(d) MACE

(e) MDF

Figure A.12. 4x super-resolution reconstructions of a measured LR EM image of gold nanorods. Each shows a field of view 563.2 nm wide. (a) Given LR image, (b) Bicubic interpolation (c) DPSRGAN, (d) DPSR, (e) MACE, (f) MDF. Note the poor reconstruction of the thickness of nanorod in (c) and (d)

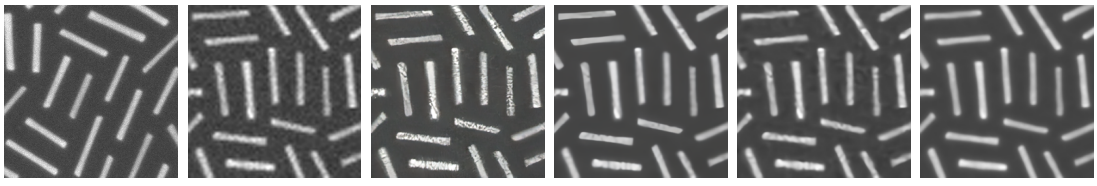


(a) HR Example Image (b) Given LR Image, Bicubically Upsampled (c) DPSRGAN



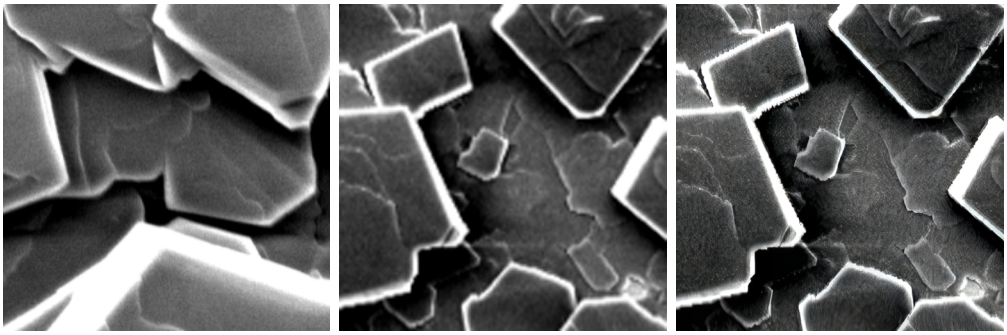
(d) DPSR (e) MACE (f) MDF

Figure A.13. 4x super-resolution reconstructions of a measured LR EM image of gold nanorods. Each shows a field of view 704 nm wide. While all methods reconstruct the nanorods' shape, only MDF is able to completely reconstruct the nanorods without internal gaps.

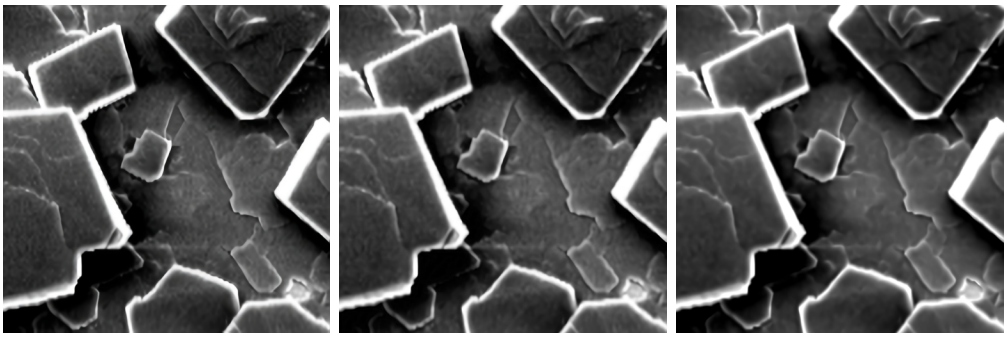


(a) HR Example Image (b) Given LR Image (c) DPSRGAN (d) DPSR (e) MACE (f) MDF

Figure A.14. Zoomed 4x super-resolution reconstructions of in Figure A.13. Each shows a field of view 352 nm wide.

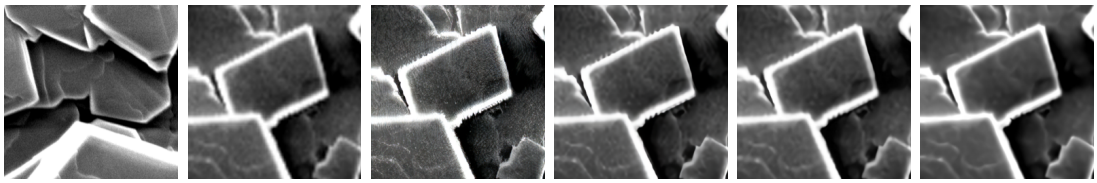


(a) HR Example Image (b) Given LR Image (c) DPSRGAN



(d) DPSR (e) MACE (f) MDF

Figure A.15. 4x super-resolution reconstructions of a measured LR EM image of pentacene crystals. Each show a zoomed field of view 21.34 μ wide. Note the significant aliasing artifacts in all non-MDF reconstructions.



(a) HR Example Image (b) Given LR Image (c) DPSRGAN (d) DPSR (e) MACE (f) MDF

Figure A.16. Zoomed 4x super-resolution reconstructions of those in A.15. Each shows a zoomed field of view 10.67 μ wide.

B. MDF PROXIMAL MAP DERIVATION

The goal of this section is to demonstrate the derivation of the proximal map for the Super Resolution Forward Model for use in the Plug & Play framework given in [18]. Namely we want to show that if $AA^T = rI$, then

$$\begin{aligned} F(\tilde{x}, \sigma_\lambda^2) &:= \underset{x \geq 0}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma_w^2} ||y - Ax||_2^2 + \frac{1}{2\sigma_\lambda^2} ||x - \tilde{x}||_2^2 \right\} \\ &= \tilde{x} + \frac{\sigma_\lambda^2}{r\sigma_\lambda^2 + \sigma_w^2} A^T (y - A\tilde{x}) \end{aligned}$$

Here the forward model is given by $y = Ax + \epsilon$ where $A \in \mathbb{R}^{M \times N}$ represents the PSF of the electron microscope and $\epsilon \sim N(0, \sigma_w^2)$ is a M dimensional vector of i.i.d. additive white Gaussian noise.

Define the objective function as

$$f(x) := \frac{1}{2\sigma_w^2} ||y - Ax||_2^2 + \frac{1}{2\sigma_\lambda^2} ||x - \tilde{x}||_2^2$$

Differentiating the objective with respect to x and multiplying by σ_λ^2 yields the first-order optimality condition

$$\frac{\sigma_\lambda^2}{\sigma_w^2} A^T (Ax - y) + (x - \tilde{x}) = 0. \tag{B.1}$$

Defining $\rho = \sigma_\lambda^2 / \sigma_w^2$ and solving for the lone x gives

$$x = \tilde{x} + \rho A^T (y - Ax).$$

This shows that x has the form $x = \tilde{x} + A^T z$ for some z. Using this form in (B.1) gives

$$\begin{aligned} 0 &= \rho A^T (A(\tilde{x} + A^T z) - y) + ((\tilde{x} + A^T z) - \tilde{x}) \\ &= \rho (A^T A \tilde{x} + A^T A A^T z - A^T y) + A^T z. \end{aligned}$$

Applying $AA^T = rI$, we have

$$\rho(A^T A\tilde{x} + rA^T z - A^T y) + A^T z = 0,$$

or

$$(1 + \rho r)A^T z = \rho A^T (y - A\tilde{x}).$$

This has solution

$$\begin{aligned} z &= \left(\frac{\rho}{1 + r\rho}\right)(y - A\tilde{x}) \\ &= \left(\frac{\sigma_\lambda^2}{\sigma_w^2 + r\sigma_\lambda^2}\right)(y - A\tilde{x}). \end{aligned}$$

Then using $x = \tilde{x} + A^T z$ gives

$$x = \tilde{x} + \left(\frac{\sigma_\lambda^2}{\sigma_w^2 + r\sigma_\lambda^2}\right) A^T (y - A\tilde{x}) \tag{B.2}$$

———-

The above shows that

$$f(x) := \frac{\rho}{2}||y - Ax||_2^2 + \frac{1}{2}||x - \tilde{x}||_2^2 \tag{B.3}$$

has solution

$$x = \tilde{x} + \left(\frac{\rho}{1 + r\rho}\right) A^t (y - A\tilde{x}). \tag{B.4}$$

Consider now the objective function below, where $AA^T = L^2 I$.

$$\begin{aligned} \frac{\rho}{2}\left\|\frac{1}{L^2}Ax - y\right\|^2 + \frac{1}{2}\|x - \tilde{x}\|^2 &= \frac{\rho}{2}\left|\frac{1}{L^2}\right|^2 \left\|(Ax - L^2 y)\right\|^2 + \frac{1}{2}\|x - \tilde{x}\|^2 \\ &= \frac{\rho}{2L^4}\|Ax - L^2 y\|^2 + \frac{1}{2}\|x - \tilde{x}\|^2. \end{aligned}$$

We make the following variable assignments $\hat{y} = L^2 y$, $\hat{\rho} = \rho / L^4$, which transforms the objective function to the form of (B.3) with $\hat{y}$ in place of y and $\hat{\rho}$ in place of ρ.

Then using $r = L^2$ and the closed form in (B.4) gives

$$x = \tilde{x} + \left(\frac{\hat{\rho}}{1 + L^2 \hat{\rho}} \right) A^T (\hat{y} - A\tilde{x})$$

Using $\hat{\rho} = \sigma_\lambda^2 / (L^4 \sigma_w^2)$ and $\hat{y} = L^2 y$ yields

$$x = \tilde{x} + \left(\frac{\sigma_\lambda^2 / (L^4 \sigma_w^2)}{1 + L^2 \sigma_\lambda^2 / (L^4 \sigma_w^2)} \right) A^T (L^2 y - A\tilde{x})$$

or

$$x = \tilde{x} + \left(\frac{\sigma_\lambda^2}{L^4 \sigma_w^2 + L^2 \sigma_\lambda^2} \right) A^T (L^2 y - A\tilde{x})$$

or

$$x = \tilde{x} + \left(\frac{\sigma_\lambda^2}{L^2 \sigma_w^2 + \sigma_\lambda^2} \right) A^T (y - \frac{1}{L^2} A\tilde{x})$$

C. RELAXED ADJOINT PROJECTION PROOFS

The contents of this chapter appear in [1].

We begin with a proposition necessary to define several maps.

Proposition C.0.0.1 *Let ϕ be maximally monotone. Then $(I+\phi)^{-1}$ is globally defined, single-valued, and nonexpansive.*

Proof: See [37, section 6]. ■

In the theorem below, we use maps $\mathbf{F}_{\mathrm{i}}$ that are implicitly defined in the sense that the map ϕ_{i} is evaluated at the output of the corresponding map $\mathbf{F}_{\mathrm{i}}$. This is a generalization of the condition satisfied by a proximal map and is equivalently written as the resolvent of ϕ_{i}, as indicated.

Theorem C.0.0.1 *Assume that each of ϕ_{i} and $R_{\mathrm{i}}\phi_{\mathrm{i}}$ are maximal monotone functions from $\mathbb{R}^n$ to itself for each $\mathrm{i}=1,\ldots,K$, where each R_{i} is an $n\times n$ matrix with $\sum_{\mathrm{i}} R_{\mathrm{i}}$ invertible. Define*

- $F_{\mathrm{i}}(v_{\mathrm{i}}) = v_{\mathrm{i}} - \phi_{\mathrm{i}}(F_{\mathrm{i}}(v_{\mathrm{i}})) = (I+\phi_{\mathrm{i}})^{-1}(v_{\mathrm{i}})$
- $F_{\mathrm{i}}^{R}(v_{\mathrm{i}}) = v_{\mathrm{i}} - R_{\mathrm{i}}\phi_{\mathrm{i}}(F_{\mathrm{i}}^{R}(v_{\mathrm{i}})) = (I+R_{\mathrm{i}}\phi_{\mathrm{i}})^{-1}(v_{\mathrm{i}})$
- $\boldsymbol{G}_{\mathrm{i}}(\boldsymbol{v}) = \frac{1}{K}\sum_{\mathrm{i}} v_{\mathrm{i}}$
- $\boldsymbol{G}_{\mathrm{i}}^{R}(\boldsymbol{v}) = (\sum_{\mathrm{i}} R_{\mathrm{i}})^{-1}(\sum_{\mathrm{i}} R_{\mathrm{i}}v_{\mathrm{i}})$

Then there is a map from solutions $\boldsymbol{v}^*$ of

$$\boldsymbol{F}^{R}(\boldsymbol{v}^*) = \boldsymbol{G}(\boldsymbol{v}^*)$$

to solutions $\hat{\boldsymbol{v}}^*$ of

$$\boldsymbol{F}(\hat{\boldsymbol{v}}^*) = \boldsymbol{G}^{R}(\hat{\boldsymbol{v}}^*)$$

such that for each such pair, $\boldsymbol{G}(\boldsymbol{v}^*) = \boldsymbol{G}^{R}(\hat{\boldsymbol{v}}^*)$. There is also such a map from $\hat{\boldsymbol{v}}^*$ to $\boldsymbol{v}^*$. Moreover, the common value x^* in the stacked vector $\boldsymbol{G}(\boldsymbol{v}^*)$ satisfies $\sum_{\mathrm{i}}(R_{\mathrm{i}}\nabla f_{\mathrm{i}}(x^*)) = 0$.

Note that $\mathbf{G}(\mathbf{v}^*)$ is formed by stacking copies of the consensus solution x^*, so this theorem says that the two formulations in (i) and (ii) have exactly the same set of consensus solutions.

Proof: Assume that $\mathbf{F}^R(\mathbf{v}^*) = \mathbf{G}(\mathbf{v}^*)$, and define x^* to be the identical entries of $\mathbf{G}(\mathbf{v}^*)$, so that $F_{\mathrm{i}}(v_{\mathrm{i}}^*) = x^*$ for each i. Applying $\mathbf{G}$ to both sides yields $\mathbf{G}(\mathbf{F}^R(\mathbf{v}^*)) = \mathbf{G}^2(\mathbf{v}^*) = \mathbf{G}(\mathbf{v}^*)$. Expanding using the definitions of F_{i}^R and $\mathbf{G}_{\mathrm{i}}$ yields

$$\frac{1}{K}\sum_{\mathrm{i}}(v_{\mathrm{i}}^* - R_{\mathrm{i}}\phi_{\mathrm{i}}(F_{\mathrm{i}}(v_{\mathrm{i}}^*))) = \frac{1}{K}\sum_{\mathrm{i}} v_{\mathrm{i}}^*. \tag{C.1}$$

Multiplying by K and canceling the sum of the v_{i}^* gives

$$\sum_{\mathrm{i}} R_{\mathrm{i}}\phi_{\mathrm{i}}(x^*) = 0. \tag{C.2}$$

Conversely given x^* such that $\sum_{\mathrm{i}} R_{\mathrm{i}}\phi_{\mathrm{i}}(x^*) = 0$, define $v_{\mathrm{i}}^* = x^* + R_{\mathrm{i}}\phi_{\mathrm{i}}(x^*)$ for all i. We will show that $\mathbf{F}^R(\mathbf{v}^*) = \mathbf{G}(\mathbf{v}^*)$. Since $F_{\mathrm{i}}^R(v_{\mathrm{i}}) = (I + R_{\mathrm{i}}\phi_{\mathrm{i}})^{-1}(v_{\mathrm{i}})$, we have

$$F_{\mathrm{i}}^R(v_{\mathrm{i}}^*) = (I + R_{\mathrm{i}}\phi_{\mathrm{i}})^{-1}(x^* + R_{\mathrm{i}}\phi(x^*)) = x^*. \tag{C.3}$$

Also,

$$\mathbf{G}(\mathbf{v}^*)_{\mathrm{i}} = \frac{1}{N}\sum_{\mathrm{i}}(x^* + R_{\mathrm{i}}\phi(x^*)) = x^* + \frac{1}{N}\sum_{\mathrm{i}} R_{\mathrm{i}}\phi_{\mathrm{i}}(x^*). \tag{C.4}$$

Note that the second term in this sum is 0 by assumption, so $\mathbf{G}(\mathbf{v}^*)_{\mathrm{i}} = x^*$ for all i and hence $\mathbf{F}^R(\mathbf{v}^*) = \mathbf{G}(\mathbf{v}^*)$.

Assume now that $\mathbf{F}(\hat{\mathbf{v}}^*) = \mathbf{G}^R(\hat{\mathbf{v}}^*)$, and let $\hat{x}^*$ be the identical entries of $\mathbf{F}(\hat{\mathbf{v}}^*)$. As before, $\mathbf{G}^R(\mathbf{F}(\hat{\mathbf{v}}^*)) = \mathbf{G}^R(\hat{\mathbf{v}}^*)$ and this with the definitions yields

$$\begin{aligned}(\textstyle\sum R_{\mathrm{i}})^{-1}(\textstyle\sum R_{\mathrm{i}}(\hat{v}_{\mathrm{i}}^* - \phi_{\mathrm{i}}(F_{\mathrm{i}}(\hat{v}_{\mathrm{i}}^*))))& \\ = (\textstyle\sum R_{\mathrm{i}})^{-1}(\textstyle\sum R_{\mathrm{i}}\hat{v}_{\mathrm{i}}^*).&\end{aligned} \tag{C.5}$$

Applying $(\sum R_{\mathrm{i}})$ to both sides, canceling $\sum R_{\mathrm{i}}\hat{v}_{\mathrm{i}}^*$, and using $F_{\mathrm{i}}(\hat{v}_{\mathrm{i}}^*) = \hat{x}^*$ gives $\sum_{\mathrm{i}} R_{\mathrm{i}}\phi_{\mathrm{i}}(\hat{x}^*) = 0$.

Conversely given $\hat{x}^*$ such that $\sum R_{\mathrm{i}}\phi_{\mathrm{i}}(x^*) = 0$, define $\hat{v}_{\mathrm{i}}^* = \hat{x}^* + \phi_{\mathrm{i}}(x^*)$ for all i. A calculation similar to the previous case shows that $\mathbf{F}(\hat{\mathbf{v}}^*) = \mathbf{G}^R(\hat{\mathbf{v}}^*)$.

Hence for each $\mathbf{v}^*$ satisfying (i), the identical entries x^* of $\mathbf{G}(\mathbf{v}^*)$ satisfy (C.2). This condition then determines $\hat{\mathbf{v}}^*$ satisfying (ii) and so that x^* is the common entry in $\mathbf{G}^R(\mathbf{v}^*)$. The map from $\hat{\mathbf{v}}^*$ to $\mathbf{v}^*$ is the same in reverse. ■

Lemma C.0.0.1 *The map F in (2.18) can be expressed as either*

1. $F(x) = (I + \nabla f)^{-1}(x)$
2. $F(x) = (I - r\nabla f)(x)$,

where $\sigma^2 = \frac{\sigma_\lambda^2}{\sigma_w^2}$, $f = \frac{\sigma^2}{2}\|y - Ax\|_2^2$ and $r = 1/(1 + \sigma^2 L^2)$.

Proof: We'll first establish the form in 1. Note that the first order optimality condition for $F(x) = \mathrm{argmin}_v \left\{ f(v) + \frac{1}{2}\|v - x\|^2 \right\}$ is $\nabla f(v) + v - x = 0$. Solving for v gives $v^* = (I + \nabla f)^{-1}(x)$. Since f is a positive, semi-definite quadratic penalty, it has a maximal monotone subdifferential, hence this inverse is well-defined by Proposition 1.

Using $\nabla f(v) = \sigma^2 A^T(Av - y)$ in the first order optimality condition above and isolating v^* gives

$$v^* = x - \sigma^2 A^T(Av^* - y). \tag{C.6}$$

Therefore, v^* is of the form $v^* = x + A^T z$ for some z. Using this form of v^* in (C.6) gives

$$x + A^T z = x - \sigma^2 A^T(A(x + A^T z) - y). \tag{C.7}$$

Some algebra and $AA^T = L^2 I$ gives

$$A^T(y - Ax) = (1 + L^2\sigma^2)A^T z, \tag{C.8}$$

with a solution of $z = \frac{\sigma^2}{1+\sigma^2 L^2}(y - Ax)$. We substitute this into $v^* = x + A^T z$ to obtain $F(x) = v^* = x + r\sigma^2 A^T(y - Ax)$, or $(I - r\nabla f)(x)$ where $r = \frac{1}{1+\sigma^2 L^2}$. ■

Lemma C.0.0.2 *Suppose* $Lip(\psi) \leq \alpha < 1$. *Then* $\Psi = I + \psi$ *is invertible and*

$$Lip(\Psi^{-1}) \leq \frac{1}{1-\alpha} \tag{C.9}$$

$$Lip(I - \Psi^{-1}) \leq \frac{\alpha}{1-\alpha} \tag{C.10}$$

Proof: If ψ is Lipschitz with constant α, then $\Psi = I + \psi$ is also Lipschitz. Consequently Ψ will be differentiable almost everywhere by Rademacher's theorem. The forward and reverse triangle inequalities imply

$$(1-\alpha)\|x - z\| \leq \|\Psi(x) - \Psi(z)\| \leq (1+\alpha)\|x - z\|. \tag{C.11}$$

These bounds show that the singular values of Ψ are bounded away from 0, which implies its Jacobian is of full rank. The Lipschitz Inverse Function Theorem [38] implies that Ψ is invertible. Defining $w = \Psi(x)$ and $v = \Psi(z)$ transforms the bounds on $\|\Psi(x) - \Psi(z)\|$ to

$$\begin{aligned}(1-\alpha)\|\Psi^{-1}(w) - \Psi^{-1}(v)\| &\leq \|w - v\| \\ &\leq (1+\alpha)\|\Psi^{-1}(w) - \Psi^{-1}(v)\|.\end{aligned} \tag{C.12}$$

Dividing the left hand side of the inequality by $1 - \alpha$ yields $Lip(\Psi^{-1}) \leq \frac{1}{1-\alpha}$.

Now fix v_1 and v_2, and let $w_j = \Psi(v_j) = v_j + \psi(v_j)$. Using the Lipschitz constants for ψ and Ψ^{-1} gives

$$\|(I - \Psi^{-1})(w_1) - (I - \Psi^{-1})(w_2)\| \tag{C.13}$$

$$= \|\Psi(v_1) - v_1 - (\Psi(v_2) - v_2))\| \tag{C.14}$$

$$= \|\psi(v_1) - \psi(v_2)\| \tag{C.15}$$

$$\leq \alpha\|v_1 - v_2\| \tag{C.16}$$

$$= \alpha\|\Psi^{-1}(w_1) - \Psi^{-1}(w_2)\| \tag{C.17}$$

$$\leq \frac{\alpha}{1-\alpha}\|w_1 - w_2\|. \tag{C.18}$$

■

Lemma C.0.0.3 *Assume $\sigma^2 < 1/L^2$ and let f and r be as in Lemma C.0.0.1 with this σ^2. Then there exist constants $\delta, C_1 > 0$ such that if R is a matrix satisfying $\|R - I\| < \delta$, then there exists a matrix $\tilde{R}$ depending on R so that*

- $\|\tilde{R} - I\| \leq C_1 \|R - I\|$
- $\tilde{R}\nabla f$ *is maximal monotone*

and so that

$$x - rR\nabla f(x) = (I + \tilde{R}\nabla f)^{-1}(x). \tag{C.19}$$

Proof: Define $W = \sigma^2 A^T A$. Note that $A^T A$ can be factored as a projection followed by scaling by L^2, hence $\|rW\| = \sigma^2 L^2/(1 + \sigma^2 L^2) < 1/(1 + \sigma^2 L^2)$ by assumption. Hence there exists $\delta_1 > 0$ so that if $\|R - I\| < \delta_1$, then $\|rWR\| < 1$, hence $(I - rWR)$ is invertible by Lemma C.0.0.2. As motivated below, let

$$\tilde{R}x = \begin{cases} rR(I - rWR)^{-1}x & \text{for } x \in \text{range}(A^T) \\ Rx & \text{for } x \in \text{null}(A). \end{cases} \tag{C.20}$$

Since the orthogonal complement of range(A^T) is null(A), we can extend by linearity to all of $\mathbb{R}^n$.

Since $AA^T = L^2 I$, induction shows that $W^k A^T = (\sigma^2 L^2)^k A^T$. Since $\sigma^2 L^2 < 1$, we can expand the first part of (C.20) at $R = I$ using a convergent power series to get

$$r(I - rW)^{-1}A^T = r\sum_{k=0}^{\infty} r^k W^k A^T \tag{C.21}$$

$$= r\sum_{k=0}^{\infty} (r\sigma^2 L^2)^k A^T \tag{C.22}$$

$$= \frac{r}{1 - r\sigma^2 L^2} A^T. \tag{C.23}$$

Since $r = 1/(1 + \sigma^2 L^2)$, this is A^T.

The same idea shows that for R sufficiently close to the identity, say $\|R - I\| < \delta_2$ for some $\delta_2 \in (0, 1)$, $\tilde{R} = \tilde{R}(R)$ restricted to range(A^T) can be written as a power series in R

and satisfies $\tilde{R}(I) = I$. Expanding this power series about $R = I$ gives constants $c_1, c_2 > 0$ so that

$$\|\tilde{R} - I\| \leq (c_1 + c_2\|R - I\|)\|R - I\|. \tag{C.24}$$

Define $C_1 = \max(1, c_1 + c_2)$. Since $\|R - I\| \leq 1$, we have

$$||\tilde{R} - I|| \leq C_1||R - I|| \tag{C.25}$$

on range(A^T), hence on all of $\mathbb{R}^n$ since $\tilde{R} = R$ on null(A).

We now show that $\tilde{R}\nabla f(x)$ is maximal monotone. Note that

$$\tilde{R}\nabla f(x) - \tilde{R}\nabla f(w) = \sigma^2 \tilde{R}A^T A(x - w), \tag{C.26}$$

so $\tilde{R}\nabla f$ is maximal monotone when $\tilde{R}A^T A$ is positive semi-definite Note that if $x \in$ null(A), then $x^T \tilde{R}A^T Ax = 0$. If $x \in$ range(A^T), then $x = A^T z$ for some $z \in \mathbb{R}^n$. Substituting this for the right-side x gives

$$x^T \tilde{R}A^T Ax = x^T \tilde{R}A^T AA^T z. \tag{C.27}$$

Since $AA^T = L^2 I$, this is $L^2 x^T \tilde{R}x$. However by reducing δ_2 if needed, (C.25) implies that $\tilde{R}$ is positive definite. Extending by linearity shows that $\tilde{R}A^T A$ is positive semi-definite, so $\tilde{R}\nabla f$ is maximal monotone. Let $\delta = \min\{\delta_1, \delta_2\}$.

Finally, let $\tilde{F}(x) = x - rR\nabla f(x)$, so $\tilde{F}(x) = (I - rR\nabla f)(x)$. Then $\tilde{F}(x) = (I + \tilde{R}\nabla f)^{-1}(x)$ exactly when

$$(I + \tilde{R}\nabla f) \circ (I - rR\nabla f)(x) = x \tag{C.28}$$

Expanding and rearranging gives

$$\tilde{R}\nabla f \circ (I - rR\nabla f) = rR\nabla f. \tag{C.29}$$

Let $p = \sigma^2 A^T y$, so that $\nabla f(x) = Wx - p$. Using this in (C.29) gives

$$\tilde{R}(Wx - rWR(Wx - p) - p) = rR(Wx - p), \tag{C.30}$$

and collecting terms gives

$$\tilde{R}(I - rWR)(Wx - p) = rR(Wx - p). \tag{C.31}$$

Since $Wx - p$ maps $\mathbb{R}^n$ onto range(A^T), this is equivalent to $\tilde{R}(I - rWR) = rR$ on range(A^T), which is consistent with (C.20). Hence $\tilde{R}$ as defined in (C.20) satisfies (C.19), thus completing the proof. ■

Lemma C.0.0.4 *Let f be as in Lemma C.0.0.1 and let ϕ be a Lipschitz and strongly maximal monotone function with Lipschitz constant k. Then there exists a constant $\delta > 0$ such that if R is a matrix satisfying $\|R - I\| < \delta$, then $\tilde{R}^{-1}\phi$ is maximal monotone, where the matrix $\tilde{R}$ is from Lemma C.0.0.3. Moreover, there exists a function Φ_R and constant C such that*

$$Lip(\Phi_R - I) \leq C\|R - I\|$$

and so that

$$(I + \phi)^{-1} \circ \Phi_R = (I + \tilde{R}^{-1}\phi)^{-1}.$$

Proof: It suffices to show $\Phi_R(x) = (I + \phi)(I + \tilde{R}^{-1}\phi)^{-1}(x)$ has the desired properties. We add and subtract ϕ and factor out $(I + \phi)^{-1}$ to obtain

$$\Phi_R = (I + \phi)(I + \phi + (\tilde{R}^{-1} - I)\phi)^{-1} \tag{C.32}$$

$$= (I + \phi)[(I + (\tilde{R}^{-1} - I)\phi(I + \phi)^{-1})(I + \phi)]^{-1} \tag{C.33}$$

$$= (I + (\tilde{R}^{-1} - I)\phi(I + \phi)^{-1})^{-1}. \tag{C.34}$$

Hence $\Phi_R = (I+\psi)^{-1}$ with $\psi = (\tilde{R}^{-1}-I)\phi(I+\phi)^{-1}$. By Lemma C.0.0.3, $\|\tilde{R}-I\| \leq C_1\|R-I\|$. Restrict R so that this is less than 1, so by Lemma C.0.0.2 with $\Psi = \tilde{R}$,

$$\|\tilde{R}^{-1} - I\| \leq \frac{C_1\|R-I\|}{1-C_1\|R-I\|} \leq C_2\|R-I\| \tag{C.35}$$

where $C_2 = C_1/(1-C_1\|R-I\|)$. Let $d_1 = Lip((I+\phi)^{-1})$ which is at most 1 since the resolvent of a monotone operator is nonexpansive [37]. Then

$$Lip(\psi) \leq C_2 k d_1 \|R-I\|. \tag{C.36}$$

Hence there exists $\delta_1 > 0$ such that if $\|R-I\| < \delta_1$, then $Lip(\psi) < 1$, in which case Lemma C.0.0.2 implies $\Phi_R = (I+\psi)^{-1}$ is well-defined with $Lip(\Phi_R) \leq 1/(1-Lip(\psi))$. Lemma C.0.0.2 with $\Psi = I+\psi = \Phi_R^{-1}$ implies

$$Lip(I-\Phi_R) = Lip(I-\Psi^{-1}) \leq \frac{Lip(\psi)}{1-Lip(\psi)}. \tag{C.37}$$

By (C.36), we can choose $\delta > 0$ so that if $\|R-I\| < \delta$, then $Lip(I-\Phi_R) \leq C\|R-I\|$, where $C = 2C_2kd_1$.

Recall that ϕ strongly monotone means that there exists $m > 0$ so that for all x, v, $(x-v)^T(\phi(x)-\phi(v)) \geq m||x-v||_2^2$. Let $\eta = \tilde{R}^{-1} - I$. By (C.35), we can reduce δ to get $\|\eta\| < \frac{m}{2k}$. To show that $\tilde{R}^{-1}\phi$ is maximal monotone, note that

$$\begin{aligned}&(x-v)^T(\tilde{R}^{-1}\phi(x) - \tilde{R}^{-1}\phi(v)) \\ &\quad = (x-v)^T(\phi(x)-\phi(v)) - (x-v)^T(\eta(\phi(x)-\phi(v))\end{aligned} \tag{C.38}$$

By assumption, the first term of the sum is bounded below by $m||x-v||_2^2$. Additionally $\|\phi(x)-\phi(v)\| \leq k\|x-v\|$ by assumption, so

$$(x-v)^T(\tilde{R}^{-1}\phi(x) - \tilde{R}^{-1}\phi(v)) \geq m||x-v||_2^2 - k\|\eta\|\|x-v\|_2^2, \tag{C.39}$$

which gives a lower bound of $\frac{m}{2}\|x-v\|_2^2$. Hence $\tilde{R}^{-1}\phi$ is strongly monotone, and since $\tilde{R}^{-1}$ is linear, $\tilde{R}^{-1}\phi$ is maximal monotone. ■

Theorem C.0.0.2 *Suppose ϕ is a Lipschitz and strongly maximal monotone function and let $H = (I+\phi)^{-1}$ and $\mu_1 = \mu_2 = 1/2$. Assume $\sigma^2 < 1/L^2$. There exist $\alpha > 0$ and $C > 0$ such that if R is a matrix satisfying $RA^T = B$ and $\|R - I\| \leq \alpha < 1$, there exists Φ_R a Lipschitz map depending on H and R with $Lip(\Phi_R - I) \leq C\|R-I\|$ such that the following two choices lead to the same set of solutions x^* in (2.23):*

- $\boldsymbol{F}_1 = \tilde{F}$ *is the RAP update in (4.1) and* $\boldsymbol{F}_2 = H$*;*
- $\boldsymbol{F}_1 = F$ *is the standard update in (2.18) and* $\boldsymbol{F}_2 = H \circ \Phi_R$*.*

This theorem says that the effect of RAP with a denoiser H can be explained by using the standard data-fitting term together with a modified denoiser defined by a Lipschitz transformation of the image domain followed by the original denoiser.

Proof: In order to apply Theorem 1, we verify that each $\mathbf{F}_\mathrm{j}$ is of the form $(I+\omega)^{-1}$ for ω a maximal monotone function. By Lemma C.0.0.1, we have that $F = (I+\nabla f)^{-1}$ and ∇f is maximal monotone. As ϕ is maximal monotone, H is of the desired form, and H is well-defined by Proposition C.0.0.1. Since $\sigma^2 < 1/L^2$, Lemma C.0.0.3 with the same f, r as defined in Lemma C.0.0.1 implies that $\tilde{F} = (I + \tilde{R}\nabla f)^{-1}$. Lemma C.0.0.3 also gives that $\tilde{R}\nabla f$ is maximal monotone. Finally as ϕ is assumed to be a Lipschitz and strongly maximal function, Lemma C.0.0.4 gives that $H \circ \Phi_R = (I + \tilde{R}^{-1}\phi)^{-1}$ is well-defined and that $\tilde{R}^{-1}\phi$ is maximal monotone. Thus we may apply the results of Theorem 1 to each pair $(\tilde{F}, H)$ and $(F, H \circ \Phi_R)$.

From the proof of Theorem 1, since $\tilde{F} = (I + \tilde{R}\nabla f)^{-1}$ and $H = (I+\phi)^{-1}$ we see that $\mathbf{v}^* = (v_1^*, v_2^*)$ is a solution of

$$\begin{bmatrix} \tilde{F}(v_1^*) \\ H(v_2^*) \end{bmatrix} = \mathbf{G}(\mathbf{v}^*)$$

if and only if

$$\tilde{R}\nabla f(x^*) + \phi(x^*) = 0,$$

where x^* is the common entry of the stacked vector $\mathbf{G}(\mathbf{v}^*)$. Since $\tilde{R}$ is invertible, this is equivalent to

$$\nabla f(x^*) + \tilde{R}^{-1}\phi(x^*) = 0.$$

Since $F = (I + \nabla f)^{-1}$ and $H \circ \Phi_R = (I + \tilde{R}^{-1}\phi)^{-1}$, again the proof of Theorem 1 implies that this is if and only if $\tilde{\mathbf{v}}^* = (\tilde{v}_1^*, \tilde{v}_2^*)$ is a solution of

$$\begin{bmatrix} F(\tilde{v}_1^*) \\ H(\Phi_R(\tilde{v}_2^*)) \end{bmatrix} = \mathbf{G}(\tilde{\mathbf{v}}^*)$$

with x^* the common entry of the stacked vector $\mathbf{G}(\tilde{\mathbf{v}}^*)$.

This implies that the two formulations have the same set of consensus solutions, x^*. ■

Proposition C.0.0.2 *If A is an $n \times n$ symmetric matrix with eigenvalues in $(0, 1]$ and $b \in \mathbb{R}^n$, then the mapping $F(x) = Ax + b$ is a proximal map.*

Proof: The conditions on A imply that $A^{-1} - I$ is symmetric and positive semidefinite, so the Cholesky decomposition gives an invertible R such that $R^T R = \frac{1}{\sigma^2}(A^{-1} - I)$ (for specified $\sigma^2 > 0$). Define p so that $\sigma^2 A R^T p = b$ and consider the proximal map defined by

$$\underset{u}{\operatorname{argmin}} \left\{ \frac{1}{2}||Ru - p||^2 + \frac{1}{2\sigma^2}||u - x||^2 \right\}. \tag{C.40}$$

The first-order optimality condition yields

$$R^T(Ru^* - p) + \frac{1}{\sigma^2}(u^* - x) = 0, \tag{C.41}$$

and gathering the u^* terms and multiplying by σ^2 gives

$$(I + \sigma^2 R^T R)u^* = x + \sigma^2 R^T p. \tag{C.42}$$

Noting that $(I + \sigma^2 R^T R) = A^{-1}$ and using the choice of p gives $u^* = Ax + b$. Hence $F(x) = Ax + b$ is a proximal map. ■

Proposition C.0.0.3 *Let $F(x) = Wx + q$ where $W = V\Lambda V^{-1}$ with Λ diagonal having eigenvalues in $(0, 1]$ and $q \in \mathbb{R}^N$. For $x \in \mathbb{R}^N$ define $\hat{x} = V^{-1}x$. Then $\hat{F}(\hat{x}) = V^{-1}F(V\hat{x})$ is a proximal map in the coordinates $\hat{x}$.*

Proof: Expanding $\hat{F}$ using $W = V\Lambda V^{-1}$ gives $\hat{F}(\hat{x}) = \Lambda\hat{x} + V^{-1}q$. Since Λ is diagonal with eigenvalues in $(0, 1]$, the previous proposition implies that $\hat{F}(\hat{x})$ is a proximal map. ■

Theorem C.0.0.3 *Let F and $\hat{F}$ be as in Proposition C.0.0.3. Let H be a denoiser such that $\hat{H}(\hat{x}) = V^{-1}H(V\hat{x})$ is nonexpansive in the coordinates $\hat{x}$. Then the PnP algorithm converges using the operators F and H.*

Proof: An expansion of **F** and **G** shows that Algorithm 1 with two operators is equivalent to the standard PnP algorithm of [16]. By Proposition C.0.0.3, $\hat{F}$ is a proximal map, and by assumption, $\hat{H}$ is nonexpansive. Hence by [17], Algorithm 1 using operators $\mathbf{F}_1 = \hat{F}$ and $\mathbf{F}_2 = \hat{H}$ converges to a fixed point.

The bilinear change of variables $\hat{x} = V^{-1}x$ yields a one-to-one correspondence

$$\hat{F}(\hat{x}) = V^{-1}F(V\hat{x}) \tag{C.43}$$

$$\hat{H}(\hat{x}) = V^{-1}H(V\hat{x}) \tag{C.44}$$

Applying this to each component and map in Algorithm 1 produces a shadow sequence equivalent to running Algorithm 1 with $\mathbf{F}_1 = F$ and $\mathbf{F}_2 = H$. This shadow sequence converges by continuity of V and V^{-1}, so the PnP algorithm converges using F and H. ■

D. RELEVANT CODE

All code is publicly available at https://github.com/emmajreid/MDF.

A^T function

```
import numpy as np
import torch
import cv2
from skimage.io import imread
import scipy

#This code performs upsampling by block replication in L x L grids.
#This maps an N x N image to an NL x NL image.

def ATgen(imagearr, L):

    outarr = np.zeros((imagearr.shape[0]*L, imagearr.shape[1]*L))
    for i in range(0,imagearr.shape[0]):
        for j in range(0, imagearr.shape[1]):
            outarr[L*i:L*(i+1), L*j:L*(j+1)] = imagearr[i,j]

    return outarr
```

A function

```
import torch
import cv2
import numpy as np

#This code performs image decimation by taking the sum of the pixel values
#in a L x L grid. From a N x N image, it returns an N/L x N/L image.
```

```
class Conv2D:

    def __init__(self, in_channel, o_channel, kernel_size, stride, mode):
        self.i = in_channel
        self.out = o_channel
        self.ker = kernel_size
        self.stride = stride
        self.mode = mode

    def forward(self, input_image, L):
        #Here using the // operator under the assumption that we'll only be applying
        #kernels that result in no empty columns.
        m = self.ker
        ##Just K1
        if self.out == 1 and self.mode == 'known':
            kernel = torch.FloatTensor(torch.ones((L,L)))
            kernel_list = [kernel]
            kernel = torch.stack(kernel_list)
        [M,N] = input_image.size()
        stride = self.stride
        [n,p,q] = kernel.size()
        temp = torch.tensor([0])
        out_image = torch.zeros((M-m)//stride+1,(N-m)//stride+1, n)

        for i in range(0,M-m+1,stride):
            for j in range(0,N-m+1,stride):
                for l in range(0,n):
        #Multiply the input image with the kernel
                        sub_img = input_image[i:i+m,j:j+m]
                        temp = torch.FloatTensor([torch.sum(sub_img * kernel[l,:])])
```

```
                        out_image[i//stride,j//stride] =
                        out_image[i//stride,j//stride]+(temp)
                        temp=torch.tensor([0])
        return out_image

def Agen(imagearr, L):
    img = torch.from_numpy(imagearr)
    img = img.float()
    params = Conv2D(1,1,L,L,'known')
    out_image= params.forward(img,L)
    out_image = out_image[:,:,0]
    return out_image.numpy()
```

PSNR function

```
from math import log10
from math import sqrt
import numpy as np

#This code computes the Peak Signal to Noise Ratio.
#PSNR assumes that gt and recon are 2D image arrays in the range [0,1].

def mse(groundtrutharr, imarr):
    imvec = imarr.reshape(-1)
    groundtruthvec = groundtrutharr.reshape(-1)
    mse = np.mean(np.power(imvec-groundtruthvec,2))
    rmse = sqrt(mse)
    nrmse = sqrt(mse)/(np.mean(np.power(groundtruthvec,2)))
    return [mse, rmse, nrmse]

def error(arr1, arr2):
```

```
    #Error (x,v)
    vec1 = arr1.reshape(-1)
    vec2 = arr2.reshape(-1)
    err = sqrt(np.sum(np.power(vec2-vec1,2)))
    errnorm = err/sqrt(np.sum(np.power(vec2,2)))
    return errnorm

def psnr(gt,recon):
    MSE = mse(gt,recon)[0]
    out = 10*log10(1/MSE)
    return out
```

MACE framework with MDF

```
# General imports
import sys
import os
import numpy as np
import argparse
import cv2

from utils import conv2d
from utils import atranspose
from utils import psnr

import torch
from models.network_dncnn import DnCNN as net

def mace(LR,numagents,c,args):
    '''
```

```
    This is a Python implementation of the MACE framework written about in 4D X-
Ray CT Reconstruction using Multi-Slice Fusion by S. Majee et. al (2019).
    Fundamentally it consists of stacked application of the prior and forward models
    to our image vectors (L) followed by a weighted averaging (G) and an update step.

    Inputs:
        LR: Low resolution image scaled to be in [0,1]
        args: Dictionary containing command line arguments
            SRval: Super resolution factor
            beta: User-defined parameter for calculating \sigma^2_\lambda, our
            Lagrangian parameter, and c.
              This parameter plays a role when sigy is assumed to be nonzero.
            iterstop: User-defined parameter for the stopping number of Mann iterations
            in the MACE framework.
            sign: Noise level that the denoising prior is trained to remove.
            For all provided prior models,sign = 0.1.
            sigy: Assumed level of noise in the ground truth image
            mu:   This is the weight of the forward model in the MACE framework,
            in [0,1]
               mu = 0 -----> Only considering the output of the prior model.
               mu = 0.5 ---> Equal consideration of the outputs.
               mu = 1 -----> Only considering the output of the forward model.
            rho: This is the step size that we take throughout the MACE framework,
            generally in (0,1).
             Larger rho tends to lead to faster convergence.
            model_dir: Path to the directory containing the saved priors.
            model_name: Name of the denoising prior model.
            hrname: Name of the HR ground truth image.
            forwards: Choice of forward model, either using the standard or RAP.
            denoisers: Choice of denoising prior model.
```

```
Outputs:
    OutW: Super resolved image
    PSNR: Array containing the PSNR values for each step of the algorithm.
    MACE: Array containing the MACE errors for each step of the algorithm.
'''

# Generate initial guess as starting point for algorithm, write to an image,
# and use it to initialize X and W.

init = cv2.resize(LR,None, fx = args.SRval, fy = args.SRval,
interpolation = cv2.INTER_CUBIC)

mdim,ndim = init.shape
cv2.imwrite('images/results/init.png', init*255)
init = init.reshape(-1)
init = init.reshape(init.shape[0],1)
X = np.tile(init, (1,numagents))
W = np.copy(X)

# Initialize vectors for metric analysis.
PSNR = np.zeros(args.iterstop)
maceerr = np.zeros(args.iterstop)

# MACE Framework
for i in range(0, args.iterstop):
    print("Currently on Iteration", i)
    X = L(W,X,numagents, mdim,ndim,c, LR, args)
    Z = G(2*X-W,numagents,args.mu)
    W = W+2*args.rho*(Z-X)
```

```
     # Save metrics for this iteration to the vector.
        PSNR[i] = psnr.psnr(gt/255,G(W,numagents,args.mu)[:,0].reshape(mdim,ndim))

        Y=np.copy(W)
        maceerr[i]=(1/args.sign)*np.linalg.norm(G(Y,numagents,args.mu)-L(Y,Y,numagents,
        mdim, ndim, c, LR, args))/np.linalg.norm(G(Y,numagents,args.mu))

    return G(W,numagents,args.mu)[:,0], PSNR, maceerr

#L takes in all of the state vectors and applies denoisers to the first k state
#vectors and the forward model to the last state vector.

def L(W,X,numagents, mdim,ndim,c, LR, args):
    # Prior Model Application
    Lout = np.copy(X)
    for i in args.denoisers:
        iternum = 0
        xi =np.copy(W[:,iternum])
        xi = xi.reshape(mdim,ndim)

        if (args.model_name == 'dncnn_25.pth'):
            denoiser = net(in_nc=1, out_nc=1, nc=64, nb=17, act_mode='R')

        else:
            denoiser = net(in_nc=1, out_nc=1, nc=64, nb=17, act_mode='BR')
        denoiser.load_state_dict(torch.load(os.path.join(args.model_dir,
        args.model_name)), strict=True)
        denoiser.eval()

        xi = torch.FloatTensor(xi)
```

```
        if torch.cuda.is_available()==True:
            denoiser.cuda()
        if torch.cuda.is_available() == True:
            imagei = (xi.cuda()).reshape(1,1,mdim,ndim)
        else:
            imagei = xi.reshape(1,1,mdim,ndim)
        x = denoiser(imagei)
        if torch.cuda.is_available() == True:
            x = x.cpu().detach().numpy()
        else:
            x = x.detach().numpy()
        if torch.cuda.is_available():
            torch.cuda.synchronize()
        x = x.reshape(mdim*ndim)
        Lout[:,iternum] = np.copy(x)

        iternum += 1
    wi =np.copy(W[:,iternum])
    wi = wi.reshape(mdim,ndim)

    #Forward Model Application
    x = forward(wi, LR, c, args)
    x = x.reshape(mdim*ndim)
    Lout[:,iternum] = np.copy(x)
    return Lout

# G outputs the weighted average of the state vectors, weighting the Forward Model's
vector with mu and all others with (1-mu)/(N-1)
# where N is the number of agents.
def G(X,numagents,mu):
```

```
    if numagents ==2:
        x = mu*X[:,numagents-1,np.newaxis]+
        ((1-mu)/(numagents-1))*np.sum(X[:,0:numagents-1,np.newaxis],1)
    else:
        x = mu*X[:,numagents-1]+(1-mu)/((numagents-1))*np.sum(X[:,0:numagents-
1],1)
    x = x.reshape(x.shape[0],1)
    Xnew = np.tile(x, (1,numagents))
    return Xnew

def forward(wi, LR,c,args):

    new = LR-(1/args.SRval**2)*conv2d.Agen(wi,args.SRval)

    # 0 is for AT, 1 for bicubic
    for i in args.forwards:
        if i==0:
            filt = atranspose.ATgen(new,args.SRval)
        elif i==1:
            filt = cv2.resize(new,None,fx=args.SRval,fy=args.SRval,
            interpolation=cv2.INTER_CUBIC)
        else:
            print("Invalid filter choice")
            break
    out = wi+c*filt
    if args.sigy==0:
        out = np.clip(out, a_min=0, a_max=None)
    return out

if __name__ == '__main__':
```

```
'''
Variable Definitions:
    beta: User-defined parameter for calculating \sigma^2_\lambda,
    our Lagrangian parameter, and c.
          This parameter plays a role when sigy is assumed to be nonzero.
    iter: User-defined parameter for the number of Mann iterations in
    the MACE framework.
          Complete convergence is usually achieved by 200 iterations.
    sign: Noise level that the denoising prior is trained to remove.
    For all provided prior models,sign = 0.1.
    sigy: Assumed level of noise in the ground truth image
    mu:   This is the weight of the forward model in the MACE framework, in [0,1]
          mu = 0 -----> Only considering the output of the prior model.
          mu = 0.5 ---> Equal consideration of the outputs.
          mu = 1 -----> Only considering the output of the forward model.
    rho: This is the step size that we take throughout the MACE framework,
    generally in (0,1).
         Larger rho tends to lead to faster convergence.
'''

parser = argparse.ArgumentParser(description="Gather P&P input parameters.")
parser.add_argument('--SRval', type=int, default=4,help='Super-Resolution factor')
parser.add_argument('--beta', type=float, default= 0.5,
help='Regularization factor')
parser.add_argument('--iterstop', type=int, default=1,
help='Number of iterations to run')
parser.add_argument('--sign', type=float,default=0.1,
help='Noise level trained to remove')
parser.add_argument('--sigy', type=float, default=0,
```

```
help='Noise level in image')
parser.add_argument('--mu', type=float, default=0.5,
help='Weighting factor')
parser.add_argument('--rho', type=float, default=0.5,
help='Convergence factor')

parser.add_argument('--model_dir', default=os.path.join('priors'),
help='directory of the model')

#Options for model names are dncnn_25.pth and MDF.pth
parser.add_argument('--model_name', default='nano.pth', type=str,
help='the model name')
parser.add_argument('--hrname', default='nanotest.png', type=str,
help='the HR image name')

# Choices for forward and prior models should be entered as arrays
separated by commas
# Currently there is only one option for a prior model,
but this will be updated in the future.
parser.add_argument('forwards', nargs = '*', default = [1])
# 0 is for AT, 1 for bicubic
parser.add_argument('denoisers', nargs = '*', default = [1])

# Read in the user-defined arguments.
args = parser.parse_args()

# Name of high-resolution ground truth.
sys.stdout.flush()
base_path = os.path.dirname(os.path.relpath(__file__))
testimg_file = os.path.join(base_path, 'images/')
```

```
lrimg_file = os.path.join(base_path, 'images/LR images')
resultsimg_file = os.path.join(base_path, 'images/results')

lrname = 'lrL='+str(args.SRval)+str(args.hrname)+'noise'+str(args.sigy)+'.png'

# Read in the high-resolution ground truth.
gt = cv2.imread(os.path.join(testimg_file,args.hrname),0)/255

# Initialize synthetic low-resolution image.

if os.path.exists('LR Images/lrL='+str(args.SRval)+str(args.hrname)+'noise'
+str(args.sigy)+'.png'):
    lr = cv2.imread('LR Images/lrL='+str(args.SRval)+str(args.hrname)+'noise'+
    str(args.sigy)+'.png',0)/255
else:
    lr = conv2d.Agen(gt,args.SRval)/(args.SRval*args.SRval)+
    np.random.normal(0,args.sigy,(gt.shape[0]//args.SRval,gt.shape[1]//args.SRval))
lr = np.float64(lr)
cv2.imwrite(os.path.join(lrimg_file, lrname), lr*255)
LR = cv2.imread(os.path.join(lrimg_file, lrname), 0)/255

# Initialize other parameters.
varn = args.sign**2
vary = args.sigy**2
varlam = varn/args.beta
c = varlam/(vary/args.SRval + varlam)
numagents = len(args.denoisers) + len(args.forwards)

# Run the MACE algorithm.
outW, PSNR, maceerr= mace(LR,numagents, c, args)
```

```
cv2.imwrite(os.path.join(resultsimg_file,str(args.iterstop)+
'iters-DnCNNmaceout'+str(args.mu)+'.noise'+str(args.sign)+'.png'),
outW.reshape(LR.shape[0]*args.SRval,LR.shape[0]*args.SRval)*255)

# Load images for metric purposes
gt = cv2.imread(os.path.join(testimg_file,args.hrname),0)
srx = cv2.imread(os.path.join(resultsimg_file,str(args.iterstop)+
'iters-DnCNNmaceout'+str(args.mu)+'.noise'+str(args.sign)+'.png'),0)
print("Path to the reconstruction is: ", resultsimg_file)



# Calculate PSNR for the final reconsruction and return our convergence metric.
ps = psnr.psnr(gt/255,srx/255)
print("PSNR for Our Reconstruction: ", ps)
print("MACE Error for Our Reconstruction: ", maceerr[-1])

#Display our reconstruction
cv2.imshow('Reconstruction', srx)
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.waitKey(1)
```

VITA

EDUCATION:

P.h.D, Applied Mathematics (Expected Aug 2021) **GPA: 3.7/4.0**

Purdue University

B.S. Mathematics (May 2015) **GPA: 3.81/4.0**

University of Nebraska - Lincoln

ACADEMIC POSITIONS:

Research Assistant August 2017 - Present

- Worked in tandem with the Mathematics and Electrical Engineering departments in researching methods in fluorescence microscopy and applications to neural networks.

Teaching Assistant August 2015 - August 2017

- Instructed for Calculus I and II, Applied Calculus, and Differential Equations.
- Wrote quizzes and exams for the various courses, in addition to working in the help room.

Undergraduate Coordinator of All Girls All Math March 2015 - August 2015

- Planned 2 week-long summer camps for girls interested in mathematics.
- Served as a teaching assistant for a cryptography course, covering such topics as modular arithmetic and RSA.

Undergraduate Learning Assistant January 2014 - May 2015

- Assist in teaching college algebra curriculum to undergraduate students.
- Work collaboratively with a graduate instructor to develop strategies to improve the course.

Athletic Tutor January 2013 - May 2015

- Worked with student athletes to deepen their understanding of coursework.

- Completed CRLA's International Tutor Training Program Certification to become a Certified Tutor, Level 1.

PROFESSIONAL EXPERIENCE:

Autonomy Technology Research Center Summer Intern May 2020 - Aug. 2020

- Continued development of algorithmic and deep learning strategies to accomplish super resolution on general microscopy images.

Autonomy Technology Research Center Summer Intern May 2019 - Aug. 2019

- Developed algorithmic and deep learning strategies to accomplish super resolution on bacterial biofilms.
- Collaborated with multiple branches of the Air Force Research Lab to fuse methodologies from biology and electrical engineering.

Langley Aerospace Research Student Scholars Program June 2014 - Aug. 2014

- Continued research from 2013, specifically towards model validation and verification.
- Performed error estimation of the National Transonic Facility test section temperature map using experimental test data.

Langley Aerospace Research Student Scholars Program June 2013 - Aug. 2013

- Developed methodology for multi-fidelity data fusion for use in the National Transonic Facility during model testing and tunnel characterization.
- Developed a composite temperature profile map to predict the state of the test section temperature distribution.

ACADEMIC HONORS:

- Best Graduate Presentation at ATRC Summer Review Summer 2020
- PEO Indiana Chapter Nominee for the PEO Scholar Award Selected Fall 2019

- Best Graduate Poster at ATRC Summer Review Summer 2019
- Accepted to Purdue's Computational Interdisciplinary Graduate Program Spring 2019
- Received the Excellence in Teaching Award from the Department of Mathematics Fall 2018
- PEO Indiana Chapter Nominee for the PEO Scholar Award Selected Fall 2018
- Mervin L. Keedy Scholarship (Purdue) Awarded Spring 2015
- Regents Scholarship (UNL) Awarded Fall 2011
- D & F Eastmann Scholarship (UNL) Awarded Fall 2013
- Dean's List, College of Arts & Sciences (UNL) Fall 2012 - Spring 2015

PUBLICATIONS:

- Submitted:

  Multi-Resolution Data Fusion for Super Resolution Imaging of Biological Materials

PRESENTATIONS:

- Invited speaker at the Computational Interdisciplinary Graduate Programs - Computational Science and Engineering's Spring Symposium
- Invited speaker at Oak Ridge National Lab's seminar March 2021
- Invited speaker at the Air Force Research Lab's MachIne And Computational Learning Exploration (MIrACLE) seminar February 2021
- Invited speaker at Argonne National Lab's seminar February 2021
- Accepted as a Presenter for the 2021 Electronic Imaging Conference Winter 2020

- Accepted as a Presenter for the 2020 SIAM Conference on Imaging Science Summer 2020
- Invited speaker at Air Force Research Lab's biweekly Bio-RT meeting Winter 2019
- Accepted as a Presenter for the 2020 Electronic Imaging Conference Winter 2019

LEADERSHIP AND INVOLVEMENT:

- Reviewer for IEEE Transactions on Image Processing Summer 2019 - present
- Graduate Student Representative for College of Science Grade Appeals Fall 2019 - present
- Department Senator in Purdue Graduate Student Government Fall 2018 - Spring 2019
- Graduate Representative for the Purdue Department of Mathematics Fall 2017 - Spring 2018
- Pi Mu Epsilon - Nebraska Alpha Chapter, President Fall 2014 - Spring 2015
- Math Club - President Fall 2014 - Spring 2015
- American Institute of Aeronautics and Astronautics, Student Chapter Member Fall 2012 - Spring 2015
- Alpha Delta Pi Sorority - Executive Committee Member Fall 2014

SKILLS:

Programming Languages: Python, MATLAB, Julia, C, LaTeX