Integrating Physical Models and Deep Priors for Computational Imaging

Yu Sun
Washington University in St. Louis

Follow this and additional works at: https://openscholarship.wustl.edu/eng_etds

Part of the Computer Sciences Commons

Recommended Citation
https://openscholarship.wustl.edu/eng_etds/802

This Dissertation is brought to you for free and open access by the McKelvey School of Engineering at Washington University Open Scholarship. It has been accepted for inclusion in McKelvey School of Engineering Theses & Dissertations by an authorized administrator of Washington University Open Scholarship. For more information, please contact digital@wumail.wustl.edu.
Integrating Physical Models and Deep Priors for Computational Imaging
by
Yu Sun

A dissertation presented to
the McKelvey School of Engineering
of Washington University in
partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

August 2022
St. Louis, Missouri
Table of Contents

List of Figures ........................................................................................................ viii
List of Tables .......................................................................................................... xxii
Acknowledgments .................................................................................................. xxiv
Abstract .................................................................................................................. xxvii

I Introduction ......................................................................................................... 1

Chapter 1: Introduction ................................................................................................. 2
  1.1 Main Contributions ............................................................................................. 4

Chapter 2: Background ................................................................................................. 7
  2.1 Imaging Inverse Problems ................................................................................ 8
      2.1.1 Forward Model ....................................................................................... 8
      2.1.2 Prior & Bayesian Inference ................................................................. 12
  2.2 Computational Algorithms for Imaging............................................................ 15
      2.2.1 Model-based Methods ........................................................................ 16
      2.2.2 Learning-based Methods .................................................................... 21
      2.2.3 Discussion ............................................................................................ 25
  2.3 Advances in Fusing Physics and Learning....................................................... 26
      2.3.1 Iterative Methods using Learning Priors ............................................. 26
      2.3.2 Internal Learning/Neural Representation Learning ......................... 28
  2.4 Monotone Operator Theory .............................................................................. 30
      2.4.1 Properties of Monotone Operators ................................................ 30
      2.4.2 Properties of Block-Coordinate Monotone Operators ....................... 35
      2.4.3 Operator Properties for Convex Function, Subdifferentials, and Proxi-
mal Operators ......................................................................................... 38
# II Plug-and-Play Priors

Chapter 3: Overview ................................................................................ 47

3.1 From Proximal Operator to Image Denoiser ........................................ 47

3.2 Challenges ....................................................................................... 50

Chapter 4: Online Plug-and-Play Proximal Gradient Method .................... 52

4.1 Introduction ..................................................................................... 52

4.2 Related Work ................................................................................... 53

4.3 Batch PnP-PGM .............................................................................. 54

4.4 Online PnP-PGM ............................................................................. 58

4.5 Numerical Simulations ....................................................................... 62

4.5.1 Diffraction tomography ............................................................. 62

4.5.2 Convergence of PnP-OPGM ...................................................... 65

4.5.3 Benefits of online processing ...................................................... 67

4.6 Summary ........................................................................................ 71

Chapter 5: Regularized Fourier Ptychography using Online Plug-and-Play PGM.................................................................................................... 72

5.1 FPM as an inverse problem ............................................................... 73

5.2 Numerical Validation ....................................................................... 74

5.2.1 Experimental Setup ..................................................................... 75

5.2.2 Benefits of PnP-OPGM ............................................................ 75

5.2.3 Validation on real data ............................................................. 78

5.3 Summary ........................................................................................ 79

Chapter 6: Incremental Plug-and-Play Alternating Direction Method of Multipliers ........................................................... 80

6.1 Introduction ..................................................................................... 81

6.2 Related Work ................................................................................... 82

6.3 Incremental PnP-ADMM .................................................................... 83

6.4 Theoretical Analysis ......................................................................... 86
III Regularization by Denoising

Chapter 7: Overview

7.1 Development of RED

7.2 Challenges

Chapter 8: Block Coordinate Regularization by Denoising

8.1 Introduction

8.2 Block Coordinate RED

8.3 Convergence Analysis and Compatibility with Proximal Optimization

8.3.1 Fixed Point Convergence of BC-RED

8.3.2 Convergence for Proximal Operators

8.4 Numerical Validation

8.5 Summary

Chapter 9: Asynchronous Block Parallel Stochastic Regularization by Denoising

9.1 Introduction

9.2 Asynchronous RED

9.2.1 Async-RED using Batch Gradient

9.2.2 Async-RED using Stochastic Gradient

9.3 Convergence Analysis of Async-RED

9.4 Numerical Validation

9.4.1 Convergence Behavior

9.4.2 Effectiveness for Computational Imaging
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Introduction</td>
<td>141</td>
</tr>
<tr>
<td>10.2</td>
<td>Proposed Method</td>
<td>145</td>
</tr>
<tr>
<td>10.2.1</td>
<td>Forward Model</td>
<td>146</td>
</tr>
<tr>
<td>10.2.2</td>
<td>Radial Encoding</td>
<td>148</td>
</tr>
<tr>
<td>10.2.3</td>
<td>Network Details</td>
<td>150</td>
</tr>
<tr>
<td>10.3</td>
<td>Experiment</td>
<td>151</td>
</tr>
<tr>
<td>10.3.1</td>
<td>Setup</td>
<td>151</td>
</tr>
<tr>
<td>10.3.2</td>
<td>Results</td>
<td>154</td>
</tr>
<tr>
<td>10.4</td>
<td>Summary</td>
<td>161</td>
</tr>
<tr>
<td>11.1</td>
<td>Introduction</td>
<td>163</td>
</tr>
<tr>
<td>11.2</td>
<td>Coordinate-based Internal Learning</td>
<td>165</td>
</tr>
<tr>
<td>11.2.1</td>
<td>Measurement-field encoding with MLP</td>
<td>165</td>
</tr>
<tr>
<td>11.2.2</td>
<td>Image reconstruction in CoIL</td>
<td>169</td>
</tr>
<tr>
<td>11.3</td>
<td>Numerical Validations</td>
<td>175</td>
</tr>
<tr>
<td>11.3.1</td>
<td>Sparse view CT and experimental setup</td>
<td>175</td>
</tr>
<tr>
<td>11.3.2</td>
<td>Effectiveness of the FFM layer</td>
<td>177</td>
</tr>
<tr>
<td>11.3.3</td>
<td>Evaluation of sinogram interpolation</td>
<td>181</td>
</tr>
<tr>
<td>11.3.4</td>
<td>Evaluation of reconstruction performance</td>
<td>182</td>
</tr>
<tr>
<td>11.4</td>
<td>Summary</td>
<td>183</td>
</tr>
<tr>
<td>12.1</td>
<td>Summary</td>
<td>186</td>
</tr>
<tr>
<td>12.2</td>
<td>Future Work</td>
<td>189</td>
</tr>
</tbody>
</table>
Appendix A: Supplement for Chapter 4
A.1 Proof of Proposition 4.1 [210]
A.2 Proof of Proposition 4.2 [211]
A.3 Proof of Proposition 4.3 [212]
A.4 Proof of Proposition 4.4 [213]
A.5 Proof of Proposition 4.5 [215]
A.6 List of Selected Hyperparameters [218]

Appendix B: Supplement for Chapter 6
B.1 Convergence Analysis of IPA [221]
B.2 Analysis of IPA for Strongly Convex Functions [226]
B.3 Fixed Point Interpretation [228]
B.4 Convergence Analysis of PnP-ADMM [231]
B.5 Variants of PnP/RED Algorithms [233]
B.6 Additional Technical Details [234]

Appendix C: Supplement for Chapter 8
C.1 Proof of Theorem 8.1 [240]
C.2 Proof of Theorem 8.2 [243]
C.3 Coordinate-Friendly Implementations [245]
C.4 Additional Details and Discussion [247]
   C.4.1 Architecture and Training of DnCNN* [247]
   C.4.2 Influence of the Lipschitz Constant on Performance [248]
   C.4.3 Influence of padding in patch-wise denoising [248]

Appendix D: Supplement for Chapter 9
D.1 Memory Access without Global Lock [250]
D.2 Proof of Analysis [251]
   D.2.1 Proof of Theorem 9.1 [252]
   D.2.2 Proof of Theorem 9.2 [259]
D.3 Additional Technical Details [266]
   D.3.1 Architecture and Training of the DnCNN Prior [266]
List of Figures

Figure 1.1: Visual illustrations of the challenges faced in the design of computational imaging algorithms. (a) Reflected light microscope as an example that cannot measure the complete light due to the system configuration. (b) Two examples of the noise sources. (c) A visual illustration of the high computational complexity when solving the image reconstruction problem. ................................................... 3

Figure 2.1: Visual illustration of the computational imaging pipeline considered in this dissertation. (a) The unknown object $x$ that we want to characterize. The characterization can be either the surface or the internal structure of the object. (b) An imaging system of interest $A$ is used to sense the object. (c) The measurements $y$ generated by the imaging system. These measurements are often not interpretable by human eyes and are corrupted by noise $e$. (d) Computational algorithms are used to recover interpretable images from the measurements. The development of such algorithms is the focus of this dissertation. (e) The final reconstructed image. We refer the process from (a) to (c) as the forward model, while the inverse problem denotes the inverse process from (d) to (e). ............................................................ 9

Figure 2.2: A variety of computational imaging systems can be mathematically described as linear inverse problems. .......................................................... 11

Figure 2.3: Visualization of the widely-adopted DL scheme in computational imaging. bring the measurement $y$ to the image domain via simple backprojection and then use a CNN to further refine the low-quality image $\hat{x}$ to generate the final image $\hat{\hat{x}}$. A function measuring the difference between $\hat{\hat{x}}$ and groundtruth $x$ is constructed to optimize the network. ........................................................................ 22

Figure 2.4: The U-Net architecture introduced in [94]. ............................................... 23

Figure 2.5: The DnCNN architecture introduced in [262]. ................................. 24

Figure 3.1: Visualization of a single iteration in PnP-PGM and PnP-ADMM. .... 50
Figure 4.1: Test images used. Top row from left to right: *Barbara, Boat, Foreman, House*. Bottom row from left to right: *Lenna, Monarch, Parrot, Peppers.*

Figure 4.2: Illustration of the influence of the step-size $\gamma$ on the convergence of PnP-OPGM with BM3D as the denoiser. The distance to a fixed point is plotted against the iteration number for 3 distinct step-sizes for both accelerated (solid) and basic (dashed) variants of PnP-OPGM for $B = 30$. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. This plot illustrates that the empirical performance of PnP-OPGM under BM3D is consistent with Proposition 4.5, where the accuracy improves with smaller $\gamma$.

Figure 4.3: Illustration of the influence of the minibatch size $B$ on the convergence of PnP-OPGM with BM3D as a denoiser. The distance to a fixed point is plotted against the iteration number for 3 distinct minibatch sizes for both accelerated (solid) and basic (dashed) variants of PnP-OPGM for $\gamma = 1/L$. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. This plot illustrates that the empirical performance of PnP-OPGM using BM3D is consistent with Proposition 4.5, where the accuracy improves with larger $B$.

Figure 4.4: Illustration of the influence of the step and minibatch sizes on the convergence of PnP-OPGM with TV as the denoiser. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. A proximal operator is $(1/2)$-averaged, which means that it perfectly satisfies the assumptions of Proposition 4.5.

Figure 4.5: Comparison between the batch and online PnP algorithms for a fixed reconstruction time. SNR (dB) is plotted against the time in seconds for three algorithms: PnP-OPGM, PnP-APGM, and PnP-ADMM. Both PnP-APGM and PnP-ADMM use the full set of 60 illuminations at every iteration, while PnP-OPGM uses a random subset of 10 or 30 illuminations. This lower per-iteration cost, leads to a substantially faster convergence of PnP-OPGM.

Figure 4.6: Comparison between the batch and online PnP algorithms under a fixed measurement budget. SNR (dB) is plotted against the number of iterations for three algorithms: PnP-OPGM, PnP-APGM, and PnP-ADMM. From the top to the bottom, figures show the performance when the budget is 10, 20, and 30 illuminations, respectively. The plot illustrates that for the same per iteration cost, PnP-OPGM can significantly outperform its batch counterparts.
Figure 4.7: Visual illustration of the reconstructed Monarch and Parrot images obtained using PnP-OPGM, PnP-APGM, and PnP-ADMM, all under BM3D. The original images are displayed in the first column. The second and the third columns show the results of PnP-APGM and PnP-ADMM with the budget of 30 illuminations, and the fourth and the fifth columns present the results of the PnP-OPGM with the budget of 10 and 30 illuminations. Visual differences are highlighted using the rectangles drawn inside the images. Each reconstruction is labeled with its SNR (dB) value with respect to the original image.

Figure 5.1: Visual comparison of the reconstructed images of Cameraman obtained by PnP-OPGM and PnP-APGM. Minibatch $B = 60$ was used in the simulation. The first column (Original) shows the original image. The second column (LS) presents the result of least-square fitting using no regularizer. The third (Online-LS), fourth (OPGM (TV)) and fifth (PnP-OPGM (BM3D)) columns present the results of OPGM using no regularizer, using TV and using BM3D, respectively. The last column (PnP-APGM (BM3D)) shows the result of PnP-APGM using BM3D and all 293 measurements. Each image is labeled with its SNR value with respect to the original image.

Figure 5.2: Evolution of average SNR across iterations for batch and online PnP algorithms using different priors. The corresponding labels are shown at the bottom-right corner inside the plot. The purple dotted line, PnP-PGM (BM3D), indicates the performance when using all 293 measurements. Note that PnP-OPGM (BM3D) achieves the SNR performance of PnP-PGM (BM3D) at a lower computational cost.

Figure 5.3: Comparison between PnP-OPGM (BM3D) and PnP-PGM (BM3D) for a fixed reconstruction time. The average SNR is plotted against the time in seconds for both algorithms. PnP-OPGM (BM3D) uses only 60 measurements per iteration, while PnP-OPGM (BM3D) uses all 293 measurements. The lower per iteration cost leads to a substantially faster convergence of PnP-OPGM (BM3D).

Figure 5.4: Comparison of online and batch algorithms on the FPM dataset containing HeLa cells. Each algorithm uses the budget of 60 measurements per iteration. The first row illustrates the results of LS, PGM (TV), and PnP-PGM (BM3D). The second row shows Online-LS, OPGM (TV), and PnP-OPGM (BM3D). Visual differences are illustrated by the white rectangles drawn inside the images. The green arrows highlight the artifacts.
Figure 6.1: Illustration of the influence of the penalty parameter $\gamma > 0$ on the convergence of IPA for a DnCNN prior. The average normalized distance to $\text{zer}(S)$ and SNR (dB) are plotted against the iteration number with the shaded areas representing the range of values attained over 12 test images. The accuracy of IPA improves for smaller values of $\gamma$. However, the SNR performance is nearly identical, indicating that in practice IPA can achieve excellent results for a range of fixed $\gamma$ values. ................................................................. 91

Figure 6.2: Illustration of scalability of IPA and several widely used PnP algorithms on problems of different sizes. The parameters $n$ and $b$ denote the image size and the number of acquired intensity images, respectively. The average SNR is plotted against time in seconds. Both IPA and PnP-OPGM use random minibatches of 60 measurements at every iteration, while PnP-ADMM and PnP-APGM use all the measurements. The figure highlights the fast empirical convergence of IPA compared to PnP-OPGM as well as its ability to address larger problems compared to PnP-ADMM and PnP-APGM. ...................... 94

Figure 7.1: Visual comparison of a single iteration in PnP-ADMM/PnP-PGM and GM-RED. While PnP algorithms adopt a serial two-step procedure, GM-RED descends along the direction that is a linear combination of the gradient and noise residual. ................................................. 103

Figure 8.1: Ten randomly selected test images from the fastMRI knee dataset [259]. 115

Figure 8.2: The architecture of two variants of DnCNN$^\ast$ used in our simulations. Each neural net is trained to remove AWGN from noisy input images. **Residual** denoiser is trained to predict the noise from the input. The final desired denoiser $D_\sigma$ is obtained by simply subtracting the predicted noise from the input $D_\sigma(z) = z - \text{DnCNN}^\ast(z)$. **Direct** denoiser is trained to directly output a clean image from a noisy input $D_\sigma(z) = \text{DnCNN}^\ast(z)$. In some experiments, we further constrain the Lipschitz constant (LC) of the direct denoiser to LC = 1 and of the residual denoiser to LC = 2 using spectral normalization [195]. LC = 1 implies a nonexpansive denoiser. A residual $R = I - D_\sigma$ with LC = 2 provides a necessary (but not sufficient) condition for a nonexpansive denoiser. ................................................................. 116
Figure 8.3: Evolution of the images reconstructed by BC-RED using the DnCNN* denoiser for different values of $\tau$. The first row corresponds to Fourier matrix with 30 dB noise, while the second row corresponds to the Radon matrix with 40 dB noise. Each reconstructed image is marked with its SNR value with respect to the ground truth image. The optimal parameters $\tau^*$ for the two problems are 0.0037 and 2.35, respectively. The denoiser used in this simulation is the residual DnCNN* with a Lipschitz constant $LC = 2$. This figure illustrates how $\tau$ enables an explicit tradeoff between the data-fit and the regularization.

Figure 8.4: Illustration of the convergence of BC-RED under two DnCNN* priors. The left plot correspond to the direct DnCNN* with the $LC = 1$, while the right plot correspond to the residual DnCNN* with $LC = 2$. The average normalized distance to $\text{zer}(G)$ is plotted against the iteration number for the Radon matrix with the shaded areas representing the range of values attained over all test images. Note that $LC = 1$ implies a nonexpansive denoiser, and $LC = 2$ provides a necessary (but not sufficient) condition for a nonexpansive denoiser.

Figure 8.5: Illustration of the run-time convergence of BC-RED under two DnCNN* priors. The left plot correspond to the direct DnCNN* with the $LC = 1$, while the right plot correspond to the residual DnCNN* with $LC = 2$. The average normalized distance to $\text{zer}(G)$ is plotted against the run-time for the Radon matrix with the shaded areas representing the range of values attained over all test images. The run-time convergence of PGM with TV is also plotted for reference.

Figure 8.6: Illustration of the influence of the parameter $\tau > 0$ for solving TV regularized least-squares problem using BC-RED. As $\tau$ increases, BC-RED provides an increasingly accurate approximation to the TV optimization problem.

Figure 8.7: Visual comparison between BC-RED and RED against PGM (TV) and U-Net for all three matrices with 30 dB noise. For BC-RED and RED, we selected the denoiser resulting in the best reconstruction performance. Every image is marked by its SNR value with respect to the ground truth. We highlight the excellent agreement between BC-RED and RED in all experiments. Note the strong degradation in the image quality for U-Net, due to the mismatch between the training and testing.
Figure 9.1: Visual illustration of serial and parallel image recovery on a multicore system. (a) Serial processing uses only one core of the system for every iteration. (b) Synchronous parallel processing has to wait for the slowest core to finish before starting the next iteration. (c) Asynchronous parallel processing can continuously iterate using all the cores without waiting. (d) Asynchronous parallel processing using the stochastic gradient leads to additional flexibility. (a), (b), and (c) use all the corresponding measurements at every iteration, while (d) uses only a small random subset at a time. Async-RED adopts the schemes shown in (c) and (d)... 125

Figure 9.2: Convergence of Async-RED-BG for different numbers of accessible cores $n_c \in \{2, 4, 6, 8\}$. The left figure plots the average normalized distance to $\text{zer}(G)$ against the iteration number; the middle and right figures plot these values, as well as SNR, plotted against the actual runtime in seconds. The shaded areas represent the range of values attained over the test images. ........................................... 133

Figure 9.3: Left: Evolution of the convergence accuracy of Async-RED-SG as the minibatch size $w$ increases. The average distance is plotted against the number of iterations with the shaded areas representing the range of values attained over the test images. Middle & Right: Comparison of convergence speed between Async-RED-BG/SG and other baselines. The right table summarizes the total runtime and the speed-up compared with GM-RED for all algorithms. ........................................ 134

Figure 9.4: CT reconstruction with a time budget of 1 hour by Async-RED-BG/SG and GM-RED. The colormap is adjusted for the best visual quality... 137

Figure 10.1: Conceptual illustration of DeCAF for IDT. (a) DeCAF reconstructs the RI volume by learning a neural field parameterized by a multilayer perceptron (MLP). The network is trained to map the 3D coordinate $(x, y, z)$ to the corresponding RI value by minimizing a loss that penalizes measurement mismatch and imposes regularization. (b) Our IDT system uses a programmable LED array to illuminate a sample from different angles and uses a digital camera to record the intensity measurements of the scattered light. By changing the illumination patterns, our system can implement different IDT modalities. (c) Our experiments consider three illumination patterns, that is, dense, annular, and multiplexed illuminations, each of which corresponds to a different formulation of the forward model. (d) DeCAF can reconstruct high-quality 3D RI maps from intensity-only and limited-angle measurements. (e) DeCAF learns a continuous representation and can render samples on a pixel grid of the desired density (illustration with $3.5 \times$, $7.5 \times$, and $31.8 \times$ upsampling). ........................................ 143
Figure 10.2: Visual illustration of the network structure and the encoding strategy used in DeCAF. (a) The overall structure of network $\mathcal{M}_\phi$. (b) Illustration of positional encoding for $z$ coordinate. (c) Illustration of radial encoding for the coordinates in the $(x, z)$ plane.

Figure 10.3: Reconstruction of Spirogyra Algae acquired by dIDT. (a) 2D rendering obtained by accumulating all the $z$ slices from DeCAF. Scale bar 65 $\mu$m. (b) Visual comparison of the axial views at $z \in \{4, 16, 28, 40\}$ $\mu$m reconstructed using three methods: DeCAF, SIMBA, and Tikhonov. We adjusted the RI range of each method for the best visual quality with the corresponding colormaps shown on the right. (c) & (d) Axial views corresponding to the colored lines in (a). Each plot uses the same colormap associated with the method presented in (b). Scale bar 40 $\mu$m. (e) Line profiles from the dashed lines in (c) and (d). The label at the upper right of each plot indicates the corresponding dashed line. This figure illustrates the ability of DeCAF to reconstruct high-contrast RI maps by also significantly reducing the missing-cone artifacts.

Figure 10.4: Reconstruction of Diatom Algae acquired by aIDT. (a) 3D illustration of the DeCAF reconstruction showing the overall structure of the sample. (b) Axial view at $z = 0$ $\mu$m (focal plane) reconstructed by DeCAF. Scale bar 10 $\mu$m. (c) & (d) $y$-$z$ and $x$-$z$ lateral views corresponding to the colored paths in (b). The results of Tikhonov are also presented for reference. Scale bar 10 $\mu$m. (e) Visual illustrations of the axial views at $z \in \{-1.0, 16, 28, 40\}$ $\mu$m, highlighting better removal of artifacts compared to Tikhonov. (f) Visual demonstration of DeCAF’s ability to perform continuous RI upsampling along the $x$ and $y$ dimensions. DeCAF’s results are consistent with that of the classic interpolation methods but provide finer details highlighted by the arrows.

Figure 10.5: Reconstruction of Human Buccal Epithelial Cell Cluster acquired by aIDT. (a) Example intensity measurement collected by our aIDT setup for the cell cluster. Note that the background light is removed from the image. Scale bar 13 $\mu$m (b) & (c) $x$-$z$ and $y$-$z$ lateral views of the DeCAF reconstruction associated with the paths in (a). Superior performance in artifact removal and axial separation is demonstrated over the Tikhonov regularization. Scale bar 10 $\mu$m. (d), (e), & (f) The axial views at various depth of the two sub-cell clusters shown in (b) and (c). These results further highlight the strong axial sectioning effects as well as the fine details preserved by DeCAF. (h) & (i) Visual demonstration of DeCAF’s ability to perform continuous RI upsampling along the $z$ dimension. Smooth and consistent transition in the appearance of bacteria is observed.
Figure 10.6: **Reconstruction of C. Elegans acquired by mIDT.** (a) & (b) The reconstructed RI distribution at $z = 0 \, \mu m$ (focal plane) by DeCAF. (c), (d), & (e) Lateral views corresponding to the paths shown in (a) and (b). Biological structures are highlighted by the arrows and circles. (f), (g), & (h) Axial views of the regions highlighted in (a) and (b) at $z \in \{-1, 1, 3\} \, \mu m$. Note how DeCAF provides higher contrast and finer details than Tikhonov.

Figure 11.1: The conceptual illustration of CoIL in the context of sparse-view CT. A multilayer perceptron (MLP) is used to represent the full measurement field by learning to map the measurement coordinate $(\theta, l)$ to its response $r$. Visual examples compare the recovered images with and without CoIL for total variation (TV). CoIL is used to generate 360 views from the data consisting of 120 noisy views of 40 dB input SNR. The quantitative and visual results in this chapter highlight the ability of CoIL to significantly improve the imaging quality for several widely-used image reconstruction methods.

Figure 11.2: Illustration of the CoIL workflow for a tomographic imaging system with free parameters $\mathbf{v} \in \mathbb{R}^v$. First, a set of $N > 0$ measurements are acquired by the system under different realization of $\mathbf{v}$. Then, the coordinate-response pairs $\{(\mathbf{v}_i, r_i)\}_{i=1}^N$ are used to train a coordinate-based MLP $\mathcal{M}_{\phi}: \mathbf{v} \rightarrow r$ for encoding the full measurement field. Once the training is finished, the encoded field is extracted from $\mathcal{M}_{\phi}$ with an arbitrary resolution by querying the relevant coordinates. In the final stage, the CoIL field and the actual measurements are jointly used for image reconstruction using a user-defined method.

Figure 11.3: Visualization of the coordinate-based MLP used in the CoIL methodology. The network $\mathcal{M}_{\phi} = \mathcal{N}_{\phi} \circ \gamma(\mathbf{v})$ is a concatenation of a single Fourier feature mapping (FFM) layer $\gamma(\mathbf{v})$ and a conventional MLP $\mathcal{N}_{\phi}$. As training on example pairs $\{(\mathbf{v}_i, r_i)\}_{i=1}^N$, $\mathcal{M}_{\phi}$ is able to learn a continuous mapping from a coordinate to its response. Hence, $\mathcal{M}_{\phi}$ becomes an implicit neural representation of the full-measurement field.

Figure 11.4: Eight $512 \times 512$ images from the scans of two patients in the AAPM human phantom dataset [146] were used for testing.
Figure 11.5: Illustration of the benefit of including the Fourier feature mapping (FFM) layer into CoIL. We plot sinograms and their FBP reconstructions in the first and second row, respectively. The proposed FFM in CoIL is compared against No FFM strategy (which does not have any FFM layer), positional encoding (Pos Enc) [151], and random Gaussian sampling (Random) [223]. The four MLPs are used to generate 360 views from the $P = 120$ projections with $I = 40$ dB noise. Both sinograms and images are labeled with the SNR values with respect to the ground truth shown in the right-most column. The bounding boxes highlight areas of significant visual difference. This comparison shows the benefit of using the FFM layer with linear spacing in the Fourier space.

Figure 11.6: Quantitative evaluation of the CoIL field for different projection numbers ($P$) and noise levels ($I$). The plot is divided into three regions, corresponding to $P$ equal to 60, 90, and 120, respectively. Within each region, the average SNR values of the generated sinograms are plotted against different input SNR values, which are also drawn by the dotted horizontal lines for better visualization. First, note how CoIL generally produces measurement fields of better SNR than the noise level in the measurements. Second, the figure highlights that the quality of the generated CoIL field improves as the number of views increases or the noise level decreases.

Figure 11.7: SNR improvements due to CoIL for each reconstruction algorithm. The plot is divided into three regions, corresponding to 60, 90, and 120 projections, respectively. Within each region, the average SNR improvement is plotted against the reconstruction method. The vertical axis is in log-scale for better visualization. Note that CoIL consistently improves the average SNR values for all the considered algorithms in every scenario.

Figure 11.8: Visual illustration of reconstruction with and without CoIL using the several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 90 (used for FBP-UNet) views from $P = 60$ measurements with $I = 40$ dB noise. Each image is labeled with its SNR value with respect to the ground truth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure 11.9: Visual illustration of reconstruction with and without CoIL using the several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 40$ dB noise. Each image is labeled with its SNR value with respect to the ground truth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. ............................................ 179

Figure 11.10: Visual illustration of reconstruction with and without CoIL using the several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 180 (used for FBP-UNet) views from $P = 120$ measurements with $I = 40$ dB noise. Each image is labeled with its SNR value with respect to the ground truth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Visual examples reconstructed with and without CoIL using the considered methods........................ 180

Figure 12.1: Visualization of the connections between the analysis of each of our proposed PnP/RED algorithms. ................................................ 187

Figure B.1: Illustration of the architecture of DnCNN used in all experiments. Vectors $\hat{x}$ and $x$ denote the denoised image and ground truth, respectively. The neural net is trained to remove the AWGN from its noisy input image. We also constrains the Lipschitz constant of $R_\sigma$ to be smaller than 1 by using the spectral normalization technique in [195]. This provides a necessary condition for the satisfaction of Assumption 6.2. ..................................................... [233]

Figure B.2: Illustration of the convergence of IPA for a DnCNN prior under drastically changed $\gamma$ values. The average normalized distance to zer($S$) and SNR (dB) are plotted against the iteration number with the shaded areas representing the range of values attained over 12 test images. In practice, the convergence speed improves with larger values of $\gamma$. However, IPA still can achieve same level of SNR results for a wide range of $\gamma$ values. .......................................................... [235]
Figure B.3: Visual examples of the reconstructed House (upper) and Parrot (bottom) images by IPA and PnP-ADMM. The first and last columns correspond to PnP-ADMM under DnCNN with 5 fixed measurements and with the full 60 measurements, respectively. The second, third, and fourth column correspond to IPA with a small minibatch of size 5 under TV, BM3D, and DnCNN, respectively. Each image is labeled by its SNR (dB) with respect to the original image, and the visual difference is highlighted by the boxes underneath. Note that IPA recovers the details lost by the batch algorithm with the same computational cost and achieves the same high-quality results as the full batch algorithm.

Figure C.1: **Left:** The averaged SNR values obtained by BC-RED for the Random matrix with 40 dB noise and *patch-wise* residual DnCNN∗, where the denoiser input includes an additional padding around the patch, while the output has the size of the patch. **Center and Right:** The convergence speed of BC-RED under *patch-wise* residual DnCNN* with 40 px padding and the *full-image* residual DnCNN*. Distance to zer(G) – corresponding to the *full-image* denoiser – and SNR are plotted against time. As a reference, we provide the convergence of RED using the full-image DnCNN∗ and BM3D denoisers.

Figure D.1: Illustration of the architecture of DnCNN used in all experiments. The neural net is trained to remove the AWGN from its noisy input image. We also constrains the Lipschitz constant of $R_\sigma$ to be smaller than 2 by using the spectral normalization technique in [196]. This provides a necessary condition for the satisfaction of Assumption 9.4.

Figure D.2: Six test images used in the experiments on CS. From the left to right, there are *cameraman, house, pepper, starfish, butterfly*, and *jet.*

Figure D.3: Visualization of the recovered images from the compressed measurements by Async-RED-BG/SG, Sync-RED, and Gm-RED. Each algorithm is run with a time budget of 700 seconds.

Figure D.4: Convergence Illustration of Async-RED-BG/SG and Gm-RED for CT reconstruction with a time budget of 1 hour. Here, ASYNC-RED-SG randomly uses one-third of the total measurements at every iteration.

Figure D.5: Visualization of the reconstructed CT images by PGM-Tv, Gm-RED, Sync-RED, and Async-RED-BG/SG. Each algorithm is run with a time budget of 1 hour. The colormap is adjusted for the best visual quality.
Figure E.1: **Visual illustration of the block-wise training procedure for DeCAF.** (a) General workflow of the proposed training procedure. (b) Illustration of padding. (c) Illustration of view enlargement. (e) Illustration of measurement separation. ........................................[281]

Figure E.2: **Network architecture of DnCNN.** DnCNN is trained to map the noisy input to the noise residual by using the BSD500 training dataset [140], which consists only of natural images. .........................[282]

Figure E.3: **Convergence of DeCAF for different samples.** In each figure, the mean absolute error (MAE) between the predicted and real test measurements is plotted against the iteration number. ..................[282]

Figure E.4: **Visual comparison of different x-y encoding strategies on C. Elegans (head).** Visual differences are highlighted using arrows. (a), (b) & (c) Axial slices of the C. elegans’ head at $z \in \{-1, 1, 3\} \mu m$ reconstructed by using Radial (ours), Positional, and Gaussian encoding of the $x$-$y$ plane. The encoding of $z$, as well as the rest of DeCAF, is set to be the same in the comparison. (d), (e) & (f) The associated lateral slices of the buccal cavity and grinder shown in the axial views...........................................................[283]

Figure E.5: **Demonstration of the implicit regularization due to the MLP on Spirogyra Algae.** Visual differences are highlighted using boxes. (a) DeCAF, which includes an explicit regularizer. (b) DeCAF-Noreg relies only on implicit regularization by MLP. (c) SIMBA. (d) Tikhonov. Axial slices at $\{4, 8, 12\} \mu m$ and $\{28, 32, 36\} \mu m$ are selected. .................................................................................[284]

Figure E.6: **Demonstration of the effectiveness of explicit regularization on C. Elegans (body).** Visual differences are highlighted by arrows. (a), (b) & (c) Axial slices of the C. Elegans body at $z \in \{-1, 1, 3\} \mu m$ reconstructed by using full regularization (DeCAF), only the axial continuity (DeCAF-AC), only the noise reduction (DeCAF-NR), and no regularization (DeCAF-Noreg). The rest of DeCAF is set to be same in the comparison. (d), (e) & (f) The associated lateral slices of the lumens of intestine shown in the axial views...........................................................[285]

Figure F.1: Illustration of the improvement in reconstruction SNR for RED and TV for different values of as $\alpha \geq 0$. The figure is divided into three columns, corresponding to the input SNR values of 30, 40, and 50 dB, respectively. The results indicate that CoIL synthesized measurements always improve the reconstruction performance, but need to be properly balanced with the true measurements. ............[291]
Figure F.2: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 90 (used for FBP-UNet) views from $P = 60$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................... [292]

Figure F.3: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................... [293]

Figure F.4: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 180 (used for FBP-UNet) views from $P = 120$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................... [294]

Figure F.5: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 90 (used for FBP-UNet) views from $P = 60$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................... [295]

Figure F.6: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................... [296]
Figure F.7: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 180 (used for FBP-UNet) views from $P = 120$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it. ............................................[297]
List of Tables

Table 4.1: Minimal distance averaged over the test image set ......................... 65
Table 4.2: Individual reconstruction SNRs for each image………………………… 70
Table 6.1: Final average SNR (dB) and Runtime obtained by several PnP algo-
rithms on all test images. .............................................................................. 95
Table 6.2: Per-iteration memory usage specification for reconstructing 1024×1024
images .............................................................................................................. 96
Table 8.1: Average signal-to-noise ratios (SNRs) computed over 10 test images
for different inverse problems and noise levels. The best SNR for each
experiment is highlighted in bold-italic, while the best denoiser prior is
in light-green. .................................................................................................. 118
Table 10.1: List of algorithmic hyperparameters ............................................. 149
Table 10.2: Quantitative Illustration of the scalability of DeCAF due to its off-the-
grid feature using the C. elegans specimen as an example. Note that
the space required for DeCAF is independent of the reconstruction
grid .................................................................................................................. 160
Table 11.1: The average SNR of the sinograms generated by No FFM, Pos
Enc, Gaussian and CoIL in the scenarios corresponding to $P \times I =$
{60, 90, 120} × {30, 40, 50} ........................................................................... 172
Table 11.2: The average SNR of the sinograms generated by Linear, Bicubic,
ConvNet [122], and CoIL in the scenarios corresponding to $P \times I =$
{60, 90, 120} × {30, 40, 50} ........................................................................... 174
Table 11.3: The average SNR values obtained with and without CoIL by using
FBP, PGM-TV, GM-RED, and FBP-UNet in the scenarios corre-
spanding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$............................... 176
Table A.1: List of optimal $\sigma$ values for each test image. ............................... [218]
Table B.1: Overview of several existing PnP/RED algorithms ................................... [234]

Table B.2: Per-iteration memory usage specification for reconstructing 512×512 images ........................................................................................................................................................................ [235]

Table B.3: Optimized SNR (dB) obtained by IPA under different priors for images from Set12 from [262] ........................................................................................................................................................................ [239]

Table C.1: Average SNR achieved by BC-RED for two variants of DnCNN* at different Lipschitz constant (LC) values. .......................................................... [246]

Table D.1: SNR values obtained by Async-RED-BG using different block sizes on CS task. ........................................................................................................................................................................ [268]

Table D.2: SNR values obtained by Async-RED-SG using different minibatch sizes on CS task. ........................................................................................................................................................................ [268]

Table D.3: SNR values obtained by Async-RED-BG using different pad size on CS task. ........................................................................................................................................................................ [269]

Table F.1: The average SNR of the sinograms generated by multiple variants of CoIL in the scenarios corresponding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$ ........................................................................................................................................................................ [288]

Table F.2: The average SNR values obtained with the variants of CoIL by using FBP in the scenarios corresponding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$. ........................................................................................................................................................................ [290]
Acknowlegements

I want to express my gratitude to many people who made my time at WUSTL a truly exceptional experience.

I want to immediately thank my advisor, Prof. Ulugbek S. Kamilov, for his never-ending support. It is my greatest pleasure to work under his guidance. His rigorous scholarship, excellent teaching, and passion for science makes him a model professor that I wish one day I can become. Additionally, he is also a great mentor in life. In almost every week, he joins the lab for lunch, during which he will share his life experience and give advises. As a junior researcher, I learned a lot from these happy and inspiring talks.

I want to also thank other members of my committee, Prof. Tao Ju, Prof. Netanel Raviv, Dr. Brendt Wohlberg, and Prof. William Yeoh, for reviewing the dissertation. I additionally want to express my sincere thanks to Dr. Brendt Wohlberg for his support on my research and career. I also want to thank Prof. Tao Ju for his valuable inputs on job hunting.

I thank the fellow past and present members of the Computational Imaging Group (CIG): Xiaojian Xu, Jiaming Liu, Weijie Gan, Yuyang Hu, Shirin Shoushtari, Long Fa, Jialong Zhang, Chunyuan Li, Joseph Han, Zachary Pewitt, Shiqi Xu, Zihui Wu, Mingyang Xie, Guangxiao Song, Renhao Liu, and Wento Shangguan. In particular, I sincerely thank my colleagues Xiaojian Xu, Jiaming Liu, Weijie Gan, Guangxiao Song, Yuyang Hu, Zhixin Sun and Shirin
Shoushtari for their constant help and companion. Without them, my life at WUSTL would be much less colorful. I additionally thank Shiqi Xu, Zihui Wu, Mingyang Xie, Renhao Liu, and Wentao Shangguan for the fruitful collaboration. I wish them prosperity in their future academic life.

I would like to express my gratitude to other WUSTL colleagues: Dr. Zhen Zhang, Nancy Ning, Maria Medrano, Jiameng Sun and many others. I really appreciate all good moments we shared together.

I would like to thank my collaborators: Prof. Hongyu An, Prof. Dmitriy Yablonskiy, Dr. Brendt Wohlberg, and Dr. Orazio Gallo. Thank you for the help on my research.

I would like to thank Dr. Marion Neumann, who encouraged and helped me to apply for a Ph.D..

I want to thank my mother, who has left me in 2017, for her teaching. I missed her so much. I want to thank my father who never ceases to support my pursuit. I thank my grand parents for their unconditional love. Without my family, I would never grow to the stage where I am now.

Yu Sun

Washington University in St. Louis
August 2022
Dedicated to my mother, Wei Zhang (1968-2017), and my family.
ABSTRACT OF THE DISSERTATION

Integrating Physical Models and Deep Priors for Computational Imaging

by

Yu Sun

Doctor of Philosophy in Computer Science
Washington University in St. Louis, 2022
Professor Ulugbek Kamilov, Chair

This dissertation addresses integrating physical models and learning priors for computational imaging. The motivation of our work is driven by the recent discussion of learning-based methods that solve the imaging inverse problem by directly learning a measurement-to-image mapping from the existing data: they achieve superior performance over the traditional model-based methods but lack the physical model to impose sufficient interpretation and guarantee of the final image. We adopt the classic statistical inference as the underlying formulation and integrate learning models as implicit image priors, such that our framework is able to simultaneously leverage physical models and learning priors. Additionally, the growing sizes of the image as well as the measurements in modern computational imaging systems place a significant burden on both computation and memory. Another purpose of the dissertation is to extend our framework to those scenarios by incorporating large-scale optimization techniques.

The dissertation significantly extends three algorithmic frameworks—that is, plug-and-play priors (PnP, Part II), regularization by denoising (RED, Part III), and neural fields (NF, Part IV)—with multiple contributions including the design of novel algorithms, establishment of unified theory, and applications to real imaging problems. In Part II, we present in-depth discussions of two popular PnP algorithms: PnP-PGM and PnP-ADMM. Our contributions
here include the proof of their fixed-point convergence under deep denoising priors and the proposal of scalable PnP variants for processing a large set of measurements by using online gradients or proximal maps. In Part III, we conduct similar investigations on RED. We first prove the fixed-point convergence for gradient-method RED (GM-RED) algorithm and propose two variants for efficiently inferring large images by using block coordinate and parallel computing techniques. In particular, our analysis framework, based on monotone operator theory, is unified for PnP and RED and has not been established in the existing literature. In Part IV, we extend NF—a novel self-supervised learning paradigm—to computational imaging by developing two novel methods. Our first method, DeCAF, investigates NF’s regularization ability in the image domain with spatial coordinates, while our second method, CoIL, leverages the representation power of NF to complete under-sampled measurements in the measurement domain with non-spatial coordinates.
Part I

Introduction
Chapter 1

Introduction

Computational imaging refers to a compendium of technologies that visualize indirect physical quantities by synergistically using hardware sensors and computational algorithms. In contrast to the direct imaging approaches, computational imaging enables unprecedented imaging capabilities long thought impossible. Examples of such capabilities include multi-camera systems that can capture pictures in dark environments [152], medical imaging systems that can visualize human internal body structure without invasive surgeries [36, 237], and high-resolution microscopes that can see nanoscale objects [202, 211],

Image reconstruction algorithms are central in computational imaging systems. As the component directly responsible for image formation, their performance significantly impact the final imaging quality of the desired object. Image reconstruction algorithms aim to computationally solve the imaging inverse problem, which corresponds to the inference of the unknown image from the known measurements collected by the hardware system. However, solving imaging inverse problems is challenging due to the following reasons:
Figure 1.1: Visual illustrations of the challenges faced in the design of computational imaging algorithms. (a) Reflected light microscope as an example that cannot measure the complete light due to the system configuration. (b) Two examples of the noise sources. (c) A visual illustration of the high computational complexity when solving the image reconstruction problem.

- **Incomplete measurements.** Limited by the system configuration, the measurements we collect are incomplete in terms of the information contained for charactering the desired object. For example, reflected light microscopes cannot measure the transmitted light because the sensors are places at the same side of the specimen as the source. With less knowns than the unknowns, imaging inverse problem becomes an *ill-posed* problem with multiple solutions consistent with the measurements.

- **Noise contamination.** Imaging measurements are corrupted by noise coming from multiple sources. Examples include thermal agitation, wave interference, hardware limitation, and etc. The existence of noise further deteriorates the ill-posedness of the imaging inverse problem by leading to unreliable solutions.

- **High computational complexity.** Signals dealt in imaging problems are usually high-dimensional images or volumes that often consist of millions of data entries. Brute-force strategies for processing such large-scale data can lead to unacceptable latency or even intractable computation.

In this dissertation, we attempt to jointly address these challenges by developing *scalable* computational approaches that integrate *physical models* and *deep priors*. Here, we
are particularly interested in system-agnostic priors whose training is independent of the ground-truth images associated with the considered imaging modalities. We built our framework upon three algorithmic frameworks—that is, plug-and-play priors (PnP), regularization by denoising (RED), and neural fields (NF)—with multiple contributions including the design of novel algorithms, establishment of unified theory, and applications to real imaging problems. In the next section, we summarize our key contributions made in this dissertation.

1.1 Main Contributions

This dissertation contains five major contributions to computational imaging.

- We present a new fixed-point theory for analyzing the convergence of both PnP and RED frameworks. By leveraging the concept of monotone operators, our theory can prove the explicit $O(1/t)$ convergence to the set of fixed points under denoisers that not necessarily correspond to proximal operators. Note that such results have never been shown in the prior work on PnP and RED. Additionally, our theory is completely backward-compatible with the established results in convex optimization, thus linking PnP/RED to classic interpretations as well as providing new understandings of proximal methods.

- We introduce deep image denoisers into PnP/RED frameworks with convergence guarantees. We develop a general strategy for training monotone deep neural networks (DNN), which follow certain monotone operator properties, by leveraging spectral normalization techniques. Numerical validations show that the DNNs trained using our strategy empirically act as the desired monotone operators, successfully leading to the convergence of PnP/RED algorithms.
• We substantially extend PnP/RED framework to large-scale imaging problems by proposing multiple new scalable algorithms. Our proposed algorithms widely explore the techniques of online gradients, incremental proximal operators, block-coordinate gradients, and asynchronous parallel computing, enabling much more efficient processing of large measurement datasets over the existing PnP/RED counterparts. Beyond algorithmic design, we further show the convergence for these scalable algorithms by extending our fixed-point theory, establishing a unified theory characterizing both original and scalable PnP/RED algorithms.

• We develop a new self-supervised deep learning method for continuous microscopic image reconstruction based on NF—a novel scheme for representing fields by using DNNs to map spatial coordinates to corresponding physical quantities. Our method, deep continuous artifact-free refractive index field (DeCAF), is the first for reconstructing continuous 3D refractive index (RI) maps, an important source of contrast used in label-free optical tomographic imaging, from discrete 2D measurements. The continuous representation in DeCAF allows to generate images at voxel grids of arbitrary density without retraining of the DNN, which is useful for addressing computational and memory bottlenecks in image reconstruction and analysis. Additionally, we explored the synergistic integration of the implicit regularization provided by the coordinate-based DNN and the explicit regularization by image priors, enabling significant suppression of image artifacts without losing fine details.

• We further extend the concept of neural field to the measurement domain that no longer has a spatial coordinate system. We propose coordinate-based internal learning (CoIL) for learning a continuous representation of measurements. CoIL trains a DNN to encode the complete measurement field by mapping the geometry parameters of the imaging system to their measurement response. CoIL is a self-supervised method that requires
no training examples besides the measurements of the test object itself. Once the DNN is trained, CoIL can generate new measurements that can be used within a majority of image reconstruction methods. Our results on computed tomography demonstrate the ability of CoIL to consistently improve the performance of both model-based and learning-based methods by providing high-fidelity measurement fields. In general, this and previous contributions together constitute new and complementary applications of NF in computational imaging.

- Our last contribution is to apply several of our methods to two real optical tomographic imaging applications: intensity diffraction tomography (IDT) and Fourier ptychography. The two modalities are advanced imaging techniques that aim to recover the RI map of the sample from the intensity-only measurements of the light it scatters. Our validation on experimentally-collected datasets clearly demonstrate the quantitative and visual improvements using deep denoising priors, the ability of scalable PnP/RED algorithms to efficiently process large datasets, and the potential of NF-based methods in computational imaging.
Chapter 2

Background

This chapter introduces the background material underlying the technical contributions developed in the thesis. We start by introducing the concept of imaging inverse problem with three specific instances corresponding to linear models, phase retrieval, and image denoising. These three models will be extensively used in the development of our proposed algorithms. We next review the computational algorithms that are commonly used to solve inverse problems, including the traditional model-based methods and the recent learning-based methods. This review aims to assist readers to understand the benefits of integrating physical models and learning priors. We additionally review the recent advances in achieving this goal, including a short discussion of plug-and-play priors (PnP), regularization by denoising (RED), neural fields (NF), and other related frameworks. Lastly, we review the monotone operator theory that will serve as the foundation in the theoretical analysis of PnP/RED algorithms.
2.1 Imaging Inverse Problems

Formally, an inverse problem refers to the process of recovering from a set of observations the casual factors that produced them.\(^1\) In the context of imaging, the casual factors are the physical quantities of the unknown object that can be used to form an image, and the observations are the measurements collected by the imaging system. As signals are often stored and processed digitally, we focus on the discretized inverse problems. We refer the interested readers to [8, 25, 57, 58, 101] for a discussion on continuous inverse problems.

2.1.1 Forward Model

Knowing how the measured data is produced is the prerequisite for defining an inverse problem. We refer to this process as the forward model. Specifically, the forward model of an imaging system, describing the mapping from an image \(x \in \mathbb{R}^n\) to the measurements \(y \in \mathbb{R}^m\), can be formulated as

\[
y = A(x) + e. \tag{2.1}
\]

For notational convenience, we assume that the image and measurements are both real-valued; nonetheless, all the algorithms developed in this dissertation can be easily extended to complex-valued images. We refer to \(A : \mathbb{R}^n \to \mathbb{R}^m\) as the forward operator that is assumed to be known and accurately characterize the deterministic physical process of the measurement acquisition. Here, we define \(A\) as a generic mapping in order to maintain generality. At the end of this section, we will introduce three specific formulations that are commonly adopted in computational imaging. We refer to the vector \(e \in \mathbb{R}^m\) as the noise introduced by the system, which accounts for the non-deterministic part of the forward model. Although accurate modeling of the noise coming from each source is often intractable, it is possible to

\(^1\) Definition from the Wikipedia entry 'Inverse Problem'.
characterize the distribution of the overall noise when treated as a whole. In particular, by assuming the noise sources are independent, the central limit theorem states that noise $e$ follows an independent and identically distributed (i.i.d) zero-mean Gaussian distribution, and we refer to such noise as the additive white Gaussian noise (AWGN).

From the statistical point of view, the relationship between $x$ and $y$ can be also stated as a generic likelihood function $y \sim p_{y|x}(y|x)$, where the uncertainty is introduced by the noise. Under the assumption of AWGN, the forward model $p_{y|x}$ depicts the relationship via the

Figure 2.1: Visual illustration of the computational imaging pipeline considered in this dissertation. (a) The unknown object $x$ that we want to characterize. The characterization can be either the surface or the internal structure of the object. (b) An imaging system of interest $A$ is used to sense the object. (c) The measurements $y$ generated by the imaging system. These measurements are often not interpretable by human eyes and are corrupted by noise $e$. (d) Computational algorithms are used to recover interpretable images from the measurements. The development of such algorithms is the focus of this dissertation. (e) The final reconstructed image. We refer the process from (a) to (c) as the \textit{forward model}, while the \textit{inverse problem} denotes the inverse process from (d) to (e).
Gaussian distribution

\[ y \sim \mathcal{N}(Ax, \sigma^2 I), \]  

(2.2)

where \( \mathcal{N}(m, \Sigma) \) denotes the multivariate normal distribution with mean \( m \) and covariance \( \Sigma \), \( \sigma \) the standard deviation, and \( I \) the identity matrix. It is worth noticing that not all noise can be described in this manner. For example, Poisson noise occurs in the digital cameras under a low-light exposure. We note that the algorithms developed in this dissertation make no specific assumptions on the distribution of the noise.

**Linear Models**

*Linear inverse problem* is one of the most common abstractions adopted for describing computational imaging systems. It defines the forward model as a linear system

\[ y = Ax + e. \]  

(2.3)

where the forward operator \( A \) is simplified to be a linear measurement matrix \( A \in \mathbb{R}^{m \times n} \), and \( e \) denotes the AWGN. A variety of computational imaging systems can be characterized as linear models (see Fig. 2.2), including digital camera, magnetic resonance imaging (MRI), X-Ray computed tomography (CT), and optical microscopy. The biggest advantage of linear models is *linearity*, which makes the problem formulation simple and enables the use of linear algebra to derive and interpret the solution. It has become a common practice for researchers to first derive the linear formulations for general inverse problems.

**Phase Retrieval**

There are problems with unneglectable intrinsic nonlinearity that cannot be formulated as linear models. We refer to these problems as *nonlinear inverse problems*. In this section, we
Image reconstruction can be generalized as an inverse problem.

![Diagram of various imaging systems and their mathematical descriptions](image)

Figure 2.2: A variety of computational imaging systems can be mathematically described as linear inverse problems.

Introduce one special case of such problems, known as phase retrieval (PR), that is central in many imaging systems. Consider the forward model of PR

\[ y = |Ax|^2 + e, \quad (2.4) \]

where \( A \in \mathbb{C}^{m \times n} \) is linear measurement matrix acting on complex signals, \( |\cdot|^2 \) computes the element-wise squared absolute value, and \( e \in \mathbb{R}^m \) is real-valued AWGN. Due to the absolute-value operation, the phase information of the complex image \( x \in \mathbb{C}^{m \times n} \) is lost during the acquisition, leading to a set of real-valued measurements \( y \in \mathbb{R}^m \). Therefore, PR refers to the full recovery of complex image including the phase from the real measurements. One typical application of PR is the Fourier ptychographic microscopy to be studied in Chapter 5: the system aims to image the spatial distribution of a complex optical quantity known as refractive index by only measuring the squared modulus of the scattered light. Overall, PR is a challenging problem because the nonlinearity will lead to nonconvex reconstructions that require extra knowledge than linear algebra.
Image Denoising

We conclude our discussion on imaging inverse problem by introducing a very simple yet fundamental problem known as image denoising. The formulation of image denoising is

\[ y = x + e, \]  

(2.5)

where the measurement \( y \in \mathbb{R}^n \) is simply the true image \( x \in \mathbb{R}^n \) corrupted by the additive noise \( e \in \mathbb{R}^n \). The formulation in Eq. 2.5 is equivalent to having an identity measurement matrix \( I \in \mathbb{R}^{n \times n} \) in linear models. Although the particular study of image denoising is not the focus of this dissertation, we are interested in this problem and its solutions because they are prerequisites for introducing the concept of image denoising priors that are fundamental to PnP/RED algorithms.

2.1.2 Prior & Bayesian Inference

Due to the ill-posedness, it is difficult to derive an analytic inverse mapping \( x = A^{-1}(y) \) that directly recovers the true image \( x \) from the measurements \( y \). One may resort to formulating a maximum likelihood estimator (MLE) to find an estimate \( \hat{x}_{\text{MLE}} \) from \( y \) as the likelihood \( p_{y|x} \) is known. However, the existence of noise often leads the search to completely inexplicable estimates violating the physical rules. To overcome this, additionally knowledge on the properties of the unknown image is necessary to narrow down the potential solution space. Bayesian statistics provides a systematic approach for imposing additional constraints on the solution. From the Bayes’ theorem, we know that the posterior of the true image \( p_{x|y} \)
is given

\[ p_{x|y}(x|y) = \frac{p_{y|x}(y|x)p_{x}(x)}{p_{y}(y)} \quad (2.6a) \]

\[ \propto p_{y|x}(y|x) p_{x}(x). \quad (2.6b) \]

where \( p_{x} \) refers to the prior probability distribution that we believe the true image should follow based on our domain knowledge, and \( \propto \) denotes the proportional relationship. By defining prior distributions associated with the desired image property, e.g. non-negativity, Gaussian smoothness, and etc., we can enforce final estimated images to favor the imposed constraint. The problem of finding the solution that maximizes the posterior is known as the maximum a posterior estimation (MAP). Mathematically,

\[ \hat{x}_{\text{MAP}} = \arg \max_{x} \{ p_{x|y}(x|y) \} \quad (2.7a) \]

\[ = \arg \min_{x} \{-\log(p_{y|x}(y|x)) - \log(p_{x}(x))\} \quad (2.7b) \]

\[ = \arg \min_{x} \{ D(x) + R(x) \}. \quad (2.7c) \]

where, in the context of inverse problem, \( D \) is usually referred to as the data-fidelity term because it ensures the consistency with the measured data, and \( R \) is called regularizer as it imposes the prior information to the final solution. The derivation in Eq. (2.7) translates MAP to a regularized optimization problem. This translation is useful in multiple aspects.

- We have extra freedom to characterize the noise model and latent image properties as the likelihood and prior are not required to be explicitly linked to any explicit distributions.
• We can leverage the rich knowledge of optimization to design various algorithms to numerically calculate the MAP estimate, even including those who do not have analytic formulations.

• We are able to mathematically analyze the numerical algorithms, providing important theoretical insights on the robustness and optimality of the image reconstruction.

**Regularized Least Squares**

When the likelihood is modeled by Gaussian distribution (i.e. Gaussian noise), the Bayesian inference can be formulated as the *regularized least-squares* problem. Consider the probability density function of the multivariate Gaussian distribution defined in Eq. (2.2)

\[
p_{y|x}(y|x) = \frac{1}{\sqrt{\det(2\pi\sigma^2 I)}} \exp\left(-\frac{1}{2\sigma^2} \|A(x) - y\|^2\right),
\]

where \(\det(\cdot)\) compute the determinant of a matrix. By plugging the above results into Eq. (2.7), we have

\[
\hat{x}_{\text{MAP}} = \arg \max_x \left\{ -\log(p_{y|x}(y|x)) - \log(p_x(x)) \right\} = \arg \min_x \left\{ \frac{1}{2\sigma^2} \|A(x) - y\|^2 + \mathcal{R}(x) \right\} = \arg \min_x \left\{ \frac{1}{2} \|A(x) - y\|^2 + \tau \mathcal{R}(x) \right\},
\]

where in the last equality we absorbed \(\sigma^2\) into the scalar \(\tau > 0\) that controls the strength of regularizations. Note that here the forward operator \(A\) defines a general mapping that may not correspond to linear models. By further assume the linearity of the forward operator, we
obtain the regularized least-squares optimization

\[
\hat{x}_{LS} = \arg \min_x \left\{ \frac{1}{2} \| Ax - y \|^2 + \tau R(x) \right\}.
\] (2.10)

We are interested in the least squares due to two reasons: (1) The problem is well-studied in the context of optimization and compressive sensing, allowing us to use off-the-shelf regularizers and optimization algorithms; (2) The objective is convex for convex regularizers, enabling theoretical analysis on optimality and algorithm’s convergence using convex optimization. Throughout the dissertation, we will extensively use Eq. (2.10) to describe the image reconstruction problems in the considered computational imaging applications.

### 2.2 Computational Algorithms for Imaging

So far, we have explained how computational imaging is related to inverse problems and presented the formulation (see Eq. (2.7)) derived from the Bayesian inference. The latter is the widely-used framework for developing image reconstruction algorithms, and we refer to these algorithms as model-based methods due to the use of the forward models derived from physics. In the past years, deep learning (DL) has quickly emerged as a powerful tool for learning representation of data at multiple levels of abstraction [121]. It has enabled revolutionary improvements in a wide range of research areas including natural language processing, computer vision, computer graphics, biology, and etc.. Recently, learning-based methods have been proposed as an alternative paradigm to model-based methods for computational imaging. These algorithms generally rely on training deep neural networks (DNN) to represent a given dataset as a generic prior for a class of images. In this section, we present a brief review of these two types of methods. Note that the intent of this review is not to enumerate
all imaging algorithms, but rather to provide an algorithmic background that allows us to better elucidate our contributions.

2.2.1 Model-based Methods

Consider the general regularized optimization

$$\hat{x} = \arg \min_x f(x), \quad \text{where} \quad f(x) = g(x) + r(x), \quad (2.11)$$

where we use function $g$ to denote the data-fidelity term, and $r$ to denote the regularizer with the strength parameter $\tau$ absorbed. Hereinafter, we will adopt the notations in (2.11) as they are more common in the literature of optimization. As the freedom of adjusting the data-fidelity term is limited (i.e. often assumes Gaussian noise), a more effective way to improve the final image quality is to design better image priors that can accurately characterize distribution of the latent image. In the past decades, there has been significant progress in developing sophisticated image priors, including those based on transform-domain sparsity [73, 191], self-similarity [55], and dictionary learning [68]. For example, one widely-used prior is total variation (TV), which penalizes the $\ell_1$-norm of the image gradient

$$r(x) := \tau \|Dx\|_1, \quad (2.12)$$

where $\tau > 0$ is the regularization parameter and $D$ is the discrete gradient operator. However, many popular regularizers, such as the ones based on the $\ell_1$-norm, are nonsmooth, which precludes the use of gradient descent for solving Eq. (2.11).
Proximal Optimization

The family of proximal methods (PM) are effective in solving nondifferentiable convex optimization problems in computational imaging. PM algorithms avoid differentiating the nonsmooth regularizer by leveraging a mathematical concept known as proximal operator, which itself is a minimization of the squared error between the input $x$ and the output $z$ regularized by $r$,

$$\text{prox}_{\mu r}(z) := \arg \min_{x \in \mathbb{R}^n} \left\{ \frac{1}{2} \|x - z\|^2_2 + \mu r(x) \right\}.$$  \hfill (2.13)

Here, $\mu > 0$ adjusts the regularization strength. In Eq (2.13), the squared error enforces $z$ to be close to $x$, while $r$ penalizes the solutions falling outside the desired set. Proximal operator can be interpreted as relaxed projection operator. If the regularizer is set to $r(x) = 1_{X}(x)$, where $1_{X}(\cdot)$ is the indicator function of some set $X$, Eq (2.13) becomes the definition of the traditional Euclidean projection. Proximal operator transforms the regularization into a sub-optimization problem that can be efficiently solved for many widely-used regularizers [168]. For example, fast solution to TV regularizer can be achieved by considering its dual formulation [20, 40]. Additionally, proximal operator also corresponds to the regularized least squares of the image denoising problem for AWGN [20]. This equivalence recently prompted a line of work exploring denoisers as regularizers [188, 236], which we will discuss in Part II.

Proximal Gradient Method

Proximal gradient method (PGM), also known as iterative shrinkage/thresholding algorithm (ISTA), is one of the most widely used algorithms in image reconstruction [18, 56, 72]. To understand PGM, consider the optimization problem (2.11), where both $g$ and $r$ are convex,
but where \( r \) is possibly non-differentiable. The iterates of PGM can then be expressed as

\[
\begin{align*}
\mathbf{z}^k &\leftarrow \mathbf{x}^{k-1} - \gamma \nabla g(\mathbf{x}^{k-1}) \\
\mathbf{x}^k &\leftarrow \text{prox}_{\gamma r}(\mathbf{z}^k)
\end{align*}
\]  
(2.14a)
(2.14b)

where \( \gamma > 0 \) is the step-size. Hence, PGM first computes a gradient-descent step with respect to \( g \) and then evaluates the proximal operator of \( r \) defined in Eq. (2.13). Under linear forward models and AWGN, i.e. Eq (2.10), \( \nabla g \) is given as

\[
\nabla g = \mathbf{A}^\top (\mathbf{A} \mathbf{x} - \mathbf{y}),
\]
(2.15)

where \( \top \) denotes the transpose for real-valued matrices or conjugate transpose for complex-valued matrices. When \( \nabla g \) is Lipschitz continuous with constant \( L > 0 \), PGM can be shown to converge for any \( \gamma \in (0, 1/L] \) to a minimizer of the objective function with rate \( O(1/t) \), where \( t \geq 1 \) is the number of PGM iterations [19]. Accelerated proximal gradient method (APGM) is an extension of PGM that includes an additional extrapolation step in each iteration

\[
\begin{align*}
\mathbf{z}^k &\leftarrow \mathbf{s}^{k-1} - \gamma \nabla g(\mathbf{s}^{k-1}) \\
\mathbf{x}^k &\leftarrow \text{prox}_{\gamma r}(\mathbf{z}^k) \\
\mathbf{s}^k &\leftarrow \mathbf{x}^k + \frac{q_{k-1} - 1}{q_k} (\mathbf{x}^k - \mathbf{x}^{k-1}).
\end{align*}
\]  
(2.16a)
(2.16b)
(2.16c)

It is clear that when \( q_k = 1 \) for all \( k \geq 1 \), PGM and APGM are perfectly equivalent. On the other hand, when \( \{q_k\} \) are updated as follows [160]

\[
q_k \leftarrow \frac{1}{2} \left( 1 + \sqrt{1 + 4q^2_{k-1}} \right),
\]
(2.17)
Algorithm 1 (Accelerated) Proximal Gradient Method (PGM/APGM)

1: **input:** Initial input $x^0 = s^0 \in \mathbb{R}^n$, stepsize $\gamma > 0$, and sequence $\{q_k\}_{k \in \mathbb{N}}$ with $q_0 = 0$
2: **for** $k = 1, 2, \ldots$ **do**
3: \quad $z^k \leftarrow s^{k-1} - \gamma \nabla g(s^{k-1})$ \hspace{1cm} \triangleright \text{Gradient of data-fidelity}
4: \quad $x^k \leftarrow \text{prox}_{\gamma r}(z^k)$ \hspace{1cm} \triangleright \text{Proximal of prior}
5: \quad $s^k \leftarrow x^k + ((q_{k-1} - 1)/q_k)(x^k - x^{k-1})$
6: **end for**

* The algorithm is simplified to PGM when $\{q_k\}$ is set to $q_k := 1$.

one can accelerate the convergence of the algorithm [19]. In Eq. 2.17, we set the initial values $q_0 = 1$. In such accelerated settings, it is possible to show that APGM converges to the minimizer of the objective function $f$ with rate $O(1/t^2)$ for any step-size $\gamma \in (0, 1/L]$, which has been proven to be optimal for gradient-based methods [160]. Algorithm 1 summarizes the details of PGM/APGM. The state-of-the-art performance of APGM has been extensively demonstrated in a wide range of areas, including image restoration [41], computational microscopy [106], and other data science applications [83].

**Alternating Directions Method of Multipliers**

PGM/APGM requires the data-fidelity term to be differentiable. However, for some computational imaging tasks, it is desirable to use nondifferentiable $g$ that better characterizes the physical process of the system. *Alternating direction method of multipliers (ADMM)* [168] is one PM algorithm that can accommodate nonsmooth data-fidelities. Similar to the strategy for handling $r$, ADMM relies on the proximal operator of $g$ to circumvent differentiation (See Algorithm 2). To develop ADMM, we consider the following optimization problem over $(x, z)$ equivalent to Eq. (2.11)

$$
\min \ g(z) + r(x) \quad \text{s.t.} \quad z = x.
$$

(2.18)
Algorithm 2 Alternating Direction Method of Multipliers (ADMM)

1: **input:** Initial input \(x^0 = z^0 \in \mathbb{R}^n\), Dual variable \(s^0 = 0\), and penalty parameter \(\gamma > 0\)
2: **for** \(k = 1, 2, \ldots\) **do**
3: \(z^k \leftarrow \text{prox}_{\gamma g}(x^{k-1} - s^{k-1})\) \(\triangleright\) Proximal of data-fidelity
4: \(x^k \leftarrow \text{prox}_{\gamma r}(z^k + s^{k-1})\) \(\triangleright\) Proximal of prior
5: \(s^k \leftarrow s^{k-1} + (z^k - x^k)\)
6: **end for**

This process of introducing an additional variable \(z\) is known as *variable splitting*. To solve this constrained optimization problem, we form the *augmented Lagrangian* [164]

\[
L_{\gamma}(z, x, \mu) = g(z) + r(x) + \mu^T(z - x) + \frac{1}{2\gamma} \|z - x\|^2_2
\]

\[
= g(z) + r(x) + \frac{1}{2\gamma} \|z - x + \gamma \mu\|^2 - \frac{\gamma}{2}\|\mu\|^2_2, \tag{2.19b}
\]

where \(\gamma > 0\) is a parameter and \(\mu \in \mathbb{R}^n\) is the dual variable. We can re-write the augmented Lagrangian by introducing the *scaled dual variable* \(s := \gamma \mu\), which leads to

\[
L_{\gamma}(z, x, s) = g(z) + r(x) + \frac{1}{2\gamma} \|z - x + s\|^2 - \frac{1}{2\gamma} \|s\|^2_2. \tag{2.20}
\]

This optimization problem can be solved via the *method of multipliers* [164] that has the following form for \(k \geq 1\)

\[
(z^k, x^k) \leftarrow \arg \min_{x, z} L_{\gamma}(z, x, s^{k-1}) \tag{2.21a}
\]

\[
s^k \leftarrow s^{k-1} + (z^k - x^k), \tag{2.21b}
\]

starting from \(s^0 = 0\). Note, however, the difficulty of running this algorithm due to the need to jointly minimize over both \(z\) and \(x\). ADMM precisely circumvents this issue by splitting this step into two as follows
\[ z^k \leftarrow \arg \min_{z \in \mathbb{R}^n} L_{\gamma}(z, x^{k-1}, s^{k-1}) \]  \hspace{1cm} (2.22a)

\[ x^k \leftarrow \arg \min_{x \in \mathbb{R}^n} L_{\gamma}(z^k, x, s^{k-1}) \]  \hspace{1cm} (2.22b)

\[ s^k \leftarrow s^{k-1} + (z^k - x^k), \]  \hspace{1cm} (2.22c)

which directly leads to Algorithm 2 by dropping the negative terms. In particular,

\[
\text{prox}_{\gamma g}(x) = \arg \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - x \|_2^2 + \frac{\gamma}{2} \| Az - y \|_2^2 \right\} \\
= [I + \gamma A^T A]^{-1}(x + \gamma A^T y).
\]  \hspace{1cm} (2.23)

The matrix inversion in 2.23 can make ADMM updates computationally expensive for problems where the measurement matrix is not easily invertible. On the other hand, ADMM is known to be fast for matrices that can be inverted efficiently [36]–[38]. When the data-fidelity \( g \) is convex, closed, and proper, but not necessarily differentiable, the iterate \( \{x^k, z^k\} \) of ADMM can converge to the minimizers of the cost function at the rate of \( O(1/t) \) [240], and the difference between \( x^k \) and \( s^k \) will convergence to 0 as \( t \to \infty \). While the provable convergence rate of ADMM is suboptimal relative to the \( O(1/t^2) \) convergence rate of FISTA, ADMM is known to achieve fast empirical convergence within the first few iterations [4]. Successful applications of ADMM to computational imaging problems include image restoration [4], microscopy [97, 178], and medical imaging [3]

### 2.2.2 Learning-based Methods

Past years have witnessed a significant increase of various DL algorithms for computational imaging [138, 145, 165, 238]. The majority adopts the scheme of inverse learning, which refers to training DL models to learn a direct inverse mapping from the measurements to the image
Figure 2.3: Visualization of the widely-adopted DL scheme in computational imaging. bring
the measurement \( y \) to the image domain via simple backprojection and then use a CNN to
further refine the low-quality image \( \tilde{x} \) to generate the final image \( \hat{x} \). A function measuring
the difference between \( \hat{x} \) and groundtruth \( x \) is constructed to optimize the network.

in purely data-driven fashion. A widely-adopted model is to first bring the measurements
\( \{y_i\}_{i=1}^N \) to the image domain via simple backprojection and then use a convolutional neural
network (CNN) to map the resulting low-quality images \( \{\tilde{x}_i\}_{i=1}^N \) to high-quality images
\( \{x_i\}_{i=1}^N \). Here, \( N > 0 \) denotes the total number of training examples. Typically, these CNNs
are trained by minimizing the averaged loss

\[
\ell(\psi) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(F_{\psi}(\tilde{x}_i), x_i),
\]  

(2.24)

where \( F_{\psi} \) denotes the network parametrized by \( \psi \), and function \( \mathcal{L} \) quantifies the discrepancy
between the network output \( F_{\psi}(\tilde{x}_i) \) and \( x_i \). Popular choices for \( \mathcal{L} \) include the \( \ell_1 \)- and
\( \ell_2 \)-norms

\[
\mathcal{L}_1 = \|F_{\psi}(\tilde{x}_i) - x_i\|_1 \quad \text{and} \quad \mathcal{L}_2 = \|F_{\psi}(\tilde{x}_i) - x_i\|_2^2.
\]  

(2.25)

Empirical investigations demonstrate that the \( \ell_1 \)-norm often yields better performance [268].
We next introduce two common CNN architectures, extensively used in many inverse learning
methods.

**U-Net.** U-Net was originally invented by Ronneberger *et al* [190] for conducting image
segmentation tasks. It was Jin *et al* [94] who first adapted this architecture for image
reconstruction and validated its the state-of-the-art performance in the context of X-Ray CT; Later, U-Net has become one of the most common CNN architecture in DL methods for computational imaging tasks. Fig. 2.4 shows the architecture of U-Net introduced in [94]. There are two key properties that recommend U-Net for learning a complicate mapping from the input to the output image:

1. **Multi-resolution decomposition:** The decoder employs a contraction-expansion structure based on the max-pooling and the up-convolution. This means that given a fixed size convolution kernel ($3 \times 3$ in most cases), the effective receptive field of the network increases as the input goes deeper into the network.

2. **Local-global composition:** In each resolution level, the outputs of the convolutional block in the contraction are directly connected and concatenated with the input of the convolutional block in the expansion. The skip connection enables the later layers to reconstruct the feature maps with both the local details and the global texture.
DnCNN. DnCNN is a network architecture that is popular for image denoising. It was first proposed by Zhang et al [262] and validated for the removal of AWGN. DnCNN quickly gains popularity due to its state-of-the-art performance and relatively simpler architecture. Fig. 2.5 shows the architecture of DnCNN. As shown, it only consists of a sequence of convolutional layers with batch normalization, excluding the modules that change the spatial resolution. DnCNN employs residual learning to map the input noisy image to its noise residual, which indicates that the network implicitly removes the latent clean image, reversely representing a general image prior. Empirical evidence indicate that DnCNN can act beyond an AWGN denoiser, being capable of handling other imaging problems such as single image super-resolution and JPEG image deblocking [262].

We note that there are other models considering to directly map \{y_i\} to the reconstructed images \{x_i\}. These methods often adopt mixed network architectures that contain fully connected layers for learning either an approximation of the inverse \((AA^T)^{-1}\) [250] or an inversion to some implicit image manifolds [274]. However, due to the use of fully connected layers, their inputs are restricted to certain pre-defined spatial size (e.g. width & height), leading to undesired inflexibility to deal with size-diverse image datasets.

Unsupervised Extensions. Additionally, various training strategies that avoid the use of high-quality images are proposed for inverse learning methods. One line of work leverages the adversarial training and forward model to jointly enable the cross-domain supervision
via cyclic mapping [109, 119, 123, 130, 203, 208, 269]; Another line of work approximates
the fully supervised training by cross-mapping multiple measurements of the same object
that contain complementary information [12, 35, 67, 69, 84, 92, 116, 117, 120, 206, 231, 257].
More discussion and work can be found in literature [138, 145, 165, 238].

2.2.3 Discussion

We have reviewed two classes of methods: model-based and learning-based methods. Model-
based methods essentially solve the optimization problem translated from Bayesian inference,
while learning-based methods pivots on the DL models trained over a set of data. To better
view the difference, we summarizes the pros and cons of each class as follows:

<table>
<thead>
<tr>
<th>Model-based Methods</th>
<th>Learning-based Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pros:</strong></td>
<td><strong>Pros:</strong></td>
</tr>
<tr>
<td>1. Explicit use of forward model, ensuring tight consistency with measurements.</td>
<td>1. Excellent capability of representing complicate image priors directly from the training data, which often leads to superior performance over model-based methods.</td>
</tr>
<tr>
<td>2. Rigorous derivation from the Bayesian inference, enabling probabilistic interpretations.</td>
<td>2. Potential to bypass the processing of forward model when the latter is computationally intractable or inaccessible.</td>
</tr>
<tr>
<td>3. Robust and reproducible performance cross different objects that obey distinct distributions.</td>
<td></td>
</tr>
<tr>
<td><strong>Cons:</strong></td>
<td><strong>Cons:</strong></td>
</tr>
<tr>
<td>1. Handcrafted regularizers cannot fully characterize distribution of images, often leading to clear visual artifacts.</td>
<td>1. Requiring a large training dataset of (ground-truth) images, and being vulnerable to objects that mismatch the distributions of the training data.</td>
</tr>
<tr>
<td>2. Iterative evaluations of forward model and regularizer can be time-consuming and unscalable to large datasets.</td>
<td>2. Lack of explicit data-consistency, making the solution less interpretable.</td>
</tr>
</tbody>
</table>
Clearly, the major advantage of model-based methods is their interpretability of the final solution and robustness to outliers, while that of learning-based methods is the strong capability of automatically representing complicated image priors from data. On the other hand, handcrafted yet mathematically-sound priors in the former are inflexible in charactering image distribution, while the lack of forward model in the latter leads to the unwanted sensitivity to outliers and difficulty in interpreting the solution.

2.3 Advances in Fusing Physics and Learning

After the discussion of model-based and learning-based methods, a natural question is asked in the field, that is, “Can we fuse model-based and learning-based methods to design new algorithms that maximally inherit the advantages, but minimally involve with the disadvantages from both sides?” To answer this question is the driving force of this dissertation. In this section, we briefly review the current work on combining forward models and learning models.

2.3.1 Iterative Methods using Learning Priors

There are considerable interests in leveraging pre-trained DL models as priors within the traditional model-based methods. The key motivation is to keep the model-based optimization as the backbone for solution interpretability and stability, but allows adaptive priors learned from the data.

Plug-and-Play Priors. The framework of PnP was first proposed by Venkatakrishnan et al. [236]. The key idea is to generalize proximal methods, such as PGM and ADMM, by replacing the proximal operator with an arbitrary AWGN denoiser \( D_\sigma : \mathbb{R}^n \to \mathbb{R}^n \), with \( \sigma > 0 \) controlling the denoising strength. This simple replacement enables PnP to use advanced denoisers, including those based on DNNs [207, 262, 265], for regularizing the inverse problem.
The PnP algorithms have been shown to be effective in various imaging applications [6, 209, 261, 263]. However, $D_\sigma$ may not correspond to any explicit regularizer $r$ in (2.11), in which case PnP loses its interpretation as optimization. In Chapter 3-6, we significantly extend PnP by proposing new algorithms based on stochastic (sub)gradient and using the monotone operator theory to analyze both normal and online variants.

**Regularization by Denoising.** The RED framework, proposed by Romano *et al.* [188], represents an alternative strategy to PnP for imposing denoiser-driven regularization. Different from PnP, where the denoiser is interpreted as a generalized proximal operator, RED incorporates the *noise residual*

$$R(x) = \tau(x - D_\sigma(x)),$$

into iterative model-based algorithms to constrain the solution. RED with deep denoisers has been reported to be effective in image super-resolution [142], phase retrieval [148], and tomographic imaging [249]. When the denoiser $D_\sigma$ is locally homogeneous and has a symmetric Jacobian [185, 188], $R(x)$ corresponds to the gradient of the following regularizer [188]

$$r(x) = \frac{\tau}{2}x^T(x - D_\sigma(x)).$$ (2.26)

It is quite beneficial to have an explicit regularizer because many optimization algorithm becomes immediately usable and their convergence can be analyzed by using the classic convex optimization theory. Nevertheless, it has clarified that many denoisers do not satisfy the assumptions necessary for having an explicit cost [185]. In Chapter 7-9, we extended monotone operator theory to RED and establish more general theoretical results. Additionally, we proposed new algorithms that combine RED with block-coordinate processing for inferring large-scale signals and remain analyzable by our theory.
**Generative Priors.** An alternative family of methods has used deep generative models as the image priors [26, 89, 197]. Here, deep generative models aim to train a generator network \( G : z \rightarrow x \) such that the latent variables \( z \), which follow a simple distribution such as standard Gaussian, are mapped into samples obeying a potentially more complex distribution defined by the training data [200]. Consider the following constrained optimization

\[
\hat{z} = \arg \min_{z \in \mathbb{R}^z} \| \mathbf{A} G(z) - y \|_2^2 \quad \text{s.t.} \quad \hat{x} = G(\hat{z}),
\]  

(2.27)

where \( \mathbf{A} \) denote the linear measurement matrix, \( G \) is a pre-trained generative network, and \( \hat{x} \) is the final solution restricted to the range of \( G \). By leveraging numerical algorithms like gradient descent, we can iteratively search for the optimal encoding \( \hat{z} \) such that the mapped image \( \hat{x} = G(\hat{z}) \) has the smallest distance to the acquired data \( y \) after passing through the forward model. Usually, generative adversarial networks (GAN) are often used as the embodiment of \( G \) due to its excellent performance in capturing complicated distributions. It is worth mentioning that the recovery properties under GAN have been analyzed in the context of compressive sensing [26, 89, 197].

### 2.3.2 Internal Learning/Neural Representation Learning

*Internal learning* is a recent DL paradigm that explores the internal correlation of the test signal for learning a neural network prior solely from data of the testing signal itself. Successful examples include [201, 277] which address image/video spatial or temporal super-resolution by training a DNN to exploit the cross-scale/cross-time similarity within images. In this section, we are particularly interested in methods that consider the following optimization

\[
\hat{\psi} = \arg \min_{\psi} \| \mathbf{A} \mathcal{F}_\psi(z) - y \|_2^2 \quad \text{s.t.} \quad \hat{x} = \mathcal{F}_\psi(z),
\]  

(2.28)
where $z$ is either a random or structural vector. Though similar to each other, the optimization in (2.28) fundamentally differs from that in (2.27) in the sense that the iterate in (2.28) is the network weights $\psi$, while the generative network $G$ in (2.27) is pre-trained and fixed through the minimization. This means that the former internally trains the network $F$ to perform measurement fitting, but the latter searches for the optimal latent encoding $z$ without touching $G$. Recently, the optimization in (2.28) is also referred to as neural representation learning. Next, we introduce two example methods with elaboration of their key benefits.

**Deep image prior.** The first example is deep image prior (DIP), which optimizes a CNN as $F$ to parameterize the reconstructed image $\hat{x} = F_\psi(z)$ [75, 134, 233]. Here, the input $z$ is a vector randomly sampled from simple distributions such as the Gaussian distribution. Discovered in [233, 234], the key benefit of DIP is the use of the implicit prior embedded in the structure of the neural networks that helps to constrain the image. This means that, for example, U-Net will impose different regularization to the latent image from DnCNN as their network architectures are different. The power of such implicit image prior has been demonstrated in computational photography [234], compressive sensing [235], CT/MRI [11, 254], and optical tomography [272].

**Neural Fields.** The second example is neural fields (NF) [251], which also goes by various names in the vision/graphics literature, including neural coordinate-based representations or neural implicit models. Different from DIP, NF considers learning a mapping from the structural vector $z$, i.e. spatiotemporal coordinate, to the 2D/3D field of varying physical quantitates, instances of which include light radiance field [141, 151], image [205, 223], volume [47, 141, 151, 169, 204], video [127, 198], and etc. [251]. Additionally, NF models usually leverage fully-connected networks, also known as multilayer perceptron (MLP), as $F$ to parameterize the physical field. Beyond the implicit regularization, NF enables continuous representation of the latent signal, meaning that one can retrieve the signal value at any
location by simply querying the relevant coordinates. Such feature is potentially beneficial for computational imaging in the sense that it (1) allows the use of customized sampling of the signal when perform testing, and (2) decouples representation from an explicit voxel grid, enabling efficiently storage and processing of large signals. In Chapter 10 and 11, we present a detailed explorations of NF for both image and measurement representation by proposing new algorithms.

2.4 Monotone Operator Theory

Monotone operator theory (MOT) constitutes the foundation of our unified theoretical analysis for PnP/RED algorithms, which is one of the key contributions of this dissertation. MOT is a rich area of nonlinear analysis that has huge impacts on a vast array of disciplines, including functional analysis, partial differential equations, convex optimization, signal and image processing, and machine learning [52]. Recently, applications of MOT to analyzing the properties of DNNs becomes an active research area [70, 79, 153, 195].

In this section, we intended to briefly introduce the definitions and propositions in MOT that are preliminary to our analysis. We note that the results in this section are well-known in the optimization literature and can be found in different forms in standard textbooks [14, 30, 159, 187]. A more complete description of these ideas can be found in literature [13, 168, 192].

2.4.1 Properties of Monotone Operators

**Definition 2.1.** An operator $T$ is Lipschitz continuous with constant $\tau > 0$ if

$$\|Tx - Ty\| \leq \tau \|x - y\|, \quad x, y \in \mathbb{R}^n.$$
When \( \tau = 1 \), we say that \( T \) is nonexpansive. When \( \tau < 1 \), we say that \( T \) is a contraction.

**Definition 2.2.** \( T \) is firmly nonexpansive if

\[
\|T x - T y\|^2 + \|(I - T)x - (I - T)y\|^2 \leq \|x - y\|^2, \quad x, y \in \mathbb{R}^n.
\]

**Definition 2.3.** \( T \) is monotone if

\[
(Tx - Ty)^T(x - y) \geq 0, \quad x, y \in \mathbb{R}^n.
\]

We say that it is strongly monotone or coercive with parameter \( \mu > 0 \) if

\[
(Tx - Ty)^T(x - y) \geq \mu \|x - y\|^2, \quad x, y \in \mathbb{R}^n.
\]

**Definition 2.4.** \( T \) is cocoercive with constant \( \beta > 0 \) if

\[
(Tx - Ty)^T(x - y) \geq \beta \|Tx - Ty\|^2, \quad x, y \in \mathbb{R}^n.
\]

When \( \beta = 1 \), we say that \( T \) is firmly nonexpansive.

The following results are derived from the definition above.

**Proposition 2.1.** Consider \( R = I - T \) where \( T : \mathbb{R}^n \to \mathbb{R}^n \).

\( T \) is nonexpansive \( \iff \) \( R \) is \((1/2)\)-cocoercive.

**Proof.** First suppose that \( R \) is \( 1/2 \) cocoercive. Let \( h := x - y \) for any \( x, y \in \mathbb{R}^n \). We then have

\[
\frac{1}{2} \|Rx - Ry\|^2 \leq (Rx - Ry)^T h = \|h\|^2 - (Tx - Ty)^T h.
\]
We also have that
\[
\frac{1}{2}\|Rx - Ry\|^2 = \frac{1}{2}\|h\|^2 - (Tx - Ty)^T h + \frac{1}{2}\|Tx - Ty\|^2.
\]

By combining these two and simplifying the expression
\[
\|Tx - Ty\| \leq \|h\|.
\]

The converse can be proved by following this logic in reverse.

Proposition 2.2. Consider \( R = I - T \) where \( T : \mathbb{R}^n \to \mathbb{R}^n \).

\( T \) is Lipschitz continuous with constant \( \tau < 1 \Rightarrow R \) is \((1 - \tau)\)-strongly monotone.

Proof. By using the Cauchy-Schwarz inequality, we have for all \( x, y \in \mathbb{R}^n \)
\[
(Rx - Ry)^T (x - y) = \|x - y\|^2 - (Tx - Ty)^T (x - y)
\]
\[
\geq \|x - y\|^2 - \|Tx - Ty\| \|x - y\| \\
\geq \|x - y\|^2 - \tau \|x - y\|^2 \geq (1 - \tau) \|x - y\|^2.
\]

Definition 2.5. For a constant \( \alpha \in (0, 1) \), we say that \( T \) is \( \alpha \)-averaged, if there exists a nonexpansive operator \( N \) such that
\[
T = (1 - \alpha)I + \alpha N.
\]

Proposition 2.3. For a nonexpansive operator \( T \), a constant \( \alpha \in (0, 1) \), and the operator \( R := I - T \), the following are equivalent
Proposition 2.4. For nonexpansive operators $T_1$ and $T_2$ with a constant $\alpha \in (0, 1)$, then the convex combination of the two operators $(1-\alpha)T_1 + \alpha T_2$ is nonexpansive.

Proof. Let $T := (1-\alpha)T_1 + \alpha T_2$. For any $x, y \in \mathbb{R}^n$, we can write
\[
\|T(x) - T(y)\| \leq (1-\alpha)\|T_1(x) - T_1(y)\| + \alpha \|T_2(x) - T_2(y)\| \leq \|x - y\|
\]

Proposition 2.5. Let $T_1$ be $\alpha_1$-averaged and $T_2$ be $\alpha_2$-averaged. Then, the composite operator $T := T_2 \circ T_1 = T_2 T_1$ is
\[
\alpha := \frac{\alpha_1 + \alpha_2 - 2\alpha_1\alpha_2}{1 - \alpha_1\alpha_2}
\] averaged operator.

Proof. See Proposition 4.44 in [13].

Proposition 2.6. Consider $T : \mathbb{R}^n \to \mathbb{R}^n$ and $\beta > 0$. Then, the following are equivalent

(a) $T$ is $\beta$-cocoercive

(b) $\beta T$ is firmly nonexpansive

(c) $I - \beta T$ is firmly nonexpansive.
(d) $\beta T$ is $(1/2)$-averaged.

(e) $I - 2\beta T$ is nonexpansive.

Proof. For any $x, y \in \mathbb{R}^n$, let $h := x - y$. The equivalence between (a) and (b) is readily observed by defining $P := \beta T$ and noting that

$$
(Px - Py)^T h = \beta(Tx - Ty)^T h
$$

and

$$
\|Px - Py\|^2 = \beta^2 \|Tx - Ty\|^2. \quad (2.30)
$$

Define $R := I - P$ and suppose (b) is true, then

$$
(Rx - Ry)^T h = \|h\|^2 - (Px - Py)^T h
$$

$$
= \|Rx - Ry\|^2 + (Px - Py)^T h - \|Px - Py\|^2
$$

$$
\geq \|Rx - Ry\|^2.
$$

By repeating the same argument for $P = I - R$, we establish the full equivalence between (b) and (c).

The equivalence of (b) and (d) can be seen by noting that

$$
2\|Px - Py\|^2 \leq 2(Px - Py)^T h
$$

$$
\Leftrightarrow \|Px - Py\|^2 \leq 2(Px - Py)^T h - \|Px - Py\|^2
$$

$$
= \|h\|^2 - (\|h\|^2 - 2(Px - Py)^T h + \|Px - Py\|^2)
$$

$$
= \|h\|^2 - \|Rx - Ry\|^2.
$$
To show the equivalence with (e), first suppose that $N := I - 2P$ is nonexpansive, then $P = \frac{1}{2}(I + (-N))$ is 1/2-averaged, which means that it is firmly nonexpansive. On the other hand, if $P$ is firmly nonexpansive, then it is 1/2-averaged, which means that from Proposition 2.3(b) we have that $(1 - 2)I + 2P = 2P - I = -N$ is nonexpansive. This directly means that $N$ is nonexpansive.

\[\square\]

### 2.4.2 Properties of Block-Coordinate Monotone Operators

In the analysis of RED, we extend MOT to block coordinate settings. For completeness, we summarize the key results useful for our analysis by restating them in a block-coordinate form. Most of the concepts in this part come from the traditional monotone operator theory [14, 192] adapted for block-coordinate operators.

To begin with, we consider the decomposition of $\mathbb{R}^n$ into $b \geq 1$ subspaces

$$\mathbb{R}^n = \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \cdots \times \mathbb{R}^{n_b} \quad \text{with} \quad n = n_1 + n_2 + \cdots + n_b.$$  

For each $i \in \{1, \ldots, b\}$, we define the matrix $U_i : \mathbb{R}^{n_i} \to \mathbb{R}^n$ that injects a vector in $\mathbb{R}^{n_i}$ into $\mathbb{R}^n$ and its transpose $U_i^T$ that extracts the $i$th block from a vector in $\mathbb{R}^n$. Then, for any $x = (x_1, \ldots, x_b) \in \mathbb{R}^n$

$$x = \sum_{i=1}^{b} U_i x_i \quad \text{with} \quad x_i = U_i^T x \in \mathbb{R}^{n_i}, \ i = 1, \ldots, b$$

which is equivalent to $\sum_{i=1}^{b} U_i U_i^T = I$. Note that the above equation directly implies the norm preservation $\|x\|_2^2 = \|x_1\|_2^2 + \cdots + \|x_b\|_2^2$ for any $x \in \mathbb{R}^n$.  

35
Definition 2.6. We define the block-coordinate operator \( T_i : \mathbb{R}^n \to \mathbb{R}^{n_i} \) of \( T : \mathbb{R}^n \to \mathbb{R}^n \) as

\[
T_i x := [T x]_i = U_i^T T x \in \mathbb{R}^{n_i}, \quad x \in \mathbb{R}^n.
\]

The operator \( T_i \) applies \( T \) to its input vector and then extracts the subset of outputs corresponding to the coordinates in the block \( i \in \{1, \ldots, b\} \).

Remark. When \( b = 1 \), we have that \( n = n_1 \) and \( U_1 = U_1^T = I \). Then, all the properties in this section reduce to their standard counterparts from the monotone operator theory in \( \mathbb{R}^n \). In such settings, we simply drop the word block from the name of the property.

Definition 2.7. \( T_i \) is block Lipschitz continuous with constant \( \tau_i > 0 \) if

\[
\| T_i x - T_i y \| \leq \tau_i \| h_i \|, \quad x = y + U_i h_i, \quad y \in \mathbb{R}^n, h_i \in \mathbb{R}^{n_i}.
\]

When \( \tau_i = 1 \), we say that \( T_i \) is block nonexpansive.

Definition 2.8. An operator \( T_i \) is block cocoercive with constant \( \beta_i > 0 \) if

\[
(T_i x - T_i y)^T h_i \geq \beta_i \| T_i x - T_i y \|^2,
\]

where \( x = y + U_i h_i, \ y \in \mathbb{R}^n, h_i \in \mathbb{R}^{n_i} \). When \( \beta_i = 1 \), we say that \( T_i \) is block firmly nonexpansive.

The following propositions are conclusions derived from the definition of above.

Proposition 2.7. Let \( T_{ij} : \mathbb{R}^n \to \mathbb{R}^{n_i} \) for \( j \in J \) be a set of block nonexpansive operators. Then, their convex combination

\[
T_i := \sum_{j \in J} \theta_j T_{ij}, \quad \text{with} \quad \theta_j > 0 \text{ and } \sum_{j \in J} \theta_j = 1,
\]
is nonexpansive.

**Proof.** By using the triangular inequality and the definition of block nonexpansiveness, we obtain

\[
\|T_i x - T_i y\| \leq \sum_{j \in J} \theta_j \|T_{ij} x - T_{ij} y\|
\]

\[
\leq \left( \sum_{j \in J} \theta_j \right) \|h_i\| = \|h_i\|
\]

for all \( y \in \mathbb{R}^n \) and \( h_i \in \mathbb{R}^{n_i} \) where \( x = y + U_i h_i \).

**Proposition 2.8.** Consider \( R_i = U_i^T - T_i \) where \( T_i : \mathbb{R}^n \to \mathbb{R}^{n_i} \).

\( T_i \) is block nonexpansive \( \iff \) \( R_i \) is \((1/2)\)-block cocoercive.

**Proof.** First suppose that \( R_i \) is \( 1/2 \) block cocoercive. Let \( x = y + U_i h_i \) for all \( y \in \mathbb{R}^n \) and \( h_i \in \mathbb{R}^{n_i} \). We then have

\[
\frac{1}{2} \|R_i x - R_i y\|^2 \leq \langle R_i x - R_i y, h_i \rangle = \|h_i\|^2 - \langle T_i x - T_i y, h_i \rangle.
\]

We also have that

\[
\frac{1}{2} \|R_i x - R_i y\|^2 = \frac{1}{2} \|h_i\|^2 - \langle T_i x - T_i y, h_i \rangle + \frac{1}{2} \|T_i x - T_i y\|^2.
\]

By combining these two and simplifying the expression, we obtain that

\[
\|T_i x - T_i y\| \leq \|h_i\|.
\]

The converse can be proved by following this logic in reverse. \( \square \)
2.4.3 Operator Properties for Convex Function, Subdifferentials, and Proximal Operators

It is convenient to link properties of a function \( f : \mathbb{R}^n \to \mathbb{R} \), \( x \mapsto y = f(x) \), to the properties of operators derived from it. The key properties for our analysis are related to continuity and convexity.

**Proposition 2.9.** Let \( f \) be continuously differentiable function with \( \nabla f \) that is \( L \)-Lipschitz continuous. Then,

\[
f(y) \leq f(x) + \nabla f(x)^{\top}(y - x) + \frac{L}{2}\|y - x\|^2
\]

\[
= f(x) + \nabla f(x)^{\top}h + \frac{L}{2}\|h\|^2
\]

for all \( x \in \mathbb{R}^n \) and \( h \in \mathbb{R}^n \), where \( y = x + h \). The above inequality also holds for block \( L_i \)-Lipschitz \( \nabla_i f \)

\[
f(y) \leq f(x) + \nabla f(x)^{\top}(y - x) + \frac{L_i}{2}\|y - x\|^2
\]

\[
= f(x) + \nabla_i f(x)^{\top}h_i + \frac{L_i}{2}\|h_i\|^2
\]

where \( y = x + \cup_i h_i \) with \( h_i \in \mathbb{R}^{n_i} \)

**Proof.** The proof is a minor variation of the one presented in Section 2.1 of [159]. \( \square \)

**Proposition 2.10.** Consider a continuously differentiable \( f \) such that \( \nabla f \) is \( L \)-Lipschitz continuous. Let \( x^* \in \mathbb{R}^n \) denote the global minimizer of \( f \). Then, we have that

\[
\frac{1}{2L}\|\nabla f(x)\|^2 \leq (f(x) - f(x^*)) \leq \frac{L}{2}\|x - x^*\|^2;
\]

(2.31)
where \( x = x^* + h, \ x \in \mathbb{R}^n, \ h \in \mathbb{R}^n \). Similarly, we have

\[
\frac{1}{2L} \| \nabla_i f(x) \|^2 \leq (f(x) - f(x^*)) \leq \frac{L}{2} \| x - x^* \|^2; \tag{2.32}
\]

where \( x = x^* + \cup_i h_i, \ h_i \in \mathbb{R}^{n_i} \).

Proof. The proof is a minor variation of the discussion in Section 9.1.2 of [30]. \(\square\)

**Proposition 2.11.** For a convex and continuously differentiable function \( f \), we have

\[
\nabla f \text{ is } L\text{-Lipschitz continuous} \iff \nabla f \text{ is } (1/L)\text{-cocoercive.}
\]

The above proposition also holds for block-Lipschitz \( \nabla_i f \).

\[
\nabla_i f \text{ is block } L_i\text{-Lipschitz continuous} \iff \nabla_i f \text{ is block } (1/L_i)\text{-cocoercive.}
\]

Proof. The proof is a minor variation of the one presented as Theorem 2.1.5 in Section 2.1 of [159]. \(\square\)

**Proposition 2.12.** For a continuously differentiable function \( f \) and a constant \( M > 0 \), we have

\[
\nabla f \text{ is strongly monotone with a constant} \iff f \text{ is strongly convex with a constant } M.
\]

Proof. A proof can be found in Theorem 2.1.9 of [159]. \(\square\)
Proposition 2.13. Let $f$ be a proper, closed, and convex function. Then for all $x, y \in \mathbb{R}^n$, $g \in \partial f(x)$, and $h \in \partial f(y)$, $\partial f$ is a monotone operator

$$(g - h)^T (x - y) \geq 0.$$ 

Additionally if $f$ is strongly convex with constant $\mu > 0$, then $\partial f$ is strongly monotone with the same constant.

$$(g - h)^T (x - y) \geq \mu \| x - y \|^2.$$

Proof. Consider a strongly convex function $f$ with a constant $\mu \geq 0$. Then, we have that

$$\begin{cases}
    f(y) \geq f(x) + g^T (y - x) + \frac{\mu}{2} \| y - x \|^2 \\
    f(x) \geq f(y) + h^T (x - y) + \frac{\mu}{2} \| x - y \|^2 \\
    \Rightarrow \quad (g - h)^T (x - y) \geq \mu \| x - y \|^2.
\end{cases}$$

The proof for a weakly convex $f$ is obtained by considering $\mu = 0$ in the inequalities above.

It is well-known that the proximal operator is firmly nonexpansive.

Proposition 2.14. Proximal operator $\text{prox}_{\gamma f}$ of a proper, closed, and convex $f$ is firmly nonexpansive.
Proof. Denote with $x_1 = Gz_1 = \text{prox}_{\gamma f}(z_1)$ and $x_2 = Gz_2 = \text{prox}_{\gamma f}(z_2)$, then

$$\begin{cases}
(z_1 - x_1) \in \gamma \partial f(x_1) \\
(z_2 - x_2) \in \gamma \partial f(x_2)
\end{cases} \Rightarrow (z_1 - x_1 - z_2 + x_2)^T(x_1 - x_2) \geq 0$$

$$\Rightarrow (Gz_1 - Gz_2)^T(z_1 - z_2) \geq \|Gz_1 - Gz_2\|^2.$$

The following proposition is sometimes referred to as Moreau-Rockafellar theorem. It establishes that for functions defined over all of $\mathbb{R}^n$, we have that $\partial f = \partial f_1 + \cdots + \partial f_m$.

**Proposition 2.15.** Consider $f = f_1 + \cdots + f_m$, where $f_1, \ldots, f_m$ are proper, closed, and convex functions on $\mathbb{R}^n$. Then

$$\partial f_1(x) + \cdots + \partial f_m(x) \subset \partial f(x) \quad x \in \mathbb{R}^n.$$

Moreover, suppose that convex sets $\text{ri}(\text{dom} f_i)$ have a point in common, then we also have

$$\partial f(x) \subset \partial f_1(x) + \cdots + \partial f_m(x) \quad x \in \mathbb{R}^n.$$

**Proof.** See Theorem 23.8 in [186].

**Proposition 2.16.** Let $f$ be a convex function, then we have that

$$f \text{ is Lipschitz continuous with constant } L > 0 \iff \|g(x)\| \leq L, \quad g(x) \in \partial f(x) \quad x \in \mathbb{R}^n.$$
Proof. First assume that \( \|g(x)\| \leq L \) for all subgradients. Then, from the definition of subgradient

\[
\begin{align*}
    f(x) &\geq f(y) + g(y)^T(x - y) \\
    f(y) &\geq f(x) + g(x)^T(y - x)
\end{align*}
\]

\[\iff g(y)^T(x - y) \leq f(x) - f(y) \leq g(x)^T(x - y).\]

Then, from Cauchy-Schwarz inequality, we obtain

\[-L\|x - y\| \leq -\|g(y)\|\|x - y\| \leq f(x) - f(y) \leq \|g(x)\|\|x - y\| \leq L\|x - y\|.

Now assume that \( g \) is \( L \)-Lipschitz continuous. Then, we have for any \( x, y \in \mathbb{R}^n \)

\[g(x)^T(y - x) \leq f(y) - f(x) \leq L\|y - x\|.

Consider \( v = y - x \neq 0 \), then we have that

\[g(x)^T\left(\frac{v}{\|v\|}\right) \leq L.

Since, this must be true for any \( v \neq 0 \), we directly obtain \( \|g(x)\| \leq L. \)

### 2.4.4 Moreau smoothing and proximal operators

In this section, we consider a class of functions that are proper, closed, and convex, but are not necessarily differentiable. The proximal operator is a widely-used concept in such nonsmooth optimization problems [154, 187].
Definition 2.9. Consider a proper, closed, and convex $h$ and a constant $\mu > 0$. We define the proximal operator

$$\text{prox}_{\mu h}(x) := \arg \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - x \|^2 + \mu h(z) \right\}$$

and the Moreau envelope

$$h_{\mu}(x) := \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - x \|^2 + \mu h(z) \right\}.$$

Proposition 2.17. The function $h_{\mu}$ is convex and continuously differentiable with a 1-Lipschitz gradient

$$\nabla h_{\mu}(x) = x - \text{prox}_{\mu h}(x), \quad x \in \mathbb{R}^n.$$

Proof. We first show that $h_{\mu}$ is convex. Consider

$$q(x, z) := \frac{1}{2} \| z - x \|^2 + \mu h(z),$$

which is convex $(x, z)$. Then, for any $0 \leq \theta \leq 1$ and $(x_1, z_1), (x_2, z_2) \in \mathbb{R}^{2n}$, we have

$$h_{\mu}(\theta x_1 + (1 - \theta)x_2) \leq q(\theta x_1 + (1 - \theta)x_2, \theta z_1 + (1 - \theta)z_2)$$

$$\leq \theta q(x_1, z_1) + (1 - \theta)q(x_2, z_2), \quad (2.33)$$

where we used the convexity of $q$. Since this inequality holds everywhere, we have

$$h_{\mu}(\theta x_1 + (1 - \theta)x_2) \leq \theta h_{\mu}(x_1) + (1 - \theta)h_{\mu}(x_2),$$

with

$$h_{\mu}(x_1) = \min_{z_1} q(x_1, z_1) \quad \text{and} \quad h_{\mu}(x_2) = \min_{z_2} q(x_2, z_2).$$
To show the differentiability, note that
\[
    h_\mu(x) = \frac{1}{2}\|x\|^2 - \max_{z \in \mathbb{R}^n} \left\{ x^T z - \mu h(z) - \frac{1}{2}\|z\|^2 \right\}
\]
\[
    = \frac{1}{2}\|x\|^2 - \phi^*(x) \quad \text{with} \quad \phi(z) := \frac{1}{2}\|z\|^2 + \mu h(z),
\]
where \( \phi^* \) denotes the conjugate of \( \phi \). The function \( \phi \) is closed and 1-strongly convex. Hence, we know that \( \phi^* \) is defined for all \( x \in \mathbb{R}^n \) and is differentiable with gradient [30]

\[
    \nabla \phi^*(x) = \arg \max_{z \in \mathbb{R}^n} \left\{ x^T z - \mu h(z) - \frac{1}{2}\|z\|^2 \right\} = \text{prox}_{\mu h}(x).
\]

Hence, we conclude that

\[
    \nabla h_\mu(x) = x - \nabla \phi^*(x) = x - \text{prox}_{\mu h}(x).
\]

Note that since the proximal operator is firmly nonexpansive, \( \nabla h_\mu \) is also firmly nonexpansive, which means that it is 1-Lipschitz.

The next result shows that the Moreau envelope can serve as a smooth approximation to a nonsmooth function.

**Proposition 2.18.** Consider \( h \in \mathbb{R}^n \) and its Moreau envelope \( h_\mu(x) \) for \( \mu > 0 \). Then,

\[
    0 \leq h(x) - \frac{1}{\mu} h_\mu(x) \leq \frac{\mu}{2} G_x^2
\]

with \( G_x^2 := \min_{x_i \in \partial h(x)} \|x_i\|^2, \quad x \in \mathbb{R}^n \).

**Proof.** First note that

\[
    \frac{1}{\mu} h_\mu(x) = \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2\mu}\|z - x\|^2 + h(z) \right\} \leq h(x), \quad x \in \mathbb{R}^n,
\]
which is due to the fact that $z = x$ is potentially suboptimal. We additionally have for any $x_i \in \partial h(x)$

$$h_{\mu}(x) - \mu h(x) = \min_{z \in \mathbb{R}^n} \left\{ \mu h(z) - \mu h(x) + \frac{1}{2} \| z - x \|^2 \right\}$$

$$\geq \min_{z \in \mathbb{R}^n} \left\{ \mu x_i^T (z - x) + \frac{1}{2} \| z - x \|^2 \right\}$$

$$= \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - (x - \mu x_i) \|^2 - \frac{\mu^2}{2} \| x_i \|^2 \right\} = -\frac{\mu^2}{2} \| x_i \|^2.$$

This directly leads to the conclusion.
Part II

Plug-and-Play Priors
Chapter 3

Overview

Plug-and-Play Priors (PnP) is a flexible methodology for imposing statistical priors without explicitly forming an objective function [209, 236]. PnP algorithms alternate between imposing data consistency by minimizing a data-fidelity term and imposing a statistical prior through the manifestation of an image denoiser. Recently, the popularity of deep learning has led to a wide adoption of PnP for exploiting learned priors specified through pre-trained deep neural networks (DNN), leading to the state-of-the-art performance in a variety of applications [6, 62, 242, 261, 263]. In this chapter, we introduce the basic formulations of two widely-used PnP algorithms: PnP-PGM and PnP-ADMM, followed by a discussion of the challenges in the existing PnP algorithms.

3.1 From Proximal Operator to Image Denoiser

Consider the reconstruction problem of an unknown image $x \in \mathbb{R}^n$ from a set of noisy measurements $y \in \mathbb{R}^m$. The task is frequently formulated as an optimization problem shown
Algorithm 3 PGM/APGM

1: input: $x^0 = s^0 \in \mathbb{R}^n$, $\gamma > 0$, $\sigma > 0$, and 
   $\{q_k\}_{k \in \mathbb{N}}$
2: for $k = 1, 2, \ldots$ do
3: \hspace{1em} $z^k \leftarrow s^{k-1} - \gamma \nabla d(s^{k-1})$
4: \hspace{1em} $x^k \leftarrow \text{prox}_\gamma(z^k)$
5: \hspace{1em} $s^k \leftarrow x^k + ((q_k - 1)/q_k)(x^k - x^{k-1})$
6: end for

Algorithm 4 ADMM

1: input: $x^0 \in \mathbb{R}^n$, $s^0 = 0$, $\gamma > 0$, and
   $\sigma > 0$
2: for $k = 1, 2, \ldots$ do
3: \hspace{1em} $z^k \leftarrow \text{prox}_{\gamma d}(x^{k-1} - s^{k-1})$
4: \hspace{1em} $x^k \leftarrow \text{prox}_{\gamma r}(z^k + s^{k-1})$
5: \hspace{1em} $s^k \leftarrow s^{k-1} + (z^k - x^k)$
6: end for

in Eq. (2.11), and we recall

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} f(x) \quad \text{with} \quad f(x) = g(x) + r(x),$$

where $g$ is the data-fidelity term that penalizes the mismatch to the measurements and $r$ is the regularizer that imposes prior knowledge regarding the unknown image. PnP draws its inspiration from the proximal algorithms extensively used in nonsmooth composite optimization [168] (see discussion in Section 2.2.1). Nevertheless, unlike proximal algorithms which rely on regularizers having explicit formulations, PnP algorithms use denoising functions, which may not be linked to explicit regularizers, to regularize the solution image. Consider the original PGM and ADMM summarized in Algorithm 3 and 4. Both algorithms have modular structures in the sense that the prior on the image is only imposed via the proximal operator, which is essentially solving a regularized image denoising problem of AWGN

$$\begin{equation}
\mathbf{y} = \mathbf{x} + \mathbf{e} \quad \overset{\text{solve}}{\underset{\mathbf{x} \in \mathbb{R}^n}{\arg\min}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|^2_2 + \mu r(\mathbf{x}) \right\}. \tag{3.1}
\end{equation}$$

To see this, consider the first squared error term as the data-fidelity for the image denoising and the second term $r$ as the regularizer. Compared with the formulation in (2.10), the proximal operator in (3.1) is in-fact a regularized least-square estimator with forward model matrix $A$. 

48
being the identity. By viewing proximal operators as image denoisers, it is clear that the key idea of PnP [236] is to replace the proximal operator with a more general denoising operator $D_\sigma(\cdot)$ for imposing regularization. This generalization is meaningful because it extends the explicit image priors to the implicit ones characterized by the state-of-the-art image denoisers (e.g. BM3D [55], TNRD [46], and DnCNN [262]) that often achieves superior denoising performance. Extensive results have demonstrated the superior imaging quality achieved by PnP over the classical model-based methods in various imaging inverse problems [6, 62, 242, 261, 263].

**PnP-PGM & PnP-PGM**

While the original formulation of PnP [236] relies on ADMM, it has been shown that it can be as effective when used with other proximal algorithms [103, 147, 166], or with another class of algorithms known as approximate message passing (AMP) [74, 149, 150]. AMP-based algorithms have been shown to be effective for problems where $A$ is large and random [16, 64], but are also known to be unstable for general matrices $A$ [38, 181, 182]. Therefore, in this dissertation, our focus will be exclusively on the variants of PnP based on PGM and ADMM, summarized in Algorithm 5 and 6, respectively.
Figure 3.1: Visualization of a single iteration in PnP-PGM and PnP-ADMM.

Figure 3.1 visualizes a single iteration of the PnP-PGM and PnP-ADMM algorithm. Intuitively, we can interpret the iteration of PnP as a two-step procedure: the algorithm first computes intermediate iterate \( z \) from the input \( x \) along the descending direction of the data-fidelity term \( g \), and then denoises \( z \) such that the next iterate \( x^+ \) is pulled towards to set of noise-free images defined by \( D_\sigma \). By iteratively evaluating the above procedure, PnP algorithm can numerically compute an image that best balances the data-fidelity term and denoising regularizer.

3.2 Challenges

Despite the progress, two challenges still remain in the existing PnP algorithms.

1. **Lack of scalability.** Current PnP algorithms for solving the problem are iterative *batch* procedures, which means that they use the full set of measurements at every iteration. This effectively precludes their application to very large datasets [28] common in 3D imaging or in imaging of dynamic objects [59, 107]. The growing size of the measurement in modern computational imaging applications calls for scalable algorithm to efficiently process the data.

2. **Lack of theoretical interpretation.** Since the objective function is missing for denoisers, it remains a challenge to analyze its convergence under different settings.
Although some prior work addressed this issue by imposing symmetric-gradient [209] and bounded denoiser [42], counter examples has shown that they are either unnecessary or insufficient.

In the following chapters, we simultaneously address these two challenges by proposing new online/incremental PnP algorithms with new fixed-point analysis that interprets both original formulation and its scalable variants.
Chapter 4

Online Plug-and-Play Proximal Gradient Method

4.1 Introduction

Using online gradients in first-order algorithms has been extensively studied in stochastic optimization and machine learning for processing big data [28]. In this chapter, we introduce a new online PnP algorithm called online plug-and-play proximal gradient method (PnP-OPGM) by incorporating the online procedure. At every iteration, the proposed algorithm uses only a subset of measurements, making it scalable to processing a large number of measurements. We present a new theoretical convergence analysis, for both batch and online variants of PnP-PGM, for denoisers that do not necessarily correspond to proximal operators. We also present simulations illustrating the applicability of the algorithm to image reconstruction in diffraction tomography. The results in this chapter have the potential to
expand the applicability of the PnP framework to very large datasets. Specifically, our key contributions are two-fold.

- We present a detailed theoretical convergence analysis of batch PnP-PGM under a set of explicit assumptions. Our analysis complements the recent theoretical results on PnP-ADMM by Sreehari et al. [209] and Chan et al. [42] in two major ways. We show that for PnP-PGM the symmetric gradient assumption from [209] is not necessary, while the bounded denoiser assumption from [42] is not sufficient to establish the convergence.

- We extend the traditional batch PnP framework with our novel online algorithm PnP-OPGM. We prove the theoretical convergence of the algorithm to the same set of fixed points as batch PnP-PGM and PnP-ADMM. This makes PnP-OPGM a powerful and theoretically sound alternative for large-scale image reconstruction. We also illustrate its applicability with several numerical simulations on image reconstruction problems encountered in diffraction tomography [98].

4.2 Related Work

Several prior publications have analyzed the theoretical convergence of PnP algorithms [42, 147, 209, 225]. Sreehari et al. [209] have established the convergence of PnP-ADMM to the global minimum of some implicitly defined objective function. Specifically, by building on the theoretical analysis by Moreau [154], they show that $D_\sigma$ is a valid proximal operator of some implicit regularizer if it is nonexpansive and $\nabla D_\sigma(x)$ is a symmetric matrix for all $x \in \mathbb{R}^n$. Chan et al. [42] have proved a fixed-point convergence of PnP-ADMM for bounded denoisers, which are defined as denoisers satisfying

$$\frac{1}{n}\|D_\sigma(x) - x\|_2^2 \leq \sigma^2 c,$$

(4.1)
for any $x \in \mathbb{R}^n$, where $c > 0$ is a constant independent of $n$ and $\sigma$. Meinhardt et al. [147] have shown that for continuous denoisers several PnP algorithms admit an equivalent fixed-point iteration. Recently, Teodoro et al. [225] considered a special class of denoisers based on Gaussian mixture models (GMMs) and showed that PnP-ADMM converges when the GMM denoiser is simplified to be a linear function of its input.

Our theoretical analysis is also closely related to the convergence results established for first-order methods by Nesterov [159] and Beck and Teboulle [18]. In particular, our work is related to inexact proximal-gradient optimization that has been extensively investigated by several researchers [23, 61, 65, 78, 99, 102, 194, 275]. We extend this prior work beyond traditional optimization, where denoising operators do not necessarily correspond to proximal operators of a given objective. To achieve this, we adopt the monotone operator theory [13, 192], which enables a unified analysis of PnP methods by expressing them as finding zeros of an operator.

### 4.3 Batch PnP-PGM

In this section, we present a detailed theoretical convergence analysis of batch PnP-PGM. The results are based on the fixed point analysis of Algorithm 5 and rely on basic convex and monotone analysis, summarized in Section 2.4.1.

The central building block of PnP-PGM is the following denoiser-gradient operator

$$P(x) := D_\sigma(x - \gamma \nabla g(x)),$$  \hspace{1cm} (4.2)

which first computes the gradient-step with respect to the function $d$ and then denoises the result with a given denoiser. Throughout this chapter, we assume that the function $d$ is
convex and has a Lipschitz continuous gradient with constant \( L > 0 \). We are interested in convergence of Algorithm 5 to the set of fixed points of the operator \( P \)

\[ \text{fix}(P) := \{ x \in \mathbb{R}^n : x = P(x) \}. \tag{4.3} \]

Note that when \( D_\sigma \) is the proximal operator of a convex function, \( \text{fix}(P) \) coincides with the set of solutions of (3.1).

**Proposition 4.1.** Let \( D_\sigma(\cdot) = \text{prox}_{\gamma r}(\cdot) \) for \( \gamma, \sigma > 0 \). Then \( x^* \in \text{fix}(P) \) if and only if it minimizes \( f = g + r \).

**Proof.** See Appendix A.1. \( \square \)

Our central goal, however, is to generalize \( D_\sigma \) beyond proximal operators. The key assumption that we adopt for our analysis is that the denoiser is averaged (see Appendix A).

**Definition 4.1.** Consider an operator \( D_\sigma \) and a constant \( \theta \in (0, 1) \). \( D_\sigma \) is \( \theta \)-averaged if and only if the operator \((1 - 1/\theta)I + (1/\theta)D_\sigma\), where \( I \) denotes the identity operator, is nonexpansive.

The class of averaged operators is a superset of proximal operators and a subset of nonexpansive operators. In fact, the proximal operator is an averaged operator with \( \theta = 1/2 \). Note that given any nonexpansive denoiser, it is always possible to make it averaged by defining a damped operator \( D := (1 - \theta)I + \theta D_\sigma \), with \( \theta \in (0, 1) \), which has the same set of fixed points as \( D_\sigma \) [168].

**Assumption 4.1.** We analyze PnP-PGM under the following assumptions:

(a) The function \( d \) is convex and differentiable with a Lipschitz continuous gradient of constant \( L > 0 \).
(b) $D_\sigma$ is $\theta$-averaged with $\theta \in (0, 1)$ for any $\sigma > 0$.

c) There exists $x^* \in \mathbb{R}^n$ such that $x^* \in \text{fix}(P)$

We can then establish the following convergence result.

**Proposition 4.2.** Run PnP-PGM for $t \geq 1$ iterations under Assumption 4.1 with step-size $\gamma \in (0, 1/L]$ and $q_k = 1$ for all $k \in \{1, \ldots, t\}$. Then, for any $x^* \in \text{fix}(P)$, we have that

$$\frac{1}{t} \sum_{k=1}^{t} \|x^{k-1} - P(x^{k-1})\|^2_2 \leq \frac{2}{t} \left(\frac{1 + \theta}{1 - \theta}\right) R^2,$$

where $R > 0$ is a constant such that $\|x^0 - x^*\|_2 \leq R$.

**Proof.** See Appendix A.2. \qed

The direct consequence of Proposition 4.2 is that

$$\min_{k \in \{1, \ldots, t\}} \left\{\|x^{k-1} - P(x^{k-1})\|^2_2\right\} = O(1/t), \quad (4.4)$$

that is under Assumption 4.1, the iterates of PnP-PGM can get arbitrarily close to the set of fixed points $\text{fix}(P)$ with rate $O(1/t)$. This result is different from the traditional monotonic $O(1/t)$ convergence of PGM to the minimum of an objective function [18]. The convergence in (4.4) is not monotonic and is expressed in terms of the smallest distance to $x = P(x)$ in the window of $t \geq 1$ previous iterations. This is because PnP-PGM is not necessarily minimizing any objective function. On the other hand, the result still guarantees that, given a sufficient number of $t \geq 1$ iterations, the iterates of PnP-PGM can get arbitrarily close to the set $\text{fix}(P)$.  

56
Recently, Meinhardt et al. [147] have showed that for continuous denoisers, the fixed-points of several PnP algorithms coincide. The following proposition is a minor variant of their result tailored for PnP-ADMM.

**Proposition 4.3.** *Under Assumption 4.1, the set of fixed-points of PnP-ADMM coincides with* $\text{fix}(P)$. 

*Proof.* See Appendix A.3. □

In the context of the work by Sreehari et al. [209], the propositions above indicate that the symmetric gradient assumption is not necessary for the convergence of PnP-PGM. Since the symmetry of $\nabla D_\sigma(x)$ in [209] ensures that the denoiser is an implicitly defined proximal operator, the results here provide a generalization of the convergence beyond proximal operators. Moreover, both PnP-PGM and PnP-ADMM are equivalent in the sense that they have the same set of solutions specified by $\text{fix}(P)$.

The bounded denoiser assumption (4.1) is a more relaxed assumption on the denoising operator and was used to analyze PnP-ADMM. However, we argue that it is not sufficient to guarantee the convergence of PnP-PGM. The following proposition builds on a specific counter example.

**Proposition 4.4.** *There exists a function* $g$ *that is convex and has a Lipschitz continuous gradient of constant* $L$, *and a denoiser* $D_\sigma$ *that satisfies* (4.1), *such that PnP-PGM with the step* $\gamma \in (0, 1/L)$, $q_k = 1$ *for all* $k \in \mathbb{N}$, *and* $\sigma > \gamma / \sqrt{c}$ *diverges.*

*Proof.* See Appendix A.4. □

Definition 4.1 makes verifying that a denoiser is averaged equivalent to verifying nonexpansiveness of some operator. As was argued in several recent publications [42, 209, 225] the task
is more difficult for some denoisers than it is for others and there exist denoisers for which this condition does not hold. However, all recently designed denoisers for PnP from [209, 225] satisfy our assumptions. In fact, the denoisers that satisfy conditions outlined in [209] correspond to implicit proximal operators, which implies that they are $\theta = 1/2$ averaged operators. For example, the modified nonlocal means (NLM) filter specifically designed in [209] is by definition an averaged operator.

### 4.4 Online PnP-PGM

We now introduce our second key contribution: the new online variant of PnP-PGM called PnP-OPGM. We additionally prove its convergence for averaged denoisers.

In many imaging applications, the data-fidelity term $d$ consists of a large number of component functions

$$
g(x) = \mathbb{E}[g_i(x)] = \frac{1}{I} \sum_{i=1}^{I} g_i(x), \quad (4.5)$$

where each $d_i$ typically depends only on the subset $y_i$ of the measurements in $y$. For example, in tomographic imaging each $y_i$ corresponds to a single projection of an object along a specific angle [98]. Note that in equation (4.5), the expectation is taken over a uniformly distributed random variable $i \in \{1, \ldots, I\}$. The computation of the gradient

$$
\nabla g(x) = \mathbb{E}[\nabla g_i(x)] = \frac{1}{I} \sum_{i=1}^{I} \nabla g_i(x), \quad (4.6)
$$

scales with the total number of components $I$, which means that when the latter is large, the memory requirements or computation time of the classical batch PnP algorithms may become impractical. The central idea of PnP-OPGM, summarized in Algorithm 7, is to approximate
Algorithm 7 Plug-and-Play Online Proximal Gradient Method (PnP-OPGM)

1: **input:** $x^0 = s^0 \in \mathbb{R}^n$, $\gamma > 0$, $\sigma > 0$, $\{q_k\}$, and $B \geq 1$
2: **for** $k = 1, 2, \ldots$ **do**
3: $\hat{\nabla}d(s^{k-1}) \leftarrow \text{minibatchGradient}(s^{k-1}, B)$
4: $z^k \leftarrow s^{k-1} - \gamma \hat{\nabla}g(s^{k-1})$
5: $x^k \leftarrow D_{\sigma}(z^k)$
6: $s^k \leftarrow x^k + ((q_{k-1} - 1)/q_k)(x^k - x^{k-1})$
7: **end for**

the gradient at every iteration with an average of $B \ll I$ component gradients

$$\hat{\nabla}g(x) = \frac{1}{B} \sum_{b=1}^{B} \nabla g_{i_b}(x), \quad (4.7)$$

where $i_1, \ldots, i_B$ are independent random indices that are distributed uniformly over $\{1, \ldots, I\}$. The minibatch size parameter $B \geq 1$ controls the number of gradient components used at every iteration.

**Assumption 4.2.** We analyze PnP-OPGM under the following assumptions:

(a) The functions $d_i$ are all convex and differentiable with the same Lipschitz constant $L > 0$.

(b) $D_{\sigma}$ is $\theta$-averaged with $\theta \in (0, 1)$ for any $\sigma > 0$.

(c) There exists $x^* \in \mathbb{R}^n$ such that $x^* \in \text{fix}(P)$. Additionally, there exists a constant $R > 0$ such that for any iteration $k$

$$\|x^k - x^*\|_2 < R.$$
(d) At every iteration, the gradient estimate is unbiased and has a bounded variance:

\[ \mathbb{E}[\hat{\nabla} g(x)] = \nabla g(x) \quad \text{and} \quad \mathbb{E}[\|\nabla g(x) - \hat{\nabla} g(x)\|_2^2] \leq \frac{\nu^2}{B}, \]

for some constant \( \nu > 0 \).

Note that Assumption 4.2(a) implies that the complete data-fidelity term \( d \) is also convex and has a Lipschitz continuous gradient of constant \( L \). Compared to Assumption 4.1(c), Assumption 4.2(c) additionally assumes the existence of the bound \( R \) for all iterates. This assumption is mild as many denoisers have bounded range spaces. In particular, this is true for a number of image denoisers whose outputs live within the bounded subset \([0, 255]^n \in \mathbb{R}^n\).

The key difference between Assumption 4.1 and Assumption 4.2 is the last condition. The fact that the minibatch gradient is unbiased is the direct consequence of (4.7). The bounded variance assumption is a standard assumption used in the analysis of online and stochastic algorithms [22, 78, 100].

**Proposition 4.5.** Run PnP-OPGM for \( t \geq 1 \) iterations under Assumption 4.2 with step-size \( \gamma \in (0, 1/L] \) and \( q_k = 1 \) for all \( k \in \{1, \ldots, t\} \). Then, for any \( x^* \in \text{fix}(P) \), we have that

\[ \mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \|x^{k-1} - P(x^{k-1})\|_2^2 \right] \leq 2 \left( \frac{1 + \theta}{1 - \theta} \right) \left[ \frac{\gamma^2 \nu^2}{B} + \frac{2\gamma \nu}{\sqrt{B}} R + \frac{R^2}{t} \right], \]

where \( P(\cdot) \) is given by (4.2).

**Proof.** See Appendix A.5. \( \square \)
This result shows that the convergence in expectation of PnP-OPGM to an element of \( \text{fix}(P) \) is proportional to the step-size \( \gamma \) and inversely proportional to the mini-batch size \( B \). By controlling these two parameters, we can obtain the following convergence rates.

**Corollary 4.1.** Consider Proposition 4.5 with the following fixed (i.e., independent of iteration \( k \)) parameters.

(a) For \( \gamma = 1/(L\sqrt{t}) \) and \( B = 1 \), we have that

\[
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \| x^{k-1} - P(x^{k-1}) \|_{2}^{2} \right] \leq \frac{A}{\sqrt{t}},
\]

(b) For \( \gamma = 1/L \) and \( B = t \), we have that

\[
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \| x^{k-1} - P(x^{k-1}) \|_{2}^{2} \right] \leq \frac{A}{\sqrt{t}},
\]

(c) For \( \gamma = 1/(L\sqrt{t}) \) and \( B = t \), we have that

\[
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \| x^{k-1} - P(x^{k-1}) \|_{2}^{2} \right] \leq \frac{A}{t},
\]

where

\[
A := 2 \left( \frac{1 + \theta}{1 - \theta} \right) \left( R + \frac{\nu}{L} \right)^{2}.
\]

Corollary 4.1(c) implies the worst-case convergence rate

\[
\mathbb{E} \left[ \min_{k \in \{1, \ldots, t\}} \{ \| x^{k-1} - P(x^{k-1}) \|_{2}^{2} \} \right] = O(1/t), \tag{4.8}
\]

which means that under Assumption 4.2 and with a particular selection of parameters \( B \) and \( \gamma \), the iterates of PnP-OPGM (in expectation) can get arbitrarily close to \( \text{fix}(P) \) as \( O(1/t) \).
4.5 Numerical Simulations

We now empirically validate PnP-OPGM in the context of diffraction tomography (DT) using three popular denoisers: TV [191], BM3D [54], and TNRD [46]. Our goal is not to justify the PnP framework, as its benefits have been well illustrated in prior work [103, 209, 236], but to focus on the aspects that relate to online processing of data. Therefore, we first discuss empirical convergence of PnP-OPGM, and then highlight the benefit of using it for processing a large number of measurements.

4.5.1 Diffraction tomography

DT is a technique used to form an image of the distribution of dielectric permittivity within an object from multiple measurements of light it scatters [98, 246]. This problem is common in a number of applications—including ultrasound [33] and optical microscopy [221]—and is known to be highly data-intensive. A typical reconstruction task uses hundreds or thousands
Figure 4.2: Illustration of the influence of the step-size $\gamma$ on the convergence of PnP-OPGM with BM3D as the denoiser. The distance to a fixed point is plotted against the iteration number for 3 distinct step-sizes for both accelerated (solid) and basic (dashed) variants of PnP-OPGM for $B = 30$. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. This plot illustrates that the empirical performance of PnP-OPGM under BM3D is consistent with Proposition 4.5, where the accuracy improves with smaller $\gamma$.

of measurements for forming a single image. As is common in DT, we adopt the first-Born approximation [246], which leads to the linear inverse problem formulation of image reconstruction.

Note that PnP-OPGM is applicable beyond DT and our choice of the latter is only due to the fact that image reconstruction in DT requires the processing of a large number of distinct measurements. Additionally, our focus is not on the experimental application of DT, but rather on the demonstration of our online algorithm for image reconstruction. Hence, we restrict our study here to image reconstruction from purely simulated DT data, which enables optimal parameter tuning and quantitative comparisons.

Consider an object with the permittivity distribution $\epsilon(\mathbf{r})$ within a bounded domain $\Omega \subseteq \mathbb{R}^2$ with a background medium of permittivity $\epsilon_b$. The object is illuminated with a monochromatic and coherent incident electric field $u_{\text{in}}(\mathbf{r})$ emitted by one of $N$ transmitters. The incident field is assumed to be known both inside $\Omega$ and at the sensor domain $\Gamma \subseteq \mathbb{R}^2$. The measurements correspond to the field scattered by the object recorded by $M$ receivers located within $\Gamma$. Under the first-Born approximation, the measurement matrix for a single illumination
Figure 4.3: Illustration of the influence of the minibatch size $B$ on the convergence of PnP-OPGM with BM3D as a denoiser. The distance to a fixed point is plotted against the iteration number for 3 distinct minibatch sizes for both accelerated (solid) and basic (dashed) variants of PnP-OPGM for $\gamma = 1/L$. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. This plot illustrates that the empirical performance of PnP-OPGM using BM3D is consistent with Proposition 4.5, where the accuracy improves with larger $B$.

can be represented as $A = S \text{diag}(u_m)$, where $u_m \in \mathbb{C}^n$ is the input field $u_m$ inside $\Omega$, and $S \in \mathbb{C}^{m \times n}$ is the discretization of the Green’s function evaluated at $\Gamma$ [132]. In practice, the image reconstruction relies on the set of illuminations $\{u_m^i\}_{i \in \{1, \ldots , I\}}$, with each individual illumination resulting in a measurement $y^i \in \mathbb{C}^m$ and a distinct measurement matrix $A_i$.

The objects we reconstruct correspond to the eight standard grayscale images shown Figure D.2. The physical size of an image is set to $18 \text{ cm} \times 18 \text{ cm}$, discretized to a grid of $256 \times 256$. The wavelength of the illumination was set to $\lambda = 0.84 \text{ cm}$ and the background medium was assumed to be air with $\epsilon_b = 1$. We additionally set the number of transmitters to $N = 60$, distributed uniformly along a circle of radius 1.6 meters, and for each illumination, the corresponding scattered field is measured by $m = 360$ receivers around the object. The simulated measurements were additionally corrupted by an additive white Gaussian noise (AWGN) corresponding to 40 dB of input signal-to-noise ratio (SNR). The quantitative evaluation of the experimental results is also provided in terms of SNR defined as

$$\text{SNR (dB)} := 10 \log_{10} \left( \frac{\|x\|^2_2}{\|\hat{x} - x\|^2_2} \right),$$

64
\[ \| x_k - P(x_k) \|_2 \]

Figure 4.4: Illustration of the influence of the step and minibatch sizes on the convergence of PnP-OPGM with TV as the denoiser. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. A proximal operator is \( (1/2) \) of PnP-OPGM with TV as the denoiser. The dotted line at the bottom shows the minimal distance to a fixed point attained by the algorithm. A proximal operator is \( (1/2) \)-averaged, which means that it perfectly satisfies the assumptions of Proposition 4.5.

<table>
<thead>
<tr>
<th>Denoiser</th>
<th>Step-size (( \gamma ))</th>
<th>Mini-batch size (( B ))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( 1/L )</td>
<td>( 1/4L )</td>
</tr>
<tr>
<td>TV</td>
<td>4.35e-4</td>
<td>2.86e-5</td>
</tr>
<tr>
<td>BM3D</td>
<td>4.08e-4</td>
<td>1.01e-4</td>
</tr>
<tr>
<td>TNRD</td>
<td>1.19e-1</td>
<td>2.20e-2</td>
</tr>
</tbody>
</table>

where \( \hat{x} \) and \( x \) are the reconstructed and the ground truth images, respectively. We use the term average SNR to indicate the SNR averaged over all the test images. In each experiment, all algorithmic hyperparameters were optimized for the best SNR performance with respect to the ground truth test image.

### 4.5.2 Convergence of PnP-OPGM

One of the key conclusions of Proposition 4.5 is that the final accuracy of PnP-OPGM to a fixed point is proportional to the step-size and inversely proportional to the minibatch size. In order to numerically evaluate the convergence, we define the distance to fix(\( P \)) at the \( k \)th iteration as

\[ \text{dist}(x^k) := \| x^k - P(x^k) \|_2^2, \]  

(4.9)
Figure 4.5: Comparison between the batch and online PnP algorithms for a fixed reconstruction time. SNR (dB) is plotted against the time in seconds for three algorithms: PnP-OPGM, PnP-APGM, and PnP-ADMM. Both PnP-APGM and PnP-ADMM use the full set of 60 illuminations at every iteration, while PnP-OPGM uses a random subset of 10 or 30 illuminations. This lower per-iteration cost, leads to a substantially faster convergence of PnP-OPGM.

where $P$ is given by (4.2). As the sequence $\{x^k\}$ approaches $\text{fix}(P)$, $\text{dist}(x^k)$ approaches zero.

Figure 4.2 and Figure 4.3 empirically evaluate the evolution of the distance to a fixed point for different step and minibatch sizes, respectively. PnP-OPGM, with BM3D as a denoiser, is run until convergence with $\gamma \in \{1/L, 1/(4L), 1/(16L)\}$ and $B \in \{10, 20, 30\}$. Here, the quantity $L > 0$ denotes the Lipschitz constant, which, for linear inverse problems, corresponds to the squared largest singular value of the measurement matrix [18]. We show the performance of both basic and accelerated variants of PnP-OPGM, where the latter is obtained by setting $\{q_k\}$ as in (2.17). The plots clearly illustrate the improvement in final accuracy for smaller $\gamma$ and larger $B$, which is consistent with Proposition 4.5. Additionally, they indicate that the convergence is significantly improved when using the accelerated variant of the algorithm. Note that our theoretical analysis does not predict monotonic reduction of the distance, which also seems to be consistent with the empirical performance of PnP-OPGM. In Figure 4.4, we provide a reference plot showing the performance of PnP-OPGM under TV, which is a valid
proximal operator and hence is known to be a $1/2$-averaged operator. We can again observe that the convergence behavior of PnP-OPGM is consistent with Proposition 4.5. Finally, the summary in Table 4.1, highlights the same convergence trends for all three algorithms, where both $\gamma$ and $B$ control the accuracy of PnP-OPGM.

### 4.5.3 Benefits of online processing

We now highlight the higher efficiency of PnP-OPGM against PnP-PGM and PnP-ADMM for larger number of measurements. Specifically, we consider two scenarios where: (a) the total time budget is fixed; (b) the number of measurements is fixed. While we use BM3D as our plug-in operator of choice, we note that our observations here directly generalize to any other denoiser.

Figure 4.5 compares the average reconstruction SNR of PnP-OPGM, PnP-APGM, and PnP-ADMM for a fixed run-time. The batch algorithms use the full 60 illuminations at every iteration, while PnP-OPGM uses 10 and 30 illuminations per iteration. This gives PnP-OPGM a significantly lower per iteration cost compared to the batch algorithms. Specifically,
Figure 4.7: Visual illustration of the reconstructed Monarch and Parrot images obtained using PnP-OPGM, PnP-APGM, and PnP-ADMM, all under BM3D. The original images are displayed in the first column. The second and the third columns show the results of PnP-APGM and PnP-ADMM with the budget of 30 illuminations, and the fourth and the fifth columns present the results of the PnP-OPGM with the budget of 10 and 30 illuminations. Visual differences are highlighted using the rectangles drawn inside the images. Each reconstruction is labeled with its SNR (dB) value with respect to the original image.

the average per iteration time for PnP-OPGM using $B = 10$, PnP-OPGM using $B = 30$, PnP-APGM, and PnP-ADMM was 8.86 seconds, 22.10 seconds, 44.94 seconds, and 382.83 seconds, respectively. The higher cost of PnP-ADMM is the result of the forward model inversion in 2.23. The figure illustrates that, in practice, the solution of PnP-OPGM is close to that of the batch algorithm with the final SNRs for $B = 10$ and $B = 30$ being within 0.2 dB and 0.01 dB, respectively, from that of PnP-APGM. Additionally, PnP-OPGM achieves a significant speedup due to the reduction in per-iteration complexity. This indicates to the potential of the algorithm for efficient image reconstruction from a large number of measurements.
Figure 4.6 compares the average reconstruction SNR of PnP-OPGM, PnP-APGM, and PnP-ADMM for a fixed per-iteration measurement budget. The batch algorithms are allowed to use only 10 (top figure), 20 (middle figure), or 30 (bottom figure) uniformly distributed illuminations. Convergence of each algorithm was observed in 1000 iterations, and the figure displays the average SNR within the first 250 iterations since PnP-OPGM has already converged in all three plots. Similarly, PnP-OPGM uses the same number of illuminations per iteration, but randomly cycles through all the measurements. This means that in each figure both PnP-OPGM and PnP-APGM have the same per-iteration computational complexity. The computational complexity of PnP-ADMM is higher due to the need to invert the measurement matrix. Table B.3 shows the final SNR obtained by all three algorithms on each individual image in the dataset. Additionally, two visual illustrations on Monarch and Parrot are shown in Figure 4.7. The two rectangles under each image show areas rich in texture that were selected to highlight the visual differences in the results. As expected, PnP-OPGM achieves dramatically higher SNR compared to batch algorithms, since it makes use of the full set of measurements. Additionally, we note the comparable final SNR performance of PnP-OPGM with $B = 10$ and $B = 30$, with the latter leading to a faster convergence speed. These results again highlight the potential of PnP-OPGM for large-scale PnP image reconstruction.

The simulations in this section highlight the benefit of PnP-OPGM in tomographic imaging, where each measurement contains information from a large portion of the object. PnP-OPGM leverages this setting to improve the computational and memory efficiency of processing a large number of measurements. Whether this benefit of using PnP-OPGM would persist in other imaging problems—such as inpainting or deblurring, where the information on the unknown is heavily localized in the measurements—is still an open question and a potential avenue of future research.
Table 4.2: Individual reconstruction SNRs for each image.

<table>
<thead>
<tr>
<th>Images</th>
<th>PnP-ADMM (10)</th>
<th>PnP-ADMM (20)</th>
<th>PnP-APGM (10)</th>
<th>PnP-APGM (20)</th>
<th>PnP-OPGM (10)</th>
<th>PnP-OPGM (20)</th>
<th>PnP-OPGM (30)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barbara</td>
<td>15.62</td>
<td>19.16</td>
<td>21.18</td>
<td>13.32</td>
<td>16.60</td>
<td>20.21</td>
<td>23.61</td>
</tr>
<tr>
<td>Boat</td>
<td>15.94</td>
<td>19.82</td>
<td>23.10</td>
<td>13.69</td>
<td>18.40</td>
<td>22.01</td>
<td>24.87</td>
</tr>
<tr>
<td>Foreman</td>
<td>23.10</td>
<td>27.27</td>
<td>29.19</td>
<td>18.46</td>
<td>26.64</td>
<td>28.61</td>
<td>29.61</td>
</tr>
<tr>
<td>House</td>
<td>19.23</td>
<td>23.46</td>
<td>26.43</td>
<td>15.68</td>
<td>25.36</td>
<td>26.79</td>
<td>28.29</td>
</tr>
<tr>
<td>Lenna</td>
<td>15.52</td>
<td>20.32</td>
<td>23.17</td>
<td>13.49</td>
<td>20.57</td>
<td>22.91</td>
<td>25.30</td>
</tr>
<tr>
<td>Monarch</td>
<td>11.46</td>
<td>15.82</td>
<td>19.66</td>
<td>8.80</td>
<td>17.38</td>
<td>20.69</td>
<td>23.51</td>
</tr>
<tr>
<td>Parrot</td>
<td>17.29</td>
<td>21.49</td>
<td>24.05</td>
<td>13.73</td>
<td>22.29</td>
<td>24.32</td>
<td>26.38</td>
</tr>
<tr>
<td>Pepper</td>
<td>15.49</td>
<td>20.46</td>
<td>22.90</td>
<td>11.67</td>
<td>20.89</td>
<td>22.96</td>
<td>24.92</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>16.71</strong></td>
<td><strong>20.98</strong></td>
<td><strong>23.71</strong></td>
<td><strong>14.26</strong></td>
<td><strong>21.02</strong></td>
<td><strong>23.73</strong></td>
<td><strong>25.85</strong></td>
</tr>
</tbody>
</table>

To conclude this section, let us put the results here in the context of our theoretical analysis. Proposition 4.5 reveals that PnP-OPGM converges to the same set of fixed points $\text{fix}(P)$ as PnP-PGM and PnP-ADMM, up to a term that depends on the minibatch size $B \geq 1$. Larger $B$ leads to a higher accuracy of PnP-OPGM with respect to $\text{fix}(P)$, which was empirically confirmed in Figure 4.3. The SNR results here additionally reveal that even with a relatively small $B$, PnP-OPGM is accurate in terms of image quality. For example, in Table B.3, we can observe that the average SNR difference between PnP-OPGM with $B = 10$ and $B = 30$ is within 0.2 dB of each other. Additionally, in Figure 4.5, we observe that the batch and online algorithms approximately achieve the same final SNR performance. These observations suggest that while there is an order of magnitude difference in accuracy between $B = 10$ and $B = 30$ when measured in terms of the distance to a fixed point (see Figure 4.3), the difference is relatively mild when measured in terms of image quality (see Figure 4.7), with smaller $B$ nearly matching the image quality of the batch algorithm.
4.6 Summary

The online PnP algorithm developed in this chapter is beneficial in the context of large-scale image reconstruction, when the amount of data is too large to be processed jointly. We have presented an in-depth theoretical convergence analysis for both batch and online variants of PnP-PGM. Our work represents a substantial extension of the current convergence theory of PnP-algorithms for image reconstruction. Related experiments are also presented to empirically confirm the proposed propositions and to elucidate the higher efficiency of PnP-OPGM in different representative situations. Future work will aim to apply the algorithm to other image reconstruction tasks, relax some of the assumptions, and extend the theoretical results in this chapter to accommodate acceleration techniques.
Chapter 5

Regularized Fourier Ptychography using Online Plug-and-Play PGM

Fourier Ptychographic Microscopy (FPM) is a popular optical bio-imaging technology that can generate high-resolution image [270]. However, the image reconstruction in FPM is challenging due to two reasons: (1) both the amount and size of FPM measurements are large, resulting in the requirement of significant computation and memory; (2) FPM only measures the intensity of the light scattered by the sample, completely losing the phase information. This makes the inverse problem associated with FPM a phase retrieval problem that is known to be severely ill-posed (see Section 2.1.1). In this chapter, we leverage PnP-OPGM to simultaneously address these two challenges. Results on both simulated and experimentally-collected data demonstrate the faster processing speed as well as higher image quality than the existing methods.
5.1 FPM as an inverse problem

Consider an unknown object with a complex transmission function \( o(r) \), where \( r \) denotes the spatial coordinates at the object plane. A total of \( N \) LED sources are used to illuminate the object. Each illumination is treated as a local plane wave with a unique spatial frequency \( k_i \), \( i \in \{1, ..., N\} \). The exit wave from the object is described by the product: \( u(r) = o(r)e^{i(k_i \cdot r)} \), which indicates that the center of the sample’s spectrum is shifted to \( k_i \) [Zheng:2013aa, 229, 266]. At the pupil plane, the shifted Fourier transformation of the exit wave is further filtered by the pupil function \( p(k) \). For a single illumination, the discrete FPM model can be mathematically described by the following inverse problem

\[
y = |As(x)|^2 + e, \quad \text{with} \quad A := F_c^{-1}\text{diag}(P)SF_o
\]

where \( s(x) = e^{ix} \) denotes the discretized transmittance, with \( x \in \mathbb{R}^n \) being the vectorized representation of the desired object properties, \( y \in \mathbb{R}^m \) represents the corresponding low-resolution light-intensity measurements, and \( e \) is the noise vector. The operator \(|\cdot|\) computes the element-wise absolute value. The complex matrix \( A \in \mathbb{C}^{m \times n} \) is implemented by taking the Fourier transform \( (F_o) \) of the object, shifting and truncating the low frequency region \( (S) \), multiplying it by a pupil function in the frequency domain \( (P) \), and taking the inverse Fourier transform \( (F_c^{-1}) \) with respect to the spectrum.

The inverse problem associated with FPM can be formulated as

\[
\arg \min_{x \in \mathbb{C}^n} f(x) \quad \text{where} \quad f(x) = \left( |As(x)|^2 - y \right)^2 + r(x)
\]

(5.2)
where we recall that $g$ measures the data-fidelity, and $r$ denotes the regularizer. Note that the final objective in FPM is nonconvex due to the absolute value operator in the data-fidelity term (5.2). Both PGM-based and ADMM-based PnP algorithms can be used for solving (5.2), but they differ from each other in the use of gradient or proximal operator to impose data-fidelity. In the context of FPM, we have

$$\nabla d(x) = |\nabla \Psi(x)|^\dagger \text{diag} \Psi(x)(\Psi(x) - y)$$

(5.3)

with $\Psi(x) := As(x)$, $\nabla \Psi(x) = \text{diag} \exp(ix) A^\dagger$, (5.4)

where $\dagger$ denotes the conjugate transpose, and

$$\text{prox}_{\gamma d}(x) = \arg\min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - x \|_2^2 + \frac{\gamma}{2} \| A s(z) \|^2 - y \|_2^2 \right\}.$$  

(5.5)

Because the data-fit in FPM is not convex, the optimization in (5.5) is difficult to solve. Additionally, the calculation of (5.5) is computationally expensive when the number of measurements is large. These two factors makes PnP-OPGM more applicable to FPM.

## 5.2 Numerical Validation

In this section, we validate our PnP-OPGM on simulated and experimental data by considering two representative denoisers: TV [191] and BM3D [55]. Note that our focus is to demonstrate the effectiveness of the proposed PnP method for FPM rather than to test different denoisers, although the algorithm is readily compatible with other state-of-the-art denoising methods.
Figure 5.1: Visual comparison of the reconstructed images of *Cameraman* obtained by PnP-OPGM and PnP-APGM. Minibatch $B = 60$ was used in the simulation. The first column (Original) shows the original image. The second column (LS) presents the result of least-square fitting using no regularizer. The third (Online-LS), fourth (OPGM (TV)) and fifth (PnP-OPGM (BM3D)) columns present the results of OPGM using no regularizer, using TV and using BM3D, respectively. The last column (PnP-APGM (BM3D)) shows the result of PnP-APGM using BM3D and all 293 measurements. Each image is labeled with its SNR value with respect to the original image.

### 5.2.1 Experimental Setup

We set up the simulation to match the FPM system used for the experimental data [229]. The sample is placed 70 mm below a $32 \times 32$ surface-mounted LED array with a 4 mm pitch. All LED sources generate light with a wavelength 513 mm and the bandwidth 20 nm. We individually illuminate the sample with 293 LEDs centered in the array and record the corresponding intensity measurements with a camera placed under the sample. The numerical aperture (NA) of the objective is 0.2 [229]. We achieve the synthetic NA of around 0.7 by summing the NA of the objective and illumination.

### 5.2.2 Benefits of PnP-OPGM

We first quantitatively analyze the performance of PnP-OPGM by reconstructing six common gray-scale images discretized to $500 \times 500$ pixels: *Cameraman, House, Jet, Lenna, Pepper* and *Woman*. The simulated measurements were obtained by solving the forward model
Figure 5.2: Evolution of average SNR across iterations for batch and online PnP algorithms using different priors. The corresponding labels are shown at the bottom-right corner inside the plot. The purple dotted line, PnP-PGM (BM3D), indicates the performance when using all 293 measurements. Note that PnP-OPGM (BM3D) achieves the SNR performance of PnP-PGM (BM3D) at a lower computational cost.

defined in (5.1). Additionally, the measurements were corrupted by an additive Gaussian noise (AWGN) corresponding to 40 dB of input signal-to-noise ratio (SNR). The quantitative evaluation of reconstruction results is also based on SNR defined as

$$\text{SNR (dB)} := 10\log_{10}(\frac{\|x\|^2}{\|x - \hat{x}\|^2}), \quad (5.6)$$

where the $\hat{x}$ and $x$ are the reconstructed and ground truth images, respectively. We use average SNR to indicate the SNR averaged over all the test images. In simulations, all algorithmic hyperparameters were optimized for the best SNR performance with respect to the original image.

Figure 5.2 illustrates the evolution of average SNR across iterations for different priors and PnP variants. With the minibatch $B = 60$, the online algorithms randomly select different subset of measurements at each iteration, whereas the batch algorithms use the same fixed 60 measurements in the reconstruction. Hence, as expected, by eventually cycling through all the measurements, Online-LS achieves a higher average SNR than LS, which only uses the fixed 60 illuminations. Additionally, the performance is further enhanced by using priors.
Figure 5.3: Comparison between PnP-OPGM (BM3D) and PnP-PGM (BM3D) for a fixed reconstruction time. The average SNR is plotted against the time in seconds for both algorithms. PnP-OPGM (BM3D) uses only 60 measurements per iteration, while PnP-OPGM (BM3D) uses all 293 measurements. The lower per iteration cost leads to a substantially faster convergence of PnP-OPGM (BM3D).

For example, OPGM (TV) and PnP-OPGM (BM3D) increase the average SNR from 18.48 dB to 19.42 dB and 19.74 dB, respectively. A visual illustration on Cameraman is presented in Figure B.3, where PnP-PGM (BM3D) is shown as reference. We observe that, even with $B = 60$, the solution of PnP-OPGM (BM3D) is only 0.25 dB lower than PnP-PGM (BM3D), which denotes the batch PnP algorithm using all 293 measurements.

Figure 5.3 compares the average SNR performance of online and batch PnP algorithms within a fixed run-time. In the test, PnP-OPGM (BM3D) uses only 60 measurements per iteration, while PnP-PGM (BM3D) uses all 293 measurements. The lower per iteration cost makes the reconstruction of PnP-OPGM (BM3D) substantially faster than that of its batch rival. In particular, the averaged single-iteration run-time of PnP-OPGM (BM3D) and PnP-PGM (BM3D) was 9.07 seconds and 19.66 seconds, respectively. We also note that PnP-OPGM (BM3D) and PnP-PGM (BM3D) eventually converge to the same average SNR, which agrees with the plots in Figure 5.2. Additionally, PnP-OPGM achieves a substantial speedup due to its reduction in per-iteration computational complexity, which makes the algorithm applicable to very large datasets.
Figure 5.4: Comparison of online and batch algorithms on the FPM dataset containing HeLa cells. Each algorithm uses the budget of 60 measurements per iteration. The first row illustrates the results of LS, PGM (TV), and PnP-PGM (BM3D). The second row shows Online-LS, OPGM (TV), and PnP-OPGM (BM3D). Visual difference are illustrated by the white rectangles drawn inside the images. The green arrows highlight the artifacts.

5.2.3 Validation on real data

We now validate the performance of PnP-OPGM on experimental FPM data. The sample used in the experiment consists of the human cervical adenocarcinoma epithelial (HeLa) cells [229]. The system corresponds to the FPM described in Section 5.2.1 with total 293 measurements for reconstruction.

Figure 5.4 compares the images of HeLa cells reconstructed by PnP-OPGM and PnP-APGM. Each image has the resolution of $900 \times 900$ pixels. The green arrows in the white rectangles highlight the artifacts. We consider the scenario with a limited memory budget sufficient...
only for 60 measurements. Since batch algorithms use a fixed subset of measurements, they produce strong unnatural features, such as the horizontal streaking artifacts in LS. Both TV-Batch and BM3D-Batch mitigate this artifact by using priors, but they generate blockiness and vertical grids in the cell, respectively. Online algorithms generally improve the visual quality by making use of all the data. Finally, PnP-OPGM (BM3D) alleviates the artifacts and reconstructs a high-quality image, while Online-LS and OPGM (TV) still have streaking artifacts and blockiness.

5.3 Summary

In this chapter, we propose a novel online plug-and-play algorithm for the regularized Fourier ptychographic microscopy. Numerical simulations demonstrate that PnP-OPGM converges to a nearly-optimal average SNR in a shorter amount of time. The experiments for FPM confirm the effectiveness and efficiency of PnP-OPGM in practice, and shows its potential for other imaging applications. More generally, this work shows the potential of PnP-OPGM to solve inverse problems beyond traditional convex data-fidelity terms.
Chapter 6

Incremental Plug-and-Play

Alternating Direction Method of Multipliers

This chapter investigates another scalable PnP algorithm that is based on alternating direction method of multipliers (ADMM). The new algorithm is different from PnP-OPGM in terms of its treatment of the data-fidelity term. We first provide an intuitive interpretation of the solution obtained by PnP-ADMM and then theoretically analyze the convergence of the algorithm under a set of explicit assumptions. Note that our analysis is also backward compatible with the original PnP-ADMM formulation. Additionally, we show the effectiveness of our algorithm with nonsmooth data-fidelity terms and deep neural networks priors, its fast convergence compared to existing PnP algorithms, and its scalability in terms of speed and memory.


6.1 Introduction

From the discussion in Section 2.2.1, we know that PnP-ADMM imposes the data-fidelity through the manifestation of a proximal operator $\text{prox}_{\gamma g}$ rather than gradient $\nabla g$, which rules out the application of online gradient used in PnP-OPGM. Instead, we leverage a concept known as *proximal average* to approximate the full-batch proximal operator by averaging the proximal operators evaluated on minibatches of measurements. We dub the new algorithm as *incremental PnP-ADMM (IPA)*. As an extensions of the widely used PnP-ADMM, IPA can integrate statistical information from a data-fidelity term and a pre-trained DNN. However, unlike PnP-ADMM, IPA can effectively scale to datasets that are too large for traditional batch processing by using a single element or a small subset of the dataset at a time. The memory and per-iteration complexity of IPA is independent of the number of measurements, thus allowing it to deal with very large datasets. Additionally, unlike PnP-PGM [214], IPA can effectively address problems with *nonsmooth* data-fidelity terms, and generally has faster convergence. We present a detailed convergence analysis of IPA under a set of explicit assumptions on the data-fidelity term and the denoiser. Our analysis extends the recent fixed-point analysis of PnP-ADMM in [193] to partial randomized processing of data. To the best of our knowledge, the proposed scalable PnP algorithm and corresponding convergence analysis are absent from the current literature in this area. Our numerical validation demonstrates the practical effectiveness of IPA for integrating nonsmooth data-fidelity terms and DNN priors, its fast convergence compared to PnP-OPGM, and its scalability in terms of both speed and memory. In summary, we establish IPA as a flexible, scalable, and theoretically sound PnP algorithm applicable to a wide variety of large-scale problems.
6.2 Related Work

We refer to Section 2.2.1 and 3.1 for the introduction of ADMM and PnP-ADMM, respectively. The empirical success of PnP-ADMM has spurred a great deal of theoretical work [42, 193, 209]. A elegant fixed-point convergence analysis based on monotone operator theory was presented in [193]. By substituting $v^k = z^k - s^{k-1}$ into PnP-ADMM, the algorithm is expressed in terms of an operator

$$P := \frac{1}{2}I + \frac{1}{2}(2G - I)(2D_\sigma - I) \quad \text{with} \quad G := \text{prox}_{\gamma g},$$

(6.1)

where $I$ denotes the identity operator. The convergence of PnP-ADMM is then established through its equivalence to the fixed-point convergence of the sequence $v^k = P(v^{k-1})$. The equivalence of PnP-ADMM to the iterations of the operator (6.1) originates from the well-known relationship between ADMM and the Douglas-Rachford splitting [37, 66, 168, 193]. Nevertheless, we note that the analysis in [193] assumes strongly convex data-fidelity term that is usually violated in real applications.

Scalable optimization algorithms have become increasingly important in the context of large-scale problems arising in machine learning and data science [29]. Stochastic and online optimization techniques have been investigated for traditional ADMM [87, 167, 222, 239, 271], where $\text{prox}_{\gamma g}$ is approximated using a subset of observations (with or without subsequent linearization). Our work contributes to this area by investigating the scalability of PnP-ADMM that is not minimizing any explicit objective function. Since PnP-ADMM can integrate powerful DNN denoisers, there is a need to understand its theoretical properties and ability to process a large number of measurements.
Algorithm 8 Incremental Plug-and-Play ADMM (IPA)

1: **input:** initial values $\mathbf{x}^0, s^0 \in \mathbb{R}^n$, parameters $\gamma, \sigma > 0$.
2: **for** $k = 1, 2, 3, \ldots$ **do**
3: Choose an index $i_k \in \{1, \ldots, b\}$
4: $z^k \leftarrow G_{i_k}(\mathbf{x}^{k-1} + s^{k-1})$ where $G_{i_k} := \text{prox}_{\gamma g_{i_k}}$
5: $\mathbf{x}^k \leftarrow D_\sigma(z^k - s^{k-1})$
6: $s^k \leftarrow s^{k-1} + \mathbf{x}^k - z^k$
7: **end for**

6.3 Incremental PnP-ADMM

Batch PnP algorithms operate on the whole observation vector $\mathbf{y} \in \mathbb{R}^m$. We are interested in partial randomized processing of observations by considering the decomposition of $\mathbb{R}^m$ into $b \geq 1$ blocks

$$\mathbb{R}^m = \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \times \cdots \times \mathbb{R}^{m_b} \quad \text{with} \quad m = m_1 + m_2 + \cdots + m_b.$$  

We thus consider data-fidelity terms of the form

$$g(\mathbf{x}) = \frac{1}{b} \sum_{i=1}^{b} g_i(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n, \quad (6.2)$$

where each $g_i$ is evaluated only on the subset $\mathbf{y}_i \in \mathbb{R}^{m_i}$ of the full data $\mathbf{y}$.

The proposed IPA algorithm seeks to avoid the direct computation of $\text{prox}_{\gamma g}$ in PnP-ADMM. As shown in Algorithm 8, it extends stochastic variants of traditional ADMM [87, 167, 222, 239, 271] by integrating denoisers $D_\sigma$ that are *not* associated with any $h$. Its per-iteration complexity is independent of the number of data blocks $b$, since it processes only a single component function $g_i$ at every iteration.
It is important to note that in some applications [7, 180, 245], the $\text{prox}_{g}$ step of PnP-ADMM can be efficiently evaluated by leveraging the structure of the measurement operator (such as diagonalization by Fourier transform). Nonetheless, IPA provides flexibility for controlling the number of measurements $1 \leq m_i \leq m$ used in every iteration, which makes it a useful alternative to PnP-ADMM, when the memory/computational benefits for evaluating $\text{prox}_{g_i}$ (which uses only $y_i \in \mathbb{R}^{m_i}$ and $A_i \in \mathbb{R}^{m_i \times n}$) outweigh those of $\text{prox}_{g}$ (which uses $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$).

In principle, IPA can be implemented using different block selection rules. The strategy adopted for our theoretical analysis focuses on the usual strategy of selecting indices $i_k$ as independent and identically distributed (i.i.d.) random variables distributed uniformly over $\{1, \ldots, b\}$. An alternative would be to proceed in epochs of $b$ consecutive iterations, where at the start of each epoch the set $\{1, \ldots, b\}$ is reshuffled, and $i_k$ is selected from this ordered set [23]. In some applications, it might also be beneficial to select indices $i_k$ in an online data-adaptive fashion by taking into account the statistical relationships among observations [111, 230].

Unlike PnP-OPGM, IPA does not require smoothness of the functions $g_i$. Instead of computing the partial gradient $\nabla g_i$, as is done in PnP-OPGM, IPA evaluates the partial proximal operator $G_i$. Nonsmooth data-fidelity terms have been extensively used in many applications, including wavelet inpainting, tensor factorization, feature selection, dictionary learning, and phase unwrapping [10, 43, 85, 91, 105, 137, 162]. The maximal benefit of IPA over PnP-OPGM is expected for problems in which $G_i$ is efficient to evaluate. This is a case for a number of functions commonly used in many applications (see the extensive discussion on proximal operators in [17]). For example, the proximal operator of the $\ell_2$-norm data-fidelity term
\[ g_i(x) = \frac{1}{2} \| y_i - A_i x \|_2^2 \] has a closed-form solution

\[ G_i(z) = \text{prox}_{\gamma g_i}(z) = (I + \gamma A_i^T A_i)^{-1} \left( z + \gamma A_i^T y_i \right) \]  

for \( \gamma > 0 \) and \( z \in \mathbb{R}^n \). Prior work has extensively discussed efficient strategies for evaluating (6.3) for a variety of linear operators, including convolutions, partial Fourier transforms, and subsampling masks [4, 7, 180, 245]. As a second example, consider the \( \ell_1 \)-data fidelity term \( g_i(x) = \| y_i - A_i x \|_1 \), which is nonsmooth. The corresponding proximal operator has a closed form solution for any orthogonal operator \( A_i \) and can also be efficiently computed in many other settings [17].

IPA can also be implemented as a \textit{minibatch} algorithm, processing several blocks in parallel at every iteration, thus improving its efficiency on multi-processor hardware architectures. Algorithm 9 presents the minibatch version of IPA that averages several proximal operators evaluated over different data blocks. When the minibatch size \( p = 1 \), Algorithm 9 reverts to Algorithm 8. The main benefit of minibatch IPA is its suitability for parallel computation of \( \hat{G} \), which can take advantage of multi-processor architectures.

Minibatch IPA is related to the \textit{proximal average} approximation of \( G = \text{prox}_{\gamma g} \) [15, 258]

\[ \bar{G}(x) = \frac{1}{b} \sum_{i=1}^{b} \text{prox}_{\gamma g_i}(x) \quad x \in \mathbb{R}^n . \]

When Assumption 6.1, introduced in Section 6.4, is satisfied, then the approximation error is bounded for any \( x \in \mathbb{R}^n \) as

\[ \| G(x) - \bar{G}(x) \| \leq 2\gamma L . \]

Minibatch IPA thus simply uses a minibatch approximation \( \hat{G} \) of the proximal average \( \bar{G} \). One implication of this is that even when the minibatch is \textit{exactly} equal to the full measurement
Algorithm 9 Minibatch IPA

1: **input:** initial values $x^0, s^0 \in \mathbb{R}^n$, parameters $\gamma, \sigma > 0$, minibatch size $p \geq 1$.
2: **for** $k = 1, 2, 3, \ldots$ **do**
3:    Choose indices $i_1, \ldots, i_p$ from the set $\{1, \ldots, b\}$.
4:    $z^k \leftarrow \hat{G}(x^{k-1} + s^{k-1})$ where $\hat{G} := \frac{1}{p} \sum_{j=1}^{p} \text{prox}_{\gamma g_{i_j}}$
5:    $x^k \leftarrow D_\sigma(z^k - s^{k-1})$
6:    $s^k \leftarrow s^{k-1} + x^k - z^k$
7: **end for**

vector, minibatch IPA is not exact due to the approximation error introduced by the proximal average. However, the resulting approximation error can be made as small as desired by controlling the penalty parameter $\gamma > 0$.

6.4 Theoretical Analysis

We now present a theoretical analysis of IPA. We first present an intuitive interpretation of its solutions, and then present our convergence analysis under a set of explicit assumptions.

6.4.1 Fixed Point Interpretation

PnP cannot be interpreted using the standard tools from convex optimization, since its solution is generally not a minimizer of an objective function. Nonetheless, we develop an intuitive operator based interpretation.

Consider the following set-valued operator

$$T := \gamma \partial g + (D_\sigma^{-1} - I) \quad \gamma > 0$$

where $\partial g$ is the subdifferential of the data-fidelity term and $D_\sigma^{-1}(x) := \{z \in \mathbb{R}^n : x = D_\sigma(z)\}$ is the inverse operator of the denoiser $D_\sigma$. The details for obtaining (6.4) from PnP-ADMM

86
(see Algorithm 6) are provided in Appendix B.3. Note that this inverse operator exists even when $D_\sigma$ is not one-to-one [66, 192]. By characterizing the fixed points of PnP algorithms, it can be shown that their solutions can be interpreted as vectors in the zero set of $T$

$$\mathbf{0} \in T(x^*) = \gamma \partial g(x^*) + (D^{-1}_\sigma(x^*) - x^*)$$

$$\Leftrightarrow \space x^* \in \text{zer}(T) := \{x \in \mathbb{R}^n : \mathbf{0} \in T(x)\} .$$

Consider the following two sets

$$\text{zer}(\partial g) := \{x \in \mathbb{R}^n : \mathbf{0} \in \partial g(x)\} \quad \text{and}$$

$$\text{fix}(D_\sigma) := \{x \in \mathbb{R}^n : x = D_\sigma(x)\} ,$$

where $\text{zer}(\partial g)$ is the set of all critical points of the data-fidelity term and $\text{fix}(D_\sigma)$ is the set of all fixed points of the denoiser. Intuitively, the fixed points of $D_\sigma$ correspond to all vectors that are not denoised, and therefore can be interpreted as vectors that are noise-free according to the denoiser.

If $x^* \in \text{zer}(\partial g) \cap \text{fix}(D_\sigma)$, then $x^* \in \text{zer}(T)$, which implies that $x^*$ is one of the solutions. Hence, any vector that minimizes a convex data-fidelity term $g$ and noiseless according to $D_\sigma$ is in the solution set. On the other hand, when $\text{zer}(\partial g) \cap \text{fix}(D_\sigma) = \emptyset$, then $x^* \in \text{zer}(T)$ corresponds to an equilibrium point between two sets.

This interpretation of PnP highlights one important aspect that is often overlooked in the literature, namely that, unlike in the traditional formulation (3.1), the regularization in PnP depends on both the denoiser parameter $\sigma > 0$ and the penalty parameter $\gamma > 0$, with both influencing the solution. Hence, the best performance is obtained by jointly tuning both parameters for a given experimental setting. In the special case of $D_\sigma = \text{prox}_{\gamma h}$ with $\gamma = \sigma^2$, 

87
we have

\[
\text{fix}(D_\sigma) = \{ x \in \mathbb{R}^n : 0 \in \partial h(x) \} \quad \text{and}
\]

\[
\text{zer}(T) := \{ x \in \mathbb{R}^n : 0 \in \partial g(x) + \partial h(x) \},
\]

which corresponds to the optimization formulation (3.1) whose solutions are independent of \( \gamma \).

### 6.4.2 Convergence Analysis

Our analysis requires three assumptions that jointly serve as sufficient conditions.

**Assumption 6.1.** Each \( g_i \) is proper, closed, convex, and Lipschitz continuous with constant \( L_i > 0 \). We define the largest Lipschitz constant as \( L = \max\{L_1, \ldots, L_b\} \).

This assumption is commonly adopted in nonsmooth optimization and is equivalent to existence of a global upper bound on subgradients [31, 167, 258]. It is satisfied by a large number of functions, such as the \( \ell_1 \)-norm. The \( \ell_2 \)-norm also satisfies Assumption 6.1 when it is evaluated over a bounded subset of \( \mathbb{R}^n \). We next state our assumption on \( D_\sigma \).

**Assumption 6.2.** The residual \( R_\sigma := I - D_\sigma \) of the denoiser \( D_\sigma \) is firmly nonexpansive.

We review firm nonexpansiveness and other related concepts in the Appendix B.3. Firmly nonexpansive operators are a subset of nonexpansive operators (those that are Lipschitz continuous with constant one). A simple strategy to obtain a firmly nonexpansive operator is to create a \((1/2)\)-averaged operator from a nonexpansive operator [168]. The residual \( R_\sigma \) is firmly nonexpansive if and only if \( D_\sigma \) is firmly nonexpansive. It is worth noting that (a) any explicit or implicit proximal operator is firmly nonexpansive, and (b) any symmetric matrix
with eigenvalues in $[0, 1]$ is firmly nonexpansive. This implies that many recently designed
denoisers for PnP, such as those discussed in [76, 155, 209, 225, 226] automatically satisfy
Assumption 6.2.

The rationale for stating Assumption 6.2 for $\mathbb{R}_\sigma$ is based on our interest in residual DNNs.
The success of residual learning in the context of image restoration is well known [262]. Prior
work has also shown that Lipschitz constrained residual networks yield excellent performance
without sacrificing stable convergence [193, 219]. Additionally, there has recently been an
explosion of techniques for training Lipschitz constrained and firmly nonexpansive DNNs [70,
153, 193, 227].

**Assumption 6.3.** The operator $T$ in (6.4) is such that $\text{zer}(T) \neq \emptyset$. There also exists $R < \infty$
such that
\[
\|x^k - x^*\|_2 \leq R \quad \text{for all } x^* \in \text{zer}(T).
\]

The first part of the assumptions simply ensures the existence of a solution. The existence
of the bound $R$ often holds in practice, as many denoisers have bounded range spaces. In
particular, this is true for a number of image denoisers whose outputs live within the bounded
subset $[0, 255]^n \subset \mathbb{R}^n$.

We will state our convergence results in terms of the operator $S : \mathbb{R}^n \to \mathbb{R}^n$ defined as
\[
S := D_\sigma - G(2D_\sigma - I).
\] (6.5)

Both IPA and PnP-ADMM can be interpreted as algorithms for computing an element in
$\text{zer}(S)$, which is equivalent to finding an element of $\text{zer}(T)$ (see details in Appendix B.3).

We are now ready to state our main result on IPA.
Theorem 6.1. Run IPA for \( t \geq 1 \) iterations with random i.i.d. block selection under Assumptions 6.1-6.3 using a penalty parameter \( \gamma > 0 \). Then, the sequence \( v^k = z^k - s^{k-1} \) satisfies

\[
E \left[ \frac{1}{t} \sum_{k=1}^{t} \|S(v^k)\|_2^2 \right] \leq \frac{(R + 2\gamma L)^2}{t} + \max\{\gamma, \gamma^2\} C ,
\]  

(6.6)

where \( C := 4LR + 12L^2 \) is a positive constant.

In order to contextualize this result, we also review the convergence of the traditional PnP-ADMM.

Theorem 6.2. Run PnP-ADMM for \( t \geq 1 \) iterations under Assumptions 6.1-6.3 using a penalty parameter \( \gamma > 0 \). Then, the sequence \( v^k = z^k - s^{k-1} \) satisfies

\[
\frac{1}{t} \sum_{k=1}^{t} \|S(v^k)\|_2^2 \leq \frac{(R + 2\gamma L)^2}{t} .
\]  

(6.7)

Both proofs are provided in the Appendix B.1. The proof of Theorem 6.2 is a modification of the analysis in [193], obtained by relaxing the strong convexity assumption in [193] by Assumption 6.1 and replacing the assumption that \( R_\sigma \) is a contraction in [193] by Assumption 6.2. Theorem 6.2 establishes that the iterates of PnP-ADMM satisfy \( \|S(v^t)\| \to 0 \) as \( t \to \infty \). Since \( S \) is firmly nonexpansive (see Appendix B.3) and \( D_\sigma \) is nonexpansive, the Krasnosel’skii-Mann theorem (see Section 5.2 in [14]) directly implies that \( v^t \to \text{zer}(S) \) and \( x^t = D_\sigma(v^t) \to \text{zer}(T) \).

Theorem 6.1 establishes that in expectation, IPA has a similar convergence behavior to PnP-ADMM up to an error term that depends on the penalty parameter \( \gamma \). One can precisely control the accuracy of IPA by setting \( \gamma \) to a desired level. In practice, \( \gamma \) can be treated as a hyperparameter and tuned to maximize performance for a suitable image quality metric, such as SNR or SSIM. Our numerical results in Section 6.5 corroborate that excellent SNR
Figure 6.1: Illustration of the influence of the penalty parameter $\gamma > 0$ on the convergence of IPA for a DnCNN prior. The average normalized distance to $\text{zer}(S)$ and SNR (dB) are plotted against the iteration number with the shaded areas representing the range of values attained over 12 test images. The accuracy of IPA improves for smaller values of $\gamma$. However, the SNR performance is nearly identical, indicating that in practice IPA can achieve excellent results for a range of fixed $\gamma$ values.

The performance of IPA can be achieved without taking $\|S(v^t)\|_2$ to zero, which simplifies practical applicability of IPA. (Note that the convergence analysis for IPA in Theorem 6.1 can be easily extended to minibatch IPA with a straightforward extension of Lemma B.1 in Appendix B.1 to several indices, and by following the steps of the main proof in Appendix B.1.)

Finally, note that the convergence of the IPA iterates can also be analyzed under assumptions adopted in [193], namely that $g_i$ are strongly convex and $R_\sigma$ is a contraction. Such an analysis leads to the statement

$$
\mathbb{E} \left[ \|x^t - x^*\|^2 \right] \leq \eta^t (2R + 4\gamma L) + (4\gamma L)/(1 - \eta) ,
$$

where $0 < \eta < 1$. Equation (6.8) establishes a linear convergence to $\text{zer}(T)$ up to an error term. A proof of (6.8) is provided in the Appendix B.2. As corroborated by our simulations in Section 6.5, the actual convergence of IPA holds even more broadly than suggested by both sets of sufficient conditions. This suggests a possibility of future analysis of IPA under more relaxed assumptions.
6.5 Numerical Validation

Recent work has shown the excellent performance of PnP for smooth data-fidelity terms using advanced denoising priors. Our goal in this section is to extend these studies with simulations validating the effectiveness of IPA for nonsmooth data-fidelity terms and DNN priors, as well as demonstrating its scalability to large-scale inverse problems. We consider two applications of the form \( y = Ax + e \), where \( e \in \mathbb{R}^m \) denotes the noise and \( A \in \mathbb{R}^{m \times n} \) denotes either a random Gaussian matrix in compressive sensing (CS) or the transfer function in intensity diffraction tomography (IDT) [131].

Our DNN prior is based on the DnCNN architecture [262], with its batch normalization layers removed for controlling the Lipschitz constant of the network via spectral normalization [195] (see details in Appendix B.6). We train a nonexpansive residual network \( R_\sigma \) by predicting the noise residual from its noisy input. While this means that \( R_\sigma \) is not trained to be firmly nonexpansive, we observed that nonexpansiveness was sufficient for empirical convergence. Note also that a nonexpansive \( R_\sigma \) satisfies the necessary (but not sufficient) condition for firm nonexpansiveness of \( D_\sigma \). It is also worth mentioning that denoiser design, which is not our main focus, is an active area of research in the context of PnP. The training data is generated by adding AWGN to the BSD400 images [140]. The reconstruction quality is quantified using the signal-to-noise ratio (SNR) in dB. We pre-train several DNN models as denoisers for \( \sigma \in [1, 10] \), using \( \sigma \) intervals of 0.5, and use the denoiser achieving the best SNR.
6.5.1 Integration of Nonsmooth Data-Fidelity Terms and Pretrained Deep Priors

We first test IPA on non-smooth data-fidelity terms. The matrix \( A \) is generated with i.i.d. zero-mean Gaussian random elements of variance \( 1/m \), and \( e \) as a sparse Bernoulli-Gaussian vector with the sparsity ratio of 0.1. This means that, in expectation, ten percent of the elements of \( y \) are contaminated by AWGN. The sparse nature of the noise motivates the usage of the \( \ell_1 \)-norm \( g(x) = \|y - Ax\|_1 \), since it is less sensitive to extreme values. The nonsmoothness of \( \ell_1 \)-norm prevents the usage of gradient-based algorithms such as PnP-OPGM. On the other hand, the application IPA is facilitated by efficient strategies for computing the proximal operator [20, 40].

Note that the focus of this section is on using CS as a convenient application for demonstrating some of the key properties of IPA, and is not on achieving the state-of-the-art subsampling in CS [118, 150, 199, 255, 260]. For any subsampling rate, the reconstruction quality of IPA is expected to match that of PnP-ADMM, which has been extensively studied in prior work. In particular, a recent work [136] has extensively compared the recovery performance of PnP relative to several widely-used algorithms in CS.

We set the measurement ratio to be approximately \( m/n = 0.7 \) with AWGN of standard deviation 5. Twelve standard images from Set12 [262] are used in testing, each resized to \( 64 \times 64 \) pixels for rapid parameter tuning and testing. We quantify the convergence accuracy using the normalized distance \( \|S(v^k)\|_2^2/\|v^k\|_2^2 \), which is expected to approach zero as IPA converges to a fixed point.

Theorem 6.1 characterizes the convergence of IPA in terms of \( \|S(v^k)\|_2 \) up to a constant error term that depends on \( \gamma \). This is illustrated in Fig. 6.1 for three values of the penalty parameter.
Figure 6.2: Illustration of scalability of IPA and several widely used PnP algorithms on problems of different sizes. The parameters $n$ and $b$ denote the image size and the number of acquired intensity images, respectively. The average SNR is plotted against time in seconds. Both IPA and PnP-OPGM use random minibatches of 60 measurements at every iteration, while PnP-ADMM and PnP-APGM use all the measurements. The figure highlights the fast empirical convergence of IPA compared to PnP-OPGM as well as its ability to address larger problems compared to PnP-ADMM and PnP-APGM.

$\gamma \in \{\gamma_0, \gamma_0/2, \gamma_0/4\}$ with $\gamma_0 = 0.02$. The average normalized distance $\|S(v^k)\|_2^2/\|v^k\|_2^2$ and SNR are plotted against the iteration number and labeled with their respective final values. The shaded areas represent the range of values attained across all test images. IPA is implemented to use a random half of the elements in $y$ in every iteration to impose the data-consistency. Fig. 6.1 shows the improved convergence of IPA to $\text{zer}(S)$ for smaller values of $\gamma$, which is consistent with our theoretical analysis. Specifically, the final accuracy improves approximately $3\times$ (from $1.07 \times 10^{-5}$ to $3.59 \times 10^{-6}$) when $\gamma$ is reduced from $\gamma_0$ to $\gamma_0/4$. On the other hand, the SNR values are nearly identical for all three experiments, indicating that in practice different $\gamma$ values lead to fixed points of similar quality. This indicates that IPA can achieve high-quality result without taking $\|S(v^k)\|_2$ to zero.

### 6.5.2 Scalability in Large-scale Optical Tomography

We now discuss the scalability of IPA on intensity diffraction tomography (IDT), which is a data intensive computational imaging modality [131]. The goal is to recover the spatial distribution of the complex permittivity contrast of an object given a set of its intensity-only measurements. In this problem, $A$ consists of a set of $b$ complex matrices $[A_1, \ldots, A_b]^T$, where
Table 6.1: Final average SNR (dB) and Runtime obtained by several PnP algorithms on all test images.

<table>
<thead>
<tr>
<th>Simulations</th>
<th>Parameters</th>
<th>Algorithms</th>
<th>SNR in dB (Runtime)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n = 512²</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(b = 300)</td>
<td>PnP-APGM</td>
<td>22.60 (19.4 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-OPGM (60)</td>
<td>22.31 (7.1 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-ADMM</td>
<td>24.23 (7.4 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IPA (60)</td>
<td>23.65 (1.7 min)</td>
</tr>
<tr>
<td></td>
<td>n = 512²</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(b = 600)</td>
<td>PnP-APGM</td>
<td>22.79 (42.6 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-OPGM (60)</td>
<td>22.74 (5.2 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-ADMM</td>
<td>24.40 (14.7 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IPA (60)</td>
<td>23.88 (2 min)</td>
</tr>
<tr>
<td></td>
<td>n = 1024²</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(b = 600)</td>
<td>PnP-APGM</td>
<td>23.56 (8.1 hr)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-OPGM (60)</td>
<td>23.42 (44.3 min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PnP-ADMM</td>
<td>25.50 (1.4 hr)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IPA (60)</td>
<td>24.95 (11 min)</td>
</tr>
</tbody>
</table>

each $A_i$ is a convolution corresponding to the $i$th measurement $y_i$. We adopt the $\ell_2$-norm loss $g(x) = \|y - Ax\|^2_2$ as the data-fidelity term to empirically compare the performance of IPA and PnP-OPGM on the same problem. PnP-OPGM has been implemented with Nesterov acceleration, as in [214].

In the simulation, we follow the experimental setup in [131] under AWGN corresponding to an input SNR of 20 dB. We select six images from the CAT2000 dataset [27] as our test examples, each cropped to $n$ pixels. We assume real permittivity functions, but still consider complex valued measurement operator $A$ that accounts for both absorption and phase [131]. Due to the large size of data, we process the measurements in epochs using minbatches of size 60.

Fig. 6.2 illustrates the evolution of average SNR against runtime for several PnP algorithms, namely PnP-ADMM, PnP-APGM, PnP-OPGM, and IPA, for images of size $n \in \{512 \times 512, 1024 \times 1024\}$ and the total number of intensity measurements $b \in \{300, 600\}$. The final values of SNR as well as the total runtimes are summarized in Table 6.1. The table highlights the overall best SNR performance in bold and the shortest runtime in light-green.

In every iteration, PnP-ADMM and PnP-APGM use all the measurements, while IPA and PnP-OPGM use only a small subset of 60 measurements. IPA thus retains its effectiveness for
Table 6.2: Per-iteration memory usage specification for reconstructing 1024×1024 images

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>PnP-ADMM</th>
<th>IPA (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>size</td>
<td>memory</td>
</tr>
<tr>
<td>{A_i}</td>
<td>real</td>
<td>1024 × 1024 × 600</td>
</tr>
<tr>
<td></td>
<td>imag.</td>
<td>1024 × 1024 × 600</td>
</tr>
<tr>
<td>{y_i}</td>
<td></td>
<td>1024 × 1024 × 600</td>
</tr>
<tr>
<td>others combined</td>
<td></td>
<td>1024 × 1024 × 600</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

large values of $b$, while batch algorithms become significantly slower. Moreover, the scalability of IPA over PnP-ADMM becomes more notable when the image size increases. For example, Table 6.1 highlights the convergence of IPA to 24.95 dB within 11 minutes, while PnP-ADMM takes 1.4 hours to reach a similar SNR value. Note the rapid progress of PnP-ADMM in the first few iterations, followed by a slow but steady progress until its convergence to the values reported in Table 6.1. This behavior of ADMM is well known and has been widely reported in the literature (see Section 3.2.2 “Convergence in Practice” in [32]). We also observe faster convergence of IPA compared to both PnP-OPGM and PnP-APGM, further highlighting the potential of IPA to address large-scale problems where partial proximal operators are easy to evaluate.

Another key feature of IPA is its memory efficiency due to incremental processing of data. The memory considerations in optical tomography include the size of all the variables related to the desired image $x$, the measured data $\{y_i\}$, and the variables related to the forward model $\{A_i\}$. Table 6.2 records the total memory (GB) used by IPA and PnP-ADMM for reconstructing a 1024 × 1024 pixel permittivity image, with the smallest value highlighted in light-green. PnP-ADMM requires 37.63 GB of memory due to its batch processing of the whole dataset, while IPA uses only 3.88 GB—nearly one-tenth of the former—by adopting incremental processing of data. In short, our numerical evaluations highlight both fast and
stable convergence and flexible memory usage of IPA in the context of large-scale optical tomographic imaging.

6.6 Summary

This work provides several new insights into the widely used PnP methodology in the context of large-scale imaging problems. First, we have proposed IPA as a new incremental PnP algorithm. IPA extends PnP-ADMM to randomized partial processing of measurements and extends traditional optimization-based ADMM by integrating pre-trained DNNs. Second, we have theoretically analyzed IPA under a set of realistic assumptions, showing that in expectation IPA can approximate the convergence behavior of PnP-ADMM to a desired precision by controlling the penalty parameter. Third, our simulations highlight the potential of IPA to handle nonsmooth data-fidelity terms, large number of measurements, and DNN priors. We observed faster convergence of IPA compared to several baseline PnP methods, including PnP-ADMM and PnP-OPGM, when partial proximal operators can be efficiently evaluated. IPA can thus be an effective alternative to existing algorithms for addressing large-scale imaging problems. For future work, we would like to explore strategies to further relax our assumptions and explore distributed variants of IPA to enhance its performance in parallel settings.
Part III

Regularization by Denoising
Chapter 7

Overview

Regularization by Denoising (RED) is an alternative framework to PnP that leverages image denoisers to impose regularization within an iterative algorithm. It was first proposed for addressing the PnP’s problem of not having an explicit regularizer, but recently is generalized to denote algorithms that incorporate image denoisers via the formulation of noise residual. In this chapter, we provide a short review of the RED framework, followed by an introduction of the widely-used gradient-method RED (GM-RED) algorithm. Meanwhile, we also discuss the differences between PnP and RED throughout the review.

7.1 Development of RED

Similar to PnP, RED also considers the image reconstruction problem in Eq. (2.11)

\[ \hat{x} = \arg \min_{x \in \mathbb{R}^n} f(x) \quad \text{with} \quad f(x) = g(x) + r(x). \]  

(7.1)
Due to the replacement of proximal operator with an image denoiser, PnP is not always accompanied by a clear definition of the regularizer $r$, losing the interpretation as an optimization. Proposed by Romano et al. [188], RED aims to formulate an explicit function $r$ for a given denoiser. Specifically, the RED regularizer they proposed is of the following form

$$ r(x) = \frac{\tau}{2} x^T(x - D_\sigma(x)). \quad (7.2) $$

where the penalty is the inner product of the image $x$ and its noise residual $x - D_\sigma(x)$, and parameter $\tau > 0$ adjusts the regularization strength. Additionally, the RED regularizer has a simple gradient similar as the noise residual

$$ \nabla r(x) = \tau (x - D_\sigma(x)) \quad (7.3) $$

if the some conditions hold for the denoiser. Assumption 7.1 summarizes the assumptions required for $D_\sigma$.

**Assumption 7.1.** We assume the image denoiser satisfies

(a) **(Local) Homogeneity** [188]. A denoiser applied to a positively scaled image should result in a scaled version of the original image, that is, for any (small) $c \geq 0$

$$ D_\sigma(c \cdot x) = c \cdot D_\sigma(x), $$

which further implies

$$ D_\sigma(x) = \nabla D_\sigma(x) x. $$
(b) **Strong Passivity** [188]. The Jacobian $\nabla D_\sigma(\mathbf{x})$ of the denoising function is stable, meaning

$$\eta(\nabla D_\sigma(\mathbf{x})) \leq 1,$$

where $\eta(\cdot)$ computes the spectral radius of the input matrix.

(c) **Symmetric Jacobian** [185]. The Jacobian $\nabla D_\sigma(\mathbf{x})$ is symmetric, that is

$$[\nabla D_\sigma(\mathbf{x})]^T = \nabla D_\sigma(\mathbf{x}).$$

The benefits of having an explicit $r$ are quite attractive. Firstly, it makes RED interpretable as an optimization, linking denoising priors to the classic literature of regularizers. Secondly, various iterative algorithms can be used to solve (2.11) adapted for RED by using the RED gradient, making the framework more general. Thirdly, with an explicit objective, one can further analyze the optimality of the RED solution by leveraging the existing theoretical results in convex/nonconvex optimization. These benefits has spurred broad interests in using RED to solve various imaging inverse problems, including phase retrieval [148, 248], image restoration [142], and tomographic imaging [249].

Nevertheless, a recent work [185] has empirically shown that many state-of-the-art image denoisers (e.g. BM3D, TNRD, and DnCNN), which have been commonly used within RED, do not satisfy Assumption 7.1(c), thus leading to the nonexistence of RED regularizer (see Theorem 1 in [185]). Instead, they suggested that the noise residual gradient in Eq. (7.3) [185] is a desirable property that accounts for the excellent performance of RED. Until now, the theoretical justification of the RED framework still remains an open yet important question.
Many prior work have theoretically analyzed RED algorithms. Reehorst and Schniter [185] proposed a new algorithm called score-matching by denoising (SMD) and proved their fixed-point convergence under nonexpansive denoisers without relying on an explicit regularizer. Cohen et al. [50] reintroduced the RED via fixed-point projection strategy that corresponds to the following optimization

$$\arg \min_{x \in \mathbb{R}^n} g(x) \quad \text{s.t.} \quad x = D_\sigma(x).$$

The authors further proposed a hybrid steepest descent (HSD) algorithm to solve problem (see Algorithm 4.1 in [50]). Under demicontractive denoisers with non-empty fixed-point set, it can be shown that the above ptimization is convex and HSD can converge to the optimal solution.

**Gradient-Method RED**

One widely-used RED algorithm is based on the gradient-method formulation

$$x^t \leftarrow x^{t-1} - \gamma G(x^{t-1}), \quad \gamma > 0,$$  \hspace{1cm} (7.4a)

where the update direction also relies on a denoising function

$$G(x) := \nabla g(x) + \tau (x - D_\sigma(x)), \quad \tau > 0.$$  \hspace{1cm} (7.4b)

Here, the operator $G$ act like a composite gradient that combines gradient of the data-fidelity term and the noise residual, and $\tau$ adjusts the balance. Figure 7.1 visually compares a single iteration in GM-RED and PnP-ADMM/PGM. Essentially, both PnP and RED aims at finding the solution that balances the data-fidelity and denoising regularizer. On the other
Figure 7.1: Visual comparison of a single iteration in PnP-ADMM/PnP-PGM and GM-RED. While PnP algorithms adopts a serial two-step procedure, GM-RED descends along the direction that is a linear combination of the gradient and noise residual.

hand, GM-RED parallelly differs from PnP by parallelly computing the gradient $\nabla g$ and noise residual $x - D_\sigma(x)$, and descending along the direction that is a linear combination of the two. When the algorithm converges, it converges to the vectors in the zero set of $G$

$$G(x^*) = \nabla g(x^*) + \tau(x^* - D_\sigma(x^*)) = 0$$

$$\iff x^* \in \text{zer}(G) := \{x \in \mathbb{R}^n : G(x) = 0\}. \quad (7.5)$$

Consider the following two sets

$$\text{zer} (\nabla g) := \{x \in \mathbb{R}^n : \nabla g(x) = 0\}$$

and

$$\text{fix}(D_\sigma) := \{x \in \mathbb{R}^n : x = D_\sigma(x)\}, \quad (7.6)$$

where $\text{zer}(\nabla g)$ is the set of all critical points of the data-fidelity and $\text{fix}(D_\sigma)$ is the set of all fixed points of the denoiser. Intuitively, the fixed points of $D_\sigma$ correspond to all the vectors that are not denoised, and therefore can be interpreted as vectors that are noise-free according to the denoiser.
Note that if $\mathbf{x}^* \in \text{zer}(\nabla g) \cap \text{fix}(D_\sigma)$, then $G(\mathbf{x}^*) = \mathbf{0}$ and $\mathbf{x}^*$ is one of the solutions of BC-RED. Hence, any vector that is consistent with the data for a convex $g$ and noiseless according to $D_\sigma$ is in the solution set. When $\text{zer}(\nabla g) \cap \text{fix}(D_\sigma) = \emptyset$, $\mathbf{x}^* \in \text{zer}(G)$ is an equilibrium point [37] balancing the direction towards higher data-fit $\nabla g(\mathbf{x})$ by the direction towards higher regularity $\mathbf{x} - D_\sigma(\mathbf{x})$, explicitly weighted by $\tau > 0$. This explicit control is one of the key differences between RED and PnP.

7.2 Challenges

In the following chapters, we extend the RED framework from two aspects, that is, (1) improving the efficiency in the inference of high-dimensional vectors (2) and providing general theoretical analysis of convergence beyond the original RED conditions. The first challenge differs from what we have addressed for PnP algorithms in the sense that the later focuses on how to process a large dataset of measurements $\mathbf{y}$, while the former aims at efficiently inferring the large unknown signal $\mathbf{x}$. The second challenge is similar to PnP as both frameworks lack an explicit regularizer $r$ useful for analysis; instead, we adopt the monotone operator theory to establish theory to characterize the convergence behavior for RED algorithms.
Chapter 8

Block Coordinate Regularization by Denoising

In this chapter, we develop a new block coordinate RED algorithm that decomposes a large-scale estimation problem into a sequence of updates over a small subset of the unknown variables. We theoretically analyze the convergence of the algorithm and discuss its relationship to the traditional proximal optimization. Our analysis complements and extends recent theoretical results for RED-based estimation methods. We numerically validate our method using several denoiser priors, including those based on convolutional neural network (CNN) denoisers.

The outline for the rest of the chapter is as follows. In Section 8.2, we introduce BC-RED and present its fixed point interpretation. In Section 8.3, we analyze the convergence of BC-RED under several transparent assumptions. In Section 8.4, we provide numerical experiments that illustrate key properties of our method. Section 8.5 concludes the chapter.
8.1 Introduction

Solving the corresponding estimation problem is still a significant computational challenge, especially in the context of high-dimensional vectors $\mathbf{x}$, typical in imaging applications. For example, in optical tomographic microscopy or astronomical image processing, the size of the desired image often reaches the level of megapixels ($10^7$). To address this, we extend the current family of RED algorithms by introducing a new block coordinate RED (BC-RED) algorithm. The algorithm relies on partial updates on $\mathbf{x}$, which makes it scalable to images that would otherwise be prohibitively large for direct processing. Additionally, as we shall see, the overall computational complexity of BC-RED can sometimes be lower than corresponding methods operating on the full image. This behavior is consistent with the traditional coordinate descent methods that can outperform their full gradient counterparts by being able to better reuse local updates and take larger steps [21, 71, 158, 232, 247]. We present two theoretical results related to BC-RED. We first theoretically characterize the convergence of the algorithm under a set of transparent assumptions on the data-fidelity and the denoiser. Our analysis complements the recent theoretical analysis of the traditional RED algorithms in [185] by considering block-coordinate updates and establishing the explicit worst-case convergence rate. Our second result establishes backward compatibility of BC-RED with traditional proximal optimization. We show that when the denoiser corresponds to a proximal operator, BC-RED can be interpreted as an approximate MAP estimator, whose approximation error can be made arbitrarily small. To the best of our knowledge, this explicit link with proximal optimization is missing in the current literature on RED. BC-RED thus provides a flexible, scalable, and theoretically sound algorithm applicable to a wide variety of large-scale imaging problems. We demonstrate BC-RED on image recovery from linear measurements using several denoising priors, including those based on deep neural networks.
8.2 Block Coordinate RED

All the current RED algorithms operate on vectors in \( \mathbb{R}^n \). We propose BC-RED, shown in Algorithm 10, to allow for partial randomized updates on \( x \). Consider the decomposition of \( \mathbb{R}^n \) into \( b \geq 1 \) subspaces

\[
\mathbb{R}^n = \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \cdots \times \mathbb{R}^{n_b} \quad \text{with} \quad n = n_1 + n_2 + \cdots + n_b.
\]

For each \( i \in \{1, \ldots, b\} \), we define the matrix \( U_i : \mathbb{R}^{n_i} \to \mathbb{R}^n \) that injects a vector in \( \mathbb{R}^{n_i} \) into \( \mathbb{R}^n \) and its transpose \( U_i^T \) that extracts the \( i \)th block from a vector in \( \mathbb{R}^n \). Then, for any \( x = (x_1, \ldots, x_b) \in \mathbb{R}^n \)

\[
x = \sum_{i=1}^{b} U_i x_i \quad \text{with} \quad x_i = U_i^T x \in \mathbb{R}^{n_i}, \; i = 1, \ldots, b
\] (8.1)

which is equivalent to \( \sum_{i=1}^{b} U_i U_i^T = I \). Note that (8.1) directly implies the norm preservation \( \|x\|_2^2 = \|x_1\|_2^2 + \cdots + \|x_b\|_2^2 \) for any \( x \in \mathbb{R}^n \). We are interested in a block-coordinate algorithm that uses only a subset of operator outputs corresponding to coordinates in some block \( i \in \{1, \ldots, b\} \). Hence, for an operator \( G : \mathbb{R}^n \to \mathbb{R}^n \), we define the block-coordinate operator \( G_i : \mathbb{R}^n \to \mathbb{R}^{n_i} \) as

\[
G_i(x) := [G(x)]_i = U_i^T G(x) \in \mathbb{R}^{n_i}, \quad x \in \mathbb{R}^n.
\] (8.2)

The proposed BC-RED algorithm is summarized in Algorithm 10. Note that when \( b = 1 \), we have \( n = n_1 \) and \( U_1 = U_1^T = I \). Hence, the theoretical analysis in this chapter is also applicable to the full-gradient RED algorithm in (7.4a).

107
Algorithm 10 Block Coordinate Regularization by Denoising (BC-RED)

1: **input:** initial value $\mathbf{x}^0 \in \mathbb{R}^n$, parameter $\tau > 0$, and step-size $\gamma > 0$.
2: **for** $k = 1, 2, 3, \ldots$ **do**
3: Choose an index $i_k \in \{1, \ldots, b\}$
4: $\mathbf{x}^k \leftarrow \mathbf{x}^{k-1} - \gamma U_{i_k} G_{i_k}(\mathbf{x}^{k-1})$
5: where $G_{i_k}(\mathbf{x}) := U_{i_k}^\top G(\mathbf{x})$
6: with $G(\mathbf{x}) := \nabla g(\mathbf{x}) + \tau(\mathbf{x} - D_\sigma(\mathbf{x}))$.
7: **end for**

As with traditional coordinate descent methods (see [247] for a review), BC-RED can be implemented using different block selection strategies. The strategy adopted for our theoretical analysis selects block indices $i_k$ as independent and identically distributed (i.i.d.) random variables distributed uniformly over $\{1, \ldots, b\}$. An alternative is to proceed in epochs of $b$ consecutive iterations, where at the start of each epoch the set $\{1, \ldots, b\}$ is reshuffled, and $i_k$ is then selected consecutively from this ordered set. We numerically compare the convergence of both BC-RED variants in Section 8.4.

BC-RED benefits from considerable flexibility compared to the full-gradient RED. Since each update is restricted to only one block of $\mathbf{x}$, the algorithm is suitable for parallel implementations and can deal with problems where the vector $\mathbf{x}$ is distributed in space and in time. However, the maximal benefit of BC-RED is achieved when $G_{i_k}$ is efficient to evaluate. Fortunately, it was systematically shown in [176] that many operators—common in machine learning, image processing, and compressive sensing—admit coordinate friendly updates.

For a specific example, consider the least-squares data-fidelity $g$ and a patch-wise denoiser $D_\sigma$. Define the residual vector $r(\mathbf{x}) := A\mathbf{x} - \mathbf{y}$ and consider a single iteration of BC-RED that produces $\mathbf{x}^+$ by updating the $i$th block of $\mathbf{x}$. Then, the update direction and the residual
update are computed as

\[ G_i(x) = A_i^T r(x) + \tau(x_i - D_\sigma(x_i)) \]

and

\[ r(x^+) = r(x) - \gamma A_i G_i(x), \tag{8.3} \]

where \( A_i \in \mathbb{R}^{m \times n_i} \) is a submatrix of \( A \) consisting of the columns corresponding to the \( i \)th block. In many problems of practical interest [176], the complexity of working with \( A_i \) is roughly \( b \) times lower than with \( A \). Also, many advanced denoisers can be effectively applied on image patches rather than on the full image [34, 68, 276]. Therefore, in such settings, the speed of \( b \) iterations of BC-RED is expected to be comparable to a single iteration of the full-gradient RED (see also the discussion in Appendix C.3).

8.3 Convergence Analysis and Compatibility with Proximal Optimization

In this section, we present two theoretical results related to BC-RED. We first establish its convergence to an element of \( \text{zer}(G) \) and then discuss its compatibility with the theory of proximal optimization.

8.3.1 Fixed Point Convergence of BC-RED

Our analysis requires several assumptions that together serve as sufficient conditions for convergence.
Assumption 8.1. The operator $G$ is such that $\text{zer}(G) \neq \emptyset$. There is a finite number $R$ such that the distance of the initial $x^0 \in \mathbb{R}^n$ to the farthest element of $\text{zer}(G)$ is bounded, that is
\[
\max_{x^* \in \text{zer}(G)} \|x^0 - x^*\|_2 \leq R.
\]

This assumption is related to the existence of minimizers in the literature on traditional coordinate minimization [21, 158, 232, 247]. The assumption that $\text{zer}(G) \neq \emptyset$ is analogous to assuming that $\text{zer}(\nabla f) \neq \emptyset$ when minimizing a convex and smooth function $f$. In this setting, $\text{zer}(\nabla f)$ fully characterizes the minimizers of $f$. Thus, $\text{zer}(\nabla f) = \emptyset$, implies that $f$ does not have minimizers. Similarly, $\text{zer}(G) = \emptyset$ implies that (7.5) has no solutions, which this assumption precludes.

The next two assumptions rely on Lipschitz constants along directions specified by specific blocks. We say that $G_i$ is block Lipschitz continuous with constant $\lambda_i > 0$ if
\[
\|G_i(x) - G_i(y)\|_2 \leq \lambda_i \|h_i\|_2,
\]
where $x = y + U_i h_i$, $y \in \mathbb{R}^n$, and $h_i \in \mathbb{R}^{n_i}$. When $\lambda_i = 1$, we say that $G_i$ is block nonexpansive. Note that if an operator $G$ is globally $\lambda$-Lipschitz continuous, then it is straightforward to see that each $G_i = U_i^T G$ is also block $\lambda$-Lipschitz continuous.

Assumption 8.2. The function $g$ is continuously differentiable and convex. Additionally, the block gradient $\nabla_i g$ is block Lipschitz continuous with constant $L_i > 0$ for each $i \in \{1, \ldots, b\}$. We define the largest block Lipschitz constant as $L_{\max} := \max\{L_1, \ldots, L_b\}$.

Let $L > 0$ denote the global Lipschitz constant of $\nabla g$. We always have $L_{\max} \leq L$ and, for some $g$, it may even happen that $L_{\max} = L/b$ [247]. As we shall see, the largest possible step-size $\gamma$ of BC-RED depends on $L_{\max}$, while that of the traditional full-gradient RED on
L. Hence, one natural advantage of BC-RED is that it can often take more aggressive steps compared to the full-gradient RED.

**Assumption 8.3.** The denoiser $D_\sigma$ is such that each block denoiser $D_i^\sigma$ is block nonexpansive.

Since the proximal operator is nonexpansive [168], it automatically satisfies this assumption. We revisit this scenario in a greater depth in Section 8.3.2. We can now establish the following result for BC-RED.

**Theorem 8.1.** Run BC-RED for $t \geq 1$ iterations with random i.i.d. block selection under Assumptions 9.2-9.4 using a fixed step-size $0 < \gamma \leq 1/(L_{\text{max}} + 2\tau)$. Then, we have

$$
E \left[ \frac{1}{t} \sum_{k=1}^{t} \| G(x^{k-1}) \|_2^2 \right] \leq \frac{b(L_{\text{max}} + 2\tau)}{\gamma t} R^2.
$$

(8.5)

A proof of the theorem is provided in Appendix C.1. Theorem 8.1 implies that $E[\|G(x^k)\|_2^2]$ is summable and $E[\|G(x^k)\|_2^2] \to 0$, which establishes the fixed-point convergence of BC-RED in expectation to $\text{zer}(G)$ with $O(1/t)$ rate, thus matching the rate of the traditional gradient-based methods [159]. The proof relies on the monotone operator theory [14, 192], widely used in the context of convex optimization [168], including in the unified analysis of various traditional coordinate descent algorithms [48, 174]. The theorem does not assume the existence of any regularizer $r$, which makes it applicable to denoisers beyond those restricted to Assumption 7.1.

Since $L_{\text{max}} \leq L$, one important implication of Theorem 8.1, is that the upper-bound on the convergence rate (in expectation) of $b$ iterations of BC-RED is better than that of a single iteration of the full-gradient RED (to see this, note that the full-gradient rate is obtained by setting $b = 1, L_{\text{max}} = L$, and removing the expectation in (8.5)). This implies that in coordinate friendly settings (as discussed at the end of Section 8.2), the overall computational
complexity of BC-RED can be lower than that of the full-gradient RED. This gain is primarily due to two factors: (a) possibility to pick a larger step-size $\gamma = 1/(L_{\text{max}} + 2\tau)$; (b) immediate reuse of each local block-update when computing the next iterate (the full-gradient RED updates the full vector before computing the next iterate).

In the special case of $D_{\sigma}(\mathbf{x}) = \mathbf{x} - (1/\tau)\nabla h(\mathbf{x})$, for some convex $r$, BC-RED reduces to the traditional coordinate descent method applied to (7.1). Hence, under the assumptions of Theorem 8.1, one can rely on the analysis of traditional randomized coordinate descent methods in [247] to obtain

$$\mathbb{E}[f(\mathbf{x}^t)] - f^* \leq \frac{2b}{\gamma t} R^2$$

(8.6)

where $f^*$ is the minimum value in (7.1). A proof of (8.6) can be found in [247]. Therefore, such denoisers lead to explicit convex RED regularizers and $O(1/t)$ convergence of BC-RED in terms of the objective. However, as discussed in Section 8.3.2, when the denoiser is a proximal operator of some convex $r$, BC-RED is not directly solving (7.1), but rather its approximation.

Note that Theorem 8.1 only provides sufficient conditions for the convergence of BC-RED. As corroborated by our numerical studies in Section 8.4, the actual convergence of BC-RED is more general and often holds beyond nonexpansive denoisers. One plausible explanation for this is that such denoisers are locally nonexpansive over the set of input vectors used in testing. On the other hand, the recent techniques for spectral-normalization of deep neural nets [79, 153, 195] provide a convenient tool for building globally nonexpansive neural denoisers that result in provable convergence of BC-RED.
8.3.2 Convergence for Proximal Operators

One of the limitations of the current RED theory is in its limited backward compatibility with the theory of proximal optimization. For example, as discussed in [188] (see section “Can we mimic any prior?”), the popular total variation (TV) denoiser [191] cannot be justified with the original RED regularization function (7.2). In this section, we show that BC-RED (and hence also the full-gradient RED) can be used to solve (7.1) for any convex, closed, and proper function $r$. We do this by establishing a formal link between RED and the concept of Moreau smoothing, widely used in nonsmooth optimization [154, 187, 258]. In particular, we consider the following proximal-operator denoiser

$$D_{\sigma}(z) = \text{prox}_{(1/\tau)h}(z)$$

$$= \arg \min_{x \in \mathbb{R}^n} \left\{ \frac{1}{2} \|x - z\|^2_2 + \frac{1}{\tau} h(x) \right\},$$

where $\tau > 0$, $z \in \mathbb{R}^n$, and $r$ is a closed, proper, and convex function [168]. Since the proximal operator is nonexpansive, Assumption 9.4 is automatically satisfied. Our analysis, however, requires an additional assumption using the constant $R$ defined in Assumption 9.2.

**Assumption 8.4.** There is a finite number $G$ that bounds the largest subgradient of $r$, that is

$$\max\{\|x_i(x)\|_2 : x_i(x) \in \partial h(x), x \in B(x^0, R)\} \leq G,$$

where $B(x^0, R) := \{x \in \mathbb{R}^n : \|x - x^0\|_2 \leq R\}$ denotes a ball of radius $R$, centered at $x^0$.

This assumption on boundedness of the subgradients holds for a large number of regularizers used in practice, including both TV and the $\ell_1$-norm penalties. We can now establish the following result.
Theorem 8.2. Run BC-RED for $t \geq 1$ iterations with random i.i.d. block selection and the denoiser (8.7) under Assumptions 9.2-8.4 using a fixed step-size $0 < \gamma \leq 1/(L_{\max} + 2\tau)$. Then

$$
\mathbb{E} [f(x^t)] - f^* \leq \frac{2b}{\gamma t} R^2 + \frac{G^2}{2\tau},
$$

where the function $f$ is defined in (7.1) and $f^*$ is its minimum.

The theorem is proved in Appendix C.2. It establishes that BC-RED in expectation approximates the solution of (7.1) with an error bounded by $(G^2/(2\tau))$. For example, by setting $\tau = \sqrt{t}$ and $\gamma = 1/(L_{\max} + 2\sqrt{t})$, one obtains the following bound

$$
\mathbb{E} [f(x^t)] - f^* \leq \frac{1}{\sqrt{t}} \left[ 2b(L_{\max} + 2)R^2 + G^2 \right].
$$

When $r(x) = -\log(p_x(x))$, the proximal operator corresponds to the MAP denoiser, and the solution of BC-RED corresponds to an approximate MAP estimator. This approximation can be made as precise as desired by considering larger values for the parameter $\tau > 0$. Note that this further justifies the RED framework by establishing that it can be used to compute a minimizer of any proper, closed, and convex (but not necessarily differentiable) $r$. Therefore, our analysis strengthens RED by showing that it can accommodate a much larger class of explicit regularization functions.

8.4 Numerical Validation

There is a considerable recent interest in using advanced priors in the context of image recovery from underdetermined ($m < n$) and noisy measurements. Recent work [24, 142, 148, 185, 188] suggests significant performance improvements due to advanced denoisers (such as BM3D [54] or DnCNN [262]) over traditional sparsity-driven priors (such as TV [191]). Our
goal is to complement these studies with simulations validating our theoretical analysis and providing additional insights. The simulations in this chapter were performed on a machine equipped with an Intel Xeon E5-2620 v4 that has 8 cores of 2.1 GHz and 264 GBs of DDR memory. We trained all neural nets using NVIDIA RTX 2080 GPUs (see Appendix C.4.1 for the details on training).

We consider inverse problems of form $y = Ax + e$, where $e \in \mathbb{R}^m$ is an AWGN vector and $A \in \mathbb{R}^{m \times n}$ is a matrix corresponding to either a sparse-view Radon transform, i.i.d. zero-mean Gaussian random matrix of variance $1/m$, or radially subsampled two-dimensional Fourier transform. Such matrices are commonly used in the context of computerized tomography (CT) [98], compressive sensing [39, 63], and magnetic resonance imaging (MRI) [115], respectively. In all simulations, we set the measurement ratio to be approximately $m/n = 0.5$ with AWGN corresponding to input signal-to-noise ratio (SNR) of 30 dB and 40 dB. Throughout this chapter, we define SNR using the follows equation

$$\text{SNR}(\hat{y}, y) \triangleq 20 \log_{10} \left( \frac{\|y\|_2}{\|y - \hat{y}\|_2} \right)$$
Figure 8.2: The architecture of two variants of DnCNN* used in our simulations. Each neural net is trained to remove AWGN from noisy input images. **Residual** denoiser is trained to predict the noise from the input. The final desired denoiser $D_\sigma$ is obtained by simply subtracting the predicted noise from the input $D_\sigma(z) = z - \text{DnCNN}^*(z)$. **Direct** denoiser is trained to directly output a clean image from a noisy input $D_\sigma(z) = \text{DnCNN}^*(z)$. In some experiments, we further constrain the Lipschitz constant (LC) of the direct denoiser to LC = 1 and of the residual denoiser to LC = 2 using spectral normalization [195]. LC = 1 implies a nonexpansive denoiser. A residual $R = 1 - D_\sigma$ with LC = 2 provides a necessary (but not sufficient) condition for a nonexpansive denoiser.

where $\hat{y}$ represents the noisy vector and $y$ denotes the ground truth. Figure D.2 shows the images we used, which correspond to 10 images randomly selected from the NYU fastMRI dataset [259], resized to be $160 \times 160$ pixels. BC-RED is set to work with 16 blocks, each of size $40 \times 40$ pixels. The reconstruction quality is quantified using SNR averaged over all ten test images.

In addition to well-studied denoisers, such as TV and BM3D, we design our own deep neural net denoiser denoted DnCNN*, which is a simplified version of the popular DnCNN denoiser (see Figure 8.2 for illustration and Appendix C.4.1 for details). This simplification reduces the computational complexity of denoising, which is important when running many iterations of BC-RED. Additionally, it makes it easier to control the global Lipschitz constant (LC)
Figure 8.3: Evolution of the images reconstructed by BC-RED using the DnCNN\(^*\) denoiser for different values of \(\tau\). The first row corresponds to Fourier matrix with 30 dB noise, while the second row corresponds to the Radon matrix with 40 dB noise. Each reconstructed image is marked with its SNR value with respect to the ground truth image. The optimal parameters \(\tau^*\) for the two problems are 0.0037 and 2.35, respectively. The denoiser used in this simulation is the residual DnCNN\(^*\) with a Lipschitz constant LC = 2. This figure illustrates how \(\tau\) enables an explicit tradeoff between the data-fit and the regularization.

of the neural net via spectral-normalization [195]. We train DnCNN\(^*\) for the removal of AWGN at four noise levels corresponding to \(\sigma \in \{5, 10, 15, 20\}\). For each experiment, we select the denoiser achieving the highest SNR value. Note that the \(\sigma\) parameter of BM3D is also fine-tuned for each experiment from the same set \(\{5, 10, 15, 20\}\).

Theorem 8.1 establishes the convergence of BC-RED in expectation to an element of \(\text{zer}(G)\). This is illustrated in Figure 8.4 (left) for the Radon matrix with 30 dB noise and a nonexpansive DnCNN\(^*\) denoiser. The average value of \(\|G(x^k)\|^2_2/\|G(x^0)\|^2_2\) is plotted against the iteration number for the full-gradient RED and BC-RED, with \(b\) updates of BC-RED (each modifying a single block) represented as one iteration. We numerically tested two block selection rules for BC-RED (i.i.d. and epoch) and observed that processing in randomized epochs leads to a faster convergence. For reference, the figure also plots the normalized squared norm of the gradient mapping vectors produced by the traditional PGM with TV [20]. The shaded
Table 8.1: Average signal-to-noise ratios (SNRs) computed over 10 test images for different inverse problems and noise levels. The best SNR for each experiment is highlighted in bold-italic, while the best denoiser prior is in light-green.

<table>
<thead>
<tr>
<th>Forward models</th>
<th>FISTA (TV)</th>
<th>UNet</th>
<th>RED</th>
<th>BC-RED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>TV</td>
<td></td>
</tr>
<tr>
<td>Radon 30 dB</td>
<td>20.66</td>
<td></td>
<td>20.79</td>
<td>20.78</td>
</tr>
<tr>
<td></td>
<td>24.40</td>
<td></td>
<td>21.55</td>
<td>21.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24.38</td>
<td>21.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24.42</td>
<td>24.42</td>
</tr>
<tr>
<td>Radon 40 dB</td>
<td>26.07</td>
<td></td>
<td>25.64</td>
<td>25.70</td>
</tr>
<tr>
<td></td>
<td>28.42</td>
<td></td>
<td>27.82</td>
<td>27.88</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>27.80</td>
<td>27.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>28.30</td>
<td>28.30</td>
</tr>
<tr>
<td>Radon 50 dB</td>
<td>28.74</td>
<td></td>
<td>29.67</td>
<td>28.71</td>
</tr>
<tr>
<td></td>
<td>30.42</td>
<td></td>
<td>29.89</td>
<td>29.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>29.97</td>
<td>29.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.32</td>
<td>30.32</td>
</tr>
<tr>
<td>Random 30 dB</td>
<td>29.99</td>
<td></td>
<td>29.97</td>
<td>29.79</td>
</tr>
<tr>
<td></td>
<td>29.62</td>
<td></td>
<td>29.89</td>
<td>29.89</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>29.97</td>
<td>29.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.32</td>
<td>30.32</td>
</tr>
<tr>
<td>Random 40 dB</td>
<td>30.42</td>
<td></td>
<td>30.42</td>
<td>30.42</td>
</tr>
<tr>
<td></td>
<td>30.62</td>
<td></td>
<td>30.62</td>
<td>30.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.62</td>
<td>30.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.62</td>
<td>30.62</td>
</tr>
<tr>
<td>Fourier 30 dB</td>
<td>28.74</td>
<td></td>
<td>28.67</td>
<td>28.71</td>
</tr>
<tr>
<td></td>
<td>29.99</td>
<td></td>
<td>29.97</td>
<td>29.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.32</td>
<td>30.32</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.32</td>
<td>30.32</td>
</tr>
<tr>
<td>Fourier 40 dB</td>
<td>29.99</td>
<td></td>
<td>29.97</td>
<td>29.99</td>
</tr>
<tr>
<td></td>
<td>30.42</td>
<td></td>
<td>30.42</td>
<td>30.42</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.62</td>
<td>30.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.62</td>
<td>30.62</td>
</tr>
</tbody>
</table>

Figure 8.4: Illustration of the convergence of BC-RED under two DnCNN* priors. The left plot correspond to the direct DnCNN* with the LC = 1, while the right plot correspond to the residual DnCNN* with LC = 2. The average normalized distance to zer(G) is plotted against the iteration number for the Radon matrix with the shaded areas representing the range of values attained over all test images. Note that LC = 1 implies a nonexpansive denoiser, and LC = 2 provides a necessary (but not sufficient) condition for a nonexpansive denoiser.

areas indicate the range of values taken over 10 runs corresponding to each test image. The results highlight the potential of BC-RED to enjoy a better convergence rate compared to the full-gradient RED, with BC-RED (epoch) achieving the accuracy of $10^{-10}$ in 104 iterations, while the full-gradient RED achieves the same accuracy in 190 iterations. Additionally, the faster convergence in time of BC-RED, compared to the full-gradient RED, is also empirically highlighted in Figure 8.5 for the Random matrix with 30 dB noise and the nonexpansive DnCNN* denoiser. Specifically, BC-RED (epoch) achieves the accuracy of $10^{-13}$ in 100 seconds, while the full-gradient RED achieves the accuracy of $10^{-9}$ in the same amount of
Figure 8.5: Illustration of the run-time convergence of BC-RED under two DnCNN* priors. The left plot correspond to the **direct DnCNN** with the **LC = 1**, while the right plot correspond to the **residual DnCNN** with **LC = 2**. The average normalized distance to \( \text{zer}(G) \) is plotted against the run-time for the Random matrix with the shaded areas representing the range of values attained over all test images. The run-time convergence of PGM with TV is also plotted for reference.

Figure 8.6: Illustration of the influence of the parameter \( \tau > 0 \) for solving TV regularized least-squares problem using BC-RED. As \( \tau \) increases, BC-RED provides an increasingly accurate approximation to the TV optimization problem.

time. This speedup shows the potential of of BC-RED to lead to faster convergence when applied to coordinate friendly reconstruction problems.

Theorem 8.2 establishes that for proximal-operator denoisers, BC-RED computes an approximate solution to (7.1) with an accuracy controlled by the parameter \( \tau \). This is illustrated in Figure 8.6 for the Fourier matrix with 40 dB noise and the TV regularized least-squares problem. The average value of \( (f(x^k) - f^*)/(f(x^0) - f^*) \) is plotted against the iteration number for BC-RED with \( \tau \in \{0.01, 0.1, 1\} \). The optimal value \( f^* \) is obtained by running the traditional PGM until convergence. As before, the figure groups \( b \) updates of BC-RED
as a single iteration. The results are consistent with our theoretical analysis and show that as $\tau$ increases BC-RED provides an increasingly accurate solution to TV. Since the range of possible values for the step-size $\gamma$ depends on $\tau$, the speed of convergence to $f^*$ is also influenced by $\tau$.

In BC-RED, the parameter $\tau$ controls the tradeoff between $\text{zer}(\nabla g)$ and $\text{fix}(D\sigma)$. Figure 8.3 illustrates evolution of images reconstructed by BC-RED for different $\tau$. The first row corresponds to the reconstruction from the Fourier measurements with 30 dB noise, while the second row corresponds to the Radon measurements with 40 dB noise. The figure clearly shows how $\tau$ explicitly adjusts the balance between the data-fit and the denoiser. In particular, small $\tau$, corresponding to weak denoising, results in unwanted artifacts in the reconstructed images, while large $\tau$ promotes denoising strength but smooths out desired features and details. The leftmost images in Figure 8.3 shows the optimal balance introduced by $\tau^*$.

The benefits of the full-gradient RED algorithms have been well discussed in prior work [24, 142, 148, 185, 188]. Table B.3 summarizes the average SNR performance of BC-RED in comparison to the full-gradient RED for all three matrix types and several priors. Corresponding visual results are illustrated in Figure 8.7. Unlike the full-gradient RED, BC-RED is implemented using block-wise denoisers that work on image patches rather than the full images. We empirically found that 40 pixel padding on the denoiser input is sufficient for BC-RED to match the performance of the full-gradient RED (see Appendix C.4.3 for additional details). The table also includes the results for the traditional PGM with TV [20] and the widely-used end-to-end U-Net approach [80, 95]. The latter first backprojects the measurements into the image domain and then denoises the result using U-Net [189]. The model was specifically trained end-to-end for the Radon matrix with 30 dB noise and applied as such to other measurement settings. All the algorithms were run until convergence with hyperparameters optimized for SNR. The DnCNN$^*$ denoiser in the table corresponds to the residual network.
with LC = 2. The overall best SNR in the table is highlighted in bold-italic, while the best RED prior is highlighted in light-green. First, note the excellent agreement between BC-RED and the full-gradient RED. This close agreement between two methods is encouraging as BC-RED relies on block-wise denoising and our analysis does not establish uniqueness of the solution, yet, in practice, both methods seem to yield solutions of nearly identical quality. Second, note that BC-RED and RED provide excellent approximations to PGM-TV solutions. Third, note how (unlike U-Net) BC-RED and RED with DnCNN* generalize to different measurement models. Finally, no prior seems to be universally good on all measurement settings, which indicates to the potential benefit of tailoring specific priors to specific measurement models.

Figure 8.7 visually compares the images recovered by BC-RED and RED and two baseline methods. First, the images visually illustrate the excellent agreement between BC-RED and RED. Second, leveraging advanced denoisers in BC-RED largely improves the reconstruction quality over PGM with the traditional TV prior. For instance, BC-RED under DnCNN* outperforms PGM under TV by 1 dB for Fourier matrix. Finally, we note the stability of BC-RED using the deep neural net denoiser versus the deteriorating performance of U-Net, which is trained end-to-end for Radon matrix with 30 dB noise. This fact highlights one key merit of the RED framework, that the denoiser, only trained once, can be directly applied in different scenarios for different tasks with no degradation.
Figure 8.7: Visual comparison between BC-RED and RED against PGM (TV) and U-Net for all three matrices with 30 dB noise. For BC-RED and RED, we selected the denoiser resulting in the best reconstruction performance. Every image is marked by its SNR value with respect to the ground truth. We highlight the excellent agreement between BC-RED and RED in all experiments. Note the strong degradation in the image quality for U-Net, due to the mismatch between the training and testing.
8.5 Summary

Coordinate descent methods have become increasingly important in optimization for solving large-scale problems arising in data analysis. We have introduced BC-RED as a coordinate descent extension to the current family of RED algorithms and theoretically analyzed its convergence. Our analysis provides two complementary interpretations for BC-RED: (i) equilibrium interpretation where the algorithm balances the direction towards higher data-fit by the direction towards higher regularity; (ii) minimization interpretation where the algorithm can perform traditional regularized inversion for arbitrary convex regularizers. Preliminary experiments suggest that BC-RED can be an effective tool in large-scale estimation problems arising in image recovery. More experiments are certainly needed to better assess the promise of this approach in various estimation tasks. The method and analysis presented in this chapter can be extended in several complementary ways. One interesting direction would be to allow the algorithm to consider overlapping blocks, as is typically done in patch-based image denoising [34, 54]. Another interesting direction would be to consider improving the efficiency of the algorithm by designing data-adaptive block selection strategies [156]. Finally, one might consider going beyond gradient-based algorithms by designing block-coordinate variants of ADMM operating on image patches.
Chapter 9

Asynchronous Block Parallel Stochastic Regularization by Denoising

Parallel Computing is particularly proposed and beneficial for inferring high-dimensional vector. In this chapter, we propose a new asynchronous RED (Async-RED) algorithm that enables asynchronous parallel processing of data, making it significantly faster than its serial counterparts for large-scale inverse problems. The computational complexity of Async-RED is further reduced by using a random subset of measurements at every iteration. We present complete theoretical analysis of the algorithm by establishing its convergence under explicit assumptions on the data-fidelity and the denoiser. We validate Async-RED on image recovery using pre-trained deep denoisers as priors.
Figure 9.1: Visual illustration of serial and parallel image recovery on a multicore system. (a) Serial processing uses only one core of the system for every iteration. (b) Synchronous parallel processing has to wait for the slowest core to finish before starting the next iteration. (c) Asynchronous parallel processing can continuously iterate using all the cores without waiting. (d) Asynchronous parallel processing using the stochastic gradient leads to additional flexibility. (a), (b), and (c) use all the corresponding measurements at every iteration, while (d) uses only a small random subset at a time. Async-RED adopts the schemes shown in (c) and (d).

9.1 Introduction

In the previous chapters, we have discussed several PnP/RED algorithms, including those based on online gradients (PnP-OPGM/IPA) and block coordinate updates (BC-RED). Although these algorithms have significantly improved from their original batch counterparts, they inherently adopt a serial procedure. As illustrated in Figure 9.1, this makes them suboptimal on multicore systems that are mostly beneficial for processing large-scale datasets [183]. We address this gap by proposing a novel asynchronous RED algorithm. The algorithm decomposes the inference problem into a sequence of partial (block-coordinate) updates on $\mathbf{x}$ executed asynchronously in parallel over a multicore system. Async-RED leads to a more efficient usage of available cores by avoiding synchronization of partial updates. Async-RED is also scalable in terms of the number of measurements, since it processes only a small random subset of $\mathbf{y}$ at every iteration. We present two new theoretical results on the convergence of Async-RED based on a unified set of explicit assumptions on the data-fidelity
and the denoiser. Specifically, we establish its fixed-point convergence in the batch setting and extend this analysis to the randomized minibatch scenario. Our results extend recent work on serial block-coordinate RED (BC-RED) [213] and are fully consistent with the traditional asynchronous parallel optimization methods [128, 212]. We numerically validate Async-RED on image recovery from linear and noisy measurements using pre-trained deep denoisers as image priors.

9.2 Asynchronous RED

Async-RED allows efficient processing of data by simultaneously considering the asynchronous partial updates of solution $x$ and the use of randomized subset of measurements $y$. In this section, we introduce the algorithmic details of our method. We start with the basic batch formulation of Async-RED (Async-RED-BG) followed by its minibatch variant (Async-RED-SG).

9.2.1 Async-RED using Batch Gradient

When the gradient uses all the measurements $y \in \mathbb{R}^m$, Async-RED-BG is the asynchronous extension of the recent block-coordinate RED (BC-RED) algorithm [213]. Consider the decomposition of the variable space $\mathbb{R}^n$ into $b \geq 1$ blocks

$$x = (x_1, \ldots, x_b) \in \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_b} = \mathbb{R}^n \quad \text{with} \quad n = n_1 + n_2 + \cdots + n_b.$$  

For each $i \in \{1, \ldots, b\}$, we introduce the operator $U_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^n$ that injects a vector in $\mathbb{R}^{n_i}$ into $\mathbb{R}^n$ and its transpose $U_i^T$ that extracts the $i$th block from a vector in $\mathbb{R}^n$. This directly
implies that

\[ I = U_1 U_1^T + \cdots + U_b U_b^T \quad \text{and} \quad \|x\|_2^2 = \|x_1\|_2^2 + \cdots + \|x_b\|_2^2 \quad \text{with} \quad x_i = U_i^T x. \quad (9.1) \]

In analogy to the RED operator \( G \) in (7.5), we define the block-coordinate operator \( G_i \) as

\[ G_i(x) := U_i U_i^T G(x), \quad \text{with} \quad x \in \mathbb{R}^n \quad \text{and} \quad G_i : \mathbb{R}^n \rightarrow \mathbb{R}^n. \quad (9.2) \]

Due to the asynchrony in the block updates, the iterate might be updated several times by different cores during a single update cycle of a core, which means that the evaluation of \( x^{k+1} \) relies on a stale iterate \( \tilde{x}^k \)

\[ x^{k+1} \leftarrow x^k - \gamma G_{i_k}(\tilde{x}^k), \quad \text{with} \quad \tilde{x}^k = x^k + \sum_{s=k-\Delta_k}^{k-1} (x^s - x^{s+1}), \quad \Delta_k \leq \lambda. \quad (9.3) \]

Here, we assume that the stale iterate \( \tilde{x}^k \) exits as a state of \( x \) in the shared memory, and the delay between them is bounded by a finite number \( \lambda \in \mathbb{Z}_+ \). These two assumptions are often referred to as the consistent read [183] and the bounded delay [133] in the traditional asynchronous block coordinate optimization. Although we implement the consistent read in Async-RED, the algorithm never imposes a global lock on \( x^k \). We refer to Appendix D.1 for the related discussion.

The first variant, Async-RED-BG, is summarized in Algorithm 11, where \( \text{read}(\cdot) \) reads a block from the shared memory to the local memory. When the algorithm is run on a single core system without parallelization (that is to say \( \tilde{x}^k = x^k \)), it reduces to the normal BC-RED algorithm. Hence, our analysis is also applicable to BC-RED.

We specifically consider the random block selection strategy in Async-RED-BG, namely that every block index \( i_k \) is selected as an i.i.d random variable uniformly distributed over
Algorithm 11 Asynchronous RED with Batch Gradient (Async-RED-BG)

1: **input:** \( \mathbf{x}^0 \in \mathbb{R}^n, \gamma > 0, \tau > 0. \)
2: **setup:** A multicore system with one shared memory storing \( \mathbf{x} \) and global iteration \( k. \)
3: **for** global \( k = 1, 2, 3, \ldots \) **do**
4: \( \tilde{\mathbf{x}}^k \leftarrow \text{read}(\mathbf{x}) \)
5: \( G_{i_k}(\tilde{\mathbf{x}}^k) \leftarrow U_{i_k}U^T_{i_k}G(\tilde{\mathbf{x}}^k) \) with random \( i_k \in \{1, \ldots, b\} \)  \( \triangleright \) Block Operation
6: \( \mathbf{x}^k \leftarrow \text{read}(\mathbf{x}) \)
7: \( \mathbf{x}^{k+1} \leftarrow \mathbf{x}^k - \gamma G_{i_k}(\tilde{\mathbf{x}}^k) \)
8: update \( \mathbf{x} \) in the shared memory using \( \mathbf{x}^{k+1} \)
9: **end for**

\( \{1, \ldots, b\} \). Such a strategy is commonly adopted for simplifying the convergence analysis. Nevertheless, our method and analysis can be generalized to the scenario where \( i_k \) follows some arbitrary probability \( P(i_k = i) = p_i \) specified by the user.

Compared with serial RED algorithms, Async-RED-BG enjoys considerable scalability by dividing the computation of the full operator \( G \) into \( b \) parallel evaluation of \( G_{i_k} \) distributed across all cores. Thus, without any modification to the algorithmic design, one can easily improve the performance of the algorithm by simply integrating more cores into the system. In Section 9.4, we experimentally demonstrate the significant speed-up and scale-up in solving the context of image recovery.

9.2.2 Async-RED using Stochastic Gradient

The scale of measurements is another important factor influencing the computational complexity in the large-scale inference tasks. Async-RED-SG improves the applicability of Async-RED to these cases by further considering the decomposition of the measurement space \( \mathbb{R}^m \) into \( \ell \geq 1 \) blocks

\[
\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_\ell) \in \mathbb{R}^{m_1} \times \cdots \times \mathbb{R}^{m_\ell} = \mathbb{R}^m \quad \text{with} \quad m = m_1 + m_2 + \cdots + m_\ell.
\]
Algorithm 12 Asynchronous RED with Stochastic Gradient (Async-RED-SG)

1: **input:** $x^0 \in \mathbb{R}^n$, $\gamma > 0$, $\tau > 0$.
2: **setup:** A multicore system with one shared memory storing $x$ and global iteration $k$.
3: **for** global $k = 1, 2, 3, \ldots$ **do**
4: $\tilde{x}^k \leftarrow \text{read}(x)$
5: $\hat{G}(\tilde{x}^k) \leftarrow \text{minibatch}\ G(\tilde{x}^k, w)$ with random $j_w \in \{1, \ldots, \ell\}$ \hspace{1cm} $\triangleright$ Minibatch Gradient
6: $\hat{G}_{ik}(\tilde{x}^k) \leftarrow U_{ik} U_{ik}^T \hat{G}(\tilde{x}^k)$ with random $i_k \in \{1, \ldots, b\}$ \hspace{1cm} $\triangleright$ Block Operation
7: $x^k \leftarrow \text{read}(x)$
8: $x^{k+1} \leftarrow x^k - \gamma \hat{G}_{ik}(\tilde{x}^k)$
9: update $x$ in the shared memory using $x^{k+1}$
10: **end** for

Hence, Async-RED-SG considers the following data-fidelity $g$ and its gradient $\nabla g$

$$g(x) = \frac{1}{\ell} \sum_{j=1}^{\ell} g_j(x) \quad \Rightarrow \quad \nabla g(x) = \frac{1}{\ell} \sum_{j=1}^{\ell} \nabla g_j(x), \quad (9.4)$$

where each $g_j$ is evaluated on the subset $y_j \in \mathbb{R}^{m_j}$ of the full $y$. From (9.4), we know that the computation of $\nabla g(x)$ is proportional to the total number $\ell$. To reduce the per-iteration cost, we follow the idea of stochastic optimization to approximate the batch gradient by using the stochastic gradient that relies on a minibatch of $w \ll \ell$ measurements

$$\hat{\nabla} g(x) = \frac{1}{w} \sum_{s=1}^{w} \nabla g_{j_s}(x), \quad (9.5)$$

where $j_s$ is picked from the set $\{1, \ldots, \ell\}$ as i.i.d uniform random variable. Based on the minibatch gradient, we define the block stochastic operator $\hat{G}_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$ as

$$\hat{G}_i := U_i U_i^T \hat{G}(x), \quad \text{with} \quad \hat{G} := \hat{\nabla} g(x) + \tau (x - D_{\sigma}(x)), \quad \hat{G} : \mathbb{R}^n \rightarrow \mathbb{R}^n. \quad (9.6)$$

Note that the computation of $\hat{G}_i$ is now dependent on the minibatch size $w$ that is adjustable to cope with the computation resources at hand. Async-RED-SG is summarized in Algorithm 12.
The operation \( \text{minibatchG}(\cdot) \) computes the estimate of \( G \) based on \( w \) randomly selected measurements. We clarify the difference between Async-RED-SG and Async-RED-BG via a specific example. Consider the least-squares \( g \) with a block-friendly operator \( A \) and a block-efficient denoiser \( D_\sigma \). We can write the update of Async-RED-BG regarding a single iteration as

\[
G_i(\tilde{x}) = A_i^T(A_i\tilde{x} - y_i) + \tau(\tilde{x}_i - D(\tilde{x}_i)),
\]  
(9.7)

where \( \tilde{x} \) is the delayed iterate for \( x \), and \( A_i \in \mathbb{R}^{m \times n_i} \) is a submatrix of \( A \) consisting of columns corresponding to the \( i \)th blocks. Although the per-iteration complexity is reduced by roughly \( b = n/n_i \) times by working with \( A_i \) instead of \( A \), Async-RED-BG still needs to work with all the measurements \( y_i \) related to the \( i \)th block at every iteration. Consider the corresponding update of Async-RED-SG with one measurement used at a time

\[
\tilde{G}_i(\tilde{x}) = A^T_{ji}(A_{ji}\tilde{x} - y_{ji}) + \tau(\tilde{x}_i - D(\tilde{x}_i)),
\]  
(9.8)

where \( y_{ji} \) denotes the \( j \)th measurement of \( x_i \), and \( A_{ji} \in \mathbb{R}^{m_j \times n_i} \) is the submatrix crossed by the rows and columns corresponding to the \( j \)th measurement and the \( i \)th blocks. This indicates that the reduction of the per-iteration complexity from Async-RED-BG to Async-RED-SG can be up to \( \ell = m/m_j \) times. In the practice, it is common to use \( w > 1 \) measurements at a time to optimize the total runtime. Note that if \( U = U^T = I \), Async-RED-SG becomes the asynchronous stochastic RED algorithm. In the next section, we will present a complete analysis of Async-RED and theoretically discuss its connection to the related algorithms.

### 9.3 Convergence Analysis of Async-RED

The proposed analysis is based on the following explicit assumptions. Note that these assumptions serve as sufficient conditions for the convergence.
**Assumption 9.1.** We assume bounded maximal delay $\lambda < \infty$. Hence, during any update cycle of an agent, the estimate $\mathbf{x}$ in the shared memory is updated at most $\lambda \in \mathbb{Z}_+$ times by other cores.

The value of $\lambda$ is often dependent on the number of cores involved in the computation [247]. If every core takes a similar amount of time to compute its update, $\lambda$ is expected to be a multiple of the number of cores. Related work has investigated the convergence with unbounded maximal delays in the context of traditional optimization [82, 177, 273].

**Assumption 9.2.** The operator $G$ is such that $\text{zer}(G) \neq \emptyset$, and the distance of the initial $\mathbf{x}^0 \in \mathbb{R}^n$ to any element in $\text{zer}(G)$ is bounded, that is $\|\mathbf{x}^0 - \mathbf{x}^*\| \leq R$ for all $\mathbf{x}^* \in \text{zer}(G)$ with $R < \infty$.

This assumption ensures the existence of a solution for the RED problem and is related to the existence of minimizers in traditional coordinate minimization [21, 158].

**Assumption 9.3.** (a) Every component function $g_i$ is convex differentiable and has a Lipschitz continuous gradient of constant $L_i > 0$. (b) At every update, the stochastic gradient is unbiased estimator of $\nabla g$ that has a bounded variance:

$$
\mathbb{E}\left[\hat{\nabla} g(\mathbf{x})\right] = g(\mathbf{x}), \quad \mathbb{E}\left[\|\hat{\nabla} g(\mathbf{x}) - \nabla g(\mathbf{x})\|^2\right] \leq \frac{\nu^2}{w}, \quad \mathbf{x} \in \mathbb{R}^n, \quad \nu > 0.
$$

The first part of the assumption implies that $g$ is also convex and has Lipschitz continuous gradient with constant $L = \max\{L_1, \ldots, L_\ell\}$. The second part is a standard assumption on the unbiasedness and variance of the stochastic gradient [78, 128]. Our final assumption is related to the deep denoiser used in Async-RED.

**Assumption 9.4.** The denoiser $D_\sigma$ is a nonexpansive operator $\|D_\sigma(\mathbf{x}) - D_\sigma(\mathbf{y})\| \leq \|\mathbf{x} - \mathbf{y}\|$.
Compared with the RED conditions stated in Section 7.1 (namely, that it is locally homogeneous with a symmetric Jacobian), our requirement on the denoiser is milder. One can train a nonexpansive $D_\sigma$ by constraining the Lipschitz constant of $D_\sigma$ via the spectral normalization, which is an active area of research in deep learning [9, 153, 196, 227].

We can now state the theorems on Async-RED.

**Theorem 9.1.** Let Assumptions 9.1-9.4 hold true. Run Async-RED-BG for $t > 0$ iterations with uniform i.i.d block selection using a fixed step-size $\gamma \in (0, 1/((1 + 2\lambda)(L + 2\tau))]$. Then, the iterates of the algorithm satisfy

$$\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \frac{(L + 2\tau)b}{\gamma t} R^2. \quad (9.9)$$

where $D = 2\lambda^2/(1 + \lambda)^2$ is a constant.

Theorem 9.1 establishes the convergence of Async-RED-BG to the fixed-point set $\text{zer}(G)$ at the rate of $O(1/t)$. Our result is consistent with the existing results in the literature. In particular, when the algorithm adopts serial block updates, that is $\lambda = 0$ and $\tilde{x}^k = x^k$, the recovered convergence is nearly the same as BC-RED [213] scaled by some constant. On the other hand, our convergence rate $O(1/t)$ is also consistent with the rate proved for the asynchronous block coordinate descent in nonconvex optimization [212].

**Theorem 9.2.** Let Assumptions 9.1-9.4 hold true. Run Async-RED-SG for $t > 0$ iterations with uniform i.i.d selections of blocks and measurements using a fixed step-size $\gamma \in (0, 1/((1 + 2\lambda)(L + 2\tau))]$. Then, the iterates of the algorithm satisfy

$$\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \frac{(L + 2\tau)b}{\gamma t} R^2 + \left[ \frac{2D}{b} + 2 \right] \frac{\gamma}{w} C. \quad (9.10)$$

where $C = (L + 2\tau)(1 + \lambda)\nu^2$ and $D = 2\lambda^2/(1 + \lambda)^2$ are constants.
Figure 9.2: Convergence of Async-RED-BG for different numbers of accessible cores \( n_e \in \{2, 4, 6, 8\} \). The left figure plots the average normalized distance to \( \text{zer}(G) \) against the iteration number; the middle and right figures plot these values, as well as SNR, plotted against the actual runtime in seconds. The shaded areas represent the range of values attained over the test images.

Theorem 9.2 states that Async-RED-SG approximates the solution obtained by Async-RED-BG up to a finite error that decreases for larger values of the minibatch size \( w \). This relationship is consistent with the recent theoretical results on the online PnP and RED algorithms \([214, 249]\). In practice, the selection of \( w \) must balance the actual memory capacity of the system and the desired runtime for obtaining a reasonable solution. Our numerical evaluation in Section 9.4 demonstrates the excellent approximation of Async-RED-SG to the batch-gradient solution by using a small subset of data.

By carefully choosing the stepsize \( \gamma \), we can state the following remark on Theorem 9.2.

**Remark 1.** Set the stepsize to be \( \gamma = 1/\sqrt{wt} \). If the maximal delay satisfies \( \lambda \leq (1/2)[\sqrt{wt}/(L + 2\tau) - 1] \), then after \( t > 0 \) iterations we have

\[
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \frac{(L + 2\tau)b}{\sqrt{wt}} R^2 + \left[ \frac{2D}{b} + 2 \right] \frac{C}{\sqrt{wt}}.
\]  

(9.11)

This establishes the fixed-point convergence to the set \( \text{zer}(G) \) at the rate of \( O(1/\sqrt{wt}) \) under specific conditions. If we treat entire \( x \) as a block, namely that \( U = U^T = 1 \) and \( b = 1 \), Async-RED-SG then becomes the asynchronous stochastic RED algorithm. Hence, the proposed remark immediately holds true for the later. Note that our convergence rate \( O(1/\sqrt{wt}) \) is
We now present a numerical validation of Async-RED. Our goals are first to validate the random matrix is implemented with the block-diagonal structure $A$ of our algorithm on the large-scale problem. We consider two image recovery tasks that have the form $y = Ax + e$, where the measurement matrix $A$ corresponds to either the random matrix in compressive sensing (CS) or the Radon transform in computed tomography (CT), and the noise $e$ is assumed to be additive white Gaussian (AWGN). In particular, the random matrix is implemented with the block-diagonal structure $A = \text{diag}([A_1, ..., A_k])$ for consistent with the rate proved for the serial [157] and parallel [60, 128] stochastic gradient methods.

All the proofs are presented in the supplement. We note that the analysis above does not assume the existence of an explicit regularizer associated with the operator $D_\sigma$. Moreover, it does not require $D_\sigma$ to be a Gaussian denoiser. Our analysis is hence applicable to all nonexpansive operators, such as the traditional proximal operators or the more recent artifact-removal operators [261].

### 9.4 Numerical Validation

We now present a numerical validation of Async-RED. Our goals are first to validate the proposed theorems in Section 9.3 and then to demonstrate the effectiveness and the efficiency of our algorithm on the large-scale problem. We consider two image recovery tasks that have the form $y = Ax + e$, where the measurement matrix $A$ corresponds to either the random matrix in compressive sensing (CS) or the Radon transform in computed tomography (CT), and the noise $e$ is assumed to be additive white Gaussian (AWGN). In particular, the random matrix is implemented with the block-diagonal structure $A = \text{diag}([A_1, ..., A_k])$ for

![Figure 9.3: Left: Evolution of the convergence accuracy of Async-RED-SG as the minibatch size $w$ increases. The average distance is plotted against the number of iterations with the shaded areas representing the range of values attained over the test images. Middle & Right: Comparison of convergence speed between Async-RED-BG/SG and other baselines. The right table summarizes the total runtime and the speed-up compared with GM-RED for all algorithms.](image-url)
fast validation, while the Radon transform is used as its full matrix form to demonstrate the
effectiveness of Async-RED for overcoming the computation bottleneck. Our deep neural
net prior adapts the DnCNN architecture [262]. We used the signal-to-noise ratio (dB) to
quantify the quality of the reconstructed images. For each experiments, we selected the
denoiser that achieves the best SNR performance from the ones corresponding to five noise
levels \( \sigma \in \{5, 10, 15, 20, 25\} \). The value of \( \sigma \) is fixed across all iterations of the algorithm.
Appendix D.3 provides additional technical details.

9.4.1 Convergence Behavior

We validate our theorems on the CS task with 6 test images selected from the Set 12
dataset [262]. Each test image is rescaled to the size of 240 x 240 pixels (see Figure D.2 in
the supplement for the visualization). The block-diagonal matrix \( A \) is set to consist of 9
submatrices, corresponding to a 3 x 3 grid of blocks with the size of 80 x 80 pixels in every
image. The elements in \( A \) are i.i.d zero-mean Gaussian random variables of variance of 1/m,
and the compression ratio is set to be \( m/n = 0.7 \), which indicates that the total number of
measurements is 4480 for each block. We obtain the measurements by multiplying \( A \) with
each vectorized image and adding additional noise corresponding to the input SNR of 30
dB. Finally, we use the normalized distance \( \|G(x^k)\|_2^2/\|G(x^0)\|_2^2 \) to quantify the fixed-point
convergence, with \( b \) block updates grouped as one iteration. The distance is expected to
approach zero as the algorithm converges to a fixed point. The average performance of all
methods is obtained by running a single trial for each image.

Theorem 9.1 establishes the convergence of Async-RED-BG to the fixed point set \( \text{zer}(G) \).
This is illustrated in Figure 9.2 for four different numbers of accessible cores \( n_c \in \{2, 4, 6, 8\} \).
In the left figure, the average normalized distance is plotted against the iteration number,
while the middle and right figures plot the corresponding distance and SNR values against the
actual runtime in seconds. The shaded areas representing the range of values attained across all test images. We also plot the results of serial BC-RED using the dashed line as reference. Async-RED-BG is implemented to be run asynchronously on multiple cores, while BC-RED can only use one core to perform the computation. The left figure highlights the fixed-point convergence of Async-RED-BG in iteration for different $n_c$, with all variants agreeing with the serial BC-RED. Since Async-RED-BG uses more cores, the middle and right figures demonstrate the significantly faster in-time convergence of Async-RED-BG than BC-RED to the same SNR value. Specifically, BC-RED takes 1.8 hours to achieve 29.00 dB, while Async-RED-BG ($n_c = 8$) takes only 17.9 minutes to obtain the same value, corresponding to a $6\times$ improvement in computation time.

Theorem 9.2 establishes the convergence of Async-RED-SG to $\text{zer}(G)$ up to some error term, which is inversely proportional to the minibatch size $w$. This is illustrated in Figure 9.3 (left) for three different minibatch sizes $w \in \{1120, 2240, 3360\}$. As before, we plotted the average distance against the iteration number with the shading area representing the variance. Note that the log-scale of y-axis highlights the change for smaller values. Figure 9.3 demonstrates the improved convergence of Async-RED-SG to $\text{zer}(G)$ for larger $w$, which is consistent with our theoretical analysis. Figure 9.3 (middle) compares the convergence speed between Async-RED-BG/SG, gradient-method RED (GM-RED), and synchronous parallel RED (SYNC-RED). For Async-RED-SG, we use $w = 1120$. In particular, Async-RED-SG takes fewer total runtime (from 17.9 min to 13.0 min) to obtain the similar result (29.01 dB and 28.03 dB) and achieves $8.4\times$ speedup compared with GM-RED. The table in Figure 9.3 summarizes the detailed results.
Figure 9.4: CT reconstruction with a time budget of 1 hour by Async-RED-BG/SG and GM-RED. The colormap is adjusted for the best visual quality.

9.4.2 Effectiveness for Computational Imaging

We additionally demonstrate the effectiveness of our algorithm by reconstructing a $800 \times 800$ CT image from its 180 projections. For block parallel updates, the image is decomposed into 16 blocks, each having the size of $200 \times 200$ pixels. The Radon matrix used in the experiment corresponds to 180 angles with 1131 detectors, and the noise level is set to 70 dB. We refer to Appendix D.3.2 for additional technical details. Figure 9.4 shows the visual illustration of the reconstructed images by Async-RED-BG/SG and GM-RED. Each algorithm starts from the filtered back-projection (FBP) of the measurements and runs for 1 hour. Here, Async-RED-SG randomly uses one-third of the total measurements at every iteration. Given the same amount of time, Async-RED-BG/SG successfully mitigates the noise-artifacts, while the result of GM-RED is still noisy. In particular, the per-iteration time cost of Async-RED-BG/SG and GM-RED is 5.23, 3.21, and 19.19 seconds, respectively. This experiment clearly illustrates the fast processing speed of the asynchronous procedure.
9.5 Summary

Asynchronous parallel methods have gained increasing importance in optimization for solving large-scale imaging inverse problems. We have introduced Async-RED as an extension of the recent RED framework and theoretically analyze its convergence in batch and stochastic settings. We have validated its convergence guarantees and demonstrated its effectiveness in CT image reconstruction. We note that this work is complementary to traditional acceleration strategies, such as Nesterov acceleration and variance-reduction, commonly used in optimization. Future work will investigate Async-RED with Nesterov acceleration (as was done in [81] for traditional asynchronous block-coordinate algorithms) and variance-reduction (as was done in [96] for traditional stochastic gradient method) to better understand the tradeoffs between acceleration and scalability in multicore systems. We will additionally investigate theoretical limits of Async-RED in the unbounded maximal delay setting.
Part IV

Neural Fields Imaging
Chapter 10

Self-Supervised 3D Neural-Fields

Computational Microscopy

Neural Fields (NF), as reviewed in Section 2.3.2 in Chapter 2, is a self-supervised learning scheme that uses the forward model \( A \) to enable direct supervision from the measurements of the unknown object itself. In this chapter, we develop a novel NF-based image reconstruction algorithm for 3D image recovery in intensity diffraction tomography (IDT). IDT refers to a class of optical microscopy techniques for imaging the 3D refractive index (RI) distribution of a sample from a set of 2D intensity-only measurements. The reconstruction of artifact-free RI maps is a fundamental challenge in IDT due to the loss of phase information and the missing cone problem. We present DeCAF as the first NF-based IDT method that can learn a high-quality continuous representation of a RI volume directly from its intensity-only and limited-angle measurements. We show on three different IDT modalities and multiple biological samples that DeCAF can generate high-contrast and artifact-free RI maps. In summary, the contributions of this chapter include both algorithmic design and experimental validation on real data.
10.1 Introduction

Refractive index (RI) is an optical property that describes the interaction between light and matter within a sample. The real part of RI characterizes the phase velocity while its imaginary part characterizes the attenuation. RI can thus serve as an endogenous source of optical contrast for imaging samples without staining or labeling them using external contrast agents. By quantitatively characterizing the three-dimensional (3D) distribution of the RI, one can visualize cellular or subcellular structures useful in morphogenesis [113], oncology [253], cellular pathophysiology [112], biochemistry [53], and beyond (see the review papers [93, 171]).

IDT is a recent technique for producing 3D RI maps of a sample by measuring the light it diffracts. In the standard IDT setup, a sample is illuminated multiple times from different angles, and a set of two-dimensional (2D) intensity projections are captured by the camera (see Figure 10.1(a)). A tomographic image reconstruction algorithm is then used to computationally reconstruct the desired 3D RI distribution from the set of 2D measurements. Unlike traditional optical diffraction tomography (ODT) that uses interferometry to record the complex-valued light fields [104, 172, 221], IDT only measures the squared amplitude of the scattered light, leading to an easy setup on standard transmission optical microscopes with inexpensive hardware modifications. Such flexibility has spurred different IDT variants integrating object scanning [77, 90], angled illumination [45, 125, 131, 228], pupil engineering [161, 241], and multiple scattering [44, 49]. Setups achieving high resolution [49] and fast acquisition [124] have also been reported.

Despite the rich literature on IDT, image reconstruction remains a fundamental challenge. The first issue is that the phase of the scattered light field is missing from the measurements, resulting in a nonlinear measurement system that is not characterizable by the classical linear
Fourier diffraction theory [98]. This rules out the usage of the standard filtered-backprojection methods and calls for advanced computational algorithms. The second issue is the well-known \textit{missing cone} problem, causing elongation of the object along the optical axis (z dimension) and hence reduction of the axial resolution. The missing cone problem is due to the limited-angle tomographic setup, where illuminations can come only from one side of the sample plane with a limited range for angle variation (less than about 40° in our setups). This leads to an incomplete coverage of the 3D Fourier spectra with a cone-shape missing region in the axial direction. These missing phase and missing cone problems make image reconstruction in IDT a severely ill-posed \textit{inverse problem}.

Regularization methods are widely-used for mitigating ill-posed nature of many inverse problems. These methods are based on minimizing a cost function consisting of a data-fidelity term and a regularization term, where the former uses a physical-model to quantify the mismatch between the predicted and acquired measurements, while the latter promotes solutions that are consistent with \textit{a priori} knowledge on the sample. For example, the least-squares loss and Tikhonov regularizer ($\ell_2$-penalty) are widely-used for obtaining closed-form solutions to inverse problems [131]. The work on plug-and-play priors has generalized the notion of image priors to implicit regularizers characterized by image denoisers (see our discussion in Part II). Recently, deep learning (DL) has emerged as a powerful paradigm for image reconstruction. A traditional DL reconstruction is based on training a convolutional neural network (CNN) over a large dataset to learn a mapping from low-quality images to their high-quality counterparts. The state-of-the-art performance of such methods has been demonstrated in X-ray computed tomography [94, 108], magnetic resonance imaging [5, 274], optical tomography [126, 216], and seismic imaging [267] (see the reviews [1, 129, 238]). While DL has significantly improved image reconstruction in many modalities, traditional DL methods are impractical for image reconstruction in IDT where it is difficult to acquire
Figure 10.1: Conceptual illustration of DeCAF for IDT. (a) DeCAF reconstructs the RI volume by learning a neural field parameterized by a multilayer perceptron (MLP). The network is trained to map the 3D coordinate \((x, y, z)\) to the corresponding RI value by minimizing a loss that penalizes measurement mismatch and imposes regularization. (b) Our IDT system uses a programmable LED array to illuminate a sample from different angles and uses a digital camera to record the intensity measurements of the scattered light. By changing the illumination patterns, our system can implement different IDT modalities. (c) Our experiments consider three illumination patterns, that is, dense, annular, and multiplexed illuminations, each of which corresponds to a different formulation of the forward model. (d) DeCAF can reconstruct high-quality 3D RI maps from intensity-only and limited-angle measurements. (e) DeCAF learns a continuous representation and can render samples on a pixel grid of the desired density (illustration with 3.5×, 7.5×, and 31.8× upsampling).

High-quality ground-truth RI maps in experiments. Although a physics-based simulator has been proposed to generate datasets for training IDT artifact-suppressing CNNs, the results are still limited by the mismatch between the simulation and experiments [144].

Neural fields (NF) is a recent DL paradigm that has gained popularity in computer vision and graphics for representing and rendering 3D scenes using coordinate-based deep neural networks [204, 205]. It has been shown that NF can learn a high-quality representation of a
complex scene from a sparse set of data without any external training dataset. Motivated by this property, we propose **Deep Continuous Artifact-free RI Field (DeCAF)** as a first NF-based IDT method for learning a high-quality continuous 3D RI map from intensity-only and limited-angle measurements without any external training dataset of ground-truth RI maps. Figure 10.1 provides a conceptual illustration of DeCAF. The key features of DeCAF are as follows:

- The central component of DeCAF is a *multilayer perceptron (MLP)*, which is a fully-connected (non-convolutional) deep network, for learning a function that maps 3D coordinates \((x, y, z)\) to the corresponding RI values. The trained MLP thus provides a *continuous* neural representation of the RI map. The RI value at any spatial location can be retrieved by querying the trained MLP with the corresponding coordinate. By decoupling representation from an explicit voxel grid, DeCAF can efficiently store and process large 3D volumes.

- DeCAF is a *self-supervised* method, meaning that it does not require training using an external dataset of ground-truth RI maps. This is possible since the same MLP is used at every 3D location, enabling it to learn natural redundancies and correlations within a RI volume. The MLP is trained directly at test time by using only the IDT measurements of the sample that we seek to reconstruct. The IDT forward model is used to map the output of MLP to the intensity measurements and using the gradient back-propagation to update the MLP weights.

- DeCAF enables easy integration of additional prior knowledge on the unknown sample using an explicit regularization term in the loss function. In this chapter, we explored the potential of such synergistic integration by including a regularizer that relies on a deep denoising CNN pre-trained on *natural images* to remove additive white Gaussian noise (AWGN) [217, 220]. While our regularizer has not been trained explicitly on RI
values, we show through ablation studies that it improves the performance by mitigating noise and imaging artifacts.

## 10.2 Proposed Method

The pipeline of the proposed method is visually illustrated in Figure 10.1(c). In the training phase, the input of DeCAF is a set of spatial coordinates, \( c = \{(x_i, y_i, z_i)\}_{i=1}^n \), taken from a pre-defined grid. DeCAF first maps the input coordinates to encodings using a non-trainable radial expansion, followed by a standard fully connected neural network to map the encodings to the RI values at the input coordinates. We introduced a novel type of encoding called \textit{radial encoding}, which enables reconstruction of higher-quality RI maps (see details in Section 10.2.2). DeCAF is trained to solve the following optimization with an objective consisting of a measurement loss \( \mathcal{L} \) and regularizer \( \mathcal{R} \)

\[
\phi^* = \arg\min_{\phi} \{ \mathcal{L}(\mathcal{F}(x), y) + \mathcal{R}(x) \}
\]

such that \( x = \mathcal{M}_\phi(c) \),

\[\text{(10.1)}\]

where \( x \) is the predicted RI map, \( y \) is the intensity measurements of the test sample, \( \mathcal{F} \) is the IDT forward model, and \( \mathcal{M}_\phi \) is the MLP (which includes the radial encoding) parameterized by weights \( \phi \). Note that the test measurements are the only input required in DeCAF. After the optimal \( \phi^* \) is learned, one can render the test sample on a pixel grid with arbitrary density by simply querying \( \mathcal{M}_{\phi^*} \) using the corresponding coordinates, as illustrated in Figure 10.1(e).

Prior applications of NF include novel view synthesis [141, 151, 256], dynamic scene representation [127, 170, 173], object lightning [210, 244], and computed tomography [184]. Our work has several novel contributions to the existing NF literature: \( (i) \) DeCAF is the first
method that considers learning NF by accounting for the diffraction effects due to the wave nature of the light, while the existing work in the area has mostly focused on ray-tracing models in graphics. (ii) DeCAF is the first method that considers the recovery of the phase information from intensity-only data. We are not aware of any prior work that has considered NF for phase recovery. (iii) DeCAF is the first NF-based image reconstruction method that combines an implicit MLP regularization with an additional explicit image regularizer (for example, based on a deep denoiser) to achieve the best of both worlds, that is to improve over the separate usage of an implicit and explicit regularization. (iv) DeCAF introduces radial encoding as a novel type of encoding layer for improving the ability of NF to represent complex samples. In the next section, we present our results showing the ability of DeCAF to produce high-quality RI maps for various biological samples.

10.2.1 Forward Model

A linearized approximation of IDT forward measurement system can be described by equation (10.2)

\[ y_\rho \approx A_\rho \Delta \epsilon, \]

where \( \Delta \epsilon = \Delta \epsilon_{re} + j \Delta \epsilon_{im} \) is the unknown volume of complex-valued permittivity contrast, \( y_\rho \) is the collection of the background-removed intensity measurements corresponding to the LED illuminations emitted at a set of locations \( \rho \), and \( A_\rho \) is the measurement matrices that model the sample-intensity mapping associated with these illuminations. The reconstruction of \( \Delta \epsilon \) is equivalent to the reconstruction of the RI distribution via equation (10.3)

\[ n_{re} = \sqrt{\frac{1}{2}((n_0^2 + \Delta \epsilon_{re}) + \sqrt{(n_0^2 + \Delta \epsilon_{im})^2 + \Delta \epsilon_{im}^2})} \quad \text{and} \quad n_{im} = \frac{\Delta \epsilon_{im}}{2 \cdot n_r}, \]

146
where \( n_{re} \) and \( n_{im} \) are the real and imaginary parts of the sample’s RI, and \( n_0 \) is the RI of the background medium (where the attenuation is often assumed to be zero). In equation (10.3), all operations are evaluated in an element-wise manner. We derived the formulations of \( A_\phi \) by following the prior work on dIDT [131], aIDT [124], and mIDT [143], summarized in the Appendix E.1.

The central piece of DeCAF is a coordinate-based MLP, \( \mathcal{M}_\phi \), which maps the 3D coordinate \((x, y, z)\) to the corresponding values of \( \Delta\epsilon_{re} \) and \( \Delta\epsilon_{im} \). We normalize the coordinate grid to a cube \([-1,1]^3\) before feeding them into \( \mathcal{M}_\phi \). The deep network \( \mathcal{M}_\phi \) consists of two subnetworks, where the first one is an encoding layer \( \gamma(x, y, z) \), pre-defined before training, and the second one is a standard MLP \( \mathcal{N}_\phi : \gamma(x, y, z) \rightarrow (\Delta\epsilon_{re}, \Delta\epsilon_{im}) \) parameterized by the trainable parameters \( \phi \). A visual illustration of the detailed network architecture is provided in Figure 10.2(a).

Figure 10.2: Visual illustration of the network structure and the encoding strategy used in DeCAF. (a) The overall structure of network \( \mathcal{M}_\phi \). (b) Illustration of positional encoding for \( z \) coordinate. (c) Illustration of radial encoding for the coordinates in the \((x, z)\) plane.
10.2.2 Radial Encoding

It has been shown that a Fourier-type encoding of the spatial coordinates is essential for a MLP to represent high-frequency variations in the signal [151] and impose implicit regularization [218]. In DeCAF, we consider a decomposition of the input coordinate \((x, y, z)\) into \((x, y)\) and \(z\), and use different strategies to expand \((x, y)\) and \(z\). This is due to the non-isotropic resolution of the imaging system along the \(x-y\) plane and the \(z\) dimension. Our experiments showed that existing encoding strategies, such as positional [151] and Gaussian [223] encoding, lead to suboptimal reconstruction of RI images along the \(x-y\) dimensions. We propose radial encoding as an alternative for expanding \(v := (x, y)\)

\[
\gamma_{rad}(v) = \begin{pmatrix}
\sin(2^0 \pi R_{\theta} v), \cos(2^0 \pi R_{\theta} v), \\
\vdots \\
\sin(2^{L_{xy}-1} \pi R_{\theta} v), \cos(2^{L_{xy}-1} \pi R_{\theta} v)
\end{pmatrix}
\]

with
\[
R_{\theta} = \begin{bmatrix}
cos(\theta_k) & -\sin(\theta_k) \\
\sin(\theta_k) & \cos(\theta_k)
\end{bmatrix}
\]

\[
k = 1, \ldots, K
\]

(10.4)

Here, \(\sin\) and \(\cos\) compute the (element-wise) sinusoidal and cosinusoidal values, respectively, \(R_{\theta}\) denotes a collection of rotation matrices that translate the coordinate by the angles \(\theta\), and \(L_{xy} > 0\) controls the number of the expanded frequency. By incorporating rotation, our strategy enables a frequency expansion that can efficiently account for the dependencies within the \(x-y\) plane. The difference between radial encoding and positional encoding is conceptually illustrated in Figure 10.2(b) and Figure 10.2(c). In the experiments, we observed that the radial encoding improves the representation of small textures that are otherwise lost by other
Table 10.1: List of algorithmic hyperparameters

<table>
<thead>
<tr>
<th>Hyperparameters</th>
<th>Algae</th>
<th>Diatom</th>
<th>Cells</th>
<th>C.elegans</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modality type</td>
<td>IDT</td>
<td>aIDT</td>
<td>aIDT</td>
<td>mIDT</td>
</tr>
<tr>
<td>(</td>
<td>\theta</td>
<td>)</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>Number of radial encoding lines distributed evenly around 360°</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(L_{xy})</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Number of expanded frequencies for the (x-y) plane</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(L_z)</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Number of (z) expanded frequency</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(N)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Number of layers in MLP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(M)</td>
<td>208</td>
<td>208</td>
<td>208</td>
<td>208</td>
</tr>
<tr>
<td>Number of Hidden neurons</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skip layer</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Denoiser strength</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\alpha)</td>
<td>(1 \times 10^{-5})</td>
<td>(1 \times 10^{-6})</td>
<td>(3 \times 10^{-7})</td>
<td>(3 \times 10^{-6}) (head)</td>
</tr>
<tr>
<td>Regularization strength of (x-y) noise reduction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\beta)</td>
<td>(1 \times 10^{-6})</td>
<td>(3 \times 10^{-8})</td>
<td>(1.5 \times 10^{-7})</td>
<td>(3 \times 10^{-6})</td>
</tr>
<tr>
<td>Regularization strength of (z)-dimension continuity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Encodings. We adopted the standard positional encoding for the expansion of \(z\)

\[
\gamma_{pos}(z) = \begin{pmatrix}
\sin(2^0 \pi z), \cos(2^0 \pi z), \\
\vdots \\
\sin(2^{L_z-1} \pi z), \cos(2^{L_z-1} \pi z)
\end{pmatrix}, \quad (10.5)
\]

where \(L_z > 0\) denotes the total number of frequencies. We fine-tuned \(\theta\), \(L_{xy}\), and \(L_z\) for every sample for optimal results and summarize their values in Table 10.1. The ablation experiment on the challenging C. elegans specimen (see details in Figure E.4 in Appendix E.4) demonstrates the superior performance of the proposed encoding.
10.2.3 Network Details

**MLP architecture.** The network architecture of \( \mathcal{N}_\phi \) is illustrated in Figure 10.2(a). Network \( \mathcal{N}_\phi \) is composed of \( N \) fully-connected (FC) layers. The first \( N - 1 \) layers have \( M \) hidden neurons activated by the leaky rectified linear unit (Leaky ReLU), while the last layer has \( M \) unactivated hidden neurons. A skip connection is implemented at the \( \lfloor N/2 \rfloor \)th FC layer to concatenate the original input of \( \mathcal{N}_\phi \) with the intermediate outputs, which has been shown beneficial for improving the representation performance [169]. We used one network configuration for all tested samples, which is summarized in Table 10.1.

**Regularized loss function.** At test time, we train \( \mathcal{M}_\phi \) to minimize equation (10.6) by using a customized Adam [114] optimizer (see details in Appendix E)

\[
\mathcal{L}(\phi; y_\rho) = \| A_\rho(\mathcal{M}_\phi(c)) - y_\rho \|_1 + \alpha \| \mathcal{M}_\phi(c) - D_\sigma(\mathcal{M}_\phi(c)) \|_2^2 + \beta \sum_j \| \mathcal{M}_\phi^j(c) - \mathcal{M}_\phi^{j-1}(c) \|_1,
\]

where \( c = \{(x_i, y_i, z_i)\}_{i=1}^n \) is a collection of all coordinates on the grid, and \( \mathcal{M}_\phi^j \) denotes the \( j \)th axial slice of the predicted RI map. The loss defined in equation (10.6) can be divided into three terms serving different purposes, with \( \alpha \geq 0 \) and \( \beta \geq 0 \) balancing their contributions. The first term is a widely-used \( \ell_1 \)-norm loss that ensures the consistency with the test measurements. The second and third terms are the regularizers imposing \( x-y \) plane noise reduction and continuity along \( z \), respectively. \( D_\sigma \) denotes a 2D image denoiser with \( \sigma > 0 \) controlling the denoising strength. We selected DnCNN as our denoiser due to its state-of-the-art denoising performance [262]. A detailed description of the architecture and training of DnCNN is presented in the Appendix E.3. We fine-tuned \( \alpha \) and \( \beta \) using standard grid search for each sample and summarized their values in Table 10.1. In the Appendix E.4,
we provide the empirical evidence to demonstrate the necessity of the explicit regularization for imaging the complex organism (see visual results in Figure E.6).

10.3 Experiment

We validated the ability of DeCAF on experimentally collected IDT data to recover high-quality RI maps with accurate biological characteristics and minimal artifacts. We used DeCAF on four biological samples: spirogyra, diatom, human buccal epithelial cells, and Caenorhabditis elegans. We adopted the existing light-propagation models to formulate the inverse problems associated with dense [131], annular [124], and multiplexed [143] illumination patterns. Since absorption provides lower contrast for the considered samples, we focus on comparing the phase images. In the subsequent sections, we use $x$, $y$, and $z$ to denote length, width, and depth, respectively.

10.3.1 Setup

IDT Experiments

Dense IDT. Our dense IDT system consists of a Nikon TE 2000-U microscope equipped with a custom programmable LED array, approximately illuminating plane wave with central wavelength $\lambda = 632$ nm; a 10×/0.25 NA objective (Nikon, CFI Plan Achromat), and an sCMOS camera (PCO.Edge 5.5). The LED array is placed about 79 mm away from the sample. It is controlled via a microcontroller and is synchronized with the camera. A small subset of the LEDs on the array, which contains the 89 LEDs within the brightfield region, is used to illuminate the sample sequentially.
**Annular IDT.** Our annular IDT system consists of a Nikon ECLIPSE E200 microscope equipped with a programmable ring LED unit (Adafruit, 1586 NeoPixel Ring). The microscope objective is 40×/0.65 NA (Nikon, CFI Plan Achromat), and each LED approximately provides a plane wave with central wavelength $\lambda = 515$ nm. The ring LED unit has 24 LED lights and is 60 mm in diameter. It is centered at the optical axis and placed approximately 35 mm away from the sample, which sets the angle between the wave vector and the optical axis to about 40° and complies with the microscope objective NA.

**Multiplexed IDT.** Our multiplexed IDT system has the same hardware specification as the dense IDT system except that the microscope objective is 40×/0.65 NA (Nikon, CFI Plan Achromat). Besides, the subset of the LEDs used in the experiment changes to 96 LEDs corresponding to the NA range from 0.3 to 0.575. This design contains 16 disjoint illumination patterns and the multiplexed illumination quantity of each pattern is 6. The camera is synchronized with the LED array and captures 16 measurements corresponding to each illumination pattern.

**Sample and data preparation**

**Spirogyra Algae.** This sample is a part of Fisher Science Education algae basic slide set S68786. We captured 89 intensity-only bright field measurements. We pre-processed each measurement by removing the background intensity followed by normalization. The same pre-processing procedure is also applied to other samples. We consider a reconstruction volume of $665.6 \times 665.6 \times 80 \mu m^3$, positioned between $-30 \mu m$ and $50 \mu m$ around the focal plane. The volume is discretized into 40 slices along the $z$ axis, with each slice having $1024 \times 1024$ pixels. Here, a single voxel corresponds to $6.5 \times 6.5 \times 2 \mu m^3$. 

152
**Diatom Algae.** This sample is a part of Fisher Science Education algae basic slide set S68786. We captured 24 measurements and consider a reconstruction volume of $113.75 \times 113.75 \times 26 \, \mu m^3$, positioned between $-10 \, \mu m$ and $16 \, \mu m$ around the focal plane. The volume is discretized into 52 slices along the $z$ axis, with each slice having $700 \times 700$ pixels. Here, a single voxel corresponds to $0.1625 \times 0.1625 \times 0.5 \, \mu m^3$.

**Human Buccal Epithelial Cells.** This sample was swabbed from a researcher’s buccal. The individual rinsed the mouth with clean water and then twirled a wooden swab against the inner cheek. The end of the swab was immersed in a drop of purified water on a glass slide and covered by a coverslip. We captured 24 measurements of the cell cluster and consider two volumes in the region as shown in Figure 10.5(b) and 10.5(c). The former has $81.25 \times 81.25 \times 16 \, \mu m^3$ and the latter has $97.5 \times 97.5 \times 16 \, \mu m^3$. Both volumes are positioned between $-8 \, \mu m$ and $8 \, \mu m$ around the focal plane. They are discretized to 32 slices of $500 \times 500$ pixels and $600 \times 600$ pixels. Here, A single voxel corresponds to $0.1625 \times 0.1625 \times 0.5 \, \mu m^3$.

**Caenorhabditis Elegans.** Young adult C. elegans were mounted on 3% agarose pads in a drop of nematode growth medium (NGM) buffer. Glass coverslips were then gently placed on top of the pads and sealed with a 1:1 mixture of paraffin and petroleum jelly. As the C. elegans was alive and moving during data acquisition, we captured a video at 4 frames per second, where each frame contained 16 measurements, and each measurement used 6 LED lights. We picked two frames at $1.5s$ and $44s$ for reconstruction, where the sample was relatively steady. We consider a unified reconstruction volume of $162.5 \times 162.5 \times 20 \, \mu m^3$, positioned between $-10 \, \mu m$ and $10 \, \mu m$ around the focal plane. The volume is divided into 40 slices along the $z$ axis, with each slice having $1000 \times 1000$ pixels. Here, a single voxel corresponds to $0.1625 \times 0.1625 \times 0.5 \, \mu m^3$. 
10.3.2 Results

We first show the effectiveness of DeCAF for dense IDT (dIDT) on stained spirogyra (Fisher Scientific S68786), which is unicellular algae containing helical arrangement of chloroplasts oriented in the 3D space. We immersed the sample in water (RI ≈ 1.33) and collected in-total 89 brightfield intensity measurements by using a 0.25 NA objective. Two measurement examples are presented in Figure 10.1(b) (see images a-b). In the experiment, we compared DeCAF against two existing IDT reconstruction baselines: Tikhonov regularization (Tikhonov) [131] and SIMBA [249]. SIMBA is a recently proposed model-based algorithm that leverages a deep learning denoiser as an image prior. The final reconstructed RI volume by each method consists of 40 slices of 1024 × 1024 pixels equally spaced between -30 µm and 50 µm, forming a volume of 665.6 × 665.6 × 80 µm³. Throughout the chapter, we define \( z = 0 \) µm as the focal plane.

Figure 10.3 visualizes the experimental results. To demonstrate the overall structure of the sample, a rendered 2D image accumulating all \( z \) layers of the DeCAF reconstruction is presented in Figure 10.3(a). As shown, DeCAF successfully reconstructed the spiral structure of the spirogyra. Figure 10.3(b) compares the 2D axial slices obtained by DeCAF, SIMBA, and Tikhonov at the depths \( z \in \{4, 16, 28, 40\} \) µm. The images associated with DeCAF are visualized in the first row, and the results of SIMBA and Tikhonov regularization are visualized in the second and third rows, respectively. The results show that DeCAF provides superior sectioning ability—meaning that a pattern emerges only in the slices it belongs to and fades away as we go axially to different depths—than other two methods. For example, note the clarity and sharpness of the spirals (highlighted in the cyan box) that appear at a specific depth, showing that DeCAF removes the artifacts (highlighted in the white box) generated by the content in adjacent slices. Note also how these artifacts remain in the reconstructions.
Figure 10.3: Reconstruction of Spirogyra Algae acquired by dIDT. (a) 2D rendering obtained by accumulating all the z slices from DeCAF. Scale bar 65 µm. (b) Visual comparison of the axial views at $z \in \{4, 16, 28, 40\} \mu$m reconstructed using three methods: DeCAF, SIMBA, and Tikhonov. We adjusted the RI range of each method for the best visual quality with the corresponding colormaps shown on the right. (c) & (d) Axial views corresponding to the colored lines in (a). Each plot uses the same colormap associated with the method presented in (b). Scale bar 40 µm. (e) Line profiles from the dashed lines in (c) and (d). The label at the upper right of each plot indicates the corresponding dashed line. This figure illustrates the ability of DeCAF to reconstruct high-contrast RI maps by also significantly reducing the missing-cone artifacts.

by SIMBA and Tikhonov. We further evaluate the axial resolution of each reconstruction by comparing the lateral views corresponding to the paths shown in Figure 10.3(c) and 10.3(d). DeCAF significantly reduces the elongation artifacts caused by the missing-cone problem. Line profiles presented in Figure 10.3(e) visually characterize the improvement of the axial resolution brought by DeCAF.
Figure 10.4: Reconstruction of Diatom Algae acquired by aIDT. (a) 3D illustration of the DeCAF reconstruction showing the overall structure of the sample. (b) Axial view at $z = 0 \, \mu m$ (focal plane) reconstructed by DeCAF. Scale bar 10 \( \mu m \). (c) & (d) $y$-$z$ and $x$-$z$ lateral views corresponding to the colored paths in (b). The results of Tikhonov are also presented for reference. Scale bar 10 \( \mu m \). (e) Visual illustrations of the axial views at $z \in \{-1.0, 16, 28, 40\} \, \mu m$, highlighting better removal of artifacts compared to Tikhonov. (f) Visual demonstration of DeCAF’s ability to perform continuous RI upsampling along the $x$ and $y$ dimensions. DeCAF’s results are consistent with that of the classic interpolation methods but provide finer details highlighted by the arrows.

We next applied DeCAF to annular IDT (aIDT) to explore its capability for efficient data processing. We imaged two distinct classes of biological samples, that is, diatom microalgae (S68786, Fisher Scientific) and unstained human epithelial buccal cells. The former is a unicellular algae with regular arrangement of punctae, while the latter is a complex cell environment consisting of intracellular bacteria. We acquired 24 intensity images using 0.65 NA objective under oblique illuminations for each sample. During acquisition, the diatom and cell cluster samples are fixed in glycerin gelatin (RI $\approx 1.47$) and water, respectively. We included Tikhonov as the baseline method for comparison.
Figure 10.5: Reconstruction of Human Buccal Epithelial Cell Cluster acquired by aIDT.  
(a) Example intensity measurement collected by our aIDT setup for the cell cluster. Note that the background light is removed from the image. Scale bar 13 µm. (b) & (c) x-z and y-z lateral views of the DeCAF reconstruction associated with the paths in (a). Superior performance in artifact removal and axial separation is demonstrated over the Tikhonov regularization. Scale bar 10 µm.  
(d), (e), & (f) The axial views at various depth of the two sub-cell clusters shown in (b) and (c). These results further highlight the strong axial sectioning effects as well as the fine details preserved by DeCAF.  
(h) & (i) Visual demonstration of DeCAF’s ability to perform continuous RI upsampling along the z dimension. Smooth and consistent transition in the appearance of bacteria is observed.

Figure 10.4 presents the results for diatom algae. Two measurement examples are provided in Figure 10.1(b) (see images c-d). Both DeCAF and Tikhonov were configured to reconstruct 52 slices of 700 × 700 pixels equally spaced between -10µm to 16µm, forming a volume of 113.75 × 113.75 × 26 µm³. The 3D illustration of the volume reconstructed by DeCAF is presented in Figure 10.4(a), demonstrating the overall reconstruction quality. Figure 10.4(c) presents the slices reconstructed by each method at three depths z ∈ {-1.0, 1.0, 3.0} µm. DeCAF demonstrates better sectioning effect than Tikhonov regularization. Superior removal
Figure 10.6: Reconstruction of *C. Elegans* acquired by mIDT. (a) & (b) The reconstructed RI distribution at $z = 0 \mu m$ (focal plane) by DeCAF. (c), (d), & (e) Lateral views corresponding to the paths shown in (a) and (b). Biological structures are highlighted by the arrows and circles. (f), (g), & (h) Axial views of the regions highlighted in (a) and (b) at $z \in \{-1, 1, 3\} \mu m$. Note how DeCAF provides higher contrast and finer details than Tikhonov.
of the missing-cone artifacts is also shown in the lateral views in Figure 10.4(c) and 10.4(d). Because DeCAF learns a continuous representation of the RI distribution, it can generate images on arbitrarily dense voxel grids without additional retraining. Figure 10.4(f) demonstrates this unique ability of DeCAF by interpolating $26.7 \times$ more pixels in the $x$-$y$ planar region shown in Figure 10.4(b). For comparison, we used nearest neighbor ($\text{Pixel}$), bilinear ($\text{Bilinear}$), and bicubic ($\text{Bicubic}$) interpolation methods to upsample the same region. Our results show that DeCAF is able to resolve small features with strong cross-scale consistency while avoiding interpolation artifacts highlighted by the arrows.

We next present the results of epithelial buccal cell clusters in Figure 10.5. A background-removed (b.r.) intensity measurement showing the distribution of the whole cell cluster is presented in Figure 10.5(a). We focus on two difficult regions where cells overlap with each other to highlight the sectioning ability of DeCAF in depth in Figure 10.5(b) and 10.5(c). The size of the two volumes are $81.25 \times 81.25 \times 16 \mu m^3$ and $97.5 \times 97.5 \times 16 \mu m^3$, discretized to 32 slices of $500 \times 500$ pixels and $600 \times 600$, respectively. DeCAF successfully resolves different cells with clear separation while Tikhonov generates strong artifacts that blur the boundaries. Visual demonstration of the axial slices of these cells are provided in Figure 10.5(d)-10.5(f). In each reconstructed slice, DeCAF recovers clear cell membrane, cytoplasm, micronuclei, and bacteria while removing the diffraction artifacts.

We further show the continuous representation learned by DeCAF by upsampling it along $z$, meaning that DeCAF is used to interpolate an entire axial slice that was not part of the grid used during training. Figure 10.5(h) and 10.5(i) present the interpolated slices of the bacteria clusters highlighted in Figure 10.5(d) and 10.5(e), respectively. In each figure, a $z$ axis is provided to show the axial location of each slice. Note that $\{-5.5, -4.5, -3.5\} \mu m$ and $\{1.5, 2.5, 3.5\} \mu m$ are the axial points pre-defined in the training grid. The interpolated slices in Figure 10.5(h) and 10.5(i) clearly show the appearance and disappearance of the
Table 10.2: Quantitative Illustration of the scalability of DeCAF due to its off-the-grid feature using the C. elegans specimen as an example. Note that the space required for DeCAF is independent of the reconstruction grid.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Training</th>
<th>Inference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Grid size)</td>
<td>1000 × 1000 × 40</td>
</tr>
<tr>
<td>DeCAF</td>
<td>3MB</td>
<td>3MB</td>
</tr>
<tr>
<td>Tikhonov</td>
<td>-</td>
<td>3.73GB</td>
</tr>
</tbody>
</table>

bacteria clusters for different values of z. As shown, the interpolated biological features are consistent with the ones lying in the pre-defined grid, making the whole transition smooth across axial layers. This strong axial consistency preserved in DeCAF enable it to produce high-fidelity interpolations without any additional re-training.

We finally validate DeCAF on the multiplexed IDT microscopy (mIDT). This modality allows more measurements under the same acquisition time by simultaneously illuminating the sample from multiple angles. We imaged a Caenorhabditis elegans (C. elegans) worm specimen by using a 0.65 NA objective to acquire 16 measurements, with each from the simultaneous illuminations of 6 different LED sources. Figure 10.1(b) (see images e-f) shows two measurement examples. Note that the sample is very challenging due to its thickness and complicated arrangement of organs. As the worm is alive and moving during the acquisition, we reconstructed two volumes of 162.5 × 162.5 × 20 µm³, discretized to 40 slices of 1000 × 1000 pixels, at different time to cover the worm body with interested biological features.

Figure 10.6(a) and 10.6(b) present the reconstructed RI maps of C. elegans at the focal plane. DeCAF successfully recovers the sample’s structure with clear quantification of the internal biological tissues. For example, the buccal cavity, anterior and terminal pharyngeal bulbs, isthmus, and intestine are clearly restored in our reconstruction. Smaller features are also distinguishable with a high contrast, shown in the regions expanded in Figure 10.6(f)-10.6(h). For example, lysosomes, grinder, and lumen of intestine are accurately visualized with
clear separation from the other tissues. Figure 10.6(c)-10.6(e) show the $y$-$z$ lateral views. In all figures, the oval shape of C. elegans is clearly reconstructed without missing-cone artifacts, and fine features such as buccal cavity, grinder, and lumen are preserved and reconstructed at different axial layers. DeCAF successfully recovers the ‘Y’ shape of the grinder in Figure 10.6(d), agreeing with the existing knowledge from biology.

Table 10.2 highlights the memory used for storing the neural representation in DeCAF. Since the representation is decoupled from a predefined voxel grid, DeCAF can be trained on a sparser grid to reduce the memory cost, but can still produce the final reconstruction on a grid of desired density. The memory reduction is demonstrated by comparing the memory requirements of DeCAF and Tikhonov for the reconstruction of the C. elegans worm. DeCAF retains the small memory size of 3MB across different grid densities while that of Tikhonov increases as the grid becomes denser.

### 10.4 Summary

We proposed a novel self-supervised deep learning method, DeCAF, for enabling high-quality 3D reconstruction of the RI distribution from intensity-only measurements. We extensively validated DeCAF on the experimentally collected datasets of multiple biological samples under three different IDT setups. The results show that DeCAF can mitigate the missing-cone artifacts while maintaining the fine details of small biological features. The continuous representation in DeCAF also allows to generate images at voxel grids of arbitrary density without retraining of the deep network, which is useful for addressing computational and memory bottlenecks in image reconstruction and analysis.
Chapter 11

Neural Interpolator for Measurement Completion via Coordinate-Based Internal Learning

COORDINATE-BASED INTERNAL LEARNING (CoIL) represents an alternative approach to leverage NF for computational imaging. Unlike DeCAF that learns a representation of the object, CoIL functions in the measurement domain, aiming at learning a neural representation of the full continuous measurement field rather than the object by exploiting the internal information within the subsampled and noisy measurements. The learned full measurement field acts just like the ordinary measurements and can be immediately used as extra source of information in a majority of image reconstruction methods. We validate CoIL on sparse-view computed tomography using several widely-used reconstruction methods, including purely model-based methods and those based on deep learning. Our
results demonstrate the ability of CoIL to consistently improve the performance of all the considered methods by providing high-fidelity measurement fields.

This chapter is organized as follows. Section 11.1 presents an introduction to CoIL and summarizes the contributions. Section 11.2 introduces the technical details of the proposed framework and discusses integration of CoIL into several widely-used image reconstruction methods. Section 11.3 presents the numerical validation of CoIL on sparse-view CT. Section 11.4 concludes the chapter by discussing potential applications and limitations of CoIL.

### 11.1 Introduction

The core of CoIL is a multilayer perceptron (MLP) that maps the measurement coordinates to the corresponding sensor responses. The measurement coordinates are the parameters corresponding to the geometry of the imaging system that determine the response measured by the sensors. For example, in computed tomography (CT) two parameters characterizing the response are the view angle $\theta$ of the incoming ray beam and the spatial location $l$ of the relevant detector on the sensor plane. By training MLP on the coordinate-response pairs extracted from the measurements of a desired object, CoIL is able to build a continuous mapping from the coordinates to the sensor responses. Thus, the learned MLP corresponds to a neural representation of the full measurement field. By querying the MLP with the relevant coordinates, CoIL can generate the full field that can be used for image reconstruction. Figure 11.1 provides a conceptual illustration of the CoIL methodology. Note that CoIL is not restricted to a specific image reconstruction method, but is compatible with a majority of methods including those based on model-based optimization or DL.

The main contributions of this chapter are as follows:
Figure 11.1: The conceptual illustration of CoIL in the context of sparse-view CT. A multilayer perceptron (MLP) is used to represent the full measurement field by learning to map the measurement coordinate $(\theta, l)$ to its response $r$. Visual examples compare the recovered images with and without CoIL for total variation (TV). CoIL is used to generate 360 views from the data consisting of 120 noisy views of 40 dB input SNR. The quantitative and visual results in this chapter highlight the ability of CoIL to significantly improve the imaging quality for several widely-used image reconstruction methods.

- We propose CoIL as a novel imaging methodology that leverages coordinate-based neural representations for estimating high-fidelity measurement fields [141, 151, 264]. Unlike other recent coordinate-based methods that focus on representing the unknown object, CoIL represents the measurement field, which can be subsequently combined with other information sources during reconstruction.

- We propose a new MLP architecture that uses a linear mapping strategy for the input coordinates of the network. We observe that this input mapping strategy is effective for representing measurement fields and achieves better performance compared to positional encoding [151] and random sampling [223].
• We extensively validate our method in the context of sparse-view CT. We show that CoIL synergistically combines with a majority of widely-used methods by being able to generate high-fidelity full-view sinograms. In all our experiments the methods with CoIL consistently outperform the ones without it.

11.2 Coordinate-based Internal Learning

In this section, we present the details of the CoIL methodology that leverages the power of coordinate-based learning for addressing imaging inverse problems. Figure 11.2 illustrates the general workflow of CoIL for a given imaging system. We first explain the proposed MLP network and then discuss its integration into several common image reconstruction methods.

11.2.1 Measurement-field encoding with MLP

The coordinate-based MLP is the central component of CoIL. The network can be expressed as

$$\mathcal{M}_\phi : \mathbf{v} \rightarrow r \quad \text{with} \quad \mathbf{v} \in \mathbb{R}^v, r \in \mathbb{R},$$

where $\mathbf{v}$ represents the coordinate in the given imaging system, and $r$ is the corresponding sensor response. The network can be conceptually separated into two parts, where the first part is a single Fourier feature mapping (FFM) layer $\gamma(\mathbf{v})$ that is pre-defined before training, while the second part is a standard MLP $\mathcal{N}_\phi : \gamma(\mathbf{v}) \rightarrow r$ whose parameters $\phi$ needs to be optimized. A visual illustration of the complete network architecture is provided in Figure 11.3. While the numerical studies presented in this chapter focus on CT, CoIL is also applicable to other imaging modalities by simply changing the coordinate-response pairs in the MLP representation. For example, one can potentially integrate CoIL into optical diffraction tomography (ODT) [106, 178, 221] by letting $\mathbf{v}$ denote the sensor location and
Figure 11.2: Illustration of the CoIL workflow for a tomographic imaging system with free parameters \( \mathbf{v} \in \mathbb{R}^v \). First, a set of \( N > 0 \) measurements are acquired by the system under different realization of \( \mathbf{v} \). Then, the coordinate-response pairs \( \{(\mathbf{v}_i, r_i)\}_{i=1}^N \) are used to train a coordinate-based MLP \( \mathcal{M}_\phi : \mathbf{v} \rightarrow r \) for encoding the full measurement field. Once the training is finished, the encoded field is extracted from \( \mathcal{M}_\phi \) with an arbitrary resolution by querying the relevant coordinates. In the final stage, the CoIL field and the actual measurements are jointly used for image reconstruction using a user-defined method.

the angle of the incident light and letting \( r \) have two elements corresponding to the real and imaginary components of the light-field.

**Fourier feature mapping**

Although neural networks are known to be universal function approximators [86], it has been found that standard MLPs perform poorly in representing high-frequency variations [151, 179]. In our experiments, we also experienced such issues when we directly applied \( \mathcal{N}_\phi \) to learning the mapping \( \mathbf{v} \rightarrow r \) (see *No FFM* in Figure 11.5). In order to overcome the limitations of standard MLPs, we include the FFM layer to expand the input coordinate \( \mathbf{v} \) as a combination
Figure 11.3: Visualization of the coordinate-based MLP used in the CoIL methodology. The network $\mathcal{M}_\phi = \mathcal{N}_\phi \circ \gamma(v)$ is a concatenation of a single Fourier feature mapping (FFM) layer $\gamma(v)$ and a conventional MLP $\mathcal{N}_\phi$. As training on example pairs $\{(v_i, r_i)\}_{i=1}^N$, $\mathcal{M}_\phi$ is able to learn a continuous mapping from a coordinate to its response. Hence, $\mathcal{M}_\phi$ becomes an implicit neural representation of the full-measurement field.

of different frequency components

$$
\gamma(v) = \begin{pmatrix}
\sin(k_1\pi v), \cos(k_1\pi v), \\
\vdots \\
\sin(k_L\pi v), \cos(k_L\pi v)
\end{pmatrix},
$$

(11.1)

where $\sin$ and $\cos$ compute element-wise sinusoidal and cosinusoidal values, respectively, and $\{k_i\}_{i=1}^L$ determine the frequencies in the mapping. The FFM layer pre-defines the frequency components so that the network $\mathcal{N}_\phi$ can actively select the ones that are the most useful for encoding sensor responses by learning the weights in the first layer. By manipulating the coefficients $k_i$ and total number of components $L > 0$, we can explicitly control the expanded spectrum and thus impose some implicit regularization. The FFM layer was first introduced in NeRF as positional encoding of spatial coordinates [151], and a follow-up work [223] has further explored its functionality by using a concept known as neural tangent kernels [88]. The original formulation of $\gamma(v)$ in [151] sets $k_i$ as an exponential function $k_i = 2^{i-1}$ with $L = 10$. We discovered that the presence of very high-frequency components leads to the overfitting of the MLP to the noise in the measurements. We thus adopted a linear sampling $k_i = \pi i/2$ in the Fourier space that leads to a higher number of frequency components.
in the low-frequency regions. Our empirical results show that our strategy can effectively improve $M_\phi$ in representing high-frequency variations and preventing overfitting to noise (see Figure 11.5 for examples). We also compare our linear-expansion strategy with the recent random-projection mapping \[223\]

$$
\gamma(v) = (\cos(2\pi Bv), \sin(2\pi Bv)),
$$

(11.2)

where $B \in \mathbb{R}^{d\times v}$ is a random matrix with each entry sampled from i.i.d. Gaussian distribution $\mathcal{N}(0, \sigma^2)$. Numerical evaluations in Section 11.3.2 show that our formulation achieves better performance for higher noise level and fewer projections.

**MLP Architecture**

The network implementing $N_\phi$ is composed of 17 fully-connected (FC) layers. The first 16 layers have 256 hidden neurons whose outputs are activated by the rectified linear unit (ReLU), while the last layer has 128 hidden neurons without any activation. We implement 7 skip connections after every even-numbered (less than 16) FC layer to concatenate the original input of $N_\phi$ with the intermediate outputs. The use of the skip connections has been shown to be beneficial for fast training \[47\] and better accuracy \[169\]. Note that although $M_\phi$ is a fully connected network, its input corresponds to a single coordinate, which enables element-wise processing of all the measurements. Additional evaluations of several other architectures that can be used as alternatives to CoIL are provided in the Appendix F.

**Additional implementation details**

CoIL trains a separate MLP to represent the full measurement field for each test objects. This means that the training pairs $\{(v_i, r_i)\}_{i=1}^N$ are obtained by extracting the measurements
8 Figure 11.4: Eight $512 \times 512$ images from the scans of two patients in the AAPM human phantom dataset [146] were used for testing.

of the test object only, without any training dataset. The network $M_\phi$ is trained by using Adam [114] to minimize the standard $\ell_2$-norm loss

$$\ell(\psi) = \frac{1}{N} \sum_{i=1}^{N} \|M_\phi(v_i) - r_i\|_2^2.$$  \hspace{1cm} (11.3)

We implement a decreasing learning rate, which decays exponentially as the training epoch increases, to smooth our optimization. Although $M_\phi$ is a MLP, the network has a significantly smaller size ($\approx 4.2$ MB on disk) compared to the standard UNet architecture ($\approx 108$ MB on disk) used in many DL-based models.

\subsection*{11.2.2 Image reconstruction in CoIL}

After training, one can generate an arbitrary number of measurements by querying $M_\phi$ using the relevant coordinates. We will refer to the corresponding measurement field as \textit{CoIL field}. The CoIL field can be readily integrated into the majority of image reconstruction methods. Here, we discuss the integration of CoIL into four widely-used methods.
Figure 11.5: Illustration of the benefit of including the Fourier feature mapping (FFM) layer into CoIL. We plot sinograms and their FBP reconstructions in the first and second row, respectively. The proposed FFM in CoIL is compared against No FFM strategy (which does not have any FFM layer), positional encoding (Pos Enc) [151], and random Gaussian sampling (Random) [223]. The four MLPs are used to generate 360 views from the $P = 120$ projections with $I = 40$ dB noise. Both sinograms and images are labeled with the SNR values with respect to the ground truth shown in the right-most column. The bounding boxes highlight areas of significant visual difference. This comparison shows the benefit of using the FFM layer with linear spacing in the Fourier space.

**Linear reconstruction**

Filtered backprojection (FBP) is a classic method for bringing the measurements into the image domain [98]. Since the CoIL field is essentially a set of measurements, we can directly feed the field as input to FBP for image reconstruction. A slightly different way to apply FBP is to form a combined input that includes both the original measurements and those generated by CoIL. The key benefit of the latter approach is that it directly uses the real data while also complementing it with CoIL measurements.
Model-based optimization

Model-based methods reconstruct images by solving optimization problems of form (2.11).

\[
\hat{x} = \arg \min_x f(x), \quad \text{where} \quad f(x) = g(x) + r(x),
\]

(11.4)

The CoIL field can be incorporated into the formulation by including an additional “data-fidelity” term \( \tilde{g} \) in the objective

\[
f(x) = (1 - \alpha)g(x) + \alpha \tilde{g}(x) + r(x).
\]

(11.5)

Here, the parameter \( 0 \leq \alpha \leq 1 \) controls the tradeoff between the real data and the generated field. In practice, we can fine-tune the value of \( \alpha \) to obtain a good balance between two terms. For example, consider the least-squares function

\[
\tilde{g}(x) = \frac{1}{2} \| \tilde{A}x - M_\phi(\tilde{v}) \|_2^2,
\]

(11.6)

where \( \tilde{A} \in \mathbb{R}^{m \times n} \) corresponds to the sampling geometry of the CoIL field, \( \tilde{v} \) represents all the query coordinates for the trained MLP\( M_\phi(\tilde{v}) \). Since the network is pre-trained, one can directly use any existing image regularizer and solve the optimization problem with a standard iterative algorithm, such as PGM or ADMM.

End-to-end DL models

As reviewed in Section 2.2.2, most end-to-end DL models are trained to directly map the low-quality images \( \{\tilde{x}_i\}_{i=1}^N \) to the high-quality images \( \{x_i\}_{i=1}^N \), making them vulnerable to unseen outliers. For example, this adversely influences the performance of DL, when there
Table 11.1: The average SNR of the sinograms generated by No FFM, Pos Enc, Gaussian and CoIL in the scenarios corresponding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$

<table>
<thead>
<tr>
<th># Views ($P$)</th>
<th>Noise Level ($I$)</th>
<th>MLP Architectures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No FFM</td>
<td>Pos Enc</td>
</tr>
<tr>
<td>60</td>
<td>30</td>
<td>33.95</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>42.62</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>46.33</td>
</tr>
<tr>
<td>90</td>
<td>30</td>
<td>34.93</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>43.82</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>48.08</td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>36.24</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>44.81</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>49.68</td>
</tr>
</tbody>
</table>

is a mismatch between training and testing angles. CoIL can be used to address this issue by generating the measurement field corresponding to the same subsampling rate as the measurements used for training the DL model

$$\hat{x} = \mathcal{F}_\psi(FBP(\mathcal{M}_\phi(\tilde{v}))),$$  \hspace{1cm} (11.7)

where $\mathcal{F}_\psi$ denotes the pre-trained CNN. Alternatively, one can include the original test image in the input by averaging the $\hat{x}$ and $FBP(\mathcal{M}_\phi(\tilde{v}))$ using a weight $\alpha$

$$\hat{x} = \mathcal{F}_\psi((1 - \alpha)\hat{x} + \alpha FBP(\mathcal{M}_\phi(\tilde{v}))).$$  \hspace{1cm} (11.8)

This approach enables the usage of the learned measurements by MLP with the true measurements from the imaging system. Our results in Section 11.3 show that this CoIL-based strategy achieves better results than training a DL model directly on the measurements.
Figure 11.6: Quantitative evaluation of the CoIL field for different projection numbers \((P)\) and noise levels \((I)\). The plot is divided into three regions, corresponding to \(P\) equal to 60, 90, and 120, respectively. Within each region, the average SNR values of the generated sinograms are plotted against different input SNR values, which are also drawn by the dotted horizontal lines for better visualization. First, note how CoIL generally produces measurement fields of better SNR than the noise level in the measurements. Second, the figure highlights that the quality of the generated CoIL field improves as the number of views increases or the noise level decreases.

Denoising-driven approches

PnP/RED algorithms can be interpreted as extensions of model-based algorithms balancing consistency with the measurements against deep denoising priors [193, 213]. Consider gradient-based RED (GM-RED)

\[
x^+ \leftarrow x - \gamma \left[ \nabla g(x) + \tau (x - D_\sigma(x)) \right]
\]

(11.9)

where \(\gamma > 0\) is the stepsize, and \(\nabla g\) is the gradient of the data-fidelity term. Similar to the modification of model-based optimization, one straightforward way to integrate CoIL into
Table 11.2: The average SNR of the sinograms generated by Linear, Bicubic, ConvNet \[122\], and CoIL in the scenarios corresponding to \( P \times I = \{60, 90, 120\} \times \{30, 40, 50\}\)

<table>
<thead>
<tr>
<th># Views (P)</th>
<th>Noise Level (I)</th>
<th>Interpolation methods</th>
<th>CoIL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Linear</td>
<td>Bicubic</td>
</tr>
<tr>
<td>60</td>
<td>30</td>
<td>30.37</td>
<td>29.51</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>35.21</td>
<td>34.82</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>36.28</td>
<td>36.10</td>
</tr>
<tr>
<td>90</td>
<td>30</td>
<td>31.18</td>
<td>30.16</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>37.95</td>
<td>37.36</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>40.14</td>
<td>39.94</td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>31.23</td>
<td>30.42</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>39.36</td>
<td>38.80</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>43.03</td>
<td>42.86</td>
</tr>
</tbody>
</table>

GM-RED is to include the gradient of \( \tilde{g} \) as an extra term

\[
x^+ \leftarrow x - \gamma \left[ (1 - \alpha) \nabla g(x) + \alpha \nabla \tilde{g}(x) + \tau(x - D_\sigma(x)) \right],
\]

where the new update ensures the consistency with the real measurements as well as the field generated by CoIL, with \( \alpha \) controlling the relative weighting. This idea is also applicable to PnP, for example, by integrating CoIL within PnP-PGM

\[
x^+ \leftarrow D_\sigma(s - \gamma [(1 - \alpha) \nabla g(s) + \alpha \nabla \tilde{g}(s)])
\]

(11.10a)

\[
s^+ \leftarrow x^+ + ((q^+ - 1)/q^+) (x^+ - x)
\]

(11.10b)

where the acceleration parameter \( q > 0 \) is updated as

\[
q^+ \leftarrow \frac{1}{2} \left( 1 + \sqrt{1 + 4q^2} \right).
\]
In the next section, we will provide results highlighting the performance of CoIL in the context of all these algorithms.

11.3 Numerical Validations

We numerically validate CoIL in the context of sparse-view CT. We first substantiate the effectiveness of the proposed form of FFM, and then demonstrate the benefits of using CoIL for image reconstruction. We consider four reconstruction methods, FBP, PGM-TV, GM-RED, and FBP-UNet. FBP-UNet refers to the end-to-end model proposed in [94] and PGM-TV refers to the TV regularized inversion implemented using PGM. We integrate CoIL into these algorithm by including the parameter $\alpha$ as discussed in Section 11.2.2.

11.3.1 Sparse view CT and experimental setup

Sparse view X-ray CT is an imaging modality that aims to reconstruct a tomographic image from few projections. In medical applications, it can significantly reduce the radiation dose and hence reduce the risk of radiation exposure. The reconstruction task in CT can be formulated as the linear inverse problem of form (11.4). In our simulations, we adopt the parallel beam geometry with a measurement operator $A$ corresponding to the Radon transform.

We consider the experimental setting where the X-ray beam is emitted from the view angle $\theta \in [0, \pi]$ and its radiation attenuation is recorded by the detectors at different (normalized) sensor-plane locations $l \in [0, 1]$. Thus, the MLP is trained to map the location and angle $(l, \theta)$ to the corresponding response $r$. Figure 11.4 visualizes eight $512 \times 512$ test images used in all experiments. These images are selected from the scans of two patients in the APPM
Table 11.3: The average SNR values obtained with and without CoIL by using FBP, PGM-TV, GM-RED, and FBP-UNet in the scenarios corresponding to \( P \times I = \{60, 90, 120\} \times \{30, 40, 50\} \).

<table>
<thead>
<tr>
<th># Views ( (P) )</th>
<th>Noise Level ( (I) )</th>
<th>without CoIL</th>
<th>with CoIL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>FBP</td>
<td>PGM-TV</td>
</tr>
<tr>
<td>60</td>
<td>30</td>
<td>0.15</td>
<td>22.66</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>9.09</td>
<td>26.08</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>14.25</td>
<td>29.37</td>
</tr>
<tr>
<td>90</td>
<td>30</td>
<td>1.95</td>
<td>23.32</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>16.07</td>
<td>30.76</td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>3.21</td>
<td>23.79</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>12.10</td>
<td>27.59</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>17.07</td>
<td>31.53</td>
</tr>
</tbody>
</table>

For each test image, CoIL trains separate MLPs to represent its full measurement field in different scenarios \( P \times I = \{60, 90, 120\} \times \{30, 40, 50\} \). We conducted all the experiments, as

human phantom dataset\(^1\) [146], while the scans of other patients are used for training the FBP-UNet and the deep denoiser in GM-RED. We implemented \( A \) and its adjoint \( A^T \) by using RayTransform from the Operator Discretization Library (ODL) [2], which allows fast computation using a GPU backend. We synthesized the test sinograms corresponding to \( P \in \{60, 90, 120\} \) projection views, each further contaminated by three noise levels equivalent to the input signal-to-noise ratio (SNR) of \( I \in \{30,40,50\} \) dB. SNR is also used as a metric to quantify the reconstruction quality

\[
\text{SNR}(\hat{x}, x) \equiv 20 \log_{10} \left( \frac{\|x\|_2}{\|x - \hat{x}\|_2} \right).
\]  

We denote the SNR values averaged over all test images as average SNR.

---

\(^1\)The 2016 NIH-AAPM-Mayo Clinic Low Dose CT Grand Challenge
Figure 11.7: SNR improvements due to CoIL for each reconstruction algorithm. The plot is divided into three regions, corresponding to 60, 90, and 120 projections, respectively. Within each region, the average SNR improvement is plotted against the reconstruction method. The vertical axis is in log-scale for better visualization. Note that CoIL consistently improves the average SNR values for all the considered algorithms in every scenario.

well as the training of all neural networks, on a machine equipped with an Intel Xeon Gold 6130 Processor and four Nvidia GeForce GTX 1080 Ti GPUs. It takes about 30 minutes to train one MLP on this machine using one GPU.

11.3.2 Effectiveness of the FFM layer

We first evaluate the effectiveness of the proposed FFM layer used in the coordinate-based MLP. We trained and compared three networks where: (a) the FFM layer is not implemented (No FFM); (b) the FFM layer implements the positional encoding where $k_i = 2^{i-1}$ (Pos Enc); (c) the FFM layer implements random projection (Gaussian); and (d) the FFM layer implements the proposed linear expansion $k_i = (\pi i)/2$ (CoIL). In the simulations, we use these networks to generate the sinograms corresponding to 360 views. We set $\text{Pos Enc}/\text{CoIL}$ to expand the input to $L = 10$ frequency components and Random to project the input to $d = 256$ features.
Table F.1 summarizes the average SNR values of the sinograms generated by the three networks in all scenarios. Here, we use SNR as the quality metric in the sinogram space because it enables straightforward comparison with the original measurements whose noise level is characterized by input SNR. As shown in the table, CoIL consistently achieves significantly higher SNR values than both No FFM and Pos Enc. Our interpretation is that No FFM is unable to represent the high-frequency variations in the measurement field, while Pos Enc overfits to noise by containing too many high-frequency components. We observe that the sampling pattern in CoIL better captures the nature of the measurements without overfitting to the noise. Furthermore, CoIL also performs better than Random in most
Figure 11.9: Visual illustration of reconstruction with and without CoIL using the several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 40$ dB noise. Each image is labeled with its SNR value with respect to the ground truth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows.

scenarios. In particular, CoIL leads to about 2 dB improvements for fewer measurements at higher noise levels ($I = 30$ and $P = 60$). To better illustrate the difference, a set of visual examples are presented in Figure 11.5, which plots the sinograms and their FBP reconstructions obtained by each network for $P = 120$ and $I = 40$ dB. Specifically, No FFM is able to represent the general structure of the sinogram but fails in generating the details; Pos Enc produces strong artifacts in its sinogram due to its FFM layer. Both Random and CoIL succeed in both representing the high-frequency details and avoiding strong artifacts in the generated measurements. However, CoIL still quantitatively outperforms Random in terms of SNR in most scenarios. In particular, the SNR margin of CoIL is greater than 1.5 dB when the number of views is relatively limited ($P = \{60, 90\}$) and the noise level is
Figure 11.10: Visual illustration of reconstruction with and without CoIL using the several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 180 (used for FBP-UNet) views from $P = 120$ measurements with $I = 40$ dB noise. Each image is labeled with its SNR value with respect to the ground truth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Visual examples reconstructed with and without CoIL using the considered methods.

We have also investigated the evolution of the sinogram quality for different number of views and noise levels. Figure 11.6 plots the SNR of the sinograms obtained by CoIL against the input SNR ($I \in \{30, 40, 50\}$) for different number of views ($P \in \{60, 90, 120\}$). The three dotted horizontal lines in the figure highlight each $I$ value. We first note that CoIL generates sinograms that generally have higher SNR than the noise level in the measurements. In particular, when $I = 30$ dB, the average SNR values are more than 7 dB higher for every high ($I = 30$). The improvement in the sinogram quality is also reflected in the SNR values obtained after FBP reconstruction. Note how CoIL significantly differs from other approaches in the regions highlighted by the bounding boxes.
This highlights the ability of CoIL to generate high-quality sinograms. The figure also demonstrates that the SNR values improve as the number of views increases or noise level decreases. This highlights that the quality of the CoIL fields can be improved by having more measurements or acquiring those that are less noisy.

11.3.3 Evaluation of sinogram interpolation

In this section, we evaluate the interpolation performance of CoIL relative to three interpolation methods: (1) Linear, (2) Bicubic, and (3) interpolation using a pre-trained CNN (ConvNet). ConvNet is implemented by following the work [122], where separate CNNs are trained to map the sinograms of $P = \{60, 90, 120\}$ views to that of 360 views. Note that the training of ConvNet requires a dataset of ground-truth sinograms, while CoIL is trained exclusively on a single under-sampled sinogram of the desired object itself.

Table 11.2 summarizes the average SNR values of the sinograms generated by each aforementioned algorithm with respect to the ground truth for all projection views ($P$) and input SNR values ($I$). The best SNR values and the results of CoIL are respectively indicated by bold and underlined fonts. Note how CoIL significantly outperforms the two conventional interpolation methods, Linear and Bicubic, in all scenarios, showing the benefit of using CoIL for capturing complex patterns in the measurements. Additionally, CoIL achieves reasonable results compared to ConvNet, even though, unlike ConvNet, CoIL is not a traditional supervised learning method. The performance gap between CoIL and ConvNet is the smallest when the input SNR is high, demonstrating that the benefit of ConvNet is primarily due to its ability to separate noise from the data, achieved via traditional supervised training.
11.3.4 Evaluation of reconstruction performance

We next highlight the benefit of CoIL for image reconstruction. We trained all our MLPs by using the FFM layer based on our linear expansion. We implemented FBP by using \texttt{fbp-op} from the ODL package under the default parameter setting. We used DnCNN \cite{Zhang2017} to build the deep denoiser within GM-RED. In every experiment, we selected the network achieving the highest SNR value from the ones corresponding to five noise levels $\sigma \in \{5, 10, 15, 20, 25\}$. For FBP, PGM-TV, and GM-RED, CoIL generates the measurement field with 360 projection views from the test measurements. FBP-UNet corresponds to our own implementation of the architecture used in \cite{Ronneberger2015}. The network was trained in the usual supervised fashion by directly predicting the ground truth from the FBP reconstruction using the $\ell_2$-loss \cite{Sapiro1999}. For FBP-UNet using CoIL, we trained separate CNN models on the datasets consisting of the measurements having $1.5 \times P = \{90, 135, 180\}$ projection views and used CoIL to generate additional measurements to achieve that number. As a baseline, we trained a separate FBP-UNet that directly predicts the ground truth from the test measurements. Note that the baseline networks correspond to the optimal performance that FBP-UNet can achieve for the test measurements without integrating CoIL. In order to stabilize FBP-UNet, we trained these networks using the data with random fluctuations in both projection views ($\pm 15$) and noise amount ($\pm 5$ dB).

Figure F.1 quantitatively evaluates the improvements in imaging quality due to CoIL for all the considered reconstruction algorithms. For each algorithm, we plot the difference between the SNR obtained with and without CoIL. We can clearly observe that CoIL leads to significant quality improvements for all the algorithms. Remarkably, for the higher noise level ($I = 30$ dB), the average improvement by CoIL can sometimes be as high as 20 dB for FBP. In general, CoIL leads to SNR improvements for all algorithms, including for model-based and DL-based
methods. In particular, for $P = 60$ and $I = 50$ dB, FBP-UNet without CoIL achieves 29.52 dB, while FBP-UNet with CoIL achieves 30.54 dB, which is nearly 1 dB improvement. The exact numbers obtained by each algorithm are summarized in Table 11.3. These results highlight that CoIL is able to accurately represent the measurement field and generate high-fidelity measurements that can be used to improve image reconstruction. One can note that the improvements due to CoIL are less dramatic for regularized inversion methods, which we attribute to the strong influence of the regularizer on the final reconstruction quality, even when combined with CoIL.

Figure 11.8 presents visual comparisons of images reconstructed with and without CoIL for $P = 60$ and $I = 40$. Each image is labeled with its SNR with respect to the ground truth and the visual differences are highlighted by arrows in the bounding boxes. This comparison highlights visual improvements due to CoIL. For example, consider the visual differences for FBP-Unet, where one can clearly see additional visual details after integration of the CoIL field. The better reconstruction quality with CoIL is also reflected in the higher SNR values. Additional visual comparisons in Figure 11.9 and Figure 11.10 also highlight the benefit of image reconstruction with CoIL.

### 11.4 Summary

The CoIL methodology developed in this chapter is a new computational-imaging approach that leverages coordinate-based neural representations. CoIL seeks to represent the full measurement field as a single MLP network trained to map the measurement coordinates to their sensor responses. This makes CoIL a self-supervised model that can be trained without any external dataset. Extensive empirical results presented here demonstrate the
improvements due to CoIL in the context of sparse-view CT, highlighting its great potential to work synergistically with existing image reconstruction methods.

The application of CoIL in this chapter are focused on sparse-view CT. One can anticipate the use of CoIL in other imaging modalities where the measurements, viewed as a function of sensing coordinates, exhibit learnable patterns, structures, or features. For example, we envision applications of CoIL in seismic imaging and diffraction tomography, where the measurements exhibit structural patterns relative to transmitter/receiver locations. On the other hand, it is unclear if the current version of CoIL would be applicable to applications where measurements are less structured or randomized relative to sensor locations, which is a common situation in traditional compressive sensing based on random measurement matrices.

It is worth mentioning potential limitations of CoIL. One limitation is the computational overhead for training the MLP, which when integrated to an overall imaging pipeline can significantly reduce the speed of image formation. A partial solution to this issue is to parallelize the MLP training across multiple GPUs, thus reducing the CoIL training overhead. Another possible limitation of CoIL is that, by generating additional measurements, it can increase the per-iteration complexity of image reconstruction algorithms. This suggests that one has to carefully balance the number of synthesized measurements to achieve the best imaging performance under the computational constraints. One potential workaround for this problem in large-scale measurement settings is to replace batch reconstruction algorithms with their stochastic/online counterparts, as discussed in [213, 215, 217, 220, 249].
Part V

Conclusion
Chapter 12

Conclusion

In this dissertation, we introduced multiple novel algorithms that constitute in-depth extensions of the three popular image reconstruction frameworks: plug-and-play priors (PnP), regularization by denoising (RED), and neural fields (NF). In Section 12.1, we summarize the key contributions made by our work. In Section 12.2, we discuss the potential directions of interest for future work.

12.1 Summary

**Fixed-point convergence analysis for PnP/RED.** We provided a unified analysis to prove the fixed-point convergence of both PnP and RED frameworks. The key assumptions required in our analysis are related across different algorithms. Consider $g$ as the data-fidelity term and $D_\sigma$ as the image denoiser. Figure 12.1 visualizes the connections between the analysis for each of our proposed algorithms.
We always assume function $g$ to be convex. The variation occurs in the assumption of differentiability that is dependent on the treatment $g$. For the analysis of gradient-based algorithms, we need $g$ to be differentiable with Lipschitz gradient. On the other hand, proximal-based algorithms require $g$ to be proper, closed, and Lipschitz continuous, but not necessarily differentiable.

We assume the denoiser $D_\sigma$ to satisfy different monotone operator properties for each analysis. The relationship between nonexpansive operator, averaged operator, and firmly nonexpansive operator is visualized in Figure 12.1: nonexpansive $\supset$ averaged $\supset$ firmly nonexpansive. In particular, averaged operator is a $1/2$-averaged operator, and the proximal operator of a convex function is firmly nonexpansive. Note that the fixed-pointed analysis is fully backward compatible with the classic optimization theory.
Our fixed point analysis provide insights on the convergence of PnP/RED algorithms from a monotone operator perspective, covering general denoiers that cannot be linked to explicit regularizers. Intuitively, our analysis shows that the PnP/RED algorithms converge to the set of fixed points where the true global minimizers live if they exist. We believe that our analysis is general because the assumption on denoiers is also applicable to other type of operators such as artifact remover [135].

**Scalable PnP/RED algorithms with convergence analysis.** The high dimensionality of the data faced in computational imaging necessitates the employment of scalable image reconstruction algorithms. To that end, we proposed multiple scalable PnP/RED algorithms that combines deep denoising priors and large-scale optimization. Experimental results has demonstrated that these algorithms can efficiently process the high-dimensional data that is otherwise computationally costly for batch algorithms. We additionally extend our fixed-point analysis to explain their algorithmic convergence, establishing explicit convergence rates. In particular, we trained monotonic deep dneoisers by using the spectral normalization technique to control the Lipschitz of the deep neural network.

**Novel NF-based computational imaging algorithms.** We introduce two novel computational imaging algorithms that are based on NF. The first algorithm, DeCAF, functions in the image domain by representing the desired physical field as a deep neural network, which maps the spatial coordinates to the corresponding physical quantities. In addition to the internal regularization provided by the network, DeCAF additionally integrates external denoising priors to synergistically enhance the final performance. Our validation on quantitative phase microscopy show that DeCAF can recover significantly better images of various biological samples. The second algorithm, CoIL, represents an alternative by functioning in the measurement domain. The purpose of CoIL is to generate the full measurement by exploiting internal information of the given under-sampled measurements. The generated
full measurements can readily used in the subsequent reconstruction algorithms as an extra source of information. Numerical study on computed tomography (CT) highlight the universe improvement brought by CoIL for common reconstruction algorithms.

12.2 Future Work

**Interpretation for the solutions achieved by PnP/RED.** Although our fixed-point analysis justifies the convergence of PnP/RED algorithms, how to interpret their solutions is still an open question due to the absence of the objective function. One interesting direction is to construct an explicit regularizer by training a deep neural network whose output is a scalar and gradient is an image denoiser [51]. In this way, the denoiser is linked to a functional via a neural network, and we can analyze the optimality of the solution by using the objective.

**Extending priors beyond ‘image denoiser’**. Once we adopt the monotone operator perspective, we need not restrict to the concept of image denoiser. Recently, Liu *et al* has empirically extend deep denoising priors to deep artifact removers that is coupled with the imaging modality. One challenge in this direction is to how constrain deep operator so that they obey the conditions but maintain the performance. To address this, investigations on the Lipschitz constant of the neural network is required to develop more precise spectral normalization techniques.

**The use of continuous regularizer in NF.** One benefit of NF is that the desired signal is represented in its original continuous form and need no discretization. This enables the use of continuous regularizers which have been shown successful in off-the-grid compressive sensing [224]. By linking NF back to this classic yet rich literature, potential research directions include the establishment of the recovery guarantee, convergence analysis, and solution optimality.
References


<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Journal Details</th>
</tr>
</thead>
</table>


207


Appendix A

Supplement for Chapter 4

We base on the monotone operator theory summarized in Section 2.4.1 to establish a unified analysis of PnP-PGM and PnP-OPGM. In Appendix A.1, we show that the fixed points of $P$ is the minimizer of $f$. In Appendix A.2, we present the proof for convergence for the batch PnP-PGM. In Appendix A.3, we show that the fixed points of PnP-PGM are also fixed points of PnP-ADMM. In Appendix A.4, we formulate a counter example of bounded dnoise which can lead to divergence of PnP-PGM. In Appendix A.5, we provide the proof for convergence for the online PnP-OPGM.

A.1 Proof of Proposition 4.1

Proposition 4.1 is a direct consequence of the well-known fixed-point interpretation of PGM (see p. 150 in [168]). We provide the proof here for completeness by using the following characterization of the proximal operator

\[ x = \text{prox}_{\gamma r}(z) \iff \frac{z - x}{\gamma} \in \partial r(x), \]  

(A.1)
valid for all \( z \in \mathbb{R}^n \), where \( \partial r(x) \) is the subdifferential of \( r \) at \( x \) [31]. Let \( D_\sigma(\cdot) = \text{prox}_{\gamma r}(\cdot) \) and \( x^* \in \text{fix}(P) \). Then, from (A.1), we have that

\[
x^* = P(x^*) = \text{prox}_{\gamma r}(x^* - \gamma \nabla g(x^*))
\]

\[
\Leftrightarrow -\nabla g(x^*) \in \partial r(x^*)
\]

\[
\Leftrightarrow 0 \in \nabla g(x^*) + \partial r(x^*),
\]

which establishes the desired result.

A.2 Proof of Proposition 4.2

The iterative application of an averaged operator is well known as Krasnosel’skii-Mann iteration [110, 139] and its convergence has been extensively discussed in literature [13, 37, 192]. Below, we use this theory to establish a novel convergence result for PnP-PGM.

From our assumptions, the denoiser \( D_\sigma \) is \( \theta \)-averaged and the gradient-step operator \( G_\gamma \) is \((\gamma L/2)\)-averaged for any \( \gamma \in (0, 2/L) \). From Proposition 2.5, we have that their composition \( P = D_\sigma \circ G_\gamma \) is

\[
\alpha = \frac{\theta + \frac{\gamma L}{2} - \theta \gamma L}{1 - \frac{\theta \gamma L}{2}}
\]

averaged. Consider a single iteration \( x^+ = P(x) \), then we have for any \( x^* \in \text{fix}(P) \) that

\[
\|x^+ - x^*\|_2^2 = \|P(x) - P(x^*)\|_2^2
\]

\[
\leq \|x - x^*\|_2^2 - \left(1 - \frac{\alpha}{\alpha}\right) \|x - P(x) - x^* + P(x^*)\|_2^2
\]

\[
= \|x - x^*\|_2^2 - \left(1 - \frac{\alpha}{\alpha}\right) \|x - P(x)\|_2^2,
\]
where we used Proposition 2.3(c) in Section 2.4.1 and the fact that $x^* = P(x^*)$. By considering the iteration $k \geq 1$ and rearranging the terms, we obtain

$$
\|x^{k-1} - P(x^{k-1})\|_2^2 \\
\leq \left( \frac{\alpha}{1 - \alpha} \right) \left[ \|x^{k-1} - x^*\|_2^2 - \|x^k - x^*\|_2^2 \right].
$$

By averaging this inequality over $t \geq 1$ iterations and dropping the last term $\|x^t - x^*\|_2^2$, we obtain

$$
\frac{1}{t} \sum_{k=1}^{t} \|x^{k-1} - P(x^{k-1})\|_2^2 \leq \frac{1}{t} \left( \frac{\alpha}{1 - \alpha} \right) \|x^0 - x^*\|_2^2.
$$

To obtain the result that depends on $\theta \in (0, 1)$, we note that for any $\gamma \in (0, 1/L]$, we can write

$$
\frac{\alpha}{1 - \alpha} = \frac{\theta + \frac{\gamma L}{2} - \theta \gamma L}{(1 - \theta)(1 - \frac{\gamma L}{2})} \leq \frac{\theta + \frac{1}{2}}{\frac{1 - \theta}{2}} \leq 2 \left( \frac{1 + \theta}{1 - \theta} \right).
$$

(A.2)

To express the result as in (4.4), simply take the minimum of $\|x^{k-1} - P(x^{k-1})\|_2^2$ over a window of past $t \geq 1$ iterations out of the sum to form a lower bound. The desired result is obtained by rearranging the terms.

**A.3 Proof of Proposition 4.3**

Proposition 4.3 is a variant of the result in [147]. For completeness, we provide a proof based on the fixed-point interpretation of ADMM (see p. 157 in [168]).
First note that both $D_\sigma$ and $\text{prox}_{\gamma d}$ are continuous (since they are nonexpansive). Fixed points $x^*, z^*, s^*$ of PnP-ADMM satisfy

\begin{align*}
  z^* &= \text{prox}_{\gamma d}(x^* - s^*) \quad (A.3a) \\
  x^* &= D_\sigma(z^* + s^*) \quad (A.3b) \\
  s^* &= s^* + z^* - x^*. \quad (A.3c)
\end{align*}

From (A.3c), we conclude that $z^* = x^*$. By using the smoothness of $d$ and the characterization (A.1) in (A.3a), we obtain

\[ x^* - s^* - z^* = \gamma \nabla g(z^*) \implies s^* = -\gamma \nabla g(x^*). \]

Finally, by using this in (A.3b), we obtain

\[ x^* = D_\sigma(x^* - \gamma \nabla g(x^*)) = P(x^*), \]

which means that $x^* = z^* \in \text{fix}(P)$ and completes the proof.

**A.4 Proof of Proposition 4.4**

We prove by providing a specific counter example. For simplicity, we assume $n = 1$, but the same example can be generalized for any $n \in \mathbb{N}$. Consider the data fidelity given by the Huber function

\[ d(x) := \begin{cases} 
\frac{1}{2}x^2 & \text{if } |x| \leq 1 \\
|x| - \frac{1}{2} & \text{if } |x| > 1.
\end{cases} \quad (A.4) \]
This function is convex and has a Lipschitz continuous gradient with constant $L = 1$

$$d'(x) = \begin{cases} 
  x & \text{if } |x| \leq 1 \\
  \text{sgn}(x) & \text{if } |x| > 1
\end{cases},$$  \hspace{1cm} (A.5)

where $\text{sgn}(\cdot)$ denotes the sign function. We also consider the denoiser defined as

$$D_\sigma(z) := z + \sigma \sqrt{c} \text{sgn}(z),$$  \hspace{1cm} (A.6)

where $c > 0$ is some constant independent of $\sigma > 0$. Since

$$|D_\sigma(x) - x|^2 = \sigma^2 c,$$  \hspace{1cm} (A.7)

this denoiser satisfies the definition of boundedness in (4.1). Then, for $q_k = 1$, a single iteration of PnP-PGM can be re-written as

$$x = D_\sigma(z) = z + \sigma \sqrt{c} \text{sgn}(z)$$

$$z^+ = x - \gamma d'(x) = \begin{cases} 
  (1 - \gamma) x & \text{if } |x| \leq 1 \\
  x - \gamma \text{sgn}(x) & \text{if } |x| > 1
\end{cases},$$

where we assume any $\gamma \in (0, 1)$. By combining these equations, we obtain

$$z^+ = \begin{cases} 
  (1 - \gamma)(|z| + \sigma \sqrt{c}) \text{sgn}(z) & \text{if } |z| \leq 1 - \sigma \sqrt{c} \\
  (|z| + \sigma \sqrt{c} - \gamma) \text{sgn}(z) & \text{if } |z| > 1 - \sigma \sqrt{c},
\end{cases}$$
where we used the fact that \( \text{sgn}(x) = \text{sgn}(z) \) and expressed \( z = |z| \text{sgn}(z) \). For \( |z| \leq 1 - \sigma \sqrt{c} \), we have that

\[
|z^+| = (1 - \gamma)(|z| + \sigma \sqrt{c}) \\
= |z| + \sigma \sqrt{c} - \gamma |z| - \gamma \sigma \sqrt{c} \\
\geq |z| + \sigma \sqrt{c} - \gamma (1 - \sigma \sqrt{c}) - \gamma \sigma \sqrt{c} \\
= |z| + \sigma \sqrt{c} - \gamma.
\]

On the other hand, for \( |z| > 1 - \sigma \sqrt{c} \), we have that

\[
|z^+| = |z| + \sigma \sqrt{c} - \gamma.
\]

This means that the iterates of PnP-PGM satisfy

\[
|z^t| \geq |z^0| + t(\sigma \sqrt{c} - \gamma), \quad \forall t \in \mathbb{N}.
\]

Therefore, for any \( \sigma > \gamma/\sqrt{c} \) and any \( z^0 \in \mathbb{R} \), the sequence \( \{z^t\}_{t \in \mathbb{N}} \) generated by PnP-PGM diverges. Since the denoiser is bounded, this implies that the sequence \( \{x^t\}_{t \in \mathbb{N}} \) also diverges. This completes the proof.

### A.5 Proof of Proposition 4.5

We define the full proximal-gradient operator

\[
P(x) := D_\sigma (x - \gamma \nabla g(x))
\]  
(A.8)
and its online variant over a minibatch of size $B \geq 1$

$$\hat{P}(x) := D_\sigma(x - \gamma \hat{\nabla} d(x)),$$  
(A.9)

where $\hat{\nabla} d$ denotes the minibatch gradient. The variance bound in Assumption 4.2(d) implies that for all $x \in \mathbb{R}^n$, we have that

$$E \left[ \|P(x) - \hat{P}(x)\|_2^2 \right]$$

$$= E \left[ \|D_\sigma(x - \gamma \nabla g(x)) - D_\sigma(x - \gamma \hat{\nabla} d(x))\|_2^2 \right]$$

$$\leq E \left[ \|x - \gamma \nabla g(x) - x + \gamma \hat{\nabla} d(x)\|_2^2 \right]$$

$$\leq \gamma^2 E \left[ \|\nabla g(x) - \hat{\nabla} d(x)\|_2^2 \right] \leq \frac{\gamma^2 \nu^2}{B},$$  
(A.10)

where in the third row we used the nonexpansiveness of $D_\sigma$. Consider a single iteration $x^k = \hat{P}(x^{k-1})$, then we have for any $x^* \in \text{fix}(P)$ that

$$\|x^k - x^*\|^2_2 = \|\hat{P}(x^{k-1}) - P(x^{k-1}) + P(x^{k-1}) - P(x^*)\|^2_2$$

$$= \|P(x^{k-1}) - P(x^*)\|^2_2 + \|\hat{P}(x^{k-1}) - P(x^{k-1})\|^2_2$$

$$+ 2(\hat{P}(x^{k-1}) - P(x^{k-1}))^T(P(x^{k-1}) - P(x^*))$$

$$\leq \|x^{k-1} - x^*\|^2_2 - \left(1 - \frac{\alpha}{\alpha'}\right) \|x^{k-1} - P(x^{k-1})\|^2_2$$

$$+ \|\hat{P}(x^{k-1}) - P(x^{k-1})\|^2_2$$

$$+ 2\|\hat{P}(x^{k-1}) - P(x^{k-1})\|_2 \cdot \|P(x^{k-1}) - P(x^*)\|_2,$$  
(A.11)

where we used Proposition 2.3(c) in Section 2.4.1 and the Cauchy-Schwarz inequality. Note that due to nonexpansiveness of the operator $P$, we have that

$$\|P(x^{k-1}) - P(x^*)\|_2 \leq \|x^{k-1} - x^*\|_2 \leq R.$$  
(A.12)

[216]
Additionally, by applying Jensen’s inequality to (A.10), we conclude that for all $\mathbf{x} \in \mathbb{R}^n$

$$
\mathbb{E} \left[ \| \mathbf{P}(\mathbf{x}) - \hat{\mathbf{P}}(\mathbf{x}) \|_2 \right] = \mathbb{E} \left[ \sqrt{\| \mathbf{P}(\mathbf{x}) - \hat{\mathbf{P}}(\mathbf{x}) \|_2^2} \right] \\
\leq \sqrt{\mathbb{E} \left[ \| \mathbf{P}(\mathbf{x}) - \hat{\mathbf{P}}(\mathbf{x}) \|_2^2 \right]} \leq \frac{\gamma \nu}{\sqrt{B}}.
$$

(A.13) (A.14)

By taking a conditional expectation of (A.11) and using these bounds, we obtain

$$
\mathbb{E} \left[ \| \mathbf{x}^k - \mathbf{x}^* \|_2^2 - \| \mathbf{x}^{k-1} - \mathbf{x}^* \|_2^2 \mid \mathbf{x}^{k-1} \right] \\
\leq \left( \frac{\alpha - 1}{\alpha} \right) \| \mathbf{x}^{k-1} - \mathbf{P}(\mathbf{x}^{k-1}) \|_2^2 \\
+ \frac{2\gamma \nu}{\sqrt{B}} R + \frac{\gamma^2 \nu^2}{B},
$$

which can be rearranged into

$$
\| \mathbf{x}^{k-1} - \mathbf{P}(\mathbf{x}^{t-1}) \|_2^2 \\
\leq \left( \frac{\alpha}{1 - \alpha} \right) \left[ \frac{\gamma^2 \nu^2}{B} + \frac{2\gamma \nu}{\sqrt{B}} R \\
+ \mathbb{E} \left[ \| \mathbf{x}^{k-1} - \mathbf{x}^* \|_2^2 - \| \mathbf{x}^k - \mathbf{x}^* \|_2^2 \mid \mathbf{x}^{k-1} \right] \right].
$$

By averaging the inequality over $t \geq 1$ iterations, taking the total expectation, and dropping the last term, we obtain

$$
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^t \| \mathbf{x}^{k-1} - \mathbf{P}(\mathbf{x}^{k-1}) \|_2^2 \right] \\
\leq \frac{\alpha}{1 - \alpha} \left[ \frac{\gamma^2 \nu^2}{B} + \frac{2\gamma \nu}{\sqrt{B}} R + \frac{R^2}{t} \right],
$$

[217]
where we used the law of total expectation. By using the inequality (A.2), we can rewrite this expression as

\[
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \| x^{k-1} - P(x^{k-1}) \|_2^2 \right] 
\leq 2 \left( \frac{1 + \theta}{1 - \theta} \right) \left[ \frac{\gamma^2 \nu^2}{B} + \frac{2 \gamma \nu}{\sqrt{B}} R + \frac{R^2}{t} \right]
\]

Note that to obtain the results in Corollary 4.1, simply replace given values for $\gamma$ and $B$ into the inequality, and use the following bounds that are valid for any $t \in \mathbb{N}$

\[
\frac{1}{t} \leq \frac{1}{\sqrt{t}} \quad \text{and} \quad \frac{1}{t^2} \leq \frac{1}{t}.
\]

This establishes the desired results.

### A.6 List of Selected Hyperparameters

We optimized the algorithmic hyperparameters of PnP-OPGM, PnP-APGM, and PnP-ADMM for each DT reconstruction with the fixed per-iteration budget of measurements. The $\gamma > 0$ was empirically evaluated at the 300th iterations by using APGM with the backtracking
selection of step-size. The parameter $\rho > 0$ of PnP-ADMM is fixed to $1 \times 10^{-3}$ in favor of a better searching range of $\sigma > 0$, which controls strength of denoising. Table A.1 lists the optimal $\sigma$ for each algorithm for the reconstruction of every test image.
Appendix B

Supplement for Chapter 6

We adopt monotone operator theory [14, 192] for a unified analysis of IPA. In Appendix B.1, we present the convergence analysis of IPA. In Appendix B.2, we analyze the convergence of the algorithm for strongly convex data-fidelity terms and contractive denoisers. In Appendix B.3, we discuss interpretation of IPA’s fixed-points from the perspective of monotone operator theory. For completeness, in Appendix B.4, we discuss the convergence results for traditional PnP-ADMM [193]. In Appendix B.5, we summarize the major similarities and differences of variations of PnP and RED algorithms. In Appendix B.6, we provide additional technical details of our deep neural net architecture, the computation of proximal operators, and results of additional simulations. In Appendix B.6, we provide additional numerical results on the comparison with different image priors, the results on the validation of the firmly nonexpansiveness of our denoising neural network, the performance of IPA with different minibatch sizes and the detailed derivations on how the dual formulation was evaluated to estimate the solution of proximal operator used in our intensity diffraction tomography experiment.

[220]
For the sake of simplicity, we use $\| \cdot \|$ to denote the standard $\ell_2$-norm in $\mathbb{R}^n$. We will also use $D(\cdot)$ instead of $D_{\sigma}(\cdot)$ to denote the denoiser, thus dropping the explicit notation for $\sigma$.

**B.1 Convergence Analysis of IPA**

In this section, we present the convergence analysis of IPA. A fixed-point convergence of averaged operators is well-known under the name of Krasnosel’skii-Mann theorem (see Section 5.2 in [14]). Additionally, PnP-ADMM was analyzed for strongly convex data-fidelity terms $g$ and contractive residual denoisers $R_{\sigma}$ [193]. Our analysis extends these results to IPA by providing an explicit upper bound on the convergence. In Appendix B.1, we present the main steps of the proof, while in Appendix B.1 we prove two technical lemmas useful for our analysis.

**Proof of Theorem 6.1**

Appendix B.3 establishes that $S$ defined in (6.5) is firmly nonexpansive. Consider any $v^* \in \text{zer}(S)$ and any $v \in \mathbb{R}^n$, then we have

\[
\|v - v^* - Sv\|^2 \leq \|v - v^*\|^2 - \|Sv\|^2,
\]

where we used the firm nonexpansiveness of $S$ and $Sv^* = 0$. The direct consequence of (B.1) is that

\[
\|v - v^* - Sv\| \leq \|v - v^*\|.
\]
We now consider the following two equivalent representations of IPA for some iteration $k \geq 1$

\[
\begin{align*}
    z^k &= G_{i_k}(x^{k-1} + s^{k-1}) \\
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

(B.2a)

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

(B.2b)

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

\[
\begin{align*}
    x^k &= D(z^k - s^{k-1}) \\
    s^k &= s^{k-1} + x^k - z^k \\
    x^{k-1} &= D(v^{k-1}) \\
    z^k &= G_{i_k}(2x^{k-1} - v^{k-1}) \\
    v^k &= v^{k-1} + z^k - x^{k-1},
\end{align*}
\]

where $i_k$ is a random variable uniformly distributed over $\{1, \ldots, b\}$, $G_i = \text{prox}_{\gamma g_i}$ is the proximal operator with respect to $g_i$, and $D$ is the denoiser. To see the equivalence between (B.2a) and (B.2b), simply introduce the variable $v^k = z^k - s^{k-1}$ into (B.2b) [193]. It is straightforward to verify that (B.2a) can also be rewritten as

\[
\begin{align*}
    v^k &= v^{k-1} - S_{i_k}(v^{k-1}) \quad \text{with} \quad S_{i_k} := D - G_{i_k}(2D - I).
\end{align*}
\]

(B.3)

Then, for any $v^* \in \text{zer}(S)$, we have that
∥v^k - v^*∥^2
= ∥v^{k-1} - v^* - Sv^{k-1}∥^2 + ∥Sv^{k-1} - S_{ik} v^{k-1}∥^2
+ 2(Sv^{k-1} - S_{ik} v^{k-1})^T(v^{k-1} - v^* - Sv^{k-1})
≤ ∥v^{k-1} - v^*∥^2 + ∥Sv^{k-1}∥^2 + ∥Sv^{k-1} - S_{ik} v^{k-1}∥^2
+ 2∥Sv^{k-1} - S_{ik} v^{k-1}∥∥v^{k-1} - v^*∥
≤ ∥v^{k-1} - v^*∥^2 + ∥Sv^{k-1}∥^2 + ∥Sv^{k-1} - S_{ik} v^{k-1}∥^2
+ 2(R + 2γL)∥Sv^{k-1} - S_{ik} v^{k-1}∥,

where in the first inequality we used Cauchy-Schwarz and (B.1), and in the second inequality we used Lemma B.2 in Appendix B.1. By taking the conditional expectation on both sides, invoking Lemma B.1 in Appendix B.1, and rearranging the terms, we get

∥Sv^{k-1}∥^2 ≤ ∥v^{k-1} - v^*∥^2 - E[∥v^k - v^*∥^2 | v^{k-1}] + 4γLR + 12γ^2L^2.

Hence, by averaging over \( t \geq 1 \) iterations and taking the total expectation, we obtain

\[ E \left[ \frac{1}{t} \sum_{k=1}^{t} ∥Sv^{k-1}∥^2 \right] ≤ \frac{(R + 2γL)^2}{t} + 4γLR + 12γ^2L^2. \]

The final result is obtained by noting that

\[ 4γLR + 12γ^2L^2 ≤ \max\{γ, γ^2\}(4LR + 12L^2). \]
Lemmas Useful for the Proof of Theorem 6.1

This section presents two technical lemmas used in our analysis in Appendix B.1.

**Lemma B.1.** Assume that Assumptions 6.1-6.3 hold and let $i_k$ be a uniform random variable over $\{1, \ldots, b\}$. Then, we have that

$$
\mathbb{E} \left[ \| S_{i_k} v - S v \|^2 \right] \leq 4 \gamma^2 L^2, \quad v \in \mathbb{R}^n.
$$

**Proof.** Let $z_i = G_i(x)$ and $z = G(x)$ for any $1 \leq i \leq b$ and $x \in \mathbb{R}^n$. From the optimality conditions for each proximal operator

$$
G_i x = \text{prox}_{\gamma g_i}(x) = x - \gamma g_i(z_i), \quad g_i(z_i) \in \partial g_i(z_i)
$$

and

$$
G x = \text{prox}_{\gamma g}(x) = x - \gamma g(z)
$$

such that

$$
g(z) = \frac{1}{b} \sum_{i=1}^{b} g_i(z) \in \partial g(z),
$$

where we used Proposition 2.15 in Section 2.4.3. By using the bound on all the subgradients (due to Assumption 6.1 and Proposition 2.16 in Section 2.4.3), we obtain

$$
\| G_i(x) - G(x) \| = \| \text{prox}_{\gamma g_i}(x) - \text{prox}_{\gamma g}(x) \|
$$

$$
= \gamma \| g_i(z_i) - g(z) \| \leq 2\gamma L,
$$

where $L > 0$ is the Lipschitz constant of all $g_i$’s and $g$. This inequality directly implies that

$$
\| S v - S_i v \| = \| G(2D v - v) - G_i(2D v - v) \| \leq 2\gamma L.
$$
Since, this inequality holds for every $i$, it also holds in expectation.

**Lemma B.2.** Assume that Assumptions 6.1-6.3 hold and let the sequence $\{v^k\}$ be generated via the iteration (B.3). Then, for any $k \geq 1$, we have that

$$\|v^k - v^*\| \leq (R + 2\gamma L) \quad \text{for all} \quad v^* \in \zer(S).$$

**Proof.** The optimality of the proximal operator in (B.3) implies that there exists $g_{ik}(z^k) \in \partial g_{ik}(z^k)$ such that

$$z^k = G_{ik}(2x^{k-1} - v^{k-1})$$

$$\Leftrightarrow 2x^{k-1} - v^{k-1} - z^k = \gamma g_{ik}(z^k).$$

By applying $v^k = v^{k-1} - S_{ik}(v^{k-1}) = v^{k-1} + z^k - x^{k-1}$ to the equality above, we obtain

$$x^{k-1} - v^k = \gamma g_{ik}(z^k) \quad \Leftrightarrow \quad v^k = x^{k-1} - \gamma g_{ik}(z^k).$$

Additionally, for any $v^* \in \zer(S)$ and $x^* = D(v^*)$, we have that

$$S(v^*) = D(v^*) - G(2D(v^*) - v^*) = x^* - G(2x^* - v^*) = 0$$

$$\Rightarrow \quad x^* - v^* = \gamma g(x^*) \quad \text{for some} \quad g(x^*) \in \partial g(x^*).$$

Thus, by using Assumption 6.3 and the bounds on all the subgradients (due to Assumption 6.1 and Proposition 2.16 in Section 2.4.3), we obtain

$$\|v^k - v^*\| = \|x^{k-1} - \gamma g_{ik}(z^k) - x^* - \gamma g(x^*)\|$$

$$\leq \|x^{t-1} - x^*\| + 2\gamma L \leq (R + 2\gamma L).$$

[225]
B.2 Analysis of IPA for Strongly Convex Functions

In this section, we perform analysis of IPA under a different set of assumptions, namely under the assumptions adopted in [193].

**Assumption B.1.** Each $g_i$ is proper, closed, strongly convex with constant $M_i > 0$, and Lipschitz continuous with constant $L_i > 0$. We define the smallest strong convexity constant as $M = \min\{M_1, \ldots, M_b\}$ and the largest Lipschitz constant as $L = \max\{L_1, \ldots, L_b\}$.

This assumption further restricts Assumption 6.1 to strongly convex functions.

**Assumption B.2.** The residual $R_\sigma := I - D_\sigma$ of the denoiser $D_\sigma$ is a contraction. It thus satisfies

$$\|R x - R y\| \leq \epsilon \|x - y\|,$$

for all $x, y \in \mathbb{R}^n$ for some constant $0 < \epsilon < 1$.

This assumption replaces Assumption 6.2 by assuming that the residual of the denoiser is a contraction. Note that this can be practically imposed on deep neural net denoisers via spectral normalization [153]. We can then state the following.

**Theorem B.1.** Run IPA for $t \geq 1$ iterations with random i.i.d. block selection under Assumptions 6.3-B.2 using a fixed penalty parameter $\gamma > 0$. Then, the iterates of IPA satisfy

$$\mathbb{E} \left[ \|x^k - x^*\| \right] \leq \eta^k (2R + 4\gamma L) + \frac{4\gamma L}{1 - \eta}, \quad 0 < \eta < 1.$$

Proof. It was shown in Theorem 2 of [193] that under Assumptions B.1 and B.2, we have that

$$\|(1 - S)x - (1 - S)y\| \leq \eta \|x - y\| \quad \text{(B.4)}$$
with
\[ \eta := \left( \frac{1 + \epsilon + \epsilon \gamma M + 2 \epsilon^2 \gamma M}{1 + \gamma M + 2 \epsilon \gamma M} \right), \]
for all \( x, y \in \mathbb{R}^n \), where \( S \) is given in (6.5). Hence, when
\[ \frac{\epsilon}{\gamma M(1 + \epsilon - 2 \epsilon^2)} < 1, \]
the operator \((1 - S)\) is a contraction. Using the reasoning in Appendix B.1, the sequence \( v^k = z^k - s^{k-1} \) can be written as
\[ v^k = v^{k-1} - S_{i_k}(v^{k-1}) \quad \text{with} \quad S_{i_k} := D - G_{i_k}(2D - I) . \] (B.5)

Then, for any \( v^* \in \text{zer}(S) \), we have that
\[
\begin{align*}
\|v^k - v^*\|^2 & = \|(1 - S)v^{k-1} - (1 - S)v^*\|^2 \\
& + 2((1 - S)v^{k-1} - (1 - S)v^*)^T((1 - S)v^{k-1} - (1 - S)v^*) \\
& + \|(1 - S)v^{k-1} - (1 - S)v^*\|^2 \\
& \leq \eta^2\|v^{k-1} - v^*\|^2 + 2\eta\|v^{k-1} - v^*\|^2\|S_{i_k}v^{k-1} - Sv^{k-1}\| \\
& + \|S_{i_k}v^{k-1} - Sv^{k-1}\|^2 ,
\end{align*}
\]
where we used the Cauchy-Schwarz inequality and the fact that \((1 - S)\) is \( \eta \)-contractive. By taking the conditional expectation on both sides, invoking Lemma B.1 in Appendix B.1, and completing the square, we get
\[ \mathbb{E} \left[ \|v^k - v^*\|^2 | v^{k-1} \right] \leq \left( \eta\|v^{k-1} - v^*\| + 2\gamma L \right)^2 . \]
Then, by applying the Jensen inequality and taking the total expectation, we get

\[
\mathbb{E} \left[ \|v^k - v^*\| \right] \leq \eta \mathbb{E} \left[ \|v^{k-1} - v^*\| \right] + 2\gamma L.
\]

By iterating this result and invoking Lemma B.2 from Appendix B.1, we obtain

\[
\mathbb{E} \left[ \|v^k - v^*\| \right] \leq \eta^k (R + 2\gamma L) + \frac{(2\gamma L)}{(1 - \eta)}.
\]

Finally by using the nonexpansiveness of \((1/(1 + \epsilon))D\) (see Lemma 9 in [193]) and the fact that \(x^* = D(v^*)\), we obtain

\[
\mathbb{E} \left[ \|x^k - x^*\| \right] \leq (1 + \epsilon) \left[ \eta^k (R + 2\gamma L) + \frac{2\gamma L}{1 - \eta} \right]
\leq \eta^k (2R + 4\gamma L) + \frac{4\gamma L}{1 - \eta}.
\]

This concludes the proof.

### B.3 Fixed Point Interpretation

Fixed points of PnP algorithms have been extensively discussed in the recent literature [37, 147, 193]. Our goal in this section is to revisit this topic in a way that leads to a more intuitive equilibrium interpretation of PnP. Our formulation has been inspired from the classical interpretation of ADMM as an algorithm for computing a zero of a sum of two monotone operators [66].
Equilibrium Points of PnP Algorithms

It is known that a fixed point \((\mathbf{x}^*, \mathbf{z}^*, \mathbf{s}^*)\) of PnP-ADMM (and of all PnP algorithms [147]) satisfies

\[
\mathbf{x}^* = G(\mathbf{x}^* + \mathbf{s}^*) \quad \text{and} \quad \mathbf{x}^* = D(\mathbf{x}^* - \mathbf{s}^*),
\] (B.6)

with \(\mathbf{x}^* = \mathbf{z}^*\), where \(G = \text{prox}_{\gamma g}\). Consider the inverse of \(D\) at \(\mathbf{x} \in \mathbb{R}^n\), which is a set-valued operator \(D^{-1}(\mathbf{x}) := \{\mathbf{z} \in \mathbb{R}^n : \mathbf{x} = D\sigma(\mathbf{z})\}\). Note that the inverse operator exists even when \(D\) is not a bijection (see Section 2 of [192]). Then, from the definition of \(D^{-1}\) and optimality conditions of the proximal operator, we can equivalently rewrite (B.6) as follows

\[
\mathbf{s}^* \in \gamma \partial g(\mathbf{x}^*) \quad \text{and} \quad -\mathbf{s}^* \in D^{-1}(\mathbf{x}^*) - \mathbf{x}^*.
\]

This directly leads to the following equivalent representation of PnP fixed points

\[
0 \in T(\mathbf{x}^*) := \gamma \partial g(\mathbf{x}^*) + (D^{-1}(\mathbf{x}^*) - \mathbf{x}^*). \quad (B.7)
\]

Hence, a vector \(\mathbf{x}^*\) computed by PnP can be interpreted as an equilibrium point between two terms with \(\gamma > 0\) explicitly influencing the balance.

Equivalence of Zeros of \(T\) and \(S\)

Define \(\mathbf{v}^* := \mathbf{z}^* - \mathbf{s}^*\) for a given fixed point \((\mathbf{x}^*, \mathbf{z}^*, \mathbf{s}^*)\) of PnP-ADMM and consider the operator

\[
S = D - G(2D - I) \quad \text{with} \quad G = \text{prox}_{\gamma g},
\]
which was defined in (6.5). Note that from (B.6), we also have $x^* = D(v^*)$ and $v^* = x^* - s^*$ (due to $z^* = x^*$). We then have the following equivalence

\[
0 \in T(x^*) = \gamma \partial g(x^*) + (D^{-1}(x^*) - x^*),
\]

\[
\Leftrightarrow \begin{cases} 
    x^* = G(x^* + s^*) \\
    x^* = D(x^* - s^*) 
\end{cases}
\]

\[
\Leftrightarrow \begin{cases} 
    x^* = G(2x^* - v^*) \\
    x^* = D(v^*) 
\end{cases}
\]

\[
\Leftrightarrow S(v^*) = D(v^*) - G(2D(v^*) - v^*) = 0,
\]

where we used the optimality conditions of the proximal operator $G$. Hence, the condition that $v^* = z^* - s^* \in \text{zer}(S)$ is equivalent to $x^* = D(v^*) \in \text{zer}(T)$.

**Firm Nonexpansiveness of S**

We finally would like to show that under Assumptions 6.1-6.3, the operator $S$ is firmly nonexpansive. Assumption 6.2 and Proposition 2.14 in Section 2.4.3 imply that $D$ and $G$ are firmly nonexpansive. Then, Proposition 2.6 in Section 2.4.1 implies that $(2D - I)$ and $(2G - I)$ are nonexpansive. Thus, the composition $(2G - I)(2D - I)$ is also nonexpansive and

\[
(1 - S) = \frac{1}{2}I + \frac{1}{2}(2G - I)(2D - I) \quad \text{(B.8)}
\]

is $(1/2)$-averaged. Then, Proposition 2.6 in Section 2.4.1 implies that $S$ is firmly nonexpansive.
B.4 Convergence Analysis of PnP-ADMM

The following analysis has been adopted from [193]. For completeness, we summarize the key results useful for our own analysis by restating them under the assumptions in Section 6.4.

Equivalence between PnP-ADMM and PnP-DRS

An elegant analysis of PnP-ADMM emerges from its interpretation as the Douglas–Rachford splitting (DRS) algorithm [193]. This equivalence is well-known and has been extensively studied in the context of convex optimization [66]. Here, we restate the relationship for completeness.

Consider the sequences of DRS (top) and ADMM (bottom)

\[
\begin{align*}
  x^{k-1} &= D(v^{k-1}) \\
  z^k &= G(2x^{k-1} - v^{k-1}) \\
  v^k &= v^{k-1} + z^k - x^{k-1}
\end{align*}
\]

\[
\begin{align*}
  z^k &= G(x^{k-1} + s^{k-1}) \\
  x^k &= D(z^k - s^{k-1}) \\
  s^k &= s^{k-1} + x^k - z^k,
\end{align*}
\]

where \( G := \text{prox}_{\gamma g} \) is the proximal operator and \( D \) is the denoiser. To see the equivalence between them, simply introduce the variable change \( v^k = z^k - s^{k-1} \) into DRS. Note also the DRS sequence can be equivalently written as

\( v^k = v^{k-1} - S(v^{k-1}) \) with \( S := D - G(2D - I) \).
To see this simply rearrange the terms in DRS as follows

\[ v^k = v^{k-1} + G(2x^{k-1} - v^{k-1}) - x^{k-1} \]

\[ = v^{k-1} - [D(v^{k-1}) - G(2D(v^{k-1}) - v^{k-1})] \] .

**Convergence Analysis of PnP-DRS and PnP-ADMM**

It was established in Appendix B.3 that \( S \) defined in (6.5) is firmly nonexpansive.

Consider a single iteration of DRS \( v^+ = v - Sv \). Then, for any \( v^* \in \text{zer}(S) \), we have

\[ \|v^+ - v^*\|^2 = \|v - v^*\|^2 - 2(Sv - Sv^*)^T(v - v^*) + \|Sv\|^2 \]

\[ \leq \|v - v^*\|^2 - \|Sv\|^2 , \]

where we used \( Sv^* = 0 \) and firm nonexpansiveness of \( S \). By rearranging the terms, we obtain the following upper bound at iteration \( k \geq 1 \)

\[ \|Sv^{k-1}\|^2 \leq \|v^{k-1} - v^*\|^2 - \|v^k - v^*\|^2 . \]  

(B.9)

By averaging the inequality (B.9) over \( t \geq 1 \) iterations, we obtain

\[ \frac{1}{t} \sum_{k=1}^{t} \|Sv^{k-1}\|^2 \leq \frac{\|v^0 - v^*\|^2}{t} \leq \frac{(R + 2\gamma L)^2}{t} \]

where used the bound on \( \|v^0 - v^*\| \leq (R + 2\gamma L) \) that can be easily obtained by following the steps in Lemma B.2 in Appendix B.1.
This result directly implies that $\|Sv^t\| \to 0$ as $t \to 0$. Additionally, Krasnosel’skii-Mann theorem (see Section 5.2 in [14]) implies that $v^t \to \text{zer}(S)$. Then, from continuity of $D$, we have that $x^t = D(v^t) \to \text{zer}(T)$ (see also Appendix B.3). This completes the proof.

### B.5 Variants of PnP/RED Algorithms

Several variants of PnP/RED algorithms are summarized in Table B.1, focusing on two properties (a) the ability to handle nonsmooth data-fidelity terms, and (b) the ability to handle online/minibatch processing of the measurements. The table highlights the way IPA complements existing work by addressing both (a) and (b).
Table B.1: Overview of several existing PnP/RED algorithms

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Nonsmooth</th>
<th>Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>PnP-ADMM [42, 193, 209, 236]</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>PnP-ISTA/PnP-FISTA [103, 147, 214]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>PnP-SPGM [214]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>RED-SD [188]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>RED-ADMM [185, 188]</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>prDeep [148]</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>RED-PG/RED-APG [185]</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>SIMBA/On-RED [248, 249]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>IPA (proposed)</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

B.6 Additional Technical Details

In this section, we present several technical details of our experiments. Section B.6 discusses the architecture and training of the DnCNN prior. Section B.6 explains the computation of the proximal operators used in our experiments. Section B.6 presents extra details and validations that complement the experiments in Section 6.5 with additional insights for IPA.

Architecture and Training of the DnCNN Prior

Fig. D.1 illustrates the architectural details of the DnCNN prior used in our experiments. In total, the network contains 7 layers, of which the first 6 layers consist of a convolutional layer...
Table B.2: Per-iteration memory usage specification for reconstructing 512×512 images

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>IPA (60)</th>
<th>PnP-ADMM (300)</th>
<th>PnP-ADMM (600)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>size memory</td>
<td>size memory</td>
<td>size memory</td>
</tr>
<tr>
<td>{A_i} real</td>
<td>512 × 0.23 GB</td>
<td>512 × 1.17 GB</td>
<td>512 × 2.34 GB</td>
</tr>
<tr>
<td></td>
<td>512 × 60</td>
<td>512 × 300</td>
<td>512 × 600</td>
</tr>
<tr>
<td>{A_i} imaginary</td>
<td>512 × 0.23 GB</td>
<td>512 × 1.17 GB</td>
<td>512 × 2.34 GB</td>
</tr>
<tr>
<td></td>
<td>512 × 60</td>
<td>512 × 300</td>
<td>512 × 600</td>
</tr>
<tr>
<td>{y_i}</td>
<td>512 × 0.47 GB</td>
<td>512 × 2.34 GB</td>
<td>512 × 4.69 GB</td>
</tr>
<tr>
<td></td>
<td>512 × 60</td>
<td>512 × 300</td>
<td>512 × 600</td>
</tr>
<tr>
<td>others combined</td>
<td>— 0.03 GB</td>
<td>— 0.03 GB</td>
<td>— 0.03 GB</td>
</tr>
<tr>
<td>Total</td>
<td>0.97 GB</td>
<td>4.72 GB</td>
<td>9.41 GB</td>
</tr>
</tbody>
</table>

Figure B.2: Illustration of the convergence of IPA for a DnCNN prior under drastically changed γ values. The average normalized distance to \( \text{zer}(S) \) and SNR (dB) are plotted against the iteration number with the shaded areas representing the range of values attained over 12 test images. In practice, the convergence speed improves with larger values of γ. However, IPA still can achieve same level of SNR results for a wide range of γ values.

and a rectified linear unit (ReLU), while the last layer is just a convolution. A skip connection from the input to the output is implemented to enforce residual learning. The output images of the first 6 layers have 64 feature maps while that of the last layer is a single-channel image. We set all convolutional kernels to be 3 × 3 with stride 1, so that intermediate images have the same spatial size as the input image. We generated 11101 training examples by adding AWGN to 400 images from the BSD400 dataset [140] and extracting patches of 128 × 128 pixels with stride 64. We trained DnCNN to optimize the mean squared error by using the Adam optimizer [114].

[235]
Figure B.3: Visual examples of the reconstructed House (upper) and Parrot (bottom) images by IPA and PnP-ADMM. The first and last columns correspond to PnP-ADMM under DnCNN with 5 fixed measurements and with the full 60 measurements, respectively. The second, third, and fourth column correspond to IPA with a small minibatch of size 5 under TV, BM3D, and DnCNN, respectively. Each image is labeled by its SNR (dB) with respect to the original image, and the visual difference is highlighted by the boxes underneath. Note that IPA recovers the details lost by the batch algorithm with the same computational cost and achieves the same high-quality results as the full batch algorithm.

We use the spectral normalization technique in [195] to control the global Lipschitz constant (LC) of DnCNN. In the training, we constrain the residual network $R_{\sigma}$ to have LC smaller than 1. Since the firm non-expansiveness implies non-expansiveness, this provides a necessary condition for $R_{\sigma}$ to satisfy Assumption 6.2. The training of DnCNN with and without spectral normalization takes 4 and 1.82 hours, respectively, on the same hardware. Thus, for about $2 \times$ increase in the denoiser pre-training time, one can make IPA/PnP-ADMM convergent.
Computation of Proximal Operators

In the CS experiments, the measurement matrix $A$ is a random matrix, and the data-fidelity term is based on the $\ell_1$-norm: $\|Ax - y\|_1$. While closed form solution of the proximal operator is inaccessible in this setting, we can efficiently approximate the proximal solution in the dual domain by using projected gradient method (PGM) [20]. Note that the closed-form solution is also unavailable for other $\ell_1$-based proximal operators [20, 99]. The stopping criteria for the PGM algorithm are that either that the total iterations exceeds 200, or that the relative change between two iterates is below $1 \times 10^{-4}$.

For intensity diffraction tomography (IDT), we adopted the linearized forward model developed in [131], which is based on the Fourier transform. For the $i^{th}$ measurement, the forward model for the 2-dimensional case is described as $A_i = F^H H_i F$, where $F$ and $F^H$ denote the discrete Fourier transform and its inverse, respectively, and $H_i$ corresponds to light transfer function of the $i^{th}$ illumination. Under the $\ell_2$-norm, we can directly derive the closed-form solution of the proximal operator in the Fourier space [4, 245].

Extra Details and Validations for Optical Tomography

All experiments were run on the machine equipped with an Intel Core i7 Processor that has 6 cores of 3.2 GHz and 32 GBs of DDR memory. We trained all neural nets using NVIDIA RTX 2080 GPUs. We define the SNR (dB) used in the experiments as

$$\text{SNR}(\hat{x}, x) \triangleq \max_{a, b \in \mathbb{R}} \left\{ 20 \log_{10} \left( \frac{\|x\|_2}{\|x - a\hat{x} + b\|_2} \right) \right\},$$

where $\hat{x}$ is the estimate and $x$ is the ground truth.
For intensity diffraction tomography, we implemented an epoch-based selection rule due to the large size of data. We randomly divide the measurements (along with the corresponding forward operators) into non-overlapping chunks of size 60 and save these chunks on the hard drive. At every iteration, IPA loads only a single random chunk into the memory while the full-batch PnP-ADMM loads all chunks sequentially and process the full set of measurements. This leads to the lower per iteration cost and less memory usage of IPA than PnP-ADMM.

Table B.2 shows extra examples of the memory usage specification for reconstructing 512 × 512 pixel permittivity images. These results follow the same trend observed in Table 6.2. We also conduct some extra validations that provide additional insights into IPA. In these simulations, we use images of size 254 × 254 pixels from Set 12 as test examples. We assume real permittivity functions with the total number of measurement \( b = 60 \).

Fig. B.2 illustrates the evolution of the convergence of IPA for different values of the penalty parameter. We consider three different values of \( \gamma \in \{\gamma_0, \gamma_0/20, \gamma/400\} \) with \( \gamma_0 = 20 \). The average normalized distance \( \|S(v^k)\|^2_2/\|v^k\|^2_2 \) and SNR are plotted against the iteration number and labeled with their respective final values. The shaded areas represent the range of values attained across all test images. IPA randomly selects 5 measurements in every iteration to impose the data-consistency. Fig. B.2 complements the results in Fig 6.1 by showing the fast convergence speed in practice with larger values of \( \gamma \). On the other hand, this plot further demonstrates that IPA is stable in terms of the SNR results for a wide range of \( \gamma \) values.

Our final simulation compares the reconstruction performance of IPA using TV, BM3D, and DnCNN. Since TV has a proximal operator, it serves as a baseline. The reconstruction performance of IPA on House and Parrot are presented in Fig. B.3, while average SNR values for additional images are presented in Table B.3. We include the results of PnP-ADMM using 5 fixed measurements and the full batch as reference. First, note the significant improvement
Table B.3: Optimized SNR (dB) obtained by IPA under different priors for images from Set12 from [262]

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>PnP-ADMM</th>
<th>IPA (Ours)</th>
<th>PnP-ADMM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Fixed 5)</td>
<td>(Random 5 from full 60)</td>
<td>(Full 60)</td>
</tr>
<tr>
<td>Denoisers</td>
<td>DnCNN</td>
<td>TV</td>
<td>BM3D</td>
</tr>
<tr>
<td>Cameraman</td>
<td>15.95</td>
<td>17.45</td>
<td>17.38</td>
</tr>
<tr>
<td>House</td>
<td>19.22</td>
<td>21.79</td>
<td>21.97</td>
</tr>
<tr>
<td>Pepper</td>
<td>17.06</td>
<td>18.68</td>
<td>19.55</td>
</tr>
<tr>
<td>Starfish</td>
<td>18.20</td>
<td>19.29</td>
<td>20.29</td>
</tr>
<tr>
<td>Monarch</td>
<td>17.70</td>
<td>19.81</td>
<td>18.66</td>
</tr>
<tr>
<td>Aircraft</td>
<td>17.15</td>
<td>18.67</td>
<td>18.83</td>
</tr>
<tr>
<td>Parrot</td>
<td>17.13</td>
<td>18.60</td>
<td>18.27</td>
</tr>
<tr>
<td>Lenna</td>
<td>15.41</td>
<td>16.48</td>
<td>16.32</td>
</tr>
<tr>
<td>Barbara</td>
<td>13.63</td>
<td>16.00</td>
<td>17.53</td>
</tr>
<tr>
<td>Boat</td>
<td>17.98</td>
<td>19.35</td>
<td>20.21</td>
</tr>
<tr>
<td>Pirate</td>
<td>17.93</td>
<td>19.36</td>
<td>19.45</td>
</tr>
<tr>
<td>Couple</td>
<td>15.40</td>
<td>17.31</td>
<td>17.53</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>16.90</strong></td>
<td><strong>18.57</strong></td>
<td><strong>18.83</strong></td>
</tr>
</tbody>
</table>

of IPA over PnP-ADMM under the same computational budget. Second, using learned priors in IPA leads to better reconstruction than other priors. For instance, DnCNN outperforms TV and BM3D by 0.7 dB in SNR. Finally, the agreement between IPA and the full batch PnP-ADMM highlights the nearly optimal performace of IPA at a lower computational cost and memory usage.
Appendix C

Supplement for Chapter 8

In the first two sections, Appendix C.1 and C.2, we present the proof for Theorem 8.1 and 8.2, respectively. In Appendix C.3, we discussed several strategies for block-friendly implementation. Lastly, Appendix C.4 introduced the technical details on training DnCNN networks and discussed the influences of the network’s Lipschitz constant and the padding size in image denoising. Preliminary material on monotone operator theory is summarized in Section 2.4 in Chapter 2

C.1 Proof of Theorem 8.1

A fixed-point convergence of averaged operators is well-known under the name of Krasnosel’skii-Mann theorem (see Section 5.2 in [14]) and was recently applied to the analysis of PnP [214] and several full-gradient RED algorithms in [185]. We extend these results to the block-coordinate setting and provides explicit worst-case convergence rates for BC-RED. Our analysis of BC-RED relates to the analysis of block-coordinate optimization algorithms by Tseng [232], Nesterov [158], Beck and Tetruashvili [21], and Wright [247]. The key difference
of our analysis from those prior works is that it does not require the prior to be expressible in the form of a regularization function, enabling BC-RED to exploit most effective image denoisers, such as those based on deep neural nets.

We consider the following operators \( G_i = \nabla_i g + R_i \) with \( R_i = \tau U_i^T (I - D) \). and proceed in several steps.

(a) Since \( \nabla_i g \) is block \( L_i \)-Lipschitz continuous, it is also block \( \max \)-Lipschitz continuous. Hence, we know from Proposition 2.11 that it is block \( (1/\max) \)-cocoercive. Then from Proposition 2.6, we know that the operator \( (U_i^T - (2/\max) \nabla_i g) \) is block nonexpansive.

(b) From the definition of \( R_i \) and the fact that \( D_i \) is block nonexpansive, we know that \( (U_i^T - (1/\tau) R_i) = D_i \) is block nonexpansive.

(c) From Proposition 2.7, we know that a convex combination of block nonexpansive operators is also block nonexpansive, hence we conclude that

\[
\left(\frac{2}{\max + 2\tau} \right) \left[ U_i^T - \frac{2}{\max} \nabla_i g \right] = \frac{2}{\max + 2\tau} \left[ U_i^T - \frac{1}{\tau} R_i \right]
\]

is block nonexpansive. Then from Proposition 2.6, we know that \( G_i \) is block \( 1/(\max + 2\tau) \)-cocoercive.

(d) Consider any \( x^* \in \text{zer}(G) \), an index \( i \in \{1, \ldots, b\} \) picked uniformly at random, and a single iteration of BC-RED \( x^+ = x - \gamma U_i G_i x \). Define a vector \( h_i := U_i^T (x - x^*) \in \mathbb{R}^{n_i} \).
We then have
\[
\begin{align*}
\|x^+ - x^*\|^2 \\
= \|x - x^* - \gamma U_i G_i x\|^2 \\
= \|x - x^*\|^2 - 2\gamma (G_i x - G_i x^*)^T h_i + \gamma^2 \|G_i x\|^2 \\
\leq \|x - x^*\|^2 - \frac{2\gamma - (L_{\text{max}} + 2\tau)\gamma^2}{L_{\text{max}} + 2\tau} \|G_i x\|^2 \\
\leq \|x - x^*\|^2 - \frac{\gamma}{L_{\text{max}} + 2\tau} \|G_i x\|^2,
\end{align*}
\] (C.1)

where we used \(G_i x^* = U_i^T G x^* = 0\), the block cocoercivity of \(G_i\), and the fact that \(0 < \gamma \leq 1/(L_{\text{max}} + 2\tau)\).

(e) By taking a conditional expectation on both sides and rearranging the terms, we obtain
\[
\frac{\gamma}{L_{\text{max}} + 2\tau} \mathbb{E} \left[ \|G_i x\|^2 | x \right] = \frac{\gamma}{b(L_{\text{max}} + 2\tau)} \sum_{i=1}^{b} \|G_i x\|^2 \\
\leq \mathbb{E} \left[ \|x - x^*\|^2 - \|x^+ - x^*\|^2 | x \right]
\]

(f) Hence by averaging over \(t \geq 1\) iterations and taking the total expectation
\[
\mathbb{E} \left[ \frac{1}{t} \sum_{k=1}^{t} \|G x^{k-1}\|^2 \right] \leq \frac{1}{t} \left[ \frac{b(L_{\text{max}} + 2\tau)}{\gamma} R^2 \right]. \tag{C.2}
\]

The last inequality directly leads to the result.

Remark. Eq. (C.1) implies that, under Assumptions 9.2-9.4, the iterates of BC-RED satisfy
\[
\|x^t - x^*\| \leq \|x^{t-1} - x^*\| \leq \cdots \leq \|x^0 - x^*\| \leq R, \tag{C.3}
\]
which means that the distance of the iterates of BC-RED to \(\text{zer}(G)\) is nonincreasing.
Remark. Our analysis in this section can be significantly strengthened if one adopts an additional assumption that $g$ is strongly convex. This would imply that the algorithm corresponds to repeated applications of a contractive operator [14], which would establish the existence of a unique fixed point $x^* \in \text{zer}(G)$ and the linear convergence of the algorithm. Our focus on the weaker form of convexity of $g$ comes from its broader applicability in computational imaging.

C.2 Proof of Theorem 8.2

The concept of Moreau smoothing is well-known and has been extensively used in other contexts (see for example [258]). Our contribution is to formally connect the concept to RED-based algorithms, which leads to its novel justification as an approximate MAP estimator. The basic review of relevant concepts from proximal optimization is given in Appendix 2.4.4.

For $\tau > 0$, we consider the Moreau envelope of $h$

$$h_{(1/\tau)}(x) := \min_{z \in \mathbb{R}^n} \left\{ \frac{1}{2} \| z - x \|^2 + (1/\tau)h(z) \right\}.$$  

From Proposition 2.18, we know that

$$0 \leq h(x) - \tau h_{(1/\tau)}(x) \leq \frac{G}{2\tau} \quad (C.4)$$

and from Proposition 2.17, we know that

$$\tau \nabla h_{(1/\tau)}(x) = \tau (x - \text{prox}_{(1/\tau)h}(x)). \quad (C.5)$$
Hence, we can express the function $f$ as follows
\[
f(x) = g(x) + h(x) \\
= (g(x) + \tau h_{(1/\tau)}(x)) + (h(x) - \tau h_{(1/\tau)}(x)) \\
= f_{(1/\tau)}(x) + (h(x) - \tau h_{(1/\tau)}(x)),
\]
where $f_{(1/\tau)} := g + \tau h_{(1/\tau)}$. From eq. (C.5), we conclude that a single iteration of BC-RED
\[
x^+ = x - \gamma U_i G_i x \quad \text{with} \quad G_i = U_i^T (\nabla g(x) + \tau \nabla h_{(1/\tau)}(x))
\]
is performing a block-coordinate descent on the function $f_{(1/\tau)}$. From eq. (C.4) and the convexity of the Moreau envelope, we have
\[
f_{(1/\tau)}^* \leq f_{(1/\tau)}(x^*) \leq f_{(1/\tau)}(x) \leq f(x),
\]
where $x \in \mathbb{R}^n$, $x^* \in \text{zer}(G)$. Hence, there exists a finite $f^*$ such that $f(x) \geq f^*$ with $f_{(1/\tau)}^* \leq f^*$. Consider the iteration $t \geq 1$ of BC-RED, then we have that
\[
\mathbb{E}[f(x^t)] - f^* \leq \mathbb{E}[f(x^t)] - f_{(1/\tau)}^* \\
= (\mathbb{E}[f_{(1/\tau)}(x^t)] - f_{(1/\tau)}^*) + \mathbb{E}[(h(x^t) - \tau h_{(1/\tau)}(x^t))] \\
\leq \frac{2b}{\gamma t} R^2 + \frac{G^2}{2\tau},
\]
where we applied (8.6).

The proof of eq. (8.9) is directly obtained by setting $\tau = \sqrt{t}$, $\gamma = L_{max} + 2\sqrt{t}$, and noting that $t \geq \sqrt{t}$, for all $t \geq 1$. 

[244]
Algorithm 13 BC-RED for a quadratic $g$ and block-wise $D$

1: **input:** $x^0 \in \mathbb{R}^n$, $\tau > 0$, and $\gamma > 0$.
2: **initialize:** $r^0 \leftarrow Ax^0 - y$
3: **for** $k = 1, 2, 3, \ldots$ **do**
4:  Choose an index $i_k \in \{1, \ldots, b\}$
5:  $x^k \leftarrow x^{k-1} - \gamma U_{i_k} G_{i_k}(x^{k-1})$
6:  with $G_{i_k}(x^{k-1}) = A_{i_k}^T r^{k-1} + \tau (x_{i_k} - D(x_{i_k}))$.
7:  $r^k \leftarrow r^{k-1} - \gamma A_{i_k} G_{i_k}(x^{k-1})$
8: **end for**

C.3 Coordinate-Friendly Implementations

Theoretical analysis in Section 8.3 of the Chapter 9 suggests that, if $b$ updates of BC-RED (each modifying a single block) are counted as a single iteration, the worst-case convergence rate of BC-RED is expected to be better than that of the full-gradient RED. This fact was empirically validated in Section 8.4, where we showed that in practice BC-RED needs much fewer iterations to converge. However, the overall computational complexity of two methods depends on their per-iteration complexities. In particular, the overall complexity of BC-RED is favorable when its total number of iterations required for convergence offsets the cost of solving the problem in a block-coordinate fashion. As for traditional coordinate descent methods [163, 176], in many problems of interest, the computational complexity of a single update of BC-RED will be roughly $b$ times lower than that of the full-gradient method.

The computational complexity of each block-update will depend on the specifics of the data-fidelity term $g$ and the denoiser $D$ used in the estimation problem. For example, consider the problem where $g(x) = \frac{1}{2} \|Ax - y\|_2^2$. Additionally, suppose that $x$ is such that it is sufficient represent its prior with a block-wise denoiser on each $x_i$, rather than on the full $x$. This situation is very common in image processing, where many popular denoisers are applied block-wise [276]. Then, one can obtain a very efficient implementation of BC-RED, illustrated in Algorithm 13.

[245]
Table C.1: Average SNR achieved by BC-RED for two variants of DnCNN* at different Lipschitz constant (LC) values.

<table>
<thead>
<tr>
<th>Variants of DnCNN*</th>
<th>Radon</th>
<th>Random</th>
<th>Fourier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Radon</td>
<td>Random</td>
<td>Fourier</td>
</tr>
<tr>
<td>Direct</td>
<td>Radon</td>
<td>Random</td>
<td>Fourier</td>
</tr>
<tr>
<td>Free</td>
<td>21.67 24.74</td>
<td>Diverges</td>
<td>29.40 30.35</td>
</tr>
<tr>
<td>LC = 1</td>
<td>19.33 22.98</td>
<td>19.89 20.26</td>
<td>25.06 25.40</td>
</tr>
<tr>
<td>Residual</td>
<td>Free</td>
<td>26.49 27.60</td>
<td>29.39 30.31</td>
</tr>
<tr>
<td>LC = 2</td>
<td>20.88 24.42</td>
<td>26.60 28.12</td>
<td>29.40 30.39</td>
</tr>
</tbody>
</table>

The worst-case complexity of applying $A_i$ and $A_i^T$ is $O(mn_i)$, which means that the cost of $b$ updates such updates for $i \in \{1, \ldots, b\}$ is $O(mn)$. Additionally, if the complexity of $b$ block-wise denoising operations is equivalent or less than the complexity of denoising the full vector (which is generally true for advanced denoisers), then the complexity of $b$ updates of BC-RED will be equivalent or better than a single iteration of the full-gradient RED.

Some of our simulations were conducted using denoisers applied on the full-image and others using patch-wise denoisers. In particular, the convergence simulations in Figure 8.4 relied on the full-image denoisers, in order to use identical denoisers for both RED and BC-RED and be fully compatible with the theoretical analysis. On the other hand, the SNR results in Table B.3, Table C.1, Figure 8.7, and Figure 8.3 rely on block-wise denoisers, where the denoiser input includes an additional 40 pixel padding around the block and the output has the exact size of the block. The padding size was determined empirically in order to have a close match between BC-RED and RED. We have observed that having even larger paddings does not influence the results of BC-RED.
### Figure C.1:

**Left:** The averaged SNR values obtained by BC-RED for the Random matrix with 40 dB noise and *patch-wise* residual DnCNN\(^*\), where the denoiser input includes an additional padding around the patch, while the output has the size of the patch. **Center and Right:** The convergence speed of BC-RED under *patch-wise* residual DnCNN\(^*\) with 40 px padding and the *full-image* residual DnCNN\(^*\). Distance to \(\text{zer}(G)\) – corresponding to the *full-image* denoiser – and SNR are plotted against time. As a reference, we provide the convergence of RED using the full-image DnCNN\(^*\) and BM3D denoisers.

### C.4 Additional Details and Discussion

#### C.4.1 Architecture and Training of DnCNN\(^*\)

We designed DnCNN\(^*\) fully based on DnCNN architecture. The network contains three parts. The first part is a composite convolutional layer, consisting of a normal convolutional layer and a rectified linear units (ReLU) layer. It convolves the \(n_1 \times n_2\) input to \(n_1 \times n_2 \times 64\) features maps by using 64 filters of size \(3 \times 3\). The second part is a sequence of 5 composite convolutional layers, each having 64 filters of size \(3 \times 3 \times 64\). Those composite layers further processes the feature maps generated by the first part. The third part of the network, a single convolutional layer, generates the final output image by convolving the feature maps with a \(3 \times 3 \times 64\) filter. Every convolution is performed with a stride = 1, so that the intermediate feature maps share the same spatial size of the input image. Figure 8.2 visualizes the architectural details. We generated 52000 training examples by adding AWGN to 13000 images (320 × 320) from the NYU fastMRI dataset [259] and cropping them into 4 sub-images of size 160 × 160 pixels. We trained DnCNN\(^*\) to optimize the *mean squared error* by using the Adam optimizer.
C.4.2 Influence of the Lipschitz Constant on Performance

Theorem 8.1 assumes that the denoiser $D$ is nonexpansive. It is relatively straightforward to control the global Lipschitz constants of deep neural nets via spectral normalization [79, 153, 195] and we have empirically tested the influence of nonexpansiveness to the quality of final image recovery.

Table C.1 summarizes the SNR performance of BC-RED for two common variants of DnCNN*. The first variant is trained to learn the direct mapping from a noisy input to a clean image, while the second variant relies on residual learning to map its input to noise (shown in Figure 8.2). To gain insight into the influence of the Lipschitz constant (LC) of a denoiser to its performance as a prior, we trained denoisers with both globally constrained and nonconstrained LCs via the spectral-normalization technique from [195]. For the direct network, we trained DnCNN* with LC = 1, which corresponds to a nonexpansive denoiser. For the residual network, we considered LC = 2, which is a necessary (but not sufficient) condition for the nonexpansiveness. In our simulations, BC-RED converged for all the variants of DnCNN*, except for the direct and unconstrained DnCNN*, which confirms that our theoretical analysis provides only sufficient conditions for convergence. Nonetheless, our simulations reveal the performance loss of the algorithm for the direct and nonexpansive (LC = 1) DnCNN*. On the other hand, the performance of the residual DnCNN* with LC = 2 nearly matches the performance of fully unconstrained networks in all experiments.

C.4.3 Influence of padding in patch-wise denoising

The procedure of block-wise image processing of BC-RED enables one to further reduce the overall computational complexity by using the patch-wise denoisers, where the denoising is performed only on the desired patch instead of the full image. The left table in Figure C.1...
summarizes the averaged SNR values for the \textit{patch-wise} residual DnCNN* corresponding to the paddings of size \{0, 5, 10, 20, 40\} pixels. The lower SNR for 0 px suggests the \textit{non-separability} of DnCNN*; yet, a small 5 px padding is sufficient for matching the performance of the \textit{full-image} DnCNN*. Since the patch-wise denoiser only \textit{approximates} the full-image denoiser, the final accuracy of BC-RED under the patch-wise DnCNN* to \texttt{zer}(G) is $1.92 \times 10^{-7}$, while the accuracy for the full-image DnCNN* is $1.10 \times 10^{-10}$. However, the patch-wise DnCNN* still matches the SNR performance of the full-image DnCNN* and does it faster due to its reduced denoising complexity. The slight SNR improvement for 40 px patch-wise denoising over the full-image denoising is due the fact that $\tau$ parameter of BC-RED was optimized for that case and reused in the rest of experiments. Note also the slow convergence of RED using the full-image BM3D, due to the lower convergence rate and the high complexity of denoising.
Appendix D

Supplement for Chapter 9

Our unified analysis of Async-RED is based on the monotone operator theory [192]. In Appendix D.1, we first clarify our setting for the access of the shared memory. In Appendix D.2, we present the proof of Theorem 9.1 and Theorem 9.2, proving the fixed-point convergence of Async-RED to $\text{zer}(G)$ in both batch and stochastic settings. In Appendix D.3, we include additional technical details and experiments.

D.1 Memory Access without Global Lock

In the setting of Async-RED, multiple cores may simultaneously read and update the blocks $x_i$ in shared memory. We coordinate the memory access of different cores by imposing certain local locks. For example, consider one work cycle of core $c_i$ for updating the block $x_i$. First, a local read lock is imposed to $x_i$ such that only read operations (by $c_i$ or others) can be performed on $x_i$. If, at the same time, other cores want to write $x_i$, then they have to wait until the read lock is released by the last one who finishes reading the block. However, if they want to write other blocks, their operations will not be blocked. Secondly, core $c_i$ evaluates
the RED update on $x_i$, while other cores continuously update $x$. Here, we assume that the number of updates by cores other than $c_i$ is bounded by some positive integer, which is exactly what Assumption 9.1 refers to. After the evaluation finishes, core $c_i$ imposes a local \textit{write} lock, which prevents both read and write by other cores, on $x_i$ and write the block with the computed update. Similarly, other cores have to wait until the lock is released before operating on $x_i$. Finally, when the update finishes, the local lock will be released and core $c_i$ will restart a new cycle. Note that $x$ is never locked \textit{globally} during the full update cycle, and the reads of each block are always consistent.

In order to ensure the consistent read of $x$, we leverage the dual-memory strategy for block coordinate settings proposed in [175] (see section 1.2.1 \textit{‘Block coordinate’}). Its key idea is that, before every write to a block $x_i$, a copy of the old version of the block is kept for reading. In this way, there always exists some state of $x$ in the memory for the cores to access.

**D.2 Proof of Analysis**

In this section, we first present the proof of Theorem 9.1, then followed by the proof of Theorem 9.2. For a review of monotone operators, we refer to Section 2.4 in Chapter 2.

Throughout the proof, we consider the probability space $(\Omega, \mathcal{F}, P)$, where $\Omega$ denotes the sample space, $\mathcal{F}$ the $\sigma$-algebra, and $P$ the probability measure. $x^k$ is a random variable defined in $\mathbb{R}^n$. We use $\| \cdot \|$ to denote the $\ell_2$-norm. We define the sequence of sub $\sigma$-algebra $\{\mathcal{X}^k\}_{k \in \mathbb{N}}$ of $\mathcal{F}$ as

$$\mathcal{X}^k := \sigma(x^0, ..., x^k, \Delta_0, ..., \Delta_k),$$

[251]
where $\sigma$ generates the filtration (smallest $\sigma$-algebra) from $x^0, \ldots, x^k$, and $\Delta_0, \ldots, \Delta_k$. Note that the sequence $\{X^k\}_{k \in \mathbb{N}}$ is such that $X^k \subset X^{k+1}$ for any $k \in \mathbb{N}$. We use $x^*$ to denote some fixed point in the set $\text{zer}(G)$.

### D.2.1 Proof of Theorem 9.1

Our proof needs the following lemma on the RED operator.

**Lemma D.1.** Let Assumption 9.3 and 9.4 hold for $g$ and $D_\sigma$. The composite operator $G$ is $1/(L + 2\tau)$-cocoercive, that is

$$
(G(x) - G(y))^T (x - y) \geq \frac{1}{L + 2\tau} \|G(x) - G(y)\|^2.
$$

**Proof.** This lemma is adapted from Lemma 3 in [213]. Consider the following decomposition

$$
1 - \frac{2}{L + 2\tau} G = \left( \frac{2}{L + 2\tau} \cdot \frac{L}{2} \right) \left[ I - \frac{2}{L} \nabla g \right] + \left( \frac{2}{L + 2\tau} \cdot \frac{2\tau}{2} \right) \left[ I - \frac{1}{\tau} R \right],
$$

where we recall $R = \tau(I - D_\sigma)$. According to Assumption 9.3, $g$ is convex and $\nabla g$ is $L$-Lipschitz continuous. By Proposition 2.11 in Section 2.4.3, $\nabla g$ is $1/L$-cocoercive. Hence, by Proposition 2.6 in Section 2.4.1, $1 - (2/L)\nabla g$ is nonexpansive. Since $D_\sigma = I - (1/\tau)R$, this means that $1 - (1/\tau)R$ is nonexpansive. From Proposition 2.4 in Section 2.4.1, we know that the convex combination of two nonexpansive operators is nonexpansive. Thus, $1 - (2/(L + 2\tau)) G$ is nonexpansive, which also means that $G$ is $1/(L + 2\tau)$-cocoercive according to Proposition 2.6. 

[252]
Now we can start the main proof. Under the fixed stepsize $\gamma > 0$, we begin with the following equations regarding the fixed point $x^* \in \text{zer}(G)$

$$
\begin{align*}
\mathbb{E} \left[ \| x^{k+1} - x^* \|^2 | \mathcal{X}^k \right] \\
= \mathbb{E} \left[ \| x^k - \gamma G_i(\tilde{x}^k) - x^* \|^2 | \mathcal{X}^k \right] \\
= \mathbb{E} \left[ \| x^k - x^* \|^2 | \mathcal{X}^k \right] + \gamma^2 \mathbb{E} \left[ \| G_i(\tilde{x}^k) \|^2 | \mathcal{X}^k \right] + 2\gamma \mathbb{E} \left[ (G_i(\tilde{x}^k))^T (x^* - x^k) | \mathcal{X}^k \right] \\
\leq \| x^k - x^* \|^2 + \frac{\gamma^2}{b} \| G(\tilde{x}^k) \|^2 + \frac{2\gamma}{b} (G(\tilde{x}^k))^T (x^* - x^k). 
\end{align*}
$$

(D.5)

Since $G_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is evaluated on a random block of $x_i$, we have the following conditional expectations

$$
\begin{align*}
\mathbb{E} \left[ (G_i(\tilde{x}^k))^T (x^* - x^k) | \mathcal{X}^k \right] &= \frac{1}{b} \sum_{i=1}^{b} (G_i(\tilde{x}^k))^T (x^* - x^k) = \frac{1}{b} (G(\tilde{x}^k))^T (x^* - x^k) \\
\mathbb{E} \left[ \| G_i(\tilde{x}^k) \|^2 | \mathcal{X}^k \right] &= \frac{1}{b} \sum_{i=1}^{b} \| G_i(\tilde{x}^k) \|^2 = \frac{1}{b} \| G(\tilde{x}^k) \|^2.
\end{align*}
$$

(D.3) and (D.4)

Thus, plugging the above results into (D.2)

$$
\begin{align*}
\mathbb{E} \left[ \| x^{k+1} - x^* \|^2 | \mathcal{X}^k \right] \leq \| x^k - x^* \|^2 + \frac{\gamma^2}{b} \| G(\tilde{x}^k) \|^2 + \frac{2\gamma}{b} (G(\tilde{x}^k))^T (x^* - x^k).
\end{align*}
$$

(D.5)
The term $\dagger$ can be expressed as

$$\frac{2\gamma}{b}(G(\tilde{x}^k))^T(x^* - x^k)$$

$$= \frac{2\gamma}{b}(G(\tilde{x}^k))^T(x^* - \tilde{x}^k) + \sum_{s=k-\Delta_k}^{k-1} (x^s - x^{s+1})$$

$$= \frac{2\gamma}{b}(G(\tilde{x}^k) - G(x^*))^T(x^* - \tilde{x}^k) + \frac{2\gamma}{b}(G(\tilde{x}^k))^T(\sum_{s=k-\Delta_k}^{k} (x^s - x^{s+1}))$$

$$= \frac{2\gamma}{b}(G(\tilde{x}^k) - G(x^*))^T(x^* - \tilde{x}^k) + \frac{2\gamma^2}{b} \sum_{s=k-\Delta_k}^{k-1} G(\tilde{x}^k)^T G_{is}(\tilde{x}^s), \quad (D.6)$$

where in the second line we used the definition of the stale iterate $x^{s+1} = x^s - \gamma G_{is}(\tilde{x}^k)$, and in the third line the fact that $G(x^*) = 0$. By using Lemma D.1, we obtain the upper bound for the first term in equation (D.6)

$$\frac{2\gamma}{b}(G(\tilde{x}^k) - G(x^*))^T(x^* - \tilde{x}^k) \leq -\frac{2\gamma \|G(\tilde{x}^k)\|^2}{b(L + 2\tau)}. \quad (D.7)$$

For the second term in (D.6), we have

$$\frac{2\gamma^2}{b} \sum_{s=k-\Delta_k}^{k-1} G(\tilde{x}^k)^T G_{is}(\tilde{x}^s) \leq \frac{\lambda \gamma^2 \|G(\tilde{x}^k)\|^2}{b} + \sum_{s=k-\Delta_k}^{k-1} \gamma^2 \|G_{is}(\tilde{x}^s)\|^2,$$

$$\leq \frac{\lambda \gamma^2 \|G(\tilde{x}^k)\|^2}{b} + \sum_{s=k-\Delta_k}^{k-1} \gamma^2 \|G(\tilde{x}^s)\|^2, \quad (D.8)$$

where in the first inequality we used the Young’s inequality

$$x_1^T x_2 \leq \frac{1}{2} [\|x_1\|^2 + \|x_2\|^2], \quad (D.9)$$
and in the second inequality we use
\[
\sum_{s=k-\Delta k}^{k-1} \gamma^2 \|G_s(\tilde{x}^s)\|^2 = \sum_{s=k-\Delta k}^{k-1} \|x^s - x^{s+1}\|^2 \leq \sum_{s=k-\lambda}^{k-1} \|x^s - x^{s+1}\|^2 = \sum_{s=k-\lambda}^{k-1} \gamma^2 \|G(\tilde{x}^s)\|^2.
\]

Applying (D.7) and (D.8) in (D.6) yields the overall upper bound for the term (†)
\[
\frac{2\gamma}{b} (G(\tilde{x}^k))^T (x^* - x^k) \leq \frac{(L + 2\tau)\lambda \gamma^2 - 2\gamma}{(L + 2\tau)b} \|G(\tilde{x}^k)\|^2 + \sum_{s=k-\lambda}^{k-1} \frac{\gamma^2 \|G(\tilde{x}^s)\|^2}{b}.
\]

Next, by plugging (D.10) into (D.2) and re-arranging the terms, we obtain the following inequality
\[
\mathbb{E} \left[ \|x^{k+1} - x^*\|^2 |\mathcal{X}^k \right] \\
\leq \|x^k - x^*\|^2 + \sum_{s=k-\lambda}^{k-1} \frac{\gamma^2 \|G(\tilde{x}^s)\|^2}{b} + \frac{(L + 2\tau)(1 + \lambda)\gamma^2 - 2\gamma}{(L + 2\tau)b} \|G(\tilde{x}^k)\|^2.
\]

Taking the total expectation of equation (D.11) and re-arranging the terms yields that
\[
\frac{2\gamma - (L + 2\tau)(1 + \lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \\
\leq \mathbb{E} \left[ \|x^k - x^*\|^2 \right] - \mathbb{E} \left[ \|x^{k+1} - x^*\|^2 \right] + \gamma^2 \sum_{s=k-\lambda}^{k-1} \frac{\mathbb{E} \left[ \|G(\tilde{x}^s)\|^2 \right]}{b}
\]

We then telescope-sum equation (D.12) over \( t > 0 \) iterations to have
\[
\sum_{k=0}^{t-1} \frac{2\gamma - (L + 2\tau)(1 + \lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \\
\leq \mathbb{E} \left[ \|x^0 - x^*\|^2 \right] - \mathbb{E} \left[ \|x^t - x^*\|^2 \right] + \gamma^2 \sum_{k=0}^{t-1} \sum_{s=k-\lambda}^{k-1} \frac{\mathbb{E} \left[ \|G(\tilde{x}^s)\|^2 \right]}{b}
\]
where the index $s$ always start at 0. Under the assumption of consistent read, it is true that
\[
\sum_{k=0}^{t-1} \sum_{s=k-\lambda}^{k-1} \frac{\mathbb{E} \left[ \| G(\tilde{x}^s) \|^2 \right]}{b} \leq \lambda \sum_{k=0}^{t-1} \frac{\mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right]}{b}.
\] (D.14)

In the case of inconsistent read, the above inequality does not always hold. We refer to [175] for a comprehensive analysis for asynchronous block-coordinate methods with inconsistent reads. Now, we rewrite equation (D.13) as
\[
\sum_{k=0}^{t-1} \frac{2\gamma - (L + 2\tau)(1 + 2\lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right] \leq \mathbb{E} \left[ \| x^0 - x^\ast \|^2 \right] - \mathbb{E} \left[ \| x^t - x^\ast \|^2 \right].
\] (D.15)

In order to ensure the convergence, we need the coefficient of $\mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right]$ to be positive. From basic algebra, one feasible range for the stepsize $\gamma$ is
\[
0 < \gamma \leq \frac{1}{(L + 2\tau)(1 + 2\lambda)},
\]
which directly implies that
\[
0 < \frac{\gamma}{(L + 2\tau)b} \leq \frac{2\gamma - (L + 2\tau)(1 + 2\lambda)\gamma^2}{(L + 2\tau)b}.
\]

By simplifying (D.15) with the above result and dropping the negative term, we can derive the following bound for the $\mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right]$ averaged over $t$ iterations
\[
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right] \leq \frac{(L + 2\tau)b}{\gamma t} \mathbb{E} \left[ \| x^0 - x^\ast \|^2 \right] \leq \frac{(L + 2\tau)b}{\gamma t} R^2.
\] (D.16)

The above inequality establishes that the change of the stale iterate $\tilde{x}^k$ converges to zero as $t$ increases. Next, we will use the bound to establish the similar result for the actual iterate $x^k$. 

[256]
We know that \( \| G(x^k) \|^2 \) can be bounded by

\[
\| G(x^k) \|^2 \leq (\| G(x^k) - G(\tilde{x}^k) \| + \| G(\tilde{x}^k) \|)^2
\]

\[
\leq 2\| G(x^k) - G(\tilde{x}^k) \|^2 + 2\| G(\tilde{x}^k) \|^2
\]

\[
\leq 2(L + 2\tau)^2 \| x^k - \tilde{x}^k \|^2 + 2\| G(\tilde{x}^k) \|^2
\]

(D.17)

where in the second inequality we used the Young’s inequality (D.9), and in the third inequality we used the following result implied by Lemma D.1

\[(L + 2\tau)\| x - y \| \geq \| G(x) - G(y) \|.
\]

By expressing the stale iterate \( \tilde{x}^k \), we can write equation (D.17) as

\[
\| G(x^k) \|^2 \leq 2(L + 2\tau)^2 \sum_{s=\lambda}^{k-1} \gamma G_i(\tilde{x}^s) \|^2 + 2\| G(\tilde{x}^k) \|^2
\]

\[
\leq 2\lambda(L + 2\tau)^2 \sum_{s=\lambda}^{k-1} \gamma^2 \| G_i(\tilde{x}^s) \|^2 + 2\| G(\tilde{x}^k) \|^2.
\]

(D.18)

where we use the fact

\[
\| \sum_{i=1}^{n} x_i \|^2 = \sum_{i=1}^{n} \| x_i \|^2 + \sum_{a \neq b} x_a^T x_b \leq \sum_{i=1}^{n} \| x_i \|^2 + \frac{1}{2} \sum_{a \neq b} [\| x_a \|^2 + \| x_b \|^2] = n \sum_{i=1}^{n} \| x_i \|^2
\]
Taking the expectation of equation (D.18) leads to

\[
\mathbb{E} \left[ \| G(x^k) \|^2 \right] \\
\leq 2\lambda (L + 2\tau)^2 \sum_{s=k-\lambda}^{k-1} \gamma^2 \mathbb{E} \left[ \| G_s(\bar{x}^s) \|^2 \right] + 2\mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right] \\
\leq 2\lambda (L + 2\tau)^2 \sum_{s=k-\lambda}^{k-1} \gamma^2 \mathbb{E} \left[ \| G_s(\bar{x}^s) \|^2 \right] \frac{b}{b} + 2\mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right], \quad (D.19)
\]

By averaging (D.19) over \( t > 0 \) iterations, we obtain that

\[
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \\
\leq \frac{2\lambda (L + 2\tau)^2}{t} \sum_{k=0}^{t-1} \sum_{s=k-\lambda}^{k-1} \gamma^2 \mathbb{E} \left[ \| G_s(\bar{x}^s) \|^2 \right] \frac{b}{b} + 2\sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right] \\
\leq \frac{2\lambda^2 (L + 2\tau)^2}{t} \sum_{k=0}^{t-1} \gamma^2 \mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right] \frac{b}{b} + 2\sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right] \quad (D.20)
\]

where we again used result in (D.14) in the last inequality. Re-arranging the terms in (D.20) yields

\[
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \left[ \frac{2\lambda^2 (L + 2\tau)^2}{b} \gamma^2 + 2 \right] \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\bar{x}^k) \|^2 \right] \quad (D.21)
\]

We plug the result in (D.16) into (D.21) and obtain

\[
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \left[ \frac{2\lambda^2 (L + 2\tau)^2}{b} \gamma^2 + 2 \right] \frac{(L + 2\tau)b}{\gamma t} R^2, \quad (D.22)
\]

Since it is always true that

\[
\gamma \leq \frac{1}{(L + 2\tau)(1 + 2\lambda)} \leq \frac{1}{(L + 2\tau)(1 + \lambda)}.
\]

[258]
we can simplify the bound by using the above inequality related to the stepsize $\gamma$

$$\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{2\lambda^2}{(1+\lambda)^2b} + 2 \right] \frac{(L + 2\tau)b}{\gamma t} R^2. \quad (D.23)$$

Let $D = 2\lambda^2/(1+\lambda)^2$, and we derive the desired result.

$$\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \frac{(L + 2\tau)b}{\gamma t} R^2. \quad (D.24)$$

### D.2.2 Proof of Theorem 9.2

We prove Theorem 9.2 by following the procedure in the proof of Theorem 9.1 with the adaptation to the block stochastic operator $\hat{G}_i$. In the key steps, we will highlight the difference between the two proofs. In addition to Lemma D.1, our second proof requires the following lemma related to the statistical properties of $\hat{G}$.

**Lemma D.2.** Let Assumption 9.3 and 9.4 hold for $g$ and $D_\sigma$. Then, we can establish the following statements for operator $\hat{G}$

$$\mathbb{E} \left[ \hat{G}(x) \right] = G(x), \quad \mathbb{E} \left[ \|\hat{G}(x) - G(x)\|^2 \right] \leq \frac{\nu^2}{w},$$

which further implies that

$$\mathbb{E} \left[ \|\hat{G}(x)\|^2 \right] \leq \frac{\nu^2}{w} + \|G(x)\|^2.$$

**Proof.** Since the the stochasticity happens only in the evaluation of the gradient, it is straightforward to see that

$$\mathbb{E} \left[ \hat{G}(x) \right] = \mathbb{E} [\hat{\nabla}g(x)] + D_\sigma(x) = G(x),$$

[259]
Similarly, we have that
\[
\mathbb{E} \left[ \| \hat{G}(x) - G(x) \|^2 \right] = \mathbb{E} \left[ \| \hat{\nabla} g(x) - \nabla g(x) \|^2 \right] \leq \frac{\nu^2}{w}.
\]

Given that \( \text{Tr}(\mathbb{E}[X^T X]) = \text{Tr} (\text{Cov}[X]) + \text{Tr}(\mathbb{E}[X] |^{2} ) \), we obtain that
\[
\mathbb{E} \left[ \| \hat{G}(x) \|^2 \right] = \mathbb{E} \left[ \| \hat{G}(x) - G(x) \|^2 \right] + \mathbb{E} \left[ \hat{G}(x) \right]^2 \leq \frac{\nu^2}{w} + \| G(x) \|^2,
\]
where we let \( \mathbb{E} \left[ \| \hat{G}(x) \|^2 \right] := \mathbb{E} \left[ \hat{G}(x)^\top \hat{G}(x) \right] \). Note that \( \text{Tr}(\cdot) \) and \( \text{Cov}(\cdot) \) denote the computation of the trace and covariance of a matrix and a vector, respectively.

Now we start the proof. Similar as (D.2), we write that
\[
\mathbb{E} \left[ \| x^{k+1} - x^* \|^2 | \mathcal{X}^k \right] = \mathbb{E} \left[ \| x^k - \gamma \hat{G}_i(\tilde{x}^k) - x^* \|^2 | \mathcal{X}^k \right] = \mathbb{E} \left[ \| x^k - x^* \|^2 | \mathcal{X}^k \right] + \gamma^2 \mathbb{E} \left[ \| \hat{G}_i(\tilde{x}^k) \|^2 | \mathcal{X}^k \right] + 2\gamma \mathbb{E} \left[ (\hat{G}_i(\tilde{x}^k))^\top (x^* - x^k) | \mathcal{X}^k \right] \quad (D.25)
\]

Here, the conditional expectation is taken for \( \hat{G}_i(x) = U_i U_i^\top \hat{G}(x) \). By using Lemma D.2, we can compute conditional expectations as
\[
\mathbb{E} \left[ (\hat{G}_i(\tilde{x}^k))^\top (x^* - x^k) | \mathcal{X}^k \right] = \frac{1}{b} \mathbb{E} \left[ (\hat{G}(\tilde{x}^k))^\top (x^* - x^k) | \mathcal{X}^k \right] = \frac{1}{b} (G(\tilde{x}^k))^\top (x^* - x^k) \quad (D.26)
\]
and
\[
\mathbb{E} \left[ \| \hat{G}_i(\tilde{x}^k) \|^2 | \mathcal{X}^k \right] = \frac{1}{b} \mathbb{E} \left[ \| \hat{G}(\tilde{x}^k) \|^2 | \mathcal{X}^k \right] \leq \frac{\nu^2}{wb} + \frac{\| G(\tilde{x}^k) \|^2}{b} \quad (D.27)
\]
where we first compute the expectation corresponding to the randomized block and then the expectation for the stochastic measurements. We note that the expectation of the cross term (D.26) remains the same as the result in (D.3), while the expectation in (D.27) has one
extra term related to the norm variance of the stochastic operator compared with (D.4). As we shall see in the future steps, the difference in the expectation of the operator’s squared norm leads to the most modifications. Using the above results in equation (D.25) yields that

\[
\mathbb{E} \left[ \|x^{k+1} - x^*\|^2 | \mathcal{X}^k \right] \\
\leq \|x^k - x^*\|^2 + \frac{\gamma^2}{b} \|G(\bar{x}^k)\|^2 + \frac{\gamma^2 \nu^2}{wb} + \frac{2\gamma}{b} (G(\bar{x}^k))^T (x^* - x^k). \quad (D.28)
\]

By following (D.6), we can express the term \((\dagger)\) as

\[
\frac{2\gamma}{b} (G(\bar{x}^k))^T (x^* - x^k) \\
= \frac{2\gamma}{b} (G(\bar{x}^k) - G(x^*))^T (x^* - \bar{x}^k) + \frac{2\gamma^2}{b} \sum_{s=k-\Delta_k}^{k-1} G(\bar{x}^k)^T \hat{G}_{i_s}(\check{x}^s), \quad (D.29)
\]

The upper bound of the first term is the same as shown in (D.7), which is

\[
\frac{2\gamma}{b} (G(\bar{x}^k) - G(x^*))^T (x^* - \bar{x}^k) \leq -\frac{2\gamma \|G(\bar{x}^k)\|^2}{b(L + 2\tau)}. \quad (D.30)
\]

Similarly, our second term is bounded by

\[
\frac{2\gamma^2}{b} \sum_{s=k-\Delta_k}^{k-1} G(\bar{x}^k)^T \hat{G}_{i_s}(\check{x}^s) \leq \frac{\lambda \gamma^2 \|G(\bar{x}^k)\|^2}{b} + \sum_{s=k-\lambda}^{k-1} \frac{\gamma^2 \|\hat{G}(\check{x}^s)\|^2}{b}, \quad (D.31)
\]

where we used the Young’s inequality (D.9) together with the fact that

\[
\sum_{s=k-\Delta_k}^{k-1} \|\hat{G}_{i_s}(\check{x}^s)\|^2 \leq \sum_{s=k-\lambda}^{k-1} \|\hat{G}_{i_s}(\check{x}^s)\|^2 \leq \sum_{s=k-\lambda}^{k-1} \|\hat{G}(\check{x}^s)\|^2.
\]
Equation (D.30) and (D.31) together establish the overall upper bound for the term (†)

\[
\frac{2\gamma}{b} (G(\tilde{x}^k))^\top (x^* - x^k) \leq \frac{(L + 2\tau)\lambda \gamma^2 - 2\gamma}{(L + 2\tau)b} \|G(\tilde{x}^k)\|^2 + \sum_{s=k-\lambda}^{k-1} \gamma^2 \|\hat{G}(\tilde{x}^s)\|^2 b. \tag{D.32}
\]

By plugging (D.32) into (D.25) and re-arranging the terms, we obtain that

\[
\mathbb{E} \left[ \|x^{k+1} - x^*\|^2 | x^k \right] \\
\leq \|x^k - x^*\|^2 + \frac{\gamma^2 \nu^2}{w b} + \sum_{s=k-\lambda}^{k-1} \frac{\gamma^2 \|\hat{G}(\tilde{x}^s)\|^2}{b} + \frac{(L + 2\tau)(1 + \lambda)\gamma^2 - 2\gamma}{(L + 2\tau)b} \|G(\tilde{x}^k)\|^2. \tag{D.33}
\]

Taking the total expectation of equation (D.33) and re-arranging the terms yields that

\[
\frac{2\gamma - (L + 2\tau)(1 + \lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \\
\leq \mathbb{E} \left[ \|x^k - x^*\|^2 \right] - \mathbb{E} \left[ \|x^{k+1} - x^*\|^2 \right] + \frac{\gamma^2 \nu^2}{w b} + \gamma^2 \sum_{s=k-\lambda}^{k-1} \left[ \frac{\nu^2}{w b} + \frac{\mathbb{E} \left[ \|\hat{G}(\tilde{x}^s)\|^2 \right]}{b} \right] \tag{D.34}
\]

where we use the following inequality derived by using the law of total expectation and Lemma D.2

\[
\mathbb{E} \left[ \|\hat{G}(\tilde{x}^*)\|^2 \right] = \mathbb{E} \left[ \mathbb{E} \left[ \|\hat{G}(\tilde{x}^*)\|^2 | x^s \right] \right] \leq \frac{\nu^2}{w} + \mathbb{E} \left[ \|G(\tilde{x}^*)\|^2 \right]. \tag{D.35}
\]

We telescope-sum equation (D.34) over \( t > 0 \) iterations to obtain

\[
\sum_{k=0}^{t-1} \frac{2\gamma - (L + 2\tau)(1 + \lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \\
\leq \mathbb{E} \left[ \|x^0 - x^*\|^2 \right] - \mathbb{E} \left[ \|x^t - x^*\|^2 \right] + \sum_{k=0}^{t-1} \frac{\gamma^2 \nu^2}{w b} + \gamma^2 \sum_{k=0}^{t-1} \sum_{s=k-\lambda}^{k-1} \left[ \frac{\nu^2}{w b} + \frac{\mathbb{E} \left[ \|\hat{G}(\tilde{x}^s)\|^2 \right]}{b} \right] \tag{D.36}
\]
By applying the same relaxation trick in (D.14) to (D.36)

$$
\sum_{k=0}^{t-1} \sum_{s=k-\lambda}^{k-1} \left[ \frac{\nu^2}{wb} + \frac{\mathbb{E} \left[ \|G(\tilde{x}^s)\|^2 \right]}{b} \right] \leq \lambda \sum_{k=0}^{t-1} \left[ \frac{\nu^2}{wb} + \frac{\mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right]}{b} \right],
$$

(D.37)

we then have that

$$
\sum_{k=0}^{t-1} \frac{2\gamma - (L + 2\tau)(1 + 2\lambda)\gamma^2}{(L + 2\tau)b} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \leq \mathbb{E} \left[ \|x^0 - x^*\|^2 \right] + \frac{(1 + \lambda)\gamma^2\nu^2}{wb} \cdot t,
$$

(D.38)

where we dropped the negative term. Recall that if $\gamma$ is in the range $\gamma \in (0, 1/((L + 2\tau)(1 + 2\lambda))]$, we have the inequality

$$
\frac{\gamma}{(L + 2\tau)b} \leq \frac{2\gamma - (L + 2\tau)(1 + 2\lambda)\gamma^2}{(L + 2\tau)b}.
$$

By relaxing the coefficient in the lefthand side, dividing the inequality by $t$, and re-arranging the terms, we obtain the convergence in terms of the stale iterate $\tilde{x}^k$

$$
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right] \leq \frac{(L + 2\tau)b}{\gamma t} \left[ \mathbb{E} \left[ \|x^0 - x^*\|^2 \right] + \frac{(1 + \lambda)\gamma^2\nu^2}{wb} \cdot t \right]
$$

$$
\leq \frac{(L + 2\tau)b}{\gamma t} R^2 + \frac{\gamma}{w} C
$$

(D.39)

where we used Assumption 9.2 and let $C = (L + 2\tau)(1 + \lambda)\nu^2$. Compared with the result in equation (D.16), equation (D.39) has the extra term related to the variance of $\hat{G}_i(x)$. Next, we establish the convergence in terms of actual iterate $x^k$. Following the steps from (D.17) to (D.19), we directly obtain the inequality related to $\hat{G}_i(\tilde{x})$

$$
\mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq 2\lambda(L + 2\tau)^2 \sum_{s=k-\lambda}^{k-1} \gamma^2 \mathbb{E} \left[ \|\hat{G}_i(\tilde{x}^s)\|^2 \right] + 2\mathbb{E} \left[ \|G(\tilde{x}^k)\|^2 \right]
$$

(D.40)
By using the the result in (D.35), we derive from (D.40) that

\[ \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq 2\lambda \left( L + 2\tau \right)^2 \sum_{s=k-L}^{k-1} \gamma^2 \left[ \frac{\nu^2}{wb} + \frac{\mathbb{E} \left[ \| G(\tilde{x}^s) \|^2 \right]}{b} \right] + 2\mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right]. \quad (D.41) \]

By averaging (D.40) over \( t > 0 \) iterations, we obtain that

\[ \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq 2\lambda \left( L + 2\tau \right)^2 \sum_{k=0}^{t-1} \sum_{s=k-L}^{k-1} \gamma^2 \left[ \frac{\nu^2}{wb} + \frac{\mathbb{E} \left[ \| G(\tilde{x}^s) \|^2 \right]}{b} \right] + \frac{2}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right] \]

\[ \leq \frac{2\lambda^2 \left( L + 2\tau \right)^2}{t} \sum_{k=0}^{t-1} \gamma^2 \left[ \frac{\nu^2}{wb} + \frac{\mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right]}{b} \right] + \frac{2}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right] \quad (D.42) \]

where we again used the relaxation (D.37) in the last inequality. Re-arranging the terms in (D.42) yields

\[ \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \frac{2\lambda^2 \left( L + 2\tau \right)^2 \cdot \nu^2}{wb} \cdot \gamma^2 + \left[ \frac{2\lambda^2 \left( L + 2\tau \right)^2}{b} \cdot \gamma^2 + 2 \right] \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(\tilde{x}^k) \|^2 \right] \quad (D.43) \]

We plug the result in (D.39) into (D.43) and obtain

\[ \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \frac{2\lambda^2 \left( L + 2\tau \right)^2 \cdot \nu^2}{wb} \cdot \gamma^2 + \left[ \frac{2\lambda^2 \left( L + 2\tau \right)^2}{b} \cdot \gamma^2 + 2 \right] \left[ \frac{\left( L + 2\tau \right)b \cdot \gamma t R^2}{\gamma} + \frac{\gamma C}{w} \right] \quad (D.44) \]

Similarly, we can use the fact

\[ \gamma \leq \frac{1}{\left( L + 2\tau \right)(1 + \lambda)}. \]

[264]
to simplify the bound in (D.44)

\[
\frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \\
\leq \frac{2\lambda^2 (L + 2\tau)^2 \cdot \nu^2}{wb} \cdot \frac{1}{(L + 2\tau)(1 + \lambda)} \cdot \gamma + \left[ \frac{2\lambda^2}{(1 + \lambda)^2b} + 2 \right] \left[ \frac{(L + 2\tau)b}{\gamma t} R^2 + \frac{\gamma}{w} C \right]
\]
\]

\[
= \frac{2\lambda^2}{(1 + \lambda)^2b} \cdot \frac{(L + 2\tau)(1 + \lambda)\nu^2}{w} \cdot \gamma + \left[ \frac{2\lambda^2}{(1 + \lambda)^2b} + 2 \right] \left[ \frac{(L + 2\tau)b}{\gamma t} R^2 + \frac{\gamma}{w} C \right]
\]
\]

\[
= \frac{2\lambda^2}{(1 + \lambda)^2b} \cdot \frac{C}{w} \gamma + \left[ \frac{2\lambda^2}{(1 + \lambda)^2b} + 2 \right] \left[ \frac{(L + 2\tau)b}{\gamma t} R^2 + \frac{\gamma}{w} C \right]
\]
\]

(D.45)

where we recall \( C = (L + 2\tau)(1 + \lambda)\nu^2 \). Let \( D = 2\lambda^2/(1 + \lambda)^2 \) and we can derive the result of Theorem 9.2

\[
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \left[ \frac{(L + 2\tau)b}{\gamma t} R^2 + \left[ \frac{2D}{b} + 2 \right] \frac{\gamma}{w} C \right]
\]
\]

(D.46)

which immediately implies the result in remark 1 by setting \( \gamma = 1/\sqrt{wt} \)

\[
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \frac{1}{t} \sum_{k=0}^{t-1} \mathbb{E} \left[ \| G(x^k) \|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \left[ \frac{(L + 2\tau)b}{\sqrt{wt}} R^2 + \left[ \frac{2D}{b} + 2 \right] \frac{C}{\sqrt{wt}} \right]
\]
\]

(D.47)

From basic algebra, we can derive the condition for \( \lambda \)

\[
\frac{1}{\sqrt{wt}} \leq \frac{1}{(L + 2\tau)(1 + 2\lambda)} \quad \Rightarrow \quad \lambda \leq \frac{1}{2} \left[ \frac{\sqrt{wt}}{L + 2\tau} - 1 \right].
\]

[265]
Figure D.1: Illustration of the architecture of DnCNN used in all experiments. The neural net is trained to remove the AWGN from its noisy input image. We also constrains the Lipschitz constant of $R_\sigma$ to be smaller than 2 by using the spectral normalization technique in [196]. This provides a necessary condition for the satisfaction of Assumption 9.4.

Figure D.2: Six test images used in the experiments on CS. From the left to right, there are *cameraman, house, pepper, starfish, butterfly, and jet.*

### D.3 Additional Technical Details

This section presents several technical details. Section D.3.1 presents the architecture and training of our DnCNN prior. Section D.3.2 provides extra details and validations that compliment the experiments in Section 9.4 of the main chapter.

#### D.3.1 Architecture and Training of the DnCNN Prior

Our denoiser follows the standard architecture of DnCNN [262]. Figure D.1 visualizes the architectural details of the DnCNN prior used in our experiments. Similar priors are extensively used in various PnP and RED algorithms [193, 213, 263]. In total, the network contains 7 layers, of which the first 6 layers consist of a convolutional layer and a rectified
linear unit (ReLU), while the last layer contains only a convolution operation. A skip connection from the input to the output is used to enforce the residual network $R_\sigma$ to predict the noise residual. The output images of the first 6 layers have 64 feature maps, while that of the last layer is a single-channel image. We set all convolutional kernels to be $3 \times 3$ with stride 1, which indicates that intermediate images have the same spatial size as the input image. We generated 44700 training examples by adding AWGN to 400 images from the BSD400 dataset [140] and extracting small patches of $128 \times 128$ pixels with stride 30. Our DnCNN denoiser is trained to optimize the mean squared error by using the Adam optimizer [114].

Different approaches have been used to constrain the Lipschitz constant (LC) of the denoising prior [193, 213]. We adopt the spectral normalization technique in [196] to control the LC of our DnCNN prior. In the training, we constrain the residual network $R_\sigma$ such that its LC is smaller than 2. Since the non-expansiveness of $D_\sigma$ implies that $R_\sigma$ has $LC \leq 2$, this provides a necessary condition for $D_\sigma$ to satisfy Assumption 9.4 [213].
Table D.1: SNR values obtained by Async-RED-BG using different block sizes on CS task.

<table>
<thead>
<tr>
<th>Block size</th>
<th>cameraman</th>
<th>house</th>
<th>pepper</th>
<th>starfish</th>
<th>butterfly</th>
<th>jet</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>27.77</td>
<td>30.92</td>
<td>29.60</td>
<td>28.23</td>
<td>28.89</td>
<td>28.90</td>
<td>29.05</td>
</tr>
<tr>
<td>80</td>
<td>27.75</td>
<td>30.95</td>
<td>29.58</td>
<td>28.28</td>
<td>28.78</td>
<td>28.76</td>
<td>29.01</td>
</tr>
<tr>
<td>60</td>
<td>27.74</td>
<td>30.95</td>
<td>29.65</td>
<td>28.12</td>
<td>28.85</td>
<td>28.71</td>
<td>28.99</td>
</tr>
</tbody>
</table>

Table D.2: SNR values obtained by Async-RED-SG using different minibatch sizes on CS task.

<table>
<thead>
<tr>
<th>minibatch size</th>
<th>cameraman</th>
<th>house</th>
<th>pepper</th>
<th>starfish</th>
<th>butterfly</th>
<th>jet</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1120</td>
<td>27.00</td>
<td>30.56</td>
<td>28.99</td>
<td>26.98</td>
<td>27.83</td>
<td>27.09</td>
<td>28.03</td>
</tr>
<tr>
<td>2240</td>
<td>27.31</td>
<td>30.78</td>
<td>29.01</td>
<td>27.64</td>
<td>28.29</td>
<td>28.03</td>
<td>28.51</td>
</tr>
<tr>
<td>3360</td>
<td>27.56</td>
<td>30.87</td>
<td>29.53</td>
<td>28.24</td>
<td>28.72</td>
<td>28.71</td>
<td>28.93</td>
</tr>
</tbody>
</table>

D.3.2 Extra Details and Validations

All experiments are run on the server equipped with 32 Intel(R) Xeon(R) CPU E5-2620 v4 processors of 3.2 GHz and 264 GBs of DDR memory. We trained all neural nets using NVIDIA RTX 2080 GPUs. We define the SNR (dB) used in the experiments as

$$\text{SNR}(\hat{x}, x) \triangleq 20 \log_{10} \left( \frac{\|x\|_2}{\|x - \hat{x}\|_2} \right)$$

where \(\hat{x}\) represents the reconstructed image and \(x\) denotes the ground truth. Note that our experimental setup satisfies Assumption 9.1-9.3 but provides a necessary condition for Assumption 9.4.
Table D.3: SNR values obtained by Async-RED-BG using different pad size on CS task.

<table>
<thead>
<tr>
<th>Pad size</th>
<th>camera-man</th>
<th>house</th>
<th>pepper</th>
<th>starfish</th>
<th>butterfly</th>
<th>jet</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>27.64</td>
<td>30.86</td>
<td>29.44</td>
<td>28.09</td>
<td>28.64</td>
<td>28.69</td>
<td><strong>28.89</strong></td>
</tr>
<tr>
<td>10</td>
<td>27.72</td>
<td>30.99</td>
<td>29.56</td>
<td>28.25</td>
<td>28.79</td>
<td>28.73</td>
<td><strong>29.00</strong></td>
</tr>
<tr>
<td>20</td>
<td>27.72</td>
<td>30.99</td>
<td>29.57</td>
<td>28.27</td>
<td>28.79</td>
<td>28.75</td>
<td><strong>29.01</strong></td>
</tr>
<tr>
<td>full</td>
<td>27.75</td>
<td>30.95</td>
<td>29.58</td>
<td>28.28</td>
<td>28.78</td>
<td>28.76</td>
<td><strong>29.01</strong></td>
</tr>
</tbody>
</table>

Figure D.2 shows the six test images used in the experiments of CS. They are resized to the size of $240 \times 240$ pixels by using the Matlab function `imresize`. As demonstrated in the middle figure in Figure 9.3, Async-RED-SG converges faster than Async-RED-BG given a fixed amount of time. This is further visualized in Figure D.3, where each algorithm is run for roughly 700 seconds. Since Async-RED-SG uses only one-fourth of the total measurements, the per-iteration complexity is lower than Async-RED-BG, leading to the faster convergence speed. In particular, the final SNR value obtained by Async-RED-SG is roughly 2 dB higher than Async-RED-BG. Additionally, both Async-RED-BG/SG achieves significantly better results than Sync-RED and GM-RED due to their adoption of asynchronous updates.

Table D.1 and D.2 illustrate the evolution of the reconstruction performance as the block size $b$ and minibatch size $w$ changes, respectively. Table D.1 summarizes the SNR values obtained for three block sizes $b \in \{60, 80, 120\}$. Async-RED-BG achieves almost the same SNR values under these settings. Table D.2 summarizes the SNR values for three minibatch sizes $w \in \{1120, 2240, 3360\}$, which corresponds to $1/4$, $1/2$, and $3/4$ of the full batch. As $w$ increases, the final SNR performance improves, which is consistent with our theory. On the other hand, the error term due to stochastic processing in Theorem 9.2 is also proportional to the step size $\gamma$, which means that by using smaller $\gamma$, Async-RED-SG can approximate GM-RED as accurately as desired. However, a reduction in $\gamma$ would also lead to slower

[269]
convergence. One thus needs to tradeoff the desired accuracy against the desired speed to select a suitable configuration for Async-RED-SG.

The benefit of Async-RED is fully explored when the denoiser acts like a *block-wise denoiser*, which means that it can perform denoising on blocks as effective as on the full image. A simple strategy for making denoisers block-effective is to include additional neighboring pixels at the input, but use the exact block size at the output. Table D.3 reports the results of experimenting with the idea of *input padding* for DnCNN. The results indicate that by including a small number of pixels around each block at the input of DnCNN, one can match the performance of using the full image at the input of DnCNN.

The test image used in the experiment of CT is selected from the dataset of human protein atlas [243]. We download 51 images that have the size of 3000 × 3000 pixels. We select one image for test, which is cropped to 800 × 800 pixels. We extract 39000 patches from the rest 50 images to train five specific DnCNN denoisers for the removal of AWGN with \( \sigma \in \{5, 10, 15, 20, 25\} \). We report the result that has the highest SNR values. The Radon matrix used in the experiments corresponds to 180 angles with 1131 detectors. We synthesize the measurements by multiplying the Radom matrix with the vectorized image and add AWGN corresponding to 70 dB input SNR. In all tests, Async-RED-SG randomly uses the measurements of 60 angles at each iteration, while Async-RED-BG uses the entire measurement set. Figure D.4 plots SNR against the iteration number for Async-RED-BG/SG, Async-RED, and Gm-RED. Due to the lower per-iteration complexity, Async-RED-SG achieves the highest SNR value within the time budget of 1 hour. Figure D.5 provides a corresponding visual comparison between these methods. As reference, we also include the proximal gradient method with total variation regularizer (PGM-TV). The visual result of each method is obtained by running the algorithm with a time budget of 1 hour. Specifically, the per-iteration time cost of Async-RED-BG/SG, Sync-RED, Gm-RED, PGM-TV are
Figure D.4: Convergence Illustration of Async-RED-BG/SG and Gm-RED for CT reconstruction with a time budget of 1 hour. Here, ASYNC-RED-SG randomly uses one-third of the total measurements at every iteration.

5.23, 3.21, and 13.13, 19.19, and 44.74 seconds, respectively. The results clearly demonstrate that Async-RED are indeed effective and efficient for a realistic, nontrivial imaging task on a large-scale image.
Figure D.5: Visualization of the reconstructed CT images by PGM-TV, Gm-RED, Sync-RED, and Async-RED-BG/SG. Each algorithm is run with a time budget of 1 hour. The colormap is adjusted for the best visual quality.
Appendix E

Supplement for Chapter 10

E.1 IDT forward model

We adopt the traditional IDT forward model where the scattering events throughout the sample volume are characterized using the first Born approximation [246]. Hence, a 3D sample is discretized into a stack of 2D slices along the axial direction, and an individual measurement matrix is derived for each slice to linearly map the 2D permittivity contrast to the intensity measurements. Given the $p^{th}$ illumination, the discrete IDT system is expressed as

$$y_p = \sum_{q=1}^{Q} A_{p,q} \Delta \epsilon_q + e_0,$$  

(E.1)

where $y_p \in \mathbb{R}^m$ is the intensity measurement with the background intensity removed, $\Delta \epsilon \in \mathbb{C}^n$ is permittivity contrast, $q = 1, \ldots, Q$ is the index of the axial slices, $\{A_{p,q}\}$ are the measurement matrices, and $\{e_p\}$ represent the noise. From the IDT principle [131], every $A_{p,q}$ is a composition of 2D (inverse) Fourier transform $\mathcal{F}$ ($\mathcal{F}^{-1}$) and a transfer function.
\[
A_{p,q} = \mathcal{F}^{-1} H_{p,q} \mathcal{F}.
\] (E.2)

From the equation, we note that \(H_{p,q}\) fully characterizes the permittivity-intensity mapping. Next, we describe how to compute the transfer function for each IDT modality used in our experiments.

**Dense and annular IDT.** In dense and annular IDT, the LED is turned on in a one-by-one manner. For a single-LED plane wave illumination, the analytical expression of the phase and absorption TFs \([124, 131]\) can be expressed as

\[
H_{p,q}^{ph}(u) = j \frac{k_0^2}{2} S(u_p) \left( P^*(u)P(u - u_p) \frac{\exp\{-j[\eta(u - u_p) - \eta_i] \cdot q \Delta z\}}{\eta(u - u_p)} - P(-u_p)P^*(u + u_p) \frac{\exp\{j[\eta(u + u_p) - \eta_i] \cdot q \Delta z\}}{\eta(u + u_p)} \right) \quad (E.3a)
\]

\[
H_{p,q}^{ab}(u) = -\frac{k_0^2}{2} S(u_p) \left( P^*(u_p)P(u - u_p) \frac{\exp\{-j[\eta(u - u_p) - \eta_i] \cdot q \Delta z\}}{\eta(u - u_p)} - P(-u_p)P^*(u + u_p) \frac{\exp\{j[\eta(u + u_p) - \eta_i] \cdot q \Delta z\}}{\eta(u + u_p)} \right) \quad (E.3b)
\]

where \(u\) denotes the lateral wave vector, \(j\) is the imaginary unit, \(k_0 = 2\pi/\lambda\) is the wave number, \(\lambda\) is the illumination wavelength, \(S\) is the source function, \(P\) is the objective pupil function, \(P^*\) is the conjugate transpose of \(P\), \(u_i\) is the \(i^{th}\) lateral illumination wave vector, \(\eta(u) = \sqrt{k_0^2 - |u|^2}\) is the axial wave vector, \(\eta_p(u_p) = \sqrt{k_0^2 - |u_p|^2}\) is the illumination axial wave vector, and \(\Delta z\) is the axial sampling spacing (i.e., slice spacing).

We obtain the matrix expression of these TFs by performing discretization and normalization: \(H_{p,q}^{ph} = H_{p,q}^{ph}/I_p\) and \(H_{p,q}^{ab} = H_{p,q}^{ab}/I_p\), where \(I_p\) denote the squared modulus of the incident light field. Here, the division denotes element-wise operation. A full expression of \(H_p\) for the...
$p^{th}$ illumination is given as

$$H_p = \begin{pmatrix} H_{p,1} & \cdots & H_{p,Q} \\ H_{ab,1} & \cdots & H_{ab,Q} \end{pmatrix}. \quad (E.4)$$

**Multiplexed IDT.** We consider the intensity measurement of a multiplexed illumination $y_{p}^{\text{mul}}$ as a summation of the measurements of individual sub-illuminations $y_{p}^{\text{sub}}$ [143]. The forward model in the frequency domain can be expressed as

$$\hat{y}_{p}^{\text{mul}} = \sum_{w_p=1}^{W} \hat{y}_{w_p}^{\text{sub}} = \sum_{w_p=1}^{W} \sum_{q=1}^{Q} H_{w_p,q} \Delta \hat{e}_q + \hat{e}, \quad (E.5)$$

where $\hat{\cdot}$ denotes the vector in the Fourier space, $W > 0$ is the total number of sub-illuminations, and $H_{p}^{\text{mul}}$ denotes the TF of a single multiplexed illumination. By regrouping and reordering the TFs of each sub-illumination, we can derive the formulation of $H_{p}^{\text{mul}}$ as follows

$$H_{p}^{\text{mul}} = \begin{pmatrix} H_{1,1}^{\text{ph}} & \cdots & H_{1,Q}^{\text{ph}} \\ \vdots & \ddots & \vdots \\ H_{W,1}^{\text{ph}} & \cdots & H_{W,Q}^{\text{ph}} \\ H_{1,1}^{\text{ab}} & \cdots & H_{1,Q}^{\text{ab}} \\ \vdots & \ddots & \vdots \\ H_{W,1}^{\text{ab}} & \cdots & H_{W,Q}^{\text{ab}} \end{pmatrix}. \quad (E.6)$$

[275]
E.2 Block-wise Training of DeCAF

Although the memory footprint of the MLP in DeCAF is small, its optimization involves the computation of the IDT forward model, which scales linearly by the total of number of the input coordinates, and thus can easily exceed the memory limit of a graphic processing unit (GPU). To address this, we proposed a block-wise training approach to reduce the GPU memory usage during optimization.

**General block-wise procedure.** The block-wise procedure (see Figure E.1 for visual illustration) can be described as follows: During optimization, we horizontally divide the 3D grid into different blocks. At each time, DeCAF randomly selects a single block and predicts its RI volume. Then, the volume is used to compute the measurement mismatch (by using the IDT forward model) and evaluate the regularizer. The weights $\phi$ of the MLP are then updated by leveraging the existing deep learning optimization algorithm.

**Padding, view enlargement, & measurement separation.** In order to correctly account for the IDT forward models that relies on the Fourier transform, our training procedure uses three computational techniques, that is, *padding*, *view enlargement*, and *measurement separation*. *Padding* is used to prevent boundary artifacts by letting each coordinate block overlap with others by a small region $p > 0$ (see Figure E.1(b)). *View enlargement* horizontally enlarging the predicted RI block to consider a larger 2D region of the measurement (see Figure E.1(c)), which ensures a sufficiently large field of view to cover the intensity information. *Measurement separation* separates and uses the partial measurements associated only with the predicted RI block to impose measurement consistency, by cropping the corresponding measurement block and removing cross-block interference (see Figure E.1(d)). Note that
Algorithm 14 Block-wise Adam

1: **input:** Measurement $y_\rho$, dictionary $\Phi$ tracking the partial measurements $\{y_i\}_{i=1}^B$.
2: **Initialization:** $\Phi = 0$.
3: **for** $k = 1, 2, 3, \ldots$ **do**
4: Randomly pick one block $i \in \{1, \ldots, B\}$
5: $y_i \leftarrow U^T_i (y_\rho - \sum_{j \neq i} U_j y_j) \quad \triangleright \text{Measurement separation}$
6: where $y_j \leftarrow \text{read}(\Phi, j)$
7: $\phi^{k+1} \leftarrow \text{AdamGradientUpdate}\{\mathcal{L}(\phi^k; y_i)\} \quad \triangleright \text{Backward propagation}$
8: $y_i^{\text{new}} \leftarrow A_\rho M_{\phi^{k+1}}(c_i) \quad \triangleright \text{Forward propagation}$
9: where $c_i$ denotes the coordinates of block $i$
10: $\Phi \leftarrow \text{update}(i, y_i^{\text{new}})$
11: **end for**

our measurement separation technique can be efficiently implemented inside the training algorithm for DeCAF.

**Block-wise Adam algorithm.** We customized Adam to realize the block-wise training procedure. Consider the approximation of measurement $y_\rho$ by the partial measurements $\{y_i\}_{i=1}^B$ generated by $B > 0$ padded and view-enlarged RI blocks

$$y_\rho = \sum_{i=1}^B U_i y_i, \quad \text{where} \quad y_i \in \mathbb{R}^{m_b}, \quad U_i : \mathbb{R}^{m_b} \to \mathbb{R}^m, \quad U^T_i : \mathbb{R}^m \to \mathbb{R}^{m_b} \quad (E.7)$$

where we introduce the operator $U_i : \mathbb{R}^n \to \mathbb{R}^n$ that injects a vector in $\mathbb{R}^{m_b}$ into $\mathbb{R}^n$ and its transpose $U^T_i$ that extracts the $i$th block from a vector in $\mathbb{R}^n$. Note that the two operators will not change the values in the vector. Pre-defining blocks allows us to enable efficient measurement separation by reusing the intermediate results. At every training step, the partial measurement of block $i$ can be computed by evaluating equation (E.8)

$$y_i = y_\rho - \sum_{j \neq i} y_j, \quad (E.8)$$
where \( y_j \) denotes the partial measurement of a block other than \( i \). By leveraging a dictionary \( \Psi \) to track the latest partial measurements of each block \( \{ y_i \}_{i=1}^B \), one can efficiently compute equation (E.8) over training. Algorithm 14 summarizes the algorithmic details of the customized block-wise Adam algorithm. Note that the algorithm only requires the measurements of the testing sample as input, and the Adam optimizer can be replaced by other training optimizers.

### E.3 Additional Technical Details

**Deep denoiser in the noise-reduction regularizer.** We adopted DnCNN [262] as our deep denoiser. The network structure is illustrated in Figure E.2. The network is composed of ten \( 3 \times 3 \) convolutional layers, of which the first nine layers are activated by rectified linear unit (ReLU) and the last layer is not equipped with any activation function. Prior to the inclusion into DeCAF, the DnCNN network, \( R_\sigma \), is pre-trained to map the noisy input to the noise residual. Subsequently, the final \( D_\sigma \) and noise reduction regularizer \( R_{NR} \) are expressed as

\[
D_\sigma := I - R_\sigma \quad \text{and} \quad R_{NR}(x) = \| R_\sigma(x) \|_2^2. \tag{E.9}
\]

The training dataset for DnCNN is synthesized by adding additive white Gaussian noise (AWGN) of strength \( \sigma \) to the natural images from BSD500 dataset [140]. Note that the decoupling of the denoiser and the imaging modality has been shown successful in the existing plug-and-play literature [220, 252, 263]. We trained DnCNN by using Adam optimizer for five noise levels \( \sigma \in \{1, 2, 3, 4, 5\} \) and selected the one that leads to the best visual performance for each sample. The loss function is defined as

\[
\mathcal{L}(I_{\text{input}}, I_{\text{noise}}) = \| R_\sigma(I_{\text{input}}) - I_{\text{noise}} \|_2^2 + \| R_\sigma(I_{\text{input}}) - I_{\text{noise}} \|_1 \tag{E.10}
\]
where $I_{\text{input}}$ is the input noisy image, and $I_{\text{noise}}$ is the ground-truth noise.

**Computational platform & training statistics.** We trained all of our deep learning models on a machine equipped with one AMD Threadripper 3960X 24-core CPU and four Nvidia RTX 3090 GPUs. We parallelized the training of DeCAF over two GPUs to accelerate the convergence. Under this setup, it approximately takes one day to train the model. We implement a decreasing learning rate, which decays exponentially as the training epoch increases, to smooth the optimization. Figure E.3 visually illustrates the training progress of DeCAF for each sample. The mean absolute error (MAE) between the predicted and real test measurements is plotted against the iteration number. Here, we define the MAE as

$$\text{MAE}(y_{\text{pred}}, y_{\rho}) = \frac{1}{M} \| y_{\text{pred}} - y_{\rho} \|_1, \quad (E.11)$$

where $y_{\text{pred}}$ denotes the full measurements generated by the entire predicted RI volume.

### E.4 Ablation Experiments

In this section, we present our ablation studies of DeCAF focusing on three aspects, that is, (1) comparing different $x$-$y$ encoding strategies, (2) demonstrating the implicit regularization of the MLP network, and (3) illustrating the effect of explicit regularization.

**Comparison of different $x$-$y$ encodings.** We validated our Radial encoding by comparing it against the Positional [151] and Gaussian [223] encodings on C. elegans, which contains small biological features that are difficult to recover. We controlled all the remaining configuration of DeCAF to be the same in order to observe the influence of the encoding strategies. The encoding of $z$ is set constantly to positional encoding with $L_z = 6$. Figure E.4 presents the experimental results with visual difference highlighted by arrows. By comparing the axial
slices shown in Figure E.4(a)-E.4(c), one can observe that Radial leads to a clearer recovery than the other two strategies. For example, the body wall and anterior pharyngeal bulb of Radial are sharp and refined, while that of Positional and Gaussian are either oversmoothed or noisy. The lateral slices shown in Figure E.4(d)-E.4(f) confirm our observation. Although Positional provides a brighter visualization of the grinder, Radial better visualizes other biological features that are highlighted by arrows. We report that radial encoding leads to similar visual improvement in all samples.

**Demonstration of implicit regularization.** We demonstrate the implicit regularization of the MLP network on spirogyra algae. In the experiment, we reconstructed the sample by using DeCAF equipped with explicit regularization (DeCAF) and without explicit regularization (DeCAF-Noreg). We control the remaining configuration of DeCAF to be the same for fair comparison. We additionally included SIMBA and Tikhonov as baseline methods. Figure E.5 presents the results with visual differences highlighted by boxes. By comparing Figure E.5(b), E.5(c), and E.5(d), one can observe that DeCAF-Noreg visually outperforms SIMBA and Tikhonov by a large margin, clearly showing that MLP by itself offers implicit regularization. One can additionally observe that DeCAF-Noreg nearly matches the performance of the full DeCAF by comparing Figure E.5(a) and E.5(b), though DeCAF better alleviates the dark-shade artifacts highlighted by the boxes.

**Demonstration of explicit regularization.** We now show that explicit regularization enables DeCAF to address more challenging samples that suffer from scattering-related artifacts. We consider C. elegans (body) and four DeCAF models: full regularization (DeCAF), only axial-continuity regularization (DeCAF-AC), only noise-reduction regularization (DeCAF-NR), and no explicit regularization (DeCAF-Noreg). We set all the remaining parameters to be the same. Figure E.6 presents the results with visual differences highlighted
Figure E.1: **Visual illustration of the block-wise training procedure for DeCAF.** (a) General workflow of the proposed training procedure. (b) Illustration of padding. (c) Illustration of view enlargement. (c) Illustration of measurement separation.

by arrows. Note how DeCAF-Noreg fails to reconstruct the worm (Figure E.6(d)) when the external regularization is removed, indicating that the implicit regularization posed by MLP is insufficient for correcting for artifacts in thick/strongly-scattering objects. By including 2D noise reduction (Figure E.6(c)), DeCAF-NR significantly improves the imagery quality over DeCAF-Noreg, but still leads to axial inconsistency as shown in the lateral view (Figure E.6(g)). By changing the regularization to axial continuity (Figure E.6(b)), DeCAF-AC improves both axial and lateral results, but causes the cross-slice interference as shown in the expanded regions highlighted by arrows. On the other hand, DeCAF achieves the best reconstruction performance by synergistically leveraging both regularization strategies (Figure E.6(a)). Figure E.6(a) and E.6(d) clearly show the necessity of explicit regularization under stronger scattering. Visual comparison of lateral slices in Figure E.6(e)-Figure E.6(h) demonstrates the same evolution.
Figure E.2: **Network architecture of DnCNN.** DnCNN is trained to map the noisy input to the noise residual by using the BSD500 training dataset [140], which consists only of natural images.

Figure E.3: **Convergence of DeCAF for different samples.** In each figure, the mean absolute error (MAE) between the predicted and real test measurements is plotted against the iteration number.
Figure E.4: Visual comparison of different $x$-$y$ encoding strategies on *C. Elegans* (head). Visual differences are highlighted using arrows. (a), (b) & (c) Axial slices of the C. elegans’ head at $z \in \{-1, 1, 3\} \, \mu m$ reconstructed by using Radial (ours), Positional, and Gaussian encoding of the $x$-$y$ plane. The encoding of $z$, as well as the rest of DeCAF, is set to be the same in the comparison. (d), (e) & (f) The associated lateral slices of the buccal cavity and grinder shown in the axial views.
Figure E.5: Demonstration of the implicit regularization due to the MLP on *Spirogyra Algae*. Visual differences are highlighted using boxes. (a) DeCAF, which includes an explicit regularizer. (b) DeCAF-Noreg relies only on implicit regularization by MLP. (c) SIMBA. (d) Tikhonov. Axial slices at \{4, 8, 12\}\,\mu m and \{28, 32, 36\}\,\mu m are selected.
Figure E.6: Demonstration of the effectiveness of explicit regularization on *C. Elegans* (body). Visual differences are highlighted by arrows. (a), (b) & (c) Axial slices of the C. Elegans body at $z \in \{-1, 1, 3\} \, \mu m$ reconstructed by using full regularization (*DeCAF*), only the axial continuity (*DeCAF-AC*), only the noise reduction (*DeCAF-NR*), and no regularization (*DeCAF-Noreg*). The rest of DeCAF is set to be same in the comparison. (d), (e) & (f) The associated lateral slices of the lumens of intestine shown in the axial views.
Appendix F

Supplement for Chapter 11

This chapter provides supplementary simulations for Chapter 11. Section F.1 reports additional technical details of CoIL as well as other baseline methods. Section F.2 presents the additional experiments, demonstrating the performance of multiple MLP architectures and the evolution of the reconstruction improvement as hyperparameter $\alpha$ changes.

F.1 Additional Technical Details

F.1.1 Training details of CoIL

The MLP is trained on a machine equipped with an Intel Xeon Gold 6130 Processor and four Nvidia GeForce GTX 1080 Ti GPUs. For each round of training, we use a single GPU without parallelization. We train each MLP for 3000 epochs with the batch size of 1000 coordinate-response pairs. The training roughly takes around 30 minutes, while the inference takes about a second for synthesizing 360 projections with 512 detectors each. We implement
a decreasing learning rate, which decays exponentially as the training epoch increases. The detailed setup of every parameter can be found in our publicly-available code\(^2\).

F.1.2 Strategies for fine-tuning hyperparameters

The hyperparameters for all methods (RED, TV, FBP-UNet) were tuned for each test image. For RED and TV, we have two hyperparameters: \(\alpha\) for balancing the data-fidelity terms and \(\tau\) for controlling the regularization strength. We optimized these parameters using a standard grid search, where \(\alpha\) was selected from 10 values evenly spaced in \([0, 1)\), and \(\tau\) was selected from 20 values evenly spaced in \((10^{-3}, 10^{-6}]\). For final evaluation, we selected the combination that leads to the best SNR performance. For FBP-UNet, we performed a 1-dimensional (1D) grid search on \(\alpha\) in the same set of \(\alpha\) values.

F.1.3 Details of FBP-UNet

FBP-UNet corresponds to our own implementation of the architecture proposed in the highly influential prior work [94]. The network was trained in the usual supervised fashion that directly predicts the groundtruth from the FBP reconstruction using the \(\ell_2\)-loss [216]. For FBP-UNet without CoIL, we trained separate CNN models in the scenarios corresponding to \(P \times I = \{60, 90, 120\} \times \{30, 40, 50\}\). For FBP-UNet using CoIL, we trained separate models on the datasets in the scenarios of \(1.5 \times P = \{90, 135, 180\}\) projection views and \(I = \{30, 40, 50\}\). We then used CoIL to generate additional measurements to achieve that number.

\(^2\)https://github.com/wustl-cig/Cooridnate-based-Internal-Learning
Table F.1: The average SNR of the sinograms generated by multiple variants of CoIL in the scenarios corresponding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$

<table>
<thead>
<tr>
<th># Views ($P$)</th>
<th>Noise Level ($I$)</th>
<th>8 layers</th>
<th>8 layers (skip conn. at 4th)</th>
<th>16 layers (skip conn. at 8th)</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>30</td>
<td>38.05</td>
<td>37.59</td>
<td>38.54</td>
<td><strong>38.61</strong></td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>43.50</td>
<td>43.27</td>
<td>43.64</td>
<td><strong>43.92</strong></td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>48.08</td>
<td>47.89</td>
<td>48.34</td>
<td><strong>48.84</strong></td>
</tr>
<tr>
<td>90</td>
<td>30</td>
<td>39.01</td>
<td>38.64</td>
<td><strong>39.60</strong></td>
<td>39.53</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>44.49</td>
<td>44.50</td>
<td>44.65</td>
<td><strong>44.85</strong></td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>50.33</td>
<td>50.14</td>
<td>50.38</td>
<td><strong>50.83</strong></td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>39.87</td>
<td>39.45</td>
<td><strong>40.31</strong></td>
<td>40.30</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>45.24</td>
<td>45.21</td>
<td><strong>45.31</strong></td>
<td>45.31</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>51.30</td>
<td>51.26</td>
<td>51.57</td>
<td><strong>51.93</strong></td>
</tr>
</tbody>
</table>

F.2 Additional Experiments and Visual Examples

F.2.1 Other potential MLP architectures

This section presents additional experiments that provide additional quantitative results for multiple MLP architectures. In particular, we considered four architectures: (1) 8-layer MLP with no skip connections, (2) 8-layer MLP with one skip connection at the forth layer, (3) 16-layer MLP with no skip connections, and (4) 16-layer MLP with one skip connection at the eighth layer. The architecture (2) above corresponds to the MLP architecture used in the original neural radiance field (NeRF) paper [151]. Tables F.1 and F.2 summarize the SNR performances for sinogram interpolation and image reconstruction, respectively. In both
tables, the largest and the second largest values are highlighted by bold and underlined fonts, respectively. We used filtered back-projection (FBP) as the reconstruction method for enabling faster evaluation. The results of the proposed CoIL architecture (which has 16 layers and skip connections at every two layers) are also included for completeness. As illustrated in Table F.1, 16-layer variants generally achieve higher SNR values than 8-layer variants, indicating that deeper MLP architectures can lead to better interpolation performance. However, this should be accommodated with the number of trainable parameters and the problem of interest, which is a classic trade-off in deep learning. Similarly, 16-layer architectures lead to an overall better reconstruction performance as can be seen in Table F.2.

We want to highlight that the CoIL architecture $\mathcal{M}_\theta(\cdot)$ has two parts: (a) Fourier feature mapping (FFM) layer $\gamma(\cdot)$ and (b) trainable MLP $\mathcal{N}_\phi(\cdot)$. What is new in our implementation of (a) relative to the existing literature is the linear-expansion FFM. Our implementation of (b) includes dense skip connections inspired by [47] for better performance. However, the results in Tables F.1 and F.2 suggest that dense skip connections might not be necessary for the best performance. While the current architecture leads to good results in most scenarios, we include other architectures in this document as potential MLP models for CoIL. Note that the skip connection we used in CoIL contains no trainable parameters, and thus does not increase the model complexity.

**F.2.2 Influence of parameter $\alpha$**

The parameter $0 \leq \alpha \leq 1$ controls the tradeoff between the real data and the CoIL field. In this section, we report on additional experiments to elucidate the influence of $\alpha$ on the final reconstruction improvement. We evaluated the RED, TV, and FBP-U Net on 10 fixed $\alpha$ values evenly spaced from 0 to 0.9 and plotted corresponding SNR improvement in Figure F.1. The value $\alpha = 0$ means that no CoIL measurements are used in the reconstruction. Note how
Table F.2: The average SNR values obtained with the variants of CoIL by using FBP in the scenarios corresponding to $P \times I = \{60, 90, 120\} \times \{30, 40, 50\}$.

<table>
<thead>
<tr>
<th># Views ($P$)</th>
<th>Noise Level ($I$)</th>
<th>8 layers</th>
<th>8 layers (skip conn. at 4th)</th>
<th>16 layers</th>
<th>16 layers (skip conn. at 8th)</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>30</td>
<td>19.84</td>
<td>19.42</td>
<td>20.36</td>
<td>20.32</td>
<td>19.45</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>23.15</td>
<td>23.09</td>
<td>23.06</td>
<td>23.39</td>
<td>23.48</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>24.75</td>
<td>24.77</td>
<td>24.76</td>
<td>24.97</td>
<td>24.99</td>
</tr>
<tr>
<td>90</td>
<td>30</td>
<td>20.38</td>
<td>20.03</td>
<td>20.93</td>
<td>20.82</td>
<td>20.15</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>23.85</td>
<td>23.95</td>
<td>23.72</td>
<td>24.01</td>
<td>24.14</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>25.63</td>
<td>25.66</td>
<td>25.55</td>
<td>25.69</td>
<td>25.69</td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>20.88</td>
<td>20.55</td>
<td>21.27</td>
<td>21.23</td>
<td>20.63</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>24.23</td>
<td>24.33</td>
<td>24.04</td>
<td>24.32</td>
<td>24.52</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>25.95</td>
<td>25.96</td>
<td>25.90</td>
<td>26.03</td>
<td>26.02</td>
</tr>
</tbody>
</table>

The SNR improves immediately as $\alpha$ starts to increase away from 0, highlighting the benefit of incorporating CoIL into image reconstruction. After $\alpha$ passes some optimal value, SNR improvements decrease, indicating the necessity to balance the true measurements with the CoIL synthesized ones. Figures F.2–F.7 in this supplement provide additional visual results for $I = 30$ dB and $I = 50$ dB.
Figure F.1: Illustration of the improvement in reconstruction SNR for RED and TV for different values of as $\alpha \geq 0$. The figure is divided into three columns, corresponding to the input SNR values of 30, 40, and 50 dB, respectively. The results indicate that CoIL synthesized measurements always improve the reconstruction performance, but need to be properly balanced with the true measurements.
Figure F.2: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 90 (used for FBP-UNet) views from $P = 60$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure F.3: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure F.4: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to $360^\circ$ (for FBP, TV, and RED) and $180^\circ$ (used for FBP-UNet) views from $P = 120$ measurements with $I = 30$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure F.5: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 90 (used for FBP-UNet) views from $P = 60$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure F.6: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 135 (used for FBP-UNet) views from $P = 90$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.
Figure F.7: Reconstruction with and without CoIL for several methods. CoIL generates measurement fields corresponding to 360 (for FBP, TV, and RED) and 180 (used for FBP-UNet) views from $P = 120$ measurements with $I = 50$ dB noise. Each image is labeled with its SNR value with respect to the groundtruth displayed in the left-most column. The visual differences are highlighted in the bounding boxes using green arrows. Note how CoIL enables the recovery of certain details missing in the reconstructions without it.