Fixed Point Theory and Algorithms for Sciences and Engineering

RESEARCH Open Access

Feasibility-based fixed point networks

Howard Heaton[1*], Samy Wu Fung[2*], Aviv Gibali[3*] and Wotao Yin[1]

*Correspondence:
hheaton@ucla.edu;
swufung@math.ucla.edu;
avivg@braude.ac.il
[1]Department of Mathematics, University of California, Los Angeles, Los Angeles, United States
[2]Department of Applied Mathematics and Statistics, Colorado School of Mines, Golden, United States
[3]Department of Mathematics, ORT Braude College, Karmiel, Israel

Abstract

Inverse problems consist of recovering a signal from a collection of noisy measurements. These problems can often be cast as feasibility problems; however, additional regularization is typically necessary to ensure accurate and stable recovery with respect to data perturbations. Hand-chosen analytic regularization can yield desirable theoretical guarantees, but such approaches have limited effectiveness recovering signals due to their inability to leverage large amounts of available data. To this end, this work fuses data-driven regularization and convex feasibility in a theoretically sound manner. This is accomplished using feasibility-based fixed point networks (F-FPNs). Each F-FPN defines a collection of nonexpansive operators, each of which is the composition of a projection-based operator and a data-driven regularization operator. Fixed point iteration is used to compute fixed points of these operators, and weights of the operators are tuned so that the fixed points closely represent available data. Numerical examples demonstrate performance increases by F-FPNs when compared to standard TV-based recovery methods for CT reconstruction and a comparable neural network based on algorithm unrolling. Codes are available on Github: github.com/howardheaton/feasibility_fixed_point_networks.

Keywords: Convex feasibility problem; Projection; Averaged; Fixed point network; Nonexpansive; Learned regularizer; Machine learning; Implicit depth; Deep learning

1 Introduction

Inverse problems arise in numerous applications such as medical imaging [1–4], phase retrieval [5–7], geophysics [8–13], and machine learning [14–18]. The goal of inverse problems is to recover a signal[1] $u_d^\star$ from a collection of indirect noisy measurements d. These quantities are typically related by a linear mapping A via

$$d = Au_d^\star + \varepsilon, \tag{1}$$

where ε is measurement noise. Inverse problems are often ill-posed, making recovery of the signal $u_d^\star$ unstable for noise-affected data d. To overcome this, traditional approaches

[1]While we refer to signals, this phrase is generally meant to describe objects of interest that can be represented mathematically (*e.g.* images, parameters of a differential equation, and points in a Euclidean space).

estimate the signal $u_d^\star$ by a solution $\tilde{u}_d$ to the variational problem

$$\min_u \ell(Au, d) + J(u), \tag{2}$$

where ℓ is a fidelity term that measures the discrepancy between the measurements and the application of the forward operator A to the signal estimate (*e.g.* least squares). The function J serves as a regularizer, which ensures both that the solution to (2) is unique and that its computation is stable. In addition to ensuring well-posedness, regularizers are constructed in an effort to instill prior knowledge of the true signal *e.g.* sparsity $J(u) = \|u\|_1$ [19–22], Tikhonov $J(u) = \|u\|^2$ [23, 24], total variation (TV) $J(u) = \|\nabla u\|_1$ [25, 26], and, more recently, data-driven regularizers [27–29]. A further generalization of using data-driven regularization consists of plug-and-play (PnP) methods [30–32], which replace the proximal operators in an optimization algorithm with previously trained denoisers.

An underlying theme of regularization is that signals represented in high dimensional spaces often exhibit a common structure. Although hand picked regularizers may admit desirable theoretical properties leveraging *a priori* knowledge, they are typically unable to leverage available data. An ideal regularizer will leverage available data to best capture the core properties that should be exhibited by output reconstruction estimates of true signals. Neural networks have demonstrated great success in this regard, achieving state of the art results [33, 34]. However, purely data-driven machine learning approaches do little to leverage the underlying physics of a problem, which can lead to poor compliance with data [35]. On the other hand, fast feasibility-seeking algorithms (*e.g.* see [36–40] and the references therein) efficiently leverage known physics to solve inverse problems, being able to handle massive-scale sets of constraints [36, 41–43]. Thus, a relatively untackled question remains:

Is it possible to fuse feasibility-seeking algorithms with data-driven regularization in a manner that improves reconstructions and yields convergence?

This work answers the above inquiry affirmatively. The key idea is to use machine learning techniques to create a mapping T_Θ, parameterized by weights Θ. For fixed measurement data d, $T_\Theta(\cdot; d)$ forms an operator possessing standard properties used in feasibility algorithms. Fixed point iteration is used to find fixed points of $T_\Theta(\cdot; d)$ and the weights Θ are tuned such that these fixed points both resemble available signal data and are consistent with measurements (up to the noise level).

Contribution The core contribution of this work is to connect powerful feasibility-seeking algorithms to data-driven regularization in a manner that maintains theoretical guarantees. This is accomplished by presenting a feasibility-based fixed point network (F-FPN) framework that solves a learned feasibility problem. Numerical examples are provided that demonstrate notable performance benefits to our proposed formulation when compared to TV-based methods and fixed-depth neural networks formed by algorithm unrolling.

Outline We first overview convex feasibility problems (CFPs) and a learned feasibility problem (Section 2). Relevant neural network material is discussed next (Section 3), followed by our proposed F-FPN framework (Section 4). Numerical examples are then provided with discussion and conclusions (Sections 5 and 6).

2 Convex feasibility background

2.1 Feasibility problem

Convex feasibility problems (CFPs) arise in many real-world applications *e.g.* imaging, sensor networks, radiation therapy treatment planning (see [36, 44, 45] and the references therein). We formalize the CFP setting and relevant methods as follows. Let $\mathcal{U}$ and $\mathcal{D}$ be finite dimensional Hilbert spaces, referred to as the signal and data spaces, respectively.[2] Given additional knowledge about a linear inverse problem, measurement data $d \in \mathcal{D}$ can be used to express a CFP solved by the true signal $u_d^\star \in \mathcal{U}$ when measurements are noise-free. That is, data d can be used to define a collection $\{\mathcal{C}_{d,j}\}_{j=1}^m$ of closed convex subsets of $\mathcal{U}$ (*e.g.* hyperplanes) such that the true signal $u_d^\star$ is contained in their intersection *i.e.* $u_d^\star$ solves the problem

$$\text{Find } u_d \text{ such that} \quad u_d \in \mathcal{C}_d \triangleq \bigcap_{j=1}^m \mathcal{C}_{d,j}. \tag{CFP}$$

A common approach to solving (CFP), *inter alia*, is to use projection algorithms [46], which utilize orthogonal projections onto the individual sets $\mathcal{C}_{d,j}$. For a closed, convex, and nonempty set $\mathcal{C} \subseteq \mathcal{U}$, the projection $P_{\mathcal{C}} : \mathcal{U} \to \mathcal{C}$ onto $\mathcal{C}$ is defined by

$$P_{\mathcal{C}}(u) \triangleq \operatorname*{argmin}_{v \in \mathcal{C}} \frac{1}{2}\|v - u\|^2. \tag{3}$$

Projection algorithms are iterative in nature and each update uses combinations of projections onto each set $\mathcal{C}_{d,j}$, relying on the principle that it is generally much easier to project onto the individual sets than onto their intersection. These methods date back to the 1930s [47, 48] and have been adapted to now handle huge-size problems of dimensions for which more sophisticated methods cease to be efficient or even applicable due to memory requirements [36]. Computational simplicity derives from the fact that the building bricks of a projection algorithm are the projections onto individual sets. Memory efficiency occurs because the algorithmic structure is either sequential or simultaneous (or hybrid) as in the block-iterative projection methods [49, 50] and string-averaging projection methods [36, 51–53]. These algorithms generate sequences that solve (CFP) asymptotically, and the update operations can be iteration dependent (*e.g.* cyclic projections). We let $\mathcal{A}_d^k$ be the update operator for the kth step of a projection algorithm solving (CFP). Consequently, each projection algorithm generates a sequence $\{u^k\}_{k\in\mathbb{N}}$ via the fixed point iteration

$$u^{k+1} \triangleq \mathcal{A}_d^k(u^k) \quad \text{for all } k \in \mathbb{N}. \tag{FPI}$$

A common assumption for such methods is the intersection of all the algorithmic operators' fixed point sets[3] contains or forms the desired set $\mathcal{C}_d$ *i.e.*

$$\mathcal{C}_d = \bigcap_{k=1}^\infty \operatorname{fix}(\mathcal{A}_d^k), \tag{4}$$

which automatically holds when $\{\mathcal{A}_d^k\}_{k\in\mathbb{N}}$ cycles over a collection of projections.

[2] The inner product and norm are denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ respectively. Although we use the same notation for each space, it will be clear from the context which one is used.

[3] For an operator T, its fixed point set is $\operatorname{fix}(T) \triangleq \{u : u = T(u)\}$.

2.2 Data-driven feasibility problem

As noted previously, inverse problems are often ill-posed, making (CFP) insufficient to faithfully recover the signal $u_d^\star$. Additionally, when noise is present, it can often be the case that the intersection is empty (*i.e.* $\mathcal{C}_d = \emptyset$). This calls for a different model to recover $u_d^\star$. To date, projection methods have limited inclusion of regularization (*e.g.* superiorization [54–58], sparsified Kaczmarz [59, 60]). Beyond sparsity via ℓ_1 minimization, such approaches typically do not yield guarantees beyond feasibility (*e.g.* it may be desirable to minimize a regularizer over $\mathcal{C}_d$). We propose composing a projection algorithm and a data-driven regularization operator in a manner so that each update is analogous to a proximal-gradient step. This is accomplished via a parameterized mapping $R_\Theta : \mathcal{U} \to \mathcal{U}$, with weights[4] denoted by Θ. This mapping directly leverages available data (explained in Section 3) to learn features shared among true signals of interest. We augment (CFP) by using operators $\{\mathcal{A}_d^k\}_{k\in\mathbb{N}}$ for solving (CFP) and instead solve the learned common fixed points (L-CFP) problem

$$\text{Find } \tilde{u}_d \text{ such that} \quad \tilde{u}_d \in \mathcal{C}_{\Theta,d} \triangleq \bigcap_{k=1}^{\infty} \operatorname{fix}\left(\mathcal{A}_d^k \circ R_\Theta\right). \tag{L-CFP}$$

Loosely speaking, when R_Θ is chosen well, the signal $\tilde{u}_d$ closely approximates $u_d^\star$.

We utilize classic operator results to solve (L-CFP). An operator $T : \mathcal{U} \to \mathcal{U}$ is *nonexpansive* if it is 1-Lipschitz *i.e.*

$$\left\| T(u) - T(u) \right\| \leq \|u - v\| \quad \text{for all } u, v \in \mathcal{U}. \tag{5}$$

Also, T is *averaged* if there exist $\alpha \in (0,1)$ and a nonexpansive operator $Q : \mathcal{U} \to \mathcal{U}$ such that $T(u) = (1-\alpha)u + \alpha Q(u)$ for all $u \in \mathcal{U}$. For example, the projection P_S defined in (3) is averaged along with convex combinations of projections [61]. Our method utilizes the following standard assumptions, which are typically satisfied by projection methods (in the noise-free setting with R_Θ as the identity).

Assumption 2.1 The intersection set $\mathcal{C}_{\Theta,d}$ defined in (L-CFP) is nonempty and $\{(\mathcal{A}_d^k \circ R_\Theta)\}_{k\in\mathbb{N}}$ forms a sequence of nonexpansive operators.

Assumption 2.2 For any sequence $\{u^k\}_{k\in\mathbb{N}} \subset \mathcal{U}$, the sequence of operators $\{(\mathcal{A}_d^k \circ R_\Theta)\}_{k\in\mathbb{N}}$ has the property

$$\lim_{k\to\infty} \left\| \left(\mathcal{A}_d^k \circ R_\Theta\right)\left(u^k\right) - u^k \right\| = 0 \quad \Longrightarrow \quad \liminf_{k\to\infty} \left\| P_{\mathcal{C}_{\Theta,d}}\left(u^k\right) - u^k \right\| = 0. \tag{6}$$

When a finite collection of update operations are used and applied (essentially) cyclically, the previous assumption automatically holds (*e.g.* setting $\mathcal{A}_d^k \triangleq P_{\mathcal{C}_{d,i_k}}$ and $i_k \triangleq k \bmod(m) + 1$). We use the learned fixed point iteration to solve (L-CFP)

$$u^{k+1} \triangleq \left(\mathcal{A}_d^k \circ R_\Theta\right)\left(u^k\right) \quad \text{for all } k \in \mathbb{N}. \tag{L-FPI}$$

Justification of the (L-FPI) iteration is provided by the following theorems, which are rewritten from their original form to a manner that matches the present context.

[4] Operator weights are also commonly called parameters.

Theorem 2.1 (Krasnosel'skiĭ–Mann [62, 63]) *If* $(\mathcal{A}_d \circ R_\Theta)\colon \mathcal{U} \to \mathcal{U}$ *is averaged and has a fixed point, then, for any* $u^1 \in \mathcal{U}$, *the sequence* $\{u^k\}_{k\in\mathbb{N}}$ *generated by* (L-FPI), *taking* $\mathcal{A}_d^k \circ R_\Theta = \mathcal{A}_d \circ R_\Theta$, *converges to a fixed point of* $\mathcal{A}_d \circ R_\Theta$.

Theorem 2.2 (Cegieslki, Theorem 3.6.2, [61]) *If Assumptions 2.1 and 2.2 hold, and if* $\{u^k\}_{k\in\mathbb{N}}$ *is a sequence generated by the iteration* (L-FPI) *satisfying* $\|u^{k+1} - u^k\| \to 0$, *then* $\{u^k\}_{k\in\mathbb{N}}$ *converges to a limit* $u^\infty \in C_{\Theta,d}$.

3 Fixed point networks overview

One of the most promising areas in artificial intelligence is deep learning, a form of machine learning that uses neural networks containing many hidden layers [64, 65]. Deep learning tasks in the context of this work can be cast as follows. Given measurements d drawn from a distribution $\mathbb{P}_\mathcal{D}$ and corresponding signals $u_d^\star$ drawn from a distribution $\mathbb{P}_\mathcal{U}$, we seek a mapping $\mathcal{N}_\Theta\colon \mathcal{D} \to \mathcal{U}$ that approximates a one-to-one correspondence between the measurements and signals *i.e.*

$$\mathcal{N}_\Theta(d) \approx u_d^\star \quad \text{for all } d \sim \mathbb{P}_\mathcal{D}. \tag{7}$$

Depending on the nature of the given data, the task at hand can be regression or classification. In this work, we focus on solving regression problems where the learning is *supervised i.e.* the loss function explicitly uses a correspondence between input and output data pairings. When the loss function does not use this correspondence (or when not all data pairings are available), the learning is *semi-supervised* if partial pairings are used and *unsupervised* if no pairings are used.

3.1 Recurrent neural networks

A common model for $\mathcal{N}_\Theta$ is given by recurrent neural networks (RNNs) [66], which have enjoyed a great deal of success in natural language processing (NLPs) [67], time series [68], and classification [68]. An N-layer RNN takes observed data d as input and can be modeled as the N-fold composition of a mapping T_Θ via

$$\mathcal{N}_\Theta \triangleq \underbrace{T_\Theta \circ T_\Theta \circ \ldots \circ T_\Theta}_{N \text{ times}}. \tag{8}$$

Here, $T_\Theta(u; d)$ is an operator comprised of a finite sequence of applications of (possibly) distinct affine mappings and nonlinearities, and u is initialized to some fixed value (*e.g.* the zero vector). To identify a faithful mapping $\mathcal{N}_\Theta$ as in (7), we solve a training problem. This is modeled as finding weights that minimize an expected loss, which is typically solved using optimization methods like SGD [69] and Adam [70]. In particular, we solve the training problem

$$\min_\Theta \mathbb{E}_{d\sim\mathcal{D}}\big[\ell\big(\mathcal{N}_\Theta(d), u_d^\star\big)\big], \tag{9}$$

where $\ell\colon \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ models the discrepancy between the prediction $\mathcal{N}_\Theta(d)$ of the network and the training data $u_d^\star$. In practice, the expectation in (9) is approximated using a finite subset of data, which is referred to as *training data*. In addition to minimizing over the training data, we aim for (7) to hold for a set of *testing data* that was not used during training (which tests the network's ability to *generalize*).

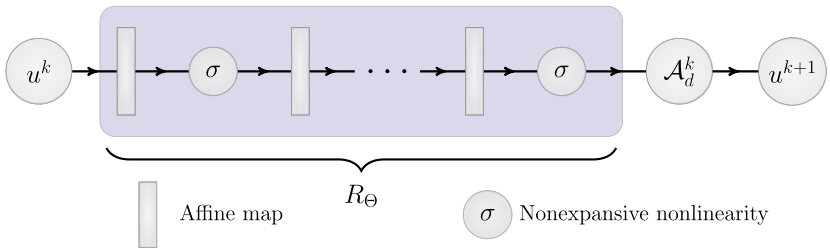

Figure 1 Diagram for update operations in the learned fixed point iteration (L-FPI) to solve (L-CFP). Here R_Θ is comprised of a finite sequence of applications of (possibly) distinct affine mappings (*e.g.* convolutions) and nonlinearities (*e.g.* projections on the nonnegative orthant *i.e.* ReLUs). For each $k \in \mathbb{N}$, we let $\mathcal{A}_d^k$ be a projection-based algorithmic operator. The parameters Θ of R_Θ are tuned in an offline process by solving (9) to ensure signals are faithfully recovered.

Remark 3.1 We emphasize that a time intensive offline process is used to find a solution $\Theta^\star$ to (9) (as is common in machine learning). After this, in the online setting, we apply $\mathcal{N}_{\Theta^\star}(d)$ to recover a signal $u_d^\star$ from its previously unseen measurements d, which is a much faster process.

Remark 3.2 If we impose a particular structure to T_Θ, as shown in Figure 1, an N-layer RNN can be interpreted as an unrolled fixed point (or optimization) algorithm that runs for N iterations. Our experiments compare our proposed method to such an unrolled scheme.

3.2 Fixed point networks

Increasing neural network depth leads to more expressibility [71, 72]. A recent trend in deep learning seeks to inquires: what happens when the number of recurrent layers N goes to infinity? Due to ever growing memory requirements (growing linearly with N) to train networks, directly unrolling a sequence generated by successively applying T_Θ is, in general, intractable for arbitrarily large N. However, the sequence limit can be modeled using a fixed point equation. In this case, evaluating a fixed point network (FPN) [73] is equivalent to finding the unique fixed point of an averaged operator $T_\Theta(\cdot; d)$ *i.e.* an FPN $\mathcal{N}_\Theta$ is defined by[5]

$$\mathcal{N}_\Theta(d) \triangleq u_{\Theta,d}, \quad \text{where } u_{\Theta,d} = T_\Theta(u_{\Theta,d}; d). \tag{10}$$

Standard results [74–76] can be used to guarantee the existence[6] of fixed points of nonexpansive T_Θ. Iteratively applying T_Θ produces a convergent sequence (Theorem 2.1). However, for different d, the number of steps to converge may vary, and so these models belong to the class of *implicit depth models*. As mentioned, it is computationally intractable to differentiate ℓ with respect to Θ by applying the chain rule through each of the N layers (when N is sufficiently large). Instead, the gradient $\mathrm{d}\ell/\mathrm{d}\Theta$ is computed via the implicit function

[5] The presented definition is a slight variation of the original work, adapted to this setting.

[6] The original FPN paper used a more restrictive contraction condition to guarantee uniqueness and justify how the weights are updated during training. However, we use their method in our more general setting since the contraction factor can be arbitrarily close to unity.

theorem [77]. Specifically, the gradient is obtained by solving the Jacobian-inverse equation (*e.g.* see [78–80])

$$\frac{\mathrm{d}\ell}{\mathrm{d}\Theta} = \mathcal{J}_\Theta^{-1}\frac{\partial T}{\partial \Theta}, \quad \text{where } \mathcal{J}_\Theta \triangleq I - \frac{dT_\Theta}{du}. \tag{11}$$

Recent works that solve the Jacobian-inverse equation in (11) to train neural networks include deep equilibrium networks [78, 81] and monotone equilibrium networks [79]. A key difficulty arises when computing the gradient via (11), especially when the signal space has large dimensions (*e.g.* when $u_d^\star$ is a high-resolution image). Namely, a linear system involving the Jacobian term $\mathcal{J}_\Theta$ must be approximately solved to estimate the gradient of ℓ. Recently, a new framework for training implicit depth models, called Jacobian-free backpropagation (JFB) [73], was presented in the context of FPNs, which avoids the intensive linear system solves at each step. The idea is to replace gradient $\mathrm{d}\ell/\mathrm{d}\Theta$ updates with $\partial T/\partial\Theta$, which is equivalent to a preconditioned gradient (since $\mathcal{J}_\Theta^{-1}$ is coercive [73, Lemma A.1]). JFB provides a descent direction and was found to be effective and competitive for training implicit-depth neural networks at substantially reduced computational costs. Since the present work solves inverse problems where the signal space has very high dimension, we leverage FPNs and JFB to solve (9) for our proposed method.

3.3 Learning to optimize

An emerging field in machine learning is known as "learning to optimize" (L2O) (*e.g.* see the survey works [82, 83]). As a paradigm shift away from conventional optimization algorithm design, L2O uses machine learning to improve an optimization method. Two approaches are typically used for model-based algorithms. Plug-and-Play (PnP) methods learn a denoiser in the form of a neural network and then plug this denoiser into an optimization algorithm (*e.g.* to replace a proximal for total variation). Here training of the denoiser is separate from the task at hand. On the other hand, unrolling methods incorporate tunable weights into an algorithm that is truncated to a fixed number of iterations, forming a neural network. Unrolling the iterative soft thresholding algorithm (ISTA), the authors in [84] obtained the first major L2O scheme learned ISTA (LISTA) by letting each matrix in the updates be tunable. Follow-up papers also demonstrate empirical success in various applications, include compressive sensing [85–93], denoising [29, 88, 93–99], and deblurring [88, 93, 95, 100–105]. L2O schemes are related to our method, but no L2O scheme has, to our knowledge, used a fixed point model as in (L-CFP). Additionally, our JFB training regime differs from the L2O unrolling and PnP schemes.

4 Proposed method

Herein we present the feasibility-based FPN (F-FPN). Although based on FPNs, here we replace the single operator of FPNs by a sequence of operators, each taking the form of a composition. Namely, we use updates in the iteration (L-FPI). The assumptions necessary for convergence can be approximately ensured (*e.g.* see Subsection A.4 in the Appendix). This iteration yields the F-FPN $\mathcal{N}_\Theta$, defined by

$$\mathcal{N}_\Theta(d) \triangleq \tilde{u}_d, \quad \text{where} \tilde{u}_d = \bigcap_{k=1}^{\infty} \mathrm{fix}\big(\mathcal{A}_d^k \circ R_\Theta\big), \tag{12}$$

Algorithm 1 Feasibility-based fixed point network (F-FPN)

1:	$\mathcal{N}_\Theta(d)$:	◁ Input data is d
2:	$u^1 \leftarrow \tilde{u}$	◁ Initialize iterate to fixed reference
3:	$k \leftarrow 1$	◁ Initialize iteration counter
4:	**while** $\|u^{k+1} - u^k\| \geq \delta$	◁ Loop to convergence
5:	$u^{k+1} \leftarrow (\mathcal{A}_d^k \circ R_\Theta)(u^k; d)$	◁ Apply regularization R_Θ and feasibility step $\mathcal{A}_d^k$
6:	$k \leftarrow k + 1$	◁ Increment counter
7:	**return** u^k	◁ Output solution estimate

assuming the intersection is unique.[7] This is approximately implemented via Algorithm 1. The weights Θ of the network $\mathcal{N}_\Theta$ are tuned by solving the training problem (9). In an ideal situation, the optimal weights $\Theta^\star$ solving (9) would yield feasible outputs (*i.e.* $\mathcal{N}_\Theta(d) \in \mathcal{C}_d$ for all data $d \in \mathcal{C}$) that also resemble the true signals $u_d^\star$. However, measurement noise in practice makes it unlikely that $\mathcal{N}_\Theta(d)$ is feasible, let alone that $\mathcal{C}_d$ is nonempty. In the noisy setting, this is no longer a concern since we augment (CFP) via (L-CFP) and are ultimately concerned with recovering a signal $u_d^\star$, not solving a feasibility problem. In summary, our model is based on the underlying physics of a problem (via the convex feasibility structure), but is also steered by available data via training problem (9). Illustrations of the efficacy of this approach are provided in Section 5.

5 Experiments

Experiments in this section demonstrate the relative reconstruction quality of F-FPNs and comparable schemes—in particular, filtered backprojection (FBP) [106], total variation (TV) minimization (similarly to [107, 108]), total variation superiorization (based on [109, 110]), and an unrolled L2O scheme with an RNN structure.

5.1 Experimental setup

Comparisons are provided for two low-dose CT examples: a synthetic dataset, consisting of images of random ellipses, and the LoDoPab dataset [111], which consists of human phantoms. For both datasets, CT measurements are simulated with a parallel beam geometry with a sparse-angle setup of only 30 angles and 183 projection beams, resulting in 5490 equations and 16,384 unknowns. Additionally, we add 1.5% Gaussian noise corresponding to each individual beam measurement. Moreover, the images have a resolution of 128×128 pixels. The quality of the image reconstructions are determined using the peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM). We use the PyTorch deep learning framework [112] and the ADAM [70] optimizer. We also use the operator discretization library (ODL) python library [113] to compute the filtered backprojection solutions. The CT experiments are run on a Google Colab notebook. For all methods, we use a single diagonally relaxed orthogonal projections (DROP) [37] operator for $\mathcal{A}_d$ (*i.e.* $\mathcal{A}_d^k = \mathcal{A}_d$ for all k), noting DROP is nonexpansive with respect to a norm dependent on A [114]. The loss function ℓ used for training is the mean squared error between reconstruction estimates and the corresponding true signals. We use a synthetic dataset consisting of random phantoms of combined ellipses as in [115]. The ellipse

[7] Uniqueness is unlikely in practice; however, this assumption is justified since we use the same initial iterate u^1 for each initialization. This makes recovery of the same signal stable with respect to changes in Θ.

training and test sets contain 10,000 and 1000 pairs, respectively. We also use phantoms derived from actual human chest CT scans via the benchmark low-dose parallel beam dataset (LoDoPaB) [111]. The LoDoPab training and test sets contain 20,000 and 2000 pairs, respectively.

5.2 Experiment methods

TV superiorization Sequences generated by successively applying the operator $\mathcal{A}_d$ are known to converge even in the presence of summable perturbations, which can be intentionally added to lower a regularizer value (*e.g.* TV) without compromising convergence, thereby giving a "superior" feasible point. Compared to minimization methods, superiorization typically only guarantees feasibility, but is often able to do so at reduced computational cost. This scheme, denoted as TVS, generates updates

$$u^{k+1} = \mathcal{A}_d\left(u^k - \alpha\beta^k D_-^\top\left(\frac{D_+ u}{\|D_+ u\|_2 + \varepsilon}\right)\right) \quad \text{for } k = 1, 2, \ldots, 20, \tag{13}$$

where D_- and D_+ are the forward and backward differencing operators, $\varepsilon > 0$ is added for stability, and 20 iterations are used as early stopping to avoid overfitting to noise. The differencing operations yield a derivative of isotropic TV (*e.g.* see [116]). The scalars $\alpha > 0$ and $\beta \in (0, 1)$ are chosen to minimize training mean squared error. See the superiorization bibliography [117] for further TVS materials.

TV minimization For a second analytic comparison method, we use anisotropic TV minimization (TVM). In this case, we solve the constrained problem

$$\min_{u \in [0,1]^n} \|D_+ u\|_1 \quad \text{such that } \|Au - d\| \leq \varepsilon, \tag{TVM}$$

where $\varepsilon > 0$ is a hand-chosen scalar reflecting the level of measurement noise and the box constraints on u are included since all signals have pixel values in the interval $[0, 1]$. We use linearized ADMM [118] to solve (TVM) and refer to this model as TV minimization (TVM). Implementation details are in the Appendix.

F-FPN structure The architecture of the operator R_Θ is modeled after the seminal work [119] on residual networks. The F-FPN and unrolled scheme both leverage the same structure R_Θ and DROP operator for $\mathcal{A}_d$. The operator R_Θ is the composition of four residual blocks. Each residual block takes the form of the identity mapping plus the composition of a leaky ReLU activation function and convolution (twice). The number of network weights in R_Θ for each setup was 96,307, a small number by machine learning standards. Further details are provided in the Appendix.

5.3 Experiment results

Our results show that F-FPN outperforms all classical methods as well as the unrolled data-driven method. We show the result on an individual reconstruction via wide and zoomed-in images from the ellipse and LoDoPab testing datasets in Figures 2 and 3 and Figures 4 and 5, respectively. The average SSIM and PSNR values on the entire ellipse and LoDoPab datasets are shown in Tables 1 and 2. We emphasize that the type of noise

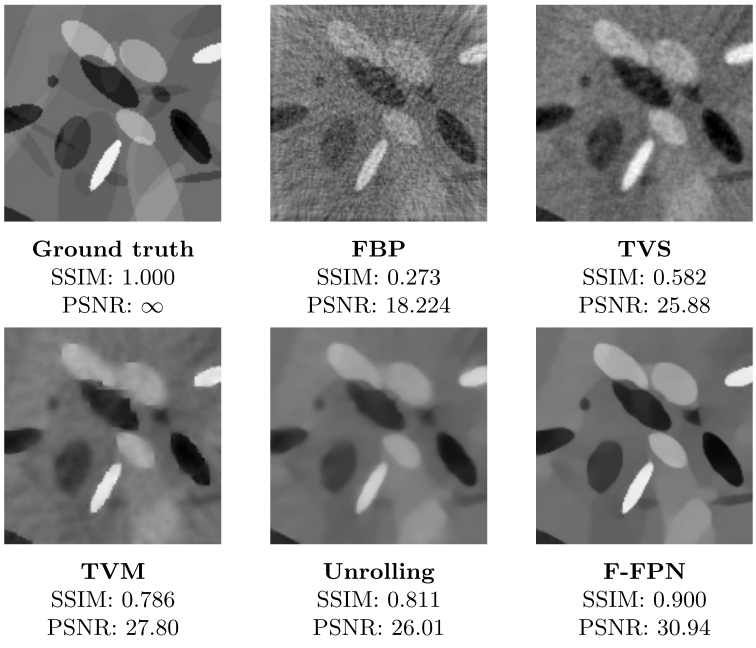

Figure 2 Ellipse reconstruction with test data for each method: filtered back projection (FBP), TV superiorization (TVS), TV minimization (TVM), unrolled network, and the proposed feasibility-based fixed point network (F-FPN).

depends on each individual ray in a similar manner to [120], making the measurements more noisy than some related works. This noise and ill-posedness of our underdetermined setup are illustrated by the poor quality of analytic method reconstructions. (However, we note improvement by using TV over FBP and further improvement by TV minimization over TV superiorization.) Although nearly identical in structure to F-FPNs, these results show the unrolled method to be inferior to F-FPNs in these experiments. We hypothesize that this is due to the large memory requirements of unrolling (unlike F-FPNs), which limits the number of unrolled steps ($\sim$ 20 steps versus 100+ steps of F-FPNs), and F-FPNs are tuned to optimize a fixed point condition rather than a fixed number of updates.

6 Conclusion

This work connects feasibility-seeking algorithms and data-driven algorithms (*i.e.* neural networks). The F-FPN framework leverages the elegance of fixed point methods while using state of the art training methods for implicit-depth deep learning. This results in a sequence of learned operators $\{\mathcal{A}_d^k \circ R_\Theta\}_{k\in\mathbb{N}}$ that can be repeatedly applied until convergence is obtained. This limit point is expected to be nearly compatible with provided constraints (up to the level of noise) and resemble the collection of true signals. The provided numerical examples show improved performance obtained by F-FPNs over both classic methods and an unrolling-based network. Future work will extend FPNs to a wider class of optimization problems and further establish theory connecting machine learning to fixed point methods.

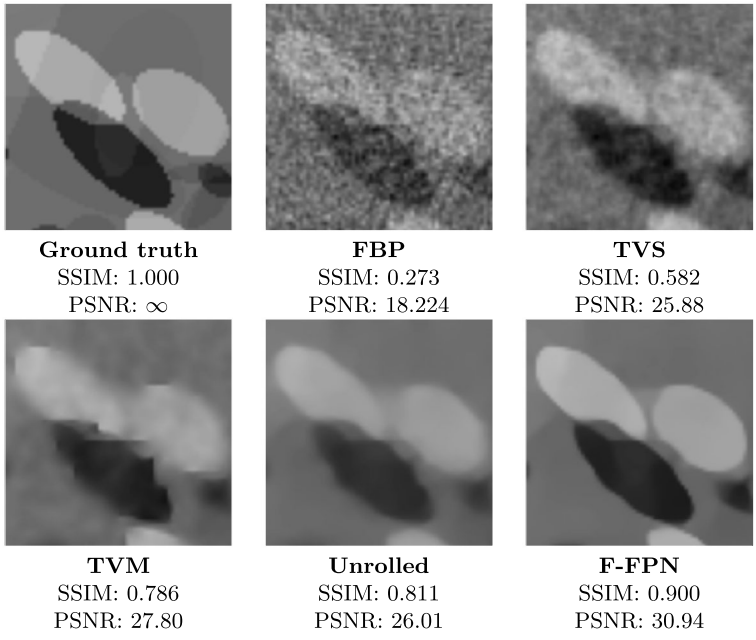

Figure 3 Zoomed-in ellipse reconstruction with test data of Figure 2 for each method: FBP, TVS, TVM, unrolling, and the proposed F-FPN.

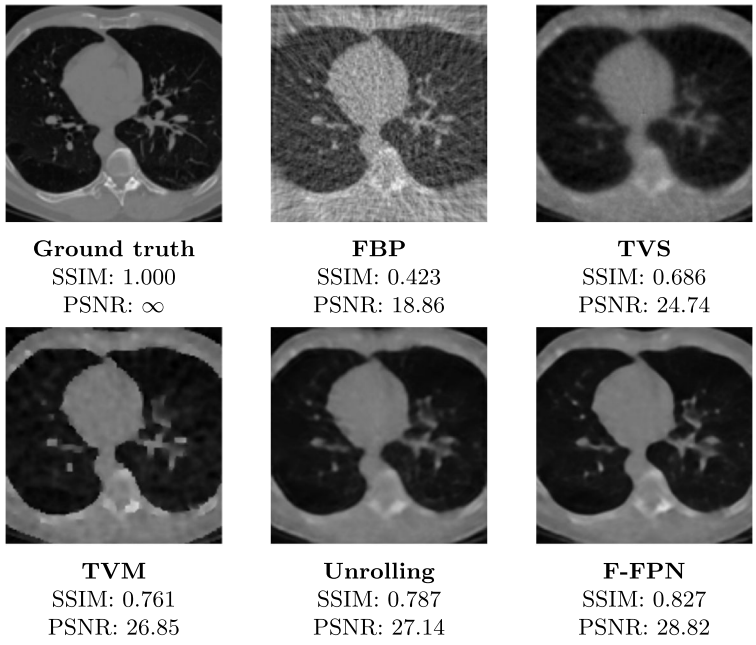

Figure 4 LoDoPab reconstruction with test data for each method: filtered back projection (FBP), TV superiorization (TVS), TV minimization (TVM), unrolled network, and the proposed feasibility-based fixed point network (F-FPN).

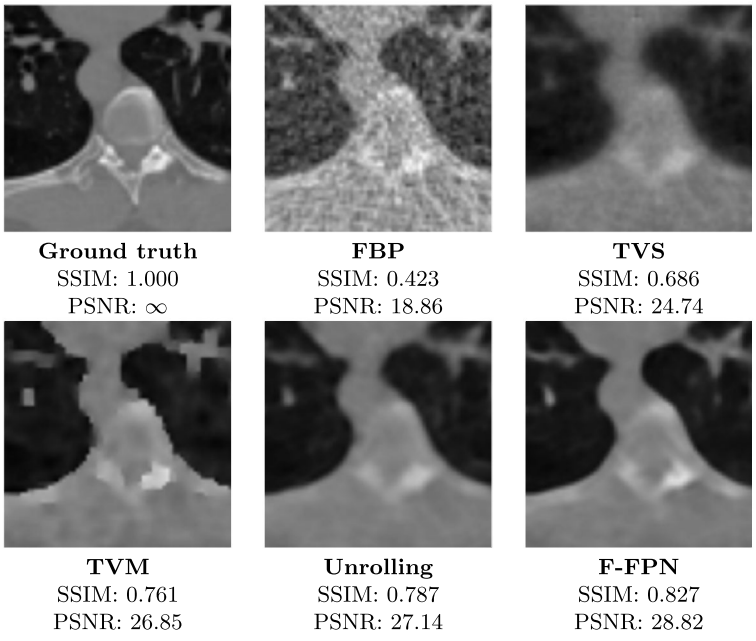

Figure 5 Zoomed-in LoDoPab reconstruction with test data of Figure 4 for each method: FBP, TVS, TVM, unrolling, and the proposed F-FPN.

Table 1 Average PSNR and SSIM on the 1000 image ellipse testing dataset.

Method	Avg. PSNR (dB)	Avg. SSIM	# Parameters
Filtered backprojection	17.79	0.211	1
TV superiorization	27.35	0.721	2
TV minimization	28.55	0.772	4
Unrolled network	30.39	0.859	96,307
F-FPN (proposed)	**31.30**	**0.877**	96,307

Table 2 Average PSNR/SSIM on the 2000 image LoDoPab testing dataset.

Method	Avg. PSNR (dB)	Avg. SSIM	# Parameters
Filtered backprojection	19.27	0.354	1
TV superiorization	26.65	0.697	2
TV minimization	28.52	0.765	4
Unrolled network	29.30	0.800	96, 307
F-FPN (proposed)	**30.46**	**0.832**	96,307

Appendix

A.1 Network structure

For our neural network architecture, we set R_Θ to be a composition of four convolutions: the first takes in one channel and outputs 44 channels. The second and third convolutions have 44 input and output channels. The final convolution maps the 44 channels back to one channel. Prior to each convolution, we use the leaky rectified linear activation function (ReLU) as the nonlinear activation function between layers. The leaky ReLU function,

denoted by ϕ, is defined as

$$\phi_a(u) \triangleq \begin{cases} u & \text{if } u \geq 0, \\ au & \text{if } u < 0, \end{cases} \tag{14}$$

where a is a number determined by the user. Exact implementation details can be found in https://github.com/howardheaton/feasibility_fixed_point_networks.

A.2 Training setup

To generate the FBP reconstructions, we use FBP operator from the operator discretization library (ODL). Since the ODL FBP operator is a built-in operator whose rows are not normalized (unlike in the remainder of the methods where DROP is used), we scale the observed data accordingly. In particular, we multiply each row of the observed data d by the rows of the original, unnormalized matrix A. For all other methods, we normalized the rows of A and scaled the measurements accordingly.

For the ellipse dataset, we train the unrolled network using a batch size of 15 for 60 epochs. The F-FPN network training used a batch-size of 15 for 50 epochs. The unrolled network architecture contains 20 total layers (*i.e.* update steps)—the number of layers was chosen on the memory capacity of the GPU.

For the LoDoPab dataset, we train the unrolled and F-FPN networks using a batch-size of 50 for 50 epochs total. The unrolled network architecture contains 14 total layers (*i.e.* update steps)—the number of layers was chosen on the memory capacity of the GPU.

A.3 TVS parameters

The TVS parameters were trained by unrolling the method indicated in (13) for 20 steps into the structure of a neural network. This unrolled network contained two parameters: α and β. We initialized α to 0.05 and β to 0.99. Then we used Adam to tune the parameters with the training data. For the ellipse experiment, the learned parameters were $\alpha = 0.0023$ and $\beta = 0.968$. For the LoDoPab experiment, the learned parameters were $\alpha = 0.0013$ and $\beta = 9607$. Note the training to tune the parameters optimized performance with respect to mean squared error.

A.4 Approximate Lipschitz enforcement

Herein we overview our technique for ensuring the composition $(\mathcal{A}_d \circ R_\Theta)$ is γ-Lipschitz in our experiments (with $\gamma \in (0, 1)$). This is accomplished in an approximate manner using batches of computed fixed points after each forward pass in training. Let B denote a set of indices corresponding to a collection of fixed points $\tilde{u}_d$, and let $\{\zeta_i\}_{i \in B}$ be Gaussian random vectors. Letting $|B|$ denote the cardinality of B, we check whether the following inequality holds:

$$\underbrace{\frac{1}{|B|} \sum_{i \in B} \left\| (\mathcal{A}_d \circ R_\Theta)(\tilde{u}_d) - (\mathcal{A}_d \circ R_\Theta)(\tilde{u}_d + \zeta_i) \right\|}_{C_1} \leq \gamma \underbrace{\frac{1}{|B|} \sum_{i} \|\zeta_i\|}_{C_2}. \tag{15}$$

If the network is γ-Lipschitz, then $C_1 \leq \gamma C_2$ for any provided batch B of samples. Now suppose that the inequality does not hold, the case where action must be taken. First assume that $\mathcal{A}_d$ is 1-Lipschitz. Then it suffices to make R_Θ γ-Lipschitz. As noted previously,

R_Θ takes the form of a composition of ResNet blocks. For simplicity, suppose

$$R_\Theta = \mathrm{I} + \phi_a(Wu + b) \tag{16}$$

for a matrix W and vector b defined in terms of the weights Θ. Let $C_3 \triangleq \gamma C_1/C_2$. To make (15) hold, it would be sufficient to replace R_Θ with $C_3 \cdot R_\Theta$. Furthermore,

$$C_3 R_\Theta = C_3\big(\mathrm{I} + \phi_a(Wu + b)\big) \tag{17a}$$

$$= \mathrm{I} + C_3\phi_a(Wu + b) + (C_3 - 1)\mathrm{I} \tag{17b}$$

$$\approx \mathrm{I} + \phi_a\big(C_3(Wu + b)\big) + (C_3 - 1)\mathrm{I} \tag{17c}$$

$$\approx \mathrm{I} + \phi_a\big(C_3(Wu + b)\big), \tag{17d}$$

where the first approximation is an equality when $Wu + b \geq 0$ (and approximately equal when a is small), and the second approximation holds whenever the inequality (15) is "close" to hold *i.e.* $C_3 \approx 1$. This shows that

$$C_3 R_\Theta \approx \mathrm{I} + \phi_a\big(C_3(Wu + b)\big). \tag{18}$$

Thus, to ensure R_Θ is approximately γ-Lipschitz, we may do the following. After each forward pass in training (*i.e.* computing $\mathcal{N}_\Theta(d)$ for a batch B of data d), we compute C_1 and C_2 as above. If (15) holds, then no action is taken. If (15) does not hold, then multiply the weights W and b by C_3, making (15) hold.

In our experiments, the structure of R_Θ was a more complicated variation of the above case (namely, the residual portion was the composition of convolutions). However, we used the same normalization factor, which forces R_Θ to be slightly more contractive than needed. And, in the general case where R_Θ is the composition of mappings of the form identity plus residual, it suffices to multiply the weights by the normalization constant C_3 raised to one over the number of layers ℓ in the residual mapping (*i.e.* $C_3^{1/\ell}$).

Remark 7.1 An important note must be made with respect to normalization. Namely, R_Θ was almost never updated by the procedure above. Because of the initialization of the weights Θ, R_Θ appears to have been roughly 1-Lipschitz. And, because the weights are tuned to improve the performance of R_Θ, it appears that this typically resulted in updates that did not make R_Θ less contractive. Consequently, the above is an approximate safeguard, but did not appear necessary in practice to obtain our results.

A.5 TV minimization

We equivalently rewrite the problem (TVM) as

$$\min_{u,p,w} \delta_{[0,1]^n}(u) + \|p\|_1 + \delta_{B(d,\varepsilon)}(w) \quad \text{such that} \quad \begin{bmatrix} D_+ \\ A \end{bmatrix} u - \begin{bmatrix} p \\ w \end{bmatrix} = 0, \tag{19}$$

where D_+ is the concatenation of forward difference operators along each image axis. Using a change of variables $\xi = (p, w)$, defining the function

$$f(\xi) \triangleq \|p\|_1 + \delta_{B(d,\varepsilon)}(w), \tag{20}$$

and setting $M = [D_+; A]$, we rewrite (19) as

$$\min_{u,\xi} \delta_{[0,1]^n}(u) + f(\xi) \quad \text{such that } Mu - \xi = 0. \tag{21}$$

Observe that (21) follows the standard form of ADMM-type problems. For scalars $\alpha, \beta, \lambda \in (0, \infty)$, linearized ADMM [118] updates take the form

$$u^{k+1} = P_{[0,1]^n}\left(u^k - \beta M^\top\left(\nu^k + \alpha\left(Mu^k - \xi^k\right)\right)\right), \tag{22a}$$

$$\zeta^{k+1} = \operatorname{prox}_{\lambda f}\left(\xi^k + \lambda\left(\nu^k + \alpha\left(Mu^{k+1} - \xi^k\right)\right)\right), \tag{22b}$$

$$\nu^{k+1} = \nu^k + \alpha\left(Mu^{k+1} - \xi^{k+1}\right). \tag{22c}$$

Expanding terms, we obtain the explicit formulae

$$r^k = D_+^\top\left(\nu_1^k + \alpha\left(D_+u^k - p^k\right)\right) + A^\top\left(\nu_2^k + \alpha\left(Au^k - w^k\right)\right), \tag{23a}$$

$$u^{k+1} = P_{[0,1]^n}\left(u^k - \beta r^k\right), \tag{23b}$$

$$p^{k+1} = \eta_\lambda\left(p^k + \lambda\left(\nu_1^k + \alpha\left(D_+u^{k+1} - p^k\right)\right)\right), \tag{23c}$$

$$w^{k+1} = P_{B(d,\varepsilon)}\left(w^k + \lambda\left(\nu_2^k + \alpha\left(Au^{k+1} - w^k\right)\right)\right), \tag{23d}$$

$$\nu_1^{k+1} = \nu_1^k + \alpha\left(D_+u^{k+1} - p^{k+1}\right), \tag{23e}$$

$$\nu_2^{k+1} = \nu_2^k + \alpha\left(Au^{k+1} - w^{k+1}\right), \tag{23f}$$

where $B(d, \varepsilon)$ is the Euclidean ball of radius ε centered at d and η_λ is the soft thresholding operator with parameter λ *i.e.*

$$\eta_\lambda(u) \triangleq \begin{cases} u - \lambda & \text{if } x \geq \lambda, \\ u + \lambda & \text{otherwise.} \end{cases} \tag{24}$$

We set $u^1 = 0$, $\nu^1 = 0$, $p^1 = D_+u^1$, and $w^1 = Au^1$. For the ellipses experiment, we use $\alpha = \beta = \lambda = 0.1$, $\varepsilon = 10$, and 250 iterations. For the LoDoPab experiment, we use $\alpha = \beta = \lambda = 0.1$, $\varepsilon = 5$, and 250 iterations. Note the computational costs of computing each signal estimate via TVM is greater than FBP and TVS.

Acknowledgements
We thank Daniel Mckenzie and Qiuwei Li for their helpful feedback prior to submitting our paper.

Funding
Samy Wu Fung is supported by AFOSR MURI FA9550-18-1-0502, AFOSR Grant No. FA9550-18-1-0167, and ONR Grants N00014-18-1-2527 snf N00014-17-1-21. Howard Heaton is supported by the National Science Foundation (NSF) Graduate Research Fellowship under Grant No. DGE-1650604. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

Availability of data and materials
All data can be downloaded in the following link:
https://drive.google.com/drive/folders/1Z0A3c-D4dnrhlXM8cpgC1b7Ltyu0wpgQ?usp=sharing. The data can also be accessed in the github link where the code is provided.

Declarations

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors contributed equally and significantly in this research work. All authors read and approved the final manuscript.

Publisher's Note

Received: 29 April 2021 Accepted: 12 October 2021 Published online: 22 November 2021

References

1. Arridge, S.R.: Optical tomography in medical imaging. Inverse Probl. **15**(2), 41 (1999)
2. Arridge, S.R., Schotland, J.C.: Optical tomography: forward and inverse problems. Inverse Probl. **25**(12), 123010 (2009)
3. Hansen, P.C., Nagy, J.G., O'leary, D.P.: Deblurring Images: Matrices, Spectra, and Filtering. SIAM, Philadelphia (2006)
4. Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An iterative regularization method for total variation-based image restoration. Multiscale Model. Simul. **4**(2), 460–489 (2005)
5. Bauschke, H.H., Combettes, P.L., Luke, D.R.: Phase retrieval, error reduction algorithm, and fienup variants: a view from convex optimization. JOSA A **19**(7), 1334–1345 (2002)
6. Candes, E.J., Eldar, Y.C., Strohmer, T., Voroninski, V.: Phase retrieval via matrix completion. SIAM Rev. **57**(2), 225–251 (2015)
7. Fung, S.W., Di, Z.W.: Multigrid optimization for large-scale ptychographic phase retrieval. SIAM J. Imaging Sci. **13**(1), 214–233 (2020)
8. Bui-Thanh, T., Ghattas, O., Martin, J., Stadler, G.: A computational framework for infinite-dimensional Bayesian inverse problems part I: the linearized case, with application to global seismic inversion. SIAM J. Sci. Comput. **35**(6), 2494–2523 (2013)
9. Fung, S.W., Ruthotto, L.: A multiscale method for model order reduction in PDE parameter estimation. J. Comput. Appl. Math. **350**, 19–34 (2019)
10. Fung, S.W., Ruthotto, L.: An uncertainty-weighted asynchronous ADMM method for parallel PDE parameter estimation. SIAM J. Sci. Comput. **41**(5), 129–148 (2019)
11. Haber, E., Ascher, U., Aruliah, D., Oldenburg, D.: Fast simulation of 3D electromagnetic problems using potentials. J. Comput. Phys. **163**(1), 150–171 (2000)
12. Haber, E., Ascher, U.M., Oldenburg, D.W.: Inversion of 3D electromagnetic data in frequency and time domain using an inexact all-at-once approach. Geophysics **69**(5), 1216–1228 (2004)
13. Kan, K., Fung, S.W., Ruthotto, L.: Pnkh-b: a projected Newton–Krylov method for large-scale bound-constrained optimization. SIAM J. Sci. Comput. **0**, 704–726 (2021)
14. Cucker, F., Smale, S.: Best choices for regularization parameters in learning theory: on the bias-variance problem. Found. Comput. Math. **2**(4), 413–428 (2002)
15. Fung, S.W.: Large-scale parameter estimation in geophysics and machine learning. PhD thesis, Emory University (2019)
16. Haber, E., Ruthotto, L.: Stable architectures for deep neural networks. Inverse Probl. **34**(1), 014004 (2017)
17. Vito, E.D., Rosasco, L., Caponnetto, A., Giovannini, U.D., Odone, F.: Learning from examples as an inverse problem. J. Mach. Learn. Res. **6**, 883–904 (2005)
18. Wu Fung, S., Tyrväinen, S., Ruthotto, L., Haber, E.: ADMM-Softmax: an ADMM approach for multinomial logistic regression. Electron. Trans. Numer. Anal. **52**, 214–229 (2020)
19. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
20. Candes, E.J., Romberg, J.: Quantitative robust uncertainty principles and optimally sparse decompositions. Found. Comput. Math. **6**(2), 227–254 (2006)
21. Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**(2), 489–509 (2006)
22. Donoho, D.L.: Compressed sensing. IEEE Trans. Inf. Theory **52**(4), 1289–1306 (2006)
23. Calvetti, D., Reichel, L.: Tikhonov regularization of large linear problems. BIT Numer. Math. **43**(2), 263–283 (2003)
24. Golub, G.H., Hansen, P.C., O'Leary, D.P.: Tikhonov regularization and total least squares. SIAM J. Matrix Anal. Appl. **21**(1), 185–194 (1999)
25. Chan, R.H., Kan, K.K., Nikolova, M., Plemmons, R.J.: A two-stage method for spectral–spatial classification of hyperspectral images. J. Math. Imaging Vis. **62**, 790–807 (2020)
26. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Phys. D, Nonlinear Phenom. **60**(1–4), 259–268 (1992)
27. Adler, J., Öktem, O.: Learned primal-dual reconstruction. IEEE Trans. Med. Imaging **37**(6), 1322–1332 (2018)
28. Kobler, E., Klatzer, T., Hammernik, K., Pock, T.: Variational networks: connecting variational methods and deep learning. In: German Conference on Pattern Recognition, pp. 281–293. Springer, Berlin (2017)
29. Lunz, S., Öktem, O., Schönlieb, C.-B.: Adversarial regularizers in inverse problems. In: Advances in Neural Information Processing Systems, pp. 8507–8516. Curran Associates, Red Hook (2018)
30. Chan, S.H., Wang, X., Elgendy, O.A.: Plug-and-play ADMM for image restoration: fixed-point convergence and applications. IEEE Trans. Comput. Imaging **3**(1), 84–98 (2016)
31. Cohen, R., Elad, M., Milanfar, P.: Regularization by denoising via fixed-point projection (red-pro) (2020). arXiv preprint 2008.00226
32. Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model based reconstruction. In: 2013 IEEE Global Conference on Signal and Information Processing, pp. 945–948. IEEE Press, New York (2013)
33. Xu, L., Ren, J.S., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. Adv. Neural Inf. Process. Syst. **27**, 1790–1798 (2014)
34. Jin, K.H., McCann, M.T., Froustey, E., Unser, M.: Deep convolutional neural network for inverse problems in imaging. IEEE Trans. Image Process. **26**(9), 4509–4522 (2017)

35. Moeller, M., Mollenhoff, T., Cremers, D.: Controlling neural networks via energy dissipation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3256–3265 (2019)
36. Censor, Y., Chen, W., Combettes, P.L., Davidi, R., Herman, G.T.: On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints. Comput. Optim. Appl. **51**(3), 1065–1088 (2012)
37. Censor, Y., Elfving, T., Herman, G.T., Nikazad, T.: On diagonally relaxed orthogonal projection methods. SIAM J. Sci. Comput. **30**(1), 473–504 (2008)
38. Gordon, D., Gordon, R.: Component-averaged row projections: a robust, block-parallel scheme for sparse linear systems. SIAM J. Sci. Comput. **27**(3), 1092–1117 (2005)
39. Censor, Y., Segal, A.: Iterative projection methods in biomedical inverse problems. Mathematical methods in biomedical imaging and intensity-modulated radiation therapy. IMRT **10**, 65–96 (2008)
40. Censor, Y., Cegielski, A.: Projection methods: an annotated bibliography of books and reviews. Optimization **64**(11), 2343–2358 (2015)
41. Bauschke, H.H., Koch, V.R.: Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. Contemp. Math. **636**, 1–40 (2015)
42. Ordoñez, C.E., Karonis, N., Duffin, K., Coutrakon, G., Schulte, R., Johnson, R., Pankuch, M.: A real-time image reconstruction system for particle treatment planning using proton computed tomography (pct). Phys. Proc. **90**, 193–199 (2017)
43. Penfold, S., Censor, Y., Schulte, R.W., Bashkirov, V., McAllister, S., Schubert, K.E., Rosenfeld, A.B.: Block-iterative and string-averaging projection algorithms in proton computed tomography image reconstruction. In: Censor, Y., Jiang, M., Wang, G. (eds.) Biomedical Mathematics: Promising Directions in Imaging, Therapy Planning and Inverse Problems, pp. 347–368. Medical Physics Publishing, Madison (2010)
44. Bauschke, H.H., Koch, V.R.: Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. Contemp. Math. **636**, 1–40 (2015)
45. Bauschke, H.H., Combettes, P.L., et al.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces, 2nd edn. Springer, New York (2017)
46. Bauschke, H.H., Borwein, J.M.: On projection algorithms for solving convex feasibility problems. SIAM Rev. **38**(3), 367–426 (1996)
47. Kaczmarz, S.: Angenaherte auflosung von systemen linearer gleichungen. In: Bulletin International de L'Académie Polonaise des Sciences et des Lettres A (1937)
48. Cimmino, G.: Cacolo approssimato per le soluzioni dei systemi di equazioni lineari. Ric. Sci. (Roma) **1**, 326–333 (1938)
49. Aharoni, R., Censor, Y.: Block-iterative projection methods for parallel computation of solutions to convex feasibility problems. Linear Algebra Appl. **120**, 165–175 (1989)
50. Byrne, C.L.: Block-iterative methods for image reconstruction from projections. IEEE Trans. Image Process. **5**(5), 792–794 (1996)
51. Censor, Y., Zaslavski, A.J.: Convergence and perturbation resilience of dynamic string-averaging projection methods. Comput. Optim. Appl. **54**(1), 65–76 (2013)
52. Censor, Y., Segal, A.: On the string averaging method for sparse common fixed-point problems. Int. Trans. Oper. Res. **16**(4), 481–494 (2009)
53. Censor, Y., Tom, E.: Convergence of string-averaging projection schemes for inconsistent convex feasibility problems. Optim. Methods Softw. **18**(5), 543–554 (2003)
54. Davidi, R., Herman, G.T., Censor, Y.: Perturbation-resilient block-iterative projection methods with application to image reconstruction from projections. Int. Trans. Oper. Res. **16**(4), 505–524 (2009)
55. Censor, Y., Davidi, R., Herman, G.T.: Perturbation resilience and superiorization of iterative algorithms. Inverse Probl. **26**(6), 065008 (2010)
56. Herman, G.T., Garduño, E., Davidi, R., Censor, Y.: Superiorization: an optimization heuristic for medical physics. Med. Phys. **39**(9), 5532–5546 (2012)
57. He, H., Xu, H.-K.: Perturbation resilience and superiorization methodology of averaged mappings. Inverse Probl. **33**(4), 044007 (2017)
58. Censor, Y.: Weak and strong superiorization: between feasibility-seeking and minimization. An. Ştiinţ. Univ. 'Ovidius' Constanţa, Ser. Mat. **23**(3), 41–54 (2017). https://doi.org/10.1515/auom-2015-0046
59. Schöpfer, F., Lorenz, D.A.: Linear convergence of the randomized sparse Kaczmarz method. Math. Program. **173**(1), 509–536 (2019)
60. Lorenz, D.A., Wenger, S., Schöpfer, F., Magnor, M.: A sparse Kaczmarz solver and a linearized Bregman method for online compressed sensing. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 1347–1351. IEEE Press, New York (2014)
61. Cegielski, A.: Iterative Methods for Fixed Point Problems in Hilbert Spaces, vol. 2057. Springer, Berlin (2012)
62. Krasnosel'skiĭ, M.A.: Two remarks about the method of successive approximations. Usp. Mat. Nauk **10**, 123–127 (1955)
63. Mann, R.: Mean value. Methods Iterat. **4**(3), 506–510 (1953)
64. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
65. Bengio, Y.: Learning Deep Architectures for AI. Now Publishers, Norwell (2009)
66. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)
67. Manning, C., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
68. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York (2009)
69. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. (2016). 1606.04838
70. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (Poster) (2015)
71. Fan, F., Xiong, J., Wang, G.: Universal approximation with quadratic deep networks. Neural Netw. **124**, 383–392 (2020)
72. Tabuada, P., Gharesifard, B.: Universal approximation power of deep neural networks via nonlinear control theory (2020). arXiv preprint 2007.06007
73. Fung, S.W., Heaton, H., Li, Q., McKenzie, D., Osher, S., Yin, W.: Fixed point networks: implicit depth models with Jacobian-free backprop (2021). arXiv preprint 2103.12803

74. Browder, F.E.: Nonexpansive nonlinear operators in a Banach space. Proc. Natl. Acad. Sci. **54**(4), 1041–1044 (1965) https://www.pnas.org/content/54/4/1041.full.pdf
75. Göhde, D.: Zum prinzip der kontraktiven abbildung. Math. Nachr. **30**(3–4), 251–258 (1965)
76. Kirk, W.A.: A fixed point theorem for mappings which do not increase distances. Am. Math. Mon. **72**(9), 1004–1006 (1965)
77. Krantz, S.G., Parks, H.R.: The Implicit Function Theorem: History, Theory, and Applications. Springer, Berlin (2012)
78. Bai, S., Kolter, J.Z., Koltun, V.: Deep equilibrium models. In: Advances in Neural Information Processing Systems, pp. 690–701 (2019)
79. Winston, E., Kolter, J.Z.: Monotone operator equilibrium networks. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 10718–10728. Curran Associates, Red Hook (2020) https://proceedings.neurips.cc/paper/2020/file/798d1c2813cbdf8bcdb388db0e32d496-Paper.pdf
80. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. In: Advances in Neural Information Processing Systems, pp. 6571–6583 (2018)
81. Bai, S., Koltun, V., Kolter, J.Z.: Multiscale deep equilibrium models. In: Advances in Neural Information Processing Systems 33 (2020)
82. Monga, V., Li, Y., Eldar, Y.C.: Algorithm unrolling: interpretable, efficient deep learning for signal and image processing. IEEE Signal Process. Mag. **38**(2), 18–44 (2021)
83. Chen, T., Chen, X., Chen, W., Heaton, H., Liu, J., Wang, Z., Yin, W.: Learning to optimize: a primer and a benchmark (2021). arXiv preprint 2103.12828
84. Gregor, K., LeCun, Y.: Learning fast approximations of sparse coding. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, pp. 399–406 (2010)
85. Rick Chang, J., Li, C.-L., Poczos, B., Vijaya Kumar, B., Sankaranarayanan, A.C.: One network to solve them all—solving linear inverse problems using deep projection models. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5888–5897 (2017)
86. Metzler, C., Mousavi, A., Baraniuk, R.: Learned D-AMP: principled neural network based compressive image recovery. In: Advances in Neural Information Processing Systems, pp. 1772–1783 (2017)
87. Chen, X., Liu, J., Wang, Z., Yin, W.: Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds. In: Advances in Neural Information Processing Systems, pp. 9061–9071 (2018)
88. Diamond, S., Sitzmann, V., Heide, F., Wetzstein, G.: Unrolled optimization with deep priors (2018). 1705.08041 [cs]
89. Perdios, D., Besson, A., Rossinelli, P., Thiran, J.-P.: Learning the weight matrix for sparsity averaging in compressive imaging. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3056–3060. IEEE Press, New York (2017)
90. Mardani, M., Sun, Q., Donoho, D., Papyan, V., Monajemi, H., Vasanawala, S., Pauly, J.: Neural proximal gradient descent for compressive imaging. Adv. Neural Inf. Process. Syst. **31**, 9573–9583 (2018)
91. Zhang, J., Ghanem, B.: Ista-net: interpretable optimization-inspired deep network for image compressive sensing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1828–1837 (2018)
92. Ito, D., Takabe, S., Wadayama, T.: Trainable ISTA for sparse signal recovery. IEEE Trans. Signal Process. **67**(12), 3113–3125 (2019)
93. Mardani, M., Sun, Q., Papyan, V., Vasanawala, S., Pauly, J., Donoho, D.: Degrees of freedom analysis of unrolled neural networks (2019). arXiv preprint 1906.03742
94. Putzky, P., Welling, M.: Recurrent inference machines for solving inverse problems (2017). 1706.04008 [cs]
95. Zhang, K., Zuo, W., Gu, S., Zhang, L.: Learning deep CNN denoiser prior for image restoration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3929–3938 (2017)
96. Chen, Y., Pock, T.: Trainable nonlinear reaction diffusion: a flexible framework for fast and effective image restoration. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1256–1272 (2017)
97. Sreter, H., Giryes, R.: Learned convolutional sparse coding. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2191–2195. IEEE Press, New York (2018)
98. Liu, J., Chen, X., Wang, Z., Yin, W.: ALISTA: analytic weights are as good as learned weights in LISTA. In: International Conference on Learning Representations (2019)
99. Xie, X., Wu, J., Liu, G., Zhong, Z., Lin, Z.: Differentiable linearized ADMM. In: International Conference on Machine Learning, pp. 6902–6911 (2019)
100. Meinhardt, T., Moller, M., Hazirbas, C., Cremers, D.: Learning proximal operators: using denoising networks for regularizing inverse imaging problems. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1781–1790 (2017)
101. Liu, R., Cheng, S., Ma, L., Fan, X., Luo, Z., et al.: A bridging framework for model optimization and deep propagation. Adv. Neural Inf. Process. Syst. **31**, 4318–4327 (2018)
102. Corbineau, M.-C., Bertocchi, C., Chouzenoux, E., Prato, M., Pesquet, J.-C.: Learned image deblurring by unfolding a proximal interior point algorithm. In: 2019 IEEE International Conference on Image Processing (ICIP), pp. 4664–4668. IEEE Press, New York (2019) https://doi.org/10.1109/ICIP.2019.8803438
103. Mukherjee, S., Dittmer, S., Shumaylov, Z., Lunz, S., Öktem, O., Schönlieb, C.-B.: Learned convex regularizers for inverse problems (2020). arXiv preprint 2008.02839
104. Zhang, K., Zuo, W., Zhang, L.: Deep plug-and-play super-resolution for arbitrary blur kernels. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1671–1681 (2019)
105. Li, Y., Tofighi, M., Geng, J., Monga, V., Eldar, Y.C.: Efficient and interpretable deep blind image deblurring via algorithm unrolling. IEEE Trans. Comput. Imaging **6**, 666–681 (2020)
106. Dudgeon, D.E., Mersereau, R.M.: Multidimensional digital signal processing. Prentice Hall Professional Technical Reference (1990)
107. O'Connor, D., Vandenberghe, L.: Primal-dual decomposition by operator splitting and applications to image deblurring. SIAM J. Imaging Sci. **7**(3), 1724–1754 (2014)
108. Goldstein, T., Osher, S.: The split Bregman method for L1-regularized problems. SIAM J. Imaging Sci. **2**(2), 323–343 (2009)
109. Penfold, S.N., Schulte, R.W., Censor, Y., Rosenfeld, A.B.: Total variation superiorization schemes in proton computed tomography image reconstruction. Med. Phys. **37**(11), 5887–5895 (2010)

110. Humphries, T., Winn, J., Faridani, A.: Superiorized algorithm for reconstruction of CT images from sparse-view and limited-angle polyenergetic data. Phys. Med. Biol. **62**(16), 6762 (2017)
111. Leuschner, J., Schmidt, M., Baguer, D.O., Maaß, P.: The LoDoPaB-CT dataset: a benchmark dataset for low-dose CT reconstruction methods (2019). arXiv preprint 1910.01113
112. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, pp. 8026–8037 (2019)
113. Adler, J., Kohr, H., Öktem, O.: (2017). Operator Discretization Library (ODL)
114. Heaton, H., Censor, Y.: Asynchronous sequential inertial iterations for common fixed points problems with an application to linear systems. J. Glob. Optim. **74**(1), 95–119 (2019)
115. Adler, J., Öktem, O.: Solving ill-posed inverse problems using iterative deep neural networks. Inverse Probl. **33**(12), 124007 (2017)
116. Lie, J., Nordbotten, J.M.: Inverse scale spaces for nonlinear regularization. J. Math. Imaging Vis. **27**(1), 41–50 (2007)
117. Censor, Y.: Superiorization and perturbation resilience of algorithms: a continuously updated bibliography (2021). arXiv preprint 1506.04219
118. Ryu, E., Yin, W.: Large-Scale Convex Optimization: Algorithm Designs via Monotone Operators. Cambridge University Press, Cambridge (2022) https://large-scale-book.mathopt.com
119. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
120. Heaton, H., Fung, S.W., Lin, A.T., Osher, S., Yin, W.: Wasserstein-based projection with applications to inverse problems (2020). arXiv preprint 2008.02200