Solving Inverse Problems Using Implicit Networks and Jacobian-free Backpropagation

Linghai Liu[*], Shuaicheng Tong[†], Lisa Zhao[‡], Samy Wu Fung [§]

July 24, 2022

Abstract

Recent efforts in applying implicit networks to solve inverse problems in imaging have achieved competitive or even better results when compared with feedforward networks. Moreover, they require less memory and can theoretically be of infinite depth. However, implicit networks are not necessarily easy to train with their fixed point due to the presence and computation of a large linear system. This paper introduces Jacobian-free Backpropagation (JFB), a fast and easy-to-implement scheme for backpropagation that circumvents such calculation and is applicable to image deblurring tasks. Our results show that JFB is effective and gives competitive results against fine-tuned optimization schemes, state-of-the-art (SOTA) feedforward methods, and existing implicit networks.

1 Introduction

Inverse problems consist of recovering a signal (e.g. an image, a parameter of a PDE, etc.) from indirect, noisy measurements. These problems arise in many applications such as medical imaging, geophysical imaging, etc.

Traditional approaches for solving inverse problems use deep unrolling; this method utilizes a fixed number of iterations which is usually chosen heuristically and can often lead to inaccuracies [4, 8]. Moreover, they are challenging to train due to memory constraints, as the activations of each layer are saved for backpropagation. Another proposed method [11] uses denoising convolutional neural networks (DnCNNs) with batch normalization and residual learning to learn how to remove the clean underlying image.

Consequently, deep equilibrium models (DEQs) were proposed [1]. These models substitute traditional feed-forward networks with implicit networks. Implicit networks backpropagate through a fixed point of the weight-tying layers, which allows them to maintain constant memory costs. Although infinite-depth networks have fixed memory costs, they are very expensive to train. This is because backpropagating through implicit networks requires the computation of a Jacobian-based linear system for every gradient evaluation [8]. Recently, a Jacobian-Free Backpropagation (JFB) approach was introduced to avoid solving the linear system by estimating the gradient [3].

---

[*]Brown University
[†]University of California, Los Angeles
[‡]University of California, Berkeley
[§]Colorado School of Mines

The theory of JFB allows us to replace the Jacobian matrix with the identity under certain conditions. By doing so, we approximate the gradient instead of directly inverting the Jacobian. JFB not only maintains a fixed memory cost, but also avoids a substantial computational cost while ensuring a descent direction. JFB has performed well on image classification tasks [3]. In this paper, we investigate its effectiveness in training implicit networks for inverse problems arising in image deblurring [1].

2 Mathematical Background

2.1 Inverse Problem Setup

Consider the case where we have N noisy blurred images $\{d_i\}_{i=1}^N$ $(d_i \in \mathbb{R}^m)$ that we refer to as *measurements*. The underlying *original images*, denoted as $\{x_i\}_{i=1}^N$ $(x_i \in \mathbb{R}^n)$, are hidden from the experimenter(s).
We use the following model [8]:
$$d = \mathcal{A}x + \varepsilon \tag{1}$$
where $\mathcal{A}$ is a mapping from signal space $\mathbb{R}^n$ of original images to measurement space $\mathbb{R}^m$ and $\varepsilon \in \mathbb{R}^m$ is a noise term that models measurement errors.

2.2 Traditional Optimization with Inverse Problems

A natural idea is to apply $\mathcal{A}^{-1}$ to Eq. 1 and obtain $x^* = \mathcal{A}^{-1}(d - \varepsilon)$ when $\mathcal{A}$ is invertible. This can amplify the noise and result in really bad reconstructions. Therefore, we solve for the true image x^* by formulating it as an optimization problem that minimizes the difference between the reconstructed image and observed image:

$$x^* = \underset{x \in \mathbb{R}^n}{\arg\min} \ \frac{1}{2}||\mathcal{A}x - d||_{L^2}^2 + \lambda R(x) \tag{2}$$

where $R(x)$ is a regularizer chosen based on prior knowledge of the data, and $\lambda > 0$ is a tunable parameter. We can solve (2) by applying the gradient descent algorithm, with a common choice of $R(x) = \frac{1}{2}||x||_{L^2}^2$. However, from our numerical experiments, the results are not ideal (low SSIM values for reconstructed images), which led to the exploration of implicit networks.

2.3 Implicit Network and Challenges

Implicit networks [2] are newly proposed models that also leverage the dataset $\{(d_i, x_i)\}_{i=1}^N$ and can represent a wide range of feedforward models. The idea is to find a fixed point for their weight-tying layers and map it to the inference space (output). We formulate our implicit network as follows:

$$\begin{aligned}[\text{Equilibrium equation}] \quad & T_\Theta(x) = x \\ [\text{Prediction equation}] \quad & x^*(d) = x\end{aligned} \tag{3}$$

The iteration to find the fixed point reads:
$$x_i^{k+1} = \underbrace{x_i^k - \eta \left(\nabla_x ||\mathcal{A}x_i^k - d_i||_{L^2}^2 + S_\Theta(x_i^k) \right)}_{:=T_\Theta(x_i^k)} \tag{4}$$

---

[1] Access the GitHub repository at `https://github.com/lliu58b/Jacobian-free-Backprop-Implicit-Networks`

where $\eta > 0$ is step size, K is the number of iterations (layers) in our neural network $\mathcal{N}_\Theta(\cdot)$, and $S_\Theta(\cdot)$ is a trainable network containing all the weights of $\mathcal{N}_\Theta(\cdot)$. Note that the equilibrium equation in (3) is satisfied as $K \to \infty$ when $T_\Theta(\cdot)$ is a contraction.

Given a noisy measurement d, we find its fixed point x^*, and define $\mathcal{N}_\Theta(d) := x^*$.

To train the weights Θ, we perform implicit differentiation on the equilibrium equation in (3)

$$\frac{dx^*}{d\Theta} = \frac{dT_\Theta(x^*)}{dx^*}\frac{dx^*}{d\Theta} + \frac{\partial T_\Theta(x^*)}{\partial \Theta} \xRightarrow{\text{rearrange terms}} \left(I - \frac{dT_\Theta(x^*)}{dx^*}\right)\frac{dx^*}{d\Theta} = \frac{\partial T_\Theta(x^*)}{\partial \Theta}$$

and substitute $\frac{dx^*}{d\Theta}$ into the gradient descent scheme after setting up a loss function ℓ:

$$\Theta \leftarrow \Theta - \alpha \frac{d\ell}{dx^*}\left(I - \frac{dT_\Theta(x^*)}{dx^*}\right)^{-1}\frac{\partial T_\Theta(x^*)}{\partial \Theta} \tag{5}$$

where $\alpha > 0$ is the learning rate and we call $(I - \frac{dT_\Theta(x^*)}{dx^*})$ the Jacobian. This update rule is costly due to the need of inverting a Jacobian, which motivates the search for other ways to optimize the backpropagation.

3 Related Works

3.1 Deep Unrolling for Inverse Problems

Deep Unrolling is used to unpack deep neural networks, whose black box nature hinders its development in networks with very large training nets. In our setup, this method corresponds to Eq. 4, with $K < +\infty$ being fixed. K is eventually a small number as a result of fine-tuning among all stages, many handcrafted parameters, and limitations in time and space for both training and testing.

3.2 Implicit Networks

Deep Equilibrium Models (DEQs), a type of implicit networks, were proposed in [1]. The advantage of DEQ lies in that it requires less memory because it only has one layer of weights—it solves for the fix-point and uses implicit differentiation to calculate the gradient for backpropagation. On the other hand, SOTA deep feed-forward networks such as Deep Unrolling have memory issues since they store intermediate values while iterating through each network layer.

3.3 Combining Implicit Networks with Classic Optimization

Three optimization techniques for implicit networks were introduced in [4] to combat the memory limitations in deep unrolling methods:

i. Gradient descent. This is formulated in Eq. 4

ii. Proximal gradient descent. The model learns the optimization problem, i.e.,

$$x^{k+1} = S_\Theta(x^k - \eta \mathcal{A}^T(\mathcal{A}x^k - d)), \quad \text{where } \eta > 0 \text{ is the step size}$$

3

iii. Alternating directions methods of multipliers (ADMM). Here an auxiliary variable is introduced and a fixed point is a long vector concatenated by both this auxiliary variable and the latent variable of the image.

Numerical experiments of these methods performed on CelebA [7] and fastMRI datasets [10, 6] have shown better benchmarks than deep unrolling methods [4].

4 Proposed Methodology

The convergence criterion of $T_\Theta(\cdot)$ in Eq. 4 (DE-GRAD) is discussed in more detail in [4], where $S_\Theta - I$ needs to be $\epsilon-$Lipschitz to ensure that $T_\Theta(\cdot)$ is contraction with parameter $\gamma \in [0, 1)$. We propose using Jacobian-free Backpropagation (JFB), a backpropagation algorithm introduced in [3], that has lower computational cost when updating Θ. The idea is to circumvent the Jacobian calculation in (5). $\left(I - \frac{dT_\Theta(x^*)}{dx^*}\right)$ is replaced with the identity I, leaving us with an approximation of the true gradient:

$$p_\Theta = \frac{d\ell}{dx^*} \frac{\partial T_\Theta(x^*)}{\partial \Theta} \tag{6}$$

which is still a descending direction for the loss function ℓ with more constraints on T_Θ [3]: (i) T_Θ is continuously differentiable with respect to Θ (ii) $M := \frac{\partial T_\Theta}{\partial \Theta}$ has full column rank and the condition number of $M^T M$ does not exceed $\frac{1}{\gamma}$.

Even though JFB is not always performing the steepest descent, its gradient is much less costly to calculate, which makes its overall cost lower while ensuring descending.

Note that $\left(I - \frac{dT_\Theta(x^*)}{dx^*}\right)^{-1}$ can be expanded using Neumann series:

$$\left(I - \frac{dT_\Theta(x^*)}{dx^*}\right)^{-1} = \sum_{j=0}^{\infty} \left(\frac{dT_\Theta(x^*)}{dx^*}\right)^j \tag{7}$$

where JFB only takes the first term of this expansion.

This greatly simplifies the implementation using PyTorch. We no longer need to hand calculate the succeeding terms of Eq. 7 and storing them.

Algorithm 1 Learning with JFB

Require: Implicit network $N_\Theta(\cdot)$ with weight-tying layers $T_\Theta(\cdot)$. Set learning rate $\alpha > 0$.
  for measurement-truth pair (d, x) in training set **do**
    Find fixed point: $x^* = T_\Theta(x^*; d)$ with `torch.no_grad()`
    Output: $\mathcal{N}_\Theta(d) = T_\Theta(x^*)$
    Calculate loss: $\ell(x^*, x)$
    Update: $\Theta \leftarrow \Theta - \alpha \frac{d\ell}{dx^*} \frac{\partial T_\Theta(x^*)}{\partial \Theta}$
  end for
---

5 Numerical Experiment

We mimic the format in [4] while implementing our JFB approach.

- **Data**: We use a subset of size 10,000 of the CelebA dataset [7], which contains around 200,000 centered human faces with annotations. Among them, 8,000 images are used for training and the rest are left for validating and testing purposes.

- **Preprocessing**: Each image is resized into 128×128 pixels with 3 channels (RGB). The blurred images are generated using Gaussian blurring kernels of size 5×5 with variance 5. The measurements are then crafted by adding white Gaussian noise with standard deviation $\sigma = 10^{-2}$ to the blurred images.

- **Training**: With the aforementioned data and preprocessing specifics, the training strictly follows Algorithm 1, where $T_\Theta(\cdot)$ is the same as defined in Eq. 4. As part of T_Θ, the trainable network $S_\Theta(\cdot)$ is pretrained as also practiced in [4] to observe an improvement in reconstruction.

- **Visualization**: We first visualize the average training loss per image over the number of epochs in Fig. 1.
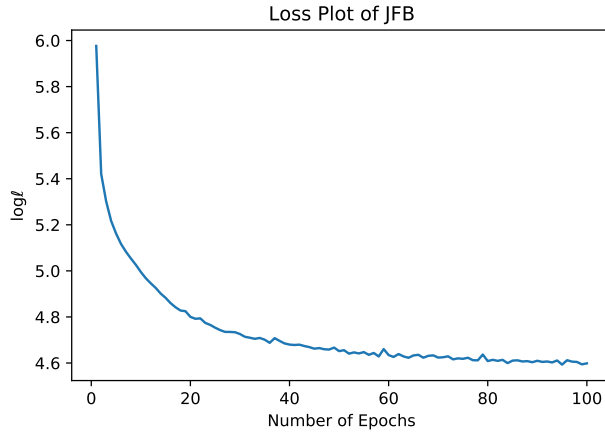
Figure 1: Loss of the DE-GRAD model, trained with JFBs

We see that as the number of epochs increases, the loss decreases. However, we are still in the process of fine tuning our parameters and expect to see even better results. We then visualize examples from the DE-GRAD model we trained using JFB in Table 1. We use the same four images for each row. Ground Truths are the images in CelebA dataset resized to 128×128 pixels. Noisy Blurred Images are generated by a 5×5 Gaussian kernel mentioned above. Direct Inverse images are the results of applying the inverse of $\mathcal{A}$, which is defined in Eq. 1. Gradient Descent images are obtained by applying gradient descent using mean squared error (MSE) as the loss function. We enforce early stopping so that the results are not corrupted. JFB images are obtained using Noisy Blurred images as inputs with JFB-trained weights for the DE-GRAD model.

- **Comparison**: We compare our results using JFB with other state-of-the-art methods using total variation (TV), standard deep neural networks, and DEQ. The metrics we use to assess the quality of reconstructed images are [5]: the peak-signal-to-noise ratio (PSNR, a positive
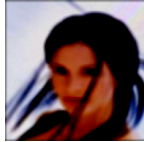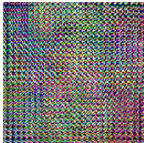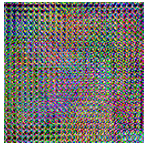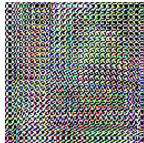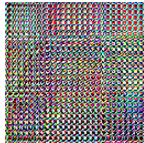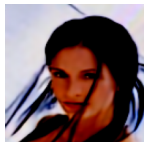
Ground Truth				
Noisy Blurred Image				
Direct Inverse				
Gradient Descent				
JFB				

Table 1: Visualization of JFB

number, best at $+\infty$) and the structural similarity index measure (SSIM, also positive, best at 1). The data in Table 2 are calculated on the testing dataset of size 2000.

	Total Variation	Plug-n-Play [9]	Deep Equilibrium [4]	JFB (Ours)
PSNR	26.79	29.77	32.43	26.88
SSIM	0.86	0.88	0.94	0.91

Table 2: Comparison across Models

Although we have not yet achieved results better than DEQs, which finds the true gradient in 5 in a complicated manner, we are observing results that are competitive with other techniques.

6 Conclusion

In this paper, we introduce Jacobian-free Backpropagation into implicit networks in the context of recovering an image from noisy and blurry images. The technique proves to be effective, in that its rudiment version recovers the images well across the testing dataset and are competitive with other state-of-the-art methods whose hand-crafted parameters are fine-tuned across stages. We are still training our model and tuning hyperparameters to obtain better performance.

7 Acknowledgements

References

[1] S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

[2] L. El Ghaoui, F. Gu, B. Travacca, A. Askari, and A. Tsai. Implicit deep learning. *SIAM Journal on Mathematics of Data Science*, 3(3):930–958, 2021.

[3] S. W. Fung, H. Heaton, Q. Li, D. Mckenzie, S. Osher, and W. Yin. Jfb: Jacobian-free backpropagation for implicit networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6648–6656, Jun. 2022.

[4] D. Gilton, G. Ongie, and R. Willett. Deep equilibrium architectures for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:1123–1133, 2021.

[5] A. Hore and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[6] F. Knoll, J. Zbontar, A. Sriram, M. J. Muckley, M. Bruno, A. Defazio, M. Parente, K. J. Geras, J. Katsnelson, H. Chandarana, Z. Zhang, M. Drozdzalv, A. Romero, M. Rabbat, P. Vincent, J. Pinkerton, D. Wang, N. Yakubova, E. Owens, C. L. Zitnick, M. P. Recht, D. K. Sodickson, and Y. W. Lui. fastmri: A publicly available raw k-space and dicom dataset of knee images for accelerated mr image reconstruction using machine learning. *Radiology: Artificial Intelligence*, 2(1):e190007, 2020.

[7] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[8] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

[9] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.

[10] J. Zbontar, F. Knoll, A. Sriram, T. Murrell, Z. Huang, M. J. Muckley, A. Defazio, R. Stern, P. Johnson, M. Bruno, et al. fastmri: An open dataset and benchmarks for accelerated mri. *arXiv preprint arXiv:1811.08839*, 2018.

[11] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.