Energy Dissipation with Plug-and-Play Priors

Hendrik Sommerhoff\textsuperscript{1} Andreas Kolb\textsuperscript{1} Michael Moeller\textsuperscript{2}
\textsuperscript{1}Computer Graphics Group, University of Siegen
\textsuperscript{2}Computer Vision Group, University of Siegen
\{hendrik.sommerhoff, andreas.kolb, michael.moeller\}@uni-siegen.de

Abstract

Neural networks have reached outstanding performance for solving various ill-posed inverse problems in imaging. However, drawbacks of end-to-end learning approaches in comparison to classical variational methods are the requirement of expensive retraining for even slightly different problem statements and the lack of provable error bounds during inference. Recent works tackled the first problem by using networks trained for Gaussian image denoising as generic plug-and-play regularizers in energy minimization algorithms. Even though this obtains state-of-the-art results on many tasks, heavy restrictions on the network architecture have to be made if provable convergence of the underlying fixed point iteration is a requirement. More recent work has proposed to train networks to output descent directions with respect to a given energy function with a provable guarantee of convergence to a minimizer of that energy. However, each problem and energy requires the training of a separate network. In this paper we consider the combination of both approaches by projecting the outputs of a plug-and-play denoising network onto the cone of descent directions to a given energy. This way, a single pre-trained network can be used for a wide variety of reconstruction tasks. Our results show improvements compared to classical energy minimization methods while still having provable convergence guarantees.

1 Introduction

In many image processing tasks an observed image \( f \) is modeled as the result of the transformation of a clean image \( \hat{u} \) under a known (linear) operator \( A \) and unknown noise \( \xi \),

\[ f = A\hat{u} + \xi. \tag{1} \]

In most cases, the problem of reconstructing \( \hat{u} \) from \( f \) and \( A \) is ill-posed and can thus not be solved by a simple inversion of \( A \), giving rise to the field of regularization theory with iterative or variational methods, see e.g. [2] for an overview. In recent years neural networks were very successful in learning a direct mapping \( G(f) \approx \hat{u} \) for a variety of problems such as deblurring [32][28], denoising [34], super-resolution [3], demosaicing [9] and MRI- or CT-reconstruction [33][14]. Even though this works well in practice, there are rarely any guarantees on the behaviour of neural networks on unseen data, making them difficult to use in safety-critical applications. Moreover, for each problem and type of noise a separate network has to be trained.

In contrast, classical variational methods try to find the solution by the minimization of a suitable energy function of the form

\[ \hat{u} = \arg\min_u H_f(u) + \alpha R(u) \tag{2} \]

where \( H_f \) is a data fidelity term, for example commonly chosen as \( H_f(u) = \frac{1}{2} \| Au - f \|^2 \), and \( R \) is a regularization function that models prior knowledge about the solution, e.g. the popular total variation (TV) regularization, \( R(u) = \| \nabla u \|_1 \). [24]. While minimizers of (2) come with many desirable theoretical guarantees, regularizations like the TV often cannot perfectly capture the complex structure of the space of natural images.
To combine the advantages of powerful feed-forward networks and model-based approaches like (2), authors have considered various hybrid models like learning regularizers (e.g. [23][1][11][8]), designing networks architectures that resemble the structure of minimization algorithms or differential equations, e.g. [25][36][15][6], interleaving networks with classical optimization steps [16][17], or using the parametrization of networks as a regularization for (2), see e.g. [29][12].

A particularly flexible approach arises from [7][37][30][13], where proximal operators with respect to the regularizer are replaced by arbitrary denoising operators, with recent works focusing on the use of denoising networks [18][4][35]. While such approaches allow to tackle different inverse problems with the same neural network, the derivation of theoretical guarantees - even in terms of the convergence of the resulting algorithmic scheme - remains difficult, see [3][27] or some discussion in [20], unless the denoiser satisfies particular properties [22].

2 Provably dissipating model-based energies with plug-and-play priors

The starting point of the above-mentioned algorithmic schemes that utilize denoising networks to regularize model-based inverse problems are methods for the minimization of (2). While most works focus on primal-dual / ADMM approaches, their convergence analysis is quite delicate even in a setting in which one still minimizes (nonconvex) energies, such that we turn to two simpler methods, gradient descent and proximal gradient methods,

\[
\begin{align*}
  u_{k+1}^{(3)} &= u_k - \tau (\nabla H_f(u_k) + \alpha \nabla R(u_k)), \\
  u_{k+1}^{(4)} &= \text{prox}_{\tau R}(u_k - \tau \nabla H_f(u_k)),
\end{align*}
\]

where \( \text{prox}_{\tau R}(v) = \arg\min_u \frac{1}{2} \|u - v\|^2 + \tau R(u) \). Following the idea of [7][37][30][13], considering either a gradient descent or a proximal step on the regularization as a generic denoising operation gives rise to the following two algorithmic schemes,

\[
\begin{align*}
  u_{k+1}^{(5)} &= \frac{1}{2} \rho(u_k; \tau) + \frac{1}{2} \mathcal{G}(u_k), \\
  u_{k+1}^{(6)} &= \mathcal{G}(\rho(u_k; \tau)),
\end{align*}
\]

where \( \mathcal{G} \) denotes any kind of denoiser, e.g. a convolutional neural network, and we define \( \rho(u_k; \tau) = u_k - \tau \nabla H_f(u_k) \) for the sake of brevity of notation. We refer to [20] for a more detailed derivation.

Algorithmic schemes like (5) or (6) combine the model-based flexibility of energy minimization methods (i.e. explicit modelling of \( H_f \)) with the expressive power of deep neural networks \( \mathcal{G} \).

Unfortunately – despite their success in various practical applications – schemes like (5) or (6) remain dangerous to be used: Figure 1 shows the result of running the iteration (6) with \( H_f = 0 \) on a noisy input image \( f = u^0 \) for 100 and 800 iterations using a DnCNN [34] preimplemented in Matlab as the denoiser \( \mathcal{G} \). As we can see the image gets completely distorted. Even more strikingly, the range of the image increased from values in \([0, 1]\) to an interval of \([-185, 218]\) within the first 1000 iterations. Clearly, the algorithmic scheme diverges.

A natural condition for the provable convergence of a scheme like (6) (at least along subsequences) would be a 1-Lipschitz continuous operator \( \mathcal{G} \). There has been previous work on computing upper
We tested our implementation with the image reconstruction tasks of Gaussian deblurring with 
standard deviation 1 and 4 times single image super resolution. In both cases we added Gaussian noise 
with standard deviation 0.02 to the corrupted image. We chose the PyTorch implementation of DnCNN [34] 
pre-trained on a noise level of 0.1 as our denoising network.

Our surrogate energy uses a TV regularization with Huber-norm instead of the $\ell^1$-norm. As our 
data term we choose $H_f(u) = \frac{1}{2}||Au - f||^2$ with $A$ either being the blurring or downsampling

\footnote{https://github.com/SaoYan/DnCNN-PyTorch}
operator. The best hyperparameters for all methods were found with a grid search. In all experiments, for scheme (conv), \( \alpha = 0 \) was the best choice for any \( \tau \), indicating that the gradient descent step on the data term does not yield much additional information, assumably because the projection onto \( C(\gamma, \nabla E(u)) \) which depends on the gradient of the data term anyway. When using (prox), we empirically found \( \tau = 30 \) to be the best choice. For the projection onto the half-space of descent directions, we used \( \gamma = 5 \) for both methods for deblurring, and \( \gamma = 50 \) in (conv) and \( \gamma = 1 \) in (prox) for super resolution. For fairness, the classical gradient descent was also implemented using backtracking line search.

Figure 2 shows the reconstruction quality of the current iterate compared to ground truth over a span of 500 iterations. The PSNR quickly peaks before slowly converging to the fixed point of the surrogate energy which is consistent with the results of [21]. Notably, the convex combination method peaks earlier but not as high as the prox method. Tables 1 and 2 show results using early stopping using a discrepancy principle. On all test images our (prox) scheme beats gradient descent.

Table 1: PSNR values for deblurring for varying images and stopping criteria. The algorithm was stopped when \( H_f(u^k) < \beta H_f(\hat{u}) \). best refers to the highest PSNR over 500 iterations and a "*" means that the stopping criterion was not triggered such that the last iteration was used instead.

<table>
<thead>
<tr>
<th>Method</th>
<th>CameraMan</th>
<th>House</th>
<th>Peppers</th>
<th>Starfish</th>
<th>Butterfly</th>
<th>Plane</th>
<th>Bird</th>
<th>Lena</th>
<th>Barbara</th>
<th>Boat</th>
<th>Man</th>
<th>Couple</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv, ( \beta = 1 )</td>
<td>21.56</td>
<td>24.10</td>
<td>22.13</td>
<td>22.06</td>
<td>22.20</td>
<td>20.74</td>
<td>20.85</td>
<td>24.88</td>
<td>21.97</td>
<td>25.22</td>
<td>26.35</td>
<td>24.94</td>
</tr>
<tr>
<td>conv, ( \beta = 0.9 )</td>
<td>23.00</td>
<td>27.50</td>
<td>23.67</td>
<td>23.60</td>
<td>23.53</td>
<td>22.34</td>
<td>22.50</td>
<td>28.49</td>
<td>23.45</td>
<td>25.32</td>
<td>26.56</td>
<td>25.06</td>
</tr>
<tr>
<td>GD, best</td>
<td>22.23</td>
<td>26.55</td>
<td>22.94</td>
<td>23.01</td>
<td>21.85</td>
<td>21.87</td>
<td>21.40</td>
<td>27.05</td>
<td>23.22</td>
<td>24.94</td>
<td>26.08</td>
<td>24.82</td>
</tr>
</tbody>
</table>

Table 2: PSNR values for super resolution for varying images and stopping criteria. The algorithm was stopped when \( H_f(u^k) < \beta H_f(\hat{u}) \). best refers to the highest PSNR over 500 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>CameraMan</th>
<th>House</th>
<th>Peppers</th>
<th>Starfish</th>
<th>Butterfly</th>
<th>Plane</th>
<th>Bird</th>
<th>Lena</th>
<th>Barbara</th>
<th>Boat</th>
<th>Man</th>
<th>Couple</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv, ( \beta = 1 )</td>
<td>23.44</td>
<td>28.57</td>
<td>24.61</td>
<td>24.41</td>
<td>24.98</td>
<td>22.86</td>
<td>23.27</td>
<td>29.21</td>
<td>23.60</td>
<td>25.87</td>
<td>26.88</td>
<td>25.39</td>
</tr>
<tr>
<td>conv, ( \beta = 0.9 )</td>
<td>23.50</td>
<td>28.78</td>
<td>24.62</td>
<td>24.54</td>
<td>25.09</td>
<td>22.91</td>
<td>23.35</td>
<td>29.38</td>
<td>23.67</td>
<td>25.96</td>
<td>27.05</td>
<td>25.40</td>
</tr>
<tr>
<td>GD, best</td>
<td>22.23</td>
<td>26.55</td>
<td>22.94</td>
<td>23.01</td>
<td>21.85</td>
<td>21.87</td>
<td>21.40</td>
<td>27.05</td>
<td>23.22</td>
<td>24.94</td>
<td>26.08</td>
<td>24.82</td>
</tr>
</tbody>
</table>

4 Conclusion

We combine deep learning and energy minimization methods for solving inverse problems in image reconstruction into a provably convergent algorithmic scheme. Still, our approach is able to generalize to different problems with a single denoising network and without the need to retrain if that problem changes. We were able to reach better results than the energy minimization baseline in our experiments, and are happy to elaborate on the above aspects in the NeurIPS workshop.
References


