

IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

---

**Beyond Wavelets: Wavelet-Inspired Invertible Neural  
Networks for Image Modelling and Approximation**

---

Author: **Andreas Floros**

CID: **01539636**

Supervisor: **Pier Luigi Dragotti**

Second Marker: **Patrick A. Naylor**

I affirm that I have submitted, or will submit, an electronic copy of my final year project report to the provided EEE link.

I affirm that I have submitted, or will submit, an identical electronic copy of my final year project to the provided Blackboard module for Plagiarism checking.

I affirm that I have provided explicit references for all the material in my Final Report that is not authored by me, but is represented as my own work.

## **Acknowledgements**

I would like to express my gratitude to Pier Luigi Dragotti for guiding me throughout this research and for providing valuable insights. To my parents, for their unconditional support and for believing in me.

## **Abstract**

Image Restoration (IR) consists of a family of ill-posed inverse problems which have been studied extensively through the lens of model and learning-based methods. Recent advances combining the two approaches have allowed denoisers to solve general IR problems, thus reducing the complexity of the tasks to just denoising. In this work, a multipurpose image processing tool is developed based on wavelet theory and Sparse Coding (SC). The proposed method exploits the recursive nature of wavelet transforms to achieve high generalisation ability and provides a powerful denoising engine. The utility of the proposed extends beyond traditional Additive White Gaussian Noise (AWGN) removal; this flexibility is demonstrated in compact representation of images, progressive loading, spatially variant noise removal, deblurring and inpainting.



# Contents

<b>1</b>	<b>Introduction</b>	<b>6</b>
	Overview . . . . .	6
	Contribution . . . . .	6
	Structure . . . . .	6
<b>2</b>	<b>Background</b>	<b>7</b>
	Image Restoration . . . . .	7
	Maximum A Posteriori Problem . . . . .	7
	Plug-and-Play Prior . . . . .	7
	Sparse Coding . . . . .	8
	Invertible Transforms . . . . .	9
	Wavelets . . . . .	9
	Normalising Flows . . . . .	10
<b>3</b>	<b>Analysis and Design</b>	<b>11</b>
	Overview . . . . .	11
	Recursive Wavelet Neural Network . . . . .	12
	Lifting Inspired Neural Network . . . . .	12
	Noise Estimation Network . . . . .	14
	Fusion Network . . . . .	15
	Iterative Thresholding Algorithm . . . . .	15
<b>4</b>	<b>Implementation</b>	<b>16</b>
	Overview . . . . .	16
	Data, Network and Training Details . . . . .	17
	Data Preparation . . . . .	17
	Optimisation Schedule . . . . .	17
	Hyperparameter Tuning . . . . .	17
<b>5</b>	<b>Testing</b>	<b>18</b>
	Properties of the Networks . . . . .	18
	Scale Invariance . . . . .	18
	Noise Invariance and Adaptivity . . . . .	19
	Ablation Study . . . . .	22
	Initialisation . . . . .	22
	Tweaking the Priors . . . . .	23
	Scaling the Transform . . . . .	24
<b>6</b>	<b>Results</b>	<b>25</b>
	Progressive Loading and Energy Compaction . . . . .	25
	Image Denoising . . . . .	26
	Towards General Problems . . . . .	27
	Image Deblurring . . . . .	27
	Image Inpainting . . . . .	29
<b>7</b>	<b>Evaluation and Conclusions</b>	<b>30</b>
	Limitations . . . . .	30
	Future Work . . . . .	30
	Conclusion . . . . .	30
	<b>Bibliography</b>	<b>31</b>

# 1 Introduction

## Overview

Advances in deep learning have enabled the use of large architectures which are capable of State-Of-The-Art (SOTA) performance in a variety of imaging tasks. Nevertheless, image modelling and approximation is broad and end-to-end learning-based approaches suffer from a lack of generalisation ability and a lack of interpretability as one typically has to train separate networks for different tasks. These observations call for the development of general, multipurpose image processing tools.

This paper focuses on Image Restoration (IR), a long-standing problem with highly practical value in various low-level vision applications [1, 2]. In this context, recent works have combined model-based optimisation with learning-based approaches to solve general problems [3, 4, 5, 1, 6]; focus has been placed on constructing rich image priors which can be transferred across tasks without the need of additional network training. These advances reveal that a robust, non-blind denoising algorithm is central to IR.

Image denoising is a fundamental inverse problem which has recently seen major breakthroughs: in [7], a fusion network was adopted to tackle blind and universal image denoising which infers noise level information. Zhang et al. [8] showed that feeding noise level maps into Convolutional Neural Networks (CNNs) can boost performance and allow for handling of spatially variant noise. Mohan et al. [9] showed that the generalisation ability of denoisers can be significantly boosted by ensuring that the networks are homogeneous, i.e.,  $f(a\mathbf{x}) = af(\mathbf{x})$  for  $a > 0$ . In [10] and [11] it was shown that Sparse Coding (SC) based architectures with adjustable thresholds are able further improve performance; these results demonstrate the relevance of domain knowledge in the deep learning era and suggest that simply scaling black box algorithms is inefficient and sub-optimal. Theoretically motivated and carefully designed algorithms have become essential in IR and are the focus of this paper.

## Contribution

This work pushes the limits of the above domain-aware designs to construct a general image processing tool for restoration tasks. Given the importance of denoisers in IR, the developed algorithm is primarily focused on noise removal so that it may be adapted for other tasks by using existing model-inspired IR frameworks. Specifically, a Recursive Wavelet Neural Network (RWNN) is proposed; it defines a family of non-linear transforms which are trained to annihilate natural images, analogously to the polynomial annihilation observed in traditional wavelets. RWNN can perform a multiresolution analysis of arbitrary depth and has strong generalisation ability with high interpretability. To tackle the denoising task, the proposed is coupled with a Convolutional Sparse Coding Network (CSCN). In short, the contribution of this work is three-fold:

- RWNN is a non-redundant and non-linear invertible transform. It inherits key wavelet properties while leveraging the approximation power of CNNs to significantly outperform traditional wavelets in energy compaction and progressive loading tasks.
- When combined with a sparsity-driven denoising network, RWNN achieves performances comparable to those of SOTA algorithms in blind image denoising with a highly competitive parameter count. Further, the proposed algorithm is flexible as it can handle spatially varying noise.
- RWNN is powerful enough to be integrated in IR frameworks and solve general inverse problems; this is demonstrated in the tasks of image deblurring and inpainting. Specifically, the proposed achieves deblurring performances similar to those of SOTA algorithms with minimal manual hyperparameter tuning.

## Structure

The rest of this paper is organised as follows: Section 2 examines the aforementioned IR frameworks and gives an overview of the construction of non-linear transforms. Section 3 is concerned with architectural details of RWNN and auxiliary networks. Section 4 focuses on training details, data preparation and hyperparameter tuning. Section 5 explores the properties of the resulting networks and also includes an ablation study. Section 6 compares the performance of the proposed method to existing schemes. Finally, Section 7 provides discussion on the proposed, highlights areas of interest for future work and draws the conclusions of this research.

## 2 Background

### Image Restoration

#### Maximum A Posteriori Problem

Traditional IR frameworks assume a degraded observation,  $\mathbf{y}$ , of a clean image,  $\mathbf{x}$ , as input. One may write  $\mathbf{y} = f(\mathbf{x}) + \sigma\mathbf{n}$  where  $f$  is a noise-independent degradation function and  $\sigma\mathbf{n}$  is white noise with standard deviation  $\sigma$ . Given this observation, an IR algorithm attempts to find the most likely associated clean image,  $\hat{\mathbf{x}}$ , by solving the following Maximum A Posteriori (MAP) problem:

$$\hat{\mathbf{x}} \in \operatorname{argmax}_{\mathbf{x}} \operatorname{Prob}(\mathbf{x}|\mathbf{y}) \quad (1)$$

Rewriting the above and recognising that  $\operatorname{Prob}(\mathbf{y})$  is independent of  $\mathbf{x}$ :

$$\hat{\mathbf{x}} \in \operatorname{argmax}_{\mathbf{x}} \frac{\operatorname{Prob}(\mathbf{y}|\mathbf{x}) \operatorname{Prob}(\mathbf{x})}{\operatorname{Prob}(\mathbf{y})} \equiv \operatorname{argmax}_{\mathbf{x}} \operatorname{Prob}(\mathbf{y}|\mathbf{x}) \operatorname{Prob}(\mathbf{x}) \quad (2)$$

Finally, by the monotonicity of the logarithm, the above is equivalent to:

$$\hat{\mathbf{x}} \in \operatorname{argmin}_{\mathbf{x}} -\log \operatorname{Prob}(\mathbf{y}|\mathbf{x}) - \log \operatorname{Prob}(\mathbf{x}) \quad (3)$$

Two components are identified; the data fidelity term,  $-\log \operatorname{Prob}(\mathbf{y}|\mathbf{x})$ , is dictated by the distribution of the noise. Assuming Additive White Gaussian Noise (AWGN), this is equivalent to  $\frac{1}{2\sigma^2} \|f(\mathbf{x}) - \mathbf{y}\|_2^2$ . Learning-based approaches usually leave the prior term,  $\lambda\Phi(\mathbf{x}) = -\log \operatorname{Prob}(\mathbf{x})$ , unspecified or loosely defined as there is no explicit description of the distribution of clean images. Therefore, the objective takes the form:

$$\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \lambda\Phi(\mathbf{x}) \quad (4)$$

#### Plug-and-Play Prior

The Plug-and-Play Prior (P3) technique [3, 1, 6] can be used to solve the MAP with an implicit  $\Phi$ . It addresses the uncertainty introduced by the prior by splitting the above minimisation problem in two. Specifically, the optimisation is augmented with an auxiliary variable,  $\mathbf{z}$ , and the objective becomes:

$$\min_{\substack{\mathbf{x}, \mathbf{z} \\ \mathbf{x}=\mathbf{z}}} \frac{1}{2\sigma^2} \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \lambda\Phi(\mathbf{z}) \quad (5)$$

Using the Half-Quadratic Splitting (HQS) approach [1], one may simplify the above as:

$$\min_{\mathbf{x}, \mathbf{z}} \frac{1}{2\sigma^2} \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \lambda\Phi(\mathbf{z}) + \frac{\mu}{2} \|\mathbf{z} - \mathbf{x}\|_2^2 \quad (6)$$

where, for  $\mu$  sufficiently large, this is equivalent to the original MAP. One can iteratively solve for  $\hat{\mathbf{x}}$  by fixing one variable and optimising for the other. Fixing  $\mathbf{z}$  yields the data sub-problem:

$$\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \frac{\mu}{2} \|\mathbf{z} - \mathbf{x}\|_2^2 \quad (7)$$

Fixing  $\mathbf{x}$  yields a denoising sub-problem with noise level  $\sqrt{\frac{\lambda}{\mu}}$ :

$$\min_{\mathbf{z}} \frac{1}{2 \left(\sqrt{\frac{\lambda}{\mu}}\right)^2} \|\mathbf{z} - \mathbf{x}\|_2^2 + \Phi(\mathbf{z}) \quad (8)$$

Observe that the data sub-problem is decoupled from the prior and can therefore be solved in closed form (for reasonable  $f$ ). The denoising sub-problem is assumed to be solved by an arbitrary denoiser and the prior term is implicitly defined by the denoising algorithm. In practice, SOTA learning-based IR algorithms leverage the P3 framework to transfer highly complex prior knowledge from one task (denoising) to another. In this way, general IR may be solved by a single network.

## Sparse Coding

While P3 uses an implicit prior to solve the MAP, it is also possible to formulate an explicit prior and obtain satisfactory results by following a model-based approach. A common assumption in imaging tasks is that of sparsity; images are structured in the sense that they span a high-dimensional manifold and one can exploit this structure to design an appropriate sparsifying transform. In doing so, structured and high-dimensional data may be mapped to relatively few large coefficients. It is therefore convenient to tackle IR problems in the transformed domains where the sparsity assumption allows for an explicit prior  $\Phi$ . Denoting  $\mathbf{W}_s$  as the synthesis kernel or inverse transform operator and  $\mathbf{g}$  as the transformed image, the optimal image is constructed by  $\hat{\mathbf{x}} = \mathbf{W}_s \hat{\mathbf{g}}$  where, for some  $\lambda > 0$ ,  $\hat{\mathbf{g}}$  satisfies:

$$\hat{\mathbf{g}} \in \underset{\mathbf{g}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{y} - \mathbf{W}_s \mathbf{g}\|_2^2 + \lambda \|\mathbf{g}\|_0 \quad (9)$$

The above problem may be solved using the Iterative Hard Thresholding (IHT) algorithm. The  $\ell_0$  pseudo-norm is often replaced with the  $\ell_1$  norm which leads to a convex relaxation of the problem and the Iterative Shrinkage Thresholding Algorithm (ISTA) [12]. It is of interest to examine the underlying thresholding operators: IHT employs  $\mathcal{H}_\lambda(x) = x1_{|x|>\lambda}$  whereas ISTA uses  $\mathcal{S}_\lambda(x) = \operatorname{sgn}(x) \max(0, |x| - \lambda)$ . It should be noted that, in the case of the  $\ell_1$  prior and dictionary learning,  $\mathbf{W}_s$  must be regularised in order to prevent arbitrary scaling.

**Non-negative sparse coding:** The SC optimisation can be modified by imposing a non-negativity constraint [13]. That is, the optimal codes will satisfy:

$$\hat{\mathbf{g}} \in \underset{\mathbf{g} \geq \mathbf{0}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{y} - \mathbf{W}_s \mathbf{g}\|_2^2 + \lambda \Phi(\mathbf{g}) \quad (10)$$

With this additional constraint, the thresholding operators are changed to  $\mathcal{H}_\lambda^+(x) = x1_{x>\lambda}$  for the  $\ell_0$  prior and  $\mathcal{S}_\lambda^+(x) = \max(0, x - \lambda)$  for the  $\ell_1$  prior. Note that  $\mathcal{S}_\lambda^+$  is commonly used in deep learning under the name of Rectified Linear Unit (ReLU) as it provides a robust non-linearity. Under some mild assumptions, one can interpret a single ReLU layer as a SC optimisation: assume  $\mathbf{W}_s$  is orthonormal and let  $\mathbf{W}_a$  denote the analysis kernel which is orthogonal to  $\mathbf{W}_s$ , i.e., the forward transform operator. Then SC is equivalent to:

$$\hat{\mathbf{g}} \in \underset{\mathbf{g} \geq \mathbf{0}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{W}_a \mathbf{y} - \mathbf{g}\|_2^2 + \lambda \Phi(\mathbf{g}) \quad (11)$$

The above can be solved in closed form with  $\mathcal{H}_{\sqrt{2\lambda}}^+(\mathbf{W}_a \mathbf{y})$  for the  $\ell_0$  prior and with  $\mathcal{S}_\lambda^+(\mathbf{W}_a \mathbf{y})$  for  $\ell_1$ . Note that the non-negativity constraint ensures a non-linear transformation of  $\mathbf{y}$  for all  $\lambda$ ; unlike unconstrained SC, this approach can reliably construct highly non-linear functions while also being theoretically motivated. In the context of deep learning, the  $\ell_1$  prior offers a more flexible construction compared to the  $\ell_0$  prior since the latter does not allow gradients to propagate to  $\lambda$ .

**Convolutional sparse coding networks:** In general, the SC problem may be solved with an Iterative Thresholding Algorithm (ITA) which takes the form:

$$\mathbf{g}_{t+1} = \operatorname{Prox}_{\mu\lambda\Phi(\cdot)} \left( \mathbf{g}_t + \mu \mathbf{W}_s^\top (\mathbf{y} - \mathbf{W}_s \mathbf{g}_t) \right), \quad 0 \leq t < T \quad (12)$$

where  $\operatorname{Prox}_{\gamma\Phi(\cdot)}(\mathbf{g}) := \operatorname{argmin}_{\mathbf{z}} \frac{1}{2} \|\mathbf{z} - \mathbf{g}\|_2^2 + \gamma\Phi(\mathbf{z})$  denotes the proximal operator,  $\top$  denotes the adjoint and  $\mu > 0$  is a step size which is typically bounded according to the singular values of  $\mathbf{W}_s$ . Usually  $\mathbf{g}_0 \in \{\mathbf{0}, \mathbf{W}_s^\top \mathbf{y}\}$ . Note that Equation 12 is also valid for convolutional operators and therefore it is straightforward to unroll  $T$  iterations and train a CNN end-to-end via gradient descent on an objective function. To boost performance, each iteration may use different parameters, slightly deviating from the theoretical model and leading to more general Residual Networks (ResNets) [14, 15, 16]. Indeed, several works [10, 11, 17] have used this construction in the context of IR and have reported highly competitive results with reduced parameter counts.

## Invertible Transforms

### Wavelets

Wavelets have been very effective in many image tasks such as compression [18, 19], denoising [20, 21, 22], deconvolution [23, 24] and inpainting [25, 26]. They are supported by the rich theory of multiresolution analysis which grants them time-frequency localisation and are therefore often superior compared to the standard Fourier transform. One can define a wavelet by considering Perfect Reconstruction (PR) filter banks as shown below:

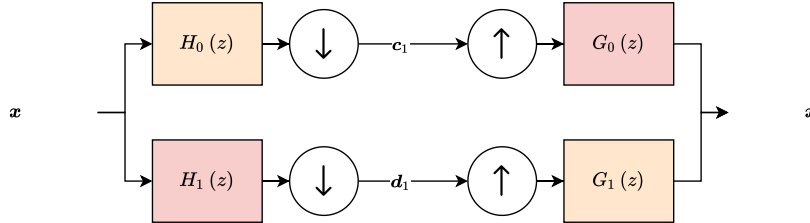


Figure 1: One level of PR filter bank.

Here, the arrows denote downsampling and upsampling by a factor of two while  $\{G_0(z), H_0(z), G_1(z), H_1(z)\}$  represent the  $Z$ -transforms of linear filters which are such that PR is satisfied, i.e., the system is the identity mapping. The intermediate signals,  $c_1$  and  $d_1$ , are referred to as coarse and detail (sparse) coefficients respectively and are an encoding of the original signal. The indices denote the decomposition level / scale which can be increased by repeating the filter bank on the coarse as shown in Figure 2.

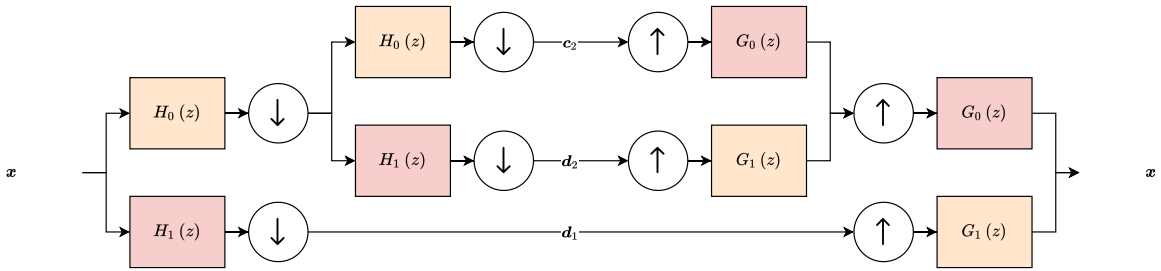


Figure 2: Two levels of PR filter bank.

**The choice of filters:** Note that this setup is not unique; one can construct orthogonal or biorthogonal transforms with the conditions  $H_0(z)G_0(z) + H_0(-z)G_0(-z) = 2$ ,  $G_1(z) = \mp z^{-1}H_0(-z)$  and  $H_1(z) = \mp zG_0(-z)$ . By controlling the number of zeros of  $G_1(z)$  at  $z = 1$ , one can shape the approximation properties of the transforms and achieve polynomial annihilation. That is, polynomial signals may be compactly represented with trivial detail coefficients. This property is especially noteworthy since it offers theoretical guarantees for the compact representation of uniformly Lipschitz signals [27].

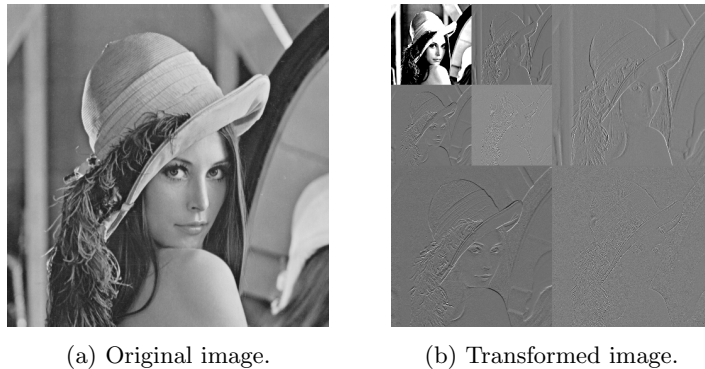


Figure 3: Example of two level decomposition using the Haar filters.

## Normalising Flows

Normalising flows [28, 29] are bijective mappings with a tractable probability change which can be evaluated through a relatively easy to compute Jacobian determinant. They have gained popularity in variational inference [30] and density estimation [31] tasks. Specifically, by treating the input,  $\mathbf{x}$ , as a random variable with a probability density function (pdf)  $p_{\mathcal{X}}(\mathbf{x})$ , flow based models,  $f$ , are able to map  $\mathbf{x}$  to a latent variable,  $\mathbf{z} = f(\mathbf{x})$ , with its pdf satisfying:

$$p_{\mathcal{X}}(\mathbf{x}) = p_{\mathcal{Z}}(\mathbf{z}) \left| \det \left( \frac{\partial \mathbf{z}}{\partial \mathbf{x}} \right) \right| \quad (13)$$

where  $\mathbf{J}_f = \frac{\partial \mathbf{z}}{\partial \mathbf{x}}$  denotes the Jacobian of  $f$ . In doing so, one typically learns the input distribution,  $\mathcal{X}$ , by performing log-likelihood maximisation with a chosen latent distribution  $\mathcal{Z}$  (often Gaussian):

$$\min_f -\log p_{\mathcal{X}}(\mathbf{x}) \equiv \min_f -\log p_{\mathcal{Z}}(\mathbf{z}) - \log |\det(\mathbf{J}_f)| \quad (14)$$

Compared to other generative schemes such as Variational AutoEncoders (VAEs) [32] and Generative Adversarial Networks (GANs) [33], normalising flows are typically simpler in their training and have the advantage of providing analytical expressions for the desired distributions. However, because they are invertible, their approximation capabilities are often limited and the requirement of a tractable  $\det(\mathbf{J}_f)$  further constraints the flows. Moreover, one may also require that the computations of  $f$  and  $f^{-1}$  are equally efficient.

**Affine coupling:** A common flow architecture for representation learning is that of affine coupling or real-valued Non-Volume Preserving transformations (real NVP) [34] which allows for efficient computation of  $f$  and  $f^{-1}$ . Specifically, let  $\mathbf{x}_1$  and  $\mathbf{x}_2$  be a partition of the input  $\mathbf{x}$ . Real NVP maps  $\mathbf{x}_1 \rightarrow \mathbf{z}_1$  and  $\mathbf{x}_2 \rightarrow \mathbf{z}_2$  as follows:

$$\mathbf{z}_1 = \exp(f_1(\mathbf{x}_2)) \odot \mathbf{x}_1 + f_2(\mathbf{x}_2) \quad (15)$$

$$\mathbf{z}_2 = \exp(g_1(\mathbf{z}_1)) \odot \mathbf{x}_2 + g_2(\mathbf{z}_1) \quad (16)$$

where  $\odot$  denotes the Hadamard product and  $f_1, f_2, g_1, g_2$  are arbitrary functions. The inverse operation is:

$$\mathbf{x}_2 = \exp(-g_1(\mathbf{z}_1)) \odot (\mathbf{z}_2 - g_2(\mathbf{z}_1)) \quad (17)$$

$$\mathbf{x}_1 = \exp(-f_1(\mathbf{x}_2)) \odot (\mathbf{z}_1 - f_2(\mathbf{x}_2)) \quad (18)$$

One can stack multiple layers of real NVP to create more complicated transforms. Especially noteworthy in the context of this work is the case of additive coupling [31], i.e., when  $f_1, g_1$  are the zero function; this results in volume preserving transforms with  $\det(\mathbf{J}_f) = 1$  which will be discussed in later sections. In general, there are more exotic constructions which trade simplicity and tractability for more expressivity [35]. The reader is referred to [28] for a detailed analysis on these architectures.

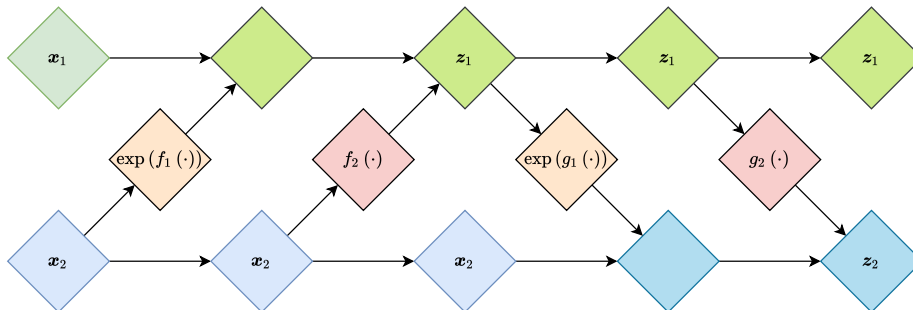


Figure 4: Forward operation of real NVP.

### 3 Analysis and Design

#### Overview

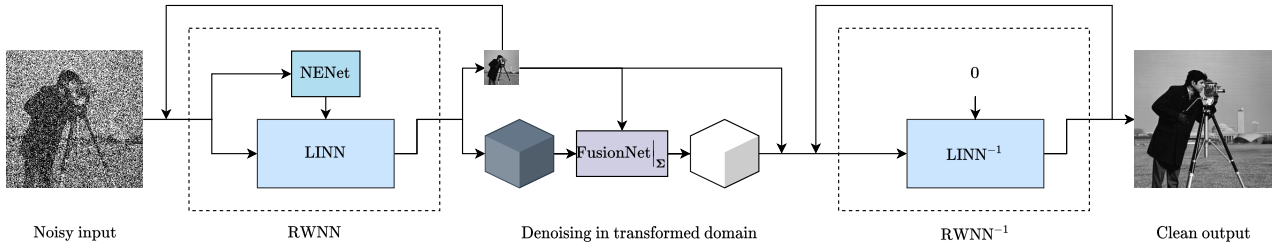


Figure 5: RWNN-F, RWNN with FusionNet.

The proposed architecture is a blind Denoising Neural Network (DNN) tailored for spatially variant Gaussian noise removal, i.e., the noise is  $\Sigma \odot \mathbf{n}$  with  $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . A high level description of the DNN is shown in Figure 5. It consists of a forward transform operator (RWNN), a processing step in the transformed domain (FusionNet) and an inverse transform (RWNN<sup>-1</sup>). RWNN consists of a Lifting Inspired Neural Network (LINN) [36] and a Noise Estimation Network (NENet) which predicts noise level maps  $\Sigma$ . It is responsible for separating the clean signal (coarse) from noisy residuals (details). In the transformed domain, a denoising sub-network based on the principles of CSCNs is used to denoise the aforementioned residuals and the result, together with the coarse, is back-projected to the original domain via RWNN<sup>-1</sup>. Note that RWNN-F may be repeated on the coarse parts to obtain better results, if the noise variance is large.

---

#### Algorithm 1: Denoising with RWNN-F.

---

**Input:** Noisy image  $\mathbf{y} = \mathbf{x} + \Sigma_{\mathbf{y}} \odot \mathbf{n}$

$\mathbf{c}_{\mathbf{y}}, \mathbf{d}_{\mathbf{y}}, \hat{\Sigma}_{\mathbf{y}} = \text{RWNN}(\mathbf{y})$  // Divide denoising problem into sub-problems

$\hat{\mathbf{c}}_{\mathbf{x}} = \text{RWNN-F}(\mathbf{c}_{\mathbf{y}})$  // Conquer coarse sub-problem recursively ( $\Sigma_{\hat{\mathbf{c}}_{\mathbf{x}}} \ll \hat{\Sigma}_{\mathbf{y}}$ ), if  $\Sigma_{\hat{\mathbf{c}}_{\mathbf{x}}} < \epsilon$  return  $\mathbf{c}_{\mathbf{y}}$

$\hat{\mathbf{d}}_{\mathbf{x}} = \text{FusionNet}(\hat{\mathbf{c}}_{\mathbf{x}}, \mathbf{d}_{\mathbf{y}}, \hat{\Sigma}_{\mathbf{y}})$  // Conquer details sub-problem with FusionNet and prior knowledge from  $\hat{\mathbf{c}}_{\mathbf{x}}$

$\hat{\mathbf{x}} = \text{RWNN}^{-1}(\hat{\mathbf{c}}_{\mathbf{x}}, \hat{\mathbf{d}}_{\mathbf{x}}, \mathbf{0})$  // Combine solutions and back-project

**Output:** Denoised image  $\hat{\mathbf{x}}$ .

---

From an algorithmic perspective, the proposed employs a divide and conquer approach as shown above. The denoising task is transformed into two sub-problems: denoising the coarse and denoising the details. The intuition is that the coarse will be much less noisy. Therefore, denoising it is an easier problem which can be solved recursively and FusionNet can use the coarse solutions as a prior to ease the denoising of the details.

**Decoupling the transform from the denoiser:** The proposed DNN consists of physically meaningful sub-networks which are of interest individually. Specifically, it is desirable for the learned RWNN to be useful beyond noise separation; inspired from the polynomial annihilation property of wavelets [27], it is of interest to approximate a natural image annihilator. In practice, such a transform has excellent energy compaction properties while also enabling a kind of super-resolution algorithm [37] and thus a wider range of applications. To ensure that the coarse representations of RWNN are physically meaningful, a noise invariance property is enforced by assuming  $\Sigma = \mathbf{0}$  in RWNN<sup>-1</sup>, i.e., denoising is discouraged in back-projection as this should be an independent operation.

**Processing in transformed domain:** Noting that RWNN is invertible and that denoising is an ill-posed inverse problem, it is not possible to perform denoising with just the learned transform. To remedy this, non-invertible FusionNet is used in the transformed domain. In general, this network takes as input the transformed signal and outputs predictions for the details, i.e., the part of the signal that is corrupted by noise. In [11], which follows a similar framework, the denoising module operates strictly on the details to make predictions. Here, it is proposed that the full signal be processed as the denoised coarse parts can act as a prior to significantly boost the prediction accuracy. In particular, the proposed architecture of FusionNet is based on the principles of fusion denoising as highlighted in [7] while also being inspired by SC methods.

# Recursive Wavelet Neural Network

## Lifting Inspired Neural Network

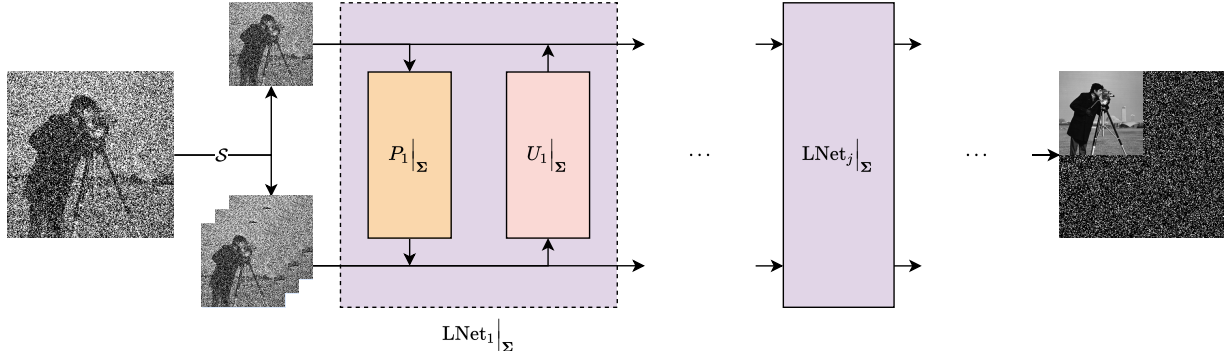


Figure 6: LINN conditioned on  $\Sigma$ .

LINNs are invertible functions composed of a split operator ( $\mathcal{S}$ ) and  $J$  lifting networks (LNETs). They effectively parameterise the space of invertible wavelet-like transforms and are also referred to as Second Generation Wavelet Transforms (SGWT) [38], additive coupling networks and NICE [31] networks in the literature. Their connection with wavelets follows from [39, 38] where traditional wavelets were proven to follow the structure of LINNs. While there are obviously many choices for invertible mappings [28], some of which are much more expressive, it is noted that LINNs are stable under recursion as they have a Jacobian determinant  $\det(\mathbf{J}_f) = 1$  and are therefore efficient in capturing the desired hypothesis class for RWNN. In this work, the LINN construction is augmented by introducing an additional input noise map,  $\Sigma$ , to the network. In doing so, a family of invertible transforms is learned with only a single set of parameters. A visual summary of the application of LINN is shown in Figure 6. The building blocks of the transformation are described next.

**Split operator:**  $\mathcal{S}$  can be any invertible function that partitions the output space into two. Denoting  $\mathbf{c}_k^J$  as the input with noise map  $\Sigma_k$ , LINN first applies  $\mathcal{S}$  on  $\mathbf{c}_k^J$  as follows:

$$\mathcal{S}(\mathbf{c}_k^J) = \begin{bmatrix} \mathbf{c}_{k+1}^0 \\ \mathbf{d}_{k+1}^0 \end{bmatrix} \quad \mathcal{S}^{-1} \left( \begin{bmatrix} \mathbf{c}_{k+1}^0 \\ \mathbf{d}_{k+1}^0 \end{bmatrix} \right) = \mathbf{c}_k^J \quad (19)$$

In wavelets,  $\mathcal{S}$  is a lazy operator which separates the input samples based on their parity. Noting that the dimensions of the output space are unconstrained, one may construct a redundant or a non-redundant transform by designing an appropriate  $\mathcal{S}$ . In this discussion, a non-redundant split is considered with the details space being three times larger than the coarse space, i.e., if  $\mathbf{c}_k^J \in \mathbb{R}^{N \times C \times H \times W}$  then  $\mathbf{c}_{k+1}^0 \in \mathbb{R}^{N \times C \times \lceil \frac{H}{2} \rceil \times \lceil \frac{W}{2} \rceil}$  while  $\mathbf{d}_{k+1}^0$  is non-rectangular and is composed of three separate tensors with dimensions  $N \times C \times \lfloor \frac{H}{2} \rfloor \times \lceil \frac{W}{2} \rceil$ ,  $N \times C \times \lceil \frac{H}{2} \rceil \times \lfloor \frac{W}{2} \rfloor$  and  $N \times C \times \lfloor \frac{H}{2} \rfloor \times \lfloor \frac{W}{2} \rfloor$ .

**Lifting networks:** LNETs consist of predictor networks,  $P$ , and updater networks  $U$ . As the names suggest, the predictors are responsible for predicting and sparsifying the details while the updaters are responsible for updating the coarse parts for further applications of LINN. In a chain of  $J$  LNETs, the  $j^{\text{th}}$  LNET is described by:

$$\text{LNet}_j \left( \begin{bmatrix} \mathbf{c}_{k+1}^{j-1} \\ \mathbf{d}_{k+1}^{j-1} \end{bmatrix}, \Sigma_k \right) = \begin{bmatrix} \mathbf{c}_{k+1}^{j-1} + U_j \left( \mathbf{d}_{k+1}^{j-1} - P_j \left( \mathbf{c}_{k+1}^{j-1}, \Sigma_k \right), \Sigma_k \right) \\ \mathbf{d}_{k+1}^{j-1} - P_j \left( \mathbf{c}_{k+1}^{j-1}, \Sigma_k \right) \end{bmatrix} \quad (20)$$

$$\text{LNet}_j^{-1} \left( \begin{bmatrix} \mathbf{c}_{k+1}^j \\ \mathbf{d}_{k+1}^j \end{bmatrix}, \Sigma_k \right) = \begin{bmatrix} \mathbf{c}_{k+1}^j - U_j \left( \mathbf{d}_{k+1}^j, \Sigma_k \right) \\ \mathbf{d}_{k+1}^j + P_j \left( \mathbf{c}_{k+1}^j - U_j \left( \mathbf{d}_{k+1}^j, \Sigma_k \right), \Sigma_k \right) \end{bmatrix} \quad (21)$$



**Predictors and updaters:** Note that  $P$  and  $U$  can be arbitrary functions and do not affect the invertibility of LNet. In this work, they follow the same ResNet architecture:

$$\mathbf{g}_0 = \mathbf{A} * \mathbf{x} \quad (22)$$

$$\mathbf{g}_{t+1} = \mathcal{S}_{\xi_t \circ \Sigma}^+ \left( \mathbf{g}_t + \mathbf{C}_t * \mathcal{S}_{\zeta_t \circ \Sigma}^+ (\mathbf{B}_t * \mathbf{g}_t) \right), \quad 0 \leq t < T \quad (23)$$

$$\text{ResNet}(\mathbf{x}, \Sigma) = \mathbf{D} * \mathbf{g}_T \quad (24)$$

where  $*$  denotes convolution with zero padding such that the height and width of the tensors are preserved and  $\{\mathbf{A}, \mathbf{D}\} \cup \{\mathbf{B}_t, \mathbf{C}_t, \zeta_t, \xi_t\}_{0 \leq t < T}$  are trainable parameters for each  $P_j$  and  $U_j$ . Each convolutional kernel has support  $p \times p$  and the encodings  $\mathbf{g}_t$  have  $L$  channels. Since the number of parameters of  $\mathbf{B}_t$  and  $\mathbf{C}_t$  scales poorly with  $L$ , these kernels are modelled with depth-wise separable convolution [40] to achieve a better complexity and performance trade-off [11]. The scaling of  $\zeta_t, \xi_t$  by  $\Sigma$  is justified as the optimal ‘‘universal threshold’’ is shown to be  $\propto \Sigma$  [20] and together with the choice of  $\mathcal{S}_\lambda^+$ , a robust, theoretically motivated non-linearity is achieved while also allowing the network to be homogeneous in the case of AWGN as input. The design of ResNet is summarised in Figure 7.

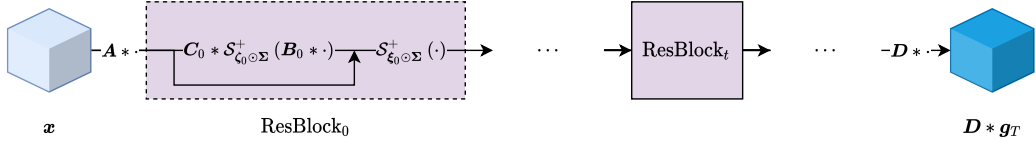


Figure 7: ResNet architecture for  $P_j$  and  $U_j$ .

The only difference between  $P$  and  $U$  networks is that their inputs and outputs have different number of channels and possibly different heights and widths (in the case of images with odd dimensions). In practice, the predictions are cropped to match the dimensions of the details while the updaters apply zero padding as a preprocessing step such that the updates match the dimensions of the coarse.

Having defined the building blocks of LINN, a one level decomposition of the signal  $\mathbf{c}_k^J$  is then given by:

$$\text{LINN}(\mathbf{c}_k^J, \Sigma_k) = \left( \bigcirc_{j=J}^1 \text{LNet}_j(\cdot, \Sigma_k) \right) \circ \mathcal{S}(\mathbf{c}_k^J) \quad (25)$$

$$\text{LINN}^{-1} \left( \begin{bmatrix} \mathbf{c}_{k+1}^J \\ \mathbf{d}_{k+1}^J \end{bmatrix}, \Sigma_k \right) = \mathcal{S}^{-1} \circ \left( \bigcirc_{j=1}^J \text{LNet}_j^{-1}(\cdot, \Sigma_k) \right) \left( \begin{bmatrix} \mathbf{c}_{k+1}^J \\ \mathbf{d}_{k+1}^J \end{bmatrix} \right) \quad (26)$$

where  $\circ$  denotes the composition. This process may be repeated by applying LINN on  $\mathbf{c}_{k+1}^J$  and to subsequent coarse tensors  $\mathbf{c}_{k+2}^J, \mathbf{c}_{k+3}^J, \dots, \mathbf{c}_{K-1}^J$  where  $K$  represents the maximum scale / depth.

**Multiscale property and generalisation:** RWNN uses the same LINN at every scale and is therefore faithful to traditional wavelet transforms which use the same filters at each scale; this recursive construction encourages a scale invariance property and allows a multiresolution analysis of arbitrary depth. As a result, RWNN has a theoretically infinite receptive field and a much more flexible understanding of the data: while capturing the notion of natural images typically requires complex setups in the form of GANs or log-likelihood maximisation in transformed domains, RWNN learns the distribution of natural images by considering coarse representations that match its input distribution (a requirement for stable recursion). Moreover, as the data is seen at different scales and noise levels, RWNN has strong generalisation ability and can form a robust prior for natural images.

## Noise Estimation Network

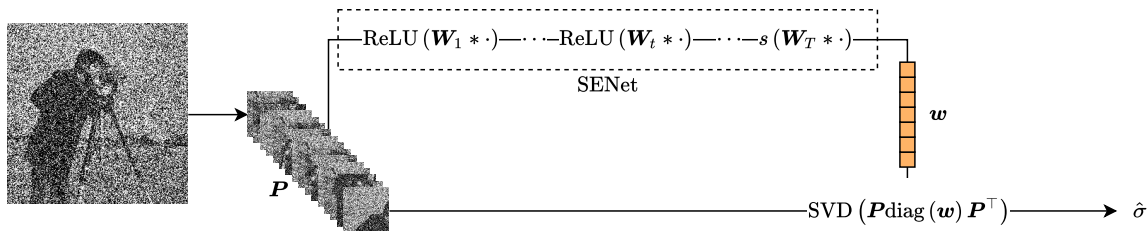


Figure 8: NENet architecture.

NENet is crucial in the construction of RWNN-F since  $\Sigma_k$  effectively parameterises the transformation of the input and controls the denoising operation at each scale. To achieve robust noise estimation, NENet converts the input image into patches,  $\mathbf{P}$ , of size  $s \times s$  which are vectorised and processed using a model-based SVD approach together with a SElection Network (SENet) as shown in Figure 8. Note that this architecture is restricted to predicting scalar  $\sigma$  which is broadcasted to a uniform  $\Sigma$ ; by focusing on this specific case, NENet can more accurately predict the noise level and therefore ease the training of RWNN-F. Moreover, it is noted that the local connectivity of CNNs allows RWNN to adapt to non-uniform  $\Sigma$  despite only being trained with uniform maps. In practice, a non-uniform  $\Sigma$  can be estimated by integrating more general SOTA noise estimators [41].

**Selection network:** In [42], an iterative algorithm for low-rank patch selection is used to predict the noise level. Instead, the authors of [11] augmented the model-based method with a SENet which predicts weights  $\mathbf{w} \in (0, 1)$  representing the rank of each patch, i.e., weights close to zero are associated with high rank whereas weights close to one are associated with low-rank patches. Specifically, SENet employs a one-dimensional, bias-free CNN architecture with  $T$  layers and intermediate features with  $L$  channels as shown below.

$$\mathbf{g}_0 = \mathbf{P} \quad (27)$$

$$\mathbf{g}_{t+1} = \text{ReLU}(\mathbf{W}_t * \mathbf{g}_t) \quad 0 \leq t < T \quad (28)$$

$$\text{SENet}(\mathbf{P}) = s(\mathbf{W}_T * \mathbf{g}_T) \quad (29)$$

The last activation,  $s(x) = \frac{1}{1+e^{-x}}$ , is a sigmoid which maps features into the desired range for the weights  $\mathbf{w}$ . With  $\mathbf{P}$  and  $\mathbf{w}$ , a weighted covariance matrix,  $\mathbf{P} \text{diag}(\mathbf{w}) \mathbf{P}^\top$ , is formed and its minimum singular value,  $\sigma_{\min}$ , is used to approximate noise variance:

$$\hat{\sigma}^2 = \frac{\sigma_{\min}}{\mathbf{1}^\top \mathbf{w}} \implies \hat{\Sigma} = \hat{\sigma} \mathbf{1} \quad (30)$$

**Noise homogeneity:** By construction of LINN, homogeneity is guaranteed in the case of AWGN if NENet can accurately track the standard deviation of the noise. That is:

$$\text{LINN}(\sigma \mathbf{n}, \Sigma) = \sigma \text{LINN}(\mathbf{n}, \mathbf{1}), \quad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \Sigma = \sigma \mathbf{1} > 0 \quad (31)$$

The above property, which shall be referred to as noise homogeneity, is a relaxation of strict homogeneity which was proposed in [9] and is preferred as it allows for strong generalisation ability while also enabling higher expressivity [11]. Noise homogeneity naturally promotes algorithms that can achieve the same amount of noise reduction for all noise levels:

$$\text{Var}\{\text{RWNN-F}(\sigma \mathbf{n})\} = \sigma^2 \text{Var}\{\text{RWNN-F}(\mathbf{n})\}, \quad \forall \sigma > 0 \quad (32)$$

Therefore, it is desirable in the context of universal denoising.

## Fusion Network

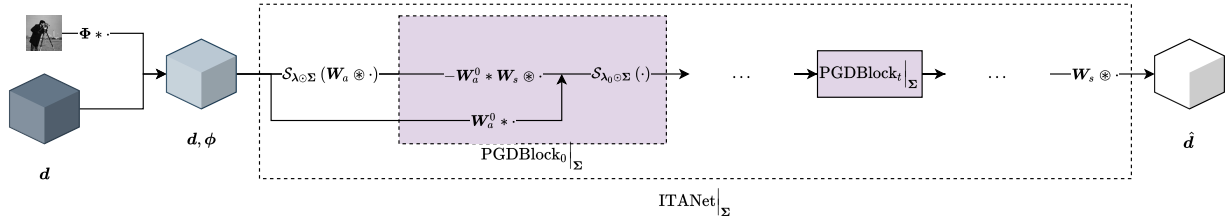


Figure 9: FusionNet architecture.

FusionNet, as shown in Figure 9, is at the core of RWNN-F as it is the only non-invertible module which can perform the denoising operation on the details. Since RWNN is encouraged to move only noise to the details, their denoising can be more challenging compared to natural images as a lot of the key structure is missing. On the other hand, predicting the clean details is easier compared to predicting clean natural images as the former follow a much simpler distribution (sparse) compared to the latter. To address the issue of a lack of structure, FusionNet extracts a prior signal,  $\phi \in \mathbb{R}^{N \times C_\phi \times H \times W}$ , from the clean coarse part,  $c \in \mathbb{R}^{N \times C \times H \times W}$ , and concatenates it with the noisy details,  $d$ , in order to perform the denoising operation through the use of an ITA based network (ITANet). In doing so, the two input signals are fused in a highly non-linear manner and the part that corresponds to the originally noisy details is used as an estimate of the clean details.

$$\text{FusionNet}(c, d, \Sigma) = \text{ITANet}([d, \Phi * c], \Sigma) \quad (33)$$

### Iterative Thresholding Algorithm

ITANet implements denoising according to the CSCN framework. As with the  $P$  and  $U$  networks, the non-rectangular  $d$  is zero-padded and tensorised before processing and cropped to the original dimensions at the end. The objective is to find a code,  $g$ , which provides a sparse representation of the joint signal  $[d, \phi]$ :

$$\min_g \|[d, \phi] - \mathbf{W}_s * g\|_2^2 + \sum_{i=1}^L \lambda_i \|g_i\|_1 \quad (34)$$

where  $\mathbf{W}_s : \mathbb{R}^{N \times L \times H \times W} \rightarrow \mathbb{R}^{N \times (3C + C_\phi) \times H \times W}$  is the synthesis operator and  $\lambda_i \geq 0$ . Since  $\phi$  is expected to be a good estimate for the desired clean details, it can effectively guide the sparse coding procedure to faster convergence. The above problem is solved by applying a Proximal Gradient Descent (PGD) technique which is inspired by quasi-Newton optimisation methods:

$$g_0 = \mathcal{S}_{\lambda_0 \odot \Sigma}(\mathbf{W}_a \circledast [d, \phi]) \quad (35)$$

$$g_{t+1} = \mathcal{S}_{\lambda_t \odot \Sigma}(g_t + \mathbf{W}_a^t * ([d, \phi] - \mathbf{W}_s \circledast g_t)) \quad 0 \leq t < T \quad (36)$$

$$[\text{ITANet}([d, \phi], \Sigma), \cdot] = \mathbf{W}_s \circledast g_T \quad (37)$$

i.e., instead of using a fixed step size and following the anti-gradient, ITANet will learn the optimal directions which are encoded in  $\mathbf{W}_a^t \in \mathbb{R}^{L \times (3C + C_\phi) \times p \times p}$ . Similar to the ResNet architecture, the thresholds<sup>1</sup> are augmented with  $\Sigma$  following the principles of [20].

**Controlling the fusion:** Even though  $d$  and  $\phi$  will generally be entangled during the above procedure,  $\Sigma \rightarrow \mathbf{0} \implies \hat{d} \approx d$  if  $\mathbf{W}_s$  and  $\mathbf{W}_a$  are orthogonal. That is, if  $\mathbf{W}_s \circ \mathbf{W}_a \approx \delta$  with  $\delta$  being the Kronecker delta, one can control the fusion and guarantee that RWNN-F performs an identity mapping in the noiseless case. For better handling of borders in  $\mathbf{W}_s \circ \mathbf{W}_a$ ,  $\mathbf{W}_s$  and  $\mathbf{W}_a$  use circular convolution  $\circledast$ . Note, however, that this method may introduce undesirable discontinuities as the borders of the images can, in general, have different noise levels. To minimise these, other kernels apply zero-padding to preserve the dimensionality.

<sup>1</sup> To enforce non-negativity, the softplus function,  $\text{Softplus}_\beta(x) = \frac{1}{\beta} \log(1 + e^{\beta x})$ , is used on unconstrained parameters,  $\beta = 20$ .

## 4 Implementation

### Overview

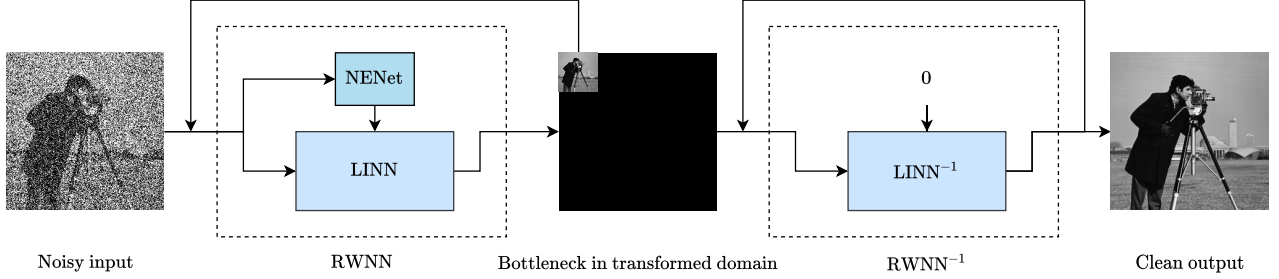


Figure 10: RWNN-DAE, RWNN as a DAE.

RWNN-F is trained in two steps to first stabilise RWNN and then fine-tune the networks for denoising.

**Denoising autoencoder:** For the first step, RWNN is trained separately from FusionNet as a Denoising AutoEncoder (DAE). The procedure is shown in Figure 10 and Algorithm 2; RWNN should provide coarse representations that are noise invariant, i.e., clean, and push most of the noise to the detail coefficients. Under this assumption, RWNN is able to compactly represent natural images within the coarse while the details vanish, i.e., are set to zero. For a noisy input  $\mathbf{y} = \mathbf{x} + \sigma\mathbf{n}$  with  $\sigma\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ , the DAE minimisation objective is:

$$\mathcal{L}_{\text{DAE}} = \mathbb{E} \left\{ \frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_c}{\text{Card}(\mathbf{x})} + \gamma_1 \|\hat{\sigma} - \sigma\|_c \right\} \quad (38)$$

where  $\hat{\mathbf{x}}$  is obtained after application of the DAE with scale  $K$ , i.e., discarding the first  $K$  detail bands,  $\hat{\sigma}$  is the noise estimate produced by NENet on the first scale and  $\gamma_1$  is a hyperparameter. Here,  $\text{Card}(\cdot)$  denotes the cardinality while  $\|x\|_c = \sqrt{x^2 + \epsilon^2}$  denotes the Charbonnier loss [43] with  $\epsilon = 10^{-3}$ . This loss is often a better choice compared to standard  $\ell_2$  [44, 45] as it inherits the robustness of  $\ell_1$  while being smooth near the origin. Robustness is important in the context of denoising since outliers may occur in the sampling of AWGN.

---

**Algorithm 2:** RWNN as a DAE.

---

**Input:** Noisy image  $\mathbf{y} = \mathbf{x} + \Sigma_{\mathbf{y}} \odot \mathbf{n}$

$\mathbf{c}_{\mathbf{y}}, \mathbf{d}_{\mathbf{y}}, \hat{\Sigma}_{\mathbf{y}} = \text{RWNN}(\mathbf{y})$  // Forward transform to separate the clean signal from noise

$\hat{\mathbf{c}}_{\mathbf{x}} = \text{RWNN-DAE}(\mathbf{c}_{\mathbf{y}})$  // Recurse to achieve further compression ( $\hat{\Sigma}_{\mathbf{c}_{\mathbf{y}}} \ll \hat{\Sigma}_{\mathbf{y}}$ ), if  $\hat{\Sigma}_{\mathbf{c}_{\mathbf{y}}} < \epsilon$  return  $\mathbf{c}_{\mathbf{y}}$

$\hat{\mathbf{d}}_{\mathbf{x}} = \mathbf{0}$  // Discard details (mostly noise)

$\hat{\mathbf{x}} = \text{RWNN}^{-1}(\hat{\mathbf{c}}_{\mathbf{x}}, \hat{\mathbf{d}}_{\mathbf{x}}, \mathbf{0})$  // Back-project

**Output:** Denoised image  $\hat{\mathbf{x}}$ .

---

**Integrating the fusion network:** After training RWNN as a DAE, RWNN-F is trained for denoising to minimise the following loss function:

$$\mathcal{L}_{\text{F}} = \mathbb{E} \left\{ \frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_c}{\text{Card}(\mathbf{x})} + \gamma_1 \|\hat{\sigma} - \sigma\|_c + \gamma_2 \frac{\|\mathbf{W}_s \otimes \mathbf{W}_a - \delta\|_2^2}{\text{Card}(\delta)} \right\} \quad (39)$$

where  $\hat{\mathbf{x}}$  is now obtained after application of RWNN-F with scale  $K$  as described in Figure 5 and Algorithm 1.  $\gamma_2$  controls the influence of the orthogonal loss,  $\frac{\|\mathbf{W}_s \otimes \mathbf{W}_a - \delta\|_2^2}{\text{Card}(\delta)}$ , which acts as a regulariser for FusionNet and is responsible for guaranteeing an identity mapping in the noiseless case. This regularisation is also responsible for boosting the generalisation ability of RWNN-F in the case of unseen and small noise levels, which occur most often in practice.

## Data, Network and Training Details

### Data Preparation

**Training data:** The 400 images of the Berkeley Segmentation Dataset (BSD400) [46] are used during training. Specifically, the dataset is split into 300 training samples and 100 validation samples. The former is further split into overlapping images of size  $64 \times 64$  for a total of 43200 patches and the batch size is set to  $N = 32$ . The data fed into the proposed algorithms is normalised to  $[-0.5, 0.5]$  and AWGN is added with a uniformly sampled standard deviation<sup>2</sup>, i.e.,  $\sigma \sim \mathcal{U}(0, 55)$  and the noise is  $\sigma \mathbf{n}$  with  $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ .

**Testing data:** The testing data consists of the 12 images from the Set12 dataset [47] and the 68 natural images from the BSD68 dataset [46].

### Optimisation Schedule

Both  $\mathcal{L}_{\text{DAE}}$  and  $\mathcal{L}_{\text{F}}$  are optimised using the same schedule. The training scale is  $K = 2$  to encourage RWNN to match its input distribution with the coarse output distribution. The regularisation hyperparameters are set to  $\gamma_1 = 1$  and  $\gamma_2 = 10$ . All parameters are learned through backpropagation using the Adam optimiser [48] with  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . The learning rate is initialised at  $lr = 10^{-3}$  and is divided by ten when the validation loss<sup>3</sup> does not improve for five epochs. Training terminates when  $lr < 10^{-6}$ .

**Transform network initialisation:** All convolutional kernels are initialised using Kaiming initialisation [49], which is a theoretically sound strategy for ReLU based networks. The adaptive thresholds / biases are initialised uniformly  $\sim \mathcal{U}(0, 1)$ , in agreement with SC which requires them to be non-negative.

**Fusion network initialisation:** The synthesis kernel,  $\mathbf{W}_s$ , is initialised by sampling from a normal distribution and all analysis kernels,  $\{\mathbf{W}_a\} \cup \{\mathbf{W}_a^t\}_{0 \leq t < T}$ , are initialised as the adjoint of  $\mathbf{W}_s$  to mimic traditional SC.

### Hyperparameter Tuning

RWNN-F is trained for  $C = 1$  channel images, i.e., grayscale. Unless otherwise stated, the following default hyperparameters are implied when referring to RWNN-DAE and RWNN-F.

**Transform network hyperparameters:** For a fair comparison with traditional wavelets, LINN uses the Lazy Wavelet Transform (LWT) as a split operator, i.e., separate pixels based on their parity. There are  $J = 4$  LNNets in LINN with each ResNet consisting of  $T = 7$  iterations. The convolutional kernels have support  $p = 3$  with  $L = 32$  features such that the receptive field of each  $P_j$  and  $U_j$  is  $33 \times 33$ . This receptive field is chosen so that each network is able to fully process the training data and leverage it at all scales. For NENet, the patch size is set to  $s = 8$  and SENet has  $T = 9$  convolutional layers with all kernels having support  $p = 5$  and  $L = 16$  features.

**Fusion network hyperparameters:** FusionNet extracts  $C_\phi = 3$  features from the coarse for fusion. ITANet has  $T = 7$  steps with  $L = 256$  latent features. The support for all kernels is  $p = 3$  and therefore the receptive field of ITANet is also  $33 \times 33$ .

With the above settings and an NVIDIA GeForce GTX 1080 Ti GPU, RWNN-DAE and RWNN-F can both be trained in under two days using the PyTorch framework. It is noted that NENet is one of the most computationally expensive modules in RWNN due to the application of SVD.

The training and testing codes are available at <https://github.com/andreasfloros/RWNN>.

---

<sup>2</sup> Raw  $\sigma$  values are quoted for simplicity.

<sup>3</sup> For more stable readings, validation is performed with a fixed  $\sigma = 25$ .

## 5 Testing

### Properties of the Networks

#### Scale Invariance

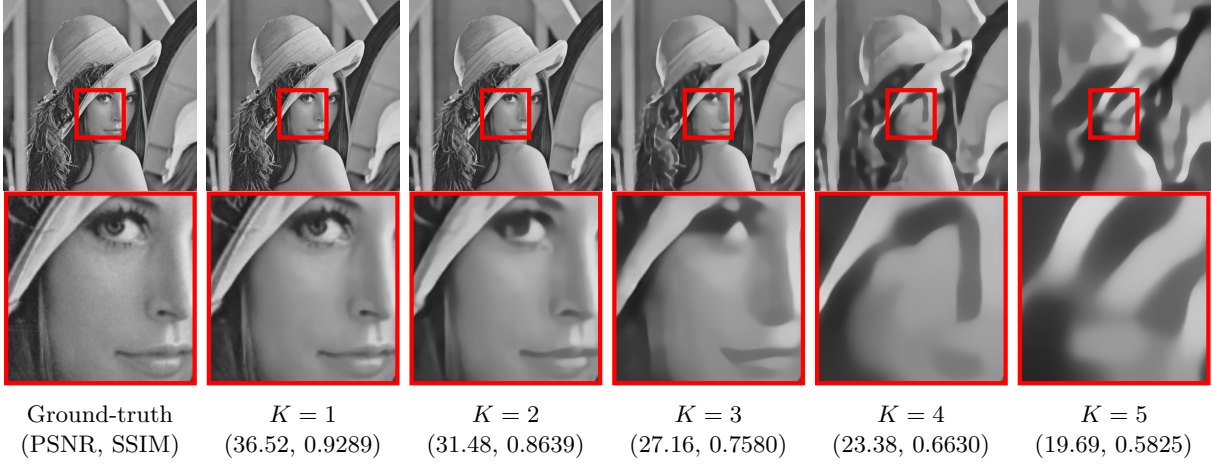
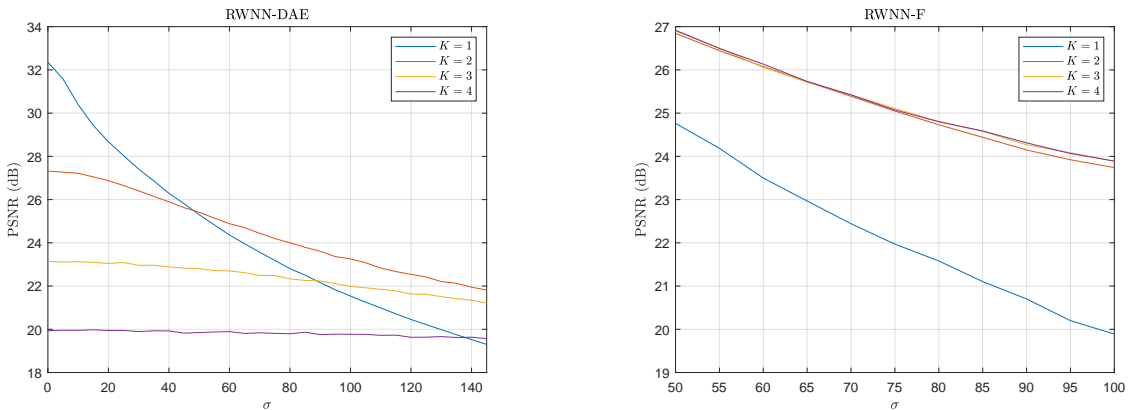


Figure 11: Progressive loading (right to left) using RWNN-DAE; the coarse parts are back-projected ( $\text{RWNN}^{-1}$ ) and all details up to scale  $K$  vanish.

RWNN can operate at different scales,  $K$ , despite being trained with a fixed  $K = 2$ ; the volume preserving property of LINN enables stable recursion and is crucial to this scale invariance.

**Progressive loading and rescaling:** JPEG-2000 introduced the feature of progressive loading [19, 18] which allows for previews during the transportation of data. The key idea is that one can incrementally build a high-resolution image by adding more detail subbands to a coarse part. RWNN-DAE mimics traditional progressive loading as shown in Figure 11 and provides a form of invertible rescaling similar to the algorithm proposed in [37]. Compared to traditional rescaling, RWNN-DAE generates fewer blocking artefacts at the cost of distorting object boundaries; this behaviour is evident for high values of  $K$ . Further analysis on this is given in Section 6.

**Incremental denoiser prior:** Figure 12 further proves the scale invariance property of RWNN; when the noise variance,  $\sigma^2$ , is large, RWNN-F can benefit from deeper decompositions, which, as shown in Figure 12 (a) and Figure 13, are expected to offer approximately noise invariant representations. Thus, starting from the deepest scale, a clean image prior may be incrementally formed.



(a) RWNN-DAE performance.

(b) RWNN-F performance.

Figure 12: RWNN performance on Set12 at different scales  $K$ .



## Noise Invariance and Adaptivity

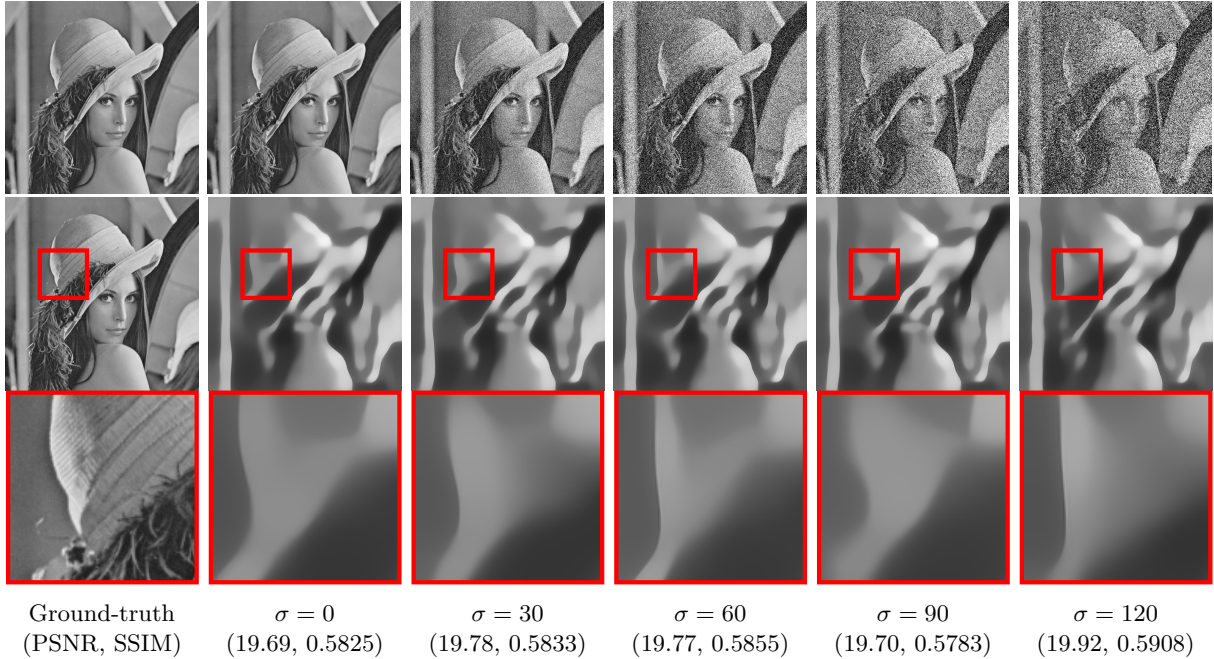


Figure 13: Noise invariance property of RWNN-DAE; the coarse parts are back-projected ( $\text{RWNN}^{-1}$ ) and all details up to scale  $K = 5$  vanish. The input image is degraded with  $\text{AWGN} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ .

RWNN forms noise invariant representations as shown in Figure 13; by comparing the back-projected images, it is evident that the coarse representations are robust. This is also confirmed by examining the PSNR and SSIM [50] scores reported; there is no correlation between recovered image quality and noise level. The noise invariance property is also present after fine-tuning RWNN-DAE to obtain RWNN-F; examples of this are shown in Figure 14. While there is some noise leakage, the majority of the noise is pushed into the detail coefficients to be denoised by FusionNet. Some example results of this denoising are shown in Figure 15.

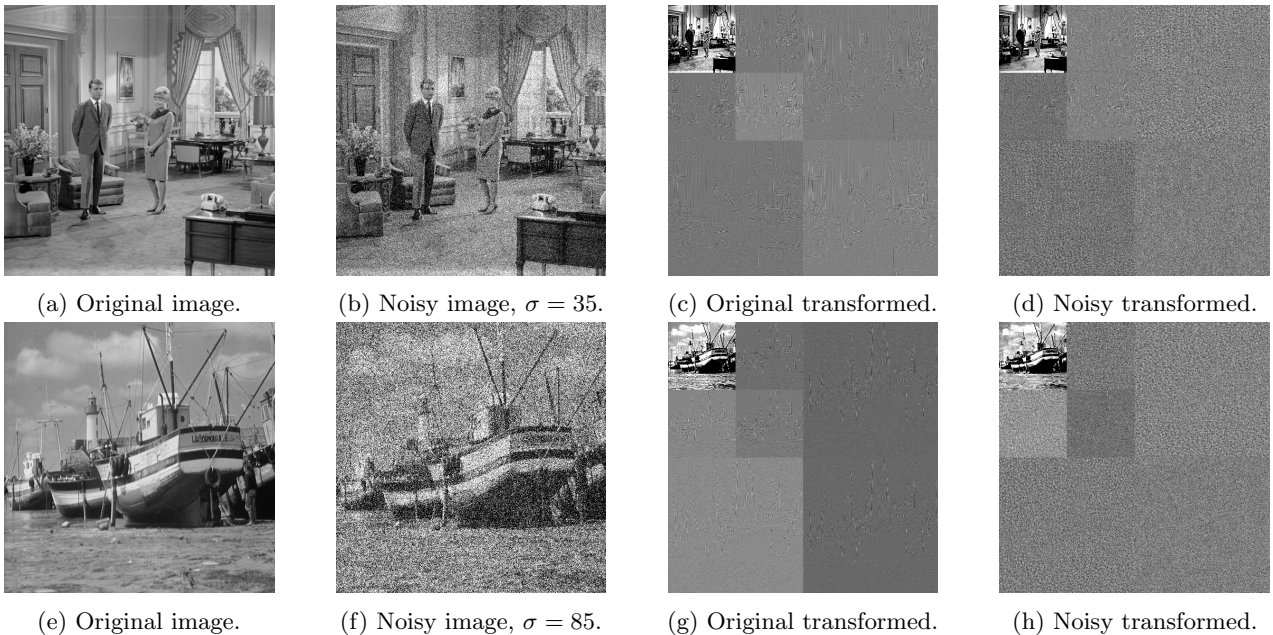


Figure 14: Example of two level decomposition using RWNN from RWNN-F.

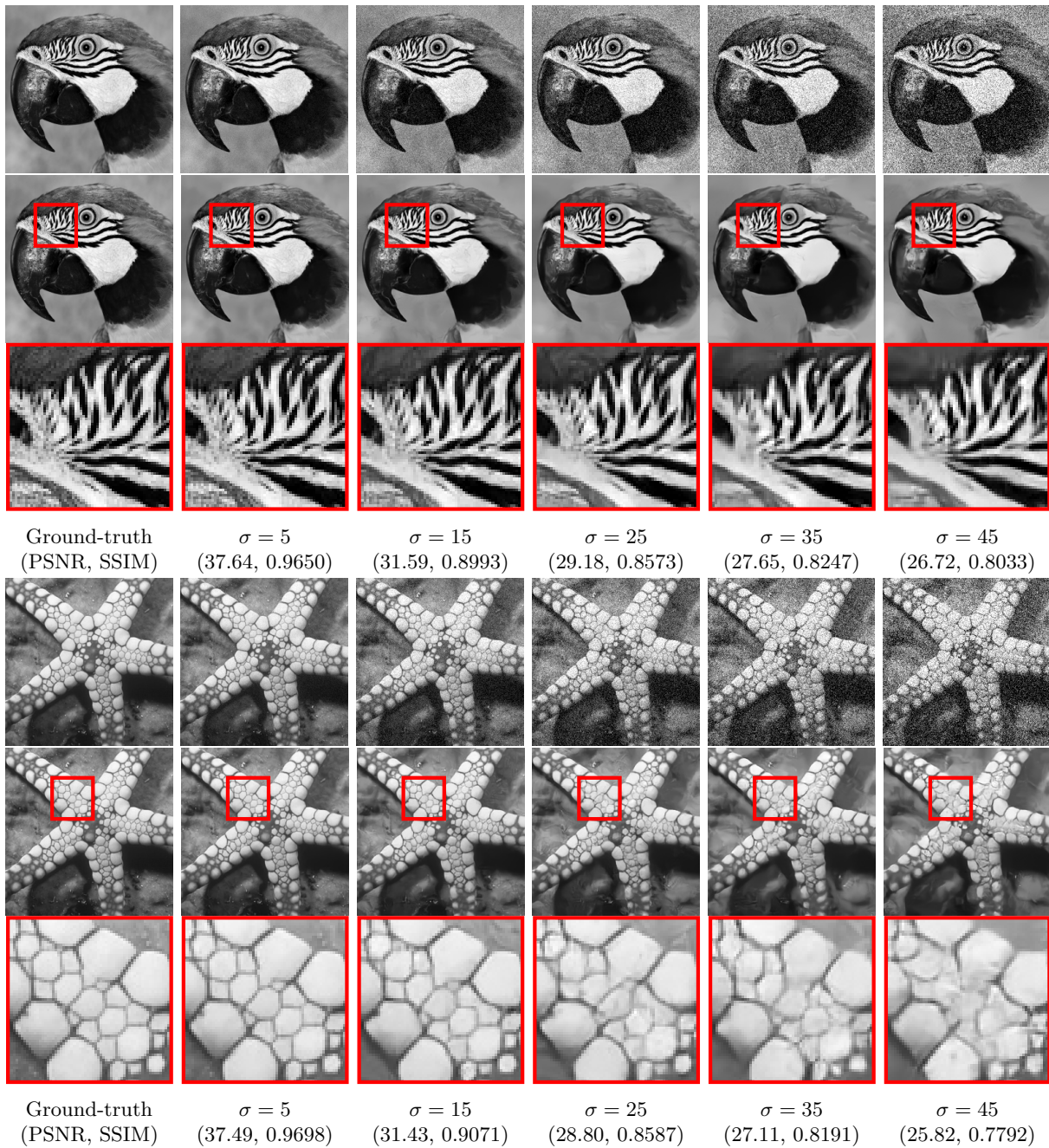


Figure 15: Denoising with RWNN-F,  $K = 2$ .



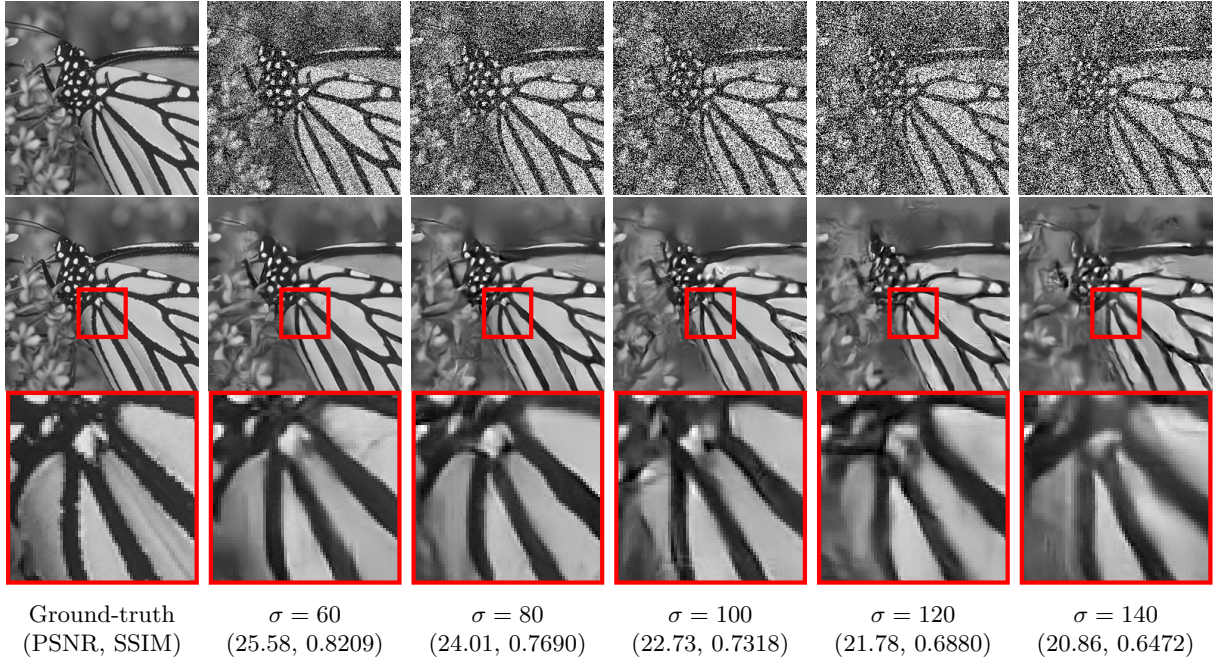


Figure 16: RWNN-F results on unseen AWGN,  $K = 4$ .

**Unseen noise levels:** Due to the noise homogeneity property, RWNN-F has strong generalisation ability and can handle unseen noise levels. Examples of this are shown in Figure 16. Note that during training  $\sigma \sim \mathcal{U}(0, 55)$ .

**Spatially variant noise:** RWNN is flexible as it can adapt for spatially variant noise removal when fed with non-uniform noise maps<sup>4</sup>. In Figure 17, the uniform noise estimate produced by NENet underperforms in areas of large noise while it provides excessive smoothing in areas of low noise; the non-uniform map allows RWNN-F to obtain significantly better results and addresses these issues.

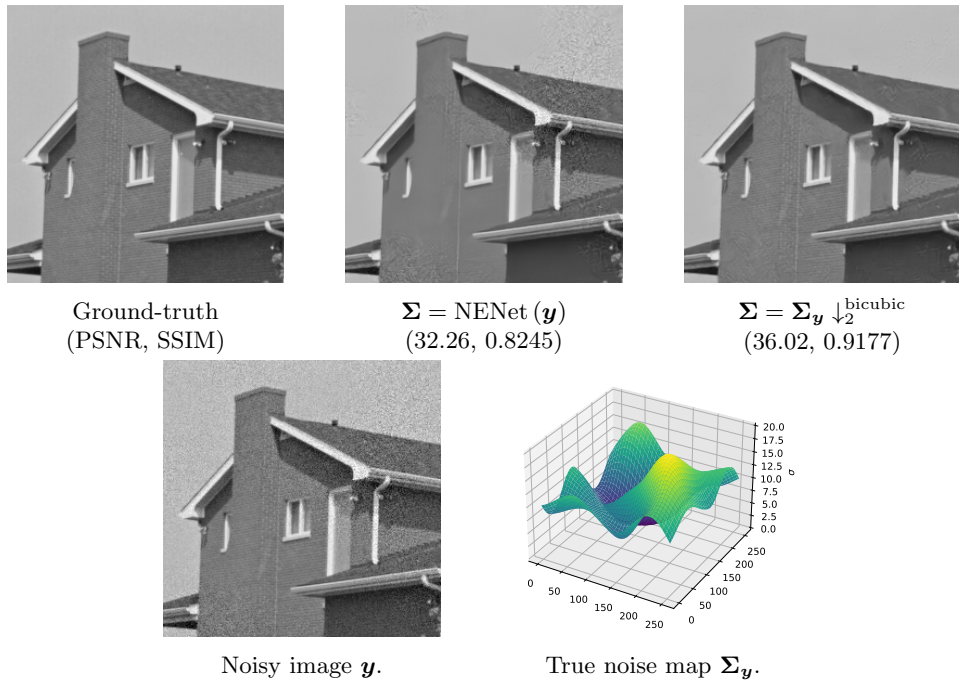


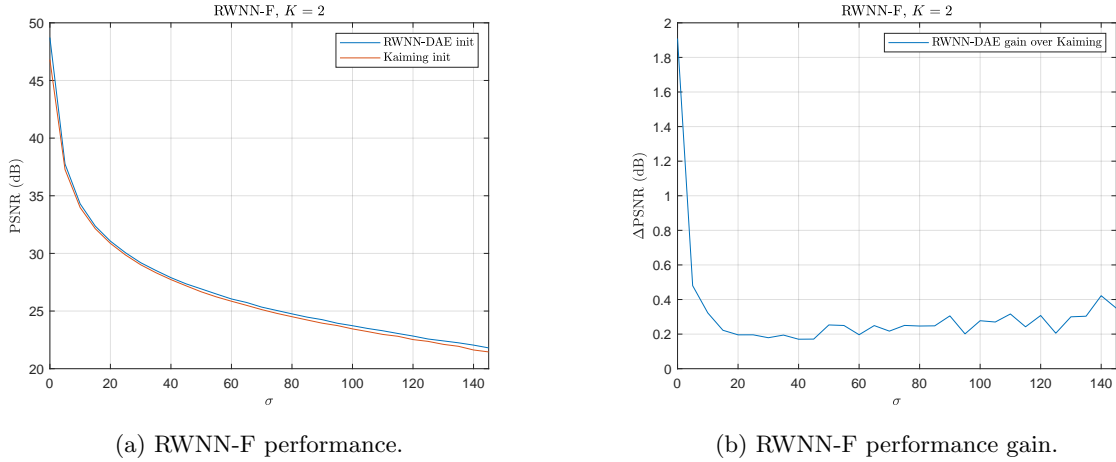
Figure 17: Spatially variant noise removal with RWNN-F,  $K = 1$ .

<sup>4</sup> Since LNetS operate on downsampled data (coarse and details), the maps are bicubically downsampled.

## Ablation Study

### Initialisation

To validate the training of RWNN-DAE separate from FusionNet, RWNN-F was also trained with standard Kaiming initialisation, i.e., skipping the first training step described in Section 4. The results, shown in Figure 18, reveal that initialisation with RWNN-DAE significantly improves performance. Specifically, a minimum gain of  $\sim 0.2$ dB is observed across all noise levels; the domain-aware initialisation of RWNN-F is key to its success.

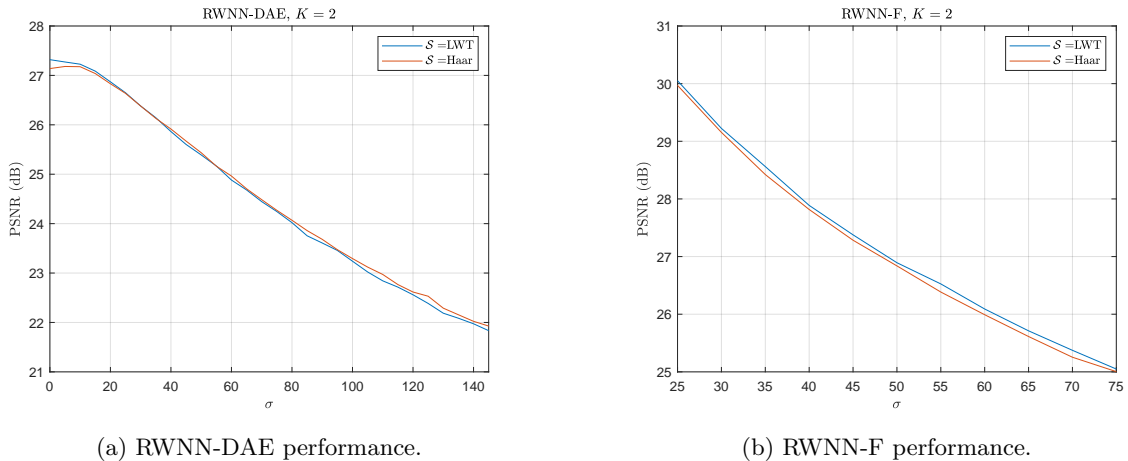


(a) RWNN-F performance.

(b) RWNN-F performance gain.

Figure 18: RWNN-F performance on Set12 for different initialisation strategies.

**Split operator:** The effect of split operators,  $\mathcal{S}$ , is shown in Figure 19. It appears that the LWT is better for RWNN-F while the Haar split is, in general, better in the DAE setting. Even though Haar or more complicated  $\mathcal{S}$  may be better choices, the standard setting for RWNN uses LWT as the goal is to explore the space of invertible transforms without handcrafted priors and biases; in principle, RWNN should be able to learn any volume preserving  $\mathcal{S}$  by making use of LNet.



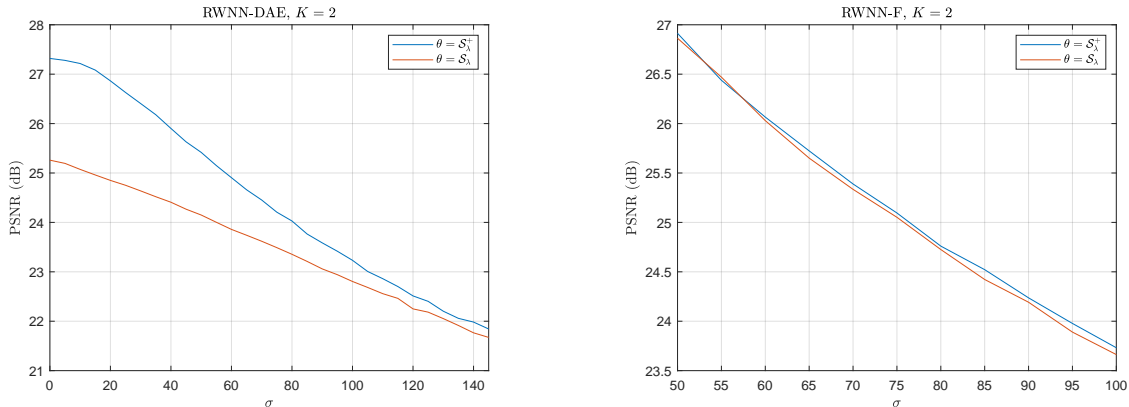
(a) RWNN-DAE performance.

(b) RWNN-F performance.

Figure 19: RWNN performance on Set12 for different split operators  $\mathcal{S}$ .

## Tweaking the Priors

**Activation function:** The choice of non-linearity proves to be crucial in RWNN-DAE, shown in Figure 20; the traditional soft thresholding non-linearity,  $\mathcal{S}_\lambda$ , consistently underperforms and does not produce robust coarse parts (no plateau for small  $\sigma$ ). The non-negative  $\mathcal{S}_\lambda^+$  is also preferred in the case of RWNN-F, though the differences between the non-linearities are not as significant in this case.

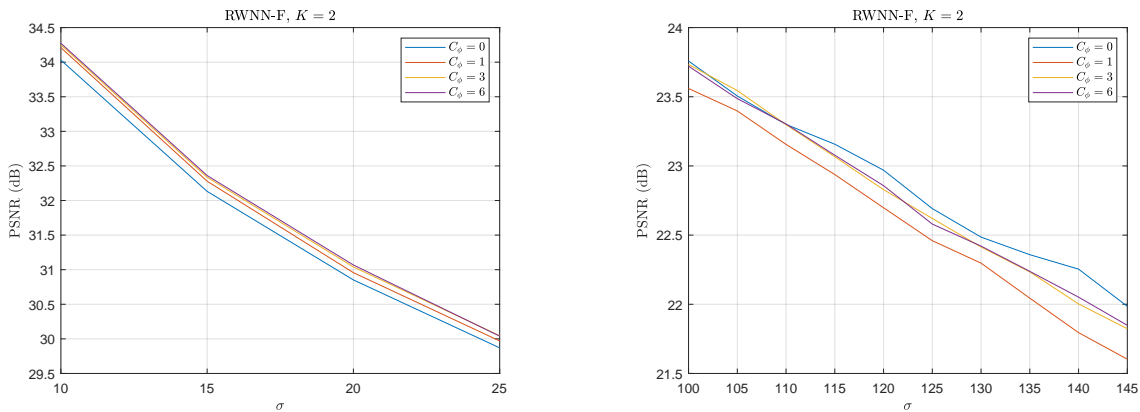


(a) RWNN-DAE performance.

(b) RWNN-F performance.

Figure 20: RWNN performance on Set12 for different activation functions  $\theta$ .

**Number of prior channels:** Figure 21 (a) shows the effect of the number of prior channels  $C_\phi$ . It is clear that the prior signal,  $\phi$ , helps improve the performance of RWNN-F. However, diminishing returns are observed when  $C_\phi$  is scaled up. To achieve a good complexity and performance trade-off,  $C_\phi = 3$  is chosen as the default.



(a)  $\sigma \in [10, 25]$ .

(b)  $\sigma \in [100, 145]$ .

Figure 21: RWNN-F performance on Set12 for different number of prior channels  $C_\phi$ .

Although the prior can boost performance, it should be noted that this only holds for appropriate combinations of  $K$  and  $\sigma$ . By examining Figure 21 (b), it is evident that for large  $\sigma$  the coarse parts at  $K = 2$  become unreliable and therefore  $\phi$  misleads ITANet; this results in  $C_\phi = 0$  being optimal. Interestingly, the cases of  $C_\phi = 3$  and  $C_\phi = 6$  have enough capacity to handle such noisy environments better than  $C_\phi = 1$ .

In practice, one can make the most out of RWNN-F and guarantee clean coarse parts by choosing a  $K$  that is either large or adaptive. For example,  $K$  should be set to the minimum scale such that NENet produces an estimate below a certain threshold.

## Scaling the Transform

**Width and depth:** The importance of the width ( $L$  features in ResNets) and the depth ( $J$  LNet) is compared in Figure 22; deeper networks are able to improve performance more reliably compared to shallow and wider networks. Moreover, it is noted that the model size scales linearly with  $J$  but quadratically with  $L$  and hence the deeper architectures are also more parameter efficient.

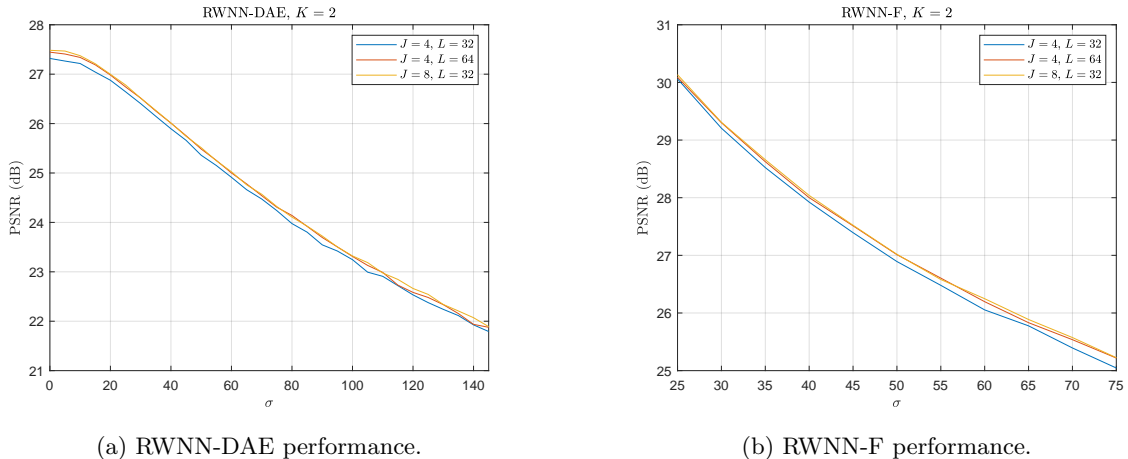


Figure 22: RWNN performance on Set12 for different activation functions  $\theta$ .

**Redundant transforms:** As an exploration, the RWNN setup is also considered with an undecimated split operator (URWNN). That is, instead of partitioning the input based on sample parity, the undecimated split simply replicates it; this results in a transform with a redundancy factor  $r = 4$ . From the results of Figure 23, it is clear that even a small  $r$  can significantly improve denoising performance without increasing the parameter count. Interestingly, despite having approximately half the model size, URWNN-DAE works better than URWNN-F as the reconstruction process in the former involves much fewer transformed coefficients and is therefore more robust.

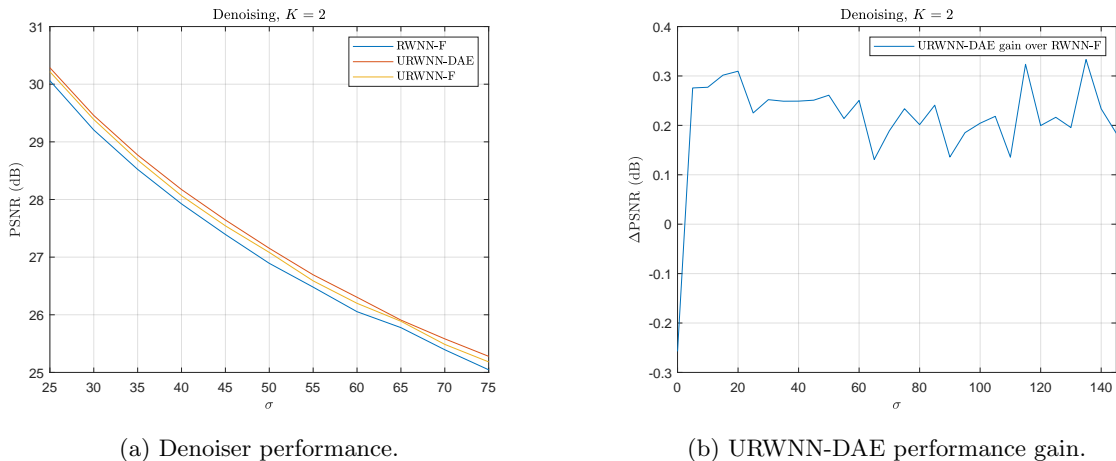


Figure 23: Performance of redundant and non-redundant denoisers on Set12.

While redundant transforms are certainly better choices in the context of denoising, they can be harder to interpret and adapt for other use cases. For example, the progressive loading task becomes trivial as there are no space constraints. Further, even though URWNN-DAE is more parameter efficient and more performant compared to RWNN-F, it is noted that this comes at the cost of a higher memory complexity which becomes increasingly unmanageable with higher scales  $K$  (if one wishes to process all coefficients of URWNN).

## 6 Results

### Progressive Loading and Energy Compaction

Table 1: Average PSNR (dB) and SSIM of different progressive loading methods evaluated on Set12. The best results are in red and the second best results are in blue.

Methods	$K = 1, 25\%$	$K = 2, 6.25\%$	$K = 3, 1.5625\%$	$K = 4, 0.3906\%$	$K = 5, 0.0976\%$
LGT 5/3 [51]	29.01, 0.8995	24.61, 0.7508	21.51, 0.5904	19.10, 0.4872	17.28, <b>0.4505</b>
CDF 9/7 [52]	<b>29.44, 0.9045</b>	<b>24.87, 0.7538</b>	<b>21.71, 0.5906</b>	<b>19.26, 0.4873</b>	<b>17.42, 0.4501</b>
RWNN-DAE	<b>32.35, 0.9183</b>	<b>27.31, 0.8120</b>	<b>23.13, 0.6620</b>	<b>19.93, 0.5322</b>	<b>17.35, 0.4578</b>

In Table 1 and Figure 24, RWNN-DAE is compared with traditional wavelet transforms for the task of progressive loading as described in Section 5. The chosen transforms are the LeGall-Tabatabai (LGT) 5/3 [51] and the Cohen–Daubechies–Feauveau (CDF) 9/7 [52] wavelets which are used in the JPEG-2000 standard.

Since CDF 9/7 has more vanishing moments than LGT 5/3, one expects it to perform better. Indeed, the above results confirm this prediction for PSNR (objective) scores. However, it is interesting to observe that, despite having fewer vanishing moments, LGT 5/3 manages to achieve similar SSIM (perceptual) scores to CDF 9/7 for  $K > 2$ . RWNN-DAE manages to significantly outperform both wavelets; for small  $K$ , the proposed achieves superior performance while also showing highly competitive PSNR and maintaining the best SSIM for large  $K$ .

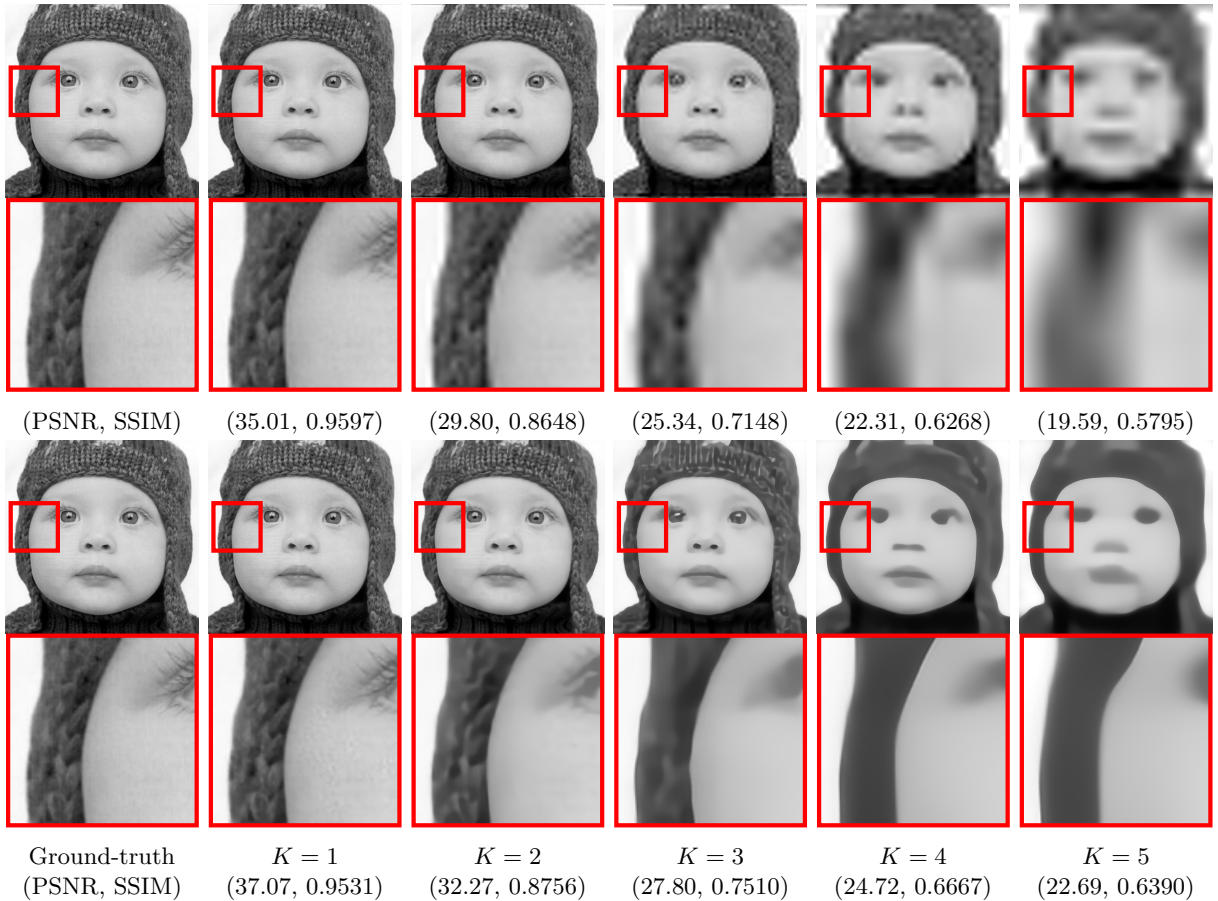


Figure 24: Progressive loading with CDF 9/7 (top) and RWNN-DAE (bottom).



## Image Denoising

Table 2: Average PSNR (dB) of different blind methods in the denoising task evaluated on the BSD68 and Set12 datasets. The best results are in red and the second best results are in blue.

Dataset	Methods	Model Size	$\sigma = 5$	$\sigma = 25$	$\sigma = 45$	$\sigma = 65$	$\sigma = 85$	$\sigma = 105$	$\sigma = 125$	$\sigma = 145$
BSD68	DnCNN-B [47]	$660 \cdot 10^3$	37.75	29.15	26.62	23.00	16.07	13.19	11.68	10.79
	BUIFD [7]	$1186 \cdot 10^3$	37.41	28.76	25.61	23.07	18.81	15.98	14.45	13.52
	BF-CNN [9]	$660 \cdot 10^3$	37.73	29.11	26.58	25.12	24.10	23.33	22.70	22.18
	WINNet [11]	$173 \cdot 10^3$	37.82	29.13	26.66	25.23	24.23	23.46	22.81	22.23
	RWNN-F	$296 \cdot 10^3$	37.41	28.86	26.42	25.03	24.07	23.36	22.76	22.23
Set12	DnCNN-B [47]	$660 \cdot 10^3$	37.88	30.38	27.68	23.52	15.95	13.18	11.78	10.92
	BUIFD [7]	$1186 \cdot 10^3$	37.34	30.18	27.01	24.27	19.41	16.28	14.66	13.73
	BF-CNN [9]	$660 \cdot 10^3$	37.81	30.33	27.58	25.83	24.54	23.55	22.74	22.07
	WINNet [11]	$173 \cdot 10^3$	38.22	30.33	27.72	26.03	24.77	23.76	22.94	22.24
	RWNN-F	$296 \cdot 10^3$	37.78	30.03	27.41	25.73	24.58	23.65	22.89	22.26

In Table 2, RWNN-F is compared with SOTA denoisers. All algorithms are trained on BSD400 with  $\sigma \in [0, 55]$  as in [11]. For the scale,  $K$ , RWNN-F employs the adaptive strategy described at the end of Section 5.

**Performance in the training range:** In general, for noise values in the training range, RWNN-F underperforms and the other algorithms appear to be more effective with DnCNN-B [47] and WINNet [11] being the best performing models. Nevertheless, RWNN-F rivals BUIFD [7] and is able to achieve results comparable to SOTA.

**Generalisation outside of training:** As can be seen, the noise homogeneous BF-CNN [9], WINNet and RWNN-F have superior performance outside of the training range while the non-homogeneous DnCNN-B and BUIFD struggle to generalise. Specifically, the proposed method significantly outperforms DnCNN-B and BUIFD while also surpassing BF-CNN and WINNet in extremely noisy environments. These observations show that noise homogeneity and an interpretable architecture are key ingredients for universal denoisers.

**Model size:** WINNet and RWNN-F achieve the smallest parameter counts while also having high generalisation ability. However, despite RWNN-F having more parameters, WINNet proves to be a more powerful denoiser in general. Architecturally, these models are similar; both are based on the principles of wavelets and use invertible modules to perform denoising in a transformed subspace. While this is the case, it is noted that RWNN is severely constrained due to being non-redundant, volume preserving and recursive whereas WINNet employs highly redundant LINNs which are universal approximators [53] and operates on a fixed  $K = 1$ . RWNN-F is more parameter efficient if one considers a two-scale WINNet, which has  $\sim 347 \cdot 10^3$  parameters [11], and is more interpretable as it can generalise and improve when deeper decompositions are performed (Figure 12).

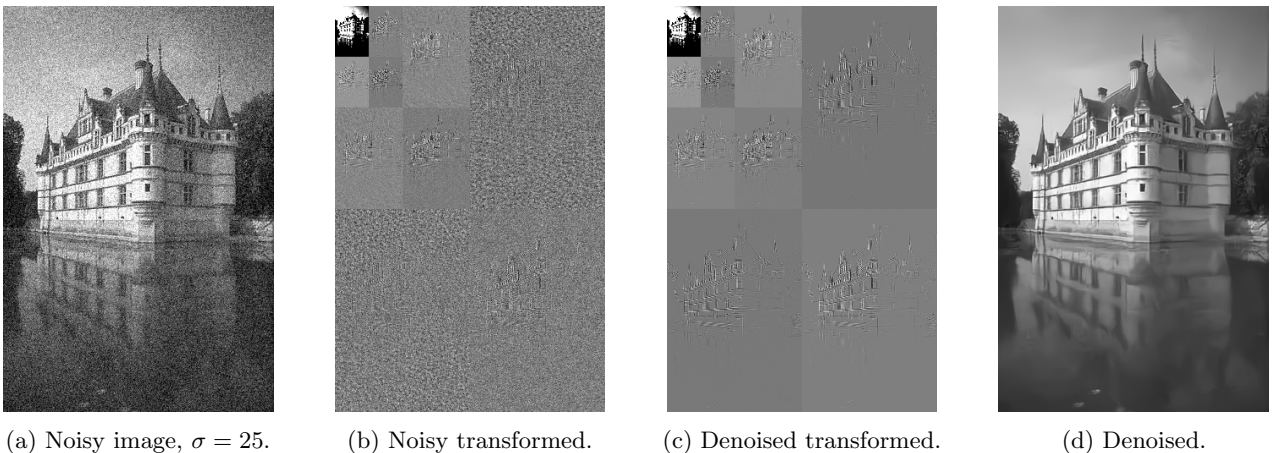


Figure 25: Example application of RWNN-F for denoising,  $K = 3$ .

## Towards General Problems

With the P3 framework [3, 1, 6], RWNN-F can be used to solve general IR problems. To demonstrate the effectiveness of this approach, RWNN-F adapted for the tasks of image deblurring and inpainting.

### Image Deblurring

The ground-truth image is convolved with a blur kernel,  $\mathbf{k}$ , and degraded with AWGN,  $\sigma\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ , to obtain the observation  $\mathbf{y}$ . The aim is to solve the following optimisation problem for some prior  $\lambda\Phi$ :

$$\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|\mathbf{k} \circledast \mathbf{x} - \mathbf{y}\|_2^2 + \lambda\Phi(\mathbf{x}), \quad \mathbf{y} = \mathbf{k} \circledast \mathbf{x} + \sigma\mathbf{n} \quad (40)$$

**Data sub-problems:** The MAP involves data sub-problems of the form:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{k} \circledast \mathbf{x} - \mathbf{y}\|_2^2 + \frac{\alpha_k}{2} \|\mathbf{z}_{k-1} - \mathbf{x}\|_2^2 \quad (41)$$

with  $\alpha_k = \mu_k\sigma^2$  and  $\mu_k$  being the step sizes of HQS. By rewriting the above in Fourier domain, it is possible to arrive at a closed form solution for the optimal  $\mathbf{x}$  [54]:

$$\mathbf{x}_k = \mathcal{F}^{-1} \left( \frac{\overline{\mathcal{F}(\mathbf{k})} \odot \mathcal{F}(\mathbf{y}) + \alpha_k \mathcal{F}(\mathbf{z}_{k-1})}{\overline{\mathcal{F}(\mathbf{k})} \odot \mathcal{F}(\mathbf{k}) + \alpha_k} \right) \quad (42)$$

where  $\mathcal{F}$ ,  $\mathcal{F}^{-1}$  denote forward and inverse Fourier transform operators,  $\overline{\mathcal{F}(\cdot)}$  denotes the complex conjugate and division is understood elementwise.

**Denosing sub-problems:** The denosing sub-problems take the form:

$$\min_{\mathbf{z}} \frac{1}{2\sigma_{k+1}^2} \|\mathbf{z} - \mathbf{x}_k\|_2^2 + \Phi(\mathbf{z}) \quad (43)$$

The optimal  $\mathbf{z}$  is obtained by application of RWNN-F and the noise level is assumed to be:

$$\sigma_{k+1} = \beta \text{NENet}(\mathbf{x}_k) = \sqrt{\frac{\lambda}{\mu_{k+1}}} \implies \alpha_k = \lambda \left( \frac{\sigma}{\sigma_k} \right)^2 \quad (44)$$

where  $\lambda$  is the regularisation parameter associated with RWNN-F's prior,  $\Phi$ , and  $\beta$  is a multiplier which increases the denosing strength to ensure fast convergence. The complete deblurring procedure is given in Algorithm 3.

---

**Algorithm 3:** RWNN-F for deblurring.

---

**Input:** Blurred image  $\mathbf{y} = \mathbf{k} \circledast \mathbf{x} + \sigma\mathbf{n}$ , kernel  $\mathbf{k}$

**Initialise:**  $\mathbf{z}_0 = \mathbf{y}$ ,  $\sigma_0 = \beta \text{NENet}(\mathbf{z}_0)$ ,  $\sigma_1 = 10\sigma_0$ ,  $k = 1$

**while**  $\sigma_k > \sigma_0$  **do**

$\mathbf{x}_k = \mathcal{F}^{-1} \left( \frac{\sigma_k^2 \overline{\mathcal{F}(\mathbf{k})} \odot \mathcal{F}(\mathbf{y}) + \lambda \sigma_0^2 \mathcal{F}(\mathbf{z}_{k-1})}{\sigma_k^2 \overline{\mathcal{F}(\mathbf{k})} \odot \mathcal{F}(\mathbf{k}) + \lambda \sigma_0^2} \right)$  // Solve data sub-problem  
 $\sigma_{k+1} = \beta \text{NENet}(\mathbf{x}_k)$  // Calculate new noise level  
 $\mathbf{z}_k = \text{RWNN-F}(\mathbf{x}_k, \sigma_{k+1})$  // Solve denosing sub-problem with the above  $\sigma_{k+1}$   
 $k = k + 1$









**end**

**Output:** Deblurred image  $\mathbf{z}_{k-1}$ .

---

**Tuning the algorithm:** Since the  $\alpha_k$  are determined automatically by NENet, the only hyperparameters for tuning are the prior parameter,  $\lambda$ , the multiplier,  $\beta$ , the number of iterations and the scale  $K$ . Following the settings in [1],  $\lambda = 0.23$  and  $\beta = 2$  as per [11] where WINNet follows a similar construction to RWNN-F while the number of iterations is adaptive. Scale  $K = 1$  is chosen since the noise levels are expected to be small in practice, but it can also be adaptive as discussed at the end of Section 5. Note that, in the latter case, one should also tune the noise levels for each decomposition level, which is undesirable.

Table 3: Average PSNR (dB) of different image deblurring methods evaluated on Set12 with kernels from [55] and  $\sigma = 2.55$ . The best results are in red and the second best results are in blue.

Methods	1 	2 	3 	4 	5 	6 	7 	8 
EPLL [56]	31.47	31.00	31.23	29.80	32.47	32.28	31.12	30.53
IRCNN [1]	32.26	31.65	30.87	31.76	31.82	31.99	31.10	31.07
WINNet [11]	32.37	32.04	32.03	31.52	32.47	33.17	32.02	31.87
RWNN-F	32.19	31.89	31.71	31.58	31.97	32.95	31.59	31.79

The performance of RWNN-F for deblurring is shown in Table 3; the proposed algorithm achieves highly competitive performance and comparable to SOTA. As expected, since WINNet [11] is more effective in low-noise environments, it performs better than RWNN-F on average.

An example summarising the deblurring operation with RWNN-F and P3 is given in Figure 26. It can be seen that the RWNN-F denoiser acts as a projection to the natural image manifold while the data solutions,  $\mathbf{x}_k$ , are with artefacts and noise. This interpretation is similar to the analysis followed in [5] and is perhaps the reason why denoisers are central to IR.

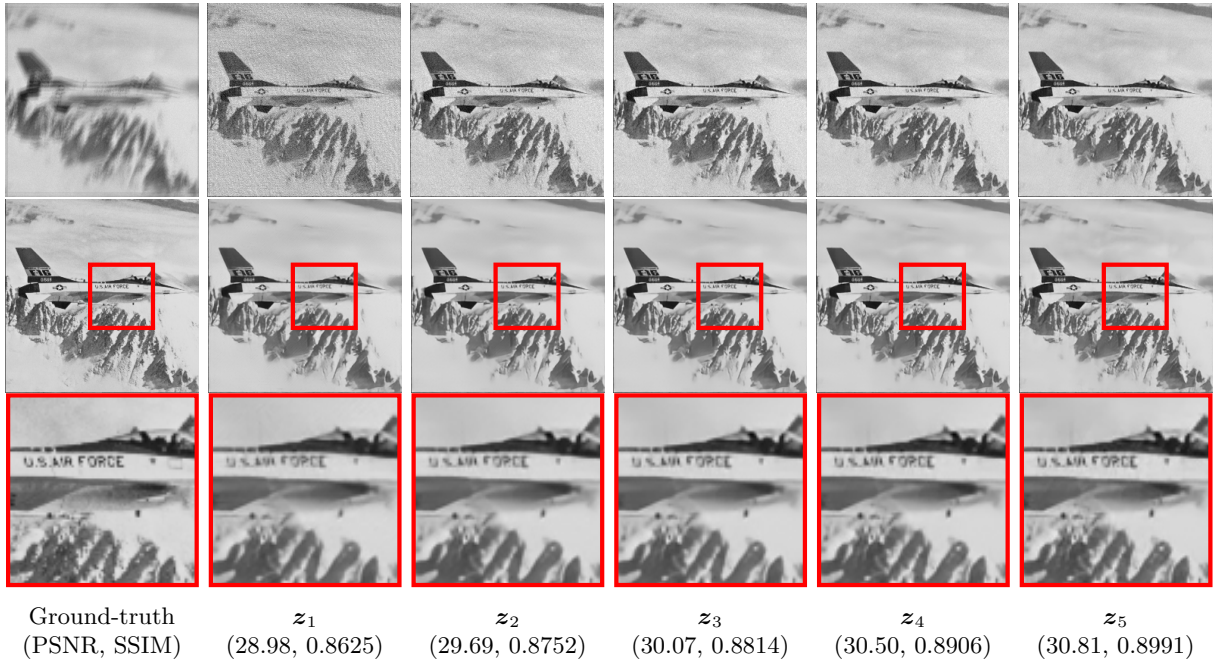


Figure 26: RWNN-F deblurring with first kernel of size  $19 \times 19$  from [55] and noise level  $\sigma = 2.55$ . The top row shows the data solutions,  $\mathbf{x}_k$ , and the bottom row shows the denoised images  $\mathbf{z}_k$ . The first column shows the degraded image and the ground-truth instead.



## Image Inpainting

For the inpainting task, the observation,  $\mathbf{y}$ , is missing pixels which need to be recovered. Specifically, the restoration process is described as the following optimisation:

$$\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|\mathbf{M} \odot \mathbf{x} - \mathbf{y}\|_2^2 + \lambda \Phi(\mathbf{x}), \quad \mathbf{y} = \mathbf{M} \odot (\mathbf{x} + \sigma \mathbf{n}) \quad (45)$$

where  $\mathbf{M}$  is a binary mask which represents the missing pixels with zero entries.

**Data sub-problems:** The MAP involves data sub-problems of the form:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{M} \odot \mathbf{x} - \mathbf{y}\|_2^2 + \frac{\alpha_k}{2} \|\mathbf{z}_{k-1} - \mathbf{x}\|_2^2 \quad (46)$$

Since  $\mathbf{M}$  is binary, the solution is simplified to:

$$\mathbf{x}_k = \frac{\mathbf{y} + \alpha_k \mathbf{z}_{k-1}}{\mathbf{M} + \alpha_k} \quad (47)$$

**Denoising sub-problems:** The denoising sub-problems have the same form as in the deblurring task and are also assumed to be solved by RWNN-F. However, unlike in the case of deblurring, Gaussian noise estimation via NENet is not an effective strategy for the inpainting task. This observation is also supported by [54] where the authors experimentally show that the effective noise tends to be sparse. Thus,  $\sigma_{k+1}$  must be manually tuned in this case. Following the intuition that  $\alpha_k = \lambda(\sigma/\sigma_k)^2$  should increase with the number of iterations, the noise levels are set to decay as  $\sigma_{k+1} = \sigma_1 d^k$  for some  $d \in (0, 1)$ . The procedure is summarised in Algorithm 4.

---

### Algorithm 4: RWNN-F for inpainting.

---

**Input:** Degraded image  $\mathbf{y} = \mathbf{M} \odot (\mathbf{x} + \sigma \mathbf{n})$ , binary mask  $\mathbf{M}$ , noise level  $\sigma$

**Initialise:**  $\mathbf{z}_0 = \mathbf{y}$ ,  $\sigma_1 = 255$ ,  $k = 1$

**while**  $\sigma_k > \max(\sigma, \epsilon)$  **do**

$\mathbf{x}_k = \frac{\sigma_k^2 \mathbf{y} + \lambda \sigma^2 \mathbf{z}_{k-1}}{\sigma_k^2 \mathbf{M} + \lambda \sigma^2}$  // Solve data sub-problem

$\sigma_{k+1} = \sigma_k d$  // Calculate new noise level

$\mathbf{z}_k = \text{RWNN-F}(\mathbf{x}_k, \sigma_{k+1})$  // Solve denoising sub-problem with the above  $\sigma_{k+1}$

$k = k + 1$

**end**

**Output:** Inpainted image  $\mathbf{z}_{k-1}$ .

---

**Tuning the algorithm:** The regularisation parameter is set to  $\lambda = 0.23$  and the scale is set to  $K = 1$  which limits tweaking to just the first level. The number of iterations is adaptive and the decay rate is set to  $d = 0.9$ .

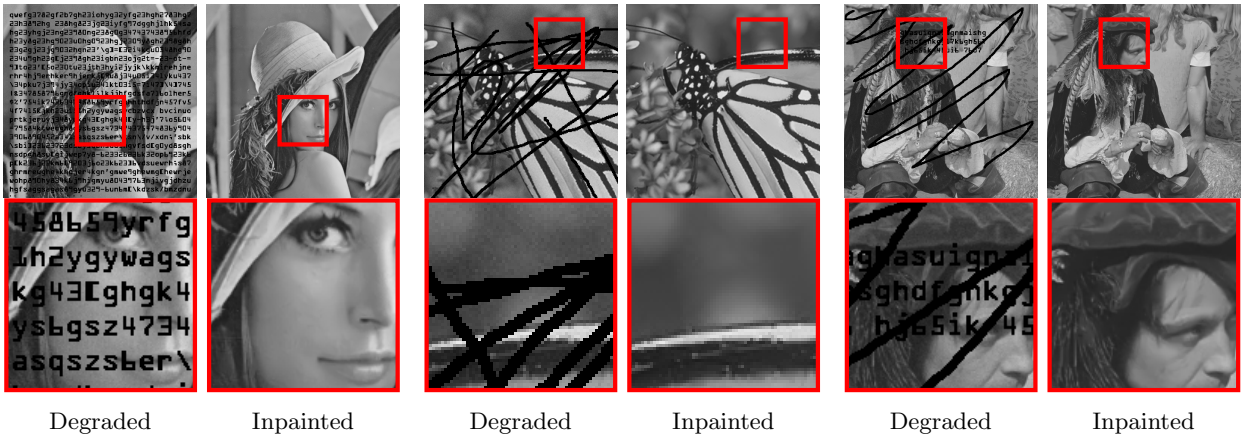


Figure 27: Inpainting with RWNN-F,  $\sigma = 2.55$ .

## 7 Evaluation and Conclusions

The paper concludes with a discussion on the limitations of the proposed RWNN and suggestions for potentially fruitful research directions to be explored. Finally, the merits of the RWNN construction are summarised.

### Limitations

**Expressivity and flexibility trade-off:** While RWNN provides a flexible algorithm, it is straightforward to see that it is not a universal approximator as its ability to recurse is enabled by volume preservation. Moreover, the approximation power is reduced further since RWNN is non-redundant. Experiments show that this lack of expressivity does in fact hinder performance and so it is not clear if the proposed construction can be scaled to compete with black box specialised algorithms. One wonders if it is possible relax some of RWNN’s constraints without compromising flexibility and generalisation ability.

**Mimicking wavelets:** RWNN follows the wavelet construction faithfully. However, as it is highly non-linear, regularising it to mimic wavelets provides a challenge. The particular use case that is relevant here is that of lossy compression; while it has been shown that RWNN-DAE significantly outperforms traditional wavelets in progressive loading, compression is a bit more involved as efficient non-linear approximations need to be devised along with efficient encoding strategies [18, 19, 57]. This is a delicate matter which was not pursued here. However, experiments showed that even preserving the properties of RWNN-DAE after fine-tuning is non-trivial; there exists a trade-off in terms of energy compaction and denoising ability.

### Future Work

**Scaling the networks:** RWNN showcases an interesting phenomenon where an image is seen at multiple scales and noise levels during training and it is intuitively understood that such a construction allows for more efficient learning. In this work, this property is only exploited up to scale  $K = 2$  as higher  $K$  would require more resources. It would be interesting to see how the RWNN performance scales with a much larger dataset, non-uniform noise estimation and a higher parameter count. Additionally, as it is well understood that conventional metrics do not capture image quality the same way humans perceive it [58], it would also be interesting to explore more exotic objective functions in the form of adversarial networks.

**Fusion algorithms:** This work primarily considered IR tasks. However, in principle, RWNN can learn to decompose images using different atoms (not necessarily based on noise level). It would be interesting to explore the properties of RWNN under generative schemes such as image fusion and style fusion where one obtains one fused image from multiple input images. The proposed SC-inspired FusionNet could also be adapted for this.

### Conclusion

The proposed construction has proven its flexibility in a variety of imaging tasks despite being non-redundant, recursive and volume preserving. Because of this domain-aware design, the inner workings of RWNN-F are understood and it is easy to draw parallels with traditional wavelets; it is this knowledge and inspiration that allows for an efficient initialisation strategy, improved generalisation ability and high interpretability. Compared to existing IR algorithms, RWNN is parameter efficient, achieves highly competitive performances and showcases the power of model-inspired, interpretable algorithms in the deep learning era.

## Bibliography

- [1] K. Zhang, W. Zuo, S. Gu, and L. Zhang, “Learning Deep CNN Denoiser Prior for Image Restoration,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3929–3938.
- [2] W. H. Richardson, “Bayesian-Based Iterative Method of Image Restoration,” *J. Opt. Soc. Am.*, vol. 62, no. 1, pp. 55–59, 1972.
- [3] S. V. Venkatakrisnan, C. A. Bouman, and B. Wohlberg, “Plug-and-Play priors for model based reconstruction,” in *2013 IEEE Global Conference on Signal and Information Processing*, 2013, pp. 945–948.
- [4] C. A. Metzler, A. Maleki, and R. G. Baraniuk, “BM3D-AMP: A new image recovery algorithm based on BM3D denoising,” in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 3116–3120.
- [5] Y. Romano, M. Elad, and P. Milanfar, “The Little Engine that Could: Regularization by Denoising (RED),” *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017.
- [6] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers, “Learning Proximal Operators: Using Denoising Networks for Regularizing Inverse Imaging Problems,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1781–1790.
- [7] M. El Helou and S. Süsstrunk, “Blind Universal Bayesian Image Denoising With Gaussian Noise Level Learning,” *IEEE Transactions on Image Processing*, vol. 29, pp. 4885–4897, 2020.
- [8] K. Zhang, W. Zuo, and L. Zhang, “FFDNet: Toward a Fast and Flexible Solution for CNN based Image Denoising,” *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4608–4622, 2018.
- [9] S. Mohan, Z. Kadkhodaie, E. P. Simoncelli, and C. Fernandez-Granda, “Robust and interpretable blind image denoising via bias-free convolutional neural networks,” 2020.
- [10] N. Janjušević, A. Khalilian-Gourtani, and Y. Wang, “CDLNet: Robust and Interpretable Denoising Through Deep Convolutional Dictionary Learning,” 2021. [Online]. Available: <https://arxiv.org/abs/2103.04779>
- [11] J.-J. Huang and P. L. Dragotti, “WINNet: Wavelet-inspired Invertible Network for Image Denoising,” 2021.
- [12] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,” *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 57, no. 11, pp. 1413–1457, 2004.
- [13] L. Zhang, Y. Hu, C. Yu, and J. Wang, “Iterative positive thresholding algorithm for non-negative sparse optimization,” *Optimization*, vol. 67, pp. 1–19, 05 2018.
- [14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” 2015.
- [15] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.
- [16] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training Recurrent Neural Networks,” 2013.
- [17] K. Gregor and Y. LeCun, “Learning Fast Approximations of Sparse Coding,” in *Proceedings of the 27th international conference on international conference on machine learning*, 2010, pp. 399–406.
- [18] M. Rabbani, “JPEG2000: Image Compression Fundamentals, Standards and Practice,” *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.
- [19] M. Charrier, “JPEG2000: A New Standard for Still Image Compression,” in *Multimedia Computing and Systems, International Conference on*, vol. 1. IEEE Computer Society, 1999, pp. 9131–9131.
- [20] D. L. Donoho and J. M. Johnstone, “Ideal Spatial Adaptation by Wavelet Shrinkage,” *biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

- [21] S. G. Chang, B. Yu, and M. Vetterli, “Adaptive Wavelet Thresholding for Image Denoising and Compression,” *IEEE transactions on image processing*, vol. 9, no. 9, pp. 1532–1546, 2000.
- [22] T. Blu and F. Luisier, “The SURE-LET Approach to Image Denoising,” *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2778–2786, 2007.
- [23] D. Donoho and M. Raimondo, “A fast wavelet algorithm for image deblurring,” *ANZIAM Journal*, vol. 46, pp. C29–C46, 2004.
- [24] N. Pustelnik, A. Benazza-Benhayia, Y. Zheng, and J.-C. Pesquet, “Wavelet-Based Image Deconvolution and Reconstruction,” *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–34, 1999.
- [25] B. Dong, H. Ji, J. Li, Z. Shen, and Y. Xu, “Wavelet Frame Based Blind Image Inpainting,” *Applied and Computational Harmonic Analysis*, vol. 32, no. 2, pp. 268–279, 2012.
- [26] L. He and Y. Wang, “Iterative Support Detection-Based Split Bregman Method for Wavelet Frame-Based Image Inpainting,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 5470–5485, 2014.
- [27] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic press, 2008.
- [28] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, “Normalizing Flows for Probabilistic Modeling and Inference,” 2019. [Online]. Available: <https://arxiv.org/abs/1912.02762>
- [29] I. Kobyzev, S. J. Prince, and M. A. Brubaker, “Normalizing Flows: An Introduction and Review of Current Methods,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, p. 3964–3979, Nov 2021. [Online]. Available: <http://dx.doi.org/10.1109/TPAMI.2020.2992934>
- [30] D. J. Rezende and S. Mohamed, “Variational Inference with Normalizing Flows,” 2016.
- [31] L. Dinh, D. Krueger, and Y. Bengio, “NICE: Non-linear Independent Components Estimation,” 2015.
- [32] D. P. Kingma and M. Welling, “An Introduction to Variational Autoencoders,” *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019. [Online]. Available: <https://doi.org/10.1561%2F22000000056>
- [33] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Networks,” 2014.
- [34] L. Dinh, J. Sohl-Dickstein, and S. Bengio, “Density estimation using Real NVP,” 2016. [Online]. Available: <https://arxiv.org/abs/1605.08803>
- [35] J. Behrmann, W. Grathwohl, R. T. Q. Chen, D. Duvenaud, and J.-H. Jacobsen, “Invertible Residual Networks,” 2018. [Online]. Available: <https://arxiv.org/abs/1811.00995>
- [36] J.-J. Huang and P. L. Dragotti, “LINN: Lifting Inspired Invertible Neural Network for Image Denoising,” in *The 29th European Signal Processing Conference, EUSIPCO 2021*, 2020.
- [37] M. Xiao, S. Zheng, C. Liu, Y. Wang, D. He, G. Ke, J. Bian, Z. Lin, and T.-Y. Liu, “Invertible Image Rescaling,” 2020. [Online]. Available: <https://arxiv.org/abs/2005.05650>
- [38] W. Sweldens, “The Lifting Scheme: A Construction of Second Generation Wavelets,” *SIAM journal on mathematical analysis*, vol. 29, no. 2, pp. 511–546, 1998.
- [39] I. Daubechies and W. Sweldens, “Factoring Wavelet Transforms into Lifting Steps,” *Journal of Fourier analysis and applications*, vol. 4, no. 3, pp. 247–269, 1998.
- [40] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.
- [41] S. G. Bahnemiri, M. Ponomarenko, and K. Egiazarian, “Learning-based Noise Component Map Estimation for Image Denoising,” 2021. [Online]. Available: <https://arxiv.org/abs/2109.11877>
- [42] X. Liu, M. Tanaka, and M. Okutomi, “Single-Image Noise Level Estimation for Blind Denoising,” *IEEE transactions on image processing*, vol. 22, no. 12, pp. 5226–5237, 2013.

- [43] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud, “Two deterministic half-quadratic regularization algorithms for computed imaging,” in *International Conference on Image Processing*, vol. 2. IEEE, 1994, pp. 168–172.
- [44] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, “Loss Functions for Neural Networks for Image Processing,” 2015. [Online]. Available: <https://arxiv.org/abs/1511.08861>
- [45] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
- [46] S. Roth and M. J. Black, “Fields of Experts,” *International Journal of Computer Vision*, vol. 82, no. 2, p. 205, 2009.
- [47] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising,” *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.
- [48] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” 2017.
- [49] K. He, X. Zhang, S. Ren, and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,” 2015.
- [50] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessment: From Error Measurement to Structural Similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [51] D. Le Gall and A. Tabatabai, “Sub-band Coding of Digital Images Using Symmetric Short Kernel Filters and Arithmetic Coding Techniques,” in *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*, 1988, pp. 761–764 vol.2.
- [52] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, “Image Coding Using Wavelet Transform,” *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 205–220, 1992.
- [53] F. Koehler, V. Mehta, and A. Risteski, “Representational aspects of depth and conditioning in normalizing flows,” 2020. [Online]. Available: <https://arxiv.org/abs/2010.01155>
- [54] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, “Plug-and-Play Image Restoration with Deep Denoiser Prior,” 2020. [Online]. Available: <https://arxiv.org/abs/2008.13751>
- [55] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, “Understanding and evaluating blind deconvolution algorithms,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1964–1971.
- [56] D. Zoran and Y. Weiss, “From Learning Models of Natural Image Patches to Whole Image Restoration,” in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 479–486.
- [57] J. Shapiro, “Embedded Image Coding Using Zerotrees of Wavelet Coefficients,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.
- [58] C. Ma, C.-Y. Yang, X. Yang, and M.-H. Yang, “Learning a No-Reference Quality Metric for Single-Image Super-Resolution,” 2016.