Image restoration with deep generative models

Jean Prost

université de BORDEAUX

Université Paris Cité

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR
DE L'UNIVERSITÉ DE BORDEAUX

ECOLE DOCTORALE MATHÉMATIQUES ET INFORMATIQUE

MATHÉMATIQUES APPLIQUÉES ET CALCUL SCIENTIFIQUE

Par **Jean PROST**

Restauration d'images avec des modèles génératifs profonds

Sous la direction de : **Nicolas PAPADAKIS**
Co-directeur : **Andrés ALMANSA**

Soutenue le 15 novembre 2023

Membres du jury :

M. Andrés ALMANSA	Directeur de recherche	Université Paris Cité	Codirecteur de thèse
Mme. Aurelie BUGEAU	Professeure des Universités	Université de Bordeaux	Présidente
M. Pierre CHAINAIS	Professeur des Universités	École Centrale de Lille	Rapporteur
M. Alasdair NEWSON	Maître de conférences	Sorbonne Université	Examinateur
M. Thomas OBERLIN	Professeur	ISAE SUPAERO Toulouse	Rapporteur
M. Nicolas PAPADAKIS	Directeur de recherche	Université de Bordeaux	Directeur de thèse
M. Marcelo PEREYRA	Professor	Heriot-Watt University	Examinateur
Mme. Pauline TAN	Maître de conférences	Sorbonne Université	Examinatrice

Membre invité :

M. Antoine HOUDARD	Cadre scientifique	Ubisoft	Invité

Restauration d'images avec des modèles génératifs profonds

Résumé : Les problèmes de restauration d'images, comme le défloutage ou la super-résolution sont des problèmes inverses, où l'on cherche à retrouver une image propre à partir d'une observation dégradée. Pour déterminer comment retrouver l'information manquante à partir de l'image dégradée, il est nécessaire de définir un modèle *a priori* des propriétés attendues de la solution. Résoudre le problème inverse revient alors à trouver l'image qui offre le meilleur compromis entre le modèle *a priori* et la fidélité à l'observation. Les modèles génératifs profonds permettent de définir des modèles probabilistes sur des données complexes, que l'on peut exploiter comme des modèles *a priori* pour résoudre des problèmes de restauration d'image difficiles. Grace a leur paramétrisation par des réseaux de neurones profonds, les modèles génératifs profonds sont particulièrement performants, mais aussi complexes à manipuler. Ainsi, l'utilisation de ces modèles pour la restauration d'images pose de nombreux défis, que l'on aborde dans ce travail. En premier lieu, on propose une méthode qui permet d'entrainer un réseau de neurones comme une fonction de régularisation lorsque l'on ne dispose seulement d'un ensemble d'exemples de patchs dégradés et d'un ensemble d'exemples de patchs propres. Pour cela, on présente une stratégie d'entrainement adversarielle, et on impose une architecture convolutionnelle au réseau pour permettre de l'entrainer seulement sur des patchs. Par la suite, on étudie l'utilisation d'auto-encodeurs variationnels (VAE) hiérarchiques pour la résolution de problèmes inverses. En particulier, on présente PnP-HVAE un nouvel algorithme flexible basé sur l'utilisation d'un VAE hiérarchique comme modèle *a priori*. PnP-HVAE prend la forme d'un algorithme d'optimisation alterné, et exploite l'encodeur du VAE pour manipuler de manière efficace les variables latentes. De plus, PnP-HVAE permet de contrôler le niveau de régularisation par le biais de la température de la distribution *a priori* sur l'espace latent du VAE. Ensuite, on présente une méthode dédiée à la super-résolution qui permet de produire des échantillons de la distribution *a posteriori* du problème de super-résolution grâce à un encodeur entrainé sur des images de basse résolution. Expérimentalement, on montre que nos méthodes basées sur l'utilisation de VAE hiérarchiques procurent un compromis avantageux entre l'efficacité calculatoire et la qualité de la restauration.
Mots-clés : Restauration d'images, Problèmes inverses, Modèles génératifs, Auto-encodeur variationnel

---

Image restoration with deep generative models

Abstract: Image restoration tasks, such as deblurring, or super-resolution, are inverse problems, as we seek to retrieve a clean image from a degraded observation. In order to determine how to recreate the missing information in the degraded observation, it is necessary to define a prior model of the expected solution. Then, solving the inverse problem amounts to finding an image that provides a good compromise between the prior model and fidelity with the observation. Deep generative models can define accurate probabilistic models of complex data distribution, that can be exploited as a prior model to solve challenging image inverse problems. Deep generative models are parameterized by deep neural networks which make them difficult to manipulate. Hence, using deep generative models for image restoration raises several challenges, that we aim to address in this thesis. First we consider to problem of defining a neural-network regularization function when training data are limited. Specifically, we introduce an adversarial strategy to train a regularization network without labeled dataset, and with only examples of small patches from clean and degraded images. Next, we investigate the use of hierarchical variational autoencoders (HVAEs) for solving image inverse problems. In particular, we introduce PnP-HVAE, a flexible algorithm that exploit a pretrained HVAE model as a prior to solve image inverse problems. PnP-HVAE is motivated by an alternate optimization scheme, and it exploits the HVAE encoder to manipulate the HVAE latent variables efficiently. Additionally, it enables us to control the strength of the regularization by tuning the temperature of the HVAE latent prior. Then we present a method specialized in super-resolution. We show that, by combining an encoder trained on low-resolution images with the HVAE generative model, we can sample from the posterior distribution of the super-resolution problem with only one network evaluation. We demonstrate that by exploiting the HVAE encoder we can develop image restoration methods that provide an advantageous trade of between computational efficiency and restoration quality.
Keywords: Image restoration, Inverse problem, Deep generative model, Variational autoencoder

---

Unité de recherche
UMR 5251 Université, 33000 Bordeaux, France.

Contents

Chapter 1

Résumé long en français

1.1 Problème inverse

Problème inverse linéaire Les problèmes de restauration d'images, comme le défloutage ou la super-résolution sont des problèmes inverses, pour lesquel on cherche à retrouver une image propre à partir d'une observation dégradée. Mathématiquement, on note $\boldsymbol{x} \in \mathbb{R}^n$ l'image propre que l'on cherche à retrouver, et $\boldsymbol{y} \in \mathbb{R}^m$ l'image observée degradée, et l'on supposera que les variables $\boldsymbol{x}$ et $\boldsymbol{y}$ sont connectées par le modèle de dégradation linéaire :

$$\boldsymbol{y} = A\boldsymbol{x} + \epsilon. \tag{1.1}$$

$A \in \mathbb{R}^{n \times m}$ est une matrice et $\epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2 \,\mathrm{Id})$ est un bruit blanc Gaussien. En ajustant l'opérateur A, on peut décrire de nombreux problèmes de restauration d'images avec le modèle (1.1). Par exemple, pour un problème de défloutage, l'opérateur A correspond à une convolution avec un noyau de flou. Pour un problème d'inpainting (c'est à dire de complétion de pixels manquants), A correspond à un masque qui cache certains pixels de l'image.

Maximum *a posteriori* Les problèmes inverses (1.1) sont généralement mal-posés, car l'opérateur A est mal-conditionné, ou n'est pas de rang plein. Ainsi, il est nécessaire de régulariser le problème afin de trouver une solution satisfaisante. Dans un cadre Bayésien, cela peut se faire en considérant un modèle statistique de la solution a-priori, que l'on notera $p(\boldsymbol{x})$. En combinant la loi *a priori* et la loi de vraisemblance $p(\boldsymbol{y}|\boldsymbol{x})$, on peut définir la loi *a posteriori* en utilisant la formule de Bayes:

$$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{y})}. \tag{1.2}$$

À partir de la distribution postérieure, on peut par exemple chercher à calculer l'estimateur du maximum *a posteriori* (MAP) :

$$\hat{\boldsymbol{x}}_{map} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}) \tag{1.3}$$

$$= \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}) + g(\boldsymbol{x}), \tag{1.4}$$

où l'on dénote $f(\boldsymbol{x})$ le terme d'attache aux données $f(\boldsymbol{x}) = \frac{1}{2\sigma^2}\|A\boldsymbol{x} - \boldsymbol{y}\|^2 = -\log p(\boldsymbol{y}|\boldsymbol{x}) + C$, et $g(\boldsymbol{x})$ le terme de régularisation $g(\boldsymbol{x}) = -\log p(\boldsymbol{x})$.

Régularisation classique Pour calculer l'estimateur MAP (1.3), il est nécessaire de définir une loi de probabilité *a priori* $p(\boldsymbol{x})$, ou la fonction de régularisation associée $g(\boldsymbol{x}) = -\log p(\boldsymbol{x})$. Par exemple, une approche classique est de définir la fonction de régularisation $g(\boldsymbol{x})$ comme la variation totale [Rudin et al., 1992a] de l'image afin de favoriser les solutions constantes par morceaux.

Régularisation apprise Récemment, de nombreux travaux ont développé des méthodes de régularisation basées sur l'apprentissage profond (deep learning), en implémentant des fonctions de régularisation ou des modèles *a priori* avec des réseaux de neurones entrainées sur de grandes bases de données. Les méthodes de régularisation "deep-learning" exploitent l'expressivité des réseaux de neurones pour résoudre des problèmes inverses difficiles. Néanmoins, l'utilisation de ces méthodes pose de nombreuses questions, que l'on va adresser dans cette thèse. En particulier, on s'intéressera aux problématiques liées à l'apprentissage de fonctions de régularisation quand le nombre de données est limité. On s'intéressera ensuite à la définition de méthodes pour la régularisation avec des modèles génératifs profonds, qui soient efficaces et qui procurent des garanties de convergence.

1.2 Régularisation adversarielle locale

Dans le chapitre 3 de ce document, on introduit la régularisation adversarielle locale, une méthode qui a pour but d'entrainer un réseau de neurones comme une fonction de régularisation quand le nombre de données d'entrainement est limité. Notre méthode permet d'entrainer une fonction de régularisation seulement avec des exemples de patchs propres et dégradés. On introduit une fonction de régularisation locale, r_θ, qui prend comme entrée un patch d'image et retourne un score scalaire.

Entrainement adversariel Pour entrainer la fonction de régularisation locale, l'on considère le critère d'entrainement inspiré par les Wasserstein GANs [Gulrajani et al., 2017] et la régularisation adversarielle [Lunz et al., 2018]. Étant donné une distribution de

patchs propres $\mathbb{P}_c$ et une distribution de patchs dégradés $\mathbb{P}_n$, le critère d'entrainement à maximiser est défini comme :

$$D(\theta) = \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_n}\left[r_\theta(\boldsymbol{z})\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_c}\left[r_\theta(\boldsymbol{z})\right] - \mu\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_i}[(||\nabla_{\boldsymbol{z}} r_\theta(\boldsymbol{z})||_2 - 1)^2]. \tag{1.5}$$

Intuitivement, ce critère permet de maximiser l'écart entre les valeurs que prend $r_\theta(\boldsymbol{z})$ sur les patchs dégradés et sur les patchs propres. De plus, un terme de régularisation (à droite), évalué sur $\mathbb{P}_i$, la distribution de toutes les lignes connectant des points échantillonnés dans $\mathbb{P}_n$ et $\mathbb{P}_c$, pénalise la norme du gradient de $r_\theta(\boldsymbol{z})$ pour forcer $r_\theta(\boldsymbol{z})$ être $1-$Lipschitz. On détaillera dans le chapitre 3 qu'il est possible de donner une interprétation géométrique à la fonction de régularisation r_θ, en établissant des connexions avec le transport optimal.

Régularisation globale On définit ensuite une fonction de régularisation "globale", à partir de la fonction de régularisation locale, Pour une image $\boldsymbol{x}$, en dénotant $\Omega_x = \{x_1, \cdots, x_n\}$ l'ensemble des patchs de $\boldsymbol{x}$, la fonction de régularisation globale est définie comme la valeur moyenne de r_θ sur l'ensemble des patchs de $\boldsymbol{x}$:

$$g(\boldsymbol{x}) = \frac{1}{|\Omega_x|}\sum_{x_i\in\Omega_x} r(x_i). \tag{1.6}$$

En pratique, on implémente la fonction de régularisation locale comme un réseau convolutionnel dont le champ réceptif correspond à la taille des patchs de la base de donnée d'entrainement, de telle sorte à ce que l'on puisse évaluer (1.6), en appliquant le réseau sur l'image entière et en moyennant la sortie du réseau.

1.3 Régularisation de problèmes inverses avec des Auto-encodeurs variationnels hiérarchiques

Auto-encodeurs variationnels Un auto-encodeur variationnel (VAE) [Kingma and Welling, 2013] permet d'apprendre les paramètres θ d'un modèle à variable latente :

$$p_\theta(\boldsymbol{z}, \boldsymbol{x}) = p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{z}|\boldsymbol{x}), \tag{1.7}$$

paramétrisé par un réseau de neurones. Dans l'équation (1.7), $\boldsymbol{z} \in \mathbb{R}^d$ est une variable latente, $p_\theta(\boldsymbol{z})$ est la loi *a priori* sur l'espace latent, et $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ est la distribution du décodeur qui transforme une variable latente $\boldsymbol{x}$ en (une distribution) sur les images. Un VAE inclut aussi un encodeur :

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x}), \tag{1.8}$$

qui est implémenté par un réseau de neurones, et qui est entrainé pour approcher la loi *a posteriori* $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ qui est n'est pas calculable facilement. Un VAE hiérarchique est

un VAE qui impose une structure hiérarchique sur la distribution *a priori* $p_\theta(\boldsymbol{z})$ pour augmenter l'expressivité du modèle génératif [Sønderby et al., 2016]. Ainsi la loi *a priori* sur l'espace latent d'un VAE hiérarchique est défini comme :

$$p_\theta(\boldsymbol{z}) = p_\theta(\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{L-1}) \tag{1.9}$$

$$= p_\theta(\boldsymbol{z}_0) \prod_{\ell=1}^{L-1} p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}), \tag{1.10}$$

et les lois conditionnelles $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$ sont définies comme des Gaussiennes, dont les statistiques sont données par des réseaux de neurones. L'encodeur d'un VAE hiérarchique est paramétré par un modèle hiérarchique similaire à (1.10). Les VAEs et les VAE hiérarchiques sont présentés dans le chapitre 4. Les VAEs hiérarchiques permettent de définir des modèles génératifs plus expressifs que les VAE classiques [Kingma and Welling, 2013]. Néanmoins, la structure hiérarchique du modèle introduit une complexité supplémentaire, qui rend son utilisation pour la régularisation de problème inverse difficile.

Optimisation alternée Pour résoudre un problème inverse avec une loi *a priori* induite par le modèle génératif d'un VAE hiérarchique, on définit le modèle augmenté induit par la composition de la loi jointe apprise par le VAE (1.7) et la vraisemblance induite par le modèle de dégradation (1.1):

$$p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) \propto p_\theta(\boldsymbol{z})^{\frac{1}{\tau^2}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{y}|\boldsymbol{x}). \tag{1.11}$$

Dans (1.11), nous avons introduit un paramètre de température τ pour contrôler la force de la régularisation. En suivant une approche similaire à l'algorithme JPMAP [González et al., 2022], on propose de calculer l'estimateur du maximum de vraisemblance "joint" :

$$\boldsymbol{x}^\star, \boldsymbol{z}^\star = \arg\max_{\boldsymbol{x},\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{y}), \tag{1.12}$$

en utilisant un algorithme d'optimisation alternée :

$$\boldsymbol{z}^{(n+1)} = \arg\max_{\boldsymbol{z}} q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(n)}\Big) p_\theta(\boldsymbol{z})^{\left(\frac{1}{\tau^2}-1\right)} \tag{1.13}$$

$$\boldsymbol{x}^{(n+1)} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{y}|\boldsymbol{x}) p_\theta\Big(\boldsymbol{x}|\boldsymbol{z}^{(n+1)}\Big) \tag{1.14}$$

Dans l'étape d'optimisation en $\boldsymbol{z}$ (1.13), on utilise l'encodeur du VAE hiérarchique $q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(n)}\Big)$ pour remplacer la loi postérieure $p_\theta\Big(\boldsymbol{z}|\boldsymbol{x}^{(n)}\Big)$ inaccessible. Cette approximation permet d'éviter d'avoir à employer une optimisation itérative avec rétropropagation couteuse pour calculer $\arg\max_{\boldsymbol{z}} p_\theta(\boldsymbol{z}|\boldsymbol{x})$. L'étape (1.13) peut être interprétée comme une forme d'interpolation entre l'encodeur $q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(n)}\Big)$ et la distribution *a priori* $p_\theta(\boldsymbol{z})$. Avec un VAE hiérarchique cette étape n'est pas évidente à résoudre de manière exacte. On propose un algorithme séquentiel qui exploite la structure "top-down" de l'encodeur pour calculer une solution approchée ne nécessitant qu'une application de l'encodeur. De plus, on démontre que cet algorithme produit la solution exacte du problème sous des conditions raisonnablement vérifiables.

Connections avec les algorithmes Plug-and-Play Dans le cas particulier où $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; \mu_\theta(\boldsymbol{z}), \gamma^2 I)$, on montre que notre algorithme peut se réécrire de la manière suivante :

$$\boldsymbol{x}_{k+1} = \text{prox}_{\gamma^2 f}(\text{HVAE}(\boldsymbol{x}_k, \boldsymbol{\tau})), \tag{1.15}$$

où l'on note $\text{HVAE}(\boldsymbol{x}, \tau)$ la reconstruction par l'auto-encodeur hiérarchique avec la "régularisation" latente induite par le paramètre τ, définie telle que $\text{HVAE}(\boldsymbol{x}, \tau) := \mu_\theta\left(\boldsymbol{z}^{(n+1)}\right)$, $\boldsymbol{z}^{(n+1)}$ est donnée par l'équation (1.13), et $f(\boldsymbol{x}) = \frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2$ est le terme d'atache aux données. Ainsi l'algorithme peut être vu comme un algorithme Plug-and-Play, où la reconstruction par le VAE hiérarchique joue le rôle du réseau débruiteur. On nomme donc notre méthode PnP-HVAE. La formulation de PnP-HVAE (1.15) nous permet d'établir une condition suffisante pour garantir la convergence vers un point fixe. Si l'opération de reconstruction par le VAE hiérarchique $\text{HVAE}(\boldsymbol{x}, \tau)$ est contractante, alors PnP-HVAE converge vers un point fixe $\boldsymbol{x}^\star$, qui vérifie :

$$\nabla f(\boldsymbol{x}^\star) = \frac{1}{\gamma^2}(\text{HVAE}(\boldsymbol{x}^\star, \boldsymbol{\tau}) - \boldsymbol{x}^\star). \tag{1.16}$$

Contrairement aux travaux précédents [González et al., 2022], ce résultat ne dépend pas de l'hypothèse $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = p_\theta(\boldsymbol{z}|\boldsymbol{x})$. En pratique, on observe que les itérations de PnP-HVAE sont effectivement stables.

Résultats Expérimentalement, on évalue PnP-HVAE sur la restauration d'images de visages, pour lesquelles on peut trouver de nombreux modèles génératifs pré-entrainés. On montre que PnP-HVAE produit de meilleurs résultats que les méthodes concurrentes basées sur des GANs [Goodfellow et al., 2014a] ou des modèles de diffusion [Sohl-Dickstein et al., 2015], tout en étant plus rapide.

1.4 Super-résolution diverse avec des auto-encodeurs variationnels hiérarchiques

Super-résolution diverse Dans cette section, correspondant au chapitre 6 de cette thèse, on s'intéresse à un problème de super-résolution. En particulier on cherche à produire des échantillons de la distribution postérieure du problème $p(\boldsymbol{x}|\boldsymbol{y})$. Pour ce faire, on définit comme loi *a priori* sur les solutions haute-résolution (HR) un modèle définit par un VAE hiérarchique,

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}, \tag{1.17}$$

Encodeur basse-résolution On propose d'entrainer un nouvel encodeur pour les images basse-résolution $q_\psi(\boldsymbol{z}|\boldsymbol{y})$, on définit ensuite notre modèle de super-résolution comme la

combinaison de l'encodeur basse-résolution et du décodeur du VAE hiérarchique :

$$p_{SR}(\boldsymbol{x}|\boldsymbol{y}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) q_\psi(\boldsymbol{z}|\boldsymbol{y}) \mathrm{d}\boldsymbol{z}. \tag{1.18}$$

En particulier, on entraine l'encodeur basse-résolution en utilisant l'encodeur du VAE hiérarchique, pour minimiser :

$$\mathcal{L}(\psi) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}[\mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||q_\psi(\boldsymbol{z}|\boldsymbol{y}))]. \tag{1.19}$$

On montre que minimiser (1.19) correspond à maximiser une borne inférieure sur la log-vraisemblance de $p_{SR}(\boldsymbol{x}|\boldsymbol{y})$ sur l'ensemble d'apprentissage $p_{\text{data}}(\boldsymbol{x}, \boldsymbol{y})$. De plus, on montre qu'il est possible d'exploiter la structure hiérarchique du VAE pour restreindre l'encodeur $q_\psi(\boldsymbol{z}|\boldsymbol{y})$ sur la sous-partition de $\boldsymbol{z}$ qui contient effectivement l'information présente dans l'image basse-résolution $\boldsymbol{y}$.

Résultats Expérimentalement, on applique notre méthode en utilisant VDVAE [Child, 2020]. On montre que notre méthode permet de produire des échantillons de qualité équivalente ou supérieure aux meilleures méthodes de l'état de l'art, basées sur des modèles de diffusion. De plus, notre méthode est significativement plus rapide ($\times$1000) que les méthodes concurrentes, car elle ne nécessite qu'une seule application successive de l'encodeur $q_\psi(\boldsymbol{z}|\boldsymbol{y})$ et du modèle génératif $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ du VAE hiérarchique.

Chapter 2

Introduction

The work presented in this document is about the development of new methodologies to restore images by using deep generative models. In this first chapter, we introduce the main concepts that will be useful throughout this document. We start by presenting the mathematical formulation of image restoration problems as inverse problems, and we provide an overview of the existing methodologies on the subject, including optimization and sampling based methods. In particular, we will discuss the strategies to define a prior over the solution, with a specific focus on recent deep learning based methods, including deep generative models. We then discuss the existing challenges of using deep generative models for restoring images, and we close this chapter by presenting our contributions and the outline of the rest of this thesis.

2.1 Image inverse problem

2.1.1 Presentation

Motivation With the increasing availability of sensors, images have become ubiquitous in our daily life, be it for recreational usage or for industrial and scientific applications. Technical limitations of the sensor, along with external factors such as motion or low-light exposure, can cause a degradation of the measured images. As such, it is of crucial importance to develop methods to restore the degraded images. Restoring an image can be viewed as an inverse problem, where we seek to retrieve a clean signal from a degraded measurement.

Linear inverse problem From a mathematical perspective, restoring an image amounts to solving an inverse problem, where we seek to retrieve a clean image from a degraded observation. Both the observed and the underlying clean images are modeled as finite dimensional vectors. Throughout this work, we will denote $\boldsymbol{y} \in \mathbb{R}^m$ the degraded ob-

servation, and $\boldsymbol{x} \in \mathbb{R}^n$ the underlying clean image. A generic degradation model, that encompasses a large class of image restoration tasks, is the linear degradation model:

$$\boldsymbol{y} = A\boldsymbol{x} + \epsilon, \tag{2.1}$$

where $A \in \mathbb{R}^{n \times m}$ is a linear operator, and $\epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2 \operatorname{Id})$ is a white Gaussian noise.

Examples By setting the linear operator A in (2.1) to be a convolution with a blur kernel h:

$$A\boldsymbol{x} = h \star \boldsymbol{x}, \tag{2.2}$$

we recover an image deblurring problem. (2.1) can also model an inpainting problem, by setting A to be a diagonal matrix, with $A_{ii} = 0$ on the masked pixel, and $A_{ii} = 1$ otherwise. For image super-resolution, a typical degradation operator is the concatenation of a low-pass filter (convolution with a blurring kernel), and a downsampling operator:

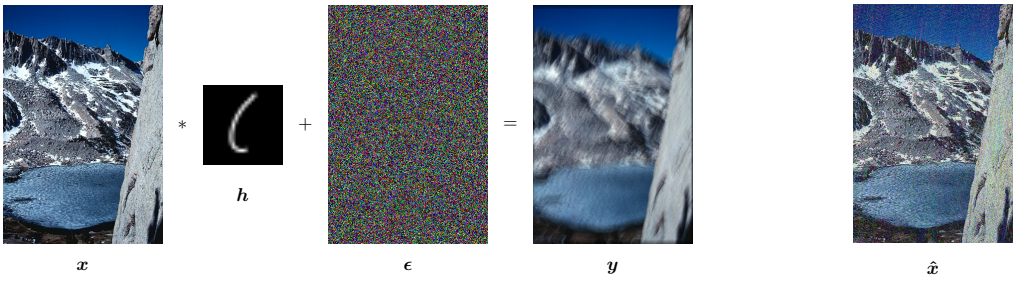
$$A\boldsymbol{x} = (h \star \boldsymbol{x}) \downarrow_s . \tag{2.3}$$

In this work, we will assume that the linear operator A is known, although in some settings, A is also unknown, and needs to be determined jointly with $\boldsymbol{x}$. We refer to those problems as blind inverse problems.

Ill-posedness Image inverse problems are typically ill-posed. This can be due to the fact that the linear system is under-determined ($m < n$, or $\operatorname{rank}(A) < n$), or that the linear operator A is ill-conditioned. For instance, for image inpainting, the system is under-determined, as the information on the missing pixels is not recoverable. For image deblurring, the linear operator associated to the blurring kernel is ill-conditioned, so that the naive solution $A^{-1}\boldsymbol{y}$ will contain severe high-frequency artifact, as can be seen in Figure 2.1b. Those artifacts are due to the noise in the observation $\boldsymbol{y}$ being amplified by the inverse A^{-1}.

2.1.2 End-to-end image restoration

Convolutional neural networks for end-to-end image restoration Since 2012 and the milestone success of AlexNet on the ImageNet large scale visual recognition challenge [Krizhevsky et al., 2017], convolutional neural networks (CNN) [Rosenblatt, 1958, LeCun et al., 1998] have revolutionized the field of computer vision, by providing state-of-the art results on a large number of vision tasks [Li et al., 2021, Dosovitskiy et al., 2015, Redmon et al., 2016, Minaee et al., 2021]. CNNs process their input by alternating local linear operations and non-linear point-wise operation in a similar fashion than classical optimization algorithms used for solving image inverse problems, motivating their usage to perform image restoration [Dong et al., 2015, Gregor and LeCun, 2010, Diamond et al.,



(a) forward model (b) inverse

Figure 2.1 – forward model (a) and solution $\hat{\boldsymbol{x}}$ of the system $h * \hat{x} = \boldsymbol{y}$ (b) for a motion deblurring problem, with $\sigma = 0.01$. A simple inversion of the linear system leads to severe artifacts. The range of the noise ϵ was augmented in the figure for visualization purpose.

2017, Wang et al., 2015]. A straight-forward way to use a CNN for image restoration is to train it to map degraded images to their clean version in an end-to-end fashion. End-to-end methods were shown to outperform their concurrent methods on large number of image restoration tasks, including image denoising [Zhang et al., 2017a], colorization [Cheng et al., 2015], or inpainting [Köhler et al., 2014].

Limitations of end-to-end methods However, end-to-end image restoration methods still have important limitations that restrain their application in practical contexts. End-to-end methods require large training datasets composed of pairs of clean and degraded images. They also lack flexibility, as one separate network needs to be trained for every different type of problem. Furthermore, because a CNN models a deterministic mapping, it only produces one solution, without accounting for the uncertainty of the solution due to the ill-posedness of the inverse problem. Plus, end-to-end methods generally do not account for the forward model (2.1), which can induce inconsistency between the restored image and the network input. A specific class of end-to-end image restoration methods based on unrolling optimization can actually account for the degradation model within their architecture, but they still inherit from the others caveats of end-to-end methods [Diamond et al., 2017].

2.1.3 Bayesian perspective

To address the limitations of end-to-ends methods, one can adopt a decoupled approach, by separating the modelling of the fidelity with the observation and the modelling of our expectation on the properties of the solution. This can be done by adopting a Bayesian perspective, as discussed in the following paragraphs.

Posterior distribution In order to tackle the ill-posedness of image inverse problems, it is necessary to incorporate some form of regularization. Under a Bayesian perspective, this can be done by considering the posterior distribution $p(\boldsymbol{x}|\boldsymbol{y})$, which, by the Bayes' law, can be written as:

$$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{y})}. \tag{2.4}$$

Equation (2.4) indicates that the posterior is proportional to the product of the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$ and the prior $p(\boldsymbol{x})$. The likelihood $p(\boldsymbol{y}|\boldsymbol{x})$ measures how likely it is to observe $\boldsymbol{y}$ knowing that the clean signal is $\boldsymbol{x}$, and is dependent on the degradation model. For the linear degradation model (2.1), it is defined as:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{y}; A\boldsymbol{x}, \sigma^2 \operatorname{Id}\right) \tag{2.5}$$

$$= \frac{1}{\sqrt{(2\pi\sigma^2)^m}} \exp\left(-\frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2\right). \tag{2.6}$$

The prior distribution $p(\boldsymbol{x})$ models our knowledge of the solution independently of the observation $\boldsymbol{y}$. Unlike the likelihood, there is no explicit, physical definition of the prior. Rather, it has to be selected so that it best fits our prior assumption on the solution. We will discuss in section 2.2 what strategies can be used to define a suitable prior.

Energy function In practice, it is more convenient to deal with the energy of the log posterior, defined (up to additive constant) as:

$$E(\boldsymbol{x}) = -\log p(\boldsymbol{x}|\boldsymbol{y}). \tag{2.7}$$

Notice that the likelihood verifies $p(\boldsymbol{y}|\boldsymbol{x}) \propto \exp(-f(\boldsymbol{x}))$, with $f(\boldsymbol{x}) = \frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2$. Assuming that the prior verifies $p(\boldsymbol{x}) =\propto \exp(-g(\boldsymbol{x}))$, the posterior energy then writes (up to additive constant):

$$E(\boldsymbol{x}) = f(\boldsymbol{x}) + g(\boldsymbol{x}) \tag{2.8}$$

The formulation (2.8) is widely used in many methods, including optimization and sampling based methods.

Maximum a posteriori estimator A widely used approach to perform image restoration is to compute the Maximum a posteriori (MAP) estimator:

$$\hat{x}_{map} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}) \tag{2.9}$$

$$= \arg\min_{\boldsymbol{x}} E(\boldsymbol{x}) \tag{2.10}$$

$$= \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}) + g(\boldsymbol{x}) \tag{2.11}$$

Hence, computing a MAP amounts to solving the optimization problem (2.11). The problem (2.11) can be viewed as a variational problem under the perspective that the unknown variable $\boldsymbol{x}$ is a function mapping pixel coordinates to their intensity values. With the Gaussian linear degradation model (2.5), $f(\boldsymbol{x})$ is convex, so if $g(\boldsymbol{x})$ is convex, $E(\boldsymbol{x})$ will be convex in turn. Then, one can rely on the extensive literature on convex optimization to derive efficient optimization algorithms, that provably converge to the solution of (2.11) with explicit convergence rates. In particular, we can use specific algorithms which take advantage of the composite structure of $E(\boldsymbol{x})$, such as forward-backward [Beck and Teboulle, 2009], alternate direction of multipliers (ADMM) [Boyd et al., 2011], or half-quadratic splitting [Geman and Yang, 1995].

Posterior sampling The MAP estimator provides one solution to the restoration problem, but, for some applications, this single point estimation is not sufficient. For instance, for creative applications, one might want to select one solution among diverse samples. In Bayesian inference, some applications such as uncertainty quantification or model selection require computing integrals of the form:

$$\int \varphi(\boldsymbol{x}) p(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x}. \tag{2.12}$$

In high dimensional setups, the integral (2.12) is typically intractable, but, when having access to a set of independent samples from the posterior $p(\boldsymbol{x}|\boldsymbol{y})$, we can compute an unbiased, low-variance estimation using the central limit theorem. Markov Chain Monte Carlo (MCMC) algorithms provide a well established framework for producing samples from the posterior distribution. MCMC methods work by constructing a Markov chain whose stationary distribution is the posterior $p(\boldsymbol{x}|\boldsymbol{y})$. Classical MCMC algorithms include Metropolis Hasting, Gibbs sampling and Langevin dynamic [Pereyra et al., 2015]. Like the above-mentioned variational methods, popular MCMC methods used for image inverse problems make use of the gradient or the proximal operator of $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ to efficiently explore the different modes of the posterior [Durmus et al., 2018]. They can also rely on variable splitting strategies similar to the one used in splitting algorithms used to compute the MAP estimator [Vono et al., 2019, Pereyra et al., 2022].

2.2 Modeling a prior on images

Contrary to the data-fidelity term, there is no clear, straight-forward choice for the regularization term $g(\boldsymbol{x})$, and it has to be defined as a way to enforce the user prior assumptions on the solution. In this subsection, we first provide a brief review of the classical strategies available in the literature, and we elaborate on the recent data-driven strategies that use powerful neural networks to define a prior.

2.2.1 Classical methods

Convex priors A widely used choice of regularization term is the total variation [Rudin et al., 1992b]:

$$g(\boldsymbol{x}) = ||\nabla \boldsymbol{x}||_2, \tag{2.13}$$

which promotes piecewise constant images. Other approaches promote sparsity in a transformed domain by penalizing the ℓ^1 norm of transform-domain coefficients:

$$g(\boldsymbol{x}) = ||W\boldsymbol{x}||_1. \tag{2.14}$$

Here, the transform W represents Wavelet frames [Donoho and Johnstone, 1994, Coifman and Donoho, 1995] or local Fourier or DCT representations [Yu and Sapiro, 2011]. It is also possible to define W as a set of learned filters with dictionary learning [Aharon et al., 2006]. Priors of the form (2.13) and (2.14) are convex (albeit non-smooth), making the optimization problem (3.2) convex in turn. We can then use efficient and well studied convex optimization algorithms to solve the variational problem, with theoretical convergence guarantees. Those terms are easily interpretable, as it is clear what information will remains in the final solution, and what information will be dismissed. Nevertheless, those handcrafted priors tend to produce over-smoothed or suboptimal results, since they only represent a rough approximation of natural image statistics and geometry.

Non-convex priors In order to better capture the statistics of images, priors based on Gaussian mixture models [Zoran and Weiss, 2011a], or fields of experts [Roth and Black, 2005] were introduced. Those priors are data-driven as their parameters are adjusted to a set of training images. Due to the complexity of fitting a statistical model on high-dimensional images, those approaches consider instead fitting a model on image patches of smaller dimension. Data driven priors give improved performance on restoration tasks compared to the aforementioned hand-crafted methods priors, while staying interpretable due to their simple formulation.

2.2.2 Deep learning priors

Adversarial regularization We can use a CNN to model a complex prior that fits the statistics of images. A direct way of doing so is to train a CNN $g_\theta(\boldsymbol{x})$ so that $g_\theta(\boldsymbol{x}) \approx \log p(\boldsymbol{x})$, in order to match the MAP interpretation (2.11). However, this is not feasible because we do not have access to the true prior energy $\log p(\boldsymbol{x})$. An explicit regularization network $g_\theta(\boldsymbol{x})$ should penalize inappropriate solutions and encourage relevant ones. Following this intuition, one can train $g_\theta(\boldsymbol{x})$ as a classifier between clean and degraded images. This approach is formalized in the adversarial regularization framework of [Lunz et al., 2018]. By exploiting optimal transport theory and the recent literature on Wasserstein GANs [Arjovsky et al., 2017], [Lunz et al., 2018] showed that the learned

regularization can be related to the distance to the clean images manifold. Further works have investigated modeling the regularization with an input convex neural network [Amos et al., 2017] in order to facilitate the resolution of the inverse problem [Mukherjee et al., 2023].

Plug-and-play regularization In the literature, the variational problem (2.11) is typically solved using optimization algorithms like forward-backward [Beck and Teboulle, 2009] or ADMM [Boyd et al., 2011]. Those algorithms do not require evaluating $g(\boldsymbol{x})$, but only require access to its gradient $\nabla g(\boldsymbol{x})$, or its proximal operator, defined as:

$$\text{prox}_{\alpha g}(\mathbf{u}) = \arg\min_{\boldsymbol{x}} \frac{1}{2}||\boldsymbol{x} - \mathbf{u}||^2 + \alpha g(\boldsymbol{x}) \tag{2.15}$$

Therefore, one can avoid the difficult problem of learning a potential function $g(\boldsymbol{x})$ by focusing instead on learning its gradient or its proximal operator. It is possible to do so by exploiting denoising autoencoders [Vincent, 2011]. Indeed, the authors of [Venkatakrishnan et al., 2013] noticed that the proximal operator in (2.15) can be viewed as solving a MAP problem on a denoising problem (with $p(\boldsymbol{y}|\boldsymbol{x}) \propto \exp\left(||\boldsymbol{x} - \boldsymbol{y}||^2/(2\alpha)\right)$), and proposed to replace the proximal operator (2.15) within the optimization process by the application of a denoiser trained on noise level $\sigma^2 = \alpha$. For instance, the plug-and-play forward-backward iteration writes as:

$$\boldsymbol{x}_{k+1} = D_{\sqrt{\alpha}}(\boldsymbol{x}_k - \tau \nabla f(\boldsymbol{x}_k)). \tag{2.16}$$

Plug-and-play regularization can exploit denoisers based on filtering methods, such as Non Local Mean [Buades et al., 2005] or BM3D [Dabov et al., 2007], although denoising autoencoders based on deep neural networks trained in a supervised fashion were shown to provide the best results in terms of restoration quality [Zhang et al., 2017a, Meinhardt et al., 2017, Zhang et al., 2021].

Denoising score matching A denoiser can also be related to the gradient of a smoother version of the image prior through Tweedie's formula [Robbins, 1992, Vincent, 2011]. Let us consider the joint model of clean and noisy data $p_\sigma(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = p_{\text{data}}(\boldsymbol{x}) p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})$ with $p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{x}}|\boldsymbol{x}, \sigma^2 I)$ a Gaussian kernel. If we train a denoiser $D_\sigma(\tilde{\boldsymbol{x}})$ so that it predicts the minimal mean square error (MMSE) estimator of the denoising problem for noise level σ:

$$D_\sigma(\tilde{\boldsymbol{x}}) = \arg\min_{\mathbf{u}} \mathbb{E}_{p_\sigma(\boldsymbol{x}|\tilde{\boldsymbol{x}})}\left[||\boldsymbol{x} - \mathbf{u}||^2\right], \tag{2.17}$$

then, according to Tweedie's formula,

$$D_\sigma(\tilde{\boldsymbol{x}}) = \tilde{\boldsymbol{x}} - \sigma^2 \nabla \log p_\sigma(\tilde{\boldsymbol{x}}), \tag{2.18}$$

Therefore, it is possible to deduce the gradient of a "noisy" version of the data distribution, $p_\sigma(\tilde{\boldsymbol{x}})$ from an image denoiser by exploiting equation (2.18). The gradient $\nabla \log p_\sigma(\tilde{\boldsymbol{x}})$ can

then be used within an optimization or a sampling framework [Bigdeli et al., 2017,Laumont et al., 2022]. Tweedie's formula is also key in the formulation of denoising diffusion model, that will be presented in section 2.3.1.

2.3 Deep generative models as an image prior

Deep generative models are a class of probabilistic models that are trained to transform noise into samples matching those from a training data distribution. In order to model complex distributions, deep generative models employ neural networks to model the transformation between noise and data. In this section we introduce the different paradigms to train a deep generative model, and we then present different existing approaches to use deep generative models as a prior to solve image inverse problems.

2.3.1 Deep generative models

Variational autoencoder The variational autoencoder (VAE) [Kingma and Welling, 2013] defines a latent variable model $p_\theta(\boldsymbol{z}, \boldsymbol{x}) = p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})$. A common choice for VAEs is to set the prior distribution over the latent variable as a Gaussian:

$$p_\theta(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; 0, I) \tag{2.19}$$

and the decoding distribution (or decoder) as another Gaussian:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; G_\theta(\boldsymbol{z}), \Sigma_\theta(\boldsymbol{z})), \tag{2.20}$$

with the mean and the covariance matrices parameterized by two neural networks:

$$\mu_\theta(.)\colon \mathbb{R}^d \to \mathbb{R}^n, \quad \text{and} \quad \Sigma_\theta(.)\colon \mathbb{R}^d \to \mathbb{R}^{n\times n}, \tag{2.21}$$

where θ corresponds to the parameters of the two neural networks. Additionally, the VAE is composed of an inference model (also known as an encoder), trained to match the (usually intractable) posterior of the model $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. VAE can model complex distributions thanks to the neural network parameterization of the decoder. However, it is not straight-forward to exploit a VAE to define a prior over images, because the marginal of the model:

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{z} \tag{2.22}$$

is intractable. We will provide a more in-depth discussion of VAEs and their usage for solving inverse problems in the following chapters.

Generative adversarial networks Generative adversarial networks (GANs) are another class of deep generative models that are trained with adversarial training [Goodfellow et al., 2014a]. A GAN generates a sample $\boldsymbol{x}$ by transforming latent variables sampled from a simple latent distribution $\boldsymbol{z} \sim p_{\mathcal{Z}}(\boldsymbol{z})$ using a generative network $\boldsymbol{x} = G_\theta(\boldsymbol{z})$. Here, θ denotes the set of parameters of the generative network. In other words, the probabilistic model learned by the GAN, $\mathbb{P}_\theta$, is defined as the push-forward of the latent distribution $p_{\mathcal{Z}}(\boldsymbol{z})$ through the generative network $G_\theta(\boldsymbol{z})$. As for the VAE, it is common to set the latent distribution to have a Gaussian density:

$$p_{\mathcal{Z}}(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; 0, I), \tag{2.23}$$

so that the distribution learned by the GAN model writes:

$$\mathbb{P}_\theta(E) = \int_{G_\theta^{-1}(E)} p_{\mathcal{Z}}(\boldsymbol{z}) d\boldsymbol{z}. \tag{2.24}$$

It should be noted that, due to the low dimensionality of $\boldsymbol{z}$, the support of $\mathbb{P}_\theta$ lies on a low-dimensional manifold of $\mathbb{R}^n$. Hence, the model distribution $\mathbb{P}_\theta$ does not admit a density function defined in $\mathbb{R}^n$, which can be a challenge for image restoration applications.

Normalizing flows Similar to GANs, a normalizing flow model is defined as the push-forward of a latent distribution through a generative network G_θ. As an additional constraint, G_θ is set to be a bijective mapping, so that the model $\mathbb{P}_\theta$ admits a density $p_\theta(\boldsymbol{x})$ that can be computed using the change of variable formula [Rezende and Mohamed, 2015]:

$$p_\theta(\boldsymbol{x}) = p_{\mathcal{Z}}(G_\theta^{-1}(\boldsymbol{x})) \left| \frac{\partial G_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}}(\boldsymbol{x}) \right|. \tag{2.25}$$

In order to enforce the bijectivity, and to be able to compute the Jacobian determinant in (2.25) efficiently, it is necessary to impose specific constraints on the architecture on the generative network [Kobyzev et al., 2020]. Hence, normalizing flows provide the ability to evaluate explicitly $p_\theta(\boldsymbol{x})$ at the cost of a limited expressivity compared to GANs and VAEs, due to architectural constraints.

Diffusion models Diffusion models, also known as score-based generative models, are a class of models that produce samples by gradually transforming noise into data using denoising autoencoders [Sohl-Dickstein et al., 2015, Song and Ermon, 2019, Ho et al., 2020, Song et al., 2021b]. Denoising diffusion models can be viewed through the lens of stochastic differential equation, by describing the sampling stage as the simulation of a stochastic differential equation (SDE) corresponding to the backward process of a diffusion process gradually transforming data into noise [Song et al., 2021b]. Formally, a forward diffusion process $\{\boldsymbol{x}_t\}_{t\in[0;T]}$ is constructed so that $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$ and $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T, 0, I)$.

For instance, the variance preserving forward diffusion process [Song et al., 2021b, Ho et al., 2020] is specified by the SDE:

$$\mathrm{d}\boldsymbol{x}_t = -\frac{\beta(t)}{2}\boldsymbol{x}_t \mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}, \tag{2.26}$$

where $\mathbf{w}$ is the standard Wiener process. The backward process associated with the diffusion process (2.26) is a solution of the reverse time SDE:

$$\mathrm{d}\boldsymbol{x}_t = \left(-\frac{\beta(t)}{2}\boldsymbol{x}_t - \beta(t)\nabla \log p\left(\boldsymbol{x}_t\right)\right)\mathrm{d}\bar{t} + \sqrt{\beta(t)}\mathrm{d}\overline{\mathbf{w}}, \tag{2.27}$$

where $\overline{\mathbf{w}}$ is a backward Wiener process, and $\mathrm{d}\bar{t}$ is an infinitesimal negative timestep [Anderson, 1982, Song et al., 2021b]. By construction, the marginals $p(\boldsymbol{x}_t)$ of the forward process (2.26) corresponds to the "smoothed" data distribution:

$$p(\boldsymbol{x}_t) = \int p_{\text{data}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_i}\boldsymbol{x}, (1-\alpha_i)\, I)d\boldsymbol{x}, \tag{2.28}$$

where α_i depends on the diffusion schedule $\beta(t)$. Hence, a score network $s_\theta(\boldsymbol{x}_t, t) \approx \nabla \log p\left(\boldsymbol{x}_t\right)$ can be trained with a denoising criterion [Vincent, 2011, Ho et al., 2020]. Using a SDE solver along with the learned score $s_\theta(\boldsymbol{x}_t, t)$, we can simulate the backward process (2.30) to produce samples matching the training data distribution.

2.3.2 Deep generative models for inverse problems

Generator inversion Deep generative models such as GANs or VAEs define a manifold on the image space corresponding to the range of the generative network $R(G_\theta) := \{\boldsymbol{x} \in \mathbb{R}^n; \exists \boldsymbol{z} \in \mathbb{R}^d \text{ so that } \boldsymbol{x} = G_\theta(\boldsymbol{z})\}$. Under the assumption that $R(G_\theta)$ corresponds to the set of clean images, one can restore a degraded image $\boldsymbol{y}$ by finding the image in the range of the generator that is the most consistent with the observation. This can be done by "inverting" the generator, that is, by finding the generator input $\boldsymbol{z}$ that corresponds the most to the (degraded) output $\boldsymbol{y}$. With a change of variable, the inversion problem is formulated as:

$$\boldsymbol{z}_{map} = \arg\min \frac{1}{2\sigma^2}||AG_\theta(\boldsymbol{z}) - \boldsymbol{y}||^2 + \lambda||\boldsymbol{z}||^2, \tag{2.29}$$

where the term $\lambda||\boldsymbol{z}||^2$ can be interpreted as a regularization term on the latent code. Enforcing the solution to lie in the range of the generator guarantees high-quality outputs, provided that the generative model is well trained. However, this constraint can also be a limitation, as there might not be any point on the generator range that correspond to a realistic solution of the inverse problem. In particular, GANs are notorious for their mode-seeking behavior, which can lead them to ignore some modes of the training distribution [Thanh-Tung and Tran, 2020]. Additionally, the problem (2.29) is non-convex, which implies that there is no guarantee to find a relevant solution even if it exists.

Algorithms for generator inversion Generative inversion has been first introduced in the context of compressed sensing in the seminal work of [Bora et al., 2017], where the cost function (2.29) is minimized using Adam optimization algorithm [Kingma and Ba, 2014]. The work of [Shah and Hegde, 2018] shows that projected gradient descent on the range of the generator converges to the solution with high-probability under certain conditions on the linear operators A. In order to relax the constraint of the solution living in the range of the generator, several works propose to extend the range of the generator by optimizing intermediate layers of the generator [Bau et al., 2019, Daras et al., 2021]. Expanding the range of the generator brings a significant improvement in terms of restoration quality [Daras et al., 2021], at the cost of increased complexity, due to the need of tuning a large number of hyperparameters for each restoration task and generator architecture.

Posterior sampling with denoising diffusion models The generative process of diffusion models involves simulating the SDE (2.30). In order to produce samples from the posterior of an inverse problem $p(\boldsymbol{x}|\boldsymbol{y})$, the reverse diffusion process (2.30) can be conditioned on an observation $\boldsymbol{y}$:

$$\mathrm{d}\boldsymbol{x}_t = \left(-\frac{\beta(t)}{2}\boldsymbol{x}_t - \beta(t)\nabla \log p\left(\boldsymbol{x}_t|\boldsymbol{y}\right)\right)\mathrm{d}\bar{t} + \sqrt{\beta(t)}\mathrm{d}\overline{\mathbf{w}}, \tag{2.30}$$

To do so, one can approximate the conditional score $\nabla \log p\left(\boldsymbol{x}_t|\boldsymbol{y}\right)$ using a conditional denoising autoencoder, as proposed in [Saharia et al., 2021b, Saharia et al., 2021a], but this approach lacks flexibility as the conditional denoising autoencoder is task dependent. A more flexible strategy is to exploit the Bayes' formula to decompose the conditional score as:

$$\nabla_{\boldsymbol{x}_t} \log p\left(\boldsymbol{x}_t|\boldsymbol{y}\right) = \nabla_{\boldsymbol{x}_t} \log p\left(\boldsymbol{y}|\boldsymbol{x}_t\right) + \nabla_{\boldsymbol{x}_t} \log p\left(\boldsymbol{x}_t\right). \tag{2.31}$$

Here, we can reuse the score-model of an unconditional denoising diffusion model to approximate $\nabla_{\boldsymbol{x}_t} \log p\left(\boldsymbol{x}_t\right)$. However, evaluating $\nabla_{\boldsymbol{x}_t} \log p\left(\boldsymbol{y}|\boldsymbol{x}_t\right)$ is not feasible, because the likelihood

$$p\left(\boldsymbol{y}|\boldsymbol{x}_t\right) = \int p(\boldsymbol{y}|\boldsymbol{x}_0)p(\boldsymbol{x}_0|\boldsymbol{x}_t)\mathrm{d}\boldsymbol{x}_0 \tag{2.32}$$

is intractable. Several works propose to approximate the likelihood score by exploiting diverse heuristics [Chung et al., 2023, Chung et al., 2022, Kawar et al., 2022a, Song et al., 2023]. For instance, in *Denoising diffusion posterior sampling* (DPS) [Chung et al., 2023], the following approximation is used:

$$p\left(\boldsymbol{y}|\boldsymbol{x}_t\right) \approx \mathcal{N}(\boldsymbol{y}; A\hat{\boldsymbol{x}}_{0:t}, \sigma^2 I), \tag{2.33}$$

where

$$\hat{\boldsymbol{x}}_{0:t} = \frac{1}{\alpha_t}\left(\boldsymbol{x}_t + \sigma_t^2 s_\theta(\boldsymbol{x}_t, t)\right) \tag{2.34}$$

$$\tag{2.35}$$

is the approximation of the posterior mean $\mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\boldsymbol{x}_0]$ derived using Tweedie formula (2.18).

2.4 Challenges

As mentioned in the previous section, several works have highlighted the potential of using deep generative models as prior to solve image restoration problems. However, a lot of challenges remain to be solved, as we discuss below.

2.4.1 Deep learning prior in low-data regime

End-to-end restoration networks require a large training dataset to perform well. For instance the super-resolution end-to-end network in [Wang et al., 2020] is trained on a dataset containing more than 10000 images. Likewise, deep generative models also need a large training dataset of clean images. For instance, datasets commonly used in the deep generative models literature, such as CelebA [Liu et al., 2018], FFHQ [Karras et al., 2019] or Cifar10 [Krizhevsky et al., 2009], respectively contain 200K, 70K and 60K images. In some imaging applications, a large dataset of paired (clean-degraded) images is not available. We can distinguish different settings, including having access to a dataset of clean images and a dataset of degraded images, without pairs, or having only access to clean example data. Furthermore, we might have to deal with datasets of limited size, making it difficult to train a deep generative model. Consequently, an important challenge to address is, **how can we learn effective deep learning based prior model under restricted data availability?** In chapter 3, we provide an effective solution to this problem.

2.4.2 Convergence guarantee with generative regularization

Deep learning based regularization methods have been shown to outperform classical convex regularizer [Meinhardt et al., 2017, Zhang et al., 2021]. However, the gain in performance comes at the cost of increased complexity. Indeed, when using neural network based regularization, the variational problem (3.2) is no longer convex, making it difficult to derive optimization algorithm that provably converge to a local minima of the variational problem (3.2). In practice, this implies that the iterations of an optimization process might diverge, inducing the need for early-stopping heuristics. For plug-and-play regularization, convergence to a fixed-point can be enforced by constraining the Lipschitz constant of the denoising autoencoder [Ryu et al., 2019]. On the other hand, when using deep generative models as regularizers, convergence guarantees remain to be found in a generic setting. In practice, state-of-the art generative regularization methods for inverse problems rely on a large-number of hyper-parameters that need to be tuned empirically to produce good results [Menon et al., 2020, Daras et al., 2021]. Hence, a key question is: **can we derive**

an algorithm that provably converge to a (local) minima of the variational problem under practical assumptions? We will adress this question in chapter 5, see section 2.5 below.

2.4.3 Efficient posterior sampling

Current sampling methods typically rely on iterative sampling algorithms such as Langevin dynamic [Laumont et al., 2022, Coeurdoux et al., 2023], or reverse diffusion process [Chung et al., 2023]. When combined with deep learning based priors, those methods become very expensive in terms of computation costs, as we have to call one network function evaluation at every iteration. For instance, the diffusion posterior sampling methods of [Chung et al., 2023] requires 1000 network evaluations and gradient evaluations to produce one sample from the posterior. As such, it is valuable to investigate **how to design a fast posterior sampling strategy while exploiting deep generative priors?** In chapter 6, we will address this question, for the specific problem of image super-resolution, with hierarchical VAE priors.

2.5 Contributions and outline

Chapter 3 In **chapter 3**, we investigate the problem of training a neural network as a regularizer when training data are limited. In particular, we focus on the setting where unpaired datasets of clean and degraded images are available. In order to reduce the data requirements, we propose to train regularizer on small images patches, and we implement the regularizer as a fully convolutional neural network to make the computation of the regularizer value and its gradient efficient. Inspired by the recent literature on generative adversarial networks, we train our regularization network as an adversarial critic, which enables us to exploit the two unpaired training distributions. Finally, we demonstrate the effectiveness of our method on denoising and deblurring applications.

Chapter 4 Despite its effectiveness, the adversarial training framework is not related to a probabilistic model of the prior. This can be a limitation, in particular for strongly ill-posed problems, such as inpainting or super-resolution. Therefore, we investigate the use of variational autoencoders, a class of image generative models that can provide a strong prior model over images. **Chapter 4** presents an in-depth introduction of the variational autoencoder. It serves as a preliminary for the remaining chapters, in which we present different ways of using VAE models to solve image inverse problems. We detail the VAE training criterion, which enables to jointly train the generative model with an associated inference model (a.k.a. encoder). We also present the different types of deep latent variable models that can be implemented within the VAE framework, with a specific focus on hierarchical VAE models. In particular, we study VDVAE, a hierarchical VAE

model that provides state-of-the art results on image generation benchmarks, and we discuss the properties of its latent representation trough visualization experiments.

Chapter 5 In **chapter 5**, we consider the problem of developing an algorithm for solving generic image inverse problems with deep generative prior that provides convergence guarantee under practical assumptions. To that purpose, we exploit a hierarchical VAE model, and we adopt an alternate optimization scheme that jointly optimizes the image and its associated latent variable. We show that we can derive an efficient strategy to exploit the hierarchical encoder to avoid using backpropagation through the generative model. Our work, inspired by the recently introduced JPMAP framework makes four novel contributions. First, we introduce a strategy to control the strength of the regularization by controlling the temperature of the Gaussian priors over the latent variables. Second, we propose a "greedy" optimization scheme to optimize the hierarchical latent variable efficiently by exploiting the top-down structure of the inference network. Third, we draw a connection with plug-and-play algorithms based on deep image denoiser, by showing that our algorithm can be formulated as a plug-and-play half-quadratic-splitting scheme where the denoising operation is replaced by the reconstruction with the hierarchical VAE. This connection enables us to prove convergence to a fixed-point under given conditions, and to characterize the property of the fixed point. Fourth, we introduce a new fully-convolutional hierarchical VAE model, patchVDVAE, that can be applied on images of any resolution. We demonstrate the effectiveness of our method on inpainting, super-resolution and deblurring problems on two datasets, namely face images from CelebA dataset and on natural images.

Chapter 6 In **chapter 6**, we tackle the problem of producing samples from the posterior distribution of the inverse problem $p(\boldsymbol{x}|\boldsymbol{y})$ for image super-resolution problems. We develop a method to exploit a powerful hierarchical VAE as a prior model over the high-resolution images. We show that we can repurpose the weights of the HVAE generative model to implement a diverse super-resolution network, that can produce samples from the posterior with only one network evaluation. To do so, we introduce an encoder on low-resolution images, and we train it to match the HVAE encoder model on the associated high-resolution images. At inference time, our super-resolution network is defined as the combination the low-resolution encoder and the high-resolution generative model given by the pretrained hierarchical VAE. Furthermore, we also demonstrate that the hierarchical representation learned by HVAE models separates the high frequency information from the image low-frequency information, which enable us to train the low-resolution encoder more efficiently by training it to only predict the part of the hierarchical latent representation that effectively encodes the low-resolution information. We validate the ability of our method to generate diverse solutions to the super-resolution problem on face super-resolution with upsampling factors $\times 4$, $\times 8$.

2.5.1 Publications and preprints

- Jean Prost, Antoine Houdard, Andrés Almansa, Nicolas Papadakis. **Learning local regularization for variational image restoration**. Scale Space and Variational Methods in Computer Vision: 8th International Conference, (SSVM 2021), Virtual Event, May 16–20, 2021, Proceedings.https://hal.science/hal-03139784v1

- Jean Prost, Antoine Houdard, Andrés Almansa, Nicolas Papadakis. **Apprentissage d'une fonction de régularisation locale pour la restauration d'images**. Congrès des jeunes chercheurs en vision par ordinateur (ORASIS'21). https://hal.science/hal-03339625/

- Jean Prost, Antoine Houdard, Andrés Almansa, Nicolas Papadakis. **Diverse super-resolution with pretrained deep hierarchical VAEs**. 2022. (preprint) https://doi.org/10.48550/arXiv.2205.10347

- Jean Prost, Antoine Houdard, Andrés Almansa, Nicolas Papadakis. **Inverse problem regularization with hierarchical variational autoencoders**. To appears in International Conference on Computer Vision 2023 (ICCV2023). https://arxiv.org/abs/2303.11217

- Lihao Liu, Jean Prost, Lei Zhu, Nicolas Papadakis, Pietro Lio, Carola-Bibiane Schönlieb, Angelica Aviles-Rivero. **SCOTCH and SODA: A Transformer Video Shadow Detection Framework**. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (CVPR2023). https://openaccess.thecvf.com/content/CVPR2023/html/Liu_SCOTCH_and_SODA_A_Transformer_Video_Shadow_Detection_Framework_CVPR_2023_paper.html

Chapter 3

Adversarial local regularization for variational image restoration

In this chapter, we present a framework to train a neural network as an explicit regularization function for solving general image restoration problems. Specifically, we define our regularizer as a fully convolutional neural network that sees the image through a receptive field corresponding to small image patches. Following the adversarial regularization framework, we train our network as a classifier discriminating clean and degraded patches. This yields a regularization function that can be incorporated in any image restoration problem. Our approach is data efficient due to the low-dimensionality of patches, and it does not need paired training data thanks to adversarial training. We demonstrate the efficiency of the framework on denoising and deblurring applications.

3.1 Introduction

Inverse problems and convex regularization. Many image restoration tasks require to solve an inverse problem. This can be addressed with a variational formulation involving a data-fidelity term and a regularization term encouraging the solution to satisfy given properties or to belong to a space of possible solutions. Some of the most famous regularization terms used for image restoration are convex non-smooth terms like the total variation [Rudin et al., 1992b], or ℓ^1 minimization of transform-domain coefficients such as Wavelet frames [Donoho and Johnstone, 1994, Coifman and Donoho, 1995] or local Fourier or DCT representations [Yu and Sapiro, 2011]. However, these strategies tend to produce over-smoothed or suboptimal results, since they represent only a rough approximation of natural image statistics and geometry.

CNN-based non-convex regularization. Later-on more accurate natural image priors emerged in the form of non-convex regularization terms, such as patch-based Gaussian

mixture models (to be discussed below) or convolutional neural networks (CNN). Most common CNN-based regularizers are, however, trained in a way that the prior or regularizer itself is only partially and implicitly known via its gradient [Bigdeli et al., 2017, Romano et al., 2017b, Reehorst and Schniter, 2019] or proximal operator [Venkatakrishnan et al., 2013, Meinhardt et al., 2017, Zhang et al., 2017b, Kamilov et al., 2017, Ryu et al., 2019]. Such implicit CNN regularizers, and the associated optimization algorithms, lack convergence guarantees or do so under overly restrictive conditions on the regularizer, the regularization parameter or the kind of inverse problems they can solve [Reehorst and Schniter, 2019, Ryu et al., 2019].

To overcome these limitations a new breed of explicit CNN-based regularizers have been proposed, either in the form of the push-forward measure of a generative model [Bora et al., 2017], a variational autoencoder [González et al., 2022], or more directly as a discriminator network [Lunz et al., 2018]. All these approaches are nevertheless limited to a particular class of image and do not generalize to images of arbitrary size.

Patch-based non-convex regularization. Learning prior information has also been widely studied from the patch point-of-view. The main idea is to learn the prior knowledge from patches, that are local sub-images of small size, instead of learning a prior from whole images. This allows to avoid the high-dimensional issues faced when working with full-size image distributions. These approaches rely on parametric models of the patch distribution such as Gaussian mixture models [Zoran and Weiss, 2011b, Houdard et al., 2018, Teodoro et al., 2018]. However, such simple models can not accurately represent the complexity of the patch space.

In this work, we introduce an explicit non-convex regularization function encoded with a fully convolutional neural network that acts as a local regularizer. This prior knowledge on the patch distribution can be applied to a whole image without size limitation. We propose (i) to learn the convolutional regularizer as a discriminator between patches using the Wasserstein GAN framework [Arjovsky et al., 2017] as in the adversarial regularization framework [Lunz et al., 2018], and (ii) to integrate this regularizer in patch-based models such as the expected patch log-likelihood framework (EPLL) [Zoran and Weiss, 2011b].

3.1.1 Setup of the problem

Linear inverse problem The main goal of this chapter is to perform image restoration by solving an inverse problem. That is, finding the underlying true image $x^\star$ from its perturbed observation y that we consider here to be of the form

$$\boldsymbol{y} = A\boldsymbol{x}^\star + \epsilon, \tag{3.1}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian white noise and A is a degradation operator that can typically be the identity (pure denoising), a mask (inpainting) or a blurring kernel

(deconvolution). These inverse problems can be addressed with a variational formulation involving a regularization term. This amounts to finding an estimate $\hat{x}$ of $x^\star$ of the form:

$$\hat{\boldsymbol{x}} \in \arg\min_{\boldsymbol{x}} \frac{1}{2\sigma^2}\|A\boldsymbol{x} - \boldsymbol{y}\|^2 + \lambda g(\boldsymbol{x}), \tag{3.2}$$

where $\|A\boldsymbol{x} - \boldsymbol{y}\|^2$ is the data-fidelity term ensuring that the recovered image $\hat{x}$ is close enough to the degraded observation $\boldsymbol{y}$, $g(\boldsymbol{x})$ is the regularization term and $\lambda \geq 0$ monitors the influence of both terms. In the case where $g(\boldsymbol{x}) = -\log(P_X(\boldsymbol{x})) + C$ is derived from a prior probability distribution P_X modeling the data $\boldsymbol{x}$, then the estimated $\hat{\boldsymbol{x}}$ from (3.2) corresponds to the *maximum a posteriori* estimator.

Patch based regularization The choice of the regularization function R has a strong impact on the final result. We propose to learn R through a local regularization functional r acting on patches. Denoting as $\Omega_x = \{x_1, \cdots, x_n\}$ the set of all patches of size $p \times p$ from an image x, this function takes as input an image patch x_i and outputs a score $r(x_i)$ that indicates how likely the patch is to be a clean one. As in EPLL [Zoran and Weiss, 2011b], we define the global regularization functional as the average value of the local scores on the set of all patches of image x:

$$g(\boldsymbol{x}) = \frac{1}{|\Omega_x|}\sum_{x_i \in \Omega_x} r(x_i). \tag{3.3}$$

Working with patches yields three main advantages. It first makes the learning phase simpler, as a patch model contains far less parameters than a full image model. Next, the number of images required for training is reduced, as a single image already provides several thousands of patches. Finally, unlike regularization methods employing networks with fixed input resolution, our regularization network can be applied on images of any size.

Convolutional patch regularizer In practice, we consider r as a CNN with receptive field size equal to the patch size $p \times p$ and taking values in $\mathbb{R}$. This representation is more general than Gaussian mixture models, and allows encoding complex distributions.

3.1.2 Contributions and outline

We propose an image restoration method that relies on a regularization function learned on patches and applied to any image size. It gathers the advantages of previous CNN methods while avoiding the constraints of implicit plug & play priors (convergence guarantee) and of GAN or VAE priors (fixed image size).

In addition, the regularization function is learned in an *unsupervised* manner, in the sense that it only relies on patch distributions of clean and degraded data, and **it does**

not require paired data. We can therefore deal with an unknown degradation model if a noisy dataset is available.

The organization of this chapter is as follows. In Section 3.2, we propose an unsupervised framework for the learning of a compact convolutional neural network modeling the local patch regularity prior. We namely obtain the local regularization functional r as a critic trained to distinguish noisy patches from clean ones using the framework of Wasserstein generative adversarial models [Arjovsky et al., 2017]. In section 3.3, we provide implementation details to make the work fully reproducible. We show in section 3.4 that the local functional r generalizes well to arbitrary levels of noise, i.e. noise level unseen during training. In Section 3.5, we demonstrate that the proposed framework is efficient for image denoising and deblurring.

3.2 Local regularization for image inverse problem

In this section we define our local image regularizer r_θ as a convolutional neural network, and we describe how to use and train it.

Patch-based methods have shown to be efficient tools for solving inverse problems in imaging [Zoran and Weiss, 2011b]. Hence we aim at defining a regularization function r_θ depending on parameters $\theta \in \Theta$ that encodes prior knowledge at a patch level. In the patch-based literature, such regularizers rely on statistical modeling of the distribution of clean patches and the model parameters are usually inferred with a maximum likelihood estimation [Houdard et al., 2018]. This leads to two main limitations. First, it requires to have access to the probability density function of the prior distribution and consequently it does not properly represent the intrinsic low dimensional manifold of clean patches. Second, maximizing the likelihood of a complex model leads to non-convex problems that are difficult to solve in practice.

In order to tackle these issues, we propose to take advantage of having two datasets of clean and degraded patches –not necessarily paired– and consider r_θ as a *critic* that tells us if a patch is more likely to be clean or degraded.

We first detail in section 3.2.1 how the local regularization function is integrated as a global regularizer on images in order to solve the variational problem (3.2). In section 3.2.2, we present the framework to learn the regularizer as a *critic* between two unpaired datasets of clean and degraded images.

3.2.1 Convolutional regularizers for variational problems

Convolutional regularizer We define, for the variational problem (3.2), a regularization term g that takes into account local prior knowledge of the images. To do so, we propose to consider a class of functions $r_\theta(\boldsymbol{x})$ defined with a fully convolutional neural network with parameter $\theta \in \Theta$ and differentiable with respect to $\boldsymbol{x}$. We enforce the perceptual

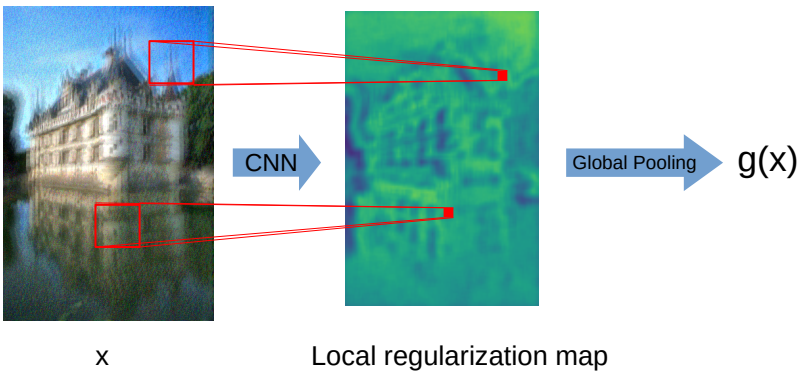


Figure 3.1 – The local regularizer function is implemented as a convolutional neural network. When applying it to full images, it outputs a regularization map, where each "pixel" value corresponds to the value of the regularizer for the corresponding patch. By averaging the values of the output map, we get the value of the regularization on the full image.

size of this network to be the patch size $p \times p$. That is, the successive convolutions operate on a window no larger than $p \times p$ pixels. Using this architecture permits to compute the global regularizer R from (3.3) by directly applying r_θ to the full image x and average the outputs, as illustrated in Figure 3.1. Once learned the local regularizer $r_\theta^\star$, the variational problem to solve becomes

$$\min_{\boldsymbol{x}} \frac{1}{2\sigma^2}\|A\boldsymbol{x} - \boldsymbol{y}\|^2 + \frac{\lambda}{|\Omega_x|}\sum_i r_\theta^\star(x_i). \tag{3.4}$$

Gradient descent optimization We propose to find a local minimizer of (3.4) by performing an explicit gradient descent method. Let x^ℓ the image at iteration ℓ, a gradient step of step size η writes

$$\boldsymbol{x}^{\ell+1} = \boldsymbol{x}^\ell - \frac{\eta}{\sigma^2}A^*(A\boldsymbol{x}^\ell - y) - \frac{\eta\lambda}{|\Omega_x|}\sum_i \nabla r_\theta^\star(\boldsymbol{x}_i^\ell), \tag{3.5}$$

where A^* is the adjoint operator of A. Contrary to plug & play methods that rely on implicit schemes [Venkatakrishnan et al., 2013], this explicit scheme converges for differentiable regularization functions and adequate time steps.

We now describe how the framework for learning the local regularization function.

3.2.2 Adversarial Local Regularizer (ALR)

Adversarial training In order to train r_θ as a critic between patch distributions, we consider the discriminator framework introduced for generative adversarial networks [Good-

fellow et al., 2014b], without the generator network. Such approach nevertheless results in a critic r_θ approximating the hard clustering between clean and degraded patches. It therefore induces steep gradients ∇r_θ that may lead to numerical instabilities during the minimization of problem (3.2).

Wasserstein adversarial loss As a consequence, we rather rely on the Wasserstein GAN [Arjovsky et al., 2017] formulation that amounts to approximate the optimal transport cost between the distribution of clean patches $\mathbb{P}_c$, and a distribution of degraded patches $\mathbb{P}_n$. Relying on the dual formulation of the optimal transport [Santambrogio, 2015], an optimal critic $r_\theta^\star$ is seen as a Kantorovitch potential and shall satisfy

$$r_\theta^\star \in \underset{\varphi \in \mathrm{Lip}_1}{\arg\max}\, \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_n}\left[\varphi(\boldsymbol{z})\right] - \mathbb{E}_{z\sim\mathbb{P}_c}\left[\varphi(\boldsymbol{z})\right]. \tag{3.6}$$

Under the assumption that the support of the clean patches distribution $\mathcal{M}$ is compact [Lunz et al., 2018], the solution of equation (3.6) corresponds to the distance function to the clean data manifold $\mathcal{M}$. Each iteration of the gradient descent on equation (3.2) thus brings our noisy data closer to the clean data.

Gradient penalty In practice, imposing a neural network to be 1-Lipschitz is a difficult task and we therefore use the gradient penalty introduced in [Gulrajani et al., 2017] to encourage the gradient norm to be close to 1. This amounts to maximize the following quantity

$$D(\theta) = \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_n}\left[r_\theta(\boldsymbol{z})\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_c}\left[r_\theta(\boldsymbol{z})\right] - \mu\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_i}[(||\nabla_{\boldsymbol{z}} r_\theta(\boldsymbol{z})||_2 - 1)^2] \tag{3.7}$$

where $\mathbb{P}_i$ is the distribution of all lines connecting samples in $\mathbb{P}_n$ and $\mathbb{P}_c$. In other words, the last term of (3.7) penalizes regularizers having gradient of norm different from one on the convex hull of the union of the support of $\mathbb{P}_c$ and $\mathbb{P}_n$. By enforcing the gradient ∇r_θ to be of norm close to 1, vanishing gradient issues are also avoided when solving problem (3.2) with gradient descent approaches. We illustrate the properties of the regularization functional with a synthetic example in Figure 3.2 containing random perturbations of clean data points located on a circle. The learned regularization function $r_\theta(z)$ therefore approximates the distance function to the circle. The gradient $\nabla r_\theta(z)$ thus indicates the direction to follow in order to transport z towards a clean point within the circle.

3.3 Practical considerations for image restoration

In this section, we provide implementation details to reproduce the proposed framework. After presenting the architecture of the regularization network r_θ, we explain the training strategy and describe how image restoration is performed.

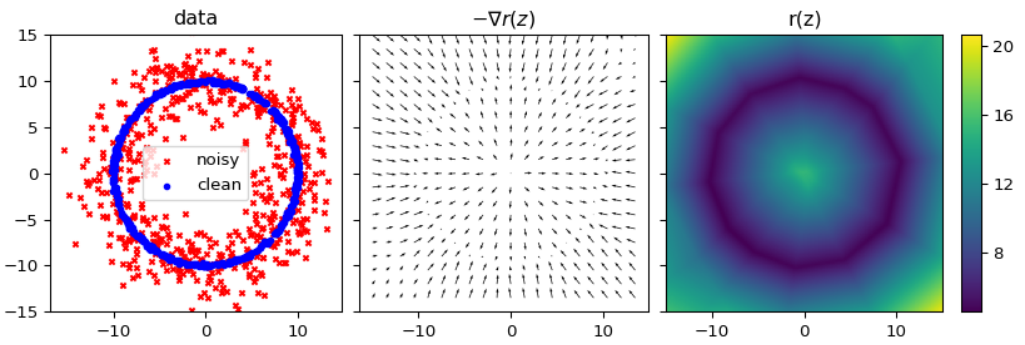


Figure 3.2 – Regularization functional $r(z)$ learned on a synthetic 2d denoising problem with clean data (blue dots) on the circle and noisy ones (red crosses). The gradient penalty ensures that the gradient ∇r is not flat close to the data manifold $\mathcal{M}$.

3.3.1 Network architecture

The local regularization functional r_θ is designed as a 6 layers convolutional network. Each layer is made of 3×3 convolution operations followed by ReLU activations [Nair and Hinton, 2010]. This network has therefore a 15×15 receptive field. No padding is used. Hence, when a patch of the size of the network receptive field is fed to the network r_θ, the output is a scalar.

3.3.2 Training the regularization functional

Training details The proposed regularization network is trained with patches matching the size of the receptive field of the network. We create the dataset $\mathcal{D}_c$ of clean patches by extracting all 15×15 patches from a 30000 image subset of the google landmarks dataset [Weyand et al., 2020]. Similarly, we create the dataset $\mathcal{D}_n$ of noisy patches by extracting all 15×15 patches from another 30000 images subset of the landmarks dataset, to which we added an additive white Gaussian noise with standard deviation σ_{train}. Following [Lunz et al., 2018], the local regularization network r_θ is trained to minimize the criterion (3.7) with Algorithm 1. We use the Adam optimizer [Kingma and Ba, 2015] with hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and an exponential learning rate decay, so that the learning rate α begins at a value of 10^{-3} for the first iteration, and ends up at 10^{-4} for the last iteration. We use a batch size of $m = 32$ and train the network for $K = 10^5$ iterations. The gradient-penalty parameter is set to $\mu = 5$.

Analysis of the local regularizer Training samples of clean and noisy patches z, with their final regularizer value $r_\theta(z) \in \mathbb{R}$, are shown in Figure 3.3. As can be observed from the functional values, there exists a slight ambiguity between texture patches ($r_\theta(z) = -0.23$

Algorithm 1 Learning the local regularization r_θ

Input: Datasets $\mathcal{D}_c$ of clean patches and $\mathcal{D}_n$ of noisy patches; gradient penalty μ; batch size m, number of iterations K
Output: regularization function r_θ
for $k = 1$ **to** K **do**
  Sample minibatches of m clean patches $\{z_j^c\}_{j=1}^m$ from $\mathcal{D}_c$ and m noisy patches $\{z_j^n\}_{j=1}^m$ from $\mathcal{D}_n$ and a random number $\alpha \in [0; 1]$
  Define interpolated patches $z_j^i = \alpha z_j^c + (1-\alpha) z_j^n$ for $j = 1, \cdots, m$
  for $j = 1$ **to** m **do**
    $D_j(\theta) = r_\theta(z_j^n) - r_\theta(z_j^c) - \mu(||\nabla_z r_\theta(z_j^i)||_2 - 1)^2$
  end for
  $\theta \leftarrow Adam(\nabla_\theta \sum_{j=1}^m D_j(\theta))$
end for

for the last patch of top row) and noisy homogeneous patches ($r_\theta(z) = -0.27$ for the first patches of bottom row). We nevertheless show in Figure 3.4 that the distributions of clean and noisy patches are globally well separated, as the regularizer $r_\theta(z) \in \mathbb{R}$ assigns a lower value on clean patches than on noisy patches, except for some textured patches, for which the regularizer assign a value that is similar to the value assigned to some noisy patches (see the patch in the middle in Figure 3.4).

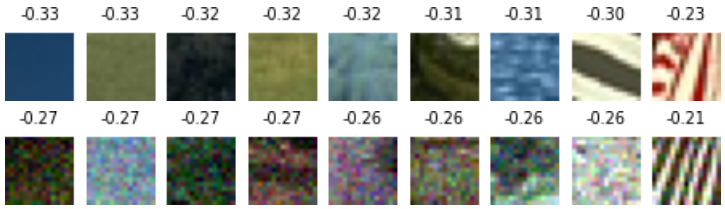


Figure 3.3 – Value of the local regularization functional r_θ trained with $\sigma_{train} = 0.1$ on clean (top row) and noisy (bottom row) patches ($\sigma = 0.1$).

3.3.3 Solving the variational problem

Image restoration is realized by solving the variational problem (3.4). To do so, we search for the minimizing image $\boldsymbol{x}$ by performing 50 iterations of Adam [Kingma and Ba, 2015], with the momentum parameter set to the default values $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and an exponential learning rate decay, with an initial learning rate of 0.1 and a final learning rate of 0.01 at the last iteration. We implement the method with the pytorch deep learning framework, so that the gradient of the global regularization functional $g(\boldsymbol{x})$ can be easily computed using automatic differentiation [Paszke et al., 2017].

In preliminary experiments, we also tried to use gradient descent instead of Adam to optimize (3.4). However, we found that the results obtained with gradient descent were significantly worse than the one obtained while using Adam.

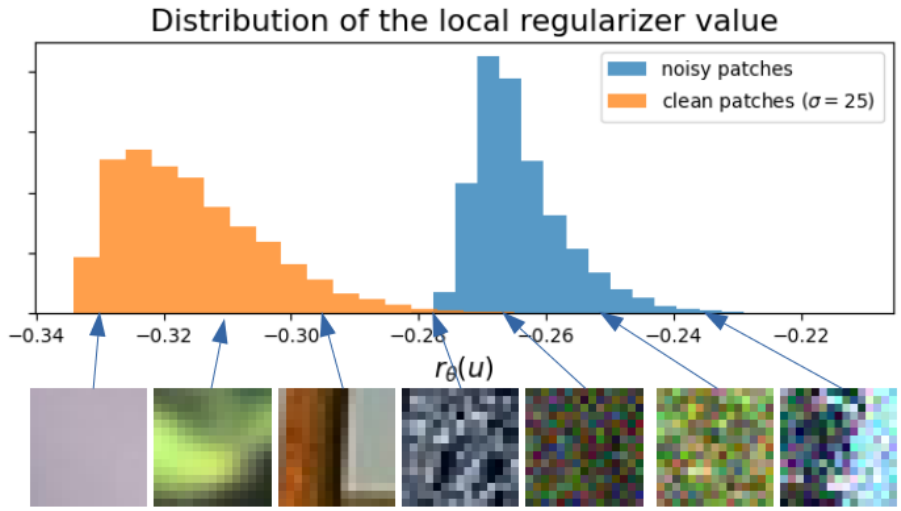


Figure 3.4 – Distribution of the values of the regularizer function on clean and noisy patches (top), and some clean and noisy patches along with their regularizer value (bottom).

3.4 Robustness to noise variations

In this section, we study the robustness of the proposed regularization function to noise variations. The adversarial training of the regularization function, presented in the previous section, requires to learn a different regularization function for every different noise level σ. We show how this limitation can be overcome.

We first analyze the behaviour of regularization functions trained on a single noise level σ_{train} and then used to denoise an image with a different noise level σ_{img}. Second, we propose to train the regularization functions with varying noise levels and demonstrate experimentally the superiority of this approach.

3.4.1 Robustness to unseen noise level

To study the ability of the local regularization function to generalize to noise levels unseen during training, we train 4 regularization functions on 4 different noise levels $\sigma_{train} \in \{0.05, 0.1, 0.2, 0.4\}$. We then evaluate the quality of the regularization of those networks on denoising tasks, for 5 different noise levels $\sigma_{img} \in \{0.05, 0.1, 0.2, 0.3, 0.4\}$. The 4 networks share the same architecture and the same training procedure as described in section 3.3.

Distribution of the regularizer values While these regularizers have only been trained to distinguish between clean patches and noisy patches for a particular noise level,

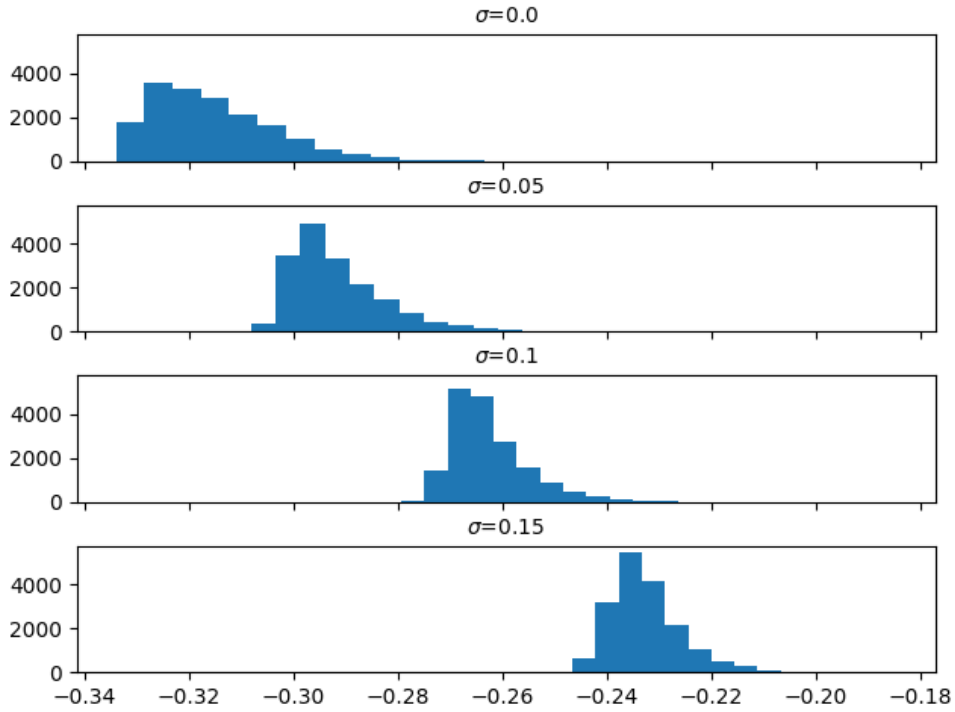


Figure 3.5 – Distribution of values $r_\theta(x)$ for a regularizer trained on noise level $\sigma_{train} = 0.1$. It generalizes well to patches x with an intermediate (0.05) or extrapolated (0.15) noise levels.

they generalize well to intermediate noise levels, in the sense that the regularizer value is an increasing function of the noise level of its input patch. Figure 3.5 illustrates this point for the noise level $\sigma_{train} = 0.1$. The overlap between the distribution for noise level $\sigma = 0$ (top) and $\sigma = 0.1$ (bottom) is small, showing the ability of the regularizer network to distinguish clean and noisy patches. Furthermore, the distribution for noise level $\sigma = 0.05$ is located in between the distributions $\sigma = 0$ and $\sigma = 0.1$, showing the ability of the network to generalize to intermediate noise levels. The network also extrapolates to larger noise level, as it returns larger values on noise level $\sigma = 0.15$ than for the training noise level $\sigma = 0.1$.

Denoising performance on noise different from training Next, we evaluate denoising quality by measuring the average PSNR on a validation set of 11 images. To that end, we solve problem (3.2) for $A = \mathrm{Id}$. We denoise images with 5 noise levels $\sigma_{img} \in \{0.05, 0.1, 0.2, 0.3, 0.4\}$, and we respectively set the regularization parameter λ to $\{0.15, 0.35, 0.6, 0.8, 1\}$. The results displayed on Table 3.1 demonstrate that the trained regularization functions generalize well to unseen noise level, as for all 5 levels of noise

Table 3.1 – Average PSNR on AWGN denoising, in function of the image noise level σ_{img}, and the noise level the regularization network was trained on σ_{train}. For each image noise level, best result is displayed in bold, and second best result is underlined. Regularization networks trained on small noise level σ_{train} generalizes well to higher noise levels σ_{img}. The regularization network trained on varying level of noise $\sigma_{train} \in [0.05; 0.3]$ performs better on high noise levels σ_{img}.

σ_{train} \ σ_{img}	0.05	0.1	0.2	0.3	0.4
0.05	**33.24**	**28.94**	<u>24.21</u>	<u>21.25</u>	18.91
0.1	<u>33.20</u>	28.82	24.17	<u>21.25</u>	<u>19.13</u>
0.2	32.42	28.23	23.80	21.03	18.96
0.4	33.01	28.23	23.84	21.03	18.96
$[0.05; 0.3]$	32.92	<u>28.90</u>	**24.91**	**22.58**	**20.84**

σ_{img}, the 4 regularization functions yield average PSNR values that are contained in an interval of size smaller than 1 dB. Furthermore, regularization networks trained on small noise levels $\sigma_{train} \in \{0.05, 0.1\}$ generalize well to higher noise levels σ_{train} as they perform even better than networks trained on the specific noise level.

Analysis We suggest that this is due to the fact that, when trained on a small noise level, the regularization function is forced to learn a tight boundary between the clean and the noisy distribution which favors denoising performance. However, for the highest noise level $\sigma_{img} = 0.4$, the regularization function trained on a small noise level $\sigma_{train} = 0.05$ gives the worst results. As patches with very high noise levels are not seen during the training of the regularization function trained for $\sigma_{train} = 0.05$, we suggest that the gradient penalty is not enforced to 1 in this region of the patch space. Thus there is no guarantee that the gradient of the regularization function ∇r_θ is indeed directed towards the space of possible solutions. This prevents the optimization algorithm from finding a relevant local minimum of (3.2).

3.4.2 Robustness to noise variation during training

Training on different noise levels We now propose to improve the robustness of the regularizer to noise level variation during training. To do so, we train a regularization function on a distribution containing patches with noise level σ_{train} uniformly sampled in the interval $[0.05, 0.30]$. We use the same network architecture and the same training procedure as in section 3.3. We evaluate the effectiveness of this regularization function by measuring the average PSNR when this function is used for denoising. We compare

the performance with the prior trained on a single noise level in the last row of Table 3.1. Results show that the regularization function trained with a varying noise level has comparable performance with the regularization function trained on a single-noise level. Furthermore, for high noise level, the regularization function trained on a varying noise level significantly outperforms the regularization function trained on a single-noise level. This illustrates the fact that training the regularization function on varying noise level is actually beneficial.

Analysis We suggest that exposing the regularization function to various noise levels during training combines two advantages. It first learns a tight boundary around the clean patches distribution, as the networks trained on low noise levels. Second, the gradient-penalty is enforced even on highly noisy patches, as the network is trained on high noise levels.

3.5 Experiments

We evaluate the effectiveness of our learned regularization functional on two image restoration tasks, image denoising and image deblurring.

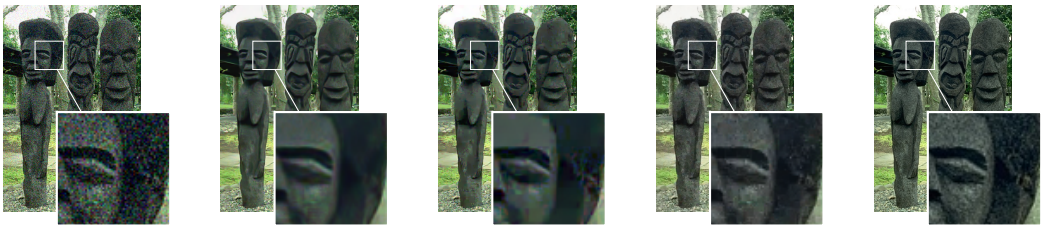
3.5.1 Denoising

Experimental setting We evaluate our method on additive white Gaussian noise denoising, which corresponds to solving (3.2) with $A = I$. We compare our method against two common patch-based denoising algorithms, BM3D [Dabov et al., 2007] and EPLL [Zoran and Weiss, 2011b], on 3 noise levels $\sigma_{img} \in \{0.1, 0.2, 0.4\}$. We use our model trained on varying noise level $\sigma_{train} \in [0.05, 0.3]$, with the regularization parameter λ respectively set to 0.15, 0.35 and 1. For BM3D, we use the implementation of [Lebrun, 2012] with default parameters, and for EPLL we use the implementation of [Hurault et al., 2018] with default parameters and a prior GMM model learned on RGB patches.

Results The average PSNR and LPIPS [Zhang et al., 2018b] on the BSD68 dataset for the 3 methods are presented in Table 3.2, and examples of denoised images are shown on Figure 3.6. For the 3 noise levels, the adversarial local regularization denoising outperform EPLL and BM3D in terms of PSNR, while having comparable perceptual quality. This illustrates the ability of convolutional neural networks to be used as local regularizers when trained the right way.

Table 3.2 – Comparisons in terms of PSNR (left) and LPIPS (right) of the patch-based denoising algorithms ALR, EPLL and BM3D, for white Gaussian noise. Results are averaged on the 68 images of the BSD68 dataset.

	PSNR			LPIPS		
σ	ALR	EPLL	BM3D	ALR	EPLL	BM3D
0.1	**28.85**	28.77	28.26	0.29	**0.28**	0.30
0.2	24.88	**24.92**	24.69	0.44	**0.42**	0.43
0.4	**21.58**	19.75	20.25	**0.57**	0.61	0.58

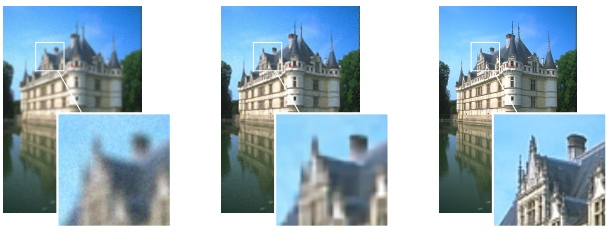


(a) noisy (20.35dB) (b) BM3D (25.48dB) (c) EPLL (25.48 dB) (d) ALR (26.96 dB) (e) clean

Figure 3.6 – Visual comparison of patch-based denoising methods for $\sigma = 0.1$.

3.5.2 Deblurring

To illustrate the adaptability of our local regularization function, we consider image deblurring. This corresponds to solving (3.2) with a linear degradation operator A taken as a convolution operation with a blur kernel k, that is $y = k * x + \varepsilon$. Figure 3.7 shows an example of image deblurring using our learned local regularization function. The image is blurred with a 7×7 Gaussian kernel with standard deviation $\sigma_k = 3$, and an additive white Gaussian noise of standard deviation $\sigma = 0.03$.



(a) blurry (22.17dB) (b) ALR (25.36dB) (c) clean

Figure 3.7 – Illustration of deblurring using a 7×7 Gaussian kernel with standard deviation $sigma_k = 3$.

3.6 Conclusion and Perspectives

We propose a new strategy to solve inverse problem in imaging using a convolutional neural network as a local regularization function. The local regularization network is trained to discriminate between clean and noisy patches, and the global regularization function is defined as the average value of the local function over the set of all image patches. Working with a local regularization function offers several advantages : it works with any image size, it requires less training data and has less parameters than a full size model. Furthermore, the fully convolutional architecture of the network makes it computationally efficient to evaluate the global regularization function and its gradient.

Experimental results on image denoising show that our method outperforms popular patch-based denoising algorithm such as EPLL and BM3D, illustrating the potential of convolutional networks to acts as regularization function for inverse imaging problems.

We believe that improving the training criterion of the regularization function could improve the performance of the regularization. Indeed, the training criterion of our local regularization network corresponds to the 1-Wasserstein distance. The regularizer thus grows linearly with the distance to the clean data manifold, whereas the data-fidelity term is quadratic. We suggest that these unbalanced terms make the variational problem difficult to solve, especially for high noise levels. We postulate that learning a regularization term based on the 2-Wasserstein distance could help to overcome this limitation, as the learned regularization function would then grow with the square of the distance to the clean manifold.

Chapter 4

Variational autoencoders priors

We have previously presented an adversarial strategy to train a neural network as regularization functional in a variational problem. However, the adversarial regularizer is not related to a probabilistic model of the prior. This can be a limitation, in particular for strongly ill-posed problems such as inpainting or super-resolution, which require to recreate the missing information. In order to define a probabilistic model of the prior, we can employ deep generative models. In the remaining of this thesis, we will focus on the use of variational autoencoder for solving image inverse problems. This chapter provides an in-depth introduction of the variational autoencoder. We present the VAE training criterion, and the different class of probabilistic models that can be learned within the VAE framework. In particular, we will focus on hierarchical VAE models, an expressive class of generative models that were shown to perform well on image modelling benchmarks.

4.1 Deep latent variable models

4.1.1 Generative modeling

Generative modeling refers to the task of modeling an unknown data distribution $p_{\text{data}}(\boldsymbol{x})$, given a dataset of independent samples from this distribution, that we will denote as $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{1 \leq i \leq N}$. A common approach is to define a class of parameterized model $\{p_\theta(\boldsymbol{x}); \theta \in \Theta\}$, and to optimize the model parameters to adjust them to the data distribution. For instance, one can compute the maximum likelihood estimator, which writes:

$$\hat{\theta}_{ml} = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta\Big(\boldsymbol{x}^{(i)}\Big) \tag{4.1}$$

For instance, when $p_\theta(\boldsymbol{x})$ is a multivariate Gaussian distribution with parameters $\theta = (\mu, \Sigma)$, and $p_\theta(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}, \mu, \Sigma)$, it can be verified that the learning problem (4.1) has a closed-form

solution. However, for more complex models $p_\theta(\boldsymbol{x})$, the maximum likelihood criterion is intractable and one has to rely on alternate strategies.

4.1.2 Latent variable models

Latent variable assumption Latent variable models provide an efficient way to design expressive generative models $p_\theta(\boldsymbol{x})$, by making the assumption that the observed data samples $\boldsymbol{x}$ depend on some underlying, latent factors, encoded within a latent variable $\boldsymbol{z}$. A latent variable model, is then described by the composition of a "prior" on the latent variable $p_\theta(\boldsymbol{z})$, and a likelihood model $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ which describes the probability of observing $\boldsymbol{x}$ knowing the latent variable $\boldsymbol{z}$. Then, the generative process of the data can be reproduced by sequentially sampling $\boldsymbol{z}$, and $\boldsymbol{x}$ conditioned on $\boldsymbol{z}$:

$$\boldsymbol{z} \sim p_\theta(\boldsymbol{z}) \tag{4.2}$$
$$\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z}). \tag{4.3}$$

By definition, the observed variable model $p_\theta(\boldsymbol{x})$ is the marginal[1]

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{z}. \tag{4.4}$$

In the literature, it is also common to refer to the latent variable as a latent code, and to $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ as a decoder, since it maps the latent variable to (a distribution over) the observed variable. We denote both $p_\theta(\boldsymbol{z})$ and $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ to underline the fact they correspond to a joint model $p_\theta(\boldsymbol{z}, \boldsymbol{x}) = p_\theta(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z})$, although it should be understood that the parameters of $p_\theta(\boldsymbol{z})$ and $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ live on disjoint subsets of Θ, and, in some case, the prior $p_\theta(\boldsymbol{z})$ does not actually depend on learnable parameters.

Deep latent variable model To design expressive latent variable models, one can implement the decoder $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ with neural networks, as proposed in [Kingma and Welling, 2013], with the following deep latent variable model:

$$\begin{cases} p_\theta(\boldsymbol{z}) & = \mathcal{N}(\boldsymbol{z}; 0, I) \\ p_\theta(\boldsymbol{x}|\boldsymbol{z}) & = \mathcal{N}(\boldsymbol{x}; \mu_\theta(\boldsymbol{z}), \Sigma_\theta(\boldsymbol{z})) \,. \end{cases} \tag{4.5}$$

In (4.5), $\boldsymbol{z} \in \mathbb{R}^d$ is a continuous variable (usually $d < n$), and the mean and covariance matrix are computed by two neural networks $\mu_\theta(.) : \mathbb{R}^d \to \mathbb{R}^n$ and $\Sigma_\theta(.) : \mathbb{R}^d \to \mathbb{R}^{n \times n}$ parameterized by (a subset of) θ. In practice, it is common to make the two networks $\mu_\theta(.)$ and $\Sigma_\theta(.)$ to share parameters, as displayed in Figure 4.1. Notice that, in this case, the prior $p_\theta(\boldsymbol{z})$ does not have any learnable parameters.

[1] in this discussion, we assume that $\boldsymbol{z}$ is continuous, although it could also be discrete. Then integrals would be replaced by summations.

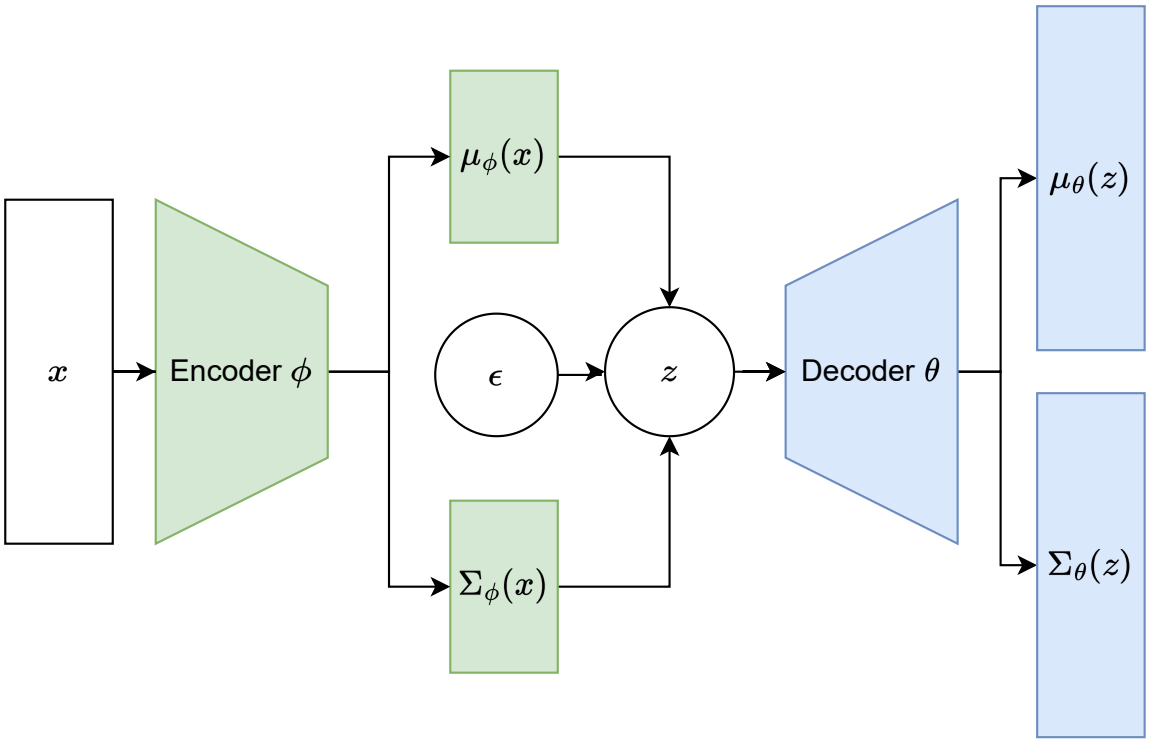


Figure 4.1 – Architecture of a classical variational autoencoder, with a Gaussian encoder and a Gaussian decoder. The encoder network (in green) predict the mean and covariance of the stochastic encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. The decoder network, in blue, takes as an input a sample $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})$ and outputs the statistics of stochastic decoder $p_\theta(\boldsymbol{z}|\boldsymbol{x})$.

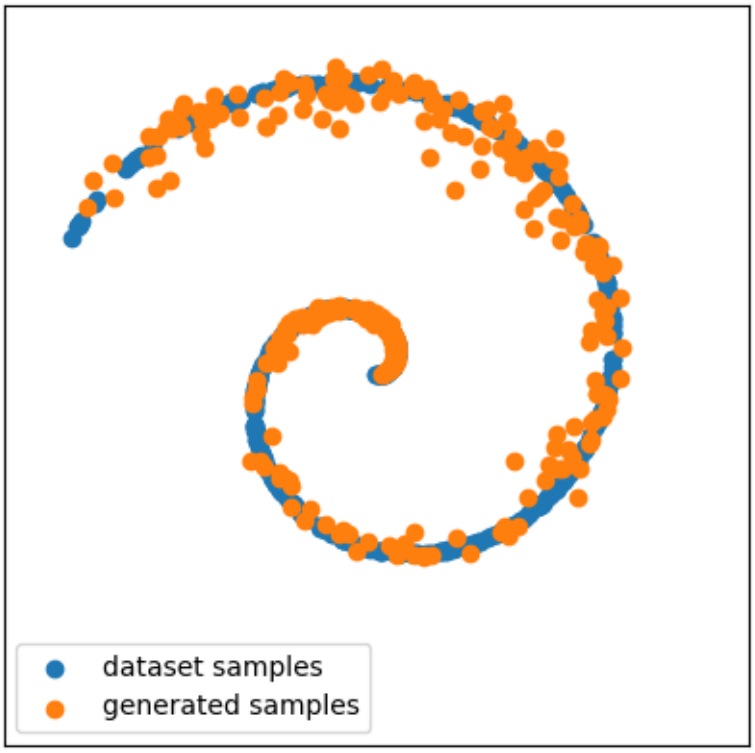


Figure 4.2 – Example of samples produced by a VAE trained on the 2-dimensional spiral dataset. The VAE can fit the manifold of the training data distribution.

Connections with ICA and PCA By imposing an isotropic Gaussian distribution for the prior $p_\theta(\boldsymbol{z})$, we make the assumption that there exists a set of independent generative factors for the observed data. In this sense, this model can be viewed as a way to extend the linear Principal Component Analysis (PCA) [Rolinek et al., 2019, Dai et al., 2017] and Independant Component Analysis (ICA) [Khemakhem et al., 2020] to the case were $\boldsymbol{z}$ and $\boldsymbol{x}$ are related by a non-linear transformation. Under a geometric perspective, the latent variable $\boldsymbol{z}$ can also be interpreted as the coordinates of a point $\boldsymbol{x}$ on the manifold defined as the image of the generative network $\mu_\theta(.)$ [Dai and Wipf, 2018, Chadebec and Allassonnière, 2022] as illustrated in Figure 4.2.

4.2 Variational autoencoder

Latent variable models training Using the maximum likelihood criterion (4.1) for training is impractical for latent variable models because of the intractability of the marginal (4.4). When the posterior of the latent variable model $p_\theta(\boldsymbol{z}|\boldsymbol{x})$[2] can be computed efficiently, as it is the case for Gaussian mixture models [Reynolds et al., 2009], one can use the Expectation-Maximization algorithm [Dempster et al., 1977] to optimize the maximum likelihood criterion. However, for deep latent variable models such as (4.5), the posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ is intractable, and one has to rely on alternative strategies.

Autoencoding Variational Bayes framework The Autoencoding Variational Bayes framework [Kingma and Welling, 2013, Kingma et al., 2019] provides an efficient way to fit complex latent variable models such as (4.5). The key idea is to jointly train an inference model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, and the parameters of the generative model $p_\theta(\boldsymbol{z}, \boldsymbol{x})$. The role of the inference model is to approximate the intractable posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. It is typically parameterized as:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{z}; \mu_\phi(\boldsymbol{x}), \Sigma_\phi(\boldsymbol{x})\right), \tag{4.6}$$

where $\mu_\phi(.) : \mathbb{R}^n \to \mathbb{R}^d$, $\Sigma_\phi(.) : \mathbb{R}^n \to \mathbb{R}^{d\times d}$ are neural networks parameterized by ϕ. In order to derive a tractable training criterion we can exploit the following decomposition of the log-likelihood:

$$\log p_\theta(\boldsymbol{x}) = \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]}_{\mathcal{L}(\boldsymbol{x};\theta,\phi)} + \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})), \tag{4.7}$$

where

$$\mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) := \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.8}$$

is the Kullback-Leibler divergence between $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ and $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ [Kullback and Leibler, 1951]. We can recognize the decomposition used in variational inference [Zhang et al., 2018a], where we would maximize the evidence lower-bound (ELBO) $\mathcal{L}\left(\boldsymbol{x}; \theta, \phi\right)$ with respect to ϕ to approximate the posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. Since the primary goal of the VAE is to learn the generative model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$, [Kingma and Welling, 2013] propose instead to jointly optimize the ELBO with respect to ϕ and θ. Due to the non-negativity of the KL divergence, the ELBO is (as suggested by its name) a lower-bound on $\log p_\theta(\boldsymbol{x})$. We can also rewrite the ELBO in a different formulation that will be useful for the optimization:

$$\mathcal{L}\left(\boldsymbol{x}; \theta, \phi\right) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})). \tag{4.9}$$

[2] to not be mistaken with the posterior of the inverse problem $p(\boldsymbol{x}|\boldsymbol{y})$ in chapter 3

The formulation (4.9) decomposes the ELBO in two terms: the first term on the right-hand side can be viewed as a reconstruction term, penalizing bad reconstruction of the latent codes given by the encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, while the second term acts as a regularization term, penalizing the encoder distributions $q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big)$ to far from the prior $p_\theta(\boldsymbol{z})$.

Optimization The lower-bound is convenient for optimization, because it is possible to estimate its gradient using the formulation (4.9). Indeed, for a Gaussian encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ (4.6) and Gaussian prior $p_\theta(\boldsymbol{z})$ (4.5), the KL term in (4.9) can be computed in close form, as:

$$\mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) = \frac{1}{2}\left(tr(\Sigma_\phi(\boldsymbol{x})) + ||\mu_\phi(\boldsymbol{x})||^2 - \log|\Sigma_\phi(\boldsymbol{x})| - d\right). \tag{4.10}$$

In practice, it is common to impose $\Sigma_\phi(\boldsymbol{x})$ to be a diagonal matrix to make the computation of (4.10) easier. On the other hand, we can estimate the gradient of the reconstruction term efficiently using the "reparametrization trick" [Kingma and Welling, 2013, Rezende and Mohamed, 2015]. In the case of a Gaussian decoder $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \mu_\phi(\boldsymbol{x}), \Sigma_\phi(\boldsymbol{x}))$[3], and for any differentiable function F, we have that:

$$\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[F(\boldsymbol{z})] = \mathbb{E}_{p_\epsilon}\Big[F(\mu_\phi(\boldsymbol{x}) + \Sigma_\phi(\boldsymbol{x})^{\frac{1}{2}}\epsilon)\Big], \tag{4.11}$$

and, consequently, the gradient with respect of the variational parameters ϕ writes:

$$\nabla_\phi\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[F(\boldsymbol{z})] = \mathbb{E}_{p_\epsilon(\epsilon)}\Big[\nabla_\phi F(\mu_\phi(\boldsymbol{x}) + \Sigma_\phi(\boldsymbol{x})^{\frac{1}{2}}\epsilon)\Big], \tag{4.12}$$

and admits an unbiased, low-variance Monte-Carlo estimator:

$$\sum_{i=1}^{N} \nabla_\phi F(\mu_\phi(\boldsymbol{x}) + \Sigma_\phi(\boldsymbol{x})^{\frac{1}{2}}\epsilon^{(i)})), \tag{4.13}$$

where $\epsilon^{(i)} \sim p_\epsilon(\epsilon)$ are i.i.d samples.

Regularized Maximum likelihood training Optimizing the evidence-lower bound can also be interpreted as an implicit form of regularized maximum likelihood training. Indeed, from the decomposition of the log-likelihood (4.7), maximizing the ELBO on a dataset with respect to θ and ϕ amounts to solving [Shu et al., 2018]:

$$\max_\theta \left(\underbrace{\sum_{i=1}^{N} \log p_\theta\Big(\boldsymbol{x}^{(i)}\Big)}_{\text{"data"}} - \underbrace{\min_\phi \sum_{i=1}^{N} \mathrm{KL}\Big(q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big)||p_\theta\Big(z|\boldsymbol{x}^{(i)}\Big)\Big)}_{\text{"regularization"}} \right). \tag{4.14}$$

[3]For the sake of simplicity, we present the reparametrization trick for a Gaussian decoder, although it could be applied to other type of decoder, as discussed in [Kingma and Welling, 2013]

From (4.14), maximizing the ELBO corresponds to finding a model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$ that provides a compromise between assigning high-probability mass to the training data $\boldsymbol{x}^{(i)}$, while having a posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ that is close to the variational family $\{q_\phi(.|\boldsymbol{x}); \phi \in \Phi\}$. When using a Gaussian encoder (4.6), we are imposing the posterior $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ to be close to a Gaussian distribution whose parameters can be predicted by a neural network. Then, we can expect a form of smoothness in the decoder mapping, as close points in the latent space should imply close generated images. This property is of interest for downstream applications that involve manipulating the latent codes.

Optimal encoder In certain scenarios, the encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ can match the model posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. Indeed, as demonstrated in [Zhao et al., 2017], the ELBO loss verifies:

$$\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\mathcal{L}(\boldsymbol{x}; \theta, \phi)] = -\mathcal{H}(p_{\text{data}}(\boldsymbol{x})) - \text{KL}(p_{\text{data}}(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{x})p_\theta(\boldsymbol{z}|\boldsymbol{x})) \tag{4.15}$$

where we denote $\mathcal{H}(p_{\text{data}}(\boldsymbol{x}))$ the entropy of the training data distribution $p_{\text{data}}(\boldsymbol{x})$. The entropy does not depend on parameters θ and ϕ. Hence, relation (4.15) implies that the VAE is trained so that the ELBO reached its upper-bound:

$$\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\mathcal{L}(\boldsymbol{x}; \theta, \phi)] = -\mathcal{H}(p_{\text{data}}(\boldsymbol{x})), \tag{4.16}$$

then,

$$\text{KL}(p_{\text{data}}(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{x})p_\theta(\boldsymbol{z}|\boldsymbol{x})) = 0 \tag{4.17}$$

and, as a corollary:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = p_\theta(\boldsymbol{z}|\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \text{supp}(p_{\text{data}}(\boldsymbol{x})), \tag{4.18}$$

where we denote $\text{supp}(p_{\text{data}}(\boldsymbol{x}))$ the support of the training data distribution. For relation (4.16) to hold, it is necessary that the VAE encoder and decoder have enough capacity, and that the variational family $\mathcal{Q} = \{q_\phi(.|\boldsymbol{x}); \phi \in \Phi\}$ is expressive enough so that there exist $\phi^\star \in \Phi$ so that $p_\theta(\boldsymbol{z}|\boldsymbol{x}) = q_{\phi^\star}(\boldsymbol{z}|\boldsymbol{x})$. Therefore, it is technically possible that the VAE encoder matches the true model posterior, but in practice, it is not easy to verify, since we do not know the value of the ELBO upper bound $-\mathcal{H}(p_{\text{data}}(\boldsymbol{x}))$.

4.3 Modeling images with hierarchical VAE

VAE for images The variational autoencoder is a powerful tool to learn complex latent variable models involving deep neural networks. Thus, they appear to be well suited for images, which are high-dimensional, and exhibit complex structures. So far, we only have discussed the simple VAE model with a Gaussian prior over the latent space. This type of model is appealing for its simple formulation, and its potential ability to "disentangle" the independent generative factors of data [Burgess et al., 2018, Chen et al., 2018]. However, images generated by the simple VAE remain somehow blurry in practice, pushing the need for more expressive generative models, adapted to high-resolution, complex images.

Levels of abstraction in images representations Images can be considered as a set of spatially correlated features. We can consider features with different levels of abstraction, ranging from high-level features (scenes, objects) to low level features (textures, pixels). A key assumption in image processing is that features at a given level of abstraction can be described as a composition of lower-level features. This paradigm is one of the core motivation for the design of deep convolutional neural networks (CNN) [LeCun et al., 1998]. For instance, in image classification, CNNs map low-level information (an array of pixel intensities) to high level information (which object is in the image), by extracting a hierarchical sequence of features with different levels of abstraction [Yosinski et al., 2015, Olah et al., 2017]. For image generation, one would like to proceed in the reverse way, that is, to map high-level information encoded within a low-dimensional latent variable to an image represented as an array of pixels. However, contrary to the low-level $\rightarrow$ high level mapping, the high-level $\rightarrow$ low-level should be a one-to-many mapping.

Hierarchical generative model Hierarchical generative models provide an efficient way to account for the compositional structure of features in images. In order to model the different factors of variations, the latent variable $\boldsymbol{z}$ is partioned into L subgroups $\boldsymbol{z} = (\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{L-1})$, and each latent subgroup is typically a 3 dimensional tensor $\boldsymbol{z}_\ell \in \mathbb{R}^{c_\ell \times h_l \times w_l}$ composed of one channel dimension and 2 spatial dimensions. Each subgroup will control a different stage of the generative process (more details below). Intuitively, the first latent subgroup in the hierarchy should encode high-level information, while the latter one should encode low-level variation. Additionally, the prior is set to have a hierarchical structure[4]:

$$p_\theta(\boldsymbol{z}) = p_\theta(\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{L-1}) \tag{4.19}$$

$$= p_\theta(\boldsymbol{z}_0) \prod_{\ell=1}^{L-1} p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}), \tag{4.20}$$

and the prior at each level l is set as a multivariate Gaussian:

$$\begin{cases} p_\theta(\boldsymbol{z}_0) & = \mathcal{N}\left(\boldsymbol{z}_0; \mu_{\theta,0}, \Sigma_{\theta,0}\right) \\ p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}) & = \mathcal{N}\left(\boldsymbol{z}_\ell; \mu_{\theta,\ell}\left(\boldsymbol{z}_{<\ell}\right), \Sigma_{\theta,l}(\boldsymbol{z}_{<\ell})\right), \end{cases} \tag{4.21}$$

where $\mu_{\theta,0}$ and $\Sigma_{\theta,0}$ can either be trainable or non-trainable constants, and the remaining mean vectors ($\mu_{\theta,l}$, and $\mu_{\phi,l}$, for $l > 0$) and covariance matrices ($\Sigma_{\theta,\ell}$ and $\Sigma_{\phi,\ell}$, for $l > 0$) are parameterized by neural networks.

[4] In some implementation, the prior can also have a Markov structure $p_\theta(\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{L-1}) = \prod_{\ell=1}^{L-1} p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{l-1})$

Top-down encoder The introduction of the Top-Down encoder [Sønderby et al., 2016], was an important contribution for training hierarchical VAEs with a large number of latent groups in a stable way. The top-down encoder is designed to infer the latent groups in the same order as the generative model:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}_0|\boldsymbol{x}) \prod_{\ell=1}^{L-1} q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x}), \tag{4.22}$$

with Gaussian conditionals:

$$\begin{cases} q_\phi(\boldsymbol{z}_0|\boldsymbol{x}) & = \mathcal{N}\left(\boldsymbol{z}_0; \mu_{\phi,0}(\boldsymbol{x}), \Sigma_{\phi,0}(\boldsymbol{x})\right) \\ q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x}) & = \mathcal{N}\left(\boldsymbol{z}_\ell; \mu_{\phi,\ell}\left(\boldsymbol{z}_{<\ell}, \boldsymbol{x}\right), \Sigma_{\phi,l}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})\right), \end{cases} \tag{4.23}$$

parametrized by neural networks $\mu_{\phi,\ell}\left(\boldsymbol{z}_{<\ell}, \boldsymbol{x}\right)$ and $\Sigma_{\phi,l}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})$. This is in opposition with the models with bottom-up inference (see Figure 4.3b), which were observed to be hard to train when L was too large [Sønderby et al., 2016]. The ordering of the latent groups in the inference model leads to a convenient KL term in the ELBO loss (4.9), as it then writes as a summation over the KL divergences between the Gaussian conditional priors $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$ and inference model $q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ at each level l:

$$\mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) = \mathrm{KL}(q_\phi(\boldsymbol{z}_0|\boldsymbol{x})||p_\theta(\boldsymbol{z}_0)) + \sum_{l=1}^{L-1} \mathbb{E}_{q_\phi(\boldsymbol{z}_{<\ell}|\boldsymbol{x})}[\mathrm{KL}(q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})||p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}))]. \tag{4.24}$$

In (4.24), each KL term has a closed form expression as the KL divergence between two multivariate Gaussians (in practice, the covariance matrices $\Sigma_{\theta,l}$ and $\Sigma_{\phi,l}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ are set to be diagonal matrix in order to fasten the computation). An estimate of the gradient of (4.24) can be computed efficiently, by using the reparametrization trick for each KL term.

VDVAE As an example of hierarchical VAE, we present Very Deep VAE (VDVAE) [Child, 2020], that we will reuse in the rest of this work. When it was introduced in 2020, VDVAE reached state-of-the art results on several challenging image datasets, as measured by the model likelihood on the test set. VDVAE implements a hierarchical generative model (4.20), with a top-down inference model (4.22) as described above. As illustrated in Figure 4.4b, a bottom-up network (left) extracts a sequence of features at different scales, and feeds them to the top-down network (right). The top-down network is composed of top-down blocks and upsampling operations. Each top-down block (Figure 4.4a) corresponds to a latent group $\boldsymbol{z}_\ell$, and is composed of a branch that will infer the parameters of the prior (in blue in Figure 4.4a), and a branch for the inference network that infers the parameters of $q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ (in green in Figure 4.4a). The inference branch takes as an input features of the image $\boldsymbol{x}$ coming from the bottom-up network.

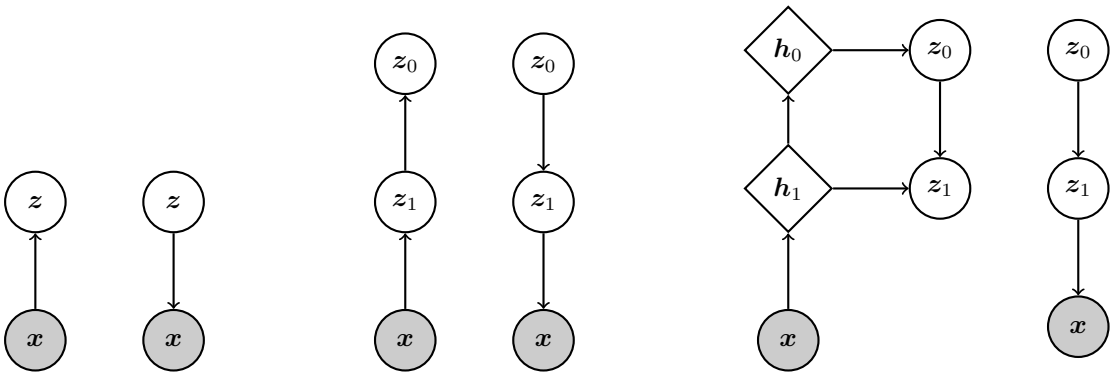
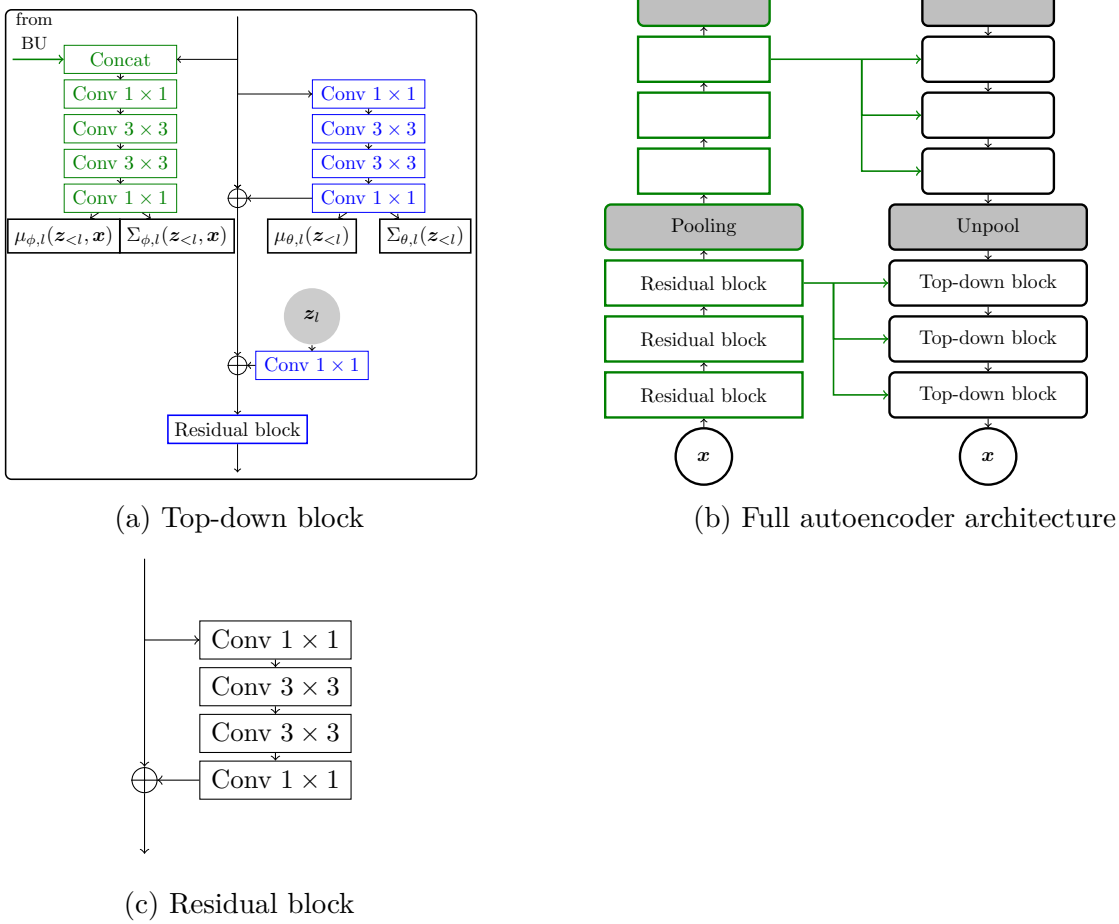


Figure 4.3 – Inference model (left) and generative model (right) for a simple (non-hierarchical) VAE (a), a hierarchical VAE with bottom-up inference (b), and hierarchical with top-down inference (c)

Properties of the hierarchical latent representation The generative model of VDVAE provides a rich hierarchical latent representation of images. We can visualize the effect of each latent group of the hierarchy on the generated image by sampling images from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<\ell})$ while keeping $\boldsymbol{z}_{<\ell}$ fixed, for different values of l. We can then deduce that the attributes that are common to all samples from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<\ell})$, are most likely to be encoded in the latent groups $\boldsymbol{z}_{<\ell}$. Our experiments in Figure 4.5 illustrate that VDVAE encodes high-level semantic information within its first latent variables.

Interpolation with VDVAE Another interesting feature of VDVAE is its ability to interpolate between images. We describe a simple strategy to interpolate between two images in algorithm 2, and we present an example of interpolation in Figure 4.6. This interpolation strategy is naive, as it ignores the hierarchical structure of the latent space, and it does not account for its geometry [Chadebec and Allassonnière, 2022]. However, it already provides a smooth transition between images. This suggests that, despite its hierarchical nature, VDVAE latent space is smooth, in the sense that close latent vectors lead to close generated images.

Low temperature model In order to improve the quality of the generated samples after training, and to make the model distribution closer to the data distribution in terms of FID metric, a trick used by practitioners is to reduce the variance of the latent prior [Kingma and Dhariwal, 2018, Vahdat and Kautz, 2020, Child, 2020, Karras et al., 2020]. For HVAE this is done by multiplying the covariance matrix of the Gaussian distribution $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$ by a factor $\tau_l^2 < 1$. As can be seen in Figure 4.7, sampling the images at a slightly reduced temperature ($\tau_\ell = 0.8$) helps reducing the artefacts visible in the samples at full temperature ($\tau = 1$). On the other hand, images sampled at a lower



(a) Top-down block

(b) Full autoencoder architecture

(c) Residual block

Figure 4.4 – Structure of the VDVAE architecture. For clarity, we omit the non-linearity after each convolution. The number of blocks differs for each dataset.

temperature provide over smoothed results with limited diversity. We denote the model with reduced prior temperature:

$$p_{\theta,\tau}\left(\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{l-1}, \boldsymbol{x}\right) = \frac{p_\theta(\boldsymbol{z}_0)^{\frac{1}{\tau_0^2}}}{Z_0} \prod_{\ell=1}^{L-1} \frac{p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\frac{1}{\tau_\ell^2}}}{Z_\ell} p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<L}), \tag{4.25}$$

(a) $l = 3$

(b) $l = 4$

(c) $l = 5$

Figure 4.5 – Samples from VDVAE model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<\ell})$ for different values of l. The first latents ($l = 3$) encode high-level semantic informations (age, genders), and the following ones encodes attributes such as face expression, skin and eyes colors.

Figure 4.6 – Image interpolation with VDVAE.

Algorithm 2 Interpolation in hierarchical VAE latent space

Input: 2 images $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$, number of interpolated images n
Sample $\boldsymbol{z}_k, \cdots, \boldsymbol{z}_0 \sim q_\phi\left(\boldsymbol{z}_k, \cdots, \boldsymbol{z}_0 | \boldsymbol{x}\right)$
Sample $\tilde{\boldsymbol{z}}_k, \cdots, \tilde{\boldsymbol{z}}_0 \sim q_\phi\left(\tilde{\boldsymbol{z}}_k, \cdots, \tilde{\boldsymbol{z}}_0 | \tilde{\boldsymbol{x}}\right)$
for $i = 1$ **to** n **do**
  $t \leftarrow \frac{i}{n+1}$
  $\boldsymbol{z}_j^{(i)} \leftarrow (1-t)\boldsymbol{z}_j + t\tilde{\boldsymbol{z}}_j, \quad \forall j \in \{0, \cdots, k\}$
  $\boldsymbol{x}^{(i)} \sim p_\theta\left(x | \boldsymbol{z}_k^{(n)}, \cdots, \boldsymbol{z}_0^{(i)}\right)$
end for
Return $\{\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(n)}\}$

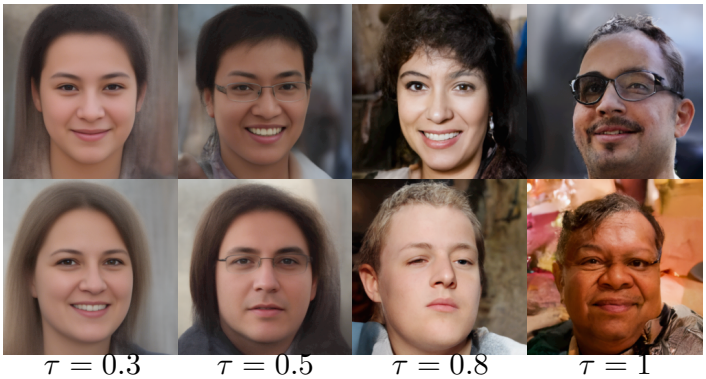


Figure 4.7 – Effect of the temperature $\tau_\ell = \tau$ on the images generated by VDVAE.

where the variables $Z_\ell := \int p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell})^{\frac{1}{\tau_\ell^2}} d\boldsymbol{z}_\ell$ are normalizing constants[5]. In (5.14), $\boldsymbol{\tau} := (\tau_0, \cdots, \tau_{L-1})$ gives the temperature for each level of the hierarchy. In the following, we use this temperature-scaled model to balance the regularization of our inverse problem. The temperature-scaled model marginal is then defined as:

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) = \int p_{\theta,\tau}\left(\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{l-1}, \boldsymbol{x}\right) d\boldsymbol{z}_0 \cdots d\boldsymbol{z}_{L-1} \tag{4.26}$$

4.3.1 Conclusion

The variational autoencoder can help to learn powerful generative models on images. They come with several attributes that will be helpful for downstream applications such as image restoration. Namely, the ELBO training criterion enforces smoothness of the generative network, and the VAE encoder allows to efficiently map images to their latent representations, enabling fast and smooth latent code manipulation. In

[5]Precisely, we have $Z_\ell = \tau_\ell^{\frac{d_\ell}{2}}$, where $d_\ell = c_\ell \times h_l \times w_l$ is the dimension of the latent variable $\boldsymbol{z}_\ell$

particular, hierarchical VAEs provide high-quality generative models, that can match the performance of other classes of generative models such as GANs, while benefiting from the aforementioned advantages of VAEs. In the following chapters, we will demonstrate how to efficiently leverage the properties of deep hierarchical VAEs such as VDVAE for different image restoration tasks.

Chapter 5

Inverse problem regularization with hierarchical variational autoencoders

In this chapter, we propose to regularize ill-posed inverse problems using a deep hierarchical variational autoencoder (HVAE) as an image prior. The proposed method synthesizes the advantages of i) denoiser-based Plug & Play approaches and ii) generative model based approaches for inverse problems. First, we exploit VAE properties to design an efficient algorithm that benefits from convergence guarantees of Plug-and-Play (PnP) methods. Second, our approach is not restricted to specialized datasets and the proposed PnP-HVAE model is able to solve image restoration problems on natural images of any size. Our experiments show that the proposed PnP-HVAE method is competitive with both SOTA denoiser-based PnP approaches, and other SOTA restoration methods based on generative models. The code for this project is available at https://github.com/jprost76/PnP-HVAE.

5.1 Introduction

Linear inverse problem In this chapter, we still focus on linear inverse problems

$$\boldsymbol{y} = A\boldsymbol{x} + \epsilon \tag{5.1}$$

in which $\boldsymbol{y} \in \mathbb{R}^m$ is the degraded observation, $\boldsymbol{x} \in \mathbb{R}^d$ the original signal we wish to retrieve, $A \in \mathbb{R}^{m \times d}$ is an observation matrix and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is an additive Gaussian noise. Many image restoration tasks can be formulated as (5.1), including deblurring, super-resolution or inpainting.

End to end restoration With the development of deep learning in computer vision, image restoration have known significant progress. The most straight-forward way to exploit deep learning for solving image inverse problems is to train a neural network to map degraded images to their clean version in a supervised fashion. However, this type of

approach requires a large amount of training data, and it lacks flexibility, as one network is needed for each different inverse problem.

Generative network inversion An alternate approach is to use deep latent variable generative models such as GANs or VAEs and to compute the Maximum-a-Posterior (MAP) estimator in the latent space:

$$\hat{\boldsymbol{z}} = \arg\max_{\boldsymbol{z}} \log p\left(\boldsymbol{y}|G(\boldsymbol{z})\right) + \lambda \log p\left(\boldsymbol{z}\right), \tag{5.2}$$

where $\boldsymbol{z}$ is the latent variable and G is the generative network [Bora et al., 2017, Menon et al., 2020]. In (5.2) the likelihood $p\left(\boldsymbol{y}|G(\boldsymbol{z})\right)$ is related to the forward model (5.1), and $p\left(\boldsymbol{z}\right)$ corresponds to the prior distribution over the latent space. After solving (5.2), the solution of the inverse problem is defined as $\hat{\boldsymbol{x}} = G\left(\hat{\boldsymbol{z}}\right)$. The latent optimization methods (5.2) provide high-quality solutions that are guaranteed to be in the range of a generative network. However, this implies highly non-convex problems (5.2) due to the complexity of the generator and the obtained solutions may lack of consistency with the degraded observation (as noticed in [Saharia et al., 2021c]). Although the convergence of latent optimization algorithms has been studied in the literature, existing convergence guarantees are either restricted to specific settings, or rely on assumptions that are difficult to enforce in practice [Huang et al., 2021]

Hierarchical VAE prior In this chapter, we propose an algorithm that exploits the strong prior of a deep generative model while providing realistic convergence guarantees. We consider a specific type of deep generative model, the hierarchical variational autoencoder (HVAE). As discussed in chapter 4, HVAE models give state-of-the-art results on image generation benchmarks [Vahdat and Kautz, 2020, Child, 2020, Hazami et al., 2022, Luhman and Luhman, 2022], and provide an encoder that will be key in the design of our proposed method.

Low-temperature model As the HVAE models differ significantly from the architecture of concurrent models, it is necessary to design algorithms adapted to their specific structure. The latent space dimension of HVAE is significantly higher than the image dimension. Hence, constraining the solution to lie in the image of the generator is not enough to regularize inverse problems. Indeed, it has been observed that HVAEs can perfectly reconstruct out-of-domain images [Havtorn et al., 2021b]. Consequently, we propose to constrain the latent variable of the solution to lie in the high probability area of the HVAE prior distribution. This can be done efficiently by controlling the variance of the prior over the latent variables.

Avoiding backpropagation The common practice of optimizing the latent variables of the generative model with backpropagation is impractical for hierarchical generative

models, because of the high dimensionality of the hierarchical latent space. Instead, we exploit the HVAE encoder to define an alternating algorithm [González et al., 2022] to optimize the joint distribution over the image and its latent variable

Connection with Plug-and-Play To derive convergence guarantees for our algorithm, we show that it can be reformulated as a Plug-and-Play (PnP) method [Venkatakrishnan et al., 2013], which alternates between an application of the proximal operator of the data-fidelity term, and a reconstruction by the HVAE. Under this perspective, we give sufficient conditions to ensure the convergence of our method, and we provide an explicit characterization of the fixed-point of the iterations. Motivated by the parallel with PnP methods, we name our method PnP-HVAE.

Contributions and outline

In this work, we introduce PnP-HVAE, a method for regularizing image restoration problems with a hierarchical variational autoencoder. Our approach exploits the expressiveness of a deep HVAE generative model and its capacity to provide a strong prior on specialized datasets, as well as convergence guarantees of Plug-and-Play methods and their ability to deal with natural images of any size.

We start by a brief review of related works on deep learning based regularization for imaging inverse problems (section 5.2), We then present in section 5.3 the specific background on VAE based inverse problem regularization, with a focus on JPMAP, an algorithm that serves as an inspiration for our proposed method. Next we present our contributions:

- In section 5.4, we introduce PnP-HVAE, an algorithm to solve inverse problems with a HVAE prior. PnP-HVAE optimizes a joint posterior on image and latent variables without backpropagation through the generative network. It can be viewed as a generalization of JPMAP [González et al., 2022] to hierarchical VAEs, with additional control of the regularization.

- In section 5.5, we demonstrate the convergence of PnP-HVAE under hypotheses on the autoencoder reconstruction. Numerical experiments illustrate that the technical hypotheses are empirically met on noisy images with our proposed architecture. We also exhibit the better convergence properties of our alternate algorithm with respect to the use of Adam for optimizing the joint posterior objective.

- In section 5.6, we demonstrate the effectiveness of PnP-HVAE through image restoration experiments and comparisons on (i) faces images using the pre-trained VDVAE model from [Child, 2020]; and (ii) natural images using the proposed PatchVDVAE architecture trained on natural image patches.

5.2 Related works

We present two prominent lines of work for deep learning based regularization of image inverse problems, namely, plug-and-play methods exploiting deep image denoisers, and methods based on deep generative models.

5.2.1 Plug-and-Play methods

Plug-and-Play (PnP) and RED methods [Venkatakrishnan et al., 2013, Romano et al., 2017a] make use of a (deep) denoiser as a proxy to encode the local information over the prior distribution. The denoiser is plugged in an optimization algorithm such as Half-Quadratic Splitting or ADMM in order to solve the inverse problem. PnP algorithms come with theoretical convergence guarantees by imposing certain conditions on the denoiser network [Ryu et al., 2019, Pesquet et al., 2021, Hurault et al., 2022]. These approaches provide state-of-the-art results on a wide variety of image modality thanks to the excellent performance of the currently available deep denoiser architectures [Zhang et al., 2021]. However, PnP methods are only implicitly related to a probabilistic model, and they provide limited performance for challenging structured problems such as the inpainting of large occlusions.

5.2.2 Deep generative models for inverse problems

Generative models represent an explicit image prior that can be used to regularize ill-posed inverse problems [Bora et al., 2017, Latorre et al., 2019, Menon et al., 2020, Daras et al., 2021, Oberlin and Verm, 2021, Pan et al., 2021, Song et al., 2021a]. They are latent variable models parameterized by neural networks, optimized to fit a training data distribution [Kingma and Welling, 2013, Goodfellow et al., 2020, Dinh et al., 2016, Ho et al., 2020].

Convergence issues Regularization with generative models (5.2) involves solving a highly non-convex optimization problem over latent variables [Bora et al., 2017, Menon et al., 2020, Oberlin and Verm, 2021], for which it is difficult to derive theoretical convergence guarantees. Researchers have been working to derive assumptions under which optimization provably converges [Shah and Hegde, 2018, Raj et al., 2019, González et al., 2022]. For compressed sensing, it has been shown that gradient descent converges to a local neighborhood of the solution with high-probability, under the assumption that the generative network has random Gaussian weights [Hand et al., 2018]. Also for compressed sensing, [Shah and Hegde, 2018] introduced a projected gradient algorithm that provably converges to the global solution with high-probability, under the assumption of the existence of an oracle projection function on the image of the generative network manifold. For

generic forward models, convergence to a local minimum, and the property of those local minima are still to be investigated. With a VAE prior, the JPMAP algorithm converges to a local minimum of a specific energy under the assumption that the VAE encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ perfectly matches the intractable VAE posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ [González et al., 2022] (more details will be given below).

5.3 Joint Posterior Maximization with Autoencoding Prior

Using a hierarchical VAE model as a prior to regularize an inverse problem is challenging, because the high-dimensionality of the latent space, and the hierarchical structure of the latent prior $p_\theta(\boldsymbol{z})$ (4.20) make approaches relying on backpropagation impractical. The Joint Posterior Maximization with Autoencoding Prior (JPMAP) algorithm of [González et al., 2022] introduces several ideas that will be key in the development of our method, including the choice of computing "joint" MAP estimator to circumvent the intractability of $p_\theta(\boldsymbol{x})$, and the use of the VAE encoder within an alternate optimization scheme to avoid backpropagation through the VAE generator. As a preliminary, we present in this section the main idea behind the JPMAP algorithm.

Joint MAP estimator A direct approach to use a VAE model as a prior to solve image inverse problems is to compute the classical MAP estimator with the prior induced by the VAE $p_\theta(\boldsymbol{x})$:

$$\boldsymbol{x}_{\text{MAP}} = \arg\min_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x}) + \log p_\theta(\boldsymbol{x}). \tag{5.3}$$

However, because evaluating the VAE model marginal $p_\theta(\boldsymbol{x})$ requires computing an intractable integral (4.4), it is not clear how to compute this MAP estimator. The main idea of the JPMAP algorithm is to consider instead the augmented model including the VAE latent variable $\boldsymbol{z}$:

$$p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) := p_\theta(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{y}|\boldsymbol{x}), \tag{5.4}$$

and to compute the associated joint MAP estimator:

$$\boldsymbol{x}^\star, \boldsymbol{z}^\star = \arg\max_{\boldsymbol{x},\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{y}). \tag{5.5}$$

For a linear forward model $p(\boldsymbol{y}|\boldsymbol{x})$ and a Gaussian VAE (4.5), solving (5.5) amounts to minimizing the energy[1]:

$$J_1(\boldsymbol{x}, \boldsymbol{z}) = \frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2 + \frac{1}{2}||\boldsymbol{x} - \mu_\theta(\boldsymbol{z})||^2_{\Sigma_\theta^{-1}(\boldsymbol{z})} + \frac{1}{2}\log\det\left(\Sigma_\theta^{-1}(\boldsymbol{z})\right) + \frac{1}{2}||\boldsymbol{z}||^2. \tag{5.6}$$

[1] we use the notation $||\boldsymbol{x}||^2_M = \boldsymbol{x}^t M \boldsymbol{x}$

Alternate optimization JPMAP employs an alternate optimization scheme to compute the joint-MAP estimator by minimizing the negative logarithm of the joint posterior (5.5):

$$\boldsymbol{z}^{(n+1)} = \arg\min_{\boldsymbol{z}} -\log p(\boldsymbol{x}^{(n)}, \boldsymbol{z}|\boldsymbol{y}) \tag{5.7}$$

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}} -\log p(\boldsymbol{x}, \boldsymbol{z}^{(n+1)}|\boldsymbol{y}). \tag{5.8}$$

It is clear from (5.6) that the subproblem (5.8) is convex and admits the closed form solution:

$$\boldsymbol{x}^{(n+1)} = \left(\frac{A^tA}{\sigma^2} + \Sigma_\theta^{-1}\left(\boldsymbol{z}^{(n+1)}\right)\right)^{-1}\left(\frac{A^t\boldsymbol{y}}{\sigma^2} + \Sigma_\theta^{-1}\left(\boldsymbol{z}^{(n+1)}\right)\mu_\theta\left(\boldsymbol{z}^{(n+1)}\right)\right), \tag{5.9}$$

On the other hand, the subproblem in $\boldsymbol{z}$ (5.7) is not convex and does not admit a closed-form solution, due to the terms $\mu_\theta(\boldsymbol{z})$ and $\Sigma_\theta^{-1}(\boldsymbol{z})$ involving neural networks. To avoid the use of an iterative optimization algorithm such as gradient descent to solve (5.7), the authors of [González et al., 2022] propose to rely on the VAE encoder to efficiently compute an approximate solution as:

$$\boldsymbol{z}^{(n+1)} = \arg\min_{\boldsymbol{z}} -\log q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(n)}\right) \tag{5.10}$$

$$\approx \underbrace{\arg\min_{\boldsymbol{z}} -\log p_\theta\left(\boldsymbol{z}|\boldsymbol{x}^{(n)}\right)}_{= \arg\min_{\boldsymbol{z}} -\log p_\theta\left(\boldsymbol{z}, \boldsymbol{x}^{(n)}|\boldsymbol{y}\right)} \tag{5.11}$$

(5.12)

Since $q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(n)}\right)$ is Gaussian (4.6), the solution of (5.10) is simply the mean of the Gaussian distribution $\mu_\phi\left(\boldsymbol{x}^{(n)}\right)$, which can be computed with the VAE encoder network. Hence, the computation of $\boldsymbol{z}^{(n+1)}$ in (5.10) simply requires one forward pass of the encoder on the current value $\boldsymbol{x}^{(n)}$, instead of an iterative optimization algorithm that would require many forward and backward passes with the decoder network. The final JPMAP iteration is then the sequence of one exact minimization of $J_1(\boldsymbol{x}, \boldsymbol{z})$ with respect to $\boldsymbol{x}$ (5.9), and an approximate minimization with respect to $\boldsymbol{z}$ (5.10):

$$\begin{cases} \boldsymbol{z}_{n+1} = & \mu_\phi(\boldsymbol{x}_n) \\ \boldsymbol{x}_{n+1} = & \left(\frac{A^tA}{\sigma^2} + \Sigma_\theta^{-1}\left(\boldsymbol{z}_{n+1}\right)\right)^{-1}\left(\frac{A^t\boldsymbol{y}}{\sigma^2} + \Sigma_\theta^{-1}\left(\boldsymbol{z}_{n+1}\right)\mu_\theta\left(\boldsymbol{z}_{n+1}\right)\right), \end{cases} \tag{5.13}$$

Convergence of JPMAP In the ideal case where the VAE encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ perfectly matches the true VAE posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, [González et al., 2022] shows that $J_1(\boldsymbol{x}, \boldsymbol{z})$ is a biconvex function. Then, the alternate scheme (5.13) corresponds to an alternate convex search, and converges to a stationary point of J_1 [González et al., 2022]. The assumption

on the decoder can be met under several assumptions discussed in subsection 4.2. However, there is no practical way to verify if this assumption is met, and several works suggest that Gaussian encoders fail to match the model posterior in practical settings [Cremer et al., 2018, Marino et al., 2018, Zhang et al., 2022]. To ensure convergence to a stationary point without assuming a perfect encoder, the authors of [González et al., 2022] use gradient descent to solve the subproblem (5.7) when the encoder approximation is not good enough to ensure decreasing value of the energy $J_1(\boldsymbol{x}_n, \boldsymbol{z}_n)$ along the iterates.

Limitations Along with the convergence issues due to the imperfection of the encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, JPMAP has limitations hindering its usage for real-world applications. First, its formulation is only adapted to simple VAE models, that lack expressivity for modeling high-resolution images, and tend to generate blurry images. Second, while classical variational methods give control on the strength of the regularization by the choice of a scalar hyperparameter, JPMAP does not provide any control of this sort. It would be tempting to add a multiplicative factor in front of the "regularization" term $||\boldsymbol{z}||^2$ in (5.6) to enforce control, however, because the "coupling" term $\frac{1}{2}||\boldsymbol{x} - \mu_\theta(\boldsymbol{z})||^2_{\Sigma_\theta^{-1}(\boldsymbol{z})}$ is not linear, it is not straightforward to adapt JPMAP to this new problem.

5.4 Regularization with HVAE Prior

In this section we introduce PnP-HVAE, our Plug-and-Play method to solve generic image inverse problems with a Hierarchical VAE prior. The formulation of our method shares the main principles introduced in JPMAP, namely, it is motivated by an alternate optimization scheme to compute a joint MAP estimator similar to (5.17), and we use the HVAE encoder to efficiently solve the sub-problem over the latent variable. Nevertheless, we introduce several novelties to overcome the main limitations of JPMAP discussed in section 5.3. In particular:

- Our method is adapted to Hierarchical VAE models with top-down inference networks, that are more expressive than the simple Gaussians VAEs used in JPMAP.
- In our method, the strength of the regularization can be controlled by tuning the temperature of the Gaussian latent priors of the HVAE model.

We show in section 5.4.1 that the strength of the regularization can be monitored by tuning the temperature of the prior in the latent space. In section 5.4.2, we propose an approximation of the low-temperature joint posterior distribution using the hierarchical VAE encoder. Using this approximation, we introduce in section 5.4.3 our final algorithm based on an alternate optimization scheme, that includes a new sequential scheme to optimize the latent variable of the HVAE.

5.4.1 Tempered hierarchical joint posterior

Mode covering behavior of hierarchical VAE models The likelihood based criterion used to train HVAE models is known to be mode-covering (see discussion in chapter 10.2 in [Bishop and Nasrabadi, 2006]). As such, it will drive the learned model to cover every mode of the training data distribution, at the risk of also assigning non-zero probability to out-of-distribution data-points. This can be a limitation for image restoration applications, in particular if the learned model assigns high probability to degraded images. Indeed, we observed in our preliminary experiments that using an HVAE model within a JPMAP like algorithm without any mechanism to increase the strength of the regularization leads to poor restoration results, as the regularization is too weak to avoid artifacts due to the noise in the observation.

Low-temperature HVAE model In order to increase the strength of the regularization, we propose to use an HVAE model with reduced temperature, as described in chapter 4, section 4.3. As a reminder, we define the joint low-temperature HVAE model as:

$$p_{\theta,\tau}\left(\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{L-1}, \boldsymbol{x}\right) = \frac{p_\theta(\boldsymbol{z}_0)^{\frac{1}{\tau_0^2}}}{Z_0} \prod_{\ell=1}^{L-1} \frac{p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\frac{1}{\tau_\ell^2}}}{Z_\ell} p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<L}), \tag{5.14}$$

where Z_ℓ are normalizing constants. Notice that when $\tau_\ell = 1$, we retrieve the original joint HVAE model, and reducing the value of τ_ℓ at each level l increases the strength of the regularization. Our motivations for using a low-temperature HVAE model are twofold. First we expect that a lower temperature τ will help reducing the value of the model $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ on out-of-distribution data points. Second, HVAE models typically produce their best samples in terms of image quality with a temperature slightly below than one (e.g. $\tau = 0.85$) [Vahdat and Kautz, 2020, Child, 2020].

Joint MAP criterion Using the low-temperature HVAE model described in equation (5.14), we define the associated tempered joint model as:

$$\begin{aligned} p\left(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}\right) &= p\left(\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{L-1}, \boldsymbol{x}, \boldsymbol{y}\right) && (5.15)\\ &:= p_{\theta,\tau}\left(\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{L-1}, \boldsymbol{x}\right) p\left(\boldsymbol{y}|\boldsymbol{x}\right). && (5.16) \end{aligned}$$

Following JPMAP motivations, we aim at finding the couple $(\boldsymbol{x}, \boldsymbol{z})$ that maximizes the joint posterior $p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{y})$:

$$\arg\min_{\boldsymbol{x},\boldsymbol{z}} -\log p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{y}). \tag{5.17}$$

Although we are only interested in finding the image $\boldsymbol{x}$, the joint Maximum A Posteriori (MAP) criterion (5.17) makes it possible to derive an optimization scheme that only relies on forward calls of the HVAE, as we describe in the following.

Effect of the temperature By ignoring the constant terms, the joint MAP estimate, solution of (5.17) is the minimizer of the energy:

$$J_1(\boldsymbol{x}, \boldsymbol{z}) := -\log p(\boldsymbol{y}|\boldsymbol{x}) - \frac{1}{\tau_0^2}\log p_\theta(\boldsymbol{z}_0) - \sum_{l=1}^{L-1}\frac{1}{\tau_\ell^2}\log p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}) - \log p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<L}). \quad (5.18)$$

Relation (5.18) illustrates the influence of the temperature factors τ_ℓ on the final objective function. The ratios $\frac{1}{\tau_\ell^2}$ control the influence of the latent prior terms $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$, like the scalar coefficient λ in front of a regularization term $g(\boldsymbol{x})$ in a typical variational problem $\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$. Reducing the temperatures τ_ℓ increases the strength of the regularization on the latent variables $\boldsymbol{z}_\ell$ in the joint model.

5.4.2 Encoder approximation of the joint posterior

We now derive an approximation of the joint model (5.15) based on the HVAE encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. This approximate model will be useful for deriving an alternate optimization scheme.

Approximate joint posterior We can rewrite the joint model (5.15) as:

$$p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) = p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x}). \quad (5.19)$$

We would like to replace the low-temperature model posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ in (5.19) by an approximation given by the HVAE encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. However, the encoder is only trained to approximate the posterior of the model $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ (that is, for $\tau = 1$), and not the posterior of the low-temperature model $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$.

Posterior of the low-temperature model In the following proposition, we show that the low-temperature posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ can be formulated as a combination of the model posterior at $\tau = 1$, $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, and the model prior $p_\theta(\boldsymbol{z})$.

Proposition 5.1. *The low-temperature model posterior satisfies:*

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_0|\boldsymbol{x})\prod_{\ell=1}^{L-1}p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x}) \quad (5.20)$$

with

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_0|\boldsymbol{x}) = \frac{1}{Z_0}p_\theta(\boldsymbol{z}_0|\boldsymbol{x})p_\theta(\boldsymbol{z}_0)^{\lambda_0} \quad (5.21)$$

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x}) = \frac{1}{Z_\ell}p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\lambda_\ell}, \quad (5.22)$$

and $\lambda_\ell := \frac{1}{\tau_\ell^2} - 1$.

We provide a detailed proof of this result in appendix A.1.1.

Relation (5.20) gives a relation between the low-temperature model posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ and the original posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. At each level l, the low-temperature model posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ is the product of the posterior $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ and the prior $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$ of the original model. The influence of the prior terms at each level is inversely proportional to the temperature τ_ℓ.

Encoder approximation of the low-temperature model Using relation (5.20), we can approximate the low-temperature model (5.14) posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ with the encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. We approximate each level of the hierarchical posterior (5.22) by using the approximation $q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$:

$$q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) := \frac{1}{Z_\ell} q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\lambda_\ell} \tag{5.23}$$

$$\approx p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}). \tag{5.24}$$

Then, we define the low-temperature encoder by plugging the approximation (5.23) in (5.20):

$$q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})} q_{\phi,\tau}(\boldsymbol{z}_0|\boldsymbol{x}) \prod_{\ell=1}^{L-1} q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}). \tag{5.25}$$

By construction, if $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = p_\theta(\boldsymbol{z}|\boldsymbol{x})$, we have that $q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) = p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$. We then define the approximate joint model by injecting the formulation of the low-temperature posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ (5.20) in (5.19):

$$q(\boldsymbol{z},\boldsymbol{x},\boldsymbol{y}) := q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) p(\boldsymbol{y}|\boldsymbol{x}) \tag{5.26}$$

$$:= p_\theta(\boldsymbol{x}) q_{\phi,\tau}(\boldsymbol{z}_0|\boldsymbol{x}) \prod_{\ell=1}^{L-1} q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) p(\boldsymbol{y}|\boldsymbol{x}). \tag{5.27}$$

In the case where $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = p_\theta(\boldsymbol{z}|\boldsymbol{x})$, we have $q(\boldsymbol{z},\boldsymbol{x},\boldsymbol{y}) = p(\boldsymbol{z},\boldsymbol{x},\boldsymbol{y})$. As discussed in subsection 4.2, this assumption can be met if the variational family $\{q_\phi(.|\boldsymbol{x}); \phi \in \Phi\}$ contains the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ and if the VAE is trained to reach the ELBO upper-bound. If this assumption appears unrealistic for vanilla (non-hierarchical) VAE [González et al., 2022], our experiments suggest that HVAE hierarchical encoders are sufficiently expressive to match the posterior to a reasonably good accuracy. Computing the joint MAP estimator for the approximate model:

$$\arg\max_{\boldsymbol{x},\boldsymbol{z}} q(\boldsymbol{z},\boldsymbol{x}|\boldsymbol{y}) \tag{5.28}$$

is equivalent to minimizing the following energy:

$$J_2(\boldsymbol{x},\boldsymbol{z}) := -\sum_{\ell=0}^{L-1} \log q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) - \log p(\boldsymbol{y}|\boldsymbol{x}) - \log p_\theta(\boldsymbol{x}). \tag{5.29}$$

We will develop in the following subsection on how to efficiently compute an approximate minimizer of $J_2(\boldsymbol{x}, \boldsymbol{z})$ with respect to $\boldsymbol{z}$.

5.4.3 Alternate optimization with PnP-HVAE

Alternate optimization We now developp our algorithm to compute the joint MAP (5.17) based on an alternate optimization scheme. The ideal optimization scheme writes:

$$\boldsymbol{z}^{(n+1)} = \arg\min_{\boldsymbol{z}} J_1(\boldsymbol{x}^{(n)}, \boldsymbol{z}) \tag{5.30}$$

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}} J_1(\boldsymbol{x}, \boldsymbol{z}^{(n+1)}). \tag{5.31}$$

Exact minimization in $\boldsymbol{z}$ The subproblem in $\boldsymbol{z}$ (5.30) does not admit a closed form solution, and is highly non-convex as the variables $\boldsymbol{z}_\ell$ are arguments of a neural network. We could use first order iterative optimization scheme to solve it, but it would be expensive in terms of time and memory, because of the high-dimensionality of the latent variables $\boldsymbol{z}_\ell$, and of the size of the generative model neural network. Furthermore, first order optimization only provides a local minimum.

Fast approximate minimization in $\boldsymbol{z}$ To avoid the difficult optimization of $J_1(\boldsymbol{x}^{(n)}, \boldsymbol{z})$ w.r.t. $\boldsymbol{z}$, we propose instead to search for a $\boldsymbol{z}$ that minimizes the encoder approximation of the joint model (5.26). This approach is motivated by the fact that, if the encoder is well trained, we have $p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) = q(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y})$, and as a consequence,

$$\boldsymbol{z}^{(n+1)} = \arg\min_{\boldsymbol{z}} J_2(\boldsymbol{x}^{(n)}, \boldsymbol{z}). \tag{5.32}$$

is the solution of the subproblem (5.30). Solving (5.32) exactly is not straight-forward, because of the nested dependencies on the $\boldsymbol{z}_\ell$ terms in $J_2(\boldsymbol{x}^{(n)}, \boldsymbol{z})$. We introduce Algorithm 3, a sequential algorithm computing the exact solution of (5.32) under additional mild assumptions. In algorithm 3, the latent variables $\boldsymbol{z}_\ell$ are inferred sequentially, starting from $l = 0$ until $l = L - 1$. At each level l, the value of $\hat{\boldsymbol{z}}_l$ is defined as the optimal value $\boldsymbol{z}_\ell$ with respect to the previous latent groups $\hat{\boldsymbol{z}}_{<\ell}$ and the current image $\boldsymbol{x}^{(n)}$, without considering the influence of $\hat{\boldsymbol{z}}_l$ on subsequent terms $q_{\phi,\tau}\Big(\boldsymbol{z}_{l+k} | \hat{\boldsymbol{z}}_{<l+k}, \boldsymbol{x}^{(n)}\Big)$ on the total cost function $J_2(\boldsymbol{x}^{(n)}, \boldsymbol{z})$.

Practical implementation Algorithm 3 is convenient to implement, because the inference order follows the order of the top-down inference network (illustrated in Figure 4.3c). To implement Algorithm 3, we apply the HVAE encoder as we would for simply encoding an image, but, at each level l, instead of sampling $\boldsymbol{z}_\ell \sim q_\phi(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}, \boldsymbol{x})$, the value $\hat{\boldsymbol{z}}_\ell$ is

Algorithm 3 Hierarchical encoding with latent regularization to minimize (5.29) w.r.t. $\boldsymbol{z}$ for a fixed $\boldsymbol{x}$

Require: image $\boldsymbol{x}$; HVAE (ϕ, θ); temperature τ_ℓ; $\lambda_\ell = \frac{1}{\tau_\ell^2} - 1$
  for $0 \leq \ell < L$ **do**
    $S_q \leftarrow \Sigma_{\phi,\ell}^{-1}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})$; $m_q \leftarrow \mu_{\phi,\ell}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ ▷ Encoder
    $S_p \leftarrow \Sigma_{\theta,\ell}^{-1}(\boldsymbol{z}_{<\ell})$; $m_p \leftarrow \mu_{\theta,\ell}(\boldsymbol{z}_{<\ell})$ ▷ Prior
    %$\arg\min_{z_\ell} - \log q_{\phi,\tau}(\boldsymbol{z}_\ell | \boldsymbol{x}, \boldsymbol{z}_{<\ell})$
    $\boldsymbol{z}_\ell \leftarrow (S_q + \lambda_\ell S_p)^{-1}(S_q m_q + \lambda_\ell S_p m_p)$
  end for
  return $E_{\boldsymbol{\tau}}(\boldsymbol{x}) = (\hat{\boldsymbol{z}}_0, \hat{\boldsymbol{z}}_1, \cdots, \hat{\boldsymbol{z}}_{L-1})$

computed as the minimizer of $q_{\phi,\tau}(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}, \boldsymbol{x})$ with respect to $\boldsymbol{z}_\ell$. From the formulation of $q_{\phi,\tau}(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell}, \boldsymbol{x})$ (5.23) as a product of Gaussian densities, the minimizer is obtained as:

$$\hat{\boldsymbol{z}}_\ell = \left(\Sigma_{\phi,\ell}^{-1}(\hat{\boldsymbol{z}}_{<\ell}, \boldsymbol{x}) + \lambda_\ell \Sigma_{\theta,\ell}^{-1}(\hat{\boldsymbol{z}}_{<\ell})\right)^{-1} \tag{5.33}$$

$$\left(\Sigma_{\phi,\ell}^{-1}(\hat{\boldsymbol{z}}_{<\ell}, \boldsymbol{x})\, \mu_{\phi,\ell}(\hat{\boldsymbol{z}}_{<\ell}, \boldsymbol{x}) + \lambda_\ell \Sigma_{\theta,\ell}^{-1}(\hat{\boldsymbol{z}}_{<\ell})\, \mu_{\theta,\ell}(\hat{\boldsymbol{z}}_{<\ell})\right). \tag{5.34}$$

Hence, at each step, the minimizer can be viewed as a weighted average of the means of the Gaussian encoder term $q_\phi(\boldsymbol{z}_\ell | \boldsymbol{x}, \boldsymbol{z}_{<\ell})$ and the Gaussian prior term $p_\theta(\boldsymbol{z}_\ell | \boldsymbol{z}_{<\ell})$, with the interpolation weights depending on the covariance matrices and the temperatures τ_ℓ (through $\lambda_\ell = \frac{1}{\tau_\ell^2} - 1$). In the following, we denote as $\hat{\boldsymbol{z}} := E_{\boldsymbol{\tau}}(\boldsymbol{x})$ the output of the hierarchical encoding of Algorithm 3.

5.4.4 Analysis of the minimization in $\boldsymbol{z}$

Global minimum of $J_2(\boldsymbol{x}, \boldsymbol{z})$ In the following proposition, we derive sufficient guarantee for which algorithm 3 provides the exact solution to the problem (5.32). To that end, we need the following assumption on the volume of the covariance matrices of the HVAE model.

Assumption 5.1 (Volume-preserving covariances)**.** *The covariance matrices of the HVAE have constant determinant (not depending on $\boldsymbol{z}_{<\ell}$, although this constant may depend on the hierarchy level l)*

$$|\Sigma_{\phi,\ell}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})| = c_\ell(\boldsymbol{x}) \tag{5.35}$$

$$|\Sigma_{\theta,\ell}(\boldsymbol{z}_{<\ell})| = d_l \tag{5.36}$$

Proposition 5.2 (Algorithm 3 computes the global minimum of $J_2(\boldsymbol{x}, \boldsymbol{z})$ with respect to $\boldsymbol{z}$)**.** *Under Assumption 5.1, $J_2(\boldsymbol{x}, \boldsymbol{z})$ has a unique global minimum in $\boldsymbol{z}$, which corresponds to the output of algorithm 3:*

$$E_{\boldsymbol{\tau}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}} J_2(\boldsymbol{x}, \boldsymbol{z}) \tag{5.37}$$

Proof. To demonstrate this result, we first notice that the low temperature encoder conditionals $q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ are unormalized Gaussian probability density functions (PDF):

$$q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) = \frac{1}{E_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x})} \exp\left(-\frac{1}{2}||\boldsymbol{z}_\ell - \mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)||^2_{\Sigma^{-1}_{\phi,\tau,\ell}(\boldsymbol{z}_{<\ell},\boldsymbol{x})}\right), \tag{5.38}$$

with

$$E_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x}) = \left((2\pi)^{n_l}\left|\Sigma_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right|\right)^{\frac{1}{2}}\left((2\pi)^{n_l}\left|\Sigma_{\theta,\ell}\left(\boldsymbol{z}_{<\ell}\right)\right|\right)^{\frac{\lambda_\ell}{2}} Z_\ell, \tag{5.39}$$

$$\Sigma_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) = \left(\Sigma^{-1}_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) + \lambda_\ell\Sigma^{-1}_{\theta,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right)^{-1}, \tag{5.40}$$

$$\mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) = \Sigma_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\left(\Sigma^{-1}_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\mu_\phi\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) + \lambda_\ell\Sigma^{-1}_{\theta,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\mu_\theta\left(\boldsymbol{z}_{<\ell}\right)\right). \tag{5.41}$$

This comes from the definition of $q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ as the product of two Gaussian PDF (5.23), and the product of Gaussian PDF rule (see for instance [Bromiley, 2003, Toussaint, 2011]). By definition of $J_2(\boldsymbol{x},\boldsymbol{z})$, we have that[2]:

$$\arg\min_{\boldsymbol{z}} J_2(\boldsymbol{x},\boldsymbol{z}) = \arg\min_{\boldsymbol{z}} -\log q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) \tag{5.42}$$

$$= \arg\min_{\boldsymbol{z}} \sum_{\ell=0}^{L-1} -\log q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}). \tag{5.43}$$

Exploiting the formulation of $q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x})$ in equation (5.38), it follows that:

$$\begin{aligned}
-\log q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) &= \sum_{\ell=0}^{L-1} -\log q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) \\
&= \sum_{\ell=0}^{L-1} \log E_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x}) + ||\boldsymbol{z}_\ell - \mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)||^2_{\Sigma^{-1}_{\phi,\tau,\ell}(\boldsymbol{z}_{<\ell},\boldsymbol{x})} \\
&= \sum_{\ell=0}^{L-1} \log C_\ell + \underbrace{\log\left|\Sigma_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right| + \lambda_\ell \log\left|\Sigma_{\theta,\ell}\left(\boldsymbol{z}_{<\ell}\right)\right|}_{B_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x})} \\
&\qquad + \underbrace{||\boldsymbol{z}_\ell - \mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)||^2_{\Sigma^{-1}_{\phi,\tau,\ell}(\boldsymbol{z}_{<\ell},\boldsymbol{x})}}_{A_\ell(\boldsymbol{z}_\ell,\boldsymbol{z}_{<\ell},\boldsymbol{x})}.
\end{aligned} \tag{5.44}$$

Under the assumption of volume preserving covariance 5.1, $B_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x}) = \tilde{B}_\ell(\boldsymbol{x})$ is constant in $\boldsymbol{z}_{<\ell}$ for all l, and

$$\arg\min_{\boldsymbol{z}} -\log q_{\phi,\tau}(\boldsymbol{z}|\boldsymbol{x}) = \arg\min_{\boldsymbol{z}} \underbrace{\sum_{\ell=0}^{L-1} A_\ell(\boldsymbol{z}_\ell,\boldsymbol{z}_{<\ell},\boldsymbol{x})}_{A(\boldsymbol{z},\boldsymbol{x})}. \tag{5.45}$$

[2]For readability, we abuse notations in the following, by denoting $q_{\phi,\tau}(\boldsymbol{z}_0|\boldsymbol{z}_{<0},\boldsymbol{x}) = q_{\phi,\tau}(\boldsymbol{z}_0|\boldsymbol{x})$

Since $A_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{<\ell}, \boldsymbol{x}) \geq 0$, we have that

$$A(\boldsymbol{z}, \boldsymbol{x}) = \sum_{\ell=0}^{L-1} A_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{<\ell}, \boldsymbol{x}) \geq 0.$$

Also,

$$A_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{<\ell}, \boldsymbol{x}) = 0 \quad \text{iff} \quad \boldsymbol{z}_\ell = \mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell}, \boldsymbol{x}\right).$$

Therefore, the output of algorithm 3 $\boldsymbol{z}^\star = E_{\boldsymbol{\tau}}\left(\boldsymbol{x}\right)$, defined as:

$$\begin{cases} \boldsymbol{z}_0^\star & = \arg\min_{z_0} -\log q_{\phi,\tau}(\boldsymbol{z}_0|\boldsymbol{x}) = \mu_{\phi,\tau,0}(\boldsymbol{x}) \\ \boldsymbol{z}_l^\star & = \arg\min_{\boldsymbol{z}_\ell} -\log q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}^\star, \boldsymbol{x}) = \mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell}^\star, \boldsymbol{x}\right) \quad \text{for } l \in \{1, \ldots, L-1\} \end{cases} \tag{5.46}$$

satisfies by construction, $A_\ell(\boldsymbol{z}_\ell^\star, \boldsymbol{z}_{<\ell}^\star, \boldsymbol{x}) = 0$ for all $l \in \{0; \cdots : L-1\}$. Hence,

$$A(\boldsymbol{z}^\star, \boldsymbol{x}) = \sum_{\ell=0}^{L-1} A_\ell(\boldsymbol{z}_\ell^\star, \boldsymbol{z}_{<\ell}^\star, \boldsymbol{x}) = 0.$$

It follows that $\boldsymbol{z}^\star$ is a minimum of $J_2(\boldsymbol{x}, .)$. Additionally, for any $\boldsymbol{z} \neq \boldsymbol{z}^\star$, there exists $j \in \{1, \ldots, L-1\}$ such that $\boldsymbol{z}_j \neq \mu_{\phi,\theta,j}(\boldsymbol{z}_{<j}, \boldsymbol{x})$, and:

$$A(\boldsymbol{z}, \boldsymbol{x}) \geq A_j(\boldsymbol{z}_j, \boldsymbol{z}_{<j}, \boldsymbol{x}) > 0$$

Hence, $\boldsymbol{z}^\star = E_{\boldsymbol{\tau}}(\boldsymbol{x})$ is the unique global minimum of $J_2(\boldsymbol{x}, .)$. □

Discussion on assumption 5.1 (volume preserving covariance) We showed in proposition 5.2 that, under assumption 5.1, Algorithm 3 computes the global minimum of $J_2(\boldsymbol{x}, \boldsymbol{z})$ with respect to $\boldsymbol{z}$. When optimizing $\boldsymbol{z}_l$ in Algorithm 3, we only consider the impact of $\boldsymbol{z}_\ell$ on the distance to the Gaussian mean in $A(\boldsymbol{z}, \boldsymbol{x})$, while ignoring its impact on the covariance volumes in the subsequent levels in the terms $B_{l'}(\boldsymbol{z}_{<l'}, \boldsymbol{x})$, for $l' > l$. If the covariance volumes are constant as stated in assumption 1, the value of $\boldsymbol{z}_\ell$ has no impact on the covariance volumes of the subsequent levels, and algorithm 3 gives the global minimizer of $J_2(\boldsymbol{x}, .)$ with respect to $\boldsymbol{z}$. In practice, the HVAE model we use does not enforce the covariance matrices of $p(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$ and $q(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ to have constant volume. However, the experiment in Figure 5.1 shows that the variation of $B_{l+1}(\boldsymbol{z}_{<l+1})$ is negligible in front of $A_l(\boldsymbol{z}_\ell)$. Hence, we can expect algorithm 3 to yield a reasonable approximation of the minimum of $J_2(\boldsymbol{x}, \boldsymbol{z})$ with respect to $\boldsymbol{z}$. For future works, we could explicitly enforce assumption 1 in the HVAE design.

Minimization in $\boldsymbol{x}$ Like for the JPMAP energy (5.6), for a linear degradation model and a Gaussian decoder (5.1), the criterion $J_1(\boldsymbol{x}, \boldsymbol{z})$ in (5.18) is convex in $\boldsymbol{x}$ and its global

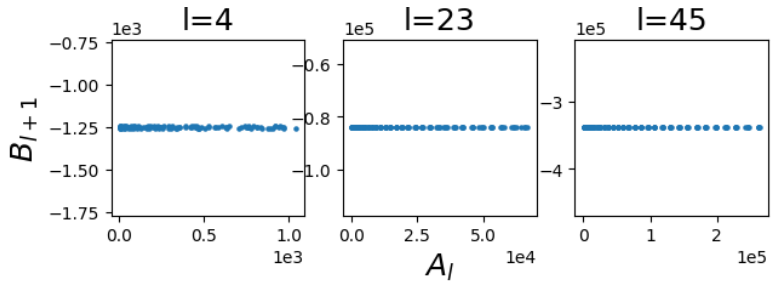


Figure 5.1 – Evolution of $B_{l+1} = \log|\Sigma^{-1}_{\phi,l+1}(\boldsymbol{z}_{<l+1}, \boldsymbol{x})| + \lambda_{l+1}\log|\Sigma_{\theta,l+1}(\boldsymbol{z}_{<l+1})|$ as a function of the distance $A_\ell = ||\boldsymbol{z}_\ell - \mu_{\phi,\tau,l}(\boldsymbol{z}_{<\ell}, \boldsymbol{x})||^2$.

minimum is:

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}} -\log p(\boldsymbol{y}|\boldsymbol{x}) - \log p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<L}) \tag{5.47}$$

$$= \arg\min_{\boldsymbol{x}} \frac{1}{2\sigma^2}\|A\boldsymbol{x} - \boldsymbol{y}\|^2 + (\boldsymbol{x} - \mu_\theta(\boldsymbol{z}_{<L}))^t \Sigma_\theta^{-1}(\boldsymbol{z}_{<L})(\boldsymbol{x} - \mu_\theta(\boldsymbol{z}_{<L})) \tag{5.48}$$

$$= \left(A^tA + \sigma^2\Sigma_\theta^{-1}\left(\boldsymbol{z}^{(n+1)}\right)\mathrm{Id}\right)^{-1}\left(A^t\boldsymbol{y} + \sigma^2\Sigma_\theta^{-1}\left(\boldsymbol{z}^{(n+1)}\right)\mu_\theta\left(\boldsymbol{z}^{(n+1)}\right)\right). \tag{5.49}$$

PnP-HVAE Our final algorithm, named PnP-HVAE, is presented in Algorithm 4. It alternates between an approximate minimization of $J_2(\boldsymbol{x}, \boldsymbol{z})$ w.r.t. $\boldsymbol{z}$ using algorithm 3, and an exact minimization of $J_1(\boldsymbol{x}, \boldsymbol{z})$ w.r.t. $\boldsymbol{x}$.

Algorithm 4 PnP-HVAE - Restoration by solving (5.18)

$k \leftarrow 0$; $res \leftarrow +\infty$; initialize $\boldsymbol{x}^{(0)}$
while $res > tol$ **do**
  % $\min_z J_2(\boldsymbol{x}^{(k)}, z)$ ▷ Optimize (5.29) w.r.t. $\boldsymbol{z}$ using Alg. 3
  $\boldsymbol{z}^{(k+1)} = E_{\boldsymbol{\tau}}(\boldsymbol{x}^{(k)})$
  % $\min_x J_1(\boldsymbol{x}, \boldsymbol{z}^{(k+1)})$ ▷ Optimize (5.18) w.r.t. $\boldsymbol{x}$
  $\boldsymbol{x}^{(k+1)} = \left(A^tA + \frac{\sigma^2}{\gamma^2}\mathrm{Id}\right)^{-1}\left(A^t\boldsymbol{y} + \frac{\sigma^2}{\gamma^2}\mu_\theta\left(\boldsymbol{z}^{(k+1)}\right)\right)$
  $res \leftarrow ||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}||$; $k \leftarrow k+1$
end while
return $\boldsymbol{x}^{(k)}$

5.5 Convergence analysis

We now analyse the convergence of Algorithm 4. Following the work of [Attouch et al., 2010], the alternate optimization scheme converges if $q_\phi(\boldsymbol{z}|x) = p_\theta(\boldsymbol{z}|\boldsymbol{x})$ and the sequential optimization scheme in Algorithm 3 actually solves $\min_{\boldsymbol{z}} J_2(\boldsymbol{x}, \boldsymbol{z})$. In practice, it is difficult

to verify if these hypotheses hold. We propose to theoretically study algorithm 4, and next verify empirically that the assumptions are met.

In section 5.5.1, we reformulate Algorithm 4 as a Plug-and-Play algorithm, where the HVAE reconstruction takes the role of the denoiser. Then we study in section 5.5.2 the fixed-point convergence of the algorithm. Finally, section 5.5.3 contains numerical experiments with the patchVDVAE architecture later presented in section 5.6.2. We empirically show that the patch architecture satisfies the aforementioned technical assumptions and then illustrate the numerical convergence and the stability of our alternate algorithm.

5.5.1 Plug-and-Play HVAE

In this section we make the assumption that the HVAE decoder is Gaussian with a constant variance on its diagonal, that is:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}\left(\boldsymbol{x}; \mu_\theta\left(\boldsymbol{z}\right), \gamma^2 I\right). \tag{5.50}$$

If the decoder distribution is not defined as in (5.50), we can replace the original encoder by a decoder distribution with constant variance. We rely on the proximal operator of a convex function f that is defined as $\text{prox}_f(\boldsymbol{x}) = \arg\min_{\boldsymbol{u}} f(\boldsymbol{u}) + \frac{1}{2}||\boldsymbol{x} - \boldsymbol{u}||^2$.

Proposition 5.3. *Assume the decoder is defined as in* (5.50)*. Denote HVAE*$(\boldsymbol{x}, \boldsymbol{\tau}) := \mu_\theta\left(E_{\boldsymbol{\tau}}\left(\boldsymbol{x}\right)\right)$*, and* $f(\boldsymbol{x}) = \frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2 \propto -\log p(\boldsymbol{y}|\boldsymbol{x})$*. Then the alternate scheme described in Algorithm 4 writes*

$$\boldsymbol{x}_{k+1} = \text{prox}_{\gamma^2 f}\left(HVAE\left(\boldsymbol{x}_k, \boldsymbol{\tau}\right)\right). \tag{5.51}$$

From relation (5.51), algorithm 4 is a Plug-and-Play Half-Quadratic Splitting method [Ryu et al., 2019] where the role of the denoiser is played by the reconstruction HVAE $\left(\boldsymbol{x}_k, \boldsymbol{\tau}\right)$. In practice, the proximal operator of the data-fidelity term of linear inverse problems $f(\boldsymbol{x}) = \frac{1}{2\sigma^2}||A\boldsymbol{x} - \boldsymbol{y}||^2$ can be computed efficiently for typical linear operator A such as the one involved in super-resolution or deblurring problems [Zhang et al., 2021]. We now derive from relation (5.51) sufficient conditions to establish the convergence of the iterations.

5.5.2 Fixed-point convergence

Let us denote T the operator corresponding to one iteration of (5.51):

$$\text{T}(\boldsymbol{x}) = \text{prox}_{\gamma^2 f}\left(\text{HVAE}\left(\boldsymbol{x}, \boldsymbol{\tau}\right)\right). \tag{5.52}$$

The Lipschitz constant of T can then be expressed as a function of f and the HVAE reconstruction operator HVAE $\left(\boldsymbol{x}_k, \boldsymbol{\tau}\right)$.

Proposition 5.4. *Assume that the decoder has a constant variance* $\Sigma_\theta^{-1}(\boldsymbol{z}) = \frac{1}{\gamma^2}\,\mathrm{Id}$ *for all* $\boldsymbol{z}$*; and the autoencoder with latent regularization is* $L_{\boldsymbol{\tau}}$*-Lipschitz,* i.e. $\forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^{\mathbf{n}}$*:* $||HVAE(\mathbf{u}, \boldsymbol{\tau}) - HVAE(\mathbf{v}, \boldsymbol{\tau})\,|| \leq L_{\boldsymbol{\tau}}||\mathbf{u} - \mathbf{v}||$*. Then, denoting as* $\lambda_{\min}$ *the smallest eigenvalue of* A^tA*, we have*

$$||\,T(\mathbf{u}) - T(\mathbf{v})|| \leq \frac{\sigma^2}{\gamma^2\lambda_{\min} + \sigma^2} L_\tau ||\mathbf{u} - \mathbf{v}||. \tag{5.53}$$

Proof. For a decoder with constant covariance $\Sigma_\theta^{-1}(\boldsymbol{z}) = \frac{1}{\gamma^2}\,\mathrm{Id}$, we have:

$$\mathrm{T}(\boldsymbol{x}) = \left(A^tA + \frac{\sigma^2}{\gamma^2}\,\mathrm{Id}\right)^{-1}\left(A^t\boldsymbol{y} + \frac{\sigma^2}{\gamma^2}\mu_\theta\left(E_{\boldsymbol{\tau}}\left(\boldsymbol{x}\right)\right)\right) \tag{5.54}$$

and then :

$$||T(\mathbf{u}) - T(\mathbf{v})|| \leq \left|\left|\left(A^tA + \frac{\sigma^2}{\gamma^2}\,\mathrm{Id}\right)^{-1}\right|\right| \frac{\sigma^2 L_\tau}{\gamma^2}||\mathbf{u} - \mathbf{v}||. \tag{5.55}$$

To conclude the proof, we use that for an invertible matrix M, $||M^{-1}|| = \frac{1}{\sigma_{\mathbf{min}}(\mathbf{M})}$, where $\sigma_{\mathbf{min}}(\mathbf{M})$ is the smallest eigenvalue of M. We also use the fact that α is an eigenvalue of $A^tA + \frac{\sigma^2}{\gamma^2}\,\mathrm{Id}$ if and only if $\alpha = \lambda + \frac{\sigma^2}{\gamma^2}$ for an eigenvalue $\lambda \geq 0$ of the positive definite matrix A^tA. □

Corollary 1. *If the Lipschitz constant of* $HVAE(\boldsymbol{x}_k, \boldsymbol{\tau})$ *verify* $L_{\boldsymbol{\tau}} < \frac{\gamma^2\lambda_{\min}+\sigma^2}{\sigma^2}$*, then iterations* (5.51) *converge.*

Proof. If $L_{\boldsymbol{\tau}} < \frac{\gamma^2\lambda_{\min}+\sigma^2}{\sigma^2}$, T is a contraction from proposition 5.4, that is:

$$||T(\mathbf{u}) - T(\mathbf{v})|| < ||\mathbf{u} - \mathbf{v}||. \tag{5.56}$$

Consequently, Banach theorem ensures the convergence of the iteration $\boldsymbol{x}_{k+1} = T(\boldsymbol{x}_k)$ to a fixed point of T. □

For problems such as inpainting or super-resolution, A is not full rank, and $\lambda_{min} = 0$. This implies that the HVAE need to be contractive ($L_{\boldsymbol{\tau}} < 1$) to ensure convergence to a fixed-point. On the other hand, for problems such as deblurring, A is full rank and $\lambda_m in > 0$.

Proposition 5.5 (Proof in appendix A.1.3)**.** $\boldsymbol{x}^\star$ *is a fixed point of* T *if and only if:*

$$\nabla f(\boldsymbol{x}^\star) = \frac{1}{\gamma^2}\left(HVAE(\boldsymbol{x}^\star, \boldsymbol{\tau}) - \boldsymbol{x}^\star\right). \tag{5.57}$$

Proposition 5.5 characterizes the solution of the PnP-HVAE algorithm, in the case where the HVAE reconstruction is a contraction. Under mild assumptions, the fixed point condition can be stated as a critical point condition

$$\nabla f(\boldsymbol{x}^*) + \nabla g(\boldsymbol{x}^*) = 0,$$

of the objective function $f(\boldsymbol{x}) + g(\boldsymbol{x}) = -\log p(\boldsymbol{y}|\boldsymbol{x}) - \log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$, where the tempered prior is the marginal $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) := \int p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z})d\boldsymbol{z}$ of the joint tempered prior defined in (5.14). As discussed in appendix A.1.4, this result follows from an interpretation of $\text{HVAE}(\boldsymbol{x}, \boldsymbol{\tau})$ as a MMSE denoiser. Tweedie's formula then provides the link between the right-hand side of equation (5.57) and ∇g.

5.5.3 Numerical convergence with PatchVDVAE

We illustrate the numerical convergence of Algorithm 4. We first analyse the Lipschitz constant of the HVAE reconstruction with the PatchVDVAE architecture proposed in section 5.6.2. Then we study the empirical convergence and stability of the algorithm.

Lipschitz constant of the HVAE reconstruction. Corollary 1 establishes the fixed point convergence of our proposed optimization algorithm under the hypothesis that the reconstruction with latent regularization is a contraction, *i.e.* $L_{\boldsymbol{\tau}} < 1$. We now show thanks to an empirical estimation of the Lipschitz constant $L_{\boldsymbol{\tau}}$ that our PatchVDVAE network empirically satisfies such a property when applied to noisy images. We present in Figure 5.2 the histograms of the ratios $r = ||\text{HVAE}(\mathbf{u}, \boldsymbol{\tau}) - \text{HVAE}(\mathbf{v}, \boldsymbol{\tau})||/||\mathbf{u} - \mathbf{v}||$, where $\mathbf{u}$ and $\mathbf{v}$ are natural images extracted from the BSD dataset and corrupted with white Gaussian noise. These ratios give a lower bound for the true Lipschitz constant $L_{\boldsymbol{\tau}}$. Although it is possible to set different temperature τ_ℓ at each level, we fixed a constant temperature amongst all levels to limit the number of hyperparameters. We realized tests for 3 temperatures $\tau \in \{0.6, 0.8, 0.99\}$, and 3 noise levels $\sigma \in \{0, 25, 50\}$. On clean images ($\sigma = 0$), the distribution of ratios in close to 1. This suggests that the HVAE is well trained and accurately models clean images. In some rare case, a ratio $r \geq 1$ is observed for clean images. This indicates that the reconstruction is not a contraction everywhere, in particular on the manifold of clean images. On noisy images $\sigma > 0$, the reconstruction behaves as a contraction, as the ratio $r < 1$ is always observed. Moreover, reducing the temperature of the latent regularization τ increases the strength of the contraction. This suggests that with the trained PatchVDVAE architecture, the hypothesis $L_{\boldsymbol{\tau}} < 1$ in Corollary 1 holds for noisy images.

Empirical convergence of Algorithm 4 We now illustrate the effectiveness of PnP-HVAE through comparisons with the optimization of the objective $J_1(\boldsymbol{x}, \boldsymbol{z})$ in (5.18) using the Adam algorithm [Kingma and Ba, 2014] for two learning rates $lr \in \{0.01, 0.001\}$.

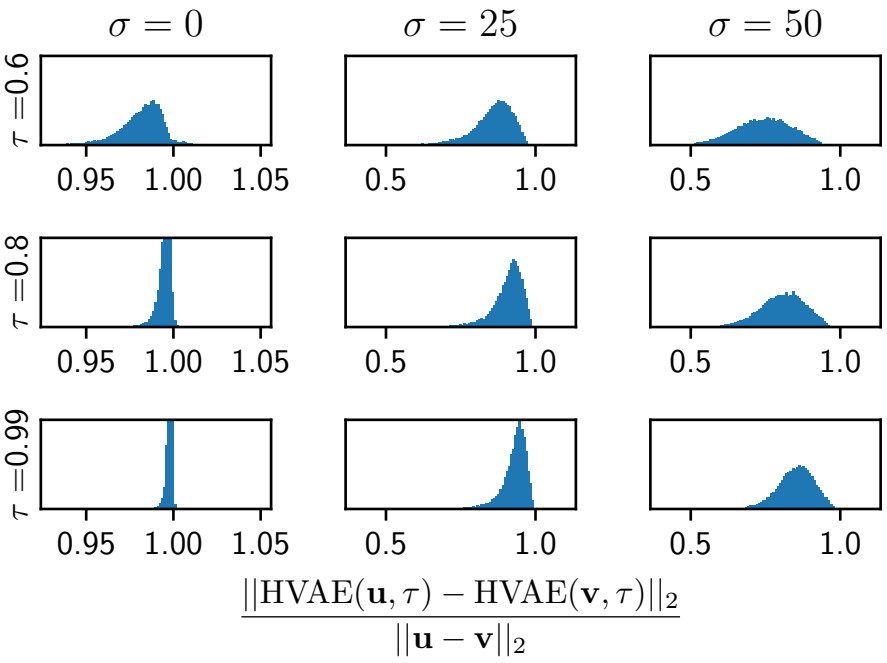


Figure 5.2 – Numerical estimation of the Lipschitz constant of PatchVDVAE reconstruction with different temperatures τ. We present the histogram of ratio values $\frac{||\text{HVAE}(\mathbf{u},\tau)-\text{HVAE}(\mathbf{v},\tau)||}{||\mathbf{u}-\mathbf{v}||}$, where $\mathbf{u}$ and $\mathbf{v}$ are natural images corrupted with white Gaussian noise of different standard deviations σ. For noisy images ($\sigma > 0$), the observed Lipschitz constant is always less than 1.

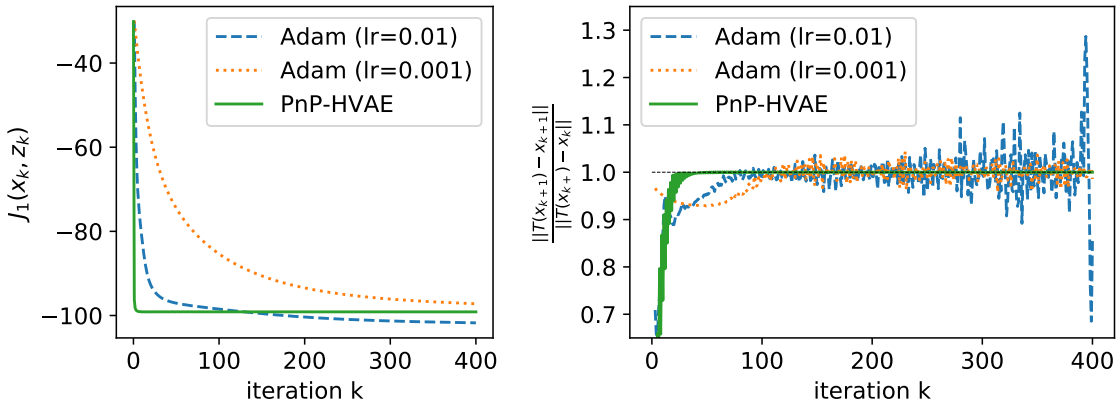


Figure 5.3 – Comparison of the convergence of PnP-HVAE algorithm 4 with respect to the baseline Adam optimizer, on a deblurring problem. Left (Convergence of the function value): PnP-HVAE converges faster to a minimum of the joint posterior $J_1(\boldsymbol{x}_k, \boldsymbol{z}_k)$ in (5.18). Right (Convergence of iterates $\boldsymbol{x}_k$): PnP-HVAE is more stable than Adam.

The left plot in Figure 5.3 shows that Adam is able to estimate a better minimum of J_1, whereas our alternate algorithm requires a smaller number of iterations to converge. On the other hand, as illustrated by the right plot in Figure 5.3, the use of Adam involves numerical instabilities. Oscillations of the ratio $L_k := \frac{||\mathrm{T}(\boldsymbol{x}_{k+1}) - \mathrm{T}(\boldsymbol{x}_k)||}{||\boldsymbol{x}_{k+1} - \boldsymbol{x}_k||}$ are even increased with larger learning rates, whereas our method provides a stable sequence of iterates. More importantly, we finally exhibit the better quality of the restorations obtained with our alternate algorithm on inpainting, deblurring and super-resolution of face images. In these experiments, we used the hierarchical VDVAE model [Child, 2020] trained on the FFHQ dataset [Karras et al., 2019]. Figure 5.4 (see 2nd and 4th columns) and Table 5.1 (PSNR, SSIM and LPIPS scores) illustrate that the quality of the images restored with our alternate optimization algorithm is higher than the ones obtained with Adam. This suggests that for image restoration purposes, our optimization method is able to find a more relevant fixed point of J_1 than the naive baseline based on Adam.

5.6 Image restoration results

We present in section 5.6.1 an application of PnP-HVAE on face images, using a pretrained state-of-the-art hierarchical VAE. Next, we study the application of our framework to natural images. To that end, we introduce in section 5.6.2 a patch hierachical VAE architecture, that is able to model natural images of different resolutions. In section 5.6.3, we provide deblurring, super-resolution and inpainting experiments to demonstrate the relevance of the proposed method.

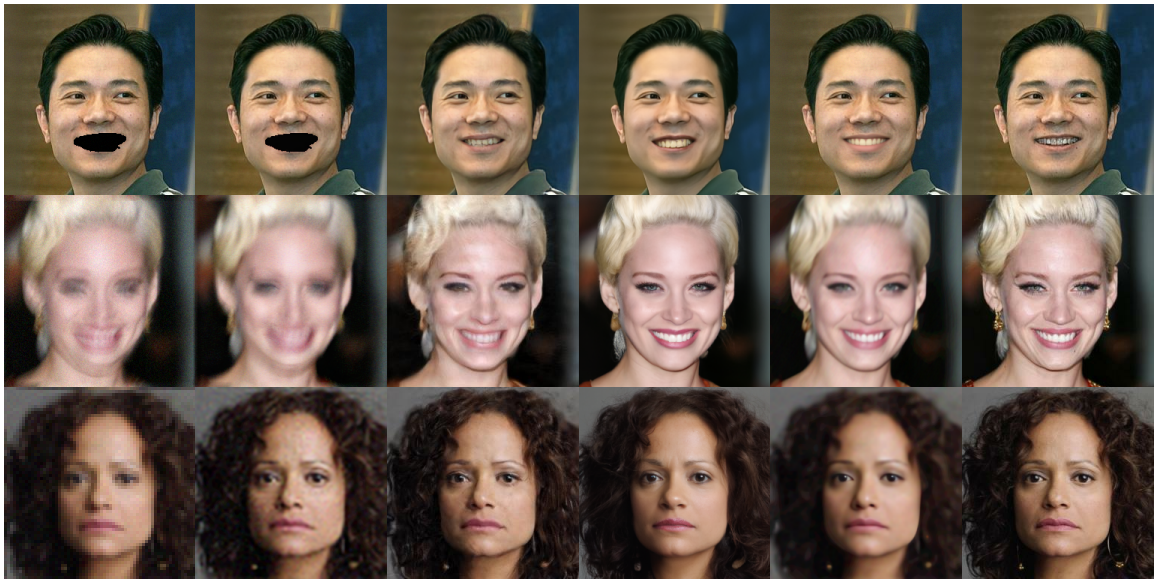


Figure 5.4 – Visual comparaison of image restoration methods based on deep generative models. We studied 3 tasks on face images: inpainting (top), deblurring (middle), super-resolution (bottom). Contrary to the optimization of the objective (5.18) with Adam, our alternate algorithm generates realistic results, on par with ILO [Daras et al., 2021], while remaining consistant with the observation.

5.6.1 Face Image restoration (FFHQ)

VDVAE model for face images We first demonstrate the effectiveness of PnP-HVAE on highly structured data, by performing super-resolution and deblurring on images of human faces. Latent variable generative models can accurately model structured images such as face images [Karras et al., 2019, Vahdat and Kautz, 2020, Child, 2020, Kingma and Dhariwal, 2018], and then be used to produce high quality restoration of such data. In our experiments, we use the VDVAE model of [Child, 2020], pre-trained on the FFHQ dataset [Karras et al., 2019], as our hierarchical VAE prior. VDVAE has $L = 66$ latent variable groups in its hierarchy and generates images at resolution 256×256.

Experimental setting For super-resolution, the degradation model corresponds to the application of a Gaussian low-pass filter followed by a $\times 4$ sub-sampling, and the addition of a Gaussian white noise with $\sigma = 3$. For deblurring, we considered motion blur and Gaussian kernels, both with a noise level $\sigma = 8$. Although VDVAE was trained on FFHQ, we evaluate our method on a subset of CelebA dataset [Liu et al., 2018], because the models used in the compared methods were not trained on the same train-test split of FFHQ. Specifically, we evaluate the methods on a subset of 100 images from the CelebA dataset.

		PSNR↑	SSIM↑	LPIPS↓	time (s)
SR ×4 $\sigma = 3$	Adam	28.56	0.75	0.38	<u>26</u>
	ILO	<u>28.80</u>	<u>0.78</u>	**0.17**	34
	PnP-HVAE	**29.32**	**0.82**	0.28	**15**
	DPS	27.53	0.76	<u>0.21</u>	153
Deblurring (motion) $\sigma = 8$	Adam	24.37	0.66	0.37	<u>12</u>
	ILO	<u>29.01</u>	<u>0.80</u>	<u>0.20</u>	34
	PnP-HVAE	**30.40**	**0.84**	**0.16**	**10**
	DPS	28.70	<u>0.80</u>	0.23	142
Deblurring (Gaussian) $\sigma = 8$	Adam	28.59	0.78	<u>0.23</u>	<u>12</u>
	ILO	29.12	0.79	**0.17**	34
	PnP-HVAE	**30.81**	**0.86**	0.24	**10**
	DPS	<u>29.14</u>	<u>0.81</u>	0.23	142

Table 5.1 – Quantitative evaluation on face restoration. Best results in **bold**, second best <u>underlined</u>.

We evaluate the performance of the restoration by measuring the distance of the restored image with the ground truth, using three distortion metrics. Namely, we use the peak signal-to-noise-ratio (PSNR) to measure the pixel-wise distortion, the Structural Similarity Index which measures the structural distortion, and the Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al., 2018b] which quantifies high-level perceptual distortion.

Compared methods We compare PnP-HVAE with two restoration methods based on different classes of generative models, namely the intermediate layer optimization algorithm (ILO) [Daras et al., 2021] and the diffusion posterior sampling method (DPS) [Chung et al., 2023]. ILO is a GAN inversion method which optimizes the image latent code along with the intermediate layer representation of a StyleGAN2 generative network [Karras et al., 2020] to generate an image consistent with a degraded observation. DPS uses denoising diffusion probabilistic model [Song et al., 2020, Ho et al., 2020] as a prior, and produces a sample from the posterior by conditioning each iteration of the sampling process on $\boldsymbol{y}$. We use the official implementation of ILO, along with a StyleGAN2 model that was trained for 550k iterations on images of resolution 256×256 from FFHQ [Seonghyeon, 2020]. For DPS, we use the official implementation as well. We provide additional details on the choice of hyperparameters for the concurrent methods in appendix A.2.2.

Results We provide a quantitative comparisons of the evaluated methods in Table 5.1, along with a visual comparison of the results in Figure 5.4. PnP-HVAE has the best PSNR and SSIM results for all the considered restoration tasks, and it also has the best perceptual results on motion deblurring. By jointly optimizing the image and its latent variable,

PnP-HVAE provides results that are both realistic and consistent with the degraded observation. On the other hand, ILO only optimizes on an extended latent space. This method generates sharp and realistic images with better LPIPS scores, but the results lack of consistency with respect to the observation, which explains the overall lower PSNR performance. DPS produces highly realistic samples (see Figure 5.4), but because DPS produces samples from the posterior, it is disadvantaged in terms of distortion metrics. DPS is also limited by its long inference time, as it requires one network function evaluation and one backpropagation operation through the network at each of the 1000 sampling steps required to generate one image.

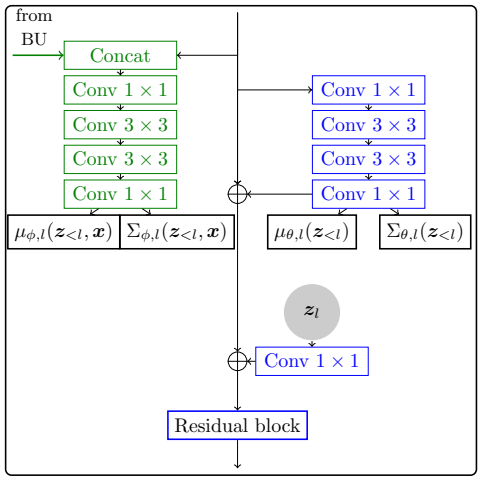
5.6.2 PatchVDVAE: a HVAE for natural images

Genericity issues of deep generative models Available generative models in the literature operate on images of fixed resolutions, and are fit on object-centric datasets, such as images of human faces [Kingma and Dhariwal, 2018, Child, 2020, Vahdat and Kautz, 2020, Karras et al., 2019], or ImageNet classes [Brock et al., 2018, Dhariwal and Nichol, 2021, Song et al., 2020, Luhman and Luhman, 2022]. Fitting an unconditional model on natural images appears to be a more difficult task, as their resolution can change, and their content is highly diverse. The complexity of the problem can be reduced by learning a prior model on patches of reduced dimension. For image restoration problems, the patch model can be reused on images of higher dimensions [Zoran and Weiss, 2011a, Prost et al., 2021, Altekrüger et al., 2022]. When the model is a full CNN, the prior on the set of the patches can be computed efficiently by applying the network on the full image [Prost et al., 2021].

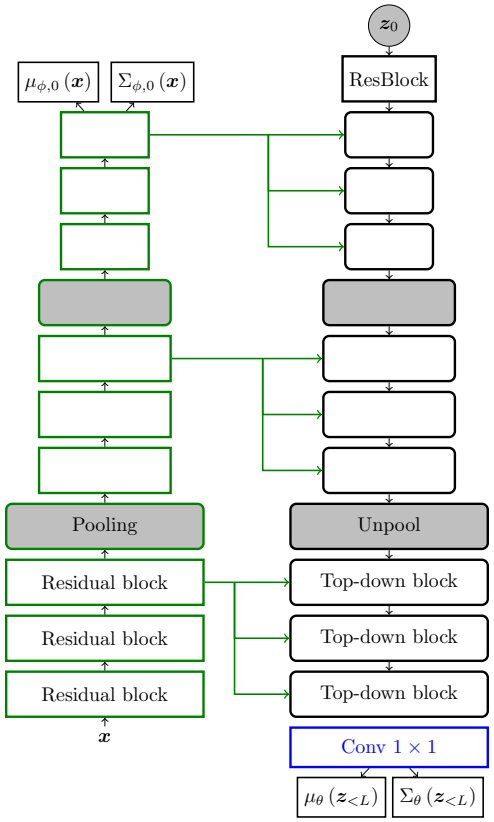
Fully convolutional HVAE We introduce patchVDVAE, a fully convolutional hierarchical VAE. Contrary to existing HVAE models whose resolution is constrained by the constant tensor at the input of the top-down block, patchVDVAE can generate images of different resolutions by controlling the dimension of the input latent. This amounts to defining a prior on patches whose dimension corresponds to the receptive field of the VAE. A similar model is used for image denoising in [Prakash et al., 2021].

PatchVDVAE architecture We provide an illustration of the architecture of PatchVDVAE in Figure 5.5. We use the same bottom-up and top-down blocks as VDVAE [Child, 2020], and replace the constant trainable input in the first top-down block by a latent variable, to make the model fully convolutional. More details on PatchVDVAE architecture are provided in appendix A.2.1.

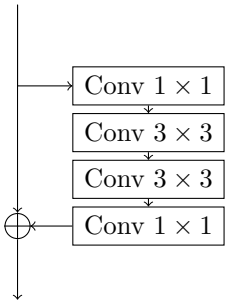
Training The training dataset is composed of 128×128 patches extracted from a combination of DIV2K [Agustsson and Timofte, 2017] and Flickr2K [Lim et al., 2017]



Top-down block



Full autoencoder architecture



Residual block

Figure 5.5 – Structure of the PatchVDVAE architecture. For clarity, we omit the non-linearity after each convolution.

datasets. We perform data augmentation by extracting patches at 3 resolutions: HR-images and $\times 2$ and $\times 4$ downscaled images. The model is trained for 7.10^5 iterations with a batch size of 64. Following the recommendation of [Hazami et al., 2022], we use Adamax optimizer with an exponential moving average and gradient smoothing of the variance.

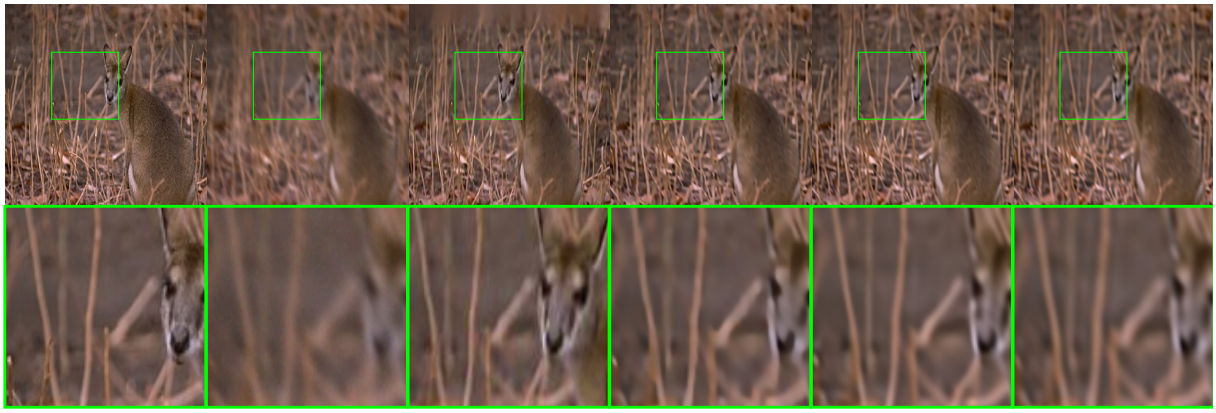
We set the decoder model to be a Gaussian with diagonal covariance, as in [Luhman and Luhman, 2022]. PatchVDVAE is fully convolutional and can generate images of dimensions that are multiple of 64 as illustrated by Figure 5.6.

Figure 5.6 – Left: 64×64 patches samples from our patchVDVAE model trained on patches from natural images. Right: PatchVDVAE is fully convolutional and it can generate images of higher resolution (here: 128×128).

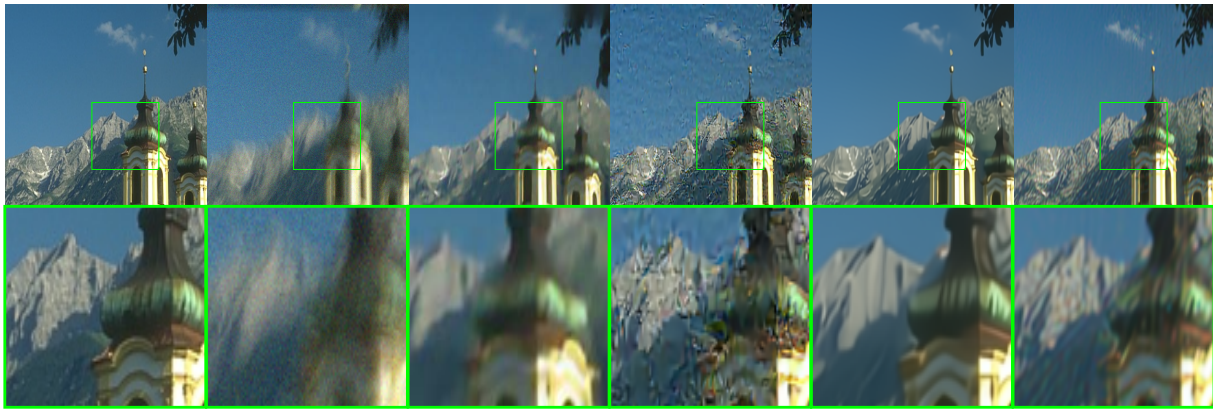
5.6.3 Natural images restoration

We evaluate PnP-HVAE on natural image restoration with patchVDVAE. For each task, we report the average value of the PSNR, the SSIM, and the LPIPS metrics on 20 images from the test set of the BSD dataset [Martin et al., 2001].

Image deblurring In the experiments, we consider 2 Gaussian kernels and 2 motion blur kernels from [Levin et al., 2009], with 3 different noise levels $\sigma \in \{2.55, 7.65, 12.75\}$. As a baseline we consider EPLL [Zoran and Weiss, 2011a], which learns a prior on image patches with a Gaussian mixture model. We also compare PnP-HVAE with PnP-MMO and GS-PnP, two competing convergent Plug-and-Play methods based on CNN denoisers. PnP-MMO [Pesquet et al., 2021] restricts the denoiser to be a contraction in order to guarantee the convergence of the PnP forward-backard algorithm. GS-PnP [Hurault et al., 2022] considers a gradient step denoiser and reaches state-of-the-art performances.We set the temperature τ in our method as 0.95, 0.8 and 0.6 for noise levels 2.55, 7.65 and 12.75 respectively, and we let it run for a maximum of 50 iterations. For the three compared methods we use the official implementations and pre-trained models provided by the respective authors. Details on the choice of hyperparameters for the concurrent methods are provided in appendix A.2.2. Visually, PnP-HVAE provides good deblurring results (Figure 5.7). For large noise levels, PnP-HVAE outperforms EPLL and PnP-MMO in terms of distortion metrics (Table 5.2), while GS-PnP provides the best results.



(a) Gaussian blur, $\sigma = 2.55$



(b) Motion blur, $\sigma = 7.65$

Figure 5.7 – Natural image deblurring

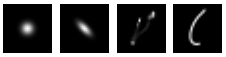
σ	Method	PSNR↑	SSIM↑	LPIPS↓
2.55	PnP-HVAE	27.75	0.79	0.31
	GS-PNP [Hurault et al., 2022]	**29.59**	**0.84**	**0.22**
	EPLL [Zoran and Weiss, 2011a]	26.49	0.71	0.36
	PnP-MMO [Pesquet et al., 2021]	<u>29.50</u>	<u>0.83</u>	<u>0.20</u>
7.65	PnP-HVAE	<u>26.36</u>	<u>0.72</u>	<u>0.40</u>
	GS-PNP [Hurault et al., 2022]	**27.33**	**0.77**	**0.31**
	EPLL [Zoran and Weiss, 2011a]	24.04	0.66	0.45
	PnP-MMO [Pesquet et al., 2021]	25.34	0.69	0.34
12.75	PnP-HVAE	<u>25.12</u>	**0.73**	<u>0.47</u>
	GS-PNP [Hurault et al., 2022]	**26.32**	**0.73**	**0.37**
	EPLL [Zoran and Weiss, 2011a]	23.28	0.61	0.51
	PnP-MMO [Pesquet et al., 2021]	22.42	0.53	0.54
	Blur and motion kernels			

Table 5.2 – Comparison of PnP-HVAE and other restoration methods on deblurring. Results are averaged on 4 kernels.

	PSNR↑	SSIM↑	LPIPS↓
PnP-HVAE	**29.54**	**0.93**	**0.06**
GS-PNP	28.52	**0.93**	0.09
EPLL	<u>29.16</u>	**0.93**	**0.06**

Table 5.3 – Quantitative evaluation for inpainting on BSD.

Image inpainting Next we consider the task of noisy image inpainting. We compose a test-set of 10 images from the validation set of BSD [Martin et al., 2001] and we create masks by occluding diverse objects of small size in the images. A Gaussian white noise with $\sigma = 3$ is added to the images. As a comparaison, we still consider GS-PnP and EPLL. For PnP-HVAE, the temperature is set to $\tau = 0.6$, and the algorithm is run for a maximum of 200 iterations, unless the residual $||\boldsymbol{x}_{k+1} - \boldsymbol{x}_k||$ is on a plateau. We provide on Table 5.3 the distortion metrics with the ground truth, as well as a visual comparison on Figure 5.8. With its hierarchical structure, PnP-HVAE outperforms the compared methods, in terms of PSNR, SSIM, and LPIPS metrics.

Effect of the temperature. PnP-HVAE gives control of the temperature of the prior over the latent space. In Figure 5.9, we illustrate that reducing the temperature increases the strength of the regularization prior. In this example the tuning $\tau = 0.7$ produces the best performance.

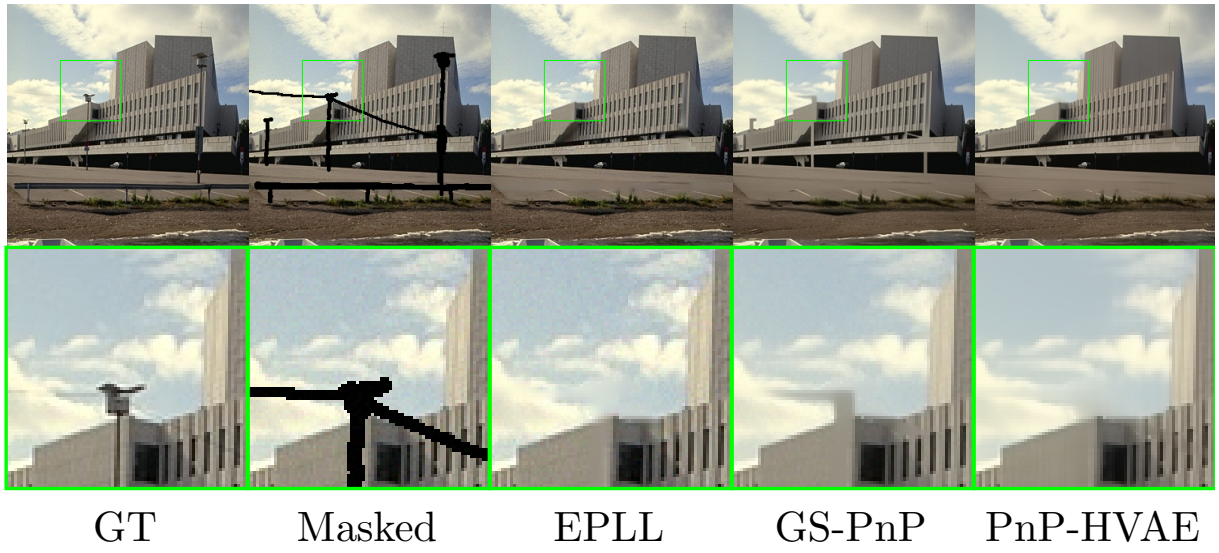


Figure 5.8 – Natural image inpainting

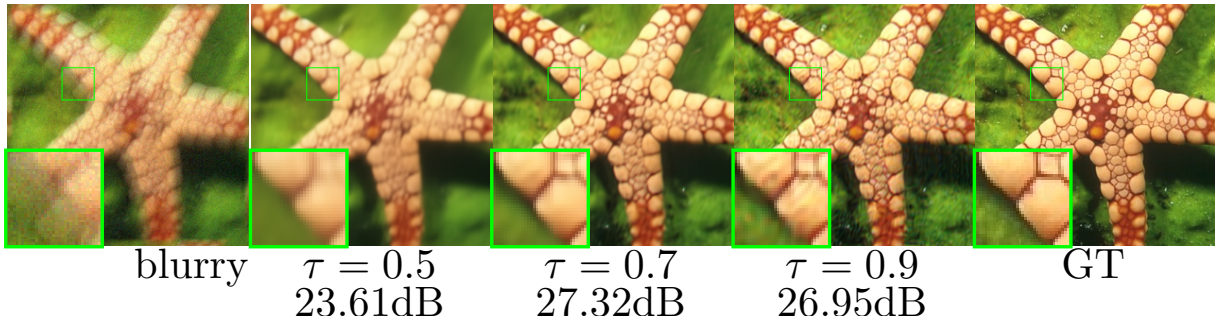


Figure 5.9 – Effect of the temperature in PnP-VAE on a deblurring problem, with $\sigma = 7.65$.

Effect of the number of latent groups We study the effect of the number of latent groups L on the hierarchical model on the restoration performance. It has been observed that HVAEs outperform non-hierarchical VAEs in terms of likelihood score [Sønderby et al., 2016], and that increasing the number of latent groups in the hierarchy improves the modelling performance of HVAE for a fixed number of parameters [Child, 2020]. Therefore we can expect that the gain in modelling performance due to a higher L translates into a gain in restoration performance using our method. We train different patchVDVAE models, with different numbers of latent groups L. In order to keep the number of trainable parameters constant, we replace stochastic top-down blocks with deterministic blocks in our network with the higher L value ($L = 36$). We evaluate the different models on image deblurring, using the same experimental settings as the one described in subsection 5.6.3. The results in Table 5.4 show that increasing the number of stochastic groups (L) has a positive effect on the evidence lower bound (B.12) evaluate on the test set, up to $L = 18$,

Table 5.4 – Effect of the number L of latent groups on the restoration performance, measured in PSNR (dB), for image deblurring. We observed similar trends for the LPIPS and SSIM metrics.

	$L = 6$	$L = 12$	$L = 18$	$L = 36$
$\sigma = 2.55$	27.25	**27.87**	<u>27.82</u>	27.71
$\sigma = 7.65$	26.10	26.41	**26.74**	<u>26.51</u>
$\sigma = 12.75$	24.78	25.16	**25.57**	<u>25.27</u>
ELBO↑(val)	-1.24	-1.14	$\mathbf{-1.10}$	$\mathbf{-1.10}$

and that a better evidence lower bound correlates with a better restoration performance.

5.7 Conclusion

We proposed PnP-HVAE, a method using hierarchical variational autoencoders as a prior to solve image inverse problems. Motivated by an alternate optimization scheme, PnP-HVAE exploits the encoder of the HVAE to avoid backpropagating through the generative network. We derived sufficient conditions on the HVAE model to guarantee the convergence of the algorithm. We have verified empirically that PnP-HVAE satisfies those conditions. By jointly optimizing over the image and the latent space, PnP-HVAE produces realistic results that are more consistent with the observation than GAN inversion on a specialized dataset. PnP-HVAE can also restore natural images of any size using our PatchVDVAE model trained on natural images patches.

On natural images, the restoration quality of PnP-HVAE is still below the performance of recent PnP methods based on deep denoisers. Existing denoisers used in the compared PnP methods are the product of numerous research iterations, whereas HVAEs trained on natural images are less than two years old [Prakash et al., 2021], and their use for PnP methods is proposed for the first time in this work. Therefore, we postulate that there is much room for future improvements on the quality of HVAE models for natural images, that would translate to better restoration performance.

Chapter 6

Diverse super-resolution with pretrained hierarchical variational autoencoders

In this chapter we investigate the problem of generating samples from the posterior distribution of an image inverse problem, with a specific focus on image super-resolution. Current methods in the literature either involve training a conditional generative model from scratch, or reuse a pretrained unconditional generative model within an expensive iterative sampling procedure. We propose to combine the best of both worlds, by developing a method sharing the sample quality of powerful unconditional deep generative models, while having the computational efficiency of fast conditional normalizing flow based methods. Our approach relies on training a lightweight stochastic encoder to encode low-resolution images in the latent space of a pretrained generative model. At inference, we combine the low-resolution encoder and the pretrained generative model to super-resolve an image. The stochastic nature of both the low-resolution encoder and the high-resolution generative decoder enables us to produce diverse highly-realistic samples. Specifically, we propose to reuse VDVAE, a hierarchical variational autoencoder, as we found that the hierarchical latent representation learned by VDVAE is well suited for our task. Furthermore, we show that the low-resolution encoder can be trained efficiently by exploiting VDVAE's expressive hierarchical encoder. We demonstrate the ability of our method to produce high-quality diverse super-resolved samples in a fast manner, on the problem of human face super-resolution.

6.1 Introduction

Single image super resolution Single image super resolution (SISR) is the task of retrieving a high-resolution (HR) image from a low-resolution (LR) observation. SISR is a

one-to-many problem as, for each LR image, there exist many HR images that are both consistent with the low-resolution one and look realistic. A common way to solve this ill-posed inverse problem (see [Wang et al., 2020] and references therein) is to estimate a regression model on paired data [Dong et al., 2014, Dong et al., 2016, Haris et al., 2018, Ledig et al., 2017, Wang et al., 2018]. However, it is not possible to explore all the potential solutions of the SR problem with regression based methods, as they only provide one single solution. This is a main issue if the provided *single* solution is not relevant enough or satisfying for the user. This work overcomes this limitation by providing diverse high-resolution solutions for each low resolution image.

Diverse image super resolution Recent works follow the *diverse* SISR paradigm, where the objective is to model the distribution of the plausible HR images conditioned on a LR image. We can distinguish two types of approaches for diverse super-resolution. First, direct methods, that only need one network evaluation to produce one sample. Those methods are based on conditional normalizing flows [Lugmayr et al., 2020,Liang et al., 2021], conditional GANs [Bahat and Michaeli, 2020] or conditional VAEs [Liu et al., 2020, Hyun and Heo, 2020, Chira et al., 2022, Gatopoulos et al., 2020]. Second, iterative methods aim at defining more expressive models, by relying on sequential sampling algorithms. Those methods are based on denoising diffusion models [Choi et al., 2021, Kawar et al., 2022b,Saharia et al., 2021b] or MCMC algorithms [Laumont et al., 2022]. Iterative methods can produce high-quality samples, but this comes at the cost of a high computational cost, since each iteration requires one (deep) network evaluation.

Deep generative prior for diverse SR As discussed in the previous chapters, unconditional generative models [Goodfellow et al., 2014a, Kingma and Welling, 2013, Rezende and Mohamed, 2015] provide a strong prior about the data distribution that can be incorporated as a prior to regularize ill-posed image inverse problems [Bora et al., 2017, Menon et al., 2020, Harvey et al., 2022]. In this work, we follow this paradigm as we propose to use a trained VDVAE network, a deep hierarchical VAE, to perform diverse super-resolution. As discussed in chapter 4, hierarchical VAEs reach state-of-the-art results among VAEs for image modelling [Child, 2020, Vahdat and Kautz, 2020]. Recent studies show that deep hierarchical variational autoencoders [Child, 2020, Vahdat and Kautz, 2020] can reach an impressive quality for image generation, while learning a latent variable representation that tends to separate the low-frequency information from the high frequency details of the generated image. Since image super-resolution is the task of recovering high-frequency details from the low-frequency information contained within a LR image, we postulate that the latent hierarchy learned by a deep hierarchical VAE can be repurposed to perform diverse image super-resolution.

Objectives In this chapter we target the following questions:

- Does the hierarchical latent representation learned by hierarchical VAEs effectively separate the image low-frequency information contained within a LR image from the high frequency details ? In particular, we study the latent representation learned by VDVAE [Child, 2020].
- Given a hierarchical VAE that separates low-frequency information and high-frequency details, how can we repurpose this VAE to perform diverse image super-resolution?

Contributions We make the following contributions:

- We study the hierarchical latent representation learned by VDVAE, and we empirically demonstrate that the low-frequency information contained within LR images is almost fully controlled by a subset of latent groups at the top of the latent hierarchy.
- We design a diverse super-resolution method that takes advantage of the specific structure of VDVAE latent representation. Specifically, we propose to combine an encoder trained on low-resolution images with VDVAE generative model to generate diverse super-resolved samples.
- We demonstrate the effectiveness of our model on face super-resolution, with upscaling factors x4, x8.

Overview of the chapter Section 6.2 provides context by reviewing related works, and the necessary technical background is introduced in Section 6.3. Then we study in Section 6.4 the properties of the latent representation learned by VDVAE [Child, 2020]. Building on those findings, we develop in Section 6.5 a diverse super-resolution method exploiting the property of its hierarchical latent representation. On a theoretical side, we derive a criterion to estimate the expected consistency error of the super-resolution model as a function of the number of predicted latent groups. In Section 6.6, we detail the practical implementation details, and provide results obtained with our proposed method on FFHQ dataset [Karras et al., 2019], with upsampling factors x4, x8 and

6.2 Related works

Conditional latent variable generative models To alleviate for the lack of diversity of end-to-end restoration methods, a new trend for designing diverse restoration methods has appeared in recent years. Diverse restoration methods are implemented as conditional latent variables generative models, such as conditional normalizing flows [Ardizzone et al., 2019, Lugmayr et al., 2020, Liang et al., 2021], conditional GANs [Bahat and Michaeli, 2020, Ohayon et al., 2021], conditional VAEs [Deshpande et al., 2017, Harvey et al., 2022]

and conditional diffusion models [Saharia et al., 2021b, Saharia et al., 2021a]. We also design a conditional latent variable generative model, but unlike previous works, we will build this model on top of a pretrained unconditional generative model.

Image restoration with pretrained generative models Another topic related to our study is the use of pretrained generative models to perform image restoration. [Bora et al., 2017, Menon et al., 2020], propose to restore an image by finding the latent code of a GAN [Goodfellow et al., 2014a] that generates an image consistent with the degraded observation. [Holden et al., 2022] propose a strategy based on MCMC to sample from the distribution of a VAE latent codes consistent with a degraded observation. Similarly, [González et al., 2022] jointly estimate the image and its latent code given in a VAE latent space, using an alternate optimization algorithm. Another strategy is to reuse denoising diffusion model by conditioning the reverse diffusion process on a degraded observation [Kawar et al., 2022b, Choi et al., 2021]. These methods are unsupervised, as they only require the knowledge of the forward degradation model. Their inference is nevertheless time-consuming, as they necessitate sampling or iterative optimization algorithms. Our work takes inspiration in the IPA framework of [Harvey et al., 2022], where restoration is performed with an encoder trained to encode degraded images in VDVAE latent space.

Image restoration with VAE Several works on image super-resolution using VAE have been proposed. [Hyun and Heo, 2020] proposes to train a conditional VAE for image super-resolution with a shared latent space between the HR and LR images. The quality of the super-resolved image is nevertheless limited by the expressivity of the simple (non-hierarchical) generative model. In [Gatopoulos et al., 2020], a 2-level hierarchical generative model is trained so that the first latent group encodes the low-frequency information and the second group the high-frequency details. In a concurrent work, [Chira et al., 2022] proposes a deep conditional hierarchical VAE architecture based on VDVAE model. Similar to us, they initialize the weight of the top-down path with the pretrained VDVAE weights, but, unlike our work, the weights of the top-down path are not frozen during training. On the topic of unsupervised denoising, [Prakash et al., 2021] exploits the ability of the hierarchical VAEs to separate the low-frequency information from the high-frequency details to denoise images.

6.3 Preliminaries

6.3.1 Diverse super-resolution

In this work we tackle the problem of diverse super-resolution: given a low-resolution (LR) image $\boldsymbol{y} \in \mathbb{R}^m$, our goal is to synthetize high-resolution (HR) images $\boldsymbol{x} \in \mathbb{R}^n$ that are

both *realistic* and *plausible* with respect to the observed $\boldsymbol{y}$. The degradation process we consider writes:

$$\boldsymbol{y} = H_s \boldsymbol{x}, \tag{6.1}$$

where H_s is a linear operator corresponding to the composition of a low-pass filter and a subsampling operation with downsampling factor s. The goal of super-resolution is to recover the high-frequency details of $\boldsymbol{x}$ erased by the degradation process (6.1). Image super-resolution is an ill-posed inverse problem, as there can be many HR images consistent with an LR image, *i.e.*, that satisfy (6.1). The objective of diverse super-resolution is to sample from the posterior distribution $p(\boldsymbol{x}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})$. The data-fidelity term $p(\boldsymbol{y}|\boldsymbol{x})$, dealing with the plausibility of the reconstruction, is given by the degradation model (6.1). On the other hand, the prior distribution term modeling realistic high-resolution images, $p(\boldsymbol{x})$, is unknown, but it can be learned with deep generative models.

VDVAE

In this chapter, we investigate the parameterization of $p(\boldsymbol{x})$ using a hierarchical VAE model. In particular, we will use the VDVAE model [Child, 2020], that we already used in chapter 5 for face image restoration. We refer the reader to chapter 4 for a detailed introduction on hierarchical VAEs and VDVAE. For the sake of completeness, we briefly introduce them again. VDVAE define a hierarchical generative model of the form:

$$p_\theta(\boldsymbol{z}, \boldsymbol{x}) = p_\theta(\boldsymbol{z}_0) \prod_{\ell=1}^{L-1} p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}) p_\theta(\boldsymbol{x}|\boldsymbol{z}), \tag{6.2}$$

where we denote $\boldsymbol{z}_{<\ell} = (\boldsymbol{z}_0, \cdots, \boldsymbol{z}_{l-1})$ and $\boldsymbol{z} = (\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{L-1})$. Each latent group is a 3-dimensional tensor $\boldsymbol{z}_\ell \in \mathbb{R}^{C^l \times H^l \times W^l}$, where C^l is the number of channels and (H^l, W^l) are the spatial dimensions. The conditional priors are set as multivariate Gaussian distributions with diagonal covariances:

$$p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}) = \mathcal{N}\left(\boldsymbol{z}_\ell; \mu_\theta\left(\boldsymbol{z}_{<\ell}\right), \Sigma_\theta\left(\boldsymbol{z}_{<\ell}\right)\right), \tag{6.3}$$

where $\mu_\theta\left(\boldsymbol{z}_{<\ell}\right)$ and $\Sigma_\theta\left(\boldsymbol{z}_{<\ell}\right)$ are parameterized by residual blocks. VDVAE inference network is composed of a deterministic bottom-up path, followed by a top-down path [Sønderby et al., 2016], sharing parameters with the generative model $p_\theta(\boldsymbol{z}, \boldsymbol{x})$. The inference network $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ infers the latent groups in the same order as in the generative model:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}_0|\boldsymbol{x}) \prod_{\ell=1}^{L-1} q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x}) \tag{6.4}$$

each conditional of the inference model $q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell}, \boldsymbol{x})$ is also set as a multivariate Gaussian distribution with diagonal covariance. Once again, we refer the reader to chapter 4 for more details about the network architecture.

6.4 Analysing the hierarchical latent representation of VDVAE

6.4.1 What information is encoded within each latent group?

Isolating the effect of the latent group The generative model of VDVAE provides a rich hierarchical latent representation of images. The effect of each latent group of the hierarchy on the generated image can be visualized by sampling images from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ while keeping $\boldsymbol{z}_{<k}$ fixed, for different values of k. Attributes that are common to all samples from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k+1})$, but not to all samples from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$, are most likely to be encoded in the latent group $\boldsymbol{z}_k$.

VDVAE hierararchical latent representation Experiments in previous works [Child, 2020] suggest that the low-frequency information of images generated by VDVAE is mostly controlled by the latent variables at the top of the hierarchy, while the image high-frequency details are dependent on the latent variables at the bottom of the hierarchy. Similar properties were observed for other hierarchical VAE architectures [Vahdat and Kautz, 2020,Havtorn et al., 2021a]. Our experiments are in line with those observations. We study

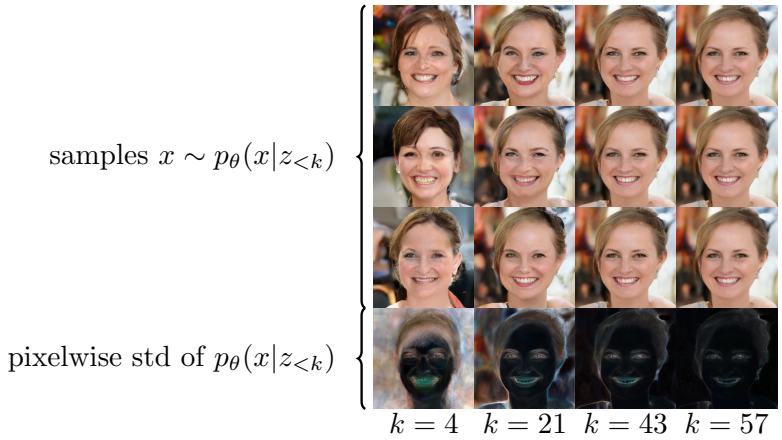


Figure 6.1 – Samples (rows 1-3) and pixel-wise standard deviation (row 4) of VDVAE hierarchical generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ (when fixing the k first latent groups of the hierarchy), for different values of k. High level semantic information and image low-frequency components is mostly controlled by the first groups of the hierarchy ($k < 21$), while image high frequency details (hairs, edges) are determined by the last latent groups ($k \geq 43$).

the VDVAE model provided by the author [Child, 2020], trained on FFHQ256 [Karras et al., 2019] with $L = 66$ groups in the latent hierarchy. Figure 6.1 shows sampled images

from the conditional generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$, where $\boldsymbol{z}_{<k}$ are k fixed latent groups at the top of the hierarchy. For a few fixed latent groups ($k = 4$), generated images share high level semantic information such as gender, skin tone or face orientation, indicating that those attributes are most likely to be encoded into the $k = 4$ first groups of the latent hierarchy. When more latent groups are fixed, generated images share the same low-frequency information, and the variation between samples is mostly due to variation of high frequency details in textures (hairs, background) or edges (face shape, eyes, mouth). Therefore, it appears that the hierarchical latent representation learned by VDVAE implicitly separates the image low-frequency information from the high-frequency details. Hence, latent groups at the top of the hierarchy monitor the low-frequency information, whereas the latent groups at the bottom of the hierarchy control high-frequency details.

6.4.2 Is VDVAE implicitly a Super-resolution network?

Average low-resolution pairwise distance between samples Our previous experiments suggest that VDVAE implicitly encodes the distribution of high frequency details conditional on low-frequency information via the hierarchical structure imposed on the prior model $p_\theta(\boldsymbol{z})$. We recall that image super-resolution is the task of recovering high-frequency details from the low-frequency information contained within a low-resolution image. We formulate the hypothesis that VDVAE conditional generative models $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ are implicit super-resolution models, generating diverse super-resolved versions of one particular low-resolution image y. To validate this hypothesis, we measure how close the image generated by the conditional models $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ are with each other, when they are downsampled with different downscaling factors. Without loss of generality, we consider the root mean square error (RMSE) as a measure of distance between samples. Thus, we estimate:

$$U_k^s := \mathbb{E}_{p_\theta(\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - H_s\tilde{\boldsymbol{x}}\|_2\right], \tag{6.5}$$

the average low-resolution pairwise distance of the generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$, when samples are downsampled by a factor s. U_k^s measures to what extent images sampled from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ differ from each other when they are downsampled.

Practical details We compute an estimations of the average sample low-resolution pairwise distance U_k^s (6.5) with ancestral Monte-Carlo sampling. We sample 50 different full latent codes $\boldsymbol{z}^{(i)}$ from the prior:

$$\boldsymbol{z}^{(i)} \sim p_\theta(\boldsymbol{z}). \tag{6.6}$$

For each latent code $\boldsymbol{z}^{(i)}$ and each number of fixed groups k, we sample five images:

$$\boldsymbol{x}^{(i,k,l)} \sim p_\theta\Big(\boldsymbol{x}|\boldsymbol{z}_{<k}^{(i)}\Big). \tag{6.7}$$

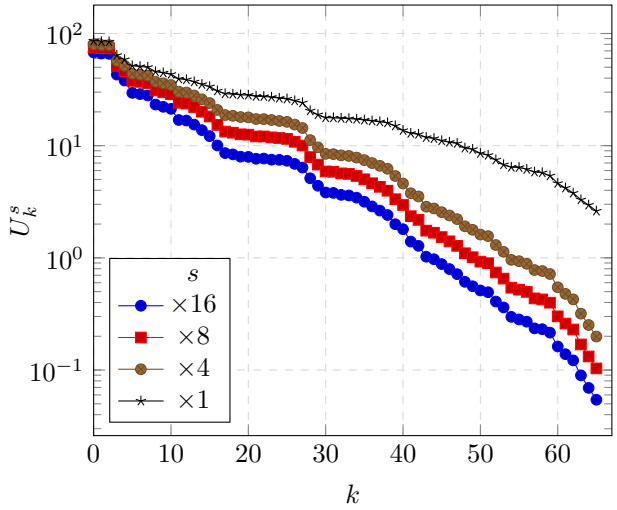


Figure 6.2 – Average low-resolution pairwise distance, U_k^s (6.5) between samples from the conditional generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ of VDVAE, for downscaling factors $s = 1, 4, 8, 16$. Image with pixel values in $[0, 255]$.

The average sample pairwise distance estimation is then computed as:

$$\hat{U_k^s} = \sum_{i=1}^{50} \sum_{1 \leq l < m \leq 5} \frac{1}{\sqrt{m}} \| H_s \boldsymbol{x}^{(i,k,l)} - H_s \boldsymbol{x}^{(i,k,m)} \|_2, \tag{6.8}$$

where H_s is the downsampling operator associated to the downscaling factor s.

Low-resolution consistency of VDVAE samples In Figure 6.2 we estimate the value of U_k^s for different downsampling factors. Results illustrate that, as the number of fixed groups k increases, the generated images get more similar. Furthermore, for a given number of fixed groups k, the low-resolution pairwise distance decreases as the downsampling factor s increases, indicating that there is more variation in the HR samples than in their LR counterparts. The gap between the average sample pairwise distance in high resolution ($s = 1$), and low-resolution ($s \in \{4, 8, 16\}$) gets larger as the number of fixed groups k increases, indicating that fixing a large number of groups k yields samples that are close at low-resolution but different at high-resolution. The average low-resolution pairwise distance U_k^s gets closer to zero as k increases. While there is no value of k such that $U_k^s = 0$, we argue that for a large enough value of k, U_k^s becomes negligible compared to the pixel intensity range (0-255), for instance $U_{60}^4 < 0.5$. Those results show that the downsampling of any image synthesized from the conditional generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ are consistent with one low-resolution image $\boldsymbol{y}$ with a certain precision inversely proportional to k. Thus, we conclude that, for a large enough value of k, all images

sampled from $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ are diverse super-resolved versions of one LR image $\boldsymbol{y}$, which is in line with the hypothesis that $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ is implicitly a diverse super-resolution model.

6.5 Diverse super-resolution with VDVAE

We propose to exploit the properties of the latent hierarchical representation learned by VDVAE to design a diverse super-resolution method. As seen in the previous section, VDVAE conditional models $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ can be viewed as implicit diverse super-resolution models, generating diverse super-resolved versions of one low-resolution image, with consistency inversely proportional to k. Thus, we propose to super-resolve an image $\boldsymbol{y}$ by estimating the latent variables $\boldsymbol{z}_{<k}$ encoding the low-frequency information contained within $\boldsymbol{y}$, and by sampling $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$, using the pretrained VDVAE generative model. In order to predict the latent variables $\boldsymbol{z}_{<k}$ that correspond to a low-resolution image $\boldsymbol{y}$, we introduce a low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$. Overall, our super-resolution model is defined as:

$$p_{SR}(\boldsymbol{x}|\boldsymbol{y}) = \mathbb{E}_{q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}[p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})], \tag{6.9}$$

which implies that we can sample from $p_{SR}(\boldsymbol{x}|\boldsymbol{y})$ by sequentially sampling $\boldsymbol{z}_{<k} \sim q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ and $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$. Our approach rely on conditionning the generative process of VDVAE. Hence, we name our method CVDVAE, for conditionned VDVAE.

In this section we first detail the training criterion of the low-resolution encoder, and we derive a criterion to estimate the consistency error of the super-resolution model as a function of the number of predicted latent groups k. Next we describe the architecture of the low-resolution encoder.

6.5.1 Training criterion of the low-resolution encoder

Low-resolution encoder We introduce a low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$, which is a neural network parameterized by $\psi \in \Psi$, where Ψ is the parameter space of the network. Considering a joint training distribution of clean-degraded image pairs $p_{\text{data}}(\boldsymbol{x}, \boldsymbol{y})$, we can show that the conditional log-likelihood of the super-resolution model has a lower-bound.

Proposition 6.1. *The conditional log-likelihood of the super-resolution model on a joint distribution $p_{data}(\boldsymbol{x}, \boldsymbol{y})$ is lower-bounded by*

$$\mathcal{O}(\psi) = \mathbb{E}_{p_{data}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\right] \tag{6.10}$$

$$\leq \mathbb{E}_{p_{data}(\boldsymbol{x},\boldsymbol{y})}[\log p_{SR}(\boldsymbol{x}|\boldsymbol{y})]. \tag{6.11}$$

Proof. This result comes from applying the lower-bound introduced in [Harvey et al., 2022] to the truncated VAE $q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})$, $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$ (a detailed proof is given in appendix B.1.2). □

Training criterion Furthermore, as detailed in appendix B.1.2, maximizing the lower-bound $\mathcal{O}(\psi)$ in relation (6.10) is equivalent to minimizing the criterion:

$$\mathcal{L}(\psi) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}[\text{KL}(q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})||q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}))]. \tag{6.12}$$

In other words, to maximize the lower-bound $\mathcal{O}(\psi)$, the low-resolution encoder has to minimize the KL divergence between the LR and the HR encoder on LR-HR images pairs. We set $\mathcal{L}(\psi)$ to be the training criterion of the low-resolution encoder.

6.5.2 Expected consistency of the super-resolution model

In this part, we derive a criterion to select the number of latent groups k to be predicted by the low-resolution encoder, based on the expected consistency error of the super-resolution model. Without loss of generality, we measure the consistency error between a high-resolution and a low-resolution image as the root-mean-square error between the downsampled HR image and the LR image $\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2$. We define the consistency error of the super-resolution model as

$$CE(k) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})}\mathbb{E}_{p_{SR}(\boldsymbol{x}|\boldsymbol{y})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right]. \tag{6.13}$$

We show in the proposition 6.2 below that the consistency error of the super-resolution model (6.13) can be predicted without using the low-resolution encoder, when the low-resolution encoder is trained with the criterion (6.12). First, we introduce

$$r(\boldsymbol{z}_{<k}, \boldsymbol{x}, \boldsymbol{y}) := p_{\text{data}}(\boldsymbol{x}, \boldsymbol{y})q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x}), \tag{6.14}$$

the joint distribution of high-resolution and low-resolution image pairs $(\boldsymbol{x}, \boldsymbol{y})$, and their latent variable $\boldsymbol{z}$ given by the high-resolution encoder, and $r(\boldsymbol{z}_{<k}|\boldsymbol{y})$ the corresponding conditional distribution. We also consider the following assumptions:

Assumption 6.1. *There exists $\psi \in \Psi$ which satisfies $r(\boldsymbol{z}_{<k}|\boldsymbol{y}) = q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all $\boldsymbol{y}$ in the support of $p_{data}(\boldsymbol{y})$.*

Assumption 6.2. *The low-resolution encoder parameters ψ are minimizers of the training criterion* (6.12)*:*

$$\psi \in \arg\min_{\tilde{\psi}} \mathcal{L}(\tilde{\psi}). \tag{6.15}$$

Assumption 6.3. *The VAE encoder $q_\phi(\boldsymbol{x}|\boldsymbol{z})$ and generative model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$ have enough capacity and are trained well enough so that ϕ and θ reaches the upper bound of the ELBO loss* (B.12)*.*

Assumption 6.1 is met if the low-resolution encoder has enough capacity while assumption 6.2 is met if the low-resolution encoder is well trained. In the next proposition, we show that the expected consistency of the super-resolution model can be expressed only as a function of the generative model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$.

Proposition 6.2 (Proof in B.2)**.** *Under assumptions 6.1, 6.2 and 6.3, the expected consistency error is equal to the average low-resolution pairwise distance* U_k^s (6.5)*:*

$$CE(k) = \mathbb{E}_{p_\theta(\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\tilde{\boldsymbol{x}} - H_s\boldsymbol{x}\|_2\right] \tag{6.16}$$

$$= U_k^s. \tag{6.17}$$

Proposition 6.2 shows that, if a low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ has enough capacity and is trained perfectly, it is possible to estimate the expected consistency error of a super resolution model relying on $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ without actually using the low-resolution encoder. It implies that $CE(k)$ can be estimated before training the low-resolution encoder, using relation (6.16). Therefore, the formulation (6.16) can be used as a criterion to select the number of latent groups k to be predicted by the low-resolution encoder, as a function of the desired consistency. Furthermore, the expected consistency as defined in (6.16) is equal to the average low-resolution pairwise distance of the conditional generative model U_k^s (6.5), displayed in Figure 6.2.

6.5.3 Low-resolution encoder

Network architecture The low-resolution encoder architecture, displayed in Figure 6.3 is built similarly to the VDVAE encoder, but it contains a reduced number of blocks due to the smaller number of latent variable groups to predict. Specifically, the low-resolution encoder is composed of a deterministic bottom-up path that extracts different levels of representation, and a top-down path that sequentially infers each latent group $\boldsymbol{z}_l$, using the representations extracted by the bottom-up path. The bottom-up path is composed of simple residual blocks, while the top-down path is composed of residual top-down blocks [Kingma et al., 2016]. Both residual blocks and residual top-down blocks follow the same design as in VDVAE.

Parameters sharing Following a common practice in hierarchical VAE design [Sønderby et al., 2016, Kingma et al., 2016, Child, 2020, Vahdat and Kautz, 2020], the top-down path of the low-resolution encoder shares its parameters with VDVAE generative model, as described in Figure 6.3. Only the parameters of the low-resolution encoder (in red in Figure 6.3) are trained, while the shared parameters (in blue in Figure 6.3) are set to the value of the corresponding parameters in the pretrained VDVAE generative model, and remain frozen during training.

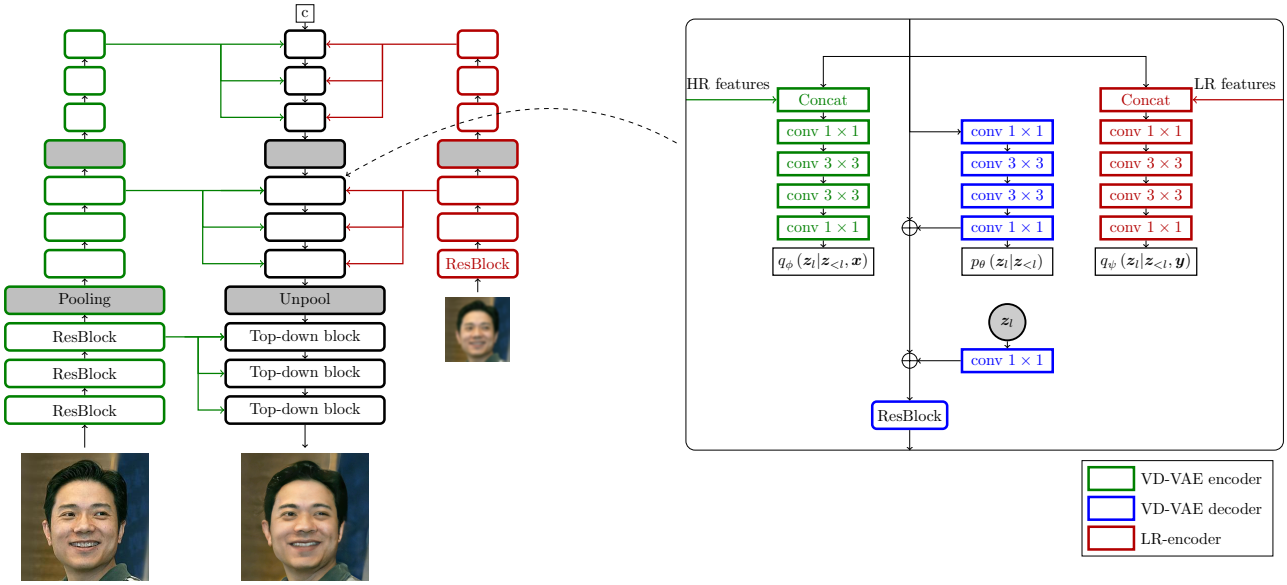


Figure 6.3 – Super-resolution model based on a pretrained VDVAE model. The low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ (in red) is trained to match VDVAE pretrained encoder $q_\phi(\boldsymbol{z}_{<k}|x)$ (in green). Both encoders share parameters with VDVAE generative model $p_\theta(\boldsymbol{z}, \boldsymbol{x})$ (in blue) in the top-down path. To super-resolve an image y, we sequentially sample $\boldsymbol{z}_{<k} \sim q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ and $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$.

6.5.4 Enforcing consistency of the super-resolution

Consistency error of the super-resolution By definition of the downsampling model (6.1), the super-resolved version of a LR image $\boldsymbol{y}$ should belong to the space of consistent solutions $\{\boldsymbol{x}|\boldsymbol{y} = H_s\boldsymbol{x}\}$. However, as many learning based super-resolution methods, our super-resolution model does not explicitly enforce this condition. According to Proposition 6.2 and the estimated average LR pairwise distance in Figure 6.2, the consistency error of our super-resolution model should be small, but remains positive. The lack of consistency can also be due to the assumption of Proposition 6.2 not being met, that is to say, the encoders and the decoder not having enough capacity and/or not being trained well enough. In particular, if the encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ does not match the intractable posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, the reconstruction error of the VAE will propagate to the low-resolution encoder and hurt the consistency of the super-resolution. In practice, we also found that the reconstruction by VDVAE was imprecise because of the 5 bits precision loss used to train the original VDVAE model, and that this reconstruction error would propagate on our low-resolution encoder.

Projection In order to generate super-resolved images consistent with respect to the low resolution input, we apply a post-processing step by projecting the output of the generative network to the space of consistent solutions $\{\boldsymbol{x}|\boldsymbol{y} = H_s\boldsymbol{x}\}$, as previously proposed in [Bahat and Michaeli, 2020]. Given a potentially inconsistent image $\hat{\boldsymbol{x}}$, the consistent solution $\hat{\boldsymbol{x}}_p$ is obtained as

$$\hat{\boldsymbol{x}}_p = (I - H_s^T(H_sH_s^T)^{-1}H_s)\hat{\boldsymbol{x}} + H_s^T(H_sH_s^T)^{-1}\boldsymbol{y}. \tag{6.18}$$

In practice, the filter $(H_sH_s^T)$ can be efficiently inversed in the frequency domain using a discrete Fourier transform [Bahat and Michaeli, 2020].

6.6 Experiments

6.6.1 Implementation details

Training Our work is built upon the official VDVAE codebase [Child, 2020], and we reuse the weights of the VDVAE network trained on FFHQ256 provided by the authors. All models are trained with Adam optimizer and the learning rate is divided by 10 when the validation loss is on a plateau. We reuse the same data split as the one used to train the original VDVAE, networks, with 63000 images in the training set. Each model was trained on 4 A100 GPUs for less than 12 hours. Using 4 GPUs allowed us to get a large enough batch size to reduce the training instability that can appear when training VAEs. Note that this can be considered a lightweight training compared to the time required to train the whole VDVAE model, namely 2 weeks on 32 GPUs. More details about the training can be found in Table 6.1.

Table 6.1 – Training details of the low-resolution encoder for each upsampling factors..

upsampling factor	$\times 4$	$\times 8$	$\times 16$
k	57	43	21
batch size	16	32	64
learning rate	5.10^{-4}	5.10^{-5}	5.10^{-6}
iterations	20K	70K	20K

Number of predicted latent groups The number of predicted latent groups k for each resolution is set so that each low-resolution encoder only predicts the latent groups z_l of spatial dimension lower than or equal to the dimension of the LR input. We found that training the low-resolution encoder to predict more groups would make the training of the low-resolution encoder harder, yielding super-resolved samples more consistent with the input but also containing more artifacts.

6.6.2 Experimental settings

Dataset and upscaling factors We test our super-resolution method on the FFHQ dataset [Karras et al., 2019], with images of resolution 256×256. We experiment on 3 upscaling factors: $\times 4$ ($64 \times 64 \rightarrow 256 \times 256$), $\times 8$ ($32 \times 32 \rightarrow 256 \times 256$) The low resolution images are initially downscaled by applying an antialiasing kernel followed by a bicubic interpolation.

Compared methods We compare our method with a conditional normalizing flow (HCFlow) [Liang et al., 2021], a conditional diffusion model (SR3) [Saharia et al., 2021b], and a method that add guidance to a non-conditional diffusion model at inference (DPS) [Chung et al., 2023]. We retrain HCFlow on FFHQ256 using the official implementation. For DPS, we also reuse the official implementation with the available pretrained model, which was trained on FFHQ. For SR3, since no official implementation is available, we use an open-source (non-official) implementation [Jiang, 2022], and we retrained the model for our task. When training SR3, we found that color shift [Deck and Bischoff, 2023] was hurting the reconstruction error. To reduce the reconstruction error due to the color shift effect, we project the super-resolved image on the space of consistent solutions at inference as described in equation (6.18). For fair comparison, we retrained both HCFlow and SR3 with the same computational budget than our conditional model. For HCFlow and CVDVAE, we set the temperature of the latent variables at $\tau = 0.8$ during sampling.

Evaluating a diverse SR method Due to the ill-posedness of the problem, evaluating a diverse super-resolution model based solely on the distortion to the ground truth is not satisfactory. Indeed, there exist many solutions that are both realistic and consistent

with the LR input while being far from the ground truth. Thus, in order to evaluate the super-resolution model, we provide a series of metrics that evaluate different expected characteristics of a diverse super-resolution model, such as the consistency of the solution, the diversity of the samples and the general visual quality. It should be noted that those metrics are not necessarily correlated: a model could propose diverse solutions, that are not consistent or realistic, or, on the opposite, it could propose solutions that are realistic and consistent but with a low diversity. Thus, to evaluate a diverse super-resolution model, it is necessary to consider these three different aspects together: diversity, consistency and visual quality.

Evaluation metrics The general quality of the super-resolved images is evaluated using the blind Image quality metric BRISQUE [Mittal et al., 2012] . Consistency with the LR input is also measured via PSNR (denoted LR-PSNR in Tables 6.2 and 6.3). Furthermore, to evaluate the diversity of the super-resolution, we evaluate the Average Pairwise distance between different samples coming from the same LR input (denoted APD in Tables 6.2 and 6.3), both at the pixel level, using the mean square error (MSE) between samples (considering pixel intensity value between 0 and 1), and at a perceptual level using LPIPS. For one LR input, the average pairwise distance is computed as the average distance between all the possible pairs of images in a set of 5 super-resolved samples. The reported APD in Tables 6.2 and 6.3 corresponds to the mean value of the single image APD over 500 LR inputs in the test set. We measure the distortion of the super-resolved samples with respect to the ground truth HR image in terms of peak Signal-to-Noise Ratio (PSNR), structural similarity (SSIM) [Wang et al., 2004] and the perceptual similarity (LPIPS) [Zhang et al., 2018b], as it is common in the super-resolution literature. All numbers reported correspond to the metric mean value on a subset of 1000 images from FFHQ256 test set.

6.6.3 Results

Quantitative evaluation Quantitative results on Table 6.2 indicate that our method provides a good trade-off between the different evaluated metrics. Indeed, our method provides the second best results in terms of distortion and visual quality, and the second or third best results in terms of diversity. It is one of the fastest one along with HCFlow. HCFlow provides the best results for distortion metrics as it explicitly penalizes bad reconstruction in its training loss. Similar to our CVDVAE, it is also very fast as it requires only one network evaluation to produce a super-resolved image. However, HCFlow lacks high-level diversity (as measured by the LPIPS average pairwise distance), compared with the concurrent methods. We postulate that this lack of diversity is due to the relative lack of expressivity of normalizing flows compared to diffusion and HVAE models. Our method, along with DPS, produces the best results in terms of visual quality as measured by the BRISQUE metric, illustrating the benefit of using a pretrained unconditional generative

Table 6.2 – Comparison of diverse SR methods face super-resolution. Best result is in bold, second best is underlined.

		Distortion			Visual Quality	Consistency	Diversity (APD)		
	model	PSNR↑	SSIM↑	LPIPS ↓	BRISQUE↓	LR-PSNR ↑	MSE ($\times 10^4$) ↑	LPIPS ($\times 10^3$) ↑	time (s)
×4	Bicubic	27.49	0.84	0.29	61.79	36.99	0	0	
	HCFlow	**31.74**	**0.89**	**0.13**	37.21	52.81	**161.8**	62.6	**0.11**
	SR3	28.87	0.73	0.25	37.17	<u>63.47</u>	20.06	**209.2**	46
	DPS	28.50	0.81	0.20	**32.21**	38.96	10.4	<u>150.0</u>	103
	CVDVAE	<u>30.24</u>	<u>0.85</u>	<u>0.16</u>	<u>32.30</u>	**75.20**	<u>88.8</u>	123.0	<u>0.14</u>
×8	Bicubic	23.50	0.70	0.45	78.42	33.61	0	0	
	HCFlow	**26.72**	**0.76**	**0.24**	36.25	51.13	**575.5**	155.3	<u>0.17</u>
	SR3	<u>26.26</u>	0.70	0.29	34.78	<u>68.6</u>	19.95	234.3	62
	DPS	24.38	0.68	0.28	**30.09**	36.97	35.68	**247.4**	103
	CVDVAE	25.47	<u>0.71</u>	<u>0.27</u>	<u>32.26</u>	**70.15**	<u>248.2</u>	<u>236.4</u>	**0.13**

model. However, DPS takes significantly more time to run ($\approx \times 1000$) than CVDVAE and HCFlow, as it requires 1000 steps of network evaluations and backpropagation through the denoiser to produce one super-resolved sample. Finally, SR3 performances are inferior to the compared method. We used the same computational budget (48h on 4 GPUs) for training the SR3 models as our CVDVAE and HCFlow. This computational budget is significantly lower than the one reported in the SR3 paper [Saharia et al., 2021b] (≈ 4 days on 64 TPUv3 chip[1]), and it is more than likely that training the SR3 models for longer would improve their performance. Like DPS, SR3 is slower than our method as it requires 2000 network evaluations to produce one super-resolved image.

Qualitative evaluation A visual comparison of the different evaluated methods is provided in Figures 6.4, 6.6, 6.5 and 6.7. More visual results from our proposed methods are displayed in Figures B.1, B.2 and B.3. Our method is able to produce diverse textures as illustrated by the facial hair variation in Figure 6.4 or the hair variation in 6.6. CVDVAE appears to produce super-resolved samples with higher semantic diversity, in terms of textures (hairs, skin), in line with the higher perceptual diversity measured in the quantitative evaluation.

Temperature control As for the unconditionnal HVAE models studied in the previous chapters, CVDVAE offers the possibility to control the conditional generation via the temperature of the latent variable distributions (see discussion in section 4.3). In order to assess the behavior of the model on both low and high temperature regime, we evaluate our method on 2 temperatures ($\tau \in \{0.1, 0.8\}$). Quantitative results in Table 6.3 show that reducing the temperature leads to a solution closer to the ground truth in terms of low-levels distortion metrics (PSNR and the SSIM), while using a higher temperature helps

[1]one TPUv3 chip has 32GB capacity [doc, 2023], while the Nvidia A100 GPU we used each had 40GB capacity

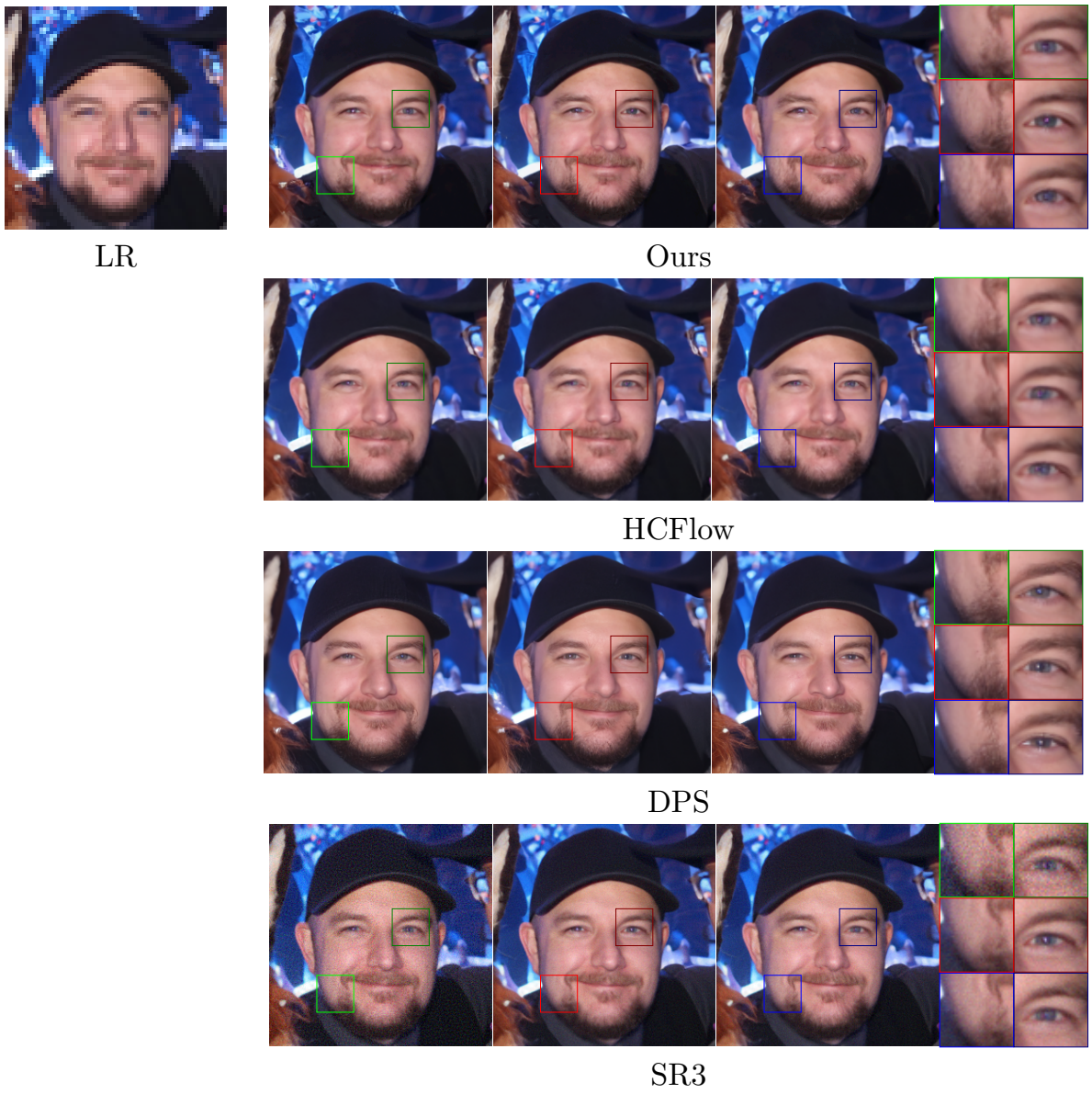


Figure 6.4 – Samples from different diverse SR methods (×4)

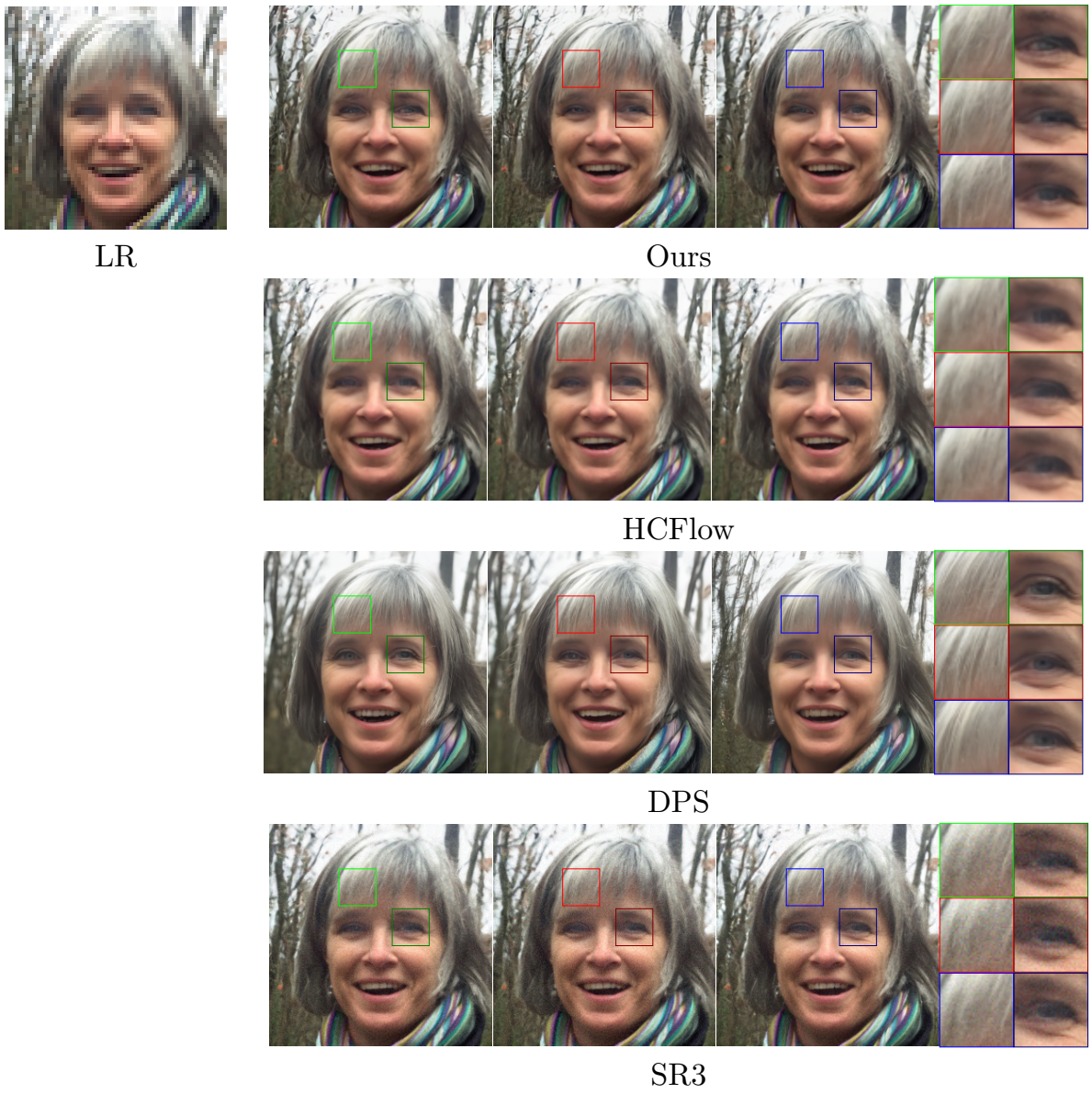


Figure 6.5 – Samples from different diverse SR methods (×4)

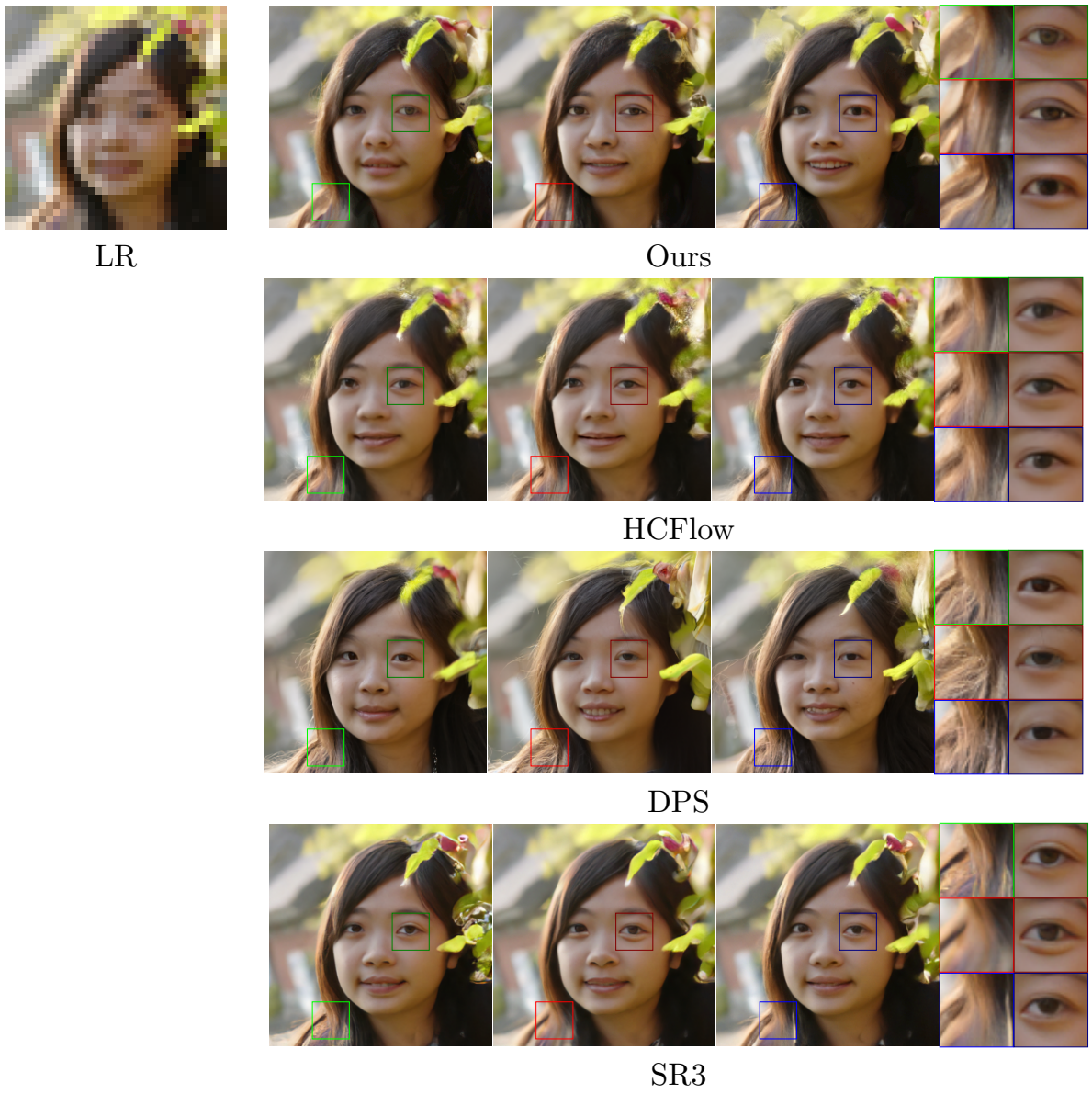


Figure 6.6 – Samples from different diverse SR methods (×8)

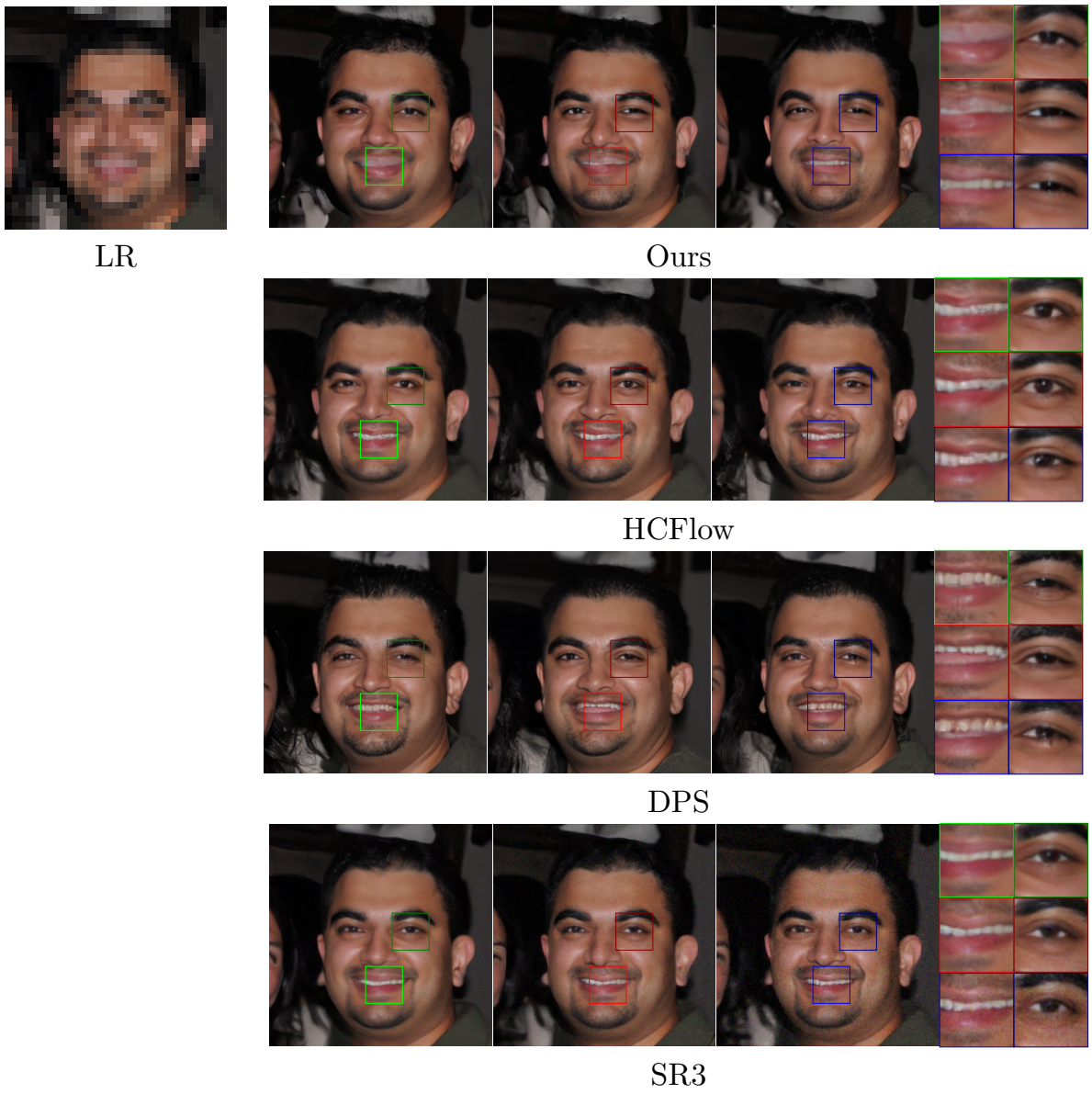


Figure 6.7 – Samples from different diverse SR methods (×8)

Table 6.3 – Effect of the sampling temperature τ on CVDVAE super-resolution results.

		Distortion			Visual Quality	Consistency	Diversity (APD)	
	τ	PSNR↑	SSIM↑	LPIPS ↓	BRISQUE↓	LR-PSNR ↑	MSE ($\times 10^4$) ↑	LPIPS ($\times 10^3$) ↑
×4	0.1	**30.75**	**0.86**	**0.15**	36.47	**75.70**	64.6	104.5
	0.8	30.24	0.85	0.16	**32.3**	75.20	**88.8**	**123.0**
×8	0.1	**26.27**	**0.75**	0.30	50.34	**71.63**	140.4	179.0
	0.8	25.47	0.708	**0.28**	**32.26**	70.15	**248.2**	**236.4**

Figure 6.8 – Effect of the sampling temperature τ on the super-resolved result. Increasing the temperature yields image with more high-frequency details.

to improve the perceptual similarity (LPIPS) with the ground-truth, as well as the general perceptual quality of the generated HR images and the diversity of the samples. On Figure 6.8, we display CVDVAE's samples at different temperatures τ. The sampling temperature correlates with the perceptual smoothness of the super-resolved sample, a higher sampling temperature inducing images containing sharper details. For out-of-distribution samples, reducing the temperature can help to reduce artefacts in the generated images, as illustrated in Figure 6.9.

Figure 6.9 – Example of failure case of our method on ×16 upsampling. The presence of uncommon attributes such as make-up can cause our method to fail. Sampling at a low-temperature can help to reduce the artifacts.

6.7 Conclusion

In this chapter, we showed that the hierarchical latent representation learned by a hierarchical variational autoencoder such as VDVAE can be efficiently repurposed for super-resolution. Consequently, we showed that we can efficiently perform diverse super-resolution by learning to encode low-resolution images in the relevant part of a pretrained VDVAE latent space. We provided an analysis to select the relevant part of the latent space, as a function of the expected consistency of the super-resolution model. The original VDVAE model needs not to be retrained, and training our low-resolution encoder takes 200 less computational resources than those required to train the full VDVAE model. Our proposed method showed promising results on face super-resolution, on par with state-of-the-art diverse SR methods, providing semantically diverse and high-quality samples. Our results illustrate the ability of conditional hierarchical generative models to perform complex image-to-image tasks.

Chapter 7

Conclusion and perspectives

7.1 Conclusion

Deep neural networks enable to define strong prior models on images that we can exploit to solve challenging inverse problems. Using deep learning and deep generative models for image restoration tasks raises new challenges. Among those challenges, we have considered in this thesis the problem of image restoration in settings where no paired data are available, the design of convergent optimization schemes for solving variational problem with a deep generative prior, and the design of efficient methods to sample from the posterior distribution of an inverse problem, given a prior induced by a deep generative model. We have presented three main contributions addressing those challenges.

To address the problem of image restoration without paired datasets, we introduced in chapter 3 the adversarial local regularization (ALR), a framework that enables training a neural network as a regularization function. By exploiting adversarial training, our method does not need paired datasets. We imposed a fully convolutional structure on the regularization network, so that we could train it with only small image patches. The adversarial regularization provides an explicit regularization function. We have demonstrated the ability of our method to outperform popular unsupervised restoration methods on image denoising.

Next we have studied the use of hierarchical VAEs as a prior. After a review on VAE and Hierarchical VAE models in chapter 4, we have shown the benefits of using a hierarchical VAE model in two different ways.

In chapter 5, we have demonstrated that HVAE models could be used to solve generic linear inverse problems with PnP-HVAE. PnP-HVAE is an iterative optimization algorithm that exploits the hierarchical encoder of HVAEs to optimize the solution without relying on expensive backpropagation through the generative network. We introduced a temperature hyperparameter that enables controlling the strength of the regularization, and we demonstrated that we could enhance the quality of the results by an appropriate tuning of

the temperature hyperparameter. Furthermore, we derived sufficient conditions on the HVAE model to guarantee the convergence of our method to a fixed point, by drawing connection with the denoising Plug-and-play algorithms. Our experiments demonstrate the ability of our method to solve challenging image inverse problems on a specialized face dataset. Our method also showed promising results on natural images restoration using our fully convolutional patchVDVAE model.

In chapter 6, we have demonstrated that hierarchical VAE models could also be efficiently repurposed to sample from the posterior distribution on an image inverse problem. We have developed a strategy to train an encoder on degraded (low-resolution) images by exploiting the HVAE hierarchical encoder. By combining this new encoder with the HVAE generative model, we showed that we could produce samples from the posterior distribution of a super-resolution problem with only one network evaluation. Then, we experimentally demonstrated on the problem of face images super-resolution that our approach provides an advantageous trade-off between sample quality and computational efficiency.

7.2 Discussion and Perspectives

7.2.1 Which deep learning regularizer should you use?

Adversarial regularization or denoising PnP? The adversarial regularization presented in chapter 3 provides an explicit regularization function parameterized by a neural network, in opposition with denoising Plug-and-Play (PnP) methods, that only indirectly model the regularization terms through its gradient or its proximal operator. After the publication of this work, the gradient-step denoiser [Hurault et al., 2021] was introduced to relate a denoiser to the potential of an explicit regularization function parameterized by a neural network. The gradient step denoiser has the advantage that its learned potential network is related to an explicit probabilistic model (through the denoising score matching theory and Tweedie's formula), while adversarial regularizers are not related to any probabilistic model. Hence, gradient step denoiser appears superior to adversarial regularization.

Denoising PnP or deep generative models? PnP methods use deep denoiser networks to model local information about the prior information. Intuitively, the denoiser should move a data point slightly closer to the high-density area of the prior distribution. However, because (by definition) denoisers where only trained to remove noise, it is not clear if it works as expected on images that differs from noisy images. For instance, for solving an inpainting problem, it is not clear if applying a denoiser to a masked image would really bring it closer to the area of high-density of the prior distribution, and in practice denoising PnP might fail when there is too much missing information. On the

other hand, deep generative models excel at filling missing information. As such they perform well for problems with a lot of missing information, such as inpainting with large size masks, or super-resolution with large upscaling factors. On the other hand, using generative regularization for moderately ill-posed problem might fail, because finding an image consistent with the degraded observation might become too difficult. Our PnP-HVAE method actually provides a good compromise as it jointly optimizes the image and its latent variables.

7.2.2 Pros and cons of HVAE priors against other deep generative models

In this work we have focused on the use of hierarchical VAEs for image restoration. Our results show that, when a pretrained HVAE model is available, using it as prior provides significant advantages compared to other types of deep generative models such as GANs or denoising diffusion models. In particular, our methods based on HVAE regularization were faster to run than the concurrent methods, while providing a similar or superior restoration quality.

However, current HVAE models available in the literature only operates on datasets with restricted diversity (such as faces), or on low-resolution images (such as image-net 64x64). As such, this limits the application of our method to those datasets, while other classes of generative models such as GANs or denoising diffusion models operate on much more diverse dataset and at larger resolution. This restriction is the main limitation of our method based on HVAE.

We postulate that there is a large room of improvement for HVAE, and that, by extending promising idea from the literature on deep generative models, and by using equivalent computing budget used for training concurrent methods, the performance of HVAE model could significantly improve, and they could be applied on more challenging dataset. The finding of our work illustrate the benefits of using an HVAE model for downstream applications. As such, we hope that those findings could motivate the research on HVAE models.

7.2.3 Toward flexible posterior sampling with HVAEs

The posterior sampling method we have presented in chapter 6 is specialized on the problem of super-resolution, and it requires to train an encoder on paired data. A question that remains to be answered is, how can we use HVAE models as prior into a flexible posterior sampling methods, that do not require training a task-specific encoder beforehand. For simple (non-hierarchical) VAEs, several works have proposed to use a Gibbs sampling scheme to produce samples from the posterior (see for instance [Mattei and Frellsen, 2018], or chapter 5 of Mario Gonzalez thesis [Olmedo, 2021]). A straightforward approach would

be to expend the Gibbs sampling idea for hierarchical VAEs. In practice, this would amount to a simple modification of the PnP-HVAE algorithm presented in chapter 5, where we would replace each minimization step by a sampling step from the corresponding probability distribution (plus an eventual Metropolis step to account for the approximation error of the encoder). We could also envisage more sophisticated methods to account for the hierarchical structure of the latent space. For instance, we could use a collapsed Gibbs sampler [Van Dyk and Park, 2008], sequential importance sampling or sequential Monte-Carlo [Doucet et al., 2001] methods. Another promising approach would be to integrate conditional generative models such as our diverse super-resolution network presented in chapter 6 within a "plug-and-play" Gibbs-sampling scheme in a similar fashion than [Coeurdoux et al., 2023]. We postulate that integrating an HVAE model within a flexible sampling scheme could provide similar benefits in terms of sample quality and computational efficiency than the one observed for our optimization based method. As such, we believe that this is a promising research direction for future works.

Bibliography

[doc, 2023] (2023). Cloud tpu documentation. `https://cloud.google.com/tpu/docs/system-architecture-tpu-vm`. Accessed: 2023-11-13. 100

[Agustsson and Timofte, 2017] Agustsson, E. and Timofte, R. (2017). Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135. 77

[Aharon et al., 2006] Aharon, M., Elad, M., and Bruckstein, A. (2006). K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322. 16

[Altekrüger et al., 2022] Altekrüger, F., Denker, A., Hagemann, P., Hertrich, J., Maass, P., and Steidl, G. (2022). Patchnr: Learning from small data by patch normalizing flow regularization. *arXiv preprint arXiv:2205.12021*. 77

[Amos et al., 2017] Amos, B., Xu, L., and Kolter, J. Z. (2017). Input convex neural networks. In *International Conference on Machine Learning*, pages 146–155. PMLR. 17

[Anderson, 1982] Anderson, B. D. (1982). Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326. 20

[Ardizzone et al., 2019] Ardizzone, L., Lüth, C., Kruse, J., Rother, C., and Köthe, U. (2019). Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*. 87

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR. 16, 28, 30, 32

[Attouch et al., 2010] Attouch, H., Bolte, J., Redont, P., and Soubeyran, A. (2010). Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality. *Mathematics of operations research*, 35(2):438–457. 69

[Bahat and Michaeli, 2020] Bahat, Y. and Michaeli, T. (2020). Explorable super resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2716–2725. 86, 87, 97

[Bau et al., 2019] Bau, D., Zhu, J.-Y., Wulff, J., Peebles, W., Strobelt, H., Zhou, B., and Torralba, A. (2019). Seeing what a gan cannot generate. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4502–4511. 21

[Beck and Teboulle, 2009] Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202. 15, 17

[Bigdeli et al., 2017] Bigdeli, S. A., Jin, M., Favaro, P., and Zwicker, M. (2017). Deep Mean-Shift Priors for Image Restoration. In *Adv. in Neural Information Proces. Systems 30*, pages 763–772. 18, 28

[Bishop and Nasrabadi, 2006] Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer. 62

[Bora et al., 2017] Bora, A., Jalal, A., Price, E., and Dimakis, A. G. (2017). Compressed sensing using generative models. In *International Conference on Machine Learning*, pages 537–546. PMLR. 21, 28, 56, 58, 86, 88

[Boyd et al., 2011] Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122. 15, 17

[Brock et al., 2018] Brock, A., Donahue, J., and Simonyan, K. (2018). Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*. 77

[Bromiley, 2003] Bromiley, P. (2003). Products and convolutions of gaussian probability density functions. *Tina-Vision Memo*, 3(4):1. 67, 128

[Buades et al., 2005] Buades, A., Coll, B., and Morel, J.-M. (2005). A non-local algorithm for image denoising. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 2, pages 60–65. Ieee. 17

[Burgess et al., 2018] Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. (2018). Understanding disentangling in beta-vae. *arXiv preprint arXiv:1804.03599*. 47

[Chadebec and Allassonnière, 2022] Chadebec, C. and Allassonnière, S. (2022). A geometric perspective on variational autoencoders. *Advances in Neural Information Processing Systems*, 35:19618–19630. 44, 50

[Chen et al., 2018] Chen, R. T., Li, X., Grosse, R. B., and Duvenaud, D. K. (2018). Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31. 47

[Cheng et al., 2015] Cheng, Z., Yang, Q., and Sheng, B. (2015). Deep colorization. In *Proceedings of the IEEE international conference on computer vision*, pages 415–423. 13

[Child, 2020] Child, R. (2020). Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*. 10, 49, 50, 56, 57, 62, 74, 75, 77, 82, 86, 87, 89, 90, 95, 97, 133

[Chira et al., 2022] Chira, D., Haralampiev, I., Winther, O., Dittadi, A., and Liévin, V. (2022). Image super-resolution with deep variational autoencoders. In *European Conference on Computer Vision*, pages 395–411. Springer. 86, 88

[Choi et al., 2021] Choi, J., Kim, S., Jeong, Y., Gwon, Y., and Yoon, S. (2021). Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*. 86, 88

[Chung et al., 2023] Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. (2023). Diffusion Posterior Sampling for General Noisy Inverse Problems. In *(ICLR) International Conference on Learning Representations*, pages 1–28. 21, 23, 76, 98, 133

[Chung et al., 2022] Chung, H., Sim, B., Ryu, D., and Ye, J. C. (2022). Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696. 21

[Coeurdoux et al., 2023] Coeurdoux, F., Dobigeon, N., and Chainais, P. (2023). Plug-and-play split gibbs sampler: embedding deep generative priors in bayesian inference. *arXiv preprint arXiv:2304.11134*. 23, 110

[Coifman and Donoho, 1995] Coifman, R. R. and Donoho, D. L. (1995). Translation-Invariant De-Noising. In *Wavelets and Statistics (Lect. Notes in Statistics, vol 103)*, chapter 5, pages 125–150. Springer. 16, 27

[Cremer et al., 2018] Cremer, C., Li, X., and Duvenaud, D. (2018). Inference suboptimality in variational autoencoders. In *International Conference on Machine Learning*, pages 1078–1086. PMLR. 61

[Dabov et al., 2007] Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. on Image Processing*, 16:2080–95. 17, 38

[Dai et al., 2017] Dai, B., Wang, Y., Aston, J., Hua, G., and Wipf, D. (2017). Hidden talents of the variational autoencoder. *arXiv preprint arXiv:1706.05148*. 44

[Dai and Wipf, 2018] Dai, B. and Wipf, D. (2018). Diagnosing and enhancing vae models. In *International Conference on Learning Representations*. 44

[Daras et al., 2021] Daras, G., Dean, J., Jalal, A., and Dimakis, A. (2021). Intermediate layer optimization for inverse problems using deep generative models. In *International Conference on Machine Learning*, pages 2421–2432. PMLR. 21, 22, 58, 75, 76

[Deck and Bischoff, 2023] Deck, K. and Bischoff, T. (2023). Easing color shifts in score-based diffusion models. *arXiv preprint arXiv:2306.15832*. 98

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22. 45

[Deshpande et al., 2017] Deshpande, A., Lu, J., Yeh, M.-C., Jin Chong, M., and Forsyth, D. (2017). Learning diverse image colorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6837–6845. 87

[Dhariwal and Nichol, 2021] Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794. 77

[Diamond et al., 2017] Diamond, S., Sitzmann, V., Heide, F., and Wetzstein, G. (2017). Unrolled optimization with deep priors. *arXiv preprint arXiv:1705.08041*. 12, 13

[Dinh et al., 2016] Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*. 58

[Dong et al., 2014] Dong, C., Loy, C. C., He, K., and Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer. 86

[Dong et al., 2015] Dong, C., Loy, C. C., He, K., and Tang, X. (2015). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307. 12

[Dong et al., 2016] Dong, C., Loy, C. C., and Tang, X. (2016). Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer. 86

[Donoho and Johnstone, 1994] Donoho, D. L. and Johnstone, J. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455. 16, 27

[Dosovitskiy et al., 2015] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766. 12

[Doucet et al., 2001] Doucet, A., De Freitas, N., and Gordon, N. (2001). An introduction to sequential monte carlo methods. *Sequential Monte Carlo methods in practice*, pages 3–14. 110

[Durmus et al., 2018] Durmus, A., Moulines, E., and Pereyra, M. (2018). Efficient bayesian computation by proximal markov chain monte carlo: when langevin meets moreau. *SIAM Journal on Imaging Sciences*, 11(1):473–506. 15

[Gatopoulos et al., 2020] Gatopoulos, I., Stol, M., and Tomczak, J. M. (2020). Super-resolution variational auto-encoders. *arXiv preprint arXiv:2006.05218*. 86, 88

[Geman and Yang, 1995] Geman, D. and Yang, C. (1995). Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946. 15

[González et al., 2022] González, M., Almansa, A., and Tan, P. (2022). Solving inverse problems by joint posterior maximization with autoencoding prior. *SIAM Journal on Imaging Sciences*, 15(2):822–859. 8, 9, 28, 57, 58, 59, 60, 61, 64, 88

[Goodfellow et al., 2014a] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial nets. *Advances in neural information processing systems*, 27. 9, 19, 86, 88

[Goodfellow et al., 2020] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144. 58

[Goodfellow et al., 2014b] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014b). Generative adversarial networks. *arXiv preprint arXiv:1406.2661*. 31

[Gregor and LeCun, 2010] Gregor, K. and LeCun, Y. (2010). Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on international conference on machine learning*, pages 399–406. 12

[Gulrajani et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of wasserstein gans. In *Adv. in Neural Information Processing Systems*, volume 30, pages 5767–5777. 6, 32

[Hand et al., 2018] Hand, P., Leong, O., and Voroninski, V. (2018). Phase retrieval under a generative prior. *Advances in Neural Information Processing Systems*, 31. 58

[Haris et al., 2018] Haris, M., Shakhnarovich, G., and Ukita, N. (2018). Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1664–1673. 86

[Harvey et al., 2022] Harvey, W., Naderiparizi, S., and Wood, F. (2022). Conditional image generation by conditioning variational auto-encoders. In *International Conference on Learning Representations*. 86, 87, 88, 93, 141, 142

[Havtorn et al., 2021a] Havtorn, J. D., Frellsen, J., Hauberg, S., and Maaløe, L. (2021a). Hierarchical vaes know what they don't know. *ArXiv*, abs/2102.08248. 90

[Havtorn et al., 2021b] Havtorn, J. D., Frellsen, J., Hauberg, S., and Maaløe, L. (2021b). Hierarchical vaes know what they don't know. In *International Conference on Machine Learning*, pages 4117–4128. PMLR. 56

[Hazami et al., 2022] Hazami, L., Mama, R., and Thurairatnam, R. (2022). Efficient-vdvae: Less is more. *arXiv preprint arXiv:2203.13751*. 56, 78

[Ho et al., 2020] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851. 19, 20, 58, 76

[Holden et al., 2022] Holden, M., Pereyra, M., and Zygalakis, K. C. (2022). Bayesian imaging with data-driven priors encoded by neural networks. *SIAM Journal on Imaging Sciences*, 15(2):892–924. 88

[Houdard et al., 2018] Houdard, A., Bouveyron, C., and Delon, J. (2018). High-dimensional mixture models for unsupervised image denoising (HDMI). *SIAM J. Imag. Sc.*, 11(4):2815–2846. 28, 30

[Huang et al., 2021] Huang, W., Hand, P., Heckel, R., and Voroninski, V. (2021). A provably convergent scheme for compressive sensing under random generative priors. *Journal of Fourier Analysis and Applications*, 27:1–34. 56

[Hurault et al., 2018] Hurault, S., Ehret, T., and Arias, P. (2018). EPLL: An Image Denoising Method Using a Gaussian Mixture Model Learned on a Large Set of Patches. *Image Processing On Line*, 8:465–489. 38

[Hurault et al., 2021] Hurault, S., Leclaire, A., and Papadakis, N. (2021). Gradient step denoiser for convergent plug-and-play. *arXiv preprint arXiv:2110.03220*. 108

[Hurault et al., 2022] Hurault, S., Leclaire, A., and Papadakis, N. (2022). Gradient step denoiser for convergent plug-and-play. In *International Conference on Learning Representations*. 58, 79, 81

[Hyun and Heo, 2020] Hyun, S. and Heo, J.-P. (2020). Varsr: Variational super-resolution network for very low resolution images. In *European Conference on Computer Vision*, pages 431–447. Springer. 86, 88

[Jiang, 2022] Jiang, L. (2022). Image super-resolution via iterative refinement. `https://github.com/Janspiry/Image-Super-Resolution-via-Iterative-Refinement`. 98

[Kamilov et al., 2017] Kamilov, U. S., Mansour, H., and Wohlberg, B. (2017). A plug-and-play priors approach for solving nonlinear imaging inverse problems. *IEEE Signal Processing Letters*, 24(12):1872–1876. 28

[Karras et al., 2019] Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410. 22, 74, 75, 77, 87, 90, 98

[Karras et al., 2020] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119. 50, 76

[Kawar et al., 2022a] Kawar, B., Elad, M., Ermon, S., and Song, J. (2022a). Denoising Diffusion Restoration Models. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, volume 2020-Decem. 21

[Kawar et al., 2022b] Kawar, B., Elad, M., Ermon, S., and Song, J. (2022b). Denoising diffusion restoration models. *arXiv preprint arXiv:2201.11793*. 86, 88

[Khemakhem et al., 2020] Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. (2020). Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR. 44

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 21, 72

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations*. 33, 34

[Kingma and Dhariwal, 2018] Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31. 50, 75, 77

[Kingma et al., 2016] Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29. 95

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. 7, 8, 18, 42, 45, 46, 58, 86

[Kingma et al., 2019] Kingma, D. P., Welling, M., et al. (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392. 45

[Kobyzev et al., 2020] Kobyzev, I., Prince, S. J., and Brubaker, M. A. (2020). Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979. 19

[Köhler et al., 2014] Köhler, R., Schuler, C., Schölkopf, B., and Harmeling, S. (2014). Mask-specific inpainting with deep neural networks. In *Pattern Recognition: 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings 36*, pages 523–534. Springer. 13

[Krizhevsky et al., 2009] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images. 22

[Krizhevsky et al., 2017] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90. 12

[Kullback and Leibler, 1951] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86. 45

[Latorre et al., 2019] Latorre, F., Cevher, V., et al. (2019). Fast and provable admm for learning with generative priors. *Advances in Neural Information Processing Systems*, 32. 58

[Laumont et al., 2022] Laumont, R., de Bortoli, V., Almansa, A., Delon, J., Durmus, A., and Pereyra, M. (2022). Bayesian imaging using Plug & Play priors: when Langevin meets Tweedie. *SIAM Journal on Imaging Sciences*, 15(12):701–737. 18, 23, 86

[Lebrun, 2012] Lebrun, M. (2012). An Analysis and Implementation of the BM3D Image Denoising Method. *Image Processing On Line*, 2:175–213. 38

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324. 12, 48

[Ledig et al., 2017] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690. 86

[Levin et al., 2009] Levin, A., Weiss, Y., Durand, F., and Freeman, W. T. (2009). Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1964–1971. IEEE. 79, 137

[Li et al., 2021] Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*. 12

[Liang et al., 2021] Liang, J., Lugmayr, A., Zhang, K., Danelljan, M., Van Gool, L., and Timofte, R. (2021). Hierarchical conditional flow: A unified framework for image super-resolution and image rescaling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4076–4085. 86, 87, 98

[Lim et al., 2017] Lim, B., Son, S., Kim, H., Nah, S., and Lee, K. M. (2017). Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 77

[Liu et al., 2018] Liu, Z., Luo, P., Wang, X., and Tang, X. (2018). Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11. 22, 75

[Liu et al., 2020] Liu, Z.-S., Siu, W.-C., and Chan, Y.-L. (2020). Photo-realistic image super-resolution via variational autoencoders. *IEEE Transactions on Circuits and Systems for video Technology*, 31(4):1351–1365. 86

[Lugmayr et al., 2020] Lugmayr, A., Danelljan, M., Gool, L. V., and Timofte, R. (2020). Srflow: Learning the super-resolution space with normalizing flow. In *European conference on computer vision*, pages 715–732. Springer. 86, 87

[Luhman and Luhman, 2022] Luhman, E. and Luhman, T. (2022). Optimizing hierarchical image vaes for sample quality. *arXiv preprint arXiv:2210.10205*. 56, 77, 79

[Lunz et al., 2018] Lunz, S., Öktem, O., and Schönlieb, C.-B. (2018). Adversarial regularizers in inverse problems. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8516–8525. 6, 16, 28, 32, 33

[Marino et al., 2018] Marino, J., Yue, Y., and Mandt, S. (2018). Iterative amortized inference. In *International Conference on Machine Learning*, pages 3403–3412. PMLR. 61

[Martin et al., 2001] Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423. 79, 81

[Mattei and Frellsen, 2018] Mattei, P.-A. and Frellsen, J. (2018). Leveraging the exact likelihood of deep latent variable models. *Advances in Neural Information Processing Systems*, 31. 109

[Meinhardt et al., 2017] Meinhardt, T., Moller, M., Hazirbas, C., and Cremers, D. (2017). Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1781–1790. 17, 22, 28

[Menon et al., 2020] Menon, S., Damian, A., Hu, S., Ravi, N., and Rudin, C. (2020). Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 2437–2445. 22, 56, 58, 86, 88

[Minaee et al., 2021] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3523–3542. 12

[Mittal et al., 2012] Mittal, A., Moorthy, A. K., and Bovik, A. C. (2012). No-reference image quality assessment in the spatial domain. *IEEE Transactions on image processing*, 21(12):4695–4708. 99

[Mukherjee et al., 2023] Mukherjee, S., Hauptmann, A., Öktem, O., Pereyra, M., and Schönlieb, C.-B. (2023). Learned reconstruction methods with convergence guarantees: A survey of concepts and applications. *IEEE Signal Processing Magazine*, 40(1):164–182. 17

[Nair and Hinton, 2010] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*. 33

[Oberlin and Verm, 2021] Oberlin, T. and Verm, M. (2021). Regularization via deep generative models: an analysis point of view. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 404–408. 58

[Ohayon et al., 2021] Ohayon, G., Adrai, T., Vaksman, G., Elad, M., and Milanfar, P. (2021). High perceptual quality image denoising with a posterior sampling cgan. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1805–1813. 87

[Olah et al., 2017] Olah, C., Mordvintsev, A., and Schubert, L. (2017). Feature visualization. *Distill*. https://distill.pub/2017/feature-visualization. 48

[Olmedo, 2021] Olmedo, M. G. (2021). *Bayesian Plug & Play Methods for Inverse Problems in Imaging*. PhD thesis, Université de Paris. 109

[Pan et al., 2021] Pan, X., Zhan, X., Dai, B., Lin, D., Loy, C. C., and Luo, P. (2021). Exploiting deep generative prior for versatile image restoration and manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7474–7489. 58

[Paszke et al., 2017] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. 34

[Pereyra et al., 2015] Pereyra, M., Schniter, P., Chouzenoux, E., Pesquet, J.-C., Tourneret, J.-Y., Hero, A. O., and McLaughlin, S. (2015). A survey of stochastic simulation and optimization methods in signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):224–241. 15

[Pereyra et al., 2022] Pereyra, M., Vargas-Mieles, L. A., and Zygalakis, K. C. (2022). The split gibbs sampler revisited: improvements to its algorithmic structure and augmented target distribution. *arXiv preprint arXiv:2206.13894*. 15

[Pesquet et al., 2021] Pesquet, J.-C., Repetti, A., Terris, M., and Wiaux, Y. (2021). Learning maximally monotone operators for image recovery. *SIAM Journal on Imaging Sciences*, 14(3):1206–1237. 58, 79, 81

[Prakash et al., 2021] Prakash, M., Delbracio, M., Milanfar, P., and Jug, F. (2021). Interpretable unsupervised diversity denoising and artefact removal. *arXiv preprint arXiv:2104.01374*. 77, 83, 88

[Prost et al., 2021] Prost, J., Houdard, A., Almansa, A., and Papadakis, N. (2021). Learning local regularization for variational image restoration. In *Scale Space and Variational Methods in Computer Vision: 8th International Conference, SSVM 2021, Virtual Event, May 16–20, 2021, Proceedings*, pages 358–370. Springer. 77

[Raj et al., 2019] Raj, A., Li, Y., and Bresler, Y. (2019). Gan-based projector for faster recovery with convergence guarantees in linear inverse problems. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5602–5611. 58

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788. 12

[Reehorst and Schniter, 2019] Reehorst, E. T. and Schniter, P. (2019). Regularization by Denoising: Clarifications and New Interpretations. *IEEE Trans. on Computational Imaging*, 5(1):52–67. 28

[Reynolds et al., 2009] Reynolds, D. A. et al. (2009). Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663). 45

[Rezende and Mohamed, 2015] Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR. 19, 46, 86

[Robbins, 1992] Robbins, H. E. (1992). An empirical bayes approach to statistics. In *Breakthroughs in Statistics: Foundations and basic theory*, pages 388–394. Springer. 17

[Rolinek et al., 2019] Rolinek, M., Zietlow, D., and Martius, G. (2019). Variational autoencoders pursue pca directions (by accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12406–12415. 44

[Romano et al., 2017a] Romano, Y., Elad, M., and Milanfar, P. (2017a). The little engine that could: Regularization by denoising (red). *SIAM J. on Im. Sc.*, 10(4):1804–1844. 58

[Romano et al., 2017b] Romano, Y., Elad, M., and Milanfar, P. (2017b). The Little Engine That Could: Regularization by Denoising (RED). *SIAM J. on Imaging Sciences*, 10(4):1804–1844. 28

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386. 12

[Roth and Black, 2005] Roth, S. and Black, M. J. (2005). Fields of experts: A framework for learning image priors. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 860–867. IEEE. 16

[Rudin et al., 1992a] Rudin, L. I., Osher, S., and Fatemi, E. (1992a). Nonlinear total variation based noise removal algorithms. *Phys. D*, 60:259–268. 6

[Rudin et al., 1992b] Rudin, L. I., Osher, S., and Fatemi, E. (1992b). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268. 16, 27

[Ryu et al., 2019] Ryu, E. K., Liu, J., Wang, S., Chen, X., Wang, Z., and Yin, W. (2019). Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. 22, 28, 58, 70

[Saharia et al., 2021a] Saharia, C., Chan, W., Chang, H., Lee, C. A., Ho, J., Salimans, T., Fleet, D. J., and Norouzi, M. (2021a). Palette: Image-to-image diffusion models. *arXiv preprint arXiv:2111.05826*. 21, 88

[Saharia et al., 2021b] Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. (2021b). Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*. 21, 86, 88, 98, 100

[Saharia et al., 2021c] Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. (2021c). Image Super-Resolution via Iterative Refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–14. 56

[Santambrogio, 2015] Santambrogio, F. (2015). Optimal transport for applied mathematicians. *Progress in Nonlinear Differential Equations and their applications*, 87. 32

[Seonghyeon, 2020] Seonghyeon, K. (2020). stylegan2-pytorch. `https://github.com/rosinality/stylegan2-pytorch`. 76

[Shah and Hegde, 2018] Shah, V. and Hegde, C. (2018). Solving linear inverse problems using gan priors: An algorithm with provable guarantees. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4609–4613. IEEE. 21, 58

[Shu et al., 2018] Shu, R., Bui, H. H., Zhao, S., Kochenderfer, M. J., and Ermon, S. (2018). Amortized inference regularization. *Advances in Neural Information Processing Systems*, 31. 46

[Sohl-Dickstein et al., 2015] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR. 9, 19

[Sønderby et al., 2016] Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., and Winther, O. (2016). Ladder variational autoencoders. *Advances in neural information processing systems*, 29. 8, 49, 82, 89, 95

[Song et al., 2023] Song, J., Vahdat, A., Mardani, M., and Kautz, J. (2023). Pseudoinverse-Guided Diffusion Models for Inverse Problems. In *(ICLR) International Conference on Learning Representations*. 21

[Song and Ermon, 2019] Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32. 19

[Song et al., 2021a] Song, Y., Shen, L., Xing, L., and Ermon, S. (2021a). Solving inverse problems in medical imaging with score-based generative models. *arXiv preprint arXiv:2111.08005*. 58

[Song et al., 2020] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*. 76, 77

[Song et al., 2021b] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021b). Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*. 19, 20

[Teodoro et al., 2018] Teodoro, A. M., Bioucas-Dias, J. M., and Figueiredo, M. A. T. (2018). Scene-Adapted Plug-and-Play Algorithm with Guaranteed Convergence: Applications to Data Fusion in Imaging. 28

[Thanh-Tung and Tran, 2020] Thanh-Tung, H. and Tran, T. (2020). Catastrophic forgetting and mode collapse in gans. In *2020 international joint conference on neural networks (ijcnn)*, pages 1–10. IEEE. 20

[Toussaint, 2011] Toussaint, M. (2011). Lecture notes: Gaussian identities. *a A*, 1:2. 67, 128

[Vahdat and Kautz, 2020] Vahdat, A. and Kautz, J. (2020). Nvae: A deep hierarchical variational autoencoder. *Advances in neural information processing systems*, 33:19667–19679. 50, 56, 62, 75, 77, 86, 90, 95

[Van Dyk and Park, 2008] Van Dyk, D. A. and Park, T. (2008). Partially collapsed gibbs samplers: Theory and methods. *Journal of the American Statistical Association*, 103(482):790–796. 110

[Venkatakrishnan et al., 2013] Venkatakrishnan, S. V., Bouman, C. A., and Wohlberg, B. (2013). Plug-and-play priors for model based reconstruction. In *IEEE Global Conference on Signal and Information Processing*, pages 945–948. 17, 28, 31, 57, 58

[Vincent, 2011] Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674. 17, 20

[Vono et al., 2019] Vono, M., Dobigeon, N., and Chainais, P. (2019). Split-and-augmented gibbs sampler—application to large-scale inference problems. *IEEE Transactions on Signal Processing*, 67(6):1648–1661. 15

[Wang et al., 2018] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., and Change Loy, C. (2018). Esrgan: Enhanced super-resolution generative adversarial

networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0. 86

[Wang et al., 2004] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612. 99

[Wang et al., 2020] Wang, Z., Chen, J., and Hoi, S. C. (2020). Deep learning for image super-resolution: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3365–3387. 22, 86

[Wang et al., 2015] Wang, Z., Liu, D., Yang, J., Han, W., and Huang, T. (2015). Deep networks for image super-resolution with sparse prior. In *Proceedings of the IEEE international conference on computer vision*, pages 370–378. 12

[Weyand et al., 2020] Weyand, T., Araujo, A., Cao, B., and Sim, J. (2020). Google landmarks dataset v2 – a large-scale benchmark for instance-level recognition and retrieval. 33

[Yosinski et al., 2015] Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. (2015). Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*. 48

[Yu and Sapiro, 2011] Yu, G. and Sapiro, G. (2011). DCT image denoising: a simple and effective image denoising algorithm. *Image Processing On Line*, 1. 16, 27

[Zhang et al., 2018a] Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. (2018a). Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026. 45

[Zhang et al., 2021] Zhang, K., Li, Y., Zuo, W., Zhang, L., Van Gool, L., and Timofte, R. (2021). Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376. 17, 22, 58, 70

[Zhang et al., 2017a] Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. (2017a). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155. 13, 17

[Zhang et al., 2017b] Zhang, K., Zuo, W., Gu, S., and Zhang, L. (2017b). Learning Deep CNN Denoiser Prior for Image Restoration. In *IEEE Conf. Comput. Vis. Pat. Recog.*, pages 2808–2817. 28

[Zhang et al., 2022] Zhang, M., Hayes, P., and Barber, D. (2022). Generalization gap in amortized inference. *Advances in Neural Information Processing Systems*, 35:26777–26790. 61

[Zhang et al., 2018b] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018b). The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595. 38, 76, 99

[Zhao et al., 2017] Zhao, S., Song, J., and Ermon, S. (2017). Learning hierarchical features from deep generative models. In *International Conference on Machine Learning*, pages 4091–4099. PMLR. 47

[Zoran and Weiss, 2011a] Zoran, D. and Weiss, Y. (2011a). From learning models of natural image patches to whole image restoration. In *2011 international conference on computer vision*, pages 479–486. IEEE. 16, 77, 79, 81

[Zoran and Weiss, 2011b] Zoran, D. and Weiss, Y. (2011b). From learning models of natural image patches to whole image restoration. In *Int. Conference on Computer Vision*, pages 479–486. 28, 29, 30, 38

Appendix A

Proofs of chapter 5

Code

The code for this project can be found on https://github.com/jprost76/PnP-HVAE

Summary

This supplementary material contains:

- proofs of the theoretical results of the main paper in section A.1
- additional implementation details in section A.2
- a discussion on the contractivity of the autoencoder and its fixed points in section A.3
- additional comparisons with the competing methods in section A.4

A.1 Proofs of the main results

In this section we provide proofs relative to Proposition 5.1, Proposition 5.4, Proposition 5.5 and the characterization of the fixed point given by Algorithm 4.

A.1.1 Proof of Proposition 5.1(Posterior of the low-temperature hierarchical model)

By definition of the joint model (5.14), the low-temperature likelihood verifies $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}|\boldsymbol{z}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})$. Hence the model posterior writes:

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})} \tag{A.1}$$

$$= \frac{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}\frac{p_\theta(\boldsymbol{z}|\boldsymbol{x})p_\theta(\boldsymbol{x})}{p_\theta(\boldsymbol{z})} \tag{A.2}$$

$$= \frac{p_\theta(\boldsymbol{x})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}\frac{p_\theta(\boldsymbol{z}|\boldsymbol{x})p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z})}{p_\theta(\boldsymbol{z})} \tag{A.3}$$

$$= \frac{p_\theta(\boldsymbol{x})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}\frac{\left(p_\theta(\boldsymbol{z}_0|\boldsymbol{x})\prod_{\ell=1}^{L-1}p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})\right)\left(\frac{p_\theta(z_0)^{\frac{1}{\tau_0^2}}}{Z_0}\prod_{\ell=1}^{L-1}\frac{p_\theta(z_\ell|z_{<\ell})^{\frac{1}{\tau_0^2}}}{Z_\ell}\right)}{\left(p_\theta(\boldsymbol{z}_0)\prod_{\ell=1}^{L-1}p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})\right)} \tag{A.4}$$

$$= \frac{p_\theta(\boldsymbol{x})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}\frac{1}{Z_0}p_\theta(\boldsymbol{z}_0|\boldsymbol{x})p_\theta(\boldsymbol{z}_0)^{\frac{1}{\tau_0^2}-1}\prod_{\ell=1}^{L-1}\frac{1}{Z_\ell}p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\frac{1}{\tau_\ell^2}-1} \tag{A.5}$$

A.1.2 Low temperature encoder

Proposition A.1. *The low temperature encoder conditionals $q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ are unormalized Gaussian probability density function (PDF):*

$$q_{\phi,\tau}(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x}) = \frac{1}{E_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x})}\exp\left(-\frac{1}{2}\left(\boldsymbol{z}_\ell-\mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right)^t\Sigma^{-1}_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\left(\boldsymbol{z}_\ell-\mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right)\right) \tag{A.6}$$

with

$$E_\ell(\boldsymbol{z}_{<\ell},\boldsymbol{x}) = \left((2\pi)^{n_l}\left|\Sigma_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right|\right)^{\frac{1}{2}}\left((2\pi)^{n_l}\left|\Sigma_{\theta,\ell}\left(\boldsymbol{z}_{<\ell}\right)\right|\right)^{\frac{\lambda_\ell}{2}}Z_\ell \tag{A.7}$$

$$= C_\ell|\Sigma_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)|^{\frac{1}{2}}|\Sigma_{\theta,\ell}\left(\boldsymbol{z}_{<\ell}\right)|^{\frac{\lambda_\ell}{2}} \tag{A.8}$$

$$\Sigma_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) = \left(\Sigma^{-1}_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)+\lambda_\ell\Sigma^{-1}_{\theta,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\right)^{-1} \tag{A.9}$$

$$\mu_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right) = \Sigma_{\phi,\tau,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\left(\Sigma^{-1}_{\phi,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\mu_\phi\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)+\lambda_\ell\Sigma^{-1}_{\theta,\ell}\left(\boldsymbol{z}_{<\ell},\boldsymbol{x}\right)\mu_\theta\left(\boldsymbol{z}_{<\ell}\right)\right) \tag{A.10}$$

Proof. This comes from the fact that the product of two univariate Gaussian PDF is an unormalized Gaussian PDF (see for instance [Bromiley, 2003, Toussaint, 2011]). This result can be extended for multivariate Gaussian PDF with diagonal covariance matrices, as it is the case for $q_\phi(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell},\boldsymbol{x})$ and $p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})$. □

A.1.3 Proof of Proposition 5.5 (fixed point of PnP-HVAE)

Proof. $\boldsymbol{x}^*$ is a fixed point of T if and only if $\boldsymbol{x}^* = T(\boldsymbol{x}^*)$. Recalling the definition of $\mathrm{T}(\boldsymbol{x}) := \mathrm{prox}_{\gamma^2} f\left(\mathrm{HVAE}\left(\boldsymbol{x}, \boldsymbol{\tau}\right)\right)$, and the definition of proximal operator $\mathrm{prox}_{\gamma^2 f}(\boldsymbol{x}) = \arg\min_{\boldsymbol{u}} \gamma^2 f(\boldsymbol{u}) + \frac{1}{2}||\boldsymbol{x} - \boldsymbol{t}||^2$, the fixed point condition is equivalent to

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{u}} \frac{1}{2}\|\boldsymbol{u} - \mathrm{HVAE}\left(\boldsymbol{x}^*, \boldsymbol{\tau}\right)\|^2 + \gamma^2 f(\boldsymbol{u}).$$

Since f is convex the above condition is equivalent to

$$\boldsymbol{x}^* - \mathrm{HVAE}\left(\boldsymbol{x}^*, \boldsymbol{\tau}\right) + \gamma^2 \nabla f(\boldsymbol{x}^*) = 0.$$

Rearranging the terms we obtain equation (5.57). □

Under mild assumptions the above result can be restated as follows: $\boldsymbol{x}^*$ is a fixed point of T if and only if

$$\nabla f(\boldsymbol{x}^*) + \nabla g(\boldsymbol{x}^*) = 0,$$

i.e. whenever $\boldsymbol{x}^*$ is a *critical point* of the objective function $f(\boldsymbol{x}) + g(\boldsymbol{x}) = -\log p(\boldsymbol{y}|\boldsymbol{x}) - \log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$, where the tempered prior is defined as the marginal

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) = \int p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z}$$

of the joint tempered prior defined in equation (5.14).

This is shown in the next section.

A.1.4 Fixed points are critical points

In this section we characterize fixed points of Algorithm 4 as critical points of a posterior density (a necessary condition to be a MAP estimator), under mild conditions. Before we formulate this caracterization we need to review in more detail a few facts about HVAE training, temperature scaling and our optimization model.

HVAE training. In section 3.1 we introduced how VAEs in general (and HVAEs in particular) are trained. As a consequence an HVAE embeds a joint prior

$$p_\theta(\boldsymbol{x}, \boldsymbol{z}) := p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z}) \tag{A.11}$$

from which we can define a marginal prior on $\boldsymbol{x}$

$$p_\theta(\boldsymbol{x}) := \int p_\theta(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z}. \tag{A.12}$$

In addition, from the ELBO maximization condition in (4.17) and Bayes theorem we can obtain an alternative expression for the joint prior, namely

$$p_\theta(\boldsymbol{x}, \boldsymbol{z}) = q_\phi(\boldsymbol{z}|\boldsymbol{x}) p_{\mathrm{data}}(\boldsymbol{x}). \tag{A.13}$$

Temperature scaling. After training we reduce the temperature by a factor $\boldsymbol{\tau}$, which amounts to replacing $p_\theta(\boldsymbol{z})$ by

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}) := \prod_{l=0}^{L-1} \frac{p_\theta(\boldsymbol{z}_\ell|\boldsymbol{z}_{<\ell})^{\frac{1}{\tau_\ell^2}}}{Z_\ell}$$

as shown in equation (5.14), leading to the joint tempered prior

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{z}) := p_\theta(\boldsymbol{x}|\boldsymbol{z})p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}). \tag{A.14}$$

The corresponding marginal tempered prior on $\boldsymbol{x}$ becomes

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) := \int p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{z})d\boldsymbol{z} \tag{A.15}$$

and the corresponding posterior is

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) := p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{z})/p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}). \tag{A.16}$$

The joint tempered prior also has an alternative expression (based on the encoder). Indeed substituting $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ from equations (A.11) and (A.13) into (A.14) we obtain

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{z}) = \frac{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z})}{p_\theta(\boldsymbol{z})} q_\phi(\boldsymbol{z}|\boldsymbol{x})p_{\text{data}}(\boldsymbol{x}). \tag{A.17}$$

Substituting this result into definition (A.16) we obtain an alternative expression for the tempered posterior

$$p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}|\boldsymbol{x})p_{\text{data}}(\boldsymbol{x})/p_\theta(\boldsymbol{z}). \tag{A.18}$$

Optimization model. Since we are using a scaled prior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ encoded in our HVAE to regularize the inverse problem, the ideal optimization objective we would like to minimize is

$$U(\boldsymbol{x}) := \underbrace{-\log p(\boldsymbol{y}|\boldsymbol{x})}_{f(\boldsymbol{x})} \underbrace{-\log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})}_{g(\boldsymbol{x})}. \tag{A.19}$$

Since $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ is intractable our algorithm seeks to minimize a relaxed objective (see equation (5.18)). Nevertheless, under certain conditions (to be specified below) this is equivalent to minimizing the ideal objective (A.19).

Fixed-point characterization. We start by characterizing $\nabla \log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ in terms of an HVAE-related denoiser (Proposition A.2). Then we relate this denoiser to the quantity $\text{HVAE}(\boldsymbol{x},\tau)$ that is computed by our algorithm (Proposition A.3). As a consequence we obtain that the fixed point condition in Proposition 5.5 can be written as $\nabla U(\boldsymbol{x}) = 0$ (see Corollary 2).

Proposition A.2 (Tweedie's formula for HVAEs.)**.** *For an HVAE with Gaussian decoder* $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; \mu_\theta(\boldsymbol{z}), \gamma^2 I)$*, the following denoiser based on the HVAE with tempered prior*

$$D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) := \int \mu_\theta(\boldsymbol{z})\, p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} \tag{A.20}$$

satisfies Tweeedie's formula

$$D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) - \boldsymbol{x} = \gamma^2 \nabla \log p_{\theta,\boldsymbol{\tau}}(x) = -\gamma^2 \nabla g(\boldsymbol{x}). \tag{A.21}$$

Proof. From the definition of $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ in equation (A.15) we have that

$$\nabla \log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) = \frac{1}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})} \int \nabla_{\boldsymbol{x}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}) d\boldsymbol{z}.$$

From the pdf of the Gaussian decoder $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ its gradient writes

$$\nabla_{\boldsymbol{x}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) = -\frac{1}{\gamma^2}(\boldsymbol{x} - \mu_\theta(\boldsymbol{z})) p_\theta(\boldsymbol{x}|\boldsymbol{z}).$$

Replacing this in the previous equation we get

$$\begin{aligned}
\nabla \log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) &= \frac{1}{\gamma^2} \int (\mu_\theta(\boldsymbol{z}) - \boldsymbol{x}) \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z})}{p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})} d\boldsymbol{z} \\
&= \frac{1}{\gamma^2} \int (\mu_\theta(\boldsymbol{z}) - \boldsymbol{x}) p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} \\
&= \frac{1}{\gamma^2} \left(\int \mu_\theta(\boldsymbol{z})\, p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} - \boldsymbol{x} \right).
\end{aligned}$$

In the second step we used the definitions of the joint tempered prior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z})$ (A.14) and the tempered posterior $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x})$ (A.16). The last step follows from the fact that $\int p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} = 1$ according to definitions (A.16) and (A.15). Finally applying the definition of the denoiser $D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ in the last expression we obtain Tweedie's formula (A.21). □

Under suitable assumptions the denoiser defined above coincides with HVAE$(\boldsymbol{x}, \boldsymbol{\tau})$ computed by our algorithm.

Assumption A.1 (Deterministic encoder)**.** *The covariance matrices of the encoder defined in equation* (4.23) *are 0,* i.e. $\Sigma_{\phi,\ell}(\boldsymbol{z}_{<\ell}, \boldsymbol{x}) = 0$ *for* $l = 0, \dots, L-1$*. Put another way* $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \delta_{E_{\boldsymbol{\tau}}(\boldsymbol{x})}(\boldsymbol{z})$ *is a Dirac centered at* $E_{\boldsymbol{\tau}}(\boldsymbol{x})$*.*

Proposition A.3. *Under Assumption A.1 the function HVAE*$(\boldsymbol{x}, \boldsymbol{\tau})$ *computed by Algorithm 4 coincides with the denoiser* $D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ *defined in equation* (A.20)*.*

Proof. HVAE$(\boldsymbol{x}, \boldsymbol{\tau})$ is defined in Proposition 5.5 as

$$\text{HVAE}(\boldsymbol{x}, \boldsymbol{\tau}) = \mu_\theta \left(E_\tau(\boldsymbol{x}) \right).$$

First observe that for a deterministic encoder we also have $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) = \delta_{E_{\boldsymbol{\tau}}(\boldsymbol{x})}(\boldsymbol{z})$. Indeed for any test function h:

$$\begin{aligned}\int h(\boldsymbol{z}) p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} &= \int h(\boldsymbol{z}) q_\phi(\boldsymbol{z}|\boldsymbol{x}) p_{\text{data}}(\boldsymbol{x}) / p_\theta(\boldsymbol{z}) d\boldsymbol{z} \\ &= h(E_{\boldsymbol{\tau}}(\boldsymbol{x})) \underbrace{p_{\text{data}}(\boldsymbol{x}) / p_\theta(E_{\boldsymbol{\tau}}(\boldsymbol{x}))}_{Z(\boldsymbol{x})}.\end{aligned}$$

And the normalization constant $Z(\boldsymbol{x})$ should be equal to 1 because $\int p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} = Z(\boldsymbol{x}) = 1$. Hence $p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \delta_{E_{\boldsymbol{\tau}}(\boldsymbol{x})}(\boldsymbol{z})$.

Finally applying the definition of $D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$ we obtain

$$\begin{aligned}D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) &= \int \mu_\theta(\boldsymbol{z}) \, p_{\theta,\boldsymbol{\tau}}(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z} = \mu_\theta \left(E_{\boldsymbol{\tau}}(\boldsymbol{x}) \right) \\ &= \text{HVAE}(\boldsymbol{x}, \tau).\end{aligned}$$

□

Combining Propositions A.3, 5.5 and A.2 we obtain a new characterization of fixed points as critical points.

Corollary 2. *Under Assumption A.1 $\boldsymbol{x}^*$ is a fixed point of T if and only if*

$$\nabla f(\boldsymbol{x}^*) + \nabla g(\boldsymbol{x}^*) = 0 \tag{A.22}$$

where $g(\boldsymbol{x}) = -\log p_{\theta,\boldsymbol{\tau}}(\boldsymbol{x})$.

Proof. From Proposition A.2 we have that

$$-\nabla g(\boldsymbol{x}) = \frac{1}{\gamma^2} \left(D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) - \boldsymbol{x} \right).$$

From Proposition A.3 we have that (under Assumption A.1) $D_{\theta,\boldsymbol{\tau}}(\boldsymbol{x}) = \text{HVAE}(\boldsymbol{x}, \tau)$. In combination with the previous result:

$$-\nabla g(\boldsymbol{x}) = \frac{1}{\gamma^2} \left(\text{HVAE}(\boldsymbol{x}, \tau) - \boldsymbol{x} \right).$$

Finally, Proposition 5.5 allows to conclude that

$$-\nabla g(\boldsymbol{x}) = \nabla f(\boldsymbol{x}).$$

□

A.2 Details on PatchVDVAE architecture

In this section, we provide additional details about the architecture of PatchVDVAE. Then, we present the choice of the hyperparameters used for the concurrent methods (presented in section 5.6 of the main paper) .

A.2.1 PatchVDVAE

Figure 5.5 provides a detailed overview of the structure of a PatchVDVAE network. The architecture follows VDVAE model [Child, 2020], except for the first top-down block, in which we replace the constant input by a latent variable sampled from a Gaussian distribution. The architecture presented in Figure 5.5 illustrates the structure of HVAE networks, but the number of blocks is different to the PatchVDVAE network used in our experiments. Our PatchVDVAE top-down path is composed of $L = 30$ top-down blocks of increasing resolution. The image features are upsampled using an unpooling layer every 5 blocks. The first unpooling layer performs a $\times 4$ upsampling, and the following unpooling layers perform $\times 2$ upsampling. The dimension of the filters is 256 in all blocks. In order to save computations in the residual blocks, the 3×3 convolutions are applied on features of reduced channel dimension (divided by 4). 1×1 convolutions are applied before and after the 3×3 convolutions to respectively reduce and increase the number of channels. The latent variables $\boldsymbol{z}_\ell$ are tensors of shape $12 \times H_l \times W_l$, where the resolution H_l, W_l corresponds to the resolution of the corresponding top-down-block. The bottom-up network structure is symmetric to the top-down network, with 5 residual blocks for each scale, and pooling layers between each scale.

A.2.2 Hyperparameters of compared methods

Face image restoration. For ILO, we found that optimizing the first 5 layers of the generative network offered the best trade-off between image quality and consistency with the observation. Hence, we optimize the 5 first layers for 100 iterations each. This choice is different from the official implementation, where they only optimize the 4 first layers for a lower number of iterations, trading restoration performance for speed. For DPS, we set the scale hyper-parameter ζ' (described in subsection C.2 in [Chung et al., 2023]) to $\zeta' = 1$ for the deblurring and super-resolution experiments reported in this paper.

Natural images restoration - Deblurring. For the three tested methods, we use the official implementation provided by the authors, along with the pretrained models. For EPLL, we use the default parameters in the official implementation.

For GS-PnP, using the notation of the paper, we use the suggested hyperparameter $\lambda_\nu = 0.1$ for the motion blur kernels and $\lambda_\nu = 0.75$ for the Gaussian kernels.

For PnP-MMO, we use the denoiser trained on $\sigma_{den} = 0.007$. On deblurring with $\sigma = 2.55$ we use the default parameters in the implementation. for higher noise levels ($\sigma = 7.65$; $\sigma = 12.75$), and we set the strength of the gradient step as $\gamma = \sigma_{den}/(2\sigma||h||)$, where h corresponds to the blur kernel.

Natural images restoration- Inpainting. For EPLL, we use the default parameters provided in the authors matlab code. For GS-PnP, after a grid-search, we chose to set $\lambda_\nu = 1$ and $\sigma_{denoiser} = 10$.

A.3 Discussion on the conctractivity of HVAE

We showed in section 5.5 that PnP-HVAE converges to a fixed point under the assumption that $\boldsymbol{x} \to \text{HVAE}(\boldsymbol{x}, \tau)$ is contractive. If this condition is met, the sequence of u_k defined by $u_{k+1} = HVAE(u_k, \tau)$ should converge to a fixed point. Figure A.1 presents the **evolution of a fixed point iteration** $u_{k+1} = HVAE(u_k, \tau)$. The image is smoothened over the iterations, and finally converges to a piececewise constant image. We used patchVDVAE for this experiment.

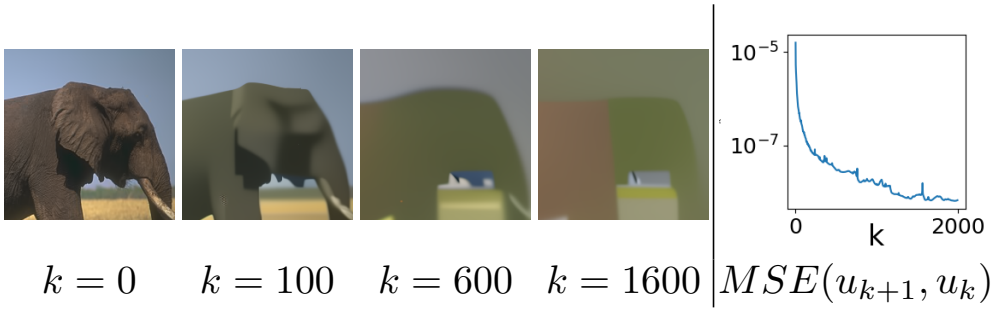


Figure A.1 – Fixed-point iterations of patchVDVAE for $\tau = 0.99$.

A.4 Comparisons

In this section, we provide additional visual results on face images and natural images.

A.4.1 Additional results on face image restoration

We provide additional comparisons with the GAN-based ILO method on inpainting (Figure A.2), $\times 4$ super-resolution (Figure A.3) and deblurring (Figure A.4). PnP-HVAE provides equally or more plausible glasses in the first column) inpaiting than ILO. For superresolution, ILO produces sharper but not realistics faces. This is an agreement with

Figure A.2 – Inpainting

the scores presented in Table 5.1). For deblurring, ILO creates textures on faces that looks realistic (low LPIPS) but are less consistent with the observation (significantly lower PSNR and SSIM).

A.4.2 Additional results on natural images restoration

We finally present additional results on natural images restoration. All the PnP-HVAE images presented below were produced using our PatchVDVAE model. We also provide visual comparisons with concurrent PnP methods and EPLL. For deblurring (Figures A.6 and A.7, PnP methods perform better than EPLL. Following quantitative results of Figure 5.2, for larger noise level, PnP-HVAE outperforms PnP-MMO and provides restoration close to GS-PnP.

For inpainting (Figure A.8), the hierarchical structure of PatchVDVAE leads to more plausible reconstructions, and PnP-HVAE outperforms the compared methods.

Figure A.3 – $\times 4$ super-resolution, with kernel (a) from Figure A.5 and $\sigma = 3$

Figure A.4 – Deblurring, with kernel (d) from Figure A.5 and $\sigma = 8$

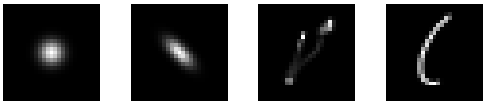
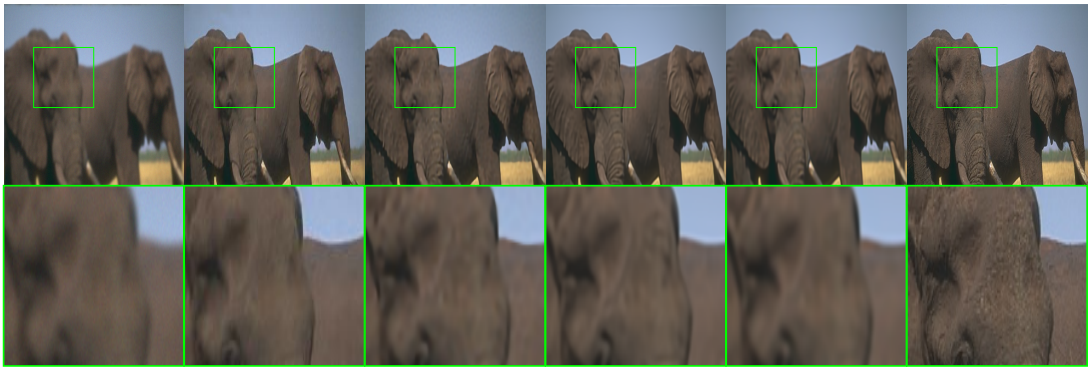


Figure A.5 – Kernels used for deblurring experiments, from [Levin et al., 2009]



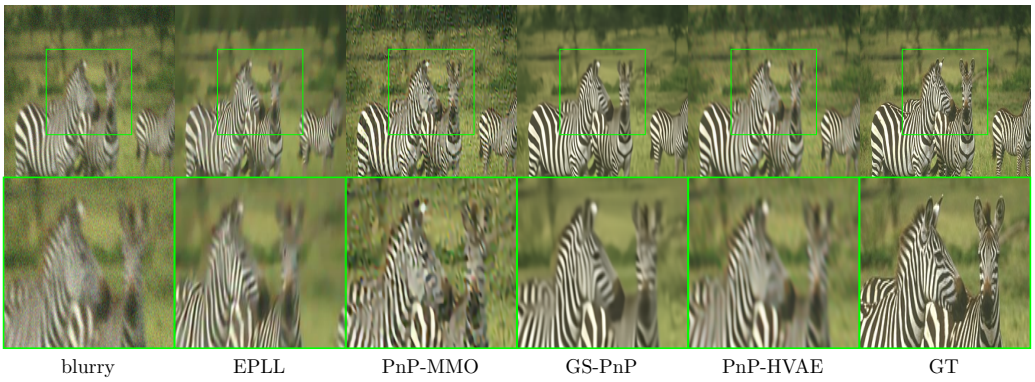
(a) kernel (a), $\sigma = 2.55$

(b) kernel (c), $\sigma = 2.55$

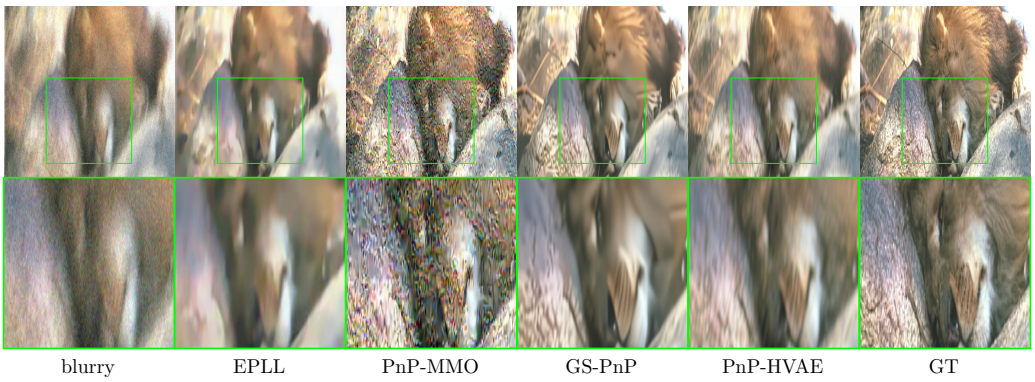
(c) kernel (a), $\sigma = 7.65$

Figure A.6 – Deblurring results on BSD

(a) kernel (d), $\sigma = 7.65$



(b) kernel (b), $\sigma = 12.75$



(c) kernel (d), $\sigma = 12.75$

Figure A.7 – Deblurring results on BSD

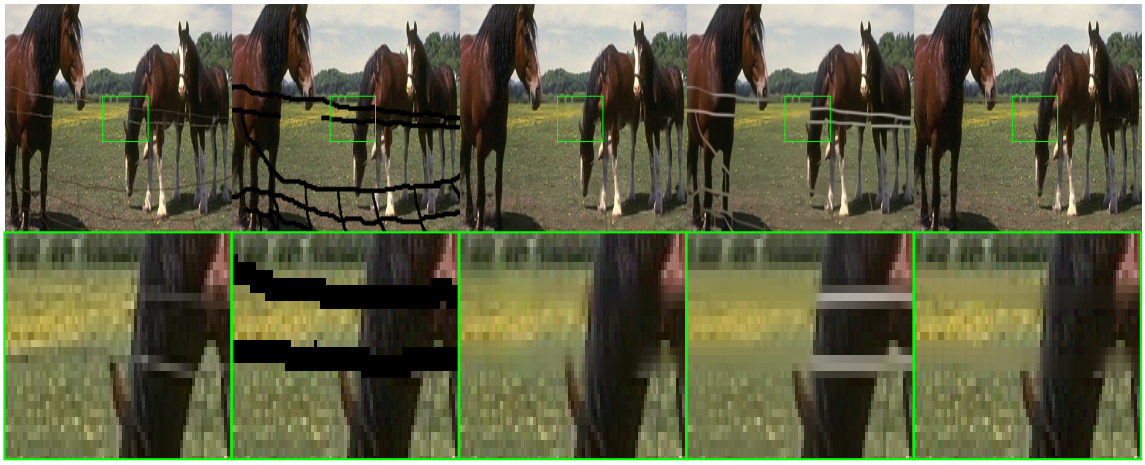


Figure A.8 – Natural images inpainting

Appendix B

Proofs of chapter 6

B.1 Connection between the training criterion and the model conditional log-likelihood

B.1.1 Lower bound on the conditional log-likelihood

In this part we detail the result about the lower-bound on the model conditional log-likelihood given in proposition 6.1, and we link the introduced lower-bound to the training criterion. The conditional log-likelihood of the super-resolution model is defined as:

$$\mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}[\log p_{SR}(\boldsymbol{x}|\boldsymbol{y})]. \tag{B.1}$$

Proposition B.1 (6.1)**.** *The conditional log-likelihood of the super-resolution model on a joint distribution $p_{data}(\boldsymbol{x},\boldsymbol{y})$ is lower-bounded by*

$$\mathcal{O}(\psi) = \mathbb{E}_{p_{data}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\right] \tag{B.2}$$

$$\leq \mathbb{E}_{p_{data}(\boldsymbol{x},\boldsymbol{y})}[\log p_{SR}(\boldsymbol{x}|\boldsymbol{y})]. \tag{B.3}$$

Proof. It is shown in [Harvey et al., 2022] that, for a conditional model written as:

$$p_{cond}(\boldsymbol{x}|\boldsymbol{y}) := \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{y})}[p_\theta(\boldsymbol{x}|\boldsymbol{z})], \tag{B.4}$$

the conditional log-likelihood on a paired data distribution $p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})$ is lower-bounded as[1]:

$$\mathcal{O}(\psi) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})q_\psi(\boldsymbol{z}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{B.5}$$

$$\leq \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}[\log p_{SR}(\boldsymbol{x}|\boldsymbol{y})]. \tag{B.6}$$

[1] In [Harvey et al., 2022], the lower bound is also defined as a function of the VAE encoder and decoder $\mathcal{O}(\theta,\phi,\psi)$. We omit the dependance on θ and ϕ since we keep those parameters constant.

Applying relation (B.5) to the truncated VAE $p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})$, $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ and the truncated latent restoration model $p_{SR}(\boldsymbol{x}|\boldsymbol{y})$ (6.9), we then have:

$$\mathcal{O}(\psi) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\right] \tag{B.7}$$

$$\leq \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}[\log p_{SR}(\boldsymbol{x}|\boldsymbol{y})]. \tag{B.8}$$

□

B.1.2 Relation between the lower-bound and the training criterion

The lower bound (B.2) can be rewritten as:

$$\mathcal{O}(\psi) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})] \tag{B.9}$$

$$+ \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\right] \tag{B.10}$$

$$= C - \mathcal{L}(\psi). \tag{B.11}$$

Therefore, for fixed θ and ϕ, minimizing $\mathcal{L}(\psi)$ amounts to maximizing the lower bound $\mathcal{O}(\psi)$ in ψ.

B.2 Expected consistency of the super-resolution model

In this section we demonstrate Proposition 6.2 on the expected consistency of the super-resolution model. To that end, we first give an intermediate result concerning optimal VAEs.

Proposition B.2 (proof in appendix B.1 of [Harvey et al., 2022])**.** *The ELBO loss of a VAE* (4.9) *can be written as:*

$$\mathcal{L}_{elbo}(\theta,\phi) = -\mathcal{H}(p_{data}(\boldsymbol{x})) - \text{KL}(p_{data}(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})), \tag{B.12}$$

where $\mathcal{H}(p_{data}(\boldsymbol{x}))$ is the entropy of the data distribution.

The formulation (B.12) indicates that maximizing the ELBO loss amounts to reducing the KL divergence from $p_{\text{data}}(\boldsymbol{x})q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to $p_\theta(\boldsymbol{z},\boldsymbol{x})$.

B.2.1 Optimal low-resolution encoder

We remind the reader that we denote

$$r(\boldsymbol{z}_{<k}, \boldsymbol{x}, \boldsymbol{y}) := p_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}) q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x}) \tag{6.14}$$

the joint distribution of high-resolution and low-resolution image pairs $(\boldsymbol{x}, \boldsymbol{y})$, and their latent variable $\boldsymbol{z}$ given by the high-resolution encoder, and $r(\boldsymbol{z}_{<k}|\boldsymbol{y})$ the corresponding conditional distribution. We now recall the technical assumptions made in section 6.5.2.

Assumption 6.1. *There exists $\psi \in \Psi$ which satisfies $r(\boldsymbol{z}_{<k}|\boldsymbol{y}) = q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all $\boldsymbol{y}$ in the support of $p_{data}(\boldsymbol{y})$.*

Assumption 6.2. *The low-resolution encoder parameters ψ are minimizers of the training criterion* (6.12)*:*

$$\psi \in \arg\min_{\tilde{\psi}} \mathcal{L}(\tilde{\psi}). \tag{6.15}$$

Assumption 6.3. *The VAE encoder $q_\phi(\boldsymbol{x}|\boldsymbol{z})$ and generative model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$ have enough capacity and are trained well enough so that ϕ and θ reaches the upper bound of the ELBO loss* (B.12)*.*

In the next proposition, we give the value of the optimal low-resolution encoder.

Proposition B.3. *Under assumptions 6.1 and 6.2, we have:*

$$q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}) = r(\boldsymbol{z}_{<k}|\boldsymbol{y}) \tag{B.13}$$

for all $\boldsymbol{y}$ in the support of $p_{data}(\boldsymbol{y})$.

Proof. The training criterion (6.12) can be written as:

$$\begin{aligned}
\mathcal{L}(\psi) =& \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\left[\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}{q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}\right]\right] && \text{(B.14)}\\
=& \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\left[\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}{r(\boldsymbol{z}_{<k}|\boldsymbol{y})}\right]\right] \\
&+ \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\left[\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{r(\boldsymbol{z}_{<k}|\boldsymbol{y})}{q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})}\right]\right] && \text{(B.15)}\\
=& \mathbb{E}_{p_{\text{data}}(\boldsymbol{x},\boldsymbol{y})}\left[\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}{r(\boldsymbol{z}_{<k}|\boldsymbol{y})}\right]\right] \\
&+ \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})}[\text{KL}(r(\boldsymbol{z}_{<k}|\boldsymbol{y})||q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}))]. && \text{(B.16)}
\end{aligned}$$

Thus $\mathcal{L}(\psi)$ is lower-bounded by the first term of the right-hand side of (B.16). If the encoder has enough capacity (6.1) and by non-negativity of the KL-divergence, the lower bound is reached (6.2) if and only if the second term of the right handside of (B.16) is zero, or equivalently $r(\boldsymbol{z}_{<k}|\boldsymbol{y}) = q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all $\boldsymbol{y}$ in the support of $p_{\text{data}}(\boldsymbol{y})$. □

Proposition B.3 states that, if the low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ has enough capacity and is trained so that it minimizes the training criterion (6.12), it matches the intractable distribution $r(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all images of the training distribution.

With a slight abuse of notation, let us now denote:

$$p_\theta(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) = p_\theta(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{y}|\boldsymbol{x}) \tag{B.17}$$

the VAE model distribution of the latent variables $\boldsymbol{z}$ and the generated high-resolution images $\boldsymbol{x}$, combined with their low-resolution counterpart $\boldsymbol{y}$ given by the degradation model (6.1). In the next proposition we show that, under additional hypothesis on the pretrained VAE, the low-resolution encoder $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ matches the conditional $p_\theta(\boldsymbol{z}_{<k}|\boldsymbol{y})$ of the model distribution (B.17).

Proposition B.4. *Under assumptions 6.1, 6.2 and 6.3, we have*

$$q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}) = p_\theta(\boldsymbol{z}_{<k}|\boldsymbol{y}) \tag{B.18}$$

for all $\boldsymbol{y}$ in the support of $p_{data}(\boldsymbol{y})$.

Proof. Refering to the definition of $r(\boldsymbol{x}, \boldsymbol{z}_{<k}, \boldsymbol{y})$ (6.14) and $p_\theta(\boldsymbol{z}_{<k}, \boldsymbol{x}, \boldsymbol{y})$ (B.17), we have, from assumption 6.3:

$$\begin{aligned}
&\mathrm{KL}(r(\boldsymbol{z}, \boldsymbol{x}) || p_\theta(\boldsymbol{z}, \boldsymbol{x})) = 0 && \text{(B.19)}\\
\implies &r(\boldsymbol{z}, \boldsymbol{x}) = p_\theta(\boldsymbol{z}, \boldsymbol{x}) && \text{(B.20)}\\
\implies &r(\boldsymbol{z}, \boldsymbol{x}) p_{\mathrm{data}}(\boldsymbol{y}|\boldsymbol{x}) = p_\theta(\boldsymbol{z}, \boldsymbol{x}) p_{\mathrm{data}}(\boldsymbol{y}|\boldsymbol{x}) && \text{(B.21)}\\
\implies &r(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) = p_\theta(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) && \text{(B.22)}\\
\implies &r(\boldsymbol{z}|\boldsymbol{y}) = p_\theta(\boldsymbol{z}|\boldsymbol{y}). && \text{(B.23)}
\end{aligned}$$

Furthermore, using proposition B.3, assumptions 6.1 and 6.2 imply that $r(\boldsymbol{z}_{<k}|\boldsymbol{y}) = q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all $\boldsymbol{y}$ in the support of $p_{\mathrm{data}}(\boldsymbol{y})$. Thus, we have:

$$\begin{aligned}
r(\boldsymbol{z}_{<k}|\boldsymbol{y}) &= q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}) && \text{(B.24)}\\
&= p_\theta(\boldsymbol{z}|\boldsymbol{y}), && \text{(B.25)}
\end{aligned}$$

for all $\boldsymbol{y}$ in the support of $p_{\mathrm{data}}(\boldsymbol{y})$. □

Proposition B.4 shows that, if the VAE encoder and decoder and the low-resolution encoder have enough capacity and are trained well enough to optimize their respective training criterion, the low-resolution encoder matches the intractable conditional $p_\theta(\boldsymbol{z}_{<k}|\boldsymbol{y})$ of the VAE model distribution (B.17).

B.2.2 Expected consistency of the super-resolution model

In the next proposition, we establish a general formula to estimate of the expected consistency error.

Proposition B.5. *Under assumptions 6.1 and 6.2, we have:*

$$CE(k) = \mathbb{E}_{p_{data}(\boldsymbol{x})}\mathbb{E}_{q_\phi(\boldsymbol{z}_{<k}|\boldsymbol{x})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\tilde{\boldsymbol{x}} - H_s\boldsymbol{x}\|_2\right]. \tag{B.26}$$

Proof. Assumptions 6.1 and 6.2 imply that $r(\boldsymbol{z}_{<k}|\boldsymbol{y}) = q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y})$ for all $\boldsymbol{y}$ in the support of $p_{\text{data}}(\boldsymbol{y})$. Notice that, by definition of $r(\boldsymbol{z}_{<k}, \boldsymbol{x}, \boldsymbol{y})$ (6.14), the marginals satisfy $r(\boldsymbol{y}) = p_{\text{data}}(\boldsymbol{y})$ and $r(\boldsymbol{x}) = p_{\text{data}}(\boldsymbol{x})$. Consequently:

$$\begin{aligned}
CE(k) &= \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})}\mathbb{E}_{r(\boldsymbol{z}_{<k}|\boldsymbol{y})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] && \text{(B.27)}\\
&= \mathbb{E}_{r(\boldsymbol{z}_{<k},\boldsymbol{y})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] && \text{(B.28)}\\
&= \mathbb{E}_{r(\tilde{\boldsymbol{x}})}\mathbb{E}_{r(\boldsymbol{z}_{<k},\boldsymbol{y}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] && \text{(B.29)}\\
&= \mathbb{E}_{p_{\text{data}}(\tilde{\boldsymbol{x}})}\mathbb{E}_{p_{\text{data}}(\boldsymbol{y}|\tilde{\boldsymbol{x}})}\mathbb{E}_{q_\psi(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right]. && \text{(B.30)}
\end{aligned}$$

□

According to Proposition B.3, the assumption $q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}) = r(\boldsymbol{z}_{<k}|\boldsymbol{y})$ is satisfied when the low-resolution encoder has enough capacity and is trained to optimality. The quantity (B.26) can be estimated with Monte-Carlo sampling, without using the low-resolution encoder. Thus, Proposition B.5 gives us a way to estimate the potential consistency error of a super-resolution model before training the low-resolution encoder.

We can now show Proposition 6.2, which states that under the additional hypothesis 6.3 on the VAE inference and generative model, the expected consistency error of the super-resolution model can be estimated only as a function of the VAE generative model $p_\theta(\boldsymbol{z}|\boldsymbol{x})$.

Proposition B.6 (6.2)**.** *Under assumptions 6.1, 6.2 and 6.3, the expected consistency error is:*

$$CE(k) = \mathbb{E}_{p_\theta(\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\boldsymbol{x}|\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\tilde{\boldsymbol{x}} - H_s\boldsymbol{x}\|_2\right]. \tag{6.16}$$

Proof. First, assumption 6.3 implies that the marginals of the model distribution (B.17) match the respective data distributions:

$$p_\theta(\boldsymbol{x}) = p_{\text{data}}(\boldsymbol{x}) \tag{B.31}$$
$$p_\theta(\boldsymbol{y}) = p_{\text{data}}(\boldsymbol{y}) \tag{B.32}$$
$$p_\theta(\boldsymbol{x}|\boldsymbol{y}) = p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y}). \tag{B.33}$$

Second, from assumptions 6.1 and 6.2, proposition B.4 can be applied, and we have:

$$q_\psi(\boldsymbol{z}_{<k}|\boldsymbol{y}) = p_\theta(\boldsymbol{z}_{<k}|\boldsymbol{y}) \tag{B.34}$$
$$= \mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{y})}[p_\theta(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}}, \boldsymbol{y})] \tag{B.35}$$
$$= \mathbb{E}_{p_{\text{data}}(\tilde{\boldsymbol{x}}|\boldsymbol{y})}[p_\theta(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}})]. \tag{B.36}$$

Thus we get:

$$CE(k) \tag{B.37}$$
$$= \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})}\mathbb{E}_{p_\theta(\boldsymbol{z}_{<k}|\boldsymbol{y})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] \tag{B.38}$$
$$= \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})}\mathbb{E}_{p_{\text{data}}(\tilde{\boldsymbol{x}}|\boldsymbol{y})}\mathbb{E}_{p_\theta(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] \tag{B.39}$$
$$= \mathbb{E}_{p_{\text{data}}(\tilde{\boldsymbol{x}})}\mathbb{E}_{p_{\text{data}}(\boldsymbol{y}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] \tag{B.40}$$
$$= \mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(\boldsymbol{z}_{<k}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_{\text{data}}(\boldsymbol{y}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right] \tag{B.41}$$
$$= \mathbb{E}_{p_\theta(\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\mathbb{E}_{p_{\text{data}}(\boldsymbol{y}|\tilde{\boldsymbol{x}})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\boldsymbol{x} - \boldsymbol{y}\|_2\right], \tag{B.42}$$

where (B.39) comes from relation (B.36), and (B.41) comes from relation (B.32). Therefore, using the fact that $p_{\text{data}}(\boldsymbol{y}|\tilde{\boldsymbol{x}}) = \delta_{\{\boldsymbol{y}=H_s\tilde{\boldsymbol{x}}\}}$, we obtain:

$$CE(k) = \mathbb{E}_{p_\theta(\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(\tilde{\boldsymbol{x}}|\boldsymbol{z}_{<k})}\mathbb{E}_{p_\theta(x|\boldsymbol{z}_{<k})}\left[\frac{1}{\sqrt{m}}\|H_s\tilde{\boldsymbol{x}} - H_s\boldsymbol{x}\|_2\right], \tag{B.43}$$

□

Proposition 6.2 show that, under adequate assumptions, the expected consistency of the super-resolution model (6.16) only depends on the generative model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$. It follows that it can be estimated before the training of the low-resolution encoder. Notice that the quantity (6.16) is equal to the low-resolution consistency U_k^s (6.5).

Figure B.1 – Super-resolved samples at $\tau = 0.8$ for upsampling factor $\times 4$.

Additional samples

For visualization purposes, we provide additional super-resolved samples produced with our method in Figures B.1, B.2 and B.3.

Figure B.2 – Super-resolved samples at $\tau = 0.8$ for upsampling factor $\times 8$.

Figure B.3 – Super-resolved samples at $\tau = 0.8$ for upsampling factor $\times 16$.