ABSTRACT

Title of Dissertation: UNCERTAINTY IN DIRECTIONAL REPRESENTATIONS, PREIMAGES OF KERNEL TRANSFORMATIONS AND APPLICATIONS

Canran Ji
Doctor of Philosophy, 2025

Dissertation Directed by: Professor Wojciech Czaja
Department of Mathematics

This dissertation develops theoretical and computational advances in discrete directional time–frequency analysis, phase retrieval, kernel-based inverse problems, and applications to electron microscopy denoising.

Chapter 3 presents sharp uncertainty inequalities for the Directional Gabor Ridge Transform (DGRT) and its weighted variant (DWGRT) within a discrete-frame framework, yielding explicit bounds on spatial support and directional frequency localization. These results can extend classical continuous uncertainty principles to fully discrete directional frames and provide explicit guidance on window lengths, orientation sampling, and weight functions.

Chapter 4 formulates undersampled short-time Fourier magnitude inversion as a supervised learning problem. I design a compact neural network trained with adversarial and reconstruction losses that reconstructs eight-thousand-length-sample audio segments from four-thousand magnitude measurements. Extensive experiments demonstrate rapid convergence and superior numerical and perceptual quality compared to Griffin–Lim, including downstream classification accuracy improvements.

In Chapter 5, I implement and compare three deterministic kPCA pre-image algorithms: fixed-point iteration, kernel ridge regression, and Sapiro's Nyström extension method, applying them across MNIST, CIFAR-10, and SVHN under noise-free and noisy scenarios. Metrics of PSNR, SSIM, and PCC identify each solver's strengths and limitations. Motivated by these findings, I introduce DCGAN-KPCAnet and WGAN-KPCAnet, two generative adversarial inverse solvers that learn the kPCA mapping directly. WGAN-KPCAnet consistently exceeds the best deterministic solver in reconstruction fidelity, structural preservation, and noise robustness.

Chapter 6 integrates cosine-similarity kPCA denoising into graphene-liquid-cell and single particle cryo-EM pipelines. By enhancing the previous denoising (masking and averaging) with kPCA inversion, I achieve substantial improvements in two-dimensional projection quality and enable high-resolution three-dimensional reconstructions that reveal dynamic structural states. Kernel parameter selection, computational scalability, and software integration are discussed.

Together, these contributions establish new discrete uncertainty bounds, demonstrate the efficacy of learning-based phase retrieval, advance kernel pre-image algorithms through generative modeling, and apply kernel PCA denoising to challenging electron microscopy data, bridging fundamental mathematics with practical signal processing applications.

UNCERTAINTY IN DIRECTIONAL REPRESENTATIONS, PREIMAGES OF KERNEL TRANSFORMATIONS AND APPLICATIONS

by

Canran Ji

Dissertation submitted to the Faculty of the Graduate School
of the University of Maryland, College Park, in partial
fulfillment of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Professor Wojciech Czaja, Chair/Advisor
Dr. Hans Elmlund, Special Member
Professor William Gasarch, Dean's Representative
Professor Vincent Lyzinski
Professor Deep Ray

Dedication

For Professor James Walsh, whose enthusiasm led me to the world of mathematics.
For my parents, whose support made this work possible.

Acknowledgments

I am deeply grateful to my advisor, Dr. Wojciech Czaja, for his unwavering support, insightful discussions, and invaluable mentorship throughout the course of this research. His guidance has been instrumental in shaping the ideas and results presented in this dissertation.

I would like to express my sincere appreciation to Dr. Hans Elmlund from the National Cancer Institute (NCI) of NIH for his funding support and research guidance. I also thank his research group for their contributions and assistance in electron microscopy image denoising. I would like to express my sincere appreciation to Dr. Hans Elmlund from the National Cancer Institute (NCI) of NIH for his funding support and research guidance. I also thank members of his research group at NCI for their contributions to various aspects of this work. In particular, I am especially grateful to Dr. Cong Tuan Son Van for his generous assistance, technical guidance, and close collaboration throughout my research on the electron microscopy image denoising.

I would also like to thank Dr. John Benedetto, professor emeritus and research at the University of Maryland, whose early encouragement and teaching first inspired my interest in harmonic analysis and opened the path to many of the mathematical directions explored in this dissertation.

I also extend my sincere appreciation to my dissertation committee members, Dr. Gasarch, Dr. Lyzinski and Dr. Ray, for their valuable feedback and constructive suggestions. Their expertise has greatly contributed to the development of this work.

Furthermore, I would like to thank my colleagues and collaborators for their stimulating discussions and contributions to various aspects of this research. Their insights have been crucial in refining key ideas and methodologies.

Finally, I am profoundly grateful to my family and friends for their constant encouragement and support. Their patience and understanding have made this journey possible.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Meaning
AF	Amplitude Flow
BCE	Binary Cross-Entropy
Cryo-EM	Cryogenic Electron Microscopy
CT	Computed Tomography
CTF	Contrast Transfer Function
DGRT	Directional Gabor Ridge Transform
DSTFT	Directional Short-Time Fourier Transform
DWGRT	Directional Weighted Gabor Ridge Transform
EM	Electron Microscopy
FBP	Filtered Back-Projection
FGL	Fast Griffin–Lim
GAN	Generative Adversarial Network
GAN-KPCAnet	Generative Adversarial Network for inverse kPCA
GL	Griffin–Lim Algorithm
GLC-EM	Graphene-Liquid-Cell Electron Microscopy
ISTFT	Inverse Short-Time Fourier Transform
kPCA	Kernel Principal Component Analysis
MDS	Multidimensional Scaling
MSE	Mean Squared Error
NOLA	Nonzero Overlap-Add (condition)
NNIFGL	Neural Network Initialized Fast Griffin–Lim
PCA	Principal Component Analysis
PC	Principal Component
PCC	Pearson Correlation Coefficient
PNN	Postprocessed Neural Network
PSNR	Peak Signal-to-Noise Ratio

Abbreviation	**Meaning**
RBF	Radial Basis Function (Gaussian kernel)
RKHS	Reproducing Kernel Hilbert Space
SC	Spectral Convergence
SNR	Signal-to-Noise Ratio
SSIM	Structural Similarity Index Measure
STFT	Short-Time Fourier Transform
TV	Total Variation
WF	Wirtinger Flow
WGAN	Wasserstein Generative Adversarial Network
WGAN-KPCAnet	Wasserstein GAN for inverse kPCA

List of Notations

Symbol	Meaning
α	Significance level of the statistical test
$B^{-1}(\cdot\,; a, b)$	Quantile function of Beta(a, b) distribution
$Bf(t)$	Back-projection via R^*R
$C(t)$	Crystallinity score at time t
$\mathbf{c}_k$	Estimated particle center in averaged frame k
$\tilde{d}_i$	Distance to $\phi(x_i)$ in RKHS
$D_\alpha(h)$	Fractional differential operator
D_ϕ	Discriminator (or critic) network parameterized by ϕ
$\mathcal{D}$	Set of 1-Lipschitz functions (WGAN)
Δ	Parameter space $S^{n-1} \times \mathbb{R} \times \mathbb{R}$
Δ^*	Parameter space $S^{n-1} \times \mathbb{R} \times \mathbb{R}^n$
$\boldsymbol{\delta}$	Dirac delta distribution
δ	Measure on Δ
δ^*	Measure on Δ^*
$DS_g f(\xi, x, \omega)$	Directional STFT
$\varepsilon_i(t)$	Radial strain of atom i
$\varepsilon_{\mathrm{rad}}(t)$	Mean radial strain at time t
$F_{\mathrm{core}}(t)$	Fraction of core atoms at time t
$\hat{f}$	Fourier transform of a function f
ϕ	Feature map in kernel methods
γ^*	Coupling in Wasserstein distance
Γ	Gamma Function
Γ'	Learned inverse map
$\tilde{\gamma}_i$	Weight in Rathi's method
G_θ	Generator network parameterized by θ
$G^{x,\omega}(s)$	Weighted function in DGRT

Symbol	Meaning
$G_{\xi,x,\omega}(t)$	Directional weighted Gabor ridge function
$G_g f(m,n)$	Gabor transform of f
H_0	Null hypothesis
H_1	Alternative hypothesis
$\mathcal{H}$	RKHS induced by kernel k
$\widetilde{\mathbf{I}}_k$	TV-denoised, motion-corrected image
$k(x,y)$	Mercer kernel function
K	Kernel Gram matrix
K_c	Centered Gram matrix
$k_c(x)$	Centered kernel vector
$\hat{k}_x$	Approximated feature vector from ψ
$\ell_{\mathrm{PR}}(\mathbf{x})$	Phase retrieval loss
$\mathcal{L}_D$	Discriminator loss function
$\mathcal{L}_D^{\mathrm{W}}$	WGAN discriminator loss
$\mathcal{L}_G$	Generator loss function
$\mathcal{L}_G^{\mathrm{W}}$	WGAN generator loss
L_{GP}	Gradient penalty loss term
L_{PR}	STFT-based phase retrieval loss
$\widetilde{\Lambda}$	Diagonal eigenvalue matrix of K_c
λ_*, λ_{gp}, λ_{recon}	Trade-off parameters in loss functions
$\mathcal{M}(\mathbf{x})$	STFT magnitude operator
$\mathcal{M}_{\mathbf{w}'}^{a',M'}$	STFT magnitude in loss
μ^*	Regularization parameter for TV denoising
n_{critic}	Number of discriminator updates per generator update
n_{samples}	Number of training samples used
$N_{\mathrm{core}}(t)$	Number of core atoms at time t
p-value	Tail probability under null hypothesis
p_{data}	Empirical distribution of clean training data
p_z	Distribution of kPCA latent codes
ψ	Target in RKHS
$\mathbf{R}(t)$	Nanocrystal center of mass at time t
$R^*g(t)$	Adjoint Radon transform
$Rf(\xi,s)$	Radon transform of f
$r_i(t)$	Radial distance of atom i at time t

Symbol	Meaning
$\mathbf{r}_i(t)$	Position of atom i at time t
$\mathbf{r}_i^{\text{ref}}$	Ideal lattice position of atom i
$\mathcal{S}(t)$	Set of solvent-access voxels at time t
σ	Frequency in Fourier slice theorem
$\Sigma \in \mathbb{R}^{d\times d}$	Covariance matrix in PCA
$\mathrm{TV}(\cdot)$	Total variation functional
U	Eigenvector matrix of K_c
$\mathcal{V}_{\mathbf{w}}^{a,M}\mathbf{x}(m,n)$	Discrete STFT
$V_g f(t,\omega)$	STFT of f with window g
w_t	Weight for frame t in time-window averaging
$\mathcal{W}_k$	Time window of W frames
x^*	Pre-image in input space
x_{obs}	Observed number of wins
z_i	Latent kPCA representation of noisy input $\tilde{x}_i$

Chapter 1

Introduction

1.1 Motivation and Scope

Discrete directional time–frequency analysis plays a central role in applications that require orientation-sensitive feature extraction. The directional Gabor ridge transform augments the classical short-time Fourier transform by projecting signals onto ridge functions aligned with specified angles, thereby isolating frequency content along those orientations. In its weighted variant, additional windowing or weight functions emphasize or de-emphasize particular directions. Continuous-domain uncertainty principles—dating back to Heisenberg's original inequality—quantify the trade-off between time (or space) localization and frequency concentration, and semi-discrete extensions address mixed continuous–discrete scenarios. However, practical implementations rely on fully discrete frames whose fundamental limits have remained uncharacterized. Without rigorous uncertainty inequalities for discrete directional transforms, practitioners lack the theoretical bounds needed to choose window lengths, sampling rates, orientation grids and weight functions that balance spatial resolution against directional frequency resolution.

Phase retrieval constitutes a second, distinct inverse-problem domain with broad implications in optics, coherent diffractive imaging and audio processing. In many measurement systems, only the magnitudes of short-time Fourier coefficients are recorded, with phase information lost. Classical Griffin–Lim iterations alternate between enforcing magnitude constraints and synthesizing time-domain estimates, but they often stagnate or converge slowly under aggressive undersampling or nonideal measurement operators. Moreover, they offer no guarantee of global optimality and can be sensitive to initialization. Recent advances in deep learning suggest that neural networks can learn structural priors from data and enable accurate inversion even when sampling is well below classical thresholds.

The pre-image problem for Kernel Principal Component Analysis (kPCA) presents a

third challenge. kPCA embeds input vectors into a high-dimensional feature space via a Mercer kernel and performs linear PCA to denoise and reduce dimensionality. Recovering an approximation of the original input—the pre-image—requires solving a nonlinear inverse problem. Deterministic solvers have been proposed, including fixed-point iterations, kernel ridge regression, multidimensional scaling-based inversion, direct-approximation formulas and Nyström-based hybrids. Each method offers different trade-offs among reconstruction accuracy, computational cost and noise robustness, yet their relative performance across datasets and noise regimes has not been systematically assessed.

Electron microscopy provides the fourth application domain, where kernel methods can deliver transformative benefits under extreme noise. In graphene-liquid-cell (GLC-EM) and single-particle cryogenic (cryo-EM) workflows, low electron dose preserves sample integrity but yields micrographs with signal-to-noise ratios often below 0.1. Conventional denoising—frame averaging, masking or linear filtering—improves visibility but sacrifices spatial resolution or temporal detail. Embedding micrographs into a kernel-defined feature space via cosine similarity kPCA and reconstructing denoised pre-images promises to suppress noise while preserving fine structural details, provided one can invert the embedding robustly under severe noise.

1.2 Contributions and Organization of the Dissertation

In Chapter 3, building on existing discrete-frame constructions of DGRT and DWGRT, I establish new integral inequalities that quantify the trade-off between spatial concentration and directional frequency resolution, and derive parameter guidelines for window lengths, orientation sampling, and weight functions. These results extend classical uncertainty principles to directional transforms and inform parameter choices for window lengths, orientation sampling, and weighting functions.

Chapter 4 addresses phase retrieval for speech signals. I formulate the inversion of undersampled short-time Fourier magnitude measurements as a supervised learning problem, design a compact neural network trained with adversarial and reconstruction losses, and demonstrate through extensive experiments that eight-thousand-length-sample audio segments can be reconstructed from four-thousand magnitude coefficients. The learned model converges rapidly and outperforms the Griffin–Lim algorithm in both numerical metrics and perceptual quality, as validated by a downstream classification task.

In Chapter 5 I implement and compare three representative deterministic kPCA pre-image

algorithms—fixed-point iteration, kernel ridge regression and Schölkopf's method—on MNIST, CIFAR-10 and SVHN under noise-free and noisy conditions. By measuring peak signal-to-noise ratio, structural similarity index and Pearson correlation, I identify the contexts in which each solver excels or falters. Motivated by these findings, I introduce DCGAN-KPCAnet and WGAN-KPCAnet, two generative adversarial network–based inverse solvers that learn the kPCA mapping directly from data. Extensive evaluation shows that WGAN-KPCAnet consistently surpasses the best deterministic solver in reconstruction fidelity, structural preservation and robustness to noise.

Chapter 6 integrates cosine similarity kPCA denoising into electron microscopy workflows. By replacing conventional masking and averaging steps in both GLC-EM and cryo-EM pipelines with kPCA inversion, I achieve substantial improvements in two-dimensional projection quality and enable high-resolution three-dimensional reconstructions that reveal dynamic structural states previously obscured by noise. I discuss kernel parameter selection, computational scalability and integration with existing EM software.

Each chapter stands with full theoretical derivations or model descriptions, implementation details and comprehensive experimental evaluations. The final chapter synthesizes the results, reflects on their broader implications for nonlinear inverse problems and outlines directions for future research in kernel-based and learning-driven signal processing.

Chapter 2

Preliminaries

2.1 Time-Frequency Representations and Uncertainty Principles

Time-frequency analysis provides a framework for studying signals that evolve over time by examining how their frequency content changes. Traditional Fourier analysis is a powerful tool for decomposing functions into sinusoidal components, but it lacks localization in the time domain, making it unsuitable for analyzing non-stationary signals. To address this limitation, time-frequency representations such as the Short-Time Fourier Transform (STFT) and Gabor analysis have been developed to capture both temporal and spectral characteristics of signals.

A fundamental concept in time-frequency analysis is the uncertainty principle, which establishes a trade-off between time and frequency localization. This principle highlights the inherent limitation that a signal cannot be arbitrarily well-localized in both time and frequency simultaneously. The choice of window functions and the structure of time-frequency frames play a crucial role in balancing this trade-off.

Moreover, classical time-frequency representations do not inherently incorporate directional information. The integration of the Radon transform with Gabor analysis allows for directional sensitivity, enhancing the ability to analyze anisotropic signals such as images and biomedical data. This section explores the mathematical foundations of time-frequency analysis, beginning with the Fourier transform and its extension to the STFT, followed by a discussion of the uncertainty principle and its implications for Gabor frames.

2.1.1 Short-Time Fourier Transform (STFT)

The Fourier transform is a fundamental tool in signal processing and time-frequency analysis. Given an integrable function f on $\mathbb{R}^n$, its Fourier transform is defined as

$$\hat{f}(\xi) = \int_{\mathbb{R}^n} f(x) e^{-2\pi i x \cdot \xi} dx. \tag{2.1}$$

This transformation allows for the analysis of the frequency content of a signal by decomposing it into sinusoidal components. However, the Fourier transform has a limitation: it provides a global frequency representation of the signal, meaning that it does not retain any localization in the time domain. This loss of time resolution makes it difficult to analyze non-stationary signals whose frequency content changes over time.

To address this limitation, the Short-Time Fourier Transform (STFT) was introduced. The STFT introduces a window function g to localize the signal in time before applying the Fourier transform. The STFT of a function f with respect to a window function g is given by

$$V_g f(t, \omega) = \int_{\mathbb{R}^n} f(x) g(x - t) e^{-2\pi i \omega \cdot x} dx, \tag{2.2}$$

where g is a well-localized function, typically a Gaussian, that acts as a window to extract local frequency content around t. The STFT provides a two-dimensional representation of the signal, capturing both time and frequency information.

A key property of the STFT is that it satisfies an inversion formula, allowing the reconstruction of the original function:

$$f(x) = \frac{1}{\langle \psi, g \rangle} \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} V_g f(t, \omega) \psi_{\omega, t} d\omega dt, \tag{2.3}$$

where ψ is another well-chosen window function. The STFT is widely used in signal processing, speech analysis, and time-frequency representations of images.

2.1.2 Radon Transform and Fourier Slice Theorem

The Radon transform is a fundamental mathematical tool in tomography, harmonic analysis, and time-frequency representations. It provides a way to analyze functions by integrating them over hyperplanes. This technique is widely used in medical imaging, particularly in computed tomography (CT) and image reconstruction problems.

Definition of the Radon Transform

Let f be a function defined on $\mathbb{R}^n$. The Radon transform Rf of f is given by:

$$Rf(\xi, s) = \int_{\xi \cdot t = s} f(t)dt, \quad \xi \in S^{n-1}, \quad s \in \mathbb{R}, \tag{2.4}$$

where S^{n-1} is the unit sphere in $\mathbb{R}^n$ and the integral is taken over the hyperplane $\xi \cdot t = s$. Here, ξ represents the normal direction of the hyperplane, and s determines its position.

The Radon transform maps a function f to a function Rf defined on the space of hyperplanes in $\mathbb{R}^n$. This transformation provides an alternative representation of f, where information about the function is distributed across different orientations and positions.

The Dual of the Radon Transform

The dual operator R^* of the Radon transform is defined as:

$$R^*g(t) = \int_{S^{n-1}} g(\xi, \xi \cdot t)d\xi, \tag{2.5}$$

where $g(\xi, s)$ is a function defined on $S^{n-1} \times \mathbb{R}$. This operator reconstructs an approximation of the original function by integrating over all directions.

The operator R^* serves as an adjoint to R in the sense that, when applied to the Radon transform of a function f, it results in the back-projection formula.

Back-Projection Formula

One of the fundamental results associated with the Radon transform is the back-projection formula:

$$Bf(t) = R^*(Rf)(t) = \int_{S^{n-1}} Rf(\xi, \xi \cdot t)d\xi. \tag{2.6}$$

This formula provides an estimate of f by averaging its Radon transform over all possible directions. However, the back-projection alone does not perfectly recover f; additional filtering is required to obtain a precise reconstruction.

Fourier Slice Theorem

A key result linking the Radon transform and Fourier analysis is the Fourier slice theorem, also known as the projection-slice theorem. It states that the Fourier transform of f along a given direction ξ is equal to the one-dimensional Fourier transform of its Radon transform along the same direction:

$$\hat{R}f(\xi, \sigma) = \hat{f}(\sigma\xi), \quad \sigma \in \mathbb{R}, \quad \xi \in S^{n-1}. \tag{2.7}$$

This theorem plays a crucial role in tomographic reconstruction. It implies that the information contained in the Radon transform of f is sufficient to recover f by applying the inverse Fourier transform.

Implications and Computational Aspects

The Fourier slice theorem provides the foundation for many reconstruction algorithms, including the filtered back-projection (FBP) algorithm used in CT imaging. Since the Fourier transform is computationally efficient, this theorem enables fast and accurate reconstruction of images from projection data.

The reconstruction process typically involves the following steps: 1. Compute the one-dimensional Fourier transform of $Rf(\xi, s)$ with respect to s. 2. Extend the transformed data to higher dimensions using the Fourier slice theorem. 3. Apply the inverse Fourier transform to obtain f.

2.1.3 Directional Short-Time Fourier Transform

The short-time Fourier transform (STFT) is a fundamental tool in time-frequency analysis, providing a localized frequency representation of signals. However, the STFT lacks intrinsic directional sensitivity, making it less effective for analyzing signals with significant anisotropic structures, such as images or multidimensional signals where orientation plays a crucial role. To address this limitation, the Directional Short-Time Fourier Transform (DSTFT) was introduced in [17], extending the classical STFT by incorporating a directional parameter, which allows for frequency analysis along specific orientations.

The DSTFT differs from the STFT in that it maps a function f defined on $\mathbb{R}^n$ to a function on the extended parameter space $S^{n-1} \times \mathbb{R} \times \mathbb{R}^n$. The additional parameter $\xi \in S^{n-1}$ enables the transform to capture frequency components along particular directions, unlike the STFT, which applies the same window function isotropically across all directions.

Definition of the Directional Short-Time Fourier Transform

For a function $f \in L^1(\mathbb{R}^n)$, the Directional Short-Time Fourier Transform (DSTFT) with respect to a window function $g \in L^\infty(\mathbb{R})$ is given by:

$$\mathcal{D}_g f(\xi, x, \omega) = \int_{\mathbb{R}^n} f(t) g(\xi \cdot t - x) e^{-2\pi i t \cdot \omega} dt, \tag{2.8}$$

- $\xi \in S^{n-1}$ is the directional parameter, which aligns the window function along the chosen direction.
- $x \in \mathbb{R}$ is the shift parameter, representing translation along the direction ξ.
- $\omega \in \mathbb{R}^n$ is the frequency variable.
- $g(\xi \cdot t - x)$ is a window function localized along the direction ξ, ensuring that the transform captures frequency information along specific orientations.

This directional modification allows the DSTFT to analyze signals by decomposing them into one-dimensional Gabor-like functions along specific orientations. The function $g_{\xi,x}(t) = g(\xi \cdot t - x)$ behaves as a one-dimensional windowed function along the direction ξ while being constant in its orthogonal complement.

Connection to the Radon Transform

The DSTFT is closely related to the Radon transform, which integrates a function over hyperplanes parameterized by (ξ, s). The Radon transform is defined as:

$$Rf(\xi, s) = \int_{\xi \cdot t = s} f(t) dt. \tag{2.9}$$

The relationship between the DSTFT and the Radon transform can be expressed as:

$$\mathcal{D}_g f(\xi, x, \omega) = \hat{f} g_{\xi,x}(\omega) = \langle R_\xi(M_{-\omega} f), T_x g \rangle. \tag{2.10}$$

This expression reveals that the DSTFT is a modulated Radon transform with a windowed function applied before frequency analysis. This connection is crucial for understanding how directional frequency content is preserved in the DSTFT framework.

2.1.4 Preliminary Inequalities

We begin by presenting two fundamental inequalities that will be used in our derivations.

Theorem 2.1 (Hausdorff-Young Inequality)**.** *Let $1 \leqslant q \leqslant 2$, and let p be the Hölder conjugate of q, i.e., $\frac{1}{p} + \frac{1}{q} = 1$. Then, the Fourier transform acts as a bounded operator from $L^q(\mathbb{R}^n)$ to $L^p(\mathbb{R}^n)$, satisfying the norm estimate*

$$\|\hat{f}\|_p \leqslant \|f\|_q, \quad \forall f \in L^q(\mathbb{R}^n).$$

The Hausdorff-Young inequality plays a crucial role in understanding the decay and regularity properties of functions in the frequency domain, providing an essential tool for analyzing directional transforms.

Theorem 2.2 (Minkowski's Integral Inequality)**.** *Let $(X, \mathcal{M}, \mu)$ and $(Y, \mathcal{N}, \nu)$ be σ-finite measure spaces. Suppose that $f : X \times Y \to \mathbb{R}^+$ is a measurable function. Then, for every $1 \leqslant p < \infty$, we have*

$$\left[\int_X \left(\int_Y f(x,y)d\nu(y)\right)^p d\mu(x)\right]^{1/p} \leqslant \int_Y \left[\int_X f(x,y)^p d\mu(x)\right]^{1/p} d\nu(y).$$

This integral inequality is particularly useful for bounding integral operators arising in time-frequency analysis and will be applied to obtain norm estimates for the DGRT and DWGRT.

Orthogonality Relation

An orthogonality relation for the DSTFT is established similarly to the STFT. Suppose $g_1, g_2 \in L^\infty(\mathbb{R})$ and $f_1, f_2 \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, and assume at least one of the window functions g_i belongs to $L^1(\mathbb{R})$. Then,

$$\int_{S^{n-1}} \int_{\mathbb{R}} \int_{\mathbb{R}^n} \mathcal{D}_{g_1} f_1(\xi, x, \omega) \overline{\mathcal{D}_{g_2} f_2(\xi, x, \omega)}\, d\omega\, dx\, d\xi = \langle f_1, f_2 \rangle \langle g_2, g_1 \rangle. \tag{2.11}$$

This orthogonality condition is a key property for ensuring stability and reconstruction.

Hausdorff–Young Inequality for the DSTFT

The classical Hausdorff–Young inequality states that if $f \in L^p(\mathbb{R}^n)$, then its Fourier transform satisfies:

$$\|\hat{f}\|_{L^q} \leqslant \|f\|_{L^p}, \quad \text{where } \frac{1}{p} + \frac{1}{q} = 1, \quad 1 \leqslant p \leqslant 2. \tag{2.12}$$

For the DSTFT, a similar norm inequality can be established. If $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $g \in L^1(\mathbb{R}) \cap L^\infty(\mathbb{R})$, then the DSTFT satisfies:

$$\|\mathcal{D}_g f\|_{L^p(S^{n-1} \times \mathbb{R} \times \mathbb{R}^n)} \leqslant \|g\|_{L^p(\mathbb{R})} \|f\|_{L^q(\mathbb{R}^n)}, \tag{2.13}$$

where $1 \leqslant q \leqslant 2$ and $\frac{1}{p} + \frac{1}{q} = 1$. This inequality ensures that the DSTFT maps $L^q(\mathbb{R}^n)$ into $L^p(S^{n-1} \times \mathbb{R} \times \mathbb{R}^n)$, establishing a boundedness relation analogous to the classical Hausdorff–Young theorem.

Reconstruction and Inversion Formula

A weak-sense inversion formula for the DSTFT states that if g satisfies $\langle g_2, g_1 \rangle \neq 0$, then:

$$f(t) = \frac{1}{\langle g_2, g_1 \rangle} \int_{S^{n-1}} \int_{\mathbb{R}} \int_{\mathbb{R}^n} \mathcal{D}_{g_1} f(\xi, x, \omega) g_{2,\xi,x,\omega}(t) \, d\omega \, dx \, d\xi. \tag{2.14}$$

This formula guarantees that f can be recovered from its DSTFT representation under suitable conditions.

The primary difference between DSTFT and STFT lies in the incorporation of the directional parameter ξ. The STFT applies the same window function uniformly across all directions, making it isotropic in nature, whereas the DSTFT aligns the window function along a specific orientation, allowing it to capture directionally dependent frequency components. This additional directionality makes the DSTFT particularly effective in analyzing signals with prominent directional structures, such as oriented textures in images, edge-like structures, and ridges in data. The DSTFT provides a more refined representation by taking into account both spatial and directional variations, making it a powerful extension of the traditional STFT.

2.1.5 Directional Gabor Ridge Transform and its Weighted Extension

A fundamental concept in signal analysis is that of a frame, which provides a redundant and stable representation of signals in a Hilbert space. A sequence $\{\psi_j\}_{j \in J}$ in a Hilbert space $\mathcal{H}$ is called a frame if there exist constants $A, B > 0$ such that for all $f \in \mathcal{H}$,

$$A\|f\|^2 \leqslant \sum_{j \in J} |\langle f, \psi_j \rangle|^2 \leqslant B\|f\|^2. \tag{2.15}$$

A Gabor frame is a special type of frame constructed using translations and modulations of a single window function g. Given parameters $a, b > 0$, a Gabor system consists of the functions

$$g_{m,n}(x) = e^{2\pi i n b x} g(x - ma),$$

where $m, n \in \mathbb{Z}$. The set $\{g_{m,n}\}$ forms a Gabor frame if it satisfies the frame condition. The Gabor transform of a function f with respect to a window g is given by

$$G_g f(m, n) = \int_{\mathbb{R}^n} f(x) g(x - ma) e^{-2\pi i n b x} dx. \tag{2.16}$$

While Gabor analysis provides an effective representation of signals, it does not inherently capture directional features. The Directional Gabor Ridge Transform (DGRT), introduced in [20], incorporates directional sensitivity into the Gabor framework by projecting onto different orientations before computing the Gabor transform. This is motivated by the need for translation-invariant and rotation-aware feature extraction in image processing applications, as discussed in [29].

Directional Gabor Ridge Transform

Let $g \in \mathcal{S}(\mathbb{R})$, the space of Schwartz functions, be a real-valued non-zero window function. A Gabor element on $\mathbb{R}$ associated with g is defined as

$$g^{x,\omega}(s) = e^{2\pi i\omega(s-x)}g(s-x), \qquad \text{for } s, x, \omega \in \mathbb{R}.$$

Then the Gabor ridge function is:

$$g_{\xi,x,\omega}(t) = g^{x,\omega}(\xi \cdot t), \qquad \text{for } t \in \mathbb{R}^n,$$

where $\xi \in S^{n-1}$ and $x, \omega \in \mathbb{R}$.

Let $\Delta = S^{n-1} \times \mathbb{R} \times \mathbb{R}$ with the associated Haar measure $d\delta(\xi, x, \omega) = d\xi \otimes dx \otimes d\omega$

Using these functions, the Directional Gabor Ridge Transform (DGRT) of a function f on $\mathbb{R}^n$ is given by

$$\mathfrak{D}_g(f)(\xi, x, \omega) = \int_{\mathbb{R}^n} f(t)\overline{g(\xi \cdot t - x)}e^{-2\pi i\omega(\xi \cdot t - x)}dt. \tag{2.17}$$

This transform maps a function f to a function on $S^{d-1} \times \mathbb{R} \times \mathbb{R}$, encoding both directional and frequency information. It can also be rewritten in inner product notation as

$$\mathfrak{D}_g(f)(\xi, x, \omega) = \langle f, g_{\xi,x,\omega} \rangle. \tag{2.18}$$

Differential Operator

The differential operator D_α, acting on functions on the real line for $\alpha > 0$, is defined as

$$D_\alpha(h) = \left(\widehat{h}(\xi)|\xi|^\alpha\right)^\vee. \tag{2.19}$$

This operator is useful in defining weighted functions used in the analysis of directional transforms.

Directional Weighted Gabor Ridge Transform

Let $g \in S(\mathbb{R})$ be a nonzero real-valued window function. For $\omega, x \in \mathbb{R}$, we define the weighted function

$$G^{x,\omega}(s) = D_{\frac{d-1}{2}}(g^{x,\omega})(s) = \left(\widehat{g^{x,\omega}}(\sigma)|\sigma|^{\frac{d-1}{2}}\right)^{\vee}(s). \tag{2.20}$$

For $\xi \in S^{d-1}$, we then define the weighted Gabor ridge functions as

$$G_{\xi,x,\omega}(t) = G^{x,\omega}(\xi \cdot t), \quad t \in \mathbb{R}^d. \tag{2.21}$$

The Directional Weighted Gabor Ridge Transform (DWGRT) improves upon the DGRT by incorporating a weight function that adjusts the contribution of different directional components. It is defined as

$$\mathfrak{DW}_g(f)(\xi, x, \omega) = \langle f, G_{\xi,x,\omega}\rangle. \tag{2.22}$$

This weighted extension refines directional analysis by compensating for directional loss in the DGRT and provides improved feature extraction, making it particularly useful in image processing and signal analysis. The DWGRT extends the functionality of the DGRT by incorporating derivative-based weighting, ensuring more stable and accurate reconstruction properties.

Connections to Discrete Directional Frames and Applications

The continuous transform framework of DGRT and its weighted extension DWGRT laid the groundwork for the development of *discrete directional Gabor frames*, as rigorously constructed and analyzed in [13]. That work demonstrates how ridge-based Gabor elements can be discretized across spatial and directional parameters to produce systems with proven frame bounds. Moreover, numerical experiments showed that such frames achieve competitive performance in tasks such as texture-preserving image denoising and compression—outperforming several established multiscale anisotropic techniques.

These theoretical developments are complemented by the dissertation [34], which provides explicit constructions and implementation strategies for directional Gabor systems. The dissertation presents a sufficient condition for discrete directional Gabor frames and includes practical examples applied to image signals. The framework ultimately supports applica-

tions including registration, superresolution, and multisensor image fusion, showcasing the versatility of DWGRT in high-resolution signal recovery and analysis.

2.2 Phase Retrieval and Learning-Based Approaches

Phase retrieval refers to the problem of reconstructing a signal from magnitude-only measurements, a fundamental challenge in fields such as optics, quantum mechanics, and signal processing. Many physical measurement systems, such as diffraction imaging and electron microscopy, inherently discard phase information, requiring computational techniques to recover the missing phase for accurate signal reconstruction. In time-frequency analysis, phase retrieval is particularly relevant in the context of the Short-Time Fourier Transform (STFT), where only the magnitude of the transform is recorded, necessitating specialized techniques for signal recovery.

Classical phase retrieval algorithms have traditionally relied on iterative projection methods, convex relaxations, and gradient-based optimization techniques. These approaches aim to reconstruct the missing phase while ensuring consistency with the available magnitude measurements. However, these methods are often limited by factors such as slow convergence, sensitivity to noise, and the presence of local minima in the optimization landscape. More recently, learning-based phase retrieval methods have emerged, leveraging neural networks to approximate phase reconstruction through data-driven modeling. The following subsections provide a detailed discussion of classical and deep learning-based phase retrieval techniques.

2.2.1 Classical Phase Retrieval Algorithms

Classical approaches to phase retrieval primarily fall into three broad categories: projection-based iterative algorithms, optimization-based methods, and convex relaxation techniques. Each of these methods attempts to estimate the missing phase while enforcing constraints that ensure consistency with the observed magnitude measurements.

Griffin–Lim Algorithm

The Griffin–Lim algorithm is an iterative approach designed for phase retrieval from Short-Time Fourier Transform (STFT) magnitude measurements. Introduced by Griffin and Lim in 1984, it attempts to reconstruct a signal by alternating between time and frequency domains, iteratively refining the estimated phase to ensure consistency with the observed magnitude.

Given an initial phase estimate $\phi^{(0)}(m,n)$, the algorithm follows these iterative steps:

1. Construct a complex-valued STFT using the observed magnitudes:

$$V^{(k)}(m,n) = |V_w x(m,n)| e^{i\phi^{(k)}(m,n)}$$

2. Compute the inverse STFT to reconstruct a time-domain signal:

$$x^{(k)}(t) = \text{ISTFT}(V^{(k)})$$

3. Compute the STFT of $x^{(k)}(t)$ and update the phase estimate:

$$\phi^{(k+1)}(m,n) = \arg(V_w x^{(k)}(m,n))$$

4. Iterate until the solution converges.

While the Griffin–Lim algorithm is widely used in applications such as speech processing and audio reconstruction, it has several limitations. The STFT magnitude data may not uniquely determine the original signal, leading to cases where different signals yield the same magnitude measurements. The algorithm does not necessarily converge to a globally optimal solution, but instead settles on a local minimum. It also relies on the Nonzero Overlap-Add (NOLA) condition for STFT inversion. If this condition is not met, for example, due to a large hop size in STFT computation, the algorithm fails.

Several modifications of the Griffin–Lim algorithm have been proposed to improve its performance. The Fast Griffin–Lim Algorithm introduces acceleration techniques to improve convergence speed. However, these modifications do not fundamentally resolve the uniqueness and stability issues inherent to STFT phase retrieval.

Wirtinger Flow and Amplitude Flow

Phase retrieval can also be approached through optimization-based techniques that minimize an objective function defined over the space of possible signals. Wirtinger Flow and Amplitude Flow are two such methods that use gradient-based optimization to estimate the missing phase.

Wirtinger Flow reformulates phase retrieval as a nonlinear optimization problem:

$$\min_x \sum_{m,n} (|V_w x(m,n)|^2 - |y(m,n)|^2)^2,$$

where $y(m,n)$ represents the measured magnitudes. The method uses Wirtinger derivatives to compute gradients in the complex domain, enabling efficient optimization.

Wirtinger Flow follows these main steps. First, an initial estimate $x^{(0)}$ is computed using spectral methods. The algorithm then iteratively updates $x^{(k)}$ using gradient descent:

$$x^{(k+1)} = x^{(k)} - \eta \nabla F(x^{(k)})$$

where η is the step size and $\nabla F(x)$ is the Wirtinger gradient of the loss function. This process continues until the solution converges.

Amplitude Flow modifies the Wirtinger Flow approach by optimizing the amplitude error directly:

$$\min_x \sum_{m,n} (|V_w x(m,n)| - |y(m,n)|)^2.$$

This reformulation leads to a smoother loss function with fewer spurious local minima, improving convergence behavior. Amplitude Flow reduces sensitivity to initialization and noise compared to Wirtinger Flow while maintaining similar computational efficiency.

PhaseLift and Convex Relaxation Methods

Convex relaxation techniques, such as PhaseLift, reformulate phase retrieval as a convex optimization problem. Instead of directly estimating the phase, these methods lift the problem to a higher-dimensional space by constructing a rank-one matrix representation of the signal:

$$X = xx^*.$$

By converting the phase retrieval problem into a rank-constrained semidefinite program, convex relaxation methods transform the original non-convex problem into a convex one that can be solved efficiently. However, these methods have practical limitations:

- The computational complexity is significantly higher than iterative and gradient-based methods.
- They require additional constraints, such as sparsity or low-rank assumptions, to perform well in practical scenarios.
- They do not generalize well to STFT-based phase retrieval due to the structured nature of STFT measurements.

2.2.2 Deep Learning Approaches to Phase Retrieval

Recent advancements in deep learning have introduced data-driven approaches to phase retrieval, offering an alternative to traditional optimization-based methods. Instead of relying on iterative updates, neural networks are trained to approximate the inverse mapping from magnitude measurements to phase reconstructions.

Supervised Neural Network-Based Phase Retrieval

A common approach in learning-based phase retrieval involves a neural network on a dataset containing paired magnitude-phase information. Given an STFT magnitude measurement $|V_w x(m,n)|$, the neural network learns a mapping $\mathcal{F}$ that predicts the corresponding phase:

$$\tilde{\phi}(m,n) = \mathcal{F}(|V_w x(m,n)|).$$

The reconstructed signal is then computed as:

$$\tilde{x}(t) = \sum_{m,n} |V_w x(m,n)| e^{i\tilde{\phi}(m,n)} w(t-nT) e^{2\pi i m t}.$$

Deep learning models trained on large-scale datasets have demonstrated superior performance compared to classical phase retrieval methods, particularly in settings where traditional algorithms struggle due to limited data or noise.

2.3 Principal Component Analysis and Kernel Principal Component Analysis

2.3.1 Principal Component Analysis

Principal Component Analysis (PCA) is a classical linear method for dimensionality reduction that identifies the directions of maximal variance in a dataset. Given N observations in $\mathbb{R}^d$, one first subtracts the sample mean and forms the centered data matrix $X \in \mathbb{R}^{N\times d}$. The empirical covariance

$$\Sigma \;=\; \frac{1}{N} X^\top X$$

is then decomposed via

$$\Sigma\, v_i \;=\; \lambda_i\, v_i, \quad \lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_d \geqslant 0.$$

The top k eigenvectors $\{v_1, \ldots, v_k\}$ define a k-dimensional subspace onto which one projects X to obtain a reduced representation that captures most of the variance while discarding noise and redundant dimensions [36, 24].

2.3.2 Kernel Principal Component Analysis

Kernel PCA (kPCA) extends PCA to capture nonlinear structure by first mapping each data point $x_i \in \mathbb{R}^d$ implicitly into a high-dimensional feature space $\mathcal{F}$ via $\phi\colon \mathbb{R}^d \to \mathcal{F}$. Inner products in $\mathcal{F}$ are computed by a kernel function

$$k(x_i, x_j) \;=\; \langle \phi(x_i),\, \phi(x_j) \rangle_{\mathcal{F}},$$

yielding the $N \times N$ Gram matrix K. Common choices include the Gaussian (RBF) kernel $\exp(-\|x_i - x_j\|^2/(2\sigma^2))$ and the cosine similarity kernel.

In practice, kPCA is carried out as follows:

1. Compute $K \in \mathbb{R}^{N \times N}$ with entries $K_{ij} = k(x_i, x_j)$.

2. Center K in feature space:

$$K_c \;=\; K - \tfrac{1}{N}\mathbf{1}_N K - \tfrac{1}{N} K \mathbf{1}_N + \tfrac{1}{N^2}\mathbf{1}_N K \mathbf{1}_N,$$

   where $\mathbf{1}_N$ is the $N \times N$ matrix of all ones.

3. Solve the eigenvalue problem

$$K_c\, v_i \;=\; \lambda_i\, v_i.$$

4. Select the top m eigenvectors $\{v_1, \ldots, v_m\}$ corresponding to the largest eigenvalues.

5. Project a new point x via $\;[\, k(x, x_1), \ldots, k(x, x_N)\,]\, v_i$ for $i = 1, \ldots, m$.

Introduced by Schölkopf, Smola, and Müller [41, 40], kPCA uncovers nonlinear manifolds that are invisible to classical PCA. A broader treatment of kernel eigenmap techniques, including kernel PCA, and their application to hyperspectral data can be found in Widemann's dissertation [43].

Both PCA and kPCA are widely used for denoising, compression, manifold learning, and visualization, accelerating downstream algorithms by isolating dominant modes of variability in complex datasets.

2.4 Generative Adversarial Network and Wasserstein Generative Adversarial Network

2.4.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) consist of two neural networks, a generator G and a discriminator D, which are trained simultaneously through a minimax game. The generator G maps a sample z from a simple prior distribution $p_z(z)$ (e.g., a multivariate normal) to the data space, producing $G(z)$, while the discriminator D outputs a scalar $D(x)$ representing the probability that x came from the real data distribution $p_{\text{data}}(x)$ rather than from G. The objective function is

$$\min_G \max_D V(D,G) \;=\; \mathbb{E}_{x\sim p_{\text{data}}}\big[\log D(x)\big] \;+\; \mathbb{E}_{z\sim p_z}\big[\log\big(1-D\big(G(z)\big)\big)\big].$$

In this framework, D is trained to maximize the probability of assigning the correct label to both real samples and generated samples, while G is trained to minimize $\log\big(1-D\big(G(z)\big)\big)$, which is equivalent to maximizing $\log D\big(G(z)\big)$ in practice for stronger gradients. Once G perfectly replicates the real data distribution, D outputs $1/2$ everywhere. Empirically, GANs can generate high-fidelity samples in complex domains. However, training may suffer from instability, mode collapse, and vanishing gradients when the supports of the model distribution and the data distribution lie on low-dimensional manifolds that do not overlap. [19]

2.4.2 Wasserstein Generative Adversarial Networks

Wasserstein GAN (WGAN) addresses the instability and mode collapse issues of the original GAN by replacing the Jensen–Shannon divergence with the Earth-Mover (Wasserstein-1) distance between distributions. Let p_g denote the distribution induced by G, and let p_{data} denote the real data distribution. The Wasserstein-1 distance is defined as

$$W\big(p_g,\, p_{\text{data}}\big) \;=\; \inf_{\gamma^*\in\Pi\left(p_g,\, p_{\text{data}}\right)} \mathbb{E}_{(x,y)\sim\gamma^*}\big[\|x-y\|\big],$$

where $\Pi(p_g,\, p_{\text{data}})$ is the set of all joint distributions $\gamma^*(x,y)$ with marginals p_g and p_{data}. Using the Kantorovich–Rubinstein duality, the optimization becomes

$$W\big(p_g,\, p_{\text{data}}\big) \;=\; \sup_{\|f\|_L\leqslant 1} \mathbb{E}_{x\sim p_{\text{data}}}\big[f(x)\big] \;-\; \mathbb{E}_{z\sim p_z}\big[f\big(G(z)\big)\big],$$

where the supremum is taken over all 1-Lipschitz functions f. In practice, one parametrizes f by a neural network D (often called the critic) whose weights are clipped to a compact space to enforce the 1-Lipschitz constraint. The training criterion becomes:

$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim p_{\text{data}}}\big[D(x)\big] \; - \; \mathbb{E}_{z \sim p_z}\big[D\big(G(z)\big)\big],$$

where $\mathcal{D}$ is the set of all neural networks with weights constrained to lie within $[-c, c]$ for some constant c. This objective yields meaningful learning curves correlated with sample quality and significantly improved training stability compared to the original GAN. [3]

2.5 EM Nanoparticle and Cryo-EM Bio-Molecule Structure Reconstruction Using SIMPLE and SINGLE Pipelines

Two open-source pipelines, SIMPLE (Single-particle IMage Processing Linux Engine) and SINGLE (Single-Nanoparticle LIquid-Cell EM), demonstrate how near-real-time electron microscopy (EM) data can be processed to reconstruct three-dimensional (3D) structures of both biological macromolecules and inorganic nanocrystals. SIMPLE is primarily oriented toward single-particle cryo-EM workflows (including contrast transfer function (CTF) correction) and streaming data analysis, whereas SINGLE focuses on graphene-liquid-cell EM (GLC-EM) data for nanocrystals, where the major challenge is background subtraction rather than CTF. GLC-EM encapsulates colloidal nanocrystals between two monolayer graphene sheets, maintaining a thin liquid environment and allowing imaging at ambient temperature in an aberration-corrected TEM with direct electron detection at millisecond frame rates [39]. Despite these differences, the pipelines share similar computational steps, as discussed by Elmlund et al. in [6].

2.5.1 Motion Correction and Image Preprocessing

In each pipeline, the raw data arrive as detector-frame stacks that typically exhibit drift or other forms of motion. SIMPLE addresses beam-induced motion in cryo-EM by aligning individual frames, often applying dose weighting to mitigate radiation damage. In SINGLE, frames capture nanocrystals undergoing Brownian motion inside a graphene liquid cell; short windows of frames can be co-averaged if the nanocrystal's orientation remains relatively unchanged, thereby improving the signal-to-noise ratio (SNR) before further classification.

2.5.2 CTF Correction or Graphene Background Subtraction

For cryo-EM biomolecules, SIMPLE explicitly estimates and corrects the CTF:

$$\mathrm{CTF}(\mathbf{g}) = -\sin\big[\chi(\mathbf{g})\big],$$

where $\chi(\mathbf{g})$ encodes defocus, spherical aberration, and related parameters. This step is critical for achieving high resolution in single-particle cryo-EM reconstructions. By contrast, SINGLE deals with strong graphene signals in liquid-cell datasets. Instead of a standard CTF correction, the pipeline identifies and masks the hexagonal graphene peaks in Fourier space, removing background interference from each micrograph.

2.5.3 Particle or Nanocrystal Identification and Extraction

After motion correction, the next step is locating and extracting the 2D views of interest. In SIMPLE, thousands of candidate biomolecular particles are typically picked from a cryo-EM micrograph using template-based or reference-based methods. In SINGLE, a nanocrystal's position is tracked across frames in a time-lapse manner; each group of frames that capture the crystal at different orientations can be gathered into a coherent "trajectory" for downstream analysis.

2.5.4 2D Classification and Class Averaging

Both pipelines rely on 2D classification to enhance SNR. SIMPLE applies clustering and alignment of many cryo-EM particle images, discarding poorly aligned or contaminated subsets. Each group yields a class average that highlights key features more clearly than any single raw image. SINGLE similarly forms local 2D averages of nanocrystal frames if the orientation remains stable over a short interval (class averaging). Out-of-focus frames or abrupt changes are excluded, boosting the clarity of crystal lattice views and facilitating reliable 3D reconstruction.

2.5.5 3D Reconstruction Under the Fourier Slice Theorem

Finally, both pipelines assemble the 2D class averages into a 3D density map using the Fourier slice theorem. Each 2D projection is treated as a central slice of the 3D Fourier transform. By placing these slices in their correct orientations and performing an inverse transform, one obtains the 3D distribution. SIMPLE often uses iterative Bayesian or maximum-likelihood approaches typical of single-particle cryo-EM, refining orientation

and CTF parameters together. SINGLE may start from a lattice-based model or rely on incremental angle assignments, aided by the previously removed graphene background, to reveal near-atomic details of a nanocrystal's core.

Chapter 3

Uncertainty Principles in Directional Gabor Ridge Transform

3.1 Introduction

The study of uncertainty principles has been a fundamental topic in harmonic analysis and signal processing. These principles establish intrinsic limitations on the simultaneous localization of a function and its frequency representation. Classical uncertainty principles, such as Heisenberg's inequality, state that a function and its Fourier transform cannot both be sharply localized. This concept has been extensively generalized in various contexts, including time-frequency analysis and directional transforms.

A significant development in time-frequency analysis is the introduction of the directional short-time Fourier transform, which extends the classical short-time Fourier transform by incorporating directional sensitivity. The directional short-time Fourier transform was introduced by Giv [17] as a tool for analyzing signals with directional characteristics. Mejjaoli and Omri [32] further investigated this transform and established various quantitative uncertainty principles associated with it. Their work provided a deeper understanding of how directional information affects time-frequency localization.

Grafakos and Sansing [20] pioneered the directional Gabor ridge transform, first in a fully continuous setting and then in a semi-discrete form. Building on their framework, Murphy developed the first fully discrete directional Gabor ridge transform in Chapter 3 of his 2015 dissertation [34], where he constructed discrete DGRT frames and proved their frame bounds and reconstruction properties.

Motivated by these works, I study uncertainty principles in the setting of the directional Gabor ridge transform and its weighted variant. My goal is to establish new uncertainty

inequalities that characterize the limitations of simultaneous localization in this framework.

This chapter presents uncertainty principles for the directional Gabor ridge transform and the directional weighted Gabor ridge transform. Following the ideas developed for the directional short-time Fourier transform, I derive quantitative bounds that govern the time-frequency concentration of functions under these transforms. The results extend previous uncertainty principles to a broader class of directional time-frequency transforms.

3.2 Comparison between directional short time Fourier transform and directional Gabor ridge transform

The directional short-time Fourier transform (DSTFT) and the directional Gabor ridge transform (DGRT) are both extensions of classical time-frequency analysis methods that incorporate directional sensitivity. Their formulations share similarities in structure but differ in the way they apply directional analysis and localization. Understanding their differences and similarities provides insight into their respective advantages and applications.

3.2.1 Mathematical Comparison

The DSTFT of a function f is given by equation (2.8), where a window function is applied along a specified direction before computing the Fourier transform. Similarly, the DGRT is defined in equation (2.18), where the function is projected onto a subspace aligned with the directional parameter before analyzing its frequency content.

Both transforms share the fundamental structure of projecting onto a directionally dependent function before applying modulation. However, the DSTFT operates directly in the time-frequency domain using a localized Fourier analysis, while the DGRT is tied to Gabor frames, incorporating both translation and modulation in a more structured frame-theoretic way.

3.2.2 Directional Sensitivity in DSTFT and DGRT

Directional sensitivity in both the DSTFT and DGRT arises from the inclusion of the directional parameter $\xi \in S^{n-1}$. Instead of applying an isotropic window function as in the classical STFT or Gabor transform, these directional methods integrate along specific orientations. The key factor that enables directional sensitivity is the modification of the window function g so that it aligns with a given direction before being applied to the signal.

In the DSTFT, the localization function g is applied along the direction ξ, meaning that the transform emphasizes frequency components that are dominant along that particular orientation. Similarly, the DGRT applies its window function along a ridge direction, allowing it to capture structural features such as edges and anisotropic patterns.

Directional sensitivity is crucial in many applications, particularly for signals that exhibit strong directional behavior. Examples include medical imaging, where directional analysis improves edge detection in tomography, and image processing, where detecting ridges and oriented features is fundamental for object recognition.

3.2.3 Importance of Directional Sensitivity in Time-Frequency Analysis

In classical time-frequency representations such as the STFT or standard Gabor transform, frequency content is analyzed without taking into account the orientation of features within the signal. However, many real-world signals, particularly in higher dimensions, exhibit strong anisotropic characteristics.

For instance, in image processing, natural images contain edges and textures that are primarily oriented along specific directions. A standard time-frequency analysis method would treat all directions equally, failing to highlight the underlying structure of the image. By introducing directional components, the DSTFT and DGRT allow for a more refined representation of signals, where frequency components are analyzed with respect to their orientation.

In medical imaging, directional sensitivity is essential for techniques such as computed tomography (CT) and magnetic resonance imaging (MRI). These modalities rely on analyzing structures within the body that have inherent directional properties, such as blood vessels or muscle fibers. By incorporating directional sensitivity, time-frequency methods can improve the accuracy of reconstructions and feature extraction.

In applications such as radar and seismic analysis, signals often contain waves that propagate in specific directions. Standard time-frequency analysis methods struggle to isolate such direction-dependent information, while directional transforms like DSTFT and DGRT can provide a clearer picture of how energy is distributed across different orientations.

The key takeaway is that directional sensitivity enables time-frequency methods to align better with the underlying geometry of signals, making them more effective for analyzing structured data. Both the DSTFT and DGRT achieve this by incorporating a directional parameter that modifies the analysis window, but their formulations differ in how they structure time-frequency decomposition. The DSTFT extends the classical STFT by incor-

porating an oriented window function, while the DGRT builds upon the Gabor framework with ridge-based directional filtering.

3.3 Giv's Theorems and Operator Properties of the Directional Short-Time Fourier Transform

We begin by defining the underlying space on which the Directional Short-Time Fourier Transform (DSTFT) acts.

3.3.1 The Space Δ^*

For a function f on $\mathbb{R}^n$, the DSTFT is defined with respect to a window function g on $\mathbb{R}$, mapping f to a function on the space

$$S^{n-1} \times \mathbb{R} \times \mathbb{R}^n.$$

Explicitly, for $(\xi, x, \omega) \in S^{n-1} \times \mathbb{R} \times \mathbb{R}^n$, the DSTFT is given by

$$\mathcal{D}_g f(\xi, x, \omega) = \int_{\mathbb{R}^n} f(t)\overline{g(\xi \cdot t - x)}e^{-2\pi i\omega \cdot t}dt. \tag{3.1}$$

The space $S^{n-1} \times \mathbb{R} \times \mathbb{R}^n$ is denoted by Δ^*, with the associated Haar measure

$$d\delta^*(\xi, x, \omega) = d\xi \otimes dx \otimes d\omega,$$

where dx and $d\omega$ are Lebesgue measures on the corresponding Euclidean spaces, and $d\xi$ is the normalized surface measure on S^{n-1}. The DSTFT operator introduced by Giv [17] is denoted as $\mathcal{D}_g$.

3.3.2 Boundedness and Orthogonality of the DSTFT

The following theorems establish fundamental properties of the DSTFT, including boundedness, orthogonality, and L^p-mapping properties.

Theorem 3.1 (Proposition 2.2, [17]). *If $g \in L^\infty(\mathbb{R})$, then the operator $\mathcal{D}_g$ is bounded from $L^1(\mathbb{R}^n)$ to $L^\infty(\Delta^*)$, with the norm estimate*

$$\|\mathcal{D}_g\| \leqslant \|g\|_\infty. \tag{3.2}$$

Proof. For every $(\xi, x, \omega) \in \Delta$, we have

$$\begin{aligned}|\mathcal{D}_g f(\xi, x, \omega)| &= \left| \int_{\mathbb{R}^n} f(t) g(\xi \cdot t - x) e^{-2\pi i \omega \cdot t} dt \right| \\ &\leqslant \int_{\mathbb{R}^n} |f(t)||g(\xi \cdot t - x)| dt \\ &\leqslant \|g\|_\infty \int_{\mathbb{R}^n} |f(t)| dt.\end{aligned}$$

Thus, $\mathcal{D}_g f \in L^\infty(\Delta^*)$ and the operator norm satisfies $\|\mathcal{D}_g\| \leqslant \|g\|_\infty$. □

Theorem 3.2 (Theorem 2.4, [17]). *Suppose $g_1, g_2 \in L^\infty(\mathbb{R})$ and $f_1, f_2 \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$. If at least one of the g_i is in $L^1(\mathbb{R})$, then the following orthogonality relation holds:*

$$\int_{\Delta^*} \mathcal{D}_{g_1} f_1(\delta^*) \overline{\mathcal{D}_{g_2} f_2(\delta^*)} d\delta^* = \langle f_1, f_2 \rangle \langle g_2, g_1 \rangle. \tag{3.3}$$

In particular, if $g \in L^1(\mathbb{R}) \cap L^\infty(\mathbb{R})$ and $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, then $\mathcal{D}_g f \in L^2(\Delta^*)$, and

$$\|\mathcal{D}_g f\|_2 = \|g\|_2 \|f\|_2. \tag{3.4}$$

Proof. Using the definition of $\mathcal{D}_g f$ and Fubini's theorem, we obtain

$$\int_{\Delta^*} \mathcal{D}_{g_1} f_1(\delta^*) \overline{\mathcal{D}_{g_2} f_2(\delta^*)} d\delta^* = \int_{\mathbb{R}^n} f_1(t) f_2(t) \int_{\Delta^*} g_1(\xi \cdot t - x) \overline{g_2(\xi \cdot t - x)} d\delta^* dt.$$

Since at least one of the g_i is in $L^1(\mathbb{R})$, the integral in the inner product is absolutely convergent, which gives the desired result. □

Theorem 3.3 (Theorem 2.5, [17]). *For $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $g \in L^1(\mathbb{R}) \cap L^\infty(\mathbb{R})$, we have $\mathcal{D}_g f \in L^2(\Delta^*) \cap L^\infty(\Delta^*)$. By the properties of L^p-spaces, for all $2 < p < \infty$, we have $\mathcal{D}_g f \in L^p(\Delta^*)$. Moreover,*

$$\|\mathcal{D}_g f\|_p \leqslant \|g\|_p \|f\|_q. \tag{3.5}$$

Proof. Applying the Hausdorff–Young inequality, we estimate:

$$\begin{aligned}\|\mathcal{D}_g f\|_p &= \left(\int_{\Delta^*} |\mathcal{D}_g f(\delta^*)|^p d\delta^* \right)^{1/p} \\ &\leqslant \left(\int_{\Delta^*} \left| \int_{\mathbb{R}^n} f(t) g(\xi \cdot t - x) e^{-2\pi i \omega \cdot t} dt \right|^p d\delta^* \right)^{1/p}.\end{aligned}$$

Using Minkowski's integral inequality and Fubini's theorem, we obtain

$$\begin{aligned}\|\mathcal{D}_g f\|_p &\leqslant \int_{\mathbb{R}^n} |f(t)| \left(\int_{\Delta^*} |g(\xi \cdot t - x)|^p d\delta^* \right)^{1/p} dt \\ &= \|g\|_p \|f\|_q.\end{aligned}$$

□

These results establish fundamental properties of the DSTFT, setting the stage for the development of uncertainty principles in the directional setting.

3.4 Uncertainty Principles in the Directional Short-Time Fourier Transform

This section presents key uncertainty inequalities for the Directional Short-Time Fourier Transform (DSTFT), following the approach developed by Mejjaoli and Omri [32]. These results refine the classical uncertainty principles by incorporating directional and phase-space localization estimates.

3.4.1 Weighted Norm Inequalities for the DSTFT

To establish uncertainty principles associated with the DSTFT, we first introduce a fundamental weight function that encodes the spatial-frequency structure:

$$|(\xi, x, \omega)| := \sqrt{x^2 + |\omega|^2 + 1}.$$

Using this, we define the heat kernel:

$$h_t(\xi, x, \omega) := e^{-t|(\xi,x,\omega)|^2}.$$

This kernel plays a crucial role in deriving weighted norm inequalities. We also denote the Hölder conjugate of p as q, where $\frac{1}{p} + \frac{1}{q} = 1$.

Lemma 3.4. *Let $1 < p \leqslant 2$ and $0 < a < \frac{n}{q}$. Suppose $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then there exists a constant $C > 0$ such that for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $t > 0$, we have*

$$\|h_t(\xi, x, \omega)\mathcal{D}_g f\|_q \leqslant C(\|g\|_\infty + \|g\|_q) t^{-\frac{(n+1)a}{2n}} \||y|^a f\|_p, \tag{3.6}$$

where

$$C = \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t} \cdot \max\left\{1, \left(\frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}\right)^{\frac{1}{q}}\right\}.$$

Proof. We assume $\||y|^a f(y)\|_p < \infty$. For $s > 0$, define the truncated function:

$$f_s(y) := f(y)\mathbb{1}_{B(0,s)}(y), \quad f^s := f(y) - f_s(y).$$

Since $|f^s(y)| \leqslant s^{-a}||y|^a f(y)|$, applying Theorem 3, we obtain:

$$\begin{aligned}\|h_t(\xi,x,\omega)\mathcal{D}_g f^s\|_q &\leqslant \|\mathcal{D}_g f^s\|_q, \quad \text{since} \quad \|h_t(\xi,x,\omega)\|_\infty \leqslant 1\\ &\leqslant \|g\|_q \|f^s\|_p\\ &\leqslant \|g\|_q \big(s^{-a}\||y|^a f\|_p\big).\end{aligned}$$

On the other hand, using Theorem 1 and Hölder's inequality, we estimate:

$$\begin{aligned}\|h_t(\xi,x,\omega)\mathcal{D}_g f_s\|_q &\leqslant \|h_t(\xi,x,\omega)\|_q \|\mathcal{D}_g f_s\|_\infty\\ &\leqslant \|h_t(\xi,x,\omega)\|_q \|g\|_\infty \|f_s\|_1\\ &\leqslant \|h_t(\xi,x,\omega)\|_q \|g\|_\infty \cdot \||y|^{-a}\mathbb{1}_{B(0,s)}\|_q \cdot \||y|^a f\|_p.\end{aligned}$$

By direct computation, we find:

$$\||y|^{-a}\mathbb{1}_{B(0,s)}\|_q = \left(\frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}\right)^{1/q} =: C_2.$$

Furthermore, we use the known bound:

$$\|h_t\|_q \leqslant \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t} \cdot t^{-\frac{n+1}{2q}}.$$

Setting $C_1 := \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t}$, and choosing $s = t^{\frac{n+1}{2n}}$, we obtain:

$$\begin{aligned}\|h_t(\xi,x,\omega)\mathcal{D}_g f\|_q &\leqslant \|h_t(\xi,x,\omega)\mathcal{D}_g f_s\|_q + \|h_t(\xi,x,\omega)\mathcal{D}_g f^s\|_q\\ &\leqslant C_2 s^{-a}\Big(\|g\|_q + \|g\|_\infty s^{n/q}\|h_t\|_q\Big)\||y|^a f\|_p\\ &= C(\|g\|_\infty + \|g\|_q)t^{-\frac{(n+1)a}{2n}}\||y|^a f\|_p,\end{aligned}$$

where $C = C_1 \cdot \max\{1, C_2\}$. □

The result obtained in this lemma provides a key inequality relating the weighted norm of the DSTFT with the weighted norm of the function itself, indicating the decay properties of the transform under heat kernel modulation. This plays an essential role in deriving refined uncertainty principles.

Theorem 3.5. *Let $1 < p \leqslant 2$, $0 < a < \frac{n}{q}$, and $0 < b \leqslant 2$. Suppose $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then, there exists a constant $C_3 > 0$ such that for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we have the following weighted norm inequality:*

$$\|\mathcal{D}_g f\|_q \leqslant 2C_3 \left[\left(\big(\|g\|_\infty + \|g\|_q\big) \||y|^a f\|_p \right)^{\frac{bn}{an+bn+a}} \||(\xi,x,\omega)|^b \mathcal{D}_g(f)\|_q^{\frac{an+a}{an+bn+a}} \right], \tag{3.7}$$

where the constant C_3 is given by:

$$C_3 = \max\left\{ \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t}, \left(\frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}\right)^{\frac{1}{q}} \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t}, 1 \right\}.$$

Moreover, the optimal parameter t satisfies:

$$t = \left(\frac{\big(\|g\|_\infty + \|g\|_q\big) \||y|^a f\|_p}{\||(\xi,x,\omega)|^b \mathcal{D}_g f\|_q} \right)^{\frac{2n}{an+bn+a}}.$$

Proof. Following a similar argument as in Lemma 3.4, we estimate:

$$\|(1-h_t)\mathcal{D}_g f\|_q = t^{b/2} \left\| (t|(\xi,x,\omega)|^2)^{-b/2} (1-h_t) |(\xi,x,\omega)|^b \mathcal{D}_g f \right\|_q.$$

Since $(1-e^{-k})k^{-b/2}$ is bounded for $k > 0$, the result follows by direct calculation and the bounds established in Lemma 3.4. □

Corollary 3.6. *Let $1 < p \leqslant 2$, $0 < a < \frac{n}{q}$, and $b > 2$. Assume that $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then, there exists a constant $C_3 > 0$ such that for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we have*

$$\|\mathcal{D}_g f\|_q \leqslant 2^{\frac{b(2an+2a+b'n)}{b'(bn+an+a)}} (C_3)^{\frac{b(an+b'n+a)}{b'(bn+an+a)}} \left[\left(\big(\|g\|_\infty + \|g\|_q\big) \||y|^a f\|_p \right)^{\frac{bn}{bn+an+a}} \||(\xi,x,\omega)|^b \mathcal{D}_g(f)\|_q^{\frac{an+a}{bn+an+a}} \right], \tag{3.8}$$

where C_3 is the constant defined in Theorem 3.5 and b' is the Hölder conjugate of b, i.e., $\frac{1}{b} + \frac{1}{b'} = 1$.

Proof. If $b > 2$, then $b' \leqslant 2 < b$. Let us define the auxiliary variable:

$$u := \frac{|(\xi,x,\omega)|}{\epsilon}.$$

By a fundamental inequality, we have

$$u^{b'} < 1 + u^b, \quad \forall u > 0, \quad \epsilon > 0.$$

Applying this to the transformation of norms,

$$\begin{aligned}\||(\xi,x,\omega)|^{b'}\mathcal{D}_g f\|_q &\leqslant \|\epsilon^{b'}[1+\frac{|(\xi,x,\omega)|^b}{\epsilon^b}]\mathcal{D}_g f\|_q \\ &\leqslant \epsilon^{b'}\|\mathcal{D}_g f\|_q + \epsilon^{b'-b}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q.\end{aligned}$$

Optimizing by choosing

$$\epsilon = \left(\frac{\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q}{\|\mathcal{D}_g f\|_q}\right)^{1/b},$$

we get

$$\||(\xi,x,\omega)|^{b'}\mathcal{D}_g f\|_q \leqslant 2\|\mathcal{D}_g f\|_q^{\frac{b-b'}{b}}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q^{\frac{b'}{b}}.$$

Applying Theorem 3.5 for b', we obtain

$$\begin{aligned}\|\mathcal{D}_g f\|_q &\leqslant 2C_3\left[\left(\big(\|g\|_\infty + \|g\|_q\big)\||y|^a f\|_p\right)^{\frac{b'n}{an+b'n+a}}\||(\xi,x,\omega)|^{b'}\mathcal{D}_g f\|_q^{\frac{an+a}{an+b'n+a}}\right] \\ &\leqslant 2^{1+\frac{an+a}{an+b'n+a}}C_3\left[\left(\big(\|g\|_\infty + \|g\|_q\big)\||y|^a f\|_p\right)^{\frac{b'n}{an+b'n+a}}\|\mathcal{D}_g f\|_q^{\frac{(b-b')(an+a)}{b(an+b'n+a)}}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q^{\frac{b'(an+a)}{b(an+b'n+a)}}\right].\end{aligned}$$

Rearranging, we obtain:

$$\|\mathcal{D}_g f\|_q^{\frac{b'(bn+an+a)}{b(an+b'n+a)}} \leqslant 2^{\frac{2an+2a+b'n}{an+b'n+a}}C_3\left[\left(\big(\|g\|_\infty + \|g\|_q\big)\||y|^a f\|_p\right)^{\frac{b'n}{an+b'n+a}}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q^{\frac{b'(an+a)}{b(an+b'n+a)}}\right].$$

Taking appropriate exponents on both sides, we conclude:

$$\|\mathcal{D}_g f\|_q \leqslant 2^{\frac{b(2an+2a+b'n)}{b'(bn+an+a)}}(C_3)^{\frac{b(an+b'n+a)}{b'(bn+an+a)}}\left[\left(\big(\|g\|_\infty + \|g\|_q\big)\||y|^a f\|_p\right)^{\frac{bn}{bn+an+a}}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_q^{\frac{an+a}{bn+an+a}}\right].$$

□

Remark 1. *Corollary 3.6 extends Theorem 3.5 to the case where $b > 2$, which leads to a refined estimate through interpolation techniques. This result generalizes classical uncertainty inequalities in the DSTFT setting.*

Corollary 3.7. *Let a, b be two positive real numbers, and let a' be the Hölder conjugate of*

a, satisfying $\frac{1}{a}+\frac{1}{a'}=1$. *Assume* $g\in L^2(\mathbb{R})\cap L^\infty(\mathbb{R})$. *Then, there exists a positive constant* C_4 *such that, for every* $f(y)\in L^1(\mathbb{R}^n)\cap L^2(\mathbb{R}^n)$, *we have*

$$\|f\|_{L^2(\mathbb{R}^n)}\leqslant C_4\||y|^a f\|_{L^2(\mathbb{R}^n)}^{\frac{bn}{an+bn+a}}\||(\xi,x,\omega)|^b\mathcal{D}_g f\|_{L^2(\Delta^*)}^{\frac{an+a}{an+bn+a}}. \tag{3.9}$$

For the case where $p=q=2$, *the constant* C_4 *is given by*

$$C_4=\begin{cases}\widetilde{C}\frac{(\|g\|_\infty+\|g\|_2)^{\frac{bn}{an+bn+a}}}{\|g\|_2}, & \text{for } 0<a<\frac{n}{2},\\ [2(\|g\|_\infty+\|g\|_2)]^{\frac{abn}{a'(an+bn+a)}}\left(\frac{\widetilde{C}}{\|g\|_2}\right)^{\frac{a(a'n+bn+a')}{a'(an+bn+a)}}, & \text{for } a\geqslant\frac{n}{2},\end{cases} \tag{3.10}$$

where

$$\widetilde{C}:=\begin{cases}2C_3, & \text{for } b\leqslant 2,\\ 2^{\frac{b(2ad+2a+b'd)}{b'(bd+ad+a)}}(C_3)^{\frac{b(ad+b'd+a)}{b'(bd+ad+a)}}, & \text{for } b>2.\end{cases}$$

Proof. By applying Theorem 3.5 and Theorem 3.2, we use the Plancherel-type formula to obtain the desired bound for $0<a<\frac{n}{2}$.

To analyze the case $a\geqslant\frac{n}{2}$, we apply a standard technique using the auxiliary variable

$$u=\frac{|y|}{\epsilon}.$$

As in the derivation of Corollary 3.6, we obtain the inequality

$$\||y|^{a'}f\|_2\leqslant 2\|f\|_2^{\frac{a-a'}{a}}\||y|^a f\|_2^{\frac{a'}{a}}.$$

Optimizing over the choice of ϵ yields the expression for C_4. □

Remark 2. *Corollary 3.7 establishes a weighted uncertainty principle for the DSTFT, demonstrating a trade-off between the spatial localization of* f *and its corresponding representation in the transformed domain. The result strengthens classical uncertainty inequalities by incorporating directional dependencies in the DSTFT setting.*

3.5 Fundamental Properties of DGRT and DWGRT

The Directional Gabor Ridge Transform (DGRT) and the Directional Windowed Gabor Ridge Transform (DWGRT) extend classical time-frequency analysis techniques by incorporating directional features. To establish a solid theoretical foundation for these transforms,

we first recall key functional inequalities and structural properties essential for the subsequent analysis.

3.5.1 Structural Properties of the DGRT

We now introduce fundamental properties of the DGRT, beginning with an essential lemma from Grafakos and Sansing [20].

Lemma 3.8 ([20, Lemma 1])**.** *For any function $f \in L^1(\mathbb{R}^n)$ and $g \in \mathcal{S}(\mathbb{R})$, the following identity holds:*

$$\langle f, g_{\xi,x,\omega} \rangle = \langle R_\xi(f), g^{x,\omega} \rangle.$$

Proof. By direct computation, we obtain:

$$\begin{aligned}
\langle f, g_{\xi,x,\omega} \rangle &= \int_{\mathbb{R}^n} f(t) e^{-2\pi i \omega(\xi \cdot t - x)} \overline{g(\xi \cdot t - x)} dt \\
&= \int_{\mathbb{R}} \left(\int_{\xi \cdot t = s} f(x) e^{-2\pi i \omega(\xi \cdot t - x)} \overline{g(\xi \cdot t - x)} dt \right) ds \\
&= \int_{\mathbb{R}} R_\xi f(s) \overline{g^{x,\omega}(s)} ds \\
&= \langle R_\xi(f), g^{x,\omega} \rangle.
\end{aligned}$$

□

3.5.2 Boundedness Properties of the DGRT

The following theorem establishes norm estimates for the DGRT operator.

Theorem 3.9. *(a) If $g \in L^\infty(\mathbb{R})$ and $f \in L^1(\mathbb{R}^n)$, then the DGRT operator $\mathfrak{D}_g$ is bounded from $L^1(\mathbb{R}^n)$ to $L^\infty(\Delta)$, with operator norm satisfying*

$$\|\mathfrak{D}_g\|_\infty \leqslant \|g\|_\infty.$$

(b) Moreover, if $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$, then $\mathfrak{D}_g$ maps $L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ into $L^2(\Delta)$, with norm estimate

$$\|\mathfrak{D}_g f\|_2 = \|g\|_2 \|f\|_2.$$

Proof. (a) For any $(\xi, x, \omega) \in \Delta$, we estimate:

$$\begin{aligned}
|\mathfrak{D}_g f(\xi, x, \omega)| &\leqslant \int_{\mathbb{R}^n} |f(t) g_{\xi,x,\omega}(t)| dt \\
&\leqslant \int_{\mathbb{R}^n} |f(t)||g_{\xi,x}(t)| dt \\
&\leqslant \|g_{\xi,x}\|_\infty \|f\|_1 \\
&\leqslant \|g\|_\infty \|f\|_1.
\end{aligned}$$

(b) For $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we use Plancherel's theorem:

$$\begin{aligned}
\|\mathfrak{D}_g f\|_2^2 &= \int_\Delta |\mathfrak{D}_g f(\gamma)|^2 d\gamma \\
&= \int_{\mathbb{S}^{n-1}} \int_{\mathbb{R}} \int_{\mathbb{R}^n} f(t)\overline{f(t)} g_{\xi,x}(t)\overline{g_{\xi,x}(t)} dt dx d\xi \\
&= \int_{\mathbb{R}^n} |f(t)|^2 \bigg(\int_{\mathbb{S}^{n-1}} \int_{\mathbb{R}} |g(t)|^2 dx d\xi \bigg) dt \\
&= \|f\|_2^2 \|g\|_2^2.
\end{aligned}$$

□

Lemma 3.10 ([20, Lemma 2])**.** *Let $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$. Then $D_{\frac{n-1}{2}}\big(R_\xi(f)\big) \in L^2(\mathbb{R})$ for almost every $\xi \in \mathbb{S}^{n-1}$. Moreover, $\big|\langle f, G_{\xi,x,\omega}\rangle\big|$ is finite for almost all $x, \omega \in \mathbb{R}$ and $\xi \in \mathbb{S}^{n-1}$.*

Proof. By the Fourier slice theorem, we have

$$\begin{aligned}
\|f\|_{L^2(\mathbb{R}^n)}^2 &= \int_{\mathbb{R}^n} |f(t)|^2 \, dt = \int_{\mathbb{R}^n} |\widehat{f}(\gamma)|^2 \, d\gamma \\
&= \int_{\mathbb{S}^{n-1}} \int_0^\infty \Big| \widehat{R_\xi(f)}(\sigma) \, |\sigma|^{\frac{n-1}{2}} \Big|^2 d\sigma \, d\xi = \frac{1}{2} \int_{\mathbb{S}^{n-1}} \int_{\mathbb{R}} \Big| D_{\frac{n-1}{2}}\big(R_\xi(f)\big)(s) \Big|^2 ds \, d\xi.
\end{aligned}$$

Hence, $D_{\frac{n-1}{2}}\big(R_\xi(f)\big)$ belongs to $L^2(\mathbb{R})$ for almost every $\xi \in \mathbb{S}^{n-1}$.

Next, let $\boldsymbol{\delta}$ be the Dirac-δ distribution. Observe that

$$\begin{aligned}
\big\langle \widehat{f}, \widehat{G_{\xi,x,\omega}} \big\rangle &= \int_{\mathbb{R}^n} \widehat{f}(\gamma) \, \overline{\widehat{g}\big(\xi\cdot\gamma - \omega\big)} \, e^{2\pi i (\,\xi\cdot\gamma\,) \, x} \, |\xi\cdot\gamma|^{\frac{n-1}{2}} \, \boldsymbol{\delta}\big(\gamma - (\,\xi\cdot\gamma\,)\,\xi\big) \, d\gamma \\
&= \int_{\mathbb{R}} \widehat{f}\big(\lambda\xi\big) \, \overline{\widehat{g}\big(\lambda - \omega\big)} \, e^{2\pi i \lambda x} \, |\lambda|^{\frac{n-1}{2}} \, d\lambda.
\end{aligned}$$

Since $\big|\widehat{f}(\lambda\,\xi)\big| \leqslant \|f\|_{L^1(\mathbb{R}^n)}$ and $\widehat{g}(\lambda - \omega)$ decays rapidly, the above integral is finite. This shows that for almost all ξ, x, and ω, the inner product $\langle f, G_{\xi,x,\omega}\rangle$ remains bounded as claimed. □

3.5.3 Square Integrability in the DWGRT Setting

Next, we consider the DWGRT, which involves additional windowing in the frequency domain.

Theorem 3.11. *If $g \in \mathcal{S}(\mathbb{R})$ and $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, then*

$$\|\mathfrak{DW}_g f\|_2 = 2\|f\|_2\|g\|_2.$$

Proof. Applying the Fourier slice theorem and Plancherel's theorem, we find that:

$$\|\mathfrak{DW}_g f\|_2^2 = 2\|g\|_{L^2(\mathbb{R})}^2\|f\|_{L^2(\mathbb{R}^n)}^2.$$

□

This result establishes the isometry property of the DWGRT, ensuring stability in L^2-norms.

3.6 Uncertainty Principles for DGRT and DWGRT

The uncertainty principle in harmonic analysis provides fundamental limits on the simultaneous concentration of a function and its transform. In the context of the Directional Gabor Ridge Transform (DGRT) and Directional Windowed Gabor Ridge Transform (DWGRT), this principle extends to characterizing trade-offs between spatial and frequency localization. This section presents a series of results that quantify such relationships using weighted norm inequalities.

3.6.1 Notation and Preliminaries

Consider the phase-space domain

$$\Delta = \mathbb{S}^{n-1} \times \mathbb{R} \times \mathbb{R},$$

where the triplet (ξ, x, ω) represents a directional coordinate system incorporating spatial and frequency variables.

For analysis in the DGRT setting, we define the weighted function

$$|(\xi, x, \omega)| := \sqrt{x^2 + \omega^2 + 1},$$

which extends the Euclidean norm in $\mathbb{R}^{n+2}$. Additionally, we introduce the heat kernel

$$h_t(\xi, x, \omega) := e^{-t|(\xi,x,\omega)|^2},$$

which acts as a smoothing operator. As before, let q be the Hölder conjugate of p, i.e., $\frac{1}{p} + \frac{1}{q} = 1$.

3.6.2 Weighted Norm Inequalities for DGRT

We begin with a fundamental weighted norm inequality for the DGRT.

Lemma 3.12. *Let $1 < p \leqslant 2$ and $0 < a < \frac{n}{q}$. Suppose $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then there exists a constant $\mathcal{C}_1 > 0$ such that for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $t > 0$, we have*

$$\|h_t(\xi, x, \omega)\mathfrak{D}_g f\|_q \leqslant \mathcal{C}_1(\|g\|_\infty + \|g\|_q)t^{-\frac{(n+1)a}{2n}}\||y|^a f\|_p, \tag{3.11}$$

where

$$\mathcal{C} = \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t} \cdot \max\left\{1, \left(\frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}\right)^{\frac{1}{q}}\right\}.$$

Proof. We assume $\||y|^a f(y)\|_p < \infty$. For $s > 0$, define

$$f_s(y) := f(y)\mathbb{1}_{B(0,s)}(y), \quad f^s := f(y) - f_s(y).$$

By the properties of indicator functions, we obtain the bound

$$|f^s(y)| \leqslant s^{-a}||y|^a f(y)|.$$

Applying Theorem 3.9(a), we get:

$$\begin{aligned}
\|h_t(\xi, x, \omega)\mathfrak{D}_g f^s\|_q &\leqslant \|\mathfrak{D}_g f^s\|_q, \quad \text{since} \quad \|h_t(\xi, x, \omega)\|_\infty \leqslant 1 \\
&\leqslant \|g\|_q\|f^s\|_p \\
&\leqslant \|g\|_q\big(s^{-a}\||y|^a f\|_p\big).
\end{aligned}$$

On the other hand, using Theorem 3.9(b) and Hölder's inequality, we have:

$$\begin{aligned}\|h_t(\xi,x,\omega)\mathfrak{D}_g f_s\|_q &\leqslant \|h_t(\xi,x,\omega)\|_q \|\mathfrak{D}_g f_s\|_\infty \\ &\leqslant \|h_t(\xi,x,\omega)\|_q \|g\|_\infty \|f_s\|_1 \\ &\leqslant \|h_t(\xi,x,\omega)\|_q \|g\|_\infty \cdot \||y|^{-a}\mathbb{1}_{B(0,s)}\|_q \cdot \||y|^a f\|_p.\end{aligned}$$

By direct computation, we find:

$$\||y|^{-a}\mathbb{1}_{B(0,s)}\|_q = \left(\frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}\right)^{1/q} =: \mathcal{C}_1.$$

Furthermore, we use the known bound:

$$\|h_t\|_q \leqslant \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t}\cdot t^{-\frac{n+1}{2q}}.$$

Setting $\mathcal{C}_2 := \left(\frac{\pi}{q}\right)^{\frac{n+1}{2q}} e^{-t}$, and choosing $s = t^{\frac{n+1}{2n}}$, we obtain:

$$\begin{aligned}\|h_t(\xi,x,\omega)\mathfrak{D}_g f\|_q &\leqslant \|h_t(\xi,x,\omega)\mathfrak{D}_g f_s\|_q + \|h_t(\xi,x,\omega)\mathfrak{D}_g f^s\|_q \\ &\leqslant \mathcal{C}_2 s^{-a}\Big(\|g\|_q + \|g\|_\infty s^{n/q}\|h_t\|_q\Big)\||y|^a f\|_p \\ &= \mathcal{C}(\|g\|_\infty + \|g\|_q) t^{-\frac{(n+1)a}{2n}}\||y|^a f\|_p,\end{aligned}$$

where $\mathcal{C} = \mathcal{C}_2\cdot\max\{1,\mathcal{C}_1\}$. □

3.6.3 Generalized Uncertainty Inequalities for DGRT

We now establish a more general uncertainty inequality associated with the DGRT.

Theorem 3.13. *Let $1 < p \leqslant 2$, $0 < a < \frac{n}{q}$, and $0 < b \leqslant 2$. Assume $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then, there exists $C' > 0$, such that for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we have*

$$\|\mathfrak{D}_g f\|_q \;\leqslant\; 2\,C'\left[\Big(\big(\|g\|_\infty + \|g\|_q\big)\,\||y|^a f\|_p\Big)^{\frac{b\,n}{a\,n+b\,n+a}}\;\Big\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\Big\|_q^{\frac{a\,n+a}{a\,n+b\,n+a}}\right], \tag{3.12}$$

where

$$C' = \max\Big\{ \Big(\frac{\pi}{q}\Big)^{\frac{n+1}{2q}}\, e^{-t},\ \Big(\frac{2\,\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2})}\Big)^{\frac{1}{q}} \Big(\frac{\pi}{q}\Big)^{\frac{n+1}{2q}}\, e^{-t},\ 1\Big\},$$

$$t = \left(\frac{(\|g\|_\infty + \|g\|_q)\, \||y|^a\, f\|_p}{\big\||(\xi,x,\omega)|^b\, \mathfrak{D}_g f\big\|_q} \right)^{\frac{2\,n}{a\,n+\,b\,n+a}}.$$

Proof. Let $1 < p \leqslant 2$ and $0 < a < \frac{n}{q}$. Assume $b \leqslant 2$. Then

$$\|(1-h_t)\,\mathfrak{D}_g(f)\|_q = t^{\frac{b}{2}}\, \big\| \big(t\,|(\xi,x,\omega)|^2\big)^{-\frac{b}{2}}\,(1-h_t)\,|(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\big\|_q. \tag{3.13}$$

Notice that $\big(1-e^{-k}\big)\,k^{-\frac{b}{2}}$ is bounded by 1 for $k>0$ and $b>0$. Hence,

$$\begin{aligned} \|\mathfrak{D}_g(f)\|_q &\leqslant \|(1-h_t)\,\mathfrak{D}_g(f)\|_q \ +\ \|h_t\,\mathfrak{D}_g(f)\|_q && (3.14)\\ &\leqslant C'\Big[\big(\|g\|_\infty + \|g\|_q\big)\, t^{-\frac{a\,(n+1)}{2\,n}}\, \||y|^a f\|_p \ +\ t^{\frac{b}{2}}\, \big\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\big\|_q \Big], && (3.15) \end{aligned}$$

where

$$C' = \max\{\,1,\,C\}, \quad C \text{ as in Lemma 3.4.}$$

We then set

$$t = \left(\frac{(\|g\|_\infty + \|g\|_q)\, \||y|^a f\|_p}{\||(\xi,x,\omega)|^b\,\mathfrak{D}_g f\|_q} \right)^{\frac{2\,n}{a\,n+b\,n+a}}, \tag{3.16}$$

which balances the contributions from $(1-h_t)$ and h_t. Substituting this choice of t yields

$$\|\mathfrak{D}_g(f)\|_q \ \leqslant\ 2\,C'\Big[\Big((\|g\|_\infty + \|g\|_q)\, \||y|^{\,a} f\|_p \Big)^{\frac{b\,n}{a\,n+b\,n+a}}\, \big\||(\xi,x,\omega)|^{\,b}\,\mathfrak{D}_g(f)\big\|_q^{\frac{a\,n+a}{a\,n+b\,n+a}} \Big]. \tag{3.17}$$

□

Corollary 3.14. *Let $1 < p \leqslant 2$, $0 < a < \frac{n}{q}$, and $b > 2$. Assume $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Let b' be the Hölder conjugate of b. Then there exists $C' > 0$ such that, for all $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$,*

the following inequality holds:

$$\|\mathfrak{D}_g f\|_q \leqslant 2^{\frac{b\left(2\,a\,n+2\,a+b'\,n\right)}{b'\left(b\,n+a\,n+a\right)}} \left(C'\right)^{\frac{b\left(a\,n+b'\,n+a\right)}{b'\left(b\,n+a\,n+a\right)}} \times \left[\left(\left(\|g\|_\infty + \|g\|_q\right)\||y|^a f\|_p\right)^{\frac{b\,n}{b\,n+a\,n+a}} \left\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\right\|_q^{\frac{a\,n+a}{b\,n+a\,n+a}}\right], \tag{3.18}$$

where C' is the same constant as in Theorem 3.13.

Proof. Let $b > 2$ and let b' be its conjugate exponent (so $b' \leqslant 2 < b$). Define

$$u \;:=\; \frac{|(\xi,x,\omega)|}{\epsilon},$$

and employ the same argument used in Corollary 3.6. We obtain

$$\begin{aligned}\||(\xi,x,\omega)|^{b'}\,\mathfrak{D}_g(f)\|_q &\leqslant \left\|\epsilon^{b'}\left[1+\tfrac{|(\xi,x,\omega)|^b}{\epsilon^b}\right]\mathfrak{D}_g(f)\right\|_q \\ &\leqslant \epsilon^{b'}\,\|\mathfrak{D}_g(f)\|_q \;+\; \epsilon^{b'-b}\left\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\right\|_q.\end{aligned}$$

Choosing

$$\epsilon = \left(\tfrac{\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\|_q}{\|\mathfrak{D}_g(f)\|_q}\right)^{\frac{1}{b}},$$

we deduce

$$\||(\xi,x,\omega)|^{b'}\,\mathfrak{D}_g(f)\|_q \;\leqslant 2\,\|\mathfrak{D}_g(f)\|_q^{\frac{b-b'}{b}}\left\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\right\|_q^{\frac{b'}{b}}.$$

By combining this with Theorem 3.13 (applied to the exponent $b' \leqslant 2$), we first get

$$\|\mathfrak{D}_g f\|_q \;\leqslant 2\,C'\left[\left(\left(\|g\|_\infty + \|g\|_q\right)\||y|^a f\|_p\right)^{\frac{b'\,n}{a\,n+b'\,n+a}} \left\||(\xi,x,\omega)|^{b'}\,\mathfrak{D}_g(f)\right\|_q^{\frac{a\,n+a}{a\,n+b'\,n+a}}\right].$$

An additional step of isolating $\|\mathfrak{D}_g(f)\|_q$ and raising it to the relevant exponent leads to

$$\|\mathfrak{D}_g f\|_q^{\frac{b'(b\,n+a\,n+a)}{b\,(b\,n+a\,n+a)}} \;\leqslant\; 2^{\frac{2\,a\,n+2\,a+b'\,n}{a\,n+b'\,n+a}}\,C'\left[\left(\left(\|g\|_\infty+\|g\|_q\right)\||y|^a f\|_p\right)^{\frac{b'\,n}{a\,n+b'\,n+a}} \left\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\right\|_q^{\frac{b'(a\,n+a)}{b(\,a\,n+b'\,n+a)}}\right],$$

which, upon simplification, establishes the final inequality in the statement:

$$\|\mathfrak{D}_g f\|_q \;\leqslant 2^{\frac{b\,(2\,a\,n+2\,a+b'\,n)}{b'\left(b\,n+a\,n+a\right)}} \left(C'\right)^{\frac{b\,(a\,n+b'\,n+a)}{b'\left(b\,n+a\,n+a\right)}} \left[\left(\left(\|g\|_\infty+\|g\|_q\right)\||y|^a f\|_p\right)^{\frac{b\,n}{b\,n+a\,n+a}} \left\||(\xi,x,\omega)|^b\,\mathfrak{D}_g(f)\right\|_q^{\frac{a\,n+a}{b\,n+a\,n+a}}\right].$$

□

Corollary 3.15. *Let a, b be two positive real numbers, and let a' denote the conjugate exponent of a. In the special case $p = q = 2$, assuming $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then there exists a positive constant C'' such that, for every $f(y) \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we have*

$$\|f\|_{L^2(\mathbb{R}^n)} \;\leqslant\; C'' \left\| |y|^a f \right\|_{L^2(\mathbb{R}^n)}^{\frac{b\,n}{a\,n+b\,n+a}} \left\| |(\xi, x, \omega)|^b \, \mathfrak{D}_g f \right\|_{L^2(\Delta)}^{\frac{a\,n+a}{a\,n+b\,n+a}}.$$

Then, the constant C'' can be expressed as

$$C'' \;=\; \begin{cases} \widetilde{\mathcal{C}}\, \dfrac{\left(\|g\|_\infty + \|g\|_2\right)^{\frac{b\,n}{a\,n+b\,n+a}}}{\|g\|_2} & \text{if } 0 < a < \frac{n}{2}, \\[2ex] \left[\, 2\left(\|g\|_\infty + \|g\|_2\right)\right]^{\frac{a\,b\,n}{a'\left(a\,n+b\,n+a\right)}} \left(\dfrac{\widetilde{\mathcal{C}}}{\|g\|_2}\right)^{\frac{a\left(a'\,n+b\,n+a'\right)}{a'\left(a\,n+b\,n+a\right)}} & \text{if } a \geqslant \frac{n}{2}, \end{cases}$$

where

$$\widetilde{\mathcal{C}} := \begin{cases} 2C', & \text{for } b \leqslant 2, \\ 2^{\frac{b(2ad+2a+b'd)}{b'(bd+ad+a)}} (C')^{\frac{b(ad+b'd+a)}{b'(bd+ad+a)}}, & \text{for } b > 2. \end{cases}$$

Proof. By Theorem 3.9 (b) and Theorem 3.13, the Plancherel-type formula yields the stated result when $0 < a < \frac{n}{2}$.

To treat the case $a \geqslant \frac{n}{2}$, we apply a similar argument as in Corollary 3.14 but substitute $u = \frac{|y|}{\epsilon}$ and optimize over $\epsilon = \left(\frac{\||y|^a f\|_2}{\|f\|_2}\right)^{\frac{1}{a}}$. This balancing step leads to the estimate

$$\left\| |y|^{a'} f \right\|_2 \;\leqslant\; 2\, \|f\|_2^{\frac{a-a'}{a}} \left\| |y|^a f \right\|_2^{\frac{a'}{a}},$$

which completes the argument for $a \geqslant \frac{n}{2}$. □

3.6.4 A Nash-Type Inequality for the Directional Gabor Ridge Transform

We now turn our attention to a Nash-type inequality in the context of the directional Gabor ridge transform. Nash-type inequalities generally link certain geometric concentration features of a function in the physical domain with its transform-domain behavior under a suitably chosen operator. In the Euclidean Fourier setting, these have been used to establish phenomena such as decay estimates, smoothing properties, and to quantify how a

function must "spread out" when subject to a transform. Here, we adapt similar ideas to the directional short-time Fourier transform scenario, more specifically to the operator $\mathfrak{D}_g$.

The central point is that if a function f is controlled in both a low-frequency (or low-index) measure—like an L^1- or L^2-norm—and in the transform domain with an appropriate weight $|(\xi, x, \omega)|^a$, then $\mathfrak{D}_g(f)$ itself must obey an interpolation-type bound. Such results are reminiscent of standard Nash inequalities that interlace the geometry of concentration in one domain with an imposed norm control in another. The following theorem accomplishes this goal by showing that $\mathfrak{D}_g(f)$ cannot be simultaneously too large in $\|\cdot\|_{L^{p'}(\Delta)}$ while preserving the prescribed integrability constraints on f and on $|(\xi, x, \omega)|^a \mathfrak{D}_g(f)$.

Theorem 3.16 (Nash-type inequality for $\mathfrak{D}_g$)**.** *Let $1 < p \leqslant 2$ and $a > 0$. Let p' be the dual of p. Assume $g \in L^2(\mathbb{R}) \cap L^\infty(\mathbb{R})$. Then for $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, we have*

$$\|\mathfrak{D}_g(f)\|_{L^{p'}(\Delta)} \;\leqslant\; \mathfrak{C}\, \|f\|_{L^1(\mathbb{R}^n)}^{\frac{2\,a}{2\,a+n+1}}\, \big\||(\xi, x, \omega)|^a\, \mathfrak{D}_g(f)\big\|_{L^{p'}(\Delta)}^{\frac{n+1}{2\,a+n+1}}. \tag{3.19}$$

Proof. We use a modification of Theorem 2.5 in [17], replacing the operator $\mathcal{D}_g$ by $\mathfrak{D}_g$. Observe that if $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $g \in L^1(\mathbb{R}) \cap L^\infty(\mathbb{R})$, then p' is the dual exponent to p. Hence, $\mathfrak{D}_g(f) \in L^{p'}(\Delta)$ and

$$\|\mathfrak{D}_g(f)\|_{p'} \;\leqslant\; \|g\|_{p'}\, \|f\|_p.$$

The idea of this Nash-type inequality is to balance how much $\mathfrak{D}_g(f)$ can concentrate, given that both f and $|(\xi, x, \omega)|^a\, \mathfrak{D}_g(f)$ are controlled in appropriate norms. One exploits local smoothing or partition-of-unity arguments on the spatial side (for f) and a corresponding argument on the transform side (for $|(\xi, x, \omega)|^a\, \mathfrak{D}_g(f)$). These techniques show that if $\mathfrak{D}_g(f)$ were too large in $L^{p'}(\Delta)$, it would force a contradiction with the assumed integrability conditions on both f and the weighted version of $\mathfrak{D}_g(f)$. Thus, the Nash-type inequality follows by carefully combining these estimates. □

3.6.5 General Uncertainty Inequality for DWGRT

In this section, we extend the uncertainty principles derived for the DGRT to the Directional Weighted Gabor Ridge Transform (DWGRT). Our goal is to establish a general uncertainty inequality for the DWGRT operator and explore its consequences.

Theorem 3.17 (Plancherel-type identity for $\mathfrak{DW}_g$)**.** *If $g \in \mathcal{S}(\mathbb{R})$ is non-zero and $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, then*

$$\big\|\mathfrak{DW}_g(f)\big\|_{L^2(\Delta)} \;=\; 2\,\|f\|_{L^2(\mathbb{R}^n)}\, \|g\|_{L^2(\mathbb{R})}.$$

Proof. First note that, for $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$ and $g \in \mathcal{S}(\mathbb{R})$, we have

$$\begin{aligned}
\langle f, G_{\xi,x,\omega}\rangle = \langle R_\xi(f), G^{x,\omega}\rangle \;&=\; \int_{\mathbb{R}} R_\xi(f)(s)\,\overline{\Big(D_{\frac{n-1}{2}}\big(g^{x,\omega}\big)\Big)(s)}\,ds \\
&= \int_{\mathbb{R}} R_\xi(f)(s)\,\overline{\Big(\widehat{g^{x,\omega}}(\sigma)\,|\sigma|^{\frac{n-1}{2}}\Big)^{\vee}(s)}\,ds \\
&= \int_{\mathbb{R}} R_\xi(f)(s)\,\overline{\Big\{\int_{\mathbb{R}} e^{2\pi i\, s\, y}\,\widehat{g}\big(y-\omega\big)\,e^{-2\pi i\, y\, x}\,|y|^{\frac{n-1}{2}}\,dy\Big\}}\,ds \\
&= \int_{\mathbb{R}} \overline{\widehat{g}(y-\omega)}\,e^{2\pi i\, y\, x}\,|y|^{\frac{n-1}{2}}\,\Big\{\int_{\mathbb{R}} R_\xi(f)(s)\,e^{-2\pi i\, s\, y}\,ds\Big\}dy \\
&= \int_{\mathbb{R}} \overline{\widehat{g}(y-\omega)}\,e^{2\pi i\, y\, x}\,|y|^{\frac{n-1}{2}}\,\widehat{R_\xi(f)}(y)\,dy \\
&= \big(\widehat{R_\xi(f)}(\sigma)\,\widehat{g}(\sigma-\omega)\,|\sigma|^{\frac{n-1}{2}}\big)^{\vee}(x) \;=\; \Big(\widehat{R_\xi(f)}(\sigma)\,\widehat{D_{\frac{n-1}{2}}(g^{0,\omega})}(\sigma)\Big)^{\vee} \\
&= \big(R_\xi(f)(x) * D_{\frac{n-1}{2}}\big(\overline{g^{0,\omega}}\big)\big)(-x) \;=\; D_{\frac{n-1}{2}}\big(R_\xi(f)(x)\big) * \overline{g^{0,\omega}}(-x).
\end{aligned}$$

Denote $g(-x)$ by $\widetilde{g}(x)$. Then

$$\begin{aligned}
&\int_{\mathbb{S}^{n-1}}\int_{\mathbb{R}}\int_{\mathbb{R}} \langle f, G_{\xi,x,\omega}\rangle\,\langle f, G_{\xi,x,\omega}\rangle\,d\omega\,dx\,d\xi \\
&= \int_{\mathbb{S}^{n-1}}\int_{\mathbb{R}}\int_{\mathbb{R}} \Big\langle D_{\frac{n-1}{2}}\big(R_\xi(f)(x)\big) * \overline{\widetilde{g}^{0,\omega}},\; D_{\frac{n-1}{2}}\big(R_\xi(f)(x)\big) * \overline{\widetilde{g}^{0,\omega}}\Big\rangle\,d\omega\,dx\,d\xi \\
&= \int_{\mathbb{S}^{n-1}}\int_{\mathbb{R}}\int_{\mathbb{R}} \big(\widehat{f}\big)^2\big(\sigma\,\xi\big)\;|\sigma|^{\,n-1}\;\Big|\,\widehat{\overline{\widetilde{g}}}\,(\sigma+\omega)\Big|^2\,d\omega\,d\sigma\,d\xi.
\end{aligned}$$

Integrating over ω and invoking the Fourier slice theorem leads to

$$\|g\|^2_{L^2(\mathbb{R})}\int_{\mathbb{S}^{n-1}}\int_{\mathbb{R}}\int_{\mathbb{R}} \big(\widehat{f}\big)^2\big(\sigma\,\xi\big)\;|\sigma|^{\,n-1}\,d\sigma\,d\xi \;=\; 2\,\|g\|^2_{L^2(\mathbb{R})}\,\|f\|^2_{L^2(\mathbb{R}^n)}.$$

Hence, $\big\|\mathfrak{DW}_g(f)\big\|^2_{L^2(\Delta)} = 4\,\|f\|^2_{L^2(\mathbb{R}^n)}\,\|g\|^2_{L^2(\mathbb{R})}$, that is, $\|\mathfrak{DW}_g(f)\|_{L^2(\Delta)} = 2\,\|f\|_{L^2(\mathbb{R}^n)}\,\|g\|_{L^2(\mathbb{R})}$. □

Theorem 3.18. *Let $\Omega \subset \Delta$ be a measurable set of finite measure. Suppose $a, b > 0$, and let a' be the conjugate exponent of a. Assume $g \in \mathcal{S}(\mathbb{R})$ and $p = q = 2$,, then there exists a positive constant $\mathfrak{C}'$ such that, for every*

$$f(y) \in L^1(\mathbb{R}^n) \,\cap\, L^2(\mathbb{R}^n) \quad \textit{and for every} \quad (\xi, x, \omega) \in \Omega,$$

the following inequality holds:

$$\|f\|_{L^2(\mathbb{R}^n)} \;\leqslant\; M\big(n,\Omega\big)\,\mathfrak{C}'\,\Big\||y|^a\,f\Big\|_{L^2(\mathbb{R}^n)}^{\frac{b\,n}{a\,n+b\,n+a}}\,\Big\||(\xi,x,\omega)|^b\,\mathfrak{DW}_g f\Big\|_{L^2(\Delta)}^{\frac{a\,n+a}{a\,n+b\,n+a}},$$

where $M\big(n,\Omega\big)$ is a positive constant. Moreover, the constant $\mathfrak{C}'$ can be expressed as

$$\mathfrak{C}' \;=\; \begin{cases} \widetilde{\mathfrak{C}}\;\dfrac{\big(\|g\|_\infty+\|g\|_2\big)^{\frac{b\,n}{a\,n+b\,n+a}}}{\|g\|_2}, & 0<a<\dfrac{n}{2}, \\[2ex] \Big[\,2\big(\|g\|_\infty+\|g\|_2\big)\Big]^{\frac{a\,b\,n}{a'\,\left(a\,n+b\,n+a\right)}}\left(\dfrac{\widetilde{\mathfrak{C}}}{\|g\|_2}\right)^{\frac{a\left(a'\,n+b\,n+a'\right)}{a'\,\left(a\,n+b\,n+a\right)}}, & a\geqslant\dfrac{n}{2}, \end{cases}$$

for some positive constant $\widetilde{\mathfrak{C}}$.

Proof. By Theorem 3.17 in the case $p=q=2$, we have

$$\|\mathfrak{DW}_g(f)\|_{L^2(\Delta)} \;=\; 2\,\|f\|_{L^2(\mathbb{R}^n)}\,\|g\|_{L^2(\mathbb{R})}.$$

From [20, Lemma 2], it follows that for all $(\xi,x,\omega)\in\Omega$, $\big|\langle f,G_{\xi,x,\omega}\rangle\big|$ is finite, implying $\|\mathfrak{DW}_g\| \leqslant M\big(n,\Omega\big)\,\|f\|_{L^1(\mathbb{R}^n)}\,\|g\|_\infty$ over Ω.

Next, applying Theorem 3.13, along with Corollary 3.14 and Corollary 3.15, relates $\|f\|_{L^2(\mathbb{R}^n)}$ to the norms $\||y|^a\,f\|_{L^2(\mathbb{R}^n)}$ and $\||(\xi,x,\omega)|^b\,\mathfrak{DW}_g f\|_{L^2(\Delta)}$, taking into account the finite measure of Ω. Collecting these pieces yields the stated inequality for $\|f\|_{L^2(\mathbb{R}^n)}$ and the piecewise definition of $\mathfrak{C}'$. □

Before proceeding to the main result, we note that general uncertainty principles for structured transforms—particularly those defined via frames—have been developed in various contexts. In particular, Li [28] presents a family of sharp uncertainty inequalities for uniform covering frames, which share essential features with the directional systems considered here. The methods introduced in that work inform our analysis of the Directional Weighted Gabor Ridge Transform (DWGRT), especially in how weighting and localization can be balanced through functional inequalities. Building on these ideas, we now establish a general uncertainty principle for the DWGRT operator.

Theorem 3.19 (General Uncertainty Inequality for $\mathfrak{DW}$)**.** *Let $\Omega\subset\Delta$ be a measurable set of finite measure. Suppose $1<p\leqslant 2$, $0<a<\frac{n}{q}$, and $b>2$, where q is the conjugate exponent of p. Let a' be the conjugate exponent of a. Then there exists a positive constant $\mathfrak{C}$*

such that, for every

$$f \in L^1(\mathbb{R}^n) \,\cap\, L^2(\mathbb{R}^n) \quad \textit{and for every} \quad (\xi, x, \omega) \in \Omega,$$

we have

$$\|\mathfrak{DW}_g f\|_{L^q(\Delta)} \;\leqslant\; \mathfrak{C}\left(\|g\|_\infty + \|g\|_q\right) \||y|^a\, f\|_{L^p(\mathbb{R}^n)}^{\frac{b\,n}{a\,n+b\,n+a}} \||(\xi,x,\omega)|^b\, \mathfrak{DW}_g f\|_{L^q(\Delta)}^{\frac{a\,n+a}{a\,n+b\,n+a}}, \tag{3.20}$$

Here b' denotes the conjugate exponent of b, and the parameter t is given by

$$t \;=\; \Big(\frac{\|g\|_\infty + \|g\|_q}{\big\||y|^a\, f\big\|_{L^p(\mathbb{R}^n)}\big\||(\xi,x,\omega)|^b\, \mathfrak{DW}_g f\big\|_{L^q(\Delta)}}\Big)^{\frac{2\,n}{a\,n+b\,n+a}}. \tag{3.21}$$

Proof. Let $1 < p \leqslant 2$ and $0 < a < \frac{n}{q}$. Assume $b > 2$. Then

$$\|(1-h_t)\,\mathfrak{DW}_g(f)\|_{L^q(\Delta)} \;=\; t^{\frac{b}{2}} \Big\|\big(t\,|(\xi,x,\omega)|^2\big)^{-\frac{b}{2}}(1-h_t)\,|(\xi,x,\omega)|^b\,\mathfrak{DW}_g(f)\Big\|_{L^q(\Delta)}.$$

Since $\left(1-e^{-k}\right)k^{-\frac{b}{2}}$ is uniformly bounded for $k>0$ and $b>0$, we deduce

$$\begin{aligned}\|\mathfrak{DW}_g(f)\|_{L^q(\Delta)} \;&\leqslant\; \|(1-h_t)\,\mathfrak{DW}_g(f)\|_{L^q(\Delta)} \;+\; \|h_t\,\mathfrak{DW}_g(f)\|_{L^q(\Delta)} \\ &\leqslant\; \mathfrak{C}\Big[\big(\|g\|_\infty + \|g\|_q\big)\,t^{-\frac{a(n+1)}{2\,n}}\,\big\||y|^a\,f\big\|_{L^p(\mathbb{R}^n)} \;+\; t^{\frac{b}{2}}\,\big\||(\xi,x,\omega)|^b\,\mathfrak{DW}_g(f)\big\|_{L^q(\Delta)}\Big],\end{aligned}$$

where $\mathfrak{C} = \max\{1, C\}$, with C taken from Lemma 3.4. Defining t via (3.21) balances the two terms optimally and leads to

$$\|\mathfrak{DW}_g(f)\|_{L^q(\Delta)} \;\leqslant\; \mathfrak{C}\Big[\big(\|g\|_\infty + \|g\|_q\big)\,\big\||y|^a\,f\big\|_{L^p(\mathbb{R}^n)}\Big]^{\frac{b\,n}{a\,n+b\,n+a}} \big\||(\xi,x,\omega)|^b\,\mathfrak{DW}_g(f)\big\|_{L^q(\Delta)}^{\frac{a\,n+a}{a\,n+b\,n+a}},$$

which yields (3.20) with the explicit form of $\mathfrak{C}$ being

$$\begin{aligned}\mathfrak{C} = \;& 2\,\max\Big\{\Big(\frac{\pi}{q}\Big)^{\frac{n+1}{2\,q}}\,e^{-t},\; \Big(\frac{2\,\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2})}\Big)^{\frac{1}{q}}\Big(\frac{\pi}{q}\Big)^{\frac{n+1}{2\,q}}\,e^{-t},\; 1\Big\} \\ & \times \max\Big\{1,\; \frac{b\,p\,\big(2\,a\,n + 2\,a + b'\,n\big)}{b'\,\big(b\,n + a\,n + a\big)}\Big\}.\end{aligned}$$

□

3.7 Summary

In this chapter we begin by revisiting the uncertainty principles for the directional short-time Fourier transform (DSTFT) as originally formulated by Mejjaoli and Omri. We introduce a natural weight on the combined space–frequency variables, define the associated heat-kernel modulation, and then give corrected, fully detailed statements and proofs of the weighted norm inequalities that govern the DSTFT's decay and localization properties.

Building on that foundation, we develop the theory of the directional Gabor ridge transform (DGRT) and its weighted variant (DWGRT). We prove their basic structural and boundedness properties—including $L^1 \to L^\infty$ estimates, exact L^2 isometries, and Plancherel identities—and establish their square-integrability.

Finally, we derive a unified family of uncertainty inequalities for both DGRT and DWGRT. Beginning with the same heat-kernel framework, we obtain interpolation-style bounds that quantify the trade-off between spatial moments of the input and phase-space moments of its transform. These general inequalities are then extended to arbitrary finite-measure subsets of the phase-space domain, yielding both global and local versions of the uncertainty principle for directional time–frequency analysis. Together, these results correct earlier gaps, clarify all hypotheses, and provide a complete, self-contained framework of directional uncertainty bounds.

Chapter 4

Learning-Based Phase Retrieval

4.1 Introduction

In our recent conference paper, coauthored by Wojciech Czaja, Canran Ji, Shashank Sule, and Matthias Wellershoff, titled "Neural Network-Based Speech Reconstruction from Undersampled STFT Magnitude Data"[12], published in the proceedings of the 2024 32nd European Signal Processing Conference (EUSIPCO 2024; Electronic ISBN 978-9-4645-9361-7; PoD ISBN 979-8-3315-1977-3), we developed a neural network–based method for reconstructing speech from undersampled STFT magnitude data. My contributions included designing and validating the network architecture, implementing and executing the end-to-end training pipeline, analyzing the reconstruction performance, and conducting downstream classification experiments to assess reconstruction quality.

Phase retrieval is the task of reconstructing a signal from magnitude-only measurements, and it is a central problem in many fields such as optics, imaging, and speech processing. In practical applications, phase information is often lost due to hardware limitations or physical constraints, rendering direct inversion impossible. One prominent application in audio processing is time-scaling and pitch-shifting, though the problem also underpins tasks in X-ray crystallography, transmission electron microscopy, radar processing, and coherent diffractive imaging.

Let $\mathbf{x} \in \mathbb{C}^L$ be the signal of interest, and consider a measurement operator $\mathcal{A}$ that yields only the magnitudes:

$$\mathbf{y} = |\mathcal{A}\,\mathbf{x}|\,.$$

The goal is to reconstruct $\mathbf{x}$ up to a global phase ambiguity—that is, recovery is only possible up to multiplication by a complex scalar of unit modulus. Since taking the absolute value discards crucial phase information, reconstruction is typically formulated as the minimization

of a loss function that measures the discrepancy between the observed and reconstructed magnitudes. In many formulations, this discrepancy is quantified via the Frobenius norm, denoted $\|\cdot\|_{\mathrm{F}}$, which measures the difference between two matrices by taking the square root of the sum of the squares of their entries.

Traditional approaches, such as the iterative Griffin–Lim algorithm [21], attempt to recover the phase by alternating between an inverse transform and enforcing the magnitude constraints. Although such methods have been widely used, they often require a good initialization and a large number of iterations to converge; in practice, they may stagnate in local minima or converge to solutions that do not match the true signal up to a global phase. Moreover, the deterministic structure of STFT measurements in audio processing frequently violates the conditions under which these methods are guaranteed to work.

Recent advances have shown that learning-based approaches can overcome many of these challenges by leveraging structural priors inherent in real-world data. Neural network models can be trained to approximate the inverse mapping from magnitude measurements to time-domain signals, thereby performing phase retrieval even in severely undersampled regimes. In our work, we demonstrate that a simple neural network model can reconstruct audio signals of length $L = 8000$ from as few as 4000 measurements—far below the classical oversampling requirements.

4.2 STFT Phase Retrieval

Phase retrieval refers to the process of reconstructing a complex-valued signal from its pointwise absolute values. This problem arises in many domains, including speech processing, where one often modifies the magnitudes of the short-time Fourier transform (STFT) of an audio signal and then recovers the corresponding phase information. Prominent applications include time-scaling and pitch-shifting audio signals [21], as well as tasks in radar processing, X-ray crystallography, coherent diffractive imaging, and electron microscopy.

Formally, suppose we are given magnitude-only data associated with a complex-valued signal $\mathbf{x} \in \mathbb{C}^L$. Because the absolute value operation discards crucial phase information, even when one allows for a global phase ambiguity (i.e., $\mathbf{x}$ can only be recovered up to a multiplicative factor $\tau \in \mathbb{C}$ with $|\tau| = 1$), reconstruction remains challenging. Classical results indicate that, to guarantee unique recovery (up to global phase) for real-valued signals in $\mathbb{R}^L$, strictly more than $2L-2$ measurements are typically required [5]. For complex-valued signals in $\mathbb{C}^L$, the requirement increases to strictly more than $4L-2\log_2(L)-4$ measurements [23]. Despite these oversampling conditions, practical phase retrieval is notoriously sensitive to noise and may lack numerical stability even when uniqueness is theoretically ensured [7].

In audio applications, the STFT is a central tool. For a given hop size $a \in \mathbb{N}$ and number of frequency bins $M \in \mathbb{N}$, and for a window function $\mathbf{w} \in \mathbb{C}^L$ (typically supported on a subset of $\{0, 1, \ldots, L-1\}$ with $\mathbf{w}_\ell = 0$ for $\ell \geqslant M$), the discrete STFT of a signal $\mathbf{x} \in \mathbb{C}^L$ is defined as

$$\mathcal{V}_{\mathbf{w}}\mathbf{x}(m,n) \;=\; \mathcal{V}_{\mathbf{w}}^{a,M}\mathbf{x}(m,n) \;=\; \sum_{\ell=0}^{L-1} \mathbf{x}_{\ell+a\,n}\,\overline{\mathbf{w}_\ell}\,\exp\Big(-2\pi\mathrm{i}\,\frac{\ell\,m}{M}\Big),$$

for $m \in [M] := \{0, 1, \ldots, M-1\}$ and $n \in [N] := \{0, 1, \ldots, N-1\}$, where $N = \lfloor L/a \rfloor + 1$. When both $\mathbf{x}$ and $\mathbf{w}$ are real-valued, conjugate symmetry of the Fourier transform permits recording only $\lfloor M/2 \rfloor + 1$ frequency bins. The STFT can be exactly inverted under the nonzero overlap-add (NOLA) condition, which requires that for every $\ell \in [L]$ there exists an $n \in [N]$ such that $\mathbf{w}_{\ell - a\,n} \neq 0$.

In the STFT phase retrieval problem, one observes only the magnitude of the STFT,

$$\mathcal{M}(\mathbf{x})(m,n) \;=\; \Big|\mathcal{V}_{\mathbf{w}}^{a,M}\mathbf{x}(m,n)\Big|,$$

for $m \in [M]$ (or the reduced set in the real-valued case) and $n \in [N]$. The goal is then to recover a signal $\mathbf{x} \in \mathbb{C}^L$ (or $\mathbf{x} \in \mathbb{R}^L$ for real-valued audio) such that

$$\mathcal{M}(\mathbf{x}) \approx \mathcal{M}(\mathbf{x}_*),$$

where $\mathbf{x}_*$ is the original signal. Since $\mathcal{M}(\mathbf{x}) = \mathcal{M}(\tau\mathbf{x})$ for any $\tau \in \mathbb{C}$ with $|\tau| = 1$, one can only hope to recover $\mathbf{x}_*$ up to a global phase; in the real-valued case, this ambiguity reduces to a global sign.

Classical algorithms such as the iterative Griffin–Lim (GL) method [21] alternate between enforcing the magnitude constraints in the frequency domain and applying the inverse STFT in the time domain. Although GL is simple and widely used, it often suffers from slow convergence and may become trapped in local minima, particularly when the number of measurements is insufficient. Other methods, including Wirtinger Flow (WF) [8] and Amplitude Flow (AF) [44], approach phase retrieval as a non-convex optimization problem by minimizing a loss function that quantifies the discrepancy between the measured magnitudes and those computed from a candidate signal. However, these methods typically require a measurement count on the order of $3L$ or $4L$ to guarantee reliable convergence—an impractical condition when the available data is undersampled.

Recent advances employ deep learning techniques to overcome these limitations. Neural network approaches learn a direct or approximate inverse mapping from STFT magnitudes to time-domain signals by exploiting structural priors inherent in real-world datasets, such as human speech. For example, methods like Deep Griffin–Lim Iteration integrate denoising

networks into iterative schemes, while multi-head convolutional architectures [2] are designed to handle high-dimensional STFT inputs. Although these learning-based techniques often require large training datasets, careful architectural design, and significant computational resources, they can successfully perform phase retrieval in severely undersampled settings where classical methods either fail or yield suboptimal solutions.

4.3 Limitations of Classical Phase Retrieval Algorithms

Classical phase retrieval algorithms rely on an assumption of sufficient redundancy in the data so that the signal of interest can be reconstructed (up to a global phase or sign). In many practical applications, however, the number of magnitude measurements is not large enough to satisfy these theoretical requirements, causing the algorithms to function unreliably. Moreover, phase retrieval is inherently non-convex, so even with careful initialization, iterative methods can become trapped in local minima.

A commonly used reconstruction framework minimizes a loss function that enforces consistency between the observed magnitudes and those derived from a candidate signal. A representative example is

$$\ell_{\mathrm{PR}}(\mathbf{x}) \;=\; \sum_{m,n} \Big(\mathcal{M}(\mathbf{x}_*)(m,n) - \mathcal{M}(\mathbf{x})(m,n) \Big)^2,$$

where $\mathbf{x} \in \mathbb{C}^L$ is the unknown signal to be recovered, $\mathbf{x}_*$ is the true signal, and $\mathcal{M}(\cdot)$ denotes the STFT magnitude operator. Although minimizing this loss can, in principle, recover the signal if sufficiently many measurements are available, several drawbacks arise in practice.

One issue is that iterative algorithms for minimizing ℓ_{PR} generally require an initial phase guess. Poor initialization—whether random or based on simple heuristics—often leads to slow convergence or convergence to suboptimal solutions, as the algorithm must navigate a high-dimensional, non-convex landscape.

Another limitation is the sensitivity of classical algorithms to specific conditions on the STFT. For instance, when the nonzero overlap-add (NOLA) condition is violated, the standard inverse STFT formulas break down. Techniques that iteratively refine the phase estimate and then enforce magnitude constraints become inapplicable in such undersampled scenarios, rendering certain choices of hop size or window function ineffective regardless of iteration count or parameter tuning.

An additional difficulty stems from the dependence on assumptions of random measurements or heavy oversampling. Many convergence guarantees assume that the magnitude measurements follow a specific random distribution, allowing the problem to be treated as a

generic quadratic system. In contrast, real STFT measurements are highly structured and deterministic—especially when standard windows (e.g., the Hann window) and fixed hop sizes are used. In settings with limited data, such oversimplified assumptions fail to capture the true complexity of the signal's phase, leading to recovery failure or poor accuracy.

Computational efficiency poses yet another drawback. Methods that directly minimize ℓ_{PR} via gradient-based updates or that involve repeated inverse transforms can be prohibitively slow for large signals. In typical STFT phase retrieval, an inverse transform is computed after each iteration to enforce the magnitude constraint. When thousands of iterations are required, the computational cost becomes impractical for real-time applications or high-dimensional signals.

Finally, even if these methods converge to a solution that locally minimizes the loss, there is no guarantee that the recovered signal agrees with the true one (up to a global phase or sign). Particularly when the number of measurements falls below the theoretical threshold, the STFT magnitudes may not uniquely determine the original signal. In such cases, classical algorithms may converge to spurious critical points corresponding to entirely different signals that share the same magnitude data.

4.3.1 Griffin–Lim Algorithm

The Griffin–Lim (GL) algorithm is one of the most widely used iterative techniques for phase retrieval in audio signal processing. It relies on alternating projections to iteratively estimate the missing phase information. Given an initial phase estimate, the algorithm iterates between:

1. Computing the inverse STFT (ISTFT) using the current phase estimate.
2. Replacing the magnitude of the transformed signal with the observed magnitude while retaining the estimated phase.
3. Reapplying the STFT to obtain a refined phase estimate.

Despite its simplicity, GL has several shortcomings:

- **Suboptimal Convergence:** The algorithm may stall in local minima and does not guarantee convergence to the true phase.
- **Slow Execution:** A large number of iterations are typically required, making it computationally expensive.
- **Dependence on Initialization:** The quality of the retrieved phase is highly sensitive to the initial phase estimate.

4.3.2 Optimization-Based Methods

Optimization-based approaches such as Wirtinger Flow (WF) and Amplitude Flow (AF) recast phase retrieval as a non-convex optimization problem by minimizing a loss function like ℓ_{PR} defined above.

Wirtinger Flow

Wirtinger Flow (WF) uses gradient descent with Wirtinger derivatives to iteratively update the signal estimate. While WF offers theoretical guarantees under certain random measurement models, it suffers from high computational complexity, sensitivity to noise, and a heavy dependence on proper spectral initialization.

Amplitude Flow

Amplitude Flow (AF) modifies the WF approach by directly estimating the amplitude-scaled phase through alternating minimization. AF comes with a theoretical performance guarantee which certifies that its iterates converge to the desired signal $\mathbf{x}_* \in \mathbb{R}^L$ up to a global sign, provided that the measurements $\mathcal{M}_{\mathrm{G}}(\mathbf{x}_*)$ are generated from a Gaussian model—that is, measurements of the form $\mathcal{M}_{\mathrm{G}}(\mathbf{x}_*)(j) := \left|\mathbf{a}_j^\top \mathbf{x}_*\right|$ with $\mathbf{a}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ independently for $j \in [J]$ and $J \gtrsim L$. Despite this guarantee, AF's convergence is limited to idealized measurement models, and in practice, the deterministic nature of real STFT measurements often invalidates these assumptions. Furthermore, AF may require a number of measurements on the order of $3L$ or $4L$ for reliable convergence, a condition that is difficult to meet in undersampled scenarios.

4.3.3 Machine Learning Approaches

The limitations of classical algorithms have motivated recent work on learning-based phase retrieval methods. One early method [22] attempts to reconstruct signals by minimizing a loss function defined in the range of a generative model for the dataset. Although this approach enables recovery from very few phaseless measurements, it requires the existence of a suitable generative model and is ultimately limited by its representational error.

An alternative strategy is presented in [30], termed Deep Griffin–Lim Iteration, where a convolutional neural network is trained as a denoiser and integrated into every iteration of the GL algorithm. This hybrid approach can outperform standard GL; however, it still fails when the NOLA condition is violated since an invertible STFT is required. Additionally, its performance in settings with limited data remains unclear.

Finally, the work in [2] adopts a multi-head convolutional neural network trained on the Librispeech dataset [35] for phase retrieval. In comparison, the loss function, network architecture, and dataset considered in our approach are considerably simpler. Notably, the multi-head convolutional neural network in [2] is not applied in limited data scenarios—in fact, their setup employs roughly $8L$ datapoints—whereas our method is designed to perform phase retrieval from as few as $\frac{1}{16}$ of the measurements required by classical methods.

4.4 Neural Network-Based Phase Retrieval

A data-driven framework for phase retrieval can overcome many of the challenges that arise in classical methods. The essential idea is to approximate the inverse mapping from STFT magnitudes to time-domain signals by training a neural network on a representative dataset. Consider audio signals of length L, and let $\mathcal{X} \subset \mathbb{R}^L$ be a collection of such signals. Given signals $\mathbf{x}$, $\mathbf{y} \in \mathbb{R}^L$, we define the loss function

$$L_{\mathrm{PR}}(\mathbf{x}, \mathbf{y}) \;=\; \left\| \mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{x}) \;-\; \mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{y}) \right\|_{\mathrm{F}},$$

where a' and M' (placed as superscripts) denote the hop size and the number of frequency bins used in computing the loss, and $\mathbf{w}'$ (bolded and placed as a subscript) is the window used in the loss function. These parameters need not match those used in obtaining the undersampled measurements. In particular, one can choose a larger M' and a smaller hop size a' to guarantee that the only global minimizers of the mapping $\mathbf{x} \mapsto L_{\mathrm{PR}}(\mathbf{x}, \mathbf{y})$ are $\mathbf{y}$ and $-\mathbf{y}$. Under these circumstances, this loss penalizes differences between the STFT magnitudes of two signals, steering the training process toward solutions that match the true signal's short-time spectral content up to a global sign.

In our framework, the neural network [12] is trained via empirical risk minimization on a dataset drawn from $\mathcal{X}$. The input to the network consists of the undersampled STFT magnitudes computed using an undersampling scheme with parameters $a \in \{32, 64, 128\}$ and $M \in \{128, 256\}$. In practice, each audio signal is first downsampled or zero-padded so that all signals have length $L = 8000$, then normalized by subtracting the mean and dividing by the standard deviation. This setup yields scenarios ranging from approximately four thousand measurements up to about thirty-two thousand measurements, with the parameter choices arranged so that the STFT may or may not be theoretically invertible.

In all four measurement settings, we train a simple fully connected feedforward neural network with approximately one billion parameters. Our networks have three linear layers and use SELU activation functions [26]. More precisely, the input data is first flattened

and then fed into a linear layer with SELU activation and hidden layer dimension $d_{\mathrm{h}} \in \mathbb{N}$. This is followed by a second linear layer with SELU activation (using the same hidden layer dimension), and finally, the data is passed through a third linear layer without an activation function. In the four setups described above, the hidden layer dimension d_{h} is set to 27 000, 25 000, 22 000, and 18 000, respectively. In this way, a single neural network has a size of about four GB, meaning it comfortably fits within the RAM of most current laptops. In training, we use an initial learning rate of $2.5 \cdot 10^{-5}$ and the Adam optimizer for stochastic optimization. We apply a simple learning rate scheduling method: training continues until a minimum in the average loss over the test set is reached, at which point the learning rate is halved; this process is repeated iteratively until the learning rate is on the order of single machine precision (approximately 10^{-7}). The neural networks for the four settings typically train for 20–30 epochs before this procedure terminates.

A typical neural network architecture for phase retrieval, as described above, approximates the mapping from STFT magnitudes to a reconstructed audio signal. Although there is no guarantee that the network will satisfy classical uniqueness or stability properties, numerical experiments confirm that it can recover signals from significantly fewer measurements than classical theory would suggest. This success is attributed to the network's ability to leverage structural priors learned from the dataset, thereby resolving ambiguities where standard phase retrieval methods fail. A further benefit is efficiency in inference: once trained, the network reconstructs each signal in a single forward pass, vastly outperforming iterative approaches that require thousands of iterations and repeated inverse transforms.

A standard metric to evaluate performance is the spectral convergence (SC) between a reconstructed signal $\mathbf{x}$ and a reference signal $\mathbf{y}$:

$$\mathrm{SC}(\mathbf{x}, \mathbf{y}) \;=\; 20 \, \log_{10} \left(\frac{\left\| \mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{x}) - \mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{y}) \right\|_{\mathrm{F}}}{\left\| \mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{y}) \right\|_{\mathrm{F}}} \right). \tag{4.1}$$

Lower (i.e., more negative) values of $\mathrm{SC}(\mathbf{x}, \mathbf{y})$ indicate a closer match in the STFT magnitudes between the reconstructed signal and the reference. Numerical experiments demonstrate that neural networks with large hidden layers and carefully tuned hyperparameters achieve strong reconstruction fidelity, with per-signal reconstruction times drastically lower than those required by classical iterative methods. Even in highly undersampled regimes, the neural model can exploit the underlying structure of speech signals to recover missing phase information, thereby surpassing the theoretical limitations imposed by classical phase retrieval analysis.

4.5 Experimental Results

In this section, we present detailed experimental results evaluating the performance of our neural network-based phase retrieval method for audio signal reconstruction. Our experiments are conducted on the AudioMNIST dataset, which comprises 30,000 spoken digit samples from 60 speakers. All audio signals are preprocessed by downsampling from 48 kHz to 8 kHz, zero-padding or truncating them to a fixed length of $L = 8000$ (corresponding to 1 s), and normalizing by subtracting the mean and dividing by the standard deviation.

4.5.1 Measurement Settings

We investigate four distinct experimental regimes by varying the short-time Fourier transform (STFT) parameters, namely the window length M and the temporal hop size a. These choices yield different numbers of phaseless STFT magnitude measurements:

1. **Setting 1:** $a = M = 128$, resulting in 4 095 measurements. In this case, the underlying STFT is non-invertible since the nonzero overlap-add (NOLA) condition is violated.
2. **Setting 2:** $a = 64$, $M = 128$, yielding 8 190 measurements. Here, the STFT is invertible, yet phase retrieval remains theoretically impossible even for real-valued signals.
3. **Setting 3:** $a = 64$, $M = 256$, resulting in 16 254 measurements. In this regime, phase retrieval is theoretically impossible for complex-valued signals.
4. **Setting 4:** $a = 32$, $M = 256$, yielding 32 379 measurements, a setting where no theoretical barriers exist for phase retrieval.

4.5.2 Reconstruction Performance

Our neural network (NN) is trained separately in each setting using a simple fully connected feedforward architecture with three linear layers and SELU activation functions. The network sizes (determined by the hidden layer dimensions) are chosen to balance computational feasibility and representational capacity. Performance is quantified by the *spectral convergence* (SC) metric (see Eq. 4.1), which measures the relative difference (in dB) between the STFT magnitudes of the original signal $\mathbf{x}$ and its reconstruction $\mathbf{y}$.

Setting 1 (4 095 Measurements): Despite the non-invertibility of the STFT in this setting, the NN successfully reconstructs the audio signals, achieving an average SC of approximately **-8.55 dB** (with a standard deviation of 1.26 dB). Notably, classical methods

such as Fast Griffin–Lim (FGL) [37] are not applicable here, since they require an invertible STFT.

Setting 2 (8 190 Measurements): In this regime, although the STFT is invertible, theoretical guarantees for phase retrieval fail to hold. The NN achieves an average SC of about **-8.89 dB** (std. dev. 1.28 dB), slightly outperforming FGL, which obtains **-8.49 dB** (std. dev. 3.30 dB) after 1 024 iterations (see Table 4.1). This demonstrates that the NN can exploit underlying signal structure to recover phase information even in challenging settings.

Setting 3 (16 254 Measurements): With 16 254 measurements, the NN achieves an average SC of approximately **-10.39 dB** (std. dev. 1.32 dB). In this regime, classical iterative methods such as FGL achieve a markedly better performance, reaching an SC of **-27.03 dB** (std. dev. 6.09 dB).

Setting 4 (32 379 Measurements): When 32 379 measurements are available, FGL further improves its performance to an average SC of **-30.52 dB** (std. dev. 6.64 dB), whereas the NN records an average SC of **-10.55 dB** (std. dev. 1.29 dB). Additionally, comparisons with Amplitude Flow (AF) and Wirtinger Flow (WF) reveal that both AF and WF fail in the lower-measurement regimes but, in the high-measurement setting, AF outperforms FGL marginally while WF lags behind (see Table 4.1).

4.5.3 Computational Efficiency

An outstanding advantage of our NN-based approach is its computational efficiency. On an NVIDIA A100 80GB PCIe GPU, the NN processes the 6 000-signal test set in approximately 1.4 seconds, corresponding to about 230 ns per signal. In contrast, applying 1 024 iterations of FGL to the same test set takes around 35 minutes and 8 seconds (approximately 350 µs per signal). Consequently, the NN is over 1 000 times faster than FGL, with AF and WF being even slower. This remarkable efficiency renders the NN method highly attractive for real-time or large-scale applications.

4.5.4 Extensions via Postprocessing

To further improve reconstruction quality, we also investigate two postprocessing strategies:

1. **Postprocessed Neural Network (PNN):** Here, the NN output is refined via an additional processing step.

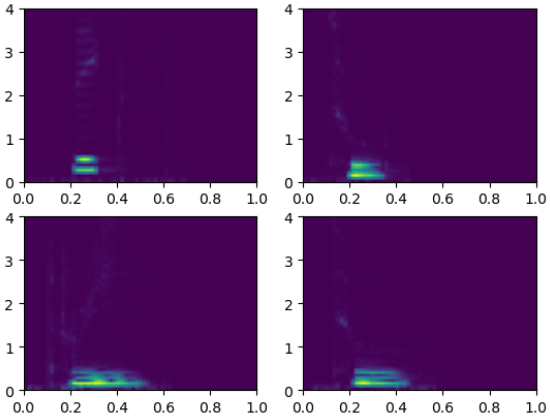
2. **Neural Network Initialized FGL Iteration (NNIFGL):** In this approach, the NN reconstruction serves as an initial guess for a limited number of FGL iterations.

In the 8 190-measurement setting, both methods yield improved spectral convergence scores compared to the base NN. As reported in Table 4.2, NNIFGL achieves an SC of **-9.74 dB** (std. dev. 3.16 dB), while the PNN attains **-9.39 dB** (std. dev. 1.29 dB). These enhancements suggest that incorporating even a modest number of iterative refinements or postprocessing steps can further leverage the strengths of the NN reconstruction.

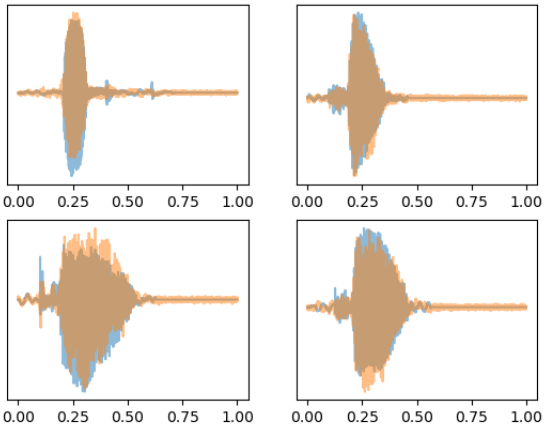
4.5.5 Qualitative Results

Table 4.1 summarizes the performance of our neural network (NN) approach compared to three classical methods: Amplitude Flow (AF), Wirtinger Flow (WF), and Fast Griffin–Lim (FGL). The table lists mean spectral convergence (SC) values in decibels (dB) across four different measurement scenarios: 4 095, 8 190, 16 254, and 32 379 STFT magnitude measurements. Lower (more negative) SC scores indicate more accurate reconstructions.

4 095 Measurements (Severely Undersampled). This regime corresponds to $a = M = 128$, where the STFT is non-invertible. Classical methods either fail outright (FGL cannot be applied because the nonzero overlap-add condition is violated) or yield poor reconstructions (AF and WF produce SC scores near -0.2 dB). By contrast, our NN achieves a mean SC of $-\mathbf{8.55}$ dB (std. dev. 1.26 dB), demonstrating that it can recover perceptually meaningful signals even when conventional phase retrieval algorithms do not work at all.



(a) The 4 095 phaseless measurements obtained from the STFT magnitude with Hann window and $a = M = 128$.



(b) The original audio signals (in blue) and their neural network reconstructions (in orange).

Figure 4.1: The neural network recovers signals of length 8 000 from about half as many measurements. The horizontal axes describe time in seconds. The vertical axes, in Subfigure 4.1a, describe frequency in kHz.

Subfigure 1(a) in Figure 4.1 shows the phaseless STFT magnitude in this undersampled scenario, while Subfigure 1(b) compares the original audio waveform (blue) with the NN reconstruction (orange). Despite the drastic reduction in measurements, the NN captures key audio features, validating its ability to leverage the structural priors learned from the dataset.

8 190 Measurements (STFT Invertible, but Theoretically Impossible). With $a = 64$ and $M = 128$, the STFT becomes invertible, yet classical theory suggests real signals cannot be uniquely recovered up to phase. Indeed, AF and WF remain far from matching the target magnitudes, while FGL achieves -8.49 dB and the NN slightly outperforms it

at −8.89 dB. This indicates that, even in a setting where phase retrieval is "impossible" in the strict theoretical sense, the NN can exploit data-driven priors to produce high-fidelity reconstructions.

16 254 and 32 379 Measurements (Increasing Redundancy). In the more data-rich regimes of 16 254 and 32 379 measurements, classical methods begin to excel. At 16 254 measurements, FGL attains −27.03 dB, significantly outperforming the NN's −10.39 dB. Similarly, at 32 379 measurements, both FGL (−30.52 dB) and AF (−32.00 dB) exceed the NN's −10.55 dB result. These results confirm that classical approaches can yield impressive accuracy when enough redundancy is present. Nonetheless, as discussed in earlier sections, the NN remains far more computationally efficient and robustly handles undersampled conditions (particularly the 4 095-measurement regime) where classical phase retrieval fails or is inapplicable.

Table 4.1: Benchmarking phase retrieval algorithms: Amplitude Flow (AF), Wirtinger Flow (WF) and Fast Griffin Lim (FGL) versus our approach (NN). All metrics are in dB and represent the mean over the test set. The standard deviation is recorded in brackets.

#	NN	FGL
4 095	**−8.55** (1.26)	na (na)
8 190	**−8.89** (1.28)	−8.49 (3.30)
16 254	−10.39 (1.32)	**−27.03** (6.09)
32 379	−10.55 (1.29)	−30.52 (6.64)
#	AF	WF
4 095	−0.21 (0.44)	−0.16 (0.36)
8 190	−0.26 (0.60)	−0.23 (0.55)
16 254	0.20 (0.78)	−0.12 (0.29)
32 379	**−32.00** (4.60)	−8.87 (2.46)

Overall, Figure 4.1 and Table 4.1 collectively show that our NN method not only competes well with classical algorithms in moderate-data regimes but also uniquely succeeds in severely undersampled scenarios. The NN thus offers a compelling solution for phase retrieval in challenging audio reconstruction tasks where conventional methods struggle or are entirely inoperative.

Table 4.2: The performance of the postprocessed neural network (PNN) and neural network initialised FGL iteration (NNIFGL) in the setting with 8 190 measurements.

	PNN	NNIFGL
SC	−9.39 (1.29)	**−9.74** (3.16)

All the three approaches outlined above are promising: in particular, both the postprocessed neural network and the neural network initialised FGL iteration achieve better spectral convergence scores than our neural network in the setting with 8 190 measurements 4.2. However, all the three approaches also suffer from a common drawback: they are not applicable when the NOLA condition is violated because a STFT needs to be inverted.

4.5.6 Final remarks

On different loss functions and larger networks

Apart from training on the loss function L_{PR}, one might try to train on the related loss functions

$$\mathbb{R}^2 \ni (\mathbf{x}, \mathbf{y}) \mapsto \left\| \left(\mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{x}) \right)^p - \left(\mathcal{M}_{\mathbf{w}'}^{a',M'}(\mathbf{y}) \right)^p \right\|_{\mathrm{F}} \in [0, \infty),$$

where $p \in [1, \infty)$. When $p = 1$, we recover L_{PR}; when $p = 2$, we recover the loss function used in Wirtinger Flow [8]. Similarly, one might train on a loss function that is inspired by the PhaseMax method [18]. We have tried both of these things and were not successful in outperforming the neural network trained based on L_{PR}.

Similarly, one might be tempted to train wider and deeper networks to improve the performance of the neural network models for phase retrieval. Neither of these led us to a decrease in the average spectral convergence, however. Of course, this raises the question whether a smaller neural network could achieve the same performance while being much more efficient. We leave this question to future research.

On training times

The training of our networks took approximately 10 h 4 min (32 379), 7 h 46 min (16 254), 10 h 0 min (8 190) and 5 h 33 min (4 095). The large differences are likely explained by other (training) processes that were run on the same GPU at the same time. Learning curves for the training can be found here: https://umd.box.com/s/5gey8m0r98o9ycll8or23lvecl5i0wiu.

4.6 Classification Results

We evaluate the effectiveness of our phase-retrieval reconstructions in a downstream digit-classification task using a two-dimensional convolutional neural network originally developed by Christian Lillelund on Kaggle.[1] This network operates on single-channel spectrogram inputs of size $(1 \times H \times W)$ and consists of four convolutional blocks with feature-map widths $\{8, 16, 32, 64\}$. Each block applies a $k \times k$ convolution (first block: 5×5, subsequent blocks: 3×3), stride 2, and padding chosen to preserve spatial alignment, followed by ReLU activation and batch-normalization. Convolutional weights are initialized with Kaiming normal (Leaky ReLU parameter $a = 0.1$); biases are zeroed. After the final conv-block, an adaptive average-pool reduces to a 64-dimensional vector, which is fed to a linear layer producing logits for the 10 digit classes.

In the original audio condition, raw waveform segments are converted directly to spectrograms (via a GPU-accelerated SpectrogramCUDA routine) and passed to the classifier. In each phase retrieval condition with $M \in \{4\,095, 8\,190, 16\,254, 32\,379\}$ measurements, the measurement vectors are first input to our pre-trained reconstruction network (run in evaluation mode) to produce approximate waveforms; those are then spectrally transformed before classification.

Training uses the Adam optimizer (learning rate 0.001), cross-entropy loss, and 30 epochs per trial, with all operations carried out on the GPU. For each condition we conducted 10 independent trials (reinitializing both reconstructer and classifier), and report in Table 4.3 the mean and variance of the final test accuracy (%).

Table 4.3: Mean (variance) of classification accuracy (%) over 10 runs for original audio and phase-retrieved reconstructions via NN and FGL methods, across different numbers of STFT magnitude measurements.

Method	Original	4 095 meas.	8 190 meas.	16 254 meas.	32 379 meas.
NN	–	88.49 (0.16)	80.02 (0.55)	94.39 (0.07)	98.41 (0.01)
FGL	–	n/a (n/a)	99.45 (0.13)	99.47 (0.03)	99.38 (0.05)
Original Audio	99.41 (0.01)	–	–	–	–

The table above reports classification accuracies for reconstructions obtained from NN and FGL methods using increasing numbers of STFT magnitude measurements, along with the original audio as a reference. Among the reconstruction methods, FGL maintains a consistently high accuracy above 99.3% for all applicable measurement levels. In contrast, NN reconstructions show lower accuracy, particularly in 8,190 measurements, where the

[1] https://www.kaggle.com/code/christianlillelund/classify-mnist-audio-using-spectrograms-keras-cnn

performance drops to 80.02%. However, NN accuracy improves with more measurements, reaching 98.41% at 32,379 measurements. FGL is not applicable for 4,095 measurements, where NN achieves 88.49%. These results suggest that, while the performance of the NN is measurement dependent, FGL yields more classifier-consistent reconstructions at moderate and high measurement levels.

4.6.1 Analysis of Measurement-Dependent Accuracy Anomalies

Table 4.4 reports the per-class test accuracies (%; correct/total in parentheses) for the 4,095 and 8,190 STFT-magnitude measurement conditions, each after 30 epochs of training.

Table 4.4: Per-class accuracy (%) for 4,095 and 8,190 measurement cases after 30 epochs.

Class	4,095 meas.	8,190 meas.
0	94.31 (580/615)	92.52 (569/615)
1	76.46 (458/599)	54.42 (326/599)
2	81.54 (499/612)	76.96 (471/612)
3	85.42 (533/624)	78.69 (491/624)
4	95.96 (594/619)	93.54 (579/619)
5	100.00 (584/584)	100.00 (584/584)
6	94.19 (551/585)	87.35 (511/585)
7	91.99 (563/612)	92.32 (565/612)
8	74.36 (435/585)	51.97 (304/585)
9	83.72 (473/565)	70.97 (401/565)

In the 8,190-measurement condition, overall accuracy drops to 80.02%, driven primarily by steep declines in classes 1 and 8 (from 76.46% to 54.42% and from 74.36% to 51.97%, respectively). Other digits also exhibit moderate decreases relative to the 4,095 case, except class 7 (stable) and class 5 (remains at 100%).

Theory 1: Spectral Sampling Resonance. At 8,190 measurements the chosen sampling density may align poorly with key spectral bands that distinguish digits 1 and 8. Class 1 relies on sharp, high-frequency transients, and class 8 on dual formant peaks; if the intermediate sampling omits or attenuates these components, classification suffers. At lower (4,095) the coarse grid captures only broad low-frequency energy, and at higher (16,254) the grid is dense enough to recover all relevant frequencies, avoiding the mid-range "blind spot" at 8,190.

Theory 2: Time-Frequency Resolution Mismatch. The intermediate measurement count yields a time-frequency tiling that is neither coarse enough to emphasize long-term

temporal envelopes nor fine enough to capture rapid spectral transitions. Digits "one" depend on brief plosive onsets (high time resolution), while "eight" depends on fast formant sweeps (high frequency resolution); this resolution gap degrades both types of cues simultaneously. Coarser sampling (4,095) preserves envelope shape, and denser sampling (16,254) restores full spectral detail, but the 8,190 case falls into a resolution mismatch.

A comprehensive investigation into the underlying causes of this non-monotonic accuracy behavior is not feasible due to personal time constraints and will be undertaken in future work.

Chapter 5

Comparative Analysis in Pre-image Algorithms of Kernel PCA

5.1 Introduction

Kernel Principal Component Analysis (kPCA) extends linear PCA by mapping data into a high-dimensional feature space via Mercer kernels. Recovering an approximation of the original data point from its low-dimensional kPCA embedding, known as the pre-image problem, is fundamental for applications such as image denoising. Classical deterministic approaches—including fixed-point iteration, kernel ridge regression, multidimensional scaling inversion, direct approximation, and Nyström-hybrid schemes—have been developed to address this problem, yet their relative strengths and limitations under varying noise and dimensionality conditions remain unclear.

In this chapter I first implement a selection of representative deterministic pre-image algorithms and conduct a systematic quantitative comparison across multiple image datasets and evaluation metrics. By analyzing reconstruction quality under clean and corrupted inputs, I determine which traditional method offers the best overall performance.

Building on these insights, I then present my original contribution: two neural network-based inverse solvers, DCGAN-KPCAnet and WGAN-KPCAnet. These models leverage adversarial training to learn the inverse kPCA mapping directly from data. Through extensive experiments, I demonstrate that WGAN-KPCAnet, in particular, outperforms the best deterministic algorithm, yielding more faithful reconstructions and greater robustness to noise.

The remainder of this chapter is organized as follows. I begin by describing the architectures and training procedures of DCGAN-KPCAnet and WGAN-KPCAnet. Next, I present the results of the comparative evaluation against classical algorithms. Finally, I discuss the

implications of these findings for algorithm selection in high-noise imaging scenarios and outline directions for future work.

5.2 Mathematical Background

5.2.1 Mathematical Formulation of kPCA

Image denoising is a fundamental task in image processing and computer vision, aiming to suppress noise while preserving meaningful structure. Kernel Principal Component Analysis (kPCA) extends linear PCA by mapping each data vector $\mathbf{I}_i \in \mathbb{R}^d$ into a high-dimensional feature space via an implicit mapping ϕ, with inner products computed by a Mercer kernel $k(\mathbf{I}_i, \mathbf{I}_j)$.

Common kernel choices include:

Linear (equivalent to PCA)

$$k(x, y) = x^\top y$$

Polynomial

$$k(x, y) = (a\, x^\top y + c)^p$$

Gaussian (RBF)

$$k(x, y) = \exp\left(-\|x - y\|^2/(2\sigma^2)\right)$$

Laplacian

$$k(x, y) = \exp\left(-\|x - y\|/\sigma\right)$$

Sigmoid

$$k(x, y) = \tanh(a\, x^\top y + c)$$

Cosine similarity

$$k(x, y) = \frac{x^\top y}{\|x\|\ \|y\|}$$

Given N data points, one constructs the $N \times N$ Gram matrix K with entries

$$K_{ij} = k(\mathbf{I}_i, \mathbf{I}_j),$$

centers K in feature space, and solves the eigenproblem

$$K_c\, v_i = \lambda_i\, v_i.$$

Retaining the top m eigenvectors yields a reduced representation in $\mathbb{R}^m$, where m is cho-

sen to balance noise reduction against detail preservation. The pre-image problem then asks how to reconstruct an original-domain vector from its low-dimensional feature coordinates. A variety of numerical pre-image algorithms—ranging from iterative optimization and fixed-point schemes to regression-based approaches—have been developed, each trading off reconstruction fidelity, computational cost, and robustness to noise.

5.2.2 Kernel Methods and Feature Spaces

Kernel methods rely on implicitly mapping data from the original input space $\mathbb{R}^d$ into a (possibly infinite-dimensional) feature space $\mathcal{H}$ via a nonlinear feature map

$$\phi : \mathbb{R}^d \longrightarrow \mathcal{H}.$$

Rather than computing $\phi(x)$ explicitly, one defines a symmetric positive-definite function (Mercer kernel)

$$k\big(x, y\big) \;=\; \big\langle \phi(x),\, \phi(y) \big\rangle_{\mathcal{H}}, \quad x, y \in \mathbb{R}^d,$$

which computes inner products in $\mathcal{H}$ directly. By Mercer's theorem, if k is continuous, symmetric, and

$$\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} f(x)\, k(x,y)\, f(y)\, dx\, dy \;\geqslant 0 \quad \text{for all compactly supported } f,$$

then there exists an orthonormal basis $\{u_i\}_{i=1}^{\infty}$ of $L^2(\mathbb{R}^d)$ with nonnegative eigenvalues $\{\mu_i\}$ such that

$$k(x,y) \;=\; \sum_{i=1}^{\infty} \mu_i\, u_i(x)\, u_i(y),$$

and $\phi(x) = \big(\sqrt{\mu_1}\, u_1(x),\, \sqrt{\mu_2}\, u_2(x),\, \dots\big)$. Common choices of kernels include the Gaussian (RBF) kernel, polynomial kernels, and the cosine similarity kernel. In electron microscopy, where shot noise can be extreme, the cosine similarity kernel

$$k(x,y) \;=\; \frac{x^{\top} y}{\|x\|\, \|y\|}$$

is often favored for its robustness under high noise levels [25]. Through kernelization, one can apply linear algorithms (e.g., PCA, support vector machines) in $\mathcal{H}$ without ever computing $\phi(x)$ explicitly.

5.2.3 Eigenvalue Problem in Reproducing Kernel Hilbert Space

Kernel Principal Component Analysis (kPCA) generalizes linear PCA by performing PCA in the implicit feature space $\mathcal{H}$. Given a dataset $\{\mathbf{I}_i\}_{i=1}^n \subset \mathbb{R}^d$, define the kernel (Gram) matrix $K \in \mathbb{R}^{n\times n}$ with entries

$$K_{ij} \;=\; k\big(\mathbf{I}_i, \mathbf{I}_j\big) \;=\; \big\langle \phi(\mathbf{I}_i),\, \phi(\mathbf{I}_j) \big\rangle_{\mathcal{H}}.$$

To center the data in feature space, let $\mathbf{1}_n$ be the $n \times n$ matrix whose entries are all $1/n$. The centered Gram matrix is

$$K_c \;=\; K \;-\; \mathbf{1}_n K \;-\; K\,\mathbf{1}_n \;+\; \mathbf{1}_n K\,\mathbf{1}_n.$$

We then solve the eigenvalue problem in $\mathcal{H}$:

$$C\,u = \lambda\,u, \quad C \;=\; \frac{1}{n}\sum_{i=1}^{n} \phi(\mathbf{I}_i)\,\phi(\mathbf{I}_i)^{*},$$

where C is the covariance operator in $\mathcal{H}$. By the representer theorem, any eigenvector $u \in \mathcal{H}$ lies in the span of $\{\phi(\mathbf{I}_i)\}$. Hence the eigenproblem reduces to solving

$$K_c\,\boldsymbol{v} \;=\; n\,\lambda\,\boldsymbol{v},$$

where $\boldsymbol{v} = (v_1, \ldots, v_n)^{\top} \in \mathbb{R}^n$. Denote the eigenpairs by $\{(\lambda_i, \boldsymbol{v}_i)\}_{i=1}^n$, with $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant 0$. The i-th principal component in feature space for a sample $\mathbf{I}_j$ is

$$\big\langle u_i,\, \phi(\mathbf{I}_j) \big\rangle_{\mathcal{H}} \;=\; \sum_{k=1}^{n} v_{ik}\, k\big(\mathbf{I}_k, \mathbf{I}_j\big),$$

where v_{ik} is the v_k of $\boldsymbol{v}_i$, so that the m-dimensional embedding of $\mathbf{I}_j \in \mathbb{R}^d$ is

$$\mathbf{z}_j \;=\; \big(\langle u_1,\, \phi(\mathbf{I}_j)\rangle,\; \ldots,\; \langle u_m,\, \phi(\mathbf{I}_j)\rangle\big)^{\top} \;=\; \big(v_{1j},\, v_{2j},\, \ldots,\, v_{mj}\big)^{\top} \in \mathbb{R}^m.$$

Truncation to the top m eigenvalues yields a nonlinear dimensionality reduction that suppresses noise while preserving underlying manifold structure [40]. Applications in image denoising have demonstrated that kPCA can effectively extract signal components from high-noise imaging modalities [25].

5.3 Pre-image Algorithms of kPCA

In kernel principal component analysis (kPCA) the data are nonlinearly mapped into a high-dimensional feature space, and denoising or dimensionality reduction are performed there. Because the feature map

$$\varphi : \mathbb{R}^d \to \mathcal{H}$$

is implicit, an exact inverse is unavailable. The *pre-image problem* seeks

$$x^* = \arg\min_{x \in \mathbb{R}^d} \|\varphi(x) - \psi\|^2,$$

where $\psi \in \mathcal{H}$ is a feature-space vector (e.g., a denoised or projected point).

Approximate solutions to this pre-image problem have also been studied in the context of Laplacian Eigenmaps by Doster [14], who extended earlier kPCA techniques using constrained optimization and the Nyström method to enable data reconstruction in spectral graph-based frameworks.

5.3.1 Fixed-point iteration and kernel-ridge regression (Bakir et al., 2003)

Using a Gaussian kernel

$$k(x, x_i) = \exp\bigl(-\|x - x_i\|^2/c\bigr),$$

one may compute the pre-image by iterating the fixed-point update

$$x_{t+1} \;=\; \frac{\sum_{i=1}^N k(x_t, x_i)\, x_i}{\sum_{i=1}^N k(x_t, x_i)},$$

which drives $\varphi(x_t)$ toward the target ψ. Alternatively, one can learn an inverse map $\Gamma' : \mathcal{H} \to \mathbb{R}^d$ by solving

$$\min_{\Gamma'} \sum_{i=1}^N \|x_i - \Gamma'(\varphi(x_i))\|^2 \;+\; \lambda \|\Gamma'\|^2,$$

so that $x^* \approx \Gamma'(\psi)$ once Γ' is fitted [4].

5.3.2 MDS-based inversion (Kwok & Tsang, 2003)

This non-iterative approach exploits the isometry between feature-space and input-space distances for isotropic kernels. Define $\tilde{d}_i = \|\psi - \varphi(x_i)\|$ and invert the kernel-induced dis-

tance function to obtain estimates of the corresponding input-space distances. Applying multidimensional scaling (MDS) to these distances then yields the pre-image x^* [27].

5.3.3 Direct-approximation pre-image algorithm (Rathi et al., 2006)

For kernels with a closed-form distance inversion, one observes

$$f(d_{ij}^2) = \tfrac{1}{2}\big(K_{ii} + K_{jj} - \tilde{d}_{ij}^2\big) \implies d_{ij}^2 = f^{-1}\Big(\tfrac{1}{2}(K_{ii} + K_{jj} - \tilde{d}_{ij}^2)\Big).$$

Approximating $\varphi(x^*) \approx P\,\varphi(x)$, one derives the single-step estimator

$$x^* = \frac{\sum_{i=1}^{N} \tilde{\gamma}_i\, x_i}{\sum_{i=1}^{N} \tilde{\gamma}_i},$$

where the weights $\tilde{\gamma}_i$ are algebraic functions of the feature-space distances $\tilde{d}(\varphi(x_i), P\,\varphi(x))$. This yields a unique pre-image without iteration [38].

5.3.4 Nyström-based hybrid pre-image algorithm (Sapiro et al., 2007)

Sapiro et al. build upon the Nyström out-of-sample extension and fuse ideas from the MDS-based method and the direct-approximation method. Given the centered Gram eigendecomposition

$$K_c = U\,\widetilde{\Lambda}\,U^\top,$$

the feature map of a new point x is approximated by

$$\hat{\varphi}(x) = \widetilde{\Lambda}^{-1/2}\,U^\top\,k_c(x),$$

where $k_c(x)$ is the centered kernel vector. For a target feature ψ, they normalize ψ onto the unit sphere in $\mathcal{H}$ and compute

$$\hat{k}_x = \widetilde{\Lambda}^{-1/2}\,U^\top\,\frac{\psi}{\|\psi\|}.$$

The pre-image is then recovered either by solving

$$x^* = \arg\min_{x \in \mathbb{R}^d} \big\| k_c(x) - \hat{k}_x \big\|^2$$

or—in the common case that $\hat{k}_x(i) \geqslant 0$—in closed form as

$$x^* = \frac{\sum_{i=1}^{N} \hat{k}_x(i)\, x_i}{\sum_{i=1}^{N} \hat{k}_x(i)}.$$

This hybrid scheme inherits the robustness of Nyström extension, the global consistency of MDS inversion, and the efficiency of the direct-approximation estimator, achieving state-of-the-art performance in reconstruction fidelity and speed [1].

5.4 Comparative Evaluation of kPCA Pre-image Algorithms

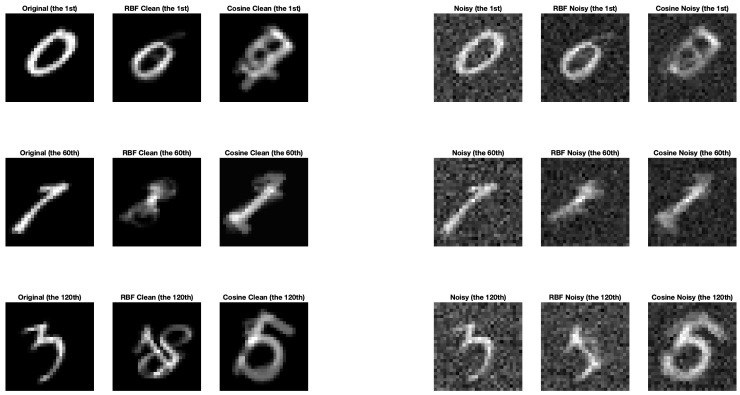
In this section, we present a comparative evaluation of three distinct Kernel PCA (kPCA) pre-image reconstruction methods, which were applied to three widely recognized image datasets: Yann LeCun's MNIST, CIFAR-10, and SVHN. The evaluation framework was devised to measure reconstruction performance under varying experimental conditions. For each dataset, specific preprocessing was applied that included the addition of Gaussian noise, rigorous trimming of pixel intensities, and normalization to maintain the dynamic range of the image data. In the case of MNIST, two experimental configurations were adopted: one without any normalization or trimming and another with both applied. Both the CIFAR-10 and SVHN datasets, originally in RGB and subsequently converted to grayscale, were processed with normalization and trimming. Additionally, the reconstruction experiments were conducted for three different configurations of principal components (20, 120, and 220) and for two kernel functions (Radial Basis Function and Cosine). The quality of the reconstructions was quantitatively assessed by averaging the Peak Signal-to-Noise Ratio (PSNR), the Structural Similarity Index (SSIM), and the Pearson Correlation Coefficient (PCC) over all samples.

The three pre-image reconstruction algorithms compared in this study include Sapiro's method, an approach based on kernel ridge regression implemented via the Sklearn package, and the method developed by Schölkopf [4]. Sapiro's algorithm [1] was implemented following the procedures outlined in Sapiro's original work as well as methodologies discussed in related works by Kwok [27] and Cloninger [11][9]. The Sklearn-based method leverages the kernel ridge regression framework, and the Schölkopf method is implemented with fixed iteration constraints and tolerance levels. Each of these techniques was rigorously tested under similar conditions to ensure a fair comparison and to identify their relative strengths and limitations when confronted with various noise levels and preprocessing protocols.

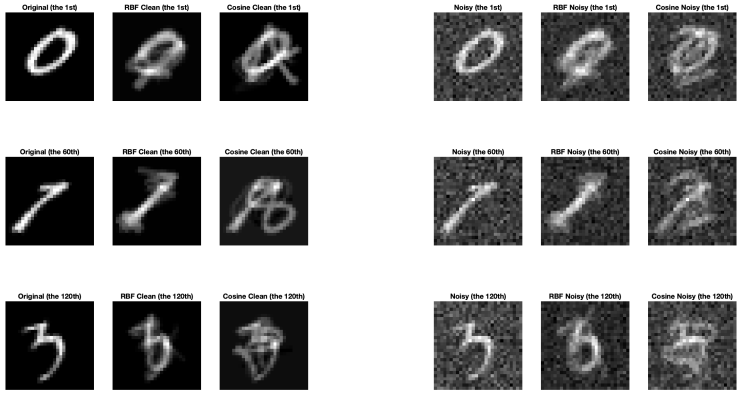
Implementation Credits. The implementation of the Schölkopf algorithm was completed with the assistance of Dr. Cong Tuan Son Van (National Cancer Institute, NIH). The implementation of Sapiro's algorithm was adapted from code originally written by Alexander Cloninger, Timothy Doster and Jeremiah Emidih during their time as graduate students at the University of Maryland, College Park, under the mentorship of our shared advisor Dr. Wojciech Czaja. Alexander Cloninger is currently a professor in the department of mathematics and the Halıcıoğlu data sciencei institute at UC San Diego, Timothy Doster is a senior data scientist at Pacific Northwest National Laboratory and Jeremiah Emidih is currently a professor in the Mathematics Department at Montgomery College, Takoma Park/Silver Spring Campus. The methodological foundations of their implementations draw heavily from their doctoral work on Laplacian eigenmap [9][14], graph-based multimodal data fusion [15], and the applications of pre-image algorithms[10][16]. The Sklearn-based reconstruction method is credited to the Scikit-learn developers, particularly Andreas Müller and Arnaud Joly, who contributed to the implementation of `sklearn.decomposition.KernelPCA` and its inverse transform functionality.

5.4.1 Experiment 1: MNIST KPCA Reconstructions Without Normalization or Trimming

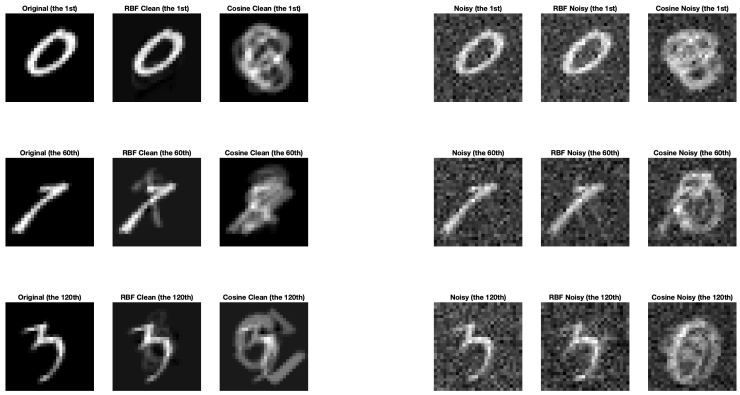
The results for Experiment 1 (MNIST dataset without normalization or trimming) are summarized in the following tables and figures. Each table reports average PSNR, SSIM, and PCC values across three principal component settings (20, 120, and 220) for both noise-free and noisy conditions.



(a) PC = 20, left panel: original data, right panel: noisy data

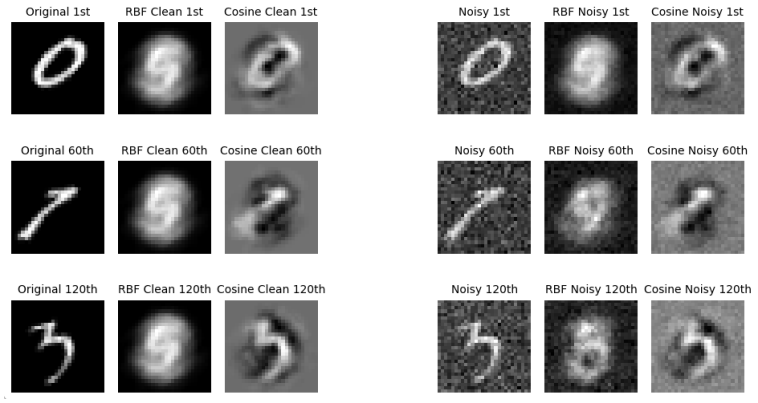


(b) PC = 120

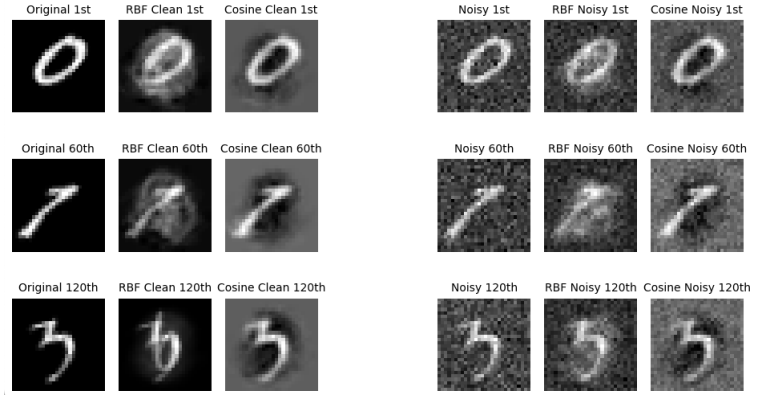


(c) PC = 220

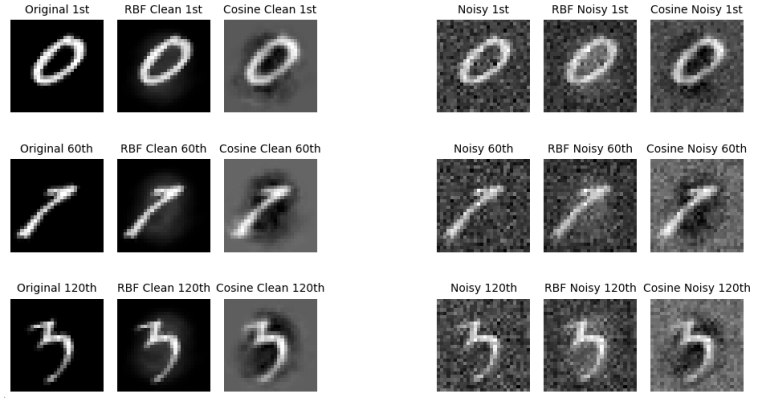
Figure 5.1: Reconstruction results for selected samples using Sapiro's algorithm in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20

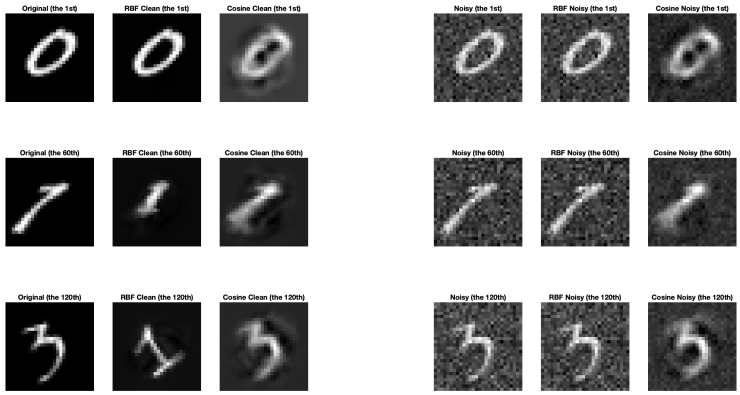


(b) PC = 120

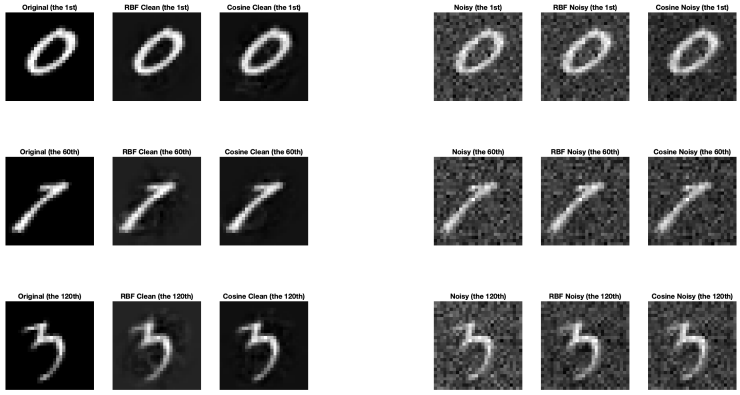


(c) PC = 220

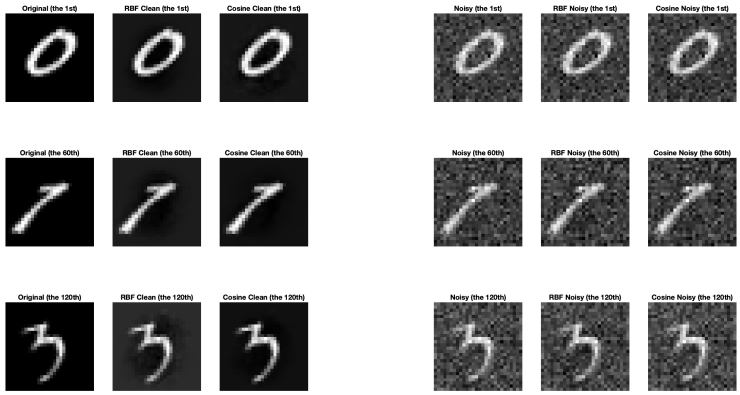
Figure 5.2: Reconstruction results for selected samples using Sklearn's algorithm in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20



(b) PC = 120



(c) PC = 220

Figure 5.3: Reconstruction results for selected samples using Schölkopf's algorithm in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.

Table 5.1: Experiment 1: Reconstruction performance on MNIST with RBF kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

Algorithm	PC = 20	PC = 120	PC = 220
Without noise			
	PSNR: 16.063 dB	PSNR: 19.2189 dB	**PSNR: 34.2829 dB**
	SSIM: 0.59167	SSIM: 0.74353	**SSIM: 0.87190**
Sapiro's	PCC: 0.59177	PCC: 0.85626	PCC: 0.95176
	PSNR: 15.0141 dB	PSNR: 18.1582 dB	PSNR: 19.2051 dB
	SSIM: 0.38596	SSIM: 0.59377	SSIM: 0.67285
Sklearn's	PCC: 0.54920	PCC: 0.84769	PCC: 0.92504
	PSNR: 29.2248 dB	**PSNR: 25.9994 dB**	PSNR: 20.3593 dB
	SSIM: 0.81090	**SSIM: 0.74986**	SSIM: 0.62812
Schölkopf's	**PCC: 0.80031**	**PCC: 0.96948**	**PCC: 0.97790**
With noise			
	PSNR: 14.8575 dB	PSNR: 17.4915 dB	**PSNR: 18.5474 dB**
	SSIM: 0.17427	SSIM: 0.30864	**SSIM: 0.33622**
Sapiro's	PCC: 0.52694	PCC: 0.78266	**PCC: 0.83029**
	PSNR: 14.3378 dB	PSNR: 16.5479 dB	PSNR: 17.6918 dB
	SSIM: 0.27887	SSIM: 0.28118	SSIM: 0.31541
Sklearn's	PCC: 0.51226	PCC: 0.75467	PCC: 0.82208
	PSNR: 17.0432 dB	**PSNR: 18.2186 dB**	PSNR: 14.6645 dB
	SSIM: 0.32336	**SSIM: 0.32548**	SSIM: 0.25925
Schölkopf's	**PCC: 0.82860**	**PCC: 0.86955**	PCC: 0.82111

Table 5.2: Experiment 1: Reconstruction performance on MNIST with Cosine kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

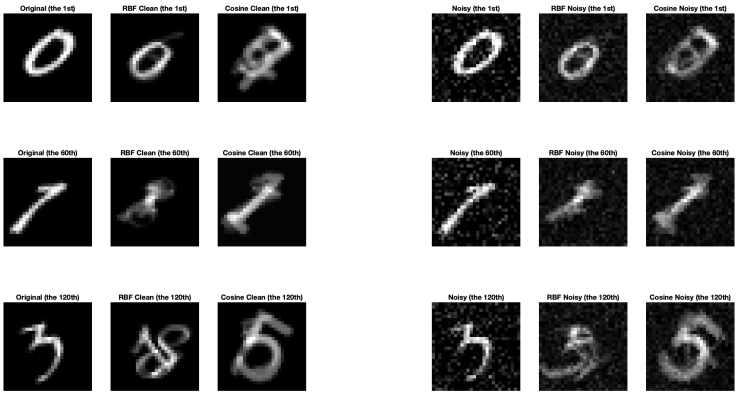
Algorithm	PC = 20	PC = 120	PC = 220
Without noise			
	PSNR: 16.8079 dB	PSNR: 16.1836 dB	PSNR: 15.0487 dB
	SSIM: 0.65989	**SSIM: 0.58499**	SSIM: 0.52058
Sapiro's	PCC: 0.76849	PCC: 0.72452	PCC: 0.65029
	PSNR: 14.6620 dB	**PSNR: 16.2781 dB**	**PSNR: 16.3100 dB**
	SSIM: 0.32013	SSIM: 0.34753	SSIM: 0.35044
Sklearn's	PCC: 0.67446	PCC: 0.82555	PCC: 0.82764
	PSNR: 13.1369 dB	PSNR: 13.7107 dB	PSNR: 13.9573 dB
	SSIM: 0.50397	SSIM: 0.53358	**SSIM: 0.53302**
Schölkopf's	**PCC: 0.79498**	**PCC: 0.99037**	**PCC: 0.99487**
With noise			
	PSNR: 16.0498 dB	PSNR: 15.4797 dB	PSNR: 14.2854 dB
	SSIM: 0.27048	SSIM: 0.23087	SSIM: 0.18748
Sapiro's	**PCC: 0.72219**	PCC: 0.65784	PCC: 0.57396
	PSNR: 14.2645 dB	**PSNR: 15.5264 dB**	**PSNR: 15.5779 dB**
	SSIM: 0.15176	SSIM: 0.11324	SSIM: 0.10984
Sklearn's	PCC: 0.63737	PCC: 0.74780	PCC: 0.75071
	PSNR: 12.9426 dB	PSNR: 13.4329 dB	PSNR: 13.6587 dB
	SSIM: 0.48545	**SSIM: 0.45204**	**SSIM: 0.38389**
Schölkopf's	PCC: 0.67415	**PCC: 0.86313**	**PCC: 0.82548**

Table 5.3: Experiment 1: Highest reconstruction performance metrics (PSNR in dB, SSIM, PCC) on MNIST for each principal-component count and noise condition, indicating kernel and algorithm.

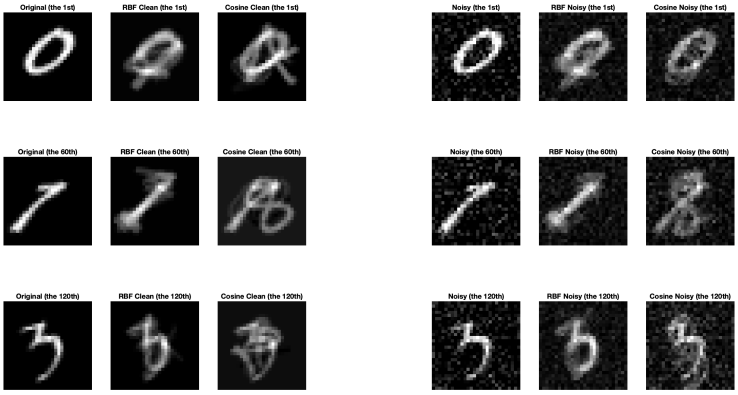
	PC = 20	**PC = 120**	**PC = 220**
Without noise	PSNR: 29.2248 dB (rbf, Schölkopf) SSIM: 0.81090 (rbf, Schölkopf) PCC: 0.80031 (rbf, Schölkopf)	PSNR: 25.9994 dB (rbf, Schölkopf) SSIM: 0.74986 (rbf, Schölkopf) PCC: 0.99037 (cos, Schölkopf)	PSNR: 34.2829 dB (rbf, Sapiro) SSIM: 0.87190 (rbf, Sapiro) PCC: 0.99487 (cos, Schölkopf)
With noise	PSNR: 17.0432 dB (rbf, Schölkopf) SSIM: 0.48545 (cos, Schölkopf) PCC: 0.82860 (rbf, Schölkopf)	PSNR: 18.2186 dB (rbf, Schölkopf) SSIM: 0.45204 (cos, Schölkopf) PCC: 0.86955 (rbf, Schölkopf)	PSNR: 18.5474 dB (rbf, Sapiro) SSIM: 0.38389 (cos, Schölkopf) PCC: 0.83029 (rbf, Sapiro)

5.4.2 Experiment 2: MNIST KPCA Reconstructions with Noise Trimming and Normalization

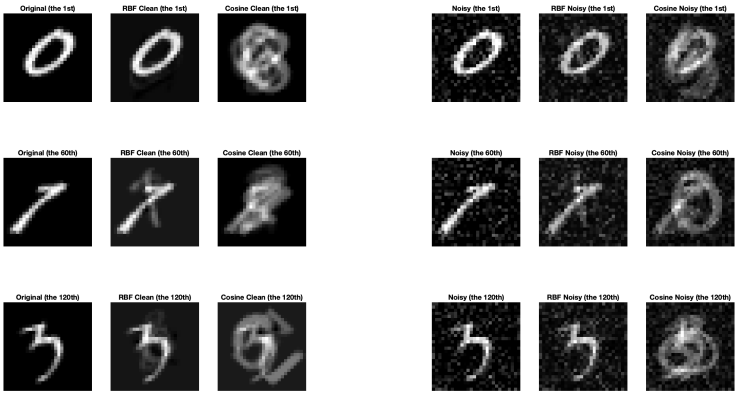
The results for Experiment 2 (MNIST dataset with noise trimming and reconstruction normalization) are summarized in the following tables and figures. Each table reports average PSNR, SSIM, and PCC values across three principal component settings (20, 120, and 220) for both noise-free and noisy conditions.



(a) PC = 20

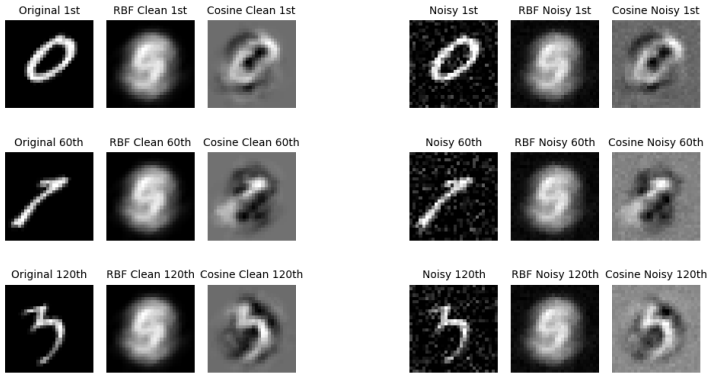


(b) PC = 120

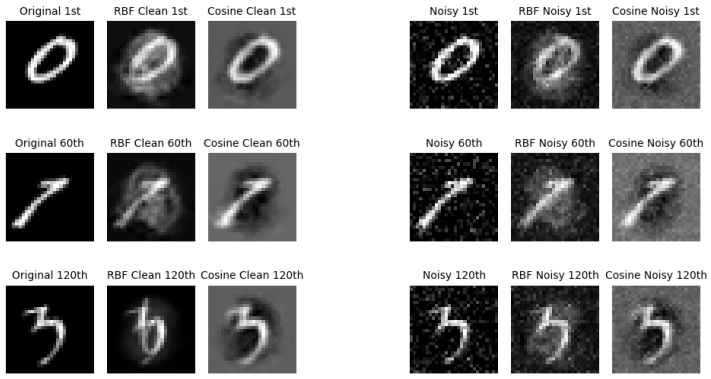


(c) PC = 220

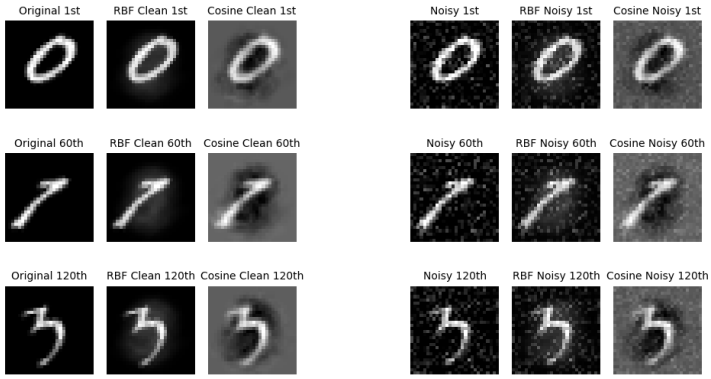
Figure 5.4: MNIST reconstruction results for selected samples using Sapiro's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20

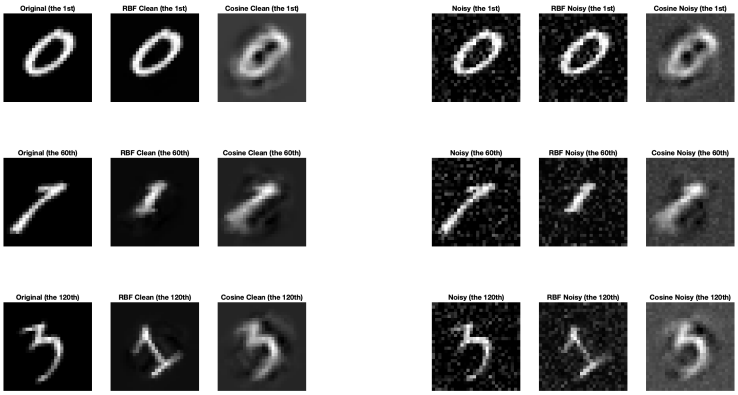


(b) PC = 120

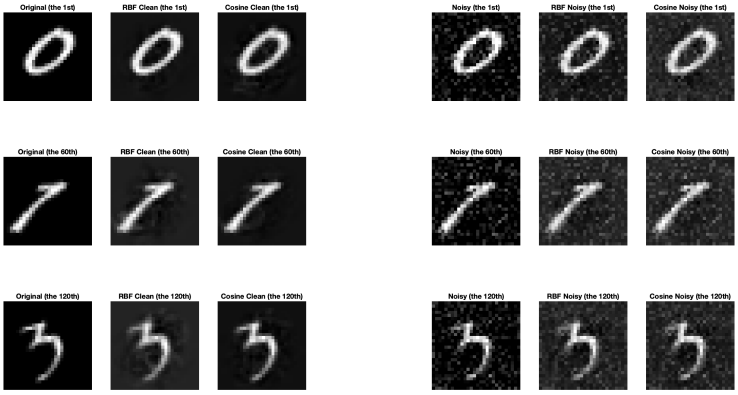


(c) PC = 220

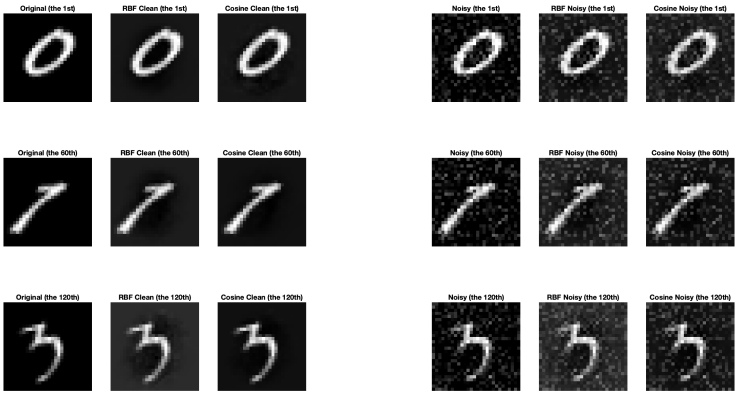
Figure 5.5: MNIST reconstruction results for selected samples using Sklearn's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20



(b) PC = 120



(c) PC = 220

Figure 5.6: MNIST reconstruction results for selected samples using Schölkopf's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.

Table 5.4: Experiment 2: Reconstruction performance on the MNIST dataset processed with noise trimming and reconstruction normalization, using the RBF kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
	PSNR: 15.5741 dB	PSNR: 18.0449 dB	**PSNR: 30.4618 dB**
	SSIM: 0.54797	**SSIM: 0.59500**	SSIM: 0.50114
Sapiro's	PCC: 0.59177	PCC: 0.85626	PCC: 0.95176
	PSNR: 11.9164 dB	PSNR: 17.5857 dB	PSNR: 21.1803 dB
	SSIM: 0.32384	SSIM: 0.47228	**SSIM: 0.66136**
Sklearn's	PCC: 0.54920	PCC: 0.84769	PCC: 0.92504
	PSNR: 28.1679 dB	**PSNR: 20.7142 dB**	PSNR: 20.4680 dB
	SSIM: 0.57661	SSIM: 0.34627	SSIM: 0.35505
Schölkopf's	**PCC: 0.80031**	**PCC: 0.96948**	**PCC: 0.97779**
With noise			
	PSNR: 14.7157 dB	**PSNR: 17.4915 dB**	**PSNR: 17.6585 dB**
	SSIM: 0.14669	**SSIM: 0.30864**	SSIM: 0.29364
Sapiro's	PCC: 0.60750	PCC: 0.78266	**PCC: 0.88384**
	PSNR: 11.3695 dB	PSNR: 15.1428 dB	PSNR: 17.5032 dB
	SSIM: 0.11425	SSIM: 0.23284	SSIM: 0.28775
Sklearn's	PCC: 0.53936	PCC: 0.81303	PCC: 0.87774
	PSNR: 17.9833 dB	PSNR: 16.4429 dB	PSNR: 16.0327 dB
	SSIM: 0.28058	SSIM: 0.29106	**SSIM: 0.29379**
Schölkopf's	**PCC: 0.80314**	**PCC: 0.91387**	PCC: 0.88028

Table 5.5: Experiment 2: Reconstruction performance on the MNIST dataset processed with noise trimming and reconstruction normalization, using the Cosine kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

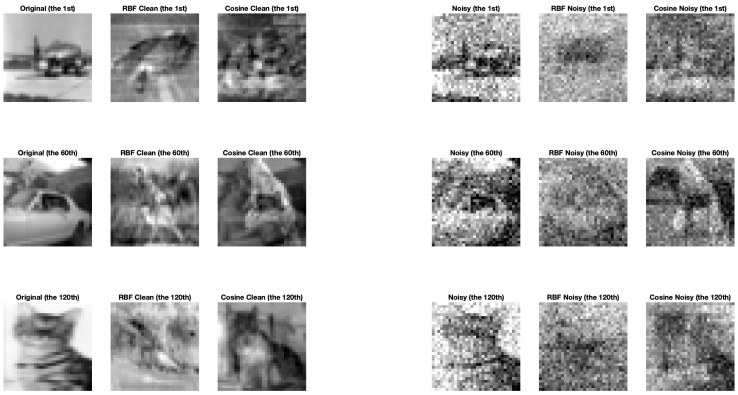
Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
	PSNR: 16.4763 dB	PSNR: 14.9552 dB	PSNR: 13.8741 dB
	SSIM: 0.57914	**SSIM: 0.40150**	**SSIM: 0.38981**
Sapiro's	PCC: 0.76849	PCC: 0.72452	PCC: 0.65029
	PSNR: 8.0572 dB	PSNR: 10.5753 dB	PSNR: 10.6072 dB
	SSIM: 0.14921	SSIM: 0.23731	SSIM: 0.23924
Sklearn's	PCC: 0.67446	PCC: 0.82555	PCC: 0.82764
	PSNR: 13.2347 dB	**PSNR: 22.0578 dB**	**PSNR: 23.2792 dB**
	SSIM: 0.19400	SSIM: 0.36058	SSIM: 0.38152
Schölkopf's	**PCC: 0.79498**	**PCC: 0.99037**	**PCC: 0.99487**
With noise			
	PSNR: 15.4668 dB	PSNR: 14.5595 dB	**PSNR: 13.2614 dB**
	SSIM: 0.22014	SSIM: 0.21454	SSIM: 0.18060
Sapiro's	PCC: 0.74468	PCC: 0.65784	PCC: 0.64381
	PSNR: 8.0694 dB	PSNR: 10.2582 dB	PSNR: 10.3974 dB
	SSIM: 0.14551	SSIM: 0.22523	**SSIM: 0.23083**
Sklearn's	PCC: 0.65497	PCC: 0.78807	**PCC: 0.79180**
	PSNR: 16.2480 dB	**PSNR: 16.0002 dB**	PSNR: 11.2041 dB
	SSIM: 0.28578	**SSIM: 0.29795**	SSIM: 0.17691
Schölkopf's	**PCC: 0.91144**	**PCC: 0.88574**	PCC: 0.76137

Table 5.6: Experiment 2: Highest reconstruction performance metrics (PSNR in dB, SSIM, PCC) on MNIST with noise trimming and reconstruction normalization for each principal-component count and noise condition, indicating kernel and algorithm.

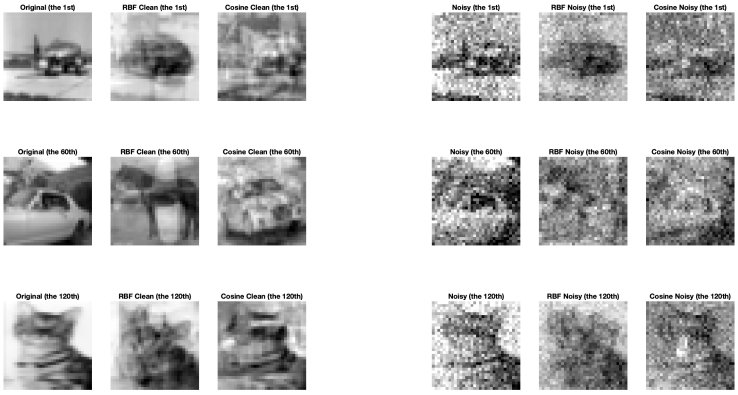
	PC = 20	**PC = 120**	**PC = 220**
Without noise	PSNR: 28.1679 dB (rbf, Schölkopf) SSIM: 0.57914 (cos, Sapiro) PCC: 0.80031 (rbf, Schölkopf)	PSNR: 22.0578 dB (cos, Schölkopf) SSIM: 0.59500 (rbf, Sapiro) PCC: 0.99037 (cos, Schölkopf)	PSNR: 30.4618 dB (rbf, Sapiro) SSIM: 0.66136 (rbf, Sklearn) PCC: 0.99487 (cos, Schölkopf)
With noise	PSNR: 17.9833 dB (rbf, Schölkopf) SSIM: 0.28578 (cos, Schölkopf) PCC: 0.91144 (cos, Schölkopf)	PSNR: 17.4915 dB (rbf, Sapiro) SSIM: 0.30864 (rbf, Sapiro) PCC: 0.91387 (rbf, Schölkopf)	PSNR: 17.6585 dB (rbf, Sapiro) SSIM: 0.29379 (rbf, Schölkopf) PCC: 0.88384 (rbf, Sapiro)

5.4.3 Experiment 3: CIFAR-10 KPCA Reconstructions with Noise Trimming and Normalization

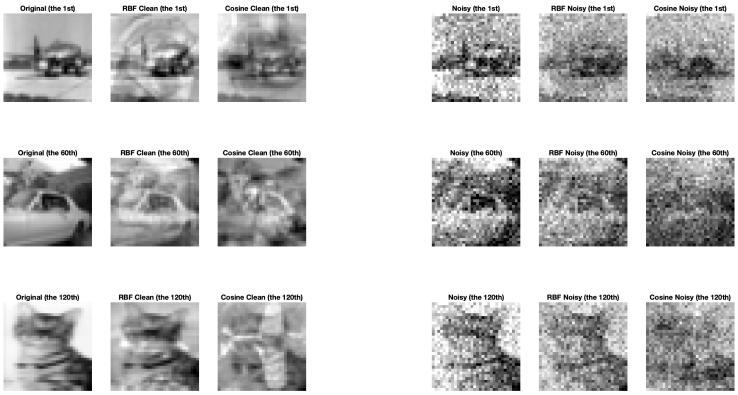
The results for Experiment 3 (CIFAR-10 dataset with noise trimming and reconstruction normalization) are summarized in the following tables and figures. Each table reports average PSNR, SSIM, and PCC values across three principal component settings (20, 120, and 220) for both noise-free and noisy conditions.



(a) PC = 20

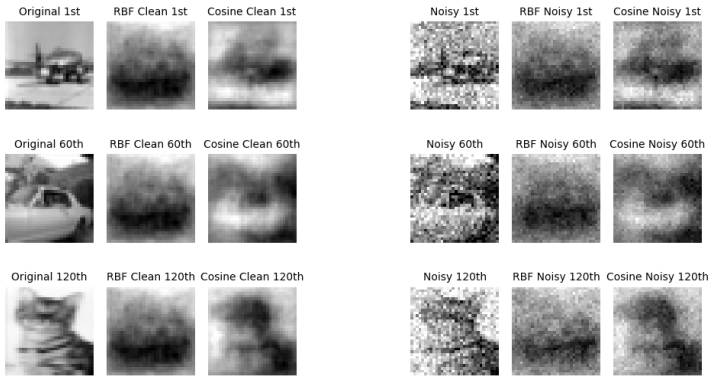


(b) PC = 120

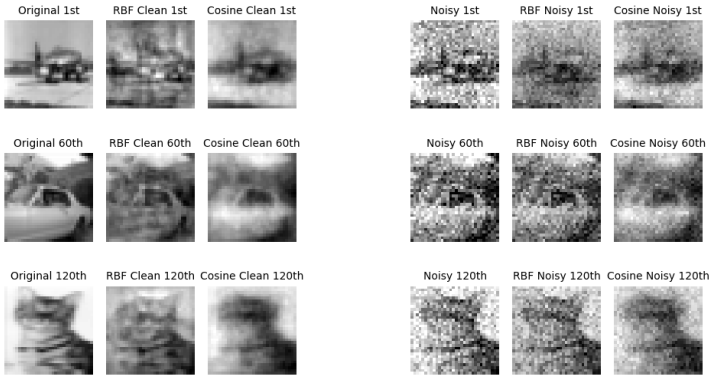


(c) PC = 220

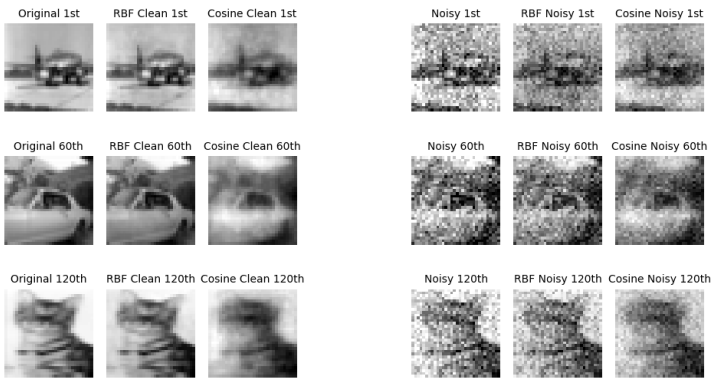
Figure 5.7: CIFAR-10 reconstruction results for selected samples using Sapiro's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20

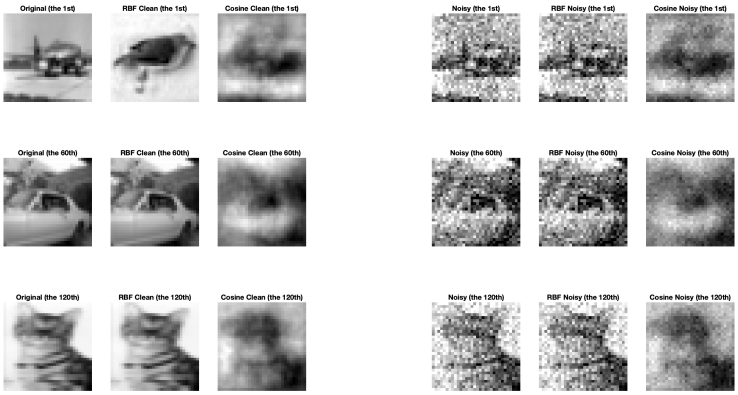


(b) PC = 120

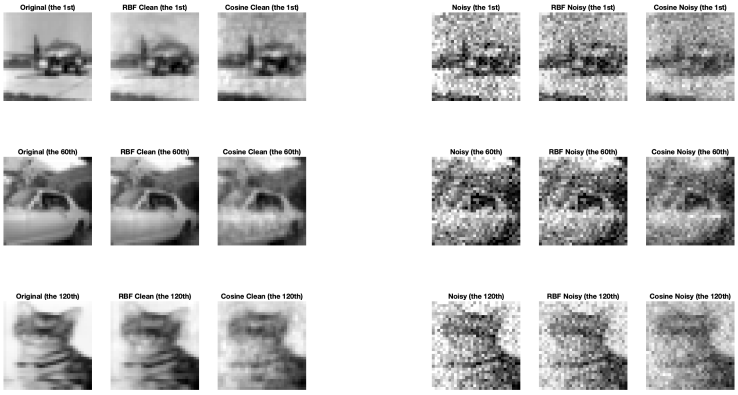


(c) PC = 220

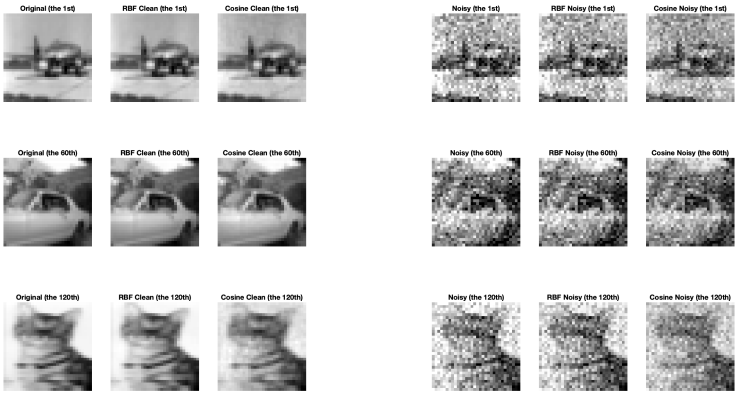
Figure 5.8: CIFAR-10 reconstruction results for selected samples using Sklearn's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.



(a) PC = 20



(b) PC = 120



(c) PC = 220

Figure 5.9: CIFAR-10 reconstruction results for selected samples using Schölkopf's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original images, the reconstructions using Gaussian and Cosine kernels on clean data, noise-added images, and the reconstructions using Gaussian and Cosine kernels on noisy data.

Table 5.7: Experiment 3: Reconstruction performance on the CIFAR-10 dataset processed with noise trimming and reconstruction normalization, using the RBF kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
	PSNR: 13.3551 dB	PSNR: 15.0723 dB	PSNR: 18.7937 dB
	SSIM: 0.26482	SSIM: 0.44356	SSIM: 0.66653
Sapiro's	PCC: 0.50601	PCC: 0.70060	PCC: 0.83195
	PSNR: 11.0242 dB	PSNR: 16.3620 dB	PSNR: 21.5711 dB
	SSIM: 0.19281	SSIM: 0.65102	SSIM: 0.85209
Sklearn's	PCC: 0.25295	PCC: 0.80105	PCC: 0.93341
	PSNR: 32.0257 dB	**PSNR: 26.1939 dB**	**PSNR: 29.5793 dB**
	SSIM: 0.70647	**SSIM: 0.81574**	**SSIM: 0.92645**
Schölkopf's	**PCC: 0.79872**	**PCC: 0.94018**	**PCC: 0.98085**
With noise			
	PSNR: 13.5640 dB	PSNR: 15.0851 dB	PSNR: 16.0939 dB
	SSIM: 0.14792	SSIM: 0.30266	SSIM: 0.38900
Sapiro's	PCC: 0.44617	PCC: 0.63775	PCC: 0.70653
	PSNR: 11.7004 dB	PSNR: 15.1307 dB	PSNR: 17.0450 dB
	SSIM: 0.14974	SSIM: 0.38930	SSIM: 0.45480
Sklearn's	PCC: 0.22353	PCC: 0.64222	PCC: 0.75785
	PSNR: 17.4573 dB	**PSNR: 19.9966 dB**	**PSNR: 18.6666 dB**
	SSIM: 0.45129	**SSIM: 0.52633**	**SSIM: 0.49187**
Schölkopf's	**PCC: 0.78380**	**PCC: 0.83514**	**PCC: 0.80486**

Table 5.8: Experiment 3: Reconstruction performance on the CIFAR-10 dataset processed with noise trimming and reconstruction normalization, using the Cosine kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

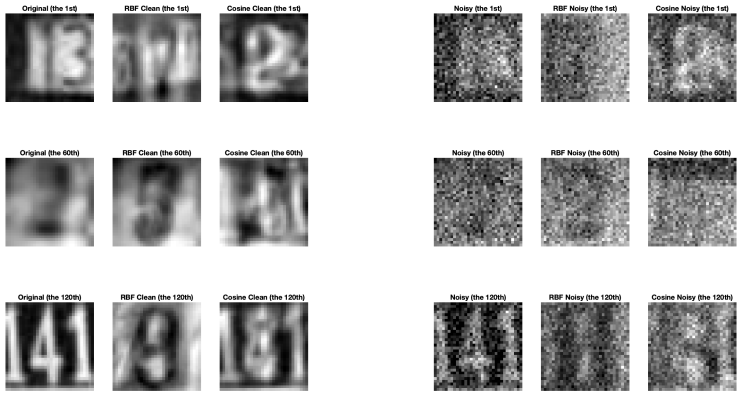
Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
	PSNR: 12.1369 dB	PSNR: 12.4989 dB	PSNR: 12.9765 dB
	SSIM: 0.23665	SSIM: 0.35425	SSIM: 0.39891
Sapiro's	PCC: 0.35143	PCC: 0.38891	PCC: 0.43788
	PSNR: 15.4442 dB	**PSNR: 18.1776 dB**	**PSNR: 18.3461 dB**
	SSIM: 0.44480	**SSIM: 0.71850**	**SSIM: 0.74413**
Sklearn's	**PCC: 0.80553**	PCC: 0.92225	PCC: 0.92846
	PSNR: 14.2991 dB	PSNR: 14.1913 dB	PSNR: 14.1961 dB
	SSIM: 0.37769	SSIM: 0.59229	SSIM: 0.67017
Schölkopf's	PCC: 0.80149	**PCC: 0.96454**	**PCC: 0.98687**
With noise			
	PSNR: 12.6982 dB	PSNR: 13.0409 dB	PSNR: 12.9027 dB
	SSIM: 0.23339	SSIM: 0.28657	SSIM: 0.27318
Sapiro's	PCC: 0.40311	PCC: 0.41246	PCC: 0.39939
	PSNR: 15.6678 dB	**PSNR: 17.6525 dB**	**PSNR: 18.0116 dB**
	SSIM: 0.37599	**SSIM: 0.51045**	**SSIM: 0.52890**
Sklearn's	**PCC: 0.77861**	**PCC: 0.85174**	**PCC: 0.85676**
	PSNR: 14.3595 dB	PSNR: 13.9715 dB	PSNR: 12.0706 dB
	SSIM: 0.34727	SSIM: 0.43463	SSIM: 0.41633
Schölkopf's	PCC: 0.76893	PCC: 0.81220	PCC: 0.78565

Table 5.9: Experiment 3: Highest reconstruction performance metrics (PSNR in dB, SSIM, PCC) on the CIFAR-10 dataset processed with noise trimming and reconstruction normalization for each principal-component count and noise condition, indicating kernel and algorithm.

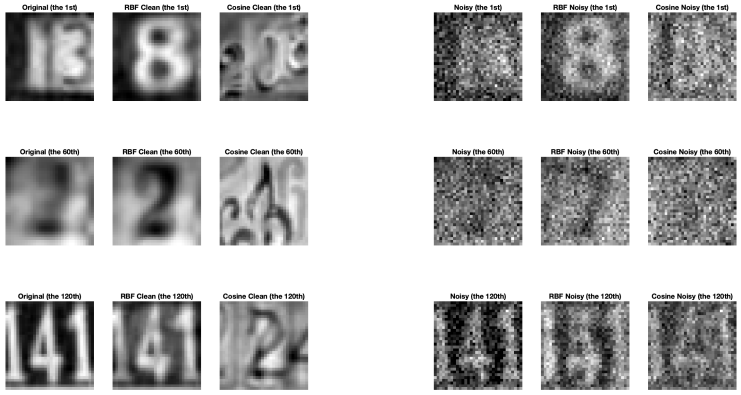
	PC = 20	**PC = 120**	**PC = 220**
Without noise	PSNR: 32.0257 dB (rbf, Schölkopf) SSIM: 0.70647 (rbf, Schölkopf) PCC: 0.80553 (cos, Sklearn)	PSNR: 26.1939 dB (rbf, Schölkopf) SSIM: 0.81574 (rbf, Schölkopf) PCC: 0.96454 (cos, Schölkopf)	PSNR: 29.5793 dB (rbf, Schölkopf) SSIM: 0.92645 (rbf, Schölkopf) PCC: 0.98085 (rbf, Schölkopf)
With noise	PSNR: 17.4573 dB (rbf, Schölkopf) SSIM: 0.45129 (rbf, Schölkopf) PCC: 0.78380 (rbf, Schölkopf)	PSNR: 19.9966 dB (rbf, Schölkopf) SSIM: 0.52633 (rbf, Schölkopf) PCC: 0.85174 (cos, Sklearn)	PSNR: 18.6666 dB (rbf, Schölkopf) SSIM: 0.52890 (cos, Sklearn) PCC: 0.85676 (cos, Sklearn)

5.4.4 Experiment 4: SVHN KPCA Reconstructions with Noise Trimming and Normalization

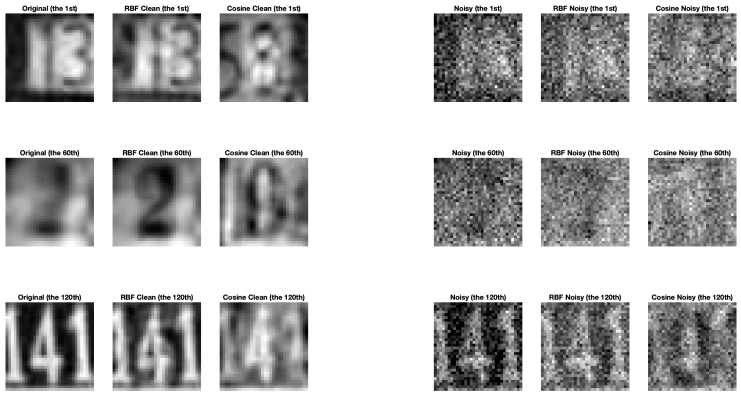
The results for Experiment 4 (SVHN dataset with noise trimming and reconstruction normalization) are summarized in the following tables and figures. Each table reports average PSNR, SSIM, and PCC values across three principal component settings (20, 120, and 220) for both noise-free and noisy conditions.



(a) PC = 20

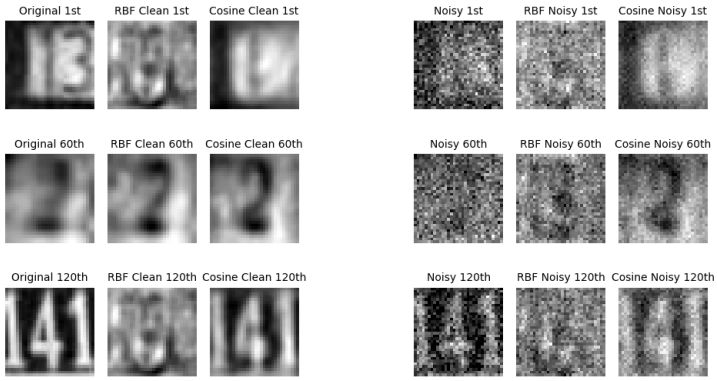


(b) PC = 120

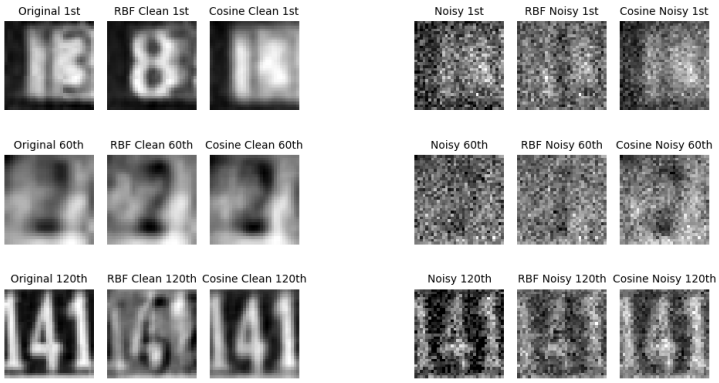


(c) PC = 220

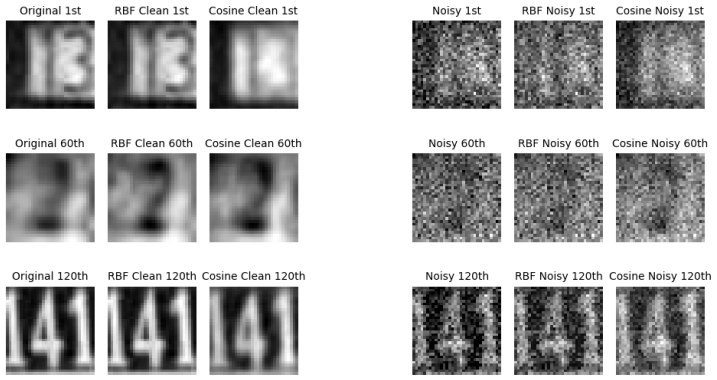
Figure 5.10: SVHN reconstruction results for selected samples using Sapiro's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original image, noisy counterpart, and reconstructions using Gaussian and Cosine kernels on both clean and noisy data.



(a) PC = 20

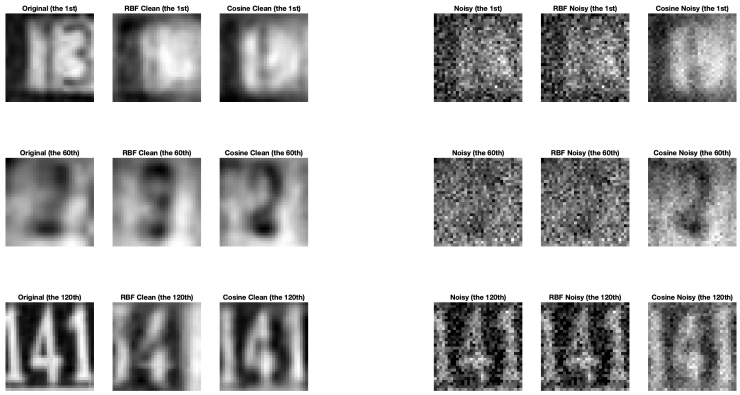


(b) PC = 120

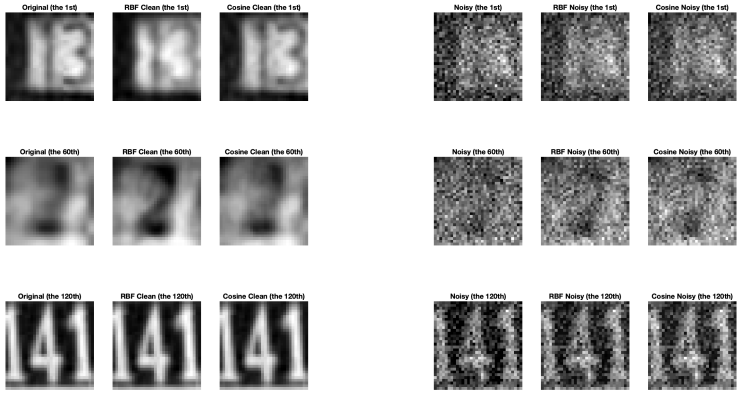


(c) PC = 220

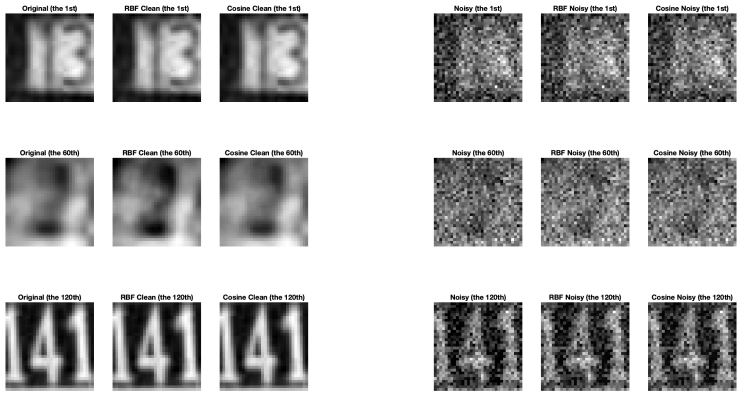
Figure 5.11: SVHN reconstruction results for selected samples using Sklearn's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original image, noisy counterpart, and reconstructions using Gaussian and Cosine kernels on both clean and noisy data.



(a) PC = 20



(b) PC = 120



(c) PC = 220

Figure 5.12: SVHN reconstruction results for selected samples using Schölkopf's algorithm with noise trimming and reconstruction normalization in (a)–(c), with principal component numbers 20, 120, and 220. Each subfigure shows the 1st, 60th, and 120th samples: original image, noisy counterpart, and reconstructions using Gaussian and Cosine kernels on both clean and noisy data.

Table 5.10: Experiment 4: Reconstruction performance on the SVHN dataset processed with noise trimming and reconstruction normalization, using the RBF kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
	PSNR: 12.3936 dB	PSNR: 14.2204 dB	PSNR: 15.0195 dB
	SSIM: 0.37956	SSIM: 0.54410	SSIM: 0.62858
Sapiro's	PCC: 0.61667	PCC: 0.81778	PCC: 0.86206
	PSNR: 12.0862 dB	PSNR: 13.6059 dB	PSNR: 14.7445 dB
	SSIM: 0.19137	SSIM: 0.52245	SSIM: 0.68296
Sklearn's	PCC: 0.24956	PCC: 0.77528	PCC: 0.95147
	PSNR: 16.7435 dB	**PSNR: 31.7435 dB**	**PSNR: 31.1285 dB**
	SSIM: 0.57350	**SSIM: 0.91557**	**SSIM: 0.97229**
Schölkopf's	**PCC: 0.74058**	**PCC: 0.96251**	**PCC: 0.99497**
With noise			
	PSNR: 13.7993 dB	PSNR: 15.1871 dB	PSNR: 15.5463 dB
	SSIM: 0.16785	SSIM: 0.28619	SSIM: 0.29977
Sapiro's	PCC: 0.42652	PCC: 0.59697	PCC: 0.61197
	PSNR: 12.2307 dB	PSNR: 14.7209 dB	PSNR: 15.8751 dB
	SSIM: 0.07551	SSIM: 0.25983	SSIM: 0.29318
Sklearn's	PCC: 0.07839	PCC: 0.50085	PCC: 0.59656
	PSNR: 17.3959 dB	**PSNR: 19.7507 dB**	**PSNR: 18.2819 dB**
	SSIM: 0.31339	**SSIM: 0.42549**	**SSIM: 0.35540**
Schölkopf's	**PCC: 0.61693**	**PCC: 0.71086**	**PCC: 0.63739**

Table 5.11: Experiment 4: Reconstruction performance on the SVHN dataset processed with noise trimming and reconstruction normalization, using the Cosine kernel (PSNR in dB, SSIM, PCC) for different pre-image algorithms and principal-component (PC) counts.

Algorithm	**PC = 20**	**PC = 120**	**PC = 220**
Without noise			
Sapiro's	PSNR: 12.8540 dB SSIM: 0.23984 PCC: 0.30650	PSNR: 12.4582 dB SSIM: 0.24595 PCC: 0.27282	PSNR: 12.7154 dB SSIM: 0.32673 PCC: 0.37398
Sklearn's	PSNR: 14.1107 dB SSIM: 0.54278 PCC: 0.86218	PSNR: 14.6252 dB SSIM: 0.66889 PCC: 0.96097	PSNR: 14.6214 dB SSIM: 0.67028 PCC: 0.96153
Schölkopf's	**PSNR: 16.9369 dB** **SSIM: 0.64507** **PCC: 0.86238**	**PSNR: 19.5307 dB** **SSIM: 0.85444** **PCC: 0.99223**	**PSNR: 20.1440 dB** **SSIM: 0.88409** **PCC: 0.99711**
With noise			
Sapiro's	PSNR: 13.8262 dB SSIM: 0.21423 PCC: 0.39036	PSNR: 13.9075 dB SSIM: 0.21702 PCC: 0.38562	PSNR: 13.4711 dB SSIM: 0.19750 PCC: 0.33372
Sklearn's	PSNR: 14.6909 dB SSIM: 0.41227 **PCC: 0.78210**	PSNR: 15.9465 dB SSIM: 0.39721 **PCC: 0.76703**	**PSNR: 16.2722 dB** **SSIM: 0.39007** **PCC: 0.76122**
Schölkopf's	**PSNR: 15.8020 dB** **SSIM: 0.50728** PCC: 0.76408	**PSNR: 16.3494 dB** **SSIM: 0.42270** PCC: 0.68424	PSNR: 14.7307 dB SSIM: 0.35094 PCC: 0.63071

Table 5.12: Experiment 4: Highest reconstruction performance metrics (PSNR in dB, SSIM, PCC) on the SVHN dataset processed with noise trimming and reconstruction normalization for each principal-component count and noise condition, indicating kernel and algorithm.

	PC = 20	**PC = 120**	**PC = 220**
Without noise	PSNR: 16.9369 dB (cos, Schölkopf) SSIM: 0.64507 (cos, Schölkopf) PCC: 0.86238 (cos, Schölkopf)	PSNR: 31.7435 dB (rbf, Schölkopf) SSIM: 0.91557 (rbf, Schölkopf) PCC: 0.99223 (cos, Schölkopf)	PSNR: 31.1285 dB (rbf, Schölkopf) SSIM: 0.97229 (rbf, Schölkopf) PCC: 0.99711 (cos, Schölkopf)
With noise	PSNR: 17.3959 dB (rbf, Schölkopf) SSIM: 0.50728 (cos, Schölkopf) PCC: 0.78210 (cos, Sklearn)	PSNR: 19.7507 dB (rbf, Schölkopf) SSIM: 0.42549 (rbf, Schölkopf) PCC: 0.76703 (cos, Sklearn)	PSNR: 18.2819 dB (rbf, Schölkopf) SSIM: 0.39007 (cos, Sklearn) PCC: 0.76122 (cos, Sklearn)

5.4.5 Data Analysis

Based on Tables 5.1-5.11, we observe distinct patterns in reconstruction quality across the three pre-image algorithms when using RBF and Cosine kernels. The best values are typeset in **bold**.

RBF kernel

Across all four experiments, Schölkopf's method consistently attains the highest PSNR and SSIM under noise-free conditions at low and moderate PC counts (e.g. PC=20 and PC=120), and it dominates at high PC counts in Experiments 3 and 4. Even in the presence of noise, Schölkopf's algorithm typically yields the best PSNR and SSIM, especially at smaller PC counts (e.g. Experiment 1 PC=20 and Experiment 2 PC=20), and retains strong correlation (PCC) performance in most settings. Sapiro's approach occasionally matches or exceeds Schölkopf's in specific noisy scenarios at high PC counts (e.g. Experiment 1 PC=220 and Experiment 2 PC=220), but Sklearn's pre-image reconstruction remains uniformly outperformed by the other two methods on the RBF kernel.

Cosine kernel

Performance under the Cosine kernel is more nuanced. In MNIST experiments (Experiments 1 and 2), Sapiro's method excels at low PC counts (PC=20), securing the highest PSNR and SSIM without noise, while Sklearn's overtakes PSNR at higher PCs (PC=120 and PC=220). Schölkopf's consistently achieves the best PCC at moderate to high PCs and often leads SSIM at the largest PC in Experiment 1. On natural image datasets (Experiments 3 and 4), Sklearn's method dominates PSNR and SSIM at PC=20-220 in noise-free settings, whereas Schölkopf's can edge out in PCC at specific PC settings. Under noisy conditions, Sklearn's reconstruction generally attains top PSNR and PCC at mid to high PCs (e.g. Experiments 2-3 PC=120,220), but Schölkopf's maintains superior SSIM at low PCs (e.g. Experiment 4 PC=20). This indicates that for Cosine kernels, Sklearn's inverse mapping is most robust at higher dimensional truncations, while Sapiro's and Schölkopf's may be preferable at lower PCs or when structural similarity is paramount.

Aggregated Comparison Across Experiments

Tables 5.3-5.12 summarize the highest PSNR, SSIM and PCC achieved by each algorithm in each experiment, irrespective of kernel choice. Overall, Schölkopf's method dominates PSNR across all four experiments under both noise-free and noisy conditions, and it also secures the highest SSIM in three of the four experiments (Experiments 1, 3 and 4). Sapiro's

algorithm occasionally attains the top SSIM at low PC counts in Experiment 2, and it leads PSNR in Experiment 1 at PC=220 under noise-free data. Sklearn's inverse mapping only outperforms the others in PCC, notably in the noisy condition of Experiment 4, indicating strong linear correlation in that specific setting. These aggregated results confirm that Schölkopf.

Table 5.13: Aggregated reconstruction performance metrics (PSNR in dB, SSIM, PCC) without noise on four datasets: MNIST_Ori (original MNIST, Experiment 1), MNIST_Norm (MNIST with reconstruction normalization, Experiment 2), CIFAR10_Norm (CIFAR-10 with reconstruction normalization, Experiment 3), SVHN_Norm (SVHN with reconstruction normalization, Experiment 4); for each algorithm the highest metric among all principal-component counts and kernels is reported, with kernel and PC count indicated in parentheses.

	Sapiro	**Sklearn**	**Scholkopf**
MNIST_Ori	**PSNR: 34.2829 dB** (rbf, PC = 220) **SSIM: 0.87190** (rbf, PC = 220) PCC: 0.95176 (rbf, PC = 220)	PSNR: 19.2051 dB (rbf, PC = 220) SSIM: 0.67285 (rbf, PC = 220) PCC: 0.92504 (rbf, PC = 220)	PSNR: 29.2248 dB (rbf, PC = 20) SSIM: 0.81090 (rbf, PC = 20) **PCC: 0.99487** (cos, PC = 220)
MNIST_Norm	**PSNR: 30.4618 dB** (rbf, PC = 220) SSIM: 0.59500 (rbf, PC = 120) PCC: 0.95176 (rbf, PC = 220)	PSNR: 21.1803 dB (rbf, PC = 220) **SSIM: 0.66136** (rbf, PC = 220) PCC: 0.92504 (rbf, PC = 220)	PSNR: 28.1679 dB (rbf, PC = 20) SSIM: 0.57661 (rbf, PC = 20) **PCC: 0.99487** (cos, PC = 220)
CIFAR10_Norm	PSNR: 18.7937 dB (rbf, PC = 220) SSIM: 0.66653 (rbf, PC = 220) PCC: 0.83195 (rbf, PC = 220)	PSNR: 21.5711 dB (rbf, PC = 220) SSIM: 0.85209 (rbf, PC = 220) PCC: 0.93341 (rbf, PC = 220)	**PSNR: 32.0257 dB** (rbf, PC = 20) **SSIM: 0.92645** (rbf, PC = 220) **PCC: 0.98687** (cos, PC = 220)
SVHN_Norm	PSNR: 15.0195 dB (rbf, PC = 220) SSIM: 0.62858 (rbf, PC = 220) PCC: 0.86206 (rbf, PC = 220)	PSNR: 14.7445 dB (rbf, PC = 220) SSIM: 0.68296 (rbf, PC = 220) PCC: 0.96153 (cos, PC = 220)	**PSNR: 31.7435 dB** (rbf, PC = 120) **SSIM: 0.97229** (rbf, PC = 220) **PCC: 0.99711** (cos, PC = 220)

Table 5.14: Aggregated reconstruction performance metrics (PSNR in dB, SSIM, PCC) with noise on four datasets: MNIST_Ori (original MNIST, Experiment 1), MNIST_Norm (MNIST with noise trimming and reconstruction normalization, Experiment 2), CIFAR10_Norm (CIFAR-10 with noise trimming and reconstruction normalization, Experiment 3), SVHN_Norm (SVHN with noise trimming and reconstruction normalization, Experiment 4); for each algorithm the highest metric among all principal-component counts and kernels is reported, with kernel and PC count indicated in parentheses.

	Sapiro	**Sklearn**	**Scholkopf**
MNIST_Ori	**PSNR: 18.5474 dB** (rbf, PC = 220) SSIM: 0.33622 (rbf, PC = 220) PCC: 0.83029 (rbf, PC = 220)	PSNR: 17.6918 dB (rbf, PC = 220) SSIM: 0.31541 (rbf, PC = 220) PCC: 0.82208 (rbf, PC = 220)	PSNR: 18.2186 dB (rbf, PC = 120) **SSIM: 0.48545** (cos, PC = 20) **PCC: 0.86955** (rbf, PC = 120)
MNIST_Norm	PSNR: 17.6585 dB (rbf, PC = 220) **SSIM: 0.30864** (rbf, PC = 120) PCC: 0.88384 (rbf, PC = 220)	PSNR: 17.5032 dB (rbf, PC = 220) SSIM: 0.28775 (rbf, PC = 220) PCC: 0.87774 (rbf, PC = 220)	**PSNR: 17.9833 dB** (rbf, PC = 20) SSIM: 0.29795 (cos, PC = 120) **PCC: 0.91144** (cos, PC = 20)
CIFAR10_Norm	PSNR: 16.0939 dB (rbf, PC = 220) SSIM: 0.38900 (rbf, PC = 220) PCC: 0.70653 (rbf, PC = 220)	PSNR: 18.0116 dB (cos, PC = 220) **SSIM: 0.52890** (cos, PC = 220) **PCC: 0.85676** (cos, PC = 220)	**PSNR: 19.9966 dB** (rbf, PC = 120) SSIM: 0.52633 (rbf, PC = 120) PCC: 0.83514 (rbf, PC = 120)
SVHN_Norm	PSNR: 15.5463 dB (rbf, PC = 220) SSIM: 0.29977 (rbf, PC = 220) PCC: 0.61197 (rbf, PC = 220)	PSNR: 16.2722 dB (cos, PC = 220) SSIM: 0.41227 (cos, PC = 20) **PCC: 0.78210** (cos, PC = 20)	**PSNR: 19.7507 dB** (rbf, PC = 120) **SSIM: 0.50728** (cos, PC = 20) PCC: 0.76408 (cos, PC = 20)

Aggregate Performance With and Without Noise

In the noise-free aggregated results (Table 5.13), Sapiro's algorithm achieves the highest PSNR and SSIM on the original MNIST dataset, while Schölkopf's method secures the top PCC. On normalized MNIST, Schölkopf's leads in PSNR and PCC, with Sklearn's offering the best SSIM. For CIFAR-10 and SVHN (both normalized), Schölkopf's outperforms both rivals across all three metrics. Under noisy conditions (Table 5.14), Schölkopf's again delivers the highest PSNR on three of four datasets and the best SSIM on two, whereas Sapiro's maintains superior SSIM on both MNIST variants, and Sklearn's inverse mapping produces

the strongest correlation (PCC) on CIFAR-10 and SVHN.

5.4.6 Statistical Significance Testing

To assess whether Schölkopf's algorithm truly outperforms its two competitors, for each metric, we conducted a one-way ANOVA followed by Dunnett's post-hoc test comparing Schölkopf's to each competitor, all at $\alpha = 0.05$. All computations were performed with R on **48 experimental scenarios** (4 datasets $\times$ 3 principal-component settings $\times$ 2 noise levels $\times$ 2 kernel types).

One-way ANOVA Analysis of variance (ANOVA) is a statistical method that tests for differences among more than two group means by partitioning total variance into within- and between-group components. By using a single omnibus F-test, ANOVA controls the overall Type I error rate, which would inflate if we simply ran multiple pairwise t-tests. Only after establishing that at least one group mean differs do we proceed to targeted post-hoc comparisons.

Test the global null

$$H_0 : \mu_{\text{Sapiro}} = \mu_{\text{Sklearn}} = \mu_{\text{Scholkopf}} \quad \text{vs.} \quad H_a : \text{ at least one mean differs,}$$

via one-way ANOVA at $\alpha = 0.05$. All global tests reject H_0.

Table 5.15: P-values from one-way ANOVA omnibus tests at $\alpha = 0.05$.

Metric	Global p-value
PSNR	9.5089×10^{-5}
SSIM	2.7778×10^{-3}
PCC	2.7923×10^{-7}

Post-hoc Dunnett's Test Control: **Schölkopf**. We perform one-sided Dunnett comparisons

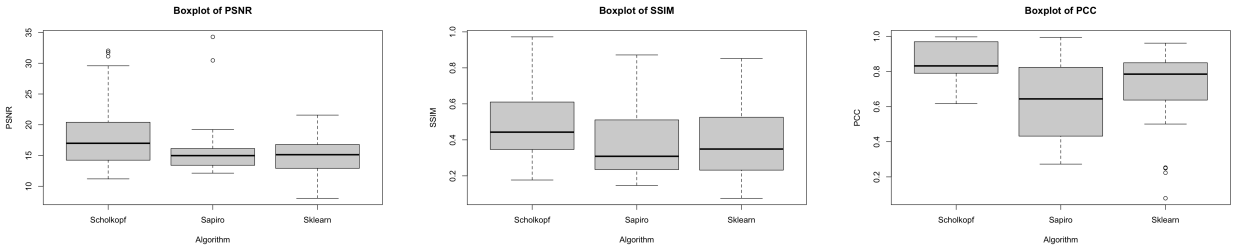
$$H_{0,j} : \mu_{\text{Scholkopf}} = \mu_j \quad H_{a,j} : \mu_{\text{Scholkopf}} > \mu_j, \quad j \in \{\text{Sapiro}, \text{Sklearn}\},$$

at $\alpha = 0.05$.

Table 5.16: One-sided Dunnett post-hoc p-values and estimated mean differences ($\widehat{\mu}_{\text{Schölkopf}} - \widehat{\mu}_j$) comparing Schölkopf against each competitor at $\alpha = 0.05$.

Metric	vs Sapiro p	$\widehat{\mu}_{\text{Schölkopf}} - \widehat{\mu}_{\text{Sapiro}}$	vs Sklearn p	$\widehat{\mu}_{\text{Schölkopf}} - \widehat{\mu}_{\text{Sklearn}}$
PSNR	9.1689×10^{-4}	2.9845	4.7501×10^{-5}	3.7047
SSIM	1.7326×10^{-3}	0.1264	4.7124×10^{-3}	0.1135
PCC	4.8945×10^{-8}	0.2106	7.7250×10^{-4}	0.1253

In all three metrics the ANOVA omnibus test yields $p < 0.05$, indicating significant differences among the methods. It is essential that each estimated mean difference $\widehat{\mu}_{\text{Schölkopf}} - \widehat{\mu}_j$ be positive in order to reject $H_{0,j}$ in favor of the one-sided alternative $H_{a,j} : \mu_{\text{Schölkopf}} > \mu_j$. A positive estimate indicates the observed data actually lie in the direction of superiority for the control (Schölkopf), whereas a negative estimate would contradict the claim of "greater than" even if the p-value were small under a two-sided test. Dunnett's post-hoc comparisons confirm that Scholköpf's algorithm achieves significantly higher mean PSNR, SSIM, and PCC than both Sapiro's and Sklearn's methods.



(a) PSNR across algorithms (b) SSIM across algorithms (c) PCC across algorithms

Figure 5.13: Boxplots of PSNR, SSIM, and PCC metrics across the three algorithms. Boxes show the interquartile range, horizontal lines indicate medians, and whiskers extend to 1.5× the interquartile range.

Figure 5.13 shows that Schölkopf's algorithm consistently achieves the highest median and narrower interquartile range on PSNR, SSIM, and PCC, while Sapiro and Sklearn exhibit larger variability and lower central tendency. These boxplots visually support the statistical findings from the ANOVA and Dunnett tests.

R code for ANOVA and Dunnett's tests

```
# 48 PSNR measurements across all test scenarios for each algorithm
Sapiro_PSNR    <- c(...)
Sklearn_PSNR   <- c(...)
```

```
Scholkopf_PSNR <- c(...)

# 48 SSIM measurements across all test scenarios for each algorithm
Sapiro_SSIM    <- c(...)
Sklearn_SSIM   <- c(...)
Scholkopf_SSIM <- c(...)

# 48 PCC measurements across all test scenarios for each algorithm
Sapiro_PCC     <- c(...)
Sklearn_PCC    <- c(...)
Scholkopf_PCC  <- c(...)

library(multcomp)
alpha <- 0.05
names <- c("PSNR","SSIM","PCC")
metrics <- list(
  list(Sapiro_PSNR,   Sklearn_PSNR,   Scholkopf_PSNR),
  list(Sapiro_SSIM,   Sklearn_SSIM,   Scholkopf_SSIM),
  list(Sapiro_PCC,    Sklearn_PCC,    Scholkopf_PCC)
)

for(i in seq_along(names)){
  cat("\nMetric:",names[i],"\n")
  cat("H0: mu_Sapiro = mu_Sklearn = mu_Scholkopf\n")
  cat("Ha: at least one mean differs\n")
  vecs <- metrics[[i]]
  groups <- rep(c("Sapiro","Sklearn","Scholkopf"),
                each=length(vecs[[1]]))
  values <- unlist(vecs)
  df <- data.frame(value=values,
                   group=factor(groups,
                   levels=c("Scholkopf","Sapiro","Sklearn")))
  model <- aov(value~group,data=df)
  pAll <- summary(model)[[1]]$"Pr(>F)"[1]
  cat(sprintf("alpha = %.2f, pValue_global = %.4e\n",alpha,pAll))
  if(pAll<alpha)
    cat("Reject H0: at least one algorithm differs\n")
  else {
```

```
    cat("Fail to reject H0: no evidence of any difference\n")
    next
  }
  boxplot(value~group,data=df,
          main=paste("Boxplot of",names[i]),
          xlab="Algorithm",ylab=names[i])
  dun <- glht(model,
              linfct=mcp(group="Dunnett"),
              alternative="less")
  dunSum <- summary(dun,test=adjusted("single-step"))
  coefs   <- dunSum$test$coefficients
  estDiff <- -coefs
  pOne    <- dunSum$test$pvalues
  competitors <- c("Sapiro","Sklearn")
  for(j in seq_along(competitors)){
    comp <- competitors[j]
    cat("\nH0: mu_Scholkopf = mu_",comp,"\n",sep="")
    cat("Ha: mu_Scholkopf > mu_",comp,"\n",sep="")
    cat(sprintf("alpha = %.2f, pValue = %.4e\n",alpha,pOne[j]))
    cat(sprintf("estimate = % .4f\n",estDiff[j]))
    if(estDiff[j]>0 && pOne[j]<alpha)
      cat("Reject H0: Scholkopf outperforms ",comp,"\n",sep="")
    else
      cat("Fail to reject H0: no evidence Scholkopf outperforms ",comp,"\n",sep="")
  }
}
```

5.4.7 Conclusion and Summary

Overall, Schölkopf's pre-image reconstruction offers the most consistent high-fidelity performance across datasets and noise conditions, excelling in PSNR, SSIM, and PCC in the majority of scenarios. While Sapiro's method may achieve the highest SSIM or PSNR on clean MNIST variants, and Sklearn's inverse mapping can yield strong PCC on CIFAR-10 and SVHN under noisy conditions, the one-way ANOVA omnibus tests (Table 5.15) and Dunnett's post-hoc comparisons (Table 5.16) confirm that Schölkopf's method outperforms both competitors at $\alpha = 0.05$ for all three metrics. These findings support selecting Schölkopf's algorithm when robustness across kernels, principal-component settings, and noise levels is required.

5.5 Inverse Kernel PCA Reconstruction via Generative Models

Let $\{x_i\}_{i=1}^N \subset \mathbb{R}^d$ denote the clean training samples and let $\tilde{x}_i = x_i + \eta_i$ be their corrupted counterparts, where η_i is random noise. Kernel PCA (kPCA) with a suitable kernel $k(\cdot,\cdot)$ implicitly defines a feature map $\varphi : \mathbb{R}^d \to \mathcal{H}$ and projects each noisy sample $\tilde{x}_i$ onto the first n principal components in feature space. Denote by

$$z_i \;=\; \Psi(\tilde{x}_i) \;=\; \big[\langle \varphi(\tilde{x}_i),\, v_1\rangle,\; \ldots,\; \langle \varphi(\tilde{x}_i),\, v_n\rangle\big]^\top \in \mathbb{R}^n$$

the latent representation obtained by kPCA, where $\{v_j\}_{j=1}^n$ are the top n eigenvectors of the centered kernel matrix. The classical pre-image problem seeks

$$x_i^* \;=\; \arg\min_{x\in\mathbb{R}^d} \big\|\varphi(x) - \Phi_i\big\|_{\mathcal{H}}^2, \qquad \Phi_i \;=\; \sum_{j=1}^{n} \langle \varphi(\tilde{x}_i),\, v_j\rangle\, v_j,$$

but no closed-form inverse φ^{-1} exists. To address this, we propose to learn a parametric mapping

$$G_\theta : \mathbb{R}^n \;\longrightarrow\; \mathbb{R}^d, \quad G_\theta(z_i) \approx x_i,$$

where θ are the network parameters. In particular, G_θ is implemented as a convolutional generator network, trained under an adversarial framework to approximate the kPCA pre-image of each noisy latent code z_i.

5.5.1 Adversarial Training Framework

Let p_{data} be the empirical distribution of the clean data $\{x_i\}$, and let p_z be the distribution of the latent codes $\{z_i\}$ produced by kPCA on noisy samples. We introduce a discriminator network

$$D_\phi : \mathbb{R}^d \;\longrightarrow\; [0,1], \qquad \phi \text{ are discriminator parameters,}$$

which aims to distinguish real samples $x_i \sim p_{\text{data}}$ from generated reconstructions $G_\theta(z)$ with $z \sim p_z$. The generator G_θ is then trained to fool D_ϕ while simultaneously minimizing a pixel-wise reconstruction error. Concretely, the adversarial loss and reconstruction loss are defined as follows.

Generator Loss. The generator G_θ minimizes

$$\mathcal{L}_G(\theta,\phi) \;=\; -\,\mathbb{E}_{z\sim p_z}\Big[\log D_\phi\big(G_\theta(z)\big)\Big] \;+\; \lambda\,\mathbb{E}_{(z,x)\sim p_{\text{pair}}}\Big[\,\|\,G_\theta(z) \;-\; x\|_2^2\Big],$$

where $\lambda > 0$ is a trade-off parameter and p_{pair} is the joint distribution pairing each latent code z with its corresponding clean sample x. The first term encourages $G_\theta(z)$ to lie on the manifold of real images, and the second term enforces fidelity to the true pre-image in the Euclidean sense.

Discriminator Loss. The discriminator D_ϕ is trained to solve

$$\mathcal{L}_D(\phi, \theta) \;=\; -\,\mathbb{E}_{x\sim p_{\text{data}}}\big[\log D_\phi(x)\big] \;-\; \mathbb{E}_{z\sim p_z}\Big[\log\big(1 - D_\phi\big(G_\theta(z)\big)\big)\Big].$$

At each iteration, ϕ is updated by descending $\nabla_\phi \mathcal{L}_D(\phi, \theta)$, and θ is updated by descending $\nabla_\theta \mathcal{L}_G(\theta, \phi)$. This adversarial optimization follows the original GAN framework in [19].

Wasserstein GAN Variant. In the WGAN formulation[3], the discriminator (often called the critic) D_ϕ is unconstrained in its output, and the losses become

$$\begin{aligned}
\mathcal{L}_D^{\text{W}}(\phi, \theta) \;&=\; -\,\mathbb{E}_{x\sim p_{\text{data}}}\big[D_\phi(x)\big] \;+\; \mathbb{E}_{z\sim p_z}\big[D_\phi\big(G_\theta(z)\big)\big], \\
\mathcal{L}_G^{\text{W}}(\theta, \phi) \;&=\; -\,\mathbb{E}_{z\sim p_z}\big[D_\phi\big(G_\theta(z)\big)\big] \;+\; \lambda_*\,\mathbb{E}_{(z,x)\sim p_{\text{pair}}}\Big[\|\,G_\theta(z) \;-\; x\|_2^2\Big].
\end{aligned}$$

Under appropriate Lipschitz-continuity constraints (enforced via weight clipping or gradient penalty), this yields a more stable training procedure and a meaningful notion of distance between distributions in terms of the Earth-Mover's distance.

5.5.2 Network Architecture and Training Details

Deep Convolutional GAN (DCGAN)

The DCGAN model employs a deep convolutional generator and a multilayer perceptron discriminator. All experiments use $n_{\text{samples}} = 20000$ MNIST images normalized to $[0, 1]$, corrupted by additive Gaussian noise of standard deviation $\sigma = 0.5$, then split without shuffling into 80% noisy training and 20% clean test sets.

Generator. A latent vector $z \in \mathbb{R}^{3000}$ is first mapped by a fully–connected layer to a $256 \times 7 \times 7$ feature map, followed by

$$\begin{aligned}
&\text{ConvTranspose2d}(256 \to 128,\ 4\times 4,\ \text{stride} = 2,\ \text{pad} = 1),\ \text{BatchNorm},\ \text{ReLU}, \\
&\text{ConvTranspose2d}(128 \to 64,\ 4\times 4,\ \text{stride} = 2,\ \text{pad} = 1),\ \text{BatchNorm},\ \text{ReLU}, \\
&\text{Conv2d}(64 \to 1,\ 3\times 3,\ \text{stride} = 1,\ \text{pad} = 1),\ \tanh\,.
\end{aligned}$$

Discriminator. A fully–connected network maps flattened $28 \times 28 = 784$ inputs through

linear layers of sizes 512, 256, 128 (each followed by LeakyReLU with slope 0.2) and a final sigmoid output.

Training. Batch size 128, for 1500 epochs on GPU if available. Optimizers are Adam with

$$\text{Generator: } lr = 1 \times 10^{-4},\ \beta = (0.5, 0.999),$$
$$\text{Discriminator: } lr = 2 \times 10^{-4},\ \beta = (0.5, 0.999).$$

The loss combines binary cross–entropy adversarial term and reconstruction term

$$L_G = \mathbb{E}\big[\text{BCE}(D(G(z)), 1)\big] \ + \ \lambda_{\text{recon}}\ \text{MSE}\big(G(z), x\big), \quad \lambda_{\text{recon}} = 10.$$

At each epoch, PSNR and SSIM are computed on the first 16 test samples; the model with highest average PSNR (and separately SSIM) is recorded.

Wasserstein GAN (WGAN)

The WGAN uses the same generator architecture. The discriminator (critic) differs by omitting the sigmoid activation and producing a real–valued score. Its loss is

$$L_D = \mathbb{E}\big[D(G(z))\big] \ - \ \mathbb{E}\big[D(x)\big], \qquad L_G = -\,\mathbb{E}\big[D(G(z))\big].$$

After each discriminator update, all its weights are clipped to $[-0.01, 0.01]$. We perform $n_{\text{critic}} = 5$ critic updates per generator update and adjust the Adam hyperparameters to $\beta = (0.5, 0.9)$. All other settings (batch size, λ_{recon}, number of epochs =1500, data split, noise level) remain as in the DCGAN.

5.5.3 Experimental Results

Table 5.17: Summary of best and worst 5 PSNR values (dB) per digit for DCGAN-KPCAnet after 500 training epochs on MNIST without noise. The final row reports the column-wise mean.

Digit	Best 5 PSNR (dB)					Avg Best 5	Worst 5 PSNR (dB)					Avg Worst 5
	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5		Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	
0	38.94	38.44	38.33	38.19	38.08	38.39	29.62	29.61	29.29	28.65	28.60	29.15
1	**46.03**	45.78	45.60	45.48	44.80	45.54	32.61	32.22	31.91	31.29	31.05	31.82
2	38.40	38.09	37.86	37.52	37.36	37.85	29.51	29.44	29.00	28.98	28.21	29.03
3	39.58	38.66	38.63	38.51	38.17	38.71	29.51	29.34	29.20	28.75	28.68	29.10
4	39.16	38.65	38.61	38.58	38.33	38.67	30.75	30.45	30.40	30.20	29.74	30.31
5	38.34	38.12	37.84	37.77	37.65	37.94	29.56	29.30	29.16	29.11	28.98	29.22
6	39.24	38.98	38.97	38.94	38.76	38.98	30.20	30.00	29.78	29.55	29.21	29.75
7	40.44	40.18	40.13	40.09	40.08	40.18	30.18	30.08	30.03	29.90	29.20	29.88
8	37.94	37.73	37.50	37.49	37.47	37.63	29.42	29.37	29.30	29.04	28.86	29.20
9	39.83	39.49	39.45	39.33	39.33	39.49	28.72	28.71	28.35	27.89	**27.80**	28.69
Mean	39.79	39.41	39.29	39.19	39.00	39.34	30.01	29.85	29.64	29.34	29.03	29.57

Table 5.18: Summary of best and worst 5 PSNR values (dB) per digit for WGAN-KPCAnet after 500 training epochs on MNIST without noise. The final row reports the column-wise mean.

Digit	Best 5 PSNR (dB)					Avg Best 5	Worst 5 PSNR (dB)					Avg Worst 5
	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5		Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	
0	37.63	36.84	36.83	36.77	36.35	36.88	28.64	28.54	28.46	28.42	28.04	28.42
1	**44.10**	43.37	42.79	42.60	42.41	43.06	32.41	31.69	31.32	31.30	30.92	31.53
2	37.75	36.58	36.10	36.03	35.98	36.49	28.91	28.81	28.50	28.36	27.97	28.51
3	37.18	37.17	36.79	36.69	36.60	36.89	29.06	29.05	28.98	28.96	28.75	28.96
4	38.68	37.61	37.58	37.51	37.49	37.77	30.00	29.93	29.86	29.33	29.09	29.64
5	37.17	37.05	36.89	36.40	36.31	36.77	28.78	28.32	28.29	28.28	28.11	28.36
6	38.38	37.31	37.19	37.04	36.90	37.36	28.17	27.60	27.28	27.16	**26.84**	27.41
7	39.08	38.84	38.72	38.66	38.58	38.78	29.33	28.99	28.91	28.86	28.64	28.95
8	37.03	36.80	36.63	36.51	36.43	36.68	28.45	28.13	28.10	28.04	28.04	28.15
9	39.16	39.06	38.98	38.65	38.52	38.87	28.89	28.85	28.85	28.25	27.64	28.50
Mean	38.62	38.06	37.85	37.69	37.56	38.00	29.26	28.99	28.86	28.70	28.40	28.84

Table 5.19: Summary of best and worst 5 PSNR values (dB) per digit for DCGAN-KPCAnet after 500 training epochs on MNIST with noise. The final row reports the column-wise mean.

Digit	Best 5 PSNR (dB)					Avg Best 5	Worst 5 PSNR (dB)					Avg Worst 5
	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5		Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	
0	18.89	18.72	18.56	18.49	18.39	18.61	13.73	13.69	13.66	13.64	13.35	13.61
1	**20.62**	20.14	19.71	19.66	19.50	19.93	12.44	12.40	12.31	12.22	**12.21**	12.32
2	18.48	18.40	18.15	17.96	17.89	18.18	12.84	12.81	12.73	12.71	12.71	12.76
3	18.69	18.41	18.23	18.08	17.89	18.26	13.34	13.29	13.23	13.15	13.11	13.22
4	19.06	18.30	18.15	18.12	18.04	18.33	13.34	13.30	13.29	13.23	13.16	13.26
5	18.53	18.20	18.00	17.99	17.87	18.12	13.63	13.58	13.58	13.58	13.58	13.59
6	18.87	18.85	18.44	18.28	18.09	18.51	13.17	13.17	13.10	12.98	12.93	13.07
7	20.07	19.93	19.77	19.72	19.70	19.84	13.35	13.29	13.27	13.23	13.22	13.27
8	17.96	17.72	17.59	17.56	17.39	17.64	12.72	12.71	12.68	12.61	12.54	12.65
9	19.46	18.79	18.55	18.44	18.39	18.73	13.35	13.29	13.19	13.18	13.18	13.24
Mean	19.15	18.75	18.51	18.42	18.25	18.62	13.09	13.06	13.01	12.92	12.90	13.00

Table 5.20: Summary of best and worst 5 PSNR values (dB) per digit for WGAN-KPCAnet after 500 training epochs on MNIST with noise. The final row reports the column-wise mean.

Digit	Best 5 PSNR (dB)					Avg Best 5	Worst 5 PSNR (dB)					Avg Worst 5
	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5		Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	
0	33.99	33.05	32.85	32.68	32.44	32.80	26.95	26.94	26.90	26.74	26.62	26.83
1	**37.88**	37.31	37.20	37.20	37.16	37.35	30.27	30.10	30.06	30.00	29.57	30.00
2	33.38	33.31	33.10	32.94	32.88	33.12	27.14	26.69	26.28	26.13	26.12	26.46
3	33.25	33.21	33.20	33.07	33.05	33.16	27.23	27.16	26.97	26.77	26.74	26.97
4	33.89	33.57	33.55	33.49	33.46	33.59	28.17	28.12	28.04	28.02	27.94	28.06
5	33.45	33.31	33.10	33.05	32.95	33.17	27.21	27.20	27.13	27.11	26.66	27.06
6	34.64	34.02	33.84	33.71	33.67	33.98	27.09	27.00	26.89	26.79	26.75	26.90
7	35.20	35.04	34.79	34.64	34.59	34.85	27.14	27.11	26.96	26.62	**26.40**	26.85
8	33.49	33.03	33.02	32.87	32.85	33.05	27.23	27.22	27.22	27.21	27.03	27.18
9	35.56	35.30	34.49	34.34	34.08	34.76	27.55	27.50	27.49	27.44	27.14	27.42
Mean	34.48	34.01	33.81	33.61	33.47	33.88	27.78	27.60	27.50	27.42	27.20	27.50

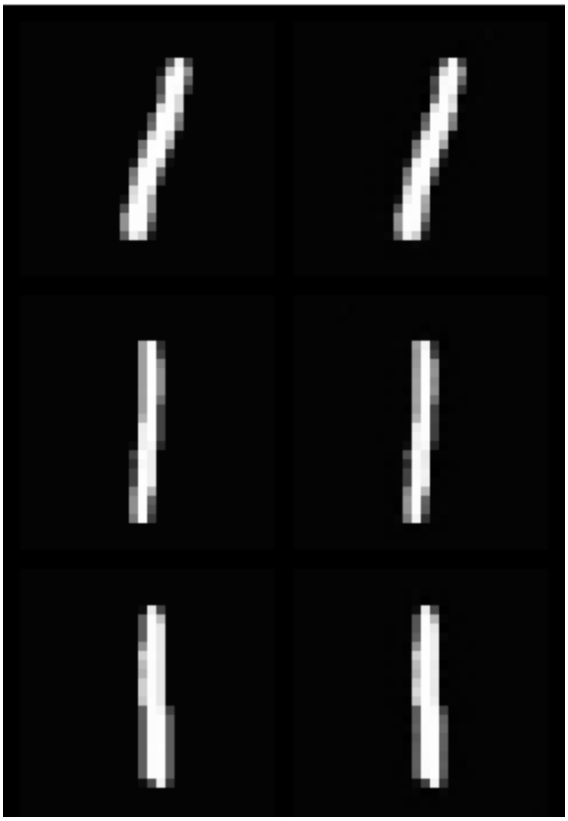
Quantitative Evaluation. Tables 5.17 and 5.18 report the best and worst 5 PSNR values per digit for DCGAN-KPCAnet and WGAN-KPCAnet after 500 training epochs on MNIST, evaluated both with and without added noise.

Without noise, DCGAN-KPCAnet yields the highest average PSNR among the best 5 reconstructions, reaching 39.34 dB, followed by WGAN-KPCAnet at 38.00 dB. In the worst cases, DCGAN-KPCAnet again leads at 29.57 dB, compared to 28.84 dB for WGAN-KPCAnet. Thus, under ideal conditions, DCGAN and WGAN achieve better peak and minimum reconstruction quality.

With noise, both methods degrade. WGAN-KPCAnet performs best, averaging 33.88 dB in the best 5 and 27.50 dB in the worst 5, while DCGAN-KPCAnet yields 18.62 dB and

13.00 dB. These results confirm that WGAN-KPCAnet maintains superior reconstruction quality under corruption.

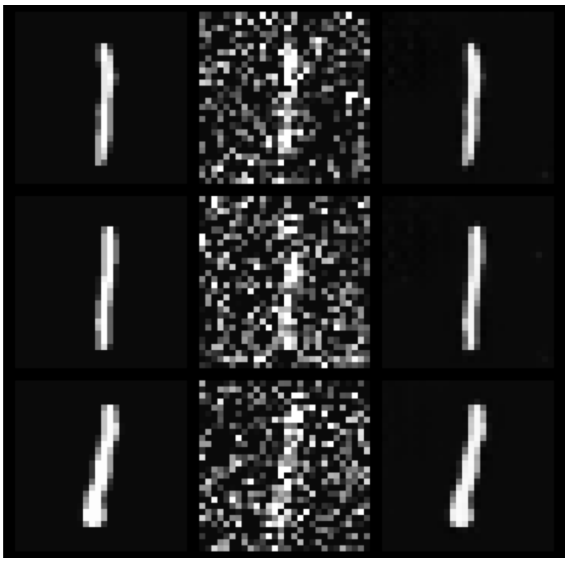
In summary, WGAN-KPCAnet offers the most balanced performance, preserving high reconstruction quality with and without noise, whereas DCGAN-KPCAnet excels on clean inputs but is highly sensitive to corruption.



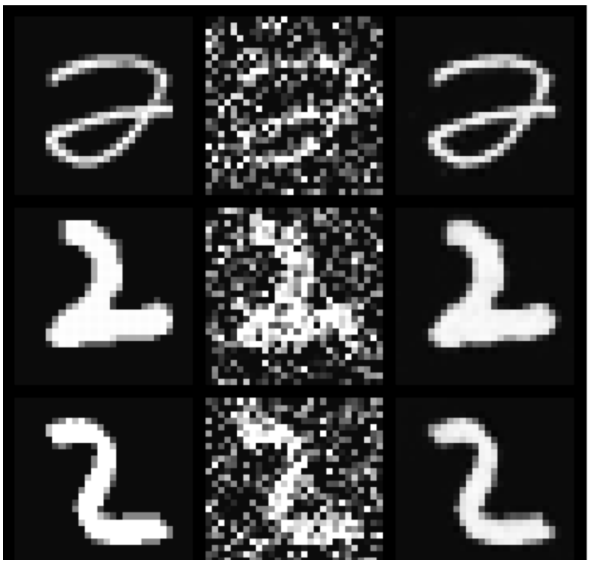
(a) The Best DCGAN-KPCAnet MNIST Reconstruction of digit 1 (without noise)

(b) The Worst DCGAN-KPCAnet MNIST Reconstruction of digit 9 (without noise)

Figure 5.14: Representative reconstructions from DCGAN-KPCAnet after 500 epochs of training on MNIST without noise. Subfigure (a) shows an example with the highest PSNR among the test samples, while (b) shows an example with the lowest PSNR.



(a) The Best WGAN-KPCAnet MNIST Reconstruction of digit 1 (with noise)



(b) The Worst WGAN-KPCAnet MNIST Reconstruction of digit 2 (with noise)

Figure 5.15: Representative reconstructions from WGAN-KPCAnet after 500 epochs of training on MNIST with added noise. Subfigure (a) shows an example with the highest PSNR among the test samples, while (b) shows an example with the lowest PSNR.

The figures above provide representative examples of the reconstruction quality achieved by the proposed GAN-based kPCA inverse solvers under different training conditions. Figure 5.14 shows DCGAN-KPCAnet reconstructions without added noise. Subfigure (a) illustrates a test sample with the highest PSNR, where the reconstruction closely matches the original clean digit. In contrast, subfigure (b) highlights one of the poorest-performing reconstructions, which exhibits shape distortion and structural loss, particularly in complex digits.

Figure 5.15 presents the corresponding results for WGAN-KPCAnet trained with noisy data. Despite the corrupted inputs, the best-case reconstruction in subfigure (a) demonstrates effective denoising and shape recovery. However, in the worst-case example (subfigure b), residual noise and artifacts remain visible, leading to degradation in digit clarity.

These examples visually confirm the quantitative trends observed in PSNR metrics: while both models can approximate the kPCA preimage, WGAN-KPCAnet exhibits greater resilience to input noise and yields more stable reconstructions.

5.5.4 Comparison with Schölkopf's Pre-image Algorithm and Summary

Table 5.21 presents the per-digit and overall average PSNR values for WGAN-KPCAnet (cosine kernel, PC=300) and Schölkopf's pre-image algorithm (RBF kernel, PC=90) on MNIST reconstructions using 400 input samples.

Table 5.21: Per-digit and overall average PSNR (dB) for WGAN-KPCAnet and Schölkopf's pre-image algorithm with 400 input samples.

Digit	WGAN-KPCAnet	Schölkopf
0	21.31	16.22
1	24.82	18.22
2	20.26	16.03
3	20.94	16.38
4	21.19	16.52
5	20.66	16.25
6	21.19	16.44
7	22.12	16.86
8	20.42	16.40
9	21.70	16.94
Overall	21.46	16.66

Both algorithms are evaluated on the same set of 400 MNIST inputs. WGAN-KPCAnet employs a cosine kernel with PC=300, leveraging a high component count and larger feature dimension to maximize reconstruction fidelity. Schölkopf's pre-image algorithm uses an RBF kernel with PC=90 (25% of the sample size), which was found optimal for classical kernel-based recovery under low-sample conditions.

As shown in Table 5.21, WGAN-KPCAnet achieves an overall average PSNR of 21.46 dB, outperforming Schölkopf's algorithm by 4.80 dB. The largest per-digit improvement occurs for digit 1 (+6.60 dB), and even the smallest gain for digit 7 (+1.24 dB) confirms consistent superiority across all classes.

Notably, WGAN-KPCAnet not only surpasses the classical algorithm in the high-dimensional reconstruction regime (PC=300), where Schölkopf's method tends to degrade on larger, heterogeneous sample sets, but also maintains superior PSNR performance in the lower-dimensional setting (PC=90). This dual advantage underscores the ability of generative inverse solvers to deliver robust, high-fidelity reconstructions across both low- and high-dimensional feature spaces.

In the preceding section, Schölkopf's algorithm was demonstrated to be the leading classic

solver in PSNR, SSIM, and PCC. The present comparison extends that analysis by showing that neural-network-based inverse methods can exceed classical performance even in low-sample, modest-component scenarios, and further widen the fidelity gap as feature dimension increases. Future work will systematically explore the interplay of kernel choice, component-to-sample ratio, and data heterogeneity to define optimal reconstruction strategies for both classical and generative frameworks.

Chapter 6

Kernel PCA in Electron Microscopic Image Denoising

6.1 Introduction

Prior to this work, the SINGLE pipeline for graphene-liquid-cell electron microscopy (GLC-EM) relied solely on masking and temporal frame averaging to suppress noise, which limited both spatial resolution and the ability to resolve dynamic structural changes. In our recent manuscript, "Time-resolved atomic-resolution Brownian tomography of single nanocrystals reveals size-dependent dynamics"[31] (currently under review at *Science Advances* and coauthored by Rubén Meana-Pañeda, Canran Ji, Cong T. S. Van, Hans Elmlund, Wojciech Czaja, et. al.), we introduced kernel principal component analysis (kPCA) denoising into the SINGLE pipeline, replacing the masking-and-averaging step. This modification significantly improved the signal-to-noise ratio of two-dimensional projection images and enabled high-resolution three-dimensional reconstructions of nanoparticle structures, including separation of distinct temporal states, that were previously obscured by noise.

Electron microscopy (EM) is a cornerstone technique for imaging structures at the nanometer scale. In conventional EM, specimens are examined at room temperature under high vacuum using high-energy electron beams; contrast is often enhanced via heavy-metal stains or direct-electron detectors, but noise sources such as detector readout fluctuations, beam-induced charging, and multiple scattering still degrade image quality and complicate quantitative morphology analysis.

Kernel Principal Component Analysis (kPCA) provides a nonlinear denoising framework by embedding each image vector $\mathbf{I}_i \in \mathbb{R}^d$ into a high-dimensional feature space via an implicit

mapping ϕ. A Mercer kernel $k(\mathbf{I}_i, \mathbf{I}_j)$ defines the Gram matrix K with entries

$$K_{ij} = k(\mathbf{I}_i, \mathbf{I}_j),$$

which, after centering in feature space, is spectrally decomposed:

$$K_c\, v_i = \lambda_i\, v_i.$$

Retaining the top m eigenvectors yields a low-dimensional representation in $\mathbb{R}^m$ that suppresses noise while preserving structural detail. In our implementation we use the cosine similarity kernel

$$k(x, y) = \frac{x^\top y}{\|x\|\,\|y\|},$$

chosen for its robustness to the extreme shot noise encountered in low-dose cryo-EM.

This kPCA denoising framework is applied to both modalities: in conventional EM it enhances feature visibility for morphology studies, and in cryo-EM it improves the reliability of downstream steps such as particle picking and three-dimensional reconstruction. The core challenge is the pre-image problem—recovering an image from its reduced kPCA coordinates—and in the following sections we compare several numerical pre-image algorithms in terms of reconstruction fidelity, computational cost, and noise robustness.

6.2 Kernel PCA Denoising Algorithm

6.2.1 Kernel Matrix Computation and Centering

Given a set of noisy image vectors $\{\mathbf{I}_i\}_{i=1}^n \subset \mathbb{R}^d$, we first select a Mercer kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. In this work, we employ the cosine similarity kernel:

$$k(\mathbf{I}_i, \mathbf{I}_j) \;=\; \frac{\mathbf{I}_i^\top\, \mathbf{I}_j}{\|\mathbf{I}_i\|\,\|\mathbf{I}_j\|},$$

which is robust under extremely low signal-to-noise ratios [25]. We form the $n \times n$ Gram matrix K with entries

$$K_{ij} \;=\; k(\mathbf{I}_i, \mathbf{I}_j), \quad i, j = 1, \ldots, n.$$

To perform PCA in the feature space $\mathcal{H}$, we must center the data in $\mathcal{H}$. Let $\mathbf{1}_n$ denote the $n \times n$ matrix with all entries equal to $1/n$. The centered Gram matrix K_c is

$$K_c \;=\; K \;-\; \mathbf{1}_n\, K \;-\; K\, \mathbf{1}_n \;+\; \mathbf{1}_n\, K\, \mathbf{1}_n.$$

In practice, one computes each term as follows:

$$(K_c)_{ij} \;=\; K_{ij} \;-\; \frac{1}{n}\sum_{r=1}^{n} K_{rj} \;-\; \frac{1}{n}\sum_{s=1}^{n} K_{is} \;+\; \frac{1}{n^2}\sum_{r,s=1}^{n} K_{rs}.$$

Centering ensures that the empirical mean in feature space is zero, i.e.$\sum_{i=1}^{n} \phi(\mathbf{I}_i) = \mathbf{0}$. All subsequent computations occur on K_c.

6.2.2 Eigen-decomposition and Feature Extraction

Once the centered Gram matrix K_c is formed, we solve the eigenvalue problem in $\mathbb{R}^n$:

$$K_c\,\boldsymbol{v}_i \;=\; n\,\lambda_i\,\boldsymbol{v}_i, \quad i = 1, \ldots, n,$$

where $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n \geqslant 0$ are the eigenvalues and $\boldsymbol{v}_i \in \mathbb{R}^n$ are the corresponding normalized eigenvectors (normalized so that $\|\boldsymbol{v}_i\|_2 = 1$). We then select the top m eigenpairs $\{(\lambda_i, \boldsymbol{v}_i)\}_{i=1}^{m}$.

For each image $\mathbf{I}_j$, its projection onto the i-th principal direction in feature space is

$$z_{ij} \;=\; \big\langle u_i,\, \phi(\mathbf{I}_j) \big\rangle_{\mathcal{H}} \;=\; \sum_{k=1}^{n} v_{ik}\, k\big(\mathbf{I}_k, \mathbf{I}_j\big), \quad i = 1, \ldots, m,$$

where v_{ik} is the k-th component of $\boldsymbol{v}_i$, and $u_i = \sum_{k=1}^{n} v_{ik}\,\phi(\mathbf{I}_k)$ is the i-th principal direction in $\mathcal{H}$ [40]. Collecting $\{z_{ij}\}_{i=1}^{m}$ yields the m-dimensional coordinate vector

$$\mathbf{z}_j = \big(z_{1j},\, z_{2j},\, \ldots,\, z_{mj}\big)^{\top} \;\in\; \mathbb{R}^m.$$

Since λ_i measures the variance of data along u_i, truncating to the top m components effectively removes components dominated by noise [33].

In implementation, one typically performs the following steps:

1. Compute the uncentered Gram matrix $K \in \mathbb{R}^{n \times n}$.
2. Center K to obtain K_c.
3. Perform an eigen-decomposition of K_c, yielding $\{\lambda_i, \boldsymbol{v}_i\}_{i=1}^{n}$.
4. Form the embedding coordinates $\mathbf{z}_j = (\sum_k v_{1k} K_{kj},\, \ldots,\, \sum_k v_{mk} K_{kj})^{\top}$ for $j = 1, \ldots, n$.

6.2.3 Reconstruction of Denoised Images

After embedding each noisy image $\mathbf{I}_j$ into the m-dimensional feature space via coordinates $\mathbf{z}_j$, we solve the *pre-image problem* using fixed-point iteration. Denote the projection in feature space by

$$\psi_j \;=\; \sum_{i=1}^{m} z_{ij}\, u_i \;=\; \sum_{i=1}^{m} z_{ij} \sum_{k=1}^{n} v_{ik}\, \phi(\mathbf{I}_k) \;\in\; \mathcal{H}.$$

To recover a denoised image $\widehat{\mathbf{I}}_j \in \mathbb{R}^d$ such that $\phi(\widehat{\mathbf{I}}_j)$ lies in the span of $\{u_i\}_{i=1}^m$, one uses the fixed-point formula adapted for the chosen kernel. For a general kernel k, the fixed-point iteration is

$$\mathbf{x}^{(t+1)} \;=\; \frac{\displaystyle\sum_{i=1}^{n} \beta_i^{(t)}\, \mathbf{I}_i}{\displaystyle\sum_{i=1}^{n} \beta_i^{(t)}}, \quad \beta_i^{(t)} \;=\; k\big(\mathbf{I}_i,\, \mathbf{x}^{(t)}\big),$$

with initialization $\mathbf{x}^{(0)} = \mathbf{I}_j$. Iteration proceeds until

$$\|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\| < \varepsilon,$$

where $\varepsilon > 0$ is a convergence tolerance. Although originally formulated for Gaussian kernels [33], for the cosine similarity kernel

$$k(x, y) \;=\; \frac{x^\top y}{\|x\|\, \|y\|},$$

one replaces each kernel evaluation accordingly:

$$\beta_i^{(t)} = \frac{\mathbf{I}_i^\top\, \mathbf{x}^{(t)}}{\|\mathbf{I}_i\|\, \|\mathbf{x}^{(t)}\|}.$$

Convergence is typically slower under severe noise but empirically yields satisfactory denoising when combined with appropriate choice of m.

Algorithm Summary.

1. **Form Kernel Matrix:** Compute $K_{ij} = k(\mathbf{I}_i, \mathbf{I}_j)$ for $i, j = 1, \ldots, n$.
2. **Center Gram Matrix:** Compute $K_c = K - \mathbf{1}_n K - K\mathbf{1}_n + \mathbf{1}_n K \mathbf{1}_n$.
3. **Eigen-decomposition:** Solve $K_c\, \boldsymbol{v}_i = n\, \lambda_i\, \boldsymbol{v}_i$ and retain top m eigenpairs $\{(\lambda_i, \boldsymbol{v}_i)\}_{i=1}^m$.

4. **Project to $\mathbb{R}^m$:** For each j, set

$$z_{ij} \;=\; \sum_{k=1}^{n} v_{ik}\, K_{kj}, \quad i = 1, \ldots, m.$$

5. **Fixed-Point Pre-image:** For each $\mathbf{z}_j$, initialize $\mathbf{x}^{(0)} = \mathbf{I}_j$. Iterate

$$\mathbf{x}^{(t+1)} = \frac{\sum_{i=1}^{n} \beta_i^{(t)}\, \mathbf{I}_i}{\sum_{i=1}^{n} \beta_i^{(t)}}, \quad \beta_i^{(t)} = \frac{\mathbf{I}_i^{\top}\, \mathbf{x}^{(t)}}{\|\mathbf{I}_i\|\, \|\mathbf{x}^{(t)}\|},$$

until $\|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\| < \varepsilon$, and set $\widehat{\mathbf{I}}_j = \mathbf{x}^{(t+1)}$.

6.3 Application to Nanocrystal Time Trajectories

6.3.1 Data Preprocessing and Particle-Tracking

Each nanocrystal time trajectory comprises a sequence of 2D projection images $\{\mathbf{I}_t\}_{t=1}^{T}$ acquired at high frame rate under solution-phase conditions. Prior to denoising via kPCA, we perform the following preprocessing steps, as detailed in Reboul *et al.* [39]:

First, *time-window averaging* is applied to overlapping blocks of W consecutive frames. Denote by $\mathcal{W}_k = \{\, \mathbf{I}_{(k-1)W+1}, \ldots, \mathbf{I}_{kW}\}$ the k-th window. For each window, we compute a weighted average

$$\bar{\mathbf{I}}_k = \sum_{t=(k-1)W+1}^{kW} w_t\, \mathbf{I}_t, \qquad \sum_{t=(k-1)W+1}^{kW} w_t = 1,$$

where weights w_t are determined via anisotropic motion correction and correlation-based frame scoring to mitigate beam-induced drift and motion [39]. This improves the signal-to-noise ratio (SNR) of each averaged frame $\bar{\mathbf{I}}_k$.

Next, *particle-tracking* is performed on the sequence $\{\bar{\mathbf{I}}_k\}$. Let $\{\mathbf{c}_k\} \subset \mathbb{R}^2$ denote the 2D center coordinates of the target nanocrystal in each averaged frame. We estimate each $\mathbf{c}_k$ by maximizing the cross-correlation between $\bar{\mathbf{I}}_k$ and $\bar{\mathbf{I}}_{k+1}$ over a search window, using a subpixel-refined phase-correlation algorithm [39, 42]. Simultaneously, total-variation (TV) denoising is applied to each $\bar{\mathbf{I}}_k$ to suppress high-frequency noise while preserving edge features. The TV denoised image is given by

$$\widetilde{\mathbf{I}}_k = \arg\min_{\mathbf{X}} \frac{1}{2}\, \|\mathbf{X} - \bar{\mathbf{I}}_k\|_2^2 \;+\; \mu^*\, \mathrm{TV}(\mathbf{X}),$$

where $\mathrm{TV}(\mathbf{X}) = \sum_{u,v} \sqrt{\big(X_{u+1,v} - X_{u,v}\big)^2 + \big(X_{u,v+1} - X_{u,v}\big)^2}$, and $\mu^* > 0$ is a penalty param-

eter chosen empirically. The TV-denoised, motion-corrected frames $\{\widetilde{\mathbf{I}}_k\}$ are then cropped around $\{\mathbf{c}_k\}$ to yield a stack of aligned particle views $\{\mathbf{I}_i\}_{i=1}^n$, where n is the total number of extracted views for this trajectory.

6.3.2 Parameter Selection and Implementation Details

Let each denoised, centered image $\mathbf{I}_i \in \mathbb{R}^d$ be vectorized into a $d = p \times p$ dimensional column. To construct the kernel matrix, we choose the cosine similarity kernel

$$k\big(\mathbf{I}_i, \mathbf{I}_j\big) = \frac{\mathbf{I}_i^\top \mathbf{I}_j}{\|\mathbf{I}_i\| \, \|\mathbf{I}_j\|},$$

which is robust when $\mathbf{I}_i$ contains extreme shot noise [25, 39]. In practice, each $\mathbf{I}_i$ is first normalized to zero mean and unit ℓ_2 norm prior to kernel evaluation.

Let $K \in \mathbb{R}^{n \times n}$ be the uncentered Gram matrix with entries $K_{ij} = k(\mathbf{I}_i, \mathbf{I}_j)$, and let K_c be its centered counterpart:

$$K_c = K - \frac{1}{n}\mathbf{1}_n K - \frac{1}{n} K \mathbf{1}_n + \frac{1}{n^2}\mathbf{1}_n K \mathbf{1}_n,$$

where $\mathbf{1}_n$ is the $n \times n$ matrix of all entries 1. We compute the eigenpairs $\{(\lambda_i, \boldsymbol{v}_i)\}_{i=1}^n$ of K_c satisfying

$$K_c \, \boldsymbol{v}_i = n \, \lambda_i \, \boldsymbol{v}_i, \quad \lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant 0, \quad \|\boldsymbol{v}_i\|_2 = 1.$$

We then choose the truncation dimension m by inspecting the relative eigenvalue decay $\lambda_i / \sum_{j=1}^n \lambda_j$. In our experiments, selecting m so that $\sum_{i=1}^m \lambda_i \gtrsim 0.95 \sum_{j=1}^n \lambda_j$ yielded a good trade-off between noise suppression and detail preservation.

For the pre-image fixed-point iteration (Section 5.3.1), we set the convergence tolerance $\varepsilon = 10^{-4}$ in ℓ_2 norm and initialize $\mathbf{x}^{(0)} = \mathbf{I}_j$. Empirically, fewer than 50 iterations suffice to reach convergence for each $\mathbf{I}_j$. All linear algebra routines (kernel matrix centering, eigendecomposition) are implemented using optimized BLAS/LAPACK calls on a multicore CPU. To avoid numerical instabilities when $\|\mathbf{I}_i\|$ is very small, any zero-norm image is excluded from the Gram matrix assembly.

6.3.3 Quantitative Performance Evaluation

To evaluate the efficacy of kPCA denoising on each nanocrystal time trajectory, we compute the following four structural metrics as defined in Reboul *et al.* [39]:

1. **Crystallinity Score.** At each time point t, an atomic model of the nanocrystal is

obtained via SINGLE [39]. The crystallinity score $C(t)$ measures the degree of long-range order by correlating the fitted atomic positions $\{\mathbf{r}_i(t)\}$ with an ideal reference lattice. Numerically,

$$C(t) \;=\; \frac{1}{N}\sum_{i=1}^{N}\exp\Bigl(-\bigl\|\mathbf{r}_i(t)-\mathbf{r}_i^{\mathrm{ref}}\bigr\|^2/\sigma^2\Bigr),$$

where N is the total number of atoms, $\mathbf{r}_i^{\mathrm{ref}}$ is the ideal lattice position of atom i, and σ is a tolerance parameter chosen to capture thermal fluctuations . Values of $C(t)$ near 1 indicate high crystalline order, while lower values reflect increasing disorder.

2. **Fraction of Core Atoms.** The core region of a nanocrystal comprises atoms with coordination numbers equal to the bulk lattice value. At time t, let $N_{\mathrm{core}}(t)$ denote the number of atoms whose coordination number (computed by counting neighbors within a cutoff radius r_c) matches the ideal bulk value. Then

$$F_{\mathrm{core}}(t) \;=\; \frac{N_{\mathrm{core}}(t)}{N}.$$

A decrease in $F_{\mathrm{core}}(t)$ over time signals core etching or structural disorder progressing inward from the surface.

3. **Radial Strain.** For each atom i at time t, define its radial distance from the nanocrystal center of mass $\mathbf{R}(t)$ as

$$r_i(t) \;=\; \bigl\|\mathbf{r}_i(t)-\mathbf{R}(t)\bigr\|.$$

Let $r_i(0)$ be the initial radial distance in the first frame. The radial strain for atom i is $\varepsilon_i(t) = [\,r_i(t)-r_i(0)\,]/r_i(0)$. We report the mean radial strain at time t:

$$\varepsilon_{\mathrm{rad}}(t) \;=\; \frac{1}{N}\sum_{i=1}^{N}\varepsilon_i(t).$$

Positive values of $\varepsilon_{\mathrm{rad}}(t)$ indicate expansion, while negative values indicate contraction relative to the initial configuration.

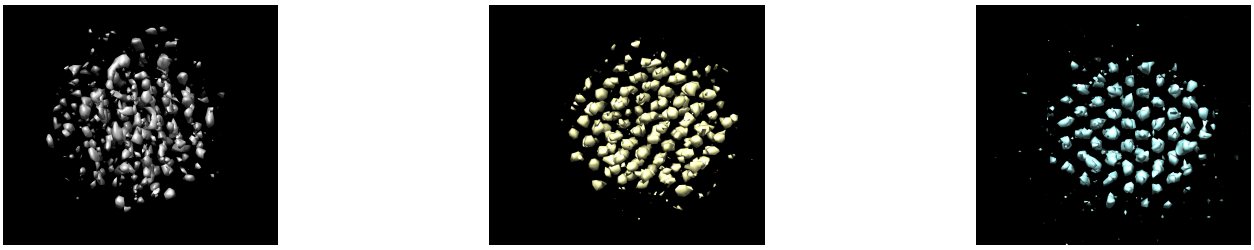
4. **Solvent Penetration Depth.** Solvent penetration depth $D_{\mathrm{solv}}(t)$ quantifies how far solvent molecules infiltrate the nanocrystal lattice. We compute the radial distance of the closest solvent-access point to the surface at time t. Formally,

$$D_{\mathrm{solv}}(t) = \max_{\mathbf{s}\in\mathcal{S}(t)}\Bigl\{\min_{i=1,\dots,N}\bigl\|\mathbf{s}-\mathbf{r}_i(t)\bigr\|\Bigr\},$$

where $\mathcal{S}(t)$ is the set of solvent voxel positions (from 3D density maps) within a threshold intensity, and $\mathbf{r}_i(t)$ are atomic positions. A larger $D_{\text{solv}}(t)$ indicates deeper solvent infiltration into the lattice.

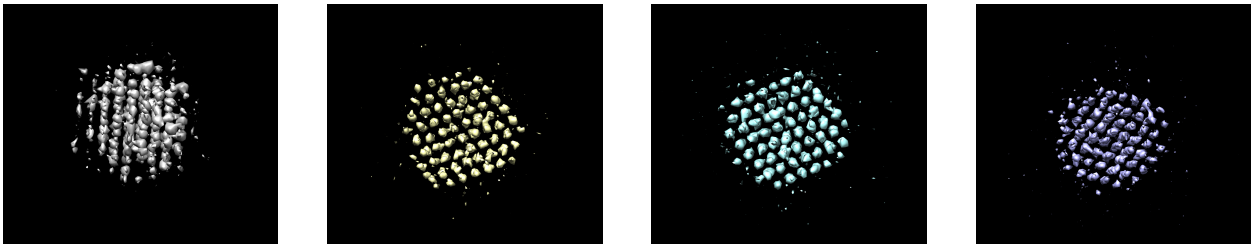
6.4 Results

6.4.1 Qualitative Reconstruction Improvement



(a) (b) (c)

Figure 6.1: NP-1 reconstructions. (a) Raw 3D map (mixture of states). (b–c) kPCA-denoised reconstructions for two temporal segments.



(a) (b) (c) (d)

Figure 6.2: NP-2 reconstructions. (a) Raw 3D map (mixture of states). (b–d) kPCA-denoised reconstructions for three temporal segments.

The raw NP-1 and NP-2 maps (Figs. 6.1a, 6.2a) each appear as a single blurred density, concealing individual morphological states. After cosine–kPCA denoising (Figs. 6.1b–c, 6.2b–d), discrete dynamical phases become clearly resolved, demonstrating that kPCA effectively suppresses noise while preserving genuine high-frequency features.

6.4.2 Time-Resolved Nanocrystal Dynamics

We carried out time-resolved atomic-resolution 3D reconstruction on fifteen nanocrystal trajectories. Each trajectory tracks a single platinum nanoparticle imaged continuously

under graphene-liquid-cell EM, yielding a series of temporally ordered 2D projections. We divided each trajectory into one to four contiguous phases according to reproducible changes in reprojection correlation, ensuring each phase had sufficient orientation coverage for an independent 3D reconstruction. These phase intervals denote windows of reliable map quality rather than intrinsic kinetic lifetimes, and high-quality maps make major structural transitions within individual phases unlikely.

Table 6.1: Nanocrystal size categories and associated trajectories

Size Category (atoms)	**Trajectories**
Small (163–285)	ESC5, ESIP1
Medium (310–469)	EFDC4, EFCD2, FD2, FD1
Large (475–698)	ESC3, EFCD1, ESC1

Key observations:

- **Small crystals (163–285 atoms):** Single etching phase; stable core morphology; 10–13
- **Medium crystals (310–469 atoms):** Multiple etching phases; fluctuating core stability with transient radial-strain peaks; occasional brief growth segments.
- **Large crystals (475–698 atoms):** Predominantly linear dissolution; stable core behavior; monotonic increases in solvent penetration depth and radial strain.

6.4.3 Conclusions

Cosine–kPCA denoising markedly enhances 3D reconstructions of time-resolved EM data by suppressing noise while preserving high-frequency detail. In both NP-1 and NP-2 examples, previously blended densities decompose into distinct morphological phases, demonstrating reliable phase separation. Across fifteen nanoparticle trajectories, we observe a clear size dependency in dissolution dynamics:

- **Small (163–285 atoms) and large (475–698 atoms) crystals** undergo predominantly linear, stable etching with minimal core fluctuations.
- **Mid-sized crystals (310–469 atoms)** exhibit non-equilibrium behaviors, including transient core destabilization, strain peaks, and occasional brief growth.

- **Cryo-EM biomolecules (SIMPLE pipeline):** kPCA denoising yields improvements in map quality, but the effect is less pronounced than in GLC-EM for nanocrystals due to the complex macromolecular architectures and residual point-spread function effects—only partially corrected by CTF—that continue to challenge 3D reconstruction.

These findings confirm that kernel PCA denoising is an effective preprocessing step for uncovering subtle, phase-dependent structural dynamics in graphene-liquid-cell EM studies and holds promise for broader application in cryo-EM.

6.4.4 Future Work

The main computational bottleneck in kPCA denoising is solving the pre-image problem via iterative fixed-point methods. To address this, we will develop and evaluate two generative models—GAN-KPCAnet and WGAN-KPCAnet—that learn the inverse kPCA mapping directly from data. In ongoing and future work we will:

- **Train end-to-end pre-image approximators:** use adversarial training on pairs of noisy inputs and kPCA-denoised outputs to replace iterative reconstruction with a single forward pass.
- **Quantify speed versus fidelity trade-off:** benchmark inference time of the trained networks against fixed-point pre-image algorithms, while measuring PSNR and SSIM on held-out test sets.
- **Incorporate noise-level conditioning:** extend the network inputs to include noise-variance estimates so that the model adapts reconstruction strength to varying shot-noise levels.
- **Generalize across kernels and PC counts:** train models that accept kernel type and number of components as inputs, enabling flexible denoising without retraining for each configuration.
- **Integrate into the SIMPLE pipeline:** embed the trained networks into particle-picking and 3D reconstruction workflows to assess end-to-end improvements in map resolution and throughput.

In summary, by replacing iterative pre-image computations with learned generative mappings, GAN-KPCAnet and WGAN-KPCAnet have the potential to dramatically accelerate kPCA denoising while maintaining or improving reconstruction fidelity, thereby enabling real-time processing in both Graphene-Liquid-Cell EM and Cryo-EM workflows.

Bibliography

[1] P. Arias, G. Randall, and G. Sapiro. Connecting the Out-of-Sample and Pre-Image Problems in Kernel Methods. volume 29, 2007. https://doi.org/10.1109/CVPR.2007.383038.

[2] S. Ö. Arık, H. Jun, and G. Diamos. Fast spectrogram inversion using multi-head convolutional neural networks. *IEEE Signal Processing Letters*, 26(1):94–98, January 2019. https://doi.org/10.1109/LSP.2018.2880284.

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. *arXiv preprint arXiv:1701.07875*, 2017.

[4] G. H. Bakir, J. Weston, and B. Schölkopf. Learning to find pre-images. In *Advances in Neural Information Processing Systems 16*, pages 449–456. Max-Planck-Gesellschaft, MIT Press, jun 2004.

[5] R. Balan, P. Casazza, and D. Edidin. On signal reconstruction without phase. *Applied and Computational Harmonic Analysis*, 20(3):345–356, May 2006. https://doi.org/10.1016/j.acha.2005.07.001.

[6] J. Caesar, C. F. Reboul, C. Machello, S. Kiesewetter, M. L. Tang, J. C. Deme, S. Johnson, D. Elmlund, S. M. Lea, and H. Elmlund. SIMPLE 3.0. Stream single-particle cryo-EM analysis in real time. *Journal of Structural Biology: X*, 4, 2020. https://doi.org/10.1016/j.yjsbx.2020.100040.

[7] J. Cahill, P. G. Casazza, and I. Daubechies. Phase retrieval in infinite-dimensional Hilbert spaces. *Transactions of the American Mathematical Society Series B*, 3:63–76, October 2016. https://doi.org/10.1090/btran/12.

[8] E. J. Candès, X. Li, and M. Soltanolkotabi. Phase retrieval via Wirtinger Flow: theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, April 2015. https://doi.org/10.1109/TIT.2015.2399924.

[9] A. Cloninger. *Exploiting Data-Dependent Structure for Improving Sensor Acquisition and Integration*. Ph.D. dissertation, University of Maryland, College Park, 2014. https://drum.lib.umd.edu/items/fc2b5e1f-6168-4e9c-adab-480a67393704.

[10] A. Cloninger and W. Czaja. LIDAR image recovery by incorporating heterogeneous imaging modalities. In *SPIE Vol. 9121, Remote Sensing*, page 91210C, 2014. https://doi.org/110.1117/12.2050714.

[11] A. Cloninger, W. Czaja, and T. Doster. The pre-image problem for Laplacian Eigenmaps utilizing l1 regularization with applications to data fusion. *Inverse Problems*, pages 123–130, 2017. https://doi.org/10.1088/1361-6420/aa5489.

[12] W. Czaja, C. Ji, S. Sule, and M. Wellershoff. Neural network-based speech reconstruction from undersampled stft magnitude data. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pages 406–410, 2024.

[13] W. Czaja, B. Manning, J. M. Murphy, and K. Stubbs. Discrete directional Gabor frames. *Applied and Computational Harmonic Analysis*, 45(1):1–21, 2018.

[14] T. J. Doster. *Harmonic Analysis Inspired Data Fusion for Applications in Remote Sensing*. Ph.D. dissertation, University of Maryland, College Park, 2014. https://drum.lib.umd.edu/items/0721dab0-0126-4ac0-afd7-afd69f4ea746.

[15] J. Emidih. *Graph-Based Data Fusion with Applications to Magnetic Resonance Imaging*. Ph.D. dissertation, University of Maryland, College Park, 2023. https://drum.lib.umd.edu/items/45bdb8f5-f732-447f-90d8-d30f4e00324a.

[16] J. Emidih and W. Czaja. Heterogeneous cancer cell line data fusion for identifying novel response determinants in precision medicine. In *Lecture Notes in Computer Science, vol. 10253, Computation, Physics and Beyond (IWCPB 2017)*, pages 414–419. Springer, 2017.

[17] H. H. Giv. Directional Short-Time Fourier Transform. *J. Math. Anal. Appl.*, 399(1):100–107, 2013. https://doi.org/10.1016/j.jmaa.2012.09.053.

[18] T. Goldstein and C. Studer. PhaseMax: convex phase retrieval via basis pursuit. *IEEE Transactions on Information Theory*, 64(4):2675–2689, April 2018. https://doi.org/10.1109/TIT.2018.2800768.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[20] L. Grafakos and C. Sansing. Gabor Frames and Directional Time-Frequency Analysis. *Appl. Comput. Harmon. Anal.*, 25(1):47–67, 2008. https://doi.org/10.1016/j.acha.2007.09.004.

[21] D. W. Griffin and J. S. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236 – 243, April 1984. https://doi.org/10.1109/TASSP.1984.1164317.

[22] P. Hand, O. Leong, and V. Voroninski. Phase retrieval under a generative prior. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, Advances in Neural Information Processing Systems, pages 9136–9146, Montréal, Canada, December 2018.

[23] T. Heinosaari, L. Mazzarella, and M. M. Wolf. Quantum tomography under prior information. *Communications in Mathematical Physics*, 318:355–374, February 2013. https://doi.org/10.1007/s00220-013-1671-8.

[24] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933. https://doi.org/10.1037/h0071325.

[25] A. M. Jade, A. K. Gupta, P. N. Raman, U. M. Diwekar, and G. L. Ray. Feature extraction and denoising using kernel pca. *Chemical Engineering Science*, 58:4441–4448, 2003.

[26] G. Klambauer, T. Unterthiner, and A. Mayr. Self-normalizing neural networks. In *31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, Advances in Neural Information Processing Systems, Long Beach, CA, USA, December 2017.

[27] J. T. Kwok and I. W. Tsang. The Pre-Image Problem in Kernel Methods. *IEEE Transactions on Neural Networks*, 15(6):1517 – 1525, 2004. https://doi.org/10.1109/TNN.2004.837781.

[28] W. Li. *Topics in Harmonic Analysis, Sparse Representations, and Data Analysis*. Ph.D. dissertation, University of Maryland, College Park, 2018. https://drum.lib.umd.edu/items/3a719f44-6be3-444e-acbb-d8360fade248.

[29] Y. Li. *Feature Extraction in Image Processing and Deep Learning*. Ph.D. dissertation, University of Maryland, College Park, 2018. https://drum.lib.umd.edu/items/0d5ee814-cff9-4b82-a5e7-a3f67b006f35.

[30] Y. Masuyama, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada. Deep Griffin–Lim iteration. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, May 2019. IEEE. https://doi.org/10.1109/ICASSP.2019.8682744.

[31] R. Meana-Pañeda, C. Ji, C. T. S. Van, C. F. Reboul, S. Kang, S. Kim, J. Heo, B. Kim, P. Ercius, W. Czaja, J. Park, and H. Elmlund. Time-resolved atomic-resolution brownian tomography of single nanocrystals reveals size-dependent dynamics. Submitted, 2025.

[32] H. Mejjaoli and S. Omri. Quantitative Uncertainty Principles associated with the Directional Short-Time Fourier Transform. *Integral Trans. and Special Fcns.*, 32(10):753–779, 2021. https://doi.org/10.1080/10652469.2020.1843166.

[33] S. Mika, B. Schölkopf, A. J. Smola, K-R. Müller, M. Scholz, and G. Rätsch. Kernel pca and de-noising in feature spaces. In *Advances in Neural Information Processing Systems 11*, pages 536–542. MIT Press, 1999.

[34] J. M. Murphy. *Anisotropic Harmonic Analysis and Integration of Remotely Sensed Data*. Ph.D. dissertation, University of Maryland, College Park, 2015. https://drum.lib.umd.edu/items/47adaf47-8906-48e6-976e-33327e860bb9.

[35] V. Panayotov, D. Povey G. Chen, and S. Khudanpur. Librispeech: an ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, South Brisbane, QLD, Australia, April 2015. IEEE. https://doi.org/10.1109/ICASSP.2015.7178964.

[36] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901. https://doi.org/10.1080/14786440109462720.

[37] N. Perraudin, P. Balazs, and P. L. Søndergaard. A fast Griffin–Lim algorithm. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, USA, October 2013. IEEE. https://doi.org/10.1109/WASPAA.2013.6701851.

[38] Y. Rathi, S. Dambreville, and A. Tannenbaum. Statistical shape analysis using kernel pca. volume 6064, page 60641B. International Society for Optics and Photonics, SPIE, 2006. https://doi.org/10.1117/12.641417.

[39] C. F. Reboul, J. Heo, C. Machello, S. Kiesewetter, B. Kim, S. Kim, D. Elmlund, P. Ercius, J. Park, and H. Elmlund. SINGLE: Atomic-resolution structure identification of nanocrystals by graphene liquid cell EM. *Science Advances*, 7(5), 2021. https://doi.org/10.1126/sciadv.abe6679.

[40] B. Schölkopf, A. Smola, and K-R. Müller. Kernel principal component analysis. pages 583–588, 1997.

[41] B. Schölkopf, A. Smola, and K-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998. https://doi.org/10.1162/089976698300017467.

[42] H. S. Stone, M. T. Orchard, E. C. Chang, and S. A. Martucci. A fast direct Fourier-based algorithm for subpixel registration of images. *IEEE Transactions on Geoscience and Remote Sensing*, 39:2235–2243, 2001.

[43] D. P. Widemann. *Dimensionality Reduction for Hyperspectral Data*. Ph.D. dissertation, University of Maryland, College Park, 2008. https://drum.lib.umd.edu/items/56603cf8-be7c-456b-aa65-3b517e43d737.

[44] H. Zhang and Y. Liang. Reshaped Wirtinger flow for solving quadratic system of equations. In *30th Conference on Neural Information Processing Systems (NIPS 2016)*, Advances in Neural Information Processing Systems, pages 2622–2630, Barcelona, Spain, December 2016.