DICTIONARY DESIGN FOR SPARSE REPRESENTATION IN COMPRESSIVE SPECTRAL IMAGING

Crisóstomo Alberto Barajas Solano

A dissertation submitted to the Department of Electrical, Electronic and Telecommunications Engineering of Universidad Industrial de Santander in fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering, Electronic Engineering Area

Advisor

Ph.D. Henry Arguello Fuentes

Universidad Industrial de Santander

Faculty of Physicomechanical Engineering

Department of Electrical, Electronic and Telecommunications Engineering

Bucaramanga

2023

To Diana, for always being there.

To my parents, for transmitting me their love for research and teaching.

To my advisor Ph.D. Henry Arguello, for having me in HDSP.

To all HDSP members, who helped me during this journey.

Contents

List of Figures

List of Tables

Resumen

Título: Diseño de Diccionarios para Representación Escasa en Sensado Espectral Comprimido [*]

Autor: Crisóstomo Alberto Barajas Solano [**]

Palabras Clave: Representación escasa, diccionarios convolucionales, imágenes espectrales.

Descripción: La Representación de diccionarios escasos convolucionales (CSDR) ha surgido como un marco robusto y flexible para representar escasamente señales de voz, escala de grises e imágenes en color. También se ha utilizado en aplicaciones médicas como imágenes de ultrasonido y ecografía, y geología. El modelo CSDR propone representar una señal como la suma de las convoluciones de una colección demasiado completa de elementos de diccionario convolucional (átomos) y mapas de coeficientes dispersos. Ambas colecciones deben cumplir una serie de restricciones. El modelo CSDR ofrece algunas ventajas interesantes frente a otros modelos de representación dispersa. Por ejemplo, el operador convolucional permite la eliminación de ruido, la invariancia de cambios, la tolerancia (hasta cierto punto) a la deformación, la rotación y la traslación. Estas propiedades hacen del CSDR un modelo interesante para su uso en imágenes espectrales compresivas (CSI). CSI establece que una imagen espectral de interés se puede recuperar a partir de un pequeño conjunto de medidas de compresión, porque un problema de optimización, con alta probabilidad, recupera la información faltante ya que se supone que los datos son escasos en algún dominio. Los métodos de última generación utilizan el modelo de representación de señal dispersa (SSR) como base de representación para recuperar la imagen espectral de tamaño completo a partir de una serie de mediciones de compresión. Este trabajo propone cambiar el modelo SSR para el marco CSDR basado en señales para aprovechar las propiedades de CSDR.

---

[*] Tesis Doctoral

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y telecomunicaciones. Director: Ph.D. Henry Arguello Fuentes

Abstract

Title: Dictionary Design for Sparse Representation in Compressive Spectral Imaging [*]

Author: Crisóstomo Alberto Barajas Solano [**]

Keywords: Sparse representation, convolutional dictionaries, spectral imaging.

Description: Convolutional Sparse Dictionary Representation (CSDR) has emerged as a robust and flexible framework for sparsely representing voice signals, gray-scale and color images. It has also been used in medical applications as ultrasound and sonography imaging, and geology. The CSDR model proposes to represent a signal as the sum of the convolutions of an overcomplete collection of convolutional dictionary elements (atoms) and sparse coefficient maps. Both collections must satisfy a series of restrictions. The CSDR model offers some interesting advantages to other sparse representation models. For example, the convolutional operator allows for noise removal, shift invariance, tolerance (to some degree) to deformation, rotation, and translation. These properties make the CSDR an interesting model for its use in compressive spectral imaging (CSI). CSI states that a spectral image of interest can be recovered from a small set of compressive measurements, because an optimization problem, with high probability, recovers the missing information since the data is assume to be sparse in some domain. The state-of-the-art methods uses the sparse signal representation model (SSR) as a representation basis in order to recover the full size spectral image from a series of compressive measurements. This work proposes to change the SSR model for the signal-based CSDR framework in order to profit on CSDR's properties.

---

[*] Doctoral Thesis, Doctorate in Engineering, Electronic Engineering Area

[**] Faculty of Physicomechanical Engineering. Department of Electrical, Electronic and Telecommunications Engineering. Advisor: Ph.D. Henry Arguello.

Introduction

Unlike a color image which divides the visible electromagnetic spectrum into three bands, or colors (i.e. red, green and blue), a spectral image (SI) divides the same spectrum into tens or hundreds of bands. Some SI even include near the infrared and ultraviolet wavelengths (Tan et al., 2016). Each spectral pixel $\mathbf{s}_{m,n} \in \mathbb{R}^L$, with $m = 1,...,M$ and $n = 1,...,N$, is considered to be unique for each physical material, allowing to identify the materials that compose a captured scene (Swamy et al., 2017). Hence, the information contained in SIs has been widely used in detection, classification and identification applications in a wide range of fields as precision agriculture, food quality monitoring and security operations (Ramirez et al., 2014). A SI is stored as a three-dimensional (3D) array $\mathbf{S} \in \mathbb{R}^{M \times N \times L}$, where $M \times N$ is the spatial resolution and L is the number of wavelength bands.

The acquisition process for SI implies expensive time and storage requirements, since the spatial and spectral coordinates have to be individually scanned and stored (Ghamisi et al., 2017). Compressive Spectral Imaging (CSI) was developed in order to reduce both the scanning time and stored measurements by capturing and compressing simultaneously a SI via multiple two-dimensional (2D) random spectral projections (Arce et al., 2014). CSI is founded on the assumption that natural SI can be represented as a lineal combination of a few waveforms in a dictionary, and therefore can be compressed. Then, a SI $\mathbf{s} \in \mathbb{R}^{MNL}$ (in its vectorized form) can be represented as a sparse collection of coefficients, $\boldsymbol{\theta} \in \mathbb{R}^{MNL}$, in some proper domain, $\boldsymbol{\Psi} \in \mathbb{R}^{MNL \times MNL}$, as $\mathbf{s} = \boldsymbol{\Psi}\boldsymbol{\theta}$ (Duarte and Baraniuk, 2012). The compressive sensing process can be modeled by means

of a compressive sensing matrix, $\mathbf{H} \in \mathbb{R}^{K \times MNL}$, where $K \ll MNL$. Then the acquisition process under the CSI framework can be modeled as $\mathbf{y} = \mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} + \omega$, where $\mathbf{y} \in \mathbb{R}^K$ are the compressive sensed measurements and $\omega \in \mathbb{R}^K$ models the sensing noise.

A reliable version of the original spectral image, $\mathbf{s}$, can be recovered from the compre- ssive sensed measurements $\mathbf{y}$ solving an inverse optimization problem (Arce et al., 2014). The inverse problem results extremely ill-posed since there exists several feasible SI that match the random projections (Candes and Wakin, 2008). To address this ill-posing problem, minimizing objectives have been explored which include regularizations related to realistic characteristics of the scene under analysis besides the data fidelity term.

In general, state-of-the-art CSI recovery methods use orthonormal analytic basis, as the Wavelet-DCT-kronecker-product basis (Arce et al., 2014) to sparsely represent a SI data cube. On the other hand, synthesis basis allow to create custom-made dictionaries which increases the sparsity level of a sparse representation. Consider the image $\mathbf{s} \in \mathbb{R}^N$ where N is the number of pixels, and the dictionary $\mathbf{D} \in \mathbb{R}^{N \times M}$, where each of the M columns is a possible image in $\mathbb{R}^N$ (Elad et al., 2010). This is, each column is an atomic image and the matrix $\mathbf{D}$ is a dictionary of atoms. The multiplication of $\mathbf{D}$ with a sparse vector $\mathbf{x}$, where $\|\mathbf{x}\|_0 = k_0 \ll M$, produces a lineal combination of k-atoms with varying weights, generating an image. The vector $\mathbf{x}$ describes which atoms and what "portions.[a]re used in the image reconstruction. This process is referred as *atom composition*.

An alternative approach to the classical overcomplete dictionaries, for sparsely repre- senting signals, is the convolutional sparse dictionary representation (CSDR) signal model. More precisely, CSDR states that a signal can be expressed as a sum of cyclic convolutions of sparse coefficient maps with an overcomplete collection of dictionary filters (Papyan et al., 2017). The main advantage of CSDR lies in the fact that the dictionary filters can be learned directly from the signal of interest, improving the reconstruction quality. Furthermore, the collection of convolutional dictionary filters can be expressed as a concatenation of banded circulant matrices, providing a local shift invariant structure and invariance to deformation (Papyan et al., 2018). These characteristics make CSDR widely used in image classification applications(Chang et al., 2014) and super resolution imaging (Gu et al., 2015).

Indeed, the CSDR signal model provides the mathematical foundation for convolutio- nal neural networks (CNN) which can automatically learn features within a signal using a hierarchical neural network, in a way similar to the process of human cognition(Liang and Li, 2016). A typical CNN is a multi-layered architecture consisting of an input, an output layer and multiple hidden layers, i.e. of a series of convolutional layers capable of assembling more complex patterns using smaller and simpler patterns through convolution operations. On the other hand, the CSDR model is a single-layer architec- ture which can be formulated as a lineal expansion equivalent to the basis pursuit (BP) scheme, as seen in (Chen et al., 2001). The only similarity between CSDR and CNN frameworks lies in the convolution operations used for representing an image.

CSDR has experienced an increment in its use thanks to recent developments of computationally effective algorithms. A particular example is the convolutional basis pursuit deNoising (CBPDN) algorithm (Wohlberg, 2016b) which has been proposed for removing noise from a stack of grayscale images. However, CBPDN expresses each grayscale image independently, missing the correlation between the color bands (when dealing with color images), leading to lower recovery qualities. A second consideration to be held is CBPDN's integration with the CSI recovery mathematical formulation. CBPDN's formulation for L independent spectral bands is hard to modify in order to integrate it within the CSI recovery formulation. Despite this, CBPDN is suitable for extension into a full 3D CSC framework which includes both spatial and spectral SI's correlation, and is suitable for integration within a CSI recovery scheme.

This work proposes to extend the 2D convolutional sparse representation to a 3D convolutional sparse representation, without deviating from the canonical CSDR formulation proposed in (Papyan et al., 2017). More precisely, we propose to represent a SI as the sum of convolutions of 3D coefficient maps with their corresponding 3D convolutional dictionary elements, or "atoms". These dictionary elements exploit both the spatial and spectral correlations within a SI, overcoming the limitations presented by approa- ches that recover every spectral band, thus leading to improved reconstruction qualities. The proposed 3D CSDR is formulated similar to a Basis Pursuit problem, as a lineal and convex formulation, and solved using the Alternating Direction Method of Multipliers (ADMM) method (Boyd et al., 2010).

The organization of the dissertation is as follows: Chapter 2 introduces the concepts of Spectral Images (SI) and the justification for Compressive Sensing Image (CSI), in addition to the concept of Sparse Signal Representation (SSR) and theoretical basis. Chapter 3 parts from the SSR concept and introduce the concept of Synthesis Basis, such as overcomplete dictionaries and its applications in classification. The chapter concludes with the introduction to the concept of Convolutional Dictionary Sparse Representation (CDSR), which serves as introduction to the next chapter. Chapter 4 contains the proposed Convolutional Sparse Representation of SIs using a 3D operator, its solution and application in denoising. This chapter also includes the application of the CSC3D framework within a CSI formulation. Finally, this chapter contains the synthetic and experimental performance evaluation of the proposed frameworks. Chapter 5 contains the natural extension of CSC3D for 4D signals such as spectral videos. This chapter includes the mathematical formulation of the proposed CSC4D framework for sparse representation and signal denoising. It also includes the extension of the proposed CSC3D-CSI into CSC4D-SR for super resolution problems and CSC4D-CSVS for compressive sensing spectral videos. Finally, this chapter contains the synthetic and experimental performance evaluation of the proposed frameworks.

1. Objectives

General Objective

To develop an algorithm to design a dictionary for sparse image representation within a compressive spectral imaging framework.

Specific Objectives

To determine the performance of the state-of-the-art dictionary design methods and compressive spectral imaging architectures.

To develop a dictionary design algorithm for obtaining both a sparse representa- tion and a dictionary to synthesize a spectral image.

To develop a dictionary design algorithm for simultaneously sparsely represent and reconstruct a spectral image within a compressive spectral imaging framework.

To evaluate the performance of the proposal dictionary design algorithm within a simulated compressive spectral imaging scheme.

To test the performance of the proposed dictionary design algorithm using data captured by experimental laboratory implementations of spectral imaging architectures.

2. Research Contributions

Most of the material presented in this doctoral dissertation appears in the following publications by the author:

Journal papers

C. Barajas-Solano, J. M. Ramirez, and H. Arguello. Convolutional Sparse Coding Framework for Compressive Spectral Imaging, Journal of Visual Communication and Image Representation, 2019, 1 (15).

C. Barajas-Solano, J. M. Ramirez, and H. Arguello. Compressive Spectral Video Sensing Using The Multidimensional Convolutional Sparse Coding Frame- work CSC4D, Journal of Visual Communication and Image Representation (submitted).

Main Conference Papers

C. Barajas-Solano, J. M. Ramirez, H. Garcia, and H. Arguello, Tridimensional convolutional sparse coding of spectral images, Optical Sensors and Sensing Congress 2019, 2019.

C. Barajas-Solano, H. Garcia, and H. Arguello, Convolutional basis pursuit denoising of spectral images using a tri-dimensional sparse representation. 2019 22nd Symposium on Image, Signal Processing and Artificial Vision, STSIVA 2019 - Conference Proceedings, 2019.

C. Barajas-Solano, J. M. Ramirez, and H. Arguello. Spectral Video Compression Using Convolutional Sparse Coding. Data Compression Conference (DCC), 2020.

This doctoral dissertation proposes a single cyclic convolutional operation for sparsely representing Spectral Images (SI) and Spectral Videos (SV), following a ND-dimensio- nal convolutional framework. The convolutional sparse representation uses a collection of small convolutional dictionary elements for representing the features within a N-dimensional signal, and a collection of sparse coefficient maps for indicating the participation of each convolutional dictionary element. The sum of all the convolutions of dictionary-coefficient pairs give as result the SI/SV of interest. The structure of the convolution operation makes the representation framework robust to noise, shifting, and deformation of the features within the represented signal.

For SIs, we propose a single 3D cyclic convolutional operator which includes all the spatial-spectral correlations of a SI without the need of additional elements. This simplifies the convex minimization to a single ℓ_2 plus a linear restriction. The results of this contribution were published in (Barajas-Solano et al., 2019a) and (Barajas-Solano et al., 2019c). For SVs, we change the 3D operator by a 4D cyclic convolutional operator. Again, this single convolutional operator includes the spatial-spectral-temporal correlations of a SV. The results of this contribution were published in (Barajas-Solano et al., 2020). The convex minimization is also a single ℓ_2 plus a linear restriction.

We propose to use a dual convex minimization scheme, with both minimizations solved alternately, for obtaining both the collection of convolutional dictionary elements and sparse coef-

ficient maps. However, given the high dimensionality of the collection of dictionary elements and sparse coefficient maps, the proposed approach has high demands on computing time and memory capacity. In this regard, this doctoral dissertation proposes a set of optimal computational routines in order to decrease the computing time and memory requirements to a fraction. The results of this contribution were published in (Barajas-Solano et al., 2019b).

The cyclic convolutional operators, 3D and 4D, have been successful in sparsely representing SIs and SVs and for applications as denoising, even for super resolution problems. However, the ND-convolutional framework was planned initially for recover- ing SIs from compressed measurements using a CSI framework. In this regard, the proposed convolutional operators have outperformed the state-of-the-art Sparse Signal Representation framework at mid and low noise levels, and matched its perfor- mance at high noise levels. However, the convolutional approach improves the sharpness in the recovered SIs and SVs. The results of this contribution were published in (Barajas-Solano et al., 2019b).

3. Spectral Images (SI), Compressive Sensing Imaging (CSI) and Sparse Signal Representation (SSR)

The purpose of this chapter is to clarify the main concepts involved in this dissertation such as spectral images, and the way how these large volume datasets can be acquired using Compressive Spectral Imaging. The chapter concludes with some state-of-the-art methods on sparse signal representation.

The notation to be used in this doctoral dissertation is introduced in Table 1 in order to facilitate the reading of the mathematical formulation.

3.1. Spectral Images (SI)

A Spectral Image (SI) captures image data within specific wavelength ranges across the electromagnetic spectrum. In comparison to an RGB image, which captures only three colors (or spectral bands) of the electromagnetic spectrum, a SI can capture dozens or hundreds of spectral bands. Therefore, a SI contains both spatial and spectral information of an scene. The latter represents the response to the absorption, or emission, of electromagnetic radiation to certain wavelengths at a given spatial coordinate (Shaw and Burke, 2003).

A SI is composed by a series of gray-scale images, with each pixel ranging from 0 to 1 that characterizes the fraction of incident light, at a given spectral wavelength band, reflected by an

Notation	Description
$M, N, L, T, d, M_d, \alpha, \beta, \gamma, \rho, \lambda, \sigma$	denote real scalars.
$\mathbf{x}$	denotes 1D arrays.
$\mathbf{X}$	denotes 2D arrays.
$\boldsymbol{\mathscr{X}}$	denotes 3D and 4D arrays.
$\|\mathbf{x}\|_0$	denotes the ℓ_0 pseudo-norm, calculated as the number of non-zero entries of the vector $\mathbf{x}$.
$\|\mathbf{x}\|_p = (\sum \|\mathbf{x}\|)^{1/p}$	denotes the ℓ_p norm for $1 \leq p \leq 2$.
$\otimes$	denotes the Kronecker product.
$\overset{n}{*}$	denotes the n-dimensional cyclic convolution operation.
$\odot$	denotes the Hadamard product.
$\mathbf{a}^t$	denotes temporal index.
$\mathbf{a}^{(j)}, \mathbf{a}^{(j+1)}$	denotes iteration step.
$\mathbf{a}^{\mathbf{T}}$ and $\mathbf{a}^{\mathbf{H}}$	denotes transpose and conjugated transpose, respectively.
$\mathbf{a}_m$	as a single sub-index, denotes the $m-th$ element of a collection.
$\mathbf{a}_{i,j}$	as a pair of sub-indexes, denotes spatial coordinates.
$[.]$	denotes horizontal concatenation.
$\text{vec}(.)$	denotes the rearrangement of an N-dimensional array into a 1D array.
$\text{diag}(\mathbf{x})$	denotes the creation of a diagonal matrix with the array $\mathbf{x} \in \mathbb{R}^N$ as its main diagonal.
$\hat{\boldsymbol{\mathscr{X}}} = \mathscr{F}_{ND}(\boldsymbol{\mathscr{X}})$ and $\boldsymbol{\mathscr{X}} = \mathscr{F}_{ND}^{-1}(\hat{\boldsymbol{\mathscr{X}}})$	denote the ND-dimensional Fourier transform and its inverse, respectively.

Table 1
Detailed list of the particular notation to be used within this doctoral dissertation.

object in a scene. In RGB imaging, the sensor pixels are covered with red, green, and blue filters, such that each pixel senses only one color, and the colorful object is obtained using techniques such as spatial deconvolution and demosaicing (Sahoo et al., 2017). In spectral imaging, the sensor simultaneously samples multiple spectral bands over a large number of spatial locations.

A SI $\mathscr{S} \in \mathbb{R}^{M \times N \times L}$ is formed by stacking all the captured spectral bands, where $M \times N$ is the spatial resolution and L is the number of spectral bands captured. Each pixel in the resulting image, $\mathbf{s}_{i,j} \in \mathbb{R}^{L}$, contains a spectral reflectance measurement and can be used to identify a material within the scene given that different materials absorb and reflect light at given wavelengths band in a unique way according to its composition (Swamy et al., 2017). This uniqueness is also called *spectral signature*. As shown in Fig. 1, each measurement is associated with a coordinate system that creates a function $s_{i,j,k}$, where i, j are the spatial coordinates and k is the spectral coordinate.

Three-dimensional spectral datacubes (i, j, k) are acquired using four basic techniques (Lu and Fei, 2014), each one with its context-dependent advantages and disadvantages:

Point scanning, where a spectral sensor (i.e. spectrophotometer) scans *point-by-point* the scene. Despite its spectral resolution, this process is time consuming and requires the scanned scene to remain still (see Fig. 2(A)).

Spatial scanning, where a slit spectra (i,k) is obtained by projecting a strip of the scene onto a slit and dispersing the slit image with a prism, or a grating, into a 2D sensor. The

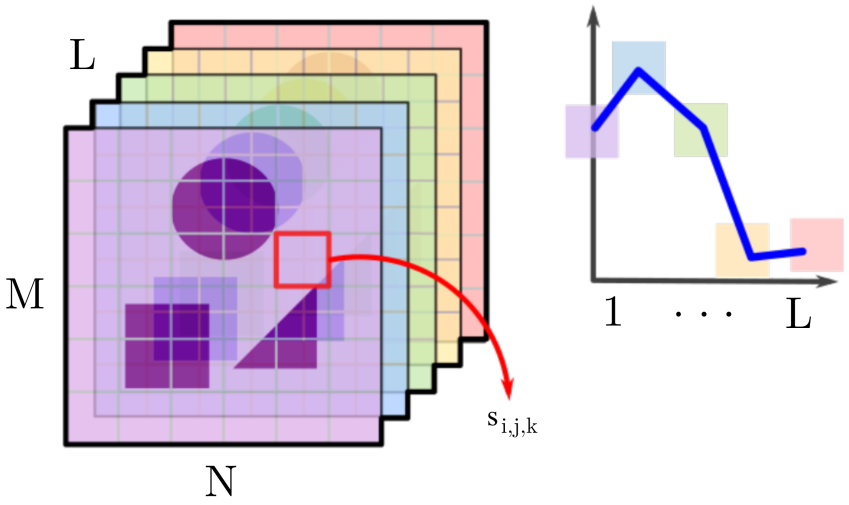


Figure 1. Spectral image scheme. A spectral pixel $\mathbf{s}_{i,j,k}$ is composed by the reflectance measurements at a wide range of wavelengths at a single spatial coordinate.

spatial dimension is collected through platform movement or scanning, with the drawbacks inherent to having mechanical parts integrated into the optical train (see Fig. 2(B)).

Spectral scanning, where optical band-pass filters (either tuneable or fixed) feeds the 2D sensor, one spectral band at the time, (i, j). The scene is spectrally scanned by exchanging one filter after another while the platform must be stationary (see Fig. 2(C)).

Non-scanning (snapshot), where the full datacube is yielded at once, without any scanning. A single snapshot represents a perspective projection of the datacube, from which its three-dimensional structure can be reconstructed. The most prominent benefits of these snapshot hyperspectral imaging systems are the snapshot advantage (higher light throughput) and shorter acquisition time (see Fig. 2(D)).

Most of the methods are related to scanning operations where multiple exposures cause

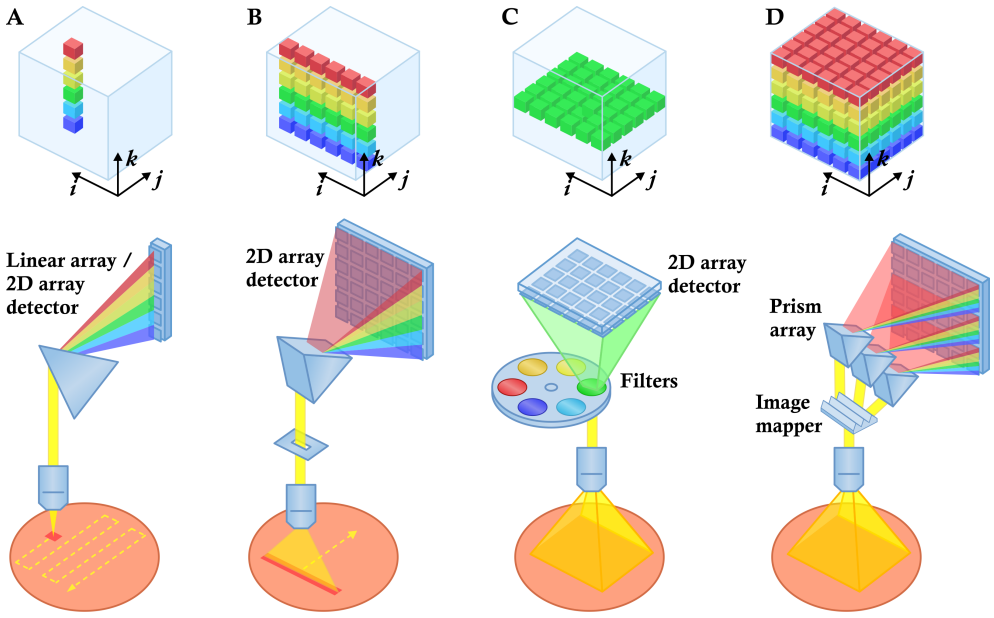


Figure 2. SI sensing techniques: (A) Point scanning. (B) Spatial scanning. (C) Spectral scanning. (D) Snapshot. Taken from (Wang et al., 2017).

motion artifacts. Then, there is a trade-off between acquisition time and the Signal to Noise Ratio (SNR): the faster each band is acquired, the fewer photons are acquired, decreasing SNR. The same problem exists in color imaging as well, but in spectral imaging, there is even lower energy per band, many more bands, and a huge amount of data that needs to be stored or transmitted.

3.2. Compressive Spectral Imaging (CSI)

Traditional SI scanning techniques follow the Nyquist/Shannon theorem (Shannon, 1949), which states that the sampling rate must be greater than twice the bandwidth of an input signal. On the other hand, Compressive Spectral Imaging (CSI) proposes to sense the spatial-spectral information of a 3D SI with a set of 2D encoded random spectral projections (Correa et al., 2014; Arguello and Arce, 2013). The success of CSI lies in that both the sensing and compressing process are carried

out simultaneously, thus reducing the number of sensed measure- ments and increasing the acquisition speed compared to traditional methods (Rueda et al., 2015). These advantages have led to CSI been applied in areas such as remote sensing (Fowler, 2014), X-ray diffraction (Greenberg et al., 2014), and biomedical imaging (Rousset et al., 2016).

CSI requires two conditions in order to make possible to recover the original spectral scene (Candes and Romberg, 2007). The first one is sparsity which requires the signal to be sparse in some domain. The second one is incoherence which is applied through the isometric property, sufficient for sparse signals.

The more known architectures used to perform CSI are the Coded Aperture Snapshot Spectral Imager (CASSI) (Arce et al., 2014), Dual Dispersive CASSI (DD-CASSI)(Gehm et al., 2007), Dual-coded Snapshot Imager (Lin et al., 2014b), the Spatial-Spectral Encoded Compressive Spectral Imager (SSCSI) (Lin et al., 2014a), and the Snapshot Colored Compressive Spectral Imager (SCCSI) (Correa et al., 2015, 2016). These architectures use an optical element named a coded aperture which let pass through or blocks the light in order to randomly encode the information. When the coded aperture codifies only the spatial dimension it can be modeled as a matrix containing 1 and 0 representing the translucent and blocking elements, respectively. When the codification is in the spatial-spectral dimensions it can be modeled as a 3D structure. Further, the encoded light is scattered into its spectral components to finally be integrated into a 2D detector. These CSI architectures can be modeled by means of a compressive sensing matrix, $\mathbf{H} \in \mathbb{R}^{MNK \times MNL}$, where

$K \ll L$ is the number of shots required to obtain the compressive sensed measurements $\mathbf{y} \in \mathbb{R}^{MNK}$, such that $\mathbf{y} = \mathbf{Hs}$ and $\mathbf{s} = \text{vect}(\mathscr{S})$

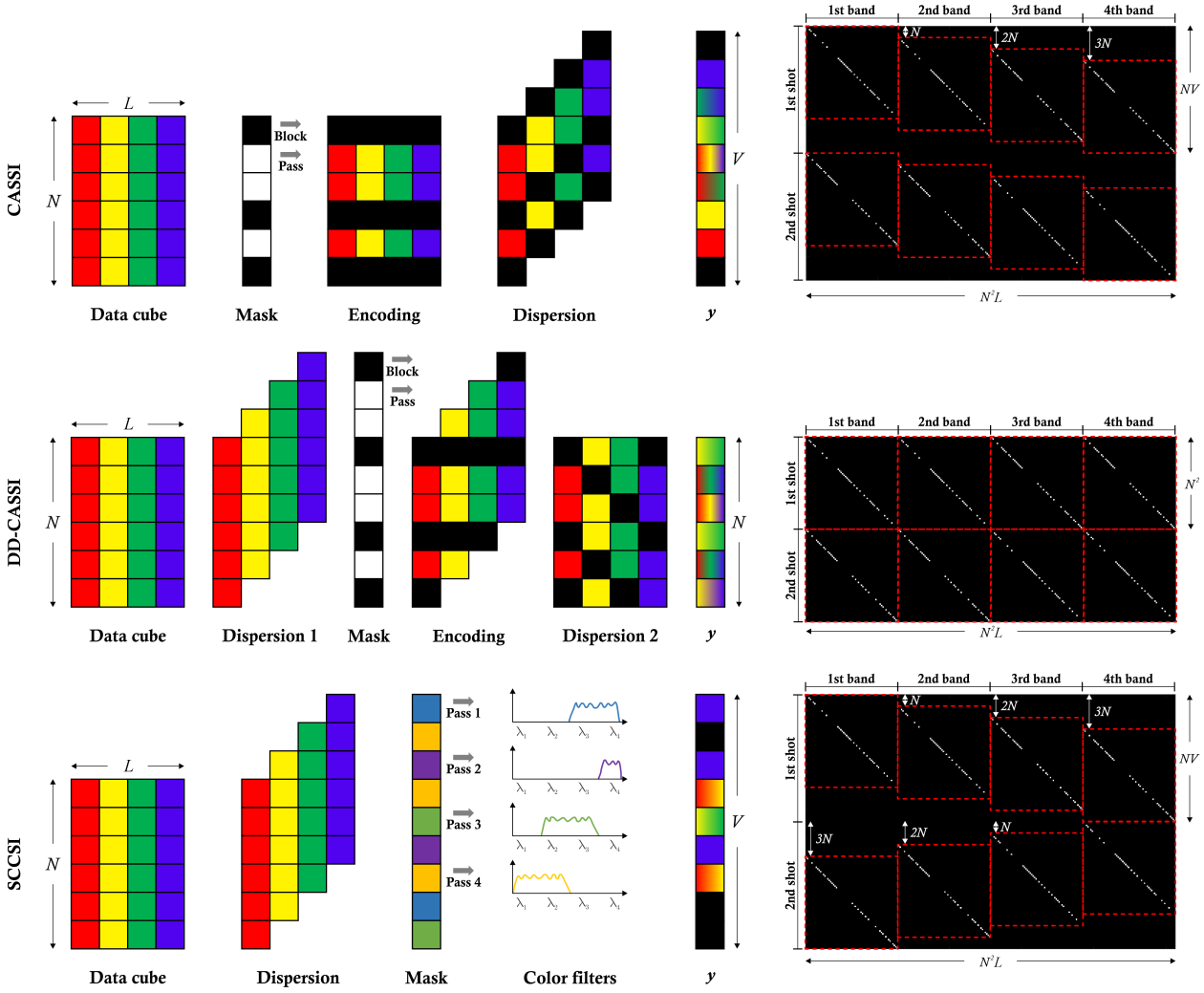


Figure 3. CASSI, DD-CASSI, and SCCSI sensing process scheme.

Figure 3 shows a top view of the sensing process for the first spatial slice and illustrates the structure of the sensing matrix $\mathbf{H}$ characterizing the CASSI, DD-CASSI, and SCCSI compressive optical architectures to acquire an SI with 6×6 spatial pixels, $L = 4$ spectral bands, and $K = 2$ snapshots. In the traditional CASSI architecture, the slice is spatially codified with a mask

containing blocking and translucent elements. The encoded light is dispersed into its spectrum components, and a sensor integrates the encoded and dispersed light. In the sensing matrix, the vectorized coded aperture is located in the diagonal and it is repeated with a N downward shift for each spectral band. The number of compressed measurements is given by $m = KN(N+L-1)$. Each snapshot is obtained by changing the spatial distribution of the mask.

In the DD-CASSI architecture, the slice is dispersed into its spectrum components. Then, the dispersed light is encoded with a mask containing blocking and translucent elements, and a second dispersion is done before the light is integrated into the sensor. The sensing matrix can be seen as the vectorized coded aperture pattern circular repeated for each spectral band. As there is a double dispersion process the number of compressed measurements is given by $m = KN^2$. Each snapshot is obtained by changing the spatial distribution of the mask.

Finally, in the SCCSI architecture, the spatial slice is decomposed into its spectral components. Then, a mask located over the sensor whose pixels contain color optical filters encodes the dispersed light in the spatial and spectral dimensions before integrate it into the sensor. In the sensing matrix, the optical filters, identified with a different color, are located in the diagonal with a N downward shift for each spectral band whereby the number of compressed measurements is given by $m = KN(N+L-1)$. Here, each snapshot is obtained by rotating the dispersive element a specified angle. In all sensing matrices, each snapshot results in a concatenation of two matrices with the structure previously described.

3.3. SI Recovery Using Sparse Signal Representation (SSR)

In order to recover a suitable version of the image of interest from compressive measurements, CSI relies in the Sparse Signal Representation (SSR) framework. According to SSR, a SI can be represented as a set of sparse coefficients on a specific basis. This is, the vectorized representation of a spectral image, $\mathbf{s} \in \mathbb{R}^{MNL}$, is S-sparse on some basis $\mathbf{\Psi} \in \mathbb{R}^{M \times M \times L}$ when only S of its coefficients are non-zero. Arce *et al.* (Arce et al., 2014) proposed to use a three-dimensional (3D) basis resulting of the Kronecker product $\mathbf{\Psi} = \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_2$, where $\mathbf{\Psi}_1$ is the two-dimensional-Wavelet Symmlet-8 basis and $\mathbf{\Psi}_2$ is the Discrete Cosine Transform (DCT) basis (Duarte and Baraniuk, 2012).

Then, $\mathbf{s}$ can be expressed as the linear product of $\mathbf{\Psi}$ with a set of coefficients $\boldsymbol{\theta} \in \mathbb{R}^{MNL}$ and $\|\boldsymbol{\theta}\|_0 = S$. However, to solve the $\|\boldsymbol{\theta}\|_0$ requires exhaustive searches over all $\boldsymbol{\theta}$, a process that has exponential complexity. Besides, the sparse recovery problem is an undetermined system of linear equations which leads the existence of uniqueness of the solution is guaranteed as soon as the signal is sufficiently sparse, and the measurement matrix satisfies the Restricted Isometry Property (RIP) at a certain level (Foucart, 2012).

Over the last decades, several sparse recovery algorithms have been proposed and can be classified into three main categories (Arjoune et al., 2017):

Convex and Relaxation Methods. This kind of algorithms solve the sparse recovery signal problem through convex relaxation problems. Example of these methods include the Basis Pursuit algorithm (Chen et al., 2001), which relaxes the ℓ_0 norm with a ℓ_1 norm, and the sparse vector $\boldsymbol{\theta}$ satisfies the equation $\boldsymbol{\Phi\theta} = \mathbf{y}$, where $\mathbf{y} \in \mathbb{R}^N$ are the measurements, as

$$\begin{aligned} &\text{argmin} \|\mathbf{x}\|_1 \\ &\text{s.t.: } \boldsymbol{\Phi}\mathbf{x} = \mathbf{y}. \end{aligned} \tag{1}$$

The Gradient Descent method (Garg and Khandekar, 2009), another example of convex and relaxation methods, is an iterative algorithm that finds the sparse solution for problem 1 where the measurement matrix $\boldsymbol{\Phi}$ satisfies the RIP with an isometric constant $\delta_{2,3} < 1/3$. This algorithm calculates iteratively a sparse signal $\mathbf{x} \in \mathbb{R}^M$ from measurements $\mathbf{y} \in \mathbb{R}^N$ using

$$\mathbf{x} = \mathbf{H}_s \left(\mathbf{x} + \frac{1}{\gamma} \boldsymbol{\Phi}^T \mathbf{r} \right), \tag{2}$$

where $\gamma = \delta_{2,3} + 1/3$, $\boldsymbol{\Phi}^T$ is the transpose of the measurement matrix, $\mathbf{r} = \mathbf{y} - \boldsymbol{\Phi}\mathbf{x}$ is the residue and $\mathbf{H}_s$ is an operator that keeps only the largest magnitude coordinates and sets all other values to zero.

Greedy Methods. Greedy methods aim to recover the sparse signal through an iterative process where each iteration makes the best local improvement to the current approximations in hope of obtaining a good overall solution. One example of greedy methods is the Orthogonal Mat-

ching Pursuit (OMP) (Tropp et al., 2006), which computes the best nonlinear approximation of sparse solution of problem 1. At each iteration, it locates the column k from the measurement matrix $\mathbf{\Phi}$ with the largest correlation residue $\mathbf{r} = \mathbf{y} - \mathbf{\Phi x}$ by taking the higher absolute value of the inner product calculated between each column and the residue as

$$k = \text{argmax}_j \left\{ \left| \mathbf{\Phi}^T \mathbf{r}_j \right| \right\}. \tag{3}$$

Then, the selected column k is appended to the set $S = S \cup \{k\}$. OMP then estimates the target variale by solving least-squares problem restricted to the columns in S and set all other components of $\mathbf{x}$ to zero by using the following formula

$$(\hat{\mathbf{x}}_i)_S = (\mathbf{\Phi}_S)^\dagger . \mathbf{y}; \; (\hat{\mathbf{x}}_i)_{S^C} = 0, \tag{4}$$

where S is the set of selected columns, S^C denotes complement of the set S, $\mathbf{\Phi}^\dagger$ denotes the pseudo-inverse of the matrix $\mathbf{\Phi}$, and $(\mathbf{\Phi}_S)^\dagger = (\mathbf{\Phi}_S^T \mathbf{\Phi}_S)^{-1} \mathbf{\Phi}_S^T$ is the pseudo-inverse of the matrix $\mathbf{\Phi}$ restricted to the set S. The residue is updated as $\mathbf{r} = \mathbf{y} - \mathbf{\Phi}\hat{\mathbf{x}}$ and the algorithm iterates by selecting a new column to be added to the set until the stopping criteria are met.

Another example of greedy methods is the Iterative Hard Thresholding proposed by Blumensath *et. al.* (Blumensath and Davies, 2009). It finds the sparse signal $\mathbf{x} \in \mathbb{R}^N$ subject to $\mathbf{y} = \mathbf{\Phi x}$ where $\mathbf{y} \in \mathbb{R}^N$ is the measurement and $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ is the measurement matrix. The solution is

updated iteratively as

$$\mathbf{x}^{n+1} = \mathbf{H}_s(\mathbf{x}^n + \mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{x}^n)), \tag{5}$$

where $\mathbf{H}_s$ is a hard thresholding operator that sets all the largest elements of $\mathbf{x}$ in term of magnitude to zero.

Bayesian Methods. Methods in the Bayesian framework solves the sparse recovery problem by taking into account a prior knowledge of the sparse signal distribution. Bayesian compressive sensing via Laplace Prior requires a definition of a joint distribution of the hierarchical model $p(\mathbf{x}, \gamma, \beta, \mathbf{y})$ (Ji et al., 2008) and is defined as

$$p(\mathbf{x}, \gamma, \beta, \mathbf{y}) = p(\mathbf{y}/\mathbf{x}, \beta).p(\mathbf{x}/\mathbf{y}).p(\gamma).p(\beta), \tag{6}$$

where γ and β are hyper parameters and the observation $\mathbf{y}$ can be described as a Gaussian distribution with zero mean and variance β^{-1}:

$$p(\mathbf{y}/\mathbf{x}, \beta) = N(\mathbf{y}/\mathbf{\Phi}\mathbf{x}, \beta^{-1}), \tag{7}$$

with a gamma prior placed on β as follows:

$$p(\beta/\alpha^{\beta}, b^{\beta}) = \Gamma(\beta/\alpha^{\beta}, b^{\beta}). \tag{8}$$

The signal model is equivalent to using a Laplacian prior on the signal $\mathbf{x}$ as

$$p(\mathbf{x}|\gamma) = \left(\frac{\gamma}{2}\right)^N \exp\left(\frac{-\gamma}{2}\|\mathbf{x}\|_1\right). \tag{9}$$

The Bayesian inference is given by

$$p(\mathbf{x},\gamma,\beta,\gamma/\mathbf{y}) = p(\mathbf{x}/\mathbf{y},\gamma,\beta,\lambda)p(\gamma,\beta,\gamma/\mathbf{y}). \tag{10}$$

Since $p(\mathbf{x}/\mathbf{y},\gamma,\beta,\lambda) \propto p(\mathbf{x},\mathbf{y},\gamma,\beta,\lambda)$, then the distribution $p(\mathbf{x}/\mathbf{y},\gamma,\beta,\lambda)$ is a multivariate Gaussian distribution $N(\mathbf{x}/\mu,\Sigma)$ with parameters $\mu = \Sigma\beta\phi^T\mathbf{y}$, $\Sigma = (\beta\phi^T\phi + \mathbf{\Lambda})^{-1}$ and $\Lambda = \text{diag}(1/\gamma_i)$. $p(\gamma,\beta,\lambda/\mathbf{y}) = [p(\gamma,\beta,\lambda,\mathbf{y})/p(\mathbf{y})] \propto p(\gamma,\beta,\lambda,\mathbf{y})$ is used to estimate the hyperparameters by maximizing the joint distribution $p(\gamma,\beta,\lambda,\mathbf{y})$ or its algorithm l

$$\begin{aligned} l =& \text{Log}(p(\gamma,\beta,\lambda,\mathbf{y})) = -\frac{1}{2}log|\mathbf{C}| - \frac{1}{2}\mathbf{y}^T\mathbf{C}^{-1}\mathbf{y} + Nlog(\lambda) - \frac{1}{2}\Sigma\gamma_i + \frac{\vartheta}{2}log\left(\frac{\vartheta}{2}\right) \\ & - log\left(\frac{\vartheta}{2}\right) + \left(\frac{\vartheta}{2} - 1\right)log(\lambda) - \frac{\vartheta}{2}\lambda + (a^\beta - 1)log(\beta) - b^\beta\beta. \end{aligned} \tag{11}$$

The updates of other parameters can be found by solving $\frac{dl}{d\lambda} = 0$ and $\frac{dl}{d\beta} = 0$. The results are given by

$$\lambda = \frac{N - 1 + \frac{\vartheta}{2}}{\Sigma_i \frac{\gamma_i}{2} + \frac{\vartheta}{2}}, \tag{12}$$

$$\beta = \frac{\frac{N}{2} + a^\beta}{\frac{\left\langle \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|^2 \right\rangle}{2}} + b^\beta. \tag{13}$$

Another probabilistic approach used to estimate the components of $\mathbf{x}$ is the Relevance Vec-

tor Machines (RVM) proposed by Badacan *et. al.* (Babacan et al., 2010). This algorithm uses a hierarchical prior to estimate a full posterior on $\mathbf{x}$ and on the variance σ^2, which defines a zero mean Gaussian prior on each element of $\mathbf{x}$. Instead of using the inverse of noise variance, RVM models the prior on $\mathbf{x}$ using the precision of a Gaussian density function α_i such that

$$p(\mathbf{x}/\alpha) = \prod_{i=1}^{N} N(\mathbf{x}_i/0, \alpha_i^{-1}), \tag{14}$$

where $N(\mathbf{x}_i/0, \alpha_i^{-1})$ denotes the Gaussian distribution with a mean equal to zero and a variance α_i^{-1}. In addition, a Gamma prior is considered over α as

$$p(\alpha/a,b) = \prod_{i=1}^{N} \Gamma(\alpha_i/a,b), \tag{15}$$

where $\Gamma(\alpha_i/a,b)$ denotes a gamma distribution. Similarly, a Gamma prior is considered over $\alpha_0 = 1/\sigma^2$ as

$$p(\alpha_0/c,d) = \prod_{i=1}^{N} \Gamma(\alpha_0/c,d). \tag{16}$$

The logarithm of the marginal likelihood can be expressed analytically as

$$\begin{aligned} \mathscr{L}(\alpha,\alpha_0) &= log(p(\mathbf{y}/\alpha,\alpha_0)), \\ &= log \int p(\mathbf{y}/\mathbf{x},\alpha_0) p(\mathbf{x}/\alpha,\alpha_0) dx, \\ &= -\frac{1}{2}\left[K log(2\pi) + log|\mathbf{C}| \, \mathbf{y}^T \mathbf{C}^{-1} \mathbf{y} \right], \end{aligned} \tag{17}$$

where $\mathbf{C} = \sigma^2\mathbf{I} + \mathbf{\Phi}\Lambda^{-1}\mathbf{\Phi}^T$, $\Lambda = diag(1/\gamma_i)$. Thus, the problem of recovering a sparse signal from few measurements in the context of Relevance Vector Machine becomes the search for the hyper parameters α and α_0.

Conclusions

The SSR model is a proven one-size-fits-all representation basis for all kind of signals, from 1D to 3D and above. The Kronecker basis proposed by Arce *et. al.* exploits the strengths of different representation basis for representing different kinds of correlations (spatial, spectral, etc). However, this size-fits-all has its limitations for representing small details such as border sharpness. In overall, the maximum reconstruction quality is limited due to the nature of the Kronecker basis itself.

The recovery methods, on the other hand, have a strong mathematical foundation, and requires strong restrictions, i.e. a convex formulation or a Bayesian approach; or require lengthy iterative process which relies in computing processing and memory capacity. Chapter 3 presents a synthetic alternative to the theoretical SSR model.

4. Sparse Dictionary Representation (SDR) and Convolutional Sparse Dictionary Representation (CSDR)

The purpose of this chapter is to clarify the concepts of sparse representation using synthetic dictionaries, and how these can represent from 1D signals to color images. The chapter concludes with the formulation of the convolutional sparse representation of N-dimensional signals, and how to profit on the DFT in order to keep such representation feasible.

Sparse representation (Bruckstein et al., 2009; Mairal and Bach, 2014) is a widely used technique for a very broad range of signal and image processing applications, such as face (Chen and Su, 2017) and pattern recognition (Wright et al., 2010), speech denoising (Jafari and Plumbley, 2011), super resolution (Yang et al., 2008), blind source separation (Li et al., 2006), and bioinformatics (Yuan et al., 2012). Given a signal $\mathbf{s}$ and an overcomplete dictionary matrix $\mathbf{D}$, sparse coding is the inverse problem of finding the sparse representation $\mathbf{x}$ with only a few non-zero entries such that $\mathbf{Dx} \approx \mathbf{s}$, as shown in Figure 4. Most sparse coding algorithms optimize a functional consisting of a data fidelity term and a sparsity inducing penalty

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{Dx} - \mathbf{s}\|_2^2 + \lambda R(\mathbf{x}), \tag{18}$$

where $R(\cdot)$ denotes a sparsity-inducing function such as the ℓ_1 norm or the ℓ_0 pseudo-norm. The two leading families of sparse coding methods are a wide variety of convex optimization algo-

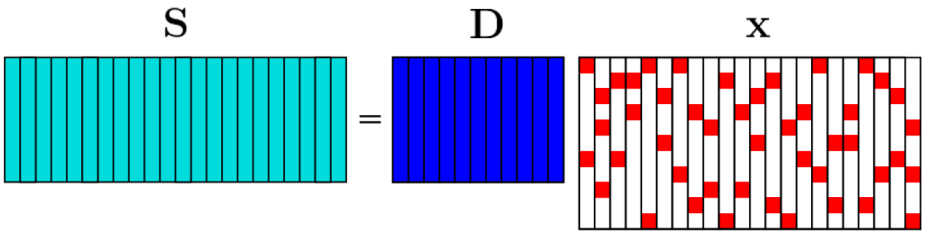


Figure 4. Sparse representation of the patches division of an image **S** using an overcomplete dictionary **D** and sparse coefficients **X**.

rithms (e.g. Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2010)) solving Eq. (18) when $R(\mathbf{x}) = \|\mathbf{x}\|_1$ and a family of greedy algorithms (e.g. Matching Pursuit (MP) (Mallat and Zang, 1993) and Orthogonal Matching Pursuit (OMP) (Pati et al., 1993)) for approximate solution when $R(\mathbf{x}) = \|\mathbf{x}\|_0$.

When applied to images, this decomposition is usually applied independently to a set of overlapping image patches covering the image; this approach is convenient, but often necessitates somewhat *ad hoc* subsequent handling of the overlap between patches, and results in a representation over the whole image that is suboptimal (Wohlberg, 2014).

4.1. Dictionary Learning and Design

Considering Eq. (18), the question then changes into how to obtain a dictionary **D** capable of representing the scene **s**. Dictionary learning techniques as MOD (Engan et al., 1999) and K-SVD (Aharon et al., 2006) are based in a iterative two stage procedure:

1. Sparse coding step: Given a fix **D**, find a sparse **x**.

2. Dictionary update step: fix **x** and update **D**.

4.1.1. Method of Optimal Directions (MOD). The Method of Optimal Directions, MOD, proposed by Engan *et al*. (Engan et al., 1999), is based on the Generalized Lloyd Algorithm, GLA (Gersho, 1992). The main steps of the algorithm are explained in Algorithm 1.

The suggested stop criteria can be: maximum number of iterations or almost constant MSE. Due to lack of guarantee for the new frame to be better than the previous, the algorithm should allow the MSE to perform iterations without terminat- ing the training.

4.1.2. K-SVD. K-SVD (Aharon et al., 2006) is an algorithm, based on the k-means clustering algorithm, aimed to learn an overcomplete dictionary $\mathbf{D} \in \mathbb{R}^{N\times K}$ that contains K signal atoms, via a singular value decomposition approach. A collection of M-length signal vectors arranged in $\mathbf{Y} \in \mathbf{R}^{N\times M}$ can be represented sparsely as a linear combinations of the atoms in **D** by solving

$$\underset{\{\mathbf{D},\mathbf{X}\}}{\operatorname{argmin}} \|\mathbf{DX}-\mathbf{Y}\|_F^2 , \tag{19}$$

with $\mathbf{X} \in \mathbf{R}^{K\times M}$ sparse column-wise. The K-SVD algorithm is summarized in Algorithm 2. K-SVD performs well for both synthetic and real images in applications such as filling in missing pixels and compression and outperforms alternatives such as the nondecimated Haar and overcomplete or unitary DCT.

Algorithm 1 MOD

Require: initial frame $\mathbf{F}_0 \in \mathbb{R}^{N\times K}$ and number of frame vectors to be used in each approximation, m. Assign counter variable $i = 1$.

1: Approximate each training vector, $\mathbf{x}_l$, using a vector selection algorithm:

$$\tilde{\mathbf{x}}_l = \sum_{k=1}^{K} w_l(k)\mathbf{f}_k \tag{20}$$

where $w_l(k)$ is the coefficient corresponding to vector $\mathbf{f}_k$, and only m of the $w_l(k)$'s are different to zero.

2: Find the residuals.

3: Given the approximations and residuals, adjust the frame vectors $\Rightarrow \mathbf{F}_i$.

4: Find the new approximations, and calculate the new residuals. If the stop criterion hasn't been reach yet, then do $i = i+1$; otherwise, stop.

Algorithm 2 K-SVD

Require: set an initial dictionary $\mathbf{D}_0$ with ℓ_2 normalized columns. Set $J = 1$.

1: *Sparse Coding Stage*: use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each sample $\mathbf{y}_i$, by approximating the solution of

$$\begin{aligned} &\underset{\{\mathbf{x}_i\}}{\operatorname{argmin}} \|\mathbf{D}\mathbf{x}_i - \mathbf{y}_i\|_2^2 \\ &\text{s.t.: } \|\mathbf{x}_i\|_0 \le T_0, \quad \forall i = 1,...,M \end{aligned} \tag{21}$$

2: *Dictionary coding stage*: for each $k = 1,...,K$ in $\mathbf{D}^{(J-1)}$, update it by:

3: Define the group of examples that use this atom, $w_k = \{i \mid 1 \le i \le N, x_T^k(i) \neq 0\}$

4: Compute the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j\neq k} \mathbf{d}_j \mathbf{x}_T^k. \tag{22}$$

5: Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to w_k, and obtain $\mathbf{E}_k^R$.

6: Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U}\mathbf{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\mathbf{\Delta}(1,1)$.

4.2. Classification Using Sparse Dictionaries

One of the earliest approaches to classification using sparse dictionaries is template matching (Scott and Nowak, 2004) where a template (or prototype) is generated and compared to the test pattern to be recognized. The metric for similarity is often a correlation measure, but if the template is modeled statistically, a likelihood measure can be used.

Given a signal $\mathbf{y}$ and a template $\mathbf{x}_p$, a measure that can be used for template matching is

$$M_p(m) = \sum_j \left|\mathbf{y}(j) - \mathbf{x}_p(j-m)\right|, \ \forall j \mid (j-m) \in \mathbf{D}, \tag{23}$$

where $\mathbf{D}$ denotes the domain of definition of the template, m indicates the amount of translation provided to the template and p identifies the class.

Thiagarajan *et al* (Thiagarajan et al., 2008) proposed a template based statistical classification frame- work in the data representation domain, where an unlabeled training data set is sparsely represented using an overcomplete dictionary. A source model is assumed, with each source using a set of dictionary elements chosen from a large dictionary, called the generating dictionary (see Fig. 5).

Chen *et al* (Chen and Su, 2017) proposed the Sparse Embedded Dictionary Learning (SEDL) for face recognition problems. Here, the training samples are composed of c classes, each vectori-

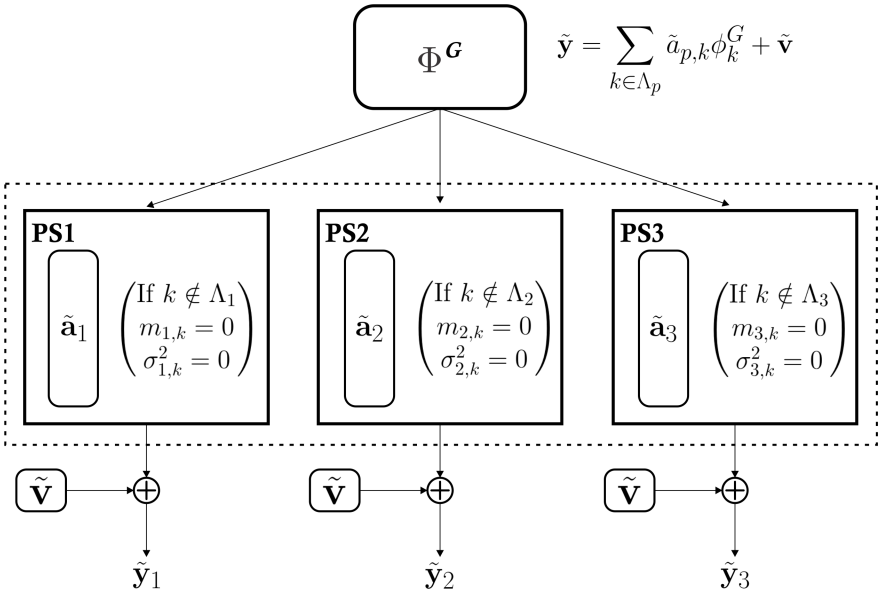


Figure 5. The probability source model. PS1, PS2, and PS3 are the sources and each source represents a class that uses a fixed set of dictionary atoms from the generating dictionary. Taken from (Thiagarajan et al., 2008).

zed as $\mathbf{y} \in \mathbb{R}m$ and the i-th class as $\mathbf{Y}_i \in [y_1, ..., y_{n_i}] \in \mathbb{R}^{m\times n_i}$, where n_i is the number of samples in the i-th class, with $i = 1, ..., c$, and the training data matrix $\mathbf{Y} = [Y_1, ..., Y_i, ..., Y_c] \in \mathbb{R}^{m\times n}$. Considering the high dimensionality of face images (Nguyen et al., 2012), dimensionality reduction aims to learn an orthogonal projection matrix $\mathbf{P} \in \mathbb{R}^{p\times m}$, where p denotes the lower dimension of data $(p < m)$.

Let $\mathbf{X} = [\mathbf{X}_1, ..., \mathbf{X}_i, ..., \mathbf{X}_c]$ denotes the coefficients of $\mathbf{Y}$ coded over dictionary $\mathbf{D}$ (see Fig. 6).The reconstruction error is then defined as

$$r(\mathbf{P}, \mathbf{Y}, \mathbf{D}, \mathbf{X}) = \|\mathbf{PY} - \mathbf{DX}\|_F^2 + \sum_{i=1}^{c} \left\|\mathbf{PY}_i - \mathbf{D}_i\mathbf{X}_i^i\right\|_F^2 + \sum_{i=1}^{c}\sum_{j=1, j\neq i}^{c} \left\|\mathbf{D}_j\mathbf{X}_i^j\right\|_F^2, \tag{24}$$

where the first term states that the dimensionality reduction $\mathbf{PY}$ can be well represented by the

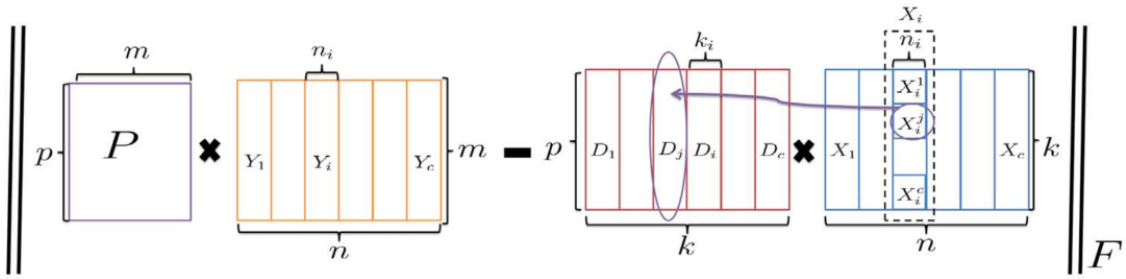


Figure 6. The relationship between variables in term $\|\mathbf{PY} - \mathbf{DX}\|_F^2$. Taken from (Chen and Su, 2017).

dictionary **D**; the second term ensures that each sub-dictionary should be able to well present the data which can enhance the discriminative ability during the classification stage and minimize the reconstruction error of source domain caused by dimensionality reduction; and the third term guarantees that the effect of other sub-dictionary is minimized.

In the case of hyperspectral images, Wang *et al* (Wang et al., 2014) propose a hinge loss function that is directly related to the classification task as the objective function for dictionary learning. The resulting online learning procedure systematically "pulls" and "pushes" dictionary atoms so that they become better adapted to distinguish between different classes. Let the data set to contain N labeled SI pixels of m spectral bands coming from C classes: $\{\mathbf{x}_i \in \mathbb{R}^m, y_i \in \{1...C\}\}$. For each class $c = 1,...,C$ there exists a dictionary $\mathbf{D}^c \in \mathbb{R}^{m \times n^c}$ of n^c atoms such that n^c is much smaller than the number of samples in class c, and any data sample in this class can be well approximated as the linear combination of a small number of active atoms selected from $\mathbf{D}^c$

$$\mathbf{x}_i \approx \mathbf{D}^c \boldsymbol{\alpha}_i^c, \ |\boldsymbol{\alpha}_i^c|_0 \leq K, \ \forall y_i = c, \tag{25}$$

where $\boldsymbol{\alpha}_i^c \in \mathbb{R}^{n^c}$ is the sparse code for pixel $\mathbf{x}_i$ with respect to the dictionary $\mathbf{D}^c$. Wang *et al* adopts an objective function so that the Learning Vector Quantization (LVQ) (Bottou, 2004) can be exploited in building a sparse dictionary classification (LSRC) as

$$\mathscr{L}_{LSRC}(\mathbf{x_i}, y_i; \mathbf{D}) = \max\left(0, r_i^{y_i} - r_i^{\hat{c}_i} + b\right), \tag{26}$$

where $\hat{c}_i$ is the most competitive class in reconstructing the signal excluding the true class y_i and b is a nonnegative parameter controlling the ?margin? between the classes. Thus, the problem of LSRC dictionary design can be formulated as

$$\mathbf{D}^* = \underset{\mathbf{D}\in\mathscr{D}}{\operatorname{argmin}} \frac{1}{N}\sum_i \mathscr{L}_{LSRC}(\mathbf{x_i}, y_i; \mathbf{D}), \tag{27}$$

which is optimized over the whole training set.

4.3. Tensorial Dictionary Sparse Representation

A notation based in tensors can effectively represent an organized multidimensional array of numerical values (Tao et al., 2007) as indicated by its order. A vector $\mathbf{a} \in \mathbb{R}^N$ is a first-order tensor, a matrix $\mathbf{A} \in \mathbb{R}^{N\times M}$ is a second-order tensor and a data cube, like a hyperspectral image, is a third-order tensor $\mathscr{A} \in \mathbb{R}^{N\times M\times L}$. In this way, the neighborhood relationship across both the spatial and spectral dimensions can be fully preserved . Recent studies have proven the effectiveness of a tensor representation on the performance of various hyperspectral applications such as: image

denoising (Liu and Tao, 2016), tensor PCA (Velasco-Forero and Angulo, 2013), and tensor neighborhood graph learning (Gao et al., 2015).

Du *et al* (Du et al., 2017) propose to split a hyperspectral image into a group of third-order tensor patches with the same size. Given a third-order tensor $\mathscr{I} \in \mathbb{R}^{L^W \times L^H \times L^S}$, it is divided into overlapping blocks of size $l^W \times l^H$ (with $l^W < L^W$ and $l^H < L^H$). Then we can reconstruct a group of 3D patches $\{\mathscr{P}_{i,j}\}_{1 \leq i \leq n^W, 1 \leq j \leq n^H} \in \mathbb{R}^{l^W \times l^H \times L^S}$ and the number of patches equal to $N = n^W n^H$. Each patch is a data cube which preserves all the spectral information from the original hyperspectral image, representing both the spatial and spectral correlations. The dictionary learning objective function can be expressed then, in terms of tensors, as

$$\begin{aligned} \underset{\mathbf{D}^W, \mathbf{D}^H, \mathbf{D}^S, \mathscr{Z}_i}{\operatorname{argmin}} & \sum_{i=1}^{N} \left\| \mathscr{P}_i - \mathscr{Z}_i \times_1 \mathbf{D}^W \times_2 \mathbf{D}^H \times_3 \mathbf{D}^S \right\|_2 \\ & \text{s.t.: } \mathscr{O}_t(\mathscr{Z}_i) \leq \mathbf{s}, \end{aligned} \tag{28}$$

where $\mathbf{D}^W \in \mathbb{R}^{l^W \times d^W}$, $\mathbf{D}^H \in \mathbb{R}^{l^H \times d^H}$ and $\mathbf{D}^S \in \mathbb{R}^{l^S \times d^S}$ (with $l^W < d^W$, $l^H < d^H$, $l^S < d^S$) are the three dictionaries, $\mathscr{Z}_i \in \mathbb{R}^{d^W \times d^H \times d^S}$ is a redundant coefficient tensor of patch $\mathscr{P}_i$, $\mathscr{O}_t(\mathscr{Z}_i) \leq \mathbf{s}$ is a generalization of the matrix sparsity in tensor form, which can constraint $\mathscr{Z}_i$ to be a sparse tensor, and $\times_j$ is the j-mode product between ta tensor and a matrix.

4.4. Convolutional Sparse Dictionary

A variation of the sparse dictionary representation is the Convolutional Sparse Dictionary Representation (CSDR), a synthesis framework for sparsely representing signals using a collection of convolutional dictionary elements and sparse coefficient maps, both learned directly from the signal of interest (Bruckstein et al., 2009). This signal specificity allows for higher reconstruction qualities.

1D CSDR. CSDR states that a given signal $\mathbf{s} \in \mathbb{R}^N$ can be represented as the sparse combination of a collection of dictionary elements $\{\mathbf{d}_m,\ m = 1,...,M_d \mid \mathbf{d}_m \in \mathbb{R}^d\}$ and its corresponding sparse coefficient maps $\{\mathbf{x}_m,\ m = 1,...,M_d \mid \mathbf{x}_m \in \mathbb{R}^N\}$, as

$$\mathbf{s} = \sum_{m=1}^{M_d} \mathbf{d}_m \overset{1}{*} \mathbf{x}_m + \boldsymbol{\omega}, \tag{29}$$

where $\boldsymbol{\omega}$ denotes a reconstruction error. It is worth noting that the sparse coefficient maps have the same dimension as $\mathbf{s}$, while the dictionary elements are much smaller, $d \ll N$. The structure of the convolution operation makes the representation frame-work robust to noise, shifting, and deformation of the features within the represented signal. These properties makes of CDSR an useful framework for denoising and machine learning (Papyan et al., 2017).

We can then replace Eq. (18) with the following formulation

$$\underset{\{\mathbf{x}_m\}}{\operatorname{argmin}} \frac{1}{2} \left\| \sum_{m=1}^{M_d} \mathbf{d}_m \overset{1}{*} \mathbf{x}_m - \mathbf{s} \right\|_2^2 + \lambda \left\| \mathbf{x}_m \right\|_1, \tag{30}$$

supposing a fixed collection of dictionary elements, and optimizing the coefficient maps.

Gray-scale Images using CSDR. For the case of a single gray-scale image $\mathbf{S} \in \mathbb{R}^{M \times N}$, Eq. (30) converts into

$$\underset{\{\mathbf{X}_m\}}{\operatorname{argmin}} \frac{1}{2} \left\| \sum_{m=1}^{M_d} \mathbf{D}_m \overset{2}{*} \mathbf{X}_m - \mathbf{S} \right\|_F^2 + \lambda \sum_{m=1}^{M_d} \left\| \mathbf{x}_m \right\|_1, \tag{31}$$

where $\overset{2}{(*)}$ represents the 2D cyclic convolution, $\{\mathbf{D}_m \in \mathbb{R}^{d \times d} | m = 1, ..., M_d\}$ is a collection of convolutional dictionary elements, and $\{\mathbf{X}_m \in \mathbb{R}^{M \times N} | m = 1, ..., M_d\}$ is a collection of sparse coefficient maps. It is worth noting that each $\mathbf{X}_m$ is the same size of the image $\mathbf{S} \in \mathbb{R}^{M \times N}$, while each $\mathbf{D}_m \in \mathbb{R}^{d \times d}$ is smaller than the image with $d \ll M, N$. Finally, $\mathbf{x}_m \in \mathbb{R}^{MN}$ is the vectorized version of each $\mathbf{X}_m$.

The CDSR approach can divided into two subproblems for its full solution:

1. Fixing the dictionary elements and updating the coefficient maps, and
2. Fixing the coefficient maps and updating the dictionary elements.

Wohlberg (Wohlberg, 2014) proposes that the coefficient maps update problem can be sol-

ved via ADMM by adding an auxiliary variable $\{\mathbf{Y}_m\}$ as

$$\begin{aligned} &\underset{\{\mathbf{X}_m\},\{\mathbf{Y}_m\}}{\operatorname{argmin}} \frac{1}{2}\left\|\sum_{m=1}^{M_d} \mathbf{D}_m \overset{2}{*} \mathbf{X}_m - \mathbf{S}\right\|_F^2 + \lambda \sum_{m=1}^{M_d} \|\mathbf{y}_m\|_1 , \\ &\text{s.t.: } \mathbf{X}_m = \mathbf{Y}_m, \quad \forall m = 1,...,M_d. \end{aligned} \tag{32}$$

Color Images using CSDR. From this point it is easy to extend to color images. Here, a RGB image is treated as a stack of gray-scale images, $\mathbf{S} \in \mathbb{R}^{M\times N\times 3}$. Wohlberg *et al.* proposed the convolutional basis pursuit denoising (CBPDN) (Wohlberg, 2016b) as a low computational cost alternative for the convolutional representation of a stack of gray-scale images by solving

$$\underset{\{\mathbf{X}_{c,m}\}}{\operatorname{argmin}} \frac{1}{2}\sum_{c=1}^{3}\left\|\sum_{m=1}^{M_d} \mathbf{D}_m \overset{2}{*} \mathbf{X}_{c,m} - \mathbf{S}_c\right\|_F^2 + \lambda \sum_{c=1}^{3}\sum_{m=1}^{M_d} \|\mathbf{X}_{c,m}\|_1 + \mu \|\mathbf{X}_{c,m}\|_{2,1} , \tag{33}$$

where $\{\mathbf{X}_{m,c},\ m = 1,...,M_d\ , c = 1,2,3 \mid \mathbf{X}_{m,c} \in \mathbb{R}^{M\times N}\}$ are the sparse coefficient maps for each dictionary element indexed by m and the c^{th} channel of the RGB image; while $\{\mathbf{D}_m,\ m = 1,...,M_d \mid \mathbf{D}_m \in \mathbb{R}^{d_M\times d_N}\}$ is a collection of dictionary elements. CBPDN represents the spatial correlation within each color channel in a RGB image, independently for each channel. This leads to missing the correlation between channels. Figure 7 shows a collection of dictionary elements obtained from a color image. Note the different textures obtained from the original image.

CSDR Solution in the Fourier Domain. While solving 2D cyclic convolutions have a high numerical cost associated, a solution in the Discrete Fourier Transform (DFT) domain has been proposed for this case (Bristow and Eriksson, 2013), profiting on the DFT properties. A 2D

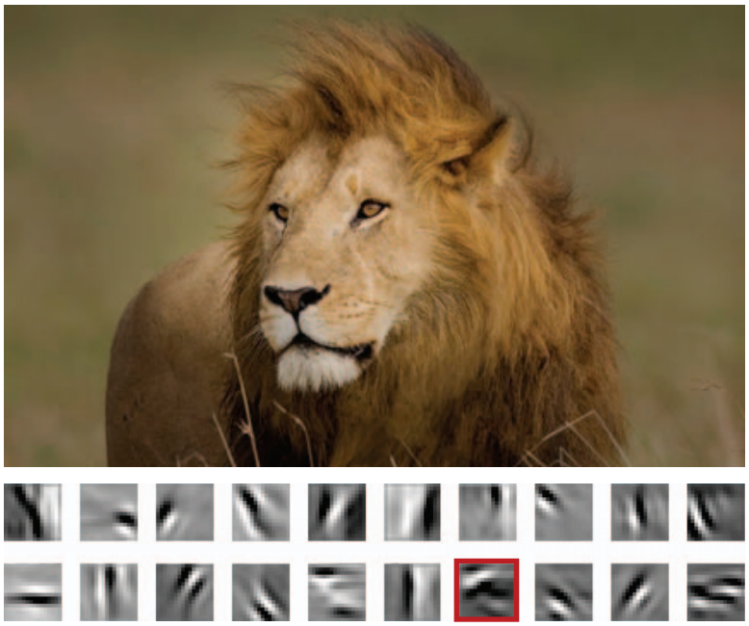


Figure 7. A selection of filters learned from an unaligned set of lions. The spatially invariant algorithm produces expression of generic Gabor-like filters as well as specialized domain specific filters, such as the highlighted ?eye?. Taken from (Bristow and Eriksson, 2013).

spatial cyclic convolution can be expressed as a Hadamard product in the Fourier domain as

$$\mathbf{A} \overset{2}{*} \mathbf{B} = \mathscr{F}_{2D}^{-1}\{\mathscr{F}_{2D}\{\mathbf{A}\} \odot \mathscr{F}_{2D}\{\mathbf{B}\}\}, \tag{34}$$

with $\odot$ denoting Hadamard product, $\mathscr{F}_{2D}\{\cdot\}$ denoting the 2D Fourier Transform, $\mathscr{F}_{2D}^{-1}\{\cdot\}$ denoting the 2D Inverse Fourier Transform. Finally, $\mathscr{F}_{2D}\{\mathbf{A}\}$ and $\mathscr{F}_{2D}\{\mathbf{B}\}$ have the same size. This can be achieved by zero-padding one or both matrices, in the spatial domain. The complexity of the cyclic convolution is estimated as $\mathscr{O}((MN)^2)$ while the DFT's is estimated as $\mathscr{O}(MN\log(MN))$ and Hadamard's as $\mathscr{O}(MN)$.

4.5. N-Dimensional CSDR

Considering the versatility of the CSC model for representing 1D signals, gray-scale and RGB images, plus the optimal computational strategies exposed in section 4.4, then it leads to think about a general model for representing multidimensional signals as

$$\mathscr{S} = \sum_{m=1}^{M_d} \mathscr{D}_m \overset{N}{*} \mathscr{X}_m + \boldsymbol{\Omega}, \tag{35}$$

where $\mathscr{S} \in \mathbb{R}^{L_1 \times \ldots \times L_N}$, $\{\mathscr{D}_m,\ m = 1,\ldots,M_d \mid \mathscr{D}_m \in \mathbb{R}^{d_1 \times \ldots \times d_N}\}$ and $\{\mathscr{X}_m,\ m = 1,\ldots,M_d \mid \mathscr{X}_m \in \mathbb{R}^{L_1 \times \ldots \times L_N}\}$, with $d_i \ll L_i$, and its solution in the Fourier domain as

$$\mathbf{A} \overset{N}{*} \mathbf{B} = \mathscr{F}_{ND}^{-1}\{\mathscr{F}_{ND}\{\mathbf{A}\} \odot \mathscr{F}_{ND}\{\mathbf{B}\}\}, \tag{36}$$

with $\mathscr{F}_{ND}\{\cdot\}$ denoting the ND Fourier Transform and $\mathscr{F}_{ND}^{-1}\{\cdot\}$ denoting the ND Inverse Fourier Transform. The complexity of the cyclic convolution is estimated then as $\mathscr{O}((L_1 \ldots L_N)^2)$, while the complexity for DFT is estimated as $\mathscr{O}(L_1 \ldots L_N \log(L_1 \ldots L_N))$, and hadamard product's as $\mathscr{O}(L_1 \ldots L_N)$. This means that it is feasible, computationally speaking, to represent higher dimension signals using the CDSR framework while keeping the computational cost within a reasonable limit. Equation (35) is the basis for the extension of the Convolutional Sparse Coding (CSC) framework presented in this doctoral dissertation, which will be presented in Chapter 4.

The CSC signal model provides the mathematical foundation for convolutional neural net-

works (CNN) which can automatically learn features within a signal using a hierarchical neural network, in a way similar to the process of human cognition (Liang and Li, 2016) (Yu et al., 2017). A typical CNN is a multi-layered architecture consisting of an input, an output layer and multiple hidden layers, i.e. of a series of convolutional layers capable of assembling more complex patterns using smaller and simpler patterns through convolution operations. On the other hand, the CSC model is a single-layer architecture, which can be formulated as a linear expansion equivalent to the basis pursuit (BP) scheme (Chen et al., 2001). The only similarity between CSC and CNN frameworks lies in the convolution operations used for representing an image.

Conclusions

The synthesis DSR is a signal-based representation which requires to be learned for each set of signals to be represented. However, this specificity allows for a higher level of reconstruction quality. The mathematical formulation of the minimization problems is somehow more lax, compared to the SSR model, but profits on being formulated as a $\ell_2 - \ell_1$ problem.

On the other hand, the CSDR model profits on the robust convolutional model, and reduces the complexity for its solution by profiting on the DFT theorem. However, the state-of-the-art reports the dimensions of the CSDR model up to 2D signals. For 3D signals (color images), the state-of-the-art suggest on expanding the use of the existing 2D model, missing the correlation of the third dimension (spectral axis), limiting the quality of the reconstructed spectral images.

Chapter 4 includes the proposal of a new 3D cyclic convolutional operator in order to represent the spatial-spectral correlations of an SI.

5. Convolutional Sparse Dictionary Representation for Spectral Images and Compressive Sensing Imaging

The purpose of this chapter is to introduce the first three contributions of this doctoral dissertation: a 3D convolutional sparse representation for SIs, its application in CSI, and the optimal numerical routines for its solution. This chapter includes the mathematical formulations and the derivation of their numerical solution. It also includes the synthetic and laboratory experiments in order to asses the performance of the proposed framework.

5.1. Definition Of Procedures And Dimensions Transformations

A series of procedures to be used in this doctoral dissertation are introduced below in order to facilitate the reading of the mathematical formulation and numerical solution. This doctoral research includes two types of $\bar{N}$-D arrays, listed bellow:

- Single $\bar{N}$-D arrays $\mathscr{A} \in \mathbb{R}^{L_1 \times \ldots \times L_{\bar{N}}}$.
- Collections of M_d $\bar{N}$-D arrays $\{\mathscr{A}_m, m = 1, \ldots, M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times \ldots \times L_{\bar{N}}}\}$

Also, this doctoral dissertation proposes four types of dimensional arrangements and domain transformations over the $\bar{N}$-D arrays:

Unfolding: To rearrange a $\bar{N}$-D array, or collection of arrays, into a single 1D array. The term comes from *unfolding* a tightly folded scarf into a single string of fabric. This dimensional

arrangement is necessary to use a $\bar{N}$-D array into a single lineal formulation, and is akin to a vectorization.

Equivalent Operator: To express the sum of convolutions, or hadamard products, as a single matrix operator. This rearrangement goes along the unfolding, or vectorization, of a collection of $\bar{N}$-D arrays.

Folding: To rearrange a 1D array into a $\bar{N}$-D array, or collection of arrays. The term comes from *folding* a string of fabric into a tightly packed manner. This dimensional arrangement is necessary in order to perform correctly the $\bar{N}$-D Fourier Transforms.

Fourier Transform: To transform a single $\bar{N}$-D array, or collection of arrays, to or from the Fourier domain. It is not advised to perform 1D Fourier transforms on 1D arrays and then folding the result as $\bar{N}$-D arrays. For this reason is necessary to perform the corresponding folding operations in order to solve $\mathscr{F}_{\bar{N}}\{\mathscr{A}\}$, and its inverse $\mathscr{F}_{\bar{N}}^{-1}\{\hat{\mathscr{A}}\}$.

Bellow are listed the several rearrangements and domain transformations over both types of $\bar{N}$-D arrays. Note that the domains $\mathbb{R}$ and $\mathbb{C}$ are interchangeable.

Algorithm 3 Unfolding a $\bar{N}$-D array

Require: Array $\mathscr{A} \in \mathbb{R}^{L_1 \times \ldots \times L_{\bar{N}}}$; dimensions $L_1, \ldots, L_{\bar{N}}$.
1: Create $\bar{M} = L_1 \cdot \ldots \cdot L_{\bar{N}}$.
2: Create $\mathbf{a} \in \mathbb{R}^{\bar{M}}$ by vectorizing $\mathscr{A}$ columnwise as $\mathbf{a} = \text{vec}(\mathscr{A})$.
3: **return** Array $\mathbf{a} \in \mathbb{R}^{\bar{M}}$, with $\bar{M} = L_1 \cdot \ldots \cdot L_{\bar{N}}$.

Algorithm 4 Unfolding a collection of M_d $\bar{N}$-D arrays

Require: Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1, ..., M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}\}$; dimensions $L_1, ..., L_{\bar{N}}$; value M_d.
1: Create $\bar{M} = L_1 \cdot ... \cdot L_{\bar{N}}$.
2: Vectorize each $\mathscr{A}_m$ columnwise as $\mathbf{a}_m = \text{vec}(\mathscr{A}_m)$, as explained in Algorithm 3.
3: Create the resulting array $\mathbf{a} = [\mathbf{a}_1^T ... \mathbf{a}_{M_d}^T]^T \in \mathbb{R}^{\bar{M}M_d}$.
4: **return** Array $\mathbf{a} \in \mathbb{R}^{\bar{M}M_d}$, with $\bar{M} = L_1 \cdot ... \cdot L_{\bar{N}}$.

Algorithm 5 Equivalent operator of sum of convolutions

Require: Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1, ..., M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}\}$; dimensions $L_1, ..., L_{\bar{N}}$; value M_d.
1: For each $\mathscr{A}_m$ create a equivalent convolutional matrix $\bar{\mathbf{A}}_m \in \mathbb{R}^{\bar{M} \times \bar{M}}$
2: Concatenate all equivalent convolutional matrices into $\bar{\mathbf{A}} = [\bar{\mathbf{A}}_1 ... \bar{\mathbf{A}}_{M_d}] \in \mathbb{R}^{\bar{M} \times \bar{M}M_d}$.
3: **return** Equivalent operator $\bar{\mathbf{A}} \in \mathbb{R}^{\bar{M} \times \bar{M}M_d}$.

Algorithm 6 Equivalent operator of sum of Hadamard products

Require: Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1, ..., M_d \mid \mathscr{A}_m \in \mathbb{C}^{L_1 \times ... \times L_{\bar{N}}}\}$; dimensions $L_1, ..., L_{\bar{N}}$; value M_d.
1: Vectorize each $\mathscr{A}_m$, as stated in Algorithm 7, to create the diagonal matrices $\bar{\mathbf{A}}_m = \text{diag}(\text{vec}(\mathscr{A}_m)) \in \mathbb{C}^{\bar{M} \times \bar{M}}$.
2: Concatenate all diagonal matrices into $\bar{\mathbf{A}} = [\bar{\mathbf{A}}_1 ... \bar{\mathbf{A}}_{M_d}] \in \mathbb{C}^{\bar{M} \times \bar{M}M_d}$.
3: **return** Equivalent operator $\bar{\mathbf{A}} \in \mathbb{C}^{\bar{M} \times \bar{M}M_d}$.

Algorithm 7 Folding a 1D array into a $\bar{N}$-D array

Require: Array $\mathbf{a} \in \mathbb{R}^{\bar{M}}$; dimensions $L_1, ..., L_{\bar{N}}$.
1: Create $\mathscr{A}$ as a $\mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}$ columnwise rearrangement.
2: **return** Array $\mathscr{A} \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}$.

Algorithm 8 Folding a 1D $\bar{M}M_d$ array into a collection of M_d $\bar{N}$-D arrays

Require: Array $\mathbf{a} \in \mathbb{R}^{\bar{M}M_d}$; dimensions $L_1, ..., L_{\bar{N}}$; value M_d.
1: Divide $\mathbf{a}$ into M_d $\mathbf{a}_m \in \mathbb{R}^{\bar{M}}$ arrays.
2: Fold each $\mathbf{a}_m$ array into $\mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}$, as explained in Algorithm 4.
3: **return** Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1, ..., M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}\}$.

Algorithm 9 Fourier transform of $\bar{N}$-D array

Require: Array $\mathscr{A} \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}$; dimensions $L_1, ..., L_{\bar{N}}$.
1: Obtain $\hat{\mathbf{A}} = \mathscr{F}_{\bar{N}}\{\mathscr{A}\}$
2: **return** Array $\hat{\mathbf{A}} \in \mathbb{C}^{L_1 \times ... \times L_{\bar{N}}}$.

Algorithm 10 Fourier transform of a collection of M_d $\bar{N}$-D arrays

Require: Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1,...,M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}\}$; dimensions $L_1,...,L_{\bar{N}}$; value M_d.
1: For each $\mathscr{A}_m$ obtain the Fourier Transform $\hat{\mathscr{A}}_m$, as stated in Algorithm 9.
2: Create the collection $\hat{\mathscr{A}} = \{\hat{\mathscr{A}}_m, m = 1,...,M_d\}$.
3: **return** Collection $\hat{\mathscr{A}} = \{\hat{\mathscr{A}}_m, m = 1,...,M_d \mid \hat{\mathscr{A}}_m \in \mathbb{C}^{L_1 \times ... \times L_{\bar{N}}}\}$.

Algorithm 11 Inverse Fourier transform of $\bar{N}$-D array

Require: Array $\hat{\mathscr{A}} \in \mathbb{C}^{L_1 \times ... \times L_{\bar{N}}}$; dimensions $L_1,...,L_{\bar{N}}$.
1: Obtain $\mathbf{A} = \mathscr{F}_{\bar{N}}^{-1}\{\hat{\mathscr{A}}\}$
2: **return** Array $\mathbf{A} \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}$.

Algorithm 12 Inverse Fourier transform of a collection of M_d $\bar{N}$-D arrays

Require: Collection $\hat{\mathscr{A}} = \{\hat{\mathscr{A}}_m, m = 1,...,M_d \mid \hat{\mathscr{A}}_m \in \mathbb{C}^{L_1 \times ... \times L_{\bar{N}}}\}$; dimensions $L_1,...,L_{\bar{N}}$; value M_d.
1: For each $\mathscr{A}_m$ obtain the Inverse Fourier Transform $\hat{\mathscr{A}}_m$, as stated in Algorithm 11.
2: Create the collection $\mathscr{A} = \{\mathscr{A}_m, m = 1,...,M_d\}$.
3: **return** Collection $\mathscr{A} = \{\mathscr{A}_m, m = 1,...,M_d \mid \mathscr{A}_m \in \mathbb{R}^{L_1 \times ... \times L_{\bar{N}}}\}$.

5.2. Convolutional Sparse Coding for Spectral Im- ages (CSC3D)

In order to represent SI's using a CSDR framework, it would be natural to extend CBPDN framework in Eq. (33) from $C = 3$ to $C = L$ channels. While being a simple escalation of CBPDN's proven algorithm, this framework misses completely the intrinsic spectral correlation of SI's by representing each channel independently.

Taking the previous into consideration, we propose a single 3D cyclic convolutional operator based in the following statements:

1. A single convolutional operator could include all SI's spatial-spectral correlations within a single operation.

2. A single operator simplifies the mathematical formulation, based in the SSR framework (section 3.3).

3. A single cyclic operator could profit extensively on the DFT solution stated in Eq. (36).

4. There is no need for the additional channel-wise summation of CBPDN, which induces additional complexity to the formulation.

Based on the previous premises, the proposed single 3D cyclic convolutional operator is then defined as

$$\mathscr{S} = \sum_{m=1}^{M_d} \mathscr{D}_m \overset{3}{*} \mathscr{X}_m + \boldsymbol{\Omega}, \tag{37}$$

where $\mathscr{S} \in \mathbb{R}^{M \times N \times L}$ is the SI of interest to be represented convolutionally; $\{\mathscr{D}_m,\ m = 1,...,M_d \mid \mathscr{D}_m \in \mathbb{R}^{d_M \times d_N \times d_L}\}$ is a collection of 3D convolutional dictionary elements with $d_i \ll M,N,L$; $\{\mathscr{X}_m,\ m = 1,...,M_d \mid \mathscr{X}_m \in \mathbb{R}^{M \times N \times L}\}$ is a collection of 3D sparse coefficient maps; and $\boldsymbol{\Omega}$ represents the reconstruc- tion error (Barajas-Solano et al., 2019c).

In order to include Eq. (37) within a minimization scheme we can express it as the single linear operation

$$\mathbf{s} = \mathbf{vec}(\mathscr{S}) = \bar{\mathbf{D}}\mathbf{x} + \boldsymbol{\omega}, \tag{38}$$

where:

- $\mathbf{s} \in \mathbb{R}^{MNL}$ results from unfolding $\mathscr{S}$ (see Algorithm 3, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).

- $\bar{\mathbf{D}} = [\bar{\mathbf{D}}_1 ... \bar{\mathbf{D}}_{M_d}] \in \mathbb{R}^{MNL \times MNLM_d}$ is created from $\{\mathscr{D}_m\}$ (see Algorithm 5, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).
- $\mathbf{x}_m \in \mathbb{R}^{MNL}$ results from unfolding $\{\mathscr{X}_m\}$ (see Algorithm 4, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).
- each equivalent convolutional matrix $\bar{\mathbf{D}}_m$ is created in such way that $\bar{\mathbf{D}}_m \mathbf{x}_m = \text{vec}(\mathscr{D}_m \overset{3}{*} \mathscr{X}_m)$.

Creating The Equivalent Convolutional Matrices. Lets us make a pause in the narrative of this doctoral dissertation for addressing the complexity of constructing the equivalent convolutional matrices $\bar{\mathbf{D}}_m$, and operator $\bar{\mathbf{D}}$, by using the example shown in figure 8 and dimensions $d_M = 2, d_N = 2, d_L = 2, M = 4, N = 4, L = 3$. Then, the matrices $\mathscr{D}_m$ and $\mathscr{X}_m$ have dimensions $\mathbb{R}^{2\times2\times2}$ and $\mathbb{R}^{4\times4\times3}$, respectively.

The first step is to create the equivalent spatial 2D cyclic convolutional sub-matrices $\mathscr{D}_{l,m} \in \mathbb{R}^{MN \times MN}$, where $l = 1, 2$ (see the highlighted red rectangle in the upper left corner in Figure 8). In this example, we have a collection of $\mathbb{R}^{16\times16}$ sub-matrices which can be seen between the white lines in Figure 8. Note the self-replicating disposition of the elements, and the circular shifting effect in the top right corner of each sub-matrix, because of the cyclic convolution.

The spectral dimension, $L = 3$, generates the equivalent 3D cyclic convolutional matrices $\mathscr{D}_m \in \mathbb{R}^{MNL \times MNL}$, or $\mathbb{R}^{48\times48}$, by self-replicating the equivalent 2D cyclic convolutional matrices in a 3×3 grid in a cyclic pattern.

As shown in Figure 8, matrix $\bar{\mathbf{D}}_m$ has a very specific structure given by the size of $\mathscr{D}_m$ and $\mathscr{X}_m$. If we were to use different values for the convolutional dictionaries and sparse coefficient maps but keeping the exact dimensions, then matrix $\bar{\mathbf{D}}_m$ would keep the same structure. However, a minor change in $\mathscr{D}_m$'s or $\mathscr{X}_m$'s dimensions changes $\bar{\mathbf{D}}_m$ completely.

The last step left is to create the operator $\bar{\mathbf{D}}$ as the concatenation of several matrices $\bar{\mathbf{D}}_m$ as shown in Figure 9. However, the construction and manipulation of operator $\bar{\mathbf{D}}$ isn't viable within a minimization scheme. For this reason we profit on the Discrete Fourier Transform for simplicity of calculations, which will be explained in section 5.2.

Creating The Dual Optimization Formulation. Once the lineal operator $\bar{\mathbf{D}}$ has been explained, then we can formulate the minimization scheme as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{D}}\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1 . \tag{39}$$

Equation (39) is called the *Coefficient Update Problem* (CUP) and obtains the optimal coefficient maps for a fixed collection of convolutional elements (Barajas-Solano et al., 2019a). It is safe to assume that an optimal 3D convolutional dictionary will rarely be available *a priori* for a specific SI. In order to obtain an optimal 3D convolutional dictionary for a given SI, we modify Eq. (39) by introducing some changes.

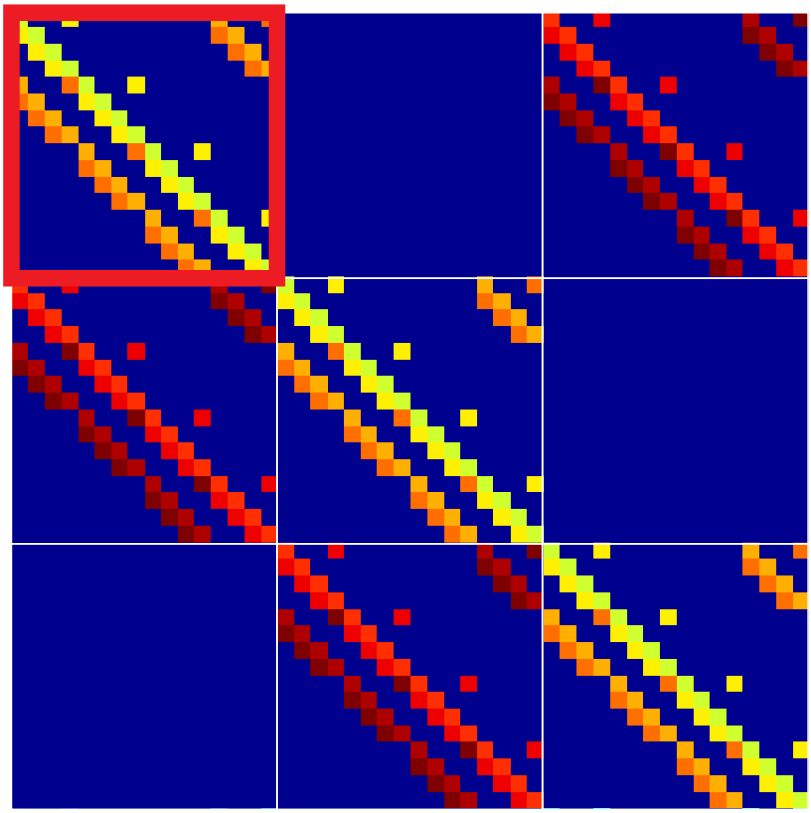


Figure 8. Example of $\bar{\mathbf{D}}_m$ for $\mathscr{D}_m \in \mathbb{R}^{2\times2\times2}$ and $\mathscr{X}_m \in \mathbb{R}^{4\times4\times3}$. The red rectangle highlights a 2D equivalent cyclic convolution matrix.

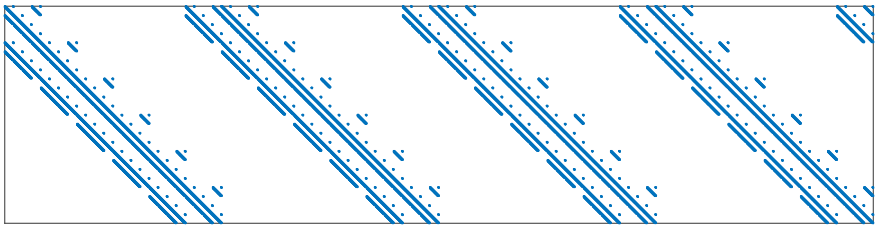


Figure 9. Schematic for $\bar{\mathbf{D}}$ with $M = 10, N = 10, L = 4, M_d = 4$ and $d_M = d_N = d_L = 2$.

First, we profit on the convolution's commutativity property so $\mathscr{D} \overset{3}{*} \mathscr{X} = \mathscr{X} \overset{3}{*} \mathscr{D}$, and shift the linear operation in Eq. (38) to

$$\mathbf{s} = \mathbf{vec}(\mathscr{S}) = \bar{\mathbf{X}}\mathbf{d} + \boldsymbol{\omega}, \tag{40}$$

where $\bar{\mathbf{X}} = [\bar{\mathbf{X}}_1 ... \bar{\mathbf{X}}_{M_d}] \in \mathbb{R}^{MNL \times MNLM_d}$ is created following Algorithm 5 ($L_1 \times ... \times L_{\bar{N}} = M \times N \times L$) so that $\bar{\mathbf{X}}_m \mathbf{d}_m = \text{vec}(\mathscr{X}_m \overset{3}{*} \mathscr{D}_m)$; and $\mathbf{d} = [\text{vec}(\mathscr{D}_1)^T ... \text{vec}(\mathscr{D}_{M_d})^T]^T \in \mathbb{R}^{MNLM_d}$ as indicated in Algorithm 4 ($L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).

Second, there is an implicit zero-padding in the formulation of array $\mathbf{d}$ in order to match the size of the dictionary elements to the size of the coefficient maps. Let be the zero-padding operator $\mathbf{Z}_p : \mathbb{R}^{d_M \times d_N \times d_L} \rightarrow \mathbb{R}^{M \times N \times L}$, with the dictionary elements being zero-padded and the desired resulting filters as $\mathbf{Z}_p^T \mathbf{d}_m$. Then, let be the constraint set, according to (Wohlberg, 2016c)

$$\mathscr{C}_{Z_p} = \left\{ \mathbf{x} \in \mathbb{R}^{MNL} : (\mathbf{I} - \mathbf{Z}_p \mathbf{Z}_p^T)\mathbf{x} = 0, \|\mathbf{x}\|_2 = 1 \right\}, \tag{41}$$

which guarantees that the obtained dictionary elements are normalized and keeps the desired size. Next, the indicator function of the constrained set is introduced as

$$\iota_{\mathscr{C}_Z}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in \mathscr{C}_{Z_p} \\ \\ \infty & \text{if } \mathbf{x} \notin \mathscr{C}_{Z_p}. \end{cases}, \tag{42}$$

and applied over each vectorized individual convolutional element $\mathbf{d}_m \in \mathbb{R}^{MNL}$, but for simplifying the notation it will be applied over the whole collection. The ℓ_1 norm is then replaced by $\iota_{\mathscr{C}_Z}$ in order to avoid the scaling ambiguity between dictionary filters and coefficients.

Finally, the unconstrained problem for obtaining a set of convolutional dictionary elements for a SIs, the *Dictionary Update Problem* (DUP), can be written as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{X}}\mathbf{d} - \mathbf{s} \right\|_2^2 + \iota_{\mathscr{C}_P}(\mathbf{d}). \tag{43}$$

Eq. (39) and (43) are solved alternately, and form the first product of this doctoral dissertation: the Convolutional Sparse Coding 3D (CSC3D) framework for representing SIs. This formulation fulfills the second specific objective of this doctoral dissertation.

Coefficients Update Problem (CUP). Eq. (39) is referred to as the *Coefficients Update Problem* (CUP) and seeks to adjust a collection of coefficient maps to a fixed collection of convolutional 3D dictionary elements in order to sparsely represent a SI of interest. Eq. (39) can be solved using the alternating directions multiplier method, ADMM (Boyd et al., 2010), by introducing an auxiliary variable as

$$\begin{aligned} &\underset{\mathbf{x},\mathbf{v}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{D}}\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \mathbf{v} \right\|_1, \\ &\text{s.t.: } \mathbf{v} = \mathbf{x}. \end{aligned} \tag{44}$$

The augmented Lagrangian for Eq. (44) can be written as stated in (Barajas-Solano et al., 2019a) as

$$\mathscr{L}\{\mathbf{x},\mathbf{v},\mathbf{g}\} = \frac{1}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{s}\right\|_2^2 + \lambda\left\|\mathbf{v}\right\|_1 + \frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}+\mathbf{g}\right\|_2^2, \tag{45}$$

where $\mathbf{g}$ is the so called dual variable. The variable updates are obtained from Eq. (45) as

$$\mathbf{x}^{j+1} := \underset{\mathbf{x}}{\operatorname{argmin}}\,\frac{1}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{s}\right\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}^j+\mathbf{g}^j\right\|_2^2, \tag{46}$$

$$\mathbf{v}^{j+1} := \underset{\mathbf{v}}{\operatorname{argmin}}\,\frac{\rho}{2}\left\|\mathbf{x}^{j+1}-\mathbf{v}+\mathbf{g}^j\right\|_2^2 + \lambda\left\|\mathbf{v}\right\|_1, \tag{47}$$

$$\mathbf{g}^{j+1} = \mathbf{g}^j + \mathbf{x}^{j+1} - \mathbf{v}^{j+1}. \tag{48}$$

The dual variable $\mathbf{g}$ can be interpreted as a vector of prices and Eq. (48) is then called a *price update* or *price adjustment step* (Boyd et al., 2010). The solutions to subproblems (46) and (47) are presented bellow.

Coefficient Maps Update. The linear representation of the 3D convolution in Eq. (46) can be solved efficiently by profiting the DFT convolution theorem (Bristow and Eriksson, 2013), which states that a N-dimensional cyclic convolution can be expressed as a Hadamard product in the n-dimensional Fourier domain, as stated in Appendix 1, as

$$\sum_{m=1}^{M_d} \boldsymbol{\mathscr{D}}_m \overset{3}{*} \boldsymbol{\mathscr{X}}_m = \mathscr{F}_{3D}^{-1}\left(\sum_{m=1}^{M_d} \mathscr{F}_{3D}(\boldsymbol{\mathscr{D}}_m) \odot \mathscr{F}_{3D}(\boldsymbol{\mathscr{X}}_m)\right). \tag{49}$$

Eq. (49) can be simplified by transforming the sums of Hadamard products into a matrix-vector product, just as in Eq. (38). The detailed steps for creating the equivalent matrix-vector product in the Fourier domain are listed in Algorithms 6 and 4. Then, the sum of 3D convolutions can be solved efficiently as

$$\sum_{m=1}^{M_d} \mathscr{D}_m \overset{3}{*} \mathscr{X}_m \simeq \mathscr{F}_{3D}^{-1}\left(\hat{\hat{\mathbf{D}}}\hat{\mathbf{x}}\right), \tag{50}$$

minding, of course, that $\hat{\hat{\mathbf{D}}}\hat{\mathbf{x}} \in \mathbb{C}^{MNL}$ must be folded first into a 3D array as explained in Appendix 1. Eq. (50) can be used to optimize the solution of Eq. (46) by expressing the latter in the Fourier domain. We begin by replacing $\mathbf{w}^j = \mathbf{v}^j - \mathbf{g}^j \in \mathbb{R}^{MNLM_d}$ and creating $\hat{\mathbf{w}}^j \in \mathbb{C}^{MNLM_d}$ by folding (Algorithm 8, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$), transforming to the Fourier domain (Algorithm 10), and unfolding (Algorithm 4, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$). Finally, we can rewrite Eq. (46) in the Fourier domain as

$$\hat{\mathbf{x}}^{j+1} := \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \frac{\rho}{2}\left\|\hat{\hat{\mathbf{D}}}\hat{\mathbf{x}} - \hat{\mathbf{s}}\right\|_2^2 + \frac{\rho}{2}\left\|\hat{\mathbf{x}} - \hat{\mathbf{w}}^j\right\|_2^2. \tag{51}$$

Solving the derivative of Eq. (51) and equaling to zero, the solution to the optimization problem is

$$\hat{\mathbf{x}}^{j+1} = \left(\hat{\hat{\mathbf{D}}}^H\hat{\hat{\mathbf{D}}} + \mathbf{I}\right)^{-1}\left(\hat{\hat{\mathbf{D}}}^H\hat{\mathbf{s}}^j + \hat{\mathbf{w}}^j\right). \tag{52}$$

Eq. (52) seems to have a simple closed form solution, but the size of matrix $\hat{\hat{\mathbf{D}}}$ makes a direct inverse procedure unfeasible. It is necessary to clarify the true dimension of the numerical cost necessary for the solution of Eq. (52) before continuing. Taking into consideration the following dimensions $M = 128$, $N = 128$, $L = 16$, $M_d = 30$ as a practical example, lets solve then the

following statements:

- The dimensions of $\hat{\hat{\mathbf{D}}} = [\hat{\hat{\mathbf{D}}}_1 ... \hat{\hat{\mathbf{D}}}_{M_d}] \in \mathbb{C}^{MNL \times MNLM_d}$ become $262.144 \times 7.864.320$; the diagonal matrix $\hat{\hat{\mathbf{D}}}_i \in \mathbb{C}^{MNL \times MNL}$ becomes 262.144×262.144, with $MNL = 262.144$ non-zero complex elements.

- Considering that a complex number takes 16bytes in RAM space as two float numbers, then $\hat{\hat{\mathbf{D}}}_i$ takes $262.144 * 16$Bytes = 4MB of RAM space.

- Considering that $\hat{\hat{\mathbf{D}}}$ contains $M_d = 30$ matrices $\hat{\hat{\mathbf{D}}}_i$, then $\hat{\hat{\mathbf{D}}}^H \hat{\hat{\mathbf{D}}}$ contains $M_d^2 = 900$ matrices $\hat{\hat{\mathbf{D}}}_i$, for a total of $900 * 4$MB = 3.52GB of RAM space.

- Matrix $\mathbf{I} \in \mathbb{C}^{MNLM_d \times MNLM_d}$ must be also float in order to operate correctly, with $MNLM_d$ non-zero elements. Then, matrix $\mathbf{I}$ takes $128 * 128 * 16 * 30 * 8$ Bytes = 60MB of RAM space.

- Matrix $\hat{\hat{\mathbf{D}}}^H \hat{\hat{\mathbf{D}}} + \mathbf{I}$ takes 3.57GB of RAM space, with dimensions $7.864.320 \times 7.864.320$.

Now, the complexity of inverting a 3.57GB matrix is evident. Eq. (52) can be solved by using the Woodbury Matrix Inverse method (Henderson and Searle, 1981), explained in Appendix 2, as

$$\hat{\mathbf{x}}^{j+1} = \mathbf{b} - \hat{\hat{\mathbf{D}}}^H \left(\mathbf{I} + \hat{\hat{\mathbf{D}}} \hat{\hat{\mathbf{D}}}^H \right)^{-1} \hat{\hat{\mathbf{D}}} \mathbf{b}, \tag{53}$$

with $\mathbf{b} = \hat{\hat{\mathbf{D}}}^H \hat{\mathbf{s}}^j + \hat{\mathbf{w}}^j$. Now, lets analyze the inversion in Eq. (53) by solving the following statements:

- Consider $\hat{\hat{\mathbf{D}}} = [\hat{\hat{\mathbf{D}}}_1 ... \hat{\hat{\mathbf{D}}}_{M_d}] \in \mathbb{C}^{MNL \times MNLM_d}$, and each $\hat{\hat{\mathbf{D}}}_i \in \mathbb{C}^{MNL \times MNL}$ a diagonal matrix.

- The product $\hat{\mathbf{D}}\hat{\mathbf{D}}^H \in \mathbb{R}^{MNL \times MNL}$ is a diagonal matrix with $MNL = 262.144$ non-zero complex elements, and takes $262.144 * 16$Bytes = 4MB of RAM space.
- Matrix $\mathbf{I} \in \mathbb{C}^{MNL \times MNL}$ takes $262.144 * 8$Bytes = 2MB of RAM space.
- We must invert matrix $\mathbf{I} + \hat{\mathbf{D}}\hat{\mathbf{D}}^H$, which only takes 6MB of RAM space.
- Matrix $\mathbf{I} + \hat{\mathbf{D}}\hat{\mathbf{D}}^H$ is diagonal, which inversion results trivial.

The previous statements give an idea on the potential RAM space saved by using the Woodbury matrix identity formula, as explained in Appendix 2. However, given the structure of matrix $\hat{\mathbf{D}}$, then Eq. (53) can be optimized even further, decreasing the overall complexity. For example, the product $\hat{\mathbf{D}}\hat{\mathbf{D}}^H$ has complexity $\mathscr{O}((MNL)^3 M_d)$. This research work proposes a collection of numerical rearrangements in order to reduce the overall complexity of solving Eq. (52) to just $\mathscr{O}(MNLM_d)$. The full explanation can be found in detail in Appendix 3, which is listed as one of the contributions of this doctoral dissertation.

Finally, the update $\mathbf{d}^{j+1}$ is obtained by folding, transforming from the Fourier domain and unfolding (Algorithms 8, 12, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).

Sparse Coefficient Maps Update. Eq. (47) has a closed form solution via soft thresholding (Rockafellar, 1970) as

$$\mathbf{v}^{j+1} = \mathscr{S}_{\frac{\lambda}{\rho}} \left(\mathbf{x}^{j+1} + \mathbf{g}^j \right). \tag{54}$$

The CUP solution is summarized in Algorithm 13.

Algorithm 13 CUP Solution for 3D CDSR

Require: $\bar{\mathbf{D}}$, $\hat{\bar{\mathbf{D}}}$, $\mathbf{v}^j$, $\mathbf{g}^j$, ρ, λ and sizes M, N, L and M_d.
1: Build $\hat{\mathbf{z}}^j$.
2: Solve $\mathbf{x}^{j+1}$ in Eq. (46) using (Eq. (53).
3: Solve $\mathbf{v}^{j+1}$ Eq. (47) using (Eq. (54).
4: Solve $\mathbf{g}^{j+1}$ in Eq. (48).
5: **return** Sparse coefficient maps $\mathbf{v}^{j+1}$, split variable update $\mathbf{v}^{j+1}$, and dual variable update $\mathbf{g}^{j+1}$.

Dictionary Update Problem (DUP). Eq. (43) is referred to as the *Dictionary Update Problem* (DUP) and seeks to adjust a collection of convolutional 3D dictionary elements to a given collection of sparse coefficient maps. Just as with CUP, Eq. (43) can be solved using the alternating directions multiplier method, ADMM (Boyd et al., 2010), by introducing an auxiliary variable as

$$\begin{aligned} &\underset{\mathbf{d},\mathbf{p},\mathbf{q}}{\operatorname{argmin}} \frac{1}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \\ &\text{s.t.: } \mathbf{q}=\mathbf{d}. \end{aligned} \tag{55}$$

The augmented Lagrangian for Eq. (55) can be written as stated in (Barajas-Solano et al., 2019a) as

$$\mathscr{L}\{\mathbf{d},\mathbf{q},\mathbf{t}\} = \frac{1}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}) + \frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}+\mathbf{t}\right\|_2^2, \tag{56}$$

where $\mathbf{q}$ is the so called dual variable. The variable updates are obtained from Eq. (56) as

$$\mathbf{d}^{j+1} := \underset{\mathbf{d}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2 + \frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}^j+\mathbf{t}^j\right\|_2^2, \tag{57}$$

$$\mathbf{q}^{j+1} := \underset{\mathbf{q}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \mathbf{d}^{j+1} - \mathbf{q} + \mathbf{t}^{j} \right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \tag{58}$$

$$\mathbf{t}^{j+1} = \mathbf{t}^{j} + \mathbf{d}^{j+1} - \mathbf{q}^{j+1}. \tag{59}$$

The dual variable $\mathbf{t}$ can be interpreted as a vector of prices and Eq. (59) is then called a *price update* or *price adjustment step* (Boyd et al., 2010). The solutions to subproblems (57) and (58) are presented bellow.

Convolutional Dictionary Update. Problem (57) can be solved efficiently by profiting the DFT convolution theorem as described for problem (46), thus rewriting problem (57) as

$$\hat{\mathbf{d}}^{j+1} := \underset{\hat{\mathbf{d}}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \hat{\hat{\mathbf{X}}} \hat{\mathbf{d}} - \hat{\mathbf{s}}^{j} \right\|_2^2 + \frac{\sigma}{2} \left\| \hat{\mathbf{d}} - \hat{\mathbf{w}}^{j} \right\|_2^2, \tag{60}$$

by replacing $\mathbf{w}^j = \mathbf{q}^j - \mathbf{t}^j \in \mathbb{R}^{MNLM_d}$ and creating $\hat{\mathbf{w}}^j = \in \mathbb{C}^{MNLM_d}$ by folding (Algorithm 8, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$), transforming to the Fourier domain (Algorithm 10), and unfolding (Algorithm 4, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$). Just as with the CUP update steps, matrix $\hat{\hat{\mathbf{X}}}$ is built from the updated coefficient maps $\mathbf{v}^{j+1}$ in Eq. (54) by folding (Algorithm 8, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$), transforming to the Fourier domain (Algorithm 10) and creating the equivalent operator (Algorithm 6, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$). Solving the derivative and equaling to zero, the solution to Eq. (60) is

$$\hat{\mathbf{d}}^{j+1} = \left(\hat{\hat{\mathbf{X}}}^H \hat{\hat{\mathbf{X}}} + \mathbf{I} \right)^{-1} \left(\hat{\hat{\mathbf{X}}}^H \hat{\mathbf{s}} + \hat{\mathbf{w}}^j \right). \tag{61}$$

Matrix $\hat{\hat{\mathbf{X}}}$ have the same concatenated diagonal structure as $\hat{\hat{\mathbf{D}}}$ in CUP, then Eq. (61) can be

solved directly using the optimized routine explained in Appendices 2 and 3 as

$$\hat{\mathbf{d}}^{j+1} = \mathbf{b} - \hat{\bar{\mathbf{X}}}^{H}\left(\mathbf{I} + \hat{\bar{\mathbf{X}}}\hat{\bar{\mathbf{X}}}^{H}\right)^{-1}\hat{\bar{\mathbf{X}}}\mathbf{b}, \tag{62}$$

with $\mathbf{b} = \hat{\bar{\mathbf{X}}}\hat{\mathbf{s}} + \hat{\mathbf{w}}^{j}$. Finally, the update $\mathbf{d}^{j+1}$ is obtained by folding (Algorithm 8,$L_1 \times ... \times L_{\bar{N}} = M \times N \times L$), transforming from the Fourier domain (Algorithm 12) and unfolding (Algorithm 4, $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).

Note that the solutions (53) and (62) has similar structures due to the similar $\ell_2 - \ell_2$ formulations. The main difference is the lineal operator $\hat{\mathbf{D}}$ and $\hat{\mathbf{X}}$, respectively. These two operators have the same concatenated diagonal matrices structure, which helps to simplify the numerical implementation.

Desired Convolutional Dictionary Update. Eq. (58) has closed solution via the proximal of $\iota_{\mathscr{C}_Z}$ (Rockafellar, 1970)

$$\mathbf{q}^{j+1} = \frac{\mathbf{Z}_p\mathbf{Z}_p^{T}(\mathbf{d}^{j+1} + \mathbf{t}^{j})}{\left\|\mathbf{Z}_p\mathbf{Z}_p^{T}(\mathbf{d}^{j+1} + \mathbf{t}^{j})\right\|}. \tag{63}$$

The DUP solution is summarized in Algorithm 14.

Algorithm 14 DUP Solution for 3D CDSR

Require: $\bar{\mathbf{X}}$, $\hat{\bar{\mathbf{X}}}$, $\mathbf{q}^{j}$, $\mathbf{t}^{j}$, σ and sizes M, N, L and M_d.
1: Build $\hat{\mathbf{z}}^{j}$.
2: Solve $\mathbf{d}^{j+1}$ in Eq. (57) using (Eq. (61).
3: Solve $\mathbf{q}^{j+1}$ Eq. (58) using (Eq. (63).
4: Solve $\mathbf{t}^{j+1}$ in Eq. (59).
5: **return** :Convolutional dictionary $\mathbf{d}^{j+1}$, split variable update $\mathbf{q}^{j+1}$, and dual variable update $\mathbf{t}^{j+1}$.

Proposed CSC3D Algorithm. The proposed CSC3D framework consists in alternately solving two related problems, CUP and DUP, in order to obtain both a collection of sparse coefficient maps and convolutional dictionary elements. Additionally, the regularization parameters ρ, λ and σ can be updated in each iteration according to (Boyd et al., 2010), section 3.3, or they can be set to a fixed value. Thus, this doctoral dissertation proposes to solve both problems alternately, as exposed in Algorithm 15.

Algorithm 15 CSC3D Algorithm

Require: $\{\mathscr{X}_m^0 \in \mathbb{R}^{M\times N\times L} | m = 1,...,M_d\}$ as zeros ; $\{\mathscr{D}_m^0 \in \mathbb{R}^{M\times N\times L} | m = 1,...,M_d\}$ as random; $M, N, L, M_d, d, \rho_0, \lambda$, σ_0 and $\mathbf{s}$.

1: Set $\{\mathscr{V}_m^0\} = \{\mathscr{X}_m^0\}$ and build the vectorization $\mathbf{v}^0$.
2: Set $\{\mathscr{Q}_m^0\} = \{\mathscr{D}_m^0\}$ and build the vectorization $\mathbf{q}^0$.
3: Build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from $\{\mathscr{D}_m^0\}$.
4: Set $j = 0$.
5: **repeat**
6: Solve CUP as explained in Algorithm 13.
7: Update ρ^{j+1} according to (Boyd et al., 2010), section 3.3
8: Fold $\mathbf{v}^{j+1}$ into $\{\mathscr{V}_m^{j+1}\}$, as explained in Algorithm 7, and build $\bar{\mathbf{X}}$ and $\hat{\bar{\mathbf{X}}}$ from it.
9: Solve DUP as explained in Algorithm 14.
10: Update σ^{j+1} according to (Boyd et al., 2010), section 3.3
11: Fold $\mathbf{q}^{j+1}$ into $\{\mathscr{Q}_m^{j+1}\}$, as explained in Algorithm 7, and build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from it.
12: **until** the residuals meet a given tolerance, or completed a number of iterations.
13: **return** the sparse coefficient maps $\{\mathscr{V}_m^{j+1}\}$ and the convolutional dictionary elements $\{\mathscr{Q}_m^{j+1}\}$

As with all iterative methods, ADMM is sensitive to initial values. Different empiric initialization alternatives for the initial dictionary elements $\{\mathscr{D}_m^0\}$ were tested, which included a full random cube and variations of zero-value cubes with some non-zero positions. The chosen alternative can be described as a centered random-value subcube $\in \mathbb{R}^{d_2\times d_2\times d_2}$, with $d_2 = d/2$, within an all zeros $\mathbb{R}^{d\times d\times d}$ cube. This variation showed the best trade-off between the highest Peak Signal-to-Noise Rate value (PSNR, for reconstruction quality) and the lowest NoN Zero percentage of elements (NNZ, for sparsity), in both mean and standard deviation.

About the convergence boundaries analysis, the proposed CSC3D is strongly based in the same Basis Pursuit scheme used in Wohlberg's CBPDN. This is, a convex $\ell_2 - \ell_1$ formulation for CUP and a ℓ_2-indicator function for DUP. Both problems are essentially an ℓ_2 and a convex restriction formulation. The main difference is that CSC3D replaces the 2D convolutions per layer in CBPDN for one full 3D convolutional sparse representation, i.e. a 3D cyclic convolution. As explained in (Bristow and Eriksson, 2013), the introduction of the convolution operations in the ADMM does not modify the convergence of the ADMM solution (Nishihara et al., 2015), even if a global minimum cannot be guaranteed. When expressed in the Fourier domain, the structures of both CBPDN and CSC3D are identical, thus implying the same convergence boundaries. However, CSC3D's higher numerical dimension requires a higher grade of optimization, as explained in Appendices 2 and 3.

Estimated Numerical Complexity. We will now estimate the numerical complexity of the more complex subproblems in CUP and DUP. Subproblem (47)'s solution, Eq. (54), is obtained by a soft-thresholding problem and subproblem (58)'s solution, Eq. (63), is akin to a hard-thresholding problem. Both complexities are negligible.

One of the most important sources of numerical complexity is the 3D convolutional operation, with complexity $\mathscr{O}((MNL)^2)$, considering that the dictionary elements are zero-padded to match the dimensions of the sparse coefficient maps. The 3D convolution complexity is reduced

to a fraction by expressing it as a Hadamard product, profiting on the Discrete Fourier Transform (DFT) Theorem, reducing the cost to $\mathscr{O}(MNL\log(MNL))$.

Finally, the greatest source of numerical complexity are the inversions in Eqs. (52) and (61), as solutions to subproblems (46) and (57). Again, the canonical complexity for inverting $\hat{\mathbf{A}}^H\hat{\mathbf{A}} + \alpha\mathbf{I} \in \mathbb{R}^{MNLM_d \times MNLM_d}$ is $\mathscr{O}((MNLM_d)^3)$. However, by profiting on the concatenated diagonal structure of $\hat{\hat{\mathbf{D}}}$ and $\hat{\hat{\mathbf{X}}}$ and the dimensions rearrangement exposed in Appendix B of (Barajas-Solano et al., 2019a), the inversion complexity falls to $\mathscr{O}(MN\ LM_d)$ as shown in Eqs. (53) and (62). Table 2 summarizes the different complexities for the stated subproblems and their solutions.

Eq.	Complexity	Solution
(47) and (58)	Negible	Implemented
(46) and (57)	$\mathscr{O}((MNLM_d)^3)$	Original
(53) and (62)	$\mathscr{O}(MNLM_d)$	Implemented

Table 2
Complexity review of the proposed CSC3D algorithm.

5.3. CSC3D Synthetic Performance Evaluation

The proposed CSC3D algorithm performance was tested with a use case, specifically a denoising scheme with $\mathbf{s}_n = \mathbf{s} + \sigma\boldsymbol{\eta}$, where $\boldsymbol{\eta}$ is a standard white Gaussian noise with standard deviation $\sigma > 0$. The noise levels included 10dB, 15dB, and 20dB of SNR. The details for the performance evaluation are listed bellow.

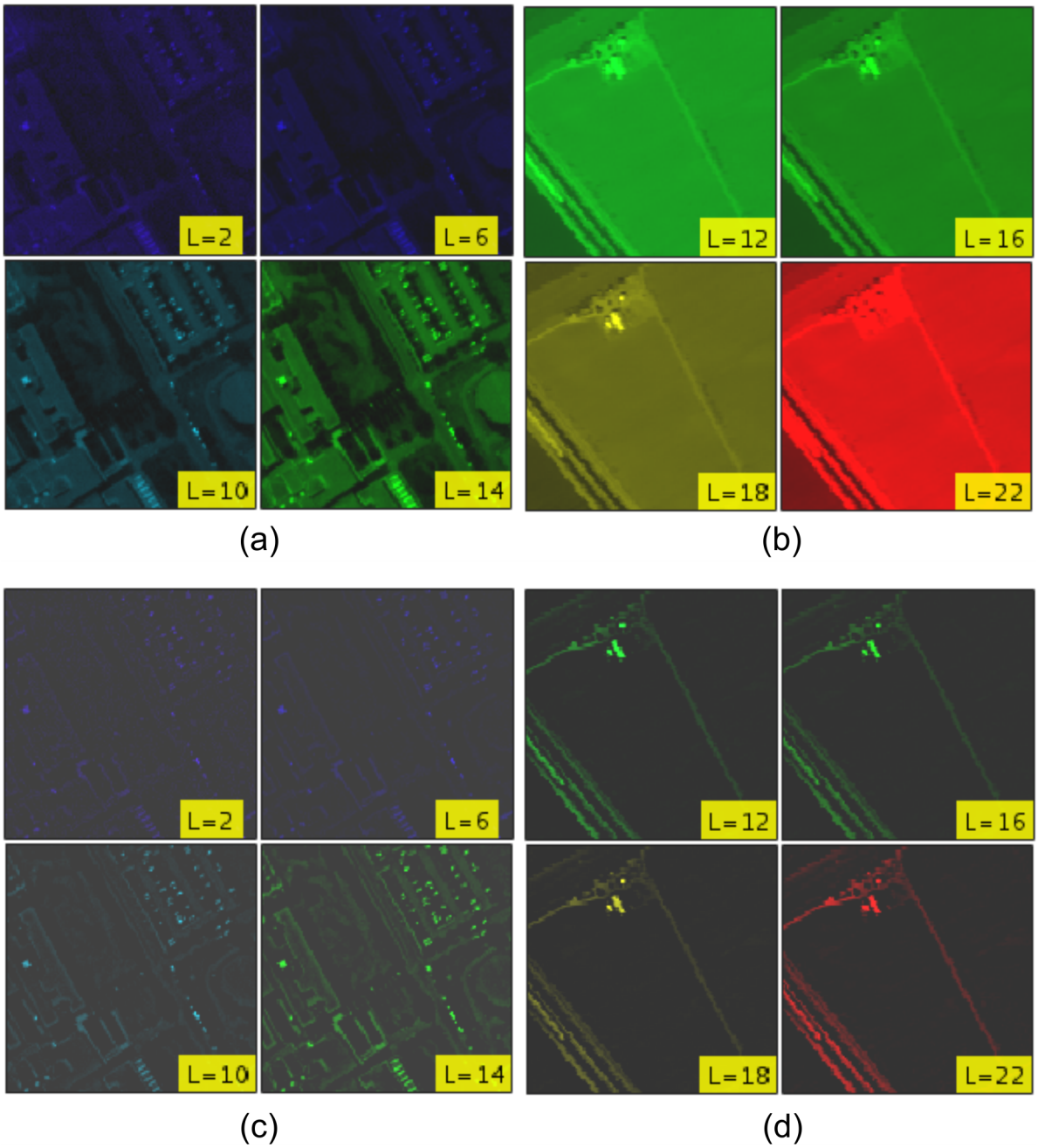


Figure 10. Example spectral bands of the test images: Pavia University (a) full frequencies and (c) high frequencies; Salinas (b) full frequencies and (d) high frequencies.

Test Images. The test images were $128 \times 128 \times 16$ sections of the Pavia University and Salinas spectral datasets, plus its high-frequencies versions (see Figure 10). Wolhberg *et al.* (Wohlberg, 2016c) reported that the convolutional dictionaries perform better when reconstructing the high frequency components of an image. Therefore, the high-frequency components for the SIs sections were extracted using a high pass filter, and included as test images.

Performance Evaluation. The recovery performances were compared against two (2) state-of-the-art techniques: a full-basis approach and a synthetic signal-based approach, as listed bellow

- Arce *et. al.* Kronecker basis (Arce et al., 2014), based in the SSR framework.
- Foi *et. al.* 3-D transform-domain collaborative filtering BM3D (Dabov et al., 2007).

The performance metrics used were the Peak Signal to Noise Ratio (PSNR) for measuring the recovery quality, and the ℓ_0 norm for measuring the sparsity of the solutions. Considering that the convolutional coefficient maps are in fact a collection of M_d sparse cubes $\mathbb{R}^{M\times N\times L\times M_d}$, compared to the single sparse cube $\mathbb{R}^{M\times N\times L}$ of the Kronecker basis, then the sparsity of the convolutional coefficient maps will be measured as

$$\text{sparsity} = \max_{m=1}^{M_d} \|\mathbf{X}_m\|_0 \,. \tag{64}$$

This is, the sparsity of the convolutional solutions will be the maximum sparsity of the individual coefficient maps.

Initializations and Regularizer Parameters. The initial values for the collection of coefficients were set to zero for the three schemes (CSC3D, SSR y BM3D), and the initial dictionary was established as a collection of random $M_d = 30$ dictionary elements of cubic size $d = 8$ for CSC3D. The proposed algorithm has proven to be sensible to the initialization of the dictionary, with the heuristic initialization strategy with the best results in terms of PSNR as

$$\mathbf{D}_m(i,j,k) = \begin{cases} \mathcal{N}(0,1) & \text{if } 3 \leq i,j,k \leq 6 \\ 0 & \text{otherwise.} \end{cases} \tag{65}$$

Results. Figure 11 shows that, when dealing with natural SIs, the proposed 3D CSC algorithm is able to match and outperform the denoising performance of the both the Kronecker basis, BM3D, and BM4D at 15dB and 20dB SNR levels (see Figure 11(a) and 11(b)). However, when dealing with higher levels of noise, 10dB SNR, the recovery quality drops. This is due to the effect of higher levels of noise in Eq. (43), affecting the estimation of the dictionary elements, leading to possible over fitting issues. When dealing with the high frequencies versions of the datasets, the proposed 3D CSC outperform the state-of-the-art at all noise levels by up to 5dB (see Figure 11(c) and 11(d)). For more information about the performance of the proposed CSC3D, please refer to Appendix 5.

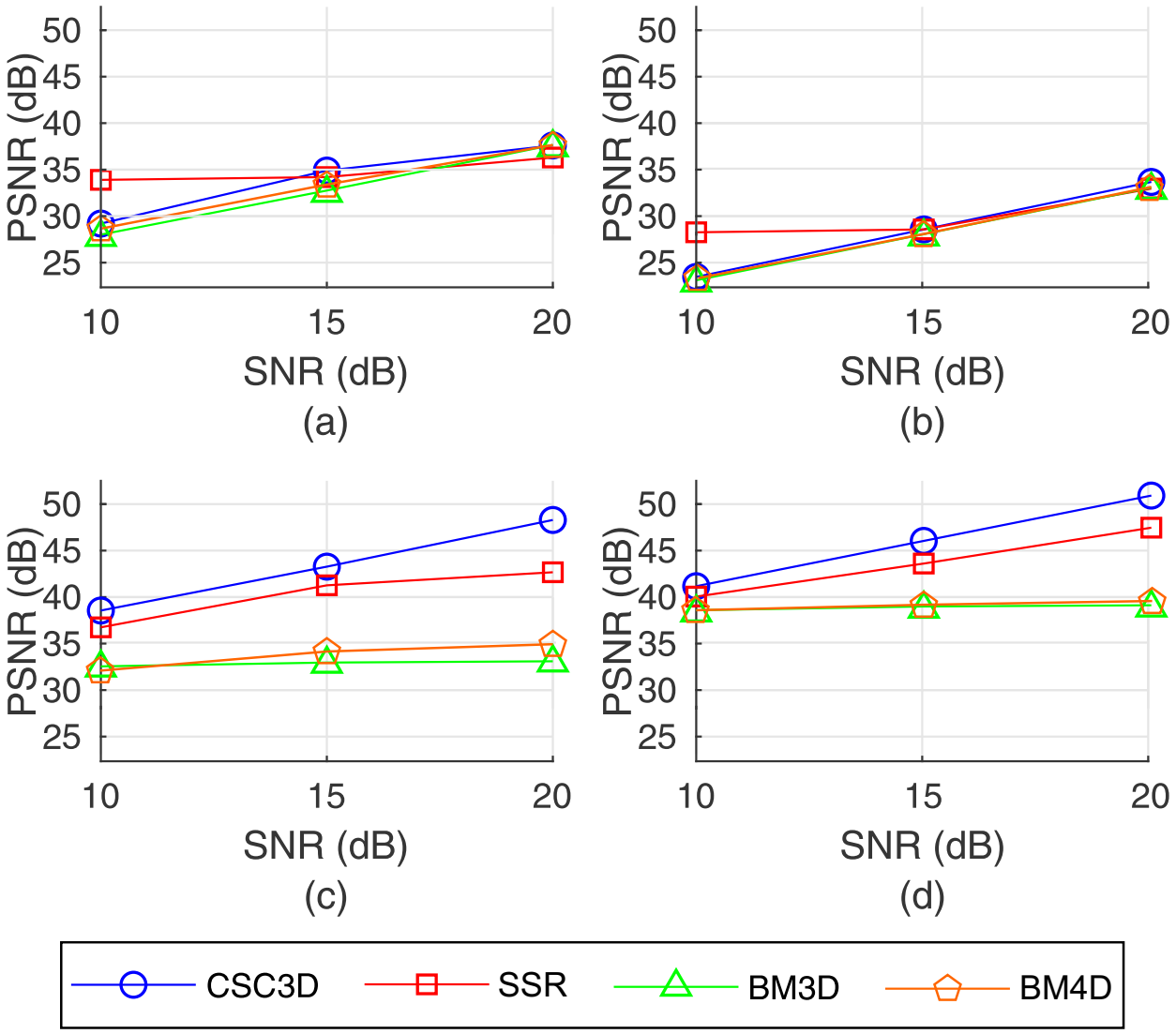


Figure 11. Recovery quality of the simulated results using the Peak Signal-to-Noise Ratio (PSNR) metric, at three different noise levels for the Pavia (a) full (b) high frequencies and Salinas (c) full (d) high frequencies test images. The proposed CSC3D is compared against state-of-the-art denoising techniques as SSR, BM3D, and DM4D.

5.4. Convolutional Sparse Coding for Compressive Spectral Imaging (CSC3D-CSI)

The main objective of this doctoral dissertation is to profit the properties of the CDSR framework for recovering a SI from compressed measurements. This means, to change the SSR based representation basis $\mathbf{\Psi}$ proposed by Arce *et. al.* in

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{H}\mathbf{\Psi}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1 , \tag{66}$$

by a CSDR model. In (Barajas-Solano et al., 2019b) we proposed the convex minimization problem

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\|\mathbf{H}\bar{\mathbf{D}}\mathbf{x} - \mathbf{y}\right\|_2^2 + \lambda \|\mathbf{x}\|_1 , \tag{67}$$

as an expansion of the formulation of CUP in Eq. (39). Eq. (67) includes the sensing matrix $\mathbf{H} \in \mathbb{R}^{K \times MNL}$ and the compressive measurements $\mathbf{y} = \mathbf{H}\text{vec}(\mathscr{S}) \in \mathbb{R}^K$, with $K < MNL$. Eq. (67) aims to learn a collection of sparse coefficients $\mathbf{x} = \text{vec}(\{\mathscr{X}_m,\ m = 1,...,M_d \mid \mathscr{X}_m \in \mathbb{R}^{M \times N \times L}\})$ from some compressed measurements $\mathbf{y}$, using a fixed collection of convolutional dictionary elements $\bar{\mathbf{D}} = f(\{\mathscr{D}_m,\ m = 1,...,M_d \mid \mathscr{D}_m \in \mathbb{R}^{d_M \times d_N \times d_L}\})$, in order to recover a SI of interest $\mathscr{S} \in \mathbb{R}^{M \times N \times L}$. As explained in section 4.1., operator $\bar{\mathbf{D}}$ can be treated as a sparsifying operator but should not be considered as a basis under any circumstance.

The complimentary problem, obtaining a collection of convolutional dictionary elements $\mathbf{d} = \text{vec}(\{\mathscr{D}_m,\ m = 1,...,M_d \mid \mathscr{D}_m \in \mathbb{R}^{d_M \times d_N \times d_L}\})$ from a collection of sparse coefficient $\bar{\mathbf{X}} =$

$f(\{\boldsymbol{\mathscr{X}}_m,\ m=1,...,M_d \mid \boldsymbol{\mathscr{X}}_m \in \mathbb{R}^{M\times N\times L}\})$ maps can be formulated as

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{H}\bar{\mathbf{X}}\mathbf{d}-\mathbf{y}\right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{d}), \tag{68}$$

with $\iota_{\mathscr{C}_Z}$ defined in Eq. (41) and (42).

Coefficients Update Problem for CSI (CUP-CSI). Eq. (67) is referred to as the *Coefficients Update Problem for CSI* (CUP-CSI) and can be solved using the alternating directions multiplier method, ADMM (Boyd et al., 2010), just as the original CSD3D problem. We begin by introducing two auxiliary **u** and **v** variables as

$$\begin{aligned} &\underset{\mathbf{x},\mathbf{u},\mathbf{v}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\mathbf{y}\right\|_2^2 + \lambda\left\|\mathbf{v}\right\|_1, \\ &\text{s.t.: } \mathbf{u}=\bar{\mathbf{D}}\mathbf{x}, \\ &\mathbf{v}=\mathbf{x}. \end{aligned} \tag{69}$$

The augmented Lagrangian for Eq. (69) can be written as

$$\mathscr{L}\{\mathbf{x},\mathbf{u},\mathbf{v},\mathbf{f},\mathbf{g}\} = \frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\mathbf{y}\right\|_2^2 + \lambda\left\|\mathbf{v}\right\|_1 + \frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}+\mathbf{f}\right\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}+\mathbf{g}\right\|_2^2, \tag{70}$$

where **f** and **g** are the so called dual variables. The variable updates are obtained from Eq. (70) as

$$\mathbf{x}^{j+1} := \underset{\mathbf{x}}{\operatorname{argmin}} \frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}^j+\mathbf{f}^j\right\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}^j+\mathbf{g}^j\right\|_2^2, \tag{71}$$

$$\mathbf{u}^{j+1} := \underset{\mathbf{u}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{H}\mathbf{u} - \mathbf{y}\|_2^2 + \frac{\rho}{2} \left\|\bar{\mathbf{D}}\mathbf{x}^{j+1} - \mathbf{u} + \mathbf{f}^j\right\|_2^2, \tag{72}$$

$$\mathbf{v}^{j+1} := \underset{\mathbf{v}}{\operatorname{argmin}} \frac{\rho}{2} \left\|\mathbf{x}^{j+1} - \mathbf{v} + \mathbf{g}^j\right\|_2^2 + \lambda \|\mathbf{v}\|_1, \tag{73}$$

$$\mathbf{f}^{j+1} = \mathbf{f}^j + \bar{\mathbf{D}}\mathbf{x}^{j+1} - \mathbf{u}^{j+1}, \tag{74}$$

$$\mathbf{g}^{j+1} = \mathbf{g}^j + \mathbf{x}^{j+1} - \mathbf{v}^{j+1}. \tag{75}$$

The dual variables $\mathbf{f}$ and $\mathbf{g}$ can be interpreted as vectors of prices, and Eqs. (74) and (75) are called *price updates* or *price adjustment steps*. The dual variables are updated separately to drive the variables into consensus, and the quadratic regularization helps pulling the variables toward their average value while still attempting to minimize variables $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{v}$ (Boyd et al., 2010). The solutions to subproblems (71) to (73) are presented bellow.

Coefficient Maps Update. Eq. (71) is similar in its formulation to Eq. (46), with a minor difference in the first ℓ_2 element. Then, just as Eq. (46), Eq. (71) can be solved efficiently in the Fourier Domain by performing the following rearrangements. First, lets replace $\mathbf{z}^j = \mathbf{u}^j - \mathbf{f}^j \in \mathbb{R}^{MNL}$ and create $\hat{\mathbf{z}}^j \in \mathbb{C}^{MNL}$ by folding, transforming to Fourier and unfolding following Algorithms 7, 9, and 3 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$. Second, lets replace $\mathbf{w}^j = \mathbf{v}^j - \mathbf{g}^j \in \mathbb{R}^{MNLM_d}$ and create $\hat{\mathbf{w}} \in \mathbb{C}^{MNLM_d}$ by folding, transforming to Fourier and unfolding following Algorithms 8, 10, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$. Finally, we can rewrite Eq.

(71) in the Fourier domain as

$$\hat{\mathbf{x}}^{j+1} := \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| \hat{\tilde{\mathbf{D}}} \hat{\mathbf{x}} - \hat{\mathbf{z}}^{j} \right\|_2^2 + \frac{\rho}{2} \left\| \hat{\mathbf{x}} - \hat{\mathbf{w}}^{j} \right\|_2^2. \tag{76}$$

Solving the derivative of Eq. (76) and equaling to zero, the solution to the optimization problem is

$$\hat{\mathbf{x}}^{j+1} = \left(\hat{\tilde{\mathbf{D}}}^{H} \hat{\tilde{\mathbf{D}}} + \mathbf{I} \right)^{-1} \left(\hat{\tilde{\mathbf{D}}}^{H} \hat{\mathbf{z}}^{j} + \hat{\mathbf{w}}^{j} \right), \tag{77}$$

which can be solved directly by using a combination of the Woodbury formula and exploiting the concatenated structure of $\hat{\tilde{\mathbf{D}}}$, as explained in Appendix 3, with solution

$$\hat{\mathbf{x}}^{j+1} = \mathbf{b} - \hat{\tilde{\mathbf{D}}}^{H} \left(\mathbf{I} + \hat{\tilde{\mathbf{D}}} \hat{\tilde{\mathbf{D}}}^{H} \right)^{-1} \hat{\tilde{\mathbf{D}}} \mathbf{b}, \tag{78}$$

with $\mathbf{b} = \hat{\tilde{\mathbf{D}}}^{H} \hat{\mathbf{z}}^{j} + \hat{\mathbf{w}}^{j} \in \mathbb{C}^{MNLM_d}$. Finally, the update $\mathbf{x}^{j+1}$ is obtained from $\hat{\mathbf{x}}^{j+1}$ by folding, obtaining the inverse transform and unfolding following Algorithms 8, 12, and 4 respectively with $L_1 \times \ldots \times L_{\bar{N}} = M \times N \times L$.

Note that the solution for $\hat{\mathbf{x}}^{j+1}$ in both CUP and CUP-CSI are similar, with a minor difference in the derivation of the auxiliary variable $\mathbf{b}$. This similarity is due to a couple of factors. First, the ADMM method generates simpler $\ell_2 - \ell_2$ minimizations for both CUP and CUP-CSI. Second, the proposed CSC3D, and its linear representation, is compact and modular enough to be integrated within the ADMM method. The modularity and formulation simplicity of the CSC3D

operator is one of its strongest advantages.

Temporal Recovery From Compressed Measurements Update. Unlike Eq. (71), Eq. (72) can be solved directly in the spatial domain by factorizing its derivative as

$$\mathbf{u}^{j+1} = \left(\mathbf{H}^T\mathbf{H} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{H}^T\mathbf{y} + \rho\mathbf{z}^j\right), \tag{79}$$

where $\mathbf{z}^j = \bar{\mathbf{D}}\mathbf{x}^{j+1} + \mathbf{f}^j \in \mathbb{R}^{MNL}$. The inverse of matrix $\mathbf{H}^T\mathbf{H} + \rho\mathbf{I} \in \mathbb{R}^{MNL\times MNL}$ is expensive to compute directly, so Eq. (79) is solved using Woodbury's matrix identity, as explained in Appendix 2, as

$$\mathbf{u}^{j+1} = \frac{1}{\rho}\left[\mathbf{b} - \mathbf{H}^T\left(\rho\mathbf{I} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{b}\right], \tag{80}$$

with $\mathbf{b} = \mathbf{H}^T\mathbf{y} + \rho\mathbf{z}^j \in \mathbb{R}^{MNL}$.

Sparse Coefficient Maps Update. Eq. (73) has a closed form solution via soft thresholding (Rockafellar, 1970) as

$$\mathbf{v}^{j+1} = \mathscr{S}_{\frac{\lambda}{\rho}}\left(\mathbf{x}^{j+1} + \mathbf{g}^j\right). \tag{81}$$

The CUP-CSI solution is summarized in Algorithm 16.

Dictionary Update Problem for CSI (DUP-CSI). Eq. (68) is referred to as the *Dictionary Update Problem for CSI* (DUP-CSI) and can also be solved using the alternating directions

Algorithm 16 CUP for CSI Solution

Require: $\bar{\mathbf{D}}$, $\hat{\bar{\mathbf{D}}}$, $\mathbf{H}$, $\mathbf{u}^j$, $\mathbf{f}^j$, $\mathbf{v}^j$, $\mathbf{g}^j$, ρ, λ and sizes M, N, L and M_d.
1: Build $\hat{\mathbf{w}}^j$ and $\hat{\mathbf{z}}^j$.
2: Solve $\mathbf{x}^{j+1}$ in Eq. (71) using (Eq. (78).
3: Solve $\mathbf{u}^{j+1}$ in Eq. (72) using (Eq. (79).
4: Solve $\mathbf{v}^{j+1}$ Eq. (73) using (Eq. (81).
5: Solve $\mathbf{f}^{j+1}$ and $\mathbf{g}^{j+1}$ in Eq. (74) and Eq. (75), respectively.
6: **return** Sparse coefficient maps $\mathbf{v}^{j+1}$, split variable updates $\mathbf{u}^{j+1}$ and $\mathbf{v}^{j+1}$, and dual variable updates $\mathbf{f}^{j+1}$ and $\mathbf{g}^{j+1}$.

multiplier method, ADMM (Boyd et al., 2010), just as the CUP-CSI problem. We begin by introducing two auxiliary variables $\mathbf{p}$ and $\mathbf{q}$ as

$$\begin{aligned} &\underset{\mathbf{d},\mathbf{p},\mathbf{q}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{H}\mathbf{p}-\mathbf{y}\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \\ &\text{s.t.: } \mathbf{p} = \bar{\mathbf{X}}\mathbf{d}, \\ &\mathbf{q} = \mathbf{d}. \end{aligned} \tag{82}$$

The augmented lagrangian for Eq. (82) is expressed as

$$\mathscr{L}\{\mathbf{d},\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{t}\} = \frac{1}{2}\|\mathbf{H}\mathbf{p}-\mathbf{y}\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}) + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{p}+\mathbf{r}\right\|_2^2 + \frac{\sigma}{2}\|\mathbf{d}-\mathbf{q}+\mathbf{t}\|_2^2, \tag{83}$$

and the ADMM iterations as follows

$$\mathbf{d}^{j+1} := \underset{\mathbf{d}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{p}^j+\mathbf{r}^j\right\|_2^2 + \frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}^j+\mathbf{t}^j\right\|_2^2, \tag{84}$$

$$\mathbf{p}^{j+1} := \underset{\mathbf{p}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{H}\mathbf{p}-\mathbf{y}\|_2^2 + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}^{j+1}-\mathbf{p}+\mathbf{r}^j\right\|_2^2, \tag{85}$$

$$\mathbf{q}^{j+1} := \underset{\mathbf{q}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \mathbf{d}^{j+1} - \mathbf{q} + \mathbf{t}^{j} \right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \tag{86}$$

$$\mathbf{r}^{j+1} = \mathbf{r}^{j} + \bar{\mathbf{X}}\mathbf{d}^{j+1} - \mathbf{p}^{j}, \tag{87}$$

$$\mathbf{t}^{j+1} = \mathbf{t}^{j} + \mathbf{d}^{j+1} - \mathbf{t}^{j}. \tag{88}$$

Just as with CUP-CSI, the dual variables $\mathbf{r}$ and $\mathbf{t}$ are interpreted as vectors of prices, and are updated separately to drive the variables $\mathbf{d}$, $\mathbf{p}$, and $\mathbf{q}$ into consensus (Boyd et al., 2010). The solutions to subproblems (72) to (86) are presented bellow.

Convolutional Dictionary Update. Eq. (84) is akin to Eq. (71) and can be can be solved efficiently by profiting the DFT convolution theorem by rewriting it in the Fourier domain as

$$\hat{\mathbf{d}}^{j+1} := \underset{\hat{\mathbf{d}}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \hat{\bar{\mathbf{X}}}\hat{\mathbf{d}} - \hat{\mathbf{z}}^{j} \right\|_2^2 + \frac{\sigma}{2} \left\| \hat{\mathbf{d}} - \hat{\mathbf{w}}^{j} \right\|_2^2. \tag{89}$$

Just as with the CUP-CSI update steps, matrix $\hat{\bar{\mathbf{X}}}$ is built from the solution $\mathbf{v}^{j+1}$ in Eq. (81) by folding, transforming to Fourier, and creating the equivalent operator following Algorithms 8, 10, and 6 respectively. Second, we replace $\mathbf{z}^{j} = \mathbf{p}_l^{j} - \mathbf{r}_l^{j}$ and create $\hat{\mathbf{z}}^{j}$ by folding, transforming to Fourier and unfolding following Algorithms 7, 9, and 3 respectively. Third, we replace $\mathbf{w}^{j} = \mathbf{q}_l^{j} - \mathbf{t}_l^{j}$ and create $\hat{\mathbf{w}}^{j}$ by folding, transforming to Fourier and unfolding following Algorithms 8, 10, and 4 respectively. Creating $\hat{\bar{\mathbf{X}}}$, $\hat{\mathbf{z}}^{j}$, and $\hat{\mathbf{w}}^{j}$ uses $Ł_1 \times ... Ł_{\bar{N}} = M \times N \times L$. Finally, solving the derivative

and equaling to zero, the solution to Eq. (89) is obtained as

$$\hat{\mathbf{d}}^{j+1} = \left(\hat{\hat{\mathbf{X}}}^H \hat{\hat{\mathbf{X}}} + \mathbf{I}\right)^{-1} \left(\hat{\hat{\mathbf{X}}}^H \hat{\mathbf{z}}_l^j + \hat{\mathbf{w}}^j\right). \tag{90}$$

Matrix $\hat{\hat{\mathbf{X}}}$ have the same concatenated diagonal structure as $\hat{\hat{\mathbf{D}}}$ in CUP-CSI, then Eq. (90) can be solved directly, using the optimized routine explained in Appendix 3, as

$$\hat{\mathbf{d}}^{j+1} = \mathbf{b} - \hat{\hat{\mathbf{X}}}^H \left(\mathbf{I} + \hat{\hat{\mathbf{X}}} \hat{\hat{\mathbf{X}}}^H\right)^{-1} \hat{\hat{\mathbf{X}}} \mathbf{b}, \tag{91}$$

with $\mathbf{b} = \hat{\hat{\mathbf{X}}}^H \hat{\mathbf{z}}_l^j + \hat{\mathbf{w}}^j \in \mathbb{C}^{MNLM_d}$. Finally, the update $\mathbf{d}^{j+1}$ is obtained by folding, transforming from the Fourier domain and unfolding (Algorithms 8, 12, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L$).

Temporal Recovery From Compressed Measurements Update. The solution to Eq (85) is also obtained by applying factorization to its derivative in the spatial domain as

$$\mathbf{p}^{j+1} = \left(\mathbf{H}^T \mathbf{H} + \sigma \mathbf{I}\right)^{-1} \left(\mathbf{H}^T \mathbf{y} + \rho \mathbf{z}^j\right), \tag{92}$$

where $\mathbf{z}^j = \bar{\mathbf{X}} \mathbf{d}^{j+1} + \mathbf{r}^j \in \mathbb{R}^{MNL}$. As with Eq. (79), the matrix inversion cannot be solved directly due to its size, and must be solved using Woodbury's matrix identity, as explained in Appendix 2, as

$$\mathbf{p}^{j+1} = \frac{1}{\sigma} \left[\mathbf{b} - \mathbf{H}^T \left(\sigma \mathbf{I} + \mathbf{H} \mathbf{H}^T\right)^{-1} \mathbf{H} \mathbf{b}\right], \tag{93}$$

with $\mathbf{b} = \mathbf{H}^T\mathbf{y} + \sigma\mathbf{z}^j \in \mathbb{R}^{MNL}$. Note that Eq. (93) has the same structure than Eq. (80) due Eqs. (72) and (85) having the same structure. The only difference lies in the auxiliary $\mathbf{b}$ which is specific for both CUP-CSI and DUP-CSI.

Desired Convolutional Dictionary Update. Eq. (86) can be rewritten as

$$\mathbf{q}_m^{j+1} := \underset{\mathbf{q}_m}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \mathbf{d}_m^{j+1} - \mathbf{q}_m + \mathbf{t}_m^j \right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}_m), \tag{94}$$

$\forall\, m = 1, ..., M_d$. Solving via the proximal of $\iota_{\mathscr{C}_Z}$ (Rockafellar, 1970), the closed solution for Eq. (94) is

$$\mathbf{q}_m^{j+1} = \frac{\mathbf{Z}_p\mathbf{Z}_p^T(\mathbf{d}_m^{j+1} + \mathbf{t}_m^j)}{\left\| \mathbf{Z}_p\mathbf{Z}_p^T(\mathbf{d}_m^{j+1} + \mathbf{t}_m^j) \right\|_2}, \tag{95}$$

and $\mathbf{q}^{j+1} = [\mathbf{q}_1^{j+1^T} ... \mathbf{q}_{M_d}^{j+1^T}]^T$.

The DUP-CSI solution is summarized in Algorithm 17.

Proposed CSC3D-CSI Algorithm. The proposed Compressive Spectral Convolutional 3D for CSI (CSC3D-CSI) frame- work consists in alternately solving two related problems, CUP-CSI and DUP-CSI, for obtaining both a collection of sparse coefficient maps and convolutional dictionary elements from compressed spectral measurements. As in CSC3D, the regularization parameters can be updated in each iteration according to (Boyd et al., 2010), section 3.3, or they can

Algorithm 17 DUP for CSI Solution

Require: $\bar{\mathbf{X}}$, $\hat{\bar{\mathbf{X}}}$, $\mathbf{H}$, $\mathbf{p}^j$, $\mathbf{r}^j$, $\mathbf{q}^j$, $\mathbf{t}^j$, σ and sizes M, N, L and M_d.

1: Build $\hat{\mathbf{w}}^j$ and $\hat{\mathbf{z}}^j$.
2: Solve $\mathbf{d}^{j+1}$ in Eq. (84) using Eq. (90).
3: Solve $\mathbf{p}^{j+1}$ in Eq. (85) using Eq. (93).
4: Solve $\mathbf{q}^{j+1}$ Eq. (86) using Eq. (95).
5: Solve $\mathbf{r}^{j+1}$ and $\mathbf{t}^{j+1}$ in Eq. (87) and Eq. (88), respectively.
6: **return** Convolutional dictionary $\mathbf{d}^{j+1}$, split variable updates $\mathbf{p}^{j+1}$, and $\mathbf{q}^{j+1}$, and dual variable updates $\mathbf{r}^{j+1}$ and $\mathbf{t}^{j+1}$.

Algorithm 18 Compressive Spectral Convolutional 3D for CSI Algorithm - CSC3D-CSI

Require: $\{\mathscr{X}^0_m\}$ as zeros $\in \mathbb{R}^{M\times N\times L\times M_d}$; $\{\mathscr{D}^0_m\}$ as random $\in \mathbb{R}^{d\times d\times d\times M_d}$; $\mathbf{H} \in \mathbb{R}^{K\times MNL}$; $M, N, L, M_d, d, \rho_0, \lambda$ and σ_0.

1: Set $\{\mathscr{V}^0_m\} = \{\mathscr{X}^0_m\}$ and build the vectorization $\mathbf{v}^0$.
2: Set $\{\mathscr{Q}^0_m\} = \{\mathscr{D}^0_m\}$ and build the vectorization $\mathbf{q}^0$.
3: Solve $\mathscr{S}^0 = \sum_{m=1}^{M_d} \mathscr{D}^0_m \overset{3}{*} \mathscr{X}^0_m$.
4: Set $\mathscr{U}^0 = \mathscr{P}^0 = \mathscr{S}^0$ and build the respective vectorizations $\mathbf{u}^0$ and $\mathbf{p}^0$.
5: Build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from $\{\mathscr{D}^0_m\}$.
6: Set $j = 0$.
7: **repeat**
8: Solve CUP-CSI as explained in Algorithm 16.
9: Update ρ^{j+1} according to (Boyd et al., 2010), section 3.3
10: Fold $\mathbf{v}^{j+1}$ into $\{\mathscr{V}^{j+1}_m\}$, as explained in Algorithm 7, and build $\bar{\mathbf{X}}$ and $\hat{\bar{\mathbf{X}}}$ from it.
11: Solve DUP-CSI as explained in Algorithm 17..
12: Update σ^{j+1} according to (Boyd et al., 2010), section 3.3
13: Fold $\mathbf{q}^{j+1}$ into $\{\mathscr{Q}^{j+1}_m\}$, as explained in Algorithm 7, and build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from it.
14: **until** the residuals meet a given tolerance, or completed a number of iterations.
15: **return** the sparse coefficient maps $\{\mathscr{V}^{j+1}_m\}$ and the convolutional dictionary elements $\{\mathscr{Q}^{j+1}_m\}$

be set to a fixed value. Thus, this doctoral dissertation propose to solve both problems alternately, as exposed in Algorithm 18.

A mayor change in CSC3D-CSI, when compared with CSC3D and CBPDN, is the introduction of a second split variable per problem, due to the presence of the matrix $\mathbf{H}$. The additional split variable simplifies the lineal formulation, generating an additional $\ell_2 - \ell_2$ convex optimization subproblem. Then, the proposed CSC3D-CSI framework solves two $\ell_2 - \ell_2$ minimizations per subproblem, compared to CSC3D and CBPDN which solves only one. Now, the additional $\ell_2 - \ell_2$ minimization has the same structure as the original one proposed in CBPDN, and does not required to be solved in the Fourier domain. This identical structure, and the performance of numerical experiments, leads to the conclusion that the convergence boundaries are not stretched far from the canonical formulation.

Estimated Numerical Complexity. We will now estimate the numerical complexity of the more complex subproblems in CUP-CSI and DUP-CSI. Subproblem (73)'s solution, Eq. (81), is obtained by a soft-thresholding problem and subproblem (86)'s solution, Eq. (95), is akin to a hard-thresholding problem. Both complexities are negligible.

Subproblems (72) and (85) have the same solution structure where the inversion of $\mathbf{H^T H} + \alpha\mathbf{I} \in \mathbb{R}^{MNLT \times MNLT}$ represents the higher complexity. According to the Appendix A of (Barajas-Solano et al., 2019a), their complexity can be reduced from $\mathscr{O}((MNL)^3)$ to $\mathscr{O}(K^2MNL)$, with

Eq.	Complexity	Solution
(73) and (86)	Negligible	Implemented
(72) and (85)	$\mathscr{O}((MNL)^3)$	Original
(80) and (93)	$\mathscr{O}(K^2MNL)$	Implemented
(71) and (84)	$\mathscr{O}((MNLM_d)^3)$	Original
(78) and (91)	$\mathscr{O}(MNLM_d)$	Implemented

Table 3
Complexity review of the proposed CSC3D+CSI algorithm.

$K \ll MNL$ and $\mathbf{H} \in \mathbb{R}^{K \times MNLT}$, by profiting on $\mathbf{H}\mathbf{H}^T = \mathbf{I}$.

One of the most important sources of numerical complexity is the 3D convolutional operation, with complexity $\mathscr{O}((MNL)^2)$, considering that the dictionary elements are zero-padded to match the dimensions of the sparse coefficient maps. The 3D convolution complexity is reduced to a fraction by expressing it as a Hadamard product, profiting on the Discrete Fourier Transform (DFT) Theorem, reducing the cost to $\mathscr{O}(MNL\log(MNL))$.

Finally, the greatest source of numerical complexity are the inversions in Eqs. (77) and (90), as solutions to subproblems (71) and (84). Again, the canonical complexity for inverting $\hat{\mathbf{A}}^H\hat{\mathbf{A}} + \alpha\mathbf{I} \in \mathbb{R}^{MNLM_d \times MNLM_d}$ is $\mathscr{O}((MNLM_d)^3)$. However, by profiting on the concatenated diagonal structure of $\hat{\hat{\mathbf{D}}}$ and $\hat{\hat{\mathbf{X}}}$ and the dimensions rearrangement exposed in Appendix B of (Barajas-Solano et al., 2019a), the inversion complexity falls to $\mathscr{O}(MNL\ M_d)$. Table 3 summarizes the different complexities for the stated subproblems and their solutions.

5.5. CSC3D-CSI Synthetic Performance Evaluation

The performance evaluation of the proposed CSC3D-CSI algorithm was carried out following a four-step scheme:

A. Sparse Representation. First, we evaluate the performance of the proposed CSC3D-CSI algorithm for $\mathbf{H} = \mathbf{I}$. The inclusion of matrix $\mathbf{H}$ introduces a second $\ell_2 - \ell_2$ minimization, increasing considerably the proposed solution complexity. It is necessary to measure the impact of the second $\ell_2 - \ell_2$ minimization in a controlled simulation, and verify that CSC3D-CSI is able to represent SIs. The performance of the proposed CSC3D-CSI algorithm is compared to the CBPDN and the SSR model. More precisely, we use the Kronecker basis as the representation basis for recovering an SI from compressive spectral measurements, using an ADMM based algorithm (Boyd et al., 2010).

The performance comparison is realized by computing the Peak Signal-to-Noise Ratio (PSNR, dB) for assessing the reconstruction quality; while the percentage of non-zero elements (NNZ, %) is used for evaluating the sparsity of the estimated coefficients. NNZ can be also described as $\|\mathbf{x}\|_0 / |\mathbf{x}|$ for the estimated coefficients vector using the SSR model, where $|\mathbf{x}|$ is the total number of coefficients. For the case of CSC3D-CSI and CBPDN, the coefficients comprise a collection of M_d elements. The sparsity of CSC3D-CSI's and CBPDN's coefficients is measured

according to (Papyan et al., 2017) as

$$\text{NNZ} = \max_{m=1}^{M_d} \frac{\|\mathbf{x}_m\|_0}{|\mathbf{x}_m|}. \tag{96}$$

B. Recovery from 3D-CASSI measurements. We evaluate the performance of the proposed CSC3D-CSI from compressive spectral measurements using the 3D-CASSI architecture (Cao et al., 2016). The quality of the recovered SI is assessed for different noise levels and various compression ratios. The performance of the proposed CSC3D-CSI algorithm is compared to four different state-of-the-art methods: a sparse signal model based method (SSR), the modified version of the Approximate Message Passing method (Tan et al., 2016), a simplified version of an unmixing approach (Vargas et al., 2019), and the low-rank minimization scheme (Gelvez et al., 2017).

C. Recovery from C-CASSI measurements. We evaluate the performance of the proposed CSC3D-CSI from compressive spectral measurements using the C-CASSI architecture (Arce et al., 2014). Preliminary tests showed low performances of the proposed CSC3D-CSI algorithm with the C-CASSI architecture. For this reason we propose a side-information acquisition scheme, based on (Galvis et al., 2017; Yuan et al., 2015). The performance of the CSC3D-CSI algorithm along the side-information scheme is assessed for different noise levels and various compression ratios, and compared to the four state-of-the-art recovery methods.

D. Robustness to acquisition noise. The proposed Gaussian noise follows an ideal estimation of the nature of the acquisition noise. The final step is to evaluate the performance of the proposed CSC3D-CSI algorithm with a more realistic noise model. The proposed algorithm is also compared against state-of-the-art recovery methods for comparison.

The test images used for the performance evaluation are sections of the Pavia University (Gamba, 2004), Salinas Valley (del Pais Vasco, 2012), and Beads SIs (Yasuma et al., 2008) datasets. More precisely, each SI section has a size of 128×128 pixels and 16 spectral bands. Considering that convolutional dictionaries have a low performance for representing low-frequency components of multidimensional signals (Wohlberg, 2016a), this work uses the high-frequencies versions of the original datasets, which are obtained by performing a high-pass filtering stage to the image data. For illustrative purposes, Fig. 12 displays some bands of the datasets under test and their respective high-frequency versions. The test methods Unmixing and Low-Rank require the number of endmembers as input. Considering that these work uses the high frequency versions of the selected datasets, then the number of endmembers were estimated according to (Bioucas-Dias et al., 2008) and (Bacca et al., 2019) as 7 for Pavia, 3 for Salinas, and 4 for Beads.

CSC3D-CSI Sparse Representation Performance. The first evaluation scheme focuses on assessing the performance of the proposed CSC3D-CSI algorithm for sparsely representing an SI. For this case, the sensing matrix is reduced to the identity matrix, i.e. $\mathbf{H} = \mathbf{I}$. In particular, the high-frequency versions of the SI under tests are recovered using the proposed algorithm, where a

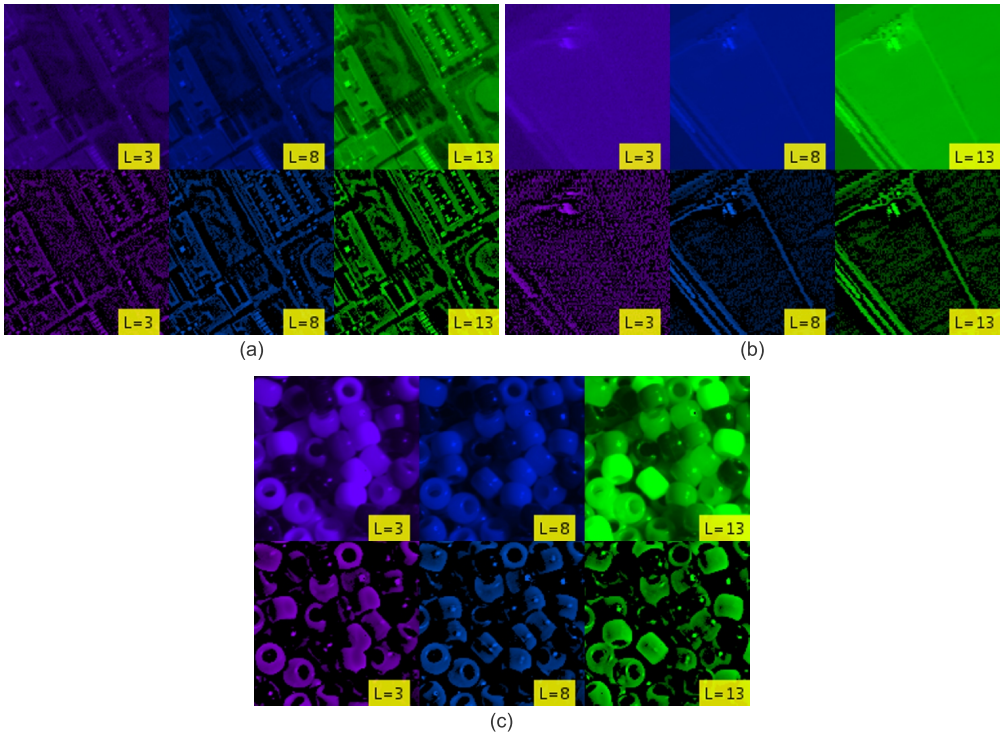


Figure 12. Example spectral bands recovered using the proposed CSC3D-CSI framework of the (a) Pavia University, (b) Salinas, and (c) Beads test datasets, and its corresponding high-frequencies-only versions.

reconstructed version of the image under test is obtained by adding the low-frequency components to the previously recovered high-frequency image.

For comparison purposes, Fig. 13 shows the evolution of the PSNR and NNZ of the proposed CSC3D-CSI as the number of iterations increases. Note that the proposed CSC3D-CSI algorithm outperforms both the SSR based approach and the CBPDN algorithm. Specifically, the proposed algorithm provides a gain up to 10dB, keeping the sparsity at competitive levels. For illustrative purposes, Fig. 14 displays some bands of the reconstructed SI using the proposed CSC3D-CSI algorithm, with their respective PSNR values.

Fig. 15 shows the effect of size of the dictionary elements $d = \{6, 8, 10\}$ and the number of dictionary elements $M_d = \{20, 30, 40\}$ on the reconstruction quality. The PSNR values are obtained by averaging 5 realizations for each combination $d - M_d$, where for each trial a different initial set of convolutional dictionaries $\{\mathscr{D}_m^0\}$ are generated. The number of dictionary elements affects directly the computational time of the proposed algorithm, so a small value for M_d is preferable. Considering those results, we use a collection of 30 convolutional dictionary elements of size $8 \times 8 \times 8$.

Performance Using the 3D-CASSI CSI Architecture. The second step in the evaluation scheme is to test the performance of the proposed CSC3D-CSI algorithm for recovering an SI from compressive measurements using the 3D-CASSI architecture (Cao et al., 2016). More precisely,

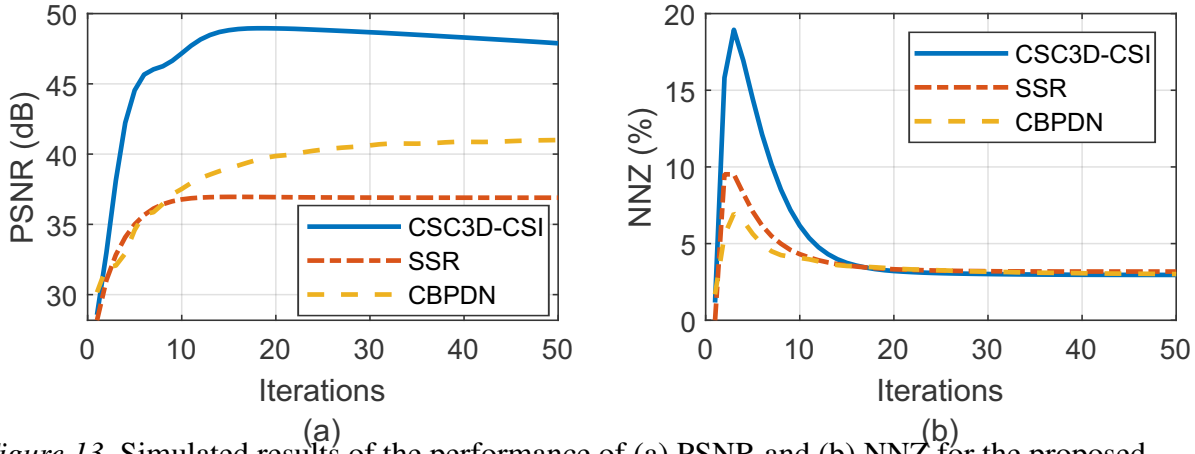


Figure 13. Simulated results of the performance of (a) PSNR and (b) NNZ for the proposed CSC3D-CSI algorithm, the Kronecker basis (SSR), and the CBPDN algorithm.

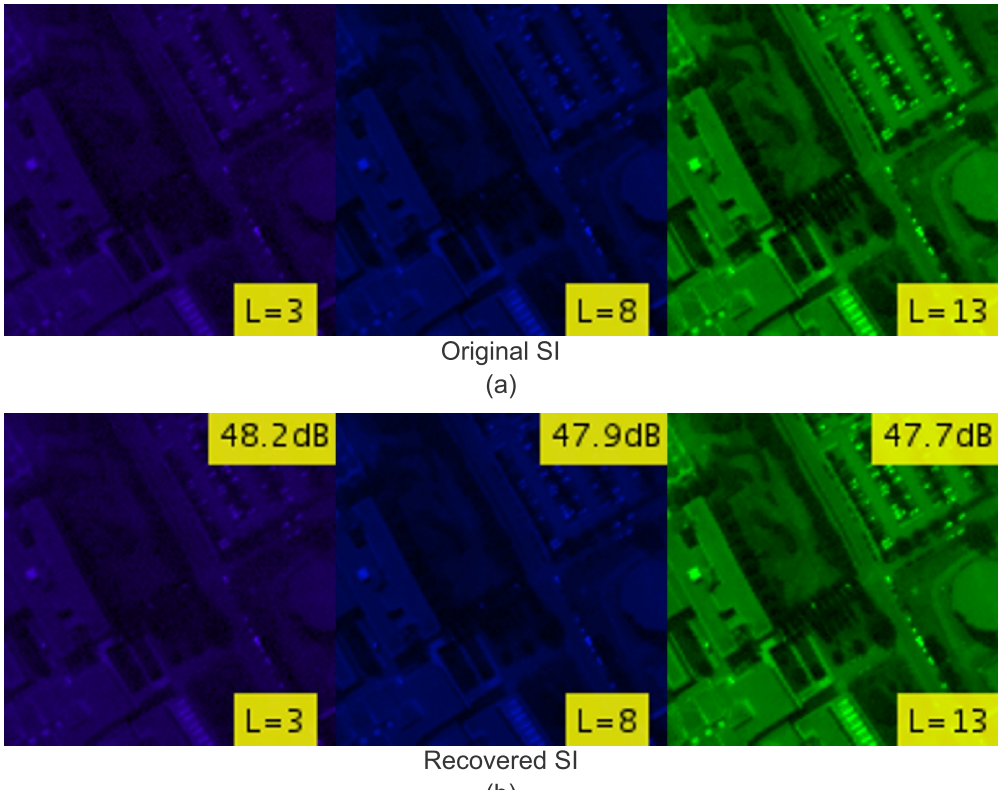


Figure 14. Simulated results of the reconstruction quality for some chosen bands using the proposed CSC3D-CSI algorithm.

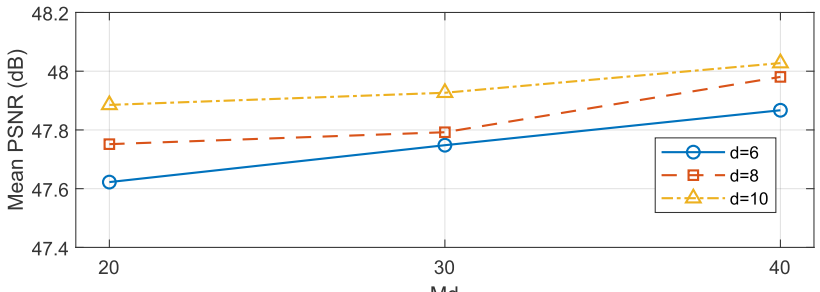
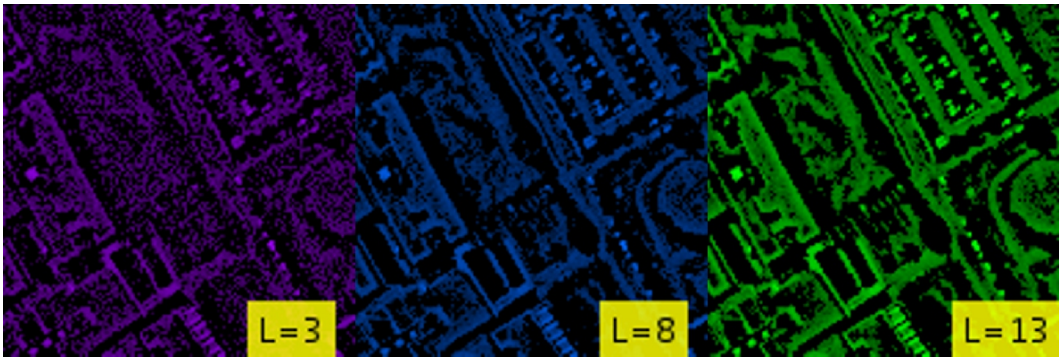


Figure 15. Mean PSNR of the simulation results at attempting to recover the Pavia scene at various values of d and Md, using the proposed CSC3D-CSI. The standard deviations were well bellow 1 % of the mean PSNR, thus are not shown.

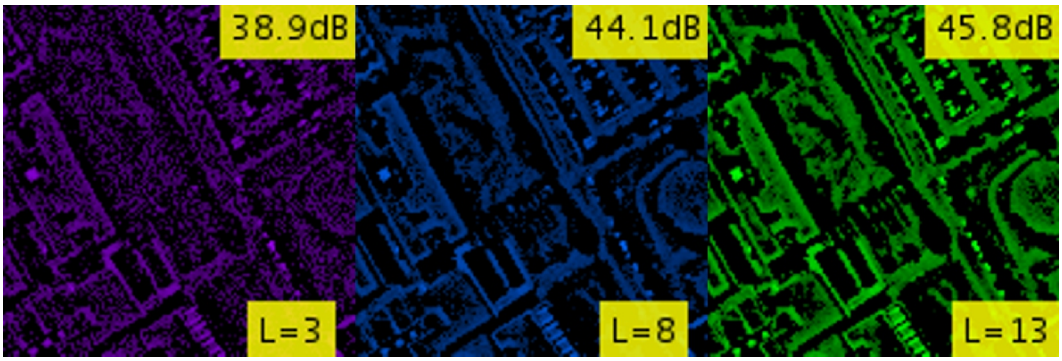
we evaluate the proposed algorithm over the high-frequency versions of the datasets, which can be experimentally obtained using an optical architecture that filters out an SI's low-frequency components. First, the proposed algorithm was tested without noise using $K = 4$ shots, which corresponds to a compression ratio of 0.25 given by the formula $\gamma = KMN/MNL$ (Cao et al., 2016). Fig. 16 exhibits some recovered bands using the proposed algorithm with their corresponding PSNR.

Second, the proposed algorithm was tested for various levels of additive noise and various levels of compression. Fig. 17 shows the effect of both the compression ratio and SNR levels over the recovery quality. For different levels of compression the SNR of the compressive measurements is fixed to 20dB. For different levels of noise the compression ratio is fixed to 4 shots.

For comparison purposes, the performance of the recovered SI using the the test methods are also displayed. The proposed algorithm outperforms the test methods, at a fixed noise level of 20dB SNR, for the entire set of compression ratios (see Fig. 17.(a)), but falls behind the SSR approach when dealing with high levels of noise (see Fig. 17(b)). This can be due to the effect of the sensing noise over the DUP-CSI problem, decreasing the proposed algorithm performance. On



Original SI
(a)



Recovered SI
(b)

Figure 16. Simulation results of the recovery quality for some chosen bands from compressive spectral measurements using the proposed CSC3D-CSI algorithm and the CSI 3D-CASSI architecture with compression ratio of 0.25.

the other hand, the SSR based approach seems to reach a maximum level of performance at 15-20dB of SNR, and non-improving with lower noise levels. The proposed CSC3D-CSI algorithm does not show this behavior and improves its performance with low noise levels. It is worth noting the low performance of the Unimixing method. This particular method includes a Total Variation component that performs poorly with high-frequency images, decreasing its global performance.

Table 4 shows the performance comparison between the proposed CSC3D-CSI, the SSR

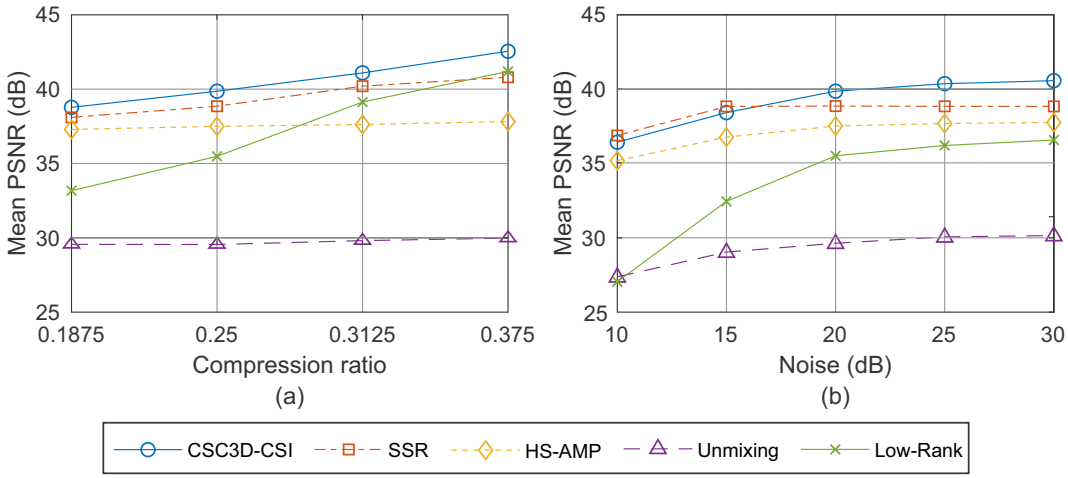


Figure 17. Mean PSNR of the simulation results at attempting to recover the Pavia SI using proposed CSC3D-CSI and the CSI 3D-CASSI architecture, for the various methods, (a) for fixed 20dB SNR and (b) fixed 4 shots. The standard deviations for five repetitions were well bellow 1 % of the mean PSNR, thus are not shown.

based approach, the HS-AMP, simplified Unfixing, and the Low-Rank methods, for five levels of noise and four different compression ratios (number of shots).

The SSR based approach performs better with high noise levels (10dB) and high compression ratios (3 shots), but the proposed CSC3D-CSI matches its performance when the noise levels decrease. The same behavior is observed for the 4 levels of compression, as with the non-increasing performance of the SSR approach above 20dB SNR. The proposed CSC3D-CSI outperforms all the state-of-the-art methods by up to 4dB with noise levels over 20dB SNR.

Performance Using the C-CASSI CSI Architecture. The third step in the performance evaluation is to test the performance of the proposed CSC3D-CSI algorithm for recovering an SI from compressive measurements obtained using the C-CASSI architecture. Preliminary tests sho-

		3 Shots (0.19)			4 Shots (0.25)			5 Shots (0.31)			6 Shots (0.38)		
		Pavia	Salinas	Beads	Pavia	Salinas	Beads	Pavia	Salinas	Beads	Pavia	Salinas	Beads
10dB	SSR	**37.36**	**45.20**	**33.38**	**36.86**	**45.66**	**34.68**	**38.17**	**46.27**	**35.00**	**38.81**	**45.67**	**34.82**
	CSC3D-CSI	35.30	42.05	31.91	36.40	44.67	32.44	36.44	44.88	33.22	37.38	45.26	33.81
	HS-AMP	34.85	41.68	31.83	35.15	42.09	33.49	35.39	42.27	33.18	35.43	42.69	33.74
	Unmixing	26.83	32.03	20.84	27.27	32.92	23.93	27.53	33.08	24.65	27.70	33.17	24.86
	Low-Rank	25.92	37.52	26.88	26.94	38.71	28.21	29.29	40.53	30.21	31.69	42.80	32.45
15dB	SSR	**37.92**	**45.63**	**35.07**	**38.83**	**46.82**	**35.12**	39.38	**46.85**	**35.62**	39.78	**47.72**	36.10
	CSC3D-CSI	37.60	44.74	34.45	38.41	46.18	35.11	**39.50**	46.21	35.55	**40.06**	47.11	**36.23**
	HS-AMP	36.59	44.29	34.12	36.75	44.54	35.87	37.16	44.86	35.97	37.15	45.57	36.49
	Unmixing	28.75	32.45	21.15	28.94	32.99	24.37	29.20	33.22	25.00	29.39	33.16	25.25
	Low-Rank	31.04	42.62	32.01	32.39	43.82	32.33	34.31	44.63	34.06	36.60	45.93	34.31
20dB	SSR	38.12	45.71	35.29	38.85	46.89	35.31	40.02	47.35	35.64	40.08	48.56	36.15
	CSC3D-CSI	**38.78**	**45.88**	**35.77**	**39.86**	**47.81**	**36.74**	**41.09**	**47.95**	**37.59**	**42.55**	**48.93**	**38.39**
	HS-AMP	37.29	45.65	35.16	37.50	46.39	36.41	37.62	46.78	37.49	37.82	47.25	38.34
	Unmixing	29.56	32.70	21.62	29.55	33.17	24.48	29.81	33.25	25.11	29.98	33.30	25.31
	Low-Rank	33.17	44.73	33.93	35.48	45.01	34.16	39.13	45.68	34.55	41.19	45.96	34.77
25dB	SSR	38.39	45.74	35.32	38.84	47.09	35.35	40.04	47.40	35.68	40.14	48.59	36.19
	CSC3D-CSI	**39.28**	**46.38**	**36.38**	**40.37**	**48.43**	**37.50**	**41.79**	**48.73**	**38.55**	**43.62**	**49.67**	**39.47**
	HS-AMP	37.53	46.13	35.47	37.67	47.09	37.26	37.87	47.28	37.95	38.03	47.87	38.69
	Unmixing	29.81	32.80	23.14	29.98	33.21	24.55	30.07	33.29	25.16	30.16	33.35	25.32
	Low-Rank	35.02	44.76	34.23	36.18	46.07	34.55	38.39	46.36	35.05	39.86	46.45	35.32
30dB	SSR	38.39	45.97	35.31	38.83	47.13	35.36	40.06	47.40	35.64	40.13	48.59	36.21
	CSC3D-CSI	**39.43**	**46.47**	**36.45**	**40.58**	**48.52**	**37.89**	**42.04**	**48.74**	**38.90**	**44.77**	**49.92**	**39.86**
	HS-AMP	37.60	46.28	35.56	37.74	47.44	37.33	37.92	47.30	38.06	38.09	48.04	38.81
	Unmixing	29.83	32.96	24.19	30.07	33.22	24.83	30.18	33.35	25.23	30.16	33.39	25.54
	Low-Rank	35.37	45.72	34.55	36.55	46.84	35.00	36.82	47.26	35.42	37.74	47.47	35.71

Table 4

Mean PSNR of the reconstructed datasets using the 3D-CASSI, for the various methods, using five levels of noise and four compression ratios (number of shots). The combinations with the best PSNR results are highlighted. The standard deviations for five repetitions were well bellow 1% of the mean PSNR, thus are not shown.

wed that the proposed algorithm can not reverse the C-CASSI band shifting, leading to recovery qualities around 24-25dB.

An exhaustive analysis of the minimization routines yielded one possible cause: the interaction between the C-CASSI sensing matrix and the linear operators $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$. Both linear operators are matrix representations of the cyclic convolution operation, therefore they have very defined structures which include the circular shifting effect.

Consider the band-shifting structure of a single C-CASSI sensing matrix $\mathbf{H}$ depicted in Figure 18, and the circular shifting effect of matrix $\bar{\mathbf{D}}$ in Figure 9. When interacting with the optical dispersion element represented in C-CASSI's sensing matrix, the circular shifting effect of the equivalent cyclic convolutional matrices is replicated off-site, as shown in Figure 19, adding noise to the recovery routine.

In order to improve CSC3D-CSI performance we implement a Side Information Acquisition scheme based in (Galvis et al., 2017; Yuan et al., 2015), as seen in Fig. 20. This modification aims to reduce the band-shifting effect over DUP-CSI by including a low-cost approximation of the morphological features within the dictionary elements initialization. Contrary to the state-of-the-art side information acquisition schemes, which capture an RGB image, the proposed side information acquisition scheme captures a single 2D grayscale version of the original SI. This 2D image contains all the important spatial features within the original (Galvis et al., 2017). Then, it

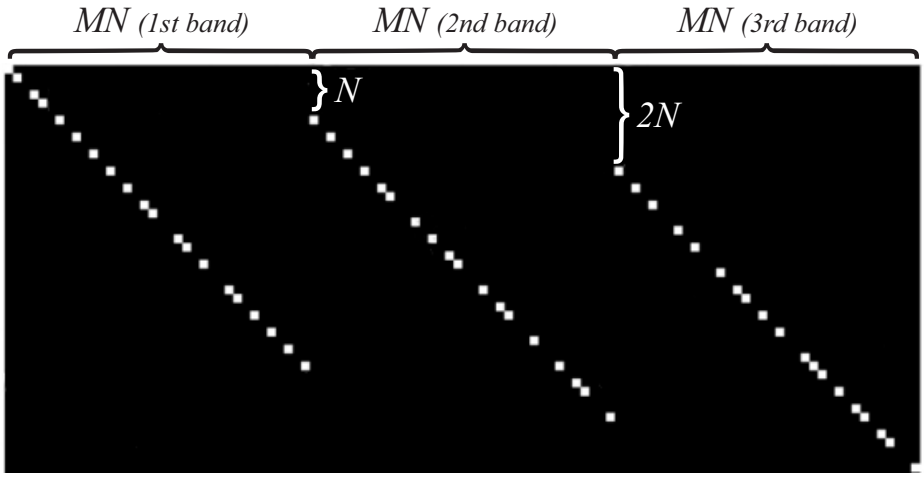


Figure 18. Detail of the band shifting for a single C-CASSI sensing matrix $\mathbf{H}^t$. Taken from (Barajas-Solano et al., 2019a).

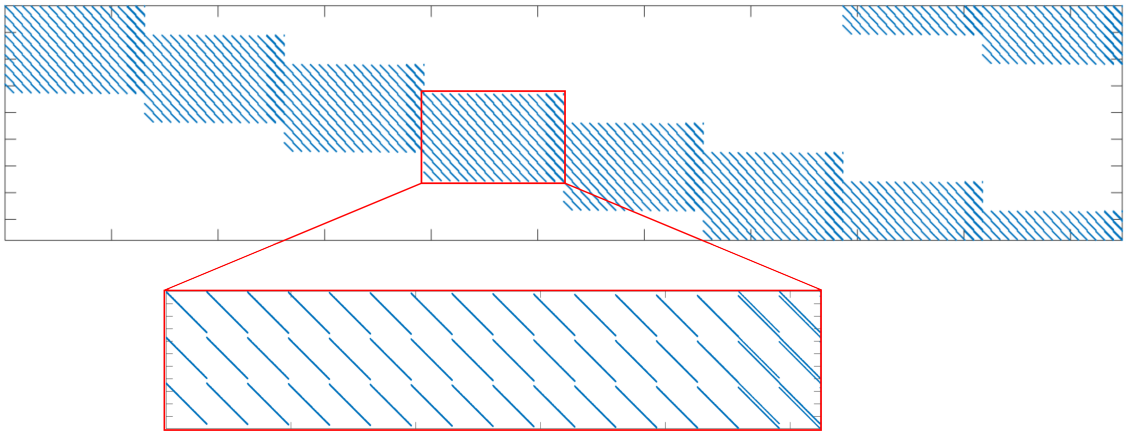


Figure 19. Product $\mathbf{H}\bar{\mathbf{D}}$ for the C-CASSI CSI architecture, and closeup. Note the off-site replicas at the right side of the closeup.

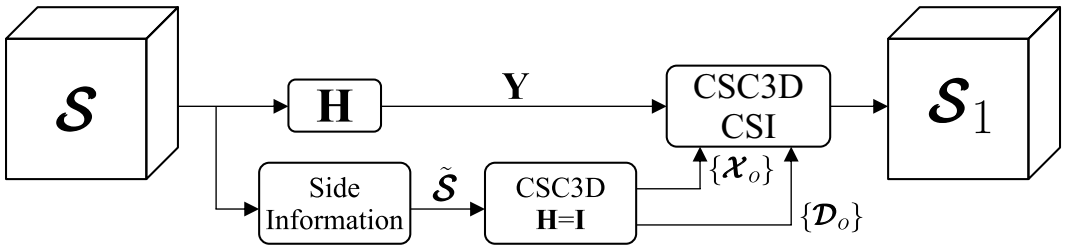


Figure 20. Side information acquisition scheme for recovering SIs from compressive spectral measurements obtained using the C-CASSI architecture.

generates a low-cost approximation of the original SI, $\tilde{\mathscr{S}} \in \mathbb{R}^{M\times N\times L}$, where each band $\tilde{\mathscr{S}}_l$ is the same grayscale version of the original SI $\mathscr{S}$. The approximation $\tilde{\mathscr{S}}$ has the same spatial morphological features as $\mathscr{S}$ (Galvis et al., 2017), and is used to train an initial collection of dictionaries $\{\mathscr{D}_o\}$ and coefficients $\{\mathscr{X}_o\}$. Finally, the CSC3D-CSI algorithm adjusts the spectral features in the resulting $\mathscr{S}_1$, without falling into solution wells provoked by the band shifting.

The proposed CSC3D-CSI is used to recover the high-frequency version of the Pavia University test image from compressive measurements. However this time we use the C-CASSI architecture along the proposed Side Information scheme. The compressive measurements include 4 shots, resulting in a compression ratio of 0.28 following the formula $\gamma = KM(N+L-1)/MNL$ (Arce et al., 2014). Note that the Side Information scheme contributes with more information, decreasing the actual compression. The recovery qualities can be seen in Fig. 21.

Second, the proposed algorithm, along the proposed side-information acquisition scheme, is tested for various levels of additive sensing noise and various levels of compression. In order to make a fair comparison, all the test methods included initializations based in $\tilde{\mathscr{S}}$. For example, the SSR based approach uses initial coefficients $\mathbf{x}_o = \mathbf{\Psi}^T\tilde{\mathbf{s}}$, where $\tilde{\mathbf{s}}$ is the vectorized version of $\tilde{\mathscr{S}}$ and $\mathbf{\Psi}^T : \mathbb{R}^{MNL} \to \mathbb{C}^{MNL}$ is the Kronecker basis direct operator.

Fig. 22 shows the effect of both the compression ratio and SNR levels over the recovery quality. For different levels of compression the SNR of the compressive measurements is fixed to

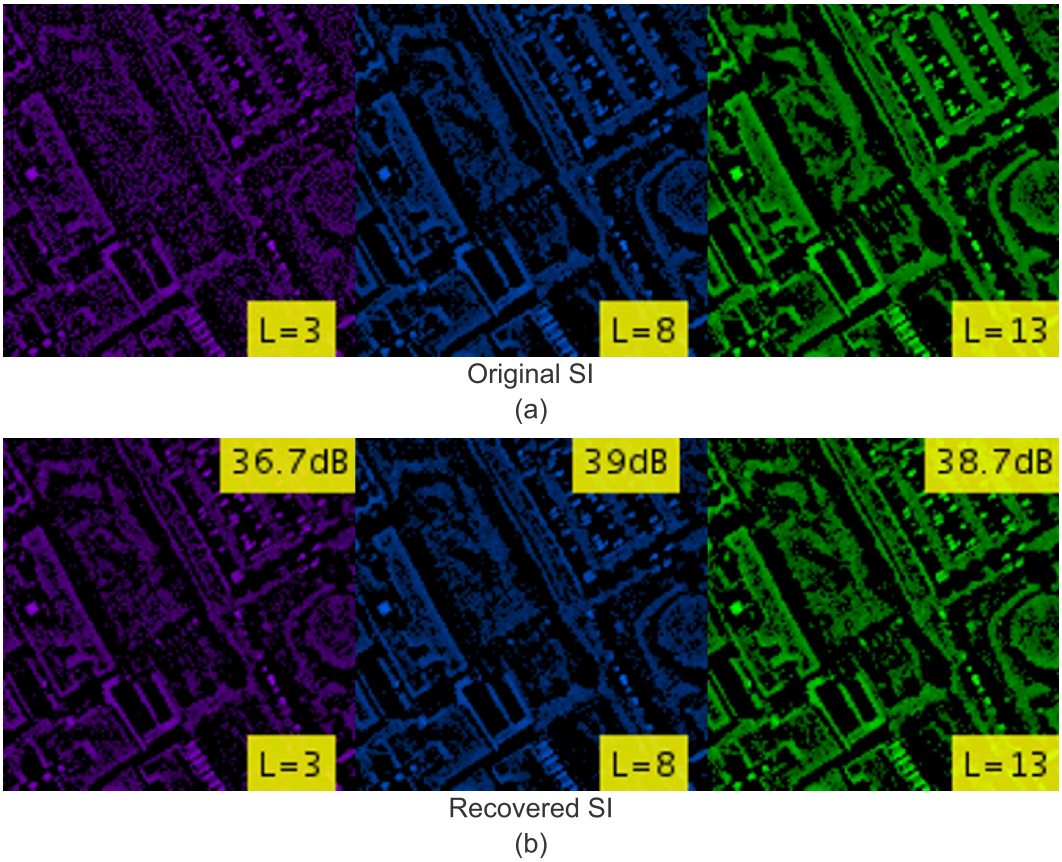


Figure 21. Simulation results of the recovery quality for some chosen bands from compressive spectral measurements using the proposed CSC3D-CSI algorithm, and the CSI C-CASSI architecture with a compression ratio of 0.28 along the Side Information scheme.

20dB. For different levels of noise the compression ratio is fixed to 4 shots. The proposed algorithm outperforms the state-of-the-art methods by up to 5dB for all the noise levels and compression ratios.

Table 5 shows the performance comparison between the proposed CSC3D-CSI, the SSR based approach, the HS-AMP, Unmixing, and the Low-Rank methods, for five levels of noise and four different compression ratios. Although methods like SSR and HS-AMP have acceptable performances, the proposed CSC3D-CSI algorithm outperforms the four state-of-the-art methods,

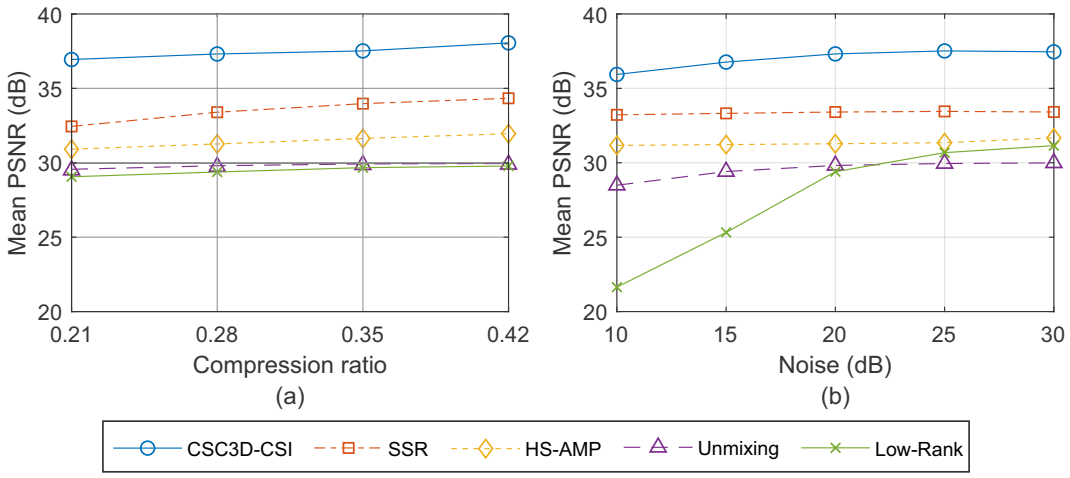


Figure 22. Mean PSNR of the simulation results at attempting to recover the Pavia SI using the proposed CSC3D-CSI and the CSI C-CASSI architecture, for the various methods, (a) for fixed 20dB SNR and (b) fixed 4 shots. The standard deviations for five repetitions were well bellow 1 % of the mean PSNR, thus are not shown.

in all the scenarios, by up to 3.5dB.

About the computing time for the proposed CSC3D-CSI and the state-of-the-art reference methods. All five methods were implemented and run on Matlab in a Windows 10, Intel i7-479 @ 3.6GHz and 16GB RAM. Table 6 shows the mean time per iteration for each method. All the methods exhibited few variations in the running time for all noise levels, compression ratios, and CSI architectures. However, the Low-Rank based approach performed slower with the C-CASSI architecture.

Robustness to acquisition noise. In this section, we evaluate the performance of the proposed CSC3D-CSI method when the measurements are contaminated with different levels of Poisson noise. The objective is to asses CSC3D-CSI's performance under conditions of photon and calibration errors . Different integration times are emulated modifying the Poisson noise, creating SNR levels between 18dB and 28dB. Considering the results obtained in sections 5.5 and 5.5, and

		3 Shots (0.21)			4 Shots (0.28)			5 Shots (0.35)			6 Shots (0.42)		
		Pavia	Salinas	Beads	Pavia	Salinas	Beads	Pavia	Salinas	Beads	Pavia	Salinas	Beads
10dB	SSR	32.15	38.98	28.10	33.22	40.02	29.51	33.91	40.56	30.24	34.27	40.89	30.65
	CSC3D-CSI	**35.79**	**40.55**	**29.26**	**35.96**	**40.98**	**29.94**	**35.67**	**41.93**	**30.32**	**36.15**	**42.80**	**31.31**
	HS-AMP	30.75	37.47	26.84	31.17	37.94	27.25	31.56	38.21	27.77	31.95	38.46	28.09
	Unmixing	28.42	32.55	22.37	28.49	32.66	23.95	28.65	32.89	24.16	28.83	32.99	24.57
	Low-Rank	21.05	28.02	22.32	21.64	28.82	22.70	22.35	30.02	23.10	23.64	31.00	23.81
15dB	SSR	32.31	38.99	28.48	33.32	40.67	29.73	33.92	41.62	30.40	34.36	41.92	30.73
	CSC3D-CSI	**36.66**	**41.29**	**30.21**	**36.77**	**41.96**	**31.06**	**37.21**	**42.67**	**31.57**	**37.50**	**43.45**	**32.73**
	HS-AMP	30.79	37.61	26.87	31.22	37.94	27.33	31.62	38.25	27.77	31.96	38.54	28.17
	Unmixing	29.19	32.97	24.34	29.41	33.05	24.50	29.49	33.18	24.92	29.65	33.24	24.88
	Low-Rank	25.10	32.81	24.06	25.32	33.55	25.23	26.22	34.29	25.32	26.72	35.47	25.55
20dB	SSR	32.46	39.24	28.66	33.41	41.37	29.78	33.98	41.99	30.43	34.35	42.40	30.79
	CSC3D-CSI	**36.95**	**41.60**	**30.64**	**37.32**	**42.38**	**31.56**	**37.52**	**43.18**	**32.15**	**38.06**	**44.04**	**33.02**
	HS-AMP	30.93	37.76	27.08	31.28	38.05	27.39	31.65	38.28	27.79	31.97	38.55	28.19
	Unmixing	29.58	32.99	24.78	29.82	33.04	25.14	29.94	33.29	25.21	29.97	33.33	25.37
	Low-Rank	29.08	37.60	26.76	29.40	38.28	26.80	29.69	38.97	27.28	29.80	39.93	27.54
25dB	SSR	32.42	39.26	28.71	33.45	41.34	29.81	34.08	42.14	30.55	34.37	42.45	35.16
	CSC3D-CSI	**37.13**	**41.80**	**30.77**	**37.52**	**42.57**	**31.48**	**37.80**	**43.37**	**32.52**	**38.27**	**44.28**	**33.46**
	HS-AMP	31.16	37.93	27.34	31.35	38.23	27.51	31.73	38.29	27.89	32.04	38.58	28.21
	Unmixing	29.59	33.18	25.12	29.95	33.24	25.31	29.98	33.31	25.46	30.19	33.34	25.51
	Low-Rank	30.40	40.50	27.63	30.68	41.20	28.03	30.76	41.88	28.47	30.90	42.04	29.09
30dB	SSR	32.42	39.28	28.67	33.41	41.33	29.79	34.07	42.08	33.66	34.39	42.48	35.37
	CSC3D-CSI	**37.20**	**41.77**	**30.89**	**37.46**	**42.50**	**31.95**	**37.67**	**43.39**	**32.35**	**38.39**	**44.37**	**33.49**
	HS-AMP	31.31	38.33	27.48	31.67	38.39	27.64	31.79	38.61	27.91	32.07	38.79	28.22
	Unmixing	29.79	33.24	25.13	29.99	33.31	25.42	30.14	33.36	25.54	30.19	33.37	25.56
	Low-Rank	31.49	41.51	27.98	31.15	41.85	28.52	31.78	42.00	28.99	32.40	42.15	29.36

Table 5

Mean PSNR of the reconstructed datasets using the C-CASSI, for the various methods, using five levels of noise and four compression ratios (number of shots). The combinations with the best PSNR results are highlighted. The standard deviations for five repetitions were well bellow 1% of the mean PSNR, thus are not shown.

	CSC3D-CSI	SSR	HS-AMP	Unmixing	Low-Rank
Average time per iteration (s)	5.5	0.2	0.9	0.2	>60

Table 6

Average time per iteration, in seconds, for the proposed CSC3D-CSI and each of the state-of-the-art reference methods. Note that this time per iteration does not take into account the computations prior to the iterative cycle.

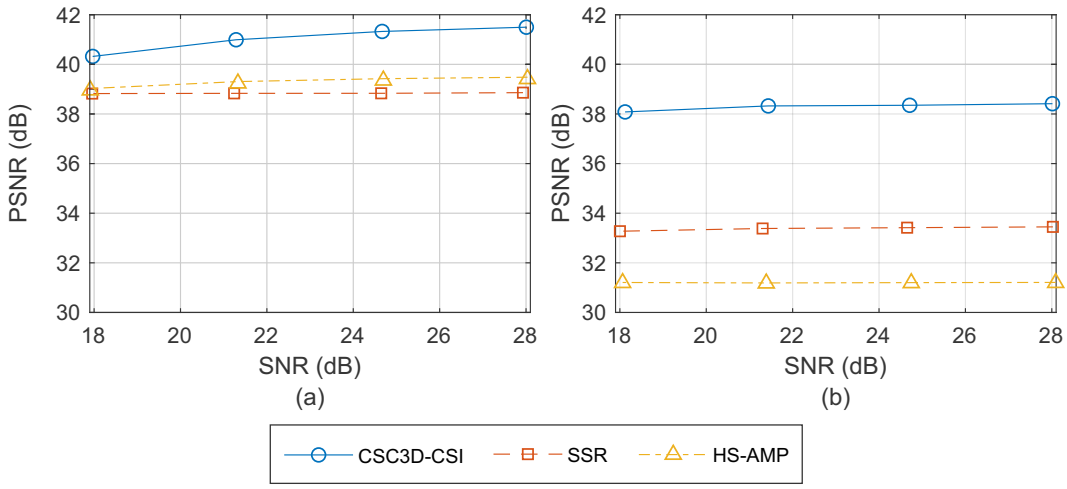


Figure 23. Mean PSNR of simulated results at recovering the Pavia University SI using the (a) 3D-CASSI and (b) C-CASSI, for the proposed CSC3D-CSI, two test methods and four different levels of Poisson acquisition noise, and a fixed compression ratio (4 shots). The standard deviations for five repetitions were well bellow 1 % of the mean PSNR, thus are not shown.

the time-per-iterations presented in Table 6, the selected state-of-the-art test methods are SSR and HS-AMP, with a fixed compression ratio of 4 shots for both CSI architectures. Fig. 23 shows the mean PSNR for the recovered Pavia dataset using four different levels of Poisson noise, both CSI architectures and a fixed compression ratio (4 shots). The proposed CSC3D-CSI algorithm proves to be robust to the change in the noise model, improving the state-of-the-art methods in up to 2dB when using the 3D-CASSI CSI architecture, and up to 5dB for the C-CASSI CSI architecture. All the above considering that the square ℓ_2-norm formulation is more related to a Gaussian noise model.

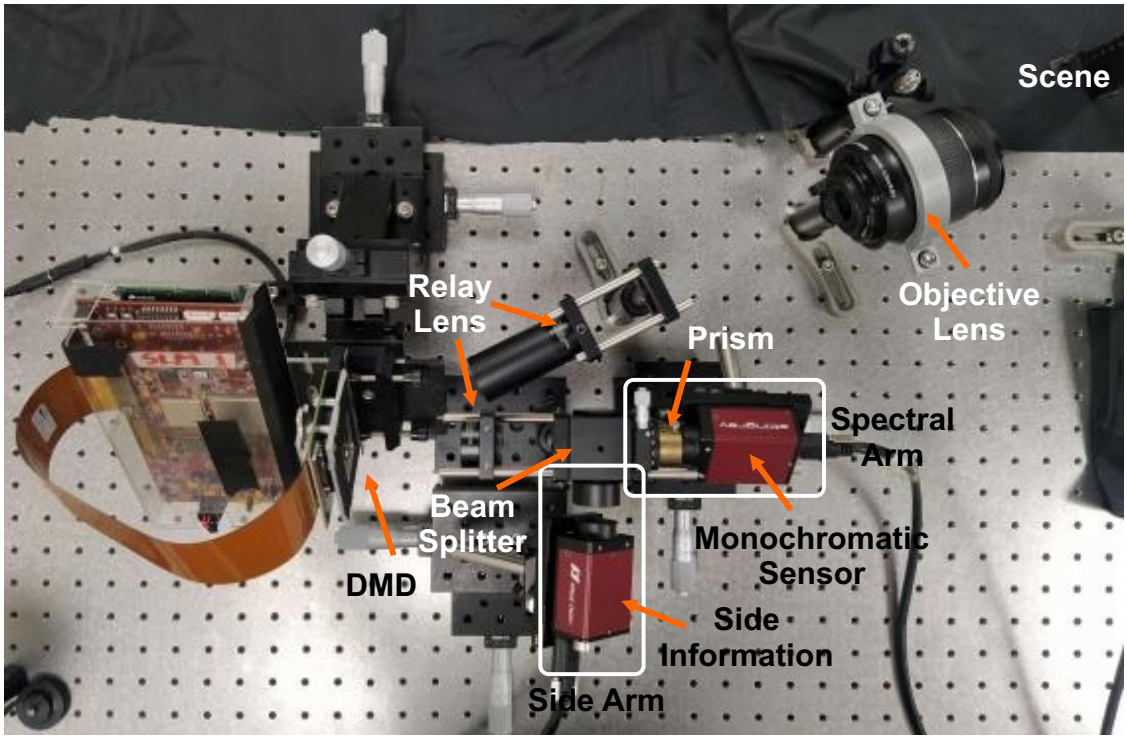


Figure 24. Experimental setup for capturing compressed measurements using both the 3D-CASSI and C-CASSI. The configuration of the DMD allows to obtain compressed measurements with one or the other CSI architecture with a single experimental setup.

5.6. CSC3D-CSI Experimental Performance Evaluation

To experimentally prove the advantages of the proposed CSC3D-CSI framework over the state-of-the-art SSR framework, we proposed the experimental acquisition setup shown in Figure 24. This setup is composed of a $18-55$mm $1:3.5-5.6$ objective lens (Canon), a DLI4130 DMD (DLInovations) with spatial resolution of 1024×768 and mirror pitch size of $13.68\ \mu m$, and two monochrome charged-coupled device detector (AVT Stingray F-145B) with spatial resolution 1388×1038 and pitch size of $6.45\ \mu m$.

The coded apertures were generated using the same routines as in section 5.5, for both 3D-CASSI and C-CASSI, and uploaded into the DMD. The compressive measurements are obtained using as a target the real scene instead of the white plate and replacing the monochromatic light with a broadband white light. The experimental setup is synchronized such that the DMD sets the pattern and the sensor captures the projection. After that, the DMD updates the coded aperture. Each scene, with each CSI architecture was sensed using K=2 and K=4 shots.

Two target scenes were used to evaluate the performance of the proposed CSC3D-CSI framework against the SSR framework. The scenes are called *Flowers* and *Lego Hulk* and both have 128×128 pixels of spatial resolution and 12 bands sensed at (457-464), (465-477), (478-487), (488-499), (500-514), (515-529), (530-549), (550-569), (570-594), (595-619), (620-649), (650-685) nm. Considering the two CSI architectures, two different shots realizations, and two different scenes, then we obtain a total of 8 experiments, plus the capture of the side information measurements.

Figures 25 and 26 show a recovered example, plus close ups, of both datasets using the 3D-CASSI architecture and K=4 shots. The first we note is the difference in colors in both recovered SIs, with the CSC3D-CSI exhibiting a warmer color. In Figure 25 we can note a higher detailed at recovering the threaded elements and less artifacts at recovering plain elements such as the leaf. In Figure 26 we note a higher detailed at recovering the abdominal muscles detail, sharper borders, and less artifacts when recovering uniform sections of the toy. In general, the CSC3D-CSI frame-

work performs qualitatively better than the SSR framework.

Figures 27 and 28 shows with a higher detail the performance of both frameworks. Here we present the spectrum of three points, per dataset, obtained using a light spectrometer (in black). For each spectrum we present the recovered spectrums obtained using the CSC3D-CSI and SSR frameworks, for both CSI architectures and shots numbers. Both figures evidence a higher performance of the proposed CSC3D-CSI (in blue) over the SSR framework (in red) for both datasets, CSI architectures and shots number.

For Figure 27, point P1 was selected because of its intense blue color; point P2 have a strong presence of the green color; while point P3 is closer to the color white. The proposed CSC3D-CSI framework performs satisfactorily in all three points, while the SSR framework seems to struggle with the green color. However, its performance improves with a higher shots number, reaching the level of performance of the CSC3D-CSI framework.

For Figure 28 the three points were chosen in order to sense colors green, blue and orange, respectively. Again the CSC3D-CSI performs satisfactorily in all three points, improving the performance of the SSR framework. Both CSC3D-CSI and SSR improves the perfomance in the point P1 with a higer number of shots. However, the SSR framework struggles at recovering the blue and orange colors. This discrepancies in the spectrums for the SSR framework could explain the dull colors in the recovered SIs.

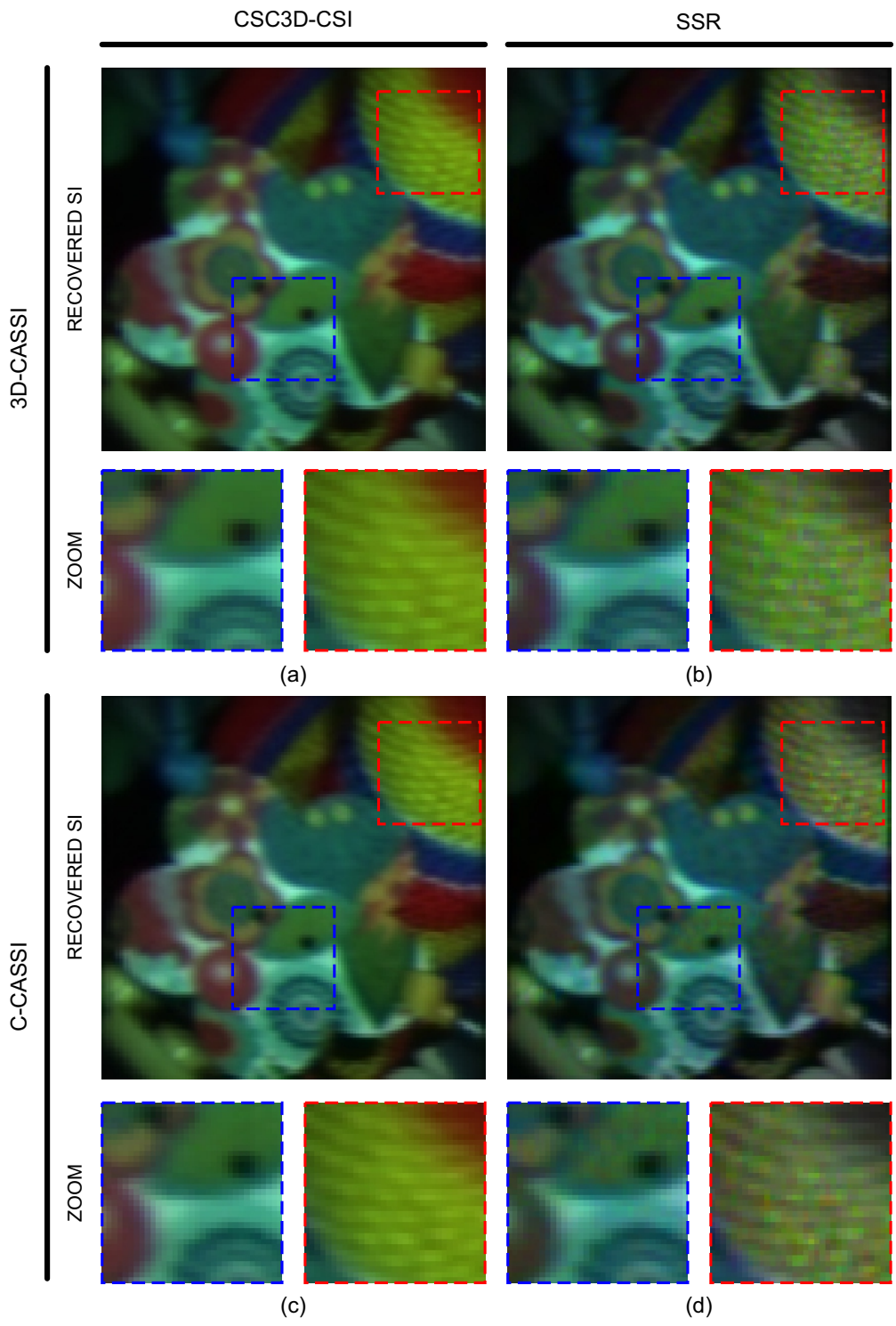


Figure 25. Experimental results of the recovered dataset Flowers at K=4 shots, and closeups, using the proposed CSC3D-CSI and SSR frameworks, and two CSI arquitectures: 3D-CASSI and C-CASSI. Note that the proposed CSC3D-CSI outperforms the SSR model at border sharpness, smooth surfaces, color warmth, and reduction of artifacts. I.e., the threaded section (red) have sharper borders while the leaf section (blue) is smoother for the proposed CSC3D-CSI at both CSI architectures.

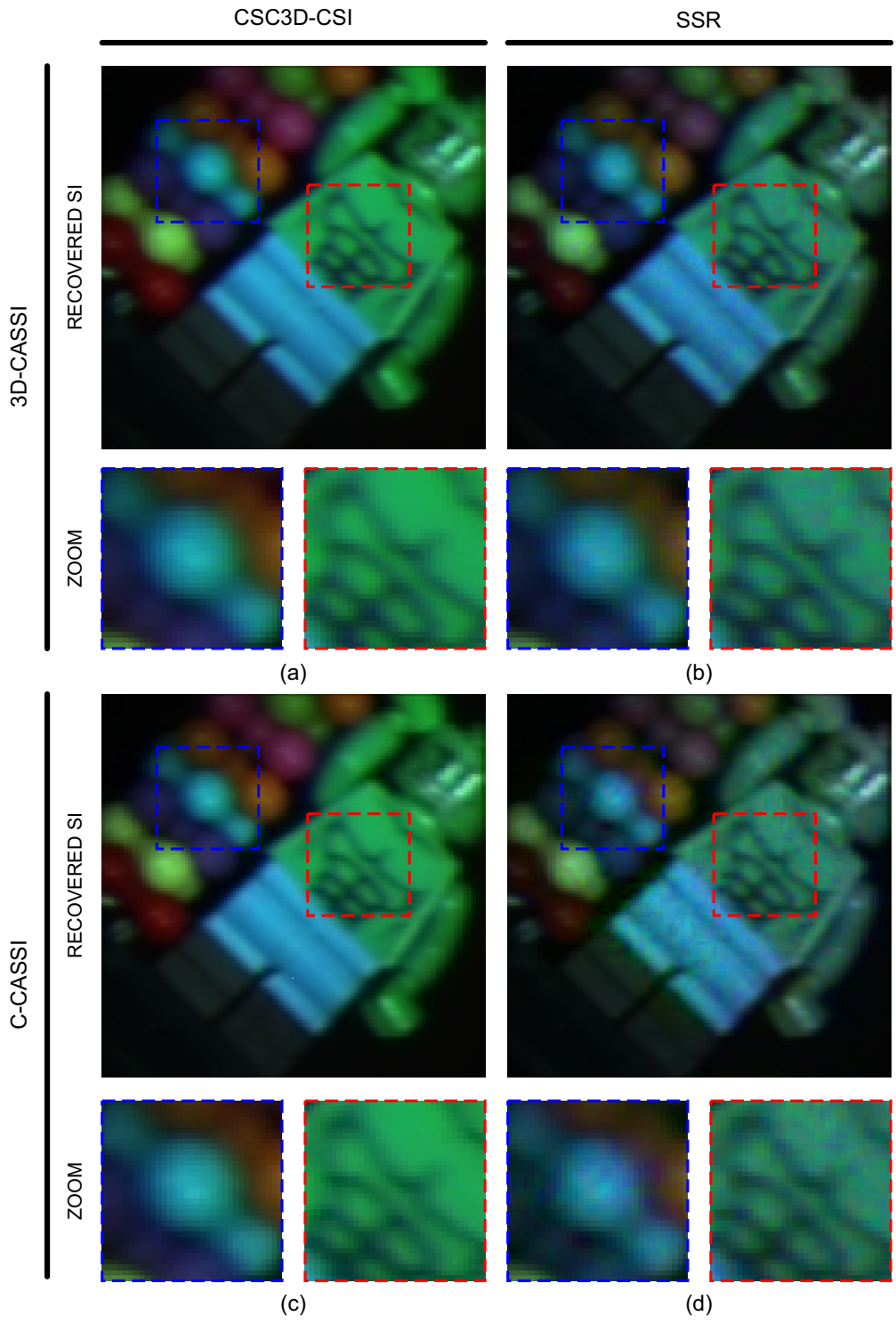


Figure 26. Experimental results of the recovered dataset Lego Hulk at K=4 shots, and closeups, using the CSC3D-CSI and SSR frameworks, and two CSI arquitectures: 3D-CASSI and C-CASSI. Note that the proposed CSC3D-CSI outperforms the SSR model at border sharpness, smooth surfaces, color warmth, and reduction of artifacts. I.e., the abdominal section (red) is smoother and the spheres section (blue) have fewer artifacts.

Dataset	Architecture	Shots	P1		P2		P3	
			CSC3D-CSI	SSR	CSC3D-CSI	SSR	CSC3D-CSI	SSR
Flowers	3D-CASSI	2	**0.021**	0.085	**0.016**	0.188	0.060	**0.061**
		4	**0.025**	0.066	**0.016**	0.179	**0.040**	0.051
	C-CASSI	2	**0.014**	0.064	**0.015**	0.252	**0.058**	0.073
		4	**0.016**	0.630	**0.020**	0.149	**0.065**	0.077
Lego Hulk	3D-CASSI	2	**0.064**	0.190	**0.083**	0.323	**0.043**	0.272
		4	**0.051**	0.117	**0.055**	0.227	**0.031**	0.164
	C-CASSI	2	**0.109**	0.350	**0.073**	0.209	**0.029**	0.360
		4	**0.099**	0.264	**0.061**	0.182	**0.028**	0.184

Table 7
SAM metric values evaluated at three different spatial points, compared to the acquired spectrum using a spectrometer, for both recovered SIs using the proposed CSC3D-CSI framework and the SSR model, and two different datasets. The evaluated SIs were recovered using K=4 shots.

Finally, Table 7 presents the estimated spectral angle mapper (SAM) values when comparing each experimental spectrum with all the recovered spectrums at all the possible dataset-architecture-shots combinations. The CSC3D-CSCI framework outperforms the state-of-the-art SSR in up to one order of magnitude.

5.7. Algorithm Convergence

In order to illustrate the good convergence of Algorithm 18, a typical evolution of cost function is shown in Eq. (39) as a function of the iteration number. Considering that the indicator function in (43) have only values 0 or ∞, then it is not included in the estimation of the cost function. The high-frequencies version of the Pavia dataset was used for this experiment, and the compressive measurements were simulated with the 3D-CASSI system, with 20dB SNR, and 4 shots. Fig. 29 confirms the convergence of the algorithm to a critical point of the objective function. Note that one iteration of Algorithm 18 includes one iteration of Algorithm 16 and 17.

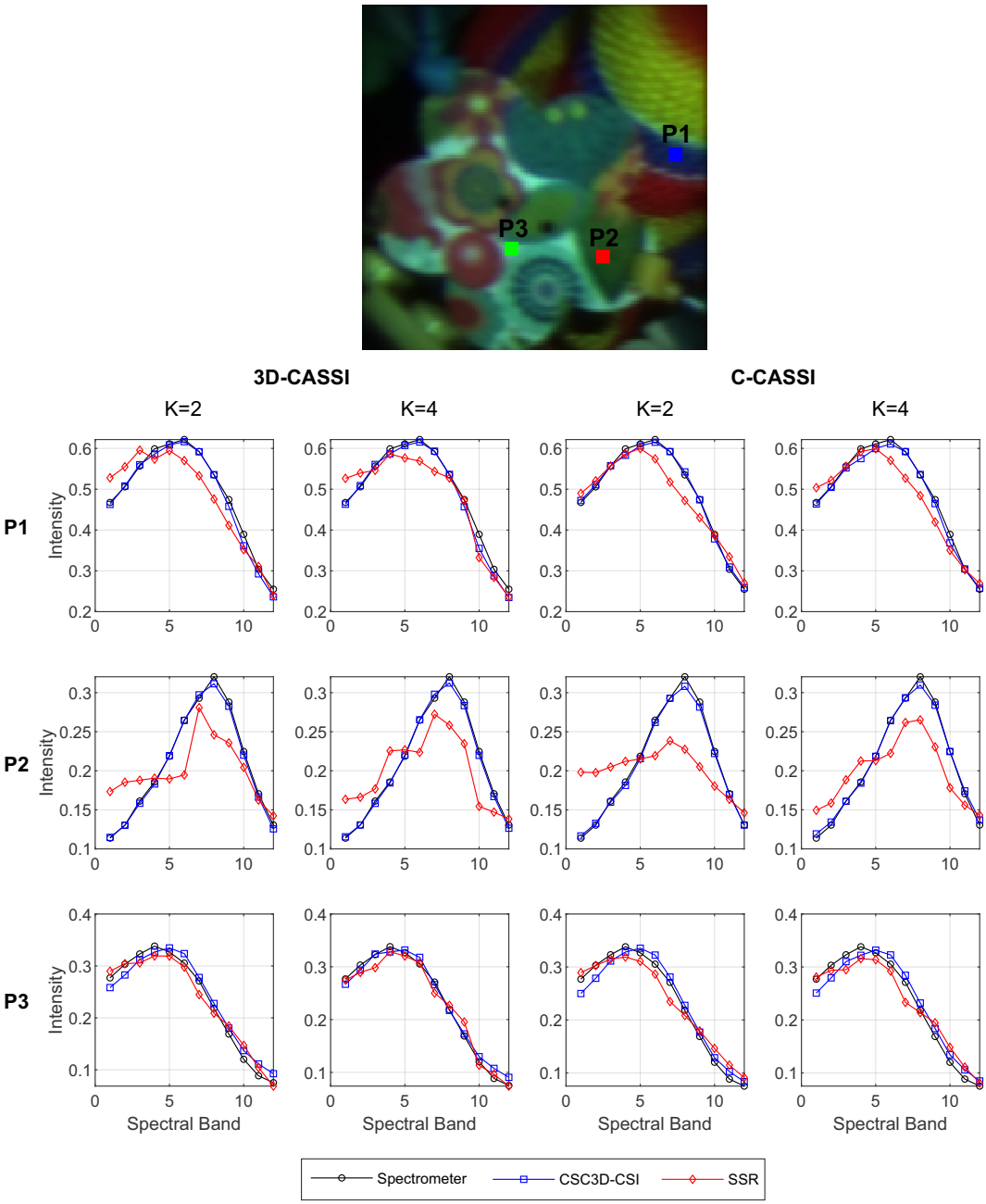


Figure 27. Comparisson of the recovered spectrums of the Flowers dataset, in three different spatial points, using K=2 and K=4 shots, and the CSI 3D-CASSI architecture for both the proposed CSC3D-CSI and SSR models. Note that the proposed CSC3D-CSI outperforms the SSR model at all compression levels with both CSI architectures.

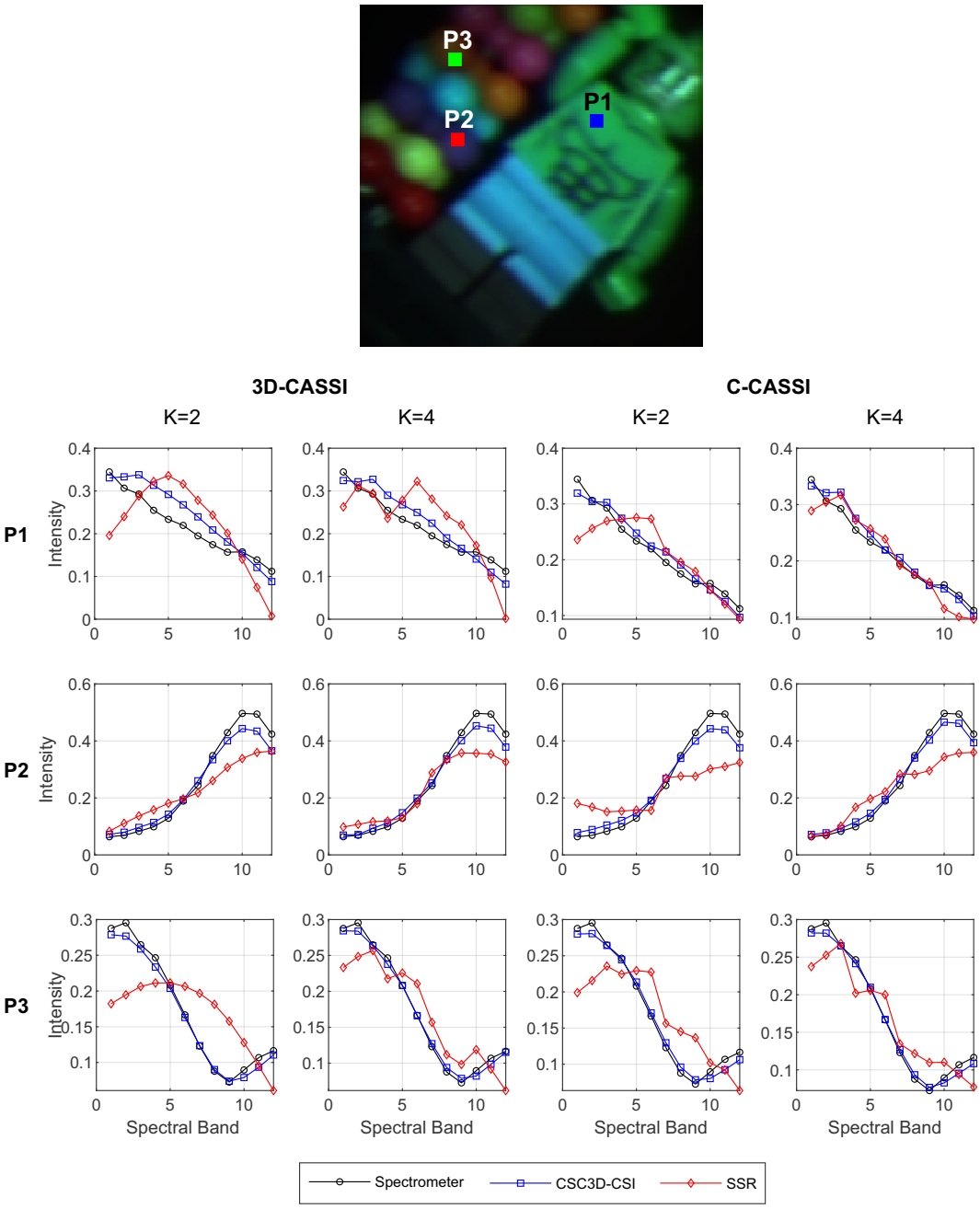


Figure 28. Comparisson of the recovered spectrums of the Lego Hulk dataset, in three different spatial points, using K=2 and K=4, shots and the 3D-CASSI architecture for both the proposed CSC3D-CSI and SSR models. Note that the proposed CSC3D-CSI outperforms the SSR model at all compression levels with both CSI architectures.

In order to analyze the sensitiviy to initialization, we ran the proposed algorithm with 200 different random initializations for the collection of dictionary elements, and all-zeros coefficient maps. The histogram of the corresponding values of the objective function is shown in Fig. (30), reporting a single mode, confirming the convexity of the proposed algorithm. The mode of the objective function histogram corresponds to a PSNR of 40.3dB.

Conclusions. A 3D CSC framework is proposed for sparsely representing the spatial-spectral correla- tions of SIs, exploiting the advantages of overcomplete convolutional dictionaries. The proposed CSC3D is able to sparsely represent SIs, even in presence of noise, outperforming the state-of-the-art approach. The proposed framework is formulated in order to fit a lineal minimization framework so that it can be integrated within a CSI recovery scheme.

When integrated with a 3D-CASSI CSI architecture, the proposed CSC3D-CSI is able to match and outperform the state-of-the-art approach at different compression ratios and noise levels, for a set of different test images. On the other hand, when the proposed CSC3D-CSI is integrated with the C-CASSI CSI architecture, it is not able to unmix the C-CASSI band-shifting and needs a better initial set of convolutional dictionary elements for preserving the spatial features.

With the suggested C-CASSI Side Information approach, the proposed CSC3D-CSI is able to outperform the state-of-the-art approach at various compression ratios and noise levels. The proposed CSC3D-CSI algorithm proved to be robust when dealing with Poisson acquisition noises,

improving various state-of-the-art for techni- ques.

When using experimental measurements, the proposed CSC3D-CSI outperforms the SSR framework in border sharpness, color warmth and absence of artifacts. This is also verified when comparing the recovered spectrums with the ones measured experimentally using a light spectrometer.

6. Convolutional Sparse Coding 4D and Compressive Spectral Video Sensing

The purpose of this chapter is to introduce the last two contributions of this doctoral dissertation: a 4D convolutional sparse representation for Spectral Videos (SVs), and its application in CSVS. This chapter includes the mathematical formulations and the derivation of their numerical solution. It also includes the synthetic experiments in order to asses the performance of the proposed framework.

It is worth noting that the mathematical formulation for the new CSC framework is heavily based on the CSC3D's formulation, profiting on the modularity of the convolutional operation; but differs in the dimensionality of the operators, rearrangements and transformations. For this reason this chapter uses the same Algorithms presented in Chapter 4, but with a change in the introduced dimensions. The fact that both CSC frameworks, 3D and 4D, uses the same formulation for representing different signal dimensions is a contribution in itself.

6.1. Spectral Video (SV) and Super Resolution (SR)

Spectral Video (SV) has emerged as an image modality used with great interest in object or human tracking (Cheng et al., 2007)(Van-Nguyen et al., 2010)(Banerjee et al., 2009), cancer detection (Leitner et al., 2013), bile duct inspection (Zuzak et al., 2013), and several types of surgery (Yi et al., 2011).

In essence, a SV $\mathscr{S} = \{\mathscr{S}^t \in \mathbb{R}^{M\times N\times L} | t = 1, ..., T\}$ is a collection of SIs $\mathscr{S}^t$ captured consecutively within a time frame at discrete intervals, analog to an RGB video (**?**). This means that each frame, a full SI, must be captured within a small time period in order to capture the time variations of interest within the observed scene. However, techniques such as push-broom (Sellar and Boreman, 2005) or optical band-pass filters (Gat, 2000) require a considerable sensing time in order to capture a single spectral frame. This limitations in the available technologies and time restrictions leads to one of two scenarios: either we resort to faster and more expensive sensing equipment in order to capture the desired SVs within the time limits; or we use available, and not too expensive, technology to capture SVs with low spatial, spectral and/or temporal resolution, within the time limits.

In this regard, Super-Resolution (SR) techniques (Buttingsrud and Alsberg, 2006; Kwan et al., 2017) have emerged has a processing tool for recovering high-resolution information from low-resolution measu- rements. In general, the SR problem describes the low-resolution measurements as a compressed version of the high-resolution data, which is recovered solving an inverse problem. In order to recover the high-resolution data, the state-of-the-art methods use the sparse signal representation model (SSR) (Correa et al., 2016; Leon-Lopez et al., 2019) as a signal framework for recovering the information of interest. In this regard, the SSR model uses predefined dictionaries, specifically orthonormal basis, for sparsely representing any signal.

The SSR-SR formulation for SVs can be expressed as

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} - \dot{\mathbf{s}}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1 , \tag{97}$$

where $\mathbf{H} \in \mathbb{R}^{M_1 N_1 L_1 T \times MNLT}$ is a decimation matrix, with $\alpha A_1 = A$ and $\alpha \in \mathbb{N}^*$; $\dot{\mathbf{s}} = \mathbf{H}\text{vect}(\mathscr{S}) \in \mathbb{R}^{MNLT}$ is the vectorization of the sensed low-resolution SV; $\boldsymbol{\Psi}$ is the SSR representation basis and $\boldsymbol{\theta}$ are the sparse coefficients. Eq. (97) aims to recover a full resolution SV from a low resolution version, using a one-size-fits-all basis. This has the same intrinsic limitation, as in SIs and CSI, of the upper-bounded reconstruction quality due to the generality of the representation basis.

On the other hand, synthesis frameworks, like the convolutional sparse coding (CSC), have emerged as an alternative approach for sparsely representing multidimen- sional signals (Wohlberg, 2016b), and recently this signal model has been expanded in order to represent the spatial-spectral correlations of spectral images (Barajas-Solano et al., 2019c). CSC's specificity for representing signals, and the attached invariance to shifting and deformation property (Papyan et al., 2017), represents an attractive framework for sparsely representing SVs.

6.2. Convolutional Sparse Coding for Spectral Videos (CSC4D)

This work proposes to expand the CSC formulation to a full 4D CSC framework, in order to represent the spatial-spectral-temporal correlations in SVs. First, lets consider a SV as a tetrahedron $\mathscr{S} \in \mathbb{R}^{M \times N \times L \times T}$. Next, let $\mathscr{D} = \{\mathscr{D}_m|_{m=1}^{M_d} \mid \mathscr{D}_m \in \mathbb{R}^{d_M \times d_N \times d_L \times d_T}\}$ be a collection of 4D convolutio-

nal dictionary elements and $\mathscr{X} = \{\mathscr{X}_m|_{m=1}^{M_d} \mid \mathscr{X}_m \in \mathbb{R}^{M\times N\times L\times T}\}$ a collection of sparse coefficient maps, where $d_M, d_N, d_L, d_T \ll M, N, L, T$. This means that while the SSR framework must integrate a new appropriate basis for the new temporal dimension, the convolutional operator only requires to scale up in order to accommodate the same information as expressed in Eq. (35).

With this scaling into consideration then a SV $\mathscr{S}$ can be represented as

$$\mathscr{S} = \sum_{m=1}^{M_d} \mathscr{D}_m \overset{4}{*} \mathscr{X}_m \simeq \bar{\mathbf{D}}\mathbf{x} = \bar{\mathbf{X}}\mathbf{d}, \tag{98}$$

where $\overset{4}{*}$ denotes the 4D cyclic convolution. Just as with CSC3D the equivalent lineal operator $\bar{\mathbf{D}} \in \mathbb{R}^{MNLT\times MNLTM_d}$ is created following Algorithm 5 with $Ł_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. Also, the vectorization $\mathbf{x} \in \mathbb{R}^{MNLTM_d}$ is created following Algorithm 4, also with $Ł_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. The matrix $\bar{\mathbf{X}}$ and vector $\mathbf{d}$ are created in the same way profiting the convolution's commutativity property.

Although the expression for the linear equivalent operator used in CSC4D is similar to the linear equivalent operator used in CSC3D, its full construction is more complicated than simply a scaling up. Figure 31 presents the example matrix $\bar{\mathbf{D}}_m$ created from $\mathscr{D}_m \in \mathbb{R}^{2\times2\times2\times2}$ and $\mathscr{X}_m \in \mathbb{R}^{4\times4\times3\times2}$. Let us begin by the 3D cyclic convolutional matrices $\bar{\mathbf{D}}_{t,m} \in \mathbb{R}^{48\times48}$ with $t = 1,2$ shown in Figure 8. The same equivalent convolutional submatrices are found hihglighted in gold in Figure 31. For the temporal dimension $T = 2$ the 3D equivalent convolutional matrices

are self-replicated in a 2×2 grid, in a cyclic pattern, increasing the size of the final matrix by T^2. Finally, we create the equivalent convolutional matrix $\bar{\mathbf{D}} \in \mathbb{R}^{96\times 96}$ ($MNLT \times MNLT$). Note again the sub-matrix-wise circular shifting effect in the top right corner in each quadrant.

Operator $\bar{\mathbf{D}}_m$ can be considered sparse given that only $(d_M d_N d_L d_T)^2$ elements out of $(MNLT)^2$ are non-zero. Besides, note that these non-zero elements have a very specific position through $\bar{\mathbf{D}}_m$, which are in turn replicated along the different dimensions of cyclic convolutions. If we were to change the values of $\mathscr{D}_m \in \mathbb{R}^{2\times 2\times 2\times 2}$, these new values would occupy the exact same positions in $\bar{\mathbf{D}}_m$. However, if we were to change $\mathscr{D}_m$'s or $\mathscr{X}_m$'s size then the self-replicating structure would change significantly. Figure 31 shows a single $\bar{\mathbf{D}}_m$ example, while Figure 32 presents an schematic of matrix $\bar{\mathbf{D}} = [\bar{\mathbf{D}}_1 ... \bar{\mathbf{D}}_{M_d}]$ with $M_d = 4$.

With this in mind we can formulate the Convolutional Sparse Coding for Spectral Videos (CSC4D) dual minimization problem, analogously to CSC3D, as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{D}}\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1, \tag{99}$$

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{X}}\mathbf{d} - \mathbf{s} \right\|_2^2 + \iota_{\mathscr{C}_P}(\mathbf{d}), \tag{100}$$

where $\mathbf{s} = \text{vect}(\mathscr{S}) \in \mathbb{R}^{MNLT}$ and ι_{C_Z} is a constraint function. Just as in the CSC3D case, each dictionary element is desired to be small and one-norm, in order to avoid the scaling ambiguity between dictionary filters and coefficients. For this, we zero-pad each of the M_d dictionary elements

using the operator $\mathbf{Z}_p : \mathbb{R}^{d_M d_N d_L d_T} \rightarrow \mathbb{R}^{MNLT}$, in order to match the dimensions of the operator $\bar{\mathbf{X}} \in \mathbb{R}^{MNLT \times MNLTM_d}$. Then, the constraint set (Wohlberg, 2016c)

$$\mathscr{C}_{Z_p} = \left\{ \mathbf{x} \in \mathbb{R}^{MNLT} : (\mathbf{I} - \mathbf{Z}_p \mathbf{Z}_p^T)\mathbf{x} = 0, \|\mathbf{x}\|_2 = 1 \right\}, \tag{101}$$

guarantees that the obtained dictionary elements are normalized and the zero-padding is removed.

Finally, the indicator function of the constrained set is introduced as

$$\iota_{\mathscr{C}_Z}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in \mathscr{C}_{Z_p} \\ \\ \infty & \text{if } \mathbf{x} \notin \mathscr{C}_{Z_p}. \end{cases}, \tag{102}$$

and applied over each vectorized individual convolutional dictionary, but for notation simplicity it will be applied over the whole collection. The single 4D cyclic convolutio- nal operator is listed as the third contribution of this doctoral dissertation and its results can be found in (Barajas-Solano et al., 2020).

Reconstruction Update Problem (RU). Eq. (99) is called the *Reconstruction Update Problem* (RU) and aims to learn the spatial-spectral-temporal contributions of each one of the elements in a given convolutional dictionary collection. Considering the nature and size of the product $\mathbf{H}\bar{\mathbf{D}}$, we propose to solve it using the alternating directions multiplier method, ADMM

(Boyd et al., 2010), by introducing one auxiliary variable as

$$\begin{aligned} &\underset{\mathbf{x},\mathbf{u},\mathbf{v}}{\operatorname{argmin}} \frac{1}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{s}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \\ &\text{s.t.: } \mathbf{v}=\mathbf{x}, \end{aligned} \tag{103}$$

with augmented Lagrangian given by

$$\mathscr{L}\{\mathbf{x},\mathbf{v},\mathbf{g}\}=\frac{1}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{s}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}+\mathbf{g}\right\|_2^2, \tag{104}$$

where $\mathbf{g}$ is the so called dual variable. The variable updates are obtained from Eq. (104) as

$$\mathbf{x}^{j+1}:=\underset{\mathbf{x}}{\operatorname{argmin}}\frac{1}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{s}\right\|_2^2+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}^j+\mathbf{g}^j\right\|_2^2, \tag{105}$$

$$\mathbf{v}^{j+1}:=\underset{\mathbf{v}}{\operatorname{argmin}}\frac{\rho}{2}\left\|\mathbf{x}^{j+1}-\mathbf{v}+\mathbf{g}^j\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \tag{106}$$

$$\mathbf{g}^{j+1}=\mathbf{g}^j+\mathbf{x}^{j+1}-\mathbf{v}^{j+1}. \tag{107}$$

The dual variable $\mathbf{g}$ can be interpreted as a vector of prices and Eq. (107) is then called a *price update* or *price adjustment step* (Boyd et al., 2010). Note that the update steps (105) to (107) for CSC4D's RU are similar in its formulation to CSC3D's CUP (46) to (48), with a different equivalent convolutional operator. Except for the equivalent convolutional operator, CSC4D's RU formulation can be solved just as CSC3D's CUP.

Coefficients Map Update. The linear representation of the 4D convolution in Eq. (105) can be solved efficiently by profiting the DFT convolution theorem (Bristow and Eriksson, 2013), which states that a n-dimensional cyclic convolution can be expressed as a Hadamard product in the n-dimensional Fourier domain as

$$\sum_{m=1}^{M_d} \mathscr{D}_m \overset{4}{*} \mathscr{X}_m = \mathscr{F}_{4D}^{-1}\left(\sum_{m=1}^{M_d} \mathscr{F}_{4D}(\mathscr{D}_m) \odot \mathscr{F}_{4D}(\mathscr{X}_m)\right). \tag{108}$$

Eq. (108) can be simplified by transforming the sums of Hadamard products into a matrix-vector product. First we create the equivalent operator $\hat{\hat{\mathbf{D}}} \in \mathbb{C}^{MNLT \times MNLTM_d}$ transforming to the Fourier domain and concatenating as described in Algorithms 10, and 6 with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. Second, we create $\hat{\mathbf{x}} \in \mathbb{C}^{MNLTM_d}$ by transforming to the Fourier domain and unfolding as described in Algorithms 10, and 4, also with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. Then, the sum of 4D convolutions can be solved efficiently as

$$\sum_{m=1}^{M_d} \mathscr{D}_m \overset{4}{*} \mathscr{X}_m \simeq \mathscr{F}_{4D}^{-1}\left(\hat{\hat{\mathbf{D}}}\hat{\mathbf{x}}\right), \tag{109}$$

minding, of course, that $\hat{\hat{\mathbf{D}}}\hat{\mathbf{x}}$ must be folded first into a 4D array. Just as in CSC3D, we profit on the Fourier solution in Eq. (109) to solve optimally Eq. (105), replacing $\mathbf{w}^j = \mathbf{v}^j - \mathbf{g}^j \in \mathbb{R}^{MNLT}$ and creating $\hat{\mathbf{w}}^j$ by folding, transforming to the Fourier domain and unfolding (Algorithms 8, 10, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$). Finally, we can rewrite Eq. (105) in the Fourier

domain as

$$\hat{\mathbf{x}}^{j+1} := \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| \hat{\mathbf{D}}\hat{\mathbf{x}} - \hat{\mathbf{s}} \right\|_2^2 + \frac{\rho}{2} \left\| \hat{\mathbf{x}} - \hat{\mathbf{w}}^j \right\|_2^2. \tag{110}$$

Just as in Eq. (51), Eq. (110) can be solved by using the Woodbury Matrix Inverse method (Henderson and Searle, 1981), explained in Appendix 2, as

$$\hat{\mathbf{x}}^{j+1} = \mathbf{b} - \hat{\bar{\mathbf{D}}}^H \left(\mathbf{I} + \hat{\bar{\mathbf{D}}}\hat{\bar{\mathbf{D}}}^H \right)^{-1} \hat{\bar{\mathbf{D}}}\mathbf{b}, \tag{111}$$

with $\mathbf{b} = \hat{\bar{\mathbf{D}}}^H \hat{\mathbf{s}}^j + \hat{\mathbf{w}}^j$. Just as with CSC3D and CSC3D-CSI, the solution for $\hat{\mathbf{x}}^{j+1}$ in CSC4D has the same structure. This is because the CSC4D formulation is a natural extension of CSC3D. The nature of a single ND-convolutional operator allows for simplifying the mathematical formulations. The real difference between Eq. (53) and Eq. (111) lies in the dimensions of the arrays, which require a particular treatment, as explained in Appendix 4. Finally, the update $\mathbf{x}^{j+1}$ is obtained by folding, transforming from the Fourier domain and unfolding (Algorithms 8, 12, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$).

Sparse Coefficient Maps Update. Eq. (106) has a closed form solution via soft thresholding (Rockafellar, 1970) as

$$\mathbf{v}^{j+1} = \mathscr{S}_{\frac{\lambda}{\rho}} \left(\mathbf{x}^{j+1} + \mathbf{g}^j \right). \tag{112}$$

The RU solution is summarized in Algorithm 19.

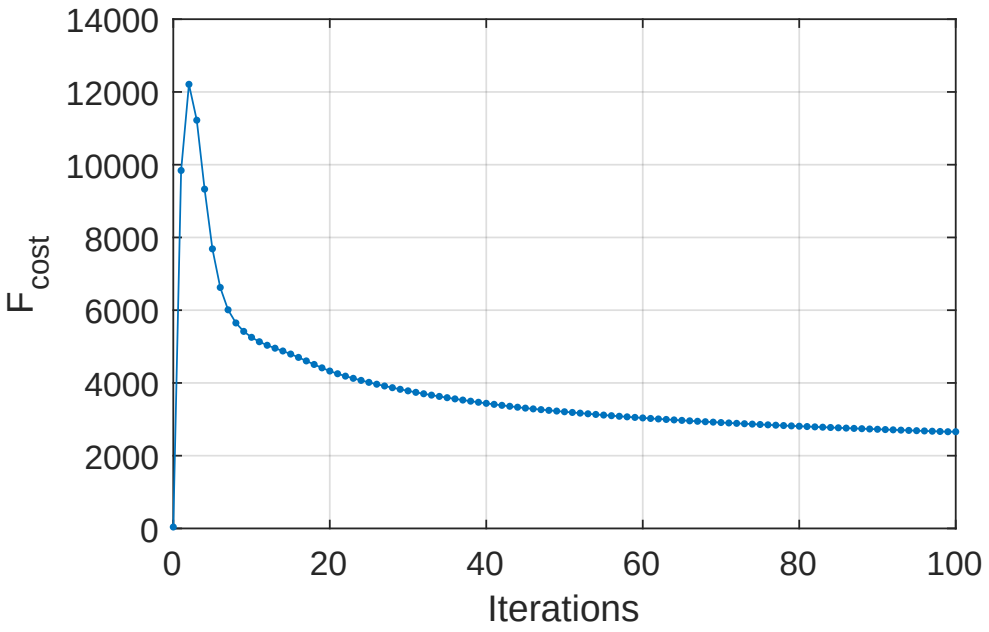


Figure 29. Typical evolution of the objective function of the proposed CSC3D-CSI using Algorithm: 18: Compressive Spectral Convolutional 3D for CSI Algorithm - CSC3D-CSI.

Algorithm 19 RU Solution for CSC4D

Require: $\bar{\mathbf{D}}$, $\hat{\bar{\mathbf{D}}}$, $\mathbf{v}^j$, $\mathbf{g}^j$, ρ, λ and sizes M, N, L, T and M_d.

1: Build $\hat{\mathbf{z}}^j$.
2: Solve $\mathbf{x}^{j+1}$ in Eq. (105) using Eq. (111).
3: Solve $\mathbf{v}^{j+1}$ Eq. (106) using Eq. (112).
4: Solve $\mathbf{g}^{j+1}$ in Eq. (107).
5: **return** Sparse coefficient maps $\mathbf{v}^{j+1}$, split variable update $\mathbf{v}^{j+1}$, and dual variable update $\mathbf{g}^{j+1}$.

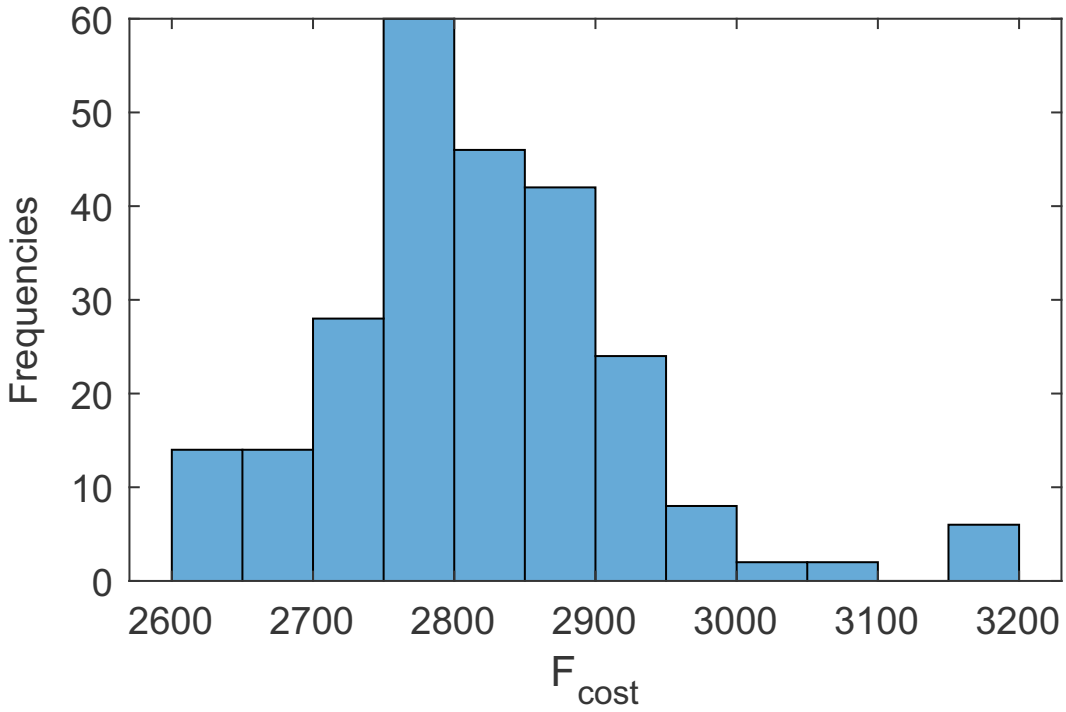


Figure 30. Histogram of the final values of the objective function of the proposed CSC3D-CSI (71) obtained after 200 different random initializations.

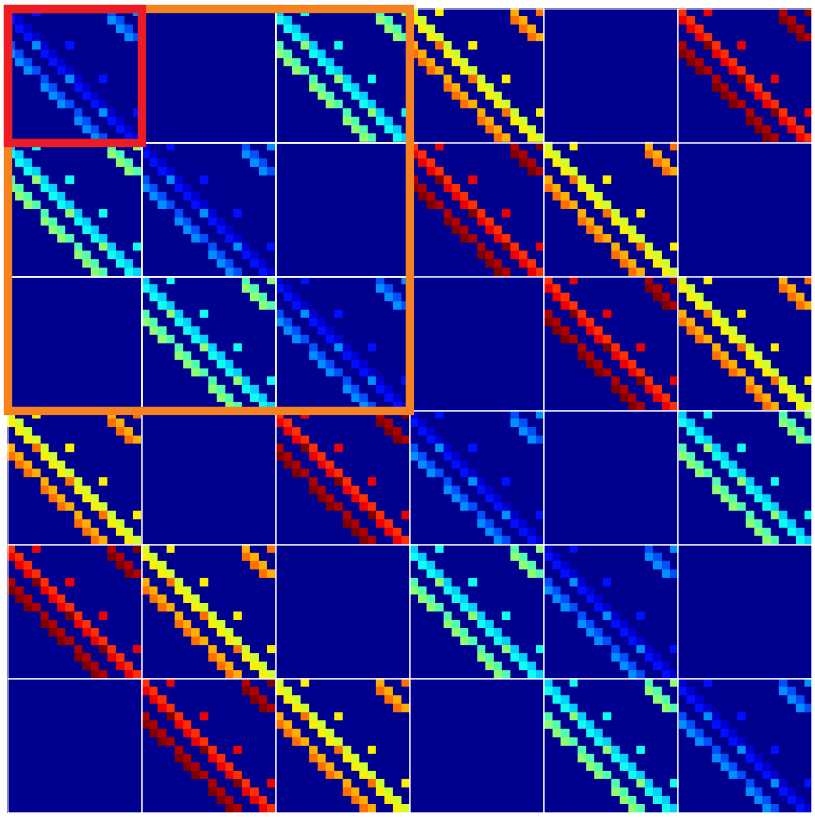


Figure 31. Example of $\bar{\mathbf{D}}_m$ for $\mathscr{D}_m \in \mathbb{R}^{2\times2\times2\times2}$ and $\mathscr{X}_m \in \mathbb{R}^{4\times4\times3\times2}$. The red rectangle highlights a 2D equivalent cyclic convolution matrix; while the gold rectangle highlights the 3D equivalent cyclic convolution matrix.

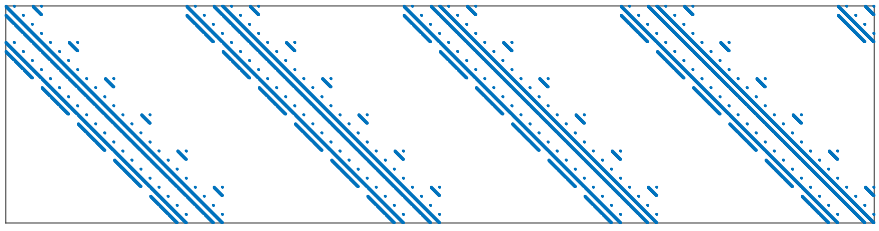


Figure 32. Schematic for $\bar{\mathbf{D}}$ with $M = 10, N = 10, L = 4, T = 6, M_d = 4$ and $d_M = d_N = d_L = d_T = 2$

Feature Extraction Problem (FE). Eq. (100) is referred to as the *Feature Extraction Problem* (FE) and seeks to adjust a collection of convolutional 4D dictionary elements to a given collection of sparse coefficient maps. Eq. (100) can be solved using the alternating directions multiplier method, ADMM (Boyd et al., 2010), by introducing an auxiliary variable as

$$\begin{aligned} &\underset{\mathbf{d},\mathbf{p},\mathbf{q}}{\operatorname{argmin}} \frac{1}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2+\iota_{\mathscr{C}_Z}(\mathbf{q}), \\ &\text{s.t.: } \mathbf{q}=\mathbf{d}. \end{aligned} \tag{113}$$

The augmented Lagrangian for Eq. (113) can be written as stated in (Barajas-Solano et al., 2019a) as

$$\mathscr{L}\{\mathbf{d},\mathbf{q},\mathbf{t}\}=\frac{1}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2+\iota_{\mathscr{C}_Z}(\mathbf{q})+\frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}+\mathbf{t}\right\|_2^2, \tag{114}$$

where $\mathbf{q}$ is the so called dual variable. The variable updates are obtained from Eq. (114) as

$$\mathbf{d}^{j+1}:=\underset{\mathbf{d}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{s}\right\|_2^2+\frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}^j+\mathbf{t}^j\right\|_2^2, \tag{115}$$

$$\mathbf{q}^{j+1}:=\underset{\mathbf{q}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\mathbf{d}^{j+1}-\mathbf{q}+\mathbf{t}^j\right\|_2^2+\iota_{\mathscr{C}_Z}(\mathbf{q}), \tag{116}$$

$$\mathbf{t}^{j+1}=\mathbf{t}^j+\mathbf{d}^{j+1}-\mathbf{q}^{j+1}. \tag{117}$$

The dual variable $\mathbf{t}$ can be interpreted as a vector of prices and Eq. (117) is then called a *price update* or *price adjustment step* (Boyd et al., 2010).

Convolutional Dictionary Update. Problem (115) can be solved efficiently by profiting the DFT convolution theorem as described for problem (105), thus we rewrite problem (115) as

$$\hat{\mathbf{d}}^{j+1} := \underset{\hat{\mathbf{d}}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \hat{\hat{\mathbf{X}}} \hat{\mathbf{d}} - \hat{\mathbf{s}}^j \right\|_2^2 + \frac{\sigma}{2} \left\| \hat{\mathbf{d}} - \hat{\mathbf{w}}^j \right\|_2^2, \tag{118}$$

by replacing $\mathbf{w}^j = \mathbf{q}^j - \mathbf{t}^j \in \mathbb{R}^{MNLT} \in \mathbb{R}^{MNLTM_d}$, folding, transforming to the Fourier domain and unfolding into $\hat{\mathbf{w}}^j \in \mathbb{C}^{MNLTM_d}$ (Algorithms 8, 10, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$); the operator $\hat{\hat{\mathbf{X}}}$ is creating by folding, transforming to the Fourier domain and creating the equivalent operator from $\mathbf{v}^{j+1}$ in Eq. (112) (Algorithms 8, 10, and 6 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$). Then, as with Eq. (60), Eq. (118) has closed solution

$$\hat{\mathbf{d}}^{j+1} = \mathbf{b} - \hat{\hat{\mathbf{X}}}^H \left(\mathbf{I} + \hat{\hat{\mathbf{X}}} \hat{\hat{\mathbf{X}}}^H \right)^{-1} \hat{\hat{\mathbf{X}}} \mathbf{b}, \tag{119}$$

with $\mathbf{b} = \hat{\hat{\mathbf{X}}} \hat{\mathbf{s}} + \hat{\mathbf{w}}^j \in \mathbb{C}^{MNLTM_d}$. Finally, the update $\mathbf{d}^{j+1}$ is obtained by folding, transforming from the Fourier domain and unfolding (Algorithms 8, 12, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$).

Desired Convolutional Dictionary Update. Eq. (116) has closed solution via the proximal of $\imath_{\mathscr{C}_Z}$ (Rockafellar, 1970)

$$\mathbf{q}^{j+1} = \frac{\mathbf{Z}_p \mathbf{Z}_p^T (\mathbf{d}^{j+1} + \mathbf{t}^j)}{\left\| \mathbf{Z}_p \mathbf{Z}_p^T (\mathbf{d}^{j+1} + \mathbf{t}^j) \right\|}. \tag{120}$$

The FE solution is summarized in Algorithm 20.

Algorithm 20 FE Solution for CSC4D

Require: $\bar{\mathbf{X}}$, $\hat{\bar{\mathbf{X}}}$, $\mathbf{q}^j$, $\mathbf{t}^j$, σ and sizes M, N, L , T and M_d.
1: Build $\hat{\mathbf{z}}^j$.
2: Solve $\mathbf{d}^{j+1}$ in Eq. (115) using (Eq. (119).
3: Solve $\mathbf{q}^{j+1}$ Eq. (116) using (Eq. (120).
4: Solve $\mathbf{t}^{j+1}$ in Eq. (117).
5: **return** :Convolutional dictionary $\mathbf{d}^{j+1}$, split variable update $\mathbf{q}^{j+1}$, and dual variable update $\mathbf{t}^{j+1}$.

Proposed CSC4D. The proposed CSC4D framework consists in alternately solving two related problems, RU and FE, in order to obtain both a collection of sparse coefficient maps and convolu- tional dictionary elements. Additionally, the regularization parameters ρ, λ, and σ can be updated in each iteration according to (Boyd et al., 2010), section 3.3, or they can be set to a fixed value. Thus, this doctoral dissertation proposes to solve both problems alternately, as exposed in Algorithm 21.

Algorithm 21 CSC4D Algorithm

Require: $\{\boldsymbol{\mathscr{X}}_m^0 \in \mathbb{R}^{M\times N\times L} | m = 1,...,M_d\}$ as zeros ; $\{\boldsymbol{\mathscr{D}}_m^0 \in \mathbb{R}^{M\times N\times L} | m = 1,...,M_d\}$ as random; $M, N, L, M_d, d, \rho_0, \lambda$, σ_0 and $\mathbf{s}$.
1: Set $\{\boldsymbol{\mathscr{V}}_m^0\} = \{\boldsymbol{\mathscr{X}}_m^0\}$ and build the vectorization $\mathbf{v}^0$.
2: Set $\{\boldsymbol{\mathscr{Q}}_m^0\} = \{\boldsymbol{\mathscr{D}}_m^0\}$ and build the vectorization $\mathbf{q}^0$.
3: Build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from $\{\boldsymbol{\mathscr{D}}_m^0\}$.
4: Set $j = 0$.
5: **repeat**
6: Solve RU as explained in Algorithm 19.
7: Update ρ^{j+1} according to (Boyd et al., 2010), section 3.3
8: Fold $\mathbf{v}^{j+1}$ into $\{\boldsymbol{\mathscr{V}}_m^{j+1}\}$, as explained in Algorithm 7, and build $\bar{\mathbf{X}}$ and $\hat{\bar{\mathbf{X}}}$ from it.
9: Solve FE as explained in Algorithm 20.
10: Update σ^{j+1} according to (Boyd et al., 2010), section 3.3
11: Fold $\mathbf{q}^{j+1}$ into $\{\boldsymbol{\mathscr{Q}}_m^{j+1}\}$, as explained in Algorithm 7, and build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from it.
12: **until** the residuals meet a given tolerance, or completed a number of iterations.
13: **return** the sparse coefficient maps $\{\boldsymbol{\mathscr{V}}_m^{j+1}\}$ and the convolutional dictionary elements $\{\boldsymbol{\mathscr{Q}}_m^{j+1}\}$

As with all iterative methods, ADMM is sensitive to initial values. Different empiric initia-

lization alternatives for the initial dictionary elements $\{\mathscr{D}_m^0\}$ were tested, which included a full random cube and variations of zero-value cubes with some non-zero positions. The chosen alternative can be described as a centered random-value subcube $\in \mathbb{R}^{d_2\times d_2\times d_2\times d_2}$, with $d_2 = d/2$, within an all zeros $\mathbb{R}^{d\times d\times d\times d}$ cube. This variation showed the best trade-off between the highest Peak Signal-to-Noise Rate value (PSNR, for reconstruction quality) and the lowest NoN Zero percentage of elements (NNZ, for sparsity), in both mean and standard deviation.

About the convergence boundaries analysis, the proposed CSC4D is equal to the CSC3D's $\ell_2 - \ell_1$ and ℓ_2-indicator function. For this reason, the same analysis applied in the CSC3D formulation works for the CSC4D formulation.

Estimated Numerical Complexity. We will now estimate the numerical complexity of the more complex subproblems in CUP and DUP. The closed solution to Eq. (106), Eq. (112), is obtained by a soft-thresholding problem and subproblem (116)'s solution, Eq. (120), is akin to a hard-thresholding problem. Both complexities are negligible.

One of the most important sources of numerical complexity is the 4D convolutional operation, with complexity $\mathscr{O}((MNLT)^2)$, considering that the dictionary elements are zero-padded to match the dimensions of the sparse coefficient maps. The 4D convolution complexity is reduced to a fraction by expressing it as a Hadamard product, profiting on the DFT Theorem, reducing the cost to $\mathscr{O}(MNLT\log(MNLT))$.

Eq.	Complexity	Solution
(106) and (116)	Negible	Implemented
(105) and (115)	$\mathscr{O}((MNLTM_d)^3)$	Original
(111) and (119)	$\mathscr{O}(MNLTM_d)$	Implemented

Table 8
Complexity review of the proposed CSC4D algorithm.

Finally, the greatest source of numerical complexity are the inversions as solutions to subproblems (105) and (115). Again, the canonical complexity for inverting $\hat{\mathbf{A}}^H\hat{\mathbf{A}}+\alpha\mathbf{I}\in\mathbb{R}^{MNLTM_d\times MNLTM_d}$ is $\mathscr{O}((MNL\,TM_d)^3)$. Howe- ver, by profiting on the concatenated diagonal structure of $\hat{\bar{\mathbf{D}}}$ and $\hat{\bar{\mathbf{X}}}$ and the dimensions rearrangement exposed in Appendix B of (Barajas-Solano et al., 2019a), the inversion complexity falls to $\mathscr{O}(MNL\,TM_d)$ as shown in Eqs. (111) and (119). Table 8 summarizes the different complexities for the stated subproblems and their solutions.

6.3. CSC4D Synthetic Performance Evaluation

We now evaluate the performance of the proposed CSC4D operator for sparsely representing SVs against the SSR model using an ADMM based algorithm (Boyd et al., 2010).

The performance comparison is carried out by computing the Peak Signal-to-Noise Ratio (PSNR, dB) for assessing the reconstruction quality; while the percentage of non-zero elements (NNZ, %) is used for evaluating the sparsity of the estimated coefficients. NNZ can be also described as $\|\mathbf{x}\|_0/|\mathbf{x}|$ for the estimated coefficients vector using the SSR model, where $|\mathbf{x}|$ is the total

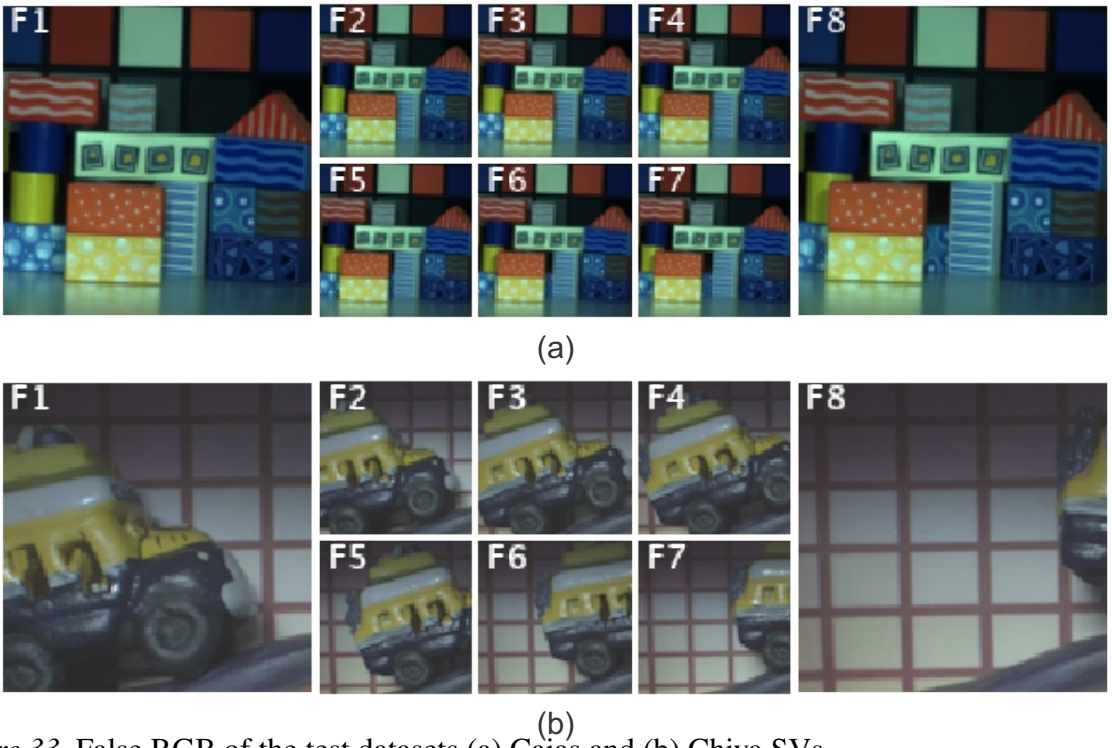


Figure 33. False RGB of the test datasets (a) Cajas and (b) Chiva SVs.

number of coefficients. The sparsity of the coefficients is measured according to (Papyan et al., 2017)

$$\text{NNZ} = \max_{m=1}^{M_d} \frac{\|\mathbf{x}_m\|_0}{|\mathbf{x}_m|}. \tag{121}$$

Two laboratory-captured spectral videos, Cajas and Chiva (see Fig 33) (Leon-Lopez et al., 2019) were used for this test. Both data sets have spatial resolution 128×128, 16 spectral bands and 8 frames long. In particular, the high-frequency versions of the SVs under tests are recovered using the proposed algorithm, where a reconstructed version of the image under test is obtained by adding the low-frequency components to the previously recovered high-frequency video.

The number of convolutional elements was fixed to $M_d = 30$ and the dictionary sizes were

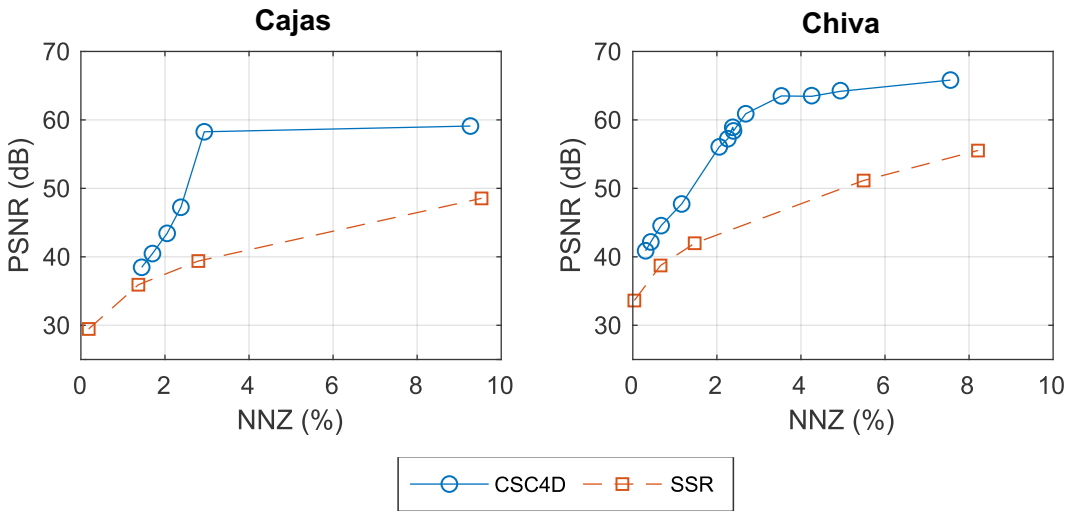


Figure 34. Reconstruction quality performance (PSNR) of the simulated results for various levels of sparsity (NNZ), for the proposed CSC4D and SSR approach with different time-compression schemes.

fixed to $d_M = d_N = d_L = 8$ and $d_T = 3$. For both methods, CSC4D and SSR, the parameter λ was varied in order to get the reconstruction quality in function of the sparsity level. All other regularizer parameters were fixed to optimal values. Fig 34 shows the sparse representation performance, for both datasets, for both frameworks. However, the main tendency is conserved. The proposed CSC4D method outperforms the SSR approach by up to 20dB.

6.4. CSC4D and SR

In order to recover a high-resolution SV from low-resolution measurements, we propose to express the low-resolution measurements as a decimated version of the objective dataset: $\dot{\mathbf{s}} = \mathbf{H}\mathbf{s}$, where $\mathbf{s} = \text{vec}(\mathscr{S}) \in \mathbb{R}^{MNLT}$, $\dot{\mathbf{s}} \in \mathbb{R}^{M_1N_1L_1T}$ is the vector containing the low-resolution measurements and $\mathbf{H} \in \mathbb{R}^{M_1N_1L_1T \times MNLT}$ is a decimation matrix with $M_1 = M/\alpha$, $N_1 = N/\beta$, $L_1 = L/\gamma$, and $\alpha, \beta, \gamma \in \mathbb{N}^*$. Using the CSC4D model then $\dot{\mathbf{s}} = \mathbf{H}\bar{\mathbf{D}}\mathbf{x} = \mathbf{H}\bar{\mathbf{X}}\mathbf{d}$. Just as in the 3D case, we propose

two linked optimization problems, solved alternately, for obtaining the convolutional dictionary elements and sparse coefficient maps. Then, the general optimization problems can be formulated as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{D}}\mathbf{x} - \dot{\mathbf{s}} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1, \tag{122}$$

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{X}}\mathbf{d} - \dot{\mathbf{s}} \right\|_2^2 + \iota_{C_Z}(\mathbf{d}), \tag{123}$$

Considering that convolutional dictionaries have a low performance for representing the low-frequency components of multidimensional signals (Wohlberg, 2016a), this work uses the high-frequencies versions of a dataset of interest. This version is obtained by performing a high-pass filtering stage to the image data. On the other hand, the low-frequencies version of the original dataset is interpolated up to scale, and added to the recovered high-resolution high-frequencies version, completing the recovered high-resolution SV $\mathscr{S}_1$, as shown in Fig 35.

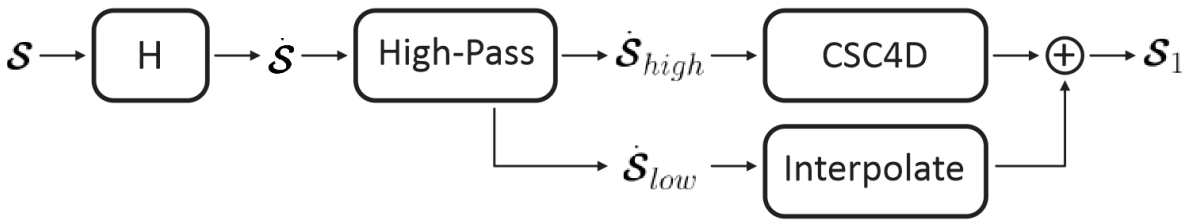


Figure 35. Proposed CSC4D-SR scheme.

Reconstruction Update Problem for SR (RU-SR). Eq. (122) is called the *Reconstruction Update problem for SR* (RU-SR) and aims to learn the spatial-spectral-temporal contributions of each one of the elements in a given convolutional dictionary collection from low-res measurements.

Considering the nature and size of the product $\mathbf{H}\bar{\mathbf{D}}$, we propose to solve it using the alternating directions multiplier method, ADMM (Boyd et al., 2010), by introducing two auxiliary variables as

$$\begin{aligned} &\underset{\mathbf{x},\mathbf{u},\mathbf{v}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\dot{\mathbf{s}}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \\ &\text{s.t.: } \mathbf{u}=\bar{\mathbf{D}}\mathbf{x}, \\ &\mathbf{v}=\mathbf{x}. \end{aligned} \tag{124}$$

Considering the model for the N-dimensional CSC framework in Eq. (35), we propose to profit on CSC3D formulation (Barajas-Solano et al., 2019a) and extend it on the required update steps. Then, considering the augmented Lagrangian for Eq. (124) given by

$$\mathscr{L}\{\mathbf{x},\mathbf{u},\mathbf{v},\mathbf{f},\mathbf{g}\}=\frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\dot{\mathbf{s}}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1+\frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}+\mathbf{f}\right\|_2^2+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}+\mathbf{g}\right\|_2^2, \tag{125}$$

we obtain the update steps

$$\mathbf{x}^{(j+1)}:=\underset{\mathbf{x}}{\operatorname{argmin}}\frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}^{(j)}+\mathbf{f}^{(j)}\right\|_2^2+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}^{(j)}+\mathbf{g}^{(j)}\right\|_2^2, \tag{126}$$

$$\mathbf{u}^{(j+1)}:=\underset{\mathbf{u}}{\operatorname{argmin}}\frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\dot{\mathbf{s}}\right\|_2^2+\frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}^{(j+1)}-\mathbf{u}+\mathbf{f}^{(j)}\right\|_2^2, \tag{127}$$

$$\mathbf{v}^{(j+1)}:=\underset{\mathbf{v}}{\operatorname{argmin}}\frac{\rho}{2}\left\|\mathbf{x}^{(j+1)}-\mathbf{v}+\mathbf{g}^{(j)}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \tag{128}$$

$$\mathbf{f}^{(j+1)}=\mathbf{f}^{(j)}+\bar{\mathbf{D}}\mathbf{x}^{(j+1)}-\mathbf{u}^{(j+1)}, \tag{129}$$

$$\mathbf{g}^{(j+1)} = \mathbf{g}^{(j)} + \mathbf{x}^{(j+1)} - \mathbf{v}^{(j+1)}, \tag{130}$$

where $\mathbf{f}$ and $\mathbf{g}$ are the so called dual variables, and the super indexes (j) and $(j+1)$ refer to the iteration steps.

Coefficient Maps Update. It should be noted that solving Eq. (126) requires the costly creation of operator $\bar{\mathbf{D}} \in \mathbb{R}^{MNLT \times MNLTM_d}$, and its transpose. By profiting on the DFT for the 4D case

$$\sum_{m=1}^{M_d} \mathscr{D}_m \overset{4}{*} \mathscr{X}_m = \mathscr{F}_{4D}^{-1}\left(\sum_{m=1}^{M_d} \mathscr{F}_{4D}(\mathscr{D}_m) \odot \mathscr{F}_{4D}(\mathscr{X}_m) \right), \tag{131}$$

we propose to solve Eq. (126) in the Fourier domain as

$$\hat{\mathbf{x}}^{(j+1)} := \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| \hat{\bar{\mathbf{D}}}\hat{\mathbf{x}} - \hat{\mathbf{z}}^{(j)} \right\|_2^2 + \frac{\rho}{2} \left\| \hat{\mathbf{x}} - \hat{\mathbf{w}}^{(j)} \right\|_2^2, \tag{132}$$

by performing the following transformations:

- $\mathbf{x} \in \mathbb{R}^{MNLTM_d}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{x}} \in \mathbb{C}^{MNLTM_d}$ as indicated in Algorithms 8, 10, and 4.
- $\mathbf{w}^{(j)} = \mathbf{v}^{(j)} - \mathbf{g}^{(j)} \in \mathbb{R}^{MNLTM_d}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{w}} \in \mathbb{C}^{MNLTM_d}$ as indicated in Algorithms 8, 10, and 4.
- $\mathbf{z}^{(j)} = \mathbf{u}^{(j)} - \mathbf{f}^{(j)} \in \mathbb{R}^{MNLT}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{x}} \in \mathbb{C}^{MNLT}$ as indicated in Algorithms 7, 9, and 3.

- $\hat{\bar{\mathbf{D}}}$ is created from $\{\mathscr{D}_m\}$ by transforming to the Fourier domain and creating the equivalent operator as indicated in Algorithms 10, and 6.

All the previous rearrangements are performed using $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. The original $\bar{\mathbf{D}}$ operator must be created as a concatenation of equivalent convolu- tional matrices with defined structure, like the example in figure 31; while the operator $\hat{\bar{\mathbf{D}}}$ can be created as the concatenation of diagonal matrices, which is much easier to build. Although there is an associated cost of $\mathscr{O}(MNLT\log(MNLT))$ for each 4D Fourier Transform, the memory save and simplicity of operating simpler diagonal matrices makes up for this cost.

Finally, as with Eq. (76), Eq. (132) has closed solution

$$\hat{\mathbf{x}}^{(j+1)} = \hat{\mathbf{b}} - \hat{\bar{\mathbf{D}}}^{\mathbf{H}} \left(\mathbf{I} + \hat{\bar{\mathbf{D}}}\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\right)^{-1} \hat{\bar{\mathbf{D}}}\hat{\mathbf{b}}. \tag{133}$$

with $\hat{\mathbf{b}} = \hat{\bar{\mathbf{D}}}^{\mathbf{H}}\hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)} \in \mathbb{C}^{MNLTM_d}$. Eq. (133) can be solved using the dimensions rearrangements proposed in the Appendices 3 and 4, minding the increase in the dimensions. Finally, the update $\mathbf{x}^{j+1}$ is obtained from $\hat{\mathbf{x}}^{j+1}$ by folding, obtaining the inverse transform and unfolding following Algorithms 8, 12, and 4 respectively with $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$.

Temporal Recovery From Low-Res Measurements Update. Eq. (127) has closed solution in the spatial domain as

$$\mathbf{u}^{(j+1)} = \left(\mathbf{H^T H} + \rho \mathbf{I}\right)^{-1} \left(\mathbf{H^T \dot{s}} + \rho \mathbf{z}^{(j)}\right), \tag{134}$$

with $\mathbf{z}^j = \bar{\mathbf{D}}\mathbf{x}^{j+1} + \mathbf{f}^j \in \mathbb{R}^{MNLT}$, and can also be solved using Woodbury's matrix identity and profiting $\mathbf{H H^T} = \mathbf{I}$ as

$$\mathbf{u}^{(j+1)} = \frac{1}{\rho}\left[\mathbf{b} - \left(\frac{1}{\rho+1}\right)\mathbf{H^T H b}\right]. \tag{135}$$

with $\mathbf{b} = \mathbf{H^T \dot{s}} + \rho \mathbf{z}^{(j)}$.

Eq. (135) can be solved optimally by performing the products right-to-left. This is, start by solving the matrix-to-vector product $\mathbf{Hb}$, and continuing left wise, instead of solving the matrix-to-matrix product $\mathbf{H}^T\mathbf{H}$ first.

Sparse Coefficient Maps Update. The last update step, Eq. (128), has a closed form solution via soft thresholding (Rockafellar, 1970) as

$$\mathbf{v}^{(j+1)} = \mathscr{S}_{\frac{\lambda}{\rho}}\left(\mathbf{x}^{(j+1)} + \mathbf{g}^{(j)}\right). \tag{136}$$

The variable updates for the RU-SR problem in CSC4D-SR are summarized in the Algorithm 22:

Algorithm 22 RU-SR for CSC4D-SR

Require: $\dot{\mathbf{s}}, \mathbf{H}, \mathbf{d}^j, \mathbf{u}^j, \mathbf{v}^j, \mathbf{f}^j, \mathbf{g}^j, \lambda$

1: create $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from $\mathbf{d}^j$
2: create $\hat{\mathbf{w}} = \mathscr{F}_{4D}\left\{\mathbf{u}^j - \mathbf{f}^j\right\}$ and $\hat{\mathbf{z}} = \mathscr{F}_{4D}\left\{\mathbf{v}^j - \mathbf{g}^j\right\}$
3: solve $\hat{\mathbf{x}}^{j+1}$ using Eq. (133), fold it into $\{\hat{\boldsymbol{\mathscr{X}}}_m^{j+1}\}$ as explained in Algorithm 7, and $\mathbf{x}^{j+1} = \text{vec}\left(\mathscr{F}_{4D}^{-1}\left\{\{\hat{\boldsymbol{\mathscr{X}}}_m^{j+1}\}\right\}\right)$
4: solve $\mathbf{u}^{j+1}$ using Eq. (135)
5: solve $\mathbf{v}^{j+1}$ using Eq. (136)
6: update $\mathbf{f}^{j+1}$ and $\mathbf{g}^{j+1}$ using Eq. (129) and (130)
7: **return** the updated split and dual variables $\mathbf{u}^{j+1}, \mathbf{v}^{j+1}, \mathbf{f}^{j+1}, \mathbf{g}^{j+1}$

Feature Extraction Problem for SR (FE-SR). Just as the sparse coefficient maps, the collection of convolutional dictionary elements are also learned from the low-res measurements. The minimization in Eq. (123) is called the Feature Extraction problem (FE), and is also solved using ADMM by introducing two auxiliary variables

$$\begin{aligned} &\underset{\mathbf{d},\mathbf{p},\mathbf{q}}{\text{argmin}} \frac{1}{2}\|\mathbf{H}\mathbf{p} - \dot{\mathbf{s}}\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \\ &\text{s.t.: } \mathbf{p} = \bar{\mathbf{X}}\mathbf{d}, \\ &\mathbf{q} = \mathbf{d}, \end{aligned} \tag{137}$$

with augmented Lagrangian

$$\mathscr{L}\{\mathbf{d},\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{t}\} = \frac{1}{2}\|\mathbf{H}\mathbf{p} - \mathbf{y}\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}) + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d} - \mathbf{p} + \mathbf{r}\right\|_2^2 + \frac{\sigma}{2}\|\mathbf{d} - \mathbf{q} + \mathbf{t}\|_2^2, \tag{138}$$

and updates

$$\mathbf{d}^{(j+1)} := \underset{\mathbf{d}}{\text{argmin}} \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d} - \mathbf{p}^{(j)} + \mathbf{r}^{(j)}\right\|_2^2 + \frac{\sigma}{2}\left\|\mathbf{d} - \mathbf{q}^{(j)} + \mathbf{t}^{(j)}\right\|_2^2, \tag{139}$$

$$\mathbf{p}^{(j+1)} := \underset{\mathbf{p}}{\text{argmin}} \frac{1}{2}\|\mathbf{H}\mathbf{p} - \dot{\mathbf{s}}\|_2^2 + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}^{(j+1)} - \mathbf{p} + \mathbf{r}^{(j)}\right\|_2^2, \tag{140}$$

$$\mathbf{q}^{(j+1)} := \underset{\mathbf{q}}{\operatorname{argmin}} \frac{\sigma}{2} \left\| \mathbf{d}^{(j+1)} - \mathbf{q} + \mathbf{t}^{(j)} \right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{q}), \tag{141}$$

$$\mathbf{r}^{(j+1)} = \mathbf{r}^{(j)} + \bar{\mathbf{X}}\mathbf{d}^{(j+1)} - \mathbf{p}^{(j)}, \tag{142}$$

$$\mathbf{t}^{(j+1)} = \mathbf{t}^{(j)} + \mathbf{d}^{(j+1)} - \mathbf{t}^{(j)}. \tag{143}$$

where $\mathbf{r}$ and $\mathbf{t}$ are the so called dual variables, and the superindexes j and $j+1$ refer to the iteration step.

Convolutional Dictionary Update. Eq. (139) and Eq. (126) have analog structures, thus can be solved as

$$\hat{\mathbf{d}}^{(j+1)} = \left(\hat{\bar{\mathbf{X}}}^{\mathbf{H}} \hat{\bar{\mathbf{X}}} + \mathbf{I} \right)^{-1} \left(\hat{\bar{\mathbf{X}}}^{\mathbf{H}} \hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)} \right), \tag{144}$$

where

- $\mathbf{d} \in \mathbb{R}^{MNLTM_d}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{d}} \in \mathbb{C}^{MNLTM_d}$ as indicated in Algorithms 8, 10, and 4.
- $\mathbf{w}^{(j)} = \mathbf{q}^{(j)} - \mathbf{t}^{(j)} \in \mathbb{R}^{MNLTM_d}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{w}} \in \mathbb{C}^{MNLTM_d}$ as indicated in Algorithms 8, 10, and 4.
- $\mathbf{z}^{(j)} = \mathbf{p}^{(j)} - \mathbf{r}^{(j)} \in \mathbb{R}^{MNLT}$ is folded, transformed to the Fourier domain and unfolded into $\hat{\mathbf{x}} \in \mathbb{C}^{MNLT}$ as indicated in Algorithms 7, 9, and 3.
- $\hat{\bar{\mathbf{X}}}$ is created from $\{\mathscr{X}_m\}$ by transforming to the Fourier domain and creating the equivalent operator as indicated in Algorithms 10, and 6.

All the previous rearrangements are performed using $L_1 \times ... \times L_{\bar{N}} = M \times N \times L \times T$. The same numerical rearrangements from Apendices 3 and 4 also apply, minding the increase in the dimensions. Eq. (144) has closed solution

$$\hat{\mathbf{d}}^{(j+1)} = \hat{\mathbf{b}} - \hat{\bar{\mathbf{X}}}^{\mathbf{H}} \left(\mathbf{I} + \hat{\bar{\mathbf{X}}} \hat{\bar{\mathbf{X}}}^{\mathbf{H}} \right)^{-1} \hat{\bar{\mathbf{X}}} \hat{\mathbf{b}}. \tag{145}$$

with $\hat{\mathbf{b}} = \hat{\bar{\mathbf{X}}}^{\mathbf{H}} \hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)} \in \mathbb{C}^{MNLTM_d}$. Finally, $\mathbf{d}^{(j+1)}$ is built from $\hat{\mathbf{d}}^{(j+1)}$ just as $\mathbf{x}^{(j+1)}$ is built from $\hat{\mathbf{x}}^{(j+1)}$.

Temporal Recovery From Low-Res Measurements Update. Again, Eq. (140) is analog to Eq. (127), thus can be solved in the spatial domain as

$$\mathbf{p}^{(j+1)} = \frac{1}{\sigma} \left[\mathbf{b} - \left(\frac{1}{\sigma + 1} \right) \mathbf{H}^{\mathbf{T}} \mathbf{H} \mathbf{b} \right], \tag{146}$$

where $\mathbf{b} = \mathbf{H}^{\mathbf{T}} \dot{\mathbf{s}} + \sigma \mathbf{z}^{(j)}$ and $\mathbf{z}^{(j)} = \bar{\mathbf{X}} \mathbf{d}^{(j+1)} + \mathbf{r}^{(j)}$.

Desired Convolutional Dictionary Update. Finally, Eq. (141) can solved via proximal for each m-4D dictionary element as

$$\mathbf{q}_m^{(j+1)} = \frac{\mathbf{Z}_p \mathbf{Z}_p^{\mathbf{T}} (\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)})}{\left\| \mathbf{Z}_p \mathbf{Z}_p^{\mathbf{T}} (\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)}) \right\|_2}, \tag{147}$$

The variable updates for the FE-SR problem are summarized in the Algorithm 23:

Algorithm 23 FE-SR for CSC4D-SR

Require: $\dot{\mathbf{s}}, \mathbf{H}, \mathbf{x}^j, \mathbf{p}^j, \mathbf{q}^j, \mathbf{a}^j, \mathbf{b}^j, \lambda$
1: create $\bar{\mathbf{X}}$ and $\hat{\bar{\mathbf{X}}}$ from $\mathbf{x}^j$
2: create $\hat{\mathbf{w}} = \mathscr{F}_{4D}\left\{\mathbf{p}^j - \mathbf{a}^j\right\}$ and $\hat{\mathbf{z}} = \mathscr{F}_{4D}\left\{\mathbf{q}^j - \mathbf{b}^j\right\}$
3: solve $\hat{\mathbf{d}}^{j+1}$ using Eq. (145), fold it into $\{\hat{\boldsymbol{\mathscr{D}}}_m^{j+1}\}$ as explained in Algorithm 7, and $\mathbf{d}^{j+1} = \text{vec}\left(\mathscr{F}_{4D}^{-1}\left\{\{\hat{\boldsymbol{\mathscr{D}}}_m^{j+1}\}\right\}\right)$
4: solve $\mathbf{p}^{j+1}$ using Eq. (146)
5: solve $\mathbf{q}^{j+1}$ using Eq. (147)
6: update $\mathbf{a}^{j+1}$ and $\mathbf{b}^{j+1}$ using Eq. (142) and (143)
7: **return** the updated split and dual variables $\mathbf{p}^{j+1}, \mathbf{q}^{j+1}, \mathbf{a}^{j+1}, \mathbf{b}^{j+1}$

Proposed CSC4D-SR. Problems RU-SR and FE-SR are solved alternately, with RU-SR's updates feeding FE-SR and vice versa, per iteration, as explained in Algorithm 24. The process continues until a desired reconstruction error, desired sparsity threshold or a maximum number of iterations is achieved.

The CSC4D-SR formulation is considered the fourth contribution of this doctoral dissertation and its results van be found in (Barajas-Solano et al., 2020).

Algorithm 24 CSC4D-SR

Require: $\dot{\mathbf{s}}, \mathbf{H}, \{\boldsymbol{\mathscr{D}}_m^0\}, \{\boldsymbol{\mathscr{X}}_m^0\}, \lambda, \rho, \sigma, M, N, L, T, Md, d_M, d_N, d_L, d_T$
1: *Initialization:*
2: build the vectorization $\mathbf{d}^0$ from $\{\boldsymbol{\mathscr{D}}_m^0\}$
3: set $\{\boldsymbol{\mathscr{V}}_m^0\} = \{\boldsymbol{\mathscr{X}}_m^0\}$ and build the vectorization $\mathbf{v}^0$
4: set $\{\boldsymbol{\mathscr{Q}}_m^0\} = \{\boldsymbol{\mathscr{D}}_m^0\}$ and build the vectorization $\mathbf{q}^0$
5: set $\boldsymbol{\mathscr{U}}^0 = \boldsymbol{\mathscr{P}}^0 = \sum_{m=1}^{M_d} \boldsymbol{\mathscr{D}}_m^0 \overset{4}{*} \boldsymbol{\mathscr{X}}_m^0$ and build de vectorizations $\mathbf{u}^0$ and $\mathbf{p}^0$
6: set j=0;
7: *Iterative Process:*
8: **repeat**
9: solve $(\mathbf{u}^{j+1}, \mathbf{v}^{j+1}, \mathbf{f}^{j+1}, \mathbf{g}^{j+1}) = \text{RU-SR}(\dot{\mathbf{s}}, \mathbf{H}, \mathbf{d}^j, \mathbf{u}^j, \mathbf{v}^j, \mathbf{f}^j, \mathbf{g}^j, \lambda, \rho)$ as explained in Algorithm 22
10: solve $(\mathbf{p}^{j+1}, \mathbf{q}^{j+1}, \mathbf{a}^{j+1}, \mathbf{b}^{j+1}) = \text{FE-SR}(\dot{\mathbf{s}}, \mathbf{H}, \mathbf{v}^{j+1}, \mathbf{p}^j, \mathbf{q}^j, \mathbf{a}^j, \mathbf{b}^j, \sigma)$ as explained in Algorithm 23
11: set $\mathbf{d}^j = \mathbf{q}^{j+1}$
12: **until** the residuals meet a given tolerance, or completed a number of iterations.
13: create $\{\boldsymbol{\mathscr{D}}_m\}$ and $\{\boldsymbol{\mathscr{X}}_m\}$ from $\mathbf{q}^{j+1}$ and $\mathbf{v}^{j+1}$, respectively
14: **return** $\boldsymbol{\mathscr{S}}_1 = \sum_{m=1}^{M_d} \boldsymbol{\mathscr{D}}_m \overset{4}{*} \boldsymbol{\mathscr{X}}_m$

Estimated Numerical Complexity. We will now estimate the numerical complexity of the more complex subproblems in FE-SR and RU-SR. Subproblem (128)'s solution, Eq. (136), is obtained by a soft-thresholding problem and subproblem (141)'s solution, Eq. (147), is akin to a hard-thresholding problem. Both complexities are negligible.

Subproblems (127) and (140) have the same solution structure where the inversion of $\mathbf{H^T H} + \alpha\mathbf{I} \in \mathbb{R}^{MNLT \times MNLT}$ represents the higher complexity. According to the Appendix A of (Barajas-Solano et al., 2019a), their complexity can be reduced from $\mathscr{O}((MNLT)^3)$ to $\mathscr{O}((M_1N_1L_1T)^2 MNLT)$, with $M_1N_1L_1 \ll MNL$ and $\mathbf{H} \in \mathbb{R}^{M_1N_1L_1T \times MNLT}$, by profiting on $\mathbf{HH}^T = \mathbf{I}$.

One of the most important sources of numerical complexity is the 4D convolutional operation, with complexity $\mathscr{O}((MNLT)^2)$, considering that the dictionary elements are zero-padded to match the dimensions of the sparse coefficient maps. The 3D convolution complexity is reduced to a fraction by expressing it as a Hadamard product, profiting on the Discrete Fourier Transform (DFT) Theorem, reducing the cost to $\mathscr{O}(MNLT \log(MNLT))$.

Finally, the greatest source of numerical complexity are the inversions as solutions to subproblems (126) and (139). Again, the canonical complexity for inverting $\hat{\mathbf{A}}^H \hat{\mathbf{A}} + \alpha\mathbf{I} \in \mathbb{R}^{MNLTM_d \times MNLTM_d}$ is $\mathscr{O}((MNLTM_d)^3)$. However, by profiting on the concatenated diagonal structure of $\hat{\hat{\mathbf{D}}}$ and $\hat{\hat{\mathbf{X}}}$ and the dimensions rearrangement exposed in Appendix B of (Barajas-Solano et al., 2019a), the inversion complexity falls to $\mathscr{O}(MNL\, TM_d)$. Table 9 summarizes the different complexities for the

stated subproblems and their solutions.

Eq.	Complexity	Solution
(128) and (141)	Negligible	Implemented
(127) and (140)	$\mathcal{O}((MNLT)^3)$	Original
(135) and (146)	$\mathcal{O}((M_1N_1L_1T)^2MNLT)$	Implemented
(126) and (139)	$\mathcal{O}((MNLTM_d)^3)$	Original
(133) and (145)	$\mathcal{O}(MNLTM_d)$	Implemented

Table 9
Complexity review of the proposed CSC4D+SR algorithm.

6.5. CSC4D-SR Synthetic Performance Evaluation

We now evaluate the performance of the proposed CSC4D method at recovering SVs from a spatially decimated version. For this test we use the same datasets as in section 5.5.1. All the regularizer parameters were fixed to optimal values, and the decimation matrix was creating using $\alpha = 2$, $\beta = 2$ and $\gamma = 1$ so $\dot{\mathbf{s}} \in \mathbb{R}^{64 \cdot 64 \cdot 16 \cdot 8}$ for both datasets. The mean reconstruction qualities were 30.07dB for CSC4D-SR and 29.23dB for SSR. For the Chiva dataset, the mean reconstruction qualities were 39.52dB for CSC4D-SR and 37.08dB for SSR (see Fig. 36(a) and 36(b)). Besides outperforming the state-of-the-art method in PSNR values, the proposed CSC4D-SR also outperforms SSR in edge reconstruction, as seen in Fig. 37(a) and 37(b). Note the sharp edges in the images reconstructed with the CSC4D-SR method, while the SSR's reconstructed images have blurred edges.

The same experiment was recreated with higher resolution versions of the same datasets,

	CSC4D-SR		SSR	
	Mean PSNR	Mean SSIM	Mean PSNR	Mean SSIM
Cajas	**30.07dB**	**0.97**	29.23dB	0.94
Chiva	**39.52dB**	**0.97**	37.08dB	0.93
Cajas 256×256	**35.93dB**	**0.98**	33.48dB	0.92
Chiva 256×256	**46.48dB**	**0.99**	44.53dB	0.97

Table 10
Mean PSNR and SSIM of the simulated results for the 8-frame SV collection, using both frameworks and four datasets using the proposed CSC4D-SR. The proposed CSC4D-SR outperforms the SSR method by up to 2dB in all scenarios. All standard deviations were bellow 1×10^{-3}*, so they are not shown.*

256×256 of spatial resolution. For the Cajas dataset, the mean reconstructi- on qualities were 35.93dB for CSC4D-SR and 33.48dB for SSR. For the Chiva dataset, the mean reconstruction qualities were 46.48dB for CSC4D-SR and 44.53dB for SSR (see Fig. 38(a) and 38(b)). Again, the proposed CSC4D-SR generates sharper edges than the state-of-the-art approach (see Fig. 39(a) and 39(b)). The edge sharpness can be measured using SSIM, as shown in Table 10.

The CSC3D approach, created for sparsely representing SIs, can be easily scaled to represent the SVs as 4D datasets, without introducing new temporal operators. The proposed CSC4D operator, and the CSC4D-SR model, profits on CSC's invariance to shifting and deformation, leading to better qualities when sparsely representing SVs and recovering SVs from decimated versions. The proposed CSC4D-SR model improves the definition of the reconstructed edges, outperforming the state-of-the-art. This improved sharpness improves the use and processing of the recovered SVs.

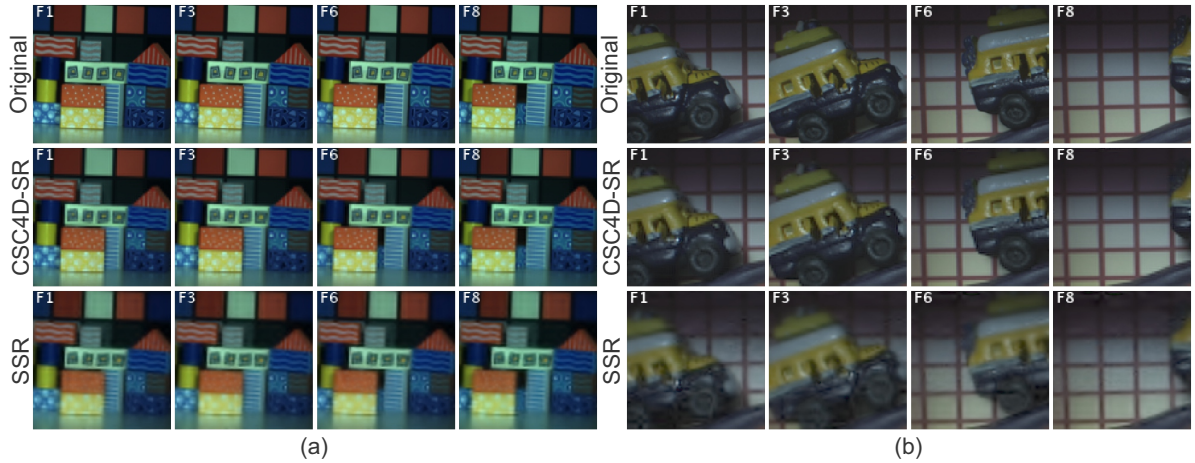


Figure 36. Simulated results of the reconstructed frames with the proposed CSC4D-SR and SSR methods, for the datasets (a) Cajas and (b) Chiva. Note that the proposed CSC4D-SR outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts.

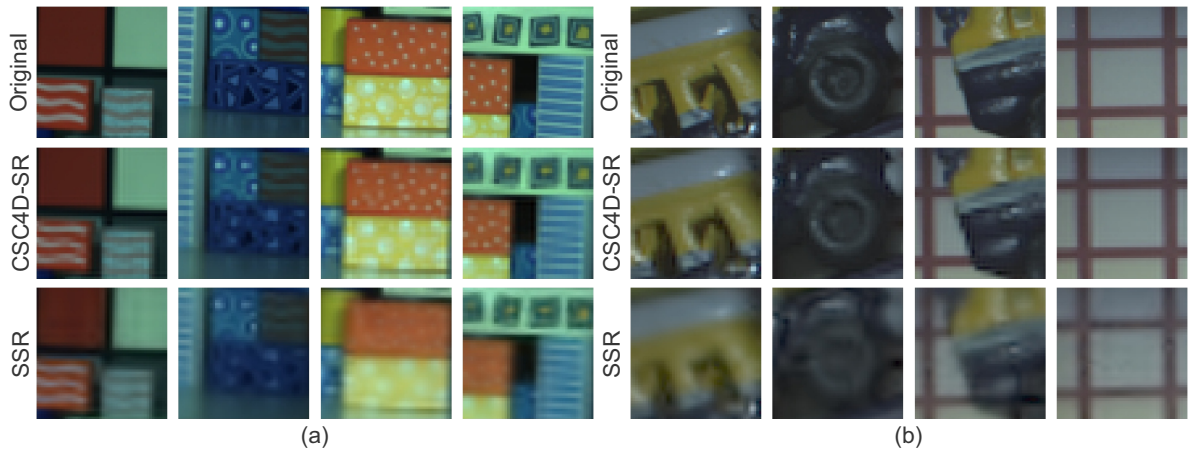


Figure 37. Simulated results of the reconstructed close-up details with the proposed CSC4D-SR and SSR methods, for the datasets (a) Cajas and (b) Chiva. For the Cajas dataset, note the sharper edges and dots definition in all panels. For the Chiva dataset, note the detail in the tire and the back of the toy, besides the lack of artifacts in the smooth areas of the last panel.

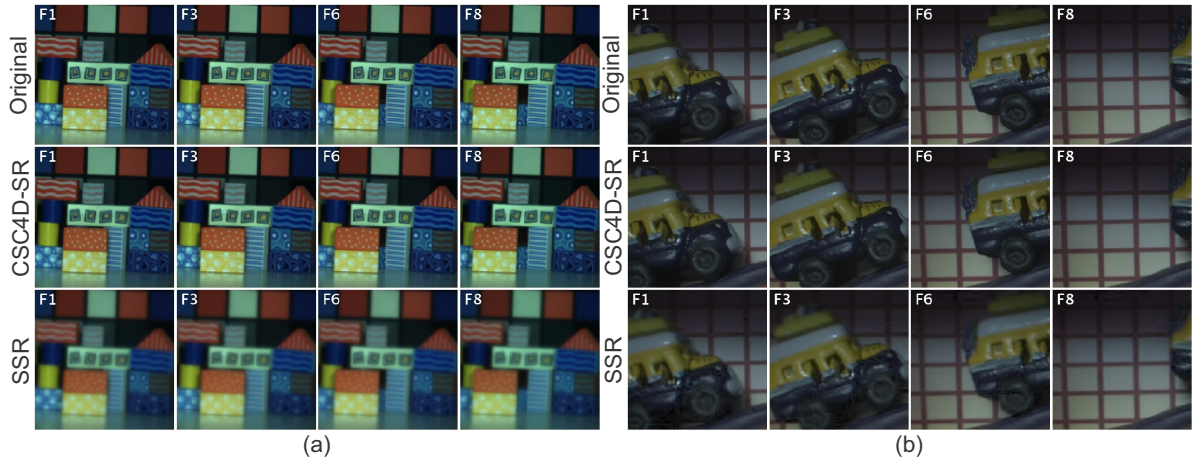


Figure 38. Simulated results of the reconstructed frames with the proposed CSC4D-SR and SSR methods, for the datasets (a) Cajas and (b) Chiva, 256×256 version. Note that the proposed CSC4D-SR outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts.

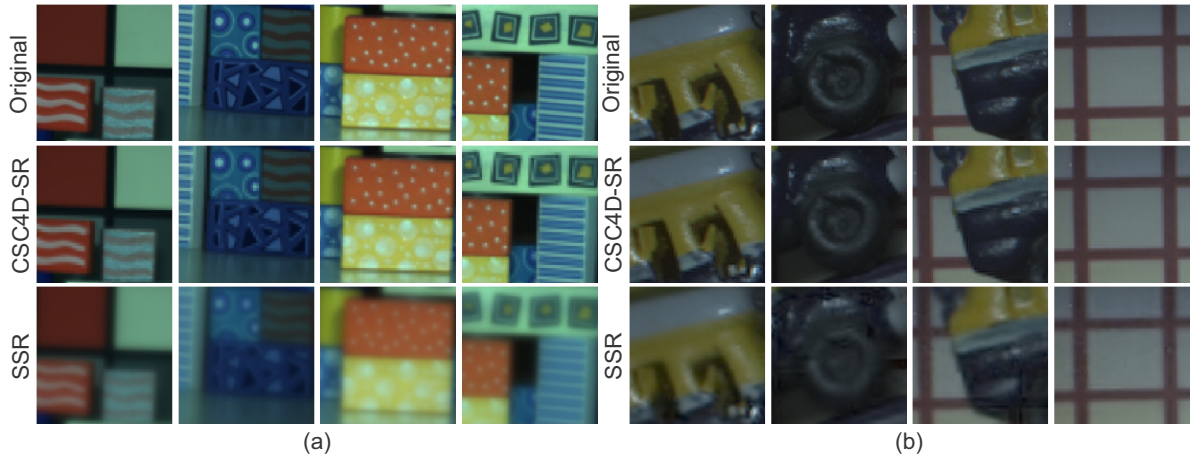


Figure 39. Simulated results of the reconstructed details with the proposed CSC4D-SR and SSR methods, for the datasets (a) Cajas and (b) Chiva, 256×256 version. Note that the proposed CSC4D-SR outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts. For the Cajas dataset, note the lack of definition around the color dots and stripes for the SSR method while the proposed CSC4D-SR matches the original dataset; for the Chiva dataset note the flawless details around the tire and toy body, besides the smooth areas.

6.6. Compressive Spectral Video Sensing (CSVS)

A more interesting approach for quickly capturing SVs is the application of the Com- pressive Sensing Imaging (CSI) framework to the SV case, also known as Compressive Spectral Video Sensing (CSVS). CSI states that a full 3D SI can be recovered from a set of 2D encoded projections. This is, a SI is simultaneously scanned and compressed, reducing both the scanning time and storage requirements (Correa et al., 2015; Arguello and Arce, 2013; Barducci et al., 2012; Wang et al., 2018). CSVS proposes to compressed sensing, consecutively, a collection of spectral frames using a collection of 2D coding apertures per spectral frame, and to recover the full SV using an extended version of the Sparse Signal Representation (SSR) model.

CSVS senses compressively each spectral frame as independent SIs, obtaining only a few measurements per spectral frame. Then, a full version of the SV is recovered from the compressed measurements using a spatial-spectral-temporal orthonormal basis, $\mathbf{\Psi}$, following the sparse signal representation model (SSR). Correa-Pugliese *et. al.* in (**?**) proposes to sparsely represent an SV using the Kronecker basis (**?**) along with the discrete cosine transform (DCT) for temporal compression, as $\mathbf{\Psi} = \text{2D Wavelet} \otimes DCT \otimes DCT$. Then, $\mathbf{s} = \mathbf{\Psi}\boldsymbol{\theta}$, where $\boldsymbol{\theta} \in \mathbb{R}^{MNLT}$ are the sparse coefficients.

The sensing matrix $\mathbf{H}$ for CSVS is a block diagonal concatenation of independent sensing

matrices $\left\{\mathbf{H}^t, t = 1, ..., T \mid \mathbf{H}^t \in \mathbb{R}^{K \times MNL}, K \ll MNL\right\}$, as (Leon-Lopez et al., 2019)

$$\begin{bmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^t \\ \vdots \\ \mathbf{y}^T \end{bmatrix} = \begin{bmatrix} \mathbf{H}^1 & 0 & \cdots & 0 & 0 \\ & \ddots & & & \\ \vdots & & \mathbf{H}^t & & \vdots \\ & & & \ddots & \\ 0 & 0 & \cdots & 0 & \mathbf{H}^T \end{bmatrix} \begin{bmatrix} \mathbf{s}^1 \\ \vdots \\ \mathbf{s}^t \\ \vdots \\ \mathbf{s}^T \end{bmatrix}, \tag{148}$$

where $\mathbf{H} \in \mathbb{R}^{KT \times MNLT}$ and $\mathbf{y} \in \mathbb{R}^{KT}$. Then, the recovery of an SV from compressed measurements had been typically addressed as the minimization problem

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} - \mathbf{y} \right\|_2^2 + \lambda \left\| \boldsymbol{\theta} \right\|_1, \tag{149}$$

where λ is a regularization constant that controls the trade off between the data fitting term and the sparsity inducing term (Arce et al., 2014). In order to improve the performance of the SSR recovery model, Lopez *et. al.* (Leon-Lopez et al., 2019) recently proposed to include an additional regularization term based on the optical flow as

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} - \mathbf{y} \right\|_2^2 + \lambda \left\| \boldsymbol{\theta} \right\|_1 + \beta \left\| \Delta \right\|_2^2, \tag{150}$$

where β is a regularization term and $\Delta = \Lambda(\bar{\mathbf{f}}_w)_{i,j} - \Lambda(\bar{\mathbf{f}}_z)_{i+u,j+v}$ is expressed in terms of the ho-

rizontal and vertical changes estimated from two upsampled and contiguous frames $\bar{\mathbf{f}}_w$ and $\bar{\mathbf{f}}_z$, obtained from a low-resolution reconstructed version of the original SV.

6.7. CSC4D and CSVS

The linear representation of CSC4D can be included within the CSVS recovery minimization in Eq. (149), replacing the SSR model as a representation basis, as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{D}}\mathbf{x} - \mathbf{y} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1, \tag{151}$$

and aims to learn a set of sparse coefficient maps $\mathbf{x}$ from compressed measurements $\mathbf{y} = \mathbf{H}\text{vect}(\mathscr{S})$, given a fixed convolutional dictionary $\bar{\mathbf{D}}$. Considering the specificity of the CSC model, the dictionary elements must also be learned from the compressed measurements by solving the minimization problem

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{X}}\mathbf{d} - \mathbf{y} \right\|_2^2 + \iota_{C_Z}(\mathbf{d}). \tag{152}$$

Eq. (151) is known as RU-CSVS and Eq. (152) as FE-CSVS, and both compose the CSC4D-CSVS model. Again, both minimization problems are solved alternately in order to recover a full version of the SV of interest from compressed measurements. It is worth noting that $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$ should not be considered, under any circumstance, as bases. Due to its rectangular size, its inverse must be obtained through a minimization scheme. Note that the proposed CSC4D-CSVS does not require to estimate a low-resolution version of the original SV in order to include the optical flow information, as in Equation 150, proposed by Lopez *et. al.* (Leon-Lopez et al., 2019). Having less

elements in the mathematical formulation simplifies its solution.

The CSC4D-CSVS model shares a similar formulation with the CSC4D-SR model, with the difference of matrix **H** and measurements **y**. For this reason, the closed-form solutions to each of CSC4D-SR's minimization problems can be repurposed with some changes in the optimized numerical routines as stated in Appendices 2 to 4.

6.8. CSC4D-CSVS Synthetic Performance Evaluation

The performance of the proposed CSC4D in CSVS was tested following a two-step scheme:

A. Recovery quality from compressed measurements:. The first step was to assess the recovery quality of the CSC4D from compressed measurements. For this case we use two collections of independent sensing matrices from two CSI architectures, 3D-CASSI (Cao et al., 2016) and C-CASSI (**?**), without the presence of noise, and compared to the SSR model. Specifically, we compared CSC4D to the classical CSVS recovery model, without the additional optical flow regularization term, considering that CSC4D does not use the optical flow information either.

B. Robustness to acquisition noise:. The second step was to assess the recovery quality of the CSC4D from compressed measurements in the presence of noise. We use two noise models, Gaussian white noise and Poisson noise, at different intensities, the latter to simulate sensing noise. Again, we compared the performance of the proposed CSC4D to the classical CSVS model.

The two-step test scheme was conducted using the same two datasets used in section 5.5.1. Both data sets have spatial resolution 128×128, 16 spectral bands and 8 frames. Considering that convolutional dictionaries have a low performance for representing low-frequency components of multidimensional signals (Wohlberg, 2016a), this work uses the high-frequencies versions of the original datasets, which are obtained by performing a high-pass filtering stage to the image data. For illustrative purposes, we add the low-frequency components to evaluate visually the recovered SVs.

The CSC approach is compared against the state-of-the-art SSR approach in all three steps of the performance test using the following metrics:

- the peak signal-to-noise ratio (PSNR) for measuring the overall reconstruction quality,
- the structural similarity index (SSIM) for edge sharpness, and
- the percentage of non-zero elements (NNZ) for measuring sparsity.

Considering that the convolutional coefficient maps are in fact a collection of M_d sparse tetrahedrons $\in \mathbb{R}^{M \times N \times L \times T}$, compared to the single sparse tetrahedron of the SSR model, then the sparsity of the convolutional coefficient maps will be measured as

$$\text{sparsity} = \max_{m=1}^{M_d} \frac{\|\boldsymbol{\mathcal{X}}_m\|_0}{MNLT}. \tag{153}$$

This is, the sparsity of the convolutional solutions will be the maximum sparsity of the individual coefficient maps. Following a series of previous experimental results, the sizes for the dictionary collection were fixed to $d_M = d_N = d_L = 8$, $d_T = 3$ and $M_d = 30$, for the entire scheme test.

Performance Using Noiseless Measurements from the 3D-CASSI and C-CASSI CSI Architecture. The first step of the test scheme is to assess the performance of the CSC4D at recovering full versions of compressed sensed SVs. First, we test CSC4D's using compressed measurements obtained using the 3D-CASSI CSI architecture. For this experiment we create collections of independently generated sensing matrices $\{\mathbf{H}^t, t = 1,...,8 \mid \mathbf{H}^t \in \mathbb{R}^{K\cdot 128\cdot 128\times 128\cdot 128\cdot 16}\}$, where $K = \{3,4\}$ are the number of shots per spectral frame, resulting in a compression of $18.75\,\%$ and $25\,\%$ respectively for both scenarios. Then, the sensing matrix can be defined as $\mathbf{H} \in \mathbb{R}^{K\cdot 128\cdot 128\cdot 8\times 128\cdot 128\cdot 16\cdot 8}$, according to Eq. (148). Table 11 presents the results of both the CSC4D+CSVS and the SSR model. CSC4D outperforms the SSR model at all ratios of compression, with both datasets.

Figures 40 and 41 show some example reconstructed frames as false RGB from compressed measurements taken with $K = 4$. Figures 42 and 43 show the error between the original SVs and the reconstructed versions from both models. Note the overall errors of the SSR model at recovering both textures and border details; while the frames recovered by the CSC4D+CSVS framework exhibit a higher overall quality and increased border sharpness, although still exist some missing specific details.

Dataset	K	Metric	CSC4D+CSVS	SSR
Cajas	3	PSNR	**36,30**	35,00
		SSIM	**0,960**	0,929
	4	PSNR	**36,53**	35,58
		SSIM	**0,961**	0,943
Chiva	3	PSNR	**51,83**	50,76
		SSIM	**0,996**	0,974
	4	PSNR	**54,60**	52,91
		SSIM	**0,997**	0,994

Table 11
Mean PSNR and mean SSIM of the simulated results at recovering the datasets using the 3D-CASSI, for both the proposed CSC4D+CSVS and SSR framework, and two compression ratios. The standard deviations for five repetitions were well bellow 1 % for both mean PSNR and mean SSIM, for this reason they are not shown.

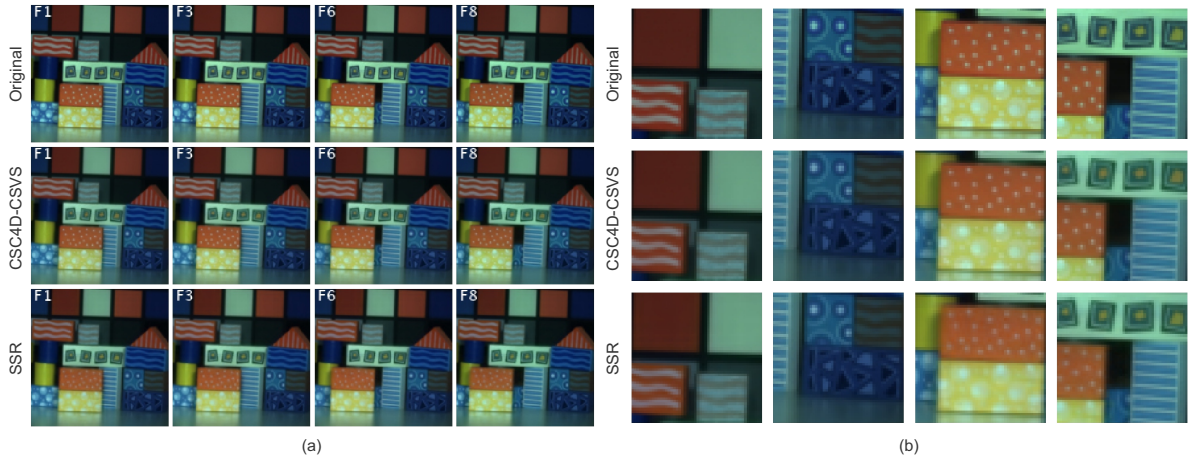


Figure 40. Example recovered frames of the simulated results from 3D-CASSI compressed measurements, at $K = 4$, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Cajas and (b) some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts. I.e., note the stripes in the first close up and the dot definitions in the third close up.

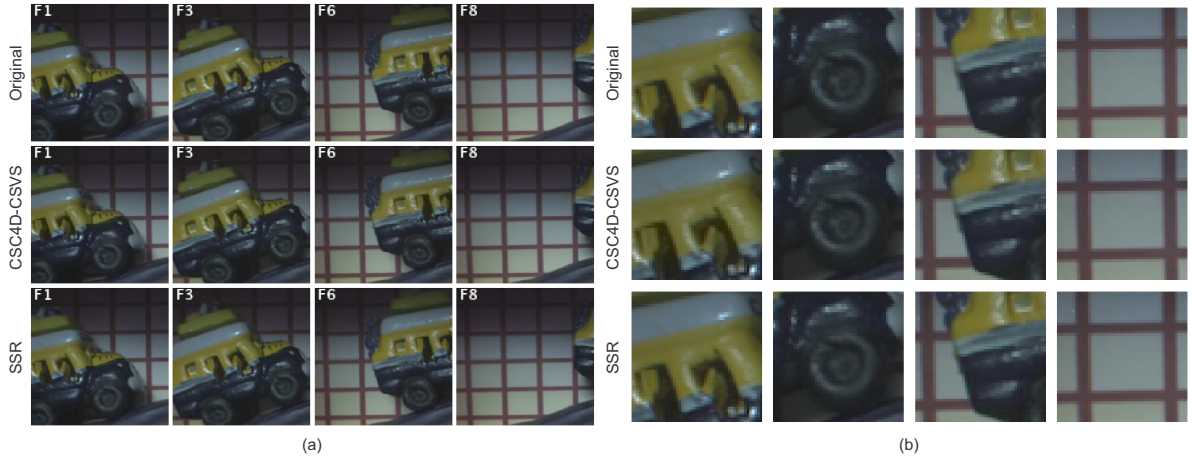


Figure 41. Example recovered frames of the simulated results from 3D-CASSI compressed measurements, at $K = 4$, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Chiva and (b) some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts.

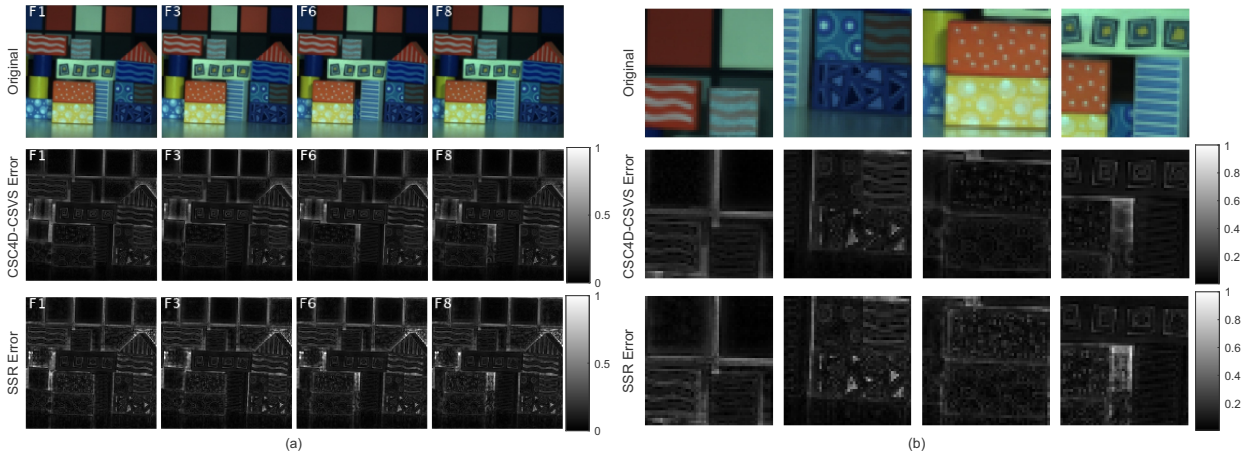


Figure 42. Reconstructed error frames of the simulated results recovered from 3D-CASSI compressed measurements, at $K = 4$, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Cajas and (b) some close-up details. Note that the proposed CSC4D-CSVS exhibits fewer errors in the uniform areas and around edges. I.e., note the body decals in the first close up and tire borders sharpness in the second close up.

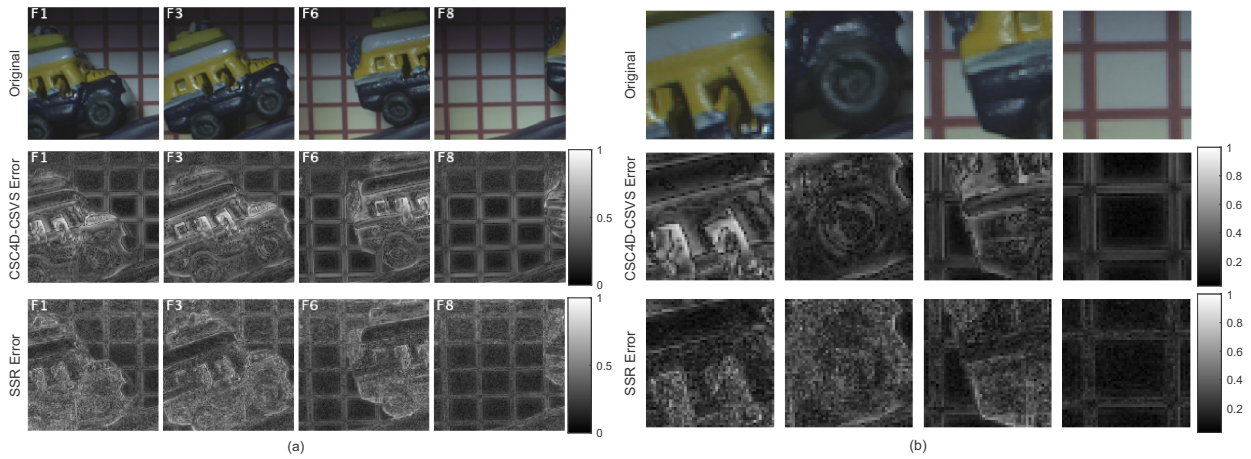


Figure 43. Reconstructed error frames of the simulated results recovered from 3D-CASSI compressed measurements, at $K = 4$, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Chiva and (b) some close-up details. Note that the proposed CSC4D-CSVS exhibits fewer errors in the uniform areas and around edges.

Second, we test the performance of the proposed CSC4D at recovering SVs from compressed measurements sensed using the C-CASSI CSI architecture. Again, we create a collection of independently generated sensing matrices $\{\mathbf{H}^t, t = 1,...,8 \mid \mathbf{H}^t \in \mathbb{R}^{K\cdot 128\cdot(128+16-1)\times 128\cdot 128\cdot 16}\}$, where $K = \{3,4\}$ are the number of shots per spectral frame, giving a compression of 20.94 % and 27.92 %, respectively, for both scenarios, according to $\gamma = KM(N+L-1)/MNL$ (**?**). Then, the sensing matrix can be defined as $\mathbf{H} \in \mathbb{R}^{K\cdot 128\cdot(128+16-1)\cdot 8\times 128\cdot 128\cdot 16\cdot 8}$, according to Eq. (148). Preliminary experiments showed that the CSC4D performs poorly when integrated with the C-CASSI CSI architectures.

As with the CSC3D-CSI case, the linear operators $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$ interact with the optical dispersion element represented in C-CASSI's sensing matrix, creating off-site replicas, as shown in Figure 19, adding noise to the recovery routine.To solve this issue, we recreated the side informa-

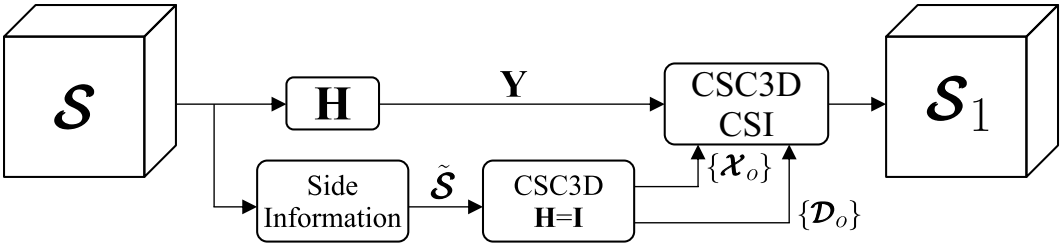


Figure 44. Side information scheme. Taken from (Barajas-Solano et al., 2019a).

tion scheme used in (Barajas-Solano et al., 2019a). The proposed side information system creates a gray-scale version of the original SV and replicates each gray-scale image at each band, so $\tilde{\mathscr{S}} \in \mathbb{R}^{M\times N\times L\times T}$. Note that $\tilde{\mathscr{S}}$ contains all the spatial patterns, structures and correlations of the original SV, but no spectral information. This additional information helps overcome the noise generated by the dispersive element in C-CASSI. Optimal initial collections $\{\tilde{\mathscr{D}}_m\}$ $\{\tilde{\mathscr{X}}_m\}$ are created from $\tilde{\mathscr{S}}$ and used as initializations for recovering $\mathscr{S}_1$, as shown in Figure 44.

The results of recovering both datasets using C-CASSI plus the proposed side information scheme are shown in Table 12, where the CSC4D model outperforms the state-of-the-art model.

Robustness to acquisition noise. The last step in the two-step test scheme is to assess the performance of the CSC4D algorithm in presence of noise. Two noise models were used: Gaussian white noise and Poisson noise, the latter for representing the acquisition noise. Three different levels of noise were added to the compressed measurements: 10dB, 20dB, and 30dB PSNR for Gaussian; 18dB, 23dB, and 28db for Poisson. Again, both CSI architectures were used, with two compression ratios, for both SVs. The side information scheme was again coupled with the C-

Dataset	K	Metric	CSC4D-CSVS	SSR
Cajas	3	PSNR	**36,21**	34,30
		SSIM	**0,935**	0,905
	4	PSNR	**36,45**	35,49
		SSIM	**0,957**	0,919
Chiva	3	PSNR	**45,53**	39,11
		SSIM	**0,974**	0,928
	4	PSNR	**46,75**	41,69
		SSIM	**0,981**	0,951

Table 12
Mean PSNR of the simulated results of the reconstructed datasets using the C-CASSI, for both the proposed CSC4D-CSVS and SSR framework and two compression ratios. The standard deviations for five repetitions were well bellow 1 % for both mean PSNR and mean SSIM, thus are not shown.

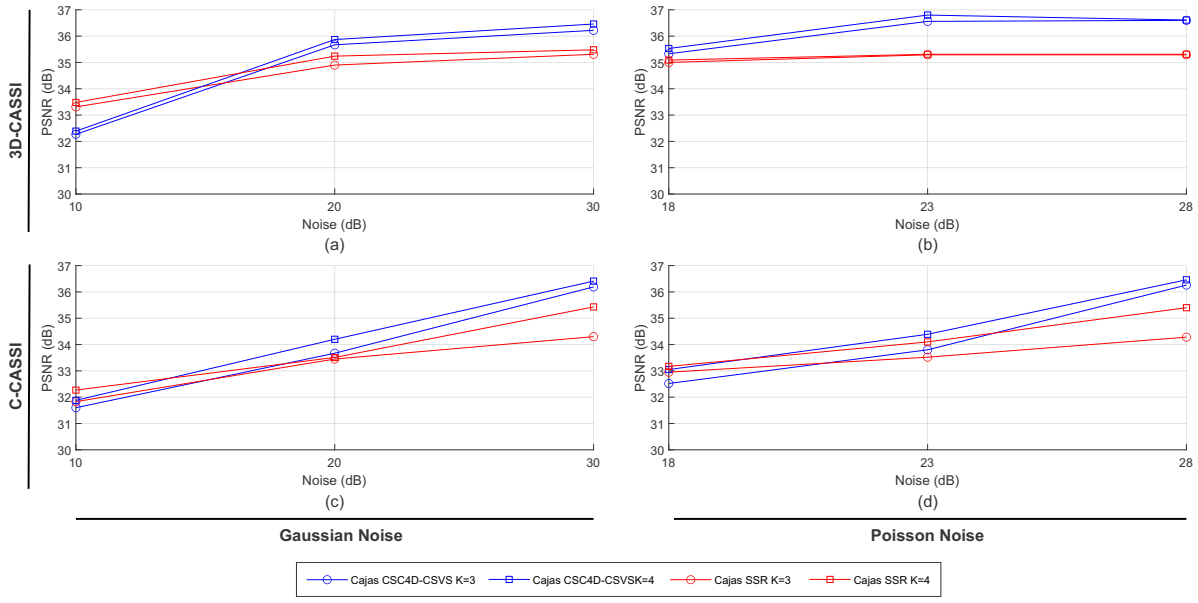


Figure 45. PSNR comparison of the simulated results at recovering the Cajas dataset for two CSVS techniques and two noise types, for both the proposed CSC4D-CSVS and SSR. The proposed CSC4D-CSVS matches the performance of the SSR method at high noise levels, and outperforms it and medium and low noise levels.

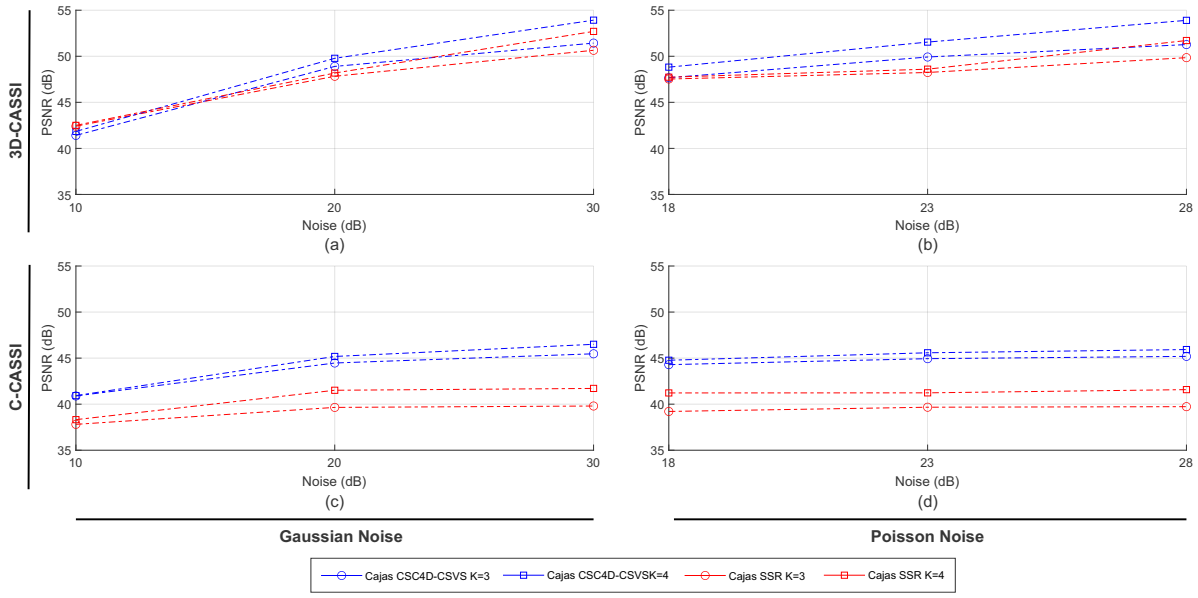


Figure 46. PSNR comparison of the simulated results at recovering the Chiva dataset for two CSVS techniques and two noise types, for both the proposed CSC4D-CSVS and SSR. The proposed CSC4D-CSVS matches the performance of the SSR method at high noise levels, and outperforms it and medium and low noise levels.

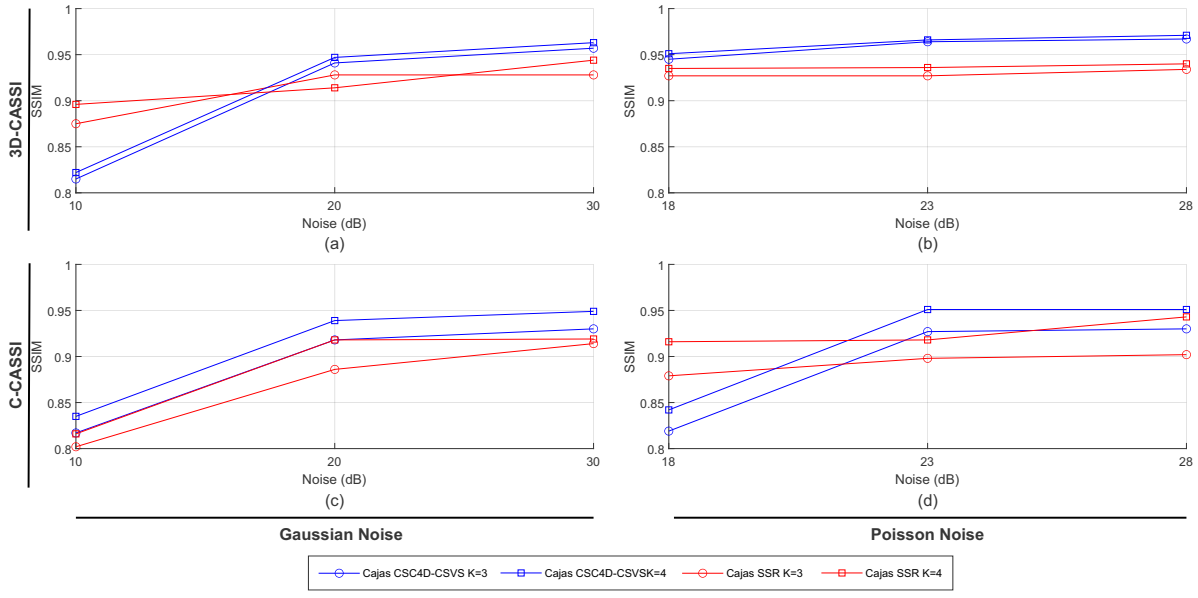


Figure 47. SSIM comparison of the simulated results at recovering the Cajas dataset for two CSVS techniques and two noise types, for both the proposed CSC4D-CSVS and SSR. The proposed CSC4D-CSVS matches the performance of the SSR method at high noise levels, and outperforms it and medium and low noise levels.

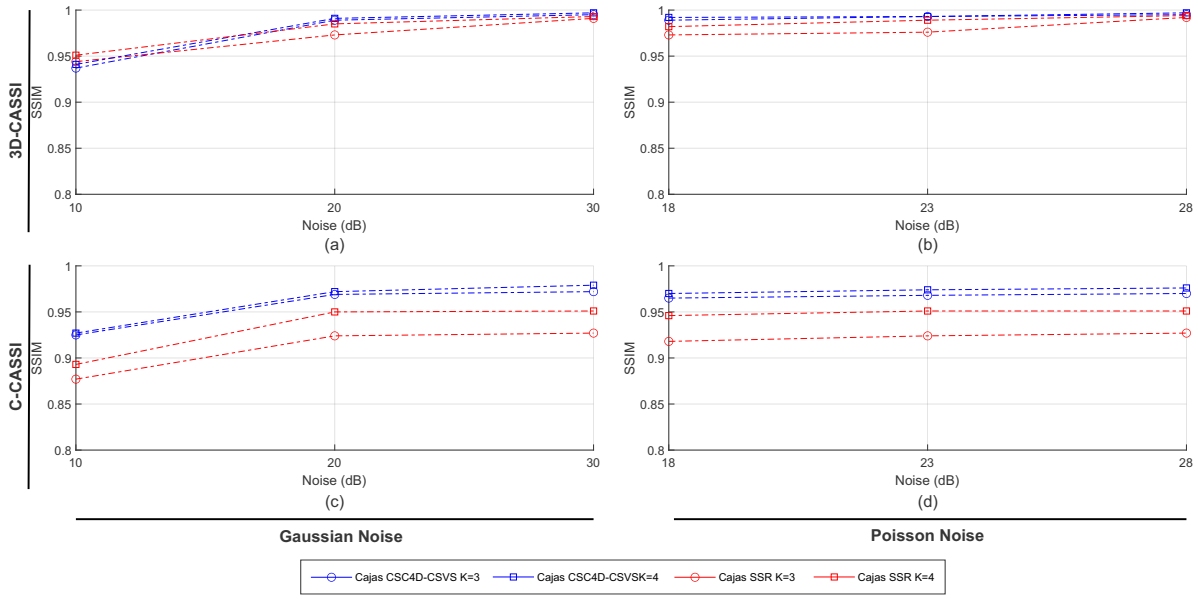


Figure 48. SSIM comparison of the simulated results at recovering the Chivas dataset for two CSVS techniques and two noise types, for both the proposed CSC4D-CSVS and SSR. The proposed CSC4D-CSVS matches the performance of the SSR method at high noise levels, and outperforms it and medium and low noise levels.

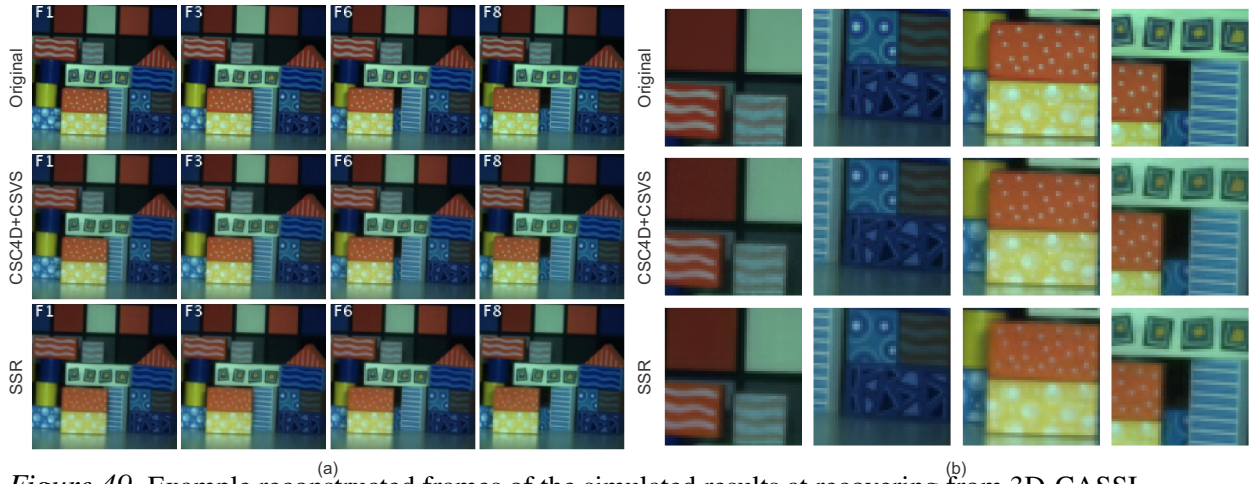


Figure 49. Example reconstructed frames of the simulated results at recovering from 3D-CASSI compressed measurements, at $K = 4$ and 20dB SNR Gauss noise, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Cajas and (b) and some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas, reduction of artifacts, even in presence of noise. I.e., note the stripes and dots definition in all close up panels

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	10	3	PSNR	32,27	**33,31**
			SSIM	0,815	**0,875**
		4	PSNR	32,39	**33,48**
			SSIM	0,822	**0,896**
	20	3	PSNR	**35,67**	34,99
			SSIM	**0,941**	0,928
		4	PSNR	**35,87**	35,24
			SSIM	**0,947**	0,914
	30	3	PSNR	**36,22**	35,31
			SSIM	**0,957**	0,928
		4	PSNR	**36,46**	35,48
			SSIM	**0,963**	0,944
Chiva	10	3	PSNR	41,41	**42,41**
			SSIM	0,937	**0,944**
		4	PSNR	41,84	**42,50**
			SSIM	0,941	**0,951**
	20	3	PSNR	**48,90**	47,83
			SSIM	**0,989**	0,973
		4	PSNR	**49,78**	48,18
			SSIM	**0,991**	0,985
	30	3	PSNR	**51,43**	50,65
			SSIM	**0,995**	0,991
		4	PSNR	**53,92**	52,70
			SSIM	**0,997**	0,993

Table 13
Mean PSNR and SSIM of the reconstructed datasets using the 3D-CASSI in presence of Gaussian white noise, for both the proposed CSC4D-CSVS and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	18	3	PSNR	**35,33**	35,00
			SSIM	**0,945**	0,927
		4	PSNR	**35,53**	35,09
			SSIM	**0,951**	0,935
	23	3	PSNR	**36,56**	35,29
			SSIM	**0,964**	0,927
		4	PSNR	**36,80**	35,31
			SSIM	**0,966**	0,936
	28	3	PSNR	**36,60**	35,29
			SSIM	**0,967**	0,934
		4	PSNR	**36,61**	35,31
			SSIM	**0,971**	0,940
Chiva	18	3	PSNR	**47,67**	47,53
			SSIM	**0,989**	0,973
		4	PSNR	**48,82**	47,73
			SSIM	**0,992**	0,982
	23	3	PSNR	**49,92**	48,24
			SSIM	**0,993**	0,976
		4	PSNR	**51,54**	48,60
			SSIM	**0,996**	0,989
	28	3	PSNR	**51,27**	49,85
			SSIM	**0,995**	0,992
		4	PSNR	**53,90**	51,71
			SSIM	**0,997**	0,994

Table 14
Mean PSNR and SSIM of the reconstructed datasets using the 3D-CASSI in presence of Poisson noise, for both the proposed CSC4D-CSVS and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	10	3	PSNR	31,60	**31,83**
			SSIM	**0,817**	0,802
		4	PSNR	31,88	**32,27**
			SSIM	**0,835**	0,816
	20	3	PSNR	**33,67**	33,45
			SSIM	**0,918**	0,886
		4	PSNR	**34,20**	33,51
			SSIM	**0,939**	0,918
	30	3	PSNR	**36,19**	34,30
			SSIM	**0,930**	0,904
		4	PSNR	**36,41**	35,43
			SSIM	**0,949**	0,919
Chiva	10	3	PSNR	**40,89**	37,81
			SSIM	**0,925**	0,877
		4	PSNR	**40,93**	38,32
			SSIM	**0,927**	0,893
	20	3	PSNR	**44,47**	39,65
			SSIM	**0,969**	0,924
		4	PSNR	**45,18**	41,51
			SSIM	**0,972**	0,950
	30	3	PSNR	**45,47**	39,80
			SSIM	**0,972**	0,927
		4	PSNR	**46,50**	41,71
			SSIM	**0,979**	0,951

Table 15
Mean PSNR and SSIM of the reconstructed datasets using the C-CASSI in presence of Gaussian white noise, for both the proposed CSC4D-CSVS and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	18	3	PSNR	32,52	**32,95**
			SSIM	0,819	**0,879**
		4	PSNR	33,05	**33,17**
			SSIM	0,842	**0,916**
	23	3	PSNR	**33,80**	33,52
			SSIM	**0,927**	0,898
		4	PSNR	**34,39**	34,10
			SSIM	**0,951**	0,918
	28	3	PSNR	**36,26**	34,28
			SSIM	**0,930**	0,902
		4	PSNR	**36,46**	35,40
			SSIM	**0,950**	0,943
Chiva	18	3	PSNR	**44,29**	39,21
			SSIM	**0,965**	0,918
		4	PSNR	**44,77**	41,22
			SSIM	**0,970**	0,946
	23	3	PSNR	**44,94**	39,67
			SSIM	**0,968**	0,924
		4	PSNR	**45,58**	41,23
			SSIM	**0,974**	0,951
	28	3	PSNR	**45,19**	39,73
			SSIM	**0,970**	0,927
		4	PSNR	**45,93**	41,58
			SSIM	**0,976**	0,951

Table 16
Mean PSNR and SSIM of the reconstructed datasets using the C-CASSI in presence of Poisson noise, for both the proposed CSC4D-CSVS and SSR framework and two compression ratios.

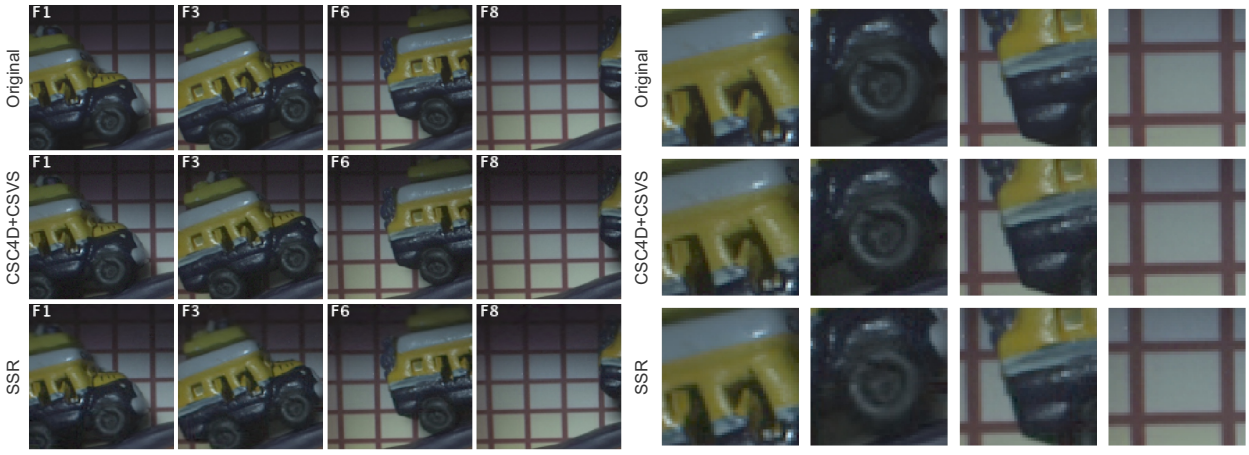


Figure 50. Example reconstructed frames of the simulated results at recovering from 3D-CASSI compressed measurements, at $K = 4$ and 20dB SNR Gauss noise, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Chiva and (b) and some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas and reduction of artifacts, even in presence of noise. I.e., note the body and tire decals in all close up panels.

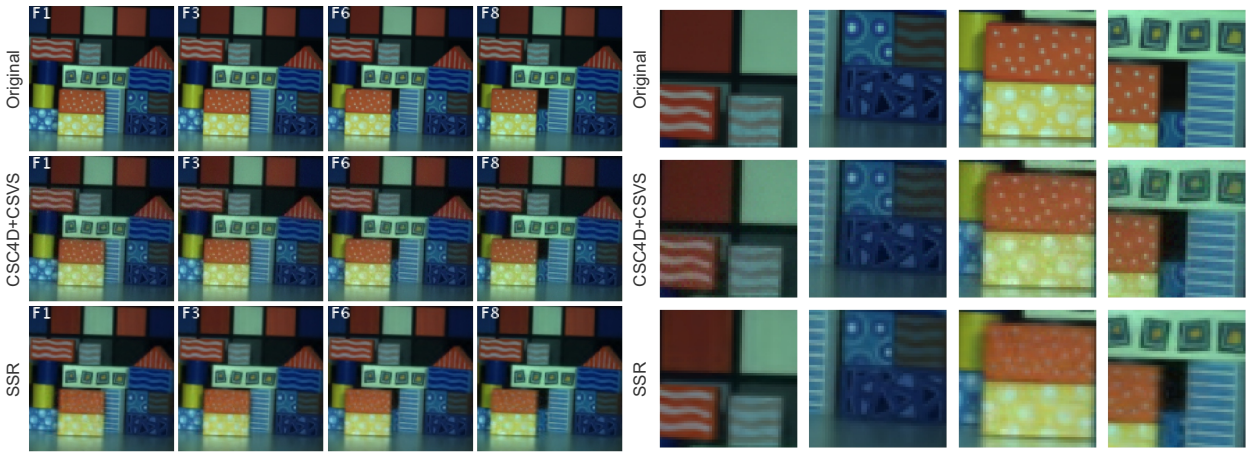


Figure 51. Example reconstructed frames of the simulated results at recovering from C-CASSI compressed measurements, at $K = 4$ and 23dB SNR Poisson noise, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Cajas and (b) some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas, reduction of artifacts, even in presence of noise. I.e., note the triangular shapes in the second close up and the orange areas in the third and fourth close up.

CASSI architecture. The results of the performance tests are shown in Figures 45 to 48 and Tables 13 to 16.

Just as in the CSC3D framework, the CSC4D model down-performs in the pressence of excessive noise (10dB Gaussian and 18dB Poisson), matching the performance of the SSR model. However, it outperforms the state-of-the-art model in medium and low noise levels, at all compression ratios, in up to 4dB. On the other hand, the SSIM values for the CSC4D model are higher than SSR's SSIM values at almost all compression and noise levels. This means that, although the CSC4D fails to recover all the SV's spatial-spectral-temporal features in presence of noise, the recovered SV has sharper and more defined borders than the SSR model.

Figures 49 to 52 show some example reconstructed frames as false RGB, recovered from compressed measurements taken with $K = 4$ and 20dB SNR Gauss and 23dB SNR Poisson noise, respectively. Note the overall quality of the recovered individual spectral frames, and the border sharpness, of the CSC4D model when compared to the SSR model.

Algorithm Convergence. To illustrate the good convergence of Algorithm 3, a typical evolution of cost function (151) as a function of the iteration number was selected. Considering that the indicator function in (152) only has values 0 or ∞, it is not included in the estimation of the cost function. The high-frequencies version of the Cajas dataset was used for this experiment, and the compressive measurements were simulated with the 3D-CASSI system, with 20dB SNR white

Gaussian noise, and $K = 4$. Fig. 53 confirms the convergence of the algorithm to a critical point of the objective function.

To analyze the sensitivity to initialization, we ran the proposed algorithm with 100 different random initializations for the collection of dictionary elements, and all-zeros coefficient maps. The histogram of the corresponding values of the objective function is shown in Fig. 54, showing a single mode, confirming the convexity of the proposed algorithm. The mode of the objective function histogram corresponds to a PSNR of 35.61dB.

Conclusions

The proposed 4D framework results from increasing the dimensionality of the CSC3D operator from a 3D cyclic convolution to a 4D cyclic convolution. This additional dimension takes into account the temporal correlation within SVs, without additional modifications. For example, the SSR model needs to carefully choose the more appropriate representation basis for the additional temporal dimension, while the CSC4D includes the temporal correlations as another convolution operation.

Being a direct scalation of the CSC3D operator, the CSC4D operator can be also represented as a matrix-vector multiplication, and included within a ℓ_2-linear restriction convex formulation. The formulation fot CSC3D and CSC4D are quite similar, being the only difference the dimensions of the convolutional equivalent operators. Now, despite these similitudes, the CSC3D's optimized

numerical routines can be modified and scaled in order to solve CSC4D with feasible resources.

The proposed CSC4D is able to sparsely represent SVs, improving on the SSR framework by up to 20dB. The proposed CSC4D framework can be also used within a SR and CSVS formulation, also improving on the SSR framework in both PSNR and SSIM metrics. However, as with the CSC3D framework, the CSC4D is also sensible to the band shifting of CSVS techniques as C-CASSI and must rely on a side-information scheme.

7. Dissertation Conclusions

ND-Convolutional Sparse Dictionary Representation

The CSDR proposed by Papyan *et. al.* for representing 1D signals can be generalized as the N-dimensional operator

$$\mathscr{S} = \sum_{m=1}^{M_d} \mathscr{D}_m \overset{N}{*} \mathscr{X}_m + \boldsymbol{\Omega}. \tag{154}$$

In order to obtain both the overcomplete set of convolutional elements and sparse coefficient maps we can transform Eq. (154) into a matrix-vector multiplication and solve the dual linear optimization scheme

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{D}}\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1, \tag{155}$$

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{X}}\mathbf{d} - \mathbf{s} \right\|_2^2 + \iota_{\mathscr{C}_P}(\mathbf{d}). \tag{156}$$

It is worth noting that Eqs. (155) (156) have no consideration for the dimension of the signal of interest $\mathbf{s}$. This dual lineal optimization scheme can be adapted to any N-dimension by only modifying the size of operators $\bar{\mathbf{D}}$, $\bar{\mathbf{X}}$ and arrays $\mathbf{x}$ and $\mathbf{d}$.

This doctoral dissertation takes the formulation proposed in Eqs. (155) (156) further from the 2D case to the 3D dimension, and also to the 4D dimension, without proposing any changes. On the other hand, this dissertation proposes the necessary adjustments for the solution of the 3D

and 4D convolutional operations, besides the corresponding reorganization of the ND-dimensional elements. The proposed solution and reorganization algorithms are necessary in order to keep the numerical complexity within reasonable and feasible limits.

There are advantages in using a signal-specific convolutional sparse representation over a traditional basis. First, we have the intrinsic robustness to translation and deformation of the convolutional operator; second, the signal specificity allows for higher detail in the reconstructed image, with fewer artifacts and sharper borders; third, and this is the most interesting one, we only need to increase the dimension of the convolutional operator in order to represent higher order signals, instead of creating custom basis with Kronecker products. However, it comes with the disadvantage of a dual optimization formulation instead of a single one. Nonetheless, the proposed numerical optimization routines keep this associated cost within feasible limits.

ND-CSC in Compressed Sensing

Proving the flexibility of the ND-convolutional sparse representation, and the robustness of the dual lineal optimization framework, is a result worth mentioning by itself. However, the main contribution of this doctoral dissertation is the application of the ND-convolutional framework in CSI and CSVS, replacing state-of-the-art represen- tation frameworks. Even more, this can be achieved without changing the proposed formulation as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{D}}\mathbf{x} - \mathbf{y} \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1 , \tag{157}$$

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{2} \left\| \mathbf{H}\bar{\mathbf{X}}\mathbf{d} - \mathbf{y} \right\|_2^2 + \iota_{\mathscr{C}_Z}(\mathbf{d}). \tag{158}$$

In this regard, this doctoral dissertation presents the following results:

1. It is possible to replace the state-of-the-art Kronecker basis $\mathbf{\Psi}$ with the CSC formulation for recovering compressed sensed spectral images without altering the formulation. However, this change a dual lineal optimization formulation.

2. The proposed CSC3D operator is a simple convolution operator, easily scalable, not a custom basis created from simpler basis, i.e. the Kronecker Basis.

3. The proposed CSC4D is a logical scalation of the CSC3D operator, while the Kronecker basis for SVs comes after trial and error of different basis.

4. The synthesis signal-based convolutional representation improves the performance of the Kronecker basis $\mathbf{\Psi}$ at recovering compressed sensed SIs and SVS, even in the presence of noise.

5. Although the numerical routines for CSC3D-CSI and CSC4D-CSVS are more than just an scalation of CSC3D's and CSC4D's, both set of routines follow the same logical formulation making them modular and reusable to certain extent.

However, there are certain observations when using the CSC framework within a compressive sensing formulation, such as

1. The CSC formulation work with the high frequency components of a signal, which is why we must perform a high-pass filtering prior to the CSC framework.

2. The CSC formulation seems to perform poorly in the presence of band shifting elements (i.e. C-CASSI CSI architecture), creating artifacts and reducing the performance.

3. Although the CSC framework outperforms the Kronecker basis in the presence of noise, it is worth noting that with higher noise levels the averaging nature of the convolutional operation down-performs compared to the Kronecer basis.

Future Work

The proposed CSC3D and CSC4D frameworks were barely tested beyond its applicability in CSI and CSVS, respectively. For example, the CSC4D framework was tested for spatial SR, but not the CSC3D. This leaves a whole field of probable applications such as image fusion, spectral SR, temporal SR and similar ones.

On the other hand, we evaluated the creation of an overcomplete set of convolutional dictionary elements and sparse coefficient maps for a signal of interest. We didn't evaluate the scenario of creating a library of pre-learned dictionary collections and its use with new signals. This scenario is valid for all ND-CSC, CSI and CSVS applications.

Finally, one of the biggest issues of the proposed CSC frameworks is the interaction with band-shifting compressed sensing architectures. Although this doctoral dissertation proposes an

ad-hoc solution in the form of the side-information scheme, this problem is worth studying in depth.

Bibliography

Aharon, M., Elad, M., and Bruckstein, A. (2006). K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54:4311–4322.

Arce, G. R., Brady, D. J., Carin, L., Arguello, H., and Kittle, D. S. (2014). Compressive coded aperture spectral imaging: An introduction. *IEEE Signal Processing Magazine*, 31:105–115.

Arguello, H. and Arce, G. R. (2013). Rank minimization code aperture design for spectrally selective compressive imaging. *IEEE Transactions on Image Processing*, 22:941–954.

Arjoune, Y., Kaabouch, N., Ghazi, H. E., and Tamtaoui, A. (2017). Compressive sensing: Performance comparison of sparse recovery algorithms. *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1–7.

Babacan, S. D., Molina, R., and Katsaggelos, A. K. (2010). Bayesian compressive sensing using laplace priors. *IEEE Transactions on Image Processing*, 19:53–63.

Bacca, J., Correa, C., and Arguello, H. (2019). Noniterative hyperspectral image reconstruction from compressive fused measurements. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12:1231–1239.

Banerjee, A., Burlina, P., and Broadwater, J. (2009). Hyperspectral video for illumination-invariant tracking.

Barajas-Solano, C., Garcia, H., and Arguello, H. (2019a). Convolutional basis pursuit denoising of spectral images using a tri-dimensional sparse representation. pages 1–5. IEEE.

Barajas-Solano, C., Ramirez, J.-M., and Arguello, H. (2019b). Convolutional sparse coding framework for compressive spectral imaging. *Journal of Visual Communication and Image Representation*, 66:1–15.

Barajas-Solano, C., Ramirez, J.-M., and Fuentes, H. A. (2020). Spectral video compression using convolutional sparse coding.

Barajas-Solano, C., Ramirez, J. M., Garcia, H., and Arguello, H. (2019c). Tridimensional convolutional sparse coding of spectral images. volume 2019.

Barducci, A., Guzzi, D., Lastri, C., Marcoionni, P., Nardino, V., and Pippi, I. (2012). Compressive sensing and hyperspectral imaging. page 105642Z.

Bioucas-Dias, J. M., MP, J., and Nascimento (2008). Hyperspectral subspace identification. *IEEE Transactions on Geoscience and Remote Sensing*, 46:2435–2445.

Blumensath, T. and Davies, M. E. (2009). Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27:265–274.

Bottou, L. (2004). Stochastic learning.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2010). Distributed optimization and

statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3:1–122.

Bristow, H. and Eriksson, A. (2013). Fast convolutional sparse coding. pages 391–398.

Bruckstein, A. M., Donoho, D. L., and Elad, M. (2009). From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51:34–81.

Buttingsrud, B. and Alsberg, B. K. (2006). Superresolution of hyperspectral images. *Chemometrics and Intelligent Laboratory Systems*, 84:62–68.

Candes, E. and Romberg, J. (2007). Sparsity and incoherence in compressive sampling. *Inverse Problems*, 23:969–985.

Candes, E. J. and Wakin, M. B. (2008). An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25:21–30.

Cao, X., Yue, T., Lin, X., Lin, S., Yuan, X., Dai, Q., Carin, L., and Brady, D. J. (2016). Computational snapshot multispectral cameras: Toward dynamic capture of the spectral world. *IEEE Signal Process. Mag.*, 33:95–108.

Chang, H., Spellman, P., Division, L. S., and Berkeley, L. (2014). Classification of histology sections via multispectral convolutional sparse. pages 3081–3088.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Review*, 43:129–159.

Chen, Y. and Su, J. (2017). Sparse embedded dictionary learning on face recognition. *Pattern Recognition*, 64:51–59.

Cheng, S. Y., Park, S., and Trivedi, M. M. (2007). Multi-spectral and multiperspective video arrays for driver body tracking and activity analysis. *Comput. Vis. Image Underst.*, 106:245–257.

Correa, C. V., Arguello, H., and Arce, G. R. (2014). Compressive spectral imaging with colored-patterned detectors. pages 7789–7793.

Correa, C. V., Arguello, H., and Arce, G. R. (2015). Snapshot colored compressive spectral imager. *JOSA A*, 32:1754–1763.

Correa, C. V., Hinojosa, C. A., Arce, G. R., and Arguello, H. (2016). Multiple snapshot colored compressive spectral imager. *Optical Engineering*, 56:041309:1–041309:10.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16:2080–2095.

del Pais Vasco, U. (2012). Hyperspectral remote sensing scenes.

Du, B., Zhang, M., Zhang, L., Hu, R., and Tao, D. (2017). Pltd: Patch-based low-rank tensor decomposition for hyperspectral images. *IEEE Transactions on Multimedia*, 19:67–79.

Duarte, M. F. and Baraniuk, R. G. (2012). Kronecker compressive sensing. *IEEE Transactions on Image Processing*, 21:494–504.

Elad, M., Figueiredo, M. A. T., and Ma, Y. (2010). Proceedings of the ieee â€“ special issue on applications of sparse representation and compressive sensing on the role of sparse and redundant representations in image processing. *PROCEEDINGS OF THE IEEE â€“ SPECIAL ISSUE ON APPLICATIONS OF SPARSE REPRESENTATION AND COMPRESSIVE SENSING*, 98:972–982.

Engan, K., Aase, S. O., and Husoy, J. H. (1999). Method of optimal directions for frame design. pages 2443–2446.

Foucart, S. (2012). Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants.

Fowler, J. E. (2014). Compressive pushbroom and whiskbroom sensing for hyperspectral remote-sensing imaging. pages 684–688.

Galvis, L., Lau, D., Ma, X., Arguello, H., and Arce, G. R. (2017). Coded aperture design in compressive spectral imaging based on side information. *Applied Optics*, 56:6332–6340.

Gamba, P. (2004). A collection of data for urban area characterization.

Gao, Y., Wang, X., Cheng, Y., and Wang, Z. J. (2015). Dimensionality reduction for hyperspectral data based on class-aware tensor neighborhood graph and patch alignment. *IEEE Trans. Neural Netw. Learn. Syst.*, 26:1582–1593.

Garg, R. and Khandekar, R. (2009). Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. pages 337–344.

Gat, N. (2000). Imaging spectroscopy using tunable filters. pages 50–64.

Gehm, M., John, R., Brady, D., Willett, R., and Schulz, T. (2007). Single-shot compressive spectral imaging with a dual-disperser architecture. *Opt. express*, 15:14013–14027.

Gelvez, T., Rueda, H., and Arguello, H. (2017). Joint sparse and low rank recovery algorithm for compressive hyperspectral imaging. *Applied Optics*, 56:6785–6795.

Gersho, A. (1992). *Vector Quantization and Signal Compression*. Kluwer Academic Publishers-Consultants Bureau.

Ghamisi, P., Yokoya, N., Li, J., Liao, W., Liu, S., Plaza, J., Rasti, B., and Plaza, A. (2017). Advances in hyperspectral image and signal processing. *IEEE Geoscience and remote sensing magazine*, 5:37–78.

Greenberg, J., Krishnamurthy, K., and Brady, D. (2014). Compressive single-pixel snapshot x-ray diffraction imaging. *Optics letters*, 39:111–114.

Gu, S., Zuo, W., Xie, Q., Meng, D., Feng, X., and Zhang, L. (2015). Convolutional sparse coding for image super-resolution. pages 1823–1831.

Henderson, H. V. and Searle, S. R. (1981). On deriving the inverse of a sum of matrices. *Siam Review*, 23:53–60.

Jafari, M. and Plumbley, M. (2011). Fast dictionary learning for sparse representations of speech signals. *selected topics in signal processing, IEEE*, 5:1025–1031.

Ji, S., Xue, Y., and Carin, L. (2008). Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56:2346–2356.

Kwan, C., Choi, J. H., Chan, S., Zhou, J., and Budavari, B. (2017). Resolution enhancement for hyperspectral images: A super-resolution and fusion approach. pages 6180–6184.

Leitner, R., De-Biasio, M., Arnold, T., Dinh, C. V., Loog, M., and Duin, R. P. W. (2013). Multi-spectral video endoscopy system for the detection of cancerous tissue. *Pattern Recognition Letters*, 34:85–93.

Leon-Lopez, K. M., Carreno, L. V. G., and Fuentes, H. A. (2019). Temporal colored coded aperture design in compressive spectral video sensing. *IEEE Transactions on Image Processing*, 28:253–264.

Li, Y., Amari, S. I., Cichocki, A., Ho, D. W. C., and Xie, S. (2006). Underdetermined blind source separation based on sparse representation. *IEEE Trans. Signal Process*, 54:423 – 437.

Liang, H. and Li, Q. (2016). Hyperspectral imagery classification using sparse representations of convolutional neural network features. *Remote Sensing*, 8:1–16.

Lin, X., Liu, Y., Wu, J., and Dai, Q. (2014a). Spatial-spectral encoded compressive hyperspectral imaging. *ACM Transactions on Graphics*, 33:233:1–233:11.

Lin, X., Wetzstein, G., Liu, Y., and Dai, Q. (2014b). Dual-coded compressive hyperspectral imaging. *Optics letters*, 39:2044–2047.

Liu, T. and Tao, D. (2016). Classification with noisy labels by importance weighting. *IEEE Trans. Patter Anal. Mach. Intell.2*, 38:447–461.

Lu, G. and Fei, B. (2014). Medical hyperspectral imaging: A review. *J. Biomed. Opt*, 19:10901.

Mairal, J. and Bach, F. (2014). Sparse modelling for image and vision processing. *Proc. IEEE Conf. Comp. Vis. Pat. Recog.*, 8:85–283.

Mallat, S. and Zang, Z. (1993). Matching pursuits with time-frequency dictionaries. *EEE Trans. Signal Proc.*, 41:3397–3415.

Nguyen, H. V., Patel, V. M., Nasrabadi, N. M., and Chellappa, R. (2012). Sparse embedding: A framework for sparsity promoting dimensionality reduction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7577 LNCS:414–427.

Nishihara, R., Lessard, L., Recht, B., Packard, A., and Jordan, M. I. (2015). A general analysis of the convergence of admm. pages 343–352.

Papyan, V., Romano, Y., Sulam, J., and Elad, M. (2018). Theoretical foundations of deep learning via sparse representations: A multilayer sparse model and its connection to convolutional neural networks. *IEEE Signal Processing Magazine*, 35:72–89.

Papyan, V., Sulam, J., and Elad, M. (2017). Working locally thinking globally: Theoretical guarantees for convolutional sparse coding. *IEEE Transactions on Signal Processing*, 65:5687–5701.

Pati, Y. C., Rezaiifar, R., and Krishnaprasad, P. S. (1993). Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. pages 40–44.

Ramirez, A., Arguello, H., Arce, G. R., and Sadler, B. M. (2014). Spectral image classification from optimal coded-aperture compressive measurements. *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING*, 52:3299–3309.

Rockafellar, R. T. (1970). *Convex Analysis*. Princeton University Press.

Rousset, F., Ducros, N., Farina, A., Valentini, G., Dâ€™Andrea, C., and Peyrin, F. (2016). Adaptive acquisitions in biomedical optical imaging based on single pixel camera: Comparison with compressive sensing. pages 680–683.

Rueda, H., Arguello, H., and Arce, G. R. (2015). Dmd-based implementation of patterned optical filter arrays for compressive spectral imaging. *JOSA A*, 32:80–89.

Sahoo, S. K., Tang, D., and Dang, C. (2017). Single shot color imaging through scattering media using a monochromatic camera. page 353â€“ 354.

Scott, C. and Nowak, R. D. (2004). Templar: A wavelet-based framework for pattern learning and analysis. *IEEE Transactions on Signal Processing*, 52:2264–2274.

Sellar, R. G. and Boreman, G. D. (2005). Classification of imaging spectrometers for remote sensing applications. *Opt. Eng.*, 44:013602–1 â€“ 013602–2.

Shannon, C. E. (1949). Communication in the presence of noise. *Proceedings of the Institute of Radio Engineers*, 37:10–21.

Shaw, G. A. and Burke, H. K. (2003). Spectral imaging for remote sensing. *Lincoln laboratory journal*, 14:3–28.

Swamy, S., Asutkar, S. M., and Asutkar, G. M. (2017). Remote sensing hsi classification and estimation of mimetite mineral spectral signatures from isro, india. pages 1095–1099.

Tan, J., Ma, Y., Rueda, H., Baron, D., and Arce, G. R. (2016). Compressive hyperspectral imaging via approximate message passing. *IEEE Journal of Selected Topics in Signal Processing*, 10:389–401.

Tao, D., Li, X., Wu, X., and Maybank, S. J. (2007). General tensor discriminant analysis and gabor features for gait recognition. *IEEE Trans. Patter Anal. Mach. Intell.*, 29:1700–1715.

Thiagarajan, J. J., Ramamurthy, K. N., and Spanias, A. (2008). Sparse representations for pattern classification using learned dictionaries. *SGAI*, pages 33–45.

Tropp, J. A., Gilbert, A. C., and Strauss, M. J. (2006). Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing*, 86:572–588.

Van-Nguyen, H., Banerjee, A., and Chellappa, R. (2010). Tracking via object reflectance using a hyperspectral video camera. pages 44–51.

Vargas, E., Arguello, H., and Tourneret, J.-Y. (2019). Spectral image fusion from compressive

measurements using spectral unmixing and a sparse representation of abundance maps. *IEEE Transactions on Geoscience and Remote Sensing*, 57:5043–5053.

Velasco-Forero, S. and Angulo, J. (2013). Classification of hyperspectral images by tensor modelling and additive morphological decomposition. *Pattern Recognition*, 46:566–577.

Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2018). Scalable online convolutional sparse coding. *IEEE Transactions on Image Processing*, 27:4850–4859.

Wang, Y. W., Reder, N. P., Kang, S., Glaser, A. K., and Liu, J. T. C. (2017). Multiplexed optical imaging of tumor-directed nanoparticles: A review of imaging systems and approaches. *Nanotheranostics*, 1:369–388.

Wang, Z., Nasrabadi, N. M., and Huang, T. S. (2014). Spatial-spectral classification of hyperspectral images using discriminative dictionary designed by learning vector quantization. *IEEE Transactions on Geoscience and Remote Sensing*, 52:4808–4822.

Wohlberg, B. (2014). Efficient convolutional sparse coding. pages 7173–7177.

Wohlberg, B. (2016a). Boundary handling for convolutional sparse representations. pages 1833–1837.

Wohlberg, B. (2016b). Convolutional sparse representation of color images. pages 57–60.

Wohlberg, B. (2016c). Efficient algorithms for convolutional sparse representations. *IEEE Transactions on Image Processing*, 25:301–315.

Wright, J., Ma, Y., Mairal, J., Sapiro, G., Huang, T. S., and Yan, S. (2010). Sparse representation for computer vision and pattern recognition. *Proc. IEEE*, 98:1031–1044.

Yang, J., Wright, J., Huang, T., and Ma, Y. (2008). Image super-resolution as sparse representation of raw image patches. pages 1–8.

Yasuma, F., Mitsunaga, T., Iso, D., and Nayar, S. K. (2008). Generalized assorted pixel camera: Post-capture control of resolution, dynamic range and spectrum.

Yi, D., Kong, L., Wang, F., Liu, F., Sprigle, S., and Adibi, A. (2011). Instrument an off-shelf ccd imaging sensor into a handheld multispectral video camera. *Photonics Technology Letters, IEEE*, 23:606–608.

Yu, S., Jia, S., and Xu, C. (2017). Convolutional neural networks for hyperspectral image classification. *Neurocomputing*, 219:88–98.

Yuan, L., Woodard, A., Ji, S., Jiang, Y., Zhou, Z. H., Kumar, S., and Ye, J. (2012). Learning sparse representations for fruit-fly gene expression pattern image annotation and retrieval. *BMC Bioinformatics*, 13:1–15.

Yuan, X., han Tsai, T., Zhu, R., Llull, P., Brady, D., and Carin, L. (2015). Compressive hyperspectral imaging with side information. *IEEE Journal of Selected Topics in Signal Processing*, 9:964–976.

Zuzak, K. J., Naik, S. C., Alexandrakis, G., Hawkins, D., Behbehani, K., and Livingston, E. (2013).

Intraoperative bile duct visualization using nearinfrared hyperspectral video imaging. pages 145–150.

Appendices

Appendix A. Efficient Solution Of The $\bar{N}$-Dimensional Convolutional Sparse Coding Representation In The Fourier Domain

Let there be a collection of convolutional dictionary elements $\{\mathscr{D}_m,\ m=1,...,M_d \mid \mathscr{D}_m \in \mathbb{R}^{d_1\times...\times d_{\bar{N}}}\}$, and a collection of sparse coeffiecient maps $\{\mathscr{X}_m,\ m=1,...,M_d \mid \mathscr{X}_m \in \mathbb{R}^{L_1\times...\times L_{\bar{N}}}\}$. Each element $\mathscr{D}_m$ and $\mathscr{X}_m$ has dimensionality $\bar{N}$, and $d_i \acute{} L_i$. Then, a $\bar{N}$-dimensional signal $\mathscr{S} \in \mathbb{R}^{L_1\times...\times L_{\bar{N}}}$ can be represented as

$$\mathscr{S} = \sum_{m=1}^{M_d} \mathscr{D}_m \overset{\bar{N}}{*} \mathscr{X}_m + \boldsymbol{\Omega}, \tag{159}$$

The sum of cyclic convolutions in Eq. (159) can be solved effectively in the Fourier domain as

$$\mathscr{S} = \mathscr{F}_{\bar{N}}^{-1}\left\{\sum_{m=1}^{M_d} \mathscr{F}_{\bar{N}}\{\mathscr{D}_m\} \odot \mathscr{F}_{\bar{N}}\{\mathscr{X}_m\}\right\}. \tag{160}$$

The sum of Hadamard products Eq. (160) can be simplified as a matrix-vector product by performing the following rearrangements:

- obtain the collections $\{\hat{\mathscr{D}}_m\}$ and $\{\hat{\mathscr{X}}_m\}$ from $\{\mathscr{D}_m\}$ and $\{\mathscr{X}_m\}$ respectively, following Algorithm 10.
- create matrix $\hat{\bar{\mathbf{D}}} = \mathbb{C}^{\bar{M}\times\bar{M}M_d}$ following Algorithm 6.
- create vector $\hat{\mathbf{x}} = \in \mathbb{C}^{\bar{M}M_d}$ following Algorithm 4.

Finally we obtain the vectorization of the Fourier transform of the signal of interest as

$$\hat{\mathbf{s}} = \text{vec}\left(\sum_{m=1}^{M_d} \mathscr{F}_{\bar{N}}\{\mathscr{D}_m\} \odot \mathscr{F}_{\bar{N}}\{\mathscr{X}_m\}\right) = \hat{\bar{\mathbf{D}}}\hat{\mathbf{x}} \in \mathbb{C}^{\bar{M}}. \tag{161}$$

In order to recover $\mathscr{S} \in \mathbb{R}^{L_1 \times \ldots \times L_{\bar{M}}}$ from $\hat{\mathbf{s}} \in \mathbb{C}^{\bar{M}}$ we must:

- fold $\hat{\mathbf{s}}$ into $\hat{\mathscr{S}} \in \mathbb{C}^{L_1 \times \ldots \times L_{\bar{M}}}$ following Algorithm 7
- obtain the inverse Fourier transform following Algorithm 11.

Appendix B. Woodbury matrix identity

Consider the linear system

$$\mathbf{A}\mathbf{x} = \left(\alpha\mathbf{I} + \mathbf{H}^T\mathbf{H}\right)\mathbf{x} = \mathbf{b}, \tag{162}$$

with $\mathbf{I} \in \mathbb{R}^{N\times N}$, $\mathbf{H} \in \mathbb{R}^{K\times N}$, $K < N$, $\mathbf{b}$ and $\mathbf{x} \in \mathbb{R}^N$. The inversion of matrix $\mathbf{A}$ in Eq. 162 can be solved using the Woodbury formula (Henderson and Searle, 1981)

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U}\right)^{-1}\mathbf{VA}^{-1}, \tag{163}$$

with $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{U} \in \mathbb{R}^{N\times K}$, $\mathbf{V} \in \mathbb{R}^{K\times N}$ and $\mathbf{C} \in \mathbb{R}^{K\times K}$. We can rewrite Eq. 162 in the form of Eq. 163 by making $\mathbf{C} = \mathbf{I}$ and factorizing α as

$$\mathbf{x} = \frac{1}{\alpha}\left[\mathbf{b} - \mathbf{H}^T\left(\alpha\mathbf{I} + \mathbf{HH}^T\right)^{-1}\mathbf{Hb}\right]. \tag{164}$$

In our problem the matrix $\mathbf{HH}^T$ is diagonal, which simplifies the complexity for solving $(\alpha\mathbf{I} + \mathbf{HH}^T)^{-1}$, and overall Eq. 164, to $\mathcal{O}(K^2N)$.

Appendix C. Optimal solution for a concatenated system

Consider the linear system

$$\left(\alpha\mathbf{I}+\mathbf{A}^H\mathbf{A}\right)\mathbf{x}=\mathbf{b}, \tag{165}$$

with $\mathbf{I}\in\mathbb{R}^{\bar{N}\bar{M}\times\bar{N}\bar{M}}$, $\mathbf{A}=[\mathbf{A}_1...\mathbf{A}_{\bar{M}}]\in\mathbb{C}^{\bar{N}\times\bar{N}\bar{M}}$, $\mathbf{A}_m\in\mathbb{C}^{\bar{N}\times\bar{N}}$ is a diagonal matrix, $m=1,...,\bar{M}$, $\mathbf{A}^H$ is the conjugated transpose of $\mathbf{A}$, and both $\mathbf{x}$ and $\mathbf{b}\in\mathbb{C}^{\bar{N}\bar{M}}$. Applying the Woodbury formula in 163 and factorizing α, Eq. 165 can be rewritten as

$$\mathbf{x}=\frac{1}{\alpha}\left[\mathbf{b}-\mathbf{A}^H\left(\alpha\mathbf{I}+\mathbf{A}\mathbf{A}^H\right)^{-1}\mathbf{A}\mathbf{b}\right]. \tag{166}$$

The product $\mathbf{A}\mathbf{A}^H$ is always a diagonal matrix with complexity $\mathscr{O}(\bar{N}^3\bar{M})$. This product, and its inverse, determinate the overall complexity of solving Eq. 166. However, we can reduce this complexity by profiting on the concatenated diagonal structure of $\mathbf{A}$.

Below we present our three step solution for the optimal solution of the inverse $\mathbf{E}_1=\left(\alpha\mathbf{I}+\mathbf{A}\mathbf{A}^H\right)^{-1}$, the right hand product $\mathbf{E}_2=\mathbf{E}_1\mathbf{A}\mathbf{b}$ and the left hand product $\mathbf{E}_3=\mathbf{A}^H\mathbf{E}_2$. Each step consists of a series of numerical rearrangements in order to reduce the overall complexity, while considering that the rearrangements themselves have no numerical cost.

Inverse $\mathbf{E}_1=\left(\alpha\mathbf{I}+\mathbf{A}\mathbf{A}^H\right)^{-1}$. $\mathbf{A}=[\mathbf{A}_1...\mathbf{A}_{\bar{M}}]$ is a concatenation of diagonal matrices $\mathbf{A}_m$. In essence, only $\frac{1}{\bar{N}}$-th of $\mathbf{A}$ is non-zero. Then we begin by eliminating all the zero-elements in $\mathbf{A}$.

- Lets rearrange $\mathbf{A} \in \mathbb{C}^{\bar{N}\times\bar{N}\bar{M}}$ into $\dot{\mathbf{A}} = [\dot{\mathbf{a}}_1 ... \dot{\mathbf{a}}_{\bar{M}}] \in \mathbb{C}^{\bar{N}\times\bar{M}}$, where each column $\dot{\mathbf{a}}_m = \mathsf{vec}(\mathbf{A}_m) \in \mathbb{C}^{\bar{N}}$, and $\mathsf{vec}(.)$ vectorize a diagonal matrix as its main diagonal.

- Now, solve $\mathbf{c} = \mathsf{sum}(\dot{\mathbf{A}} \odot \mathsf{conj}(\dot{\mathbf{A}})) \in \mathbb{C}^{\bar{N}}$, where $\mathsf{sum}(.)$ denotes the row-wise sum, $\odot$ denotes the Hadamard product and $\mathsf{conj}(.)$ denotes the element-wise conjugate.

- Finally, solve $\mathbf{d} \in \mathbb{C}^{\bar{N}}$ where each $d_i = 1/(\alpha + c_i)$. The diagonal matrix $\mathbf{D} = \mathsf{diag}(\mathbf{d}) \in \mathbb{C}^{\bar{N}\times\bar{N}}$ is equal to $(\alpha\mathbf{I} + \mathbf{A}\mathbf{A}^H)^{-1}$.

The rearrangements and operations explained above solves the matrix inversion with a fraction of memory space and computing time. The complexity of this first step is given by the Hadamard product as $\mathscr{O}(\bar{M}\bar{N})$.

Right product $\mathbf{E}_2 = \mathbf{E}_1\mathbf{A}\mathbf{b}$**.** The same dimensions rearrange strategy can be used to solve the product at the right hand of the inverse.

- Considering $\mathbf{e} = \mathbf{A}\mathbf{b} \in \mathbb{C}^{\bar{N}}$, fold the vector $\mathbf{b} \in \mathbb{C}^{\bar{N}\bar{M}}$ into the matrix $\dot{\mathbf{B}} \in \mathbb{C}^{\bar{N}\times\bar{M}}$.

- Solve $\mathbf{e} = \mathsf{sum}(\dot{\mathbf{A}} \odot \dot{\mathbf{B}})$.

- Finally, we can solve $\mathbf{f} = (\alpha\mathbf{I} + \mathbf{A}\mathbf{A}^H)^{-1}\mathbf{A}\mathbf{b} \in \mathbb{C}^N$ as $\mathbf{f} = \mathbf{d} \odot \mathbf{e}$.

Again, the complexity of the second step is given by the Hadamard product as $\mathscr{O}(\bar{N}\bar{M})$.

Left product $\mathbf{E}_3 = \mathbf{A}^H\mathbf{E}_2$**.** Finally, we can solve the product at the left hand of the inverse. First, consider $\mathbf{g} = \mathbf{A}^H(\alpha\mathbf{I} + \mathbf{A}\mathbf{A}^H)^{-1}\mathbf{A}\mathbf{b} \in \mathbb{C}^{\bar{N}\bar{M}}$ and $\dot{\mathbf{G}} = [\dot{\mathbf{g}}_1 ... \dot{\mathbf{g}}_M]$.

- Fold $\dot{\mathbf{A}}$ into the 3D array $\bar{\mathbf{A}} \in \mathbb{C}^{\bar{N}\times 1\times \bar{M}}$, with each $\mathbb{C}^{\bar{N}\times 1}$ layer as $\dot{\mathbf{a}}_m$.
- Solve $\bar{\mathbf{G}} \in \mathbb{C}^{\bar{N}\times 1\times \bar{M}}$ per layer as $\dot{\mathbf{g}}_m = \mathsf{conj}(\dot{\mathbf{a}}_m)\odot\mathbf{f} \in \mathbb{C}^{\bar{N}}$.
- Fold $\bar{\mathbf{G}}$ into $\dot{\mathbf{G}} = [\dot{\mathbf{g}}_1 ... \dot{\mathbf{g}}_{\bar{M}}] \in \mathbb{C}^{\bar{N}\times\bar{M}}$.

Finally the solution is given by $\mathbf{g} = \mathsf{vec}(\dot{\mathbf{G}}) \in \mathbb{C}^{\bar{N}\bar{M}}$. The complexity of the third step is given by the Hadamard product as $\mathscr{O}(\bar{N}\bar{M})$.

In conclusion, the concatenated diagonal structure of matrix $\mathbf{A}$ allows to rearrange it down to a $\bar{N}^{\text{th}}$ of its original size. This way the matrix products can be simplified to Hadamard products and row-wise sums, reducing the computational time.

Appendix D. Dimension Adjustments for the 4D case

As stated in Appendix 3, the lineal problem

$$\left(\alpha\mathbf{I}+\mathbf{A}^{H}\mathbf{A}\right)\mathbf{x}=\mathbf{b}, \tag{167}$$

has solution

$$\mathbf{x}=\frac{1}{\alpha}\left[\mathbf{b}-\mathbf{A}^{H}\left(\alpha\mathbf{I}+\mathbf{A}\mathbf{A}^{H}\right)^{-1}\mathbf{A}\mathbf{b}\right], \tag{168}$$

with $\mathbf{I}\in\mathbb{C}^{\bar{N}\bar{M}\times\bar{N}\bar{M}}$, $\mathbf{A}=[\mathbf{A}_1...\mathbf{A}_{\bar{M}}]\in\mathbb{C}^{\bar{N}\times\bar{N}\bar{M}}$, $\mathbf{A}_m\in\mathbb{C}^{\bar{N}\times\bar{N}}$ is a diagonal matrix, $m=1,...,\bar{M}$, $\mathbf{A}^H$ is the conjugated transpose of $\mathbf{A}$, and both $\mathbf{x}$ and $\mathbf{b}\in\mathbb{C}^{\bar{N}\bar{M}}$.

For the 3D case, $\bar{N}$ states the spatial-spectral dimensions and $\bar{M}$ states the number of convolutional elements to use in the collection of dictionaries and sparse coefficients. Following the notation $\boldsymbol{\mathscr{X}}=\{\boldsymbol{\mathscr{X}}_m\in\mathbb{R}^{M\times N\times L}|m=1,...,M_d\}$, with $\boldsymbol{\mathscr{X}}_m$ the same size as the SI of interest, then $\bar{N}=NML$ and $\bar{M}=M_d$, with N,M,L as the spatio-spectral dimensions and M_d as the size of the collection of convolutional elements.

For the 4D case, $\bar{N}$ changes to the spatial-spectral-temporal dimensions of the SV, while $\bar{M}$ states the number of convolutional elements. Following the notation $\boldsymbol{\mathscr{X}}=\{\boldsymbol{\mathscr{X}}_m\in\mathbb{R}^{M\times N\times L\times T}|m=1,...,M_d\}$, with $\boldsymbol{\mathscr{X}}_m$ the same size as the SV of interest, then $\bar{N}=NMLT$ and $\bar{M}=M_d$, with N,M,L,T as the spatio-spectral-temporal dimensions.

Note that the algorithm presented in Appendix 3 doesn't take into consideration the dimensions of the signal beyond the $\bar{N}$ and $\bar{M}$ parameters. This means that the optimized routines proposed in Appendix 3 can be used for the N-dimensional case as proposed in section 4.5, with the correspondent dimension adjustments. Again, although the proposed formulation doesn't change with an increment in the dimensions, the computational load and arrays size increase exponentially. This factors must be taking into consideration when working with a ND-formulation.

Appendix E. Extra Material on the Performance Evaluation of CSC3D

In (Barajas-Solano et al., 2019a) we compared the performance of the proposed CSC3D against the Kronecker basis and the CBPDN algorithm, as described next.

Test Images. The chosen SIs for the performance evaluation were the Pavia University and the Beads spectral scenes. This work uses three ($E = 3$) $128\times128\times16$ sections of the full spectral scenes which includes spectral bands 1 to 16 for both the Pavia SI (**?**) (see Fig. (55(a))) and Beads SI (**?**) (see Fig. (55(b))). Wolhberg *et al.* (Wohlberg, 2016c) reported that the convolutional dictionaries perform better when reconstructing the high frequency components of an image. Therefore, the high-frequency components for the SIs sections were extracted using a high pass filter, and included as test images (see Fig. 55(c) and 55(d)). The proposed algorithm was compared against an equivalent Kronecker-ADMM scheme as reference

$$\underset{\boldsymbol{\theta}_e}{\operatorname{argmin}} \|\boldsymbol{\Psi}\boldsymbol{\theta}_e - \mathbf{s}_{n,e}\|_2^2 + \lambda \|\boldsymbol{\theta}_e\|_1 , \tag{169}$$

where $\mathbf{s}_{n,e} = \mathbf{s}_e + \sigma\boldsymbol{\eta}$, $\boldsymbol{\eta}$ is a standard white Gaussian noise, $\sigma > 0$ is a noise level and $\mathbf{s}_e \in \mathbb{R}^{MNL}$ is the noise-free vectorized version of each e-SI. The noisy test images were obtained using MATLAB's *awgn* routine and three levels of noise: 10dB, 15dB and 20dB of SNR.

Performance Metrics. The performance metrics used were the Peak Signal to Noise Ratio (PSNR) for measuring the recovery quality, and the ℓ_0 norm for measuring the sparsity of the solutions. Considering that the convolutional coefficient maps are in fact a collection of M_d sparse

cubes $\mathbb{R}^{M\times N\times L\times M_d}$, compared to the single sparse cube $\mathbb{R}^{M\times N\times L}$ of the Kronecker basis, then the sparsity of the convolutional coefficient maps will be measured as

$$\text{sparsity} = \max_{m=1}^{M_d} \|\boldsymbol{\Theta}_m\|_0 \,. \tag{170}$$

This is, the sparsity of the convolutional solutions will be the maximum sparsity of the individual coefficient maps.

Initializations and Regularizer Parameters. The initial values for the collection of coefficients were set to zero for the three schemes, and the initial dictionary was established as a collection of random $M_d = 30$ dictionary elements of cubic size $d = 8$ for both the proposed and CBPDN algorithms. The proposed algorithm has proven to be sensible to the initialization of the dictionary, with the heuristic initialization strategy with the best results in terms of PSNR as

$$\mathbf{D}_m(i,j,k) = \begin{cases} \mathcal{N}(0,1) & \text{if } 3 \leq i,j,k \leq 6 \\ 0 & \text{otherwise.} \end{cases} \tag{171}$$

The initial dictionary for the CBPDN algorithm as established according to (Wohlberg, 2016b). The regularizer parameter λ, for the three solution schemes, was allowed to vary freely, choosing the value with optimal results in terms of PSNR for each SNR level.

Results

The proposed algorithm performs better than the state-of-the-art, in sparse representa- tion, Kronecker basis in presence of noise, improving both the reconstruction quality and sparsity levels.

When dealing with higher levels of noise (*i.e.* 10dB SNR), the proposed algorithm needs to be modified to improve its performance. When dealing with high-frequencies SIs, the proposed algorithm outperforms the state-of-the-art in all the scenarios.

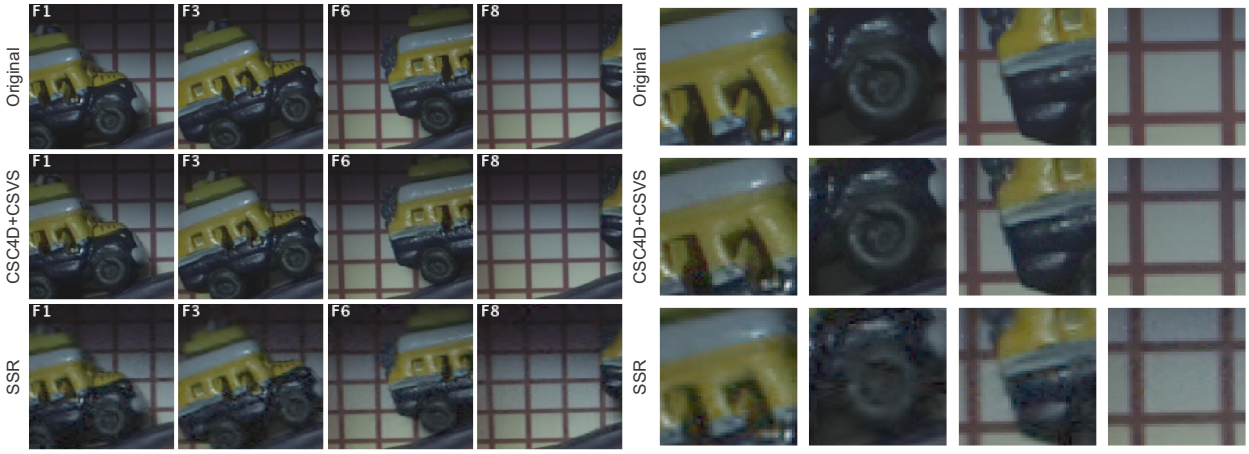


Figure 52. Example reconstructed frames of the simulated results at recovering from C-CASSI compressed measurements, at $K = 4$ and 23dB SNR Poisson noise, with the proposed CSC4D-CSVS and SSR methods for the dataset (a) Chiva and (b) some close-up details. Note that the proposed CSC4D-CSVS outperforms the SSR method with sharper edges, cleaner uniform areas, reduction of artifacts, even in presence of noise. I.e., note the tire decals in the second close up, and the body decal in the third close up.

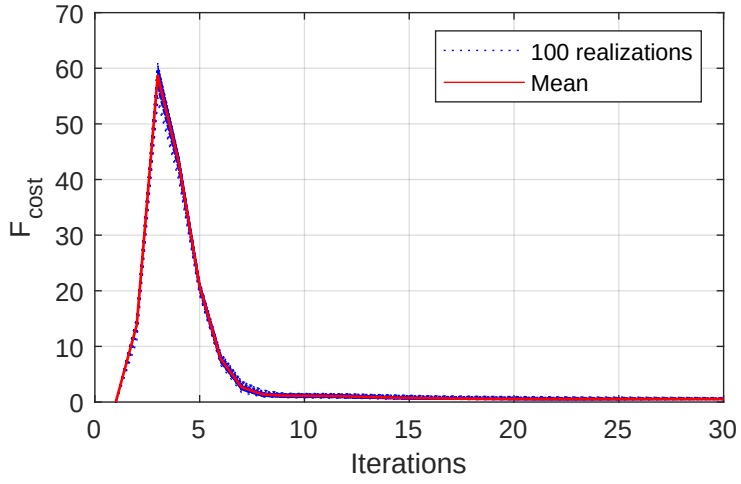


Figure 53. Mean behavior of the cost function of the proposed CSC4D-CSVS after 100 realizations.

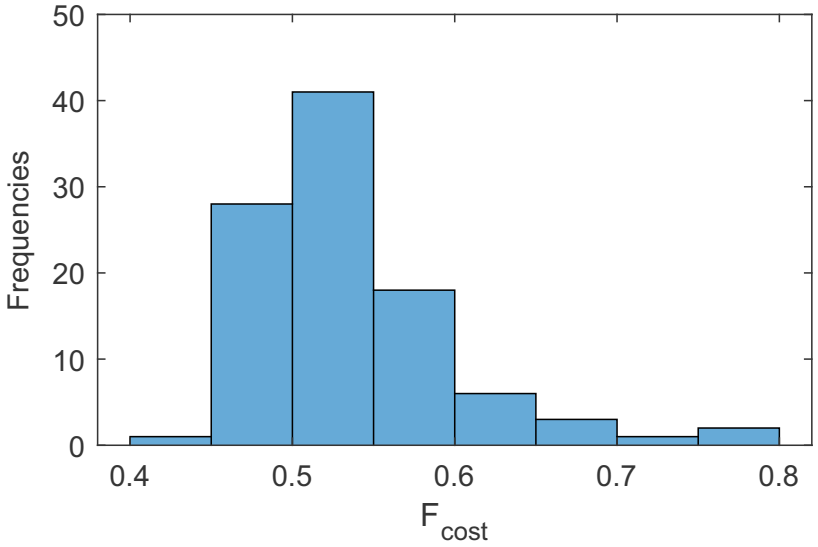


Figure 54. Histogram of the cost function of the proposed CSC4D-CSVS at random initializations.

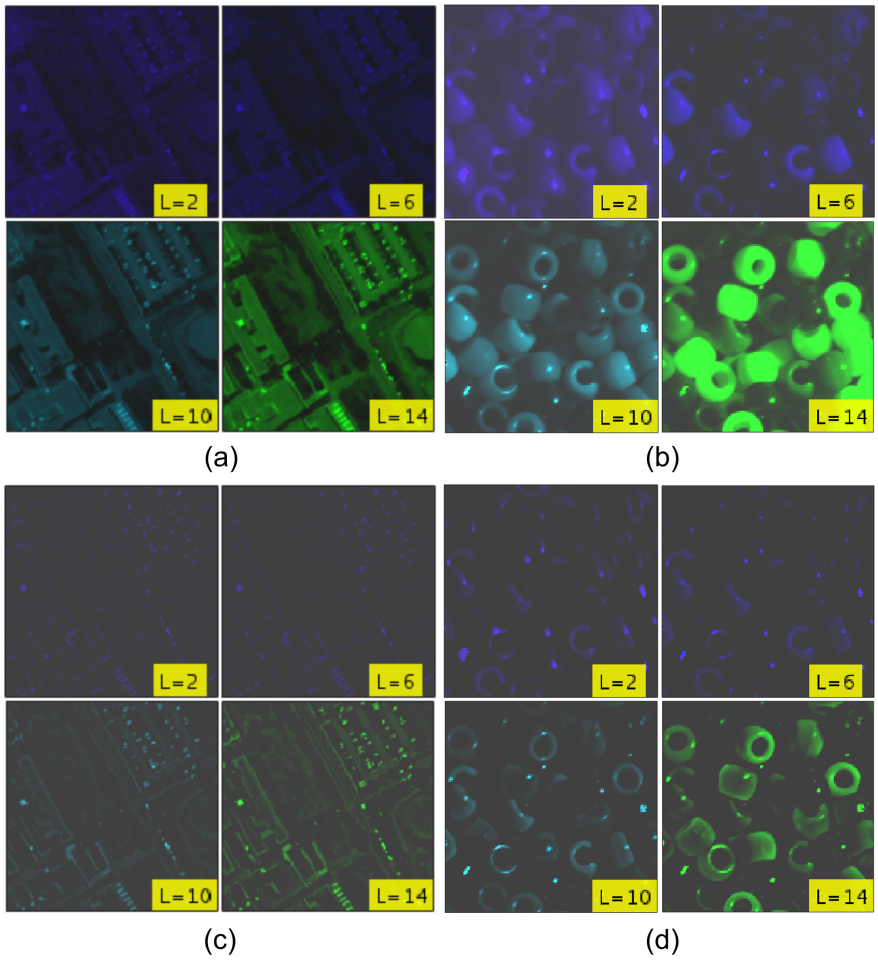


Figure 55. Example of the chosen spectral bands for one of the selected sections of (a)(c) the Pavia and (b)(d) Beads SIs, full-frequencies and high-frequencies versions respectively.

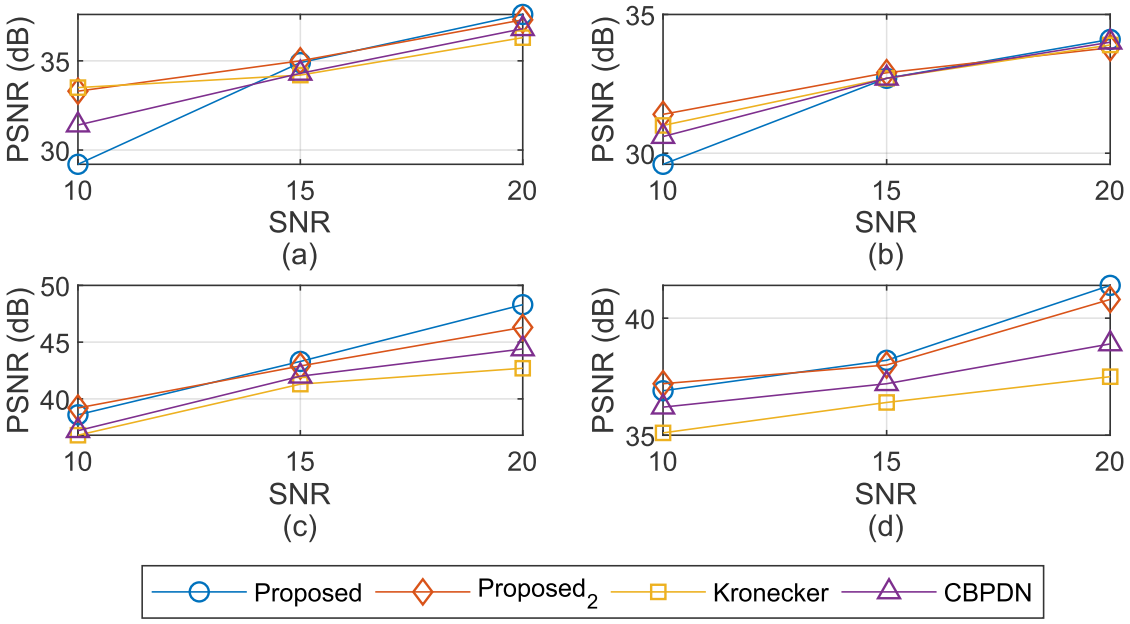


Figure 56. Performance comparison of the reconstruction quality for the 4 data sets (a) Pavia, (b) Beads, (c) Pavia high-frequencies and (d) Beads high-frequencies, using the proposed convolutional algorithm using two different random dictionary initilizations approaches (*Proposed* and $Proposed_2$), the Kronecker basis, and the CBPDN algorithm.