MCALab: Reproducible Research in Signal and Image Decomposition and Inpainting

Morphological component analysis of signals and images has far-reaching applications in science and technology, but some consider it problematic and even intractable. Reproducible research is essential to give MCA a firm scientific foundation. Researchers developed MCALab to demonstrate key MCA concepts and make them available to interested researchers.

Reproducibility is at the heart of scientific methodology and all successful technology development. In theoretical disciplines, mathematics has set the gold standard, where formal proof in principle allows anyone to reproduce the cognitive steps leading to a theorem's verification. In experimental disciplines—such as biology, physics, or chemistry—experiments must be replicable for a result to be well established. Of course, a necessary condition to truthfully replicate the experiment is that the paper describe it in enough detail that other groups can mimic it.

Computational science is a much younger field than mathematics, but is already of great importance for human welfare. However, it doesn't yet rigorously adhere to the reproducibility notion. The "reproducible research" slogan is relatively new; it tries to create a gold standard for computational reproducibility by recognizing that a research project's real outcome isn't the published article, but rather the entire environment that produced the results, including data, software, and documentation.[1] A closely related concept is *computational provenance*. In scientific experiments, provenance helps researchers interpret and understand results by examining the sequence of steps that led to them. In a sense, provenance's potential uses exceed reproducibility, as the May/June 2008 issue of *CiSE* attests.[2]

Inspired by Jon Clearbout, David Donoho began to practice reproducible research in the early 1990s in computational harmonic analysis within the now-proliferating Matlab environment.[3] In the signal-processing community, work by Mauro Barni and Fernando Perez-Gonzalez revitalized interest in reproducible research.[4] In 2007, the International Conference on Acoustics, Speech, and Signal Processing devoted an entire special session to reproducible signal-processing research.

Our reproducible research context is that of sparse-representation-based signal and image decomposition and inpainting. To achieve signal and image decomposition and inpainting, we use MCALab, a library of Matlab routines that

1521-9615/10/$26.00 © 2010 IEEE
Copublished by the IEEE CS and the AIP

Jalal M. Fadili
National Graduate School of Engineering and Research, Caen, France
Jean-Luc Starck
French Commission for Atomic Energy
Michael Elad
The Technion–Israel Institute of Technology
David L. Donoho
Stanford University

implements various algorithms.[5-7] MCALab offers the research community open source tools for sparse decomposition and inpainting, and can be used to reproduce the figures we present here and to redo them with parameter variations.

Problem Overview

Although mathematics has its million-dollar problems (the Clay Math Prizes), signal and image processing problems exceed a billion dollars. Famous ones include the phase problem (loss of information from a physical measurement) and the cocktail party problem (separate one sound from a mixture of other recorded sounds and background noises at a party). The "billion dollar" descriptor refers to the problems' enormous significance to the world economy should they be solved, as well as the extreme intellectual kick such a solution would necessarily entail. These signal-processing problems seem to be intractable according to orthodox arguments based on rigorous mathematics, and yet they keep cropping up in problem after problem.

One fundamental problem involves decomposing a signal or image into superposed contributions from different sources; think of symphonic music—which might involve superpositions of acoustic signals generated by many different instruments—and imagine the challenge of separating these contributions. More abstractly, various forms of media content are superpositions of contributions from different content types. Our goal is to separate out each type's contribution, which is a complex task. We can easily see a fundamental problem; for example, an n-pixel image created by superposing K different types offers us n data (the pixel values) but there may be as many as $n \cdot K$ unknowns (the contribution of each content type to each pixel). Traditional mathematical reasoning—in fact, the fundamental theorem of linear algebra—tells us not to attempt this because there are more unknowns than equations. On the other hand, if we have prior information about the underlying object, some rigorous results show that such separation might be possible using special techniques. We're thus left with a situation that has grounds for both hope and skepticism.

The idea to morphologically decompose a signal into its building blocks is an important problem in signal and image processing. An interesting and complicated image-content separation problem is one that targets image decomposition to texture and piecewise-smooth (cartoon) parts. Since the pioneering contribution of the French mathematician Yves Meyer on the G-space of oscillating patterns, we've witnessed a flurry of research activity in this application field. Successful methods for signal or image separation can be applied in a broad range of science and technology areas, including biomedical engineering, medical imaging, speech processing, astronomical imaging, remote sensing, and communication systems.

In morphological component analysis (MCA), we base decomposition on sparse signal representation.[5,8] MCA assumes that each signal is the linear mixture of several layers—or *morphological components*—that are morphologically distinct, such as sines and bumps. The method's success relies on the assumption that for every signal atomic behavior to be separated, there's a dictionary of atoms that enables its construction using a sparse representation. It's then assumed that each morphological component is sparsely represented in a specific transform domain. And, when all transforms (each one attached to a morphological component) are amalgamated in one dictionary, each one must lead to sparse representation over the part of the signal it's serving, while being highly inefficient in representing the other content in the mixture. If such dictionaries are identified, the use of a pursuit algorithm searching for the sparsest representation leads to the desired separation.

MCA can create atomic sparse representations containing (as a byproduct) a decoupling of the signal content. Researchers have extended MCA to simultaneous texture and cartoon images inpainting.[6] Other researchers have formalized the inpainting problem as a missing data-estimation problem.[7]

Theory and Algorithms

Before describing of the MCALab library, we'll first set a context by explaining basic terminology, reviewing the main theoretical concepts, and providing an overview of the algorithms.

Terminology

An *atom* is an elementary signal-representing template. Examples might include sinusoids, monomials, wavelets, and Gaussians. Using a collection of atoms as building blocks, we can construct more complex waveforms by linear superposition.

A *dictionary* $\mathbf{\Phi}$ is an indexed collection of atoms $(\phi_\gamma)_{\gamma \in \Gamma}$, where Γ is a countable set. The interpretation of the index γ depends on the dictionary; frequency for the Fourier dictionary (sinusoids), position for the Dirac dictionary (the standard unit vector basis, or kronecker basis), position scale for the wavelet dictionary, translation duration frequency for cosine packets, and position scale orientation

for the curvelet dictionary in two dimensions. In discrete-time finite-length signal processing, we can view a dictionary as a matrix whose columns are the atoms, which we can view as the column vectors. When the dictionary has more columns than rows, it's redundant, or *overcomplete*. Here, we're mainly interested in overcomplete dictionaries. In the overcomplete case, the signal $x = \Phi\alpha$ amounts to an underdetermined system of linear equations.

Given a dictionary, we must distinguish between analysis and synthesis operations. *Analysis operations* associate each signal x to a vector of coefficients α attached to an atom: $\alpha = \Phi^T x$. *Synthesis operations* reconstruct x by superposing atoms: $x = \Phi\alpha$. Analysis and synthesis are very different linear operations. In the overcomplete case, Φ isn't invertible and the reconstruction isn't unique (we discuss this in more detail later).

Next, we assume that a given n-sample signal or image x consists of a sum of K signals or images,

$$(x_k)_{k=1,\cdots,K}, x = \sum_{k=1}^{K} x_k,$$

that have different morphologies. Each x_k is called a *morphological component*.

Inpainting restores missing image information based upon the still available (observed) cues from destroyed, occluded, or deliberately masked subregions of the signal or image. Roughly speaking, inpainting is an interpolation of missing or occluded data.

Fast Implicit Transforms and Dictionaries

Given recent advances in modern harmonic analysis, researchers have shown many novel representations—including the wavelet transform, curvelet, contourlet, and steerable or complex wavelet pyramids—to be very effective in sparsely representing certain kinds of signals and images. For decomposition purposes, we build the dictionary by taking the union of one or several (sufficiently incoherent) transforms, each of which generally corresponds to an orthogonal basis or a tight frame.

From a practical viewpoint, given a signal x, we'll need to compute its forward (or analysis) transform by multiplying it by Φ^T. We also need to reconstruct any signal from its coefficients α. In fact, the matrix Φ and its adjoint Φ^T (corresponding to each transform) are never explicitly constructed in memory. Rather, they're implemented as fast implicit analysis and synthesis operators taking a signal vector x, and returning $\Phi^T x = \mathbf{T}x$ (such as Fourier or wavelet transforms), or taking a coefficient vector α and returning $\Phi\alpha$ (such as inverse Fourier or wavelet transforms).

Signal and Image Decomposition

Suppose that the n-sample signal or image x is the linear superposition of K morphological components. The MCA framework aims at recovering the components $(x_k)_{k=1,\ldots,K}$ from their observed linear mixture (see Figure 1). This is obviously an ill-posed inverse problem. MCA assumes that a dictionary can be built by amalgamating several transforms $(\Phi_1,\ldots,\Phi_K)$ such that for each k the representation α_k of x_k in Φ_k is sparse but α_k is not, or at least not as sparse, in other Φ_l, $l \neq k$. In other words, the subdictionaries $(\Phi_1,\ldots,\Phi_K)$ must be mutually incoherent. Thus, the dictionary Φ_k plays a role of a discriminant between content types, preferring the component x_k over the other parts.

Choosing an appropriate dictionary is a key step toward creating a good sparse decomposition and inpainting. Thus, to represent isotropic structures efficiently, a qualifying choice is the wavelet transform.[9] The curvelet system is a good candidate for representing piecewise smooth (C^2) images away from C^2 contours.[10] The ridgelet transform has proven highly effective for representing global lines in an image.[11] The local discrete cosine transform (DCT) is well suited to represent locally stationary textures.[9] These transforms are also computationally tractable, particularly in large-scale applications, and (as stated above) you should never explicitly implement Φ and $\mathbf{T}$. The associated implicit fast analysis and synthesis operators have typical complexities of $O(n)$, with n being the number of samples or pixels (such as orthogonal or bi-orthogonal wavelet transform) or $O(n \log n)$ (such as ridgelet, curvelet, or local DCT transforms).

By definition, the augmented dictionary $\Phi = [\Phi_1 \ldots \Phi_K]$ will provide an overcomplete representation of x. Again, because there are more unknowns than equations, the system $x = \Phi\alpha$ is underdetermined. In some idealized cases, we can use sparsity to find a unique solution.[12] Some researchers have proposed solving the underdetermined system of equations and estimate the morphological components $(x_k)_{k=1,\ldots,K}$ by solving the following constrained optimization problem:

$$\min_{x_1,\ldots,x_K} \sum_{k=1}^{K} \left\| \Phi_k^T x_k \right\|_p^p,$$

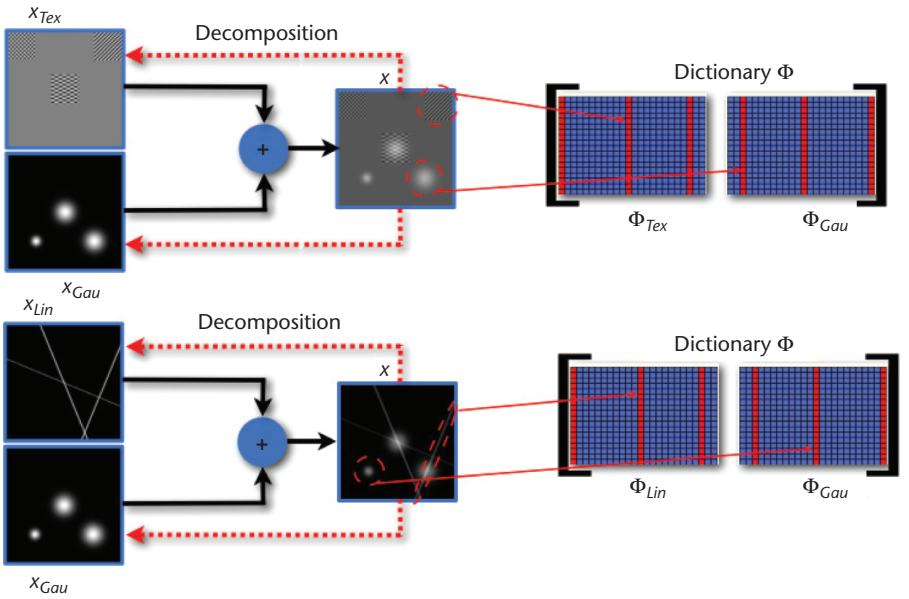


Figure 1. Illustration of the image decomposition problem with sparse representations. Each image x is a superposition of contributions from different content types (x_{Lin}, x_{Gau}, x_{Tex}). The goal is to separate out each type's contribution from the observed x by exploiting that each content type is sparse in a dedicated dictionary.

such that

$$\left\| x - \sum_{k=1}^{K} x_k \right\|_2 \leq \sigma, \tag{1}$$

where

$$\|\alpha\|_p^p = \sum_i |\alpha(i)|^p$$

is the penalty quantifying sparsity (the most interesting regime is for $0 \leq p \leq 1$), and

$$\|\alpha\|_0 = \lim_{p \to 0} \|\alpha\|_p^p$$

the ℓ_0 pseudonorm—that is, the number of nonzero components.[5,8] The constraint in this optimization problem accounts for the presence of noise and model imperfection. If there is no noise and the linear superposition model is exact ($\sigma = 0$), we can substitute an equality constraint for the inequality constraint. This formulation is flexible enough to incorporate external forces that direct the morphological components to better suit their expected content; these forces will fine-tune the separation process to achieve its task. As an example of a successful external force, Jean-Luc Starck and colleagues proposed adding a total variation (TV) penalty to the cartoon part to direct this component to fit the piecewise-smooth model.[8] This leads to

$$\min_{x_1,\cdots,x_K} \sum_{k=1}^{K} \left\| \Phi_k^T x_k \right\|_p^p + \gamma \|x_s\|_{TV},$$

such that

$$\left\| x - \sum_{k=1}^{K} x_k \right\|_2 \leq \sigma, \tag{2}$$

where s indexes the piecewise-smooth (cartoon) part, and γ the TV regularization parameter.

For the sake of completeness, we summarize Starck's MCA algorithm—which relies on an iterative thresholding scheme with varying threshold—in Algorithm 1 (see Figure 2). $\mathrm{TH}_\lambda(\alpha)$ denotes component-wise thresholding with threshold λ, hard thresholding ($p = 0$ in Equation 2), or soft thresholding ($p = 1$ in Equation 2). As far as the thresholding decreasing strategy is concerned, there are several alternatives. Researchers have, for example, advocated linear and exponential decrease.[5,8] Other researchers proposed a more elaborated strategy, called MOM (for mean-of-max).[13] Here (and in MCALab toolbox version 1.1.0), only linear and exponential decrease are implemented. As Starck suggested, an approximation of the TV regularization is implemented in MCALab by soft thresholding the undecimated Haar wavelet coefficients with threshold γ.[8]

MCA naturally handles data perturbed by additive noise ε with bounded variance $\sigma_\varepsilon^2 = \mathrm{Var}[\varepsilon] < +\infty$. In terms of the dictionary's

Algorithm 1. Morphological component analysis (MCA) component separation algorithm.
Task: Signal/image decomposition, solve (2).
Parameters: The signal/image x, the dictionary $\Phi = [\Phi_1 \ldots \Phi_K]$, number of iterations N_{iter}, γ and index of the cartoon part, stopping threshold λ_{min}, threshold update schedule.
Initialization:
- Initial solution $x_k^{(0)} = 0, \forall k$.
- Initial residual $r^{(0)} = x$.
- Initial threshold: let $k^* = \max_k \left\| \Phi_k^T x \right\|_\infty$, set $\lambda^{(0)} = \max_{k=k^*} \left\| \Phi_k^T x \right\|_\infty$.

Main iteration:
For $t = 1$ to N_{iter}
- **For** $k = 1$ **to** K,
  - Compute marginal residuals $r_k^{(t)} = r^{(t)} + x_k$.
  - Update k-th component coefficients by thresholding $\alpha_k^{(t)} = \mathrm{TH}_{\lambda(t)} (\Phi_k^T r_k^{(t)})$.
  - Update k-th component $x_k^{(t)} = \Phi_k \alpha_k^{(t)}$.
  - If k is the cartoon part and $\gamma \neq 0$, TV penalty to $x_k^{(t)}$.
- Update the residuals $r^{(t+1)} = x - \sum_{k=1}^K x_k^{(t)}$.
- Update the threshold $\lambda^{(t+1)}$ according to the given schedule.
- **If** $\lambda^{(t+1)} \leq \lambda_{min}$ **then** stop.

End iteration
Output: Morphological components $(x_k^{(N_{iter})})_{k=1,\cdots,K}$.

Figure 2. Algorithm 1 summarizes the morphological component analysis (MCA) algorithm, which relies on an iterative thresholding scheme with varying threshold. TH$\lambda(\alpha)$ denotes component-wise thresholding with threshold λ, hard thresholding ($p = 0$ in Equation 2) or soft thresholding ($p = 1$ in Equation 2).

coefficient amplitude, MCA is an iterative coarse-to-fine procedure and can handle bounded noise by simply stopping iteration when the residual is at the noise level. Assuming that we know the noise variance σ_ε^2, we can stop the algorithm at iteration t when the ℓ_2-norm of the residual satisfies

$$\| r^{(t)} \|_2 \leq \sqrt{n\sigma_\varepsilon}.$$

Alternatively, we can use a strategy reminiscent of denoising methods by taking $\lambda_{min} = \tau \sigma_\varepsilon$ where τ is a constant (typically between 3 and 4).

Signal and Image Inpainting

Assume we define the original signal/image on a bounded domain Ω. Then suppose we partially observe a signal/image x on a compact domain $\Omega_o \subset \Omega$, $n_o = \mathrm{Card}\ (\Omega_o)$, where some samples are missing (those indexed by $\Omega \backslash \Omega_o$). By "mask" M we mean the diagonal $n \times n$ matrix where the observed samples are indicated by ones and the missing ones by zeros.

Elad algorithm. Michael Elad and his colleagues describe an inpainting algorithm that is a direct extension of MCA and can fill holes in images that are a superposition of a texture and cartoon layers.[6] Thus, we can extend Equation 2 to incorporate missing data through the mask M by minimizing

$$\min_{x_1,\cdots,x_K} \sum_{k=1}^K \left\| \Phi_k^T x_k \right\|_p^p + \gamma \| x_s \|_{TV}$$

such that

$$\left\| x - M \sum_{k=1}^K x_k \right\|_2 \leq \sigma. \quad (3)$$

In our approach to solving this optimization problem, we need adapt the MCA Algorithm 1 only slightly, modifying the residual update rule to

$$r^{(t+1)} = x - M \sum_{k=1}^K x_k^{(t)}$$

to account for the masking matrix M. The other steps of the algorithm remain unchanged.[6]

Fadili algorithm. When taking $p = 1$ and dropping the TV-regularization term in Equation 3, the inpainting problem can be formalized as a missing data-estimation problem.[7] In this case, the researchers formulated the problem as a maximum penalized likelihood estimator with missing data and sparsity-promoting penalties. They carefully characterized it and solved it using tools from nonsmooth convex optimization theory. A chief advantage of this algorithm is its ability to estimate the noise variance. Let's denote the relaxation parameter

$$0 < \mu < 2 / \sum_k \left\| \Phi_k \right\|_2^2,$$

where

$$\left\| \Phi_k \right\|_2^2$$

Algorithm 2. Inpainting algorithm inspired by expectation maximization (EM).
Task: Signal/image inpainting.
Parameters: Observed masked image x, the mask $\mathbf{M}$, the dictionary $\Phi = [\Phi_1 \ldots \Phi_K]$, regularization parameter λ, initial $\sigma_\varepsilon^{(0)}$, convergence tolerance δ.
Initialization: Initial solution $\alpha_k^{(0)} = 0$, $\forall k$.
Main iteration:
Repeat
• Update the residual $r^{(t)} = x - \mathbf{M}\Phi\alpha^{(t)}$.
• Update the coefficients $\alpha^{(t)}$

$$\alpha^{(t+1)} = \text{SoftThreshold}_{\lambda\mu}\left(\alpha^{(t)} + \mu\Phi^T r^{(t)}\right).$$

• If desired, update $\sigma_\varepsilon^{2(t+1)}$ according to

$$\sigma_\varepsilon^{2(t+1)} = \frac{1}{n}\left[\sum_{i\in\Omega_c}(x(i) - (\Phi\alpha^{(t)})(i))^2 + (n - n_0)\sigma_\varepsilon^{2(t)}\right].$$

Until $\left\|\alpha^{(t+1)} - \alpha^{(t)}\right\|_2 \le \delta$.
Output: Inpainted image $\hat{x} = \Phi\alpha^{(t)} = \sum_k \Phi_k \alpha_k^{(t)}$.

Figure 3. Algorithm 2 summarizes the EM-inspired inpainting algorithm, which relies on an iterative soft thresholding scheme. $SH_\lambda(\alpha)$ denotes component-wise soft thresholding with threshold λ.

corresponds to the largest frame bound associated to Φ_k. We summarize this inpainting method in Algorithm 2 (see Figure 3); other variations are described elsewhere.[7]

Algorithm comparison. The two inpainting algorithms have differences and similarities:

- *Target.* In the Elad algorithm formulation, the targets are the morphological components x_k and component separation is a byproduct of the inpainting process. In Fadili, the problem is optimized with respect to the coefficients α_k. This distinction obviously disappears when Φ is an orthobasis.
- *Parameters.* In Elad's MCA-based inpainting, the user provides the algorithm with a threshold-lowering schedule and a stopping threshold $\lambda_{\min}$; in Fadili, the user must give a fixed regularization parameter λ. In practice, it might be more difficult to choose λ than to choose the stopping threshold.
- *Noise.* Both algorithms handle the presence of noise; Fadili's formulation can estimate the noise variance along with inpainting.
- *Optimization algorithms.* Despite apparent similarities, the two formulations use different optimization frameworks. MCA is a heuristic algorithm, formed by hybridizing matching pursuit with block-coordinate relaxation. The formulation of the Fadili algorithm uses an iterative thresholding proximal iteration with a rigorous convergence analysis guaranteed when $p \ge 1$.

We now describe the MCALab architecture, which implements the previously described algorithms and reproduces the experiments described in the corresponding papers.

MCALab Architecture

MCALab is a library of Matlab routines that implements the algorithms we describe here for signal and image decomposition and inpainting. The library is available free of charge at www.morphologicaldiversity.org/mcalab.html. The current version of MCALab is 1.1.0 and has been successfully tested under Unix Solaris, Linux, and Mac OS X (PPC and Intel) under Matlab 6.x and 7.x. MCALab is distributed under the Cecill software license (www.cecill.info/index.en.html), which doesn't require advanced permission for noncommercial use. Figure 4 shows the MCALab package's structure.

MCALab has two main directories—one for 1D signals and another for 2D images. Each of these directories has the same architecture and contains the following subdirectories:

- *Dictionaries.* The `MCALab110/xxx-D/Dictionaries` directory contains various fast implicit analysis and synthesis transforms including those discussed earlier.
- *Decomposition.* The `MCALab110/xxx-D/Decomposition` directory contains the implementation of Algorithms 1 and 2.
- *Datasets.* The `MCALab110/xxx-D/Datasets` directory contains signal and image datasets used to illustrate decomposition and inpainting. MCALab110/One-D/Datasets also contains a Matlab function that generates many artificial signals and reads some real signals.

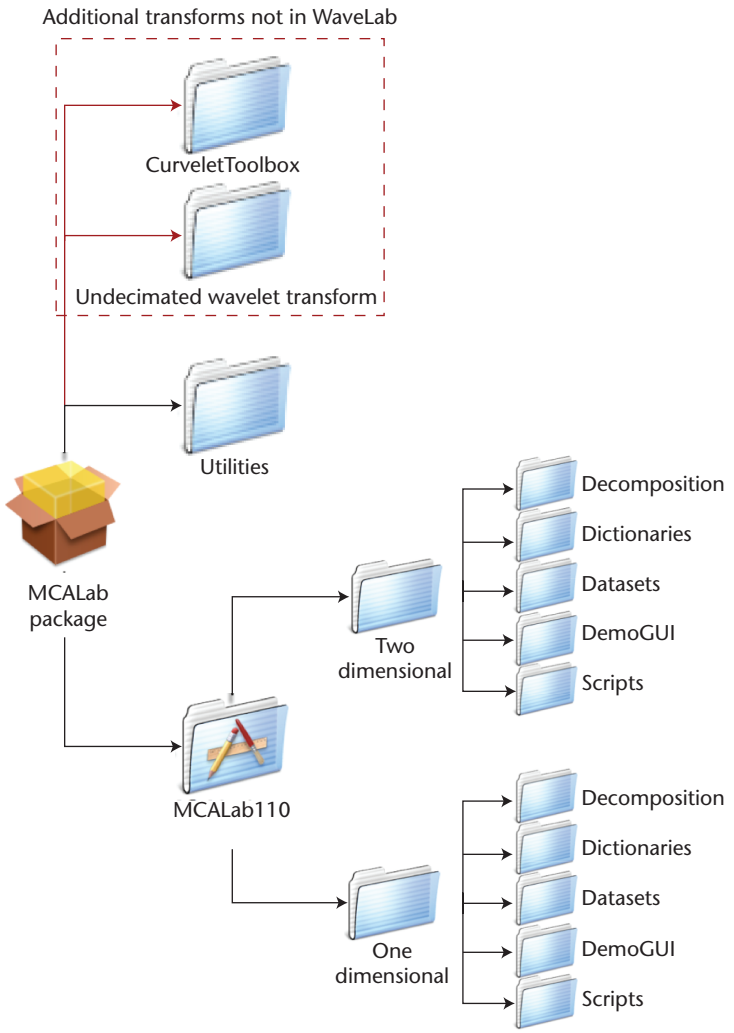


Figure 4. MCALab package architecture. MCALab has two directories—one for 1D signals and another for 2D images. Each of these directories has the same organization.

- *DemoGUI.* The `MCALab110/xxx-D/DemoGUI` directory contains a GUI to decompose and inpaint the user's own data or data supplied with MCALab.
- *Scripts.* The `MCALab110/xxx-D/Scripts` directory contains pedagogical examples with synthetic and real data that illustrate the use of MCA and inpainting algorithms.

The Scripts directory also includes examples of metadata structures to help users reproduce the results and get started with the package.

MCALab Dependencies

As we mentioned earlier, MCALab requires implementation of forward and inverse transforms. As it turns out, many of them—including orthogonal wavelet transform, discrete cosine, and sine transforms—are already implemented in WaveLab (http://playfair.stanford.edu/~wavelab). We therefore recommend that users download and install WaveLab for MCALab to work properly.

MCALab also incorporates software for two other transforms not included in WaveLab: the wrapping version of the fast discrete curvelet transform (FDCT) implemented in CurveLab (www.curvelet.org)[10] and the undecimated discrete wavelet transform (UDWT) implemented in the Rice wavelet toolbox (www.dsp.rice.edu/software/rwt.shtml). We slightly modified the FDCT Matlab functions `fdct wrapping.m` and `ifdct wrapping.m` (not the Matlab Mex files) to match our dictionary data structure and implement curvelets at the finest scale. We strongly recommend that users download this modified version or at least use our `fdct wrapping.m` and `ifdct wrapping.m`. Both of these transforms are available in the `MCALabWithUtilities` MCALab version (the dashed rectangle in Figure 4) in the `MCALabWithUtilities/CurveleToolbox` and `MCALabWithUtilities/UDWT` subdirectories (for further details, see `Contents.m` in `MCALabWithUtilities`). We also encourage users to read the transforms' software license agreements, which are available on the respective Web sites.

Because MCALab has external library dependencies, reproducibility and sensitivity to third-party libraries are a legitimate concern. However, the dependencies are essentially on the transforms, and given our implementation of the dictionaries, these transforms are called as external functions from MCALab. So, no modification is necessary on the MCALab code if such transforms are corrected or modified. Moreover, to make MCALab's behavior with these transforms more robust, we tested MCALab with both WaveLab 802 and 805 and CurveLab (versions 1.0, 2.0, 2.1.1, and 2.1.2) and obtained exactly the same results.

MCALab Metadata Generation

MCALab also generates metadata that contains necessary information to facilitate reproducibility and reusability. As Figure 5 shows, this metadata is generated as a Matlab structure object with fields that indicate

- the version of Matlab and machine architecture;
- the version of third-party libraries that generated a given result (here, for example, we used WaveLab 802, CurveLab 2.1.1, and RWT 1.2); and
- all MCALab options, including the version (here, 1.1.0), the task (decomposition or inpainting), parameters, I/O, and the dictionary description.

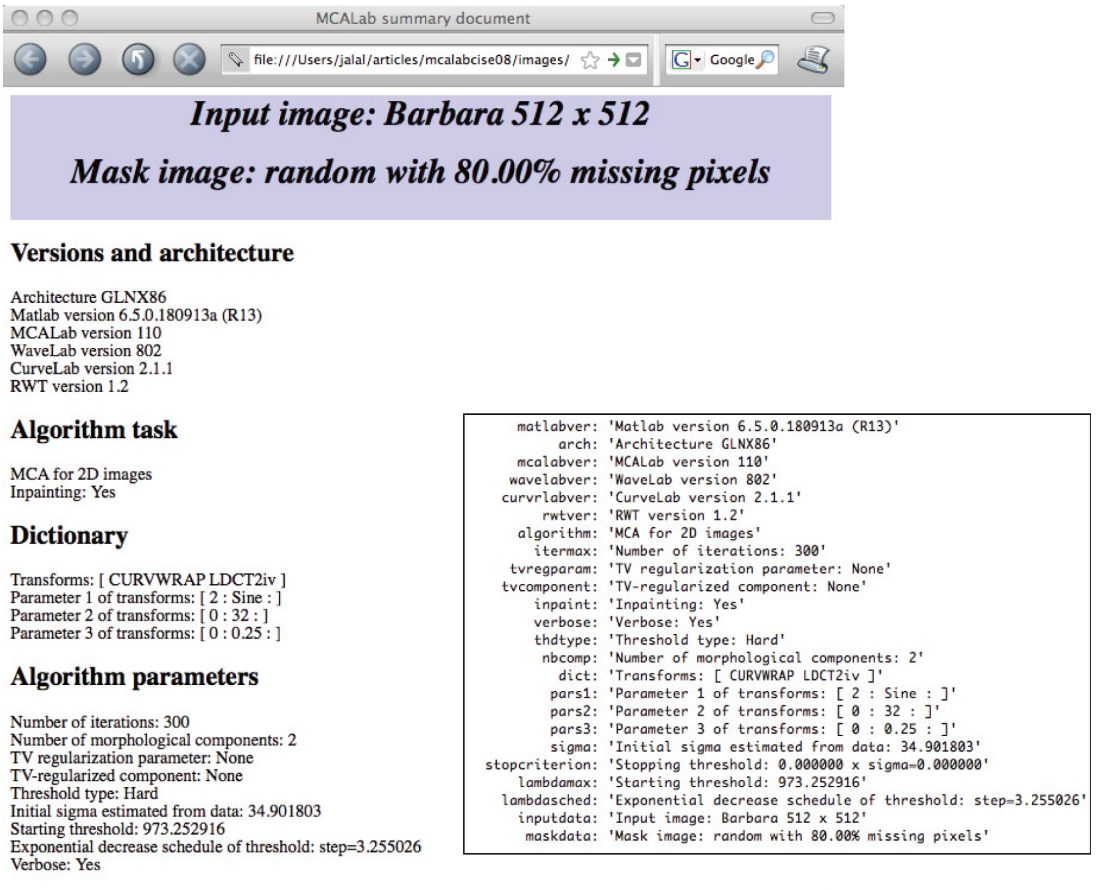


Figure 5. Metadata generated by MCALab in HTML for the experiment shown in Figure 12. The black rectangle represents the corresponding Matlab structure of a metadata object.

When we launch MCALab on the call `MCAPath.m`, this metadata object is initialized with the Matlab version, machine architecture, and external libraries versions. The object is then stored in the `Utils/` directory. The MCALab decomposition and inpainting routines load and add additional information to this object. In addition to the resulting signals and images, the test scripts can either store this object or export it in HTML format it using the MCALab function `metaoptions2html.m` (see Figure 5). Examples of metadata Matlab structure objects that the tests scripts store are available in the toolbox in the `.mat` Matlab format (version 5 for compatibility with older Matlab versions). Another interesting alternative worth investigating in a future version would be to exploit the XML `Publish` function, which is offered by Matlab but only available with versions 7.x and later.

To encourage users to reproduce our results from the metadata structure, we've also written an MCALab function `firefrommetadata.m` that can reproduce the results from previously stored metadata with as little extra information as possible. For user convenience, MCALab includes two test scripts in `MCALab110/xxx-D/Scripts` that exemplify the use of `firefrommetadata.m` on some 1D and 2D examples.

Documentation

MCALab is documented and each MCALab function has help documentation. The `README` and `INSTALL` files provide some guidelines to help users access and install the software and get started in exploring the MCALab package resources. Users can also check the MCALab Web page, which contains links to our Web sites, the abstract of this article, references to other MCALab articles, a zipped archive of MCALab package, and an email address for remarks and bug reports.

Dictionaries and Data Structures

The functions in `xxx-D/Dictionary` subdirectory provide fast implicit analysis and synthesis operators for all MCALab dictionaries. With the exception of FDCT with the wrapping implementation,[10] all MCALab dictionaries are normalized such that atoms have unit ℓ_2-norm.

The FDCT implements a non-normalized Parseval tight frame. Thus, a simple rule to compute the ℓ_2-norm of the atoms is

$$1/\sqrt{\textit{redundancy of the frame}}\,.$$

Normalization has an important impact on the thresholding step involved in the decomposition/inpainting engines. The thresholding functions coded in MCALab take care of these normalization issues for the dictionaries it implements.

The data structure and the routines are inspired by those of the Atomizer (www-stat.stanford.edu/~Atomizer), and we've implemented the dictionaries in an object-oriented way. In addition,

- the top-level routines are `FastLA` and `FastLS`;
- each transform has a fast analysis and a fast synthesis function, `FasNAMEOFTRANSFORMAnalysis` and `FasNAMEOFTRANSFORMSynthesis`, respectively;
- four arguments (`NameOfDict`, `par1`, `par2`, `par3`) specify a dictionary;
- each parameter's meaning is explained in the corresponding routine;
- each `FasNAMEOFTRANSFORMAnalysis` computes the transform coefficients—that is, the application of Φ^T—of an image or a signal and stores them in a structure array; and
- we define the `List` data structure and corresponding functions, which lets users create and easily manipulate overcomplete merged dictionaries.

Users can easily implement their own dictionaries and associate them with a new transform following the above development rules.

Decomposition and Inpainting Engines

This directory's routines perform either signal or image decomposition and inpainting using the MCA-based Algorithm 1 or signal and image inpainting using Algorithm 2. The set of routines is for dictionaries with fast implicit operators. We also designed it to return a metadata object that can be saved or exported in HTML format to facilitate result reproducibility.

Scripts and GUI

The `/MCALab110/xxx-D/Scripts` and `/MCALab110/xxx-D/DemoGUI` is the MCALab toolbox's whole raison d' être. The `/MCALab110/xxx-D/Scripts` subdirectory contains a variety of scripts, each of which contains a sequence of commands, datasets, and parameters for generating figures from our own articles, as well as other exploratory examples. Users can run the same experiments on their own data or tune the parameters by simply modifying the scripts. By studying these scripts, users can quickly learn the practical aspects of sparse-representation-based image decomposition and inpainting, as well as how to use the MCALab software library.

The `/MCALab110/xxx-D/DemoGUI` routines implement a point-and-click GUI that lets users select data and masks (signals or images), and then decompose or inpaint them using MCA Algorithm 1. Only signals and images with two morphological components—and hence two dictionaries—can be processed with this GUI. It's possible to select the type of dictionaries and their associated parameters `par1`, `par2`, or `par3` for each morphological component. The main window is invoked via the command `MCA1Demo` (signals) or `MCA2Demo` (images). Upon initialization, one main window will appear with four plots, a menu, and an interactive area on the right. Two additional top-bar pull-down menus are specific to MCALab GUI and let users load data variables from the Matlab workspace. Thus, prior to using the GUI, the target signals, images, or masks must first be read from disk and loaded to the workspace using any Matlab input-data reading routine. If your goal is to decompose and not to inpaint, it's sufficient to create a dummy variable of ones that are the same size as the data. By default, the GUI assumes the trivial mask (all ones).

Examples of Reproducible Experiments

Table 1 summarizes the data description, dictionaries, and algorithm parameters that we used in the following experiments.

Bumps and Local Cosines

In Figure 6a, the synthetic signal (dashed line) is a composite of two Gaussian bumps with different scales and two local cosine waves. In this synthetic signal, we know the ground truth decomposition. We also added white Gaussian noise with $\sigma_\varepsilon = 0.429$ (PSNR = 20dB). Based on the signal's morphological content, we decomposed and denoised it using an overcomplete dictionary containing the local DCT and the undecimated wavelet transform (see Table 1 for the dictionary details). The standard deviation σ_ε wasn't supplied; the algorithm estimated it from the noisy data using the median absolution deviation (MAD) estimator.[14]

Table 1. Experimental setup for the 1D examples.

Signal	Dictionaries	Task	Algorithm parameters
Bumps + Local cosines $n = 1{,}024$	LocalDCT (square window, width = 32, no overlap) Undecimated discrete wavelet transform (UDWT) (Symmlet 6, coarsest scale 2)	Decomposition and denoising (Algorithm 1)	$N_{iter} = 100$, σ_ε: estimated from data, $\gamma = 0$ (no total variation—TV), $\lambda_{min} = 4\,\hat{\sigma}_\varepsilon$, exponentially decreasing threshold
TwinSine + Diracs $n = 1{,}024$	GlobalDCT (fineness = 4) Dirac	Decomposition (Algorithm 1)	$N_{iter} = 200$, $\gamma = 0$ (no TV), $\lambda_{min} = 0$, linearly decreasing threshold
EEG-fMRIsignal $n = 8{,}192$	LocalDCT (squarewindow, width = 32, overlap = 50% × width) UDWT (Symmlet 6, coarsest scale 2)	Decomposition and denoising (Algorithm 1)	$N_{iter} = 100$, σ_ε: estimated from data, $\gamma = 0$ (no TV), $\lambda_{min} = 4\,\hat{\sigma}_\varepsilon$, exponentially decreasing threshold

Figures 6b through 6d show the results. In Figures 6c and 6d, the dashed line shows the original signal, while the solid line represents the locally oscillating part and localized bumps, respectively, that MCA recovered. We plotted the addition of both recovered components as a continuous line in Figure 6b superimposed on the original signal (dashed line).

TwinSine and Dirac Signals

Figure 7 shows our second example, which decomposes the TwinSine and Dirac signals. The synthetic signal of Figure 7a (dashed line) consists of three randomly located spikes and the TwinSine signal: two cosines with frequencies separated by less than the Rayleigh distance.[15] We analyzed this signal with an overcomplete dictionary formed by merging a fourfold overcomplete DCT and a standard (Dirac) basis. Table 1 summarizes the experiment's parameters and Figures 7b through 7d show the results. As Figures 7b and 7c show, MCA resolved the frequencies of the oscillating (TwinSine) component and separated it properly from the spikes (Figure 7d). Some spurious frequencies remain in the oscillating part. The locations of the true frequencies correspond to the dotted lines in Figure 7c; the crosses in Figure 7d indicate the original spikes.

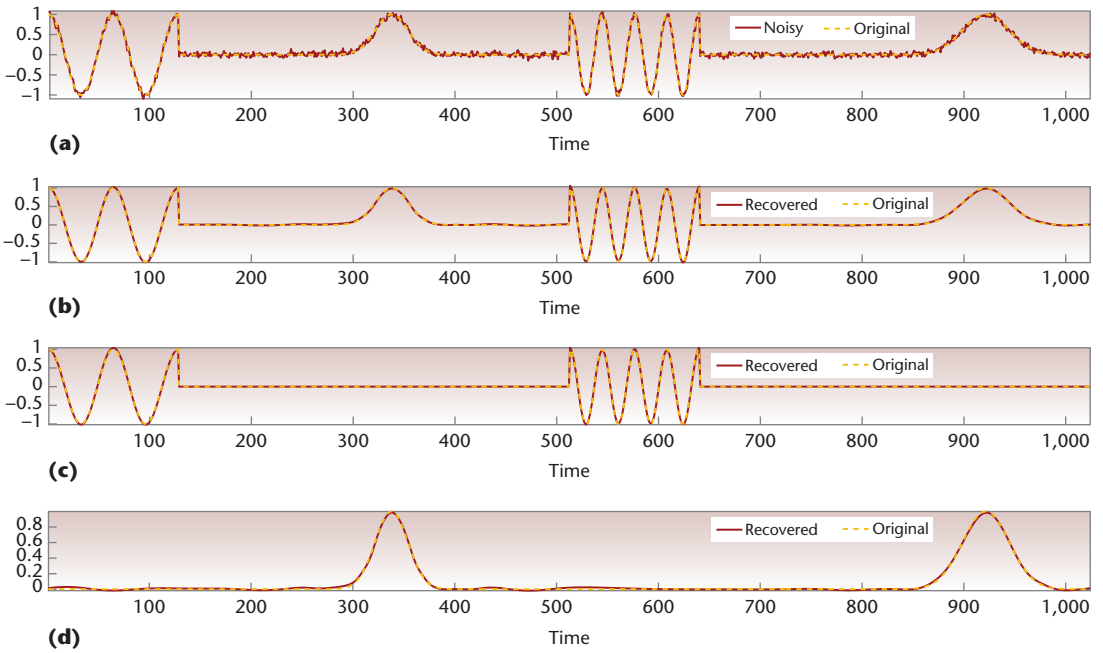


Figure 6. Decomposing and denoising local sines and bumps signal. (a) Noisy signal with a peak signal-to-noise ratio of 20 decibels, (b) a denoised morphological component analysis (MCA) with a PSNR of 38.1 dB, (c) local cosines with a PSNR of 39.9 dB, and (d) bumps with a PSNR of 38.5 dB.

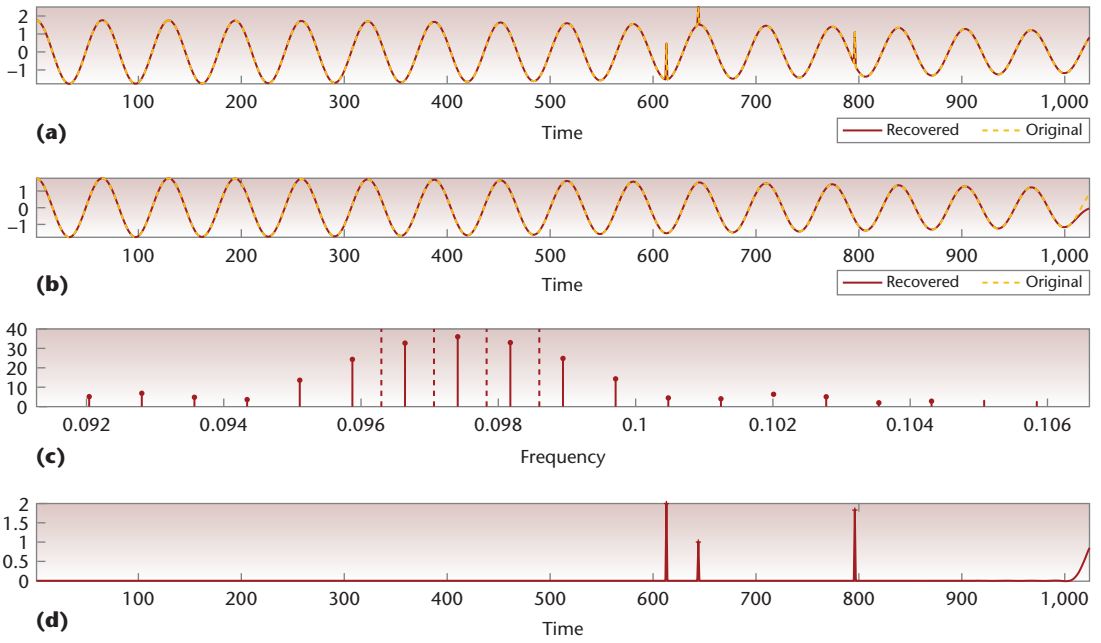


Figure 7. Decomposing TwinSine and Dirac signals. (a) The morphological component analysis (MCA) recovery with a peak signal-to-noise ratio of 67.4488 decibels, (b) the MCA TwinSine component with a PSNR of 29.3687 dB, (c) the MCA discrete cosine transform coefficient, and (d) the MCA Dirac component with a PSNR of 30.5055 dB.

EEG and fMRI

Finally, we applied MCA to a real signal acquired during a multimodal neuroimaging experiment. During this experiment, electroencephalography (EEG) and functional magnetic resonance imaging (fMRI) data are recorded synchronously to enable study of the spatio-temporal dynamics of brain activity. However, this simultaneous acquisition of EEG and fMRI data faces a major difficulty: the currents induced by the MR magnetic fields yield strong artifacts that generally dominate the useful EEG signal. In a continuous acquisition protocol, both modalities can record data. However, to exploit the EEG signal, we need a postprocessing step for EEG artifact reduction.

Figure 8a shows an example of such a signal. The periods where the MR sequence is switched on are clearly visible, inducing strong oscillations on the EEG signal with an amplitude 10 to 20 times higher that the expected EEG amplitude. This locally oscillating behavior is expected because of the shape of the MR scanner's magnetic field gradients. Therefore, to get rid of the MR component and clean the EEG signal from these artifacts, we decomposed it using an overcomplete dictionary of local DCT (for the MR part) and the UDWT (for the EEG smooth and localized component). Table 1 shows the experiment's parameters and Figures 8c and 8d show its results. As a byproduct, we also cleaned the signal of measurement noise.

2D Examples of Reproducible Experiments

Table 2 summarizes the data description, dictionaries, and algorithm parameters that we used in the following experiments.

Decomposition

We now describe our reproducible experiments on 2D image decomposition.

Boy with texture. In this experiment, we generated a synthetic image composed of a natural scene and a texture, giving us the ground truth parts to compare against.[6,8] We used the MCA decomposition Algorithm 1 with the curvelet transform for the natural scene part, and a global DCT transform for the texture; see Table 2 for details. The TV regularization parameter γ was fixed to 0.1. In this example (and for the Barbara example we describe next) our results were better if we ignored the textured part's very low frequencies. The reason for this is the evident coherence between the two dictionaries' elements at low frequencies—both claim the low-frequency content. Figure 9a and 9b show the original and recovered images (the addition of the texture and the natural parts); 9c and 9d show the original and

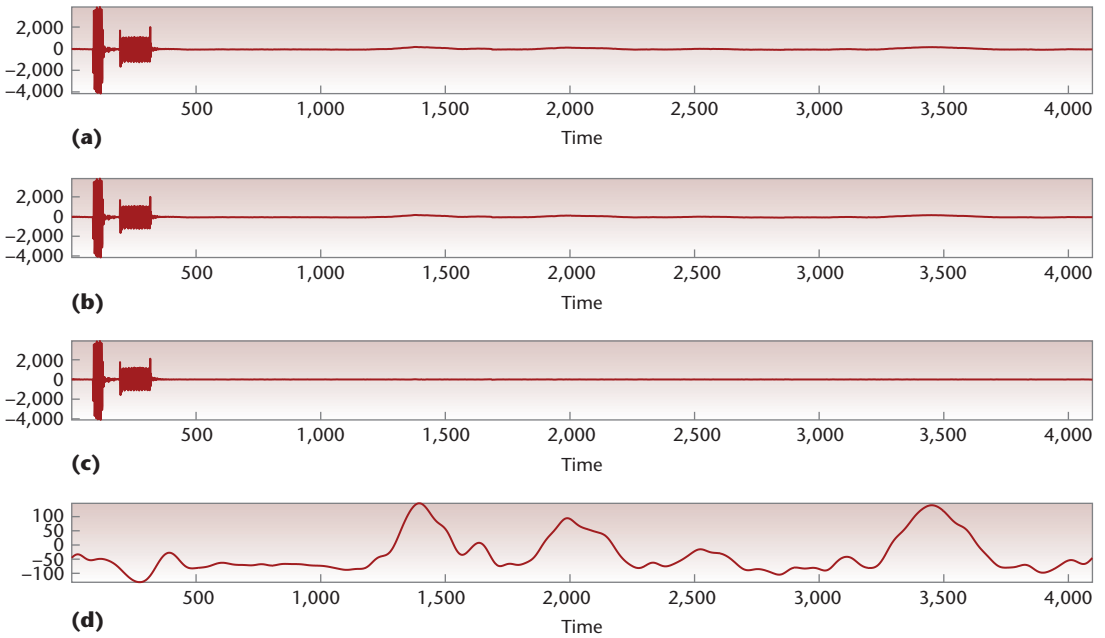


Figure 8. Decomposing and denoising an electroencephalography (EEG) and functional magnetic resonance imaging (fMRI) signal. (a) The original EEG–fMRI signal, (b) the denoised signal, (c) the morphological component analysis (MCA) MRI magnetic-field induced component, and (d) the MCA EEG component without the MRI. (Data courtesy of GIP Cyceron, Caen, France.)

Table 2. Experimental setup for the 2D examples.

Image	Dictonaries	Task	Algorithm parameters
Boy + texture 256 × 256	Curvelets (cartoon) (coarsest scale 2) GlobalDCT (texture) (low frequencies removed)	Decomposition (Algorithm 1)	$N_{iter} = 50$, $\gamma = 0.1$ (TV on cartoon part), $\lambda_{min} = 0$, exponentially decreasing threshold
Barbara 512 × 512	Curvelets (cartoon) (coarsest scale 2) LocalDCT(texture) (sine-bellwindow, width = 32, low frequencies removed)	Decomposition (Algorithm 1) Inpainting (Algorithm 1)	$N_{iter} = 300$, $\gamma = 2$ (TV on cartoon part), $\lambda_{min} = 0$, exponentially decreasing threshold $N_{iter} = 300$, $\gamma = 0$ (no TV), $\lambda_{min} = 0$, linearly decreasing threshold
Risers 150 × 501	Curvelets (lines) (coarsest scale 2) Undecimated discrete wavelet transform (UDWT) (smooth and isotropic) (Symmlet 6, coarsest scale 2)	Decomposition and denoising (Algorithm 1)	$N_{iter} = 30$, σ_ε: estimated from data, $\gamma = 2$ (TV on UDWT part), $\lambda_{min} = 3\,\hat{\sigma}_\varepsilon$, exponentially decreasing threshold
Lines + Gaussians 256 × 256	Curvelets (lines) (coarsest scale 3) UDWT (Gaussians) (Symmlet 6, coarsest scale 2)	Inpainting (Algorithm 1) Inpainting (Algorithm 2)	(Algorithm 2) $N_{iter} = 50$, $\gamma = 0$ (no TV), $\lambda_{min} = 0$, linearly decreasing threshold No noise, $\delta = 1E\text{-}6$, $\lambda = 0.3$
Lena 512 × 512	Curvelets (coarsest scale 2)	Inpainting (Algorithm 1) Inpainting (Algorithm 2)	$N_{iter} = 300$, $\gamma = 0$ (no TV), $\lambda_{min} = 0$, exponentially decreasing threshold no noise, $\delta = 1E\text{-}6$, $\lambda = 10$

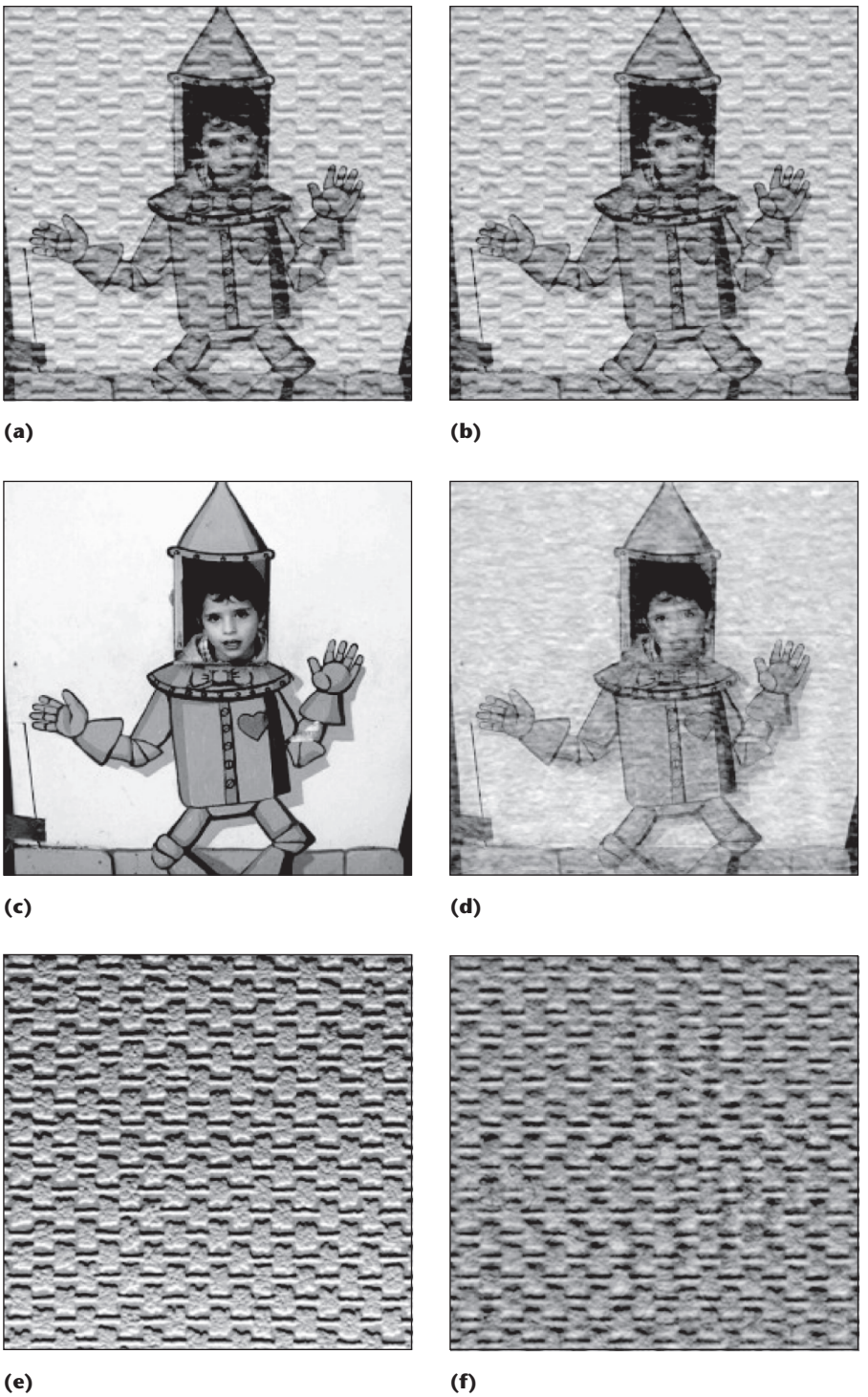


Figure 9. Decomposing the boy-texture image. (a) The original boy-texture image, (b) the recovered image by morphological component analysis, (c) the original cartoon part, (d) the MCA cartoon, (e) the original texture, and (f) the MCA texture.

reconstructed cartoon (natural) component; and 9e and 9f show the original and MCA textures.

Barbara. We also applied the MCA decomposition Algorithm 1 to the Barbara image. We used the curvelet transform for the cartoon part and a local DCT transform with a smooth sine window and a window size of 32×32 for the locally oscillating texture. We set the TV regularization parameter γ to 2. Figures 10a through 10d show the Barbara

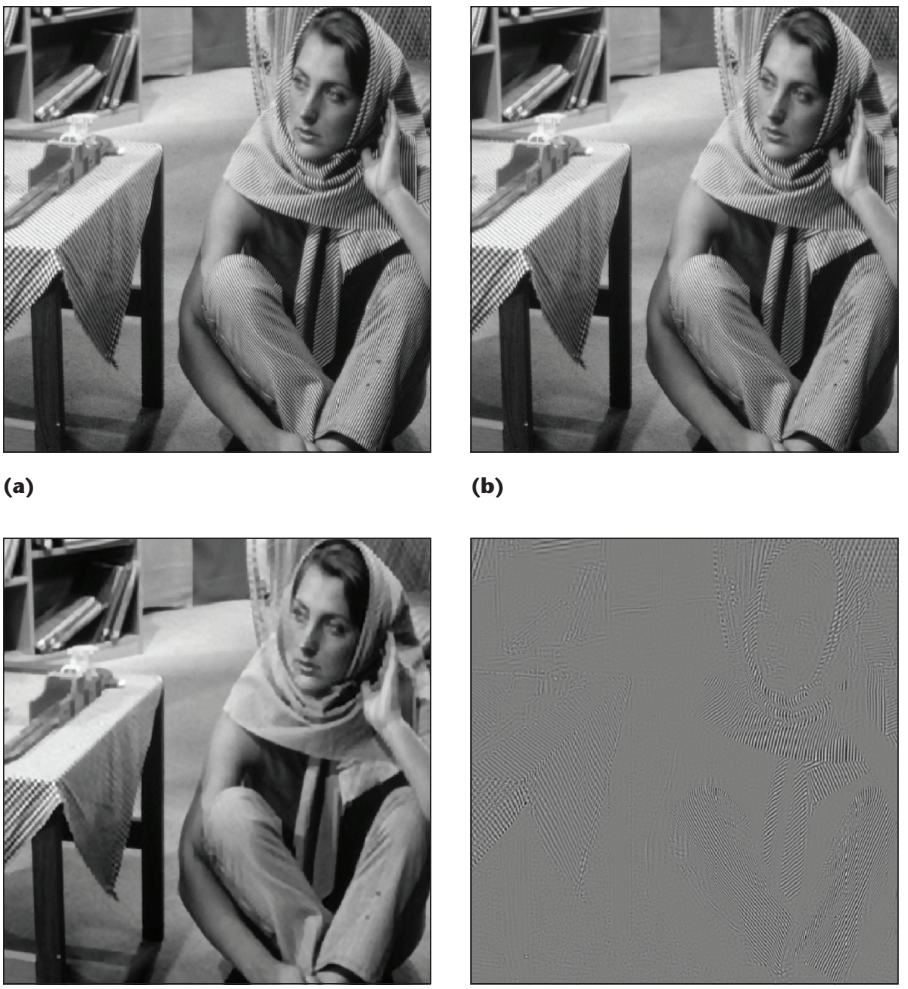


Figure 10. Decomposing the Barbara image. (a) The original Barbara image, (b) morphological component analysis (MCA), (c) the recovered cartoon component, and (d) the reconstructed texture component.

image, the recovered cartoon component, and the reconstructed texture component.

Risers. In this experiment, our goal was to illustrate MCA's usefulness in a real-life application—studying the mechanical properties of composite risers used in the oil industry.

Figure 11a shows a multilayered riser recorded using a digital x-ray camera. The riser's layers include a composite material layer, a layer of steel-made fibers with opposite lay angles, and lead-made markers used as a reference to calibrate the x-ray camera. The structures of interest are the curvilinear fibers. However, the markers that appear as black isotropic structures on the image hide the fiber structures that exist behind them. Our goal was to decompose this x-ray image into two components: one containing the curvilinear fiber structures and the other with the lead-made markers and the smooth background. Therefore, natural candidate dictionaries would be the curvelet transform for the fibers, and the UDWT for the isotropic markers and the piecewise smooth background component. We added the TV penalty to direct the image with the lead markers to fit the piecewise smooth model with $\gamma = 3$. We also denoised the image by setting $\lambda_{\min} = 3\sigma_{\varepsilon}$.

Figure 11 shows the MCA separation. Figure 11b shows the isotropic and background component recovered by MCA, while Figure 11c shows the recovered fibers. MCA clearly managed to eliminate the lead-made markers, while preserving the curvilinear fibers structure and reconstructing the unobserved fibers parts that were behind the markers.

Inpainting

We now describe our reproducible experiments on image inpainting.

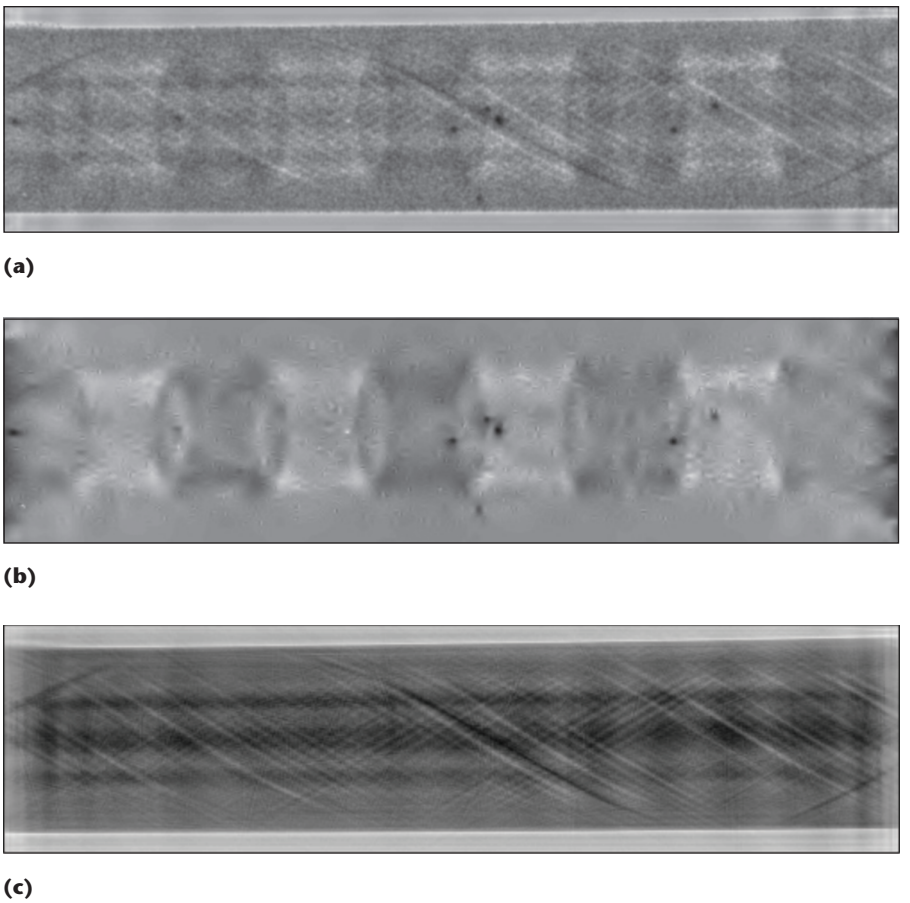


Figure 11. Decomposing a risers image. (a) The original multilayered riser image recorded using a digital x-ray camera, (b) the isotropic and background component recovered through morphological component analysis, and (c) the MCA recovered fibers.

Barbara. Figure 12 shows the Barbara image and its inpainted results for three random masks of 20 percent, 50 percent, and 80 percent missing pixels. The mask's unstructured random form makes the recovery task easier, which is intuitively acceptable using a compressed sensing argument.[16,17] Again, the dictionary contained the curvelet and local DCT transforms. The algorithm not only recovered the geometric part (cartoon), but also performs particularly well inside the textured areas.

Lines and Gaussians. We applied the two inpainting algorithms to a synthetic image that is a composite of three Gaussians and three lines (see Figure 13a). Based on this morphological content, we chose the UDWT and the curvelet transforms as candidate dictionaries. Table 2 summarizes the parameters chosen for each algorithm. Figure 13 shows the masked and filled-in images. Both inpainting algorithms performed well, although the result of Algorithm 2 is somewhat smoother.

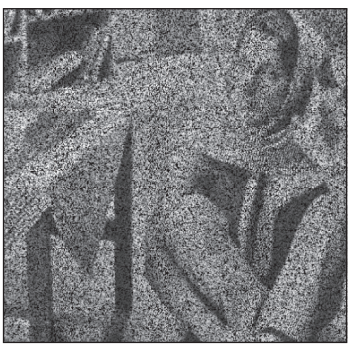
Lena. We repeated the same experiment with the 512×512 Lena image. Figure 14b shows the masked image with 80 percent of the pixels missing. The mask contains large gaps. The dictionary contained the curvelet transform. Table 2 shows the parameters we chose for each algorithm. Despite the challenges entailed in the example's large gaps, both inpainting algorithms performed well. Both managed to recover the most important details of the image that are hardly distinguishable in the masked image. The visual quality is confirmed by PSNR measures, as Figure 14 shows.

Reproducible computational research remains an exception rather than a common practice in most image and signal-processing community literature. Researchers can reproduce our results by running corresponding scripts in the MCALab package. We've also included other exploratory examples—as well as a GUI—in MCALab. We recently released a C++ batch version of

(a)

(b)

(c)



(d)

(e)

(f)

(g)

Figure 12. Inpainting results for the Barbara image. (a) The original image, (b) masked with 20 percent of the pixels missing, (c) inpainted using the morphological component analysis (Algorithm 1) with a peak signal-to-noise ratio of 35.3 decibels, (d) masked with 50 percent missing pixels, (e) inpainted using MCA with a PSNR of 31.7 dB, (f) masked with 80 percent of the pixels missing, and (g) inpainted using MCA with a PSNR of 26.5 dB.

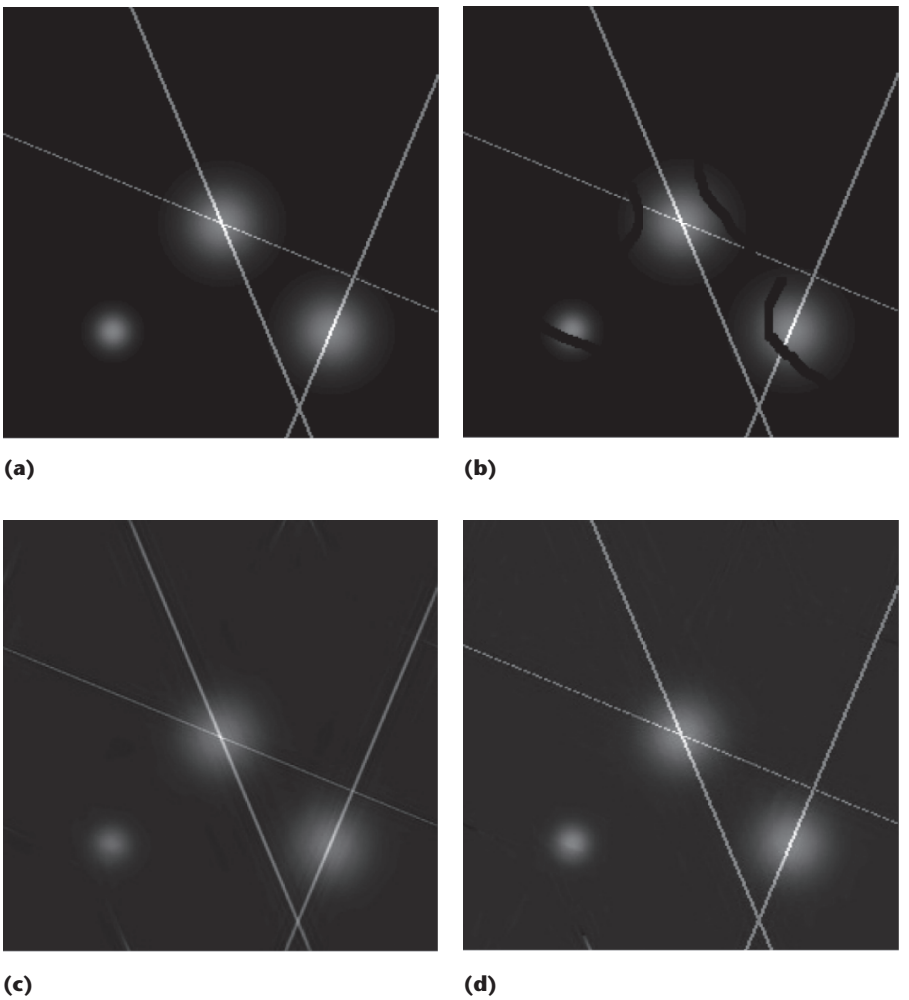


Figure 13. Inpainting lines and Gaussians image. (a) Original lines and Gaussians, (b) masked with a peak signal-to-noise ratio of 30.9 decibels, (c) inpainted using the expectation-maximization inspired algorithm (Algorithm 2) with a PSNR of 32 dB, and (d) inpainted morphological component analysis (Algorithm 1) with a PSNR of 40 dB.

MCALab that is available for download at www.greyc.ensicaen.fr/~jfadili/demos/WaveRestore/downloads/MCALabC1.1.tgz. We're continuing to develop MCALab and warmly welcome all constructive feedback, suggestions, and contributions.

References

1. M. Schwab, N. Karrenbach, and J. Claerbout, "Making Scientific Computations Reproducible," *Computing in Science & Eng.*, vol. 2, no. 6, 2000, pp. 61–67.
2. C.T. Silva and J.E. Tohline, "Computational Provenance," *Computing in Science & Eng.*, vol. 10, no. 3, 2008, pp. 9–10.
3. D.L. Donoho et al., "Reproducible Research in Computational Harmonic Analysis," *Computing in Science & Eng.*, vol. 11, no. 1, 2009, pp. 8–18.
4. M. Barni and F. Perez-Gonzalez, "Pushing Science into Signal Processing," *IEEE Signal Processing,* vol. 22, no. 4, 2005, pp. 119–120.
5. J.L. Starck, M. Elad, and D. Donohom "Redundant Multiscale Transforms and Their Application for Morphological Component Analysis," *Advances in Imaging and Electron Physics,* vol. 132, Elsevier, 2004.
6. M. Elad et al., "Simultaneous Cartoon and Texture Image Inpainting," *Applied and Computational Harmonic Analysis,* vol. 19, no 3, 2005, pp. 340–358.
7. M. J. Fadili, J. L. Starck, and F. Murtagh, "Inpainting and Zooming using Sparse Representations," *Computer J.*, vol. 52, no. 1, 2007, pp. 64–79.
8. J.L. Starck, M. Elad, and D. Donoho, "Image Decomposition via the Combination of Sparse Representations and Variational Approach," *IEEE Trans. Image Processing,* vol. 14, no. 10, 2005, pp. 1570–1582.
9. S.G. Mallat, *A Wavelet Tour of Signal Processing,* 2nd ed. Academic Press, 1998.

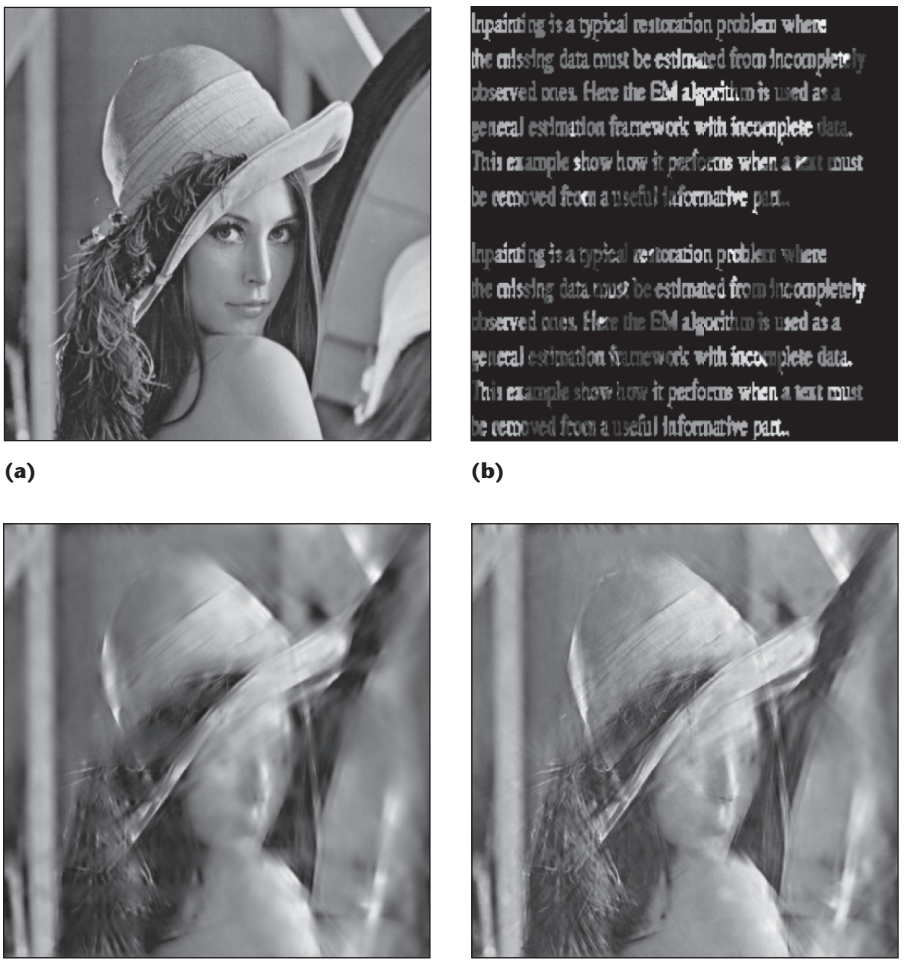


Figure 14. Inpainting the Lena image. (a) The original image, (b) masked with a peak signal-to-noise ratio of 6.44 decibels, (c) inpainted using the expectation-maximization (EM) inspired algorithm (Algorithm 2) with a PSNR of 19.9 decibels, (d) inpainted morphological component analysis (Algorithm 1) with a PSNR of 19.9 dB.

10. E. Candes et al., *Fast Discrete Curvelet Transforms,* tech. report, Applied and Computational Mathematics Dept., CalTech, 2005.
11. E. Candes and D. Donoho, "Ridgelets: The Key to High Dimensional Intermittency?" *Philosophical Trans. Royal Soc. London A,* vol. 357, 1999, pp. 2495–2509.
12. A.M. Bruckstein, D.L. Donoho, and M. Elad, "From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images," *SIAM Rev.,* vol. 51, no. 1, 2009, pp. 34–81.
13. J. Bobin et al., "Morphological Component Analysis: An Adaptive Thresholding Strategy," *IEEE Trans. Image Processing,* vol. 16, no. 11, 2007, pp. 2675–2681.
14. D.L. Donoho and I. Johnstone, "Ideal Spatial Adaptation via Wavelet Shrinkage," *Biometrika,* vol. 81, 1994, pp. 425–455.
15. S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic Decomposition by Basis Pursuit," *SIAM J. Scientific Comp.,* vol. 20, no. 1, 1999, pp. 33–61.
16. E. Candess and T. Tao, "Near Optimal Signal Recovery from Random Projections: Universal Encoding Strategies," *IEEE Trans. Inf. Theory,* vol. 52, no. 12, 2004, pp. 5406–5425.
17. D.L. Donoho, "Compressed Sensing," *IEEE Trans. Inf. Theory,* vol. 52, no. 4, 2006, pp. 1289–1306.

Jalal M. Fadili *is an associate professor of signal and image processing at the National Graduate School of Engineering and Research in Caen, France. His research interests include statistical approaches in signal and image processing, inverse problems, computational harmonic analysis, optimization, and sparse representations as applied in areas such as medical and astronomical imaging. Fadili has a PhD in signal and image processing from the University of Caen. Contact him at Jalal.Fadili@greyc.ensicaen.fr.*

Jean-Luc Starck *is a researcher at Commission for Atomic Energy (CEA), France, where he is leader of the*

multiresolution project and a core team member of the Planck European Space Agency project. His research interests include image processing, statistical methods in astrophysics, and cosmology. He is an expert in multiscale methods such as wavelets and curvelets, and is author of Image Processing and Data Analysis: The Multiscale Approach *(Cambridge University Press, 1998) and* Astronomical Image and Data Analysis *(2nd ed., Springer, 2006). Starck has a PhD in image processing from the University Nice-Sophia Antipolis, France. Contact him at jstarck@cea.fr.*

***Michael Elad** is an associate professor in the Department of Computer Science at The Technion, Israel Institute of Technology, Haifa. His research interests are in signal and image processing, focusing on inverse problems, sparse representations, and overcomplete transforms. Elad has a DSc in electrical engineering from The Technion. He is an associate editor of* IEEE Transactions on Image Processing *and SIAM's* Imaging Sciences *journal. Contact him at elad@cs.technion.ac.il.*

***David L. Donoho** is a professor of statistics at Stanford University. His research interests are in harmonic analysis, image representation, and mathematical statistics. Donoho has a PhD in statistics from Harvard University. He is a member of the US National Academy of Sciences and a fellow of the American Academy of Arts and Sciences. Contact him at donoho@stanford.edu.*

cn *Selected articles and columns from IEEE Computer Society publications are also available for free at http://ComputingNow.computer.org.*

ADVERTISER INFORMATION JANUARY/FEBRUARY 2010

Advertising Personnel
Marion Delaney
IEEE Media, Advertising Dir.
Phone: +1 415 863 4717
Email: md.ieeemedia@ieee.org

Marian Anderson
Sr. Advertising Coordinator
Phone: +1 714 821 8380
Fax: +1 714 821 4010
Email: manderson@computer.org

Sandy Brown
Sr. Business Development Mgr.
Phone: +1 714 821 8380
Fax: +1 714 821 4010
Email: sb.ieeemedia@ieee.org

Advertising Sales Representatives

Recruitment:

Mid Atlantic
Lisa Rinaldo
Phone: +1 732 772 0160
Fax: +1 732 772 0164
Email: lr.ieeemedia@ieee.org

New England
John Restchack
Phone: +1 212 419 7578
Fax: +1 212 419 7589
Email: j.restchack@ieee.org

Southeast
Thomas M. Flynn
Phone: +1 770 645 2944
Fax: +1 770 993 4423
Email: flynntom@mindspring.com

Midwest/Southwest
Darcy Giovingo
Phone: +1 847 498 4520
Fax: +1 847 498 5911
Email: dg.ieeemedia@ieee.org

Northwest/Southern CA
Tim Matteson
Phone: +1 310 836 4064
Fax: +1 310 836 4067
Email: tm.ieeemedia@ieee.org

Japan
Tim Matteson
Phone: +1 310 836 4064
Fax: +1 310 836 4067
Email: tm.ieeemedia@ieee.org

Europe
Hilary Turnbull
Phone: +44 1875 825700
Fax: +44 1875 825701
Email: impress@impressmedia.com

Product:

US East
Dawn Becker
Phone: +1 732 772 0160
Fax: +1 732 772 0164
Email: db.ieeemedia@ieee.org

US Central
Darcy Giovingo
Phone: +1 847 498 4520
Fax: +1 847 498 5911
Email: dg.ieeemedia@ieee.org

US West
Lynne Stickrod
Phone: +1 415 931 9782
Fax: +1 415 931 9782
Email: ls.ieeemedia@ieee.org

Europe
Sven Anacker
Phone: +49 202 27169 11
Fax: +49 202 27169 20
Email: sanacker@intermediapartners.de

Get the building blocks you need.

Take your career to the next level in software development, systems design, and engineering with:

- Article collections from the IEEE Computer Society
- Materials from Harvard Business School Publishing
- Computer discounts
- Online courses and certifications

Our experts. Your future.

www.computer.org/buildyourcareer