RobOMP: Robust variants of Orthogonal Matching Pursuit for sparse representations

Carlos A Loza Corresp. [1]

[1] Department of Mathematics, Universidad San Francisco de Quito, Quito, Ecuador

Corresponding Author: Carlos A Loza
Email address: cloza@usfq.edu.ec

Sparse coding aims to find a parsimonious representation of an example given an observation matrix or dictionary. In this regard, Orthogonal Matching Pursuit (OMP) provides an intuitive, simple and fast approximation of the optimal solution. However, its main building block is anchored on the minimization of the Mean Squared Error cost function (MSE). This approach is only optimal if the errors are distributed according to a Gaussian distribution without samples that strongly deviate from the main mode, i.e. outliers. If such assumption is violated, the sparse code will likely be biased and performance will degrade accordingly. In this paper, we introduce five robust variants of OMP (RobOMP) fully based on the theory of M-Estimators under a linear model. The proposed framework exploits efficient Iteratively Reweighted Least Squares (IRLS) techniques to mitigate the effect of outliers and emphasize the samples corresponding to the main mode of the data. This is done adaptively via a learned weight vector that models the distribution of the data in a robust manner. Experiments on synthetic data under several noise distributions and image recognition under different combinations of occlusion and missing pixels thoroughly detail the superiority of RobOMP over MSE-based approaches and similar robust alternatives. We also introduce a denoising framework based on robust, sparse and redundant representations that open the door to potential further applications of the proposed techniques. The five different variants of RobOMP do not require parameter tuning from the user and, hence, constitute principled alternatives to OMP.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27482v1 | CC BY 4.0 Open Access | rec: 14 Jan 2019, publ: 14 Jan 2019

RobOMP: Robust Variants of Orthogonal Matching Pursuit for Sparse Representations

Carlos A. Loza

Department of Mathematics, Universidad San Francisco de Quito, Quito, Ecuador

Corresponding author:
Carlos A. Loza

Email address: cloza@usfq.edu.ec

ABSTRACT

Sparse coding aims to find a parsimonious representation of an example given an observation matrix or dictionary. In this regard, Orthogonal Matching Pursuit (OMP) provides an intuitive, simple and fast approximation of the optimal solution. However, its main building block is anchored on the minimization of the Mean Squared Error cost function (MSE). This approach is only optimal if the errors are distributed according to a Gaussian distribution without samples that strongly deviate from the main mode, i.e. outliers. If such assumption is violated, the sparse code will likely be biased and performance will degrade accordingly. In this paper, we introduce five robust variants of OMP (RobOMP) fully based on the theory of M–Estimators under a linear model. The proposed framework exploits efficient Iteratively Reweighted Least Squares (IRLS) techniques to mitigate the effect of outliers and emphasize the samples corresponding to the main mode of the data. This is done adaptively via a learned weight vector that models the distribution of the data in a robust manner. Experiments on synthetic data under several noise distributions and image recognition under different combinations of occlusion and missing pixels thoroughly detail the superiority of RobOMP over MSE–based approaches and similar robust alternatives. We also introduce a denoising framework based on robust, sparse and redundant representations that open the door to potential further applications of the proposed techniques. The five different variants of RobOMP do not require parameter tuning from the user and, hence, constitute principled alternatives to OMP.

INTRODUCTION

Sparse modeling is a learning framework with relevant applications in areas where parsimonious representations are considered advantageous, such as signal processing, machine learning, and computer vision. Dictionary learning, image denoising, image super–resolution, visual tracking and image classification constitute some of the most celebrated applications of sparse modeling (Aharon et al., 2006; Elad and Aharon, 2006; Mallat, 2008; Wright et al., 2009; Elad et al., 2010; Xu et al., 2011). Strictly speaking, sparse modeling refers to the entire process of designing and learning a model, while sparse coding, sparse representation, or sparse decomposition is an inference process—estimation of the latent variables of such model. The latter formally emerged as a machine learning adaptation of the sparse coding scheme found in the mammalian primary visual cortex (Olshausen and Field, 1996).

The sparse coding problem is inherently combinatorial and, therefore, intractable in practice. Thus, classic solutions involve either greedy approximations or relaxations of the original ℓ_0-pseudonorm. Examples of the former family of algorithms include Matching Pursuit (MP) and all of its variants (Mallat and Zhang, 1993), while Basis Pursuit (Chen et al., 2001) and Lasso (Tibshirani, 1996) are the archetypes of the latter techniques. Particularly, Orthogonal Matching Pursuit (OMP) is usually regarded as more appealing due to its efficiency, convergence properties, and simple, intuitive implementation based on iterative selection of the most correlated predictor to the current residual and batch update of the entire active set (Tropp and Gilbert, 2007).

The success of OMP is confirmed by the many variants proposed in the literature. Wang et al. (2012) introduced Generalized OMP (GOMP) where more than one predictor or atom (i.e. columns of the

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27482v1 | CC BY 4.0 Open Access | rec: 14 Jan 2019, publ: 14 Jan 2019

measurement matrix or dictionary) are selected per iteration. Regularized OMP (ROMP) exploits a predefined regularization rule (Needell and Vershynin, 2010), while CoSaMP incorporates additional pruning steps to refine the estimate recursively (Needell and Tropp, 2009). The implicit foundation of the aforementioned variants—and, hence, of the original OMP—is optimization based on Ordinary Least Squares (OLS), which is optimal under a Mean Squared Error (MSE) cost function or, equivalently, a Gaussian distribution of the errors. Any deviation from such assumptions, e.g. outliers, impulsive noise or non–Gaussian additive noise, would result in biased estimations and performance degradation in general.

Wang et al. (2017) proposed Correntropy Matching Pursuit (CMP) to mitigate the detrimental effect of outliers in the sparse coding process. Basically, the Correntropy Induced Metric replaces the MSE as the cost function of the iterative active set update of OMP. Consequently, the framework becomes robust to outliers and impulsive noise by weighing the input samples according to a Gaussian kernel. The resulting non–convex performance surface is optimized via the Half–Quadratic (HQ) technique to yield fast, iterative approximations of local optima (Geman and Yang, 1995; Nikolova and Ng, 2005). Even though the algorithm is successful in alleviating the effect of outliers in practical applications, the main hyperparameter—the Gaussian kernel bandwidth—is chosen empirically with no theoretical validation. With this mind, we propose a generalization of CMP by reformulating the active set update under the lens of robust linear regression; specifically, we exploit the well known and developed theory of M–Estimators (Andersen, 2008; Huber, 2011) to devise five different robust variants of OMP: RobOMP. Each one utilizes validated hyperparameters that guarantee robustness up to theoretical limits. In addition, the HQ optimization technique is reduced to the Iteratively Reweighted Least Squares (IRLS) algorithm in order to provide an intuitive and effective implementation while still enjoying the weighing nature introduced in CMP.

For instance, Fig. 1 illustrates the estimated error in a 50–dimensional observation vector with a 10% rate of missing samples (set equal to zero). While Tukey–Estimator–based–OMP practically collapses the error distribution after 10 decompositions, the OMP counterpart still leaves a remnant that derives in suboptimal sparse coding. Moreover, RobOMP provides an additional output that effectively weighs the components of the input space in a [0,1] scale. In particular, the missing samples are indeed assigned weights close to zero in order to alleviate their effect in the estimation of the sparse decomposition. In terms of computer vision, Fig. 2 compares denoising mechanisms based on sparse representations. Only CMP and the proposed methods are able to not only retrieve high–fidelity reconstructions, but also provide a weight vector that explicitly discriminates between pixels from the original, uncorrupted image and outliers (image occlusion in this case). Though the results are visually similar, the residual errors differ significantly in terms of dynamic range and ℓ_2–norm; this idea is further exploited for image recognition in the sections to come.

We present three different sets of results to validate the proposed robust, sparse inference framework. First, synthetic data with access to ground truth (support of the representation) highlights the robustness of the estimators under several types of noise, such as additive non–Gaussian densities and instance–based degradation (e.g. missing samples and impulsive noise). Then, a robust sparse representation–based classifier (RSRC) is developed for image recognition under missing pixels and occlusion scenarios. The results outperform the OMP–based variants and the CMP–based classifier (CMPC) for several cases. Lastly, preliminary results on image denoising via sparse and redundant representations over overcomplete dictionaries are presented with the hope of exploiting RobOMP in the future for image denoising under non–Gaussian additive noise. The rest of the paper is organized as follows: Section 2 details the state of the art and related work concerning greedy approximations to the sparse coding problem. Section 3 introduces the theory, rationale, and algorithms regarding M–estimation–based Robust OMP: RobOMP. Section 4 details the results using synthetic data and popular digital image databases, while Section 5 discusses more in–depth technical concepts, analyzes the implications of the proposed framework, and offers potential further work. Lastly, Section 6 concludes the paper.

STATE OF THE ART AND RELATED WORK

Let $\mathbf{y} \in \mathbb{R}^m$ be a measurement vector with an ideal, noiseless, sparse representation, $\mathbf{x}_0 \in \mathbb{R}^n$, with respect to the measurement matrix (also known as dictionary), $\mathbf{D} \in \mathbb{R}^{m \times n}$. The matrix $\mathbf{D}$ is usually overcomplete, i.e. $m < n$, to promote sparse decompositions. In practice, $\mathbf{y}$ is affected by a noise component, $\mathbf{n} \in \mathbb{R}^m$.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27482v1 | CC BY 4.0 Open Access | rec: 14 Jan 2019, publ: 14 Jan 2019

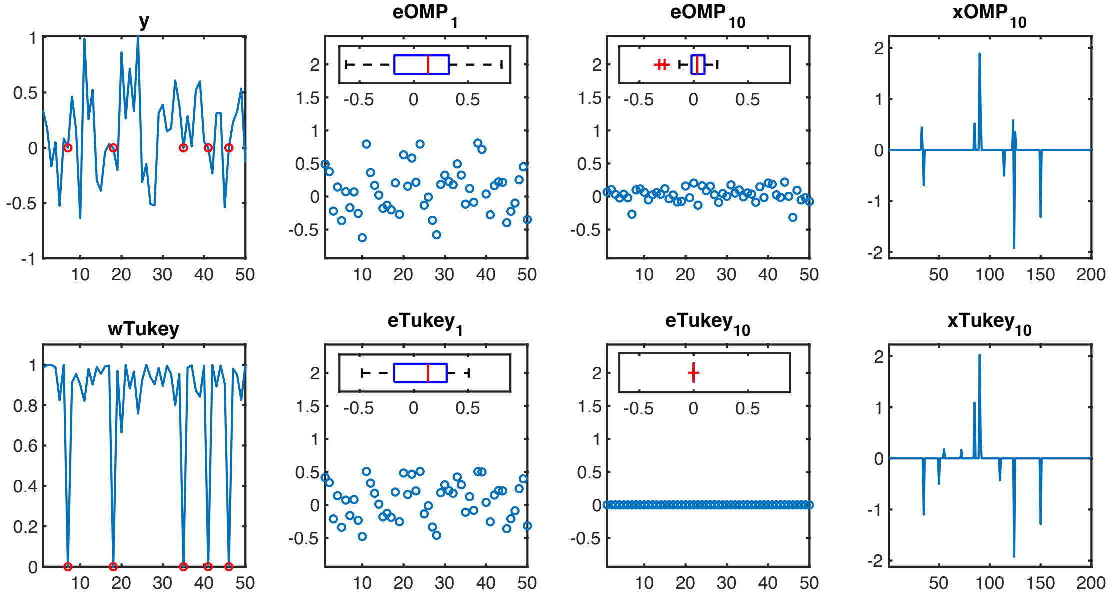


Figure 1. Illustration of the robustness of the proposed method. $y \in \mathbb{R}^{50}$ constitutes an observation vector with 5 missing samples (set to zero, marked in red). $eOMP_1$ and $eOMP_{10}$ are the resulting errors after the first and tenth iteration of OMP (with corresponding box plots as insets), respectively. $xOMP_{10}$ is the final estimated sparse decomposition after 10 OMP iterations. Their RobOMP counterparts (Tukey estimator) reduce more aggressively the dynamic range of the errors until almost collapsing to a delta distribution; this results in optimal sparse coding. $wTukey$ is the learned weight vector that assigns values close to one to values around the main mode of the data and small weights to potential outliers (red marks). $K = 10$.

This results in the following constrained, linear, additive model:

$$\mathbf{y} = \mathbf{D}\mathbf{x}_0 + \mathbf{n} \quad \text{s.t.} \quad ||\mathbf{x}_0||_0 = K_0 \tag{1}$$

where K_0 indicates the support of the sparse decomposition and $||\cdot||_0$ represents the ℓ_0–pseudonorm, i.e. number of non–zero components in $\mathbf{x}_0$. The sparse coding framework aims to estimate $\mathbf{x}_0$ given the measurement vector and matrix plus a sparsity constraint.

MSE–based OMP

Orthogonal Matching Pursuit (Tropp and Gilbert, 2007) attempts to find the locally optimal solution by iteratively estimating the most correlated atom in $\mathbf{D}$ to the current residual. In particular, OMP initializes the residual $\mathbf{r}_0 = \mathbf{y}$, the set containing the indices of the atoms that are part of the decomposition (an active set) $\Lambda_0 = \emptyset$, and the iteration $k = 1$. In the kth iteration, the algorithm finds the predictor most correlated to the current residual:

$$\lambda_k = \underset{i \in \Omega}{\operatorname{argmax}} |\langle \mathbf{r}_{k-1}, \mathbf{d}_i \rangle| \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operator, $\mathbf{d}_i$ represents the ith column of $\mathbf{D}$, and $\Omega = \{1, 2, \cdots, n\}$. The resulting atom is added to the active set via Λ, i.e.:

$$\Lambda_k = \Lambda_{k-1} \cup \{\lambda_k\} \tag{3}$$

The next step is the major refinement of the original Matching Pursuit algorithm (Mallat and Zhang, 1993)—instead of updating the sparse decomposition one component at the time, OMP updates all the coefficients corresponding to the active set at once according to a MSE criterion

$$\mathbf{x}_k = \underset{\mathbf{x} \in \mathbb{R}^n, \operatorname{supp}(\mathbf{x}) \subset \Lambda_k}{\operatorname{argmin}} ||\mathbf{y} - \mathbf{D}\mathbf{x}||_2 \tag{4}$$

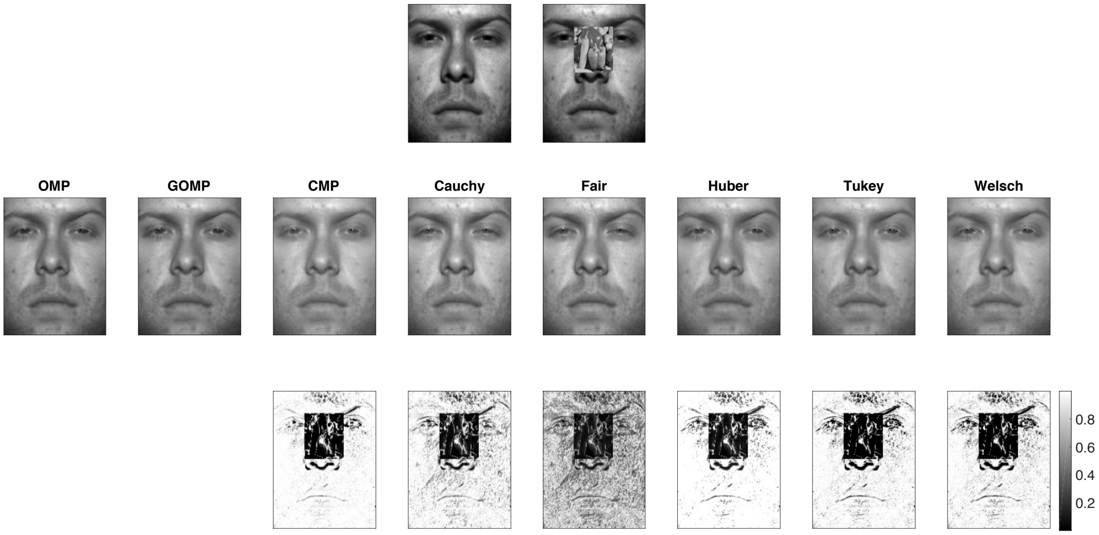


Figure 2. Image reconstruction (denoising) based on sparse decompositions. First row shows the original and corrupted (image occlusion) images. Second row depicts the reconstructions corresponding to OMP, GOMP, CMP and the five variants of RobOMP introduced in this work. Third row details the weight maps from CMP and RobOMP according to the resulting weight vector in the scale [0,1] where small values are assigned to occlusion pixels and large weights correspond to elements from the uncorrupted image.

where $\text{supp}(\mathbf{x})$ is the support set of vector $\mathbf{x}$. Equation (4) can be readily solved via OLS or Linear Regression where the predictors are the columns of $\mathbf{D}$ indexed by Λ_k and the response is the measurement vector $\mathbf{y}$. Stopping criterion for OMP typically include a set number of iterations or compliance with a set minimum error of the residue. In the end, the estimated sparse code, $\mathbf{x}$, is set as the last $\mathbf{x}_k$ obtained.

One of the major advantages of OMP over MP is the guarantee of convergence after K_0 iterations. In practice, however, the true sparsity pattern is unknown and the total number of iterations, K, is treated as a hyperparameter. For a detailed analysis regarding convergence and recovery error bounds of OMP, see Donoho et al. (2006). A potential drawback of OMP is the extra computational complexity added by the OLS solver. Specifically, each incremental update of the active set affects the time complexity of the algorithm in a polynomial fashion: $\mathcal{O}(k^2 n + k^3)$ where k is the current iteration.

Generalized Orthogonal Matching Pursuit (Wang et al., 2012) refines OMP by selecting N_0 atoms per iteration. If the indices of the active set columns in the kth iteration are denoted as $J_k[1], J_k[2], \ldots, J_k[N_0]$, then $J_k[j]$ can be defined recursively:

$$J_k[j] = \underset{i \in \Omega \setminus \{J_k[1], \ldots, J_k[j-1]\}}{\text{argmax}} |\langle \mathbf{r}_{k-1}, \mathbf{d}_i \rangle|, \quad 1 \leq j \leq N_0 \tag{5}$$

The index set $\{J_k[j]\}_{j=1}^{N_0}$ is then added to Λ_k and, likewise OMP, GOMP exploits an OLS solver to update the current active set. Both OMP and GOMP obtain locally optimal solutions under the assumption of Gaussianity (or Normality) of the errors. Yet, if such restriction is violated (e.g. by the presence of outliers), the estimated sparse code, $\mathbf{x}$, will most likely be biased.

CMP

The main drawback of MSE–based cost functions is its weighing nature in terms of influence and importance assigned to the available samples. In particular, MSE considers every sample as equally important and assigns a constant weight equal to one to all the inputs. Wang et al. (2017) proposed exploiting Correntropy (Liu et al., 2007) instead of MSE as an alternative cost function in the greedy sparse coding framework. Basically, the novel loss function utilizes the Correntropy Induced Metric (CIM) to weigh samples according to a Gaussian kernel $g_\sigma(t) = \exp\left(-t^2/2\sigma^2\right)$, where σ, the kernel bandwidth, modulates the norm the CIM will mimic, e.g. for small σ, the CIM behaves similar to the ℓ_0-pseudonorm (aggressive non–linear weighing), if σ increases, CIM will mimic the ℓ_1–norm (moderate linear weighing), and, lastly, for large σ, the resulting cost function defaults to MSE, i.e. constant weighing for all inputs. The main conclusion here is that the CIM, unlike MSE, is robust to outliers for a principled choice of σ. This outcome easily generalizes for non–Gaussian environments with long–tailed distributions on the errors.

Correntropy Matching Pursuit (CMP) exploits the CIM robustness to update the active set in the sparse coding solver. The algorithm begins in a similar fashion as OMP, i.e. $\mathbf{r}_0 = \mathbf{y}$, $\Lambda_0 = \emptyset$, and $k = 1$. Then, instead of the MSE–based update of Equation (4), CMP proceeds to minimize the following CIM–based expression:

$$\mathbf{x}_k = \underset{\mathbf{x}\in\mathbb{R}^n, \text{supp}(\mathbf{x})\subset\Lambda_k}{\text{argmin}} L_\sigma(\mathbf{y} - \mathbf{D}\mathbf{x}) \tag{6}$$

where $L_\sigma(\mathbf{e}) = \frac{1}{m}\sum_{i=1}^{m} \sigma^2(1 - g_\sigma(\mathbf{e}[i]))$ is the simplified version (without constant terms independent of $\mathbf{e}$) of the CIM loss function and $\mathbf{e}[i]$ is the ith entry of the vector $\mathbf{e}$. The Half–Quadratic (HQ) technique is utilized to efficiently optimize the invex CIM cost function (Geman and Yang, 1995; Nikolova and Ng, 2005). The result is a local minimum of Equation (6) alongside a weight vector $\mathbf{w}$ that indicates the importance of the components of the observation vector $\mathbf{y}$:

$$\mathbf{w}^{(t+1)}[i] = g_\sigma\left(\mathbf{y}[i] - \left(\mathbf{D}\mathbf{x}^{(t)}\right)[i]\right), \quad i = 1, 2, \ldots, m \tag{7}$$

where t is the iteration in the HQ subroutine. In short, the HQ optimization performs block coordinate descent to separately optimize the sparse code, $\mathbf{x}$, and the weight vector, $\mathbf{w}$, in order to find local optima. The hyperparameter σ is iteratively updated without manual selection according to the following heuristic:

$$\sigma^{(t+1)} = \left(\frac{1}{2m}\left|\left|\mathbf{y} - \mathbf{D}\mathbf{x}^{(t+1)}\right|\right|_2^2\right)^{\frac{1}{2}} \tag{8}$$

In Wang et al. (2017), the authors throughly illustrate the advantage of CMP over many MSE–based variants of OMP when dealing with non-Gaussian error distributions and outliers in computer vision applications. And even though they mention the improved performance of the algorithm when σ is iteratively updated versus manual selection scenarios, they fail to explain the particular heuristic behind Equation (8) or its statistical validity. In addition, the HQ optimization technique is succinctly reduced to a weighted Least Squares problem than can be solved explicitly. Therefore, more principled techniques that exploit weighted Least Squares and robust estimators for linear regression can easily provide the needed statistical validity, while at the same time, generalize the concepts of CMP under the umbrella of M–estimators.

ROBUST ORTHOGONAL MATCHING PURSUIT

MSE–based OMP appeals to OLS solvers to optimize Equation (4). In particular, let $\Phi \in \mathbb{R}^{m\times k}$ correspond to the *active* atoms in the dictionary $\mathbf{D}$ at iteration k, i.e. $\Phi = [\mathbf{d}_{\Lambda_k[1]}, \mathbf{d}_{\Lambda_k[2]}, \cdots, \mathbf{d}_{\Lambda_k[k]}]$, and $\beta \in \mathbb{R}^k$ be the vector corresponding to the coefficients that solve the following regression problem:

$$\mathbf{y} = \Phi\beta + \mathbf{e} \tag{9}$$

where $\mathbf{e}$ is an error vector with independent components identically distributed according to a zero–mean Normal density ($\mathbf{e}[i] \sim \mathcal{N}(0, \sigma^2)$). Then, the least squares regression estimator, $\hat{\beta}$, is the maximum likelihood estimator for β under a Gaussian density prior, i.e.:

$$\hat{\beta} = \underset{\beta}{\text{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\mathbf{e}[i]^2}{2\sigma^2}\right) = \underset{\beta}{\text{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathbf{y}[i] - (\Phi\beta)[i])^2}{2\sigma^2}\right) \tag{10}$$

which is equivalent to maximizing the logarithm of (10) over β:

$$\hat{\beta} = \underset{\beta}{\text{argmax}} \sum_{i=1}^{m}\left(-\frac{1}{2}\ln(2\pi\sigma^2) - \frac{\mathbf{e}[i]^2}{2\sigma^2}\right) = \underset{\beta}{\text{argmin}} \sum_{i=1}^{m}\left(\frac{\mathbf{e}[i]^2}{2}\right) \tag{11}$$

Since σ is assumed as constant, $\hat{\beta}$ is the estimator that minimizes the sum of squares of the errors, or residuals. Hence, the optimal solution is derived by classic optimization theory giving rise to the well known normal equations and OLS estimator:

$$\begin{aligned}\sum_{i=1}^{m} \mathbf{e}[i]^2 &= \mathbf{e}^T\mathbf{e}\\ &= (\mathbf{y}-\Phi\beta)^T(\mathbf{y}-\Phi\beta)\\ &= \mathbf{y}^T\mathbf{y}-\mathbf{y}^T\Phi\beta-\beta^T\Phi^T\mathbf{y}+\beta^T\Phi^T\Phi\beta\end{aligned}$$

At the minimum:

$$\begin{aligned}\frac{\partial}{\partial\beta}\sum_{i=1}^{m}\mathbf{e}[i]^2 = 0 &= \frac{\partial}{\partial\beta}(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\Phi\beta-\beta^T\Phi^T\mathbf{y}+\beta^T\Phi^T\Phi\beta)\\ &= 0-\Phi^T\mathbf{y}-\Phi^T\mathbf{y}+2\left(\Phi^T\Phi\right)\beta\end{aligned}$$

Consequently when $\Phi^T\Phi$ is non–singular, the optimal estimated coefficients vector has a closed–form solution equal to:

$$\hat{\beta}_{\text{OLS}} = \hat{\beta} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y} \tag{12}$$

which is optimal under a Gaussian distribution of the errors. If such assumption is no longer valid due to outliers or non–Gaussian environments, M–Estimators provide a suitable alternative to the estimation problem.

M–Estimators

If the errors are not normally distributed, the estimator of (12) will be suboptimal. Hence, a different function is utilized to model the statistical properties of the errors. Following the same premises of independence and equivalence of the optimum under the log–transform, the following estimator arises:

$$\hat{\beta}_{\text{M–Est}} = \underset{\beta}{\operatorname{argmin}}\sum_{i=1}^{m}\rho\left(\frac{\mathbf{e}[i]}{s}\right) = \underset{\beta}{\operatorname{argmin}}\sum_{i=1}^{m}\rho\left(\frac{(\mathbf{y}[i]-(\Phi\beta)[i])}{s}\right) \tag{13}$$

where $\rho(e)$ is a continuous, symmetric function (also known as the objective function) with a unique minimum at $e=0$ (Andersen, 2008). Clearly, $\rho(e)$ reduces to half the sum of squared errors for the Gaussian case. s is an estimate of the scale of the errors in order to guarantee scale–invariance of the solution. The usual standard deviation cannot be used for s due to its non–robustness; thus, a suitable alternative is usually the “re–scaled MAD”:

$$s = 1.4826 \times MAD \tag{14}$$

where the MAD (median absolute deviation) is highly resistant to outliers with a breakdown point (BDP) of 50%, as it is based on the median of the errors ($\tilde{\mathbf{e}}$) rather than their mean (Andersen, 2008):

$$MAD = \text{median}|\mathbf{e}[i]-\tilde{\mathbf{e}}| \tag{15}$$

The re–scaling factor of 1.4826 guarantees that, for large sample sizes and $\mathbf{e}[i]\sim\mathcal{N}(0,\sigma^2)$, s reduces to the population standard deviation (Hogg, 1979). M–Estimation then, likewise OLS, finds the optimal coefficients vector via partial differentiation of (13) with respect to each of the k parameters in question, resulting in a system of k equations:

$$\sum_{i=1}^{m}\Phi_{ij}\psi\left(\frac{\mathbf{y}[i]-\phi_i^T\beta}{s}\right) = \sum_{i=1}^{m}\Phi_{ij}\psi\left(\frac{\mathbf{e}[i]}{s}\right) = 0, \quad j=1,2,\ldots,k \tag{16}$$

where ϕ_i represents the ith row of the matrix Φ while Φ_{ij} accesses the jth component of the ith row of Φ. $\psi\left(\frac{\mathbf{e}[i]}{s}\right) = \frac{\partial\rho}{\partial\frac{\mathbf{e}[i]}{s}}$ is known as the score function while the weight function is derived from it as:

$$\mathbf{w}[i] = \mathbf{w}\left(\frac{\mathbf{e}[i]}{s}\right) = \frac{\psi\left(\frac{\mathbf{e}[i]}{s}\right)}{\frac{\mathbf{e}[i]}{s}} \tag{17}$$

Substituting Equation (17) into (16) results in:

$$\sum_{i=1}^{m}\Phi_{ij}\mathbf{w}[i]\frac{\mathbf{e}[i]}{s}=\sum_{i=1}^{m}\Phi_{ij}\mathbf{w}[i](\mathbf{y}[i]-\phi_i^T\beta)\frac{1}{s}=0 \qquad j=1,2,\ldots,k$$

$$\sum_{i=1}^{m}\Phi_{ij}\mathbf{w}[i](\mathbf{y}[i]-\phi_i^T\beta)=0 \qquad j=1,2,\ldots,k$$

$$\sum_{i=1}^{m}\Phi_{ij}\mathbf{w}[i]\phi_i\beta=\sum_{i=1}^{m}\Phi_{ij}\mathbf{w}[i]\mathbf{y}[i] \qquad j=1,2,\ldots,k \tag{18}$$

which can be succinctly reduced in matrix form as:

$$\Phi^T\mathbf{W}\Phi\beta=\Phi^T\mathbf{W}\mathbf{y} \tag{19}$$

by defining the weight matrix, $\mathbf{W}$, as a square diagonal matrix with non–zero elements equal to the entries in $\mathbf{w}$, i.e.: $\mathbf{W}=\mathrm{diag}(\{\mathbf{w}[i]:i=1,2,\ldots,m\})$. Lastly, if $\Phi^T\mathbf{W}\Phi$ is well-conditioned, the closed form solution of the robust M–Estimator is equal to:

$$\hat{\beta}_{\text{M–Est}}=(\Phi^T\mathbf{W}\Phi)^{-1}\Phi^T\mathbf{W}\mathbf{y} \tag{20}$$

Equation (20) resembles its OLS counterpart (Equation (12)), except for the the addition of the matrix $\mathbf{W}$ that weights the entries of the observation vector and mitigates the effect of outliers according to a linear fit. A wide variety of objective functions (and in turn, weight functions) have been proposed in the literature (for a through review, see Zhang (1997)). For the present study, we will focus on five different variants that are detailed in Table 1. Every M–Estimator weighs its entries according to a symmetric, decaying function that assigns large weights to errors in the vicinity of zero and small coefficients to gross contributions. Consequently, the estimators downplay the effect of outliers and samples, in general, that deviate from the main mode of the residuals.

Solving the M-Estimation problem is not as straightforward as the OLS counterpart. In particular, Equation (20) assumes the optimal $\mathbf{W}$ is readily available, which, in turn, depends on the residuals, which, again, depend on the coefficient vector. In short, the optimization problem for M–Estimators can be posed as finding both $\hat{\beta}_{\text{M–Est}}$ and $\hat{\mathbf{w}}_{\text{M–Est}}$ that minimize Equation (13). Likewise HQ, the common approach is to perform block coordinate descent on the cost function with respect to each variable individually until local optima are found. In the robust regression literature, this optimization procedure is the well known *Iteratively Reweighted Least Squares* or IRLS (Andersen, 2008). The procedure is detailed in Algorithm 1. In particular, the routine runs for either a fixed number of iterations or until the estimates change by less than a selected threshold between iterations. The main hyperparameter is the choice of the robust M–Estimator alongside its corresponding parameter c. However, it is conventional to select the value that provides a 95% asymptotic efficiency on the standard Normal distribution (Zhang, 1997). Throughout this work, we exploit such optimal values to avoid parameter tuning by the user (see Table 2). In this way, the influence of outliers and non-Gaussian errors are expected to be diminished in the OMP update stage of the coefficients corresponding to the active set.

M–Estimators–based OMP

Here, we combine the ideas behind greedy approximations to the sparse coding problem and robust M–Estimators; the result is RobOMP or Robust Orthogonal Matching Pursuit. We propose five variants based on five different M–Estimators (Table 1). We refer to each RobOMP alternative according to its underlaying M–Estimator; for instance, Fair–Estimator–based–OMP is simply referred to as *Fair*. As with OMP, the only hyperparameter is the stopping criterion: either K as the maximum number of iterations (i.e. sparseness of the solution), or ε, defined as a threshold on the error norm.

For completeness, Algorithm 2 details the RobOMP routine for the case of set maximum number of iterations (the case involving ε is straightforward). Three major differences are noted:

1. The robust M–Estimator–based update stage of the active set is performed via IRLS,
2. The updated residuals are computed considering the weight vector $\hat{\mathbf{w}}_k$ from IRLS, and
3. The weight vector constitutes an additional output of RobOMP.

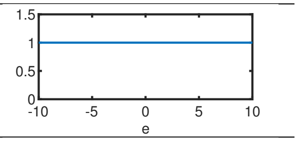
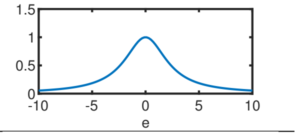
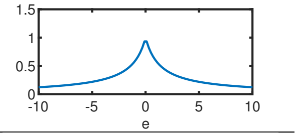
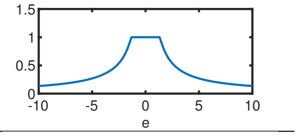
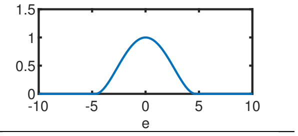
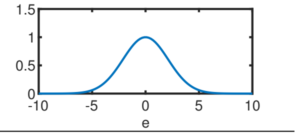
Type	$\rho(e)$	$w(e)$	$w(e)$
OLS	$\frac{1}{2}e^2$	1	
Cauchy	$\frac{c^2}{2}\log\left(1+(\frac{e}{c})^2\right)$	$\frac{1}{1+(e/c)^2}$	
Fair	$c^2\left(\frac{\lvert e\rvert}{c}-\log\left(1+\frac{\lvert e\rvert}{c}\right)\right)$	$\frac{1}{1+\lvert e\rvert/c}$	
Huber $\begin{cases} \text{if } \lvert e\rvert \leq c \\ \text{if } \lvert e\rvert \geq c \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ c\left(\lvert e\rvert-\frac{c}{2}\right) \end{cases}$	$\begin{cases} 1 \\ \frac{c}{\lvert e\rvert} \end{cases}$	
Tukey $\begin{cases} \text{if } \lvert e\rvert \leq c \\ \text{if } \lvert e\rvert > c \end{cases}$	$\begin{cases} \frac{c^2}{6}\left(1-\left(1-\left(\frac{e}{c}\right)^2\right)^3\right) \\ \frac{c^2}{6} \end{cases}$	$\begin{cases} \left(1-(\frac{e}{c})^2\right)^2 \\ 0 \end{cases}$	
Welsch	$\frac{c^2}{2}\left(1-\exp\left(-(\frac{e}{c})^2\right)\right)$	$\exp\left(-(\frac{e}{c})^2\right)$	

Table 1. Comparison between OLS estimator and M–Estimators. Objective $\rho(e)$ and weight $w(e)$ functions of OLS solution and 5 different M–Estimators. For M–Estimators, error entries are standardized, i.e. divided by the scale estimator, s. Each robust variant comes with a hyperparameter c. Exemplary plots in the last column utilize the optimal hyperparameters detailed in Table 2

Cauchy	Fair	Huber	Tukey	Welsch
2.385	1.4	1.345	4.685	2.985

Table 2. Optimal hyperparameter c of M–Estimators according to a 95% asymptotic efficiency on the standard Normal distribution.

The last two differences are key for convergence and interpretability, respectively. The former guarantees shrinkage of the weighted error in its first and second moments, while the latter provides an intuitive, bounded, m–dimensional vector capable of discriminating between samples from the main mode and potential outliers at the tails of the density.

RESULTS

This section evaluates the performance of the proposed methods in three different settings. First, sparse coding on synthetic data is evaluated under different noise scenarios. Then, we present an image recognition framework fully–based on sparse decompositions using a well known digital image database. Lastly, a denoising mechanism that exploits local sparse coding highlights the potential of the proposed techniques.

Sparse Coding with Synthetic Data

The dictionary or observation matrix, $\mathbf{D} \in \mathbb{R}^{100\times 500}$, is generated with independent entries drawn from a zero–mean Gaussian random variable with variance equal to one. The ideal sparse code, $\mathbf{x}_0 \in \mathbb{R}^{500}$, is generated by randomly selecting ten entries and assigning them independent samples from a zero–mean,

Algorithm 1 IRLS–based M–Estimation

```
function IRLS($\mathbf{y} \in \mathbb{R}^m, \Phi \in \mathbb{R}^{m \times k}, w_c(u)$)    ▷ Weight function $w(u)$ with hyperparameter $c$
    $t \leftarrow 0$
    $\beta^{(0)} = \beta_{\text{OLS}} \leftarrow (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y}$    ▷ OLS initialization
    $\mathbf{e}^{(0)} \leftarrow \mathbf{y} - \Phi \beta^{(0)}$
    $MAD \leftarrow \text{median}|\mathbf{e}^{(0)}[i] - \tilde{\mathbf{e}}^{(0)}|$
    $s^{(0)} \leftarrow 1.4826 \times MAD$
    $\mathbf{w}^{(0)}[i] \leftarrow w_c\left(\frac{\mathbf{e}^{(0)}[i]}{s^{(0)}}\right)$    $i = 1, 2, \ldots, m$    ▷ Initial weight vector
    $\mathbf{W}^{(0)} \leftarrow \text{diag}\left(\mathbf{w}^{(0)}\right)$
    $t \leftarrow 1$
    while NO CONVERGENCE do
        $\beta^{(t)} \leftarrow (\Phi^T \mathbf{W}^{(t-1)} \Phi)^{-1} \Phi^T \mathbf{W}^{(t-1)} \mathbf{y}$    ▷ Block coordinate descent
        $\mathbf{e}^{(t)} \leftarrow \mathbf{y} - \Phi \beta^{(t)}$
        $MAD \leftarrow \text{median}|\mathbf{e}^{(t)}[i] - \tilde{\mathbf{e}}^{(t)}|$
        $s^{(t)} \leftarrow 1.4826 \times MAD$
        $\mathbf{w}^{(t)}[i] \leftarrow w_c\left(\frac{\mathbf{e}^{(t)}[i]}{s^{(t)}}\right)$    $i = 1, 2, \ldots, m$    ▷ Block coordinate descent
        $\mathbf{W}^{(t)} \leftarrow \text{diag}\left(\mathbf{w}^{(t)}\right)$
        $t \leftarrow t + 1$
    return $\hat{\beta}_{\text{M–Est}} \leftarrow \beta^{(t)}$    $\hat{\mathbf{w}}_{\text{M–Est}} \leftarrow \mathbf{w}^{(t)}$    ▷ Final estimates
```

Algorithm 2 RobOMP

```
function RobOMP($\mathbf{y} \in \mathbb{R}^m, \mathbf{D} \in \mathbb{R}^{m \times n}, w_c(u), K$)
    $k \leftarrow 1$    ▷ Initializations
    $\mathbf{r}_0 \leftarrow \mathbf{y}$
    $\Lambda_0 \leftarrow \emptyset$
    while $k < K$ do
        $\lambda_k = \text{argmax}_{i \in \Omega} |\langle \mathbf{r}_{k-1}, \mathbf{d}_i \rangle|$    $\Omega = \{1, 2, \cdots, n\}$
        $\Lambda_k = \Lambda_{k-1} \cup \{\lambda_k\}$
        $\Phi = [\mathbf{d}_{\Lambda_k[1]}, \mathbf{d}_{\Lambda_k[2]}, \cdots, \mathbf{d}_{\Lambda_k[k]}]$
        $\{\hat{\beta}_{\text{M–Est}}, \hat{\mathbf{w}}_k\} \leftarrow \text{IRLS}(\mathbf{y}, \Phi, w_c(u))$    ▷ Robust linear fit
        $\mathbf{x}_k[\Lambda_k[i]] \leftarrow \hat{\beta}_{\text{M–Est}}[i]$    $i = 1, 2, \ldots, k$    ▷ Update active set
        $\mathbf{r}_k[i] \leftarrow \hat{\mathbf{w}}_k[i] \times (\mathbf{y}[i] - (\mathbf{D}\mathbf{x}_k)[i])$    $i = 1, 2, \ldots, m$    ▷ Update residual
        $k \leftarrow k + 1$
    return $\mathbf{x}_{\text{RobOMP}} \leftarrow \mathbf{x}_K, \mathbf{w} \leftarrow \hat{\mathbf{w}}_K$    ▷ Final Estimates
```

unit–variance Gaussian distribution. The rest of the components are set equal to zero, i.e. $K_0 = 10$. The resulting observation vector $\mathbf{y} \in \mathbb{R}^{100}$ is computed as the linear combination of the dictionary with weights from the ideal sparse code plus a noise component $\mathbf{n} \in \mathbb{R}^{100}$:

$$\mathbf{y} = \mathbf{D}\mathbf{x}_0 + \mathbf{n} \tag{21}$$

The first set of experiments considers different noise distributions. In particular, five noise cases are analyzed: Gaussian ($\mathcal{N}(0,2)$), Laplacian with variance equal to 2, Student's t–distribution with 2 degrees of freedom, Chi–squared noise with 1 degree of freedom, and Exponential with parameter $\lambda = 1$. Then, OMP, GOMP, CMP, and the 5 variants of RobOMP estimate the sparse code with parameter $K = 10$. For the active set update stage of CMP and RobOMP, the maximum allowed number of HQ/IRLS iterations is set to 100. For GOMP, $N_0 \in \{2, 3, 4, 5\}$ where the best results are presented.

The performance measure is defined as the normalized ℓ_2–norm of the difference between the ground truth sparse code, $\mathbf{x}_0$, and its estimate. The average results for 100 independent runs are summarized in Table 3. As expected, most of the algorithms perform similar under Gaussian noise, which highlights the adaptive nature of CMP and RobOMP. For the non–Gaussian cases, CMP and Tukey are major improvements over ordinary OMP. The rest of the RobOMP flavors consistently outperform the state

Noise	Gaussian	Laplacian	Student	Chi–squared	Exponential
OMP	5.92	5.69	7.14	5.22	4.43
GOMP	7.66	7.27	9.37	6.71	5.65
CMP	**5.57**	**4.40**	3.87	3.08	**3.49**
Cauchy	5.88	5.21	4.43	3.95	4.06
Fair	5.92	5.34	5.05	4.45	4.13
Huber	5.80	5.04	4.57	3.92	3.89
Tukey	5.85	4.78	**3.80**	**3.05**	3.64
Welsch	5.82	4.84	3.90	3.20	3.70

Table 3. Average norm of sparse code errors of MSE–based OMPs and robust alternatives for different types of noise. Best results are marked bold. $K = K_0 = 10$.

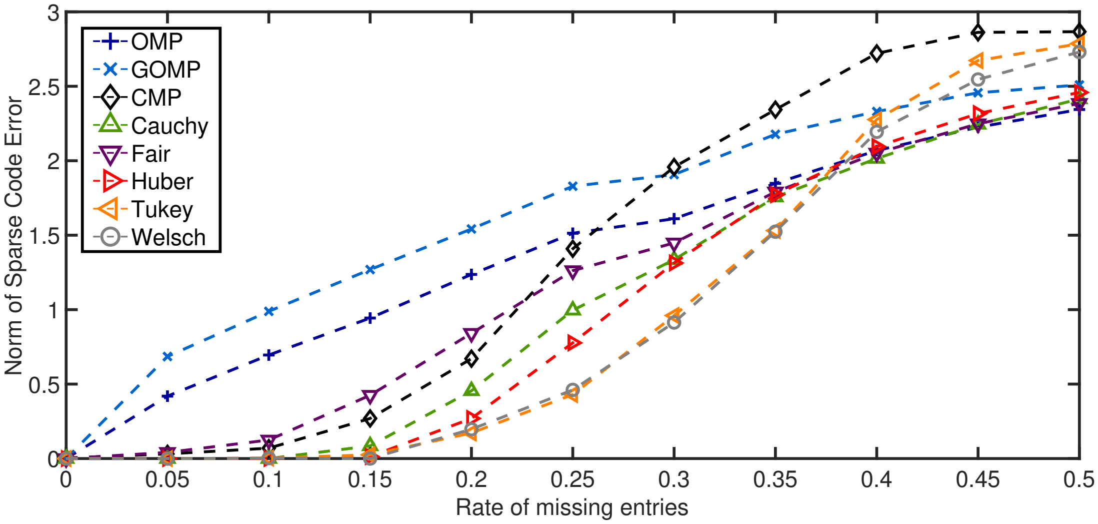


Figure 3. Average normalized norm of sparse code error of MSE–based OMPs and robust alternatives for several rates of missing entries in the observation vector. All algorithms use the ground truth sparsity parameter $K = K_0 = 10$.

of the art OMP and GOMP techniques. This confirms the optimality of MSE–based greedy sparse decompositions when the errors are Normally distributed; yet, they degrade their performance when such assumption is violated.

The second set of results deals with non–linear additive noise or instance–based degradation. Once again, $\mathbf{D}$ and $\mathbf{x}_0$ are generated following the same procedure of the previous set of results ($K_0 = 10$). Yet now, noise is introduced by means of zeroing randomly selected entries in $\mathbf{y}$. The number of missing samples is modulated by a rate parameter ranging from 0 to 0.5. Fig. 3 summarizes the average results for $K = 10$ and 100 independent runs. As expected, the performance degrades when the rate of missing entries increases. However, the five variants of RobOMP are consistently superior than OMP and GOMP until the 0.4–mark. Beyond that point, some variants degrade at a faster rate. Also, CMP achieves small sparse code error norms for low missing entries rate; however, beyond the 0.25–mark, CMP seems to perform worse than OMP and even GOMP. This experiment highlights the superiority of RobOMP over MSE–based and Correntropy–based methods.

Now, the effect of the hyperparameter K is studied. Once again, 100 independent runs are averaged to estimate the performance measure. The rate of missing entries is fixed to 0.2 while K is the free variable. Fig. 4 shows how the average error norm is a non–increasing function of K for the non–MSE–based variants of OMP (slight deviation in some cases beyond $K = 8$ might be due to estimation uncertainty and restricted sample size). On the other hand, both OMP and GOMP seem to stabilize after a certain number of iterations, resulting in redundant runs of the algorithm. These outcomes imply that RobOMP is not only a robust sparse code estimator, but also a statistically efficient one that exploits the available

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27482v1 | CC BY 4.0 Open Access | rec: 14 Jan 2019, publ: 14 Jan 2019

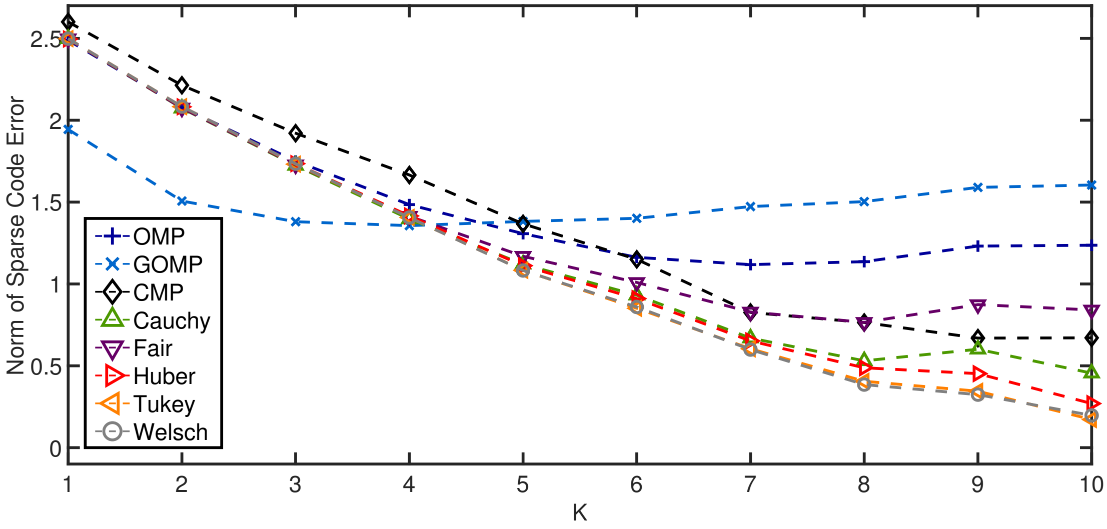
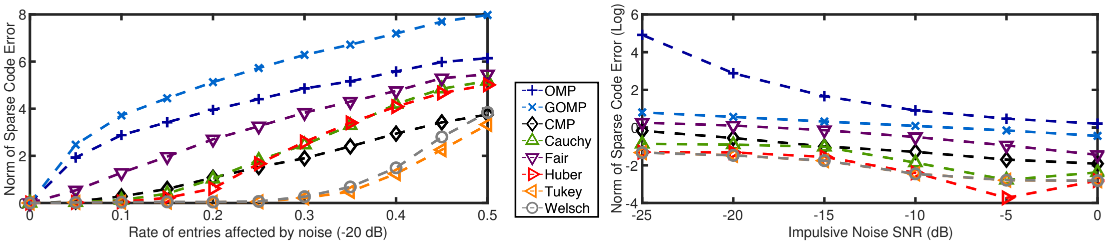


Figure 4. Average normalized norm of sparse code error of MSE–based OMPs and robust alternatives over K (number of iterations) for a 0.2 rate of missing entries in the observation vector. $K_0 = 10$.

information in the data in a principled manner. It is also worth noting that CMP underperforms when compared to most flavors of RobOMP.

Impulsive noise is the other extreme of instance–based contamination. Namely, a rate of entries in $\mathbf{y}$ are affected by aggressive high–variance noise while the rest of the elements are left intact. The average performance measure of 100 independent runs is reported for $K = K_0 = 10$. Fig. 5a details the results for varying rates of entries affected by -20 dB impulsive noise. Again, RobOMP and CMP outperform OMP and GOMP throughout the entire experiment. Tukey and Welsch seem to handle this type of noise more effectively; specifically, the error associated to the algorithms in question seem to be logarithmic or radical for OMP and GOMP, linear for Fair, Cauchy, Huber and CMP, and polynomial for Tukey and Welsch with respect to the percentage of noisy samples. On the other hand, Fig. 5b reflects the result of fixing the rate of affected entries to 0.10 and modulating the variance of the impulsive noise in the range [-25,0]. RobOMP and CMP again outperform MSE–based methods (effect visually diminished due to log–transform of the performance measure for plotting purposes). For this case, CMP is only superior to the Fair version of RobOMP.

In summary, the experiments concerning sparse coding with synthetic data confirm the robustness of the proposed RobOMP algorithms. Non–Gaussian errors, missing samples and impulsive noise are handled in a principled scheme by all the RobOMP variants and, for most cases, the results outperform the Correntropy–based CMP. Tukey seems to be the more robust alternative that is able to deal with a wide spectrum of outliers in a consistent, efficient manner.



(a) Performance measure for several rates of noisy entries (-20 dB) in the observation vector.

(b) Log–Performance measure for several noise levels in the observation vector (fixed 0.10 rate).

Figure 5. Average normalized norm of sparse code error of MSE–based OMPs and robust alternatives for 2 cases involving impulsive noise in the observation vector. All algorithms use the ground truth sparsity parameter $K = K_0 = 10$.

```
Algorithm 3 RSRC
Inputs: Normalized matrix of training samples $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N] \in \mathbb{R}^{m \times n}$
Test Example, $\mathbf{y} \in \mathbb{R}^m$
M–Estimator weight function, $w_c(u)$
Stopping criterion for RobOMP, $K$
Output: class($\mathbf{y}$)
  $(\mathbf{x}_{\text{RobOMP}}, \mathbf{w}) \leftarrow \text{RobOMP}(\mathbf{y}, \mathbf{A}, w_c(u), K)$    ▷ Compute robust sparse code and weight vector
  $r^i(\mathbf{y}) = ||\text{diag}(\mathbf{w})(\mathbf{y} - \mathbf{A}_i \delta_i(\mathbf{x}_{\text{RobOMP}}))||_2, \quad i \in \mathsf{N}$    ▷ Calculate norm of class–dependent residuals
  $\text{class}(\mathbf{y}) \leftarrow \text{argmin}_{i \in \mathsf{N}}\, r^i(\mathbf{y})$    ▷ Predict label
```

RobOMP–based Classifier

We introduce a novel robust variant for sparse representation–based classifiers (SRC) fully based on RobOMP. Let $\mathbf{A}_i = [\mathbf{a}_1^i, \mathbf{a}_2^i, \ldots, \mathbf{a}_{n_i}^i] \in \mathbb{R}^{m \times n_i}$ be a matrix with n_i examples from the ith class for $i = 1, 2, \ldots, N$. Then, denote the set $\mathsf{N} = \{1, 2, \ldots, N\}$ and the dictionary matrix of all training samples $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N] \in \mathbb{R}^{m \times n}$ where $n = \sum_{i=1}^{N} n_i$ is the number of training examples from all N classes. Lastly, for each class i, the characteristic function $\delta_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$ extracts the coefficients associated with the ith label. The goal of the proposed classifier is to assign a class to a novel test sample $\mathbf{y} \in \mathbb{R}^m$ given the generative “labeled” dictionary $\mathbf{A}$.

The classification scheme proceeds as follows: N different sparse codes are estimated via Algorithm 2 given the subdictionaries $\mathbf{A}_i$ for $i = 1, 2, \ldots, N$. The class–dependent residuals, $r^i(\mathbf{y})$ are computed and the test example is assigned to the class with minimal residual norm. To avoid biased solutions based on the scale of the data, the columns of $\mathbf{A}$ are set to have unit–ℓ_2–norm. The result is a robust sparse representation–based classifier or RSRC, which is detailed in Algorithm 3.

Similar algorithms can be deployed for OMP, GOMP and CMP (Wang et al., 2017). In particular, the original SRC (Wright et al., 2009) exploits a ℓ_1–minimization approach to the sparse coding problem; however the fidelity term is still MSE, which is sensitive to outliers. In this section we opt for greedy approaches to estimate the sparse representation. Moreover for RobOMP, the major difference is the computation of the residual—we utilize the weight vector to downplay the influence of potential outlier components and, hence, reduce the norm of the errors under the proper dictionary. CMP utilizes a similar approach, but the weight matrix is further modified due to the HQ implementation (see Wang et al. (2017) for details). We compare the 7 SRC variants under two different types of noise on the Extended Yale B Database.

Extended Yale B Database

This dataset contains over 2000 facial images of 38 subjects under different lighting settings (Lee et al., 2005). For each subject, a maximum of 64 frontal–face images are provided alongside light source angles. The original dimensionality of the images is 192 × 168 or 32256 in vector form. Fig. 2 illustrates one sample image from the database.

Due to the difference in lighting conditions, the database is usually segmented into 5 subsets (Wright et al., 2009). Let $\theta = \sqrt{A^2 + E^2}$ where A and E are the azimuth and elevation angles of the single light source, respectively. The first subset comprises the interval $0 \leq \theta \leq 12$, the second one, $13 \leq \theta \leq 25$, the third one, $26 \leq \theta \leq 54$, the fourth one, $55 \leq \theta \leq 83$, and lastly, the fifth subset includes images with $\theta \geq 84$. In this way, the subsets increase in complexity and variability, making the classifier job more challenging, e.g. subset one includes the cleanest possible examples, while the fifth dataset presents aggressive occlusions in the form of shadows. The cardinality of the 5 subsets are (per subject): 7, 12, 12, 14, and 19 images. For all the following experiments, the dictionary matrix $\mathbf{A}$ is built from the samples corresponding to subsets 1 and 2, while the test examples belong to the third subset. This latter collection is further affected by two kinds of non–linearly additive noise.

Occlusions and Missing Pixels

Two different types of noise are simulated: blocks of salt and pepper noise, i.e. occlusions, and random missing pixels. In all the following experiments, the sparsity parameter for learning the sparse code is set to $K = 5$ (for GOMP, $N_0 \in \{2, 3\}$ and the best results are presented). Also, 10 different independent runs are simulated for each noise scenario.

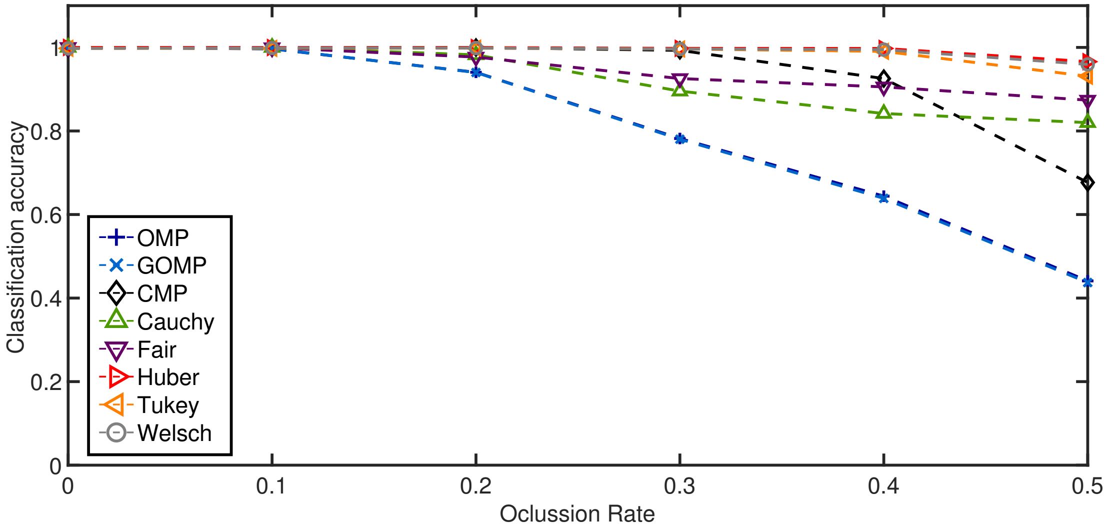
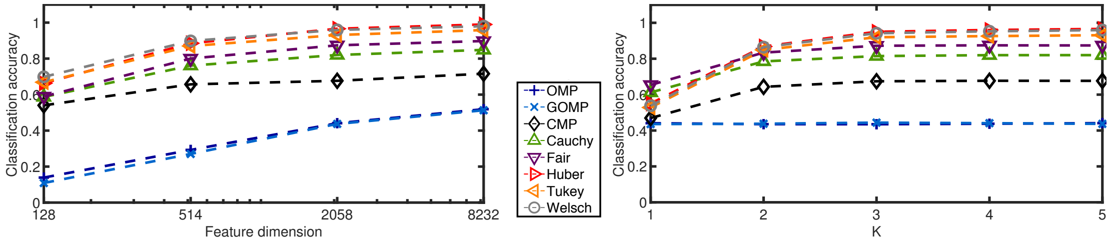


Figure 6. Average classification accuracy on the Extended Yale B Database over occlusion rate of blocks of salt and pepper noise. Feature dimension = 2058. $K = 5$.

For the occlusion blocks, a rate of affected pixels is selected beforehand in the range [0, 0.5]. Then, as in the original SRC (Wright et al., 2009), we downsampled the inputs mainly for computational reasons. In particular, we utilized factors of 1/2, 1/4, 1/8, and 1/16 resulting in feature dimensions of 8232, 2058, 514, and 128, respectively. Next, every test example is affected by blocks of salt and pepper noise (random pixels set to either 0 or 255). The location of the block is random and its size is determined by the rate parameter. Every sample is assigned a label according to SRC variants based on OMP and GOMP, CMP–based classifier (coined as CMPC by Wang et al. (2017)), and our proposed RSRC. For simplicity, we use the same terminology as before when it comes to the different classifiers. The performance metric is the average classification accuracy in the range [0,1]. Fig. 6 highlights the superiority of RSRC over OMP and GOMP. Particularly, Huber, Tukey and Welsch are consistently better than CMP while Fair and Cauchy seem to plateau after the 0.3–mark.

Next, the effects of the feature dimension and the sparsity parameter are investigated. Fig. 7 confirms the robustness of the proposed discriminative framework. As expected, when the feature dimension increases, the classification accuracy increases accordingly. However, the baselines set by OMP and GOMP are extremely low for some cases. On the other hand, CMP and RSRC outperform both MSE–based approaches, and even more, the novel M–Estimator–based classifiers surpass their Correntropy–based counterpart. When it comes to the sparsity parameter, K, it is remarkable how OMP and GOMP do not improve their measures after the first iteration. This is expected due to the lack of principled schemes to deal with outliers. In contrast, RSCR shows a non–decreasing relation between classification accuracy and K, which implies progressive refinement of the sparse code over iterations. To make these last two findings more evident, Table 4 illustrates the classification accuracy for a very extreme case: 0.3 rate of occlusion and feature dimension equal to 128, i.e. each input image is roughly 12×11 pixels in size (the downsampling operator introduces rounding errors in the final dimensionality). This scenario is very challenging and, yet, most of RSRC variants achieve stability and high classification after only 4 iterations. On the other hand, OMP and GOMP degrade their performance over iterations. This confirms the robust and sparse nature of the proposed framework.

For the missing pixels case, a rate of affected pixels is selected beforehand in the range [0, 1]. Then, every test example is affected by randomly selected missing pixels—the chosen elements are replaced by samples drawn from a uniform distribution over the range $[0, y_{\max}]$ where $y_{\max}$ is the largest possible intensity of the image in question. Figures 8 and 9 summarize similar experiments as in the occlusion case. Again, the RSRC are superior than MSE–based methods and consistently increase the performance measure as the sparsity parameter grows. The extreme case here involves a rate of 0.4 affected pixels by distorted inputs and a feature dimension of 128. Table 5 reinforces the notion that robust methods achieve higher classification accuracy even in challenging scenarios.



(a) Classification accuracy over feature dimension. $K = 5$.

(b) Classification accuracy over sparsity parameter. Feature dimension = 2058.

Figure 7. Average classification accuracy on the Extended Yale B Database for two cases concerning blocks of salt and pepper noise at a fixed rate of 0.5.

K	OMP	GOMP	CMP	Cauchy	Fair	Huber	Tukey	Welsch
1	0.38	**0.36**	0.59	0.53	0.55	0.53	0.51	0.51
2	**0.39**	0.34	0.90	0.81	0.83	0.87	0.86	0.87
3	0.39	0.30	**0.97**	**0.88**	0.88	0.98	0.98	0.98
4	0.37	0.28	0.97	0.88	**0.89**	**0.99**	**0.99**	**0.99**
5	0.36	0.28	0.97	0.88	0.88	0.99	0.99	0.99
6	0.34	0.28	0.97	0.88	0.88	0.99	0.99	0.99
7	0.34	0.28	0.97	0.88	0.88	0.98	0.99	0.99
8	0.33	0.28	0.97	0.88	0.88	0.98	0.99	0.99
9	0.32	0.28	0.97	0.88	0.88	0.98	0.99	0.99
10	0.31	0.28	0.97	0.88	0.88	0.98	0.99	0.98

Table 4. Average classification accuracy on the Extended Yale B Database over K for a fixed rate of 0.3 pixels affected by blocks of salt and pepper noise. Best result for each classifier is marked bold. Feature dimension = 128.

Lastly, it is worth noting that CMP performs better in the missing pixels case; yet, it fails to surpass the Welsch variant of RSRC which is its equivalent in terms of weight function of errors. Once again, Tukey is the algorithm with overall best results that is able to handle both kinds of noise distributions in a more principled manner.

Image Denoising via Robust, Sparse and Redundant Representations

The last set of results introduces a preliminary analysis of image denoising exploiting sparse and redundant representations over overcomplete dictionaries. The approach is based on the seminal paper by Elad and Aharon (2006). Essentially, zero–mean white and homogeneous Gaussian additive noise with variance σ^2 is removed from a given image via sparse modeling. A global image prior that imposes sparsity over patches in every location of the image simplifies the sparse modeling framework and facilitates its implementation via parallel processing. In particular, if the unknown image $\mathbf{Z}$ can be devised as the spatial (and possibly overlapping) superposition of patches that can be effectively sparsely represented given a dictionary $\mathbf{D}$, then, the optimal sparse code, $\hat{\mathbf{x}}_{ij}$, and estimated denoised image, $\hat{\mathbf{Z}}$, are equal to:

$$\{\hat{\mathbf{x}}_{ij}, \hat{\mathbf{Z}}\} = \underset{\mathbf{x}_{ij},\mathbf{Z}}{\operatorname{argmin}}\, \lambda ||\mathbf{Z}-\mathbf{Y}||_2^2 + \sum_{ij} \mu_{ij} ||\mathbf{x}_{ij}||_0 + \sum_{ij} ||\mathbf{D}\mathbf{x}_{ij} - \mathbf{R}_{ij}\mathbf{Z}||_2^2 \tag{22}$$

where the first term is the log–likelihood component that enforces close resemblance (or proximity in an ℓ_2 sense) between the measured noisy image, $\mathbf{Y}$, and its denoised (and unknown) counterpart $\mathbf{Z}$. The second and third terms are image priors that enforce that every patch, $\mathbf{z}_{ij} = \mathbf{R}_{ij}\mathbf{Z}$, of size $\sqrt{n} \times \sqrt{n}$ in every location of the constructed image $\mathbf{Z}$ has a sparse representation with bounded error. λ and μ_{ij} are regularization parameters than can easily be reformulated as constraints.

Block coordinate descent is exploited to solve (22). In particular, $\hat{\mathbf{x}}_{ij}$ is estimated via greedy approximations of the sparse code of each local block or patch. The authors suggest OMP with stopping criterion set by $||\mathbf{D}\mathbf{x}_{ij} - \mathbf{R}_{ij}\mathbf{Z}||_2^2 \leq (C\sigma)^2$ for all $\{ij\}$ combinations (sequential sweep of the image to extract all

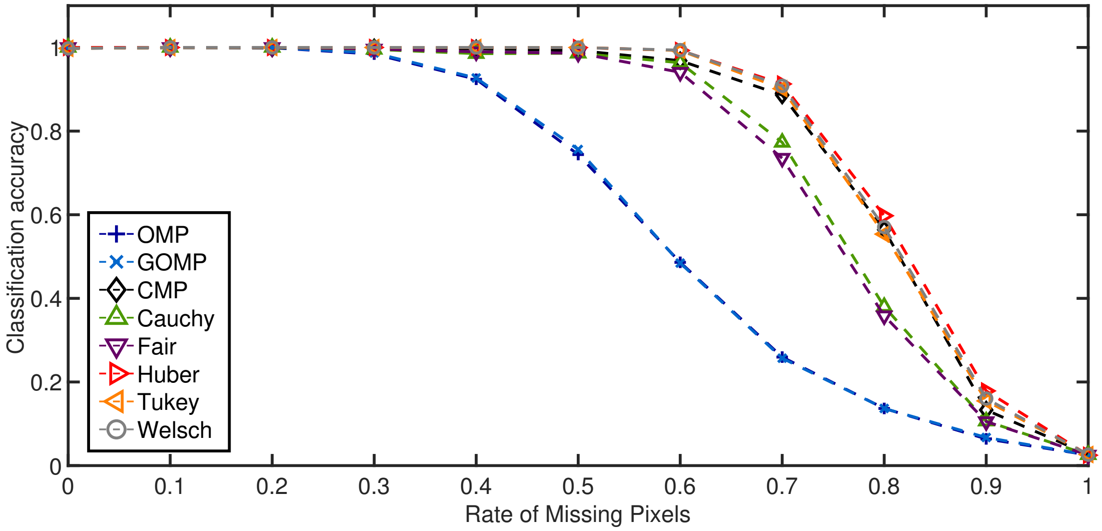
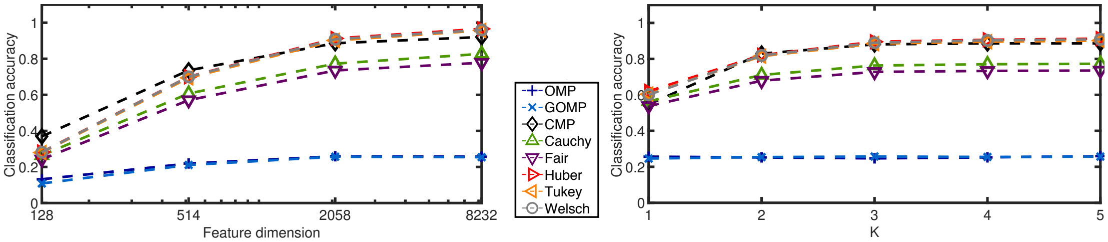


Figure 8. Average classification accuracy on the Extended Yale B Database over missing pixels rate. Feature dimension = 2058. $K = 5$.



(a) Classification accuracy over feature dimension. $K = 5$.

(b) Classification accuracy over sparsity parameter. Feature dimension = 2058.

Figure 9. Average classification accuracy on the Extended Yale B Database for two cases concerning missing pixels at a fixed rate of 0.7.

possible $\sqrt{n} \times \sqrt{n}$ blocks). Then, the estimated denoised image has the following closed form solution:

$$\hat{\mathbf{Z}} = \left(\lambda I + \sum_{ij} \mathbf{R}_{ij}^T \mathbf{R}_{ij}\right)^{-1} \left(\lambda \mathbf{Y} + \sum_{ij} \mathbf{R}_{ij}^T \mathbf{D} \mathbf{x}_{ij}\right) \tag{23}$$

where I is the identity matrix. The authors go one step further and propose learning the dictionary, $\mathbf{D}$, as well; this is accomplished either from a corpus of high–quality images or the corrupted image itself. The latter alternative results in a fully generative sparse modeling scheme. For more details regarding the denoising mechanisms, refer to Elad and Aharon (2006).

For our case, we focus on the sparse coding subproblem alone and utilize an overcomplete Discrete Cosine Transform (DCT) dictionary, $\mathbf{D} \in \mathbb{R}^{64\times256}$, and overlapping blocks of size 8×8. The rest of the free parameters are set according to the heuristics presented in the original work: $\lambda = 30/\sigma$ and $C = 1.15$. Our major contribution is the robust estimation of the sparse codes via RobOMP in order to handle potential outliers in a principled manner. Two types of zero–mean, homogeneous, additive noise (Gaussian and Laplacian) are simulated with different variance levels on 10 independent runs. Each run comprises of separate contaminations of 4 well known images (Lena, Barbara, Boats and House) followed by the 7 different denoising frameworks, each one based on a distinct variant of OMP. As before, every algorithm is referred to as the estimator exploited in the active set update stage.

Tables 6 and 7 summarize the average performance measures (PSNR in dB) for 5 different variance levels of each noise distribution. As expected, OMP is roughly the best denoising framework for additive Gaussian noise. However in the Laplacian case, Cauchy achieves higher PSNR levels throughout the

K	OMP	GOMP	CMP	Cauchy	Fair	Huber	Tukey	Welsch
1	0.51	**0.54**	0.57	0.61	0.62	0.56	0.54	0.54
2	0.54	0.52	0.89	0.87	0.86	0.88	0.88	0.88
3	**0.57**	0.48	**0.95**	0.91	**0.90**	0.93	0.94	0.94
4	0.56	0.45	0.95	**0.92**	0.90	**0.94**	0.96	0.95
5	0.55	0.45	0.95	0.92	0.89	0.94	0.96	**0.96**
6	0.54	0.45	0.95	0.91	0.89	0.94	**0.97**	0.96
7	0.53	0.45	0.94	0.91	0.89	0.94	0.97	0.96
8	0.52	0.45	0.94	0.91	0.89	0.94	0.96	0.96
9	0.51	0.45	0.94	0.91	0.89	0.94	0.96	0.95
10	0.50	0.45	0.94	0.91	0.89	0.93	0.96	0.95

Table 5. Average classification accuracy on the Extended Yale B Database over K for a fixed rate of 0.4 missing pixels. Best result for each classifier is marked bold. Feature dimension = 128.

σ	OMP	GOMP	CMP	Cauchy	Fair	Huber	Tukey	Welsch
5	36.33	36.31	35.62	**36.56**	36.55	36.52	36.20	36.29
10	32.38	32.36	31.01	**32.44**	32.22	32.39	32.17	32.17
15	**30.35**	30.33	28.95	30.25	29.88	30.21	30.01	29.97
20	**28.97**	28.96	27.85	28.78	28.40	28.76	28.58	28.53
25	**27.93**	27.92	27.12	27.70	27.39	27.70	27.55	27.51

Table 6. Grand average PSNR (dB) of estimated denoised images under zero–mean additive Gaussian noise exploiting patch–based sparse and redundant representations.

entire experiment. This suggests the Cauchy M–Estimator is more suitable for this type of non–Gaussian environment (see Fig. 10 for an example). It is worth noting though that the averaging performed in (23) could easily blur the impact of the sparse code solvers for this particular joint optimization. Also, no attempt was made to search over the hyperparameter space of λ and C, which we suspect have different empirical optima depending on the noise distribution and sparse code estimator. These results are simply preliminary and highlight the potential of robust denoising frameworks based on sparse and redundant representations.

DISCUSSION

An example is considered a univariate outlier if it deviates from the rest of the distribution for a particular variable or component (Andersen, 2008). A multivariate outlier extends this definition to more than one dimension. However, a regression outlier is a very distinctive type of outlier—it is a point that deviates from the linear relation followed by most of the data given a set of predictors or explanatory variables. In this regard, the current work focuses on regression outliers alone. The active set update stage of OMP explicitly models the interactions between the observation vector and the active atoms of the dictionary as purely linear. This relation is the main rationale behind RobOMP: regression outliers can be detected and weighted when M–Estimators replace the pervasive OLS solver. If the inference process in sparse

σ	OMP	GOMP	CMP	Cauchy	Fair	Huber	Tukey	Welsch
5	36.27	36.25	35.64	**36.59**	36.56	36.56	36.21	36.30
10	32.22	32.19	31.03	**32.44**	32.20	32.38	32.15	32.15
15	30.09	30.05	28.97	**30.20**	29.83	30.15	29.95	29.90
20	28.63	28.58	27.88	**28.70**	28.33	28.66	28.50	28.45
25	27.51	27.45	27.14	**27.60**	27.30	27.58	27.45	27.41

Table 7. Grand average PSNR (dB) of estimated denoised images under zero–mean additive Laplacian noise exploiting patch–based sparse and redundant representations.

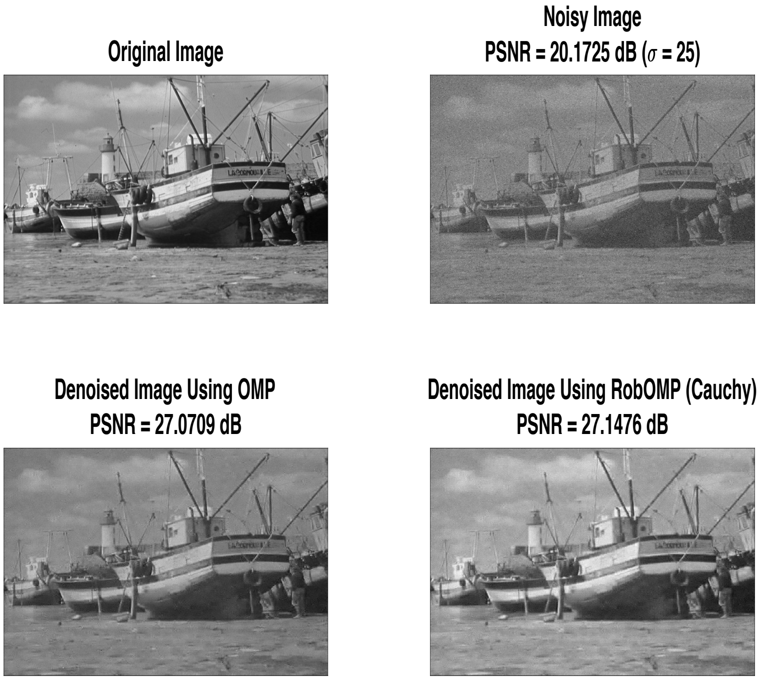


Figure 10. Example of the denoising results for the image “Boats” under Laplacian noise with $\sigma = 25$. Original, noisy, and two denoised results based on OMP and RobOMP (Cauchy variant).

modeling incorporates higher–order interactions (as in Vincent and Bengio (2002)), linear regression outliers become meaningless and other techniques are needed to downplay their influence. The relation between outliers in the observation vector and regression outliers is highly complex due to the mixing of sources during the generative step and demands for further research.

Even though other OMP variants are utilized in practice for different purposes, e.g. GOMP, ROMP and CoSaMP, we decided to disregard the last two flavors mainly due to three factors: space limitations, inherent MSE cost functions, and most importantly, they both have been outperformed by CMP in similar experiments as the ones simulated here (Wang et al., 2017). The algorithm to beat was CMP due to its resemblance to an M–Estimator–based OMP. We believe we have provided sufficient evidence to deem RobOMP (and specifically the Tukey variant) as superior than CMP in a wide variety of tasks, performance measures and datasets. In this regard, it is worth noting that CMP reduces to the Welsch algorithm with the ℓ_2–norm of the errors as the estimated scale parameter ($s = ||\mathbf{e}||_2$), and hyperparameter $c = \sqrt{m}$. The main drawback of such heuristic is the use of a non–robust estimator of the scale, which in turn, will bias the sparse code. The CMP authors introduce a data–dependent parameter of the exponential weight function (Gaussian kernel of Correntropy) that relies on the dimensionality of the input, m. The rationale behind such add–hoc choice is not fully justified, while in contrast, we provide statistically sound arguments for our choice of the weight function hyperparameter, i.e. 95% asymptotic efficiency on the standard Normal distribution. We believe this is the underlying reason behind the superiority of Welsch over CMP on most of the synthetic data experiments and the entirety of the simulations on the Extended Yale B Database.

M–Estimators are not the only alternative to robust linear regression. S–Estimators (Rousseeuw and Yohai, 1984) are based on the residual scale of M–Estimators. Namely, S–estimation exploits the residual standard deviation of the errors to overcome the weakness of the re–scaled MAD. Another option is the so–called MM–Estimators (Yohai, 1987) which fuse S–Estimation and M–Estimation to achieve high BDP and better efficiency. Optimization for both S–Estimators and MM–Estimators is usually performed via IRLS. Another common approach is the Least Median of Squares method (Rousseeuw, 1984) where the optimal parameters solve a non–linear minimization problem involving the median of squared residuals. Advantages include robustness to false matches and outliers, while the main drawback is the need for Monte Carlo sampling techniques to solve the optimization. These three approaches are left for potential further work in order to analyze and compare performances of several types of robust estimators applied to sparse coding.

In terms of image denoising via robust, sparse and redundant representations, future work will involve the use of the weight vector in the block coordinate descent minimization in order to mitigate the effect of outliers. If sparse modeling is the final goal, K–SVD (Aharon et al., 2006) is usually the preferred dictionary learning algorithm. However in the presence of non–Gaussian additive noise, the estimated dictionary might be biased as well due to the explicit MSE cost function of the sequential estimation of generative atoms. Plausible alternatives include Correntropy–based cost functions (Loza and Principe, 2016) and ℓ_1–norm fidelity terms (Loza, 2018).

In the spirit of openness and to encourage reproducibility, the MATLAB (Mathworks) code corresponding to all the proposed methods and experiments of this paper are freely available at `https://github.com/carlosloza/RobOMP`.

CONCLUSION

We proposed a novel, greedy approximation to the sparse coding problem fully based on the theory of M–Estimators under a linear model. Unlike the original Orthogonal Matching Pursuit, our framework is able to handle outliers and non–Gaussian errors in a principled manner. In addition, we introduce a novel robust sparse representation–based classifier that outperform current state of the art and similar robust variants. Preliminary results on image denoising confirm the plausibility of the methods and open the door to future applications where robustness and sparseness are advantageous. The proposed five algorithms do not require parameter tuning from the user and, hence, constitute a suitable alternative to ordinary OMP.

ACKNOWLEDGMENTS

This work was completed thanks to the support of the Poligrant program of Universidad San Francisco de Quito.

REFERENCES

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

Andersen, R. (2008). *Modern methods for robust regression*. Number 152. Sage.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159.

Donoho, D. L., Elad, M., and Temlyakov, V. N. (2006). Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on information theory*, 52(1):6–18.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745.

Elad, M., Figueiredo, M. A., and Ma, Y. (2010). On the role of sparse and redundant representations in image processing. *Proceedings of the IEEE*, 98(6):972–982.

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

Hogg, R. V. (1979). Statistical robustness: One view of its use in applications today. *The American Statistician*, 33(3):108–115.

Huber, P. J. (2011). Robust statistics. In *International Encyclopedia of Statistical Science*, pages 1248–1251. Springer.

Lee, K.-C., Ho, J., and Kriegman, D. J. (2005). Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):684–698.

Liu, W., Pokharel, P. P., and Príncipe, J. C. (2007). Correntropy: Properties and applications in non-gaussian signal processing. *IEEE Transactions on Signal Processing*, 55(11):5286–5298.

Loza, C. A. (2018). Robust K-SVD: A novel approach for dictionary learning. In *International Workshop on Artificial Intelligence and Pattern Recognition*, pages 185–192. Springer.

Loza, C. A. and Principe, J. C. (2016). A robust maximum correntropy criterion for dictionary learning. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE.

Mallat, S. (2008). *A wavelet tour of signal processing: the sparse way*. Academic press.

Mallat, S. and Zhang, Z. (1993). Matching pursuit with time-frequency dictionaries. Technical report, Courant Institute of Mathematical Sciences New York United States.

Needell, D. and Tropp, J. A. (2009). Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and computational harmonic analysis*, 26(3):301–321.

Needell, D. and Vershynin, R. (2010). Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE Journal of selected topics in signal processing*, 4(2):310–316.

Nikolova, M. and Ng, M. K. (2005). Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal on Scientific computing*, 27(3):937–966.

Olshausen, B. A. and Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607.

Rousseeuw, P. and Yohai, V. (1984). Robust regression by means of s-estimators. In *Robust and nonlinear time series analysis*, pages 256–272. Springer.

Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Tropp, J. A. and Gilbert, A. C. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on information theory*, 53(12):4655–4666.

Vincent, P. and Bengio, Y. (2002). Kernel matching pursuit. *Machine Learning*, 48(1-3):165–187.

Wang, J., Kwon, S., and Shim, B. (2012). Generalized orthogonal matching pursuit. *IEEE Transactions on signal processing*, 60(12):6202.

Wang, Y., Tang, Y. Y., and Li, L. (2017). Correntropy matching pursuit with application to robust digit and face recognition. *IEEE transactions on cybernetics*, 47(6):1354–1366.

Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. (2009). Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227.

Xu, Y., Zhang, D., Yang, J., and Yang, J.-Y. (2011). A two-phase test sample sparse representation method for use with face recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(9):1255–1262.

Yohai, V. J. (1987). High breakdown-point and high efficiency robust estimates for regression. *The Annals of Statistics*, pages 642–656.

Zhang, Z. (1997). Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, 15(1):59–76.