2014 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 21–24, 2014, REIMS, FRANCE

LARGE SCALE SPARSE OPTIMIZATION FOR OBJECT DETECTION IN HIGH RESOLUTION IMAGES

Aurélie Boisbunon[1,2,*], *Rémi Flamary*[3,†], *Alain Rakotomamonjy*[4,‡], *Alain Giros*[1], *Josiane Zerubia*[2]

[1]CNES
DCT/SI/AP
Toulouse, France

[2]INRIA
AYIN team
Sophia Antipolis, France

[3]UNS-CNRS-OCA
Lab. Lagrange
Nice, France

[4]Normandie Université
Lab. LITIS
Rouen, France

ABSTRACT

In this work, we address the problem of detecting objects in images by expressing the image as convolutions between activation matrices and dictionary atoms. The activation matrices are estimated through sparse optimization and correspond to the position of the objects. In particular, we propose an efficient algorithm based on an active set strategy that is easily scalable and can be computed in parallel. We apply it to a toy image and a satellite image where the aim is to detect all the boats in a harbor. These results show the benefit of using nonconvex penalties, such as the log-sum penalty, over the convex ℓ_1 penalty.

Index Terms— Object detection, sparsity, active set, unsupervised learning, image processing

1. INTRODUCTION

We address the problem of object detection in high resolution images with the hypotheses that, in such images, our object of interest can be approximated by a simple shape. In addition, the number of objects in the image is unknown, but typically reaches several hundreds or thousands. This problem is also known as monitoring in the remote sensing community and, in particular, we are interested in the special case of boat detection in harbors on satellite images. Images provided by Pleiades satellites are typically of size 42000×42000 px for several gigabytes in memory.

In this setting, marked point processes (MPP) are a useful model of unsupervised learning for getting a refined configuration of the objects. MPPs are point processes where the position of objects is augmented by their shape, called the mark. The associated method consists in modeling the distribution of the configuration in order to estimate both the position and the shape parameters of the objects [see *e.g.* 1]. In [2] and references therein, the probability of the configuration $\mathbf{C}$, given an image $\mathbf{Y}$ is defined by a Gibbs distribution taking into account both the image and prior information:

$$P_\theta[\mathbf{C}|\mathbf{Y}] = c_\gamma^{-1} \exp\{-\gamma U_d(\mathbf{C},\mathbf{Y}) + U_o(\mathbf{C}) + U_a(\mathbf{C})\}, \quad (1)$$

where γ is the parameter weighting the data and prior terms, c_γ is the normalizing constant, the data term $U_d(\mathbf{C},\mathbf{Y})$ is based on the contrast between the objects and their neighborhoods, and the prior terms $U_o(\mathbf{C})$ and $U_a(\mathbf{C})$ respectively model the overlapping and alignment between objects, (see Figure 1). The aim is to estimate both the configuration $\mathbf{C}$ and the parameter γ based on the image $\mathbf{Y}$. The procedure for building MPPs consists in first initializing and estimating γ, which is currently performed from the null configuration and the Stochastic Expectation-Maximization (SEM) algorithm [3], and then extracting the objects by simulated annealing [see 4, Chapter 7 and references therein]. However, the price to pay to get a fine extraction is a large computational time. Indeed, in practice, it already takes several minutes for an image of size 275×385 px [5]. It would thus take a very long time to run it on full Pleiades images.

The computational cost of MPPs could be greatly reduced if it started with a coarse initial configuration instead of the empty one. This is the reason why, in this work, we consider a different approach for the initialization of the configuration. This approach consists in representing the image as a sum of convolutions between a small image with one single object and a highly sparse matrix with non-zero entries corresponding to Dirac delta functions defining the positions of objects. Our approach thus relies on two main components: first, the construction of a dictionary containing representative instances of a proxy of the object with different rotations and scales; then, the determination of the non-zero entries of the matrix associated with each dictionary element. This latter component is based on sparse optimization problems, with *e.g.* an ℓ_1 [6] or a log-sum penalty [7]. The complexity of the problem is linear with the number of pixels in the image multiplied by the number of atoms in the dictionary, which can be very large even for medium-size images. Hence, we propose to use an active set algorithm in order to solve efficiently this

*The first author would like to thank the French Spatial Agency (CNES) for the financial support of her post-doc at INRIA.

†RF acknowledges support of CNRS MASTODONS project DISPLAY.

‡AR acknowlegdes support form grant ANR GRETA 12-BS02-004.

978-1-4799-3694-6/14/$31.00

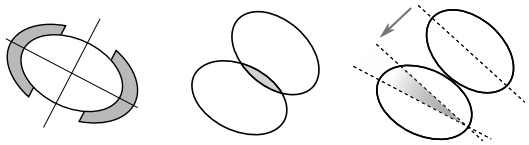

Fig. 1. Representations of the data (left), overlapping (middle) and alignment (right) terms used in the MPP model (1).

problem, in a similar fashion as in [8]. The problem we address bears resemblance to the convolutive non-negative matrix factorization as described in [9, 10]. The main difference is that, in this work, we essentially focus on the sparse approximation stage and on devising an algorithm able to recover very efficiently in a very high-dimensional setting the few active variables.

The remainder of the paper is organized as follows. Section 2 exposes the methods currently used for initialization and estimation of the parameter of the MPP model. Section 3 presents our approach based on the resolution of sparse optimization problems and the efficient active set algorithm proposed for solving it. In Section 4, we compare the performances of two penalties, namely the ℓ_1 and the log-sum penalties, on a toy image, and we show how the algorithm can be used as an initial configuration for a finer method, such as MPPs, on a real satellite image. Finally, we present some perspectives of future work in Section 5.

2. CONTEXT AND BACKGROUND

The difficulty of estimating the parameter γ in (1) comes from two facts: first, the setting is unsupervised, so that the unknowns are both the configuration $\mathbf{C}$ and the parameter γ; second, the normalizing constant c_γ itself depends on the parameter γ, resulting in a difficult optimization problem. In this section, we briefly introduce the methods currently used for the initialization and the estimation of γ. We refer the reader to [2, 5] and references therein for more details.

2.1. Initialization of MPP

In [4, Chapter 7], the initialization is performed as follows. First, a manual over-estimate β_0 of the number of objects in the image is required. This over-estimate can be taken for instance as the maximum number of average-size objects that could fit in a rectangle of the size of the image. Then, the tradeoff parameter γ is obtained by taking the root of the function

$$-\beta_0 + \int_{\mathcal{K}} \beta \exp\{-\gamma U_d(u)\}\Lambda(du),$$

where $\Lambda(\cdot)$ is the intensity distribution of a Poisson process, $\mathcal{K}$ is the compact space corresponding to the image, and the integral corresponds to the average number of objects with respect to a Papangelou conditional intensity for an empty configuration $\mathbf{C} = \emptyset$.

2.2. Estimation of the tradeoff parameter

One way to overcome the complexity of the inference problem in unsupervised setting is to use the Stochastic Expectation-Maximization (SEM) algorithm [3], and to replace the likelihood (1) by the pseudo-likelihood [11] considering independance between the objects. Starting from an initial value $\hat{\gamma}^{(0)}$, the algorithm alternates between approximating the expected likelihood based on a simulated configuration $\mathbf{C}^{(k)}$ for fixed $\widehat{\gamma}^{(k)}$, and updating the parameter $\widehat{\gamma}^{(k+1)}$ from Maximum of Pseudo-Likelihood [2]. The main issue of the SEM algorithm is that it is quite long to converge. A slightly faster estimation can be obtained by Stochastic Approximation EM (SAEM) algorithm [12], which takes into account all the simulated configurations in the approximation of the expected pseudo-likelihood. Even so, the estimation step is still long to compute in practice [5].

3. SPARSE OPTIMIZATION FOR OBJECT DETECTION

In this work, we consider a completely different approach for initializing the configuration. We believe it will considerably speed up the estimation step, since it can be started from a good coarse configuration instead of the empty one.

The general idea of our approach is to consider the image as the sum of convolutions of dictionary elements, representing each instance of the object, with Dirac delta functions on the position of the objects in the image. This actually corresponds to a sparse linear optimization problem, since we are looking for the highly sparse matrix of positions. In the following paragraphs, we first explain the main idea into more details, then we expose our algorithm and, finally, we discuss some techniques to use it in very high dimensions.

3.1. Optimization problem

We first assume that we have a dictionary with atoms $\mathbf{D}_k$, $k = 1, \ldots, K$, of size $d \times d$ corresponding to a proxy of the object with different shapes. For an input image $\mathbf{Y}$ of size $n \times m$, the problem can be stated as

$$\min_{\mathbf{X} \in \mathbb{R}_+^{n \times m \times K}} \quad \|\mathbf{Y} - \sum_{k=1}^{K} \mathbf{X}_k * \mathbf{D}_k\|_F^2 + \lambda \Omega(\mathbf{X}), \qquad (2)$$

where $\mathbf{X}_k$ is the activation matrix of size $n \times m$ corresponding to element $\mathbf{D}_k$ of the dictionary, $*$ is the convolution operation with zero-padding and conservation of the size of $\mathbf{X}_k$, $\|\cdot\|_F$ is the Frobenius norm, $\Omega(\cdot)$ is a penalty function, and λ is the tradeoff between goodness of fit and penalization. This representation is illustrated in Figure 2. Note that the regularization term $\Omega(\cdot)$ should promote sparsity in the activation

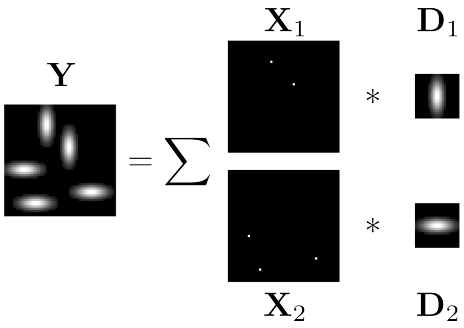

Fig. 2. Representation of an image as a sum of convolutions of activation matrices with elements of the dictionary.

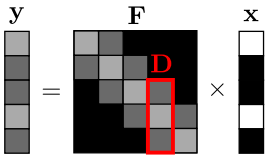

Fig. 3. The matrix $\mathbf{F}$ is formed by concatenation of K Toeplitz matrices where each column corresponds to the vectorized atoms $\mathbf{D}_k$ at one possible location in the vectorized image. Here, we only show a simple example for an image of size 5×1 with one atom.

coefficients in $\mathbf{X}$, since these coefficients correspond to the position of the objects on the image. The construction of the dictionary will be discussed in Section 4.

Problem in (2) can be rewritten in the following way:

$$\min_{\mathbf{x} \geq 0} \quad \|\mathbf{y} - \mathbf{F}\mathbf{x}\|_2^2 + \lambda \Omega(\mathbf{x}), \tag{3}$$

where $\mathbf{y}$ is the vectorized image of length nm, $\mathbf{x}$ contains all the vectorized activation matrices $\mathbf{X}_k$, $k = 1, \ldots, K$, concatenated in one single vector of size nmK and $\mathbf{F}$ is a block Toeplitz matrix that corresponds to the convolution, as displayed in Figure 3. In this work, we choose to regularize $\mathbf{x}$ with a component-wise sparsity promoting term such as an ℓ_1-norm [6], corresponding to $\Omega(\mathbf{x}) = \|\mathbf{x}\|_1$, or a more aggressive non-convex regularization term such as the log-sum penalty $\Omega(\mathbf{x}) = \sum_{i=1}^{nmK} \log(1 + |x_i|/\theta)$ [7].

3.2. 2D active set algorithm

One particularity of problem in (2) is that it is both high dimensional and extremely sparse. For instance, for an image of size 1024×1024 with 15 dictionary elements, and containing several hundreds of objects, say N, we need to recover only N non-zero entries in $\mathbf{X}_k$ out of 15 million components. Therefore, it is crucial to use a very efficient algorithm that takes this extreme sparsity into account in order to obtain the solution in reasonable time.

In [8], the authors considered an active set algorithm for solving extremely sparse nonconvex optimization problems.

Algorithm 1 2D-sparse optimization (2D-SO)

- Input: $\mathbf{Y}$, $(\mathbf{D}_k)_{k=1}^K$
- Init.: $\tilde{\mathbf{Y}} = \mathbf{0}$ (reconstructed image)
  $\mathbf{F} = [\,]$ (active elements)

1: **repeat**
2: $\quad \mathbf{R} \leftarrow \mathbf{Y} - \tilde{\mathbf{Y}}$ % Compute residue
3: $\quad \mathbf{C}_k \leftarrow \text{corr2}(\mathbf{R}, \mathbf{D}_k)$ % Correlation $\forall k$
4: $\quad i, j, c \leftarrow \arg\max_{i,j,c} C_{i,j,c}$
5: $\quad \mathbf{F} \leftarrow [\mathbf{F} \quad \text{im\_dic}(i, j, \mathbf{D}_c)]$ % add new active element
6: $\quad \mathbf{x} \leftarrow$ Solve Problem (3) with current active elements $\mathbf{F}$
7: $\quad \tilde{\mathbf{Y}} \leftarrow \text{reshape}(\mathbf{F}\mathbf{x})$ % Update reconstructed image
8: **until** stopping criterion is met

Their approach is of particular interest in our case as it allows the use of nonconvex regularization terms that improve sparsity. The main idea behind an active set is to iteratively solve the problem on a small subset of active variables. The subset of variables is updated at each iteration by adding the variable that most violates the optimality conditions.

We propose to adapt this algorithm to the context of image processing as shown in Algorithm 1. First we want to emphasize that we can take advantage of the convolution in order to have efficient computations of correlations (Line 3 in the algorithm). The inner problem (Line 6) is solved on a restricted number of active positions and with a sparse matrix $\mathbf{F}$ containing in each column ($\text{im\_dic}(i, j, \mathbf{D}_c)$), a vectorized sparse image with a unique dictionary element at position (i, j) in the image. As in [8], the inner problem is solved by using a proximal gradient descent method, such as FISTA [13] for the ℓ_1-penalty and GIST [14] for non-convex ones. Next, we discuss several strategies that have been performed in order to scale the proposed algorithm to extremely large scale optimization.

3.3. Strategies for large scale optimization

First, step 3 in Algorithm 1 consists in computing a 2D intercorrelation between the residue $\mathbf{R}$ and each of the dictionary elements $\mathbf{D}_k$. This step requires mnd^2 operations for dictionary elements of size $d \times d$ if performed directly or $2m \log_2(m) n \log_2(n)$ if performed in the Fourier domain. In our applications, objects are typically small with respect to the image and a direct computation leads to a computational complexity linear in the number of pixels in the image. Moreover, this step is a typical case of embarrassingly parallel computation as the correlations can be computed in parallel for each dictionary element.

Second, we use in the algorithm what is known as a warm-starting trick: the solution at a given iteration is given as initialization of the inner problem (line 6) for the following iteration. The inner solver being implemented by a gradient descent, the algorithm converges in a few iterations to the new solution. Another trick that allows faster convergence is to

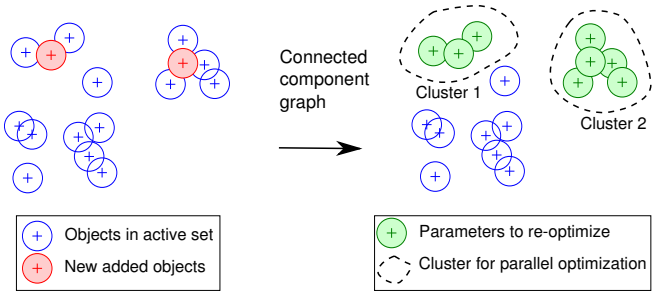

Fig. 4. Illustration of the impact of new added objects in a current active set of objects (left). Only a subset of the active variables has to be updated (in green) and this update can be performed in parallel for several connected component clusters (right).

add multiple variables to the active set at each iteration. When the number of objects in the image is important, an active set algorithm requires a lot of iterations. When the variables are added by batch, lines 4-5 are executed several times before solving the problem line 6. Hence, the final number of active set iterations can be greatly decreased as discussed in [8].

Finally we propose to use the structure of the image model and the fact that the objects are of limited size to further improve speed through a parallel version of the optimization (line 6). First, one can see from the left part of Figure 4 that, when new active variables are added (in red) to the optimization problem, the corresponding objects have a limited impact on the objects located far outside a certain range. This suggests that the re-optimization at line 6 of the algorithm can be exactly solved only on a subset of the current active variables, leading to a significant decrease in computational cost. The subset of parameters to be re-optimized can be obtained simply by using an efficient algorithm on the connected component graph of the objects as described in [15]. An additional benefit of this approach is that all the connected components in the graph (black clusters illustrated in the right part of Figure 4), can be solved independently, allowing a parallel implementation of the re-optimization line 6 in the algorithm.

4. NUMERICAL EXPERIMENTS

4.1. Generated images

In this paragraph, we compare the performances of the ℓ_1 and the log-sum penalties for a detection task in a toy image where we know exactly the position and shape of the objects. The aim is to investigate the benefit of non-convex penalties over convex ones.

The study was driven as follows. First, we built the dictionary from 15 rotations of a truncated 2D-Gaussian distribution, as displayed in the top-right part of Figure 2. Then, we created a 1024×1024 image by placing randomly over the image 10 objects per atom of the dictionary, and we added

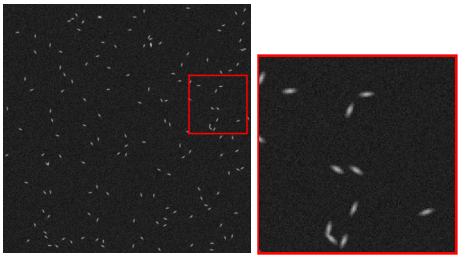

Fig. 5. 1024×1024 noisy toy image (left), and zoom (right).

a Gaussian noise with variance $\sigma^2 = .05$. One example of generated image is displayed in Figure 5 and contains 150 objects.

We run the algorithm 2D-SO (see Algorithm 1) with the ℓ_1 and the log-sum penalties and computed four measures of performance along the regularization path (for a varying λ). The first three measures are based on the F-score, defined by

$$\text{F-score} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}},$$

where TP (true positive) is the number of pixels correctly estimated as part of an object, FP (false positive) is the number of pixels that have been incorrectly estimated as part of an object, and FN (false negative) is the number of pixels that have been incorrectly estimated as part of the background. The F-score thus gives an estimation of the detection rate: the closer to 1, the better the detection is. We computed the F-score on three different matrices: first, for the full image, measuring the overlap between the binary result image and the ground truth; then on the true and reconstructed $\mathbf{X}$ which measures how well are estimated both position and shape of the objects; and finally, for the position only, thus measuring how well the position of the objects can be recovered, regardless of which dictionary atom is activated. We also computed a fourth measure of quality, namely the mean-squared error between the true matrix of activation and the estimated one.

Figure 6 shows the evolution of these four measures as a function of the number of activated objects when varying the regularization parameter λ. The results are averaged over 20 draws. First, note that, between 90 and 170 objects activated, the performances of the log-sum penalty are better than that of the ℓ_1 penalty, whatever the measure. Outside that range, it has equivalent or lower performances. Also, the best F-scores (closest to 1) and MSE (closest to 0) of the log-sum penalty are obtained for 150 activated objects in average, which is exactly the number of objects in the toy image. On the other hand, for the ℓ_1 penalty, the best results are obtained between 200 and 270 objects. These results highlight, in the detection problem, the well known issue of the ℓ_1 penalty's bias, resulting in poor performances for a small number of objects detected and in good performances when the bias is decreased, that is, when more objects are activated. Finally, the performances in F-score for the full image and for the positions are

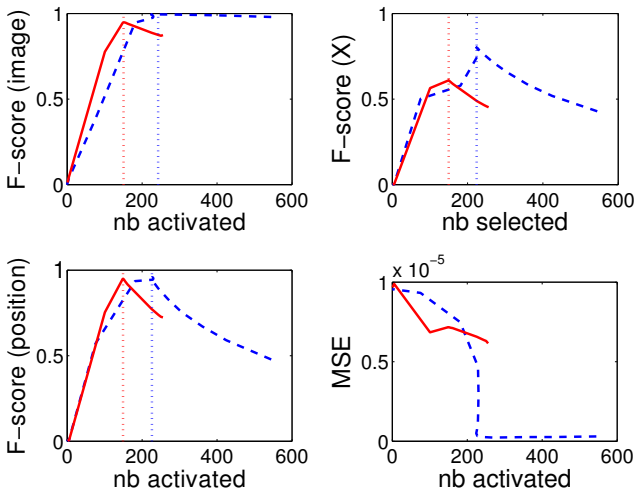

Fig. 6. Comparisons of performance of the ℓ_1 (dashed line) and the log-sum (solid line) penalties, in F-score for the full image (top left), for the positions and shapes (top right), for the positions only (bottom left), and in MSE (bottom right), averaged over 20 draws.

	ℓ_1	LSP
Activated objects	712	713
Non-overlap. objects	424	**489**
F-score (image)	0.71	**0.77**
Steps	77	78
Time (sec)	8.8	11.7

Table 1. Comparison between ℓ_1 and log-sum penalties for a dictionary with 6 scales.

Method	Final F-score	Estimation time (min.)
NC-SEM	70.42 (1.20)	8.19 (2.01)
NC-SAEM	70.14 (0.99)	7.31 (2.52)
2D-SO-SAEM	**73.67** (0.99)	**4.24** (1.35)

Table 2. Comparison of performances in F-score for the image and in computational time of the estimation step when MPPs are initialized with the null configuration (NC) and with our algorithm (2D-SO). The results are given in average over 50 examples with standard deviations given in parenthesis.

very good, close to one at the optimum, which means that both penalties are able to recover the right position of the objects and a shape at least close to the true one. However, the results in terms of F-score for both the position and the shape are less satisfactory and seem to show a difficulty of the algorithm to detect the correct rotation for a given position.

In order to illustrate the temporal complexity of our approach, we ran the algorithm on a 4096×4096 image containing 480 objects. The optimization with LSP regularization took 6.0 min on a computer with 30GB RAM for estimating a sparse vector of size 2.51×10^8.

4.2. Boat detection on real images

In this paragraph, we apply our algorithm to the problem of boat detection in satellite images of harbors. The original image is part of a satellite image, of size 275×385. As the objects all have the same orientation, we fixed it to 0.6π, both in the MPP model and in 2D-SO. Also, in the MPP model, boats are approximated by ellipses. Hence, we generated a dictionary of binary ellipses with different scaling so as to better fit the discrepancy in boat size.

In order to compare the performances between the ℓ_1 and the log-sum penalties, we tuned the penalty tradeoff parameter λ in (2) so as to activate a similar number of objects, as shown in Table 1. The table also shows the number of non-overlapping objects obtained after a post-processing step keeping the boats with highest activation coefficient, the F-score for the full image compared to a ground truth, the number of steps for the algorithm to reach convergence, and the computation time. For a similar number of objects, it is clear that LSP is more accurate as it has less overlapping objects and a better F-score than the ℓ_1 penalty. It still misses some objects, the correct number being 615, but the computation time is very good and promising.

Figure 7 displays the configurations obtained from the estimation step with SAEM (top right), and from the 2D-SO algorithm with the ℓ_1 and the log-sum penalties (top row). For 2D-SO, the images displayed result from after a post-processing step removing overlapping objects where we keep the objects having the biggest activation coefficient. From this figure, it appears that the detection with the ℓ_1 penalty is not as good as the one with LSP, since it has many spaces where no objects were detected or many objects still overlap. This suggests that running SAEM from an initial configuration obtained with 2D-SO instead of the null configuration might speed up the estimation step and improve the quality of the estimation, which is confirmed by the results in Table 2. Indeed, the estimation step is performed almost twice faster, while the gain in detection rate is of more than 3%. The big difference of computation time between MPP and 2D-SO is mainly due to the fact that MPP tests for many small random perturbations of the objects' shape to find the one that best fits the image. Hence, it results in a fine estimation of the parameters of each ellipse. Note that the detection rate is lower at the end of MPP than at the end of our algorithm. This is a side effect of the parametrization of MPPs fixing the maximum size of the boats, which can still be optimized. Indeed, the MPP program removes the larger boats, while it greatly improves the shape of the smaller boats (see bottom images of Figure 7).

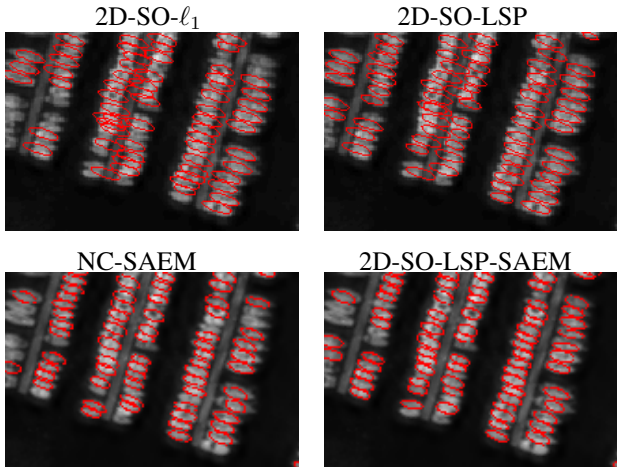

Fig. 7. Detail of the configurations obtained from 2D-SO with ℓ_1 penalty after post-processing (2D-SO-ℓ_1), and from 2D-SO with log-sum penalty after post-processing (2D-SO-LSP), from MPP with null configuration (NC-SAEM) and from MPP initialized with the results of 2D-SO-LSP (2D-SO-SAEM). Best seen in color.

5. CONCLUSION AND FUTURE WORKS

In this work, we presented an original way of treating the problem of detecting objects in an image as a sparse optimization problem. We proposed an efficient algorithm based on an active set strategy that is able to solve the problem in millions of dimensions. The numerical study confirmed that nonconvex penalties result in a more aggressive sparsity and a more accurate detection, and should be preferred over the ℓ_1 penalty.

The next step of our work is to test the scalability of our method on larger images, and its performances on other types of images, such as astronomical or medical images.In the longer-term, we intend to work on the refinement of the dictionary. Several options seem interesting: an interpolation or a vote to select between different atoms for overlapping objects, which can also be done automatically with the Elitist Lasso [16]; and finally, dictionary learning, where both the dictionary and the activations are estimated simultaneously.

References

[1] M. N. M. van Lieshout, *Markov Point Processes And Their Applications*, Imperial College Press, 2000.

[2] S. Ben Hadj, F. Chatelain, X. Descombes, and J. Zerubia, "Parameter estimation for a marked point process within a framework of multidimensional shape extraction from remote sensing images," in *Symp. Photogramm. Comput. Vis. and Image Anal. (PCV)*, Paris, France, September 2010.

[3] G. Celeux and J. Diebolt, "The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem," *Comput. Stat. Q.*, vol. 2, no. 1, pp. 73–82, 1985.

[4] X. Descombes et al., *Stochastic geometry for image analysis*, Wiley/ISTE, 2013.

[5] A. Boisbunon and J. Zerubia, "Estimation of the weight parameter with SAEM for marked point processes applied to object detection," in *Eur. Signal Process. Conf. (EUSIPCO)*, Lisbon, 2014, submitted.

[6] R.J. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Roy. Stat. Soc. B Met.*, vol. 58, no. 1, pp. 267–288, 1996.

[7] E. Candes, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, no. 5-6, pp. 877–905, 2008.

[8] A. Boisbunon, R. Flamary, and A. Rakotomamonjy, "Active set strategy for high-dimensional non-convex sparse optimization problems," in *Int. Conf. Acoust. Spee. (ICASSP)*, 2014.

[9] P. Smaragdis, "Convolutive speech bases and their application to supervised speech separation," *IEEE Trans. Audio Speech*, pp. 1–12, 2007.

[10] P. O'Grady and B. Pearlmutter, "Convolutive non-negative matrix factorisation with sparseness constraint," in *IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, 2006.

[11] J. Besag, "Statistical analysis of non-lattice data," *The Statistician*, vol. 24, no. 3, pp. 179–195, 1975.

[12] B. Delyon, M. Lavielle, and E. Moulines, "Convergence of a stochastic approximation version of the EM algorithm," *Ann. Stat.*, vol. 27, no. 1, pp. 94–128, 1999.

[13] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[14] P. Gong, C. Zhang, Z. Lu, and J. Huang, J. Ye, "A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems," in *Int. Conf. Mach. Learn. (ICML)*, 2013, vol. 28, pp. 37–45.

[15] J. Hopcroft and R. Tarjan, "Algorithm 447: efficient algorithms for graph manipulation," *Commun. ACM*, vol. 16, no. 6, pp. 372–378, 1973.

[16] M. Kowalski and B. Torrésani, "Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients," *Signal, Image and Video Process.*, vol. 3, no. 3, pp. 251–264, 2009.