Electronic Transactions on Numerical Analysis.
Volume 65, pp. 123–139, 2026.
© 2026, the Author(s), licensed via CC BY 4.0.
ISSN 1068–9613.
DOI: 10.1553/etna_vol65s123

ETNA
Kent State University and
Johann Radon Institute (RICAM)

A VARIATIONAL MODEL FOR GLYPH EXTRACTION, THEORETICAL ASPECTS AND IMPLEMENTATION[*]

ANDREA AZZARELLI[†] AND ALESSANDRO BUCCINI[†]

Abstract. A common task in archaeology is extracting incisions or glyphs from a surface. This procedure is usually done manually. Therefore, it is prone to errors and can be extremely time consuming. In this work we present a variational model to automatically extract these incisions. We model this problem as a variational problem, where we wish to split the archaeological surface into two components, a smooth one, containing the background, and a sparse one, containing the glyph. To perform the minimization, we employ the Alternating Direction Method of Multipliers (ADMM). We detail a procedure to generate realistic synthetic data, and we show the performance of the proposed method on this kind of data. Moreover, we test our algorithm on a real-world example and provide a Matlab implementation of it.

Key words. glyph extraction, variational models, alternating direction multiplier method, fractional Laplacian

AMS subject classifications. 65K10, 68U10, 49M27

1. Introduction.

In applications, we are often faced with the problem of separating some details from a background. There are several algorithms and methods to achieve this result, ranging from variational models to machine learning algorithms. In this paper we consider the problem of extracting glyphs from rock surfaces. We refer the interested reader to [13] for a review of the literature regarding this problem and possible approaches for its solution.

In this work we assume that a three-dimensional reconstruction of the archaeological surface is available. This can be obtained, for instance, using photometric stereo techniques; see, e.g., [5, 6] and the references therein. For simplicity, we assume that the surface is “flat” and can be represented as a function $f\colon \mathbb{R}^2 \to \mathbb{R}$, e.g., a bas-relief. However, the proposed approach can be easily extended to a more general case. We denote by $D \in \mathbb{R}^{n_1\times n_2}$ a sampling of the surface on an equispaced and regular grid on a rectangle. Therefore, the entry $D_{i,j}$ satisfies $D_{i,j} = f(x_i, y_j)$ for some grid point (x_i, y_j). Following a similar idea to the one in [11], we assume that D is the sum of a smooth part B, containing the background, and a sparse part G, for which most of its entries vanish and which represents the glyph; namely,

$$D = B + G.$$

We wish to construct a method that decomposes D into its two components. To this end, we first vectorize D reordering its entries in lexicographical order, obtaining $\mathbf{x}_d \in \mathbb{R}^N$, where $N = n_1 n_2$. We formulate the following variational problem:

$$(\mathbf{x}_b^*, \mathbf{x}_g^*) = \underset{\mathbf{x}_b,\mathbf{x}_g\in\mathbb{R}^N}{\arg\min}\ \frac{1}{2}\left\|L^\alpha \mathbf{x}_b\right\|_2^2 + \mu\left\|\mathbf{x}_g\right\|_1, \quad \text{s.t. } \mathbf{x}_d = \mathbf{x}_b + \mathbf{x}_g, \tag{1.1}$$

where $\mu, \alpha > 0$, $L \in \mathbb{R}^{N\times N}$ denotes a discretization of the two-dimensional Laplacian operator (see Section 2.2), and $\|\mathbf{x}\|_p^p = \sum_{j=1}^N |\mathbf{x}_i|^p$, for $p \in \{1, 2\}$. The vectors $\mathbf{x}_b^*$ and $\mathbf{x}_g^*$ contain an approximation of the vectorizations of B and G, respectively. The parameter μ determines how sparse $\mathbf{x}_g^*$ is. In particular, for large values of μ, the vector $\mathbf{x}_g^*$ is extremely

[*]Received September 30, 2025. Accepted February 10. 2026. Published online on March 20, 2026. Recommended by G. Rodriguez.

[†]Corresponding author: A. Buccini. Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy (`{andrea.azzarelli, alessandro.buccini}@unica.it`).

sparse, while small values of μ produce a non-sparse $\mathbf{x}_g^*$. We illustrate this effect in Section 5. The parameter α is tied to a smoothness measure of $\mathbf{x}_b$; see Section 2.2 for more details.

For the solution of the problem (1.1), we apply the Alternating Direction Method of Multipliers (ADMM); see [3] and Section 2.1 for more details. This algorithm decouples the problem, isolating the linear part from the nonlinear one. This, combined with the structure of L, allows for extremely fast computations.

The main contributions of this work are twofold. Firstly, the fractional Laplacian has never been used to solve this kind of problem. Secondly, we detail a way to construct synthetic, but realistic, datasets that can be used to validate the results of our proposed algorithm and other algorithms. Moreover, we provide an almost completely automatic Matlab implementation of our algorithms. To the best of our knowledge, it is the first time that a plug-and-play implementation of a fairly accurate extraction algorithm has been published in an open-source format. The attached software is user-friendly and a powerful tool for archaeologists as well as an accurate benchmark method to evaluate new extraction algorithms. Moreover, we offer software to create realistic data sets. These can be used both to validate new methods and to train machine learning algorithms.

We show the performance of our approach on synthetic data and outline some possible extensions of this method that will be the object of future research. Moreover, we test our method on a real engraving found in Sardinia, Italy; see below for more details.

This paper is structured as follows. In Section 2, we provide some needed background for the formulation of our method. Sections 3 and 4 detail our extraction method and its Matlab implementation, respectively. We test our approach on some numerical examples in Section 5 and describe a procedure for the generation of realistic synthetic data in Section 5.1. Finally, we draw our conclusions in Section 6.

2. Background.

We now provide some background concepts that are required for the formulation of our algorithm.

2.1. The Alternating Direction Method of Multipliers.

The Alternating Direction Method of Multipliers (ADMM) is an iterative algorithm for the solution of constrained minimization problems of the form

$$
\begin{aligned}
(\mathbf{x}^*, \mathbf{z}^*) = \underset{\mathbf{x}\in\mathbb{R}^n, \mathbf{z}\in\mathbb{R}^m}{\arg\min}\; & f(\mathbf{x}) + g(\mathbf{z}), \\
\text{s.t. } & A\mathbf{x} + B\mathbf{z} - \mathbf{c} = \mathbf{0},
\end{aligned}
\tag{2.1}
$$

where $A \in \mathbb{R}^{s\times n}$ and $B \in \mathbb{R}^{s\times m}$ are such that $\begin{bmatrix} A \\ B \end{bmatrix}$ is full-rank, $\mathbf{c} \in \mathbb{R}^s$, $\mathbf{0} \in \mathbb{R}^s$ denotes the zero vector, and $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ are proper, closed, and convex functions.

To formulate the ADMM, we first compute the augmented Lagrangian of (2.1) as

$$
\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}; \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(A\mathbf{x} + B\mathbf{z} - \mathbf{c}) + \frac{\rho}{2}\left\|A\mathbf{x} + B\mathbf{z} - \mathbf{c}\right\|_2^2,
$$

where $\rho > 0$ is a user-defined parameter and $\mathbf{y} \in \mathbb{R}^s$ is the Lagrangian multiplier vector. Obviously a solution of (2.1) can be determined as a saddle point of $\mathcal{L}_\rho$, i.e.,

$$
(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*) = \underset{\mathbf{x}\in\mathbb{R}^n, \mathbf{z}\in\mathbb{R}^m}{\arg\min}\; \max_{\mathbf{y}\in\mathbb{R}^s} \mathcal{L}_\rho(\mathbf{x}, \mathbf{z}; \mathbf{y}).
$$

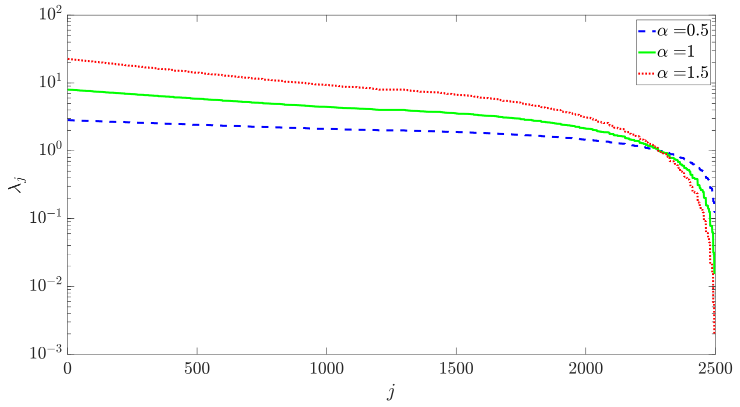

FIG. 2.1. *Eigenvalues of $L^\alpha \in \mathbb{R}^{2500\times 2500}$ for three different choices of α.*

Starting from some initial guesses for $\mathbf{y}^*$ and $\mathbf{z}^*$, we obtain the ADMM iterations by alternating the minimizations with respect to $\mathbf{x}$ and $\mathbf{z}$ and updating the Lagrangian multiplier $\mathbf{y}$

$$\begin{cases} \mathbf{x}^{(k+1)} = \arg\min\limits_{\mathbf{x}\in\mathbb{R}^n} \mathcal{L}_\rho\left(\mathbf{x}, \mathbf{z}^{(k)}; \mathbf{y}^{(k)}\right), \\ \mathbf{z}^{(k+1)} = \arg\min\limits_{\mathbf{z}\in\mathbb{R}^m} \mathcal{L}_\rho\left(\mathbf{x}^{(k+1)}, \mathbf{z}; \mathbf{y}^{(k)}\right), \\ \mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \rho\left(A\mathbf{x}^{(k+1)} + B\mathbf{z}^{(k+1)} - \mathbf{c}\right). \end{cases}$$

The following theorem holds:

THEOREM 2.1 ([3]). *Assume that the extended real-valued functions f and g are closed, proper, and convex. Moreover, assume that the unaugmented Lagrangian $\mathcal{L}_0$ has a saddle point, i.e., that there exists at least a point $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ such that the inequalities*

$$\mathcal{L}_0(\mathbf{x}^*, \mathbf{z}^*; \mathbf{y}) \le \mathcal{L}_0(\mathbf{x}^*, \mathbf{z}^*; \mathbf{y}^*) \le \mathcal{L}_0(\mathbf{x}, \mathbf{z}; \mathbf{y}^*)$$

hold for all $\mathbf{x}$, $\mathbf{z}$, and $\mathbf{y}$. Then the iterates generated by ADMM satisfy the following:

(i) Residual convergence: $\lim_{k\to\infty} \left\|A\mathbf{x}^{(k)} + B\mathbf{z}^{(k)} - \mathbf{c}\right\|_2 = 0$.

(ii) Objective convergence: $\lim_{k\to\infty} f\left(\mathbf{x}^{(k)}\right) + g\left(\mathbf{z}^{(k)}\right) = p^*$*, where p^* is the minimum of $f+g$.*

(iii) Dual variable convergence: $\lim_{k\to\infty} \mathbf{y}^{(k)} = \mathbf{y}^*$*, where $\mathbf{y}^*$ is a dual optimal point.*

We refer the interested reader to [3] for more details on ADMM.

2.2. The fractional Laplacian.

We now describe the fractional Laplacian operator that we use in our method. This operator can be derived in many ways. For simplicity we assume that $n = n_1 = n_2 = \sqrt{N}$. Denote by L_1 the matrix

$$L_1 = \begin{bmatrix} 2 & -1 & & & -1 \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ -1 & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n\times n},$$

and let $I \in \mathbb{R}^{n\times n}$ denote the identity matrix. We define $L = L_1 \otimes I + I \otimes L_1 \in \mathbb{R}^{N\times N}$, where $\otimes$ is the Kronecker product. Since L_1 is a circulant matrix, L is a block circulant with circulant blocks (BCCB) matrix. It is well-known that BCCB matrices are diagonalized by the two-dimensional Fourier matrix $F \in \mathbb{C}^{N\times N}$, i.e.,

$$L = F^*\Lambda F, \tag{2.2}$$

where the superscript $*$ denotes the conjugate transposition and $\Lambda \in \mathbb{R}^{N\times N}$ is a diagonal matrix containing the eigenvalues of L on the diagonal; see [7, Theorem 3.2.1]. Note that the eigenvalues of L are real since L is symmetric. Moreover, by the Gershgorin theorem (see, e.g., [16]), L is positive semidefinite, and therefore all the diagonal entries of Λ are nonnegative.

Since L is diagonalizable, we have that

$$L^\alpha = F^*\Lambda^\alpha F,$$

where $\Lambda^\alpha = \text{diag}\{\lambda_j^\alpha\}$. Note that the diagonal elements of Λ^α are well defined since $\lambda_j \geq 0$ for all j.

The fractional Laplacian can be derived in terms of fractional Sobolev spaces, and for a detailed theoretical analysis we refer the reader to [9] and the references therein. Another possible construction can be obtained in the graph-analysis framework; see, e.g., [1, 2]. Here we provide only an intuitive idea of the role of α and how this operator can be exploited in our method.

Intuitively, small values of α amplify high frequencies and damp low frequencies less than large values of α. In other words, if $\mathbf{x}_{\text{low}}$ is obtained by sampling a slowly oscillating function, then

$$\left\|L^{\alpha_1}\mathbf{x}_{\text{low}}\right\|_2 \geq \left\|L^{\alpha_2}\mathbf{x}_{\text{low}}\right\|_2, \qquad \alpha_1 \leq \alpha_2.$$

On the other hand, if $\mathbf{x}_{\text{high}}$ is obtained by sampling a rapidly oscillating function, then

$$\left\|L^{\alpha_1}\mathbf{x}_{\text{high}}\right\|_2 \leq \left\|L^{\alpha_2}\mathbf{x}_{\text{high}}\right\|_2, \qquad \alpha_1 \leq \alpha_2.$$

This is well illustrated by the behavior of the eigenvalues of L^α shown in Figure 2.1. We recall that, since L is a differential operator, the large and small eigenvalues are associated to high-frequency and low-frequency eigenvectors, respectively.

Thanks to this property of L^α, we are able to adaptively select α depending on the structure of the background. In particular, if the surface on which the glyphs are engraved is extremely smooth, then we pick a large value of α. On the other hand, if the surface is "rough", then we select a smaller value of α.

In Section 3.1 we describe a heuristic rule for the automatic selection of the parameter α. Extensive numerical experience suggests that the proposed rule is able to determine a value of α that produces accurate reconstructions.

3. Our proposal.

We are now in a position to describe our algorithm. We wish to apply the ADMM introduced in Section 2.1 to the minimization problem (1.1). The augmented Lagrangian of the problem is

$$\mathcal{L}_\rho(\mathbf{x}_b, \mathbf{x}_g; \mathbf{y}) = \frac{1}{2}\left\|L^\alpha \mathbf{x}_b\right\|_2^2 + \mu\left\|\mathbf{x}_g\right\|_1 + \mathbf{y}^T(\mathbf{x}_b + \mathbf{x}_g - \mathbf{x}_d) + \frac{\rho}{2}\left\|\mathbf{x}_b + \mathbf{x}_g - \mathbf{x}_d\right\|_2^2,$$

where $\rho > 0$.

Let $\mathbf{x}_g^{(0)} = \mathbf{y}^{(0)} = \mathbf{0} \in \mathbb{R}^N$. Then the ADMM iterations read

$$\begin{cases} \mathbf{x}_b^{(k+1)} = \arg\min_{\mathbf{x}_b} \mathcal{L}_\rho\left(\mathbf{x}_b, \mathbf{x}_g^{(k)}; \mathbf{y}^{(k)}\right), \\ \mathbf{x}_g^{(k+1)} = \arg\min_{\mathbf{x}_g} \mathcal{L}_\rho\left(\mathbf{x}_b^{(k+1)}, \mathbf{x}_g; \mathbf{y}^{(k)}\right), \\ \mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \rho\left(\mathbf{x}_b^{(k+1)} + \mathbf{x}_g^{(k+1)} - \mathbf{x}_d\right). \end{cases}$$

We now discuss how the $\mathbf{x}_b$- and $\mathbf{x}_g$-subproblems can be solved cheaply. Note that this discussion is similar to the one in [4].

We first consider the computation of $\mathbf{x}_b^{(k+1)}$. Disregarding the terms that are constant with respect to $\mathbf{x}_b^{(k+1)}$ in $\mathcal{L}_\rho\left(\mathbf{x}_b, \mathbf{x}_g^{(k)}; \mathbf{y}^{(k)}\right)$, we obtain

$$\begin{aligned} \mathbf{x}_b^{(k+1)} &= \arg\min_{\mathbf{x}_b} \mathcal{L}_p\left(\mathbf{x}_b, \mathbf{x}_g^{(k)}; y^{(k)}\right) \\ &= \arg\min_{\mathbf{x}_b} \frac{1}{2}\|L^\alpha \mathbf{x}_b\|_2^2 + \left(\mathbf{y}^{(k)}\right)^T \mathbf{x}_b + \frac{\rho}{2}\left\|\mathbf{x}_b + \mathbf{x}_g^{(k)} - \mathbf{x}_d\right\|_2^2 \\ &\overset{(a)}{=} \arg\min_{\mathbf{x}_b} \frac{1}{2}\|L^\alpha \mathbf{x}_b\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x}_b - \left(\mathbf{x}_d - \mathbf{x}_g^{(k)} - \frac{\mathbf{y}^{(k)}}{\rho}\right)\right\|_2^2 \\ &= \arg\min_{\mathbf{x}_b} \frac{1}{2}\left\|\begin{bmatrix} L^\alpha \\ \sqrt{\rho} I \end{bmatrix} \mathbf{x}_b - \begin{bmatrix} 0 \\ \sqrt{\rho}\left(\mathbf{x}_d - \mathbf{x}_g^{(k)} - \frac{\mathbf{y}^{(k)}}{\rho}\right) \end{bmatrix}\right\|_2^2 \\ &= \left((L^\alpha)^* L^\alpha + \rho I\right)^{-1}\left(\rho\left(\mathbf{x}_d - \mathbf{x}_g^{(k)}\right) - \mathbf{y}^{(k)}\right) \\ &\overset{(b)}{=} F^*\left(\Lambda^{2\alpha} + \rho I\right)^{-1} F\left(\rho\left(\mathbf{x}_d - \mathbf{x}_g^{(k)}\right) - \mathbf{y}^{(k)}\right), \end{aligned}$$

where (a) is obtained by rearranging the terms and adding some constant terms. Plugging in the spectral decomposition of L and recalling that $\Lambda \in \mathbb{R}^{N \times N}$ is nonnegative yield equality (b). Using the `fft` and `ifft` algorithms, the matrix-vector products with F and F^* can be computed in $O(N \log N)$ floating-point operations, and we can avoid explicitly forming the matrix F. Only the main diagonal of Λ is stored and can be obtained by applying the `fft` algorithm to the first column of L; see, e.g., [12] for a more detailed description on how to perform computations cheaply with BCCB matrices. Overall, the computational cost to compute $\mathbf{x}_b^{(k+1)}$ is $O(N \log N)$.

We now consider the computation of $\mathbf{x}_g^{(k+1)}$. Similarly as above, we obtain

$$\begin{aligned} \mathbf{x}_g^{(k+1)} &= \arg\min_{\mathbf{x}_g} \mathcal{L}_p\left(\mathbf{x}_b^{(k+1)}, \mathbf{x}_g; y^{(k)}\right) \\ &= \arg\min_{\mathbf{x}_g} \mu\|\mathbf{x}_g\|_1 + \left(\mathbf{y}^{(k)}\right)^T \mathbf{x}_g + \frac{\rho}{2}\left\|\mathbf{x}_b^{(k+1)} + \mathbf{x}_g - \mathbf{x}_d\right\|_2^2 \\ &= \arg\min_{\mathbf{x}_g} \mu\|\mathbf{x}_g\|_1 + \frac{\rho}{2}\left\|\mathbf{x}_g - \left(\mathbf{x}_d - \mathbf{x}_b^{(k+1)} - \frac{\mathbf{y}^{(k)}}{\rho}\right)\right\|_2^2 \\ &= \arg\min_{\mathbf{x}_g} \frac{1}{2}\left\|\mathbf{x}_g - \left(\mathbf{x}_d - \mathbf{x}_b^{(k+1)} - \frac{\mathbf{y}^{(k)}}{\rho}\right)\right\|_2^2 + \frac{\mu}{\rho}\|\mathbf{x}_g\|_1 \\ &= S_{\frac{\mu}{\rho}}\left(\mathbf{x}_d - \mathbf{x}_b^{(k+1)} - \frac{\mathbf{y}^{(k)}}{\rho}\right), \end{aligned}$$

where

$$S_\beta(\mathbf{z}) = \max\{|\mathbf{z}| - \beta, 0\} \cdot \operatorname{sign}(\mathbf{z})$$

and all the operations are meant element-wise. The operator $S_\beta(\mathbf{z})$ is the so-called soft-thresholding operator and is the proximal operator of the ℓ^1-norm. It is trivial to see that the cost for the computation of $\mathbf{x}_g^{(k+1)}$ is $O(N)$ operations.

We stop the iterations when both $\mathbf{x}_b^{(k)}$ and $\mathbf{x}_g^{(k)}$ no longer change significantly, namely, as soon as

$$\left\|\mathbf{x}_b^{(k+1)} - \mathbf{x}_b^{(k)}\right\| \le \tau \left\|\mathbf{x}_b^{(k)}\right\| \qquad \text{and} \qquad \left\|\mathbf{x}_g^{(k+1)} - \mathbf{x}_g^{(k)}\right\| \le \tau \left\|\mathbf{x}_g^{(k)}\right\|,$$

where $\tau > 0$ is a user-defined tolerance, or when a maximum number of iterations K is reached.

Note that the assumptions of Theorem 2.1 are trivially satisfied, and therefore the ADMM iterations applied to our problem converge to a solution of (1.1).

We summarize the computations in Algorithm 1.

Algorithm 1: ADMM for glyph extraction.

```
1  Let $\mathbf{x} \in \mathbb{R}^{N\times N}$. Fix the parameters $\alpha, \mu, \rho > 0$, a maximum number of iterations $K$,
     and a tolerance $\tau > 0$.;
2  Compute and store the main diagonal of $\Lambda$;
3  $\mathbf{x}_g^{(0)} = \mathbf{z}$;
4  $\mathbf{y}^{(0)} = \mathbf{z}$;
5  for $k = 0, 1, \dots, K$ do
6      $\mathbf{x}_b^{(k+1)} = F^*\left((\Lambda^{2\alpha} + \rho I)^{-1} F\left(\rho\left(\mathbf{x}_d - \mathbf{x}_g^{(k)}\right) - \mathbf{y}^{(k)}\right)\right)$;
7      $\mathbf{x}_g^{(k+1)} = S_{\frac{\mu}{\rho}}\left(\mathbf{x}_d - \mathbf{x}_b^{(k+1)} - \frac{\mathbf{y}^{(k)}}{\rho}\right)$;
8      if $\left\|\mathbf{x}_b^{(k+1)} - \mathbf{x}_b^{(k)}\right\| \le \tau\left\|\mathbf{x}_b^{(k)}\right\|$ and $\left\|\mathbf{x}_g^{(k+1)} - \mathbf{x}_g^{(k)}\right\| \le \tau\left\|\mathbf{x}_g^{(k)}\right\|$ then
9          exit;
10     end
11     $\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \rho\left(\mathbf{x}_b^{(k+1)} + \mathbf{x}_g^{(k+1)} - \mathbf{x}_d\right)$;
12 end
```

3.1. Selection of α. We now detail a heuristic automatic rule for the selection of the parameter α.

There are many possible criteria for the selection of the parameter α. We propose to select α such that

$$\alpha^* = \underset{\alpha \ge 0}{\arg\min} \left\|L^\alpha \mathbf{x}_d\right\|_2^2. \tag{3.1}$$

We now discuss when the minimized function in (3.1) is strictly convex, i.e., when the solution of the minimization problem (3.1) is unique.

PROPOSITION 3.1. *Let $G(\alpha) = \|L^\alpha \mathbf{x}\|_2^2$ and $\tilde{\mathbf{x}} = F\mathbf{x}$, where L and F are defined in (2.2). Denote by λ_j, for $j \in J = \{1, 2, \ldots, N\} \subset \mathbb{N}$, the eigenvalues of the matrix L, and define the following partition of the set J:*

$$J^0 = \{j \in J \mid \lambda_j = 0\}, \qquad J^- = \{j \in J \mid 0 < \lambda_j < 1\},$$
$$J^1 = \{j \in J \mid \lambda_j = 1\}, \qquad J^+ = \{j \in J \mid \lambda_j > 1\}.$$

The following hold:

(i) $G(\alpha)$ is convex for all $\mathbf{x} \in \mathbb{R}^N$.
(ii) If there exists a $j \in J^+ \cup J^-$ such that $\tilde{\mathbf{x}}_j \neq 0$, then $G(\alpha)$ is strictly convex. Therefore, α^* in (3.1) is unique.
(iii) If there exists $j \in J^+$ such that $\tilde{\mathbf{x}}_j \neq 0$, then α^* in (3.1) is finite.

Proof. Using the spectral decomposition of the matrix L, we can rewrite the function G as

$$G(\alpha) = \|L^\alpha \mathbf{x}\|_2^2 = \|F^* \Lambda^\alpha F\mathbf{x}\|_2^2 = \|\Lambda^\alpha \tilde{\mathbf{x}}\|_2^2 = \sum_{j=1}^{N} (\lambda_j^\alpha |\tilde{\mathbf{x}}_j|)^2 = \sum_{j\in J} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2, \tag{3.2}$$

where we used that F^* is unitary. Since G is a sum of convex functions, it is convex, proving (i).

We now show (ii). Let us split (3.2) into four summations using the partition of the set J.

$$\begin{aligned} G(\alpha) &= \sum_{j\in J^0} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 + \sum_{j\in J^-} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 + \sum_{j\in J^1} |\tilde{\mathbf{x}}_j|^2 + \sum_{j\in J^+} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 \\ &= \sum_{j\in J^-} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 + \sum_{j\in J^1} |\tilde{\mathbf{x}}_j|^2 + \sum_{j\in J^+} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2, \end{aligned} \tag{3.3}$$

where the second equality follows from the observation that, for $j \in J^0$, the sum vanishes. The first and third summations are strictly convex functions, therefore the overall sum is strictly convex if and only if there exists $j \in J^+ \cup J^-$ such that $\tilde{\mathbf{x}}_j \neq 0$.

We now show (iii). Consider the limits of the three sums in (3.3), for $\alpha \to \infty$. It holds that

$$\lim_{\alpha\to\infty} \sum_{j\in J^-} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 = 0,$$
$$\lim_{\alpha\to\infty} \sum_{j\in J^1} |\tilde{\mathbf{x}}_j|^2 = \sum_{j\in J^1} |\tilde{\mathbf{x}}_j|^2,$$
$$\lim_{\alpha\to\infty} \sum_{j\in J^+} \lambda_j^{2\alpha} |\tilde{\mathbf{x}}_j|^2 = \infty.$$

Therefore, we have

$$\lim_{\alpha\to\infty} G(\alpha) = \begin{cases} \infty & \text{if } \exists j \in J^+ \text{ such that } \tilde{\mathbf{x}}_j \neq 0, \\ \sum_{j\in J^1} |\tilde{\mathbf{x}}_j|^2 & \text{if } \forall j \in J^+,\ \tilde{\mathbf{x}}_j = 0. \end{cases}$$

This shows that, if there exists $j \in J^+$ such that $\tilde{\mathbf{x}}_j \neq 0$, then α^* in (3.1), which exists and is unique thanks to (i) and (ii), has to be finite, concluding the proof. □

We now wish to provide an intuitive justification of our heuristic criterion. If we assume that the glyph is not too large, then we can expect

$$\|L^\alpha \mathbf{x}_b\|_2^2 \approx \|L^\alpha \mathbf{x}_d\|_2^2,$$

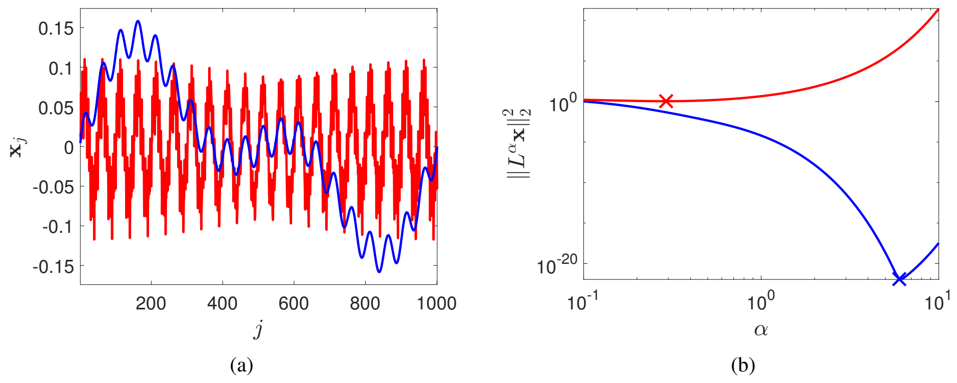

FIG. 3.1. *Comparison of the evolution of* $\|L^\alpha \mathbf{x}\|_2^2$ *against* α *for a low-frequency (blue) and a high-frequency (red) one-dimensional signals. Panel (a) reports the vectors* $\mathbf{x}$*, and panel (b) shows the evolution of* $\|L^\alpha \mathbf{x}\|_2^2$ *against* α*.*

i.e., that we can approximate the (unknown) 2-norm of $L^\alpha \mathbf{x}_b$ with the (known) one of $L^\alpha \mathbf{x}_d$. We wish to select a value of α such that $\|L^\alpha \mathbf{x}_b\|_2^2$ is as small as possible, so that all the "components" pertaining to $\mathbf{x}_b$ are correctly recognized in (1.1). Therefore, as long as we can approximate $\|L^\alpha \mathbf{x}_b\|_2^2$ with $\|L^\alpha \mathbf{x}_d\|_2^2$, we can expect that a "good" value of α is determined by (3.1).

In Figure 3.1 we consider a one-dimensional problem and two different signals, one that is mostly composed of low-frequency components and one that oscillates. In Figure 3.1b we report the evolution of $\|L^\alpha \mathbf{x}\|_2^2$ against α. We can observe that, as expected, for the slowly-oscillating function, the α that minimizes $\|L^\alpha \mathbf{x}\|_2^2$ is extremely large (roughly $\alpha \approx 6$), while for the highly-oscillating function, our criterion determines a small value of α (roughly $\alpha \approx 0.3$). This confirms our observation above: if the background is smooth, then we wish to pick a large value for α, while if the background is "rough", then we wish to select a small value of α.

4. Software.

This section describes the software accompanying this paper[1]. We implement a function that performs the splitting and which can be called as follows:

```
[B,G,OutInfo] = splitImageFract(D,mu,options);
```

Here, `D` is the matrix (or surface) to be split, and `mu` is the parameter μ as in equation (1.1). The optional input argument `options` contains additional parameters that control the behavior of the algorithm. These are listed in Table 4.1.

The function produces as outputs the two components of the surface, `B` and `G`, together with a structure, `OutInfo`, which contains information about the behavior of the algorithm; see Table 4.2.

[1]Available at
https://etna.ricam.oeaw.ac.at/volumes/2021-2030/vol65/addition/p123.php

TABLE 4.1
`options` *fields*

Field	Description	Accepted values
`alpha`	fractional exponent of the Laplacian	$0 < \alpha < \infty$ (default see (3.1))
`rho`	augmentation parameter for the ADMM	$0 < \rho < \infty$ (default $\rho = 1$)
`iter`	maximum number of iterations	`iter` $\in \mathbb{N}$ (default `iter`=100)
`tol`	tolerance of the stopping criterion; set `tol`=0 to force the method to perform all the iterations	$0 \leq$ `tol` < 1 (default `tol`=10^{-4})
`waitbar`	flag to enable a waitbar. If this field is equal to `'on'`, then a waitbar is displayed	`'on'`, `'off'` (default `waitbar` `='off')`

TABLE 4.2
`OutInfo` *fields.*

Field	Description
`alpha`	fractional exponent of the Laplacian
`J`	value of the minimized functional at each iteration
`cstr`	value of the residual at each iteration $\left\|D - B^{(k)} - G^{(k)}\right\|_F$
`iter`	number of performed iterations

4.1. Supplementary software. In Section 5.1 we describe the construction of our synthetic examples. This task is performed by the function

```
        D = createData(S,G,sigma,nu);
```

which takes as input the exact glyph `G`, a smooth background surface `S`, and two parameters, `sigma` and `nu`. These parameters control, respectively, the width and the height of the bumps added to the background in order to obtain a rougher surface (see Section 5.1). The output `D` represents the synthetic data to be used as input for the algorithm.

5. Numerical examples. The algorithm was tested on numerous synthetic examples. In this section we show some selected ones to highlight the performance of the proposed method. We also describe how we constructed the synthetic data sets used to test our method. We then consider three examples that show the performance of our method in different scenarios.

Determining if a provided reconstruction is satisfactory is not trivial. Simply looking at the distance in some norm between the exact background and the glyph from the reconstructed ones usually does not indicate whether the computed solution is satisfactory. In applications we are mostly interested in detecting the location of the glyph rather than computing its exact "depth". Once the glyph has been localized, it is a trivial task to extract it. Therefore, we consider two different measures for the reconstruction error. Let $\mathbf{x}_g^*$ denote the vectorization of the exact glyph. We define

$$f_p(\mathbf{x}_g) = \left\| \frac{\mathbf{x}_g}{\|\mathbf{x}_g\|_p} - \frac{\mathbf{x}_g^*}{\|\mathbf{x}_g^*\|_p} \right\|_p , \tag{5.1}$$

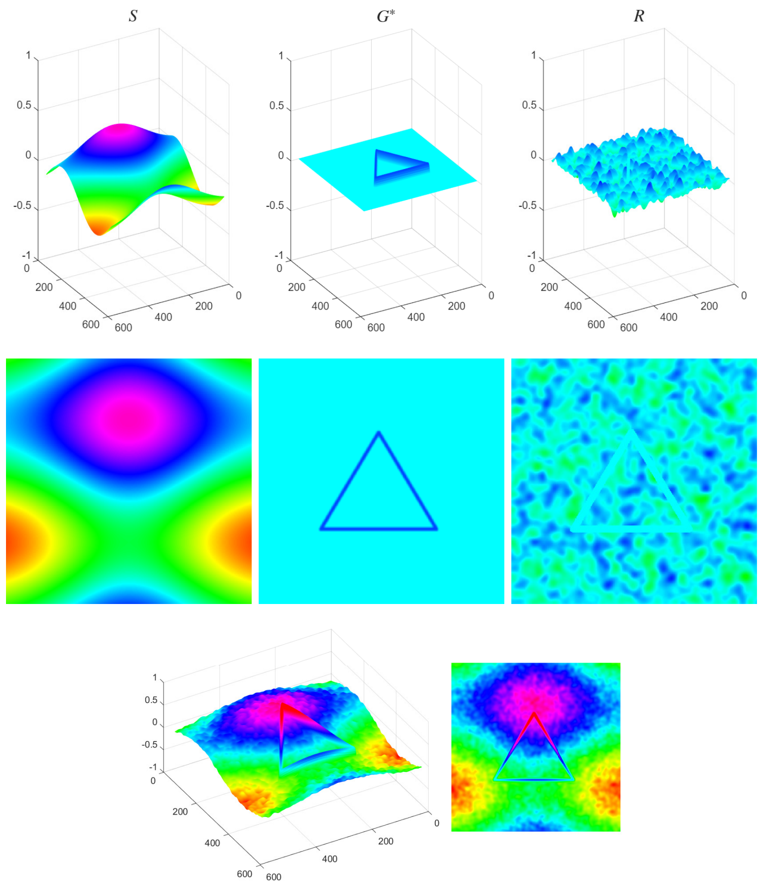

FIG. 5.1. *Construction of the synthetic data. The first two rows report the three components S, G, and R. The last row displays the assembled artificial glyph.*

where $p \in \{1,\ 2\}$. Note that, if the computed glyph satisfies $\mathbf{x}_g = \mathbf{0}$, then the measure is not defined. In the latter case, using Matlab notation, we denote the error by NaN (`Not a Number`).

For all synthetic experiments below we set the stopping criterion's parameters to $K = 2000$ and $\tau = 10^{-6}$. The values of ρ and μ are problem-dependent, and an investigation on how to automatically select them will be the matter of future research. All computations were performed using Matlab with roughly 15 decimal digits of precision.

5.1. Creation of synthetic data. We now describe how we constructed test data sets. The test surfaces are the sum of three components:

- **The discretization of a smooth surface S.** The aim of this component is to create macro areas characterized by different curvature, i.e., convex and concave regions with relative minima, maxima, and saddle points. In our experiments, we define the function S on a grid (x_i, y_j) equally spaced in $[-1, 1] \times [-1, 1]$, $i = 1 \ldots n_1$, $j = 1 \ldots n_2$.
- **A sparse glyph G^*.** This is the simulation of an incision in the surface S.
- **A roughness component R.** Since engravings are usually not made on polished surfaces, we add a component that simulates some asperities in the following way:

TABLE 5.1
Circumference test case: $f_2(G)$ for the computed reconstructions.

μ \ α	**0.01**	**0.1**	**0.5**	**1**	**5**
0.01	1.31	1.31	1.01	0.78	0.67
0.1	1.31	1.30	0.71	0.64	0.63
1	1.29	0.89	0.63	0.55	0.61
10	NaN	NaN	NaN	0.77	0.59

(a) Without the roughness component.

μ \ α	**0.01**	**0.1**	**0.5**	**1**	**5**
0.01	1.32	1.32	1.20	1.14	0.97
0.1	1.32	1.30	1.05	0.95	0.89
1	1.30	1.07	0.67	0.58	0.80
10	NaN	NaN	NaN	0.78	0.70

(b) With the roughness component.

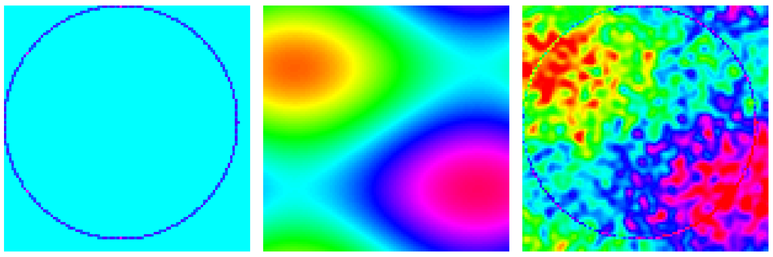
(a) (b) (c)

FIG. 5.2. *Circumference test case: (a) glyph G, (b) assembled surface with no roughness component, i.e., $S + G$, (c) assembled surface with roughness component, i.e., $S + G + R$.*

Given a random matrix E, i.e., whose entries are independent realizations of a Gaussian random variable with zero mean, we reproduce a rough surface by convolving it with a Gaussian filter of standard deviation σ. We add this component outside of the glyph, i.e., we set $E_{i,j} = 0$ if $G_{i,j} \neq 0$. The final surface is obtained as

$$R = \nu \left\| G^* + S \right\|_F E,$$

where $\nu \in [0, 1]$ is a parameter that allows us to adjust the amount of roughness and $\left\| \cdot \right\|_F$ denotes the Frobenius norm.

We provide an example in Figure 5.1. Our aim is to separate $B = S + R$ from G^*.

5.2. Examples.

Circumference. The first glyph we consider is a 1-pixel circumference. In Figure 5.2a we display the glyph. This example should be straightforward to solve since the glyph is extremely sparse and "sharp"; therefore it can be quite easily distinguished from the background surface.

We position the circumference tangent to the top and left edge of the considered area. Our purpose is to show the effect of the chosen boundary conditions in L on the computed solution.

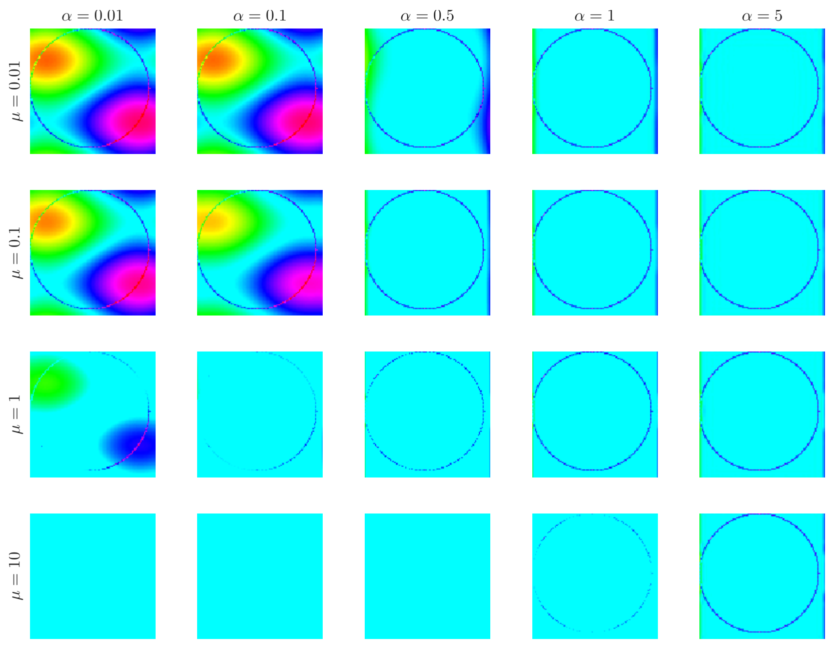

FIG. 5.3. *Circumference test case ($\nu = 0$): reconstructed glyphs obtained with different values for α and μ.*

We set $n_1 = n_2 = 100$, and the discretization of the background surface S is obtained as

$$S_{i,j} = \sin\left(\tfrac{2}{3}\pi x_i\right) + \sin\left(\pi y_j\right),$$

where, as we mentioned above, x_i and y_i are equispaced points in $[-1, 1]$.

We first test the proposed method for several choices of the parameters α and μ, while, for all cases, we set the parameter ρ to 100. For this example we consider two scenarios: one without the roughness component R (Figure 5.2b) and one with it (Figure 5.2c).

We report some reconstructions of the glyph G in the first case, i.e., for $\nu = 0$, in Figure 5.3. We can observe that for several choices of the two parameters, the obtained reconstructions are quite accurate. However, some artifacts are present on the boundaries. This suggests that the selection of the boundary conditions in L has some limited, albeit nontrivial, effect on the computed solutions. From this observation we can deduce that, if possible, it is desirable to position the glyph far from the boundaries of the considered area. This is usually not an issue in real applications. However, if this is not possible, more accurate boundary conditions for L can be considered, e.g., reflexive [12, 14], antireflexive [10, 15], and high-order boundary conditions [8].

We can observe that larger values of α produce more accurate and stable reconstructions with respect to the choice of μ. This confirms our observations that if the background is smooth, then we need to pick a large value of α. This is also supported by the values of f_2 in Table 5.1.

We now add the roughness component R, setting $\sigma = 1.5$ and $\nu = 0.4$. In Figure 5.4 we display some of the computed reconstructions. Due to the presence of the R-component, the choice of the parameters α and μ becomes crucial. We can observe that an imprudent choice of these parameters may lead to poor reconstructions. We report the values of f_2 in Table 5.1. To select the parameter α, we use the criterion in Section 3.1. We show the evolution of $\|L^\alpha \mathbf{x}_d\|_2^2$ against α in Figure 5.5. We can observe that, among the tested ones, our criterion selects

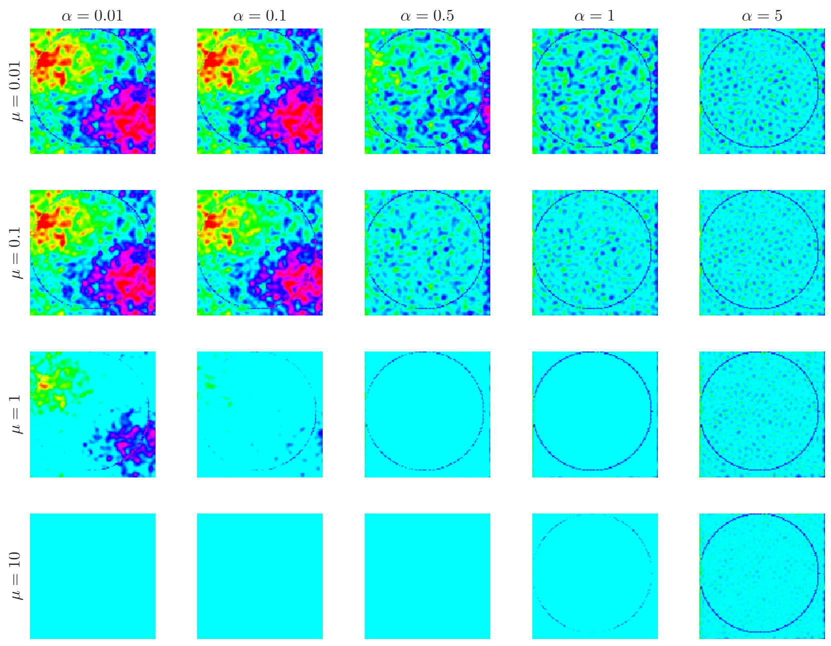

FIG. 5.4. *Circumference test case ($\nu = 0.4$): reconstructed glyphs obtained with different values for α and μ.*

$\alpha = 0.5$. This is not the optimal value of α; nevertheless, the solution computed with this choice of α, selecting a proper value of μ, is satisfactory, and we are able to determine a good approximation of the exact glyph.

Triangle. In this problem we simulate a more realistic incision of a triangle. In particular, differently from the example above, we smooth out the glyph. This is obtained by constructing a sharp surface and convolving it with a Gaussian kernel.

In this example we set $n_1 = n_2 = 560$ and define the background S by

$$S_{i,j} = (0.3)e^{-\frac{x_i^2}{2}} - (0.2)\sin(\pi y_j) - 0.2,$$

where, similarly as above, x_i and y_j are equispaced points between -1 and 1. The roughness component R is obtained by setting $\sigma = 8$ and $\nu = 0.2$. The resulting test example is displayed in Figure 5.1.

The criterion (3.1) selects $\alpha \approx 0.75$. To assess if this choice is close to the optimal one, we compute the values of f_p, for both $p = 1$ and $p = 2$, for $\alpha \in \{0.3,\ 0.75,\ 1,\ 2\}$, and for 50 values of μ logarithmically equispaced in $[10^{-7}, 1]$. In Figure 5.7 we report the evolution of f_p. We can observe that the two metrics behave similarly. We can also state that the value of α selected by (3.1) produces accurate approximations of the glyph for a good choice of the parameter μ. However, the most accurate reconstruction is obtained with $\alpha = 0.3$, but the range of good choices of μ in this case is very narrow. The visual inspection of the computed solutions in Figure 5.7 confirms that the best reconstruction is obtained in the latter case. However, we can observe that the reconstructions obtained for $\alpha = 0.3$ with small values of μ are useless, while the reconstructions obtained with our selection of α are more stable with respect to the choice of μ.

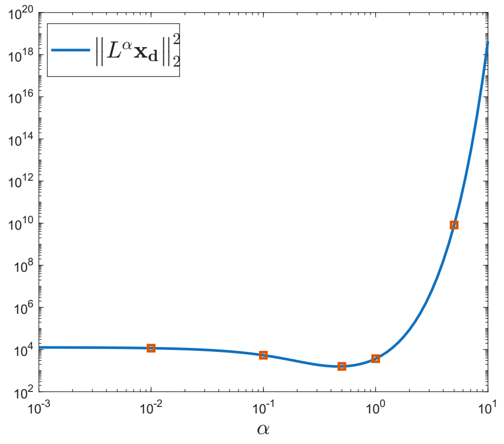

FIG. 5.5. *Circumference test case ($\nu = 0.4$): evolution of $\|L^\alpha \mathbf{x}_d\|_2^2$ against α.*

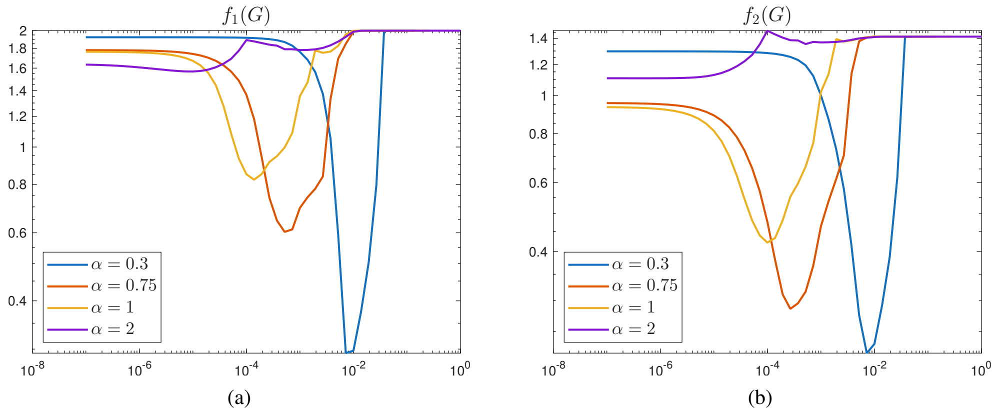

FIG. 5.6. *Triangle test case: evolution of the errors f_p against μ for four values of α. Panel (a) reports the results for $p = 1$, while panel (b) shows them for $p = 2$.*

Writings. We now wish to test the behavior of our algorithmic proposal on a petroglyph with mixed characteristics, i.e., less homogeneous than the one in the previous examples. We have created a test image that contains different types of writings; see Figure 5.8a. The glyph is composed by

- a row of thin letters,
- a row of bold letters,
- a row of shadowed numbers,
- a row of shapes and symbols both solid and empty,
- a row with a rough incision of Roman numerals,
- a row with an italic writing three times shallower than the others.

In this example we set $n_1 = n_2 = 600$ and define the background surface S by

$$S_{i,j} = (0.3)e^{-8x_i^2} - (0.2)\sin(\pi y_j) + 2.$$

We set the parameters of the filter used to create the surface R to $\sigma = 8$ and $\nu = 0.1$. The final test surface is displayed in Figure 5.8b.

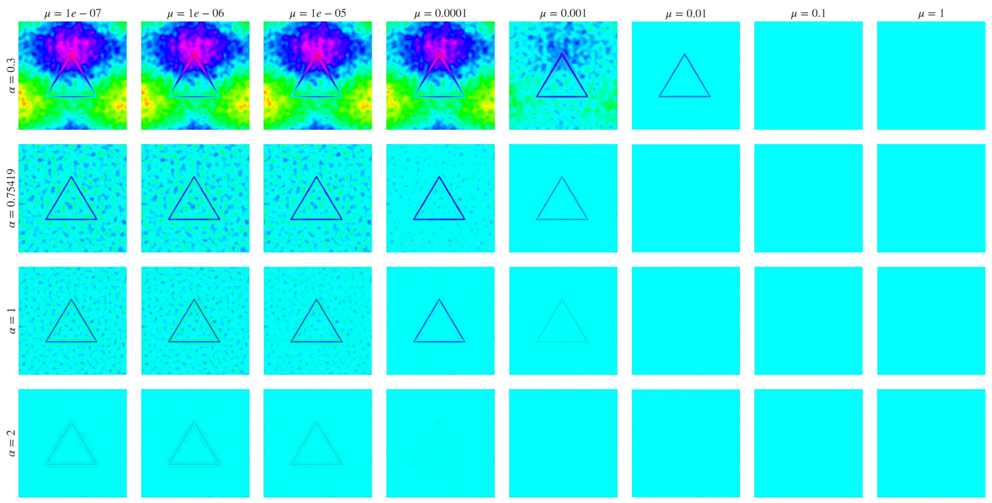

FIG. 5.7. *Triangle test case: reconstructed glyphs obtained with different values for α and μ.*

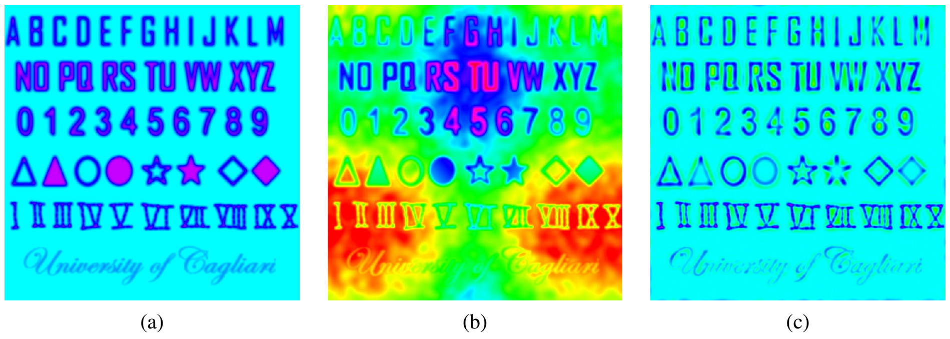

FIG. 5.8. *Writings test case: (a) glyph G, (b) assembled surface, (c) reconstruction obtained with $\alpha = 1.09$ and $\mu = 10^{-4}$.*

Using the criterion presented in Section 3.1, we obtain the parameter $\alpha = 1.09$. We hand-tune the parameter μ so that it minimizes the error metric (5.1) (see Table (5.1)), obtaining $\mu = 10^{-4}$. Note that, in this case, both metrics agree on which reconstruction is the optimal one. In Figure 5.8c we display the resulting reconstruction. From the visual inspection of the reconstruction we can observe that the algorithm was able to correctly identify the glyph with little artifacts around the edges. The main downfall of the computed approximation is that we failed to detect the filled surfaces as glyphs and rather reconstruct only the edges. This is due to the fact that the algorithm is designed to spot sudden jumps in the analyzed surface and not flat regions between them. It will be a matter of future research how to incorporate this kind of glyphs among the identifiable ones.

Application to real data. Finally, we apply our algorithm to real data. The starting surface, displayed in Figure 5.9a, is one of the surfaces reconstructed in [6]: *Domus de Janas di Corongiu*, Pimentel, (Italy). In order to avoid artifacts due to boundary conditions, we apply the algorithm to a larger portion of the surface and then crop the edges. We set the parameter $\alpha \approx 0.517$ according to the criterion (3.1) and select $\mu = 3 \times 10^{-4}$ by trial and error as the

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 5.2
Writings test case: $f_p(\mathbf{x}_g)$ with $p \in \{1, 2\}$ for the computed reconstructions.

p \ μ	$\mathbf{10^{-6}}$	$\mathbf{10^{-5}}$	$\mathbf{10^{-4}}$	$\mathbf{10^{-3}}$	$\mathbf{10^{-2}}$	$\mathbf{10^{-1}}$
1	1.136	1.049	0.914	1.321	1.932	1.999
2	0.910	0.889	0.893	1.165	1.401	1.414

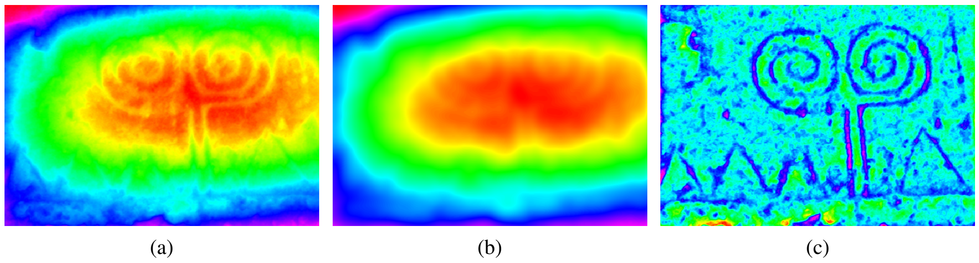
(a) (b) (c)

FIG. 5.9. *Domus de Janas di Corongiu, Pimentel, (Italy): (a) starting surface, (b) smooth component, (c) recovered glyph.*

one that produces better results visually. The hyperparameters of the algorithm were set to $K = 200$ and $\tau = 10^{-6}$.

In Figures 5.9b and 5.9c we report the computed $\mathbf{x}_b$ and $\mathbf{x}_g$, respectively. We can observe that the algorithm was able to accurately recover the engraving, removing the underlying smooth surface. The obtained result is extremely satisfactory, especially considering the roughness of the rock and the condition of the glyph after thousands of years. We would like to stress that we are able to deal with the roughness of the rock thanks to the use of the fractional Laplacian. This is confirmed by the fact that the selected parameter α is not an integer.

6. Conclusions.

In this paper we propose an automatic algorithm for the detection of petroglyphs. The method is able to separate even complex glyphs from rough surfaces without requiring the tuning of many parameters. We show the performance of our method on several examples showcasing the strengths and limits of our algorithmic proposal. Finally, we propose an approach to create completely synthetic, but realistic, datasets for testing algorithms. We apply the algorithm to a real example, and we obtain very satisfactory results also in this case. Moreover, we provide Matlab codes that are almost completely automatic to implement our algorithmic proposals.

Matters of future research include the development of completely automatic methods and the use of more complex priors to further improve the accuracy of our method and enlarge the class of recoverable glyphs. Moreover, different boundary conditions for the fractional Laplacian operator will be investigated like reflexive boundary conditions [14], antireflexive boundary conditions [10, 15], and high-order boundary conditions [8].

Supplementary material. The accompanying software is available at
`https://etna.ricam.oeaw.ac.at/volumes/2021-2030/vol65/addition/p123.php`
in form of a compressed file entitled splitImageFract.zip. Installation details are discussed in the file README.md .

Acknowledgments. The authors would like to thanks E. Crabu, F. Pes, G. Rodriguez, and G. Tanda for providing the data from [6] and for the precious discussions.

The authors are member of the GNCS group of INdAM that partially founded this work with the GNCS Project 2024 “Algebra lineare numerica per problemi di grandi dimensioni: aspetti teorici e applicazioni” (CUP E53C23001670001) and with the GNCS Project 2025 “Tecniche numeriche per problemi di grandi dimensioni” (CUP E53C24001950001).

A. B. is partially supported by the PRIN 2022 PNRR project no. P2022PMEN2 “A mathematical approach to inverse problems arising in cultural heritage preservation and dissemination” financed by the European Union-NextGenerationEU and by the Italian Ministry of University and Research (MUR), by the PRIN 2022 project “Inverse Problems in the Imaging Sciences (IPIS)” (2022ANC8HL) financed by the European Union-NextGenerationEU and by the Italian Ministry of University and Research (MUR), and by “GraphNet: models, computation and estimation in networks and graph analysis” funded by D.M. 737/2021 - Linea d’intervento Iniziative di ricerca interdisciplinare su temi di rilievo trasversale per il PNR (grant number F25F21002720001).

REFERENCES

[1] S. Aleotti, A. Buccini, and M. Donatelli, *Fractional graph Laplacian for image reconstruction*, Appl. Numer. Math., 200 (2024), pp. 43–57.
[2] M. Benzi, D. Bertaccini, F. Durastante, and I. Simunec, *Non-local network dynamics via fractional graph Laplacians*, J. Complex Netw., 8 (2020), Paper No. cnaa017, 29 pages.
[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.
[4] R. H. Chan, M. Tao, and X. Yuan, *Constrained total variation deblurring models and fast algorithms based on alternating direction method of multipliers*, SIAM J. Imaging Sci., 6 (2013), pp. 680–697.
[5] A. Concas, R. Dessì, C. Fenu, G. Rodriguez, and M. Vanzi, *Identifying the lights position in photometric stereo under unknown lighting*, in 2021 21st International Conference on Computational Science and Its Applications (ICCSA), IEEE Conference Proceedings, Los Alamitos, 2021, pp. 10–20.
[6] E. Crabu, F. Pes, G. Rodriguez, and G. Tanda, *Ascertaining the ideality of photometric stereo datasets under unknown lighting*, Algorithms, 16 (2023), Paper No. 375, 15 pages.
[7] P. Davis, *Circulant Matrices*, Wiley, New York, 1979.
[8] P. Dell’Acqua, *A note on Taylor boundary conditions for accurate image restoration*, Adv. Comput. Math., 43 (2017), pp. 1283–1304.
[9] E. Di Nezza, G. Palatucci, and E. Valdinoci, *Hitchhiker’s guide to the fractional Sobolev spaces*, Bull. Sci. Math., 136 (2012), pp. 521–573.
[10] M. Donatelli and S. Serra-Capizzano, *Antireflective boundary conditions for deblurring problems*, J. Electr. Comput. Eng., (2010), Paper No. 241467, 18 pages.
[11] A. Gholami and S. Gazzola, *Automatic balancing parameter selection for Tikhonov-TV regularization*, BIT, 62 (2022), pp. 1873–1898.
[12] P. C. Hansen, J. G. Nagy, and D. P. O’Leary, *Deblurring Images: Matrices, Spectra, and Filtering*, SIAM, Philadelphia, 2006.
[13] L. Maddalena, F. Gregoretti, and L. Antonelli, *Petroglyph segmentation: an overview*, Tech. Rep. RT-ICAR-NA-2024-01, Institute for High-Performance Computing and Networking, National Research Council of Italy, April 2024.
[14] M. K. Ng, R. H. Chan, and W.-C. Tang, *A fast algorithm for deblurring models with Neumann boundary conditions*, SIAM J. Sci. Comput., 21 (1999), pp. 851–866.
[15] S. Serra-Capizzano, *A note on antireflective boundary conditions and fast deblurring models*, SIAM J. Sci. Comput., 25 (2003/04), pp. 1307–1325.
[16] R. S. Varga, *Geršgorin and His Circles*, Springer, Berlin, 2004.