# A Heuristic Spline Interpolation Method on Signal Denoising

Therese Anne G. Basco-Uy
Department of Physical Sciences and Mathematics
University of the Philippines-Manila
theresebasco@gmail.com

Marrick C. Neri
Institute of Mathematics
University of the Philippines-Diliman
mcneri@up.edu.ph

## ABSTRACT

In this paper, we apply in a discrete setting a modification of the Spline Contact Problem (SCP) and the Taut-String Algorithm (TSA) found in (G. Steidl 2006), utilizing cubic splines linked with a heuristic step. This process aims to keep the splines within the tube region centered at the antiderivative of the observed signal data with radius equal to the regularization parameter $\lambda$. The resulting spline serves as the antiderivative of a solution to a minimization problem with quadratic data term and a total variation (TV) regularization term. The heuristic spline method proposed aims to improve recovery of corners and reduce the staircasing effect in the solution compared to that obtained when TSA is applied.

**Keywords:** Signal denoising, variation model, splines, heuristic method

## 1. INTRODUCTION

In this paper, we consider an observed signal represented by a function $f : E \to \mathbb{R}$, with $E \subset \mathbb{R}$. We are interested in obtaining a solution $u$ that is $m$-times continuously differentiable that will minimize a discrete version of the following functional:

$$\frac{1}{2}\int_E (u(x) - f(x))^2 + \lambda|\nabla u^{(m)}(x)|\, dx. \qquad (1)$$

The first term is referred to as the data fidelity term which promotes the closeness of the solution $u$ to the observed data $f$. The second term, commonly referred to as the penalty term, represents the total variation of $u$ which contributes to the preservation of the edges and smoothing of flat zones, i.e, with constant values, in the data. The constant $\lambda > 0$ controls the influence of the second term in the solution, while $m$ is the order of differentiation. This is a general version of the classical model proposed by Rudin, Osher, and Fatemi in [10] called the ROF model and is widely used for its applications in digital image processing.

In this paper, all computations are done in a discrete setting which is ideal for tools and operations used in digital signal and image processing. We focus our attention to the following discretized version of (1):

$$M(u) = \frac{1}{2}\|f - u\|_2^2 + \lambda\|J(u)\|_1, \qquad (2)$$

where $u$ and $f$ are $n\times 1$ vectors and $J(u)$ is a discrete version of the derivative of $u$.

To recover the true signal in a noise-contaminated observation, the most commonly used methods are based on the least squares concept which mainly depends on the $L_2$ norm. TV norms in image restoration models are typically $L_1$ or $L_2$ norms of derivatives, which aid in the recovery of edges or boundaries in images. Solution methods involving $L_2$ norms can be derived as closed-form expressions where the result is linear in form. They are easier to compute compared to the $L_1$ estimation which is non-linear and computationally complex [10].

Another popular approach in image denoising is by the use of a wavelet transform with a thresholding technique [13]. A similar technique was earlier applied to reduce noise of speech signals in [14] and was used to denoise ECG signals in [15]. Machine learning techniques such as neural networks have also been applied to signal denoising; see for instance [16] and [17].

It was shown in [5] that some methods for denoising discrete one-dimensional discontinuous data using model (2) with first-order TV term select reconstructions within a tube-like region. These methods are referred to as tube methods. The concept of tube methods were developed based on a constrained minimization problem formulation focused on an antiderivative $F$ of the given data $f$. For all components of $F$, the boundaries $F + \lambda$ and $F - \lambda$ are constructed enclosing the tube region $T$. The minimizer $u$ then has the property that its antiderivative $U$ is contained in $T$.

In [11], a general version of (2) was formulated as a tube problem called the Spline Contact Problem (SCP). In Section 3, it is shown that the minimization problem

$$\frac{1}{2}\|f - u\|_2^2 + \lambda\|D_{n,m}u\|_1,$$

where $D_{n,m}$ is defined as in (3) below, is equivalent to an SCP satisfying boundary, tube, and contact conditions. For $m = 1$, the discrete antiderivative of the unique solution can be obtained using TSA resulting in a piecewise-linear curve contained within a tube of radius $\lambda$, centered at $F$, and interpolating boundary and contact points. The derivative of the taut-string output is a piecewise constant function. This will reflect in the recovery process as staircasing in the reconstruction $u$, possibly flattening peaks in the original signal.

The taut-string solution can be viewed as a linear spline complying with the SCP conditions. Splines are significantly useful in problems involving interpolation and approximation. In [1], cubic spline regression was used to study the relationship between solar radiation and sunlight duration. In [2], spline interpolation techniques were applied in the field of power systems. It was used to compute heat transfer in [3]. Because of its customized property, splines are ideal tools for minimization problems requiring a certain level of smoothness. We use cubic splines to replace the linear spline solution $U$ obtained from the taut-string method to offer a smoother, if not optimal, alternative solution in the one and two dimensional cases. The boundary points and contact points with the tube based on the TSA are maintained to initialize interpolation using cubic splines. This however results to parts of the spline which may escape the tube. In Section 5 we discuss the inclusion of a heuristic step that reinserts the spline inside the tube by adding additional points inside the tube to be reinterpolated.

## 2. DISCRETE SPLINES

We are interested in a piecewise continuous function that is smooth everywhere, even at its points of connectivity. A spline function is such a function, defined as follows [6, 7]:

Definition 2.1. *Let $D = \{(x_i, y_i)\,|1 \le i \le m\}$ be a set of $m$ points in $\mathbb{R}^2$ such that $x_i < x_{i+1}$, for all $i = 1, \ldots, m-1$. A spline*

$$s(x) = \sum_{i=1}^{m-1} p_i(x)\,\chi_{[x_i, x_{i+1}]}(x)$$

*is an $n^{th}$- degree function of class $C^{n-1}$ [8] that passes through the set of knots $D$*

*where* $p_i(x) = \sum_{k=0}^{n} a_{k,i} x^k$ *and* $\chi_E(x) = \begin{cases} 1 & \text{if } x \in E \\ 0 & \text{otherwise} \end{cases}$.

The coefficients $a_{k,i}$ are chosen such that

- $s(x)$ interpolates all points $(x_i, y_i)$, i.e., $s(x_i) = y_i$, for all $i = 1, \ldots, m$ and
- adjacent sub-functions $p_i(x)$ and $p_{i+1}(x)$ and their respective derivatives are continuous at their common point $(x_{i+1}, y_{i+1})$, for all $i = 1, \ldots, m-2$.

### 2.1 Cubic Splines

By definition 2.1, a cubic spline through $D$ is a $C^2$ function $s(x)$ where

$$p_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$$

for each $i = 1, \ldots, m-1$.

Since $s^{(k)}$ is continuous for each $k = 0, 1$, then

$$\lim_{x \to x_i^+} s(x) = p_i(x_i) = a_i x_i^3 + b_i x_i^2 + c_i x_i + d_i = y_i, \text{ and}$$

$$\lim_{x \to x_{i+1}^-} s(x) = p_i(x_{i+1}) = a_i x_{i+1}^3 + b_i x_{i+1}^2 + c_i x_{i+1} + d_i = y_{i+1},$$

for all $i = 1, \ldots, m-1$.

Moreover,

$$\lim_{x \to x_{i+1}^-} s^{(k)}(x) = p_i^{(k)}(x_{i+1}) = p_{i+1}^{(k)}(x_{i+1}) = \lim_{x \to x_{i+1}^+} s^{(k)}(x).$$

For $k = 1, 2$, we have

$$3a_i x_{i+1}^2 + 2b_i x_{i+1} + c_i = 3a_{i+1} x_{i+1}^2 + 2b_{i+1} x_{i+1} + c_{i+1}$$

$$6a_i x_{i+1} + 2b_i = 6a_{i+1} x_{i+1} + 2b_{i+1}$$

for all $i = 1, \ldots, m-1$.

We include the following boundary conditions

$$\lim_{x \to x_1^+} s''(x) = 0 \text{ and } \lim_{x \to x_m^-} s''(x) = 0$$

resulting in a natural cubic spline. Other conditions can be used such as

$$\lim_{x \to x_1^+} s''(x) = \lim_{x \to x_2} s''(x)$$

and

$$\lim_{x \to x_{m-1}} s''(x) = \lim_{x \to x_m^-} s''(x)$$

which produces a parabolic run-out spline forcing the ends of the interval to behave like a parabola.

The above continuity properties are applied on $s(x)$ and are summarized in the matrix form $\mathbf{Qa} = \mathbf{y}$ where $\mathbf{Q} = [\mathbf{A}\ \mathbf{B}]$ with appropriately sized matrices

$$\mathbf{A} = \begin{bmatrix}
x_1^3 & x_1^2 & x_1 & 1 & 0 & \cdots & \cdots & \cdots & \cdots \\
x_2^3 & x_2^2 & x_2 & 1 & 0 & \cdots & \cdots & \cdots & \cdots \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
6x_1 & 2 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & 0 & 0 & x_2^3 & x_2^2 & x_2 & 1 & 0 \\
0 & 0 & 0 & 0 & x_3^3 & x_3^2 & x_3 & 1 & 0 \\
-3x_2^2 & -2x_2 & -1 & 0 & 3x_2^2 & 2x_2 & 1 & 0 & 0 \\
-6x_2 & -2 & 0 & 0 & 6x_2 & 2 & 0 & 0 & 0 \\
0 & \cdots & \cdots & \cdots & \cdots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & & & & & \ddots & \ddots & \ddots \\
\vdots & & \ddots & & & & & \ddots & \ddots \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \ddots
\end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix}
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \\
0 & \cdots & \cdots & \cdots & 6x_m & 2 & 0 & 0 & \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \\
\ddots & & & & & & & \vdots & \\
 & \ddots & & & & & & & \vdots \\
\ddots & \ddots & \ddots & \ddots & & & & 0 & \\
\ddots & 0 & 0 & 0 & x_{m-1}^3 & x_{m-1}^2 & x_{m-1} & 1 & \\
\ddots & 0 & 0 & 0 & x_m^3 & x_m^2 & x_m & 1 & \\
-3x_{m-1}^2 & -2x_{m-1} & -1 & 0 & 3x_{m-1}^2 & 2x_{m-1} & 1 & 0 & \\
-6x_{m-1} & -2 & 0 & 0 & 6x_{m-1} & 2 & 0 & 0 &
\end{bmatrix},$$

$$\mathbf{a} = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ a_2 \\ b_2 \\ c_2 \\ d_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ a_{m-1} \\ b_{m-1} \\ c_{m-1} \\ d_{m-1} \end{bmatrix}, \text{ and } \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ 0 \\ 0 \\ y_2 \\ y_3 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ y_{m-1} \\ y_m \\ 0 \\ 0 \end{bmatrix}.$$

Solving the above system will yield a column vector containing coefficients of a natural cubic spline interpolating the $m$ points in $D$.

Splines are commonly used in interpolation problems for their customized approach of traversing a given set of points. Because of their smooth nature, they are also ideal for problems requiring smooth derivatives.

In the antiderivative setting, we use natural cubic splines as tools for interpolation of data points obtained from TSA. Once the interpolation is finished, we get the derivative of the resulting cubic spline which will return a smooth parabolic curve.

## 3. SPLINE CONTACT PROBLEM

The details of the derivation of the Spline Contact Problem and the Taut-String Algorithm in the following section are taken from [11]. We apply the discretization of the penalty term $\|J(u)\|_1$ using forward differences. The $n \times n$ matrix $D_n$ and its inverse $A_n$, when applied to an $n \times 1$ vector $f$ will represent the dicrete versions of the first order derivative and antiderivative of $f$, respectively:

$$D_n = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ -1 & 1 & \dots & 0 & 0 \\ & \ddots & \ddots & & \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & -1 & 1 \end{pmatrix}, \; A_n = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 1 & 1 & \dots & 0 & 0 \\ & \ddots & \ddots & & \\ 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & \dots & 1 & 1 \end{pmatrix}.$$

The $m$ times application of each of the above matrices yields the $m$-fold differentiation and $m$-fold integration of $f$, denoted by $D_n^m$ and $A_n^m$, respectively.

Let $0_{n,m}$ denote the $n \times m$ matrix composed of zeros, and $I_n$ the $n \times n$ identity matrix. The discretization of $J(u)$ in (2) is defined as the matrix produced by pre-multiplying an $(n-m)$-by-$n$ augmented matrix to $D_n^m$:

$$D_{n,m} := \left(0_{n-m,m} | I_{n-m}\right) D_n^m. \tag{3}$$

The minimizaton problem (2) can now be restated as

$$\frac{1}{2} \|f-u\|_2^2 + \lambda \|D_{n,m} u\|_1 \to \text{ min.} \tag{4}$$

The functional defined above is strictly convex and therefore has a unique minimizer $u$ which is $n \times 1$.

We list in the proposition below the relations between $D_n^m$ and $D_{n,m}$ which are crucial to the formulation of the SCP.

PROPOSITION 3.1. *The difference matrices satisfy the following properties:*

1. $D_{n,m}^\top = (-1)^m D_n^m \begin{pmatrix} I_{n-m} \\ 0_{m,n-m} \end{pmatrix}$,
2. $D_{n,m} D_n^m = D_{n+m,2m} \begin{pmatrix} 0_{m,n} \\ I_n \end{pmatrix}$,
3. $D_{n+m,m} \begin{pmatrix} 0_{m,n} \\ I_n \end{pmatrix} = D_n^m$.

One of the results in [11] is that the problem of minimizing (2) is equivalent to a spline interpolation problem where the knots that define the spline are not known in advance but depend on the input data $f$ and $\lambda$. For $m = 1$, the resulting SCP is well examined and can be solved by TSA [5].

To begin the formulation of the SCP, we first apply the conditions for optimality. A necessary and sufficient condition for $u$ to be the minimizer of (2) is that the $n \times 1$ zero vector is an element of the functional's subdifferential, i.e.,

$$0_{n,1} \in u - f + \lambda \partial \left(\|D_{n,m} u\|_1\right). \tag{5}$$

Note that the subgradient of $|x|$ is given by

$$\frac{x}{|x|} := \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x < 0 \\ [-1,1] & \text{if } x = 0. \end{cases}$$

By Theorem 23.9 in [9], (5) can be written as

$$u = f - \lambda D_{n,m}^\top \frac{D_{n,m} u}{|D_{n,m} u|} \tag{6}$$

only if $D_{n,m} u \neq 0$ and $\frac{D_{n,m} u}{|D_{n,m} u|}$ is performed component-wise. The result of this component-wise division return values 1 or $-1$. However, if $D_{n,m} u = 0$, $\partial \left(\|D_{n,m} u\|_1\right)$ returns a value between 1 and $-1$.

The present form (6) of the statement is not convenient for the computation of the solution $u$. Multiplying both sides with the discrete $m^{th}$ antiderivative matrix $A_n^m$, we get

$$A_n^m u = A_n^m f - \lambda A_n^m D_{n,m}^\top \frac{D_{n,m} u}{|D_{n,m} u|}.$$

Applying the forward difference matrix property in Proposition 3.1.1 results to

$$A_n^m u = A_n^m f - (-1)^m \lambda \begin{pmatrix} I_{n-m} \\ 0_{m,n-m} \end{pmatrix} \frac{D_{n,m} u}{|D_{n,m} u|}. \tag{7}$$

Let $K$ be an index set. We define $F_K = \left(A_n^m f\right)_K$ to be the subvector of $A_n^m f$ with indices in $K$. We can write $A_n^m f$ and $A_n^m u$ as

$$A_n^m f =: \begin{pmatrix} F_I \\ F_R \end{pmatrix} \text{ and } A_n^m u =: \begin{pmatrix} U_I \\ U_R \end{pmatrix}, \tag{8}$$

where $F_I, U_I \in \mathbb{R}^{n-m}$ and the right boundary condition vectors $F_R, U_R \in \mathbb{R}^m$. This means that $F_I$ and $U_I$ takes in the first $n-m$ components of $A^m f$ and $A^m u$, respectively, and $F_R$ and $U_R$ takes in the remaining $m$ components.

We can now rewrite (7) as

$$U_I = F_I - (-1)^m \lambda \frac{D_{n,m}u}{|D_{n,m}u|}, \text{ and} \tag{9}$$

$$U_R = F_R. \tag{10}$$

In (8), we can obtain $f$ and $u$ by premultiplying the $m^{th}$ order forward-difference matrix $D_n^m$,

$$f = D_n^m \begin{pmatrix} F_I \\ F_R \end{pmatrix} \text{ and } u = D_n^m \begin{pmatrix} U_I \\ U_R \end{pmatrix}. \tag{11}$$

Premultiplying $D_{n,m}$ on (11) and using property (2) in Proposition 3.1.1 of the previous section, we have

$$D_{n,m}u = D_{n+m,2m} \begin{pmatrix} 0_{m,n} \\ I_n \end{pmatrix} \begin{pmatrix} U_I \\ U_R \end{pmatrix}.$$

An artificial left boundary condition $U_L := 0_{m,1}$ is introduced to replace the product $\begin{pmatrix} 0_{m,n} \\ I_n \end{pmatrix} \begin{pmatrix} U_I \\ U_R \end{pmatrix}$ which results to $\begin{pmatrix} 0_{m,1} \\ U_I \\ U_R \end{pmatrix}$.

This extends our vector $U$ to

$$U := \begin{pmatrix} U_L \\ U_I \\ U_R \end{pmatrix}.$$

The inclusions in (10) can now be written as

$$U_I = F_I - (-1)^m \lambda \frac{D_{n+m,2m}U}{|D_{n+m,2m}U|}, \tag{12}$$

$$U_R = F_R. \tag{13}$$

We summarize the above formulations as a problem solvable by spline interpolation with the conditions specified in the SCP below.

**Spline Contact Problem**

1. Boundary conditions: $U_L = F_L = 0_{m,1}$ and $U_R = F_R$
2. Tube condition: $\|F_I - U_I\|_\infty \leq \lambda$
3. Contact Condition:
   - Lower Contact Condition:
     If the $i$th component of $(-1)^m D_{n+m,2m}U$ is positive, then
     $$U(i) = F(i) - \lambda.$$
   - Upper Contact Condition:
     If the $i$th component of $(-1)^m D_{n+m,2m}U$ is negative, then
     $$U(i) = F(i) + \lambda.$$

where $F := \begin{pmatrix} F_L \\ F_I \\ F_R \end{pmatrix}$ and $U := \begin{pmatrix} U_L \\ U_I \\ U_R \end{pmatrix}$.

The boundary conditions initialize the first $m$ components of the vector $U$ with zeroes and the last $m$ components are set with the last $m$ corresponding components of vector $F$. The description of the remaining unknown components of $U$ is summarized in the tube and contact conditions. The tube condition describes that the distance of the $U_I$ components cannot be more than $\lambda$ units away from their partner $F_I$ components. We can contain the possible locations of the components of $U$ as a tube-like region with a maximum radius of $\lambda$. Figure 1 illustrates this region.

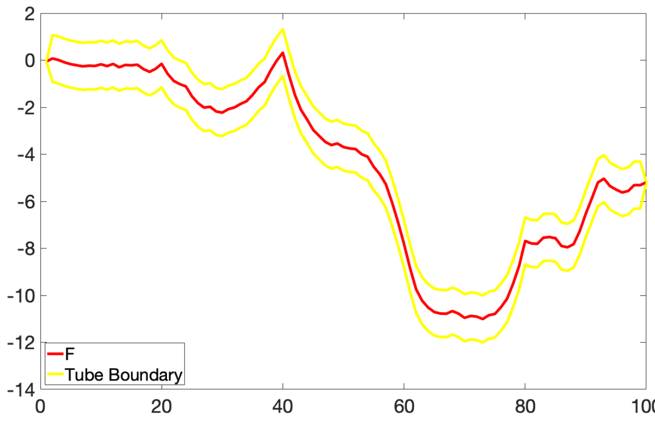


**Figure 1.** Tube boundaries enclosing possible locations or values of $U$

## 4. TAUT-STRING ALGORITHM

When $m = 1$, the simplest case of the SCP defined in (4) is equivalent to solving the Taut-String Algorithm. The solution to this algorithm is a linear spline representing a string of minimal length contained within a tube.

Implementing this algorithm on a signal with $n$ data points is equivalent to solving the following minimization problem with boundary conditions:

$$\min \sum_{j=1}^{n} \sqrt{1 + (U(j+1) - U(j))^2}.$$

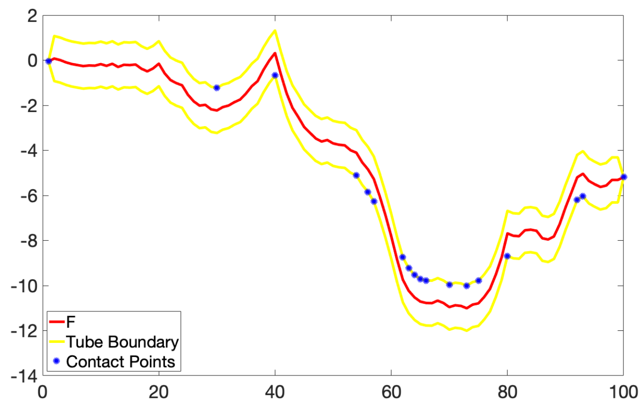
The $n \times 1$ vector $U$ appears to be a string that is taut inside the tube centered on $F$. In the tube problem, we first construct the graph of $F$ to be the "center" of the tube. This is done by premultiplying the discrete antiderivative matrix $A$ to the vector representing the observed signal $f$, i.e.,

$$F = Af.$$

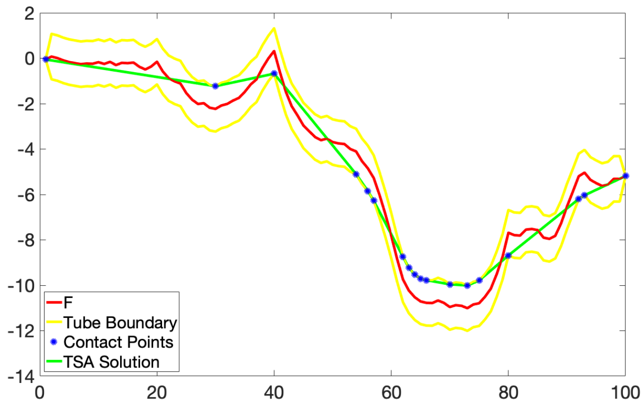
Next the boundaries of the tube are set by measuring $\lambda$ units above and below $F$. We set the boundary conditions of the tube such that the endpoints have the same values as $F$. The endpoints of the upper boundary of the tube are initialized to the endpoint values of the lower boundary vector.

Starting with the leftmost point, the greatest convex minorants of the upper bound and the smallest concave majorants are computed left to right based on the existing point. The point of intersection of these segments are included in the roster of contact points. The last point added to the roster will serve as the starting point of the next set of convex minorants and concave majorants. This process is continued until the last point of the tube is reached [4]. Figure 2(a) displays these contact points.

The set of contact points obtained are connected by linear splines. The resulting taut-string solution $U$ (e.g., see Figure 2(b)) is now premultiplied with the discrete derivative matrix $D$ to obtain the solution $u$ to the minimization problem (4), i.e., $u = DU$.



(a)



(b)

**Figure 2.** (a) Contact points based on the TSA, (b) Solution using TSA

Since the taut-string solution vector $U$ corresponds to a linear spline, its derivative counterpart $u$ represents a piecewise-constant curve. When simulated in a computational software, endpoints with same $x$-values but different $y$-values are connected together using vertical line segments. This phenomenon, called "staircasing effect" appears as a series of staircases repeatedly going up and down along the observed signal.

It is harder to obtain a solution for the SCP when $m > 1$. This is because of the implicit nature of the formulation and the increasing computational requirements for higher values of $m$.

Also, for a specific value of $m$ the solution $u$ is a spline of degree $m - 1$ which shows staircasing effects for $m = 1$.

Figure 3 illustrates the effect of staircasing on a peicewise signal.

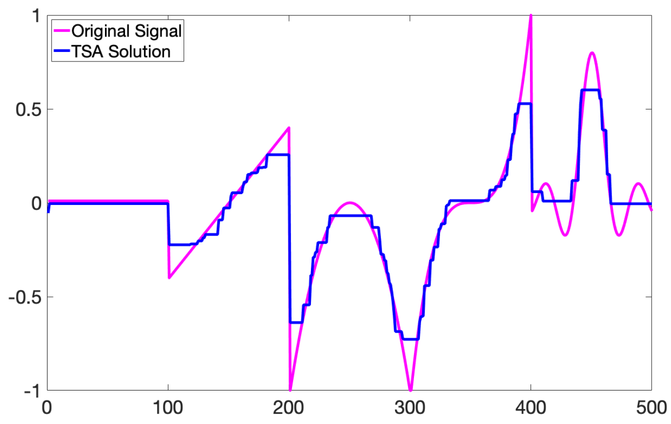


**Figure 3.** Staircasing effect on the solution to a piecewise signal with 500 data points

It is evident in this example that the staircasing greatly affects corners in the original signal. The TSA tends to lose the sharpness of the corners in its reconstruction.

## 5. A HEURISTIC STRATEGY

In this section we apply a similar version of (4), modifying its penalty term by substituting the matrix $D_n$ in place of the forward-difference matrix $D_{n,m}$. We need the antiderivative of the forward difference matrix in the tube formulation of the problem. This substitution simplifies the algorithm for the one-dimensional case by using the invertible square matrix $D_n$ as compared to $D_{n,m}$ which is of size $(n-m)$-by-$n$. Recall that our objective is to minimize the functional

$$\frac{1}{2}\left\|f-u\right\|_2^2 + \lambda\left\|D_n u\right\|_1.$$

Following the necessary and sufficient condition for optimality,

$$0_{n,1} \in u - f + \lambda\partial\left\|D_n u\right\|_1,$$

which implies

$$u \in f - \lambda D_n^\top \frac{D_n u}{|D_n u|}.$$

Integrating in the discrete sense yields

$$A_n^\top u \in A_n^\top f - \lambda A_n^\top D_n^\top \frac{D_n u}{|D_n u|}.$$

Simplifying,

$$A_n^\top u \in A_n^\top f - \lambda \frac{D_n u}{|D_n u|}.$$

Using the splitting applied in (8), we get

$$U \in F - \lambda \frac{D_n u}{|D_n u|}.$$

where $U = A_n^\top u$ and $F = A_n^\top f$

In the previous section, the TSA provides a solution for the problem stated above. In this paper, we replace the linear spline with a cubic spline interpolating the same contact points initially derived from TSA.

We use cubic splines because of its smoothness up to the third degree. We need this property so that when we return to obtaining the reconstruction $u$, by discretely differentiating $U$, we obtain a smooth derivative based on the cubic spline we used.

The method proposed in this paper starts out by implementing TSA in the tube region based on $f$. After which we implement the heuristic step which utilizes cubic splines to obtain the solution. It begins with two inputs. Firstly, the observed data $f$, which is a vector of length $n$. Secondly, the value $\lambda$ which is a nonzero constant that serves as radius of the tube to be derived from $f$. The vector $f$ is assumed to contain a collection of function values at each node from 1 to $n$.

The antiderivative $F$ is computed by premultiplying $A_n$ to $f$. We then obtain the lower and upper boundaries of the tube $F_{low}$ and $F_{upp}$, respectively. The lower boundary is obtained by simply subtracting from $F$ a $\lambda$-vector which is a vector of the same length where are the components are all $\lambda$. Similarly, we obtain the upper boundary by adding the $\lambda$-vector to $F$. These steps are found in Algorithm 1 lines 3 and 4.

We then implement the Taut-String Algorithm to determine a solution. We only note the nodes and function values of the contact points of the resulting solution as $c$ and $fc$ as seen in Algorithm 1 line 5.

The determination of knots or contact points for a cubic spline with the boundaries of the tube is more computationally complex compared to using TSA in determining the contact points for a linear spline solution. Cubic splines have the tendency to go outside of the tube, thus requiring more information than linear splines.

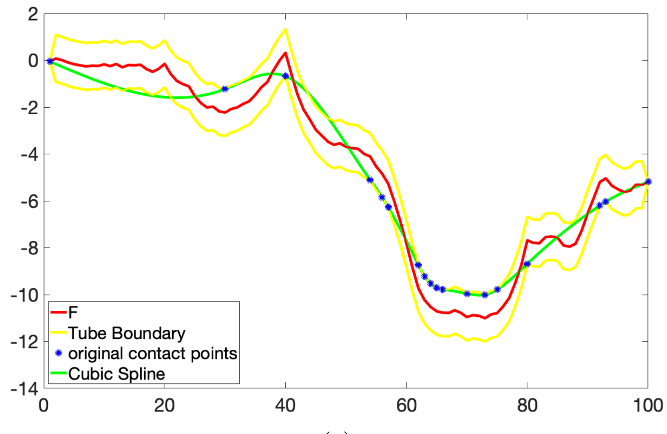
We adapt the same contact points obtained from TSA and use these points as knots for interpolation using cubic spline. See, e.g., Figure 2(a). The function values of the interpolating spline at every node from 1 to $n$ is stored in vector $U$ (this step can be found in Algorithm 1 listed as line 6). The clustering formed in figure (2) on the interval $(60, 80)$ is due to the steep decline before the interval and the following incline after the interval.

Interpolating these points using cubic splines results to parts of the spline which may escape the tube, i.e., points in the spline that may be below the lower boundary of the tube, or above the upper boundary of the tube.

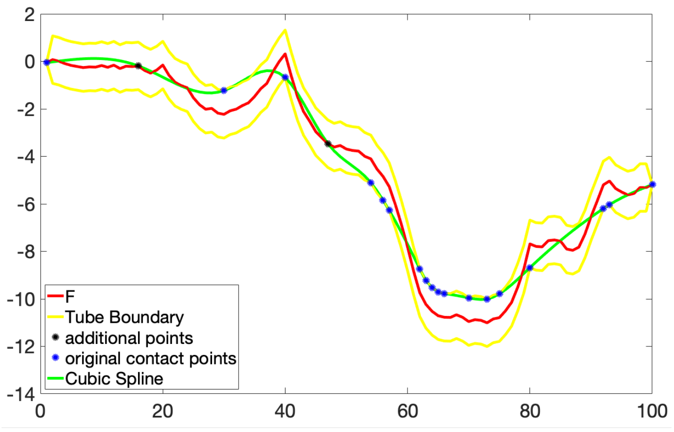
This is expected since cubic splines do not necessarily assign upper boundary contact points as maximum points of the cubic curve. A similar observation applies to the lower boundary of the tube.

Because of possible excursion of the spline outside the tube (Figure 4(a)), we include a heuristic step in the algorithm to keep the spline within the tube. This heuristic process is enveloped in a while loop as indicated in Algorithm 1 from line 7 to 14. Inside this loop, we verify per data point if the cubic spline $U$ has values that are outside the tube region. If that is the case, the "insert" subfunction is prompted. This subfunction adds one or more knots to the list of points to be interpolated by the cubic spline. These points are obtained by first determining pairs of consecutive contact points where the curve is outside the tube. Next, the middle of these pairs of contact points are reassigned values equal to the corresponding value of $F$. The updated set of nodes and function values which are stored as *cnew* and *fcnew*, respectively, are then interpolated by a cubic spline as indicated in line 10.

The process of inserting and reinterpolating is repeated until, for a certain tolerance level (*tol*), all points on the spline are in the tube. In the experiments performed, the tolerance value used is 0.1. Figure 4(b) demonstrates the result of this heuristic scheme. In figure 4(a), the cubic spline exits the tube between the first two contact points and between the third and fourth contact points. The heuristic step assigns the midpoint of each pair of contact points with the value of $F$ at that location. Re-interpolating the contact points with these additional points results in the cubic spline shown in figure 4(b). The resulting cubic spline $U$ is differentiated to obtain the solution $u$.



(a)



(b)

**Figure 4.** (a) Cubic spline interpolation of contact points, (b) Cubic spline interpolation with heuristic step

In summary, we have the following algorithm:

**Algorithm 1** One-dimensional Heuristic Spline Interpolation Algorithm

```
1: function ALGORITHM1($\lambda$, $f$)
2:     n=length(f)
3:     $F = Af$
4:     $F_{low} = F - \lambda * 1_{n,1}, F_{upp} = F + \lambda * 1_{n,1}$
5:     $(c, fc) = TautString(F_{low}, F_{upp})$
6:     $U = cubicspline(c, fc)$
7:     while $1 \neq 0$ do
8:         if $U - F_{upp} > tol$ or $F_{low} - U > tol$ then
9:             $[cnew, fcnew] = insert(c, fc, F_{upp}, F_{low})$
10:            $U = cubicspline(cnew, fcnew)$
11:        else
12:            break
13:        end if
14:    end while
15:    $u = DU$ return $u$
16: end function
```

## 6. NUMERICAL EXAMPLES

In this section, we provide numerical results and images to demonstrate the capabilities of our heuristic method in terms of restoration efficiency.

We used Signal-to-Noise Ratio (SNR) and Root Mean Square (RMS) to evaluate the effectiveness of the reconstruction using the heuristic method. In the following statistical measurements, $g$ is referred to as the column vector representing the observed noisy signal. In the implementation of Algorithm 1, $u$ is the obtained solution [11].

The Signal-to-Noise Ratio, $SNR = 10\log_{10}\left(\frac{\|g\|_2^2}{\|g-u\|_2^2}\right)$, measures the ability of the reconstruction $u$ to recover the clean signal. The Euclidean norm is used to measure the magnitude of both clean signal and removed noise. Obtaining high values for SNR implies good performance in the recovery.

The Root Mean Square, $RMS = \left\|\frac{g-u}{n}\right\|_2$, of a vector is a formula similar to the mean, measuring the central tendency of the difference between the clean signal and the solution $u$. Lower values of RMS correspond to better method performance.

We included in the summary table for each pair of figures, the value $M(u)$ of the discretized ROF functional (2). Another shorter table is included for most of the simulations. This shorter table contains the summary of the same statistical measures however it is applied only on the local extrema values of the signal. The purpose of this additional table is to ascertain which method is better at recovering the peaks or corners of the signal.

The TSA Method and proposed Algorithm 1 was run using MATLAB 2018a in a workstation with 1.8 GHz i5 processor and 8GB RAM.

For the test example, we used four test signals: blocks, bumps, heavisine, and doppler originally introduced in [12].

In the experiments performed, each of the four signals listed above is evaluated at each integer from 1 to $2^n$ where $n$ is taken to be 10 or 13. Figure 5 illustrates the four signals each with $2^{10} = 1024$ data points. The signals are corrupted with either 10% or 20% additive gaussian white noise before implementing the denoising algorithms.

For simplicity, the value for $\lambda$ is set to 0.1. Similar results were observed when $\lambda$ is changed to any other value in the interval $(0.05, 1)$. In general, when the value of $\lambda$ approaches 0, the tube formed in the antiderivative setting becomes smaller and in effect results in solutions which resemble the original noisy image. When $\lambda$ is increased, the initial contact points will only include the boundary points and will give indiscriminate results.

In Figure 6(a), the TSA reconstruction displays a better reconstruction than Algorithm 1 6(b) because both the blocks signal and TSA solutions piecewice constant. Table 1 records the different measures for comparison of the two methods.

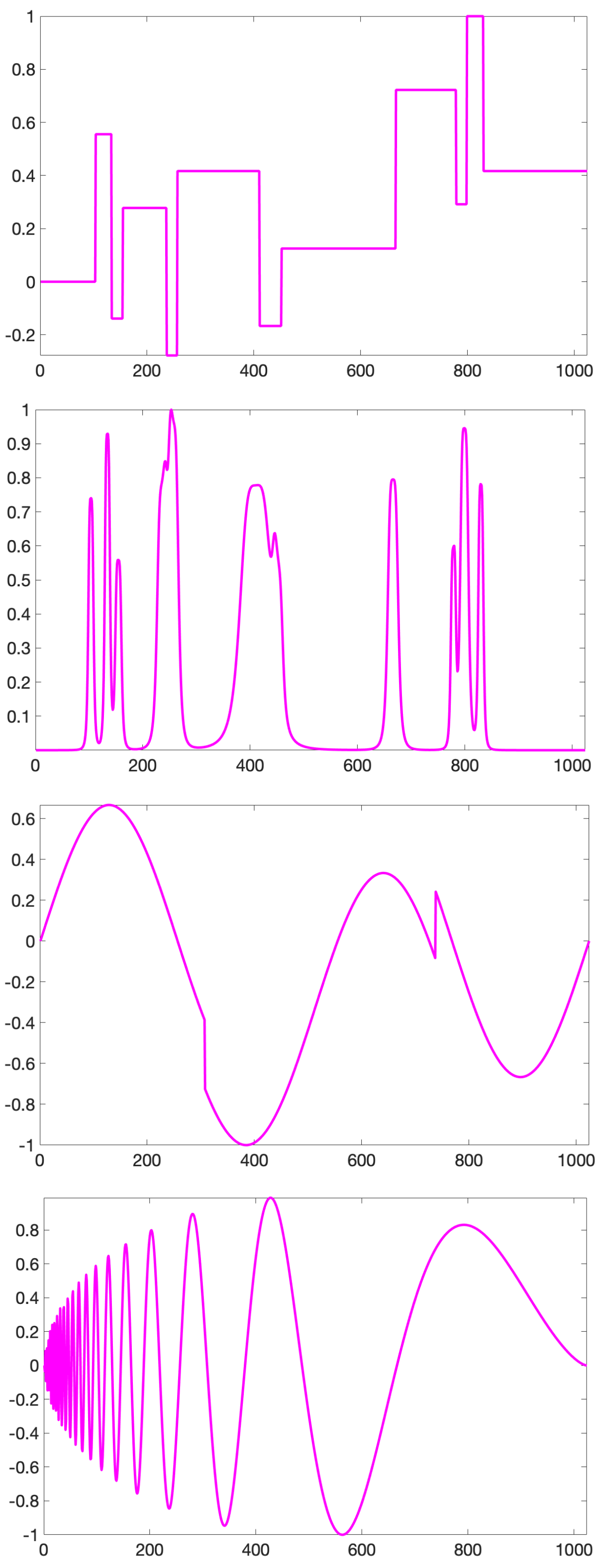
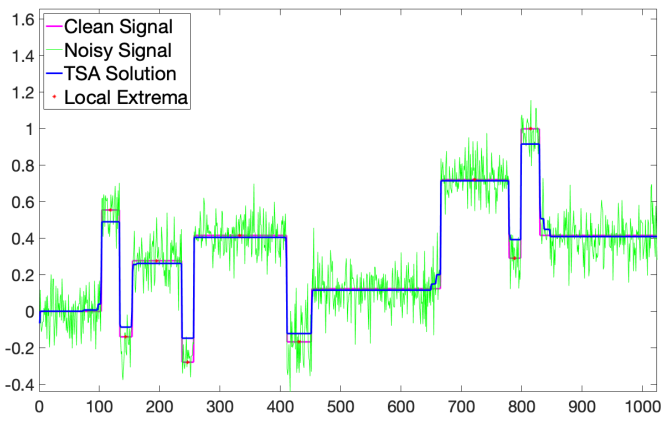
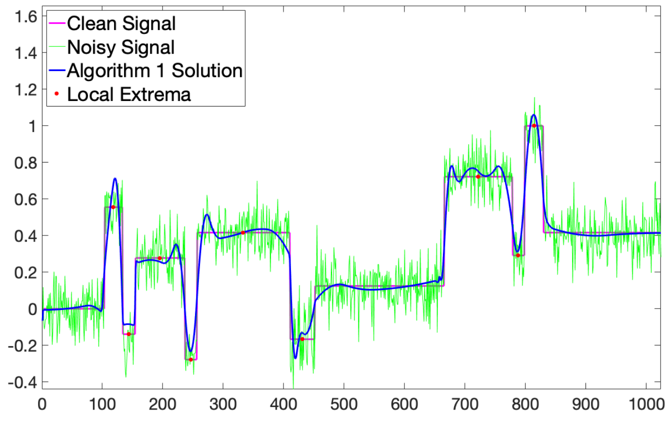


**Figure 5.** Blocks, Bumps, Heavisine, and Doppler signal with 1024 data points

Note that the TSA outperforms the heuristic algorithm when it comes to these measures.



(a)



(b)

**Figure 6.** Reconstructions on blocks signal with 1024 points and 10% noise level: (a) TSA, (b) Algorithm 1

| | M(u) | SNR | RMS | time |
|---|---|---|---|---|
| TSA | 10.74 | 12.23 | 0.1038 | 0.040786 |
| Algo 1 | 15.58 | 10.75 | 0.1232 | 0.057308 |

Table 1: Summary of results for Taut-String Algorithm and Algorithm 1 with 1024 data points and 10% noise level applied on the blocks signal

In Figure 7(a), the TSA reconstruction displays staircasing in its reconstruction and it struggles to recover the original signal at the sharp corners. On the other hand, the reconstruction using Algorithm 1 shows better recovery of sharp corners of the original signal.

Table 2 lists the different measures on the methods. As in the earlier case, the same observation is made, i.e., the values returned by TSA is better by a small margin.

If we obtain the measurements solely at the local extremum to determine the effectivity of each method in recovering corners of the signal we have the following summary:

Table 3 shows favorable results for Algorithm 1 in all mea-

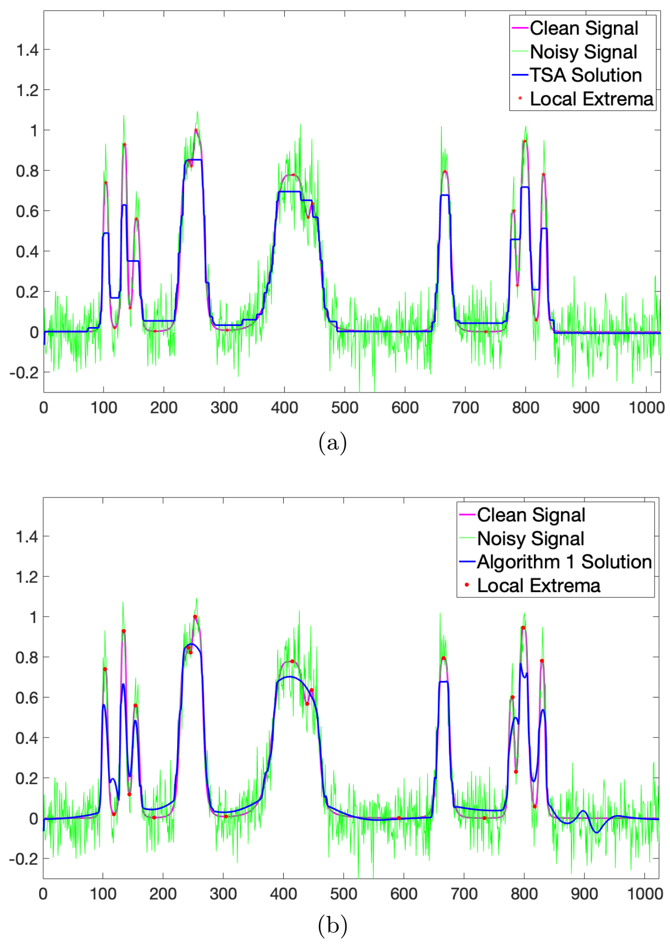


**Figure 7.** Reconstructions on bumps signal with 1024 points and 10% noise level: (a) TSA, (b) Algorithm 1

| | M(u) | SNR | RMS | time |
|---|---|---|---|---|
| TSA | 14.97 | 9.742 | 0.1144 | 0.045359 |
| Algo 1 | 16.96 | 9.624 | 0.116 | 0.072919 |

Table 2: Summary of results for Taut-String Algorithm and Algorithm 1 with 1024 data points and 10% noise level applied on the bumps signal

| | M(u) | SNR | RMS |
|---|---|---|---|
| TSA | 0.3214 | 11.67 | 0.1614 |
| Algo 1 | 0.3072 | 12.73 | 0.1427 |

Table 3: Summary of results for Taut-String Algorithm and Algorithm 1 with 1024 data points and 10% noise level applied on the bumps signal evaluated at local extremum values

sures which indicates that it is better at local extrema recovery than TSA.

Figure 8 shows the TSA and Algorithm 1 solutions for the bumps signal when the number of data points is increased to 8192.

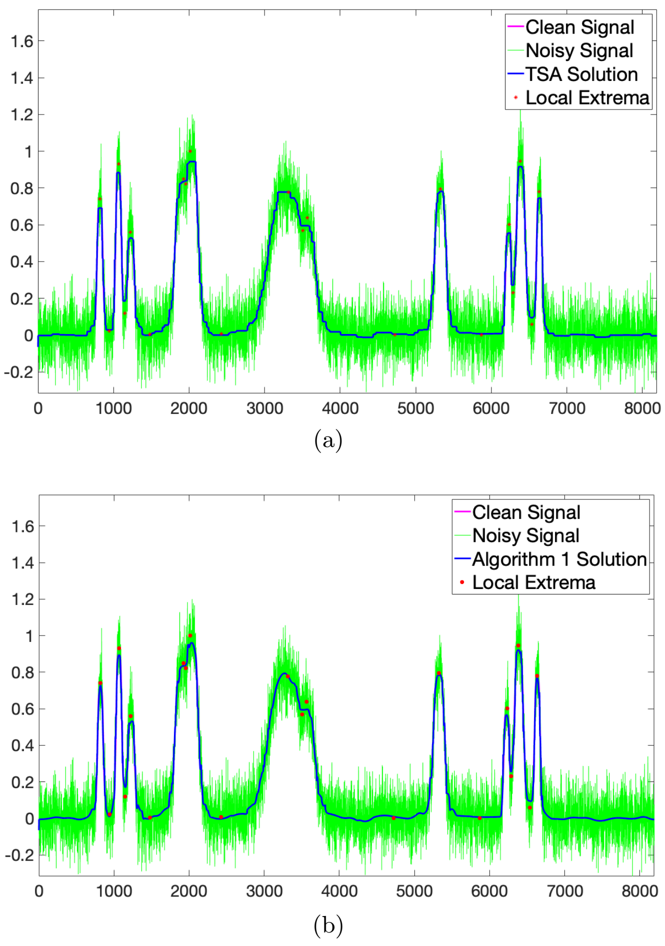


**Figure 8.** Reconstructions on bumps signal with 8192 points and 10% noise level: (a) TSA, (b) Algorithm 1

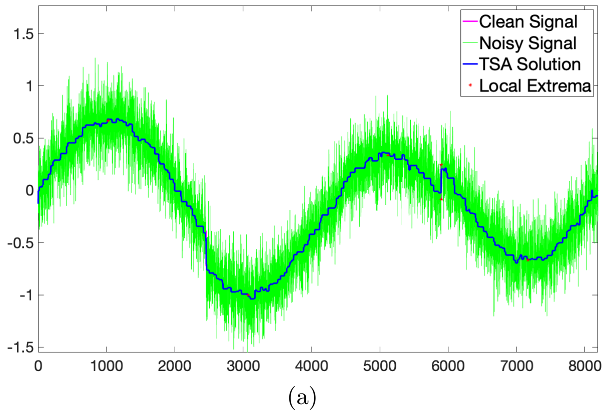
Table 4 shows that Algorithm 1 outperforms TSA in terms of SNR and RMS.

| | M(u) | SNR | RMS |
|---|---|---|---|
| TSA | 0.01204 | 25.23 | 0.03387 |
| Algo 1 | 0.01668 | 27.26 | 0.02681 |

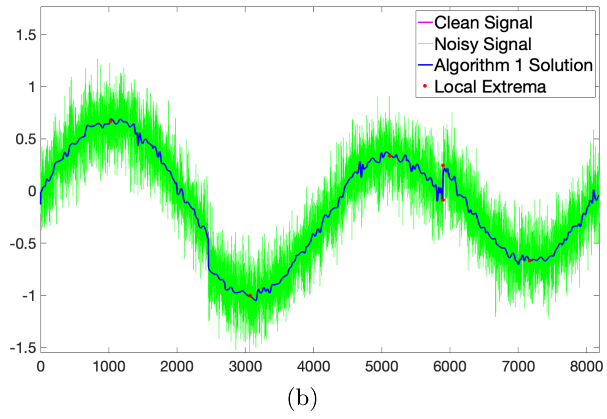
Table 4: Summary of results for Taut-String Algorithm and Algorithm 1 with 8192 data points and 10% noise level applied on the bumps signal evaluated at local extremum values

Figures 9 and 10 show the results for heavisine and doppler signals with 20% noise and 8192 data points.

Tables 5 and 6 shows that Algorithm 1 also outperforms TSA in terms of SNR and RMS for the heavisine and doppler signals with 20 % noise.



(a)



(b)

**Figure 9.** Reconstructions on bumps signal with 8192 points and 20% noise level: (a) TSA, (b) Algorithm 1

| | M(u) | SNR | RMS |
|---|---|---|---|
| TSA | 0.1086 | 15.31 | 0.1007 |
| Algo 1 | 0.1097 | 16.44 | 0.08842 |

Table 5: Summary of results for Taut-String Algorithm and Algorithm 1 with 8192 data points and 20% noise level applied on the heavisine signal evaluated at local extremum values

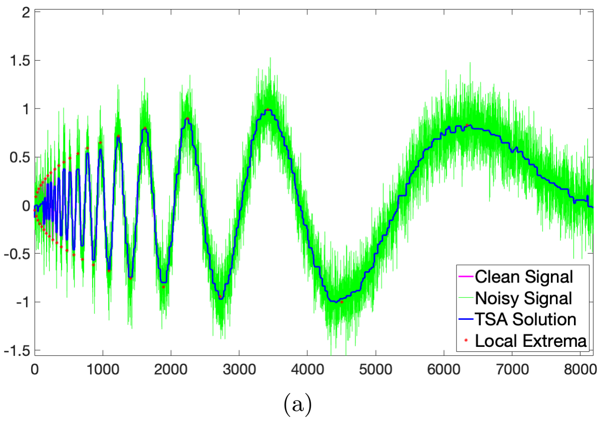
| | M(u) | SNR | RMS |
|---|---|---|---|
| TSA | 0.2821 | 13.11 | 0.1188 |
| Algo 1 | 0.3663 | 13.6 | 0.1122 |

Table 6: Summary of results for Taut-String Algorithm and Algorithm 1 with 8192 data points and 20% noise level applied on the doppler signal evaluated at local extremum values

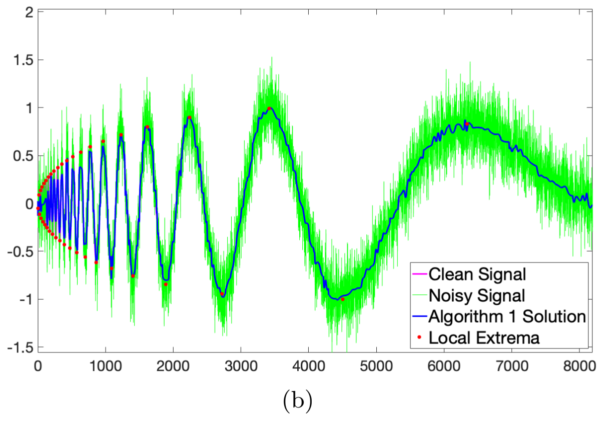
Similar results are observed for the Bumps signal with 20% noise and Heavisine and Doppler signals with 10% noise.

## 7. CONCLUSIONS AND RECOMMENDATIONS

The results obtained for a one-dimensional signal demonstrate the capacity of the heuristic spline interpolation method



(a)



(b)

**Figure 10.** Reconstructions on bumps signal with 8192 points and 20% noise level: (a) TSA, (b) Algorithm 1

in denoising problems. Corners are better restored by the method as compared with the Taut String Algorithm. The drawback with the heuristic are overall measures ($SNR$ and $RMS$) it returns, although it exhibits good results if measurements are restricted at local extrema.

A natural extension of this research is the application of the heuristic method to reconstructing two-dimensional images.

## 8. REFERENCES

[1] R.E. Childs, D. Chuah, S.L. Lee and K.C. Tan. (1984). *Analysis of solar radiation data using splines.* Solar Energy, 32: 643–653.

[2] A. Prasad, A. Manmohan, P. Karthikeyan, and D.P. Kothari. (2018). Application of Cubic Spline Interpolation Technique in Power Systems: A Review. 10.5772/intechopen.74853.

[3] C. Chikwendu, K. Oduwole, S. Okoro, Samuel. (2015). An Application of Spline and Piecewise Interpolation to Heat Transfer (Cubic Case). 5. 28-39.

[4] P.L. Davies and A. Kovac. (2001). *Local extremes, runs, strings and multiresolution.* Annals of Stat, 29:1–65.

[5] W. Hinterberger, M. Hinterḿuller, K. Kunisch, M. von Oehsen and O. Scherzer. (2003). *Tube Methods for BV Regularization.* J Math. Imag. and Vis.,

19:219–235.

[6] T. Lyche. (1976). *Discrete polynomial spline approximation methods. In: Böhmer K., Meinardus G., Schempp W. (eds) Spline Functions. Lecture Notes in Mathematics, vol 501. Springer, Berlin, Heidelberg.*

[7] T. Lyche. (1976). *Discrete cubic spline interpolation. BIT Numerical Mathematics, 16, 281-290.*

[8] F. Warner. (1983). *Foundations of differentiable manifolds and Lie groups (Graduate texts in mathematics).*

[9] R. Rockafellar. (1970). *Convex Analysis.* Princeton University Press.

[10] L. Rudin, S. Osher, and E. Fatemi. (1992). *Nonlinear total variation based noise removal algorithms.* Physica D: Nonlinear Phenomena, 60(1–4): 259–268.

[11] G. Steidl, S. Didas, and J. Neumann. (2006). *Splines in Higher Order TV Regularization.* Intl J Comp Vis, 70: 241-255.

[12] D. Donoho, I. Johnstone, G. Kerkyacharian, and D. Picard. (1995). *Wavelet Shrinkage: Asymptopia?.* Journal of the Royal Statistical Society. Series B, Vol.57, No. 2, 301-369.

[13] Ç. P. Dautov and M. S. Özerdem. (2018). *Wavelet transform and signal denoising using Wavelet method.* 26th Signal Processing and Communications Applications Conference (SIU), Izmir, 2018, pp. 1-4.

[14] R. Aggarwal, J. Singh, V. Gupta, S. Rathore, M. Tiwari, and A. Khare. (2011). *Noise Reduction of Speech Signal using Wavelet Transform with Modified Universal Threshold.* International Journal of Computer Applications. 20. 14-19. 10.5120/2431-3269.

[15] T. Yadav and R. Mehra. (2016). *Denoising and SNR improvement of ECG signals using wavelet based techniques.* 2nd International Conference on Next Generation Computing Technologies (NGCT), Dehradun, pp. 678-682.

[16] W. Zhu, S. M. Mousavi and G. C. Beroza. (2019). *Seismic Signal Denoising and Decomposition Using Deep Neural Networks.* IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 11, pp. 9476-9488.

[17] Y. Chen, M. Akutagawa, Y. Kinouchi, and Q. Zhang. (2008). *Neural network based audio signal denoising.* In: Proceedings of the 2008 International Conference on Advanced Infocomm Technology, ICAIT.