Chapter 3
Weighted Tensor Least Angle Regression for Solving Sparse Weighted Multilinear Least Squares Problems

Ishan M. Wickramasingha, Biniyam K. Mezgebo, and Sherif S. Sherif

Abstract Sparse weighted multilinear least-squares is a generalization of the sparse multilinear least-squares problem, where prior information about, e.g., parameters and data is incorporated by multiplying both sides of the original problem by a typically diagonal weights matrix. However, the introduction of arbitrary diagonal weights would result in a non-Kronecker least-squares problem that could be very large to store or solve practically. In this paper, we generalize our recent Tensor Least Angle Regression (T-LARS) algorithm to efficiently solve either L_0 or L_1 constrained multilinear least-squares problems with arbitrary diagonal weights for all critical values of their regularization parameter. To demonstrate the validity of our new Weighted Least Angle Regression (WT-LARS) algorithm, we used it to successfully solve three different image inpainting problems by obtaining sparse representations of binary-weighted images.

3.1 Introduction

Weighted least squares is a generalization of the least-squares (LS) problem, where prior information about parameters and data is incorporated by multiplying both sides of the original LS problem by a typically diagonal weights matrix. Applications of weighted least-squares in Signal Processing include signal restoration [1, 2], source localization in wireless networks [3–6], adaptive filters [4, 7–9], and image smoothing [10]. In Statistics, weighted least-squares regression is often used to reduce bias from non-informative data samples [11, 12]. Also, a best linear unbiased estimator (BLUE) is obtained by using the inverse of the data covariance matrix as the weights matrix [13].

Recently, sparsity has become a commonly desired characteristic of a least-squares solution [14, 15]. Because of its relatively small number of non-zero values, a sparse

I. M. Wickramasingha · B. K. Mezgebo · S. S. Sherif (✉)
Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Canada
e-mail: Sherif.Sherif@umanitoba.ca

R. Kountchev et al. (eds.), *New Approaches for Multidimensional Signal Processing*,
Smart Innovation, Systems and Technologies 385,
https://doi.org/10.1007/978-981-97-0109-4_3

solution could result in faster processing with lower computer storage requirements [14, 15]. A sparse solution is usually obtained by solving a least-squares problem while minimizing either the L_0 norm of the solution (non-convex optimization problem) or minimizing the L_1 norm of the solution (convex optimization problem), where the L_0 norm of a vector is its number of non-zero elements and the L_1 norm of a vector is the sum of the magnitude of its elements [14].

Several methods have been proposed to solve sparse least-squares problems, including the *Method of Frames* [16], *Matching Pursuit* (MP) [17], *Orthogonal Matching Pursuit* (OMP) [18], *Best Orthogonal Basis* [19], *Least Absolute Shrinkage and Selection Operator* (LASSO) that is also known as *Basis Pursuit* [20, 21], and *Least Angle Regression* (LARS) [21]. Both MP and OMP solve the L_0 constrained least-squares problem [22] using sequential heuristic steps that add solution coefficients in a greedy, i.e., non-globally optimal, way. LASSO relaxes the non-convex L_0 constrained least-squares problem to solve the convex L_1 constrained least-squares problem instead [20]. Among the above solution methods, only *Least Angle Regression* could efficiently solve both the L_0 and, with a slight modification, L_1 constrained least-squares problem for all critical values of their regularization parameters. This parameter is required to balance the minimization of the LS residual with the minimization of the norm of the solution [21].

In addition to incorporating a priori information, weights also could be introduced to sparse least-squares problems to improve the L_1 minimization problem results [23, 24]. Candès et al. also used a reweighted L_1 minimization approach to enhance sparsity in compressed sensing [25]. Also, weighted L_1 constrained least-squares regression has been used to extract information from large data sets for statistical applications [26, 27]. We note that sparse weighted least-squares problems could be solved using any of the above optimization methods.

Multilinear least-squares is a multidimensional generalization of least-squares [28–30], where the least-squares matrix has a Kronecker structure [31, 32]. Sparse multilinear least-squares could be either an L_0 constrained or an L_1 constrained multilinear least-squares problem. Caiafa and Cichocki introduced a generalization of OMP, *Kronecker-OMP*, to solve the L_0 constrained sparse multilinear least-squares problem [32]. Elrewainy and Sherif [33] developed *Kronecker Least Angle Regression* (*K*-LARS) to efficiently solve both L_0 and L_1 constrained sparse least-squares having a specific Kronecker matrix form, $\boldsymbol{A} \otimes \boldsymbol{I}$, for all critical values of the regularization parameter. To overcome this limitation, the authors further developed Tensor Least angle Regression (T-LARS) [30], a generalization of K-LARS that does not require any special form of the LS matrix beyond being Kronecker. T-LARS solves either large L_0 or large L_1 constrained, sparse multilinear least-squares problems (underdetermined or overdetermined) for all critical values of the regularization parameter λ with significantly lower computational complexity and memory usage than Kronecker-OMP.

Weighted multilinear least-squares is a generalization of multilinear least-squares that introduces a typically diagonal weight matrix to both sides of the original LS problem. Since an arbitrary diagonal weight matrix would not be Kronecker, the weighted LS matrix would lose its original Kronecker structure, resulting in a potentially very large non-Kronecker LS matrix. Thus solving these weighted sparse multilinear least-squares problems could become highly impractical, as it would require significant memory and computational power.

Therefore, in this paper, we extend T-LARS to Weighted Tensor Least Angle Regression (WT-LARS) that could solve efficiently both L_0 and L_1 constrained sparse weighted multilinear least-squares problems for all critical values of the regularization parameter. It is organized as follows: Sect. 3.2 includes a brief introduction to the sparse weighted multilinear least-squares problem. In Sect. 3.3, we describe our new Weighted Tensor Least Angle Regression (WT-LARS) algorithm in detail. Section 3.4 provides results of applying WT-LARS to solve three different image inpainting problems by obtaining sparse representations of binary-weighted images. We present our conclusions in Sect. 3.5.

3.2 Problem Formulation

3.2.1 Sparse Weighted Multilinear Least-Squares Problem

A multilinear transformation of a tensor $\mathcal{X}$ could be defined as, $\mathcal{Y} = \mathcal{X} \times_1 \boldsymbol{\Phi}^{(1)} \times_2 \cdots \times_N \boldsymbol{\Phi}^{(N)}$, where $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_n \times \cdots \times J_N}$ and $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_n \times \cdots \times I_N}$ are N^{th} order tensors, with the equivalent vectorized form

$$\Phi vec(\mathcal{X}) = vec(\mathcal{Y}) \tag{3.1}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{J \times I}$, and $\boldsymbol{\Phi} = \boldsymbol{\Phi}^{(N)} \otimes \cdots \otimes \boldsymbol{\Phi}^{(1)}$, and $\otimes$ is the Kronecker product operator [34].

Let $\boldsymbol{W} = \boldsymbol{S}^H \boldsymbol{S}$, be a diagonal weight matrix. We could obtain a weighted linear transformation [35] of (1) as

$$\mathbf{S}\Phi vec(\mathcal{X}) = Svec(\mathcal{Y}) \tag{3.2}$$

A sparse solution of the weighted linear system in (2) could be obtained by solving an L_p ($p = 0 \; or \; p = 1$) minimization problem,

$$\tilde{\mathcal{X}} = \underset{\mathcal{X}}{\operatorname{argmin}} \| \boldsymbol{S}\boldsymbol{\Phi}\text{vec}(\mathcal{X}) - \boldsymbol{S}\text{vec}(\mathcal{Y}) \|_2^2 + \lambda \| \text{vec}(\mathcal{X}) \|_p \tag{3.3}$$

where λ is a regularization parameter.

If $\boldsymbol{S}$ is a Kronecker matrix, then $\boldsymbol{S\Phi} = \left(\boldsymbol{S}^{(N)}\boldsymbol{\Phi}^{(N)} \otimes \cdots \otimes \boldsymbol{S}^{(1)}\boldsymbol{\Phi}^{(1)}\right)$ and we could use T-LARS [30] to obtain a sparse solution for either L_0 or L_1 optimization problem in (3) efficiently. However, $\boldsymbol{S}$ is not typically Kronecker, so $\boldsymbol{S\Phi}$ would not have a Kronecker structure, and (3) should be solved as a potentially very large vectorized (one-dimensional) sparse least-squares problem which could be very challenging in terms of memory and computational power requirements. Therefore, in this paper, we develop Weighted Tensor Least Angle Regression (WT-LARS), a computationally efficient method, to solve either L_0 or L_1 constrained sparse weighted multilinear least-squares problems in (3) for an arbitrary diagonal weights matrix $\boldsymbol{W} = \boldsymbol{S}^H \boldsymbol{S} \in \mathbb{R}^{J \times J}$.

3.3 Weighted Tensor Least Angle Regression

In this section, we develop Weighted Tensor Least Angle Regression (WT-LARS) by extending T-LARS to solve the sparse weighted multilinear least-squares problem in (3), for weights $\boldsymbol{W} = \boldsymbol{S}^H \boldsymbol{S}$ and Kronecker dictionaries $\boldsymbol{\Phi}$.

Inputs to WT-LARS are the data tensor $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_n \times \cdots \times J_N}$, *mode-n* dictionary matrices $\boldsymbol{\Phi}^{(n)}; n \in \{1, \cdots, N\}$ where $\boldsymbol{\Phi} = \boldsymbol{\Phi}^{(N)} \otimes \cdots \otimes \boldsymbol{\Phi}^{(1)}$, the diagonal weight matrix $\boldsymbol{W} = \boldsymbol{S}^H \boldsymbol{S}$, and the stopping criterion as a residual tolerance ε or the maximum number of non-zero coefficients K (*K-sparse* representation). The output is the solution tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_n \times \cdots \times I_N}$.

WT-LARS requires weighted data $\boldsymbol{S}\text{vec}(\mathcal{Y})$, and columns of the weighted dictionary $\boldsymbol{S\Phi}$ to have a unit L_2 norm. Normalized weighted data could be easily calculated by $\mathcal{Y}_W = \boldsymbol{S}\text{vec}(\mathcal{Y}) / \|\boldsymbol{S}\text{vec}(\mathcal{Y})\|_2$. However, the dictionary matrix $\boldsymbol{S\Phi}$ does not have a Kronecker structure. Hence, normalizing *mode-n* dictionary matrices $\boldsymbol{\Phi}^{(n)}$ does not ensure normalization of the columns of $\boldsymbol{S\Phi}$. Therefore, in WT-LARS, we use the normalized weighted dictionary matrix $\boldsymbol{\Phi}_W = \boldsymbol{S\Phi Q}$ instead of the normalized dictionary matrix $\boldsymbol{\Phi}$ in T-LARS, where $\boldsymbol{Q}$ is a diagonal matrix,

$$\boldsymbol{Q}_{i,i} = \frac{1}{\|(\boldsymbol{S\Phi})_i\|_2} \tag{3.4}$$

where $(\boldsymbol{S\Phi})_i$ is the i^{th} column of the weighted dictionary matrix $\boldsymbol{S\Phi}$. We can efficiently calculate the diagonal matrix $\boldsymbol{Q}$ as,

$$diag(\boldsymbol{Q}) = 1./\sqrt{\left(\boldsymbol{\Phi}^{*2}\right)^{\boldsymbol{T}} diag(\boldsymbol{W})} \tag{3.5}$$

where, $\boldsymbol{\Phi}^{*2}$ [36] denotes the Hadamard square of $\boldsymbol{\Phi}$, such that $\boldsymbol{\Phi}^{*2}_{i,j} = \left(\boldsymbol{\Phi}_{i,j}\right)^2$, "./" denotes elementwise division, and $diag(\boldsymbol{Q})$ and $diag(\boldsymbol{W})$ are diagonal vectors of $\boldsymbol{Q}$ and $\boldsymbol{W}$ respectively. We could efficiently calculate $\left(\boldsymbol{\Phi}^{*2}\right)^{\boldsymbol{T}} diag(\boldsymbol{W})$ using the full multilinear product.

WT-LARS solves the L_0 or L_1 constrained minimization problems in (3) for all critical values of the regularization parameter λ. WT-LARS starts with a large value of λ, that results in an empty active set $I = \{\}$, and a solution $\tilde{\mathcal{X}}_{t=0} = 0$. The set I denotes an active set of columns of the dictionary $\boldsymbol{\Phi}_W$, i.e., column indices where the optimal solution $\tilde{\mathcal{X}}_t$ at iteration t, is nonzero, and I^c denotes its corresponding inactive set. Therefore, $\boldsymbol{\Phi}_{W I}$ contains only the active columns of the dictionary $\boldsymbol{\Phi}_W$ and $\boldsymbol{\Phi}_{W I^c}$ contains only its inactive columns.

At each iteration t, a new column is either added (L_0) to the active set I or a new column is either added or removed (L_1) from the active set I, and λ is reduced by a calculated value δ_t^*.

As a result of such iterations, new solutions with an increased number of coefficients that follow a piecewise linear path are obtained until a predetermined residual error ε or a predetermined number of active columns K is obtained.

The regularization parameter λ is initialized to the maximum of the correlation $\boldsymbol{c}_1$, between the columns of $\boldsymbol{\Phi}_W$ and the initial residual $\boldsymbol{r}_0 = \text{vec}(\mathcal{Y})$.

$$\boldsymbol{c}_1 = \boldsymbol{\Phi}_W^T \boldsymbol{r}_0 \tag{3.6}$$

Since $\boldsymbol{\Phi}_W^T = \boldsymbol{Q}\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{S}$, we can easily calculate $\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{S}\boldsymbol{r}_0$ using the full multilinear product as

$$\mathcal{C}_1' = \mathcal{R}_{S_0} \times_1 \boldsymbol{\Phi}^{(1)^T} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)^T} \tag{3.7}$$

where $\text{vec}\big(\mathcal{R}_{S_0}\big) = \boldsymbol{S}\boldsymbol{r}_0$ and $\boldsymbol{c}_1 = \boldsymbol{Q}\text{vec}\big(\mathcal{C}_1'\big)$. The column index corresponding to the maximum correlation $\boldsymbol{c}_1$ is added to the active set. For a given active set I, the optimal solution $\tilde{\mathcal{X}}_t$ at any iteration t, could be written as

$$\text{vec}\left(\underset{t}{\tilde{\mathcal{X}}}\right) = \begin{cases} \left(\Phi_{W_{I_t}^T}\Phi_{W_{I_t}}\right)^{-1}\left(\Phi_{W_{I_t}^T}\text{vec}(\mathcal{Y}) - \lambda_t \mathbf{z}_t\right), \text{on } I \\ 0, \qquad\qquad \text{Otherwise} \end{cases} \tag{3.8}$$

where, z_t is the sign sequence of $\boldsymbol{c}_t$ on the active set I, and $\boldsymbol{c}_t = \boldsymbol{\Phi}_W^T \boldsymbol{r}_{t-1}$ is the correlation vector of all columns of the dictionary $\boldsymbol{\Phi}_W$ with the residual $\boldsymbol{r}_{t-1}$ at any iteration t.

The optimal solution at any iteration, t must satisfy the following two optimality conditions,

$$\boldsymbol{\Phi}_{W_{I_t}}^{\ T}\boldsymbol{r}_t = -\lambda_t z_t \tag{3.9}$$

$$\|\boldsymbol{\Phi}_{W_{I_t^c}}^{\ T}\boldsymbol{r}_t\|_\infty \le \lambda_t \tag{3.10}$$

where, $\boldsymbol{r}_t = \text{vec}(\mathcal{Y}) - \boldsymbol{\Phi}_W\text{vec}\left(\tilde{\mathcal{X}}_t\right)$ is the residual at iteration t, and z_t is the sign sequence of the correlation $\boldsymbol{c}_t$ at iteration t, on the active set I. The condition in (9) ensures that the magnitude of the correlation between all active columns of $\boldsymbol{\Phi}_W$ and the residual is equal to $|\lambda_t|$ at each iteration, and the condition in (10) ensures that the magnitude of the correlation between the inactive columns of $\boldsymbol{\Phi}_W$ and the residual is less than or equal to $|\lambda_t|$.

At each iteration t, λ_t is reduced by a small step size, δ_t^*, until a condition in either (9) or (10) violates. For L_0, and L_1 constrained minimization problems, if an inactive column violates the condition (10), it is added to the active set, and for L_1 constrained minimization problems, if an active column violates the condition (9), it is removed from the active set.

As λ is reduced by δ_t^*, the solution $\tilde{\mathcal{X}}_t$ changes by $\delta_t^* \boldsymbol{d}_t$ along a direction $\boldsymbol{d}_t$, where $\boldsymbol{d}_{I_t^c} = 0$ and $\boldsymbol{d}_{I_t} = \boldsymbol{G}_t^{-1} \boldsymbol{z}_t$, and $\boldsymbol{G}_t^{-1}$ is the inverse of the Gram matrix of the active columns of the dictionary $\boldsymbol{G}_t = \boldsymbol{\Phi}_{W}{}_{I_t}^{T} \boldsymbol{\Phi}_{W I_t}$.

The size of this Gram matrix would either increase (dictionary column addition) or decrease (dictionary column removal) with each iteration t. Therefore, for computational efficiency, we use the *Schur complement* inversion formula to calculate $\boldsymbol{G}_t^{-1}$ from $\boldsymbol{G}_{t-1}^{-1}$ thereby avoiding its full calculation [30, 37].

The smallest step size for L_1 constrained sparse least-squares problem $\delta_t^* = \min\{\delta_t^+, \delta_t^-\}$ is the minimum of δ_t^+, minimum step size for adding a column, and δ_t^-, minimum step size for removing a column. The minimum step size for removing a column from the active set is given by,

$$\delta_t^- = \min_{i \in I} \left\{ -\frac{\boldsymbol{x}_{t-1}(i)}{\boldsymbol{d}_t(i)} \right\} \tag{3.11}$$

The minimum step size for adding a new column to the active set is given by,

$$\delta_t^+ = \min_{i \in I^c} \left\{ \frac{\lambda_t - \boldsymbol{c}_t(i)}{1 - \boldsymbol{v}_t(i)}, \frac{\lambda_t + \boldsymbol{c}_t(i)}{1 + \boldsymbol{v}_t(i)} \right\} \tag{3.12}$$

where

$$\boldsymbol{v}_t = \boldsymbol{\Phi}_W^T \boldsymbol{\Phi}_W \boldsymbol{d}_t \tag{3.13}$$

Since $\boldsymbol{\Phi}_W = \boldsymbol{S\Phi Q}$**,** We can efficiently calculate $\boldsymbol{v}_t$ using two full multilinear products.

Let $\boldsymbol{v}_t = \boldsymbol{Q}\text{vec}(\mathcal{V}_{t'})$, where

$$\mathcal{V}_t' = \mathcal{U}_{w_t} \times_1 \boldsymbol{\Phi}^{(1)^T} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)^T} \tag{3.14}$$

and $\text{vec}(\mathcal{U}_{w_t}) = \boldsymbol{W}\text{vec}(\mathcal{D}_{t'} \times_1 \boldsymbol{\Phi}^{(1)} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)})$, and $\text{vec}(\mathcal{D}_{t'}) = \boldsymbol{Q}\boldsymbol{d}_t$.

The residual $\boldsymbol{r}_{t+1}$ is calculated at the end of each iteration using,

$$\boldsymbol{r}_{t+1} = \boldsymbol{r}_t - \delta_t^* \boldsymbol{\Phi}_W \boldsymbol{d}_t \tag{3.15}$$

where we can efficiently calculate $\boldsymbol{\Phi}_W \boldsymbol{d}_t$ using

$$\boldsymbol{\Phi}_W \boldsymbol{d}_t = \boldsymbol{S}vec(\mathcal{D}_{t'} \times_1 \boldsymbol{\Phi}^{(1)} \times_2 \cdots \times_N \boldsymbol{\Phi}^{(N)}) \tag{3.14}$$

WT-LARS stops at a predetermined residual error $\boldsymbol{r}_{t+1} \leq \varepsilon$ or when a predetermined number of active columns K is obtained.

3.3.1 Weighted Tensor Least Angle Regression Algorithm

Algorithm 1: Weighted Tensor Least Angle Regression (WT-LARS)

Input: *WT-LARS_mode* $= L_1$ *or* L_0, *normalized tensor* $\mathcal{Y} \in \mathbb{R}^{J_1\times\ldots\times J_n\times\ldots\times J_N}$; *Mode-n dictionary matrices* $\boldsymbol{\Phi}^{(n)} \in \mathbb{R}^{J_n\times I_n}$; $n \in \{1,..N\}$;

Diagonal Weights Matrix $\boldsymbol{W} \in \mathbb{R}^{(J_1\times\ldots\times J_N)\times(J_1\times\ldots\times J_N)}$; *Stopping criterion: residual tolerance:* ε *or number of non-zero coefficients: K*

```
Initialization: $\boldsymbol{S} = \sqrt{\boldsymbol{W}}$, Residual: $\boldsymbol{r}_0 = \boldsymbol{S}vec(\mathcal{Y})$; $\boldsymbol{x}_0 = 0$; $I = \{\}$;
1.  $diag(\boldsymbol{Q}) = \mathbf{1}./\sqrt{(\boldsymbol{\Phi}^2)^T\, diag(\boldsymbol{W})}$
2.  $\mathrm{vec}(\mathcal{R}_{\boldsymbol{S}_0}) = \boldsymbol{S}\boldsymbol{r}_0$
3.  $\mathcal{C}_1 = \mathcal{R}_{\boldsymbol{S}_0} \times_1 \boldsymbol{\Phi}^{(1)^T} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)^T}$
4.  $\boldsymbol{c}_1 = \boldsymbol{Q}\mathrm{vec}(\mathcal{C}_1)$
5.  $[\lambda_1, column\_idx\,] = max(\boldsymbol{c}_1)$
6.  $I = \{column\_idx\}$
7.  while $(\|\boldsymbol{r}_t\|_2 < \varepsilon \text{ or } length(I) < K)$:
8.      $\boldsymbol{z}_t$ = sign ($\boldsymbol{c}_t(I)$)
9.      $\boldsymbol{G}_t^{-1}$ =updateWeightedInverseGramMatrix( $\boldsymbol{G}_{t-1}^{-1}$, $\boldsymbol{W}$ , $\boldsymbol{Q}$ , { $\boldsymbol{\Phi}^{(1)}$, …, $\boldsymbol{\Phi}^{(N)}$}, I,
        add_column, column_idx) % See reference [30]
10.     $\boldsymbol{d}_{I_t} = \boldsymbol{G}_t^{-1}\boldsymbol{z}_t$
11.     $\mathrm{vec}(\acute{\mathcal{D}}_t) = \boldsymbol{Q}\boldsymbol{d}_t$
12.     $\mathcal{U}_t = \acute{\mathcal{D}}_t \times_1 \boldsymbol{\Phi}^{(1)} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)}$
13.     $\mathrm{vec}(\mathcal{U}_{w_t}) = \boldsymbol{W}\mathrm{vec}(\mathcal{U}_t)$
14.     $\mathcal{V}_t = \mathcal{U}_{w_t} \times_1 \boldsymbol{\Phi}^{(1)^T} \times_2 \ldots \times_N \boldsymbol{\Phi}^{(N)^T}$
15.     $\boldsymbol{v}_t = \boldsymbol{Q}vec(\mathcal{V}_t)$
16.     $\delta_{t\,1}^{+} = (\lambda_t - \boldsymbol{c}_t(I^c))./(1 - \boldsymbol{v}_t(I^c))$ % "./" - Elementwise division
17.     $\delta_{t\,2}^{+} = (\lambda_t + \boldsymbol{c}_t(I^c))./(1 + \boldsymbol{v}_t(I^c))$
18.     $\delta_t^- = -\boldsymbol{x}_{t-1}./\,\boldsymbol{d}_t$
19.     $[\delta_t^*,\ column\_idx] = min\left(\delta_{t\,1}^{+}, \delta_{t\,2}^{+}\right)$
20.     add_column == True
21.     if WT-LARS_mode == $L_1$ && min ($\delta_t^-$) < $\delta_t^*$:
22.         $[\delta_t^*,\ column\_idx]$= min ($\delta_t^-$)
23.         add_column = False
24.     end
25.     $\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \delta_t^*\boldsymbol{d}_t$
26.     $\lambda_{t+1} = \lambda_t - \delta_t^*$
27.     $\boldsymbol{c}_{t+1} = \boldsymbol{c}_t - \delta_t^*\boldsymbol{v}_t$
28.     $\acute{\mathcal{R}}_t = \acute{\mathcal{D}}_t \times_1 \boldsymbol{\Phi}^{(1)} \times_2 \cdots \times_N \boldsymbol{\Phi}^{(N)}$
29.     $\boldsymbol{r}_{t+1} = \boldsymbol{r}_t - \delta_t^*\boldsymbol{S}\mathrm{vec}(\acute{\mathcal{R}}_t)$
30.     if add_column == True:   $I = I + \{column\_idx\}$
31.     else: $I = I - \{column_{idx}\}$ end
32. end while
33. return I, $\boldsymbol{x}$
```

3.4 Experimental Results

In this section, we present experimental results for WT-LARS as a tensor completion problem [38–40], using inpainting as an example. Image inpainting has progressed significantly during last few years, specifically using machine learning methods [41–43]. However, as far as we know, no other tensor-based method is available for solving the image inpainting problem as a weighted tensor least squares problems.

For experiments shown in Figs. 3.2 and 3.1, we obtained fenced images from the Image datasets for MSBP deformable lattice detection Algorithm [44], and for the experiment shown in Fig. 3.1, we obtained a landscape image from the DIV2K dataset [45].

Our experimental results were obtained using a MATLAB implementation of WT-LARS using the MATLAB version R2017b on an MS-Windows machine: 2 Intel Xeon CPUs E5-2637 v4, 3.5 GHz, 32 GB RAM, and NVIDIA Tesla P100 GPU with 12 GB memory.

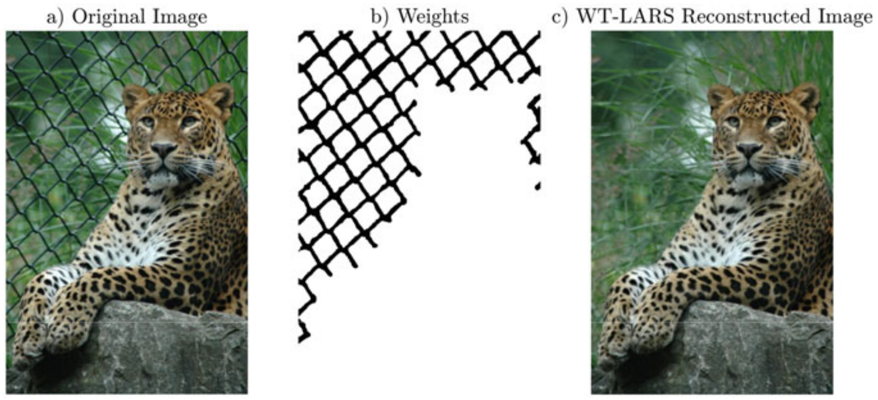

Fig. 3.1 **a** Original image with a fence **b** weights image with zero weights for the fence **c** WT-LARS reconstructed image (Fence Removed)

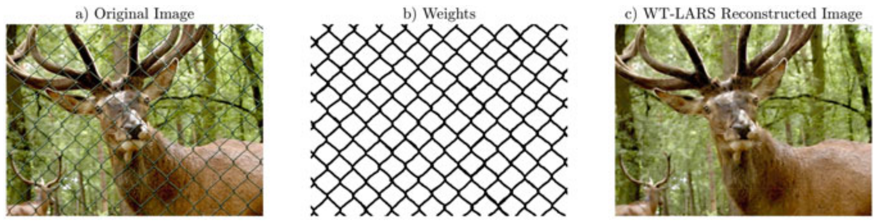

Fig. 3.2 **a** Original image with a fence **b** weights image with zero weights for the fence **c** WT-LARS reconstructed image (Fence Removed)

3.4.1 Inpainting

In this experiment, we use WT-LARS for inpainting. We obtained a sparse representation of the inpainted image using WT-LARS after applying zero weights to the missing data.

In our experimental results shown in Figs. 3.1 and 3.2, we obtained a fenceless image by considering pixels behind the fences as missing data. Figures 3.1a and 3.2a show the original image with a fence, and Figs. 3.1b and 3.2b show the respective masks applied to each pixel of the original image, where black indicates zero and white indicates one. Figures 3.1c and 3.2c show the obtained sparse representation of images behind fences using WT-LARS.

We obtained RGB image patches, $200 \times 200 \times 3$ pixels, from the original images in Figs. 3.1a and 3.2a. For each patch, we obtained a weighted K-sparse representation using WT-LARS, with 10% nonzero coefficients, for three fixed *mode-n* overcomplete dictionaries, $\mathbf{\Phi}^{(1)} \in \mathbb{R}^{200\times400}$, $\mathbf{\Phi}^{(2)} \in \mathbb{R}^{200\times400}$ and $\mathbf{\Phi}^{(3)} \in \mathbb{R}^{3\times4}$, by solving a L_1 constrained sparse weighted least squares problem. Weights consists of zeros for the pixels that belong to the fence in the original images and ones for everywhere else. Used fixed *mode-n* overcomplete dictionaries were a union of a Discrete Cosine Transform (DCT) dictionary and a Symlet wavelet packet with four vanishing moments dictionary. In the experimental results shown in Figs. 3.1 and 3.2, the RGB patches with the minimum number of nonzero samples had 79, 834 and 92, 748 nonzero samples, respectively. We collected 60 image patches from the image in Fig. 3.1a and 35 image patches from the image in Fig. 3.2a, where on average WT-LARS took 476 s to collect 12,000 (10% of $200 \times 200 \times 3$) non-zero coefficients from each image patch.

In the experimental results shown in Fig. 3.3, we use WT-LARS to obtain a landscape image occluded by a person in Fig. 3.3a. Figure 3.3b shows the weights, and Fig. 3.3c shows the inpainting result after removing the person from the foreground of the landscape image.

The RGB images in Fig. 3.3a is a scaled version of the original image with $200 \times 300 \times 3$ pixels. We obtained a weighted K-sparse representation for the scaled image in Fig. 3.3a using WT-LARS, with 20% non-zero coefficients, for three fixed *mode-n* overcomplete dictionaries, $\mathbf{\Phi}^{(1)} \in \mathbb{R}^{200\times400}$, $\mathbf{\Phi}^{(2)} \in \mathbb{R}^{300\times604}$ and $\mathbf{\Phi}^{(3)} \in$

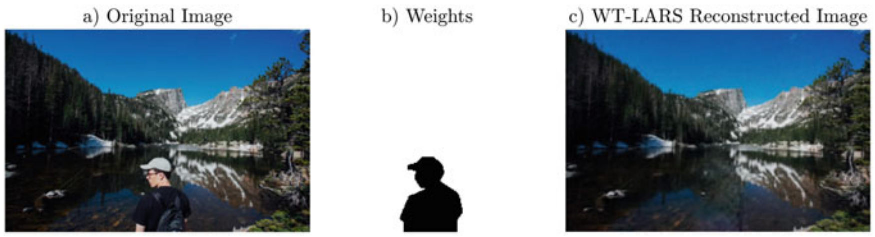

Fig. 3.3 **a** Original image with a person **b** weights image with zero weights for the person **c** WT-LARS reconstructed image (Person Removed)

$\mathbb{R}^{3\times4}$, by solving a weighted L_1 constrained sparse least squares problem. Weights consist of zeros for the pixels belonging to the person in the original image and ones for everywhere else. Used fixed *mode-n* overcomplete dictionaries were a union of a Discrete Cosine Transform (DCT) dictionary and a Symlet wavelet packet with four vanishing moments dictionary. In the experimental results shown in Fig. 3.3, a total of 170,829 nonzero samples have been used to obtain the sparse signal representation of the landscape image. The WT-LARS took 20,625 s to collect 36,000 non-zero coefficients, which is 20% of the size of the image tensor in Fig. 3.3a. Therefore, inpainting results in, Figs. 3.1c, 3.2c and 3.3c clearly show that WT-LARS can be successfully used to approximate missing/incomplete data.

3.5 Conclusions

Sparse weighted multilinear least-squares is a generalization of the sparse multilinear least-squares problem, where both sides of the Kronecker LS system are multiplied by an arbitrary diagonal weights matrix. These arbitrary weights would result in a potentially very large non-Kronecker least-squares problem that could be impractical to solve as it would require significant memory and computational power.

This paper extended the T-LARS algorithm, earlier developed by the authors [28], to become the Weighted Tensor Least Angle Regression (WT-LARS) algorithm that could solve efficiently either L_0 or L_1 constrained multilinear least-squares problems with arbitrary diagonal weights for all critical values of their regularization parameter. To validate our new WT-LARS algorithm, we used it to solve three image inpainting problems. In our experimental results using WT-LARS shown in Figs. 3.1 and 3.2, we obtained the exact sparse signal representation of RGB images behind fences after applying zero weights to the pixels representing the fences. In the experimental result using WT-LARS shown in Fig. 3.3, we successfully obtained an exact sparse signal representation of an RGB landscape image occluded by a person by applying zero weights to the pixels representing this person. These results demonstrate the validity and usefulness of our new Weighted Least Angle Regression (WT-LARS) algorithm.

Possible future applications of WT-LARS include efficiently solving weighted least-squares applications for tensor signals. Such examples include tensor completion, image/video inpainting, image/video smoothing, and tensor signal restorations.

A MATLAB GPU-based implementation of our Weighted Tensor Least Angle Regression (WT-LARS) algorithm, Algorithm 1, is available at https://github.com/SSSherif/Weighted-Tensor-Least-Angle-Regression.

References

1. Repetti, A., Chouzenoux, E., Pesquet, J.C.: A penalized weighted least squares approach for restoring data corrupted with signal-dependent noise. Eur. Signal Process. Conf., 1553–1557 (2012)
2. Wang, J., Lu, H., Wen, J., Liang, Z.: Multiscale penalized weighted least-squares sinogram restoration for low-dose X-ray computed tomography. IEEE Trans. Biomed. Eng. **55**, 1022–1031 (2008). https://doi.org/10.1109/TBME.2007.909531
3. Cheung, K.W., So, H.C., Ma, W.K., Chan, Y.T.: Least squares algorithms for time-of-arrival-based mobile location. IEEE Trans. Signal Process. **52**, 1121–1128 (2004). https://doi.org/10.1109/TSP.2004.823465
4. Zou, Y., Liu, H., Wan, Q.: An iterative method for moving target localization using TDOA and FDOA measurements. IEEE Access. **6**, 2746–2754 (2017). https://doi.org/10.1109/ACCESS.2017.2785182
5. Tarrío, P., Bernardos, A.M., Besada, J.A., Casar, J.R.: A new positioning technique for RSS-based localization based on a weighted least squares estimator. ISWCS'08 - Proc. 2008 IEEE Int. Symp. Wirel. Commun. Syst., 633–637 (2008). https://doi.org/10.1109/ISWCS.2008.4726133
6. Wang, W., Tan, W., Shi, W., Zhang, Q., Li, H.: Direction finding based on iterative adaptive approach utilizing weighted $$\ell _2$$ ℓ 2 -norm penalty for acoustic vector sensor array. Multidimens. Syst. Signal Process. **331**(33), 247–261 (2021). https://doi.org/10.1007/S11045-021-00797-6
7. Chan, S.C.K.: Adaptive weighted least squares algorithm for Volterra signal modeling. IEEE Trans. Circuits Syst. I Fundam. Theory Appl. **47**, 545–554 (2000). https://doi.org/10.1109/81.841856
8. De Courville, M., Duhamel, P.: Adaptive filtering in subbands using a weighted criterion. IEEE Trans. Signal Process. **45**, 1675 (1997). https://doi.org/10.1109/icassp.1995.480341
9. Zhao, R., Lai, X.: A fast matrix iterative technique for the WLS design of 2-D quadrantally symmetic FIR filters. Multidimens. Syst. Signal Process. 2010 224. 22, 303–317 (2010). https://doi.org/10.1007/S11045-010-0128-X
10. Min, D., Choi, S., Lu, J., Ham, B., Sohn, K., Do, M.N.: Fast global image smoothing based on weighted least squares. IEEE Trans. Image Process. **23**, 5638–5653 (2014). https://doi.org/10.1109/TIP.2014.2366600
11. Ruppert, D., Wand, M.P.: Multivariate locally weighted least squares regression. Ann. Stat. **22**, 1346–1370 (2007). https://doi.org/10.1214/aos/1176325632
12. Magee, L.: Improving survey-weighted least squares regression. J. R. Stat. Soc. Ser. B Stat. Methodol. **60**, 115–126 (1998). https://doi.org/10.1111/1467-9868.00112
13. Romano, J.P., Wolf, M.: Resurrecting weighted least squares. J. Econom. **197**, 1–19 (2017). https://doi.org/10.1016/j.jeconom.2016.10.003
14. Wickramasingha, I., Sobhy, M., Sherif, S.S.: Sparsity in Bayesian Signal Estimation. In: Tejedor, J.P. (ed.) Bayesian Inference. InTech (2017)
15. Mallat, S.: A Wavelet Tour of Signal Processing. Elsevier (2009)
16. Daubechies, I.: The wavelet transform, time-frequency localization and signal analysis. IEEE Trans. Inf. Theory **36**, 961–1005 (1990). https://doi.org/10.1109/18.57199
17. Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. **41**, 3397–3415 (1993). https://doi.org/10.1109/78.258082
18. Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.S.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: Conference Record of the Asilomar Conference on Signals, Systems & Computers, pp. 40–44. IEEE Comput. Soc. Press (1993)
19. Coifman, R.R., Wickerhauser, M.V.: Entropy-based algorithms for best basis selection. IEEE Trans. Inf. Theory **38**, 713–718 (1992). https://doi.org/10.1109/18.119732
20. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. **20**, 33–61 (1998). https://doi.org/10.1137/S1064827596304010

21. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., Ishwaran, H., Knight, K., Loubes, J.M., Massart, P., Madigan, D., Ridgeway, G., Rosset, S., Zhu, J.I., Stine, R.A., Turlach, B.A., Weisberg, S., Johnstone, I., Tibshirani, R.: Least angle regression. Ann. Stat. **32**, 407–499 (2004). https://doi.org/10.1214/009053604000000067
22. Tropp, J.A.: Greed is good: algorithmic results for sparse approximation. IEEE Trans. Inf. Theory **50**, 2231–2242 (2004). https://doi.org/10.1109/TIT.2004.834793
23. Friedlander, M.P., Mansour, H., Saab, R., Yilmaz, Ö.: Recovering compressively sampled signals using partial support information. IEEE Trans. Inf. Theory **58**, 1122–1134 (2012). https://doi.org/10.1109/TIT.2011.2167214
24. Khajehnejad, M.A., Xu, W., Avestimehr, A.S., Hassibi, B.: Weighted l1 minimization for sparse recovery with prior information. IEEE Int. Symp. Inf. Theory - Proc. 483–487 (2009). https://doi.org/10.1109/ISIT.2009.5205716
25. Candès, E.J., Wakin, M.B., Boyd, S.P.: Enhancing sparsity by reweighted$\ell 1$ minimization. J. Fourier Anal. Appl. **14**, 877–905 (2008). https://doi.org/10.1007/s00041-008-9045-x
26. Bergersen, L.C., Glad, I.K., Lyng, H.: Weighted lasso with data integration. Stat. Appl. Genet. Mol. Biol. **10** (2011). https://doi.org/10.2202/1544-6115.1703
27. Shimamura, T., Imoto, S., Yamaguchi, R., Miyano, S.: Weighted lasso in graphical Gaussian modeling for large gene network estimation based on microarray data. Genome Inform. **19**, 142–153 (2007). https://doi.org/10.1142/9781860949852_0013
28. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**, 455–500 (2009). https://doi.org/10.1137/07070111X
29. Cichocki, A., Mandic, D., De Lathauwer, L., Zhou, G., Zhao, Q., Caiafa, C., Phan, H.A.: Tensor decompositions for signal processing applications: from two-way to multiway component analysis (2015)
30. Wickramasingha, I., Sobhy, M., Elrewainy, A., Sherif, S.S.: Tensor least angle regression for sparse representations of multidimensional signals. Neural Comput. **32**, 1697–1732 (2020). https://doi.org/10.1162/neco_a_01304
31. Sulam, J., Ophir, B., Zibulevsky, M., Elad, M.: Trainlets: dictionary learning in high dimensions. IEEE Trans. Signal Process. **64**, 3180–3193 (2016). https://doi.org/10.1109/TSP.2016.2540599
32. Caiafa, C.F., Cichocki, A.: Computing sparse representations of multidimensional signals using Kronecker bases. Neural Comput. **25**, 1–35 (2012). https://doi.org/10.1162/NECO_a_00385
33. Elrewainy, A., Sherif, S.S.: Kronecker least angle regression for unsupervised unmixing of hyperspectral imaging data. Signal, Image Video Process. **14**, 359–367 (2020). https://doi.org/10.1007/s11760-019-01562-w
34. Schäcke, K.: On the Kronecker product, 1–35 (2013)
35. Moon, T.K., Stirling, W.C.: Mathematical methods and algorithms for signal processing (1999)
36. Bocci, C., Carlini, E., Kileel, J.: Hadamard products of linear spaces. J. Algebr. **448**, 595–617 (2016). https://doi.org/10.1016/j.jalgebra.2015.10.008
37. Zhao, Q., Caiafa, C.F., Mandic, D.P., Chao, Z.C., Nagasaka, Y., Fujii, N., Zhang, L., Cichocki, A.: Higher order partial least squares (HOPLS): A generalized multilinear regression method. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1660–1673 (2013). https://doi.org/10.1109/TPAMI.2012.254
38. Phan, A., Cichocki, A., Tichavsk, P., Luta, G., Brockmeier, A.: Tensor completion through multiple Kronecker product decomposition Institute of Information Theory and Automation. Proc. ICASSP. c, 3233–3237 (2013)
39. Bugg, C., Chen, C., Aswani, A.: Nonnegative Tensor Completion via Integer Optimization. (2021)
40. Yu, Q., Zhang, X., Chen, Y., Qi, L.: Low Tucker rank tensor completion using a symmetric block coordinate descent method. Numer. Linear Algebr. with Appl. **30**, e2464 (2022). https://doi.org/10.1002/nla.2464
41. Elharrouss, O., Almaadeed, N., Al-Maadeed, S., Akbari, Y.: Image inpainting: a review, https://link.springer.com/article, https://doi.org/10.1007/s11063-019-10163-0 (2020)
42. Xiang, H., Zou, Q., Nawaz, M.A., Huang, X., Zhang, F., Yu, H.: Deep learning for image inpainting: A survey. Pattern Recognit. **134**, 109046 (2023). https://doi.org/10.1016/j.patcog.2022.109046

43. Romero, A., Castillo, A., Abril-Nova, J., Timofte, R., Das, R., Hira, S., Pan, Z., Zhang, M., Li, B., He, D., Lin, T., Li, F., Wu, C., Liu, X., Wang, X., Yu, Y., Yang, J., Li, R., Zhao, Y., Guo, Z., Fan, B., Li, X., Zhang, R., Lu, Z., Huang, J., Wu, G., Jiang, J., Cai, J., Li, C., Tao, X., Tai, Y.W., Zhou, X., Huang, H.: NTIRE 2022 Image Inpainting Challenge: Report. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1149–1181 (2022)
44. Park, M., Brocklehurst, K., Collins, R.T., Liu, Y.: Deformed lattice detection in real-world images using mean-shift belief propagation. IEEE Trans. Pattern Anal. Mach. Intell. **31**, 1804–1816 (2009). https://doi.org/10.1109/TPAMI.2009.73
45. Agustsson, E., Timofte, R.: NTIRE 2017 challenge on single image super-resolution: dataset and study. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1122–1131 (2017)