mathematics

MDPI

Article

Numerical Methods for Decorrelation Stretch

Elisa Crabu †, Federica Pes *,† and Giuseppe Rodriguez †

Department of Mathematics and Computer Science, University of Cagliari, via Ospedale 72, 09124 Cagliari, Italy; elisa.crabu@unica.it (E.C.); rodriguez@unica.it (G.R.)

* Correspondence: federica.pes@unica.it

† These authors contributed equally to this work.

Abstract

Decorrelation stretch is an image enhancement technique that emphasizes color differences, which also applicable to multispectral datasets. It transforms an image so that its color plane result is uncorrelated, with assigned variances. The standard algorithm may suffer from numerical instability. Moreover, it is not able to manage degenerate cases, where color planes are linearly dependent. In this paper, we review the theory behind decorrelation stretch and propose some alternative algorithms that resolve the issues of the standard approach.

Keywords: decorrelation stretch; image enhancement; RGB images; QR factorization; singular value decomposition (SVD)

MSC: 15A04; 15A18; 62H35; 68U10

1. Introduction

Image enhancement is a well-developed topic, whose aim is to modify the appearance of an image in order to simplify the interpretation of the information contained in it. Different methods have been developed, such as linear contrast enhancement, histogram equalization, neural network, or Gaussian stretch, but some image transformation techniques or filtering methods can also be described as enhancement models [1–4]. Most of these methods try to enhance (stretch) color differences in images, operating on color bands. Some of them work on the channels separately, but, since color planes may be correlated, the stretching should not be independent for each color [5]. Therefore, it is, in general, preferable to work on the three color bands at the same time.

Decorrelation stretch is an image transformation technique used to enhance color differences, which operates on the three channels simultaneously. It aims to remove the inter-channel correlation found in image pixels; hence, the name "decorrelation stretch".

It is based on Principal Component Analysis (PCA). Starting from a color image, it finds a linear transformation that, under suitable assumptions, moves data in a reference system where the color bands are uncorrelated. Then, the bands are stretched to enhance color differences. The aim is to achieve an output image whose pixels are spread along all possible colors, resulting in an improvement in the intensity and saturation of colors. This model has been largely studied [6–9], and applied in different fields, e.g., geological studies described in [10–12], or archeological applications in [13–24], or medical studies in [25–28].

The method is most effective for images where bands (channels) are highly correlated, e.g., satellite or aerial images where different bands capture very similar information. Even in standard natural-color photographs, the red, green, and blue channels are not

check for updates

Academic Editors: Gianluca Vinti, Ivan Gerace and Arianna Travaglini

Received: 9 September 2025
Revised: 9 October 2025
Accepted: 12 October 2025
Published: 15 October 2025

Citation: Crabu, E.; Pes, F.; Rodriguez, G. Numerical Methods for Decorrelation Stretch. *Mathematics* **2025**, *13*, 3297. https://doi.org/10.3390/math13203297

independent. Decorrelation stretch reduces this correlation to highlight subtle differences in color. Our work is motivated by the necessity to understand whether there are paintings in rock walls of archeological sites that have been hidden by the passage of time or confused with the presence of moss and vegetation.

While digital color images generally consist of three matrices storing red, green, and blue (RGB) components of each pixel, multispectral images investigate information in specific wavelength ranges, even non-visible ones. They are used in remote sensing for capturing information that the human eye fails to detect, and often include 7–15 spectral bands. A particular situation is that of *hyperspectral* images, which may involve hundreds of bands.

In some applications, to be able to visualize data as a standard color image, it is common to select in advance three particular frequency bands in a multispectral image, on the basis of physical information on the phenomenon under scrutiny. However, this does not necessarily have to be performed before data processing, as decorrelation stretch, like other image enhancement techniques, does not substantially change if applied to three-plane images or to multispectral ones. Moreover, processing the complete dataset leads to automatically identifying the spectral components which are most relevant in the observation.

In this work, we first resume the decorrelation stretch method and clearly state its statistical properties. In particular, we discuss the degenerate case of linearly dependent color planes, which does not appear to have been discussed in the past. Then, we propose some new algorithms, with the aim of reducing the computational load and increasing accuracy. One of the proposed methods is also able to correctly treat degenerate cases, producing images with decorrelated color channels. Such approaches are compared to an existing implementation of the method, namely, the MATLAB function `decorrstretch`; in this paper we always refer to version R2024b of MATLAB. Reducing the complexity of the computation is particularly useful when large datasets have to be processed, or when hyperspectral images are considered.

The paper is organized as follows: Section 2 recalls some basic statistical concepts to fix the notation; Section 3 reviews the decorrelation stretching process and its MATLAB implementation `decorrstretch`; and Section 4 underlines some issues of this implementation and explains how we propose to solve them. Section 5 shows some applications and discusses the results obtained by applying the proposed methods. Lastly, Section 6 contains our final considerations.

2. Statistical Preliminaries

Let y be a random variable with the distribution function $F(y)$. Its mean value and variance are defined as

$$\mu = E[y] = \int_{-\infty}^{\infty} y \, dF(y), \tag{1}$$

$$\sigma^2 = E[(y-\mu)^2] = \int_{-\infty}^{\infty} (y-\mu)^2 \, dF(y). \tag{2}$$

Now, let $\mathbf{y} = (y_1, \ldots, y_n)^T$ be a vector of random variables. If $F(y_i, y_j)$ is the joint distribution of y_i and y_j, then the covariance of the two random variables is

$$\sigma_{i,j} = \operatorname{cov}(y_i, y_j) = E[(y_i - \mu_i)(y_j - \mu_j)] = \int_{\mathbb{R}^2} (y_i - \mu_i)(y_j - \mu_j) \, dF(y_i, y_j).$$

Notice that $\sigma_{i,i} = \sigma_i^2 = E[(y_i - \mu_i)^2]$. The variables y_i and y_j are said to be uncorrelated if $\sigma_{i,j} = 0$.

For a vector of random variables, we extend the definition of the operator E to act, componentwise, on its arguments. So, we will write $\boldsymbol{\mu} = E[\mathbf{y}] \in \mathbb{R}^n$, meaning that $\mu_i = E[y_i]$, and define the variance–covariance matrix of $\mathbf{y}$ as

$$\mathcal{V} = \mathcal{V}[\mathbf{y}] = E[(\mathbf{y} - \boldsymbol{\mu})(\mathbf{y} - \boldsymbol{\mu})^T],$$

so that $\mathcal{V}_{i,j} = \sigma_{i,j}$.

The correlation matrix of $\mathbf{y}$ is obtained by

$$\mathcal{C} = \mathcal{C}[\mathbf{y}] = \Sigma^{-1}\mathcal{V}\Sigma^{-1}, \tag{3}$$

where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$ contains the standard deviations of the components of $\mathbf{y}$. Its diagonal entries are unitary, while the off-diagonal elements are the correlation coefficients for each pair of variables, i.e., $\mathcal{C}_{i,j} = \frac{\sigma_{i,j}}{\sigma_i \sigma_j}$, for $i \neq j$. We will say that a random vector is uncorrelated when each pair of its components is as such. The correlation coefficient is undefined when the variable is constant, as it has zero variance.

The following theorem can be easily proved by applying the linearity of E.

Theorem 1. *Let M be a matrix of dimension $r \times n$ and let $\mathbf{z} = M\mathbf{y}$. If $\boldsymbol{\mu} = E[\mathbf{y}]$ and $\mathcal{V} = \mathcal{V}[\mathbf{y}]$, then the mean value and the variance–covariance matrix of the linearly transformed random vector are*

$$E[\mathbf{z}] = M\boldsymbol{\mu} \qquad \text{and} \qquad \mathcal{V}[\mathbf{z}] = M\mathcal{V}M^T.$$

When a discrete sampling $(y_1, \ldots, y_m)$ of a random variable is available, assuming uniform distribution, the mean value (1) and variance (2) are computed by

$$\mu = \frac{1}{m}\sum_{i=1}^{m} y_i, \qquad \sigma^2 = \frac{1}{m-1}\sum_{i=1}^{m}(y_i - \mu)^2.$$

In the case of a random vector $\mathbf{y}$, we will pack its realizations in a matrix

$$Y = \Big[\mathbf{y}_1, \ldots, \mathbf{y}_m\Big],$$

and write

$$\boldsymbol{\mu} = \frac{1}{m}Y\mathbf{u}, \qquad \mathcal{V} = \frac{1}{m-1}(Y - \boldsymbol{\mu}\mathbf{u}^T)(Y - \boldsymbol{\mu}\mathbf{u}^T)^T, \tag{4}$$

where $\mathbf{u} = (1, \ldots, 1)^T \in \mathbb{R}^m$.

Remark 1. *Statistical independence and linear independence, commonly encountered in numerical linear algebra, are fundamentally different concepts and should not be conflated. First of all, if two variables are statistically independent and possess finite variance, then they are necessarily uncorrelated. However, the converse is not true: uncorrelated variables are not necessarily statistically independent. Nevertheless, uncorrelatedness implies the absence of a linear relationship between the variables, but nonlinear forms of dependence may still be present.*

3. Decorrelation Stretch

Decorrelation stretch (DS), starting from a color image, seeks a linear spatial transformation which decorrelates and stretches colors. The process can be described by matrix and vector operations, and can be expressed by a single mathematical transformation that takes, as input, an image and produces the decorrelation-stretched output.

DS [6] is strictly connected to Principal Component Analysis (PCA). The basic PCA procedure leads to the construction of a vector of uncorrelated random variables, starting from a vector of correlated ones, by changing the reference system.

Let $\mathbf{y}$ be a vector of random variables with mean value $\boldsymbol{\mu}$ and variance–covariance (V-C) matrix $\mathcal{V}$. The matrix $\mathcal{V}$ is symmetric and positive semi-definite, so it admits a spectral factorization

$$\mathcal{V} = Q\Lambda Q^T,$$

where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ contains the eigenvalues $\lambda_i \geq 0$ and the columns of the orthogonal matrix $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_n]$ are the eigenvectors.

Proposition 1. *The components of the transformed vector* $\mathbf{z} = Q^T\mathbf{y}$ *are uncorrelated.*

Proof. By Theorem 1 the new vector $\mathbf{z} = Q^T\mathbf{y}$ has mean $E[\mathbf{z}] = Q^T\boldsymbol{\mu}$ and V-C matrix

$$\mathcal{V}[\mathbf{z}] = Q^T\mathcal{V}Q = \Lambda,$$

that is, the components of $\mathbf{z}$ are uncorrelated with variances equal to the eigenvalues of $\mathcal{V}$. □

Decorrelation stretch allows one to construct a new random vector of uncorrelated components with the desired variances.

Proposition 2. *Under the assumption that* $\lambda_i \neq 0$, $\forall\ i = 1, \ldots, n$, *let us consider the "DS" transformation*

$$M = \Sigma_T Q \Lambda^{-\frac{1}{2}} Q^T, \tag{5}$$

where

$$\Lambda^{-\frac{1}{2}} = \text{diag}\left(\frac{1}{\sqrt{\lambda_1}}, \ldots, \frac{1}{\sqrt{\lambda_n}}\right)$$

and $\Sigma_T = \text{diag}(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$, *for chosen target variances* $\overline{\sigma}_i^2$. *Then, the components of the transformed vector* $\mathbf{z} = M\mathbf{y}$ *are uncorrelated with variances* $\overline{\sigma}_i^2$.

Proof. Computing the V-C matrix of $\mathbf{z} = M\mathbf{y}$ by Theorem 1 and using the spectral factorization of $\mathcal{V}$, one obtains

$$\mathcal{V}[\mathbf{z}] = M\mathcal{V}M^T = (\Sigma_T Q \Lambda^{-\frac{1}{2}} Q^T)(Q\Lambda Q^T)(Q\Lambda^{-\frac{1}{2}} Q^T \Sigma_T) = \Sigma_T^2, \tag{6}$$

that is, the new variables z_i are uncorrelated with variance $\overline{\sigma}_i^2$. □

A similar transformation based on the correlation matrix (3) of $\mathbf{y}$ can be constructed. In this case, one considers its spectral factorization

$$\mathcal{C} = Q\Lambda Q^T.$$

Proposition 3. *Assuming* $\mathcal{C}$ *is symmetric and positive definite, i.e.,* $\lambda_i > 0$, *and considering the transformation*

$$N = \Sigma_T Q \Lambda^{-\frac{1}{2}} Q^T \Sigma^{-1}, \tag{7}$$

the V-C matrix of the transformed vector $\mathbf{w} = N\mathbf{y}$ *is* $\mathcal{V}[\mathbf{w}] = \Sigma_T^2$.

Proof. From Theorem 1 and from the definition of correlation matrix (3), using its spectral factorization, one obtains

$$\begin{aligned}\mathcal{V}[\mathbf{w}] = N\mathcal{V}N^T &= (\Sigma_T Q\Lambda^{-\frac{1}{2}}Q^T\Sigma^{-1})\mathcal{V}(\Sigma^{-1}Q\Lambda^{-\frac{1}{2}}Q^T\Sigma_T)\\ &= (\Sigma_T Q\Lambda^{-\frac{1}{2}}Q^T)(Q\Lambda Q^T)(Q\Lambda^{-\frac{1}{2}}Q^T\Sigma_T) = \Sigma_T^2.\end{aligned}$$

□

This shows that even if the transformed vectors $\mathbf{z} = M\mathbf{y}$ and $\mathbf{w} = N\mathbf{y}$ differ in general, they share important statistical features, as they are both uncorrelated and their V-C matrices coincide.

A dataset is considered an image when it consists of a 3D array $\mathcal{I}$ of size $r \times c \times n$. Monochrome images are characterized by $n = 1$; the resulting $r \times c$ matrix contains grayscale intensity information for each pixel. For RGB images, $n = 3$ and the layers $\mathcal{I}_k = \mathcal{I}_{(\cdot,\cdot,k)}$, $k = 1, 2, 3$, store intensity information for the red, green, and blue color channels. In multispectral and hyperspectral images, n is larger than 3 and each layer corresponds to a particular electromagnetic frequency band. Other applications may concern datasets that are not directly connected to visible phenomena or to electromagnetic waves, but that can be regarded as "images".

When images are numerically processed, they are usually "vectorized", that is, the array $\mathcal{I}$ is converted to a matrix A of size $p \times n$, where $p = rs$ is the number of pixels. Each column $A_k = A_{(\cdot,k)}$ of A is obtained by lexicographically ordering the corresponding kth layer.

In DS, one associates to an image pixel a random vector $\mathbf{y} \in \mathbb{R}^n$, whose components represent the corresponding "color" layer information for that pixel. An image contains p realizations of this random vector

$$Y = \left[\mathbf{y}_1, \ldots, \mathbf{y}_p\right] = A^T, \tag{8}$$

which are usually assumed to be uniformly distributed. This makes it possible to associate to the image a mean vector and a V-C matrix by Formula (4), that is

$$\boldsymbol{\mu} = \frac{1}{p}A^T\mathbf{u}, \qquad \mathcal{V} = \frac{1}{p-1}(A^T - \boldsymbol{\mu}\mathbf{u}^T)(A^T - \boldsymbol{\mu}\mathbf{u}^T)^T. \tag{9}$$

At this point, the linear transformation (5) is applied to Y. This rotates the reference system in that of the principal components, i.e., the eigenvectors of $\mathcal{V}$, where the variables are uncorrelated. Here, the variables are scaled to take on unitary variances, and then they are rotated back to the original reference, where they are still uncorrelated. The resulting vectors are stretched so that they assume target variances $\overline{\sigma}_i^2$, $i = 1, \ldots, n$. Finally, target mean values $\boldsymbol{\mu}_T$ can be imposed on the resulting dataset.

The complete transformation is

$$S = MY + (\boldsymbol{\mu}_T - M\boldsymbol{\mu})\mathbf{u}^T = M(Y - \boldsymbol{\mu}\mathbf{u}^T) + \boldsymbol{\mu}_T\mathbf{u}^T,$$

and the final $n \times p$ matrix S, after transposition, is reshaped in a $r \times c \times n$ "image". By default, one usually sets $\Sigma_T = \Sigma$ and $\boldsymbol{\mu}_T = \boldsymbol{\mu}$. A similar process can be constructed by substituting the correlation matrix $\mathcal{C}$ to the V-C matrix $\mathcal{V}$, and the transformation (7) to (5). The above procedure is implemented in the MATLAB function `decorrstretch`.

The Special Case of a Singular V-C Matrix

When an eigenvalue of the matrix $\mathcal{V}$ is zero, transformation (5) cannot be constructed, due to the presence of the matrix $\Lambda^{-\frac{1}{2}}$. Given definition (3), the same happens when the construction is based on the correlation matrix.

Two significant situations in which this feature appears are the following. First of all, if the ith color plane is constant, the variance of the component y_i of the random vector

$\mathbf{y}$ is zero, as well as the covariances of the pairs (y_i, y_j) for any $j \neq i$. This means that the ith color plane is completely uncorrelated to the other ones. It might be removed from the dataset before processing, and inserted back after the computation.

Another relevant case is when a color plane is a linear combination of other planes. In this case, that layer is intrinsically correlated to the ones that generated it, and DS cannot diagonalize the V-C matrix. We remark that if a group of layers is linearly dependent, it is possible to identify the layers to be removed by applying the modified Gram–Schmidt method [29] to the columns of A. Indeed, a breakdown at the kth step of the algorithm indicates that the kth vector is a linear combination of the previous ones. In this case, the vector can simply be removed before executing the next step.

The case of a singular V-C matrix, that is, a constant or perfectly linearly dependent color channel, is rather unlikely to be encountered in field photography or in remote sensing. However, it might happen in applications where the observed target is illuminated by light of a specific color, which happens in monochrome microscopy. Furthermore, a mathematical model and its software implementation should be robust with respect to input data, so the problem must be addressed.

In the following, we give a mathematical justification of the failure of DS in the presence of zero eigenvalues. Formally, it is possible to proceed even in this situation by substituting $\Lambda^{-\frac{1}{2}}$ with the square root of the pseudoinverse $\sqrt{\Lambda^{\dagger}}$. The pseudoinverse matrix can be defined in terms of the SVD factorization (see [29]), but in the case of a diagonal matrix, its construction is immediate. Indeed, if $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, then the pseudoinverse $\Lambda^{\dagger}$ is a diagonal matrix with diagonal elements

$$(\Lambda^{\dagger})_{i,i} = \begin{cases} \lambda_i^{-1}, & \lambda_i \neq 0, \\ 0, & \lambda_i = 0. \end{cases}$$

If this happens, Equation (6) becomes

$$\mathcal{V}[\mathbf{z}] = M\mathcal{V}M^T = \Sigma_T Q\sqrt{\Lambda^{\dagger}}\Lambda\sqrt{\Lambda^{\dagger}}Q^T\Sigma_T = \Sigma_T Q\mathcal{J}Q^T\Sigma_T, \tag{10}$$

where the matrix $\mathcal{J} = \sqrt{\Lambda^{\dagger}}\Lambda\sqrt{\Lambda^{\dagger}}$ is diagonal with diagonal elements

$$\mathcal{J}_{i,i} = \begin{cases} 1 & \lambda_i \neq 0, \\ 0, & \lambda_i = 0. \end{cases}$$

Due to this fact, the matrix $Q\mathcal{J}Q^T$ appearing in (10) is nondiagonal. Indeed, it is the linear transformation that orthogonally projects a vector in the space spanned by the eigenvectors of $\mathcal{V}$ corresponding to nonzero eigenvalues. As a consequence, the components of the resulting random vector $\mathbf{z}$ are not uncorrelated, and DS cannot be successfully applied.

The MATLAB function `decorrstretch` implements the above procedure based on the pseudoinverse of Λ. Additionally, it considers the transformation

$$\widetilde{\mathbf{z}} = \widetilde{M}\mathbf{y} = KM\mathbf{y},$$

where the diagonal matrix K is given by

$$K = \Sigma_T\Big(\operatorname{diag}(M\mathcal{V}M^T)^{\dagger}\Big)^{\frac{1}{2}}.$$

It is straightforward to verify that if the V-C matrix $\mathcal{V}$ is nonsingular, K is the identity matrix, so that in this case the modified algorithm does not change the result of DS.

When at least one of the eigenvalues of $\mathcal{V}$ is zero, K introduces a scaling factor in the components of $\widetilde{\mathbf{z}}$, so that its variances are the target values $\overline{\sigma}_i^2$. In any case, this transformation does not change the fact that the components are correlated, so DS has practically no effect on the enhancement of the image. For this reason, we will not consider this transformation in our computation. Our decision is supported by the fact that the presence of a linearly dependent color layer is not likely in experimental data. Moreover, in case this happens, we consider it preferable to remove the dependent layer from the image, as it does not contain significant information, and process the reduced dataset. If one chooses to process the correlation matrix $\mathcal{C}$ and applies the transformation (7), all the above considerations are still valid.

In the following, we will describe an algorithm for DS that can detect a degenerate case and automatically reduce the dataset to produce an output image with uncorrelated color channels.

4. New Numerical Approaches to DS

The algorithm outlined in Section 3 has various numerical flaws. Firstly, it stands on the computation of the matrix $C = A^T A$. It is known that such operation is heavily affected by underflows and overflows, which may cause loss of information, and is not backward stable ([29], Section 2.1.2). This last statement means that for the computed matrix $\overline{C}$, which is influenced by rounding errors, it is not possible to prove that there exists a small perturbation matrix E, such that $\overline{C} = (A + E)^T (A + E)$.

Further, we showed in Section 3 that in the case of a singular V-C matrix, the approach used in the MATLAB function `decorrstretch` does not produce uncorrelated color components for an image, and is of little use. Finally, the above procedure does not automatically deal with linearly dependent color planes and, to produce uncorrelated data, forces the user to check for the presence of numerically zero eigenvalues of the V-C matrix, remove the corresponding components from the data, and rerun the DS algorithm.

To deal with these problems, we propose to perform the computation using the singular value decomposition (SVD) of A, instead of computing $A^T A$. A clever use of a QR factorization, before computing the SVD, leads to a modified algorithm requiring less computing time and a smaller storage space, but also gives the opportunity to efficiently treat the case of a singular V-C matrix, producing uncorrelated stretched color planes. If accuracy is not an issue, but speed of processing is, we propose to apply a randomized approach to any of the previous methods to reduce their runtime.

The algorithms introduced in this paper produce results equivalent to those obtained when DS is applied to the V-C matrix. They are not suitable to be applied to the correlation matrix. In any case, we remark that the resulting random vectors are statistically equivalent because, as already noted in Proposition 3, they share the same V-C matrix. Moreover, according to our experience, the results produced by the two approaches are visually indistinguishable.

4.1. SVD-Based Decorrelation Stretch

We consider the compact SVD factorization of $(Y - \boldsymbol{\mu}\mathbf{u}^T)^T = A - \mathbf{u}\boldsymbol{\mu}^T$ (see Equations (8) and (9))

$$A - \mathbf{u}\boldsymbol{\mu}^T = UDV^T, \qquad D = \operatorname{diag}(d_1, \ldots, d_n), \tag{11}$$

where $U \in \mathbb{R}^{p \times n}$ and $V \in \mathbb{R}^{n \times n}$ have orthonormal columns, called *left and right singular vectors*, and $d_i \geq 0$ are the *singular values*. The number of nonzero singular values equals the rank of the matrix, and of $\mathcal{V}$.

While V and D are "small" matrices, of size coincident with the number of color planes in the image, U has the same size as A, so it may be quite large. In any case, the matrix U is not needed in the process. Indeed, the right singular vectors, i.e., the columns of V, are the eigenvectors of $\mathcal{V}$, and its eigenvalues coincide with $d_i^2/(p-1)$, for $i = 1, \ldots, n$. This can be easily verified by substituting the SVD (11) in the computation of the V-C matrix

$$\mathcal{V} = \frac{1}{p-1}\left(Y - \boldsymbol{\mu}\mathbf{u}^T\right)\left(Y - \boldsymbol{\mu}\mathbf{u}^T\right)^T = \frac{1}{p-1}VD^TDV^T.$$

This implies that transformation (5) based on the SVD can be written as

$$M = \sqrt{p-1}\,\Sigma_T V D^{-1} V^T. \tag{12}$$

4.2. Q-Less QR + SVD

Even if the matrix U in (11) is not required for DS, the SVD algorithm creates it. To reduce time and storage space, one can consider an approach based on the application of the compact QR decomposition

$$\left(Y - \boldsymbol{\mu}\mathbf{u}^T\right)^T = QR, \tag{13}$$

where Q is a matrix with orthogonal columns of size $p \times n$ and R is a square upper triangular matrix of size n. The factorization algorithm allows one to avoid the allocation and computation of the matrix Q. The triangularization is performed by applying n fast Householder transformations to the starting matrix [30]. This feature is implemented in the `qr` function of MATLAB and is called *Q-less* QR factorization.

Once (13) is obtained, the SVD decomposition $R = \widetilde{U}DV^T$ of the resulting triangular matrix is computed. Then, the SVD of the initial matrix is given by

$$\left(Y - \boldsymbol{\mu}\mathbf{u}^T\right)^T = Q\widetilde{U}DV^T = UDV^T,$$

where the matrix $U = Q\widetilde{U}$ is not explicitly computed, since it is not needed in (12). This makes it possible to compute transformation M by a faster algorithm and save the allocation of a $p \times n$ matrix, while preserving the stability of the procedure.

Another advantage of this approach is that it allows the direct treatment of a rank-deficient V-C matrix. Indeed, for a fixed tolerance τ , let

$$\mathcal{S} = \{i \,:\, |r_{i,i}| < \tau\}$$

be the set of indices corresponding to "zero" diagonal elements in R. Then, the columns of A with index in $\mathcal{S}$ are either constant or linear combinations of other columns. This is a consequence of the fact that a QR factorization is an orthonormalization process for the columns of A.

If the set $\mathcal{S}$ is nonempty, the corresponding columns of A are removed, as well as the rows and columns of R with index in $\mathcal{S}$. At this point, the transformation (12) is constructed and applied to the linearly independent columns of A. Finally, the columns of A with index in $\mathcal{S}$ are reintroduced in the dataset as zero columns. This produces uncorrelated color planes, differently from what happens using the procedure described in Section 3.

If, for some reason, the V-C matrix of the original dataset is needed, it can be computed with reduced complexity by

$$\mathcal{V} = \frac{1}{p-1}R^TR.$$

In particular, the variances are given by

$$\sigma_i^2 = \frac{1}{p-1}\sum_{j=1}^{i} r_{j,i}^2, \qquad i = 1, \ldots, n.$$

4.3. A Randomized Approach

When large datasets must be processed and computing speed is important, one may be prone to renouncing some accuracy to obtain faster results.

A simple solution is to consider a randomly constructed sub-image of the original one. This can be easily performed by extracting a random subsampling of the rows of A, using this reduced dataset to compute the DS linear transformation, and then applying it to the initial image. This idea can be applied to any of the algorithms described in this paper.

In our tests, we fixed a percentage $\phi \in (0, 1]$ and extracted a subsampling of $\lfloor \phi p \rfloor$ rows of A, where $\lfloor z \rfloor$ denotes the integer part of z. As expected, the error between the computation performed with the full dataset and the reduced one becomes larger as ϕ decreases, but we will show in the numerical experiments that even relatively large errors produce almost imperceptible visual differences.

The numerical methods presented in this paper are outlined in Algorithm 1. It takes as input the image to be processed and the target variances and mean values. The method can be either SVD or QR-SVD. If the parameter ϕ is 1, the standard algorithms are applied. If $\phi < 1$, the randomized methods are applied instead. Computational routines for computing SVD and QR factorizations are contained in any scientific software environment, besides MATLAB, and are available in the LAPACK library https://www.netlib.org/lapack/ (accessed on 8 September 2025), which can be linked to all high level programming languages.

Algorithm 1 Outline of SVD, QR-SVD, and rand-QR-SVD approaches to DS.

Require: image $X \in \mathbb{R}^{r\times c\times n}$, method, target variance matrix Σ_T, target mean vector $\boldsymbol{\mu}_T$, percentage $\phi \in (0, 1]$ for the randomized approach, tolerance τ
Ensure: stretched image X

convert image pixel values to double precision
reshape image X to a matrix $A \in \mathbb{R}^{p\times n}$, where $p = rc$
store in vector $\boldsymbol{\mu}$ the mean values of the columns of A
$\tilde{A} = A - \boldsymbol{\mu}$
if $\phi < 1$ **then**
    store in σ a random subsampling of $\lfloor \phi p \rfloor$ rows of A
    $\tilde{A} = \tilde{A}_\sigma$ (apply the subsampling to $\tilde{A}$)
end if
if method = SVD **then**
    compute compact SVD factorization $\tilde{A} = USQ^T$
    $\Lambda = S/\sqrt{p-1}$
    if any $\Lambda_{i,i} < \tau$ **then**
        issue an error: the V-C matrix is numerically singular
    end if
else if method = QR-SVD **then**
    compute R as triangular factor of *Q-less* QR factorization $\tilde{A} = QR$
    construct set $\mathcal{S} = \{i : |r_{i,i}| < \tau\}$
    if $\mathcal{S}$ is not empty **then**
        remove from R and Σ_T rows and columns indexed in $\mathcal{S}$
        remove from A columns indexed in $\mathcal{S}$
        remove from $\boldsymbol{\mu}$ and $\boldsymbol{\mu}_T$ components indexed in $\mathcal{S}$
    end if
    compute compact SVD factorization $R = USQ^T$
    $\Lambda = S/\sqrt{p-1}$
end if
$M = Q\Lambda^{-1}Q^T\Sigma_T$
$A = (A - \boldsymbol{\mu})M + \boldsymbol{\mu}_T$
reshape matrix $A \in \mathbb{R}^{p\times n}$ to an image $X \in \mathbb{R}^{r\times c\times n}$

5. Results and Discussion

The numerical experiments were performed on an Intel Core i9-14900KF computer with 128 GB RAM (Intel Corporation, Santa Clara, CA, USA), running Linux Debian 12 and MATLAB R2024b.

Four test images were used in the experiments:

- RAND(r, c, n): A randomly generated image with resolution $r \times c$ and n color planes, for chosen values of r, c, and n, with pixel values uniformly distributed in $[0, 1]$;
- WEST: The aerial RGB color image `westconcordaerial.png` with resolution, 394×369, available in the MATLAB library;
- PIMENTEL: A RGB color image with resolution 4012×6016, taken by the authors at the archeological site *Domus de Janas de S'Acqua Salida* (Pimentel, Italy); *domus de janas* (fairy houses) are Neolithic funerary hypogea typical of the Island of Sardinia;
- PARIS: The multispectral dataset `paris.lan` containing a Landsat image of Paris with resolution 512×512, available in the MATLAB documentation.

The MATLAB function `dextre.m`, developed for this work, is available at the web page https://bugs.unica.it/~cana/software (accessed on 8 September 2025), as well as the PIMENTEL image, in CR3 raw format.

The first numerical test concerns the speed of execution. We initially consider random RGB images RAND$(r, r, 3)$ for $r = 200, 400, \ldots, 3000$. We apply to each test image the function `decorrstretch` from MATLAB, the proposed methods of Sections 4.1 and 4.2, which we denote SVD and QR-SVD, and the randomized approach of Section 4.3 with $\phi = 10^{-3}$, labeled as rand-QR-SVD. Each method is applied 10 times and we measure the average execution time. Figure 1 reports the measured timings in seconds. The two SVD-based methods are slightly slower than the classical approach: the ratio between their timings and `decorrstretch` for large r values is about 2 for SVD and 1.3 for QR-SVD. The last method is faster for small sizes. The randomized QR-SVD method is clearly the fastest.

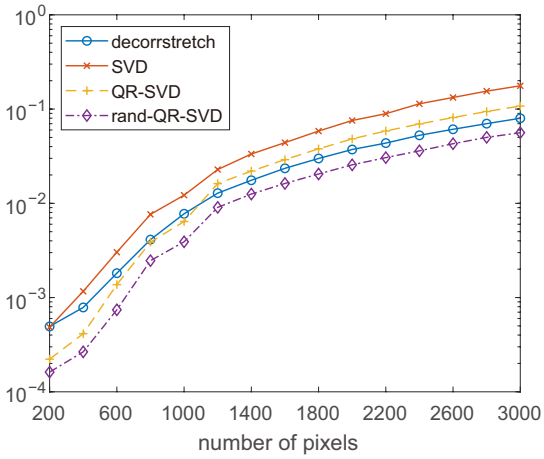

Figure 1. Execution time of the analyzed algorithms for square random RGB pictures of increasing $r \times r$ size.

In Figure 2, we repeat the same kind of test for random multispectral images RAND$(1000, 1000, n)$, with $n = 5, 10, \ldots, 50$, that is, for a fixed resolution and increasing number of color planes. The graph shows that the difference in the computing time between the methods increases with n, and that the randomized approach is essentially equivalent to `decorrstretch`. The strange behavior of the results for the SVD methods for large values of n is probably due to some kind of parallel optimization which is activated by MATLAB for large array sizes.

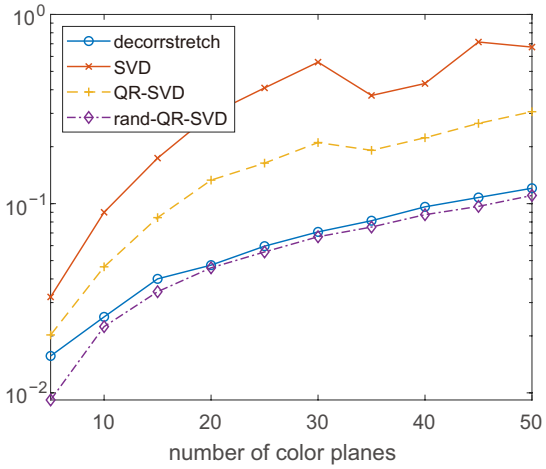

Figure 2. Execution time of the analyzed algorithms for random pictures of size 1000×1000 and increasing number of color planes.

Then, we turn to analyze the error introduced by the randomized approach. In Figure 3, we show the value of the relative error

$$\frac{\max_{i,j} |\mathcal{I}_{i,j} - \mathcal{I}^{\phi}_{i,j}|}{\max_{i,j} |\mathcal{I}_{i,j}|},$$

where $\mathcal{I}$ is the output of the QR-SVD method when applied to the WEST test problem (resolution 394×369), and $\mathcal{I}^{\phi}$ is the result produced by the same method with a random subsampling of size $\lfloor \phi p \rfloor$, for ϕ logarithmically spaced between 10^{-4} and 1. For this relatively small image, $\phi = 10^{-3}$ roughly produces a 10% error. For a large image, the error is substantially smaller, as the graph in Figure 4 shows, where the same experiment is performed on the test image PIMENTEL, of size 4012×6016.

A 10% error is practically unnoticeable, as Figure 5 shows, where the original WEST image is displayed together with the results of `decorrstretch`, QR-SVD, and rand-QR-SVD, that is, the same method with a random subsampling corresponding to $\phi = 10^{-3}$. The three images on the right are hardly distinguishable. Indeed, the *signal-to-noise ratio* (SNR) between both reconstructed images and the original one is 15.54, while the randomized method produces the SNR 15.27. The fact that the two non-randomized methods are substantially equivalent is testified by a mutual SNR of 270.21.

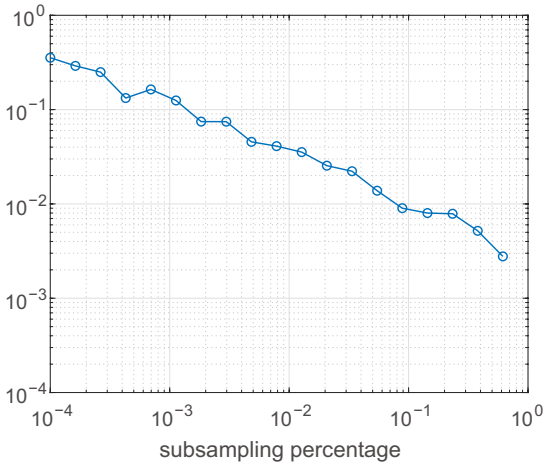

Figure 3. Relative error between the application of the QR-SVD method to test image WEST and the result obtained by rand-QR-SVD.

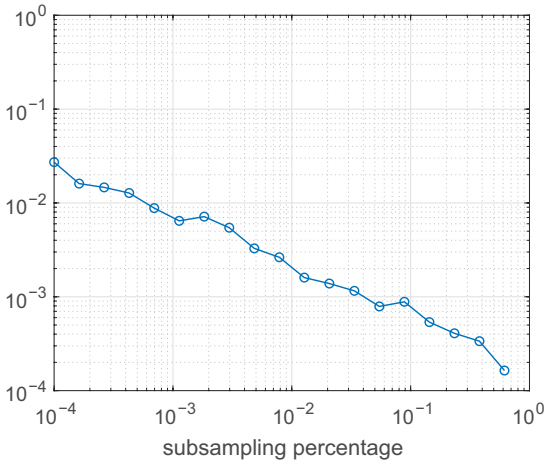

Figure 4. Relative error between the application of the QR-SVD method to test image PIMENTEL and the result obtained by rand-QR-SVD.

Figure 5. From (**left**) to (**right**): the test image WEST and the result of `decorrstretch`, QR-SVD, and rand-QR-SVD.

Figure 6 shows the original PIMENTEL image and the stretched version produced by rand-QR-SVD. Degraded paintings present on the rock wall are greatly enhanced. This helps archeologists in classifying the shapes drawn in these kinds of burial sites.

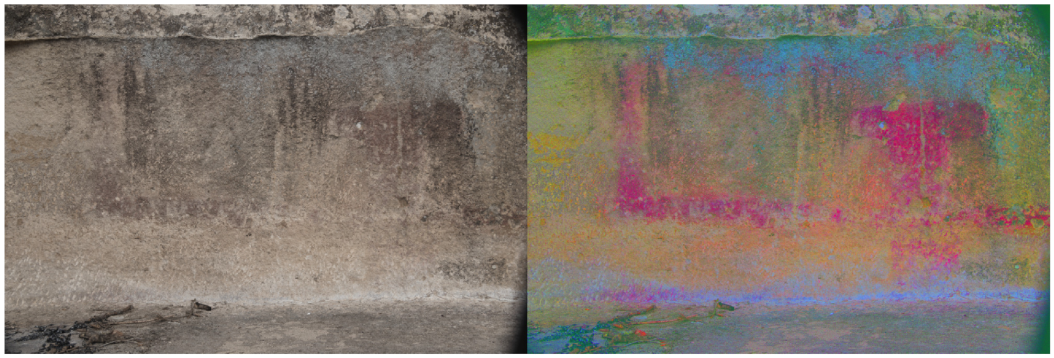

Figure 6. From (**left**) to (**right**): the test image PIMENTEL and the result of rand-QR-SVD.

As a comparison, we display in Figure 7 the images resulting from applying to the PIMENTEL image two other image enhancement techniques. We increased the contrast of the image by using the `imadjust` MATLAB function, and by histogram equalization, as implemented in the `histeq` function. The first approach requires user-defined intensity values for the colors, which cannot be estimated automatically. The second one does not require any parameters. It is clear that contrast enhancement methods act on images differently from DS and do not detect the presence of the painting.

The main advantage of the QR-SVD decorrelation stretch algorithm is its capability to automatically detect degenerate cases of linearly dependent color planes and correctly process them, generating pictures with uncorrelated color planes.

Figure 7. From (**left**) to (**right**): the effect on the PIMENTEL image produced by `imadjust` and `histeq`.

To investigate this aspect, we modified the WEST test image by making its blue color plane linearly dependent on the other two planes. We set the blue pixel values to be two times the corresponding red values plus the green ones. Figure 8 displays the modified test image together with the results of `decorrstretch` and QR-SVD. The last image clearly exhibits the best enhancement. The difference in the colors of the two processed images is due to the fact that once the QR-SVD method detects a linearly dependent color plane, it sets the corresponding pixels to zero in the result. The consequence is that the processed image contains only red and blue pixels. The same experiments have been performed on the PIMENTEL image. Results are displayed in Figure 9.

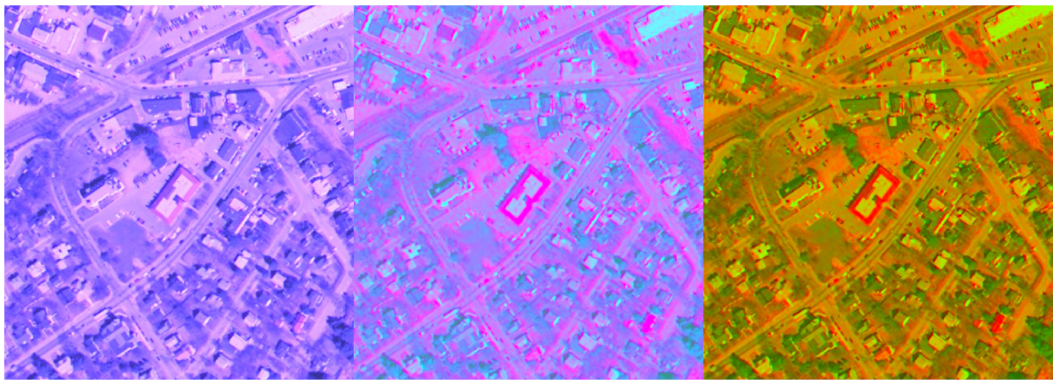

Figure 8. From (**left**) to (**right**): the test image WEST modified so that the blue color plane is linearly dependent on the other two planes, results of `decorrstretch` and QR-SVD.

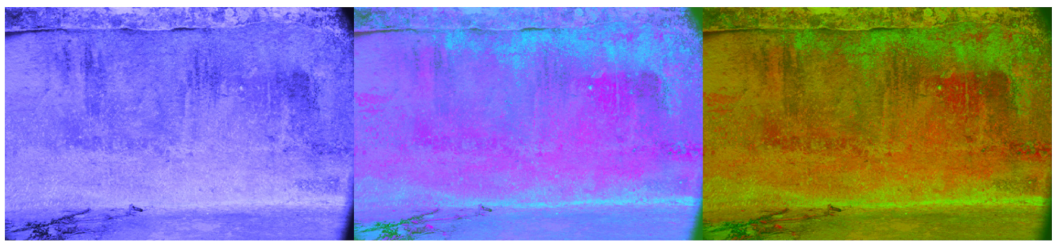

Figure 9. From (**left**) to (**right**): the test image PIMENTEL modified so that the blue color plane is linearly dependent on the other two planes, results of `decorrstretch` and QR-SVD.

To illustrate the effect of DS on a multispectral image, we processed the dataset PARIS. It contains a seven-band 512-by-512 Landsat image. Figure 10 shows the results obtained by the QR-SVD algorithm. We visualize bands $(1, 2, 3)$ (left column), $(3, 4, 5)$ (center column), and $(5, 6, 7)$ (right column). The top row reports the unprocessed color planes; the bottom row reports the results obtained by DS. To correctly display images, we adjusted them using the MATLAB function `imadjust`.

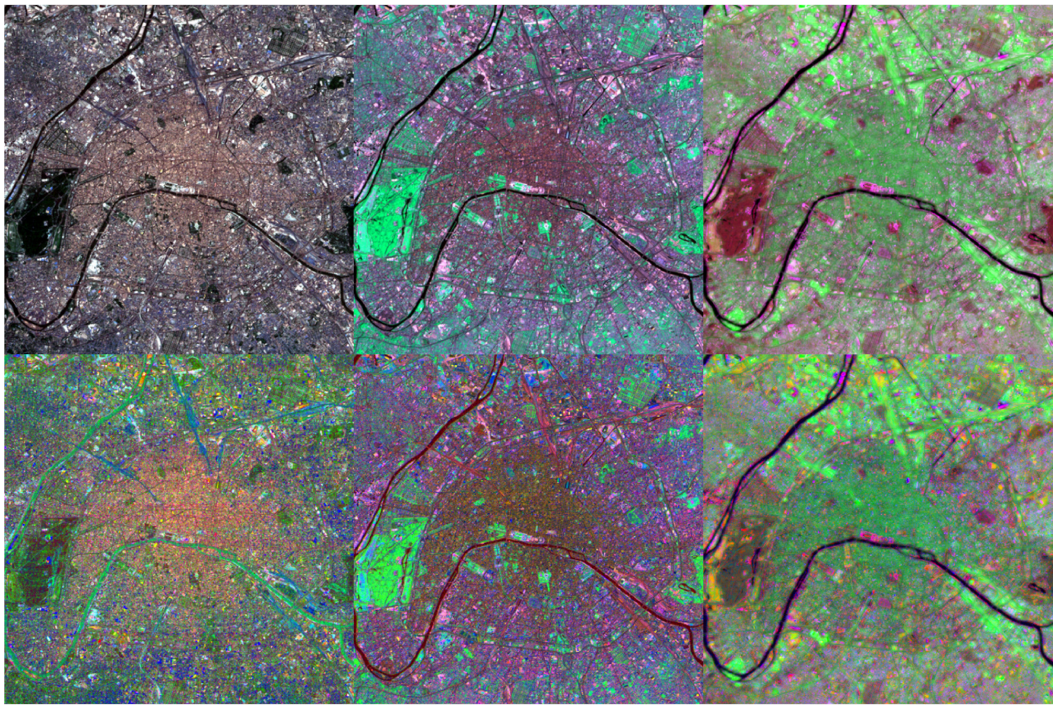

Figure 10. Results for multispectral image PARIS. From (**left**) to (**right**): color bands $(1,2,3)$, $(3,4,5)$, and $(5,6,7)$. The (**top**) row displays original data; the (**bottom**) row shows results obtained by the QR-SVD algorithm.

6. Conclusions

Decorrelation stretch is an image enhancement technique based on Principal Component Analysis. It increases color differences by removing inter-channel correlations and scaling variances to let them assume target values. It is used in many applications, e.g., remote sensing and processing of archeological images, where it is important that subtle features become more visible. In this paper, we have presented some numerical methods for its application as alternatives to the original formulation. One of the algorithms can also deal with degenerate cases, producing images with uncorrelated color channels. The computing time can be reduced by coupling a random subsampling of the image to the above method.

Author Contributions: All authors contributed to the conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, and funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: F. Pes and G. Rodriguez are partially supported by European Union–Next Generation EU, Mission 4 Component 1, CUP F53D23002700006 through the PRIN 2022 project "IPIS–Inverse Problems in the Imaging Sciences". E. Crabu and G. Rodriguez acknowledge partial support from the PRIN–PNRR 2022 project "AQuAInt–Approximation and Quadrature for Applicative Integral Models" (P20229RMLB). All authors are members of the Gruppo Nazionale per il Calcolo Scientifico–Istituto Nazionale di Alta Matematica (GNCS–INdAM). E. Crabu and F. Pes are partially supported by the INdAM–GNCS project 2025 "Tecniche numeriche per problemi di grandi dimensioni" (CUP E53C24001950001) and G. Rodriguez by the INdAM–GNCS project 2025 "Metodi di approssimazione globale per operatori integrali e applicazioni alle equazioni funzionali" (CUP E53C24001950001). F. Pes gratefully acknowledges OGS and CINECA under HPC–TRES program award number 2024–02.

Data Availability Statement: The data presented in this study are available in the CaNA research group repository at https://bugs.unica.it/~cana/software (accessed on 8 September 2025) as a DEXTRE package. Other data were derived from the following resource available in the public domain: https://www.mathworks.com/products/matlab.html (accessed on 8 September 2025) (MATLAB documentation).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mather, P.M.; Koch, M. *Computer Processing of Remotely-Sensed Images*; John Wiley & Sons: Hoboken, NJ, USA, 2022.
2. Zhang, D.; Liu, Y.; Zhao, Y.; Liang, J.; Sun, B.; Chu, S. Algorithm research on detail and contrast enhancement of high dynamic infrared images. *Appl. Sci.* **2023**, *13*, 12649. [CrossRef]
3. Lu, H.; Liu, Z.; Lan, R.; Pan, X.; Gong, J. Retinex-inspired contrast stretch and detail boosting for lowlight image enhancement. *IET Image Process.* **2023**, *17*, 1718–1738. [CrossRef]
4. Chun, S.M.; Park, J.Y.; Eom, I.K. Low-Light Image Enhancement Network Using Informative Feature Stretch and Attention. *Electronics* **2024**, *13*, 3883. [CrossRef]
5. Gillespie, A.R.; Kahle, A.B.; Walker, R.E. Color enhancement of highly correlated images. I. Decorrelation and HSI contrast stretches. *Remote Sens. Environ.* **1986**, *20*, 209–235. [CrossRef]
6. Alley, R.E. *Algorithm Theoretical Basis Document for Decorrelation Stretch*; Technical Report; NASA, Jet Propulsion Laboratory: La Cañada Flintridge, CA, USA, 1996.
7. Campbell, N.A. The decorrelation stretch transformation. *Int. J. Remote Sens.* **1996**, *17*, 1939–1949. [CrossRef]
8. Rothery, D.A.; Hunt, G.A. A simple way to perform decorrelation stretching and related techniques on menu-driven image processing systems. *Int. J. Remote Sens.* **1990**, *11*, 133–137. [CrossRef]
9. Guo, L.J.; Moore, J.M. Direct decorrelation stretch technique for RGB colour composition. *Int. J. Remote Sens.* **1996**, *17*, 1005–1018. [CrossRef]
10. Ferrari, M.C. Improved decorrelation stretching of TM data for geological applications: First results in Northern Somalia. *Int. J. Remote Sens.* **1992**, *13*, 841–851. [CrossRef]
11. Rowan, L.C.; Mars, J.C. Lithologic mapping in the Mountain Pass, California area using advanced spaceborne thermal emission and reflection radiometer (ASTER) data. *Remote Sens. Environ.* **2003**, *84*, 350–366. [CrossRef]
12. White, K. Image processing of Thematic Mapper data for discriminating piedmont surficial materials in the Tunisian Southern Atlas. *Int. J. Remote Sens.* **1993**, *14*, 961–977. [CrossRef]
13. Le Quellec, J.L.; Duquesnoy, F.; Defrasne, C. Digital image enhancement with DStretch®: Is complexity always necessary for efficiency? *Digit. Appl. Archaeol. Cult. Herit.* **2015**, *2*, 55–67. [CrossRef]
14. Evans, L.; Mourad, A.L. DStretch® and Egyptian tomb paintings: A case study from Beni Hassan. *J. Archaeol. Sci. Rep.* **2018**, *18*, 78–84. [CrossRef]
15. Gunn, R.G.; Douglas, L.C.; Whear, R.L. Interpreting polychrome paintings using DStretch. *Rock Art Res.* **2014**, *31*, 101–104.
16. Le Quellec, J.; Harman, J.; Defrasne, C.; Duquesnoy, F. DStretch® et l'amélioration des images numériques: Applications à l'archéologie des images rupestres. *Les Cah. l'AARS* **2013**, *16*, 177–198.
17. Acevedo, A.; Franco, N.V. Aplicación de DStretch-ImageJ a imágenes digitales del arte rupestre de Patagonia (Argentina). *Comechingonia Virtual Rev. Electrón. Arqueol.* **2012**, *6*, 152–175.
18. Collado, F.J.M.; Ruiz, A.J.M.; del Toro, M.S.N. Aplicación del plugin DStretch para el programa ImageJ al estudio de las manifestaciones pictóricas del abrigo Riquelme (Murcia). *Cuad. Arte Rupestre* **2013**, *6*, 113–127.
19. González, E.R.; Pastor, S.C.; Casals, J.R. Lost colours: Photogrammetry, image analysis using the DStretch plugin, and 3-D modelling of post-firing painted pottery from the south west Iberian Peninsula. *Digit. Appl. Archaeol. Cult. Herit.* **2019**, *13*, e00093.
20. Quesada, E.; Harman, J. A step further in rock art digital enhancements. DStretch on Gigapixel imaging. *Digit. Appl. Archaeol. Cult. Herit.* **2019**, *13*, e00098. [CrossRef]
21. DiBiasie-Sammons, J.F. Visualizing Dipinti: Re-Reading a Painted Inscription from Herculaneum (CIL IV. 10479) Using Decorrelation Stretch and Archival Research. *Z. Papyrol. Epigr.* **2020**, *214*, 273–284.
22. Potter, R.; Pitman, D.; Manley, H.; Rönnlund, R. Cost-effective, rapid decorrelation stretching and responsive UAS mapping as a method of detecting archaeological sites and features. *Herit. Sci.* **2023**, *11*, 89. [CrossRef]
23. Sosa-Alonso, P.J. Image analysis and treatment for the detection of petroglyphs and their superimpositions: Rediscovering rock art in the Balos Ravine, Gran Canaria Island. *Rock Art Res. J. Aust. Rock Art Res. Assoc. (AURA)* **2023**, *40*, 121–130.

24. Kazutaka, K.; Masatoshi, I.; Haruhiro, F.; Ryo, Y.; Toshiki, T.; Haruhiko, O. Statistical Image Processing (Decorrelation Stretch) and Deep Learning (CycleGANs) to Restore Images of Faded Artworks. In Proceedings of the 51th International Conference of Computer Applications and Quantitative Methods in Archaeology (CAA2024), Auckland, New Zealand, 8–12 April 2024. [CrossRef]
25. Nagaoka, T.; Nakamura, A.; Aizawa, K.; Kanazawa, M.; Kezuka, T.; Miura, M.; Usui, M.; Ohtsubo, S.; Sota, T. Application of decorrelation stretching method to hyperspectral fundus image processing. In *Ophthalmic Technologies XVII, Proceedings of the SPIE BiOS, San Jose, CA, USA, 20–25 January 2007*; SPIE: Bellingham, WA, USA, 2007; Volume 6426, pp. 387–394.
26. Al-Surmi, A.; Wirza, R.; Mahmod, R.; Khalid, F.; Dimon, M.Z. A new human heart vessel identification, segmentation and 3D reconstruction mechanism. *J. Cardiothorac. Surg.* **2014**, *9*, 161. [CrossRef] [PubMed]
27. Peikari, M.; Martel, A.L. Automatic cell detection and segmentation from H and E stained pathology slides using colorspace decorrelation stretching. In *Medical Imaging 2016: Digital Pathology, Proceedings of the SPIE Medical Imaging, San Diego, CA, USA, 27 February–3 March 2016*; SPIE: Bellingham, WA, USA, 2016; Volume 9791, pp. 292–297.
28. Giri Babu, K.; Bhavana, J. Brain Tumor Recognition by Machine Learning with Decorrelation Stretch Image Enhancement Based on MRI Images. In *Proceedings of the Second Congress on Control, Robotics, and Mechatronics*; Springer: Singapore, 2024; pp. 161–174.
29. Björck, Å. *Numerical Methods for Least Squares Problems*, 2nd ed.; SIAM: Philadelphia, PA, USA, 2024.
30. Golub, G.H.; Van Loan, C.F. *Matrix Computations*, 3rd ed.; The John Hopkins University Press: Baltimore, MD, USA, 1996.