Inverse Problems and Imaging
Volume 16, No. 2, 2022, 341–366
doi:10.3934/ipi.2021053

PCA REDUCED GAUSSIAN MIXTURE MODELS WITH APPLICATIONS IN SUPERRESOLUTION

Johannes Hertrich*
TU Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany

Dang-Phuong-Lan Nguyen
Univ. Bordeaux, Bordeaux INP, CNRS, IMB, UMR 5251, F-33400 Talence, France
Univ. Bordeaux, Bordeaux INP, CNRS, IMS, UMR 5218, F-33400 Talence, France

Jean-Francois Aujol
Univ. Bordeaux, Bordeaux INP, CNRS, IMB, UMR 5251, F-33400 Talence, France

Dominique Bernard
CNRS, Univ. Bordeaux, Bordeaux INP, ICMCB, UMR 5026, F-33600 Pessac, France

Yannick Berthoumieu
Univ. Bordeaux, Bordeaux INP, CNRS, IMS, UMR 5218, F-33400 Talence, France

Abdellatif Saadaldin
CNRS, Univ. Bordeaux, Bordeaux INP, ICMCB, UMR 5026, F-33600 Pessac, France

Gabriele Steidl
TU Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany

(Communicated by Habib Ammari)

Abstract. Despite the rapid development of computational hardware, the treatment of large and high dimensional data sets is still a challenging problem. The contribution of this paper to the topic is twofold. First, we propose a Gaussian mixture model in conjunction with a reduction of the dimensionality of the data in each component of the model by principal component analysis, which we call PCA-GMM. To learn the (low dimensional) parameters of the mixture model we propose an EM algorithm whose M-step requires the solution of constrained optimization problems. Fortunately, these constrained problems do not depend on the usually large number of samples and can be solved efficiently by an (inertial) proximal alternating linearized minimization algorithm. Second, we apply our PCA-GMM for the superresolution of 2D and 3D material images based on the approach of Sandeep and Jacob. Numerical results confirm the moderate influence of the dimensionality reduction on the overall superresolution result.

2020 *Mathematics Subject Classification.* Primary: 62H12, 62H25, 62H35; Secondary: 65K05.
Key words and phrases. Gaussian mixture models, expectation maximization algorithm, dimensionality reduction, principal component analysis, maximum likelihood estimation, superresolution.

* Corresponding author: Johannes Hertrich.

1. **Introduction.** The motivation for this work was superresolution of 3D material images taken within the project ITN MUMMERING. Superresolution aims at reconstructing high resolution images from low resolution ones. Here a common assumption is that the low resolution image is generated by $y = Ax + \epsilon$, where ϵ is some noise and A is a possibly unknown superresolution operator. Since this is an ill-posed inverse problem, methods addressing this task usually incorporate some prior information. One approach for solving the problem is based on Gaussian Mixture Models (GMMs). Usually, the GMM approximates the distribution of the patches of natural images and its parameters are learned from some given data, see also [7].

In literature, there exist several approaches to tackle superresolution by GMMs. Zoran and Weiss [32] proposed to use the negative log-likelihood function of a GMM as regularizer of the inverse problem by estimating the high resolution image x_H given the low resolution one x_L by solving

$$\operatorname*{argmin}_{x_H} \|Ax_H - x_L\|^2 - \lambda \sum_{i \in I} \log p(x_{H,i}),$$

where p is the probability density function of the GMM and $(x_{H,i})_{i \in I}$ are the patches in x_H. For a special choice of λ, the solution of this problem can be interpreted as the maximum a postiori (MAP) estimator of x_H under the prior assumption that the distribution of the patches in x_H is given by the GMM. This method is called expected patch log likelihood (EPLL) and it can be applied for several inverse problems. Various accelerations and an efficient implementation of EPLL were developed in [23]. However, EPLL requires that the operator A is known, which is usually not the case for the superresolution task. Therefore, we prefer the alternative approach of Sandeep and Jacob [26], which does not require any knowledge about the operator A. While for EPLL the GMM describes only the distribution of the patches from the high resolution image, in [26] the idea is to use a joint GMM, which describes the distribution of pairs of high and low resolution patches. Having learned a joint GMM, each high resolution patch is estimated separately from the low resolution patch and the joint GMM using the minimal mean squared error estimator. For a more detailed description of the method proposed by Sandeep and Jacob [26], we refer to Section 5.

However, any of these models requires the estimation of the parameters of a GMM using the patches of the images as data points. For this, the maximum likelihood (ML) estimator is used, which corresponds to minimizing the negative log likelihood function. The standard method to find the ML estimator is the expectation maximization (EM) algorithm [4, 8]. Unfortunately, the EM algorithm for GMMs becomes very slow as the number of data points becomes large and high dimensional, which is the case for our superresolution task, particularly if we have to deal with 3D images.

To overcome these performance issues, we reduce the dimension of the data points. The standard method for dimensionality reduction is the principal component analysis (PCA) [24]. The main assumption of the PCA is that the high dimensional data points are approximately located in a lower dimensional affine subspace. In this paper, we combine the GMM with a PCA by adding the minimization term of the PCA and the negative log likelihood function of the GMM on the dimensionality reduced data points. We rewrite this minimization problem again as the negative log likelihood function of a Gaussian mixture model which

has additional constraints on the parameters. We call this model PCA-GMM. This representation allows in particular, to use a different PCA for each component of the GMM. We derive an EM algorithm with a special M-step for finding a minimizer of our objective function. The M-step requires solutions of maximization problems with contraints on the Stiefel manifold. Fortunately, these problems do no longer depend on the large number of sampling points and they can be efficiently solved by the (inertial) proximal alternating linearized minimization algorithm (PALM) for which some convergence results can be ensured. The idea to couple parameter learning with dimension reduction is not new. So the authors of [3, 14] propose directly a GMM with constraint covariance matrices. It is based on an extension of the PCA, which was proposed in [28] to replace the affine space in the PCA by the union of finitely many affine spaces using a mixture model of probabilistic PCAs. The relation to our approach is analyzed in a remark in Section 3. Using our new PCA-GMM model within the superresolution model of Sandeep and Jacob [26], we provide numerical examples of 2D and 3D material images. For making the examples reproducible, we provide the code online[1].

The paper is organized as follows: in Section 2 we briefly review the two main ingredients for our model, namely GMMs and PCA. We derive our PCA-GMM model in Section 3. In Section 4, an EM algorithm with a special constrained optimization task in the M-step is proposed for minimizing the objective function. The solution of the constrained minimization problem via (inertial) PALM is investigated. The superresolution method using the PCA-GMM model is described in Section 5. Finally, Section 6 shows numerical examples of superresolution based on our PCA-GMM model on 2D and 3D images. Conclusions are drawn in Section 7.

2. Preliminaries.

In this section, we briefly revisit the two building blocks of our approach, namely Gaussian mixture models and principal component analysis. We need the following notation. By $\mathrm{SPD}(n) \subset \mathbb{R}^{n,n}$ we denote the cone of *symmetric positive definite* $n \times n$ *matrices*, by $O(n)$ the group of orthogonal $n \times n$ matrices, by

$$\mathrm{St}(d,n) \coloneqq \left\{U \in \mathbb{R}^{n,d} : U^{\mathrm{T}}U = I_d\right\}, \quad n \geq d,$$

the *Stiefel manifold* and by $\Delta_K \coloneqq \{\alpha = (\alpha_k)_{k=1}^K \in \mathbb{R}^K_{\geq 0} : \sum_{k=1}^K \alpha_k = 1\}$ the *probability simplex*. We write 1_n for the vector with all n components equal to 1. Further, we denote by $\|\cdot\|_F$ the Frobenius norm.

Gaussian Mixture Models. The (absolutely continuous) *n-dimensional normal distribution* $\mathcal{N}(\mu, \Sigma)$ with mean $\mu \in \mathbb{R}^n$ and positive semi-definite covariance matrix $\Sigma \in \mathrm{SPD}(n)$ has the density

$$f(x|\mu,\Sigma) = (2\pi)^{-\frac{n}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^{\mathrm{T}}\Sigma^{-1}(x-\mu)\right). \tag{1}$$

Note, that not all multivariate normal distributions are absolutely continuous, in particular, the covariance matrix is not necessarily invertible. However, for the rest of the paper, we focus on normal distributions with positive definite covariance matrices, which are invertible. A *Gaussian mixture model* (GMM) is a probability distribution with probability density function

$$p(x) = \sum_{k=1}^K \alpha_k f(x|\mu_k, \Sigma_k), \qquad \alpha \in \Delta_K.$$

[1] https://github.com/johertrich/PCA_GMMs

For samples $\mathcal{X} = \{x_1, ..., x_N\}$, the maximum likelihood (ML) estimator of the parameters $\alpha = (\alpha_k)_{k=1}^K$, $\mu = (\mu_k)_{k=1}^K$ and $\mathbf{\Sigma} = (\Sigma_k)_{k=1}^K$ of a GMM can be found by minimizing its *negative log-likelihood function*

$$L(\alpha, \mu, \mathbf{\Sigma}|\mathcal{X}) = -\sum_{i=1}^N \log\Big(\sum_{k=1}^K \alpha_k f(x_i|\mu_k, \Sigma_k)\Big)$$

for $\alpha \in \triangle_K$, $\mu_k \in \mathbb{R}^n$, and $\Sigma_k \in \mathrm{SPD}(n)$, $k = 1, \dots, K$.

In the following, we use the notation $\vartheta := (\mu, \Sigma)$ to address the parameters of a Gaussian distribution. A standard minimization algorithm for finding the ML estimator of the parameters α_k and $\vartheta_k = (\mu_k, \Sigma_k)$, $k = 1, \dots, K$ of a GMM is the so-called *EM algorithm* [4, 8] detailed in Alg. 1.

Algorithm 1 EM Algorithm for Mixture Models

Input: $x = (x_1, ..., x_N) \in \mathbb{R}^{n \times N}$, initial estimate $\vartheta^{(0)}$.
for $r = 0, 1, ...$ **do**
E-Step: For $k = 1, ..., K$ and $i = 1, \dots, N$ compute

$$\beta_{i,k}^{(r)} = \frac{\alpha_k^{(r)} f(x_i|\vartheta_k^{(r)})}{\sum_{j=1}^K \alpha_j^{(r)} f(x_i|\vartheta_j^{(r)})}$$

M-Step: For $k = 1, ..., K$ compute

$$\alpha_k^{(r+1)} = \frac{1}{N}\sum_{i=1}^N \beta_{i,k}^{(r)},$$

$$\vartheta_k^{(r+1)} = \operatorname*{argmax}_{\vartheta_k} \Big\{\sum_{i=1}^N \beta_{i,k}^{(r)} \log(f(x_i|\vartheta_k))\Big\}.$$

end for

For Gaussian density functions (1), the iterates $\vartheta_k^{(r+1)}$, $k = 1, \dots, K$, i.e. the maximization in the M-Step of Alg. 1 can be simply computed by

(2)
$$\mu_k^{(r+1)} = \frac{\sum_{i=1}^N \beta_{ik}^{(r)} x_i}{\sum_{i=1}^N \beta_{ik}^{(r)}} = \frac{1}{N\alpha_k^{(r+1)}} m_k^{(r)},$$

(3)
$$\Sigma_k^{(r+1)} = \frac{\sum_{i=1}^N \beta_{ik}^{(r)} (x_i - \mu_k^{(r+1)})(x_i - \mu_k^{(r+1)})^{\mathrm{T}}}{\sum_{i=1}^N \beta_{ik}^{(r)}} = \frac{1}{N\alpha_k^{(r+1)}} C_k^{(r)} - \mu_k^{(r+1)}(\mu_k^{(r+1)})^{\mathrm{T}},$$

where

$$m_k^{(r)} = \sum_{i=1}^N \beta_{ik}^{(r)} x_i \quad \text{and} \quad C_k^{(r)} = \sum_{i=1}^N \beta_{ik}^{(r)} x_i x_i^{\mathrm{T}}.$$

Principal Component Analysis. In many applications, the dimension of the data is huge such that dimensionality reduction methods become necessary. The working horse for dimensionality reduction is the principal component analysis (PCA). Given data samples $\mathcal{X} = \{x_1, ..., x_N\}$ in $\mathbb{R}^n$, the classical PCA finds the d-dimensional affine space $\{U\,t + b : t \in \mathbb{R}^d\}$, $1 \le d \ll n$ having smallest squared

distance from the samples by minimizing

$$P(U,b)=\sum_{i=1}^{N}\|(UU^{\mathrm{T}}-I_n)(x_i-b)\|^2$$

for $b\in\mathbb{R}^n$ and $U\in\mathrm{St}(d,n)$. It is not hard to check that the affine subspace goes through the *offset* (*bias*) $b=\bar{x}:=\frac{1}{N}(x_1+\ldots+x_N)$ so that we can reduce our attention to the minimization with respect to $U\in\mathrm{St}(d,n)$, i.e., to the consideration of

$$P(U)=\sum_{i=1}^{N}\|(UU^{\mathrm{T}}-I_n)y_i)\|^2,\quad y_i=x_i-\bar{x}.$$

Note, that a minimizer can be derived explicitly as the matrix $\hat{U}$, whose columns are given by the eigenvectors corresponding to the d largest eigenvalues of the empirical covariance matrix $\sum_{i=1}^{N}y_iy_i^{\mathrm{T}}$. This minimizer is not unique, since it holds $P(UV)=P(U)$ for any orthogonal matrix $V\in O(d)$.

3. **PCA-GMM model.** In this section, we propose a GMM which incorporates a dimensionality reduction model via PCA. More precisely, we want to consider Gaussian distributions only on smaller subspaces of the original data space.

A first idea would be to couple the GMM and the PCA model in an additive way and to minimize for data samples $\mathcal{X}=\{x_1,...,x_N\}$ in $\mathbb{R}^n$ the function

$$F(U,\alpha,\vartheta)=L\big(\alpha,\vartheta|\mathcal{X}_{\mathrm{low}}\big)+\frac{1}{2\sigma^2}P(U),\quad \sigma>0 \tag{4}$$

for $U\in\mathrm{St}(d,n)$, $\alpha\in\triangle_K$, $\mu_k\in\mathbb{R}^d$, and $\Sigma_k\in\mathrm{SPD}(d)$, $k=1,\ldots,K$, where

$$\mathcal{X}_{\mathrm{low}}:=\{U^{\mathrm{T}}y_1,\ldots,U^{\mathrm{T}}y_N\},\quad y_i=x_i-\bar{x}.$$

It is important that the negative log-likelihood function L acts with respect to ϑ only on the lower dimensional space $\mathbb{R}^d$. The function F can be rewritten as

$$\begin{aligned}F(U,\alpha,\vartheta)&=-\sum_{i=1}^{N}\left(\log\Big(\sum_{k=1}^{K}\alpha_k f\big(U^{\mathrm{T}}y_i|\vartheta_k\big)\Big)-\frac{1}{2\sigma^2}\|(UU^{\mathrm{T}}-I_n)y_i)\|^2\right)\\&=-\sum_{i=1}^{N}\left(\log\Big(\sum_{k=1}^{K}\alpha_k f\big(U^{\mathrm{T}}y_i|\vartheta_k\big)\,\exp\big(-\tfrac{1}{2\sigma^2}\|(UU^{\mathrm{T}}-I_n)y_i)\|^2\big)\Big)\right).\end{aligned} \tag{5}$$

However, knowing that the samples were taken from K different Gaussian distributions it makes more sense to reduce the dimension individually for each distribution. Based on the reformulation (5) and using the notation $\mathbf{U}=(U_k)_{k=1}^K$ and $\mathbf{b}=(b_k)_{k=1}^K$, we propose to minimize the following **PCA-GMM model**:

$$F(\mathbf{U},\mathbf{b},\alpha,\vartheta)\text{ subject to }\alpha\in\Delta_K,\,U_k\in\mathrm{St}(d,n),\Sigma_k\in\mathrm{SPD}(d),\;k=1,\ldots,K, \tag{6}$$

where $b_k\in\mathbb{R}^n$, $\mu_k\in\mathbb{R}^d$ and

$$\begin{aligned}F(\mathbf{U},\mathbf{b},\alpha,\vartheta)&:=-\sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\alpha_k f(U_k^{\mathrm{T}}y_{ik}|\vartheta_k)\,\exp\big(-\tfrac{1}{2\sigma^2}\|(I_n-U_kU_k^{\mathrm{T}})y_{ik}\|^2\big)\right),\\y_{ik}&:=x_i-b_k,\quad k=1,\ldots,K,\,i=1,\ldots,N.\end{aligned} \tag{7}$$

Clearly, if $U_k=U$ and $b_k=\bar{x}$ for all $k=1,\ldots,K$, we get back to model (4).

The next lemma shows that our PCA-GMM model can be rewritten as a GMM model whose parameters incorporate those of the PCA.

Lemma 3.1. *Let $\mu \in \mathbb{R}^d$, $\Sigma \in \mathrm{SPD}(d)$, $U \in \mathrm{St}(d,n)$, $b \in \mathbb{R}^n$ and let f be the Gaussian density function (1). Then the following relation holds true:*

$$f\big(U^{\mathrm{T}}(x-b)|\mu,\Sigma\big)\,\exp\Big(-\tfrac{1}{2\sigma^2}\|(I_n-UU^{\mathrm{T}})(x-b)\|^2\Big) = (2\pi\sigma^2)^{\frac{n-d}{2}} f(x|\tilde{\mu},\tilde{\Sigma}), \tag{8}$$

where

$$\tilde{\Sigma} = \big(\tfrac{1}{\sigma^2}(I_n - UU^{\mathrm{T}}) + U\Sigma^{-1}U^{\mathrm{T}}\big)^{-1} \in \mathrm{SPD}(n), \tag{9}$$

$$\tilde{\mu} = \tilde{\Sigma}U\Sigma^{-1}\mu + b \in \mathbb{R}^n. \tag{10}$$

Proof. 1. First of all, we verify that the matrices $\tilde{\Sigma}$ are well defined, i.e. that $\frac{1}{\sigma^2}(I_n - UU^{\mathrm{T}}) + U(\Sigma)^{-1}U^{\mathrm{T}}$ is invertible. Let $\tilde{U} \in \mathbb{R}^{n,(n-d)}$ such that $V := (U|\tilde{U})$ is an orthogonal matrix. Then we obtain

$$\begin{aligned} V^{\mathrm{T}}\tilde{\Sigma}^{-1}V &= V^{\mathrm{T}}(\tfrac{1}{\sigma^2}(I_n - UU^{\mathrm{T}}) + U\Sigma^{-1}U^{\mathrm{T}})V \\ &= \tfrac{1}{\sigma^2}(I_n - V^{\mathrm{T}}UU^{\mathrm{T}}V) + V^{\mathrm{T}}U\Sigma^{-1}U^{\mathrm{T}}V. \end{aligned}$$

Since $(V^{\mathrm{T}}U)^{\mathrm{T}} = U^{\mathrm{T}}V = (I_d|0)$, this is equal to

$$V^{\mathrm{T}}\tilde{\Sigma}^{-1}V = \tfrac{1}{\sigma^2}\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & I_{n-d} \end{array}\right) + \left(\begin{array}{c|c} \Sigma^{-1} & 0 \\ \hline 0 & 0 \end{array}\right) = \left(\begin{array}{c|c} \Sigma^{-1} & 0 \\ \hline 0 & \frac{1}{\sigma^2} I_{n-d} \end{array}\right) \tag{11}$$

and the last matrix is invertible.

2. We have to show that

$$\begin{aligned} &(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\exp\Big(-\frac{1}{2\sigma^2}\|(I_n-UU^{\mathrm{T}})(x-b)\|^2 \\ &\quad -\frac{1}{2}(U^{\mathrm{T}}(x-b)-\mu)^{\mathrm{T}}\Sigma^{-1}(U^{\mathrm{T}}(x-b)-\mu)\Big) \\ &= (2\pi)^{-\frac{n}{2}}|\tilde{\Sigma}|^{-\frac{1}{2}}\exp\Big(-\frac{1}{2}(x-\tilde{\mu})^{\mathrm{T}}\tilde{\Sigma}^{-1}(x-\tilde{\mu})\Big) \\ &= (2\pi)^{-\frac{n}{2}}|\tilde{\Sigma}|^{-\frac{1}{2}}\exp\Big(-\frac{1}{2}x^{\mathrm{T}}\tilde{\Sigma}^{-1}x + \tilde{\mu}^{\mathrm{T}}\tilde{\Sigma}^{-1}x - \frac{1}{2}\tilde{\mu}^{\mathrm{T}}\tilde{\Sigma}^{-1}\tilde{\mu}\Big). \end{aligned}$$

Straightforward calculation together with the observation that $U^{\mathrm{T}}\tilde{\Sigma}U = \Sigma$ and hence $U^{\mathrm{T}}\tilde{\Sigma}^{-1}U = \Sigma^{-1}$ gives

$$\begin{aligned} &\frac{1}{2\sigma^2}\|(I_n-UU^{\mathrm{T}})(x-b)\|^2 + \frac{1}{2}(U^{\mathrm{T}}(x-b)-\mu)^{\mathrm{T}}\Sigma^{-1}(U^{\mathrm{T}}(x-b)-\mu) \\ &= \frac{1}{2}x^{\mathrm{T}}\left(\tfrac{1}{\sigma^2}(I_n-UU^{\mathrm{T}}) + U\Sigma^{-1}U^{\mathrm{T}}\right)x \\ &\quad - \left(\tfrac{1}{\sigma^2}b^{\mathrm{T}}(I_n-UU^{\mathrm{T}}) + (\mu^{\mathrm{T}}+b^{\mathrm{T}}U)\Sigma^{-1}U^{\mathrm{T}}\right)x \\ &\quad + \frac{1}{2}(U^{\mathrm{T}}b+\mu)^{\mathrm{T}}\Sigma^{-1}(U^{\mathrm{T}}b+\mu) + \tfrac{1}{2\sigma^2}b^{\mathrm{T}}(I_n-UU^{\mathrm{T}})b \\ &= \frac{1}{2}x^{\mathrm{T}}\tilde{\Sigma}^{-1}x - \tilde{\mu}^{\mathrm{T}}\tilde{\Sigma}^{-1}x + \frac{1}{2}\tilde{\mu}^{\mathrm{T}}\tilde{\Sigma}^{-1}\tilde{\mu}. \end{aligned}$$

Finally, we see by (11) that $|\tilde{\Sigma}|^{-1} = \sigma^{-2(n-d)}|\Sigma|^{-1}$. □

By Lemma 3.1, we can rewrite our objective function F in (7) with $\tilde{\vartheta} = (\tilde{\mu},\tilde{\Sigma})$ defined by (9) and (10) with corresponding indices as

$$\begin{aligned} F(\mathbf{U},\mathbf{b},\alpha,\vartheta) &= -\sum_{i=1}^{N}\log\Big(\sum_{k=1}^{K}\alpha_k f(x_i|\tilde{\vartheta}_k)\Big) + (n-d)\log(\sqrt{2\pi\sigma^2}) \\ &= L(\alpha,\tilde{\vartheta}|\mathcal{X}) + (n-d)\log(\sqrt{2\pi\sigma^2}). \end{aligned} \tag{12}$$

Up to the constant this is a negative log-likelihood function of a GMM. However, when minimizing this function, we have to take the constraints (9) and (10) into account. More precisely, our model in (6) can be rewritten as **PCA-GMM model**:

$$\text{(13)}\quad \mathrm{F}(\mathbf{U},\mathbf{b},\alpha,\vartheta) \coloneqq L(\alpha,\tilde{\vartheta}|\mathcal{X}) \text{ subject to } U_k \in \mathrm{St}(d,n),\ \alpha \in \Delta_K,\ \Sigma_k \in \mathrm{SPD}(d),$$

where

$$\text{(14)}\quad \tilde{\Sigma}_k = \left(\tfrac{1}{\sigma^2}(I_n - U_k U_k^{\mathrm{T}}) + U_k \Sigma_k^{-1} U_k^{\mathrm{T}}\right)^{-1},\ \tilde{\mu}_k = \tilde{\Sigma}_k U \Sigma_k^{-1} \mu_k + b_k, \quad k = 1,\dots,K.$$

The choice of μ_k and b_k is redundant. This can be seen as follows, for any μ_k and b_k, define $\hat{\mu}_k = 0$ and $\hat{b}_k = \tilde{\mu}_k$. Then, it holds

$$\tilde{\Sigma}_k = \left(\tfrac{1}{\sigma^2}(I_n - U_k U_k^{\mathrm{T}}) + U_k \Sigma_k^{-1} U_k^{\mathrm{T}}\right)^{-1},\ \tilde{\mu}_k = \tilde{\Sigma}_k U \Sigma_k^{-1} \hat{\mu}_k + \hat{b}_k, \quad k = 1,\dots,K$$

such that $\mathrm{F}(\mathbf{U},\hat{\mathbf{b}},\alpha,\hat{\vartheta}) = \mathrm{F}(\mathbf{U},\mathbf{b},\alpha,\vartheta)$. Consequently, in the M-step of the EM algorithm in Section 4.1 we obtain that the update for the mean μ is given by $\mu_k = 0$.

Remark 1 (Different component dimensions)**.** So far the dimension d is the same for all components $k = 1,...,K$. But by some simple adjustments, the PCA-GMM model can also be rewritten with $U_k \in \mathrm{St}(d_k,n)$, $\mu_k \in \mathbb{R}^{d_k}$ and $\Sigma_k \in \mathrm{SPD}(d_k)$, where the d_k are not necessarily equal for all k. However, to keep the notations as simple as possible, we will restrict our analysis to the case that $d_k = d$ for $k = 1,...,K$. Nevertheless, all the results of this paper can be derived analogously for other choices of d_k.

Remark 2 (Learning σ)**.** The function F in (4), resp. (7), (12) is strictly decreasing in σ. Thus it does not make sense to minimize F with respect to σ.

However, the function $\mathrm{F} = F - \frac{n-d}{2}\log(2\pi\sigma^2)$ in (13) can be optimized with respect to σ. To keep the M-step of the EM algorithm simple, we associate to each summand in the mixture model an own σ_k, $k = 1,\dots,K$ such that $\tilde{\Sigma}$ in (14) becomes

$$\tilde{\Sigma}_k = \left(\tfrac{1}{\sigma_k^2}(I_n - U_k U_k^{\mathrm{T}}) + U_k \Sigma_k^{-1} U_k^{\mathrm{T}}\right)^{-1}.$$

In this case, we use the notation $\sigma = (\sigma_k)_{k=1}^K$.

Related Work. There are several relations of the PCA-GMM model to other models proposed in the literature, in particular to mixtures of probabilistic PCAs (MPPCA) [28], high dimensional data clustering (HDDC) [3] and high-dimensional mixture models for unsupervised image denoising (HDMI) [14]. In the following, we shortly review these methods and comment on similarities and differences to the PCA-GMM model.

For understanding the relation to other models, we first need the following reformulation of the covariance matrices $\tilde{\Sigma}$ from the PCA-GMM model. We have as in (11) for matrices $\tilde{\Sigma}$ of the form (9) and an orthogonal matrix $V = (U|\tilde{U})$ that

$$V^{\mathrm{T}} \tilde{\Sigma} V = \left(\begin{array}{c|c} \Sigma & 0 \\ \hline 0 & \sigma^2 I_{n-d} \end{array}\right),$$

so that

$$\begin{aligned}&\left\{\tilde{\Sigma} = \left(\tfrac{1}{\sigma^2}(I_n - UU^{\mathrm{T}}) + U\Sigma^{-1}U^{\mathrm{T}}\right)^{-1} : U \in \mathrm{St}(d,n),\, \Sigma \in \mathrm{SPD}(d)\right\}\\ &= \left\{Q^{\mathrm{T}}\left(\begin{array}{c|c}\mathrm{diag}(\lambda) & 0\\ \hline 0 & \sigma^2 I_{n-d}\end{array}\right)Q : Q \in O(n),\, \lambda \in \mathbb{R}^d_{>0}\right\}. \end{aligned} \tag{15}$$

As outlined in Remark 2, the σ can either be fixed a priori, or optimized within the EM algorithm, as later outlined in Section 4.1, simultaneously with the other parameters.

In [28], Tipping and Bishop propose mixture models of probabilistic PCAs (MPPCA), which are GMMs of the form

$$p(x) = \sum_{k=1}^{K} \alpha_k f(x_i|\tilde{\mu}_k, \tilde{\Sigma}_k), \tag{16}$$

where

$$\tilde{\Sigma}_k = U_k U_k^{\mathrm{T}} + \sigma_k^2 I_n, \quad U_k \in \mathrm{St}(d_k, n).$$

Here, the parameters σ_k are optimized simultaneously with the α_k and U_k via the EM algorithm. Hence, skipping the index, instead of minimizing over (15), they minimize over sets of the form

$$\left\{Q^{\mathrm{T}}\left(\begin{array}{c|c}(1+\sigma^2)I_d & 0\\ \hline 0 & \sigma^2 I_{n-d}\end{array}\right)Q : Q \in O(n)\right\}. \tag{17}$$

Since this form of the covariance matrices is very restrictive, Bouveyron, Girard and Schmid generalized MPPCA in [3] to a model called high dimensional data clustering (HDDC). Again, they minimize a special GMM (16), but here the covariances are given by

$$\tilde{\Sigma}_k = U_k \mathrm{diag}(\lambda_k) U_k^{\mathrm{T}} + \sigma_k^2 I_n, \quad U_k \in \mathrm{St}(d_k, n), \lambda_k \in \mathbb{R}^{d_k}_{>0}.$$

As for the MPPCA, the parameters are optimized via the EM algorithm. For deriving it, it is important that the parameters σ_k are not fixed a priori but are optimized within the EM algorithm. Skipping the index, instead of minimizing over (15) or (17), this corresponds to a minimization over

$$\left\{Q^{\mathrm{T}}\left(\begin{array}{c|c}\mathrm{diag}(\lambda) + \sigma^2 I_d & 0\\ \hline 0 & \sigma^2 I_{n-d}\end{array}\right)Q : Q \in O(n), \lambda \in \mathbb{R}^d_{>0}\right\}. \tag{18}$$

In contrast to (15), where the diagonal values λ are required to be strictly greater than 0, the diagonal values $\lambda + \sigma^2$ in (18) are automatically strictly greater than σ^2. Consequently, the PCA-GMM model is more general than HDDC. Note that HDDC model contains the so-called mixture factor analysis [20] as a special case. Here also the alternating expectation conditional maximization algorithm [21] is applicable [31], which is an improved version of the EM algorithm.

Finally, Houdard, Bouveyron and Delon proposed in [14] a model selection algorithm for the dimensions d_k. For this, they propose a model called HDMI, where the only difference to HDDC is, that σ is a priori fixed. They derive as an intermediate step a corresponding EM algorithm in [14, Proposition 2]. Unfortunately, the M-step only ensures that $\lambda > -\sigma^2 1_d$ and not $\lambda > 0$, such that the calculations appear to be not fully correct. However, the final model selection algorithm again ensures that $\lambda > 0$ such that this seems not to be a problem in [14].

4. **Minimization algorithm.** We propose to minimize F in (13) based on the EM algorithm, where we have to take the special structure of $\tilde{\mu}_k \in \mathbb{R}^n$ and $\tilde{\Sigma}_k \in \mathrm{SPD}(n)$ in (14) into account to work indeed in the lower d-dimensional space. This requires the solution of a special inner minimization problem within the M-Step of the EM algorithm. We describe the EM algorithm for our PCA-GMM model in Subsection 4.1. In particular, we will see that the M-Step of the algorithm requires the minimization of functions $G_k(U,b)$, $k=1,\dots,K$, of the same structure. We prove that these functions have indeed a global minimizer. In particular, these functions do not depend on the large number of input data x_i, $i=1,\dots,N$. Therefore it turns out that the E-step of the algorithm is the most time consuming one. We propose to find at least a local minimizer of G by the (inertial) proximal alternating linearized minimization (PALM) in Subsection 4.2 and provide convergence results.

4.1. **EM Algorithm for PCA-GMM.** For our setting, we obtain a special EM algorithm described in Algorithm 2. Note that the E-Step of Algorithm 2 requires only the mean and covariance matrix in $\vartheta_k^{(r)}$, $k=1,\dots,K$ with respect to the smaller space $\mathbb{R}^d$.

Algorithm 2 EM Algorithm for PCA reduced Mixture Models

Input: $\mathcal{X}=(x_1,...,x_N)\in\mathbb{R}^{n,N}$, initialization $\mathbf{U}^{(0)}$, $\mathbf{b}^{(0)}$, $\alpha^{(0)}$, $\vartheta^{(0)}=(\mu^{(0)},\mathbf{\Sigma}^{(0)})$.
for $r=0,1,...$ **do**

E-Step: For $k=1,...,K$ and $i=1,\dots,N$ compute

$$\beta_{i,k}^{(r)} = \frac{\alpha_k^{(r)} f(x_i|\tilde{\vartheta}_k^{(r)})}{\sum_{j=1}^K \alpha_j^{(r)} f(x_i|\tilde{\vartheta}_j^{(r)})}$$
$$= \frac{\frac{\alpha_k^{(r)}}{(\sigma_k^{(r)})^{n-d}} \exp\left(-\frac{1}{2(\sigma_k^{(r)})^2}\|(I_n - U_k^{(r)}(U_k^{(r)})^{\mathrm{T}})y_{i,k}^{(r)}\|^2\right) f\left((U_k^{(r)})^{\mathrm{T}} y_{i,k}^{(r)}|\vartheta_k^{(r)}\right)}{\sum_{j=1}^K \frac{\alpha_j^{(r)}}{(\sigma_j^{(r)})^{n-d}} \exp\left(-\frac{1}{2(\sigma_j^{(r)})^2}\|(I_n - U_j^{(r)}(U_j^{(r)})^{\mathrm{T}})y_{i,k}^{(r)}\|^2\right) f\left((U_j^{(r)})^{\mathrm{T}} y_{i,k}^{(r)}|\vartheta_j^{(r)}\right)},$$
$$y_{i,k}^{(r)} = x_i - b_k^{(r)}.$$

M-Step: For $k=1,...,K$ compute

$$\alpha_k^{(r+1)} = \frac{1}{N}\sum_{i=1}^N \beta_{i,k}^{(r)},$$
$$(U_k^{(r+1)}, b_k^{(r+1)}, \sigma_k^{(r+1)}, \vartheta_k^{(r+1)}) = \operatorname*{argmax}_{U,b,\mu,\Sigma} \sum_{i=1}^N \beta_{ik}^{(r)} \log(f(x_i|\tilde{\vartheta}_k))$$
$$\text{subject to} \quad U_k \in \mathrm{St}(d,n), \Sigma_k \in \mathrm{SPD}(d)$$
$$\text{with } \tilde{\vartheta}_k = (\tilde{\mu}_k, \tilde{\Sigma}_k) \text{ as in (14).}$$

end for

A convergence analysis of the EM algorithm via Kullback-Leibler proximal point algorithms was given in [5, 6], see also [16] for a nice review. The authors showed that the objective function decreases for the iterates of the algorithm. Hence we obtain the following corollary.

Corollary 1. *For the iterates* $\left(\mathbf{U}^{(r)}, \mathbf{b}^{(r)}, \alpha^{(r)}, \vartheta^{(r)}\right)_r$ *generated by Algorithm 2 the objective function* F *is decreasing.*

The interesting step is the second M-Step which requires again the maximization of a function. Based on (2) and (3) we can prove the following proposition.

Proposition 1. *Assume that $n+1$ of the points x_i, $i = 1, ..., N$ are affinely independent.*

Further, let f be the Gaussian density function (1) and $\beta_i \in \mathbb{R}_{\geq 0}$, $i = 1, \dots, N$. and let σ^2 be fixed.
i) For fixed σ^2, a solution of

$$\operatorname*{argmax}_{U,b,\mu,\Sigma} \sum_{i=1}^{N} \beta_i \log(f(x_i|\tilde{\vartheta})) \tag{19}$$

with $\tilde{\vartheta} = (\tilde{\mu}, \tilde{\Sigma})$ of the form (10) and (9) is given by

$$\hat{\mu} = 0, \quad \hat{\Sigma} = \frac{1}{\alpha}\hat{U}^{\mathrm{T}} S \hat{U}, \quad \text{and} \quad \hat{b} = \frac{1}{\alpha} m,$$

where

$$m = \sum_{i=1}^{N} \beta_i x_i, \quad C = \sum_{i=1}^{N} \beta_i x_i x_i^{\mathrm{T}}, \quad \alpha = \sum_{i=1}^{N} \beta_i, \quad S = C - \frac{1}{\alpha} m m^{\mathrm{T}},$$

and

$$\hat{U} \in \operatorname*{argmin}_{U \in \mathrm{St}(d,n)} G(U). \tag{20}$$

Here

$$G(U) := -\frac{1}{\sigma^2} \mathrm{tr}(U^{\mathrm{T}} S U) + \alpha \log(|U^{\mathrm{T}} S U|). \tag{21}$$

ii) If σ is learned, we have

$$\hat{\sigma}^2 = \tfrac{1}{\alpha(n-d)} \left(\mathrm{tr}(S) - \mathrm{tr}(\hat{U}^{\mathrm{T}} S \hat{U}) \right),$$

and G from (21) is replaced by

$$G(U) := (n-d) \log\left(\mathrm{tr}(S) - \mathrm{tr}(U^{\mathrm{T}} S U) \right) + \log(|U^{\mathrm{T}} S U|). \tag{22}$$

Note that α in the proposition is defined in another way than in the first M-step, more precisely, the factor $\frac{1}{N}$ is skipped. Before presenting the proof of the proposition, we give the following remark.

Remark 3. By definition of C in Proposition 1 we have that

$$S = \sum_{i=1}^{N} \beta_i (x_i - \tfrac{1}{\alpha} m)(x_i - \tfrac{1}{\alpha} m)^{\mathrm{T}}.$$

Since $n+1$ of the points x_i, $i = 1, ..., N$, are affinely independent, S is symmetric positive definite. In particular, it holds for G from (21) or (22) that $G(U) > -\infty$ for any $U \in \mathrm{St}(d,n)$. Further, since the function G is continuous and the Stiefel manifold is compact, we can conclude, that G has a global minimizer.

Proof of Proposition 1. i) Let σ be fixed. Using (8), we have for fixed U and b, as in the classical GMM, see (2) and (3), that the maximizer in (19) with respect to

μ and Σ fulfills

$$\mu = \frac{1}{\alpha}\sum_{i=1}^{N}\beta_i U^{\mathrm{T}}(x_i - b) = \frac{1}{\alpha}(U^{\mathrm{T}}m - \alpha U^{\mathrm{T}}b),$$

$$\begin{aligned}\Sigma &= \frac{1}{\alpha}\sum_{i=1}^{N}\beta_i \left(U^{\mathrm{T}}(x_i - b) - \mu\right)\left(U^{\mathrm{T}}(x_i - b) - \mu\right)^{\mathrm{T}} \\ &= \frac{1}{\alpha}\sum_{i=1}^{N}\beta_i \left(U^{\mathrm{T}}(x_i - \frac{1}{\alpha}m)\right)\left(U^{\mathrm{T}}(x_i - \frac{1}{\alpha}m)\right)^{\mathrm{T}} = \frac{1}{\alpha}U^{\mathrm{T}}SU.\end{aligned}$$

By Lemma 3.1, the negative objective function in (19) is given by

$$2\tilde{G}(U,b) = G_1(U,b) + G_2(U,b) + \alpha\log(|\Sigma|) + \text{const}, \tag{23}$$

$$G_1(U,b) = \tfrac{1}{\sigma^2}\sum_{i=1}^{N}\beta_i(x_i - b)^{\mathrm{T}}(I_n - UU^{\mathrm{T}})(x_i - b) + \alpha(n-d)\log(\sigma^2) \tag{24}$$

$$G_2(U,b) = \sum_{i=1}^{N}\beta_i \left(U^{\mathrm{T}}x_i - (U^{\mathrm{T}}b + \mu)\right)^{\mathrm{T}}\Sigma^{-1}\left(U^{\mathrm{T}}x_i - (U^{\mathrm{T}}b + \mu)\right).$$

In the following, we use const as a generic constant which has values independent of μ, Σ, U and b. The linear trace operator $\mathrm{tr} : \mathbb{R}^{d,d} \to \mathbb{R}$ fulfills $x^{\mathrm{T}}Ay = \mathrm{tr}(Axy^{\mathrm{T}})$ and in particular $x^{\mathrm{T}}UU^{\mathrm{T}}x = \mathrm{tr}(U^{\mathrm{T}}xx^{\mathrm{T}}U)$. Using this property we obtain

$$G_2(U,b) = \mathrm{tr}\Big(\Sigma^{-1}\underbrace{\sum_{i=1}^{N}\beta_i \left(U^{\mathrm{T}}x_i - (U^{\mathrm{T}}b + \mu)\right)\left(U^{\mathrm{T}}x_i - (U^{\mathrm{T}}b + \mu)\right)^{\mathrm{T}}}_{=\Sigma}\Big) = \mathrm{tr}(I).$$

Thus, the only term in (23) which depends on b and U is G_1. Further, minimizing G_1 is equivalent to minimizing

$$g_1(U,b) \coloneqq \sum_{i=1}^{N}\beta_i(x_i - b)^{\mathrm{T}}(I_n - UU^{\mathrm{T}})(x_i - b).$$

For fixed U, we can minimize g_1 with respect to b by setting the gradient to 0. Since g_1 is convex in b this is equivalent for being a global minimizer. This yields

$$0 = \sum_{i=1}^{N}\beta_i(I_n - UU^{\mathrm{T}})(b - x_i)$$

which is equivalent to

$$0 = (I_n - UU^{\mathrm{T}})(\alpha b - m).$$

In particular, $b = \frac{1}{\alpha}m$ is a global minimizer of g_1 resp. G_1, and it is independent of U. Using this, we get

$$\mu = \frac{1}{\alpha}(U^{\mathrm{T}}m - \alpha U^{\mathrm{T}}b) = 0.$$

Minimizing G_1 with respect to U for $b = \frac{1}{\alpha}m$ is equivalent to minimizing

$$G_1(U, \frac{1}{\alpha}m) = -\tfrac{1}{\sigma^2}\mathrm{tr}\left(U^{\mathrm{T}}SU\right) + \text{const}.$$

Further we have $\log\left(\left|\frac{1}{\alpha}U^{\mathrm{T}}SU\right|\right) = \log(|U^{\mathrm{T}}SU|) + \text{const}$. Thus, by combining the above computations, we get that minimizing (23) with respect to U is equivalent to minimizing

$$G(U) = -\frac{1}{\sigma^2}\operatorname{tr}\left(U^{\mathrm{T}}SU\right) + \alpha\log(|U^{\mathrm{T}}SU|).$$

ii) Now consider the case, where σ is learned. Again by (8), the maximizer in (19) with respect to σ is given by the maximizer of

$$\sum_{i=1}^{N}\beta_i\big(-\tfrac{1}{2\sigma^2}\|(I_n - UU^{\mathrm{T}})(x_i-b)\|^2 - (n-d)\log(\sigma)\big).$$

By setting the derivative to zero, one obtains, that

$$\sigma^2 = \tfrac{1}{\alpha(n-d)}\sum_{i=1}^{N}\beta_i(x_i-b)^{\mathrm{T}}(I_n - UU^{\mathrm{T}})(x_i-b).$$

Then the function in (24) modifies to

$$G_1(U,b) = \alpha(n-d)\log\Big(\sum_{i=1}^{N}\beta_i(x_i-b)^{\mathrm{T}}(I_n - UU^{\mathrm{T}})(x_i-b)\Big) + \text{const}. \tag{25}$$

Now the monotonicity of the logarithm implies that minimizing G_1 is again equivalent to minimizing g_1. Hence, as in case i) we get $b = \frac{1}{\alpha}m$ is a global minimizer of g_1 resp. G_1, and it is independent of U. Using this, we obtain

$$\mu = \frac{1}{\alpha}(U^{\mathrm{T}}m - \alpha U^{\mathrm{T}}b) = 0 \quad \text{and} \quad \sigma^2 = \tfrac{1}{\alpha(n-d)}\left(\operatorname{tr}(S) - \operatorname{tr}(U^{\mathrm{T}}SU)\right).$$

By (25), minimizing G_1 with respect to U for $b = \frac{1}{\alpha}m$ is equivalent to minimizing

$$G_1(U, \frac{1}{\alpha}m) = \alpha(n-d)\log\left(\operatorname{tr}(S) - \operatorname{tr}(U^{\mathrm{T}}SU)\right) + \text{const},$$

such that minimizing (23) with respect to U is equivalent to minimizing

$$G(U) = (n-d)\log\left(\operatorname{tr}(S) - \operatorname{tr}(U^{\mathrm{T}}SU)\right) + \log(|U^{\mathrm{T}}SU|).$$

□

By Proposition 1, the M-Step of Algorithm 2 reduces for $k = 1, ..., K$ to the computation of

$$\begin{aligned}
\alpha_k^{(r+1)} &= \frac{1}{N}\sum_{i=1}^{N}\beta_{i,k}^{(r)},\\
m_k = \sum_{i=1}^{N}\beta_{i,k}x_i,\ C_k &= \sum_{i=1}^{N}\beta_{i,k}x_i x_i^{\mathrm{T}},\\
(U_k^{(r+1)}, b_k^{(r+1)}) &\in \operatorname*{argmin}_{U\in\mathrm{SPD}(d,n),b\in\mathbb{R}^n} G_k(U,b) \qquad \text{with } G_k \text{ in (21)},\\
\mu_k^{(r+1)} &= \frac{1}{N\alpha_k^{(r+1)}}\left(U_k^{(r+1)}\right)^{\mathrm{T}}\left(m_k - N\alpha_k^{(r+1)}b_k^{(r+1)}\right),\\
S_k &= C_k - m_k\left(b_k^{(r+1)}\right)^{\mathrm{T}} - b_k^{(r+1)}m_k^{\mathrm{T}} + N\alpha_k^{(r+1)}b_k^{(r+1)}\left(b_k^{(r+1)}\right)^{\mathrm{T}}\\
\Sigma_k^{(r+1)} &= \frac{1}{N\alpha_k^{(r+1)}}\left(U_k^{(r+1)}\right)^{\mathrm{T}}S_k\,U_k^{(r+1)}
\end{aligned}$$

Note that the large data set $\mathcal{X}$ is involved in the computation of m_k and C_k, but it does not influence the computational time for minimizing the G_k, $k = 1, \dots, K$. Indeed, the E-Step of Algorithm 2 will be the most time consuming one.

4.2. **PALM for Minimizing G.** To minimize G in (21) we propose to use the Proximal alternating linearized minimization (PALM) [2], resp. its accelerated version iPALM [25], where the 'i' stands for inertial. As a special case the PALM algorithm can be applied to functions of the form

$$F(x) = H(x) + f(x) \tag{26}$$

where $H \in C^1(\mathbb{R}^d)$ and a lower semi-continuous function $f\colon \mathbb{R}^d \to (-\infty, \infty]$. It is based on the computation of so-called proximal operators. For a proper and lower semi-continuous function $f\colon \mathbb{R}^d \to (-\infty, \infty]$ and $\tau > 0$ the *proximal mapping* $\operatorname{prox}_\tau^f\colon \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ is defined by

$$\operatorname{prox}_\tau^f(x) = \operatorname*{argmin}_{y \in \mathbb{R}^d} \left\{ \tfrac{\tau}{2}\|x - y\|^2 + f(y) \right\},$$

where $\mathcal{P}(\mathbb{R}^d)$ denotes the power set of $\mathbb{R}^d$.

Starting with an arbitrary $x^{(0)}$ PALM performs the iterations

$$x^{(r+1)} \in \operatorname{prox}^f_{\tau^{(r)}}\left(x^{(r)} - \tfrac{1}{\tau^{(r)}}\nabla H(x^{(r)})\right)$$

Further, iPALM is detailed in Algorithm 3. Indeed, we have applied the iPALM algorithm in our numerical examples. However, although we observed convergence of the iterates numerically, we have not proved convergence theoretically so far. Alternatively, we could apply the PALM algorithm which is slightly slower. Note again, that the E-Step of the algorithm is the most time consuming one.

Algorithm 3 iPALM

Input: $\alpha^{(r)}, \beta^{(r)}$ initialization $x^{(1)}$, $x^{(0)}$

for $r = 1, 2, \dots$ **do** until a convergence criterion is reached

$$\begin{aligned} y^{(r)} &= x^{(r)} + \alpha^{(r)}(x^{(r)} - x^{(r-1)}), \\ z^{(r)} &= x^{(r)} + \beta^{(r)}(x^{(r)} - x^{(r-1)}), \\ x^{(r+1)} &\in \operatorname{prox}^f_{\tau^{(r)}}(y^{(r)} - \tfrac{1}{\tau^{(r)}}\nabla H(z^{(r)})), \end{aligned}$$

end for

In the following, we give details on PALM for our setting. For our problem (20), we choose $f(U) \coloneqq \iota_{\mathrm{St}(d,n)}$ and

$$H(U) \coloneqq G(U)\eta(\|I_d - U^T U\|_F^2), \tag{27}$$

where

$$\eta(x) \coloneqq \begin{cases} 1, & \text{if } x \in (-\rho, \rho), \\ \exp(-\frac{\rho}{\rho - (|x| - \rho)^2}), & \text{if } x \in (-2\rho, -\rho] \cup [\rho, 2\rho), \\ 0, & \text{otherwise.} \end{cases}$$

is a smooth cutoff function of the interval $(-\rho, \rho)$ for some $\rho > 0$. Then, the iteration scheme reads as

$$U^{(r+1)} \in \Pi_{\mathrm{St}(d,n)}(U^{(r)} - \tfrac{1}{\tau^{(r)}} \nabla H(U^{(r)})) \tag{28}$$

where $\Pi_{\mathrm{St}(d,n)}$ denotes the orthogonal projection onto the Stiefel manifold.

Remark 4. (Projection onto Stiefel manifolds) Concerning this orthogonal projection, it is well known [12], that for a matrix $A \in \mathbb{R}^{n,d}$, the projection $\Pi_{\mathrm{St}(d,n)}(A)$ is given by the orthonormal polar factor W from the polar decomposition

$$A = WM, \quad W \in \mathrm{St}(d,n), \quad M \in \mathrm{SPD}(d).$$

Further, this orthonormal polar factor can be computed by $W = UV$, where $A = U\Sigma V$ is the singular value decomposition of A, see [12]. The authors of [13] propose to use the so-called Schulz-iteration

$$X_{k+1} = X_k(I + \tfrac{1}{2}(I - X_k^{\mathrm{T}} X_k))$$

with $X_0 = A$ for computing the orthonormal polar factor of a full rank matrix A. Unfortunately, the convergence of this iteration requires that $\|I - A^{\mathrm{T}} A\|_F < 1$, which is usually not fulfilled in our case.

Note that for any $r \in \mathbb{N}$, the matrix $U^{(r)}$ belongs to the Stiefel manifold, such that $\eta(\|I_d - U^{\mathrm{T}} U\|_F) = 1$ in a neighborhood of $U^{(r)}$. Thus, we can replace the gradient with respect to H by the gradient with respect to G in (28). Then the iteration scheme reads as

$$U^{(r+1)} \in P_{\mathrm{St}(d,n)}(U^{(r)} - \tfrac{1}{\tau^{(r)}} \nabla G(U^{(r)})), \tag{29}$$

In particular, we do not need to choose the ρ explicitly within our algorithm.

To show convergence of the algorithm, we need the following two lemmas.

Lemma 4.1. *Let H be defined by (27). Then the function ∇H is globally Lipschitz continuous.*

Proof. The function H is twice continuously differentiable and zero outside of a compact set. Hence the second order derivative is bounded and $\nabla_U H(\cdot, b)$ is globally Lipschitz continuous. □

Further, let us recall the notation of Kurdyka-Łojasiewicz functions. For $\delta \in (0, \infty]$, we denote by Φ_δ the set of all concave continuous functions $\phi\colon [0, \delta) \to \mathbb{R}_{\geq 0}$ which fulfill the following properties:

(i) $\phi(0) = 0$.
(ii) ϕ is continuously differentiable on $(0, \delta)$.
(iii) For all $s \in (0, \delta)$ it holds $\phi'(s) > 0$.

For a proper and lower semicontinuous function $\gamma\colon \mathbb{R}^d \to (-\infty, +\infty]$ denote by $\partial\gamma$ the subdifferntial of γ.

Definition 4.2 (Kurdyka-Łojasiewicz property)**.** A proper, lower semicontinuous function $\gamma\colon \mathbb{R}^d \to (-\infty, +\infty]$ has the Kurdyka-Łojasieweicz (KL) property at $\bar{u} \in \operatorname{dom} \partial\gamma = \{u \in \mathbb{R}^d : \partial\gamma \neq \emptyset\}$ if there exist $\delta \in (0, \infty]$, a neighborhood U of $\bar{u}$ and a function $\phi \in \Phi_\delta$, such that for all

$$u \in U \cap \{v \in \mathbb{R}^d : \gamma(\bar{u}) < \gamma(v) < \gamma(\bar{u}) + \delta\},$$

it holds

$$\phi'(\gamma(u)-\gamma(\bar{u}))\mathrm{dist}\,(0,\partial\gamma(u))\geq 1.$$

We say that γ is a KL function, if it satisfies the KL property in each point $u\in \operatorname{dom}\partial\gamma$.

Lemma 4.3. *The function H defined in (27) is a KL function.*

Proof. The functions G and η are sums, products, quotients and concatenations of real analytic functions. Thus, also H is a real analytic function. This implies that it is a KL function, see [1, Remark 5] and [18, 19]. □

The following theorem follows directly from [2, Lemma 3, Theorem 1].

Theorem 4.4 (Convergence of PALM). *Let $F\colon \mathbb{R}^d\to(-\infty,\infty]$ be given by (26) and let ∇H be globally L-Lipschitz continuous. Let $(x^{(r)})_r$ be the sequence generated by PALM, where the step size parameters fulfill*

$$\tau^{(r)}\geq \gamma L$$

for some $\gamma>1$. Then, for $\eta\coloneqq(\gamma-1)L$, the sequence $(F(x^{(r)}))_r$ is nonincreasing and

$$\tfrac{\eta}{2}\|x^{(r+1)}-x^{(r)})\|_2^2\leq F(x(r))-F(x^{(r+1)}).$$

If F is in addition a KL function and the sequence $(x^{(r)})_r$ is bounded, then it converges to a critical point of F.

By Lemma 4.1 and 4.3 and the fact that G coincides with H in a neighborhood of the Stiefel manifold we obtain the following corollary.

Corollary 2. *Let $(U^{(r)})_r$ be generated by (29) with $\tau^{(r)}\geq\gamma L$, where L is the Lipschitz constant of ∇H and $\gamma>1$. Consider the sequence generated by PALM with (29). Then, the sequence $(G(U^{(r)}))_r$ is monotone decreasing and the sequence $(U^{(r)})_r$ converges to a critical point of G.*

5. Superresolution.

In this section, we adapt the superresolution method proposed by Sandeep and Jacob [26] to our PCA-GMM model. The method works in three steps. In the first step, we learn the PCA-GMM based on a reference image, where high and low resolution images are known. After that, in the second and third step, we use the learned PCA-GMM for superresolution of a low resolution image with unknown high resolution counterpart.

Learning the PCA-GMM. For given low resolution patches $x_{L,i}\in\mathbb{R}^{\tau^2}$ of an image and their higher resolution counterparts $x_{H,i}\in\mathbb{R}^{q^2\tau^2}$, $q\in\mathbb{N}$, $q>2$, $i=1,\dots,N$ we learn a PCA-GMM based on the data $x_i=\begin{pmatrix}x_{H,i}\\ x_{L,i}\end{pmatrix}\in\mathbb{R}^n$, where $n=(q^2+1)\tau^2$, by Algorithm 2. This provides us with parameters $(\mathbf{U},\mathbf{b},\alpha,\mu,\mathbf{\Sigma})$ of the reduced d-dimensional GMM. Using these parameters, we compute the parameters of the corresponding high-dimensional mixture model $(\alpha,\tilde{\mu}_k,\tilde{\Sigma}_k)$, $k=1,\dots,K$, where μ_k and Σ_k are defined as in (10) and (9). In the following, we use the notations $\tilde{\mu}_k=\begin{pmatrix}\tilde{\mu}_{H,k}\\ \tilde{\mu}_{L,k}\end{pmatrix}$ and $\tilde{\Sigma}_k=\begin{pmatrix}\tilde{\Sigma}_{H,k} & \tilde{\Sigma}_{HL,k}\\ (\tilde{\Sigma}_{HL,k})^{\mathrm{T}} & \tilde{\Sigma}_{L,k}\end{pmatrix}$.

Estimation of high resolution patches by MMSE. Now, we want to improve the resolution of a given low resolution patch $x_L \in \mathbb{R}^{\tau^2}$. First, we select the component k^*, such that the likelihood that x_L belongs to the k^*-th component is maximal, i.e., we compute

$$k^* = \operatorname*{argmax}_{k=1,\dots,K} \alpha_k f(x_L|\tilde{\mu}_{L,k}, \tilde{\Sigma}_{L,k}).$$

Then we estimate the high resolution patch $x_H \in \mathbb{R}^{q^2\tau^2}$ as the minimum mean-square estimator (MMSE). The following remark briefly reviews this estimator.

Remark 5. (MMSE) Given a random variable $Y : \Omega \to \mathbb{R}^d$ in a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, we wish to estimate a random variable $X : \Omega \to \mathbb{R}^d$, i.e., we seek an estimator $T\colon \mathbb{R}^d \to \mathbb{R}^d$ such that $\hat{X} = T(Y)$ approximates X. A common quality measure for this task is the *mean square error* $\mathbb{E}\|X - T(Y)\|_2^2$, which gives rise to the definition of the *minimum mean square estimator*

$$T_{\mathrm{MMSE}} \in \operatorname*{argmin}_T \mathbb{E}\|X - T(Y)\|_2^2. \tag{30}$$

Under weak additional regularity assumptions on the estimator T, the Lehmann-Scheffé theorem [17] states that the general solution of the minimization problem (30) is given by

$$T_{\mathrm{MMSE}}(Y) = \mathbb{E}(X|Y).$$

In general, it is not possible to give an analytical expression of the MMSE estimator T_{MMSE}. An exception are Gaussian random variables: if X and Y are jointly normally distributed, i.e.,

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix} \right),$$

then the conditional distribution of X given $Y = a$ is normally as well and reads as

$$(X|Y = a) \sim \mathcal{N}\big(\mu_{X|Y}, \Sigma_{X|Y}\big),$$

where

$$\mu_{X|Y} = \mu_X + \Sigma_{XY}\Sigma_Y^{-1}(a - \mu_Y), \qquad \Sigma_{X|Y} = \Sigma_X - \Sigma_{XY}\Sigma_Y^{-1}\Sigma_{YX}.$$

As a consequence we obtain for normally distributed random variables the MMSE estimator

$$T_{\mathrm{MMSE}}(Y) = \mathbb{E}(X|Y) = \mu_X + \Sigma_{XY}\Sigma_Y^{-1}(Y - \mu_Y). \tag{31}$$

In our superresolution task, we assume that the vector $\begin{pmatrix} x_H \\ x_L \end{pmatrix}$ is a realization of a random variable $\begin{pmatrix} X_H \\ X_L \end{pmatrix} \sim \mathcal{N}(\tilde{\mu}_{k^*}, \tilde{\Sigma}_{k^*})$. Then, by (31), the MMSE can be computed as

$$x_H = \tilde{\mu}_{H,k^*} + \tilde{\Sigma}_{HL,k^*}(\tilde{\Sigma}_{L,k^*})^{-1}(x_L - \tilde{\mu}_{L,k^*}).$$

Reconstruction of the high resolution image by patch averaging. We estimate for any patch in the low resolution image the corresponding high resolution patch. Once we have estimated the high resolution patches, we compute an estimate of the high resolution image in the following way:

Let $x_H = (x_{k,l})_{k,l=1}^{q\tau} \in \mathbb{R}^{q\tau,q\tau}$ be a two-dimensional high resolution patch. Now, we assign to each pixel $x_{k,l}$ the weight

$$w_{k,l} := \exp\Big(-\tfrac{\gamma}{2}\big((k-\tfrac{q\tau+1}{2})^2 + (l-\tfrac{q\tau+1}{2})^2\big)\Big).$$

After that, we add up for each pixel in the high resolution image the corresponding weighted pixel values and normalize the result by dividing by the sum of the weights.

6. **Numerical results.** In this section, we demonstrate the performance of our algorithm by two- and three-dimensional examples, where we mainly focus on material data which provided the original motivation for this work. More precisely, in the frame of the ITN MUMMERING, a series of multi-scale 3D images has been acquired by synchrotron micro-tomography at the SLS beamline TOMCAT. Materials of two samples were selected to provide 3D images having diverse levels of complexity:

- The first one is a sample of Fontainebleau sandstone ("FS"), a natural rock rather homogeneous and commonly used in the oil industry for flow experiments.
- The second one is a composite ("SiC Diamonds") obtained by microwave sintering of silicon and diamonds, see [29].

Slices of the corresponding 3D images are given in the first two columns of Figure 1.

All implementations were done in Python and Tensorflow and they can be parallelized on a GPU. We run all our experiments on a Lenovo ThinkStation with Intel i7-8700 6-Core processor with 32GB RAM and NVIDIA GeForce GTX-2060 Super GPU. The code is available online[2].

For the implementation of PALM and iPALM, we use the implementation framework from [11][3]. As suggested in [25] we set the extrapolation factors $\gamma_1^{(r)} = \gamma_2^{(r)} = \frac{r-1}{r+2}$ and choose $\tau_1^{(r)} = \frac{1}{\tilde{L}_1(b^{(r)})}$ and $\tau_2^{(r)} = \frac{1}{\tilde{L}_2(U^{(r+1)})}$, where $\tilde{L}_1(b^{(r)})$ and $\tilde{L}_2(U^{(r+1)})$ are estimates of the Lipschitz constant of $\nabla_U G(\cdot, b^{(r)})$ and $\nabla_b G(U^{(r+1)}, \cdot)$.
We generate pairs of high and low resolution images using the following superresolution operator:

Generation of the test examples. For convenience, we describe the generation in 2D. The 3D setting is treated in a similar way. We use the operator A from the implementation of [23][4]. This operator consists of a blur operator H and a downsampling operator S. The blur operator is given by a convolution with a Gauss kernel with standard deviation 0.5. For the downsampling operator S we use the discrete Fourier transform (DFT). Given an image $x \in \mathbb{R}^{m,n}$ the two-dimensional DFT is defined by $\mathcal{F}_{m,n} := \mathcal{F}_n \otimes \mathcal{F}_m$, where $\mathcal{F}_n = (\exp(-2\pi \mathrm{i} kl/n))_{k,l=0}^{n-1}$. Now, the downsampling operator $S\colon \mathbb{R}^{m,n} \to \mathbb{R}^{m_2,n_2}$ is given by

$$S = \frac{m_2 n_2}{mn} \mathcal{F}_{m_2,n_2}^{-1} D \mathcal{F}_{m,n},$$

[2] https://github.com/johertrich/PCA_GMMs
[3] https://github.com/johertrich/Inertial-Stochastic-PALM
[4] https://github.com/pshibby/fepll_public

where for $x \in \mathbb{R}^{m,n}$ the (i,j)-th entry of $D(x)$ is given by

$$\begin{cases} x_{i,j}, & \text{if } i \leq \frac{m_2}{2} \text{ and } j \leq \frac{n_2}{2}, \\ x_{i+m-m_2,j}, & \text{if } i > \frac{m_2}{2} \text{ and } j \leq \frac{n_2}{2}, \\ x_{i,j+n-n_2}, & \text{if } i \leq \frac{m_2}{2} \text{ and } j > \frac{n_2}{2}, \\ x_{i+m-m_2,j+n-n_2}, & \text{if } i > \frac{m_2}{2} \text{ and } j > \frac{n_2}{2}. \end{cases}$$

For a given high resolution image x, we now generate the low resolution image y by $y = Ax + \epsilon$, where ϵ is a realization of white Gaussian noise with standard deviation 0.02.

Initialization of the EM algorithms. Since the negative log-likelihood function is non-convex and admits many critical points, EM algorithms for GMMs are very sensitive with respect to the initialization. For example this can be seen by considering the case when $\vartheta_k^{(r)} = \vartheta_l^{(r)}$, $k, l = 1, ..., K$ for some $r \in \mathbb{N}$. Then we obtain that $\beta_{i,k}^{(r)} = \alpha_k$ and consequently $\vartheta_k^{(r+1)} = \vartheta_l^{(r+1)}$, $k, l = 1, ..., K$. The same effect appears for PCA-GMMs and HDDC. Consequently the initialization of the EM algorithms is of great importance. For our numerical examples, we initialize the GMMs as follows. We set $\alpha_k^{(0)} = \frac{1}{K}$, for $k = 1, ..., K$. For initializing the means, we choose randomly K distinct data points $\mu_1, ..., \mu_K$ from our training data $x_1, ..., x_N$. Finally, we choose for each $k = 1, ..., K$ the M points $y_1, ..., y_M$ from $x_1, ..., x_N$ which are the closest ones to μ_k and initialize the covariances by $\Sigma_k = \frac{1}{M} \sum_{i=1}^{M} y_i y_i^{\mathrm{T}}$. The number M is chosen according to the dimension n of the data. In our examples, we use $M = 2n$.

We initialize the PCA-GMMs and HDDC by taking the initialization for GMMs, running one E-Step from the EM algorithm for GMMs followed by the M-step of the PCA-GMMs or HDDC, respectively.

Choice of σ and K. The PCA-GMM model depends heavily on the choice of the parameter σ. As pointed out in Subsection 4.1, this parameter could be learned from the data. However, the forward model for the low resolution images $y = Ax + \epsilon$ for some (unknown) superresolution operator A, the high resolution image x and noise ϵ suggest to choose the σ according to the standard deviation of the noise ϵ. Note, that in our experiments, the low resolution images are artificially generated by applying a downsampling operator and adding some noise. Consequently, the standard deviation of ϵ is known. Nevertheless, if the noise level is unknown, it could be estimated very accurately from the data based on homogeneous area detection as done, e.g., in [10, 27].

In practice, it can be unstable to estimate the standard deviation of the noise within the optimization of the mixture model, since this requires that the image patches belong exactly (and not only approximately) to a dimensionality reduced GMM with K components, which is an unrealistic assumption. Therefore, it can be beneficial and quite more accurate to estimate the standard deviation of the noise a priori. In particular, if the standard deviation of the noise is known, fixing σ can be the better approach.

Note that the noise with standard deviation σ within the superresolution model does not necessarily imply that the eigenvalues of the covariance matrices in the mixture model are greater than or equal to σ^2 (which is assumed for HDDC), since the noise is only applied to the low resolution images.

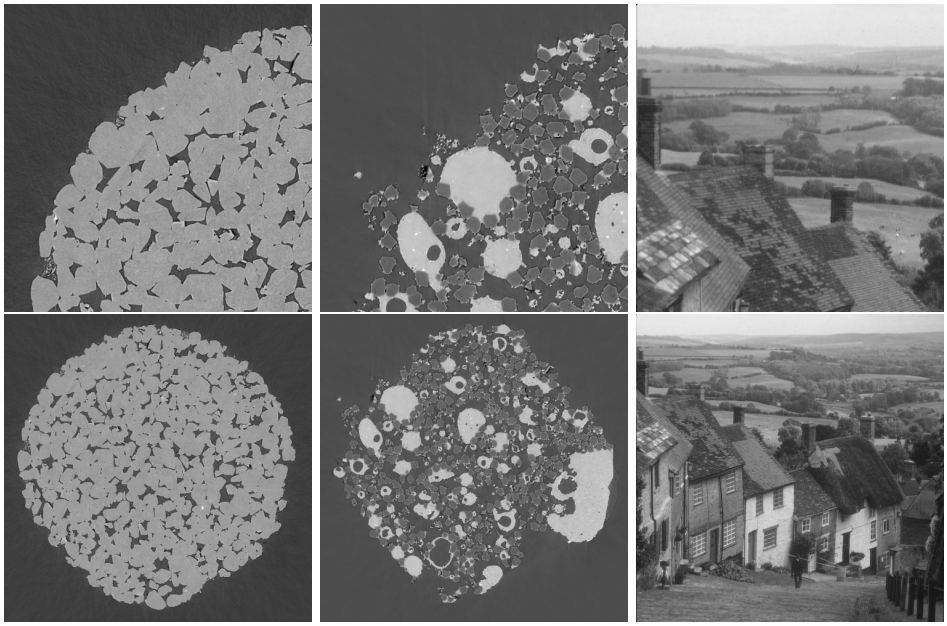


Figure 1. Top: Images for estimating the mixture models. Bottom: Ground truth for reconstruction. First column: Material "FS", second column: Material "SiC Diamonds", third column: `goldhill` image.

Also the number of components K of the mixture models can have a large impact on the results. For superresolution, a detailed comparison of the prediction quality for different choices of K was done by Sandeep and Jacob in [26]. They observed that the benefit of taking more than 100 components in the GMM is usually very small. Therefore, we take $K = 100$ components for all mixture model in our numerical examples.

Comparison of the computation times. Note that there already exist implementations of HDDC by some of the authors of [3]. However, to provide a fair comparison of the execution times, we reimplement the EM algorithm for HDDC in Python and Tensorflow, such that it supports GPU parallelization. Further, note that we compute the updates of α, m and C simultaneously to the E-step such that the corresponding execution time is contained in the E-step, even though the updates technically belong to the M-step. This has the advantage that we have to iterate only once over the whole data set and enables a better parallelization. We implemented this optimization of the order of computation for all of the models (GMM, PCA-GMM and HDDC) analogously.

2D-Data. For estimating the parameters of the mixture models, we use the upper left quarter of the image as in the top row of Figure 1. As ground truth for the reconstruction we use the whole images as in the bottom row. The images in the left and middle columns are the middle slices of the material data "FS" and "SiC Diamonds". The high resolution images have a size of 2560×2560. The right column contains the `goldhill` image, which has the size 512×512.

We estimate the parameters of a GMM and of our PCA-GMM as described in the previous sections. First, we fix the parameter σ in Algorithm 2 as the standard deviation of the noise on the low dimensional image (i.e. $\sigma = 0.02$). Second, we consider the case when σ is learned from the data and finally we compare our results with HDDC [3]. Each mixture model has $K = 100$ classes. We use the magnification factors $q \in \{2, 4\}$ and the patch size $\tau = 4$ for the low resolution patches. This corresponds to a patch size of $q\tau = 8$ or $q\tau = 16$ respectively for the high resolution images. For the material images, this leads to $N \approx 400000$ patches for $q = 2$ and $N \approx 100000$ for $q = 4$. Using the `goldhill` image, we get $N \approx 15000$ patches for $q = 2$ and $N \approx 3700$ patches for $q = 4$. We reduce the dimension of the pairs of high and low resolution patches from $n = (q^2 + 1)\tau^2 = 80$ or $n = (q^2 + 1)\tau^2 = 272$ respectively to d for $d \in \{4, 8, 12, 16, 20\}$. After estimating the mixture models, we use the reconstruction method from [26] as described in the previous section to reconstruct the ground truth from the artificially downsampled images. The resulting PSNRs are given in Table 1. As a reference we also measure the PSNR of the bicubic interpolation. The average execution times for one E-step and one M-step are given in Table 2. Figure 2 shows some small areas of the high resolution images, low resolution images and the corresponding reconstructions for GMMs and PCA-GMM with $d = 12$ and $d = 20$. The result with $d = 12$ for PCA-GMM is already almost as good as GMM, whereas the dimension of the patches was reduced by a factor between 4 and 22 (depending on the case). Further, we observed that the dimensionality reduction reduces the execution time of the E-step significantly. On the other hand, the execution time of the M-step is larger than those in the GMM for all dimension reduced models due its higher complexity. Comparing the different dimensionality reduced models, we observe that the PCA-GMM with fixed σ gives significantly better results than the other models, while HDDC achieves the fastest M-step due to the closed-form updates. However, compared to the execution time of the E-step, this advantage seems to be negligible for large data sets as, e.g., the patches from the FS and SiC Diamonds image.

Figure 3 shows a histogram of the eigenvalues of the covariance matrices Σ_k, $k = 1, ..., K$ of the PCA-GMM model with fixed $\sigma = 0.02$ for the FS and SiC Diamonds image with magnification $q = 4$. We observe, that for the SiC Diamonds image a significant amount of eigenvalues are smaller than $\sigma^2 = 4 \cdot 10^{-4}$ which is not possible within a HDDC model [3]. For the FS image, the eigenvalues are mostly greater than σ^2.

3D-Data. In the following, we present the same experiments as in the 2D-case but with 3D-data. For this experiment, we crop a $600 \times 600 \times 600$ image from the material images "FS" and "SiC Diamonds". For the estimation of the mixture model, we use the upper front left $300 \times 300 \times 300$ part of the images and crop randomly $N = 1000000$ patches.

Again, we estimate the parameters of a GMM and a PCA-GMM with $K = 100$ classes and fixed $\sigma = 0.02$ as described in the previous sections. Since we have seen in the 2D examples that the results of PCA-GMMs with learned σ and HDDC are similar, we compare our 3D results just with HDDC. As magnification factor, we use $q = 2$. For the low resolution image we use $\tau \times \tau \times \tau$-patches with patch size $\tau = 4$ and for the high resolution image we use a patch size of $q\tau = 8$. We reduce the dimension of the pairs of high and low resolution patches from $n = (q^3 + 1)\tau^3 = 576$ to d for $d \in \{20, 40, 60\}$. After estimating the mixture models, we use the reconstruction method from [26] as described in the previous paragraph to reconstruct the ground

		Magnification factor $q = 2$			Magnification factor $q = 4$		
	d	FS	Diamonds	Goldhill	FS	Diamonds	Goldhill
bicubic	-	30.57	30.67	28.99	25.27	25.19	24.66
GMM	-	35.49	37.21	31.63	30.69	30.74	27.80
PCA-GMM, $\sigma = 0.02$	20	35.44	37.24	31.25	30.75	30.74	27.64
	16	35.42	37.22	31.25	30.74	30.62	27.59
	12	35.47	37.13	31.18	30.67	30.48	27.55
	8	35.32	36.69	31.00	30.46	30.16	27.38
	4	34.69	35.23	30.42	29.78	29.24	26.89
PCA-GMM, learned σ	20	35.22	37.06	31.27	30.43	30.51	27.66
	16	35.14	37.01	31.14	30.34	30.31	27.51
	12	34.95	36.54	30.94	30.13	29.84	27.33
	8	34.43	35.47	30.54	29.62	29.08	26.88
	4	32.74	33.41	29.69	28.51	27.75	26.16
HDDC [3]	20	35.35	37.12	31.35	30.54	30.63	27.73
	16	35.31	37.10	31.25	30.47	30.48	27.62
	12	35.24	36.64	31.08	30.27	30.08	27.40
	8	34.76	35.66	30.76	29.80	29.34	27.00
	4	33.46	33.86	29.93	28.61	27.99	26.37

Table 1. PSNRs of the reconstructions of artificially downsampled 2D images using either bicubic interpolation, a GMM, PCA-GMM for different choices of d or HDDC. The magnification factor is set to $q \in \{2, 4\}$. PCA-GMM produces results almost as good as GMM, with a much lower dimensionality.

truth from of the artificially downsampled images. The resulting PSNRs are given in Table 3 and the average execution times of one E-step and one M-step are given in Table 4. As a reference we also measure the PSNR of the nearest neighbor interpolation.

7. **Conclusions.** In this paper, we presented a new algorithm to perform image superresolution. Based on previous work by Sandeep and Jacob [26], we added a dimension reduction step within the GMM model using PCA on patches. The new variational model, called PCA-GMM is of interest on its own, and can be also applied for other tasks. We solved our PCA-GMM model by an EM algorithm with the usual decreasing guarantees for the objective if the E-step and M-step can be performed exactly, see Corollary 1. However, our M-step requires to solve a non-convex constrained minimization problem. Here we propose a PALM algorithm and prove that all assumptions for the convergence of the sequence of iterates to a critical point required by [2] are fulfilled, see Corollary 2. Our algorithm has the advantage that the M-step is cheap in relation to the E-step since it does not rely on the large numbers of samples in the inner iterations.

We have demonstrated the efficiency of the new model by numerical examples, in the case of 2D and 3D images. They confirm that PCA-GMM is an efficient way of reducing the dimension of the patches, while keeping almost the same quality of the results than with a GMM algorithm. This dimension reduction is of the utmost importance when dealing with 3D images, where the size of the data gets very large.

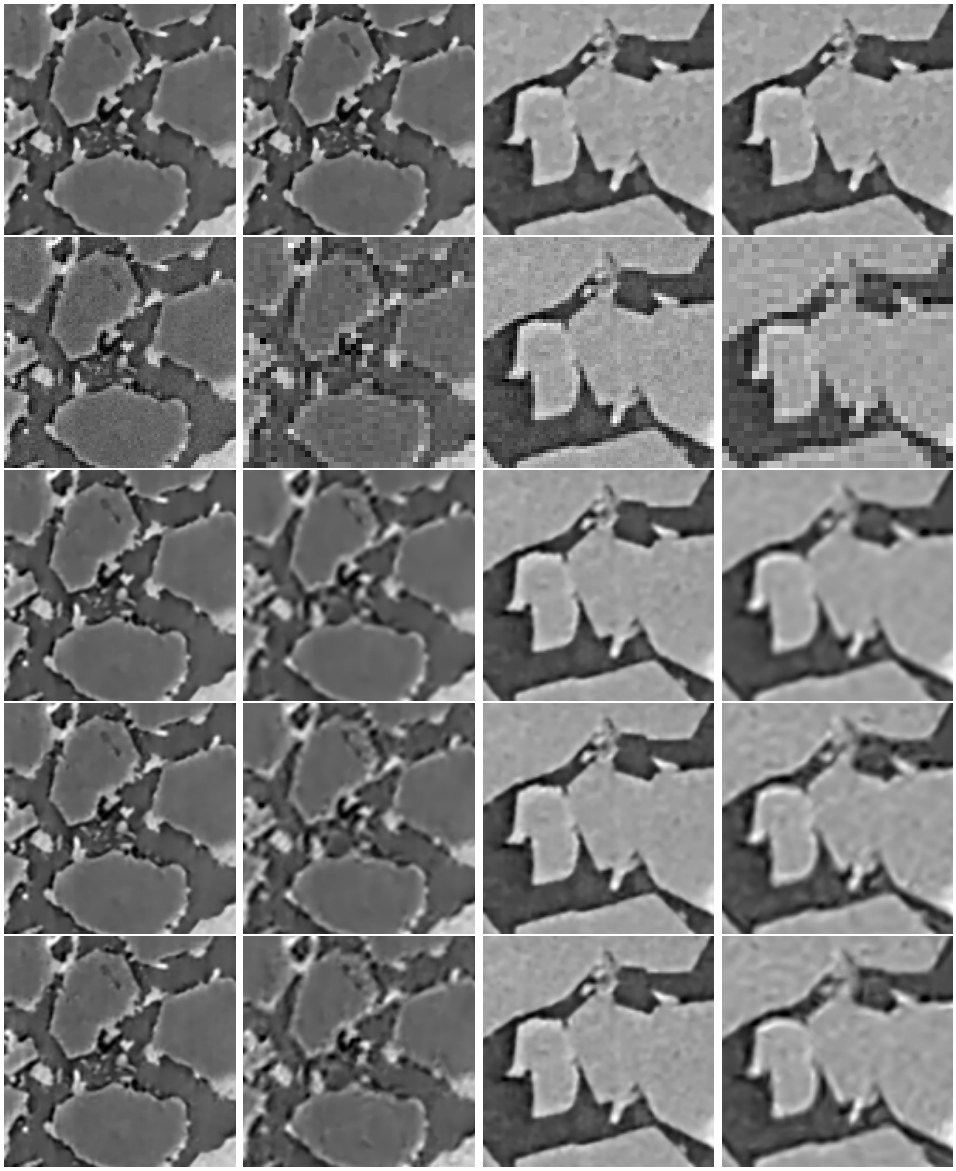
		Magnification factor $q = 2$, i.e. dimension $n = 80$					
		FS, $N = 405769$		Diamonds, $N = 405769$		Goldhill, $N = 15625$	
	d	E-step	M-step	E-step	M-step	E-step	M-step
GMM	-	10.91	0.06	10.91	0.06	0.44	0.06
PCA-GMM, $\sigma = 0.02$	20	7.25	0.74	7.42	0.57	0.28	0.54
	12	6.58	0.59	6.53	0.51	0.25	0.46
	4	6.18	0.56	6.17	0.52	0.24	0.48
PCA-GMM, learned σ	20	7.28	0.54	7.41	0.54	0.28	0.54
	12	6.59	0.47	6.53	0.45	0.25	0.47
	4	6.20	0.47	6.17	0.44	0.24	0.51
HDDC [3]	20	7.27	0.27	7.44	0.27	0.28	0.26
	12	6.64	0.26	6.64	0.26	0.25	0.26
	4	6.27	0.27	6.23	0.26	0.24	0.26

		Magnification factor $q = 4$, i.e. dimension $n = 272$					
		FS, $N = 100489$		Diamonds, $N = 100489$		Goldhill, $N = 3721$	
	d	E-step	M-step	E-step	M-step	E-step	M-step
GMM	-	17.15	0.06	17.11	0.06	0.90	0.06
PCA-GMM, $\sigma = 0.02$	20	8.65	3.54	8.68	2.03	0.44	1.83
	12	8.17	2.73	8.15	1.99	0.42	1.75
	4	7.95	2.10	7.94	2.49	0.41	1.91
PCA-GMM, learned σ	20	8.65	1.92	8.70	1.83	0.44	1.87
	12	8.17	1.99	8.17	1.74	0.42	1.74
	4	7.94	2.17	7.93	1.76	0.41	1.72
HDDC [3]	20	8.65	1.53	8.71	1.54	0.44	1.52
	12	8.16	1.54	8.14	1.53	0.42	1.52
	4	7.95	1.54	7.96	1.54	0.41	1.52

TABLE 2. Average execution time (in seconds) for the E-step and M-step in the EM algorithm for estimating the parameters of the mixture models.

	d	FS	Diamonds
Nearest neighbor	-	30.10	26.25
GMM	-	33.32	30.71
PCA-GMM, $\sigma = 0.02$	60	33.38	30.83
	40	33.36	30.75
	20	33.25	30.17
HDDC [3]	60	33.23	30.49
	40	33.24	30.29
	20	33.02	29.47

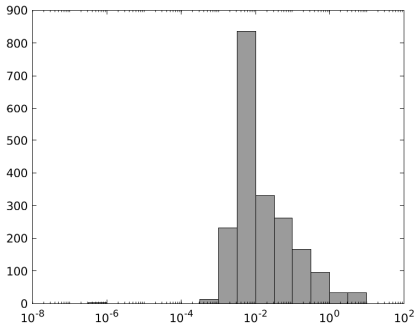
TABLE 3. PSNRs of the reconstructions of artificially downsampled 3D images using either nearest neighbor interpolation, GMM or PCA-GMM for different choices of d. The magnification factor is set to $q = 2$. As in the 2D case, PCA-GMM with small d produces results almost as good as GMM, but with a much lower dimensionality.



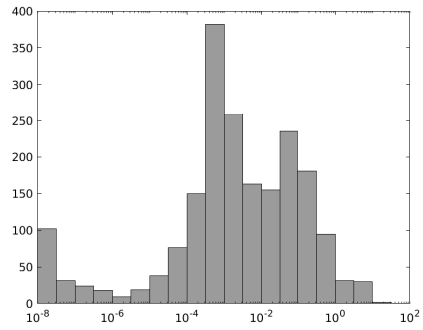
(A) Diamonds, $q = 2$ (B) Diamonds, $q = 4$ (C) FS, $q = 2$ (D) FS, $q = 4$

FIGURE 2. Reconstructions of 2D low resolution images. First row: ground truth, second row: low resolution, third row: reconstruction with GMM, fourth row: reconstruction with PCA-GMM and $d = 20$, fifth row: reconstruction with PCA-GMM and $d = 12$. The larger of d, the closer is the result of PCA-GMM to GMM.

As future work, apart from the mathematical analysis of the EM algorithm with approximate M-step, we intend to work on the robustness of the method. This could be done by using a robust PCA [22], and also by making the model invariant to contrast changes, see, e.g. [9]. Further, we aim to deal with material examples,



(A) FS with magnification $q = 4$



(B) SiC Diamonds with magnification $q = 4$

FIGURE 3. Histograms of the eigenvalues of Σ_k, $k = 1, ..., K$ for the PCA-GMM with fixed $\sigma = 0.02$ for $d = 20$.

		FS		Diamonds	
	d	E-step	M-step	E-step	M-step
GMM	-	717.91	0.07	718.13	0.07
PCA-GMM, $\sigma = 0.02$	60	338.22	12.29	337.44	17.49
	40	327.34	9.73	324.93	13.87
	20	320.00	7.85	319.46	9.80
HDDC [3]	60	337.29	4.15	337.42	4.16
	40	327.11	4.19	324.95	4.15
	20	320.03	4.20	319.07	4.15

TABLE 4. Average execution time (in seconds) of the E-step and M-step in the EM algorithm for estimating the parameters of the mixture models.

where we do not subsample the images in a synthetic way. In particular, we will not know the subsampling operator. Within ITN MUMMERING such measurements were taken, but require to undergo an advanced registration process.

Finally, we are aware of deep learning techniques for superresolution, see, e.g. [15, 30]. We will consider such approaches in the future which would also benefit from dimensionality reduction, in particular in 3D.

Acknowledgments. Funding by the German Research Foundation (DFG) within the project STE 571/16-1 as well as by the French Agence Nationale de la Recherche (ANR) under reference ANR-18-CE92-0050 SUPREMATIM, is gratefully acknowledged. The EU Horizon 2020 Marie Sklodowska-Curie Actions Innovative Training Network MUMMERING (MUltiscale, Multimodal and Multidimensional imaging for EngineeRING, Grant Number 765604) is also acknowledged. Further, we acknowledge the Paul Scherrer Institut, Villigen, Switzerland for provision of synchrotron radiation beamtime at the TOMCAT beamline X02DA of the SLS and would like to thank Federica Marone Welford for assistance.

REFERENCES

[1] H. Attouch and J. Bolte, On the convergence of the proximal algorithm for nonsmooth functions involving analytic features, *Math. Program.*, **116** (2009), 5–16.

[2] J. Bolte, S. Sabach and M. Teboulle, Proximal alternating linearized minimization for nonconvex and nonsmooth problems, *Math. Program.*, **146** (2014), 459–494.

[3] C. Bouveyron, S. Girard and C. Schmid, High-dimensional data clustering, *Comput. Statist. Data Anal.*, **52** (2007), 502–519.

[4] C. L. Byrne, *The EM Algorithm: Theory, Applications and Related Methods*, Lecture Notes, University of Massachusetts, 2017.

[5] S. Chrétien and A. O. Hero, Kullback proximal algorithms for maximum-likelihood estimation, *IEEE Trans. Inform. Theory*, **46** (2000), 1800–1810.

[6] S. Chrétien and A. O. Hero, On EM algorithms and their proximal generalizations, *ESAIM Probab. Stat.*, **12** (2008), 308–326.

[7] C. A. Deledalle, J. Salmon and A. Dalalyan, Image denoising with patch based PCA: Local versus global, *Proceedings of the British Machine Vision Conference*, **25** (2011), 1–25.

[8] A. P. Dempster, N. M. Laird and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *J. Roy. Statist. Soc. Ser. B (Methodological)*, **39** (1977), 1–38.

[9] J. H. Fitschen, K. Losch and G. Steidl, Unsupervised multi class segmentation of 3d images with intensity inhomogeneities, *Journal of Visual Communication and Image Representation*, **46** (2017), 312–323.

[10] M. Hasannasab, J. Hertrich, F. Laus and G. Steidl, Alternatives to the EM algorithm for ML estimation of location, scatter matrix, and degree of freedom of the student t distribution, *Numer. Algorithms*, **87** (2021), 77–118.

[11] J. Hertrich and G. Steidl, Inertial stochastic PALM (iSPALM) and applications in machine learning, preprint, 2020, arXiv:2005.02204.

[12] N. J. Higham, Matrix nearness problems and applications, In *Applications of Matrix Theory (Bradford, 1988)*, Inst. Math. Appl. Conf. Ser. New Ser., 22, Oxford Univ. Press, New York, 1989, 1–27.

[13] N. J. Higham and R. S. Schreiber, Fast polar decomposition of an arbitrary matrix, *SIAM J. Sci. Statist. Comput.*, **11** (1990), 648–655.

[14] A. Houdard, C. Bouveyron and J. Delon, High-dimensional mixture models for unsupervised image denoising (HDMI), *SIAM J. Imaging Sci.*, **11** (2018), 2815–2846.

[15] T. Klatzer, D. Soukup, E. Kobler, K. Hammernik and T. Pock, Trainable regularization for multi-frame superresolution, *Pattern Recognition*, Lecture Notes in Comput. Sci., 10496 Springer, Cham., 2017, 90–100.

[16] F. Laus, *Statistical Analysis and Optimal Transport for Euclidean and Manifold-Valued Data*, Ph.D Thesis, TU Kaiserslautern, 2019.

[17] E. Lehmann and H. Scheffé, Completeness, similar regions, and unbiased estimation: Part I, *Sankhyā*, **10** (1950), 305–340.

[18] S. Łojasiewicz, Une propriété topologique des sous-ensembles analytiques réels, In *Les Équations aux Dérivées Partielles (Paris, 1962)*, Éditions du Centre National de la Recherche Scientifique, Paris, 1963, 87–89.

[19] S. Łojasiewicz, Sur la géométrie semi- et sous-analytique, *Ann. Inst. Fourier (Grenoble)*, **43** (1993), 1575–1595.

[20] G. J. McLachlan, D. Peel and R. Bean, Modelling high-dimensional data by mixtures of factor analyzers, *Comput. Statist. Data Anal.*, **41** (2003), 379–388.

[21] X.-L. Meng and D. Van Dyk, The EM algorithm - an old folk-song sung to a fast new tune, *J. Roy. Statist. Soc. Ser. B*, **59** (1997), 511–567.

[22] S. Neumayer, M. Nimmer, S. Setzer and G. Steidl, On the rotational invariant l_1-norm PCA, *Linear Algebra Appl.*, **587** (2020), 243–270.

[23] S. Parameswaran, C. Deledalle, L. Denis and T. Q. Nguyen, Accelerating GMM-based patch priors for image restoration: Three ingredients for a 100x speed-up, *IEEE Trans. Process.*, **28** (2019), 687–698.

[24] K. Pearson, On lines and planes of closest fit to systems of points in space, *Philosophical Magazine*, **2** (1901), 559–572.

[25] T. Pock and S. Sabach, Inertial proximal alternating linearized minimization (iPALM) for nonconvex and nonsmooth problems, *SIAM J. Imaging Sci.*, **9** (2016), 1756–1787.

[26] P. Sandeep and T. Jacob, Single image super-resolution using a joint GMM method, *IEEE Trans. Image Process.*, **25** (2016), 4233–4244.
[27] C. Sutour, C.-A. Deledalle and J.-F. Aujol, Estimation of the noise level function based on a nonparametric detection of homogeneous image regions, *SIAM J. Imaging Sci.*, **8** (2015), 2622–2661.
[28] M. E. Tipping and C. M. Bishop, Mixtures of probabilistic principal component analyzers, *Neural Computation*, **11** (1999), 443–482.
[29] S. Vaucher, P. Unifantowicz, C. Ricard, L. Dubois, M. Kuball, J.-M. Catala-Civera, D. Bernard, M. Stampanoni and R. Nicula, On-line tools for microscopic and macroscopic monitoring of microwave processing, *Physica B: Condensed Matter*, **398** (2007), 191–195.
[30] M. Xiao, S. Zheng, C. Liu, Y. Wang, D. He, G. Ke, J. Bian, Z. Lin and T.-Y. Liu, Invertible image rescaling, preprint, 2020, arXiv:2005.05650.
[31] J.-H. Zhao and L. Philip, Fast ML estimation for the mixture of factor analyzers via an ECM algorithm, *IEEE Trans. Neural Networks*, **19** (2008), 1956–1961.
[32] D. Zoran and Y. Weiss, From learning models of natural image patches to whole image restoration, In *2011 International Conference on Computer Vision (ICCV)*, IEEE, 2011, 479–486.

Received September 2020; revised May 2021; early access September 2021.

E-mail address: j.hertrich@math.tu-berlin.de
E-mail address: lan.nguyen@math.u-bordeaux.fr
E-mail address: jean-francois.aujol@math.u-bordeaux.fr
E-mail address: dominique.bernard@icmcb.cnrs.fr
E-mail address: yannick.berthoumieu@ims-bordeaux.fr
E-mail address: abdellatif.saadaldin@icmcb.cnrs.fr
E-mail address: steidl@math.tu-berlin.de