Department of Electrical Engineering
Indian Institute of Technology, Bombay

Recovery of Joint Probability Distribution from one-way marginals Using Low Rank Tensors and Random Projections

Jian Vora, 170100026

Faculty Mentors
Prof. Ajit Rajwade & Prof. Rajbabu Velmurugan

May, 2021

Contents

1. Abstract

Given a set of N random variables $X_1, X_2, ..., X_N$, estimating the joint density $p(X_1, X_2, ..., X_N)$ is a fundamental problem in many fields such as statistics and machine learning. This probabilistic interpretation can help us make may inferences to help us aid take better decisions for the problem in hand. For a simple example, take X_1 as the time taken to drive to a particular destination and X_i, $2 \leq i \leq N$ denote a scalar number indicating the traffic on the $N-1$ streets. We would want to make predictions such as the minimum time needed for reaching the destination given we somehow know the traffic profiles of all the streets, i.e. more formally, we would want to find the following : min $p(X_1|X_2, X_3, ..., X_N)$. Maximum a posteriori(MAP) estimation of the joint probability density is used in classification problems. Probabilistic models can also be used for anomaly detection in various fields like law, medicine. Capturing the joint density is helpful as we can infer a variety of queries of such form easily(?). Most the work done can be classified in a few categories namely:

1. If the realisations of these random variables are discrete, then standard histogramming would work but it requires a large number of samples (exponential in N) to make the estimate of the joint close to the actual density and is clearly not scalable for large N. In the big data era, it is very common for data to be high dimensional (an image of size 200 $\times$200 is 40k dimensional), and hence such a naive approach for modelling densities without accounting for the inherent structure which data posseses fails in these scenarios.

2. A parametric model for estimating the underlying distribution. A Gaussian mixture model is the simplest of a capturing the probability density by modelling it as a mixture of gaussians. This is indeed shown to be a universal approximator, i.e. with enough number of components, it can approximate any density perfectly. This however is not very practical to use because the number of components increases very rapidly for even commonly occuring densities. The expressive power of these models is limited.

3. Bayesian networks or Markov random fields which explicitly model some (in)dependencies between the set of random variables and hence scalable to large data however require a lot of approximation techniques of a variety of common queries. These techniques model a directed/undirected acyclic graph with the nodes representing the random variables and edges with weights representing the conditional probabilities. Although these are the most popular methods for current sized datasets, learning the structure of these graphs is arduous coupled with a variety of approximation algorithms to perform inference once these models are learned.

Given the relevance of the problem of density estimation, we would want to find the density of the N variables while still saving us from the curse of dimensionality. Thus, density estimation without assuming some structure in the data is mission impossible. This report shall aim to answer the following questions pertinent to density estimation:

1. What is a good prior on the joint probability density tensor which holds true for a variety of real-world datasets?

2. Given just marginals of the joint density, can we reconstruct the entire tensor using the prior as mentioned in (1)?

3. Can transforming the data to a different space help us in the pursuit of the above goals?

This report is organized as follows - Chapter 2 introduces the canonical polyadic model which we shall use to model the probability tensor and the link between radon tranform with density estimation. Chapter 3 introduces the algorithm to reconstruct the joint tensor given the marginals in the radon space. Chapter 4 includes some preliminary results of our algorithm on recovering 2-way joint densities and Chapter 5 includes empirical results for recovering high dimensional densities. Chapter 6 concludes the discussion and mentions some avenues for future work. Chapter 7 contains ideas for extending methods from joint distribution estimation on discrete data to continuous data. Chapter 8 contains relevant references on the top of which this work is built.

2. Probability Tensor Decomposition

2.1 Low Rank of the Joint Probability Tensor

Going back to the setting of N random variables $X_1, X_2, ..., X_N$, we only consider the setting where each random variable X_i takes a discrete number of, say I_i, states. Thus we want to model $p(X_1, X_2, ..X_N)$ which is an $N-$way tensor(we shall refer to this as $\boldsymbol{X}$ henceforth) which each axis having a length of I. Thus the sum of entries of $\boldsymbol{X}$ is one with $\boldsymbol{X} \in {\mathbb{R}_+}^{I_1 \times I_2 \times I_3 ... \times I_N}$ and our aim is recovering this core pmf tensor from some sample realisations of the set of random variables. More specifically, we have $\boldsymbol{X}(x_1, x_2, ...x_N) = p(X_1 = x_1, X_2 = x_2, ..X_N = x_N)$. One structure which this core tensor $\boldsymbol{X}$ follows is that of a low-rank. This is a rational assumption in the sense that for much real-world datasets, most of the random variables are reasonably (in)dependent. For example, in the case of images, we know neighbouring pixels values are highly correlated while value of far off pixels are almost independent. This is also true for a large variety of temporal data where realisations of current random variables are independent to a large extend to values far off back in time which is incorporates using a Markov model. Thus, our aim is recovering the tensor $\boldsymbol{X}$ from partial entries exploiting the low-rankness of this tensor. To make this assumption explicit, we see how this low-rankness assumption holds for the case of just 2 random variables.

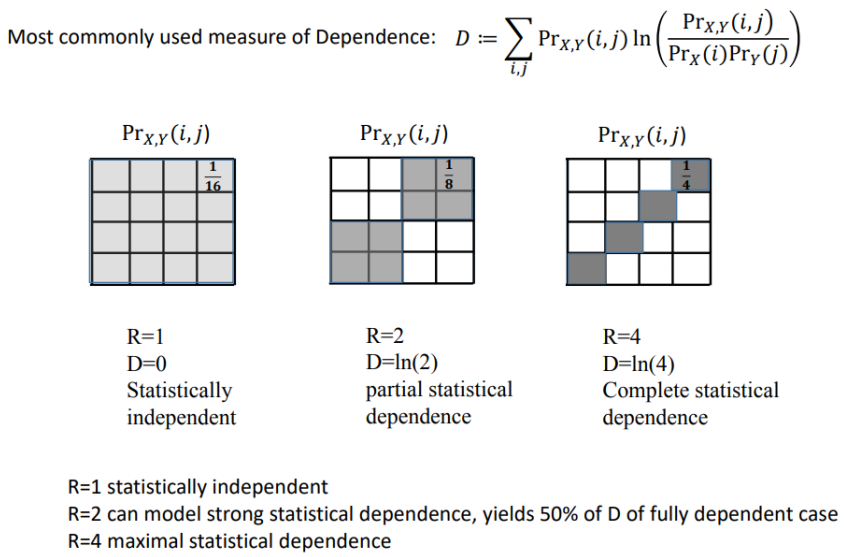

Figure 2.1: Illustration of low-rankness of the core probability tensor, this work is in the regime of $R = 2$, i.e. partial statistical dependence. Credits: [7]

Our problem formulation is that of a low rank tensor recovery from partially observable information about the tensor entries. We employ a Canonical Polyadic Decomposition on $\boldsymbol{X}$ to force low-rankess which is elaborated in the subsequent section.

2.2 Canonical Polyadic Decomposition of a Tensor

$N-$way tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times ... \times I_N}$ admits a Canonical Polyadic Decomposition(CPD) if it can be decomposed as a sum of F rank-1 tensors.Let $a \otimes b$ denote the outerproduct of two vectors, then formally the CPD model can be stated as follows:

$$\boldsymbol{X} = \sum_{f=1}^{f=F} \boldsymbol{\lambda}(f) \boldsymbol{A_1}(:, f) \otimes \boldsymbol{A_2}(:, f) \otimes \boldsymbol{A_3}(:, f) \otimes ... \otimes \boldsymbol{A_N}(:, f)$$

F is the smallest number for which such a decomposition exists. Thus whenever we refer to a tensor $\boldsymbol{X}$ having rank F, it implies that $\boldsymbol{X}$ can be decomposed into F rank one tensors using the above CPD model. $\boldsymbol{\lambda} \in {\mathbb{R}_+}^F$ and $\boldsymbol{A_i} \in {\mathbb{R}_+}^{I_i \times F}$ and $[\boldsymbol{\lambda}, \boldsymbol{A_1}, \boldsymbol{A_2}, ..., \boldsymbol{A_N}]$ can uniquely determine the core tensor. $\boldsymbol{A_i}$s are referred to as mode latent factors in the subsequent parts of this report. Hence recovering the core pmf tensor $\boldsymbol{X}$ is equivalent of recovering the mode latent factors. The above model has various forms as follows:

$$\boldsymbol{X} = \sum_{f=1}^{f=F} \boldsymbol{\lambda}(f) \prod_{n=1}^{n=N} \boldsymbol{A_n}(i_n, f)$$

$$\text{vec}(\boldsymbol{X}) = (\boldsymbol{A_N} \otimes \boldsymbol{A_{N-1}} \otimes ... \otimes \boldsymbol{A_1}) \boldsymbol{\lambda}$$

There is link between the CPD model and the naive bayes model. Assume a latent variable H taking F distinct states, then the same CPD model as described above can be formulated in the following manner:

$$Pr(X_1 = i_1, X_2 = i_2, .., X_N = i_n) = \sum_{f=1}^{f=F} Pr(H = f) \prod_{n=1}^{n=N} Pr(X_n = i_n | H = f)$$

Thus the mode latent factors have an interpretation of conditional densities. Thus, comparing with the original CPD model, we get $\boldsymbol{\lambda}(f) = Pr(H = f)$ and $\boldsymbol{A_n}(i_n, f) = Pr(X_n = i_n | H = f)$. Thus, as it is clearly evident, the entries of $\boldsymbol{\lambda}$ and each column of the mode latent factor should sum to one to be valid densities along with the non-negativity constraints. F is an important hyperparameter which explores how much dependency between the variables would we like to model. Typically, we want $F << min_k \prod_{n \neq k} I_n$, for such a decompositon to be meaningful to capture low-rankness of the tensor. This is also reasonable in practice as random variables are not completely dependent. We now see when is the CPD decomposition unique in the next section.

2.3 Uniqueness of the Canonical Polyadic Decomposition

In this section, we see the various constraints on F, the rank of the tensor to see when is the CPD decomposition unique. Here, uniqueness refers to essential uniqueness defined as:

Lemma 1: For a tensor $\boldsymbol{X}$ of rank F, we say that a decomposition $\boldsymbol{X}= [\boldsymbol{\lambda}, \boldsymbol{A_1}, \boldsymbol{A_2}, .., \boldsymbol{A_N}]$ is essentially unique if the factors are unique up to a common permutation and scaling / counter-scaling of columns of the mode latent factors.

We clearly do not have the scaling ambiguity if we the non-negativity and sum to one constraints on the columns of the latent factors. Let $\boldsymbol{X} = [\boldsymbol{A_1}, \boldsymbol{A_2}, \boldsymbol{A_2}]$, where $\boldsymbol{A_1} \in \mathbb{R}^{I_1 \times F}$, $\boldsymbol{A_2} \in \mathbb{R}^{I_2 \times F}$, and $\boldsymbol{A_3} \in \mathbb{R}^{I_3 \times F}$ with $I_1 \leq I_2 \leq I_3$ without loss of generality, then:

Lemma 2: If $\min(I_1, I_2) \geq 3$ and $F \leq I_3$, then, $\text{rank}(\boldsymbol{X}) = F$ and the decomposition of $\boldsymbol{X}$ is essentially unique, almost surely, if and only if $F \leq (I_1 - 1)(I_2 - 1)$

3. Tomography in Density Estimation

3.1 Random Linear Projections

For a given vector $X \in \mathbb{R}^N$, we define the following operation as random linear projections:

$$Y = \Phi X$$

where $\Phi \in \mathbb{R}^{M \times N}$ is composed of entries drawn i.i.d from a standard normal Gaussian or a bernoulli $\{-1, 1\}$ distribution with the columns scaled accordingly. If $M < N$, then we call the projection as compressive. Such a forward model is commonly used for compressive sensing of signals and random projections have also been shown to do dimensionality reduction much more computationally efficiently as compared to some other expensive methods like PCA.

3.2 Preliminaries from Tomography

3.2.1 Radon Transform of a Function

The Radon transform of a 2-D function $f(x, y)$ is defined as:

$$R(r, \theta)[f] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y)\delta(r - xcos\theta - ysin\theta)dxdy$$

where r is the perpendicular distance of a line from the origin and θ is the angle between the line and the y-axis. According to the Fourier slice theorem, this transformation is invertible with minimal reconstruction loss and that transform is called the Inverse Radon operation.

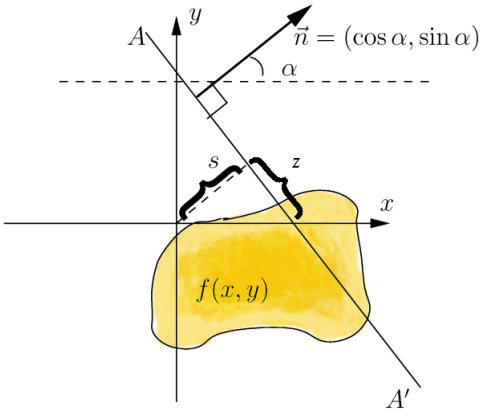

Figure 3.1: Graphical Illustration of the Radon Transform. Source: Wikipedia

The radon transform $R(r,\theta)[f]$ can be inverted to reconstruct the function $f(x,y)$ using many computationally efficient algorithms with the filtered back projection(FBP) algorithm being the most prominent which is described in the next section.

3.2.2 Filtered Back Projection(FBP) Algorithm

Analytical reconstruction methods consider continuous tomography. So the problem is: given the radon transform we want to recover the object described in (x,y) coordinates. The most common type of image analytical reconstruction is Filtered Back Projection (FBP). Fourier slice theorem states that for a $2-D$ function $f(x,y)$, the $1-D$ Fourier transforms of the Radon transform along r, are the $1-D$ radial samples of the $2-D$ Fourier transform of $f(x,y)$ at the corresponding angles. Using the Fourier slice theorem, we can see that if we are given the Fourier transform of a projection at enough angles, the projections could be assembled into a complete estimate of the two-dimensional transform and then simply inverted to arrive at an estimate of the object. This procedure is known as the Filtered Back Projection algorithm. It has been shown to be extremely accurate and amenable to fast implementation. The back projection algorithm mathematically expressed as:

$$\hat{f}(x,y) = \int_0^{\pi} R(r,\theta) d\theta$$

The filtered back projection algorithm is also based on a similar principle but the signal $R(r,\theta)$ is modulated by a filter. Let $R(v,\theta)$ be the $1-D$ fourier transform of $R(r,\theta)$ keeping the θ fixed. Then for a variety of filters with the transfer function $H(v)$, we have:

$$\hat{f}(x,y) = \int_0^{\pi} \int_{-\infty}^{+\infty} R(v,\theta) H(v) dv d\theta$$

There are several famous choices for the filter H such as the Ram-Lak filter, Cosine filter and the Shepp-Logan filter. Thus, whenever we refer to inverse radon tranform, we refer to the filtered back projection algorithm with a filter implicitly assumed.

3.3 Random Projections for Density Estimation

Consider a random vector $X \in \mathbb{R}^N$ and another random vector $\theta \in \mathbb{R}^N$. θ can be considered as one column of the random matrix Φ defined earlier in this chapter. We are interested in estimating $p(X)$ from random linear projections of the data. $\theta^T X$ is a scalar variable and hence we can estimate $p(\theta^T X)$ which is a 1D density estimate and is fairly easy to do using either histogramming or kernel density estimate. Let $\theta = [\theta_1, \theta_2, ..., \theta_N]$ and $X = [X_1, X_2, ..., X_N]$, then

$$p(\theta^T X = t) = p(\sum_i \theta_i X_i = t)$$

$$p(\theta^T X = t) = \sum_{a_1} \sum_{a_2} .. \sum_{a_{N-1}} p(\theta_1 X_1 = a_1, \theta_2 X_2 = a_2, ...\theta_N X_N = t - \sum_{i=1}^{i=N-1} a_i)$$

Thus the 1D density estimate of a linear projection of a data vector is infact the radon transform of the joint probability tensor taken at an angle θ. Thus, if we collect densities of the form $p(\theta_i^T X)$ for various i, then we can use concepts of inverse radon transform to reconstruct the core probability tensor. Furthermore, if we incorporate the low-rankness of the tensor, we can recover the tensor using 1D marginals in the radon space. We now investigate the usage of this method of radon transform for density estimation on just 2 random variables X_1 and X_2.

3.4 Empirical Results of Density Estimation with Tomography

Suppose we have N random variables and want to estimate $p(X_1, X_2, ...X_N)$, then on arranging then as a vector X, we take linear combinations of them through a linear ϕ operator. Let the number of projections be m and the number of samples S.

$$Y = \phi X; Y = [y_1, y_2, ..., y_m]^T$$

From S samples we get the following - $p(y_1), p(y_2), ..., p(y_m)$ and use this for getting the original $p(x_1, x_2, ...x_n)$. Certain notations are as follows:

N : Number of random variables
S : Number of samples of X
m : Number of projections
I : Number of discrete states each random variable takes
$numbins$: Number of values in which each y_i shall be discretized

Some experimental conditions the first preliminary experiment were as follows:

N : 2
S : Variable for analysis
m : Variable for analysis
I : 10
$numbins$: 16

Data Generation Process : We take a truncated 2D gaussian (from -2 to +2 in either directions) with an identity covariance matrix and 0 mean. This is put into 10 bins as cummulative interval measures to get the PDF matrix. PDF of y_i is generated using **histograms** with the $numbins$ parameter.

Implementation details : iradon implemented with Ram-Lak filter and without any filter was studies angles obtained by using 'atand' command for each row of ϕ

Evaluation Metric: Jensen-Shannon Divergence

$$JS(P,Q) = \frac{KL(P||(P+Q)/2) + KL(Q||(P+Q)/2)}{2}$$

```
Number of Projections = [5, 20, 50, 100, 200, 500, 1000]
Number of Samples     = [10, 100, 1000, 5000, 10000, 20000]
```

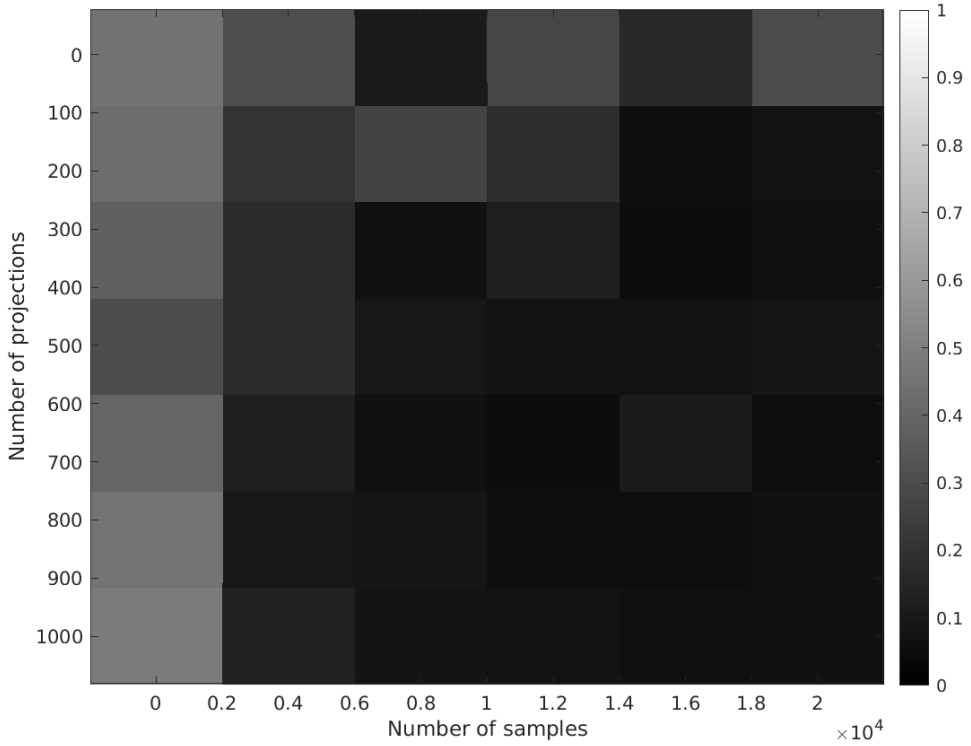

Figure 3.2: JSD (smaller the better) errors using Ram-Lak Filter

Comparision with other 2D density estimators

Number of samples	JSD Radon	JSD Histogramming
100	0.0608	0.1902
500	0.0478	0.0529
1000	0.0221	0.0203
10000	0.0107	0.0019

Table 3.1: Reconstruction Comparisons with 1000 projections for Radon

We conclude that the radon approach for simple density estimation performs better than standard histogramming approach especially when the number of samples are less. Further, we

need the underlying densities to be continuous and should have decaying fourier coefficients in order to be able to apply inverse radon transform efficiently. Thus, most the this work is concerned with the low sample regime with continuous densities. Next, to validate our claim, we look at capturing the densities of a variety of gaussian mixture models.

Data Generation Process : We take a truncated 2D gaussian (from -8 to +8 in either directions) with varying number of components and random means and full covariances while ensuring the positive semi-definite condition. This is put into 500 bins as cummulative interval measures to get the PDF matrix. PDF of y_i is generated using histograms with the *numbins* parameter.

```
Number of Projections = [2,10,40,100,500]
Number of Samples     = [100,500,1000,5000,10000,20000]
```

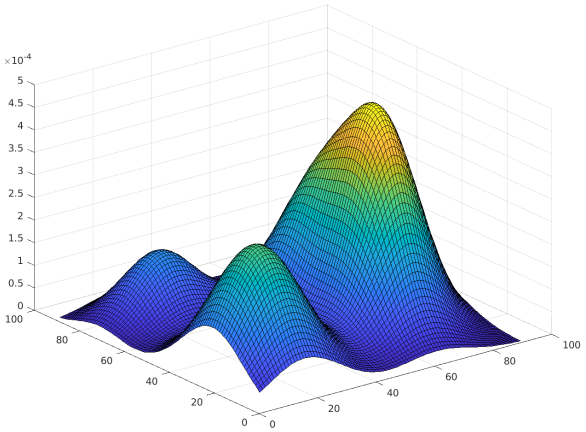

Figure 3.3: GMM 1 (4 components)

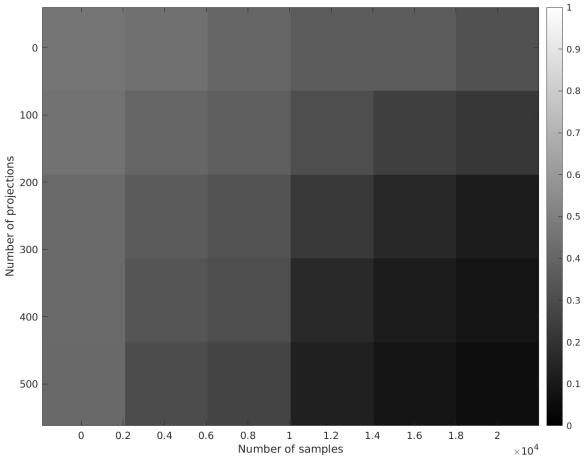

Figure 3.4: GMM 1 Errors

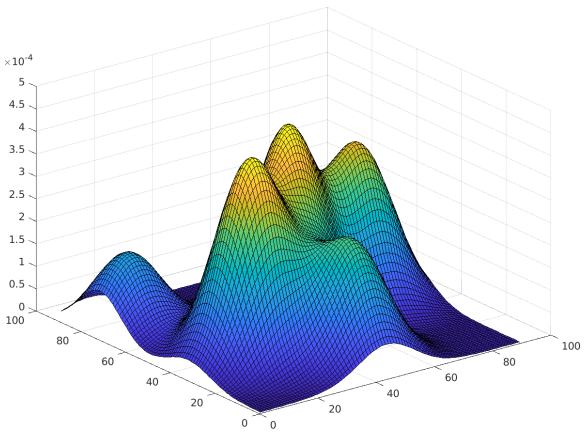

Figure 3.5: GMM 2 (9 components)

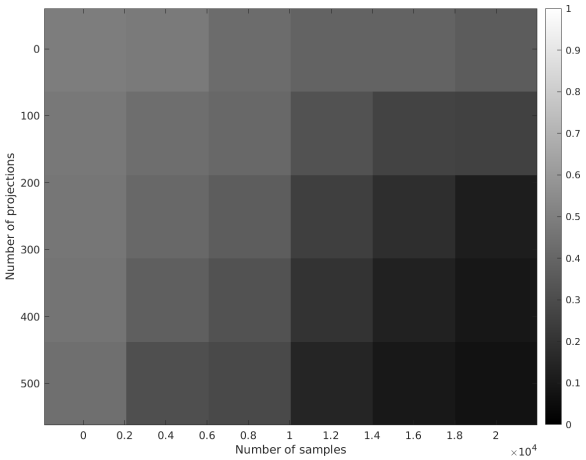

Figure 3.6: GMM 2 Errors

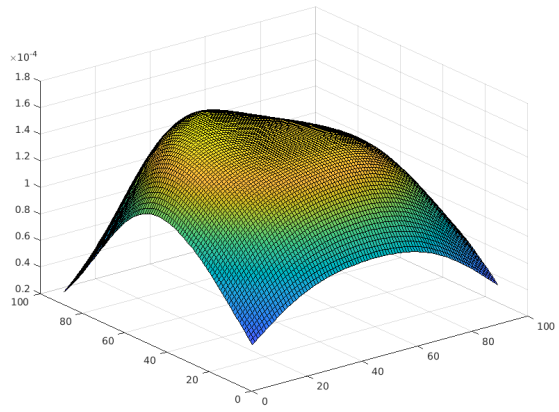

Figure 3.7: GMM 3 (4 components)

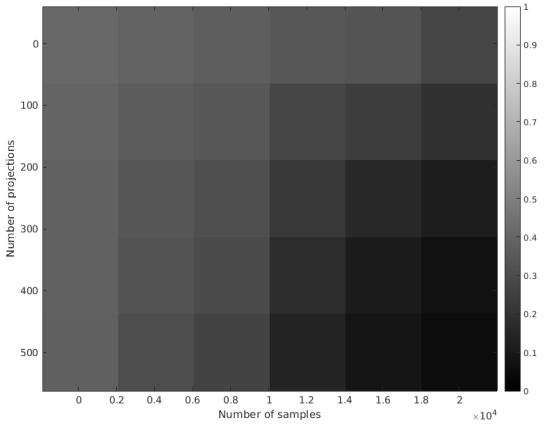

Figure 3.8: GMM 3 Errors

Thus, in this chapter, we introduced the link between tomography and density estimation. The take away message is that by estimating the densities of random projection of data, we can reconstruct the original density using the inverse radon transform. We presented results on simple 2D densities which will be used later to reconstruct the core joint pmf tensor using the low rank canonical polyadic decomposition model which is described in the subsequent sections.

4. Literature Review

4.1 Joint Density Estimation from Random Projections

Multidimensional Density Estimation by Tomography by Yudi Pawitan was the first paper linking random projections of data with density estimation. It formulated the problem of pmf estimation as inverting the radon transform. However, there were two major things lacking in this formulation:

1. Experiments were conducted for only two-dimensional joint densities as inverting the radon transform is costly as the number of dimensions keep on increasing.

2. Did not leverage the property that joint density tensors have a low-rank

4.2 Joint Density Estimation from 3-way Marginals

This was the first work concerning density estimation from marginals by Kargas et.al. In this paper, they first estimate 3-way marginals of the form $p(X_i, X_j, X_k)$ from the data using standard histogramming. Once we have the 3-way marginals, we can decompose that uniquely to get the corresponding mode factors and once we have all the model factors, then we can uniquely determine the core pmf tensor. F is to be such that the decomposition of the 3-way marginals is unique in accordance with the lemma stated in Chapter 2. They use the ADMM to solve the proposed optimization problem of joint factorisation. Their algorithm can be described as follows:

Procedure: Joint PMF Recovery From Triples

1. Estimate $\boldsymbol{X}_{j,k,l}$ from the data
2. Jointly factor $\boldsymbol{X}_{j,k,l} = [\boldsymbol{\lambda}, \boldsymbol{A_j}, \boldsymbol{A_k}, \boldsymbol{A_l}]$ using the CPD model of rank F
3. Once all the latent factors are available using the joint factorisation, reconstruct using

$$\boldsymbol{X} = \sum_{f=1}^{f=F} \boldsymbol{\lambda}(f)\boldsymbol{A_1}(:, f) \otimes \boldsymbol{A_2}(:, f) \otimes \boldsymbol{A_3}(:, f) \otimes ... \otimes \boldsymbol{A_N}(:, f)$$

For solving the joint factorisation problem, we use the following least squares optimization problem using alternating minimization:

$$\min_{\{\boldsymbol{A_n}\},\boldsymbol{\lambda}} \sum_j \sum_{k>j} \sum_{l>k} \|\boldsymbol{X}_{j,k,l} - [\boldsymbol{\lambda}, \boldsymbol{A_j}, \boldsymbol{A_k}, \boldsymbol{A_l}]\|_F^2$$

subject to appropriate normalization of $\boldsymbol{\lambda}$ and $\boldsymbol{A_n}, n = 1, 2, 3, ..., N$

4.3 Joint Density Estimation using Expectation Maximization

The formulation of CPD model is similar to that of a gaussian mixture model where instead of the mixture weights, we have $\boldsymbol{\lambda}$ and instead of estimating the means and covariances of the gaussians, we have to estimate the mode latent factors. Thus, work by Yeredor et.al. tries to find the parameters by maximising the likelihood of observing the data. It uses an EM update like the one used for training the GMMs.

$$\log\ P(y[1], y[2], ..., y[T]) = \sum_{t=1}^{t=T} \log\ \sum_{f=1}^{f=F} \lambda_f \prod_{n=1}^{n=N} \boldsymbol{A_n}(y[t], f)$$

Thus, we formulate the following optimization problem for minimizing the negative log-likelihood:

$$\min_{\{\boldsymbol{A_n}\},\boldsymbol{\lambda}} - \sum_{t=1}^{t=T} \log\ \sum_{f=1}^{f=F} \lambda_f \prod_{n=1}^{n=N} \boldsymbol{A_n}(y[t], f)$$

subject to appropriate normalization of $\boldsymbol{\lambda}$ and $\boldsymbol{A_n}, n = 1, 2, 3, ..., N$

4.4 Joint Density Estimation from 2-way Marginals

In this work by Ibrahim et.al., they try to recover the joint density from 2-way marginals. Given the data matrix, we first estimate the 2-way marginals $\boldsymbol{X}_{j,k}$ from histogramming and then try to recover the mode latent factors in the following manner:

$$\boldsymbol{X}_{j,k} = \boldsymbol{A_j} \boldsymbol{D}(\boldsymbol{\lambda}) \boldsymbol{A_k^T}$$

However, we cannot directly use non-negative matrix factorisation as F is generally greater than I, the number of states each variable takes and hence the matrix factorisation wont't be unique. We however, can split the N variables into 2 sets with M being the index of the split. Thus we have 2 splits namely $\{1, 2, ..., M\}$ and $\{M+1, M+2, ..., N\}$ We then arrange the 2-way marginals in the form a matrix $\boldsymbol{A}$ where the block $\boldsymbol{A_{i,j}} = \boldsymbol{X_{i,M+i}}$ where NMF can be applied to get the mode factors. This matrix $\boldsymbol{A} \in \mathbb{R}^{MI \times (N-M)I}$.

They use the Successive Projection Algorithm(SPA) for carrying out the NMF. After this, they also propose running EM with the initial values of the EM algorithm as the output of the SPA algorithm which would be a better initialisation and the optimization would not get stuck in a local minima which vanilla EM is infamous for.

5. High-Dimensional Density Estimation for Discrete Data

5.1 Problem Statement and Algorithm

Consider N discrete RVs $\{X_i\}_{i=1}^{i=N}$ each existing in I different states. Our aim is to estimate the joint PMF $p(X_1, X_2, .., X_N)$ where $p(.)$ is a low-rank tensor which follows the CPD equation. However, recovering the tensor p from its Radon projections taken in general directions $\{\boldsymbol{\phi}_m \in \mathbb{R}^N\}$ is an immensely costly operation. Hence we consider only sparse direction vectors $\{\boldsymbol{\phi}_m\}$ with just two non-zero entries (i.e. we consider linear combinations of just two RVs at a time), which facilitates speedy FBP implementation. Define the set $\mathcal{B} \triangleq \{(j,k) : 1 \leq j < k \leq N\}$. For M random vectors $\boldsymbol{\phi}_m \in \mathbb{R}^2, 1 \leq m \leq M$, and for $\boldsymbol{X_{j,k}} \triangleq [X_j, X_k]^t$ for $(j,k) \in \mathcal{B}$, we obtain $M|\mathcal{B}|$ random projections of the form $\boldsymbol{\phi}_m^t \boldsymbol{X_{j,k}} \in \mathbb{R}$. We wish to recover the PMF tensor from the 1D PMFs of the form $p(\boldsymbol{\phi}_m^t \boldsymbol{X_{j,k}})$. Define $\boldsymbol{Z}_{j,k}$ as the joint probability $p(X_j, X_k)$ for the two RVs X_j and X_k. On fixing j and k and stacking M one-dimensional PMFs $p(\boldsymbol{\phi}_m^t \boldsymbol{X_{j,k}})$ row-wise, we obtain a matrix $\boldsymbol{Y}_{j,k} \in \mathbb{R}^{M \times I}$. In case of infinite samples,

$$\boldsymbol{Y}_{j,k} = \mathfrak{R}(\boldsymbol{Z}_{j,k}) = \mathfrak{R}(\boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T), \tag{5.1}$$

where $D(.)$ is the diagonal operator, and $\mathfrak{R}$ stands for the Radon transform in multiple directions $\{\boldsymbol{\phi}_m\}_{m=1}^M$. However the 1D PMFs, and hence each $\boldsymbol{Y}_{j,k}$ can only be *estimated*. For each element in set $\mathcal{B}$, we perform random projections and estimate *one dimensional* PMFs empirically from the data using histogramming, and thus assemble $\boldsymbol{Y}_{j,k}$. The goal is now to determine the underlying mode factors $\boldsymbol{\lambda}, \boldsymbol{A_1}, \boldsymbol{A_2}, ..., \boldsymbol{A_N}$ (and thus the joint PMF of $\{X_i\}_{i=1}^N$), which represent these stacked 1D PMF estimates $\{\boldsymbol{Y}_{j,k}\}_{(j,k)\in\mathcal{B}}$ as faithfully as possible. To this end, we formulate the following objective function:

$$J(\{\boldsymbol{A_n}\}_{n=1}^N, \boldsymbol{\lambda}) = \sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \mathfrak{R}(\boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T)\|_F^2. \tag{5.2}$$

We cannot directly optimise $J(.)$, say via gradient descent updates, as the mode factors will not be identifiable when $F > I$ as argued in [6]. To circumvent, we introduce an auxiliary variable $\boldsymbol{Z}_{j,k}$ with the constraint $\boldsymbol{Z}_{j,k} = \boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T$ and transform it into an unconstrained problem by adding a penalty term, namely

$$\begin{aligned} J_1(\{\boldsymbol{A_n}\}_{n=1}^N, \boldsymbol{\lambda}, \{\boldsymbol{Z}_{j,k}\}) \triangleq \sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \mathfrak{R}(\boldsymbol{Z}_{j,k})\|_F^2 + \\ \rho \|\boldsymbol{Z}_{j,k} - \boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T\|_F^2. \end{aligned} \tag{5.3}$$

We perform the following three steps to obtain identifiable mode factors: (i) setting the hyper-parameter $\rho = 0$, compute $\boldsymbol{Z}_{j,k}$ from the 1D densities $\boldsymbol{Y}_{j,k}$ using least-squares, (ii) construct $\widetilde{\boldsymbol{Z}}$ from $\boldsymbol{Z}_{j,k}$, and (iii) determine the matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ as outputs of the function SPA(.) which are the factors of $\widetilde{\boldsymbol{Z}}$. The mode factors $\{\boldsymbol{A_j}\}$ which are sub-matrices of $\boldsymbol{W}$ and $\boldsymbol{H}$ will now be identifiable as shown in [6]. We further refine our estimates by choosing ρ via cross-validation, and updating the mode factors and $\boldsymbol{Z_{j,k}}$ till convergence as described in Alg. 1. The resultant mode factors obtained from optimizing $J_1(.)$ can then be used as a good initial condition for a projected gradient descent on the original objective function $J(.)$ with adaptive step-size. For Alg. 1, we assume that we have an access to a function SPA(.) which takes in $\widetilde{\boldsymbol{Z}}$ as input and outputs $\boldsymbol{W, H}$. In Alg. 1, the operators $\mathfrak{R}$ and $\mathfrak{R}^T$ are implemented as function handles using the `radon` and `iradon` functions of MATLAB respectively. *It is important to note that a procedure which stops at the end of step 4 in Alg. 1 (equivalent to FBP to obtaining $\boldsymbol{Z}_{j,k}$ from $\boldsymbol{Y}_{j,k}$, followed by the algorithm from [6]), would necessarily ignore the fact that the 2-way marginals $\boldsymbol{Z}_{j,k}$ are inter-dependent due to common mode factors.* This motivates the further steps in Alg. 1 which account for such dependencies. We additionally note that the cost function $J_1(.)$ in Eqn. 5.3 can be easily modified to include further prior information about the density such as it being piece-wise flat (true for PMFs) or smooth. Our algorithm can be viewed as 3 sequential processing blocks which refine the estimates produced by the previous steps in the pipeline. The three blocks are: G1 (lines 2-4), G2 (lines 6-13), and G3 (lines 15-20). We refer to our method as JUROR: **J**oint distribution recovery **U**sing **R**andom projections to **O**ne dimensional **R**egion.

Algorithm 1 Recovering Mode Latent Factors from 1D Densities of Random Projections

procedure JUROR
  Set $\{\boldsymbol{Z}_{j,k}\}_{(j,k)\in B}$ to be equal to argmin $\sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \mathfrak{R}(\boldsymbol{Z}_{j,k})\|_F^2$
  Assemble $\widetilde{\boldsymbol{Z}}$ using $\{\boldsymbol{Z}_{j,k}\}_{(j,k)\in B}$
  $\boldsymbol{W}^{(0)}, \boldsymbol{H}^{(0)} \leftarrow \text{SPA}(\widetilde{\boldsymbol{Z}})$
  converged $\leftarrow$ False, $\rho \leftarrow \rho_0$, $q \leftarrow 1$
  while converged == False **do**
    Fetch $\{\boldsymbol{A}_n\}_{n=1}^{n=N}, \boldsymbol{\lambda}$ from $\boldsymbol{W}^{(q-1)}, \boldsymbol{H}^{(q-1)}$
    $\boldsymbol{Z}_{j,k}^{(n)} = (\mathfrak{R}^T\mathfrak{R} + \rho I)^{-1}(\mathfrak{R}^T\boldsymbol{Y}_{j,k} + \rho \boldsymbol{A}_j D(\boldsymbol{\lambda})\boldsymbol{A}_k^T)$
    Assemble $\widetilde{\boldsymbol{Z}}$ using $\{\boldsymbol{Z}_{j,k}^{(n)}\}_{(j,k)\in B}$
    $\boldsymbol{W}^{(q)}, \boldsymbol{H}^{(q)} \leftarrow \text{SPA}(\widetilde{\boldsymbol{Z}})$
    $q = q + 1$
    if $J_1(.) < \epsilon$ **then** converged $\leftarrow$ True
  Fetch $\{\boldsymbol{A}_n^0\}_{n=1}^{n=N}, \boldsymbol{\lambda}^0$ from $\boldsymbol{W}^{(q)}, \boldsymbol{H}^{(q)}$
  converged$\leftarrow$ False, $q \leftarrow 1$
  while converged == False **do**
    for k in 1 to N **do**
      $\boldsymbol{A}_k^{(q)} \leftarrow \text{ProjectOnSimplex}(\boldsymbol{A}_k^{(q-1)} - \eta_q \frac{\partial J_1}{\partial \boldsymbol{A}_k})$
    $\boldsymbol{\lambda}^{(q)} \leftarrow \text{ProjectOnSimplex}(\boldsymbol{\lambda}^{(q-1)} - \eta_q \frac{\partial J_1}{\partial \boldsymbol{\lambda}})$
    $q = q + 1$
    if $J(.) < \epsilon$ **then** converged $\leftarrow$ True
  return $\{\boldsymbol{A_n}\}_{n=1}^{n=N}, \boldsymbol{\lambda}$

5.2 Empirical Results on Synthetic and Real Data

In this section, we present several PMF estimation results on both synthetic and real-world datasets. For synthetic data, we present results for both discrete and continuous RVs. The synthetic data are created from their mode factors, in the following manner: (1) For PMFs of discrete RVs are created, each entry in $\boldsymbol{A_i}$ is generated i.i.d. from Uniform$[0, 1]$ followed by normalizing the columns to have unit ℓ_1 norm. Elements of $\boldsymbol{\lambda}$ are generated in the same manner. (2) For continuous RVs, we consider cumulative interval measures (CIMs) instead of PDFs and represent them as tensors. For the CIM, each column of $\boldsymbol{A_i}$ is generated by sampling from sinusoidal waves of varying amplitudes, phase and frequency, at regular intervals. For synthetic data, F is known to the algorithm before hand and it tries to find the mode factors given just the data sampled from the underlying distribution. For all the experiments, the number of

random projections $M = 200$ and $|\mathcal{B}| = \binom{N}{2}$. Let $\boldsymbol{\lambda}, \{\boldsymbol{A_n}\}_{n=1}^N$ and $\widehat{\boldsymbol{\lambda}}, \{\widehat{\boldsymbol{A_n}}\}_{n=1}^N$ be the mode factors of the true and estimated PMFs/CIMs respectively. Let $\kappa \in (0, 1]$ be the fraction of observed (as opposed to missing) entries in the data-points. Since storing the tensor with I^N entries may be infeasible, we compute the mean squared estimation error in terms of the mode factors: MSE $\triangleq \sum_{i=1}^{i=N} \frac{\|\widehat{\boldsymbol{A}_i} - \boldsymbol{A}_i\|_F^2}{N} + \|\widehat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2^2$. The results are reported in Tables 5.1 and 5.3 for different number of samples (denoted by N_s) for PMF/CIM estimation. In both cases $F > I$ and $\kappa = 1$. The methods compared are (1) CTF using 3-way marginals from [2]; (2) the SPA method using 2-way marginals from [6]; (3) our technique from Alg. 1 termed JUROR with various combinations of the stages of the algorithm defined as A: $\mathsf{G1} + \mathsf{G2} + \mathsf{G3}$, B: $\mathsf{G1} + \mathsf{G3}$, C: $\mathsf{G1} + \mathsf{G2}$; and (4) the EM technique from [3] with random initial conditions referred to as RAND-EM. In Table 3 we consider the real-world scenario where not all features are observed for every data-point and present results for $\kappa = 0.8$ as the probability of observing each feature of a sample. In many real-world applications not all features are observed for every data-point. PMF/CIM estimation from marginals is particularly useful in these scenarios, because simple histogramming will discard samples with missing data. In Table 5.4 we present results with $\kappa = 0.8$ as the probability of observing each feature of a sample.

N_s	100	1000	5000	10000	50000
CTF	0.339	0.294	0.233	0.172	0.103
SPA	0.295	0.264	0.196	0.143	0.084
JUROR-A	**0.253**	**0.205**	**0.174**	0.138	0.092
JUROR-B	0.267	0.229	0.182	**0.131**	**0.087**
JUROR-C	0.262	0.217	0.185	0.140	0.098
RAND-EM	0.284	0.246	0.204	0.149	0.106

Table 5.1: MSE for PMFs for $F = 25, I = 10, N = 6, \kappa = 1$

We also study the absolute tensor errors in addition to MSE in mode factors when storing the tensor is feasible. If $\mathcal{Z}$ is the original tensor and $\hat{\mathcal{Z}}$ is the estimate of the joint distribution tensor, then we measure the mean absolute error defined as MAE $\triangleq \frac{\|\hat{\mathcal{Z}} - \mathcal{Z}\|_F^2}{\|\mathcal{Z}\|_F^2}$. The results for MAE comparisions are shown in Table 5.2.

N_s	100	1000	5000	10000
CTF	0.215	0.181	0.123	0.096
SPA	0.175	0.151	0.114	0.063
JUROR-A	**0.158**	**0.136**	**0.093**	0.054
JUROR-B	0.169	0.148	0.102	**0.048**
JUROR-C	0.162	0.139	0.108	0.059
RAND-EM	0.183	0.174	0.116	0.082

Table 5.2: MAE for PMFs for $F = 15, I = 10, N = 4, \kappa = 1$

The MSE results in Tables 5.1, 5.2, 5.3, 5.4 are obtained after averaging over 5 random runs, and with $M = 200$ Radon projections per pair of RVs. As expected, our algorithm performs better in the low-sample regime where estimating 1D marginals is much more reliable than higher-D marginals. All the methods start converging in the high-sample regime where out method isn't far from the lowest MSE. The results for high-D PMFs with $N = 15$ are presented in Table 5.5. From the above tests it is evident that G2 of the algorithm is crucial when we have less samples and the final step G3 is more effective in the high-sample regime. We also study the effect of number of projections M on density estimation using the algorithm SPA-R-A. The results are shown in Table 5.6.

N_s	1000	5000	10000	50000	100000
CTF	0.318	0.289	0.227	0.191	0.164
SPA	0.296	0.274	0.238	0.182	0.158
JUROR-A	**0.271**	**0.259**	**0.194**	**0.162**	0.149
JUROR-B	0.283	0.264	0.209	0.175	**0.137**
JUROR-C	0.276	0.268	0.204	0.179	0.155
RAND-EM	0.281	0.268	0.198	0.187	0.157

Table 5.3: MSE for CIMs for $F = 90, I = 50, N = 5, \kappa = 1$

N_s	100	1000	10000	50000
CTF	0.316	0.151	0.142	**0.092**
SPA	0.238	0.162	0.126	0.117
JUROR-A	**0.205**	**0.147**	0.131	0.103
JUROR-B	0.225	0.159	**0.118**	0.107
JUROR-C	0.218	0.154	0.126	0.114
RAND-EM	0.269	0.173	0.152	0.136

Table 5.4: MSE for PMFs for $F = 20$, $I = 15$, $N = 5$, $\kappa = 0.8$

In real-world applications where the true underlying PMF is unknown, we compared the different methods in terms of classification accuracy computed after estimating the joint PMFs of the form $p(\boldsymbol{x}, y)$, where y is the label associated with data-point $\boldsymbol{x} \in \mathbb{R}^N$. The classification is done by assigning class label $\hat{y} = \text{argmax}_y p(y|\boldsymbol{x}) = \text{argmax}_y p(\boldsymbol{x}, y)/p(\boldsymbol{x})$ to $\boldsymbol{x}$. We test our algorithms against some common classification methods on two datasets from the UCI repository - the Car (6D) and the Mushroom (22D) datasets. The value of F was set by cross-validating with the accuracy obtained on the validation split. Commonly used discriminative classifiers such as SVM-RBF and neural networks, were trained using MATLAB's classification toolbox. The train-val-test split was 70-10-20 for the car dataset and 50-20-30 for the mushroom dataset. As seen in Table 5.7, our method JUROR outperforms other techniques.

N_s	1000	5000	10000	50000	100000
CTF	0.268	0.242	0.201	**0.154**	**0.104**
SPA	0.245	0.237	0.226	0.176	0.126
JUROR-A	**0.229**	**0.208**	**0.184**	0.162	0.115
JUROR-B	0.258	0.217	0.205	0.158	0.109
JUROR-C	0.241	0.225	0.197	0.171	0.119
RAND-EM	0.248	0.226	0.217	0.197	0.143

Table 5.5: MSE for PMFs for $F = 20$, $I = 10$, $N = 15$, $\kappa = 1$

$M(\downarrow)$	100	1000	5000	10000	50000
200	0.253	0.205	0.174	0.138	0.092
150	0.264	0.214	0.188	0.151	0.106
100	0.289	0.267	0.223	0.184	0.129
50	0.384	0.357	0.305	0.264	0.213

Table 5.6: MSE for PMFs for $F = 25, I = 10, N = 6, \kappa = 1$

Algorithm	Car	Mushroom
CTF	84.92	95.13
SPA	86.45	96.01
JUROR-A	87.59	**96.72**
JUROR-B	86.37	95.12
JUROR-C	**88.38**	95.79
RAND-EM	82.68	94.65
LOGISTIC REGRESSION	82.37	95.86
SVM-RBF	78.32	95.74
NAIVE BAYES	84.39	89.67
NEURAL NET	85.16	96.37

Table 5.7: Classification Accuracies on real-world datasets

6. High-Dimensional Density Estimation for Continuous Data

6.1 Introduction and Motivation

In the previous chapters, we have seen how a PMF can be respresented as a low-rank tensor and then we can use 1D density estimates of random projections of data to recover the joint density. In this chapter we shall explore the above ideas however in estimating PDFs, i.e., when the data is continuous. Now, we would need to estimate an infinite dimensional tensor in the truly non-parametric case. In the subsequent sections, we shall look in some previous work in this space and then present the exact problem statement about learning gaussian mixture models from marginals.

6.2 Literature Review

6.2.1 Learning Mixtures of Smooth Product Distributions

One may note that mixture of smooth product distributions is essentially the CPD model where each column of the mode factors is a pdf instead of a pmf. This particular work assumes that each mode factor CDF is smooth and band-limited and thus it can be reconstructed from sufficiently closely spaced discrete samples followed by invoking the Nyquist sampling theorem.

Proposition: *A PDF that is (approximately) bandlimited with cutoff frequency ω_c can be recovered from uniform samples of the associated CDF taken π/ω_c apart.* [12]

If the CDF follows the above property, then we can simply estimate the (finite) samples of the CDF at regular intervals (which in turn can be estimated by any method discussed in the previous chapter) and then reconstruct the entire function using sinc interpolation as given by:

$$F_X(x) = \sum_{n=-\infty}^{n=\infty} F_X(nT)\text{sinc}\left(\frac{x - nT}{T}\right)$$

where $T = \pi/\omega_c$. The PDF can then be determined by differentiation, which amounts to linear interpolation of the CDF samples using the derivative of the sinc kernel. The problem of this approach is the strict assumption of the bandlimitness which does not occur even for common distributions (say a Gaussian with a small variance).

6.2.2 A Low-Rank Characteristic Function Approach for Density Estimation

In this work, they model the characteristic function (fourier transform of the PDF) as a low rank tensor from which they try to recover the original PDF.

Proposition: *Truncating the multidimensional Fourier series (sampled multivariate characteristic function) of any compactly supported random vector is equivalent to approximating the corresponding multivariate density by a finite mixture of separable densities.* [11]

Given sample realizations in the form of $\{x_i\}_{i=1}^{M}$, they do the following main steps:

1. estimate $\mathbf{\Phi}[\boldsymbol{k}] = \frac{1}{M}\sum_i e^{j2\pi \boldsymbol{k}^T x_i}$
2. fit the low rank model $\mathbf{\Phi}[\boldsymbol{k}] \approx \sum_h p_H(h) \prod_{n=1}^{n=N} \Phi_{X_n|H=h}[k_n]$
3. invert to find each column of the mode factor as $\sum_{k_n=-K}^{K} \Phi_{X_n|H=h}[k_n] e^{-j2\pi k_n x_n}$

Even here, one has to choose a hyperparameter K to replace the infinite summation of the inverse Fourier transform with a finite sum which can be calculated. Further they also require that the support of the PDF is bounded in order to write the characterisitc function in terms on summation rather than an integral.

6.2.3 Sliced Wasserstein Distance for Learning Gaussian Mixtures

Finite Gaussian Mixture Models (GMMs) are parametric and probabilistic tools that are widely used as flexible models for density estimation. They have been shown to be universal approximators which can approximate any given distribution given we take large number of mixture components.
Generally, GMM parameters are learnt using expectation maximization (EM) which is known to be prone to local minima. Inspired by the relationship between the negative log-likelihood function and the Kullback-Leibler (KL) divergence, they propose an alternative formulation for estimating the GMM parameters using the sliced Wasserstein distance [12]. In this new algorithm , they link the same idea of taking random projections of data and then finding the high dimensional PDF from these 1D estimate PDFs. The difference is that they do not consider two random variables at a time, but rather take the entire data vector for random projection. This means that their number of random projections required for a good high dimensional recovery scales exponentially in the dimensionality of the data d. The main motivation of using the wasserstein distance is that it is better than maximizing log likelihood as traditionally done for GMM's because it is less sensitive to initial conditions and the energy landscape of Wasserstein is a lot smoother. The main steps of the algorithm can be summarized as follows:

1. Randomly project the samples on 1D space where the direction of projection will have all N entries in it (basically, not sparse)
2. Fit either a kernel density estimator on the projected 1D samples or have point masses on the projections to get the base density I_y
3. This density I_y is approximated by a Radon transformed GMM where the distance used in approximation is the 1D Wasserstein distance. The parameters of the GMM are updated so that the 1D Wasserstein distance of the Radon transformed GMM is closer to I_y

As noted earlier, the main issue of this approach is that the number of projections should be extremely large in order to compute the transport map with high accuracy (over which the parameters of the GMM are to be optimised). In the subsequent section, we shall see how to use the above method while still requiring much lesser number of projections.

6.3 Learning GMMs from 1-way Marginals - Problem Statement

Recalling the CPD model, if we let $A_n(:, f) = \mathcal{N}(\mu_i, \Sigma_i)$, then the density we recover is a GMM where each component has a diagonal covariance matrix. Hence, there is this implicit connection between tensor decomposition and GMMs if the mode factors are chosen intelligently.

The question we ask is the following - Can we use our previous idea of projecting using 2-sparse vectors only and recovering the high dimensional GMM? Can we integrate this framework with the previously done work on sliced wasserstein distance for learning GMMs? We adopt the following approach for the same:

1. As done before, estimated densities on one-dimensional space for all the $\binom{N}{2}$ pairs of random variables. Now, for each of these pairs, we can invoke the sliced wasserstein module in order to get two-way marginals of the form $p(X_i, X_j)$ which will also be a GMM. Note that this is not expensive as we are just going to 2-way form 1-way marginals. This method is polynomial in the dimensionality d as opposed to the exponential in d cost decribed earlier.
2. Try to estimate the high-dimensional GMM in such a manner that it's 2D marginals as as close as possible to the ones estimated from step 1. We use the MMD metric with the RBF kernel which has a closed form for 2 GMMs: Given two GMMs P and Q with

$$P = \sum_i \alpha_i \mathcal{N}(\mu_i, \Sigma_i) \text{ and } Q = \sum_i \beta_i \mathcal{N}(m_i, S_i)$$

The MMD distance between the two GMMs P and Q is defined as below:

$$MMD(P, Q) = \sqrt{K(P, P) + K(Q, Q) - 2K(P, Q)}$$

For GMMs P and Q, we define the $K(.,.)$ function as follows:

$$K(P,Q) = \sum_{i,j} \alpha_i \beta_j K(P_i, Q_j) \text{ where } X_i \text{ is the ith component of GMM } X$$

$$K(P_i, Q_j) = (2\pi\sigma^2)^{d/2} \mathcal{N}(\mu; \hat{\mu}, \Sigma + \hat{\Sigma} + \sigma^2 I)$$

In the above σ is the variance of the RBF kernel whose value can be chosen by cross-validation and is generally in the range of the variance in the dataset. We formulate our problem as minimizing the MMD loss over all the 2-way marginals. On one hand we have estimating 2-way marginals from Step 1 of the algorithm and on the other hand we have the 2-way marginals of the true high-dimensional GMM (in terms of it's parameters w_i, μ_i, Σ_i). We minimize the following loss function:

$$J = \sum_k MMD^2(P_k, \hat{P}_k)$$

where k is an index covering $\binom{N}{2}$ values, P_k is the parameterised (which are to estimated) 2-way marginal and $\hat{P}_k$ is the estimated 2-way marginal. We simply perform a gradient descent over each parameter to minimize the loss function. However there are certain caveats which we mention below:

a. We need to ensure that the convariance matrix is positive-semi-definite after each iteration and this requires projection on the SPD manifold. This can be done either by setting the negative eigenvalues to 0 or using a more sophisticated scheme of directly optimising over the SPD manifold as shown in [14].

b. The mixing weights of the components are also projected to ensure that they follow the simplex constraints - they should be non-negative and should sum to one.

6.4 Learning GMMs from 1-way Marginals - Experiments

In this section, we present a preliminary experiment to demonstrate the idea described above. This is the synthetic experiment where samples were generated from an actual guassian mixture model. Using this, 2-way marginals were estimated directly (skipping step 1 in the algorithm for now). We also started off with just a single component GMM in order to verify the correctness of the MMD idea.

Experimental Conditions: Dimensionality of the dataset: 3
Number of Samples: 1000
Final Value of Loss Function: 0.157
True Log-Likelihood of the Samples: -113.46
Log-Likelihood of the Samples from the estimated GMM: -115.34

Experimental Conditions: Dimensionality of the dataset: 8
Number of Samples: 50000
Final Value of Loss Function: 0.229
True Log-Likelihood of the Samples: -538.92
Log-Likelihood of the Samples from the estimated GMM: -542.18

We see that for a single Gaussian, we can estimate it from two-way marginals close enough (as indicated by the log-likelihoods). However, here the initial conditions of the GMM to be estimated were taken from the 2-way marginals. The variances and means of the true GMM were given the average values from those obtained from the 2-way marginals and the covariances were kept as it is. This will not be possible in case of GMMs as we do not know the association of components between any 2 GMMs. GMMs would need random initialization with which the algorithm is suspectible to local minimas, we would need to solve this issue which is left for future work.

7. Conclusion and Future Work

We presented a link between density estimation and tomography. We try to solve the problem of density estimation from random projections of data and then use the low-rankness of the pmf tensor to recover it from the marginals. Some future work can be as follows:

1. Improvise the CPD model in order to make it more expressive (such as a graph structure) as opposed to a simple naive bayes model and see if the joint recovery from marginals theory applies there.

2. Extend the Radon theory to joint PDF recovery from marginals for a large variety of mixture model (GMMs are just a starting point). The broad long term goal could be to design a framework so that we can handle a big class of density estimators and standard methods like GMM learning, KDE etc. come out to be as special cases of this framework.

3. Scale the algorithm to very high dimensional data sets(for example 500) which is not possible in the current framework because of the limitations of the SPA algorithm (in the discrete setting). A hope could be use of continuous parametric models like GMMs would help in this goal.

Acknowledgments

I would like to thank Prof. Ajit Rajwade and Dr. Karthik Gurumoorthy for constantly advising me throughout the past year on the above project. They gave me enough freedom to try out new pathways and used to patiently listen to my half-baked ideas. Most of the work of the thesis is an outcome of our bi-weekly brainstorming meets. The idea of the JUROR algorithm was first presented by Karthik which was refined later to bring it in it's current form. I would like to thank my batchmates who made it possible for me to survive the online semesters. Last but not the least, I would like to thank my parents who have always fully supported me in everything I wanted to ever do.

8. References

1. K. P. Murphy, Machine learning: a probabilistic perspective. MIT press, 2012.

2. N. Kargas, N. D. Sidiropoulos, and X. Fu, Tensors, learning, and Kolmogorov extensionfor finite-alphabet random vectors, IEEE Trans. Signal Process., vol. 66, no. 18, pp. 48544868, 2018.

3. A. Yeredor and M. Haardt, Maximum likelihood estimation of a lowrank probability mass tensor from partial observations, IEEE Signal Process. Lett., vol. 26, no. 10, pp. 15511555, Oct 2019

4. X. Fu, K. Huang, N. D. Sidiropoulos, and W.-K. Ma, Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications, IEEE Signal Process. Mag., vol. 36, no. 2, pp. 5980, March 2019.

5. N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, Tensor decomposition for signal processing and machine learning, IEEE Trans. Signal Process., vol. 65, no. 13, pp. 35513582, 2017.

6. S. Ibrahim, X. Fu, Recovering Joint Probability of Discrete Random Variables from Pairwise Marginals - https://arxiv.org/abs/2006.16912

7. Tensors and Probability: An Intriguing Union - N. Sidiropoulos, N. Kargas, X. Fu, https://sigport.org/sites/default/files/docs/KeynoteGlobalSIP2018-Sidiropoulos.pdf

8. S. Dasgupta. Experiments with random projection. Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI), 2000

9. S. Dasgupta. Learning probability distributions. Ph.D. dissertation, University of California at Berkeley, 2000

10. Finbarr O'Sullivan and Yudi Pawitan, Multidimensional Density Estimation by Tomography, Journal of the Royal Statistical Society. Series B (Methodological) , 1993, Vol. 55, No. 2 (1993), pp. 509-521

11. M Amiridi, N Kargas, ND Sidiropoulos, Nonparametric Multivariate Density Estimation: A Low-Rank Characteristic Function Approach - arXiv preprint arXiv:2008.12315, 2020 - arxiv.org

12. Kolouri, Soheil, Gustavo K. Rohde, and Heiko Hoffmann. Sliced Wasserstein Distance for Learning Gaussian Mixture Models. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)

13. N. Kargas and N. D. Sidiropoulos, Learning Mixtures of Smooth Product Distributions: Identifiability and Algorithm, International Conference on Artificial Intelligence and Statistics (AISTATS), 2019

14. Zhi Gao, Yuwei Wu, Yunde Jia, Mehrtash Harandi; Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 7700-7709