L-Moments Method: Introducing Novel Approach of Mixture Parameters Estimation and Its Implementation in Python Library Pysatl-Mpest

Danil Totmyanin, Viacheslav Gorikhovskii
Saint-Petersburg State University
Saint-Petersburg, Russian Federation
dtotmyanin@yahoo.com, v.gorikhovskii@spbu.ru

***Abstract*—This paper addresses the problem of estimating the parameters of a mixture of distributions. One of the most popular ways to solve this problem is the Expectation Maximization algorithm. We introduce a novel alternative method for estimating the parameters of a mixture of distributions, based on numerical characteristics of distribution known as L-moments. Experimental results demonstrate significant speedup and comparable accuracy compared to classical methods.**

I. Introduction

In various applied problems, methods of mathematical statistics find wide applications. One of these methods is the construction of a probabilistic model for describing a particular process. The model is built on the basis of a sample obtained from a series of experiments. To build a model, it is often necessary to be able to estimate distribution parameters based on some initial data.

For example, Latent Dirichlet Allocation (LDA) used in text generation requires estimation of the Dirichlet distribution parameters [1].

In addition to single distributions, there are mixtures of distributions, the parameters of which also need to be estimated. For instance, the wrapped Gaussian mixture models are used for modeling and high-rate quantization of phase data of speech [2]. Moreover, Gaussian mixtures can be used in solving the image inverse problem [3].

Estimating the parameters of mixtures in which different families are represented can also be useful, such as estimating the mixture parameters of the Student's t and Watson distributions for brain shift compensation [4].

The introduced problem can be solved with the Expectation Maximization algorithm (EM algorithm), which uses a maximum likelihood estimation (MLE) method [5]. However, while effective, this method can be computationally intensive, especially for large datasets.

To overcome these limitations, we introduce a novel algorithm based on numerical characteristics of distributions such as L-moments, which offers a potentially huge speedup while maintaining acceptable accuracy.

II. Background

This section overviews mixtures of distributions and the EM algorithm, which underpin our proposed method, and introduces their implementations in the module of the PySATL python library — mpest.

A. Mixture of distributions

Assume we have a sample composed of k subsamples, each of which is distributed according to some distinct probability distribution $f_i(x) \in F$, where $i = 1, ..., k$. The probability density function (PDF) of a mixture distribution with k components is then given by:

$$p(x \mid F, \Omega, \Theta) = \sum_{i=1}^{k} \omega_i \cdot f_i(x \mid \theta_i)$$

where $f_i \in F$, $\omega_i \in \Omega$ — are the mixture weights that represent prior probability such that $\sum \omega_i = 1$ and $\omega_i \geq 0$, $\theta_i \in \Theta$ — parameters of the i-th component.

An example of mixture distribution with two components is shown in Fig. 1

We assume that F is known and Ω, Θ are unknown. In other words, this article addresses the problem of estimating the parameters of a mixture with known components.

For a detailed exposition on mixture distributions, see [6].

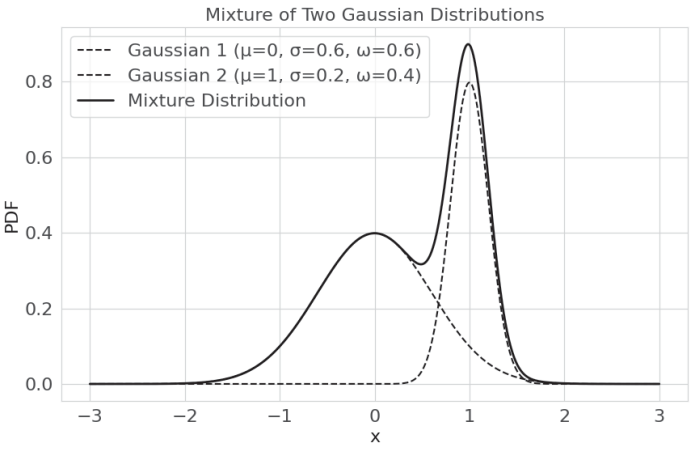

Fig. 1. Example of a mixture distribution with two Gaussian components with weights 0.6 and 0.4

B. EM-algorithm

The most popular method to estimate parameters of a mixture is the EM algorithm [7]. Let z_{ij} be the probability that the i-th observation is distributed according to the j-th component; then the algorithm relies on two steps:

With the given sample and mixture, z_{ij} can be calculated using the Bayesian formula as follows:

$$z_{ij} = \frac{\omega_j \cdot f_j(x_i \mid \theta_j)}{\omega_1 \cdot f_1(x_i \mid \theta_1) + ... + \omega_k \cdot f_k(x_i \mid \theta_k)} \quad (1)$$

With the known matrix Z, weights of the given mixture can be estimated as follows:

$$\omega_j = \frac{\sum_{i=1}^{n} z_{ij}}{n} \quad (2)$$

Therefore, the mixture parameters can be iteratively approximated closer to the true values using the EM algorithm:

- **E-step:** Calculate the matrix $Z = (z_{ij})$ using formula 1.
- **M-step:** Update the mixture weights using formula 2 and parameters by maximizing the logarithm of the likelihood function.

Repeat iterations until convergence, defined by a small change in the likelihood function or parameters between iterations. Another stopping criterion can be reaching a maximum number of iterations, which will be utilized in the experiments.

C. Pysatl-mpest

Pysatl-mpest[1] is a module that provides an implementation of the EM algorithm, using the likelihood function, and provides the functionality for working with mixtures [8]. The current architecture of the module is structured as in Fig. 2.

- **EM:** A central component of the library. Contains a class, which implements the EM algorithm in general form.
- **Methods:** A component that implements various methods based on the EM algorithm.
- **Core:** This component contains implementations of Weibull, exponential and Gaussian distributions and a mixture distribution class.
- **ADistributionChecker:** It is designed to reduce the impact of degenerate distributions or distributions with incorrect parameters in the mixture by removing them.
- **ABreakPointer:** Contains classes that implement algorithm-stopping conditions denoted in Section II.B.
- **Optimizers:** Contains implementations of optimizers for use in the EM algorithm. All calculations are carried out using the SciPy library.

III. L-moments of single distribution

The main results, on which this work and many others are based, are described in the fundamental work of J.R.M. Hosking [9]. In his work, the concept of L-moments was introduced, and some theory was built around them.

This section describes L-moments of single distributions, explains their advantage over conventional moments, and gives an example. Furthermore, we will review existing implementations relevant to this research.

[1] https://github.com/PySATL/pysatl-mpest

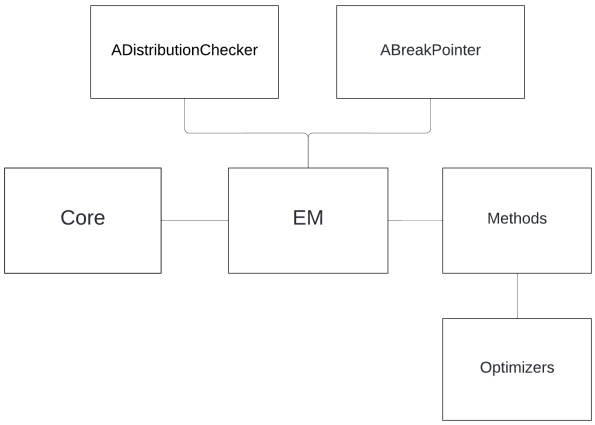

Fig. 2. Diagram of components of the pysatl-mpest module

A. L-moments

L-moments are numerical characteristics of a distribution analogous to raw moments, describing its shape. The main difference is that the definition of L-moments is based on order statistics.

Consider a sorted sample $x_{(1)} < x_{(2)} < ... < x_{(n)}$, then the r-th sample L-moment of the sample is given by [9]:

$$l_r = \sum_{k=0}^{r-1} p_{r-1,k} \cdot \frac{\sum_{j=k+1}^{n} \binom{j-1}{k} \cdot x_{(j)}}{n \cdot \binom{n-1}{k}} \quad (3)$$

where

$$p_{r,k} = (-1)^{r-k} \cdot \binom{r}{k} \cdot \binom{r+k}{k}$$

Analytical formulas for estimating parameters by L-moments are derived for most distributions. For example, for Gaussian distribution:

$$\mu = l_1, \quad \sigma = \sqrt{\pi} \cdot l_2$$

An important benefit of L-moments over conventional moments is their wider applicability. L-moments exist under weaker conditions, requiring only a finite mean, whereas raw moments of higher orders may not exist for many distributions. This is supported by a theorem introduced by Hosking.

B. Illustrative Example

Consider the generalized Pareto distribution (GPD) with probability density function [10]:

$$f(x \mid \alpha, k, \xi) = \begin{cases} \alpha^{-1} \cdot \exp\left(-\frac{x-\xi}{\alpha}\right), & k = 0 \\ \alpha^{-1} \left(1 - \frac{k(x-\xi)}{\alpha}\right)^{\frac{1-k}{k}}, & k \neq 0 \end{cases}$$

Let $\xi = 0,\ \alpha = 1,\ k = -\frac{1}{2}$, then the PDF of GPD is equal to:

$$f_X(x \mid 1, -\frac{1}{2}, 0) = (1 + \frac{1}{2}x)^{-3}$$

By definition of raw moments:

$$\mu_1 = \mathbb{E}[X] = \int_0^{+\infty} x \cdot (1 + \frac{1}{2}x)^{-3} dx = 2$$

$$\mu_2 = \mathbb{E}[X^2] = \int_0^{+\infty} x^2 \cdot (1 + \frac{1}{2}x)^{-3} dx = +\infty$$

But 2nd L-moment [9]:

$$\lambda_2 = \frac{\alpha}{(1+k)(2+k)} = \frac{4}{3}$$

This example shows that the range of application of L-moments is wider than that of ordinary moments. However, if the distribution does not have a mean (for example, the Cauchy distribution), then their use is impossible. For such a case, there is a generalization of L-moments — Trimmed L-moments (TL-moments) [11].

C. Existing implementations

- **Lmo**[2]: Python library providing a wide range of capabilities, from calculating L-moments to estimating the parameters of many distributions. The disadvantage is that Lmo does not involve working with mixtures of distributions and therefore does not solve the problem of estimating the parameters of a mixture distribution.
- **WeibullDist**[3]: Abandoned Python library primarily focusing on the Weibull mixtures. It includes parameters estimation of Weibull mixtures with the L-moments method. Strong restrictions on the families of the mixture components make this package not relevant for mixtures of any distribution [12].
- **lmoments3**[4]: Python implementation of Fortran library created by J.R.M. Hosking. Library has a function for calculating L-moments for distributions mentioned in Hosking's work.

There are several implementations of the EM algorithm for mixtures with various distributions on GitHub[5 6], but no general case has been found.

The libraries studied, although making individual contributions to specific aspects of L-moment methodologies, collectively do not provide a generalized solution for estimating mixture model parameters.

The pysatl-mpest, with its built-in ELM algorithm, directly addresses this need by offering a more generalized and extensible Python library for L-moments-based mixture analysis.

IV. ELM ALGORITHM

In this section, we present an adaptation of the formula of L-moments for mixtures and introduce the Expectation L-moments algorithm (ELM algorithm).

[2]https://github.com/jorenham/Lmo
[3]https://github.com/MakuhIlyukh/WeibullDist
[4]https://github.com/Ouranosinc/lmoments3
[5]https://github.com/rezaahmadzadeh/Expectation-Maximization
[6]https://github.com/dmetivie/ExpectationMaximization.jl

A. Mixture L-moments

The definition of L-moments in Section III.A assumes that all observations come from a single distribution. This assumption is violated in mixture models, where samples are distributed according to mixture distribution, making a direct application of the standard L-moments calculation inappropriate. Therefore, the formula is adapted as follows:

$$l_r^{(j)} = \sum_{k=0}^{r-1} p_{r-1,k} \cdot \frac{\sum_{i=k+1}^{n} b_{ik}^{(j)} \cdot z_{ij} \cdot x_{(i)}}{\sum_{i=1}^{n} z_{ij} \cdot \binom{n-1}{k}} \tag{4}$$

where $l_r^{(j)}$ — r-th L-moment of the j-th component of mixture, z_{ij} — probability that $x_{(i)}$ distributed from the j-th component of the mixture, $b_{ik}^{(j)} = \binom{\lfloor \sum_{l=1}^{i} z_{ij} \rceil}{k}$, symbol $\lfloor \rceil$ denotes rounding to nearest integer

It should be noted that formula 4 was derived empirically, through iterative evaluation of algorithm performance.

The necessity to compute binomial coefficients in formula 4 introduces a significant computational overhead, especially as the sample size and order of L-moment increase. Precomputations of factorials up to $n = 1000$ were implemented to speed up calculations. For $n > 1000$, we propose using Stirling's formula. Since the accuracy of calculating the binomial coefficients is not very important for large n, using approximations will allow one to obtain a significantly faster result without serious loss of accuracy.

B. Implementation of algorithm

Algorithm 1 ELM algorithm

Initialization:
$\Omega^{(0)} = \{\omega_j^{(0)}\}$
$\Theta^{(0)} = \{\theta_j^{(0)}\}$
$Z^{(0)} = \{z_{ij}^{(0)}\}$
for $t = 0, 1, \ldots$ **do**
  LM-step:
  for $j = 1, \ldots, k$ **do**
    $L_j \leftarrow$ **calculate_lm**()
    $\theta_j^{(t+1)} \leftarrow$ **estimate**(L_j)
  end for
  E-step:
  for $i = 1, \ldots, n$, $j = 1, \ldots, k$ **do**
    $Z^{(t+1)} \leftarrow$ **calc_indicators**()
    $\omega_j^{(t+1)} \leftarrow$ **update_weights**()
  end for
  for $j = 1, \ldots, k$ **do**
    if $\omega_j^{(t+1)} < \delta$ **or** $\theta_j^{(t+1)}$ is incorrect **then**
      Delete j-th component
    end if
  end for
  if end_condition() **then**
    Exit.
  end if
end for

As the base for the ELM algorithm, the EM algorithm from Section II.A. will be used. While the E-step remains the same, the Maximization step (M-step) is substituted with the L-moments step (LM-step).

Thus, the algorithm has the form shown in pseudocode in Algorithm 1

Upper indices, e.g. $\Theta^{(t)}$ denotes a variable at step t.

In words, the algorithm looks like this:

- **LM-step:** For all components f_j, calculate L-moments up to the required order r using formula 4. Using formulas introduced by J.R.M. Hosking, calculate components parameters θ_j.
- **E-step:** This step is identical to that of the EM algorithm. The function **calc_indicators** calculates matrix Z by formula 1. In the function **update_weights**, formula 2 is used to calculate mixture weights Ω.

After each iteration, using the class of type *ADistributionChecker* a j-th component is removed if its parameters $\theta_j^{(t+1)}$ are deemed invalid or if its weight $\omega_j^{(t+1)}$ is less than the threshold δ.

The **end_condition** function checks the fulfillment of the algorithm end conditions specified in Section II.C.

V. Experiment pipeline

Since ELM and EM algorithms are implemented in pysatl-mpest, we will compare them under the conditions of this library. To conduct the experiment conveniently, an experimental environment was implemented. The experimental environment is able to save all intermediate results and compare methods by criteria.

In this section, we pose research questions about the ELM algorithm. To address research questions outlined and evaluate the proposed algorithm, we design and execute a series of experiments.

A. Research questions

To measure the efficiency and accuracy of the provided method in Section IV, we formulated three research questions:

- **RQ1:** To what extent is the scope of applicability of the new method within the set conditions coinciding with the EM algorithm? The answer to this question will reveal the limitations that may overlap with the use of the L-moment method.
- **RQ2:** What is the computational speed of the ELM algorithm in comparison with the maximum likelihood method? Within the framework of this issue, it remains to be seen whether the L-moments method is faster or slower than the classic.
- **RQ3:** Is the ELM algorithm more or less accurate than the EM algorithm, and what factors influence their relative accuracy? The answer to this question will allow you to understand in which cases it is better to use the L-moment method and in which it is better to use the EM algorithm.

B. Experiment pipeline

To ensure the reproducibility of the experiment, a fixed seed of 42 was utilized for all random processes.

The experiment is separated into different steps. As a step 0, the configuration is fixed: the sample size and F — families of components. Then for each configuration three steps defined as follows:

- **Step 1:** Data sampling from configured mixture with random Ω and Θ. Samples and base mixtures of each run are saved.
- **Step 2:** Parameter estimation by EM and ELM on each sample from Step 1. For each sample, a single set of weights Ω and parameters Θ is generated randomly. This random initial guess is then used for both EM and ELM algorithms. The resulting mixtures are saved for subsequent analysis.
- **Step 3:** Evaluation metric values representing a similarity between the resulting mixture and base mixture and execution time. Following the analysis of all Step 2 results, samples of metric values and algorithm running times are obtained. For these samples, descriptive statistics such as mean, median, and standard deviation are then computed.

Scripts were written to perform each step, which can be found in the branch of the pysatl-mpest repository[7].

To evaluate the accuracy of methods, the mean integrated square error (MISE) implemented in pysatl-mpest will be used:

$$\int_{-\infty}^{+\infty} \Big(p(x \mid F, \Omega, \Theta) - p(x \mid F, \Omega^*, \Theta^*)\Big)^2 dx \qquad (5)$$

where $p(x \mid F, \Omega, \Theta)$ — density function of mixture with true parameters Ω and Θ, $p(x \mid F, \Omega^*\Theta^*)$ — density function of result mixture with estimated parameters Ω^* and Θ^*.

The integral is calculated using SciPy library.

C. Experiment setup

To answer the research questions posed, we developed an experimental setup:

- Mixtures contain only exponential, Gaussian, and two-parametric Weibull components.
- For the parameters estimation using the EM algorithm, all optimizers were used. Among all the results, the one with the best metric value is chosen.
- Samples size: 200, 500 and 1000
- Components number — 2
- Runs per configuration — 100
- Maximum number of algorithms steps — 16

If an error occurs at any step, the algorithm terminates and returns the resulting mixture.

The mixture of exponential and Weibull distributions is not included in the experiment setup because the exponential is a special case of the Weibull.

[7]https://github.com/iraedeus/pysatl-mpest/tree/exp_scripts

VI. Experiment results

The pipeline of the experiment given in Section V yielded meaningful results.

In this section, we will present the results of comparing the accuracy of the algorithms in Tables I and II and their runtime in Table III. Subsequently, we will delve into an analysis of these results and provide answers to the research questions outlined in Section 5.

Results of all steps of the experiment pipeline are contained in the GitHub repository[8].

A. Accuracy comparison

TABLE I. Information about the MISE of the resulting mixture of the EM algorithm compared to the true one

Sample size	Mixture	MISE		
		μ	σ	Median
200	Exp+Exp	0.0057	0.0083	0.0024
	Exp+Gauss	0.06	0.114	0.017
	Gauss+Gauss	0.0031	0.0022	0.0025
	Weib+Gauss	0.11	0.23	0.028
	Weib+Weib	0.027	0.12	0.0045
500	Exp+Exp	0.0029	0.0042	0.0015
	Exp+Gauss	0.074	0.17	0.018
	Gauss+Gauss	0.0012	0.0008	0.0011
	Weib+Gauss	26.0	237.0	0.029
	Weib+Weib	0.077	0.59	0.0016
1000	Exp+Exp	0.0014	0.0006	0.002
	Exp+Gauss	0.079	0.18	0.017
	Gauss+Gauss	0.008	0.025	0.0007
	Weib+Gauss	0.1	0.22	0.026
	Weib+Weib	0.013	0.06	0.0013

Tables I and II, which present accuracy metrics, shows that the median and mean decrease steadily as the sample size increases. The exception is the Weibull and Gaussian mixture, which are characterized by outliers.

It can be seen that, unlike the ELM, the accuracy of the EM algorithm almost does not change with increasing sample size for a mixture of Exponential and Gaussian.

Comparing medians and means, it is noticeable that the EM works better for smaller sample sizes. However, for a sample size of 1000, the situation is completely opposite. The ELM algorithm shows better results for larger samples compared to the classic EM.

As for standard deviation, its dependence on sample size is similar to the dependence of median and mean. This statistic also decreases with sample size for all mixtures except those containing a Weibull component.

Comparing the two algorithms based on these statistics, it can be seen that the ELM algorithm has a significant advantage starting from a sample size of 1000.

TABLE II. Information about the MISE of the resulting mixture of the ELM algorithm compared to the true one

Sample size	Mixture	MISE		
		μ	σ	Median
200	Exp+Exp	0.0057	0.0083	0.0024
	Exp+Gauss	0.0086	0.02	0.0037
	Gauss+Gauss	0.0034	0.0026	0.0029
	Weib+Gauss	0.67	5.0	0.003
	Weib+Weib	0.36	1.9	0.005
500	Exp+Exp	0.0029	0.0042	0.0015
	Exp+Gauss	0.0034	0.0055	0.0015
	Gauss+Gauss	0.0014	0.0011	0.0012
	Weib+Gauss	0.13	0.92	0.0012
	Weib+Weib	0.064	0.55	0.0017
1000	Exp+Exp	0.0014	0.002	0.0006
	Exp+Gauss	0.0014	0.0023	0.0007
	Gauss+Gauss	0.0006	0.0005	0.0006
	Weib+Gauss	180.0	1790.0	0.0007
	Weib+Weib	0.033	0.66	0.001

TABLE III. Comparison of mean execution speed of algorithms

Mixture	Sample size	Execution Time (seconds)	
		ELM	EM
Exp+Exp	200	0.067 ± 0.003	0.41 ± 0.11
	500	0.21 ± 0.04	1.2 ± 0.46
	1000	0.37 ± 0.17	1.7 ± 0.47
Exp+Gauss	200	0.11 ± 0.012	2.4 ± 1.2
	500	0.33 ± 0.066	6.6 ± 4.0
	1000	0.67 ± 0.28	12.0 ± 6.2
Gauss+Gauss	200	0.16 ± 0.01	2.9 ± 1.8
	500	0.47 ± 0.096	8.9 ± 5.5
	1000	0.89 ± 0.048	15.0 ± 10.0
Weib+Gauss	200	0.14 ± 0.019	4.3 ± 2.4
	500	0.47 ± 0.11	13.0 ± 8.7
	1000	0.74 ± 0.046	23.0 ± 12.0
Weib+Weib	200	0.12 ± 0.056	1.7 ± 0.96
	500	0.37 ± 0.19	5.1 ± 3.5
	1000	0.62 ± 0.39	7.8 ± 4.3

B. Runtime comparison

Table III shows that the execution speed of the presented algorithm is significantly lower in comparison with the EM algorithm. This follows obviously from the fact that EM algorithm performs optimization of the logarithm of the likelihood function while the ELM algorithm just computes L-moments.

Also, the standard deviation indicates that the execution speed of the ELM algorithm is much more stable than that of the EM algorithm, which may be due to its resistance to

[8]https://github.com/iraedeus/EM-algo-experiment

initial conditions or to the fact that the EM algorithm solves the function optimization problem.

VII. Results discussion

A. RQ1

Our results demonstrate that the ELM algorithm works well in the same domain as the EM algorithm. A common characteristic of both algorithms is outliers in metric values when dealing with the mixtures that contain Weibull and Gaussian components.

The comparable accuracy in parameter estimation for the considered distributions and mixture configurations allows the use of ELM as an alternative.

B. RQ2

In all cases, the ELM algorithm is significantly faster than the EM algorithm. Specifically, for datasets with a sample of size 200, ELM execution time is under 200 ms.

From the data in Table III on the increase in time with increasing sample size, it follows that the asymptotic complexity of both algorithms is most likely $O(n \cdot k)$. However, from the data on the absolute difference in the execution time of the algorithms with sample sizes of 1000 and 200, it follows that the hidden constant in the complexity of ELM is significantly smaller.

C. RQ3

The accuracy of the EM algorithm by the metric proposed in formula 5 is higher on samples with a size of 200 than ELM. On samples with higher sizes (500, 1000), the introduced approach has higher or comparable accuracy than the classical method.

To be more specific, let's look at different mixtures.

- Exp+Exp: The mean accuracy of both algorithms is absolutely identical. However, the median accuracy of ELM algorithm exhibits a significant improvement at a sample size of 1000.
- Exp+Gauss: The EM algorithm works worse with increasing sample size, but the ELM algorithm works with increasing accuracy at the same time.
- Gauss+Gauss: The accuracy of both algorithms is comparable.
- Weib+Gauss: Based on the median values, the ELM algorithm outperforms the EM algorithm at the sample sizes of 500 and 1000.
- Weib+Weib: Presents ambiguous results. The EM algorithm underperforms across all metrics only at a sample size of 200. At a sample size of 500, the ELM algorithm exhibits lower median accuracy, while at a size of 1000, it's better only at the mean and standard deviation.

VIII. Conclusion

In this work we introduced and implemented a novel approach to estimate parameters of mixtures using L-moments. Moreover, to implement ELM algorithm, we adapted the L-moment formula for the case of mixture distributions. Based on the results of the study, we can conclude that the algorithm is quite promising. However, it is necessary to continue its research for more accurate results.

Therefore, in the future, it is planned to expand the class of distributions implemented in pysatl-mpest. To conduct more experiments with larger samples as well as mixtures of size 3 or more, it is planned to refine the experimental environment. To support distributions with an uncertain mean (e.g., Cauchy), it is planned to develop an algorithm based on TL moments, as well as its implementation in the pysatl-mpest module.

Acknowledgment

This work was supported by St. Petersburg State University (Pure ID 116636233).

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, no. null, p. 993–1022, Mar. 2003.

[2] Y. Agiomyrgiannakis and Y. Stylianou, "Wrapped gaussian mixture models for modeling and high-rate quantization of phase data of speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 17, pp. 775 – 786, 06 2009.

[3] G. Yu, G. Sapiro, and S. Mallat, "Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity," *IEEE Transactions on Image Processing*, vol. 21, 06 2010.

[4] S. Bayer, N. Ravikumar, M. Strumia, X. Tong, Y. Gao, R. Fahrig, M. Ostermeier, and A. Maier, "Intraoperative brain shift compensation using a hybrid mixture model," 09 2018.

[5] E. E. Elmahdy and A. W. Aboutahoun, "A new approach for parameter estimation of finite weibull mixture distributions for reliability modeling," *Applied Mathematical Modelling*, vol. 37, no. 4, pp. 1800–1810, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0307904X12002545

[6] G. McLachlan and D. Peel, *Finite Mixture Models*, ser. Wiley Series in Probability and Statistics. Wiley, 2004. [Online]. Available: https://books.google.fi/books?id=c2_fAox0DQoC

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 12 2018. [Online]. Available: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

[8] K. Anton, "Efficient parameter estimation of distribution mixture," 2024. [Online]. Available: http://hdl.handle.net/11701/46017

[9] J. R. M. Hosking, "L-moments: Analysis and estimation of distributions using linear combinations of order statistics," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 52, no. 1, pp. 105–124, 1990. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1990.tb01775.x

[10] J. R. M. Hosking and J. R. Wallis, "Parameter and quantile estimation for the generalized pareto distribution," *Technometrics*, vol. 29, no. 3, pp. 339–349, 1987. [Online]. Available: http://www.jstor.org/stable/1269343

[11] E. E. A. Habib, "Probability distribution theory, generalisations and applications of l-moments," pp. 40–47, 2001. [Online]. Available: http://etheses.dur.ac.uk/3987/

[12] M. I. Mikhailovich, "Parameter estimation of weibull mixture distributions," 2023.