Department of Electrical Engineering
EE5111 - Estimation Theory - Project Report

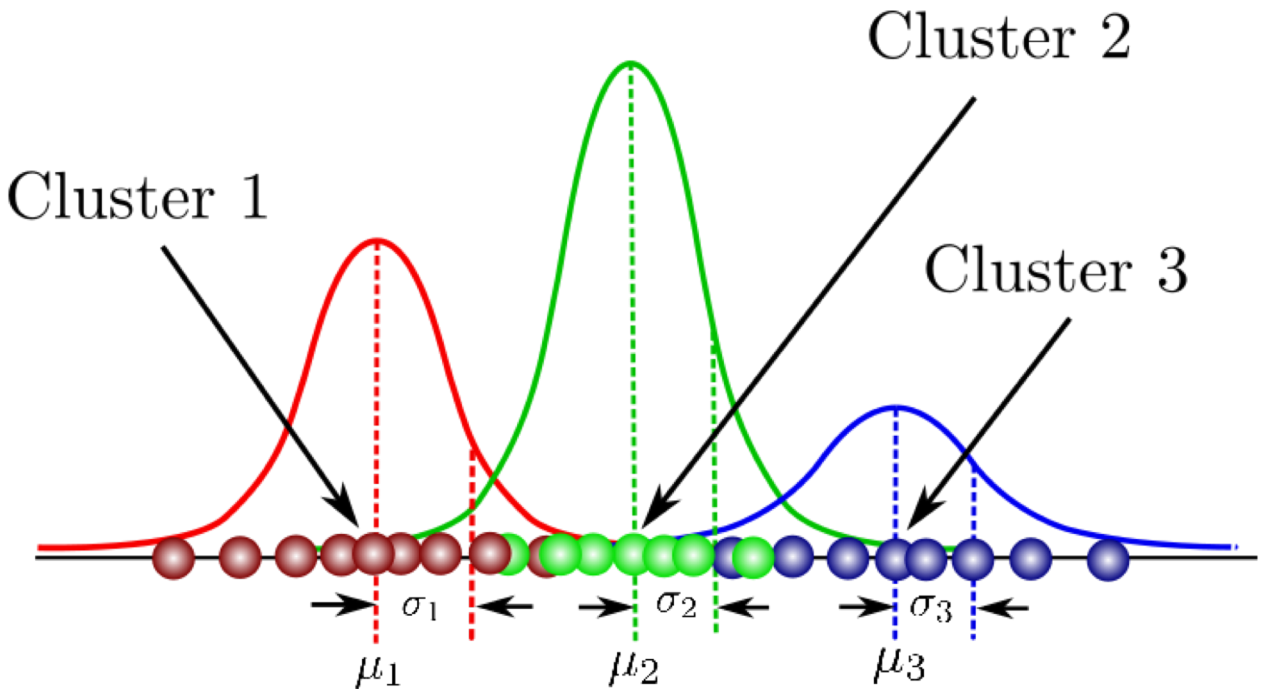
Convergence of the EM Algorithm for Gaussian Mixtures with Unbalanced Mixing Coefficients

Arvind Menon EP17B017
Vallabh Ramakanth EE17B068
Aditya Sundararajan EE17B102

May 2020

Gaussian Mixture Models

Gaussian Mixture Models can be used to model complex multimodal distributions while still maintaining the theoretical and computational benefits of Gaussian Models.



The Model

We will assume that our data arrives from K Gaussians, each with their own means μ_i and covariances Σ_i for $1 \leq i \leq K$ such that,

$$\mathcal{P}(X_i) = \sum_{i=1}^{K} \alpha_i \mathcal{N}(X_i|\mu_i, \Sigma_i)$$

Where α_i are the mixing coefficients and $\mathcal{N}(X_i|\mu_i, \Sigma_i)$ is the probability that a Gaussian with mean μ_i and covariance matrix Σ_i takes value X_i

Expectation Maximization Algorithm

The Expectation Maximization (EM) Algorithm can be used to iteratively find the Maximum Likelihood Estimate of parameters in statistical models with latent variables.

In our case since each observation randomly picked from a cluster i with probability α_i, the latent variable Z will indicate which cluster a particular observation has come from (but is not directly measured)

EM Algorithm: E-Step

$$Q(\boldsymbol{\theta}^*, \boldsymbol{\theta}) = \sum_{i=1}^{N} \sum_{j=1}^{K} h_{ij} \left(\ln \alpha_i + \ln \mathcal{N}\left(X_i | \mu_j, \Sigma_j\right) \right)$$

Which is the Expectation of the Complete Log Likelihood and needs to be maximized in the M-Step. h_{ij} is the probability that the observation X_i came from cluster j and is given by:

$$h_{ij} = \frac{\alpha_j \mathcal{N}(X_i | \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \alpha_k \mathcal{N}(X_i | \mu_k, \Sigma_k)}$$

EM Algorithm: M-Step

On maximizing Q computed in the E-Step, we get the following update rules:

$$\mu_j(t+1) = \frac{\sum_{i=1}^{N} h_{ij}(t) X_i}{\sum_{i=1}^{N} h_{ij}(t)}$$

$$\Sigma_j(t+1) = \frac{\sum_{i=1}^{N} h_{ij}(t)(X_i - \mu_j)(X_i - \mu_j)^T}{\sum_{i=1}^{N} h_{ij}(t)}$$

$$\alpha_j(t+1) = \frac{\sum_{i=1}^{N} h_{ij}(t)}{N}$$

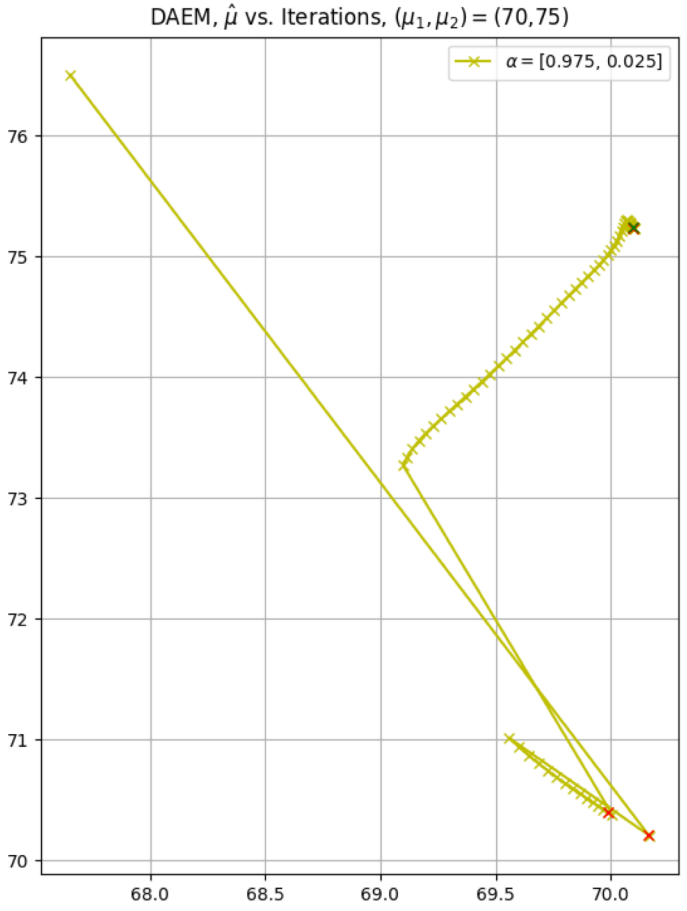
Deterministic Anti-Annealing Variant of the EM Algorithm (DAEM)

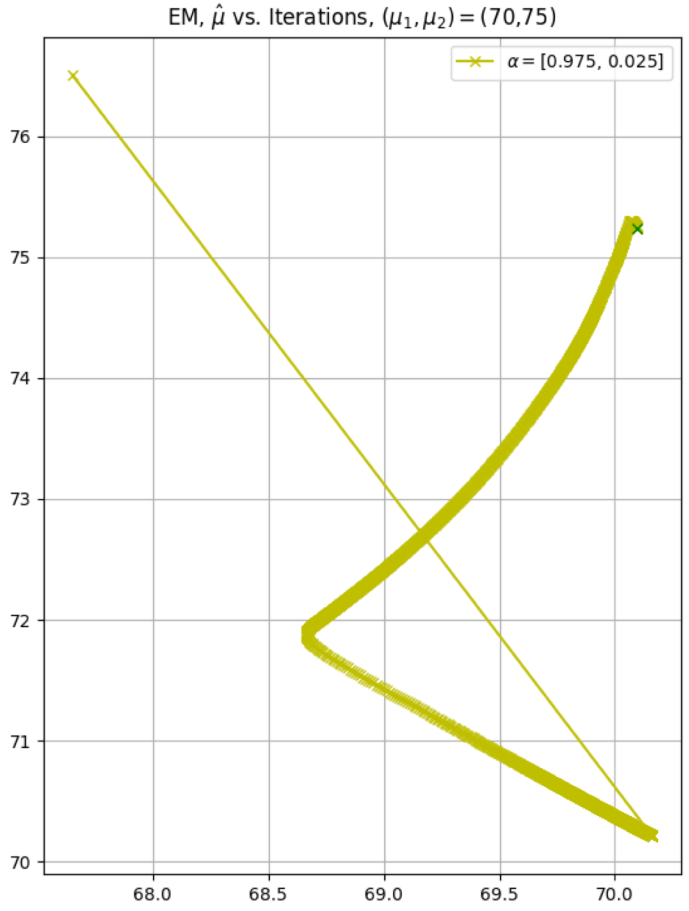
This variant of the EM algorithm can help speed up the rate of convergence and also prevent it from getting stuck in local maxima. The DAEM algorithm modifies h_{ij} from the E-Step as:

$$h_{ij} = \frac{\left(\alpha_j \mathcal{N}(X_i|\mu_j, \Sigma_j)\right)^{\beta}}{\sum_{k=1}^{K}(\alpha_k \mathcal{N}(X_i|\mu_k, \Sigma_k))^{\beta}}$$

Where β is the scheduling parameter and will be swept from a low value to β_{max} which is greater than 1 and then brought back down to 1

Performance in Simulation



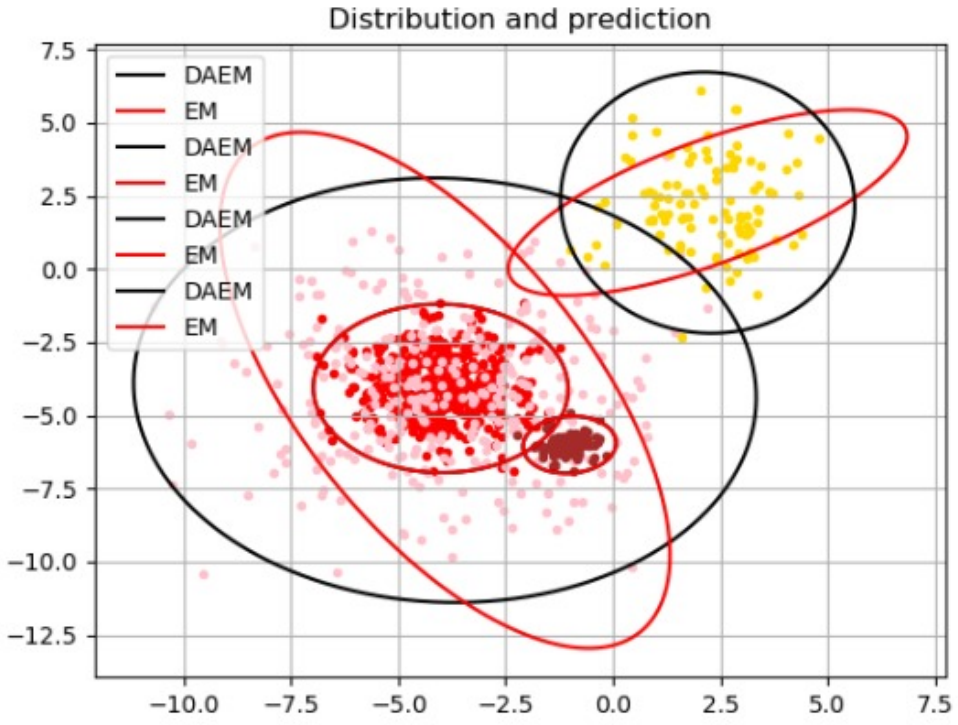


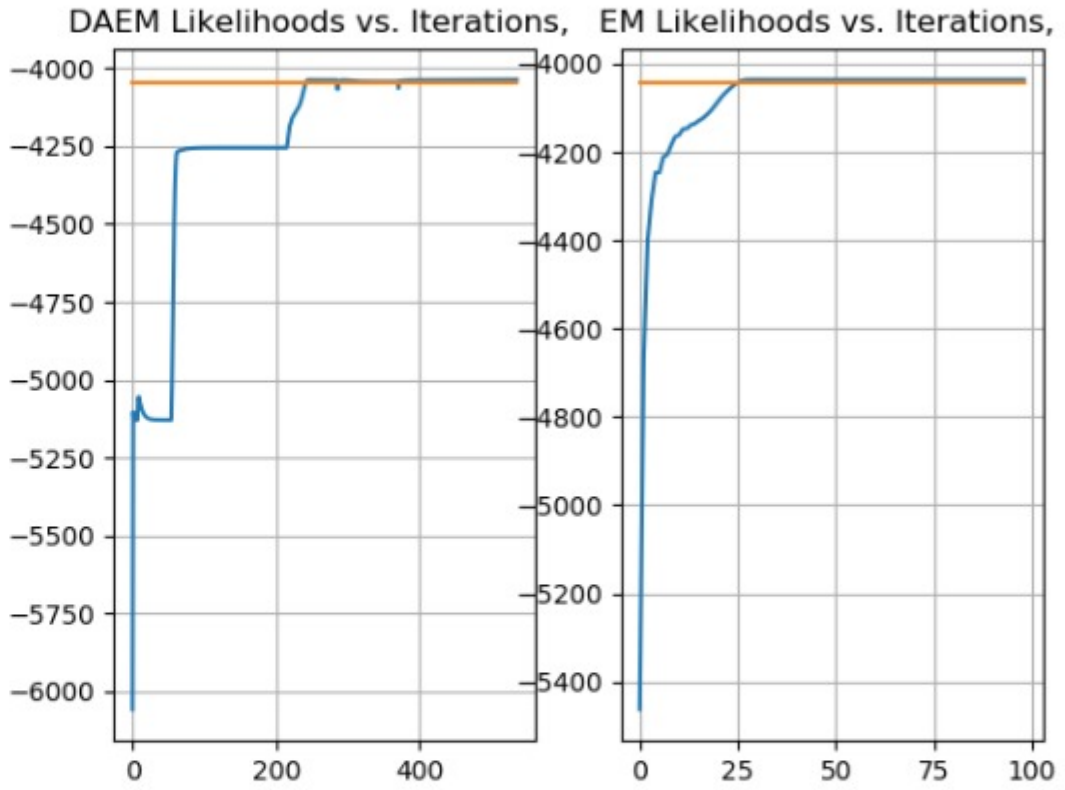
Fast convergence of DAEM if clusters are overlapping and are heavily unbalanced.

Sample points = 1000

DAEM performs a better and robust search in given parameter space.

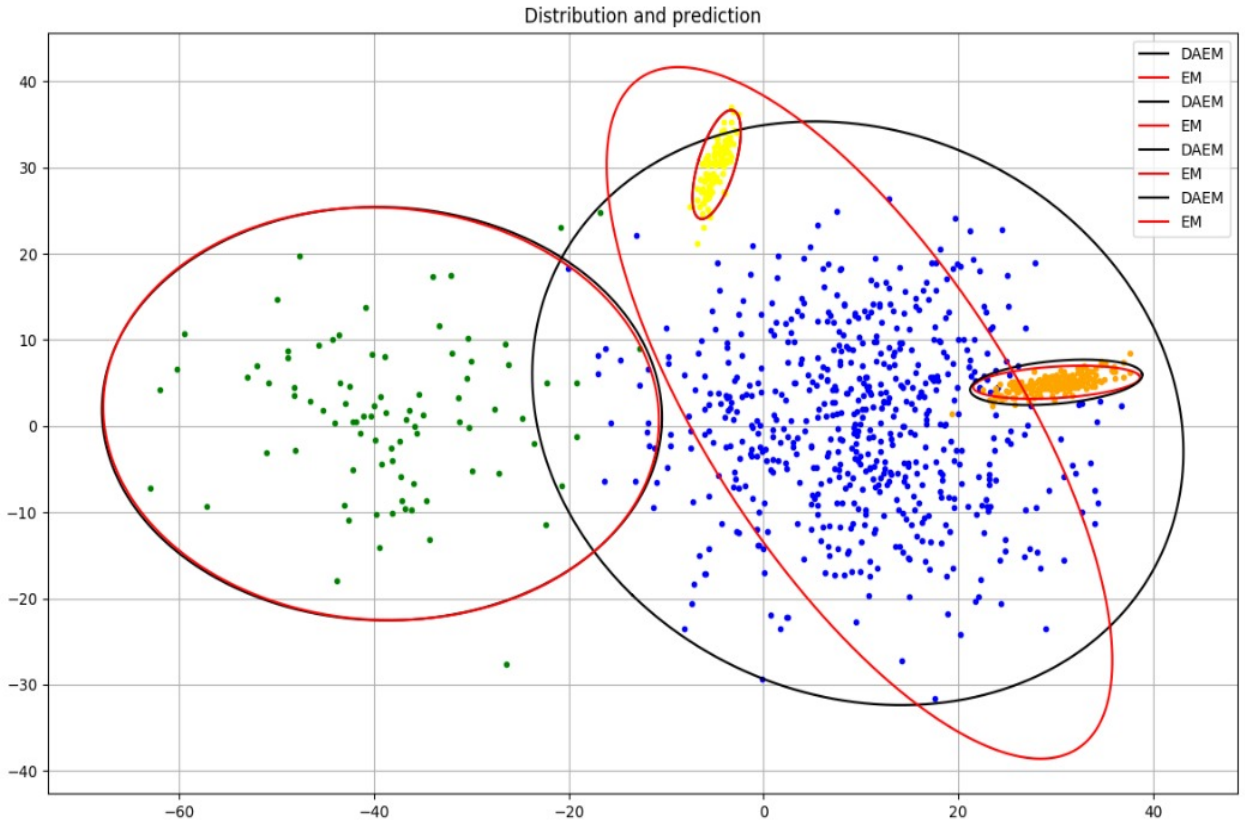
Performance in Simulation





Separating 4 different Bivariate Gaussian Clusters, sample size = 1000

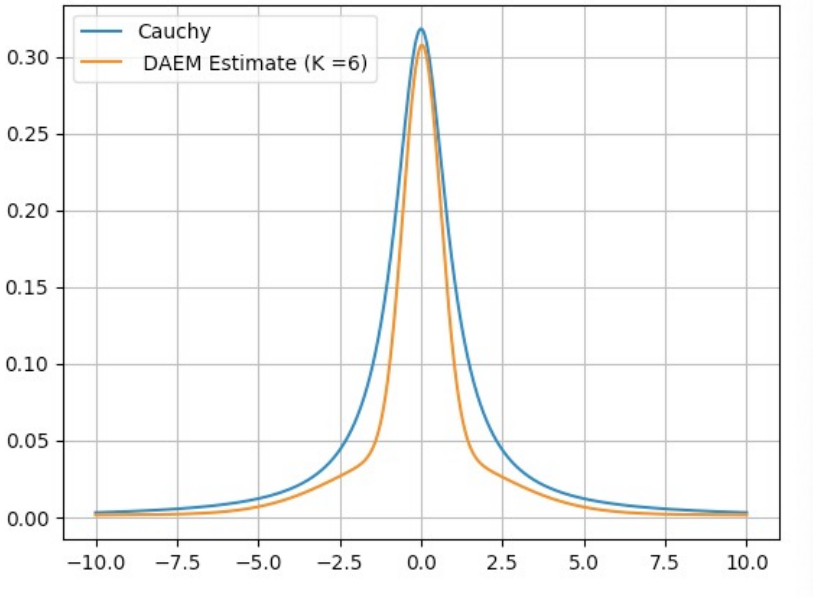
Performance in Simulation



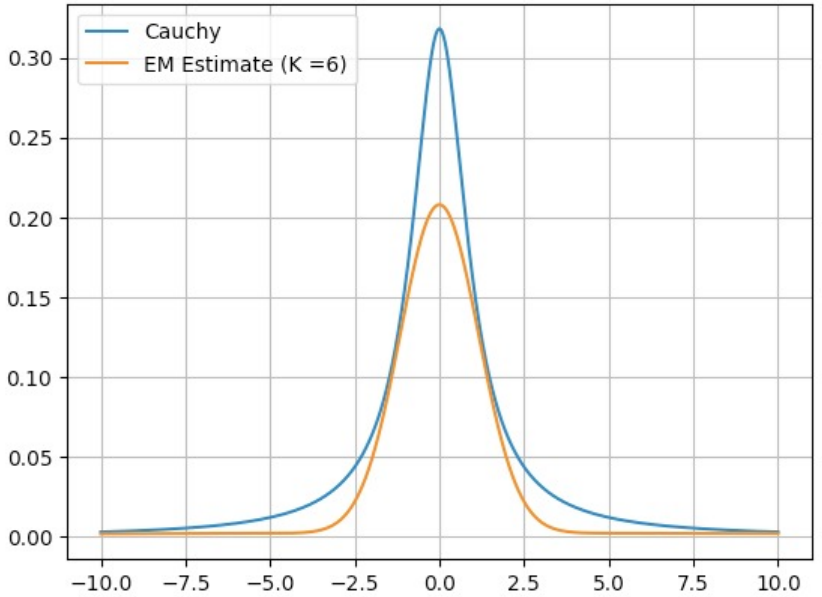
Separating 4 different Bivariate Gaussian Clusters, sample size = 1000

Application of DAEM: Modelling Non-Gaussian Noise

Non-Gaussian noise can be modelled as a finite mixture of Gaussian pdfs. This performs far better than conventional methods in the case of additive Non-Gaussian noise. Here we have modelled a Cauchy distribution using GMMs

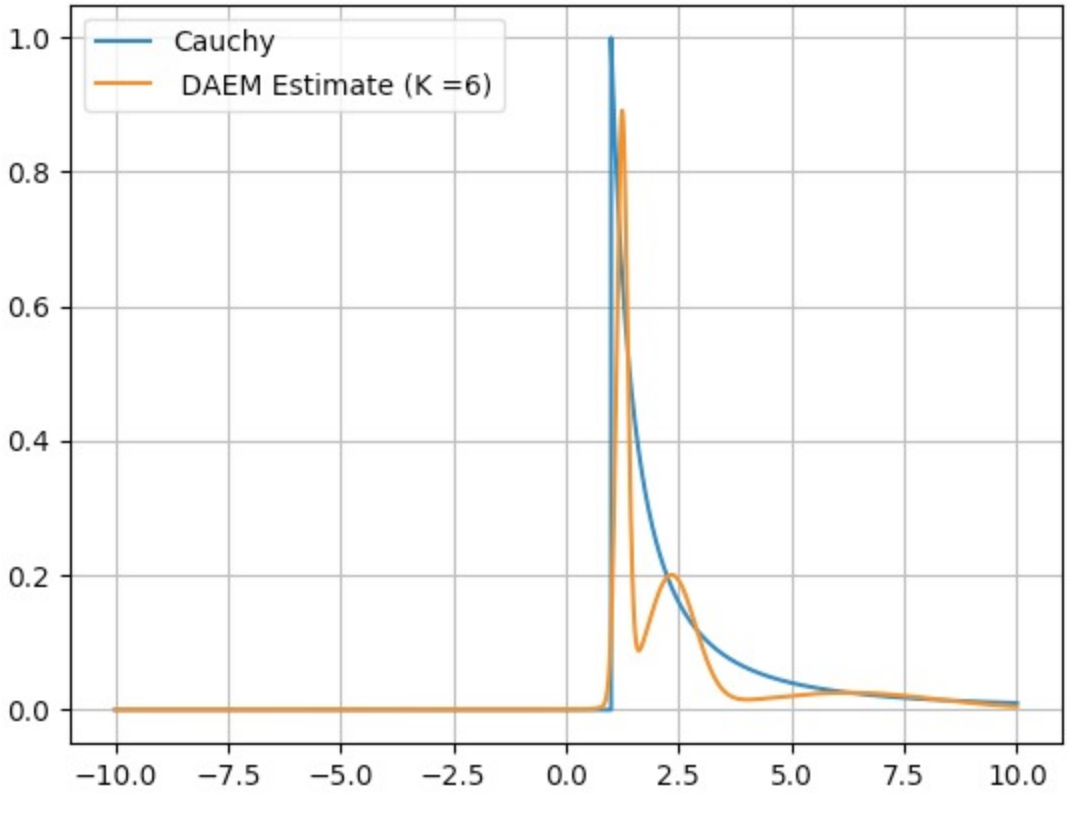


KL Divergence for DAEM = 0.3468

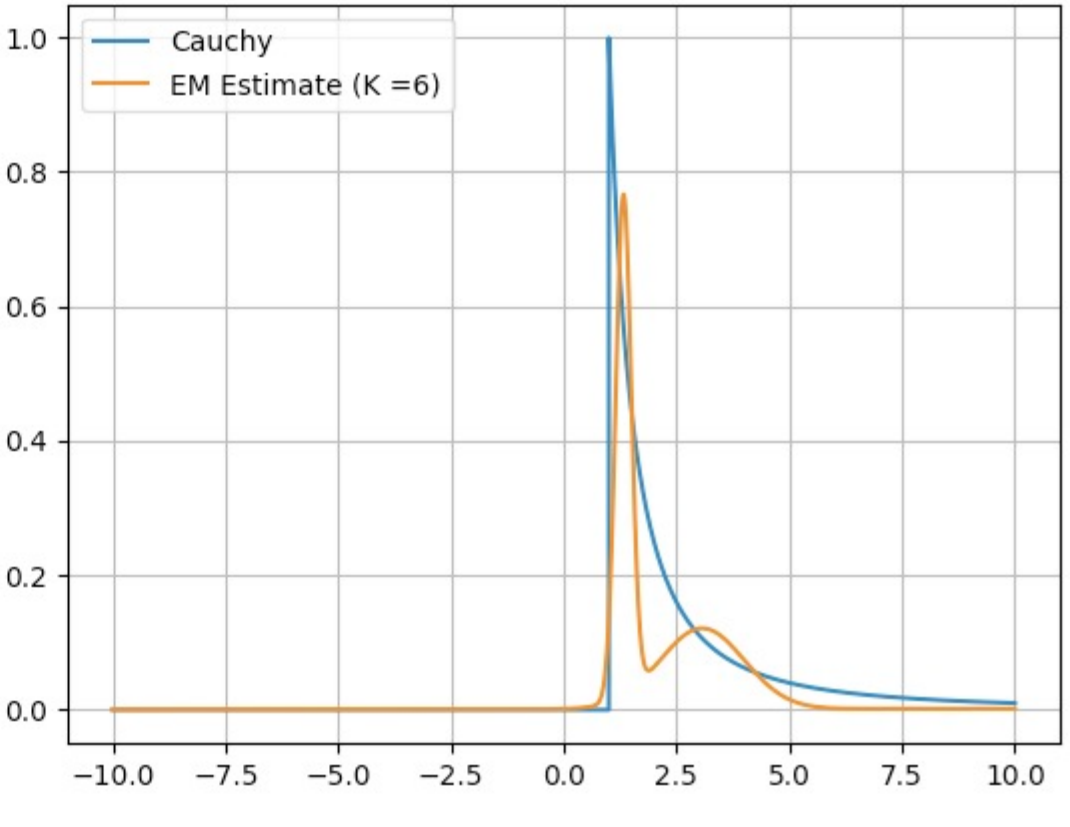


KL Divergence for EM = 0.4349

Modelling the Pareto Distribution



KL Divergence for DAEM = 0.46



KL Divergence for EM = 0.62

Channel Estimation with Non-Gaussian Noise

$$\boldsymbol{y} = A\boldsymbol{s} + \boldsymbol{w}$$

The above equation relates the received signal $\boldsymbol{y}$ with the channel vector $\boldsymbol{s}$ in the presence of complex non-Gaussian Noise.

$\boldsymbol{w}$ is hence modelled as a K-component GMM with means μ_i and variances ${\sigma_i}^2$ for $1 \leq i \leq K$

A is given by:

$$A = XF$$

$X = \text{diag}(x_1, x_2, \ldots, x_N)$ where x_i is the ith pilot symbol.

F is the DFT Matrix.

Update Rules

$$\boldsymbol{s}(t+1) = (A^T B A)^{-1} A^T B \boldsymbol{y}$$

$$\sigma_j^2(t+1) = (\boldsymbol{y} - A\boldsymbol{s})^H \left(\widetilde{H}_j - \widetilde{\boldsymbol{h}}_j \widetilde{\boldsymbol{h}}_j^{\,H}\right)(\boldsymbol{y} - A\boldsymbol{s})$$

$$\mu_j(t+1) = \widetilde{\boldsymbol{h}}_j^{\,H}(Y - As)$$

$$\alpha_j(t+1) = \frac{\sum_{i=1}^{N} h_{ij}}{N}$$

h_{ij} is the probability that an observation X_i comes from cluster j and is defined as before in conventional EM.

$$\boldsymbol{h}_j := (h_{1j}, h_{2j}, \ldots, h_{Nj})^T$$

$$\widetilde{\boldsymbol{h}}_j := \frac{\boldsymbol{h}_j}{\sum_{i=1}^{N} h_{ij}}$$

$$\widetilde{H}_j := \mathrm{diag}(\tilde{h}_{1j}, \tilde{h}_{2j}, \ldots, \tilde{h}_{Nj})$$

$$B := \sum_{j=1}^{K} \frac{\widetilde{H}_j - \widetilde{\boldsymbol{h}}_j \boldsymbol{h}_j^T}{\sigma_j^2(t)}$$

Simulation Details

Channel model generated using:

$$s[k] = \frac{1}{\|\boldsymbol{p}\|}(a[k] + jb[k])p[k]$$

$$p[k] = e^{-0.2k}$$

$$k \in \{0, 1, \dots, L-1\}$$

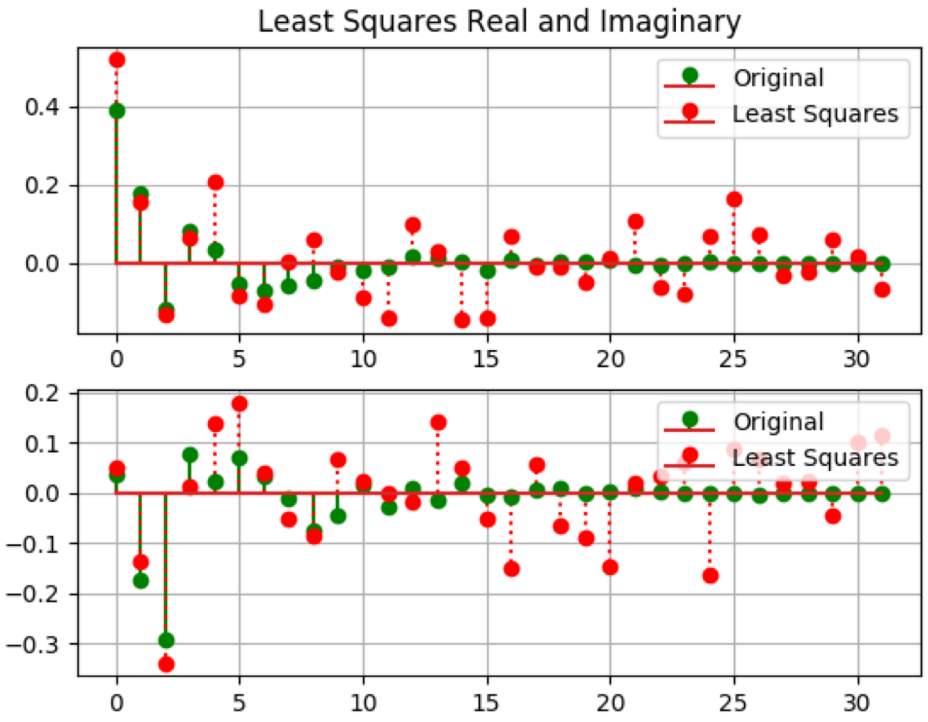
$$a[k], b[k] \sim \mathcal{N}(0, 0.5)$$

Channel is assumed to have L = 32 taps and there are N = 128 pilot symbols.

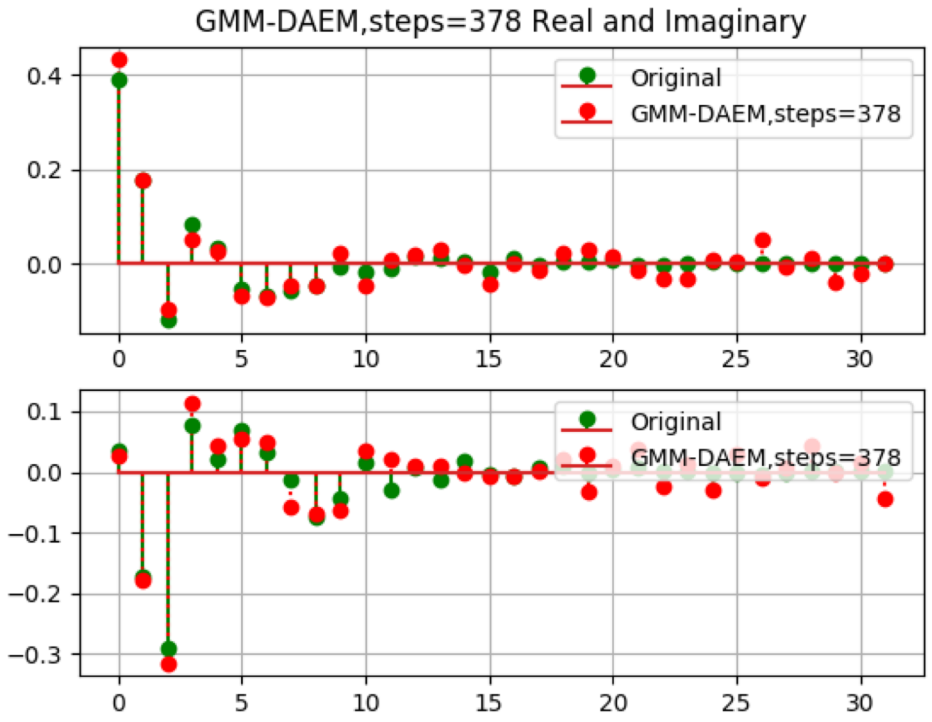
Process noise added is assumed to be a zero-mean complex Cauchy RV with scale parameter $\gamma = 0.1$.

The GMM is assumed to have 4 clusters.

Simulation Results

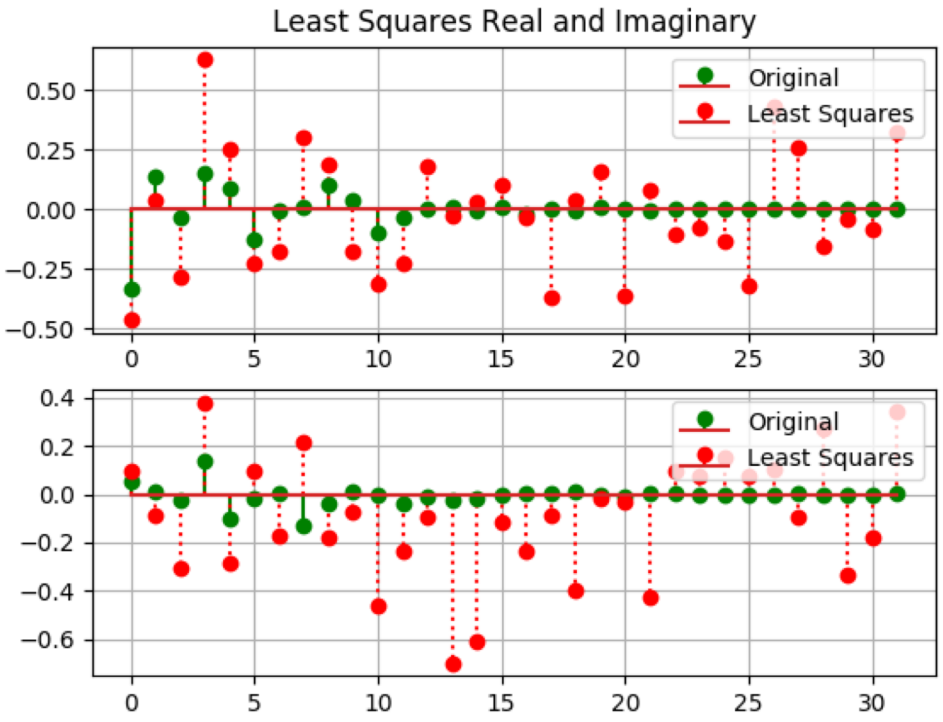


SE = 0.4072

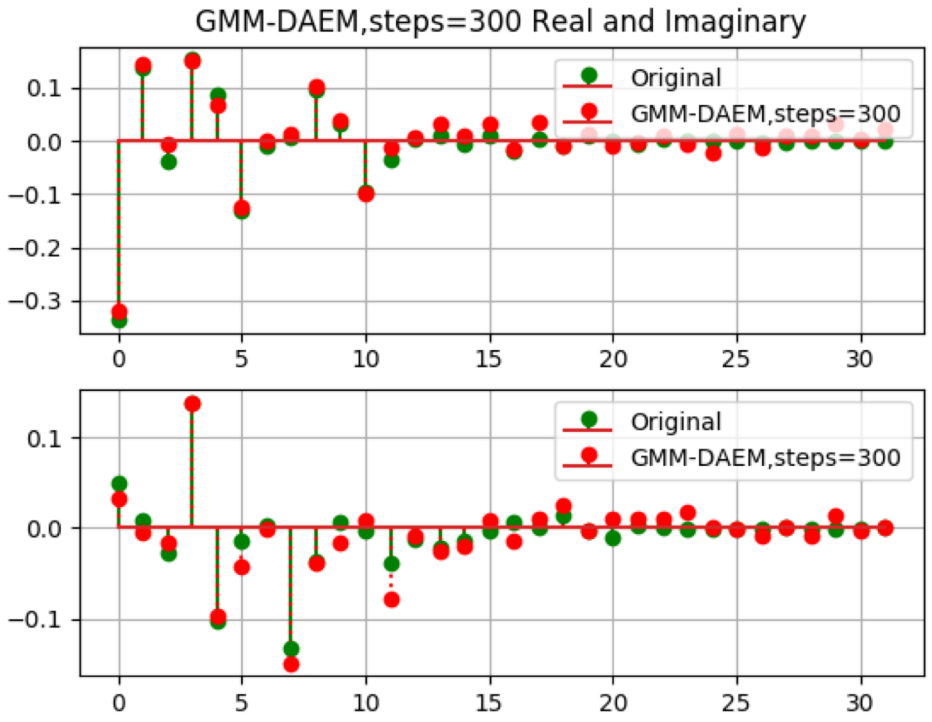


SE = 0.0324

Simulation Results



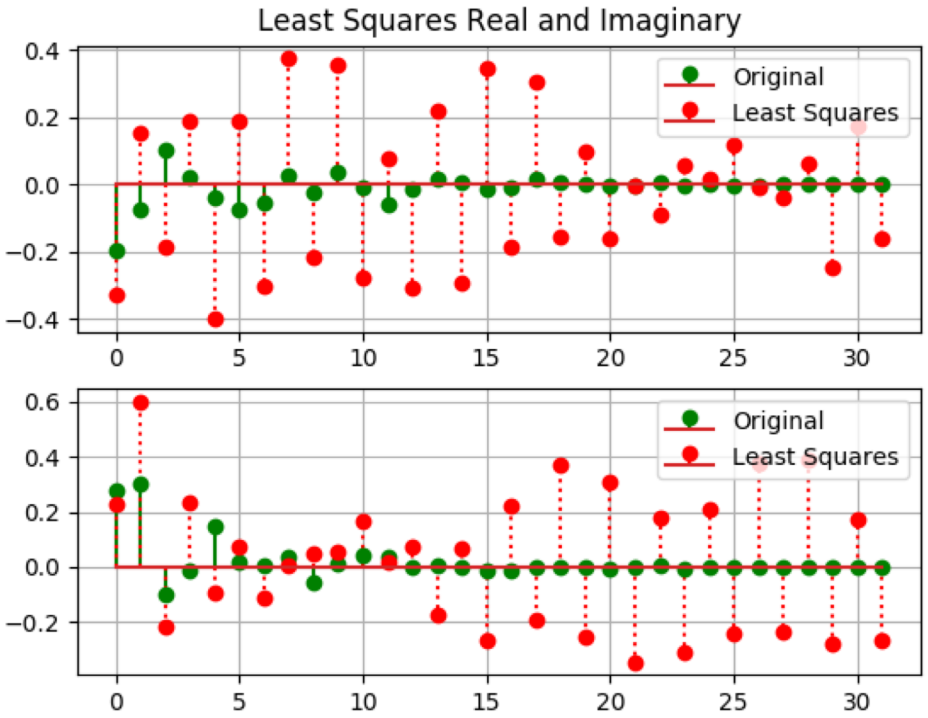
SE = 3.7392

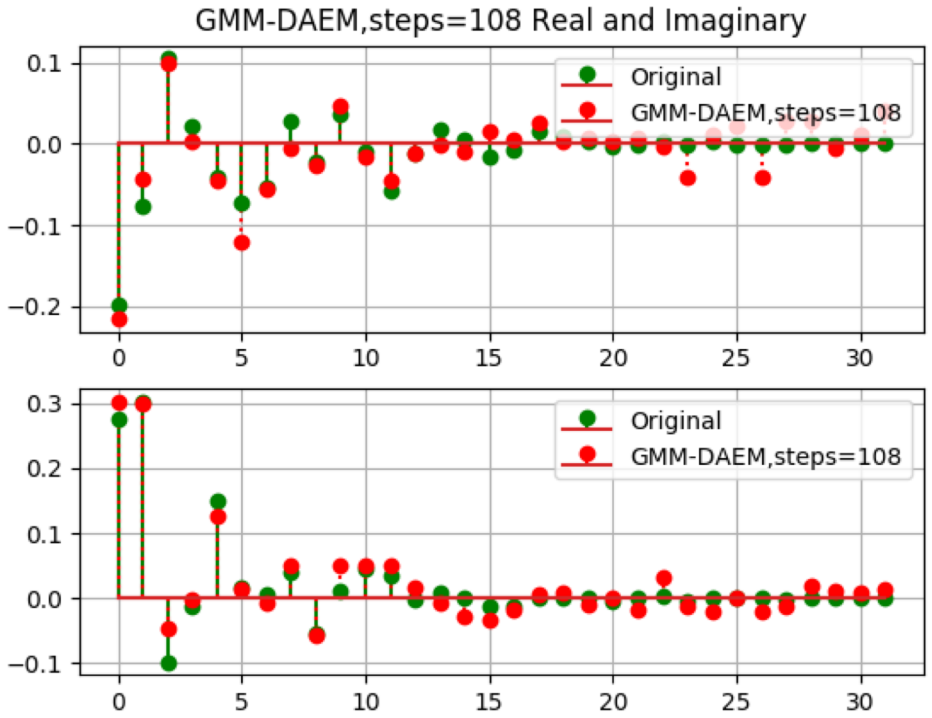


SE = 0.0133

Works well even with unknown non-zero mean. True mean of noise = 4.

Simulation Results





SE = 3.10

SE = 0.026

Robust to randomly added Brownian noise on top of Cauchy Noise (PSD proportional to $1/f^2$)

$N = 256$