Statistical Signal Processing Project Report Group-7

Aarsh Prakash Agarwal
150004
aarshp@iitk.ac.in

Harshvardhan
150283
harshv@iitk.ac.in

Krishna Chandra
150337
krishc@iitk.ac.in

Shubhanshu Khandelwal
150705
shubhke@iitk.ac.in

Vishal Srivastava
150820
vishalsr@iitk.ac.in

Abstract

The project involves the comprehension and implementation of state of the art techniques involved in the estimation of direction of source. We ended implementing two state of the art techniques which involves different experimental settings. We will discuss about them in the upcoming sections.

1 Direction of arrival estimation using Information Geometry (1)

The paper explores the idea of using information geometry as the similarity measure between the observations.

1.1 Description of Matlab files

There are basically two files. The first file implements the algorithm and plots the spectrum while comparing its with state of the art MUSIC and MVDR algorithms, while the second file implements the Monte Carlo simulation and compares their performance on different SNR values. Following is the description:

1.1.1 doa.m

The file implements the signal model and IG algorithm, and compares its spatial spectrum with MUSIC and MVDR. MUSIC and MVDR are implemented via in-built phased library functions present in MATLAB.

1.1.2 monte_carlo_doa.m

The file implements the Monte Carlo simulation for all the three algorithms. The simulation is run for different values of SNR depending upon the problem. Sources are kept to 2 in this file.

1.2 Description of Functions and Modules from MATLAB

In the paper we had to compare the results of the proposed method IGPencil as against standard methods of source angle estimation MUSIC and MVDR. The implementation of MUSIC and MVDR is already present in the **Phased System Array Toolbox** of MATLAB. Functions used in the setup model are as follows:

- **phased.ULA** : For modelling a uniform linear array (ULA) containing 11 isotropic antennas spaced 0.5 meters apart.

Submitted for course Project EE-602 at Electrical Engineering department, IIT Kanpur

- **sensorsig** : For the multichannel signal received by ULA.
- **phased.MVDREstimator** : Scans an MVDR beam over the specified region
- **phased.MUSICEstimator** : Scans a MUSIC beam over the specified region

1.3 Steps of Execution

There have been two major implementations done by us:

- **IGPencil** : In this method we need to plot the spectrum of

$$f(\phi) = \frac{1}{(\log \boldsymbol{a}(\phi)^H \hat{\boldsymbol{R}}_{xx}^{-1} \boldsymbol{a}(\phi))^2}, \qquad \phi \in [-\pi/2, \pi/2] \tag{1}$$

  and the source angles could be estimated from the maximas of the spectrum.
- **Monte Carlo** : We have to run the IGPencil, MUSIC and MVDR models for 1000 different signals at a particular SNR which has to be varied in a certain range. Then we plotted the the graph of MMSE vs SNR (in dB) for measuring the statistical performance.

The application of algorithm can be broken down in following steps

1. Create a uniform linear array model of 11 elements placed at separation of $\lambda/2$ with equal power of unity. Allocate sources to the desired directions and generate 100 instances of signal at each element with desired level of White Gaussian Noise.
2. Proceed to calculate the energy covariance matrix $\hat{R}_{xx}$ which is just equal to $signal * signal^H$. This would serve the purpose of true covariance matrix.
3. For an angle span of $[-90°, 90°]$, calculate the IG Pencil function 1 for each degree.
4. Maximas of the above function will give you the direction of the sources.
5. Normalize the function and calculate the spatial spectrum, in decibles.
6. Compare it with algorithms, MUSIC and MVDR (implemented using in-built functions) by plotting the spatial spectrum.

The Monte Carlo simulation and comparisons can be done in following manner.

1. Create 20 divisions for the range of SNR you want to evaluate. For each chosen SNR in the range, generate 100 samples (Monte Carlo samples) of signal. Sources are limited to two in thses simulations
2. Implement all three algorithms and calculate parameters such as estimated angle, variance, root mean squared error for each SNR value for each source.
3. Plot the root mean squared error curve and the estimated angle for each algorithm for each source.

1.4 Experimental Setup

Set up a uniform linear antenna array of 11 elements spaced 0.5 $\lambda/2$ metres apart. For the first experiment SNR is kept at 10dB and the sample size at each antenna is kept at 100. For Monte Carlo simulations, at each SNR, 100 samples of signal model is generated. We'll run the code for 4 setups with varying number of sources and varying the values of SNR as well.

1. Generate a set of 10 sources with equal unitary power and uniform separation in degrees from range $[-60°60°]$ buried under white noise of SNR 10dB and a set of K = 100 time snapshots have to be considered.
2. Take two sources at angles $[-20°30°]$ where the noise is varied for the sources. For simulations we conduct 1000 Monte Carlo experiments with SNR from range -20dB to 20dB.
3. Take two sources at angles $[-20°25°]$ where the noise is varied for the sources. For simulations we conduct 100 Monte Carlo experiments with SNR from range -5dB to 40dB.

4. Lastly take 13 sources with equal unitary power and uniform separation in degrees from range $[-60° 60°]$ buried under white noise of SNR 10dB and a set of K = 100 time snapshots have to be considered.

1.5 Results

Performing the first experiment in which 10 sources were located at equidistant from each other with in the range of $[-60° 60°]$, we compared the spatial spectrum of all the three algorithms, IG, MUSIC, and MVDR as shown in Figure 1. It was found out that the performance of IG method was at par or similar to MUSIC and MVDR methods.

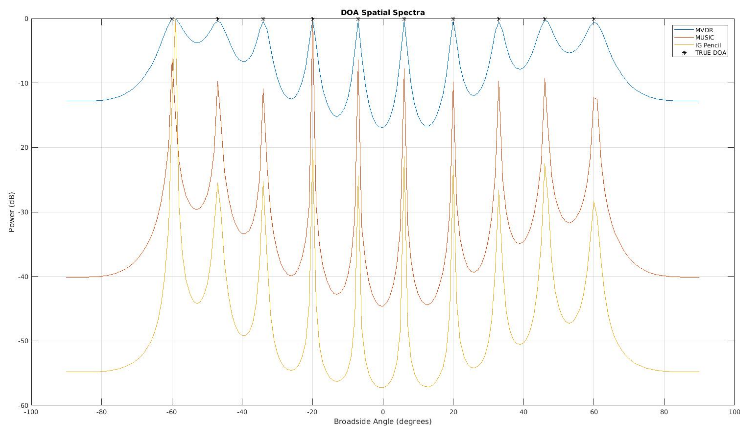

Figure 1: Spatial Spectrum Comparison of IG with MUSIC and MVDR when 10 sources are kept equidistant from $[-60°, 60°]$

To investigate further, we ran Monte Carlo simulations for two sources kept far apart at $\theta = (-20°, 30°)$. The SNR for this experiment was varied between $[-20, 20]$dB.To measure the performances, we plotted the mean square root error v/s SNR as shown in Figure 2. It can be seen that as we increase the SNR, performance of IG exactly follows the MVDR and also, at lower levels it is comparable to MUSIC, and MVDR.

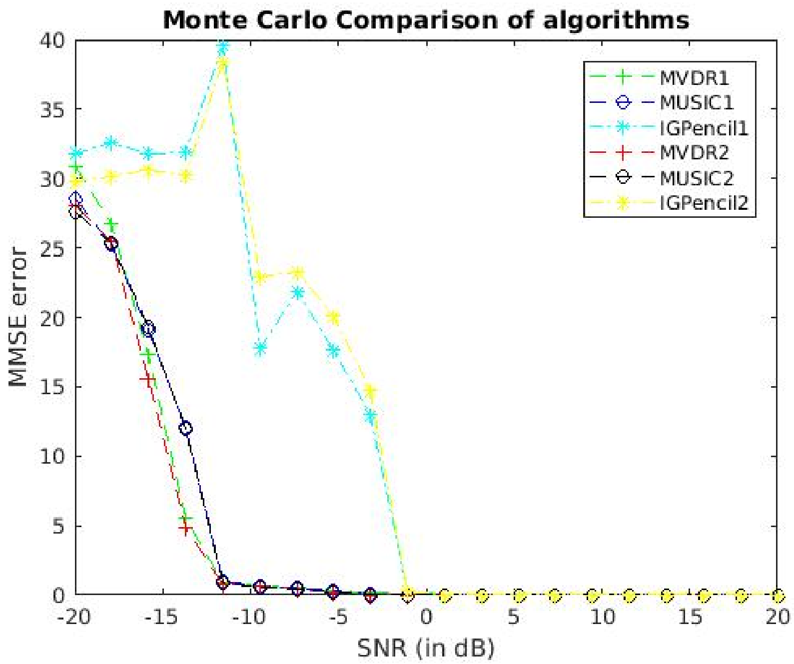

Figure 2: Root Mean square error comparison of Monte Carlo simulation of IG with MUSIC and MVDR at different SNRs in range $(-20, 20)$ dB when 2 sources are kept far apart at $(-20°, 30°)$

But, when sources are placed closer at $\theta = (-20°, -25°)$, and compare the performances for SNR range $(-5, 40)$ dB as shown in Figure 3, we found that the error for IG method performed much better than MUSIC, or MVDR especially at lower SNR values. Comparing the error graphs of all three methods in th figures 4, 5 and 6, the argument that IG provided the better resolution at lower SNR values is strengthened.

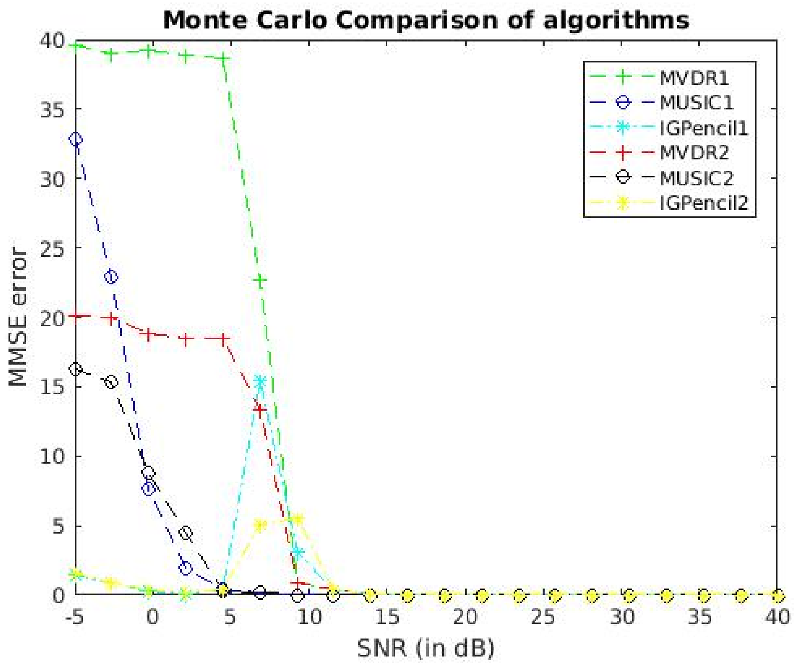

Figure 3: Root Mean square error comparison of Monte Carlo simulation of IG with MUSIC and MVDR at different SNRs in range $(-5, 40)$ dB when 2 sources are kept close at $(-25°, 20°)$

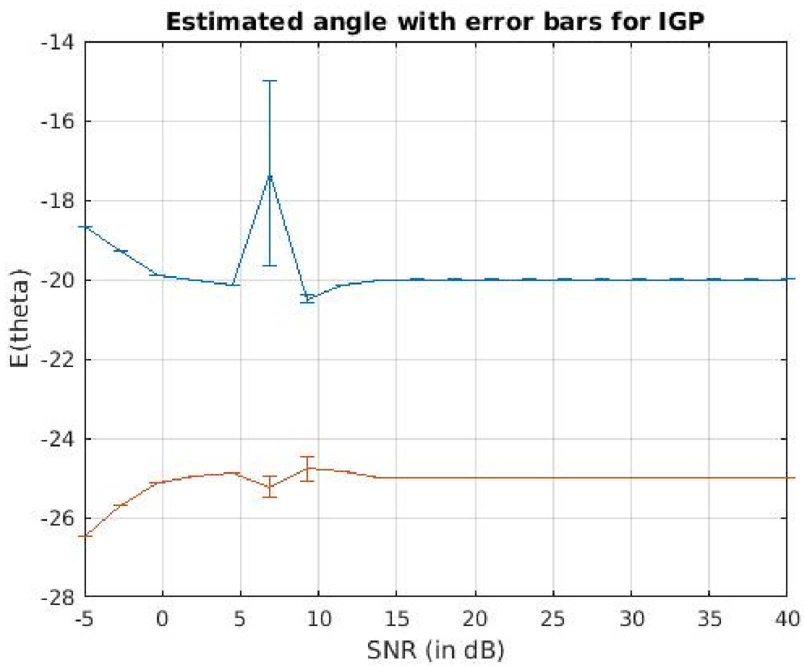

Figure 4: Error graph: IG pencil method

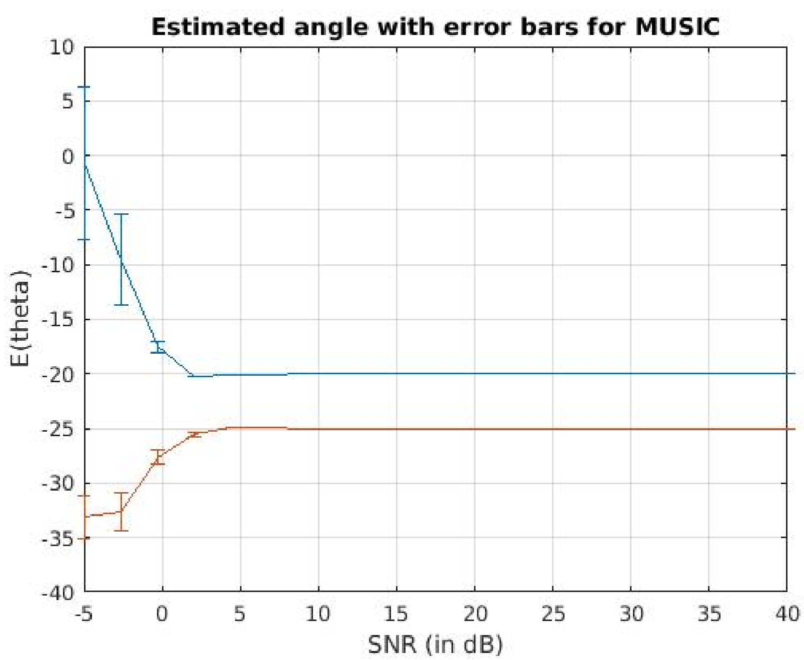

Figure 5: Error graph: MUSIC

Finally, we increased the number of sources distributed to 13, taken at SNR of 10 dB. MVDR and MUSIC even failed to recognize more than 8 sources. But, IG method identified all the sources and located them correctly. It can be seen from spatial Spectrum plots in Figure 7

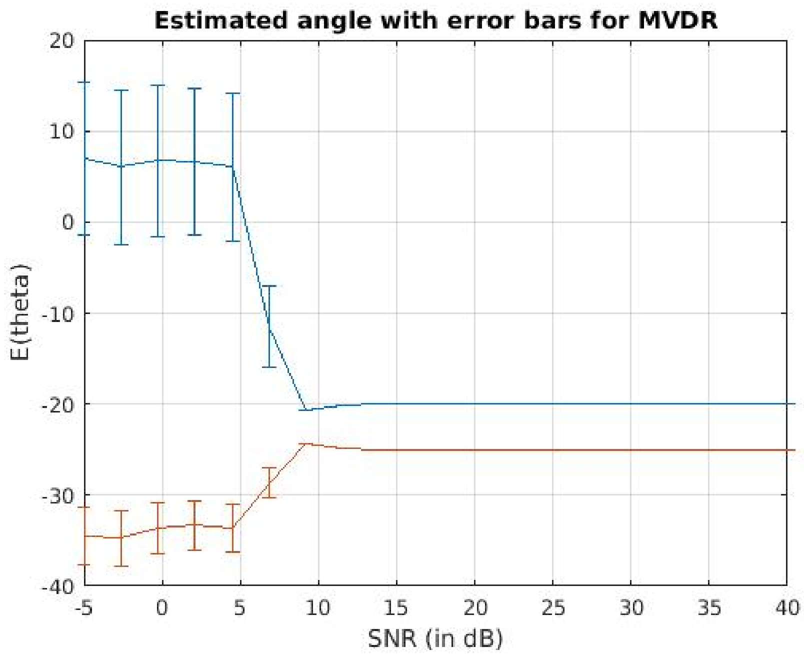

Figure 6: Error graph: MVDR

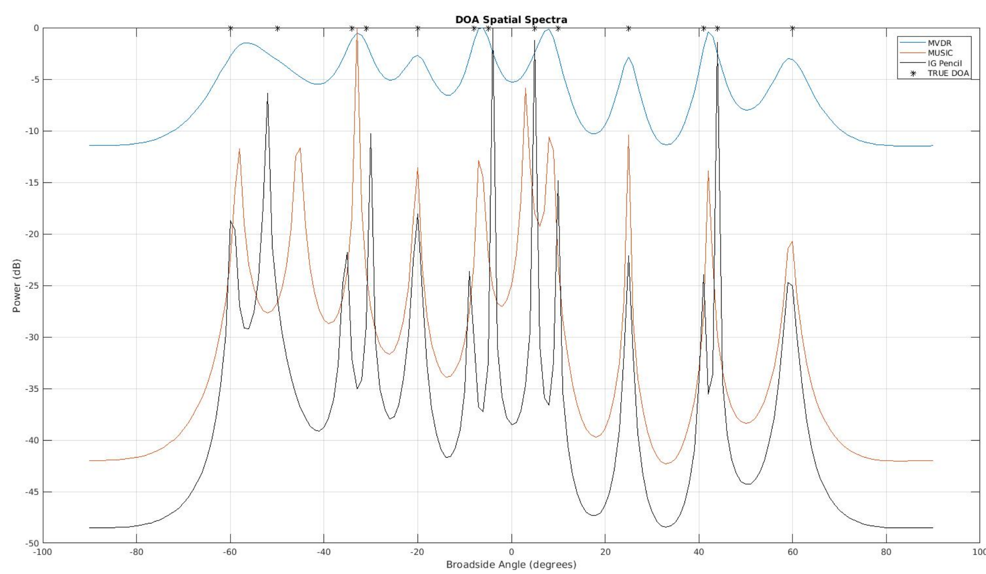

Figure 7: Spatial Spectrum Comparison of IG with MUSIC and MVDR when 13 sources are kept in $[-60^{\circ}, 60^{\circ}]$

1.6 Report from MATLAB

Published report from MATLAB is attached at last.

2 Source tracking using moving microphone arrays for robot audition (2)

2.1 Description of MATLAB files and functions

The following files were used while implementing the algorithm:

2.1.1 calc_distance.m

Function that takes two matrices as input and returns the maximum of the absolute value of the difference between the two.

2.1.2 complex_gauss.m

Function that takes point and covariance matrix (as a complex number) as input and returns the probability of the point from the circular Gaussian distribution with zero mean and given covariance matrix.

2.1.3 SSP_EM.m

This file runs the subroutine EM algorithm for at max 50 iterations. It uses certain threshold for the convergence. This file assumes following input from file **sampler.m**[2.1.4]

- microphone position
- J sampled state(x, y, v_x, v_y) of particles.(size = $4 \times$ J)
- STFT of the signal recieved by microphones.(size = $2 \times$ K; K = no. of frequency bins.)
- ϵ to calculate the $\Gamma(t, k)$ for each instant t and each frequency bin k.

This subroutine returns probability vector which inturn is used to update weights $w_j^{(t)}$. This subroutine calls complex_gauss and calc_distance as helper functions.

2.1.4 sampler.m

This is the main script for the second paper's implementation. It defines the 3 parts of the algorithm, source motion, sampling possible source positions and updating the weights for the sampled source position using SSP_EM function on the convolution of the RIR and STFT of the observed speech signal. Then it plots the KDE contour plots for each time step.

2.1.5 stft.m (3)

Function takes in the time domain signal values, sampling rate, window, and hop length(here it is the same as window length) and outputs the STFT of the given signal. This function has been ported from MATLAB File Exchange.

2.1.6 RIR (4)

Function has been ported from the RIR library. It computes the RIR for given source-microphone geometries when the reverberation time and room dimensions have been supplied.

2.2 mvnpdf

Function generates the given number of samples from a multivariate normal distribution with given mean and covariance matrix.

2.3 ksdensity

Calculates the probability density for given set of sampled points and their corresponding weights over a given region using the Kernel Density Estimator. Gives probability density values for the pdf approximated as

$$p(\mathrm{s}(t)|\mathrm{Z}_{1:t},\ \phi_y(t,\ k),\ \phi_R(t,\ k)) \approx \sum_{j=1}^{j} \tilde{w}^{(j)}(t)\delta_{\hat{\mathrm{s}}^{(j)}(t)}(\mathrm{s}(t)), \tag{15}$$

2.4 Steps of Execution

- We generate the microphone positions from its initial and final positions and the source model according to the given equation for a Langevin Model.

  $$\mathrm{s}(t) = \mathrm{F}(t)\mathrm{s}(t-1) + \mathrm{u}(t),\ \mathrm{u}(t) \sim \mathcal{N}(0_{4\times 1},\ \mathrm{Q}(t)) \tag{2}$$

  The initial positions for the actual source and the sampled source positions are randomly assigned inside the room and a 2s signal from the TIMIT database.
- For each time step the following actions are taken –
  - The new true source position is sampled from the source model.
  - STFT of the part of signal observed in this time step is generated and convolved with the RIR generated for the current source-microphone positions.
  - For each frame of the STFT, we sample the J possible source positions according to the source motion model. Then for this J-long set of sampled positions, we calculate the weights to approximate the source position distribution using SSP_EM function. Note: We also remove the zero valued frequency components from STFT
  - We plot the kernel density contours for the current sampled source positions using the ksdensity function.

2.4.1 EM

The file assumes some information like speed of sound and sampling frequency, and calculates Γ and $\mathbf{h}$. For calculating Γ the diagonal loading factor is taken to be $\epsilon = 0.1$. It then finds $\phi_{R,j}(t,\ k)$ and $\phi_{y,j}(t,\ k)$ and $\Phi_j(t,\ k)$ for each of the j and k. Using $\Phi_j(t,\ k)$ it goes to while loop and finds μ and ψ using μ alternately, until convergence. The equations for both of them is given as-

$$\mu^{(\ell-1)}(t,\ k,\ j) \triangleq \mathbb{E}[x(t,\ k,\ \theta_j)|\mathrm{z}(t,\ k), \theta^{(\ell-1)}] = \frac{\psi_j^{(\ell-1)}\mathcal{N}^c(\mathrm{z}(t,\ k)|0_{2\times 1}, \Phi_j(t,\ k))}{\sum_{j=1}^{J}\psi_j^{(\ell-1)}\mathcal{N}^c(\mathrm{z}(t,k)|0_{2\times 1}, \Phi_j(t,k))}$$

and

$$\psi_j^{(\ell)} = \frac{1}{K}\sum_{k=1}^{K}\mu^{(\ell-1)}(t,\ k, j)$$

Finally after convergence it finds the probabilities $p(\mathrm{Z}_t|\theta_t^{(j)})$ that are returned.

2.5 Experimental Setup

- Source, microphone and room params : Experimental setup consist of a 6m $\times$ 6m $\times$ 3.5m dimensional room. The source state is modelled by Langevin motion model. Source position is observed after every 0.375s. The 2 microphone start form ([1.35,1,1.5]; [1.65,1,1.5]) and are moving in a straight line upto position ([4.85,1,1.5];[5.15,1,1.5]) for 10s. The initial positions for the source is selected randomly uniformly from a region which is $1.5m$ away from each of the walls. The source motion model params are $\bar{v} = 1m\backslash s$ and $\beta = 2$.
- Signal : 10s signal generated from TIMIT database by joining 3 signals and clipping it till 10s. However, we analyse only the first 2s of this signal to reduce the time taken for computation. $fs = 8kHz$ for the 10s sound signal. For the signals of the timit database, sampling is done at 16kHz, so we downsample these signals.
- RIR params : Speed of sound taken as $c = 340$ and the reverberation time is $0.5s$
- The number of sampled source positions are $J = 100$ which has been reduced from that mentioned in the paper to reduce computation time. The initialisation for these J points is similar to the initialisation of the true source. We do not ignore the sampled points which lie

outside the room but count their number and check their value at the end to see how valid our assumption was. The model for sampling was same as the source motion model but the interval between 2 samplings is now the frame length so the Q and F matrices were changed accordingly to accomodate this.

- STFT params : We set the number of points for STFT as 512 and rectangular window with frame length 50ms. We take a hop equal to frame length, thus preventing overlap.
- We remove all frequency components which have 0 contribution from the STFT as these result in singularities when using the SSP_EM procedure.
- ksdensity is calculated only for the region of the room for 61 points along each axis.

.

For EM no extra parameter was required, all the required parameters were known from sampler file. The ψ vector of length J was initialized by constant vector of $1/J$, Using this vector the alternating optimization scheme was initialized. The only assumption taken was to calculate $w^{(j)}(t) = w^{(j)}(t-1)\psi_j^{(L)}\mathcal{N}^c(\mathrm{z}(t,\ k)|0_{2\times 1}, \Phi_j(t,\ k))$ as $w^{(j)}(t) = w^{(j)}(t-1)\psi_j^{(L)} \max_k(\mathcal{N}^c(\mathrm{z}(t,\ k)|0_{2\times 1}, \Phi_j(t,\ k))$. We should have taken $w^{(j)}(t) = w^{(j)}(t-1)\psi_j^{(L)} \prod_k \mathcal{N}^c(\mathrm{z}(t,\ k)|0_{2\times 1}, \Phi_j(t,\ k))$ but it was giving infinite values of weights. The convergence criterion is either 50 iterations of EM or either the maximum absolute difference between the elements of μ is less than 0.0001.

2.6 Results

The results differ a bit from the actual plots which can be because we have scaled down the problem to the level that it has become difficult to obtain convergence. The reductions in maxiter for EM algorithm and the length of the sound signal and the number of particles were done to get a reasonable computation time on a laptop. Thus, we might not have achieved optimum convergence conditions. Our code takes approximately 60 minutes for running.
Also, the number of points which lie outside the room among the true source positions and the sampled source positions are and which shows that some sort of truncation needs to be applied on the sampling procedure. We propose that the paper should have used a truncated gaussian (truncated to be always inside the room).
In the EM procedure, we take the maximum of the probability of the K components and not their multiplication as it leads to very large values many being Inf, which makes kernel density estimation impossible . One workaround for this can be clipping the probability density values to some large number but even then, the KDE performance would differ from optimal.
We thus believe that there were many important implementational details not mentioned in the paper and for these we had to make appropriate assumptions to get valid results. Some of these assumptions might be different from that made by the authors. Finally, the computational capacity used by the authors might be very large of which there is no mention in the paper . Published file from MATLAB is attached in the end.

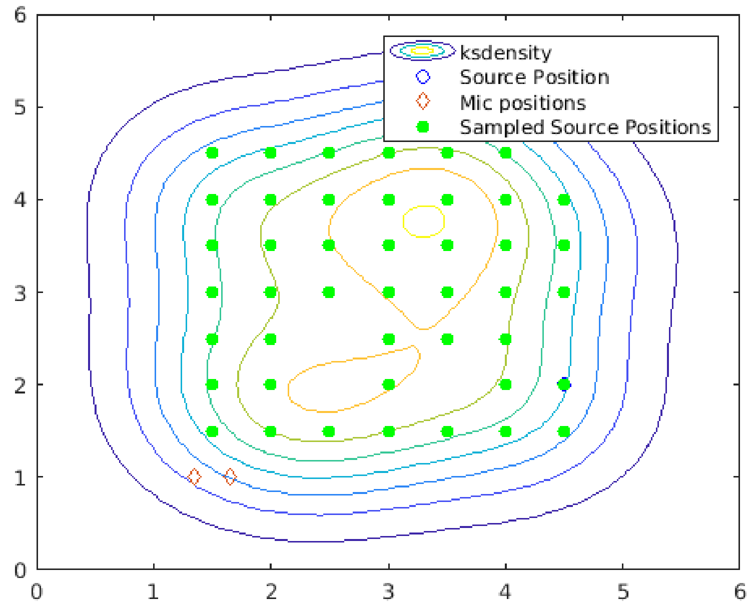

Figure 8: distribution of particles at t=0 sec

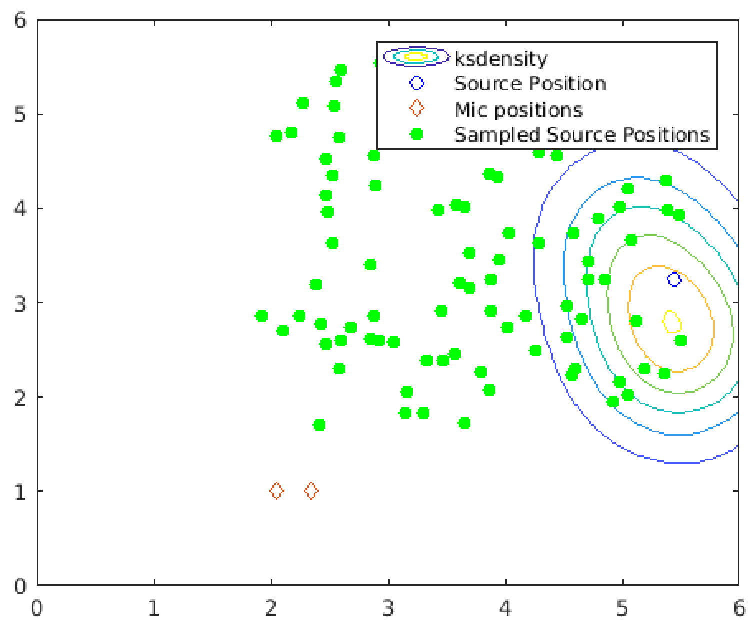

Figure 9: distribution of particles at t=0.375 sec

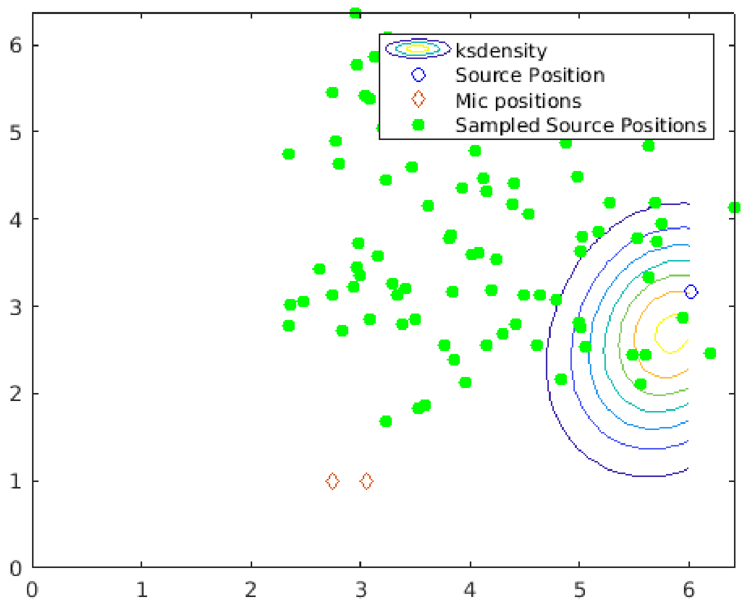

Figure 10: distribution of particles at t=0.75sec

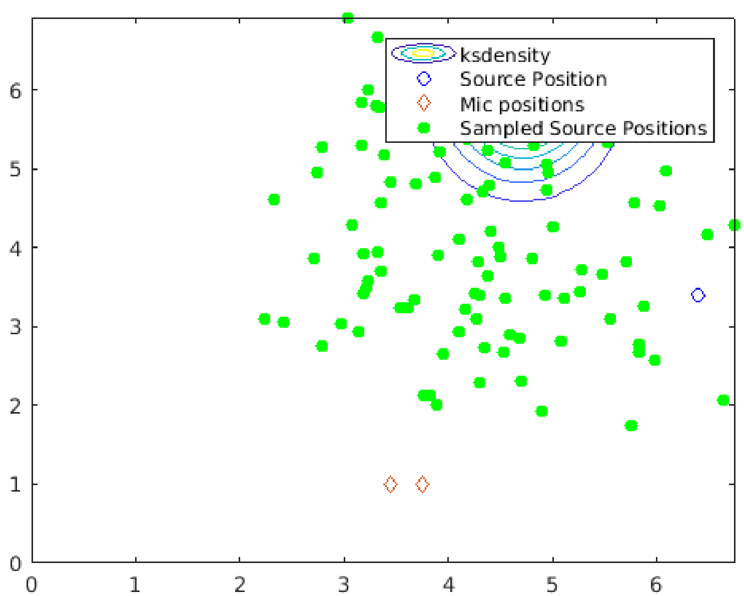

Figure 11: distribution of particles at t=1.125 sec

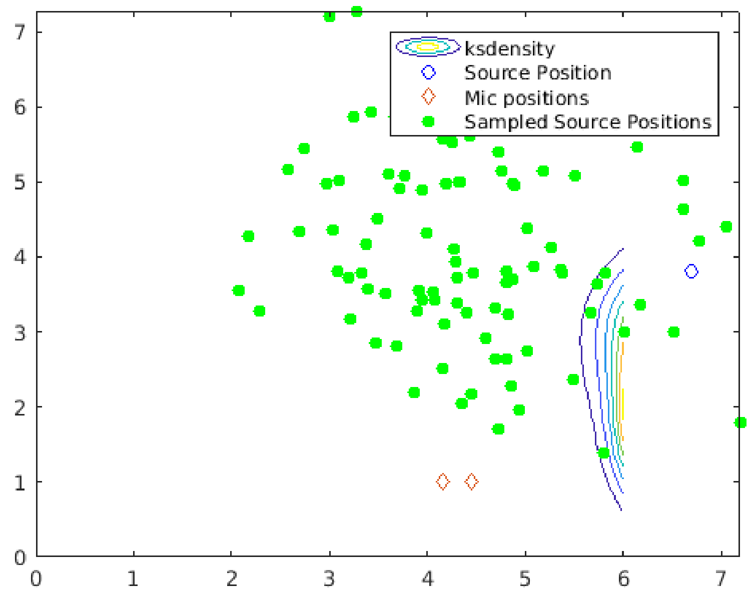

Figure 12: distribution of particles at t=1.5 sec

References

[1] M. Coutino, R. Pribic, and G. Leus, "Direction of arrival estimation based on information geometry," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3066–3070, 2016.

[2] C. Evers, Y. Dorfan, S. Gannot, and P. A. Naylor, "Source tracking using moving microphone arrays for robot audition," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6145–6149, 2017.

[3] H. Zhivomirov, "Short-Time Fourier Transform (STFT) with Matlab." `https://in.mathworks.com/matlabcentral/fileexchange/45197-short-time-fourier-transform-stft-with-matlab`.

[4] J. Allen and D. Berkley, "RIR Generator." `https://www.audiolabs-erlangen.de/fau/professor/habets/software/rir-generator`.

3 PUBLISHED REPORTS FROM MATLAB