TEXAS A&M
UNIVERSITY

INVESTIGATING MULTIMODAL DENSITY SAMPLING METHOD

REPELLING-ATTRACTING METROPOLIS

STAT 654 - Term Project - Group 8

Professor – Dr. David E Jones

Abhishek Kumar Sinha (529005619)
Abhishek Reddy Soma (230002596)
Shubham Satish Zope (930001594)
Srujan Jabbireddy (430000474)
SuryaNarayana Raju Nadumpalli (529005747)

MOTIVATION OF THE PROJECT

We selected this topic as we were intrigued by the Monte Carlo Markov Chain and Metropolis algorithm and wanted to explore more as they were highly applied in our domain field of industrial engineering. We saw an example of 'Sensor Network Localization' in our course in which the posterior density had multiple modes. The optimization method 'BFGS' had to be evaluated several times with different random starting values to explore all the solutions. The Metropolis Hasting algorithm did not do a good job of exploring the multimodal density and once converged to a stationary distribution was unlikely to jump modes. As a result, we wanted to explore a method that maintains the ease of implementation of Metropolis Hastings and more likely to jump between modes. As we were exploring, we came across RAM [1] - Repelling Attracting Metropolis Algorithm, which was developed by Hyungsuk Tak, Xiao-Li Meng, David A. van Dyk which suited our needs.

GOALS OF THE PROJECT

Our goals for this project were developed based on the motivation of the project i.e., to study the latest statistical advancement in metropolis hasting to explore multiple modes and their application in the practical world. Therefore, we divided our goals for the project as follows:

1. Study the developed alternative Metropolis Hastings multimodal sampler called the RAM method for Multimodality and develop an algorithm to test the model on the sensor network.
    a. This step involves reproducing the results in the 'A Repelling-Attracting Metropolis Algorithm for Multimodality' paper by 'Hyungsuk Tak'.
    b. We further went ahead and produced a simulated sensor network with noisy distance data and applied the RAM algorithm to see how the multimodal samples are generated in a larger network.
2. Next, we wanted to compare the computational time and speed of this algorithm of this new algorithm with the BFGS optimization method.
    a. We know that they are not similar techniques to compare, we wanted to see the advantages of RAM method in computational speed and time.
    b. This step was incorporated to apply the knowledge we learned as a part of this statistical computing course
3. In the final step, we implement the Metropolis-Hastings Algorithm and the RAM algorithm on a real dataset. We work upon the '7-dimensional posterior form an exoplanet detection problem'.
    a. A detailed performance analysis between the two algorithms is performed and reported.

1. REPELLING-ATTRACTING METROPOLIS

Repelling-Attracting Metropolis (RAM) is a Metropolis-Hastings algorithm with a unique joint jumping density and an easy-to-compute acceptance probability that preserves the target marginal distribution. This algorithm improves metropolis ability to jump between the modes more often than Metropolis using a jumping rule that is tuned to optimize Metropolis for the multimodal target, and with less tuning requirements than tempering methods.

RAM generates a proposal in an interesting way via forced downhill and forced uphill Metropolis transitions. In order to encourage downward move, the forced downhill Metropolis transition uses a reciprocal ratio of the target densities in its acceptance probability. The forced uphill Metropolis transition generates a final proposal with a standard Metropolis ratio making local modes attractive. Together, the downhill and uphill transitions form a proposal for a Metropolis-Hastings sampler.

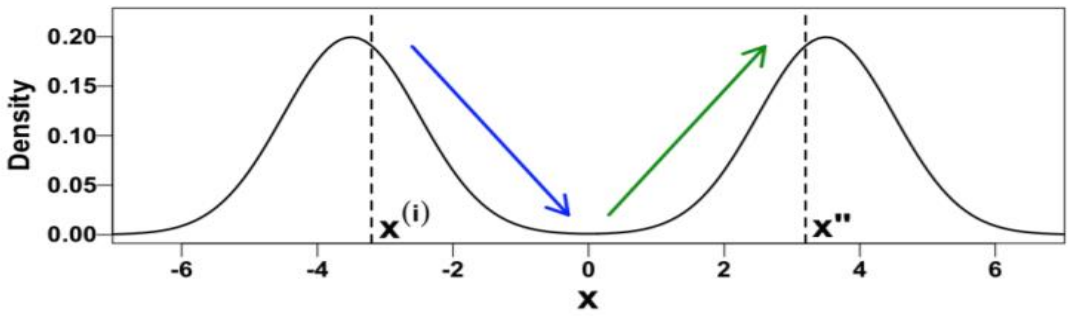

Fig 1: Downhill and Uphill Metropolis-Hastings sampler

Thus, RAM generates a proposal via three forced Metropolis transitions but accepts the proposal with an easy-to-compute acceptance probability.

RAM ALGORITHM

The concept of Repelling-Attracting Metropolis can compose into following 4 steps:

a) Steps 1-3 generates a joint proposal (z'', x'')
b) Step 4 accepts or rejects the joint proposal (z'', x'')
c) ε is introduced to avoid 0/0 values and its value is chosen to be 10e-308 (min. positive number in R)

Table 1: A repelling-attracting Metropolis algorithm.

Set initial values $x^{(0)}$ and $z^{(0)}$ $(= x^{(0)})$. For $i = 0, 1, \ldots$

Step 1: ($\searrow$) Repeatedly sample $x' \sim q(x' \mid x^{(i)})$ and $u_1 \sim \text{Uniform}(0,1)$ until $u_1 < \min\left\{1, \frac{\pi(x^{(i)})+\epsilon}{\pi(x')+\epsilon}\right\}$.

Step 2: ($\nearrow$) Repeatedly sample $x^* \sim q(x^* \mid x')$ and $u_2 \sim \text{Uniform}(0,1)$ until $u_2 < \min\left\{1, \frac{\pi(x^*)+\epsilon}{\pi(x')+\epsilon}\right\}$.

Step 3: ($\searrow$) Repeatedly sample $z^* \sim q(z^* \mid x^*)$ and $u_3 \sim \text{Uniform}(0,1)$ until $u_3 < \min\left\{1, \frac{\pi(x^*)+\epsilon}{\pi(z^*)+\epsilon}\right\}$.

Step 4: Set $(x^{(i+1)}, z^{(i+1)}) = (x^*, z^*)$ if $u_4 < \min\left\{1, \frac{\pi(x^*)\min\{1,(\pi(x^{(i)})+\epsilon)/(\pi(z^{(i)})+\epsilon)\}}{\pi(x^{(i)})\min\{1,(\pi(x^*)+\epsilon)/(\pi(z^*)+\epsilon)\}}\right\}$, where $u_4 \sim \text{Uniform}(0,1)$, and set $(x^{(i+1)}, z^{(i+1)}) = (x^{(i)}, z^{(i)})$ otherwise.

2. SENSOR NETWORK WITH RAM:

In order to implement the RAM algorithm, consider sensor network localization discussed in our coursework. Motivation behind selecting sensor network localization problem is, as it is known to produce a high-dimensional, banana-shaped, and multimodal joint posterior distribution. It involves searching for unknown sensor locations within a network using the noisy distance data.

The simulated distances y_{ij} among the six stationary sensor locations, x_1, x_2, x_3, x_4, x_5, x_6 are displayed if observed.
The observation indicator w_{ij} is one if y_{ij} is specified and is zero otherwise.
The locations of the sensors are
x_1 = (0.57, 0.91), x_2 = (0.10, 0.37), x_3 = (0.26, 0.14), x_4 = (0.85, 0.04), x_5 = (0.50, 0.30), and x_6 = (0.30, 0.70), where the first four locations, x_1, x_2, x_3, and x_4, are assumed to be unknown.

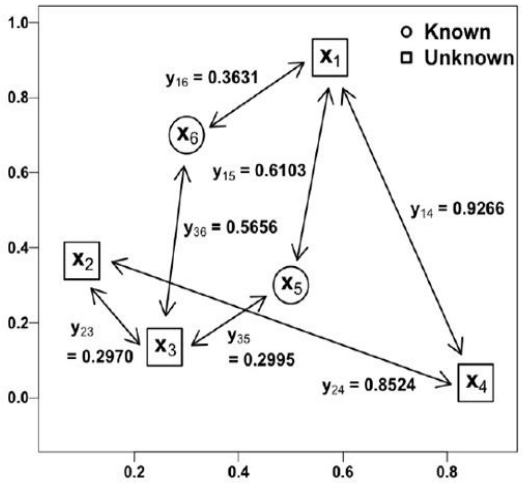

Fig 2: Sensor Network Localization Initialization

The Likelihood function is defined as:

$$L(x_1, x_2, x_3, x_4) \propto \prod_{j>i} \left[\exp\left(-\frac{(y_{ij} - \|x_i - x_j\|)^2}{2 \times 0.02^2} \right)^{w_{ij}} \times \exp\left(-\frac{w_{ij} \times \|x_i - x_j\|^2}{2 \times 0.3^2} \right) \times \left(1 - \exp\left(-\frac{\|x_i - x_j\|^2}{2 \times 0.3^2} \right) \right)^{1-w_{ij}} \right]$$

y_{ij} is the normal measurement error. $(x_i - x_j)$ is the distance between x_i and x_j. w_{ij} is an indicator variable.

Multiplying the likelihood with the prior, we get the posterior density. The full posterior distribution is

$$\pi(x_1, x_2, x_3, x_4 \mid y, w) \propto L(x_1, x_2, x_3, x_4) \times \exp\left(-\frac{\sum_{k=1}^{4} x_k^\top x_k}{2 \times 10^2} \right)$$

RAM algorithm was implemented within a Gibbs sampler for 500,000 iterations with the first 200,000 as burn-in. At first iteration proposal for the first 2 parameters was generated. At the next iteration we generated proposals for the third and fourth parameter given the updated value of 1st and 2nd parameter and so on. We used a normal proposal density and sd of the density was one of the tuning parameters. We set it to 1.08 for all the parameters.

The posterior samples of each unknown sensor location are plotted in trace plots, scatter plots and histograms. From the trace plots, we can observe the jumping nature of RAM. The trace plots are plotted after the initial burn-in period; hence the algorithm have converged to the different modes.

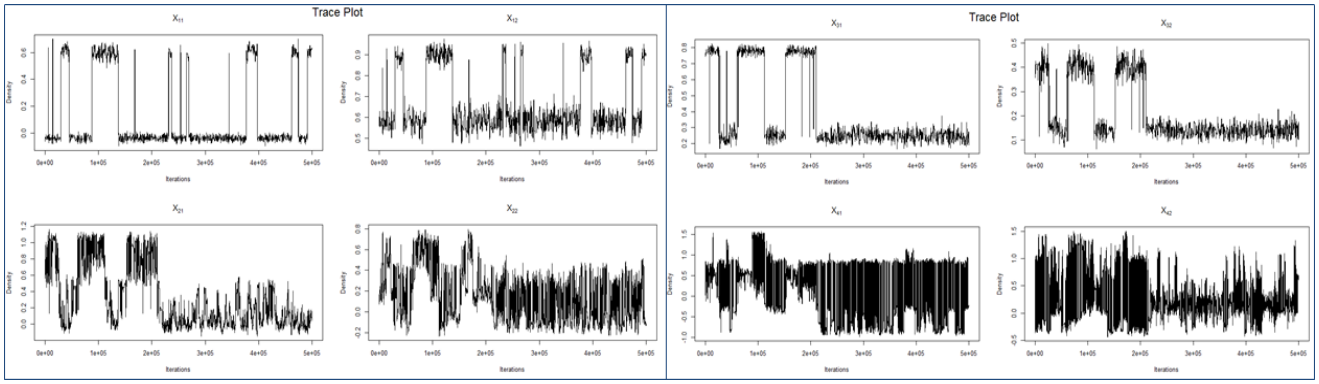

Fig 3: Trace plots for posterior samples

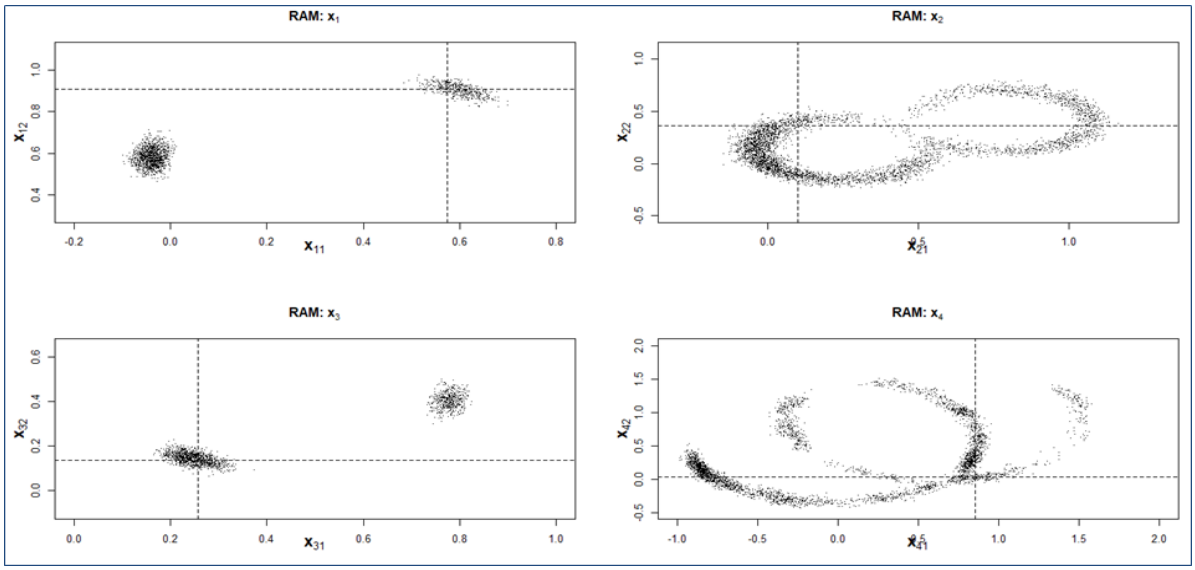

Fig 4: Scatter plots for posterior samples

All the 4 unknown points i.e. x_1, x_2, x_3 and x_4 is plotted in the graph. The black dotted horizontal lines show the true value of the parameters where the black dots show the distribution obtained from the RAM algorithm.

We can see that our algorithm does a good job of predicting the correct location of the 1st and the 3rd points. For x_1 and x_3 we can see clearly that there are 2 distinct modes. Whereas for x2 and x4, there seem to be multiple modes which are not well distinguishable. The bottom plot shows the density plots of the parameters.

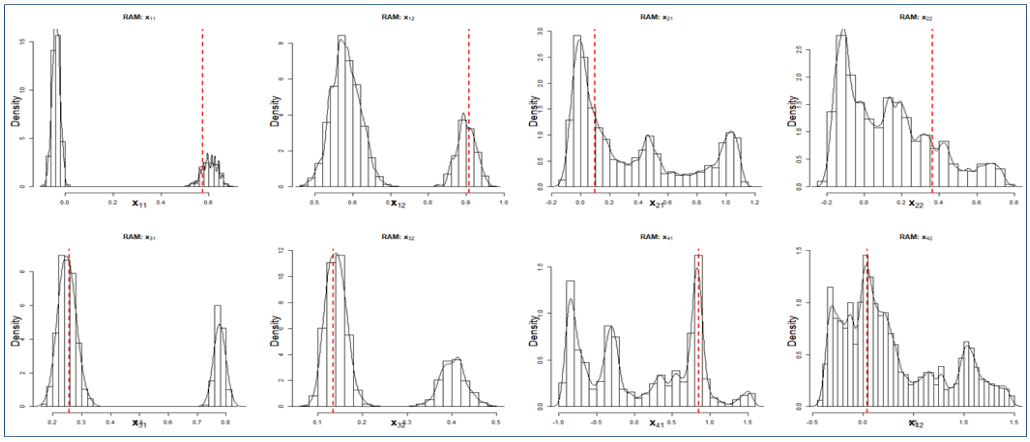

Fig 5: Histograms for posterior samples

Inferences

RAM algorithm does a good job of obtaining the expected densities in most of the cases namely in x_{12}, x_{31}, x_{32}, x_{41} and x_{42}. For the other cases the actual points are slightly off the modes. The reason for that may be the number of iterations or enough information was not available to predict the true values.

Parameters	No. of iterations	Burn in	Avg. No. of downhill proposals	Avg. no. of uphill proposals	Avg. No. of downhill proposals for z*	Total Proposals	Average acceptance rate	Total accepted proposals
X_1	500,000	200,000	1.0001	7.19	1.07	9.26	0.003625	1632.5
X_2	500,000	200,000	1.0003	6.54	1.07	8.61	0.0084	4200
X_3	500,000	200,000	1.0001	7.29	1.05	9.34	0.00348	1740
X_4	500,000	200,000	1.0003	6.98	1.13	9.11	0.007475	3737.5

Table: Performance statistic of the RAM on the sensor network localization problem

2.1 Comparison of RAM with BFGS Optimization Method

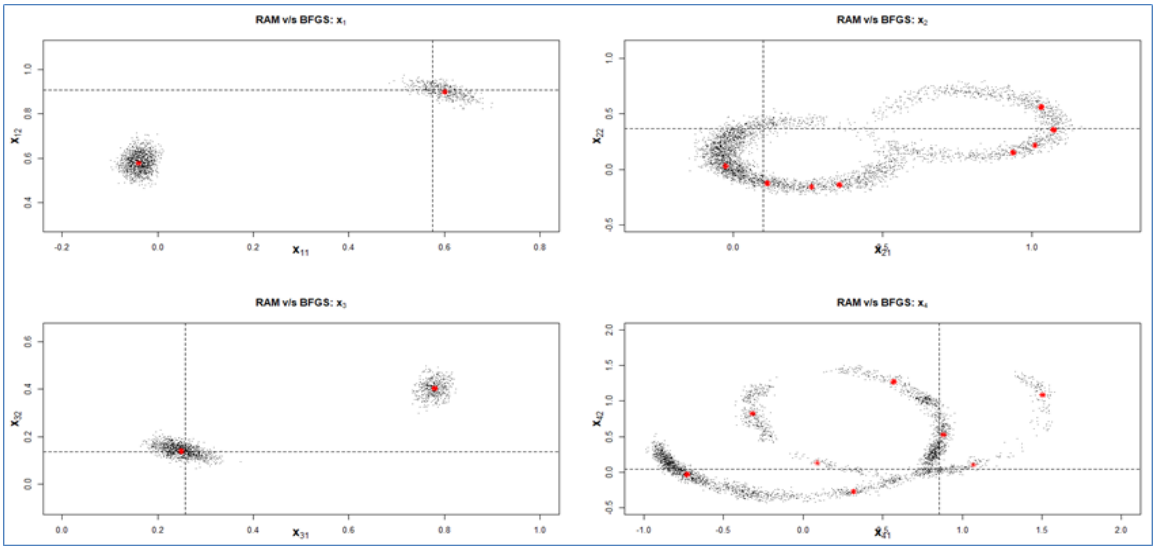

Fig 9: RAM with BFGS scatter plot

We compare the results of the 6 nodes sensor problem with that obtained from BFGS. BFGS is an iterative method for solving nonlinear optimization problems. It gives a point solution i.e. a single posterior mode in a run. RAM on the other hand gives sample from the posterior distribution. We ran the BFGS for 1000 times each time starting at different random values.

We can see in the result; the red point comes from the BFGS optimization whereas the black points are from RAM method. We observe that the BFGS gives results which are in the range of distribution obtained from RAM and hence consistent with our previous results. We would also like to conclude that in terms of accuracy, RAM is a better method than BFGS since BFGS just gives a point solution whereas RAM gives a distribution, which gives a greater understanding of the parameter values and its uncertainty.

2.2 Simulated Sensor Network Localization with 9 nodes:

We conduct an experiment like the previous example of sensor network localization. However, this is a simulated sensor network example containing 9 numbers of sensor nodes. We generated random points between 0 and 1. We calculated distance between all the points, added random normal error with sd 0.02 and then randomly removed some of the points. Here we consider points 7,8 and 9 to be known whereas points 1-6 are unknown. So, there are a total of 12 parameters to be estimated i.e. the x and y coordinates of the 6 points.

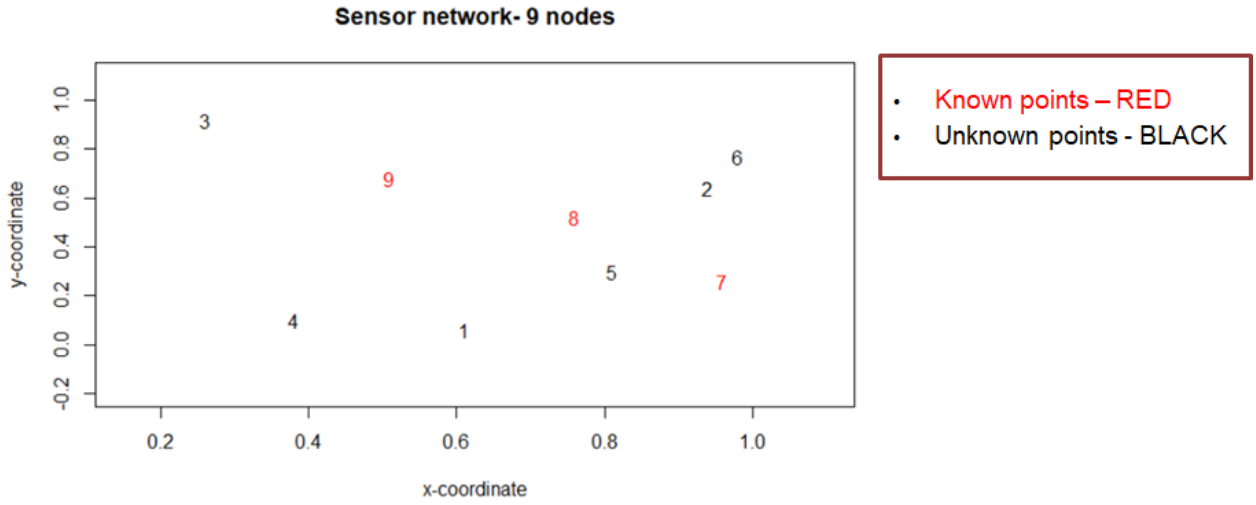

Fig 6: Sensor network with additional nodes

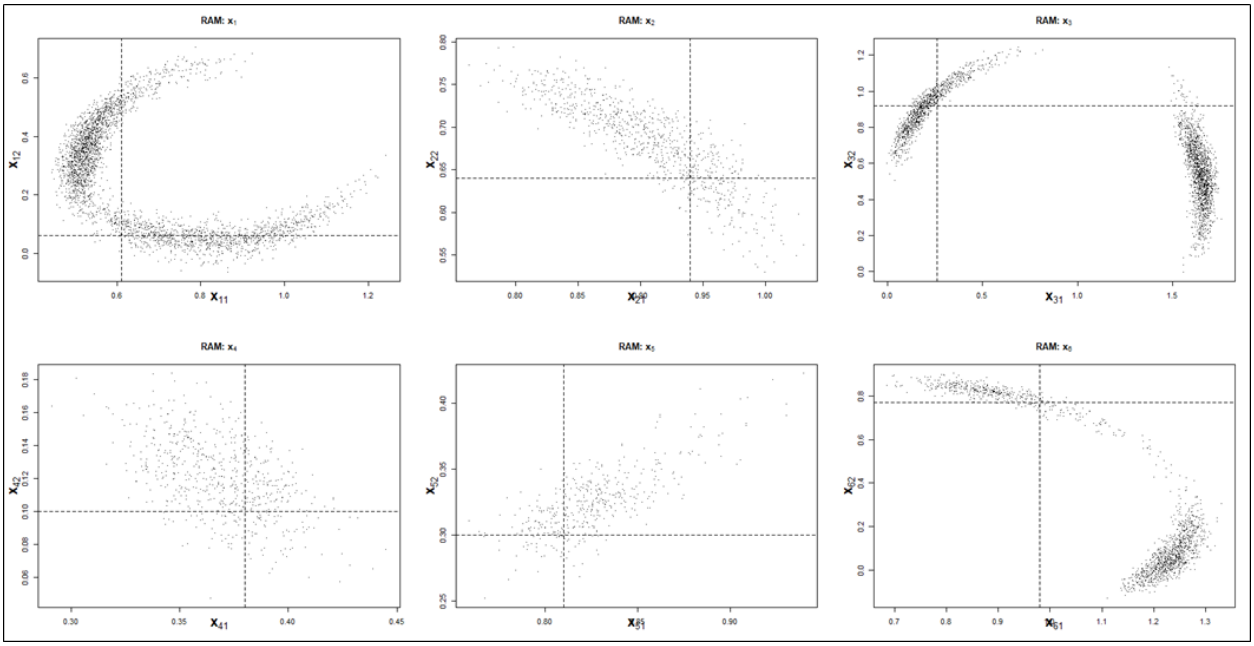

Fig 7: Scatter plots for new posterior density

We observe the scatterplots of the 6 parameters. The black dotted vertical and horizontal lines intersect at the true location. We see from the plots that the density of the parameters is much more distributed as compared to the previous example. We get a better estimate for the points 2 and 4 as compared to the other points.

One of the reasons may be that in this case a larger no. of parameters needs to be estimated and more distance data may be needed to have a posterior distribution that predicts the true value more accurately. Also, as it is a larger network, the computational time was much more and hence we had to limit the number of iterations which may have resulted in lower effective sampling size.

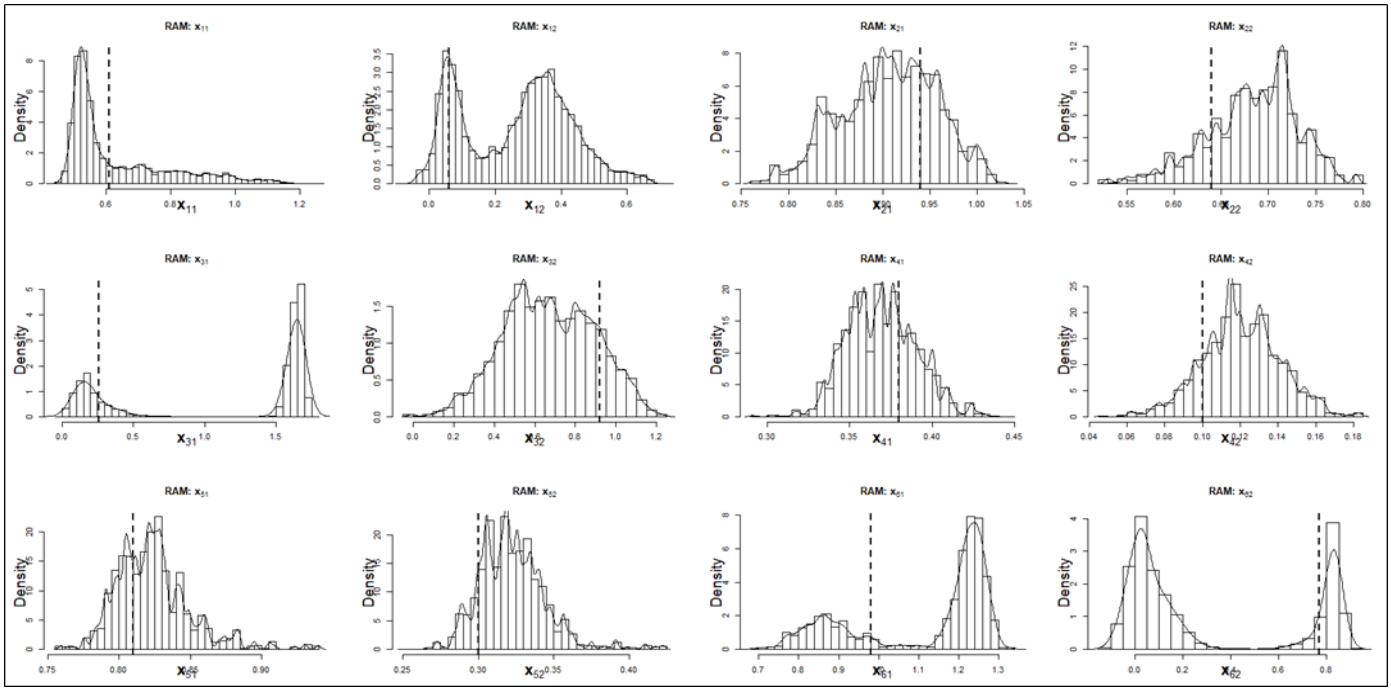

Fig 8: Posterior density histogram plots

The density histogram plots are observed in the above figure. The black vertical line represents the true locations.

3. Exoplanet: Posterior Sampling of a Simulated Radial Velocity Dataset

After applying the RAM algorithm on sensor network localization, we wanted to explore on a real dataset and work towards its usage in real world application. We explore exoplanet dataset which has 6 simulated datasets with anywhere from zero to three planets in each dataset. For purpose of our model generation, we use the first dataset and a one planet model. The Data set properties are as follows:

- The dataset is a simple timeseries, including the time of observation (t_i), measured radial velocity (v_i), and a measurement uncertainty (σ_i)
- Number of observations: 600 days
- The dataset includes between zero and three planets (inclusive)
- The dataset includes a single velocity offset and correlated, Gaussian noise to represent stellar activity

The Statistical model of exoplanet model is as follows:

Each simulated data point is generated according to

$$v_i = v_{\mathrm{pred}}(t_i|\theta) + \epsilon_i,$$

Where the first term is the velocity predicted at time t_i by a model parameterized by Θ and ε_i
The appropriate likelihood is a multi-variable normal distribution centered on the predictions of the model,

$$\log \mathcal{L}(\theta) = -\frac{1}{2}(\mathbf{v} - \mathbf{v}_{\mathrm{pred}}(\theta))^T \Sigma^{-1} (\mathbf{v} - \mathbf{v}_{\mathrm{pred}}(\theta)) - \frac{1}{2}\log|\det\Sigma| - \frac{n_{\mathrm{obs}}}{2}\log(2\pi)$$

The Gaussian noise is correlated from one observation to the next. The covariance matrix is given by

$$\Sigma_{i,j} = K_{i,j} + \delta_{i,j}\left(\sigma_i^2 + \sigma_J^2\right)$$

For the quasi-periodic kernel $K_{i,j,}$ we assume

$$K_{i,j} = \alpha^2 \exp\left[-\frac{1}{2}\left\{\frac{\sin^2[\pi(t_i - t_j)/\tau]}{\lambda_p^2} + \frac{(t_i - t_j)^2}{\lambda_e^2}\right\}\right],$$

Next, we define the priors for each of the 7 parameters.

Para.	Prior	Mathematical Form	Min	Max
$T(days)$	Jeffreys	$\frac{1}{T \ln\left(\frac{T_{max}}{T_{min}}\right)}$	39.81	44.66
$K(ms^{-1})$	Mod. Jeffreys	$\frac{(K+K_0)^{-1}}{\ln\left(\frac{K_0+K_{max}}{K_0}\right)}$	1.0	999.0
$V(ms^{-1})$	Uniform	$\frac{1}{V_{max}-V_{min}}$	-1000	1000
e	Uniform	1	0	1
ϖ	Uniform	$\frac{1}{2\pi}$	0	2π
χ	Uniform	1	0	1
$s(ms^{-1})$	Mod. Jeffreys	$\frac{(s+s_0)^{-1}}{\ln\left(\frac{s_0+s_{max}}{s_0}\right)}$	1	99

where T – Planet's Orbital Period
K – Planet's RV Semi Amplitude
e – Planet's eccentricity
w – Planet's argument of pericenter
X – Planet's mean anomaly
s – Additional white noise term
V – RV Velocity offset

The first one is the planet's orbital period for which we consider a Jeffrey's prior. The minimum and maximum value of this parameter depends on the dataset and on the number of planets considered in the model. These values are for a one planet model. The next parameter is Planet's RV Semi Amplitude who's prior is a Modified Jeffreys prior with value range from 1 to 999. The priors for the next 4 parameters i.e. RV Velocity offset, Planet's eccentricity, Planet's argument of pericenter and Planet's mean anomaly is a uniform distribution.

RESULTS:

First, we compare the trace plot of the results obtained by metropolis and RAM to show the contrast between both the methods in jumping between modes. Below is the trace plot for one of the parameters i.e. for 'Mean Anomaly' with both the RAM and Metropolis Hasting. We run both the methods for 10k iterations and burn-in period of 2000 iterations. We observe that the Metropolis has converged to a single parameter value whereas the RAM is jumping between 2 values.

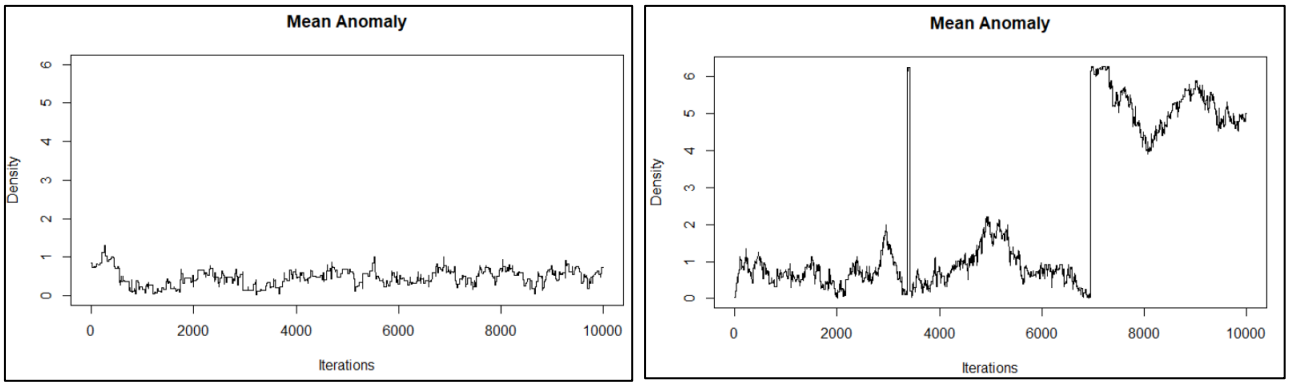

Fig 9: Trace plots: Metropolis Hastings (Left) and RAM (Right)

Below is the trace plot for 7 parameter values. We observe that all the parameters have converged. The trace plots are shown after a burn-in period of 2000 iterations. We also observe that the ‘Argument of epicenter’ and ‘Mean anomaly’ has multiple modes. All the other plots appear to have a single mode.

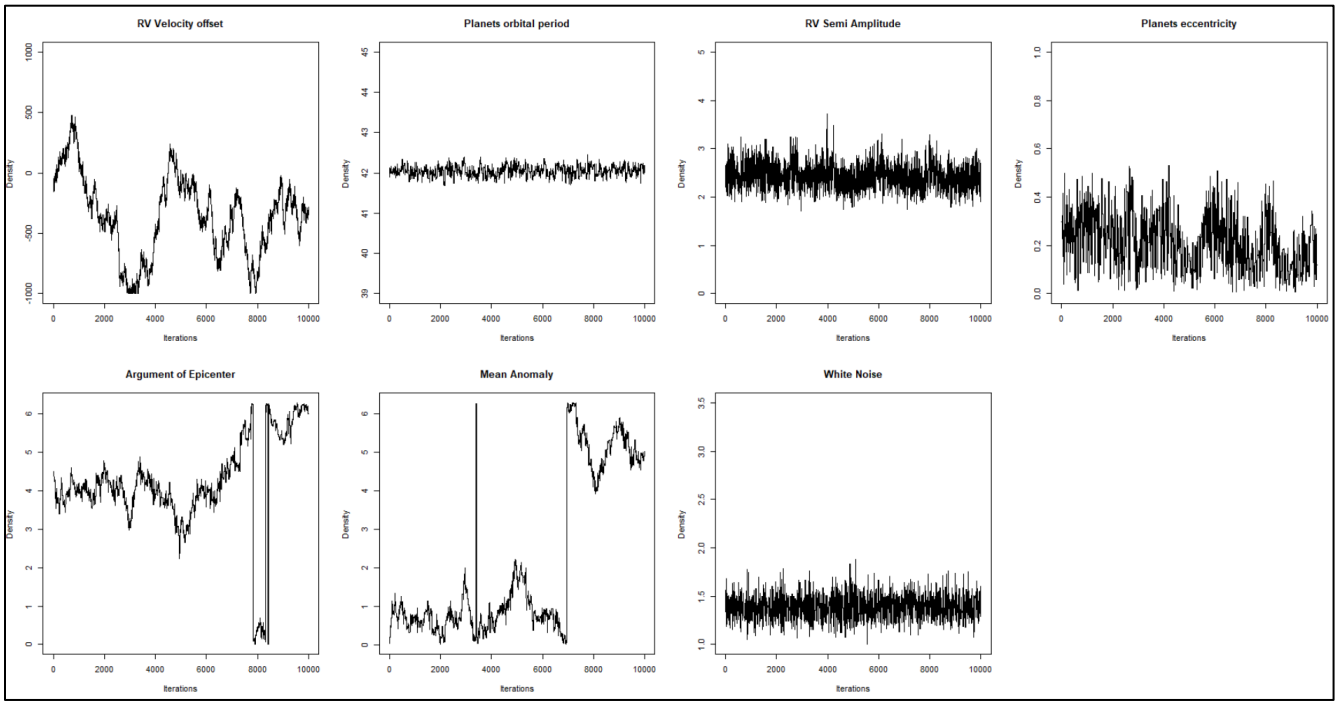

Fig 10: Final Trace plots

To get a better understanding we look at the density plots. One of the tuning parameters which is the sd of the proposal density for all the 7 parameters needed to be determined. Based on the range of the parameters, and after running the algorithm several times, we set this value of 10 for the first parameter and 2 for the rest of the parameters.

The most important parameter of interest in a planet’s orbital period, which as we can confirm from the distribution, appears to be converged at a value of 42.05.

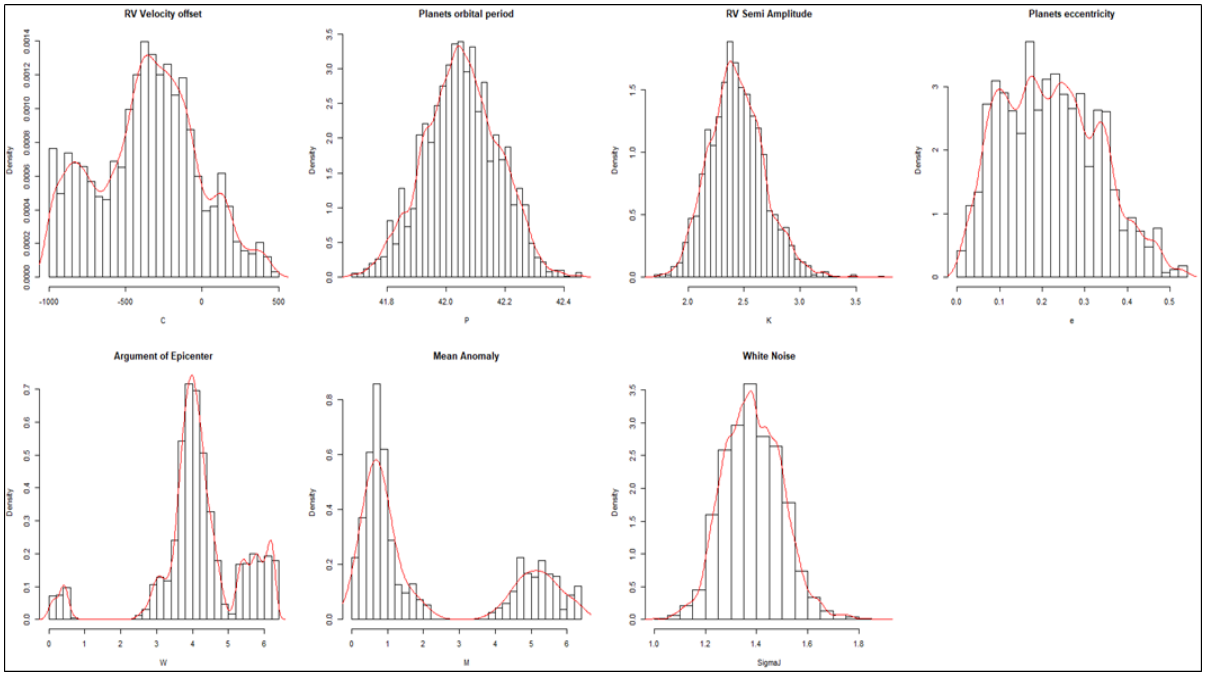

Fig 11: Final Histogram plot

REFERENCES:

1. Tak, H., Meng, X.-L., & Dyk, D. A. V. (2018). A Repelling–Attracting Metropolis Algorithm for Multimodality. Journal of Computational and Graphical Statistics, 27(3), 479–490. doi: 10.1080/10618600.2017.1415911
2. Loredo, T. J., Berger, J. O., Chernoff, D. F., Clyde, M. A., & Liu, B. (2012). Bayesian methods for analysis and adaptive scheduling of exoplanet observations. Statistical Methodology, 9(1-2), 101–114. doi: 10.1016/j.stamet.2011.07.005