Baysian Navigation

Nghia Vu Duc

11/08/2021

1. Introduction

Position resection and intersection are methods to determine an unknown geographic position by measuring angles with respect to known positions. In resection, the one point with unknown coordinates is occupied and sightings are taken to the known points. This project will demonstrate how these methods work in context of Bayesian Statistics.

This is the settings for the project, suppose you are at a location near Costco Wholesale(50.443492, -104.510223) and you do not know your position but you know coordinates of three other locations: Point1(50.484095, -104.579288) near Co-op Refinery Point2(50.5180655, -104.556831) near Kodiak LED Lighting Point3(50.450717, -104.610967)near Cornwall Center The coordinates are latitude and longitude respectively. YOur current location and reference points can be shown in the map as below.

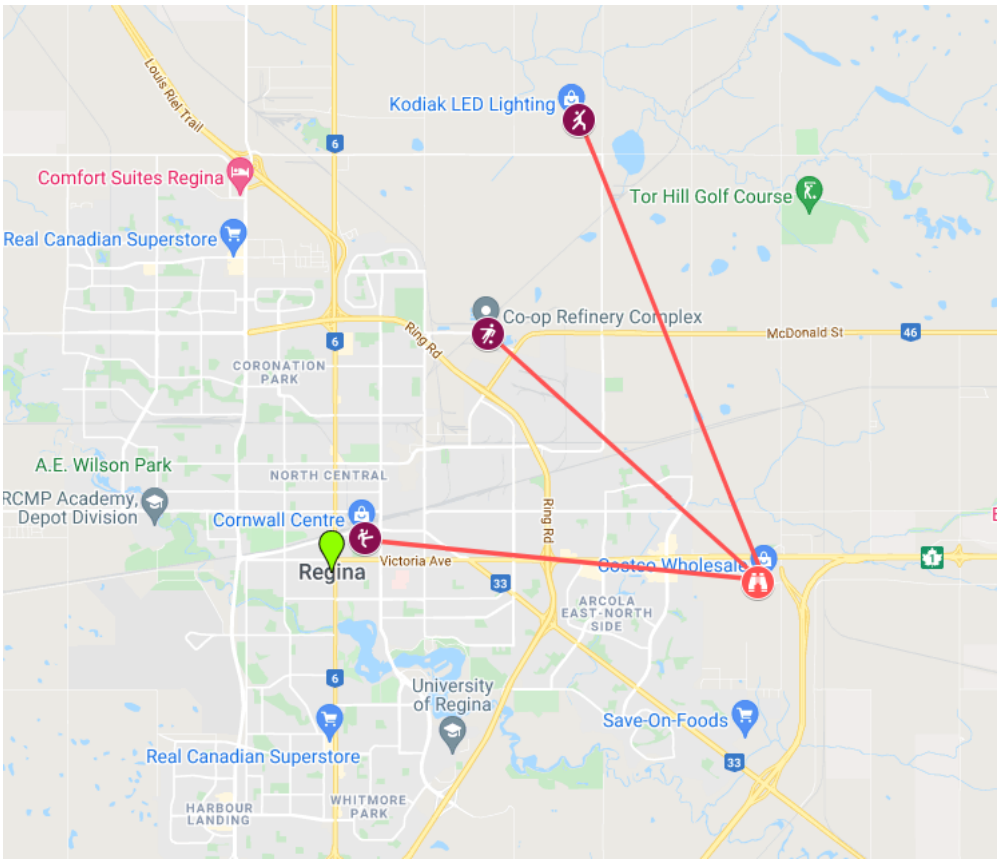


Figure 1: Figure 1:Reference Points

Use a compass to measure the bearings from your location to reference points, we can get three bearings respectively for Point 1, Point 2, Point 3: alpha <- 312 #bearing between your position and Point 1 beta <- 338 #bearing between your position and Point 2 gamma <- 276 # bearing between your position and Point 3

Ideally, the intersection of three lines will go though one point, that is your position. However, there are many factors that affect to our measurement then the intersections between lines creates a triangle as below.

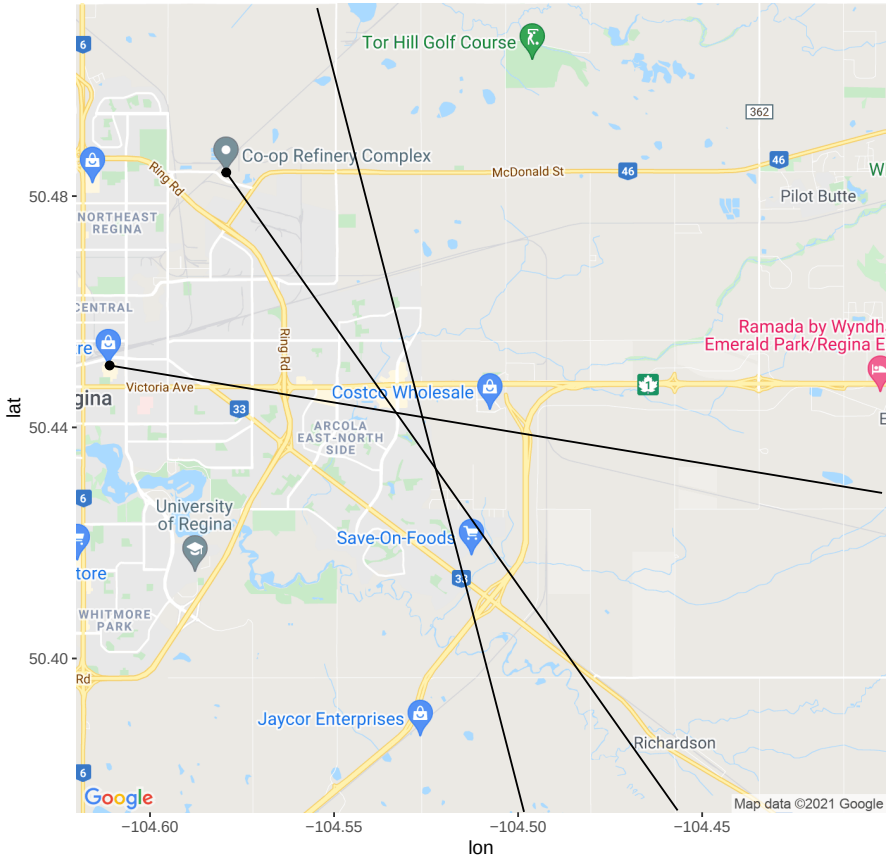


Figure 2: Zooming

So, your location is somewhere inside or around the triangle. Our aim of the project is to calculate the posterior of your location using known information about reference points and priors. In order to complete this project, you may need below R software packages: mvtnorm, coda, tmvtnorm, ggmap, osmar, geosphere, tidyverse,osmdata, showtext, rvest and sf.

2. Methods and Analysis

2.1 Assumptions

For given pair of coordinates $P_1 = (\lambda_1, \phi_1)$ and $P_2 = (\lambda_2, \phi_2)$, the bearing angle from point P1 to point P2 is computed by following formula:

$$B(P_1, P_2) = (360 + arctan(\frac{\lambda_2 - \lambda_1}{\phi_2 - \phi_1}).\frac{180}{\pi}) \ mod \ 360$$

This formula assumes that the Earth is flat, and this gives almost correct values when we work on marginal area.The errors of measured bearings and calculated bearings follow normal distribution with zero-mean and variance of σ^2. We suppose that that the measurements of three bearings α, β, γ were made independently. Therefore, the likelihood function can be construct as below:

$$p(\alpha, \beta, \gamma|\lambda, \phi, \sigma) = N(\alpha|B((\lambda, \phi), (\lambda_1, \phi_1)) * N(\alpha|B((\lambda, \phi), (\lambda_2, \phi_2)) * N(\alpha|B((\lambda, \phi), (\lambda_3, \phi_3))$$

Where $(\lambda_1, \phi_1) = (-104.579288, 50.484095)$ is the longitude and latitude for Point 1 $(\lambda_2, \phi_2) = (-104.556831, 50.5180655)$ is the longitude and latitude for Point 2 $(\lambda_3, \phi_3) = (-104.610967, 50.450717)$ is the longitude and latitude for Point 3 (λ, ϕ) is unknown longitude and latitude of your position

The final assumptions are for prior distributions of λ, ϕ, σ. Suppose, they have following distribution: $\lambda \sim$ Uniform(-105,-104) $\phi \sim$ Uniform(50,51) $\sigma \sim$ Exponential(20)

2.2 Sampling

2.2.1 Stage 1-Uniform prior

Given measured bearings, we need to calculate posteriors of your location as well as variance of bearing error, they are noted as a vector of three parameter params=(λ, ϕ, σ). There are various methods of sampling such as fine grid , Sequential Importance Sampling, Metropolis-Hasting, Gibbs sampler(special case of Metropolis-Hastings). In this section, we will go with Metropolis-Hastings and use the Gelman-Rubin PSRF statistic to monitor the convergence of the simulation.

From the assumptions mentioned in previous sections, we implement loglikelihood and logprior functions in R in order to calculate posteriors. Note that we use logarithm of density function instead of direct use. Since the target distribution is unknown and very complicated that we can not express in a closed form, we should use special sampling method. We use Metropolis-Hasting algorithm for this purpose since it guarantees the convergence of chains.

We will simulate four chains simultaneously and monitor the convergence process, the final samples are stored in a R file "stage1_final.RData" for reproduction purpose and to avoid time consuming when generating report. Good proposal distribution is key to make our chains converge rapidly. After many experiments, We choose a multivariate normal distribution with covariance matrix COV.

Table 1: Stage 1-Covariance Matrix

4e-06	0e+00	0.000
0e+00	4e-06	0.000
0e+00	0e+00	0.002

It is noticed that our third parameter σ must be greater than zero, so We need to make some correction for our acceptance rate. We will use truncated multivariate normal distribution dtmvnorm from R package tmvtnorm as correction component. Some noticeable parameter of this function as below:

dtmvnorm(x, mean = rep(0,nrow(sigma)), sigma = diag(length(mean)), lower=rep(-Inf, length = length(mean)), upper=rep(Inf, length = length(mean)))

x: Vector or matrix of quantiles

mean: Mean vector

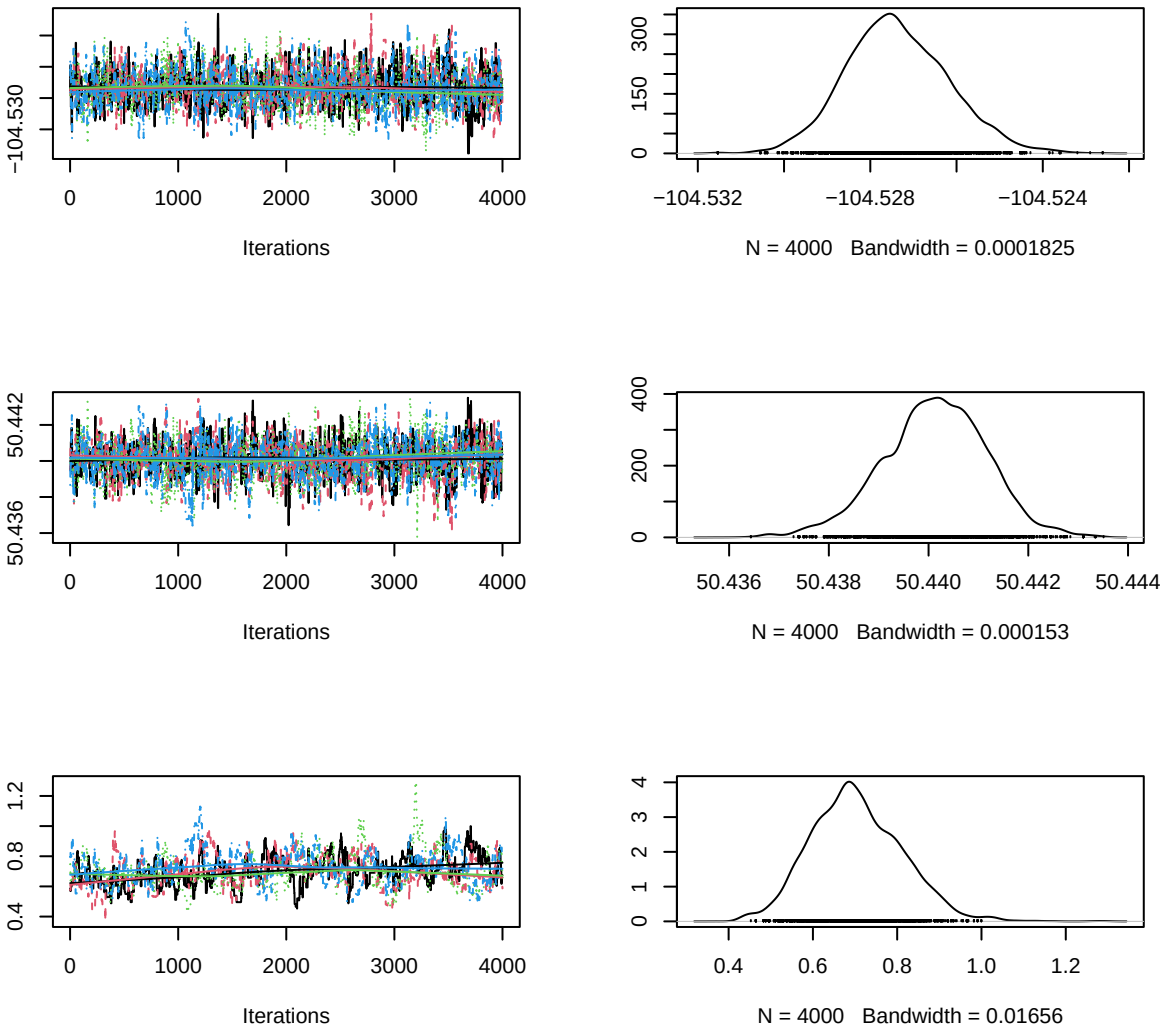
sigma: Covariance matrix

lower: Vector of lower truncation points, default is rep(-Inf, length = length(mean)).

upper: Vector of upper truncation points, default is rep(Inf, length = length(mean)).

After few iterations, we get converged chains for our location. Statistical factors such as potential scale reduction factor (PSRF) (less than 1.1) and traceplot show a truely stationary distribution.It is noted that two figures in the first row present the line and density chart of λ , the second row connected to ϕ and the third row connected to σ

```
## Upper Limits of PSRF: 1.033387 1.034269 1.036991
```



2.2.3 Prior-Distance to Nearest Road

Difference with previous part, in this stage our priors depend on the distance from our location to the nearest road. Here is the prior:

$$\rho(\lambda, \phi) \propto N(\rho(\lambda, \phi)|0, 6^2)$$

$$\sigma \sim Exponential(20)$$

Where $\rho(\lambda, \phi)$ is the shortest distance to the middle of the nearest road from the Open-StreetMap database. The R software packages osmdata, rvest, sf and geosphere support us to calculate this distance. That

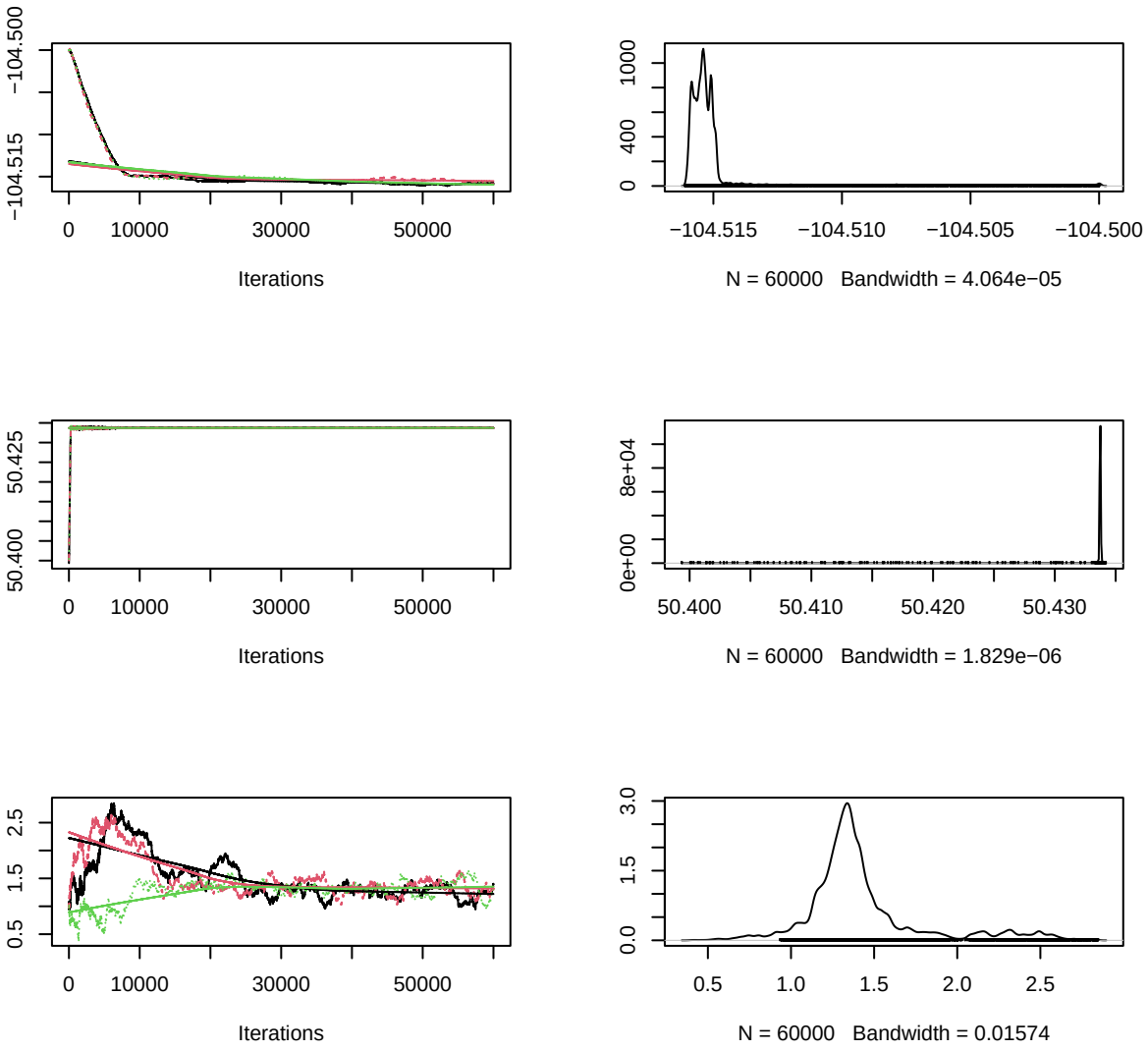
include several steps: identify intersection bearings of two bearings, then load all roads within 50 meters of the intersection of the first and the second bearing lines then calculate the distance.

The sampling process is similar to the previous part but this case is more complicated and may need very long run. So, We will use visualization of the chains instead of using Gelman-Rubin PSRF statistic to monitor the convergence. We will perform simulation with three chains, each has 80000 samples. The proposal is a multi-normal distribution with covariance matrix as:

Table 2: Stage 2

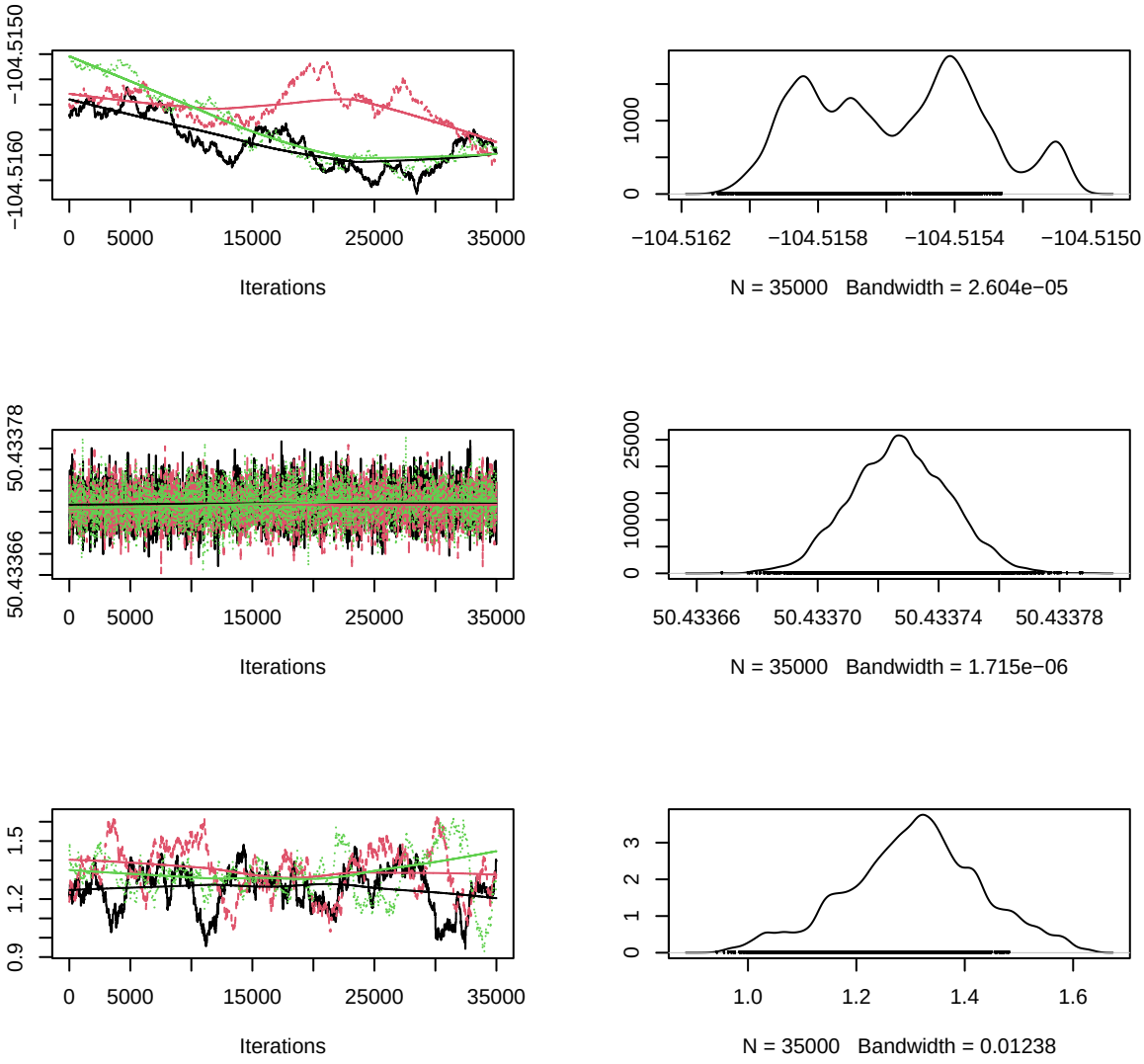
1e-10	0e+00	0e+00
0e+00	1e-07	0e+00
0e+00	0e+00	3e-04

The sample is saved into R as final_S2.RData for reproduction and analysis. We can visualize the chains as below:



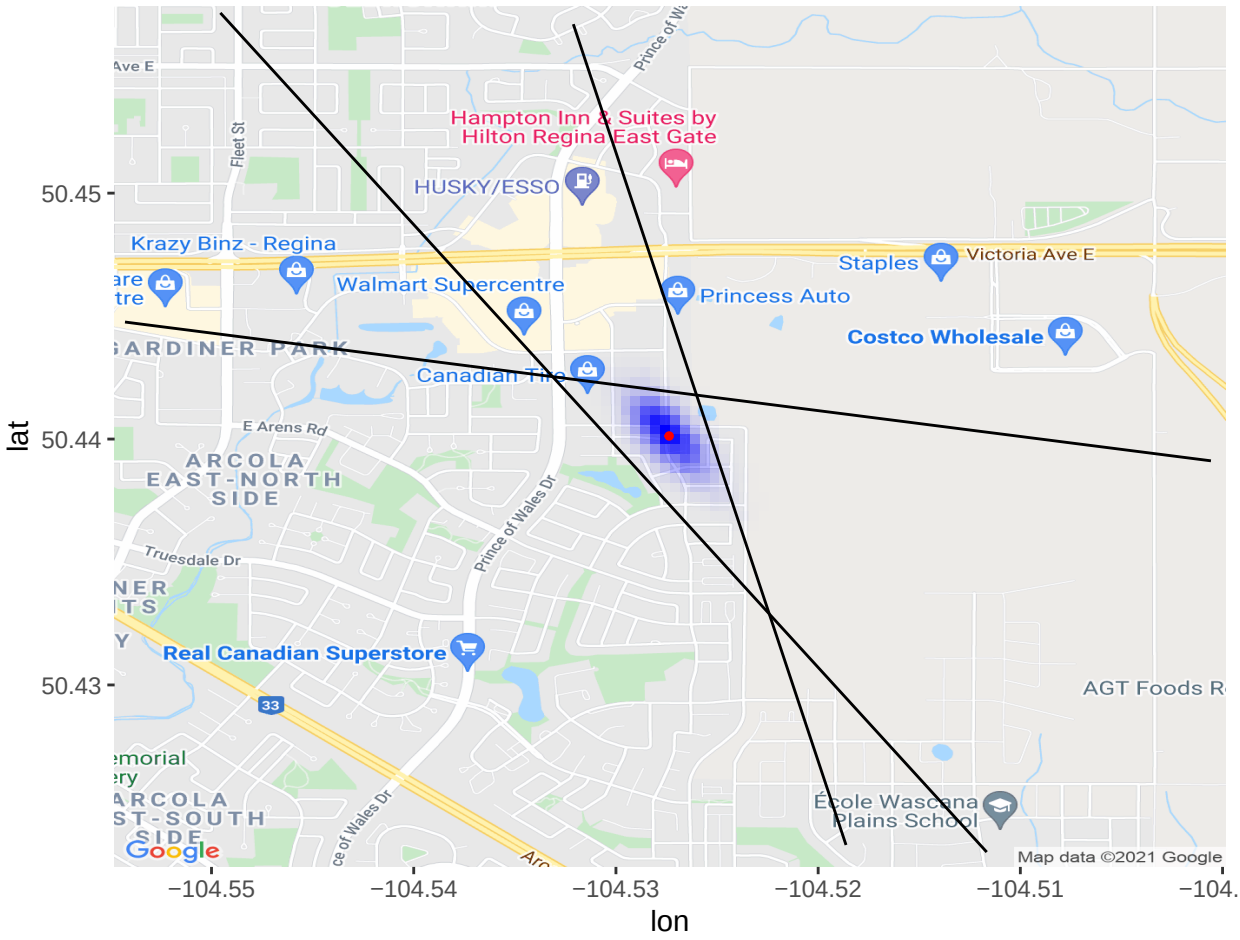
It is obvious that the first 25000 samples are “burn in” period, we can ignore this period and get final samples

for us. We can visualize the final sample again and see that the chain of ϕ converge perfectly, while chains of λ, σ can be weakly stationary. We may need longer chains to see the clear convergence.

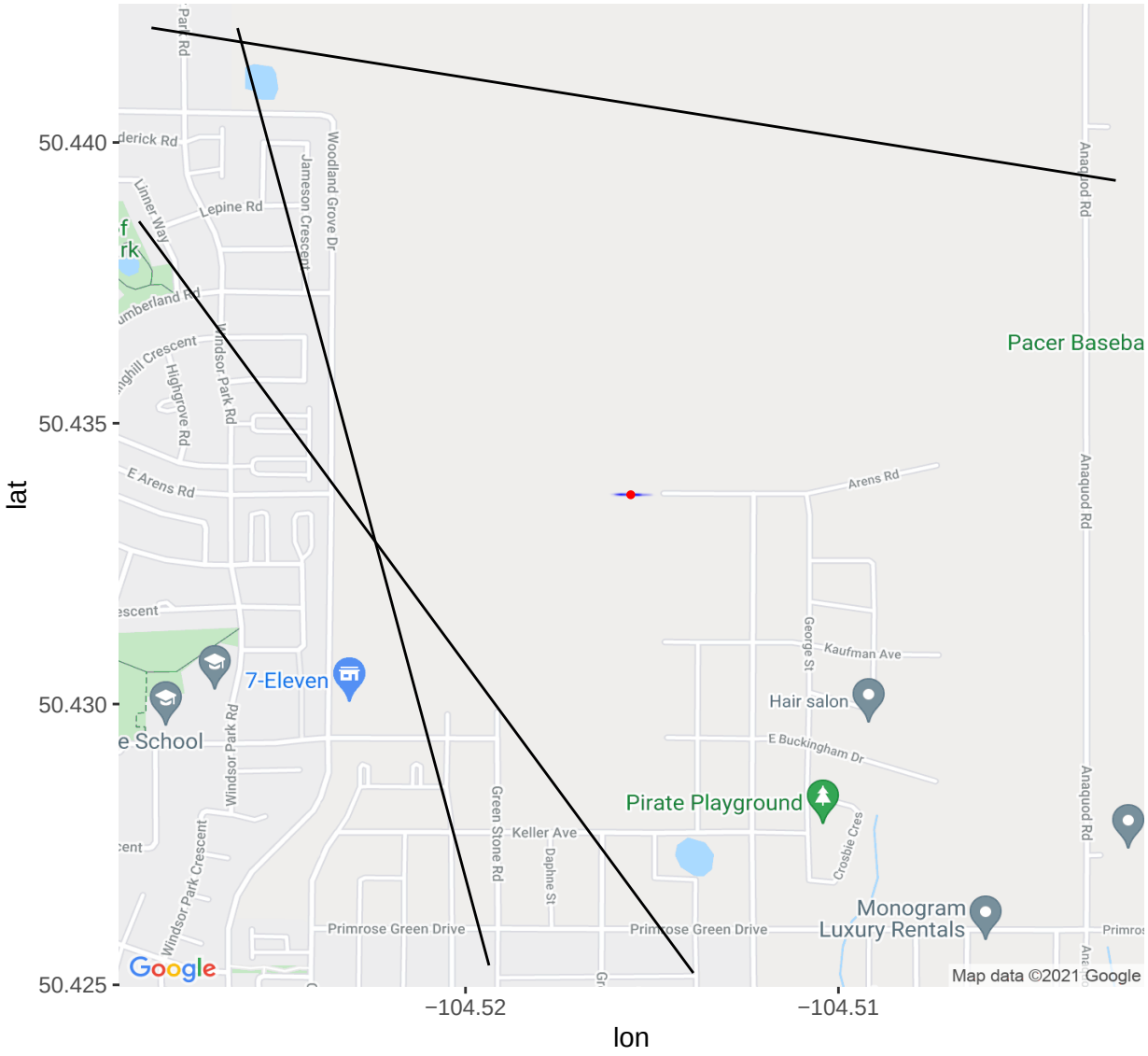


3. Results

In the first stage, we suppose the priors of our location follow uniform distribution and then applied Metropolis-Hasting algorithms we have samples for λ and ϕ as well the error variance σ. The trace plots show that our chains converge after some iterations, the density map give us more concrete confirmation on that as detected area is entirely inside the triangle and very close to our expected location. It proves that our algorithms work well.



In the stage 2, it is assume that you are on the road. Although the traceplot do not show a perfect stationary chains but the density map indicates well your position on the road that which near our location. It is noted that as our sample for this case is large, so using full sample to plot density may cause your computer crash, so I just take a subsets of 8000 samples to present here only.



4. Conclusion

In this project we see that our target distributions are very complicated that we can not solve or express them in some closed forms. And Metropolis-Hastings Sampler proved to be useful in such cases. In the first stage, samples generated from this algorithm converged quickly. This is because we have a good proposal to start with. During tuning parameters in this stage, we learn that the variance of proposal distribution will move in opposite direction with acceptance rate. When acceptance rate is too high we may need to rise the variance to lower values to decrease the acceptance rate. Some papers mentioned that the acceptance rate is from 10% to 60% is fine and I try to control it between 20% to 50%.

In the second stage, our priors differ significantly with stage 1 and we assume that you are on the road then the chains run much longer to converge. The results shows that predicted locations are perfectly fit to the road near our expected locations. When perform this simulation, I found that sometimes low value of PSFR (less than 1.2) do not guarantee the convergence, we must visualize to double check the results. And in some cases, changing different parameters, proposal do not work, we need to extend our chain for looking to the convergence.