Bayesian Statistical Modeling: Analytical Methods / Bayes Monte Carlo / Grid Methods

Eason Cai

2025-02-14

1. Introduction

In this report, we are addressing an safety concern an emergency landing system. In an emergency, the airplane system must quickly estimate the usable fuel remaining based on sensor readings and known consumption rates.

Our goal is to use Bayesian methods to update our belief about the fuel level and then propagate this uncertainty to the available flight time. Finally, we address the likelihood of meeting operational requirements and the risk of running out of fuel.

Here are the given information and assumptions:

- **Fuel tank capacity:** 182 liters.
- **Fuel sensor reading:** 34 liters.
- **Fuel sensor error:** $\mathcal{N}(0, 20^2)$.
- **Fuel consumption rate:** $\mathcal{N}(18, 2^2)$.
- The fuel level and consumption are assumed **uncorrelated**.

---

2. The Naive Estimate Based only on the Fuel Gauge Reading

Define F to be the true usable fuel in liters, then without any additional information, the likelihood function behaves as such

$$\mathbb{P}(F \mid \text{data}) \propto \mathcal{N}(34, 20^2).$$

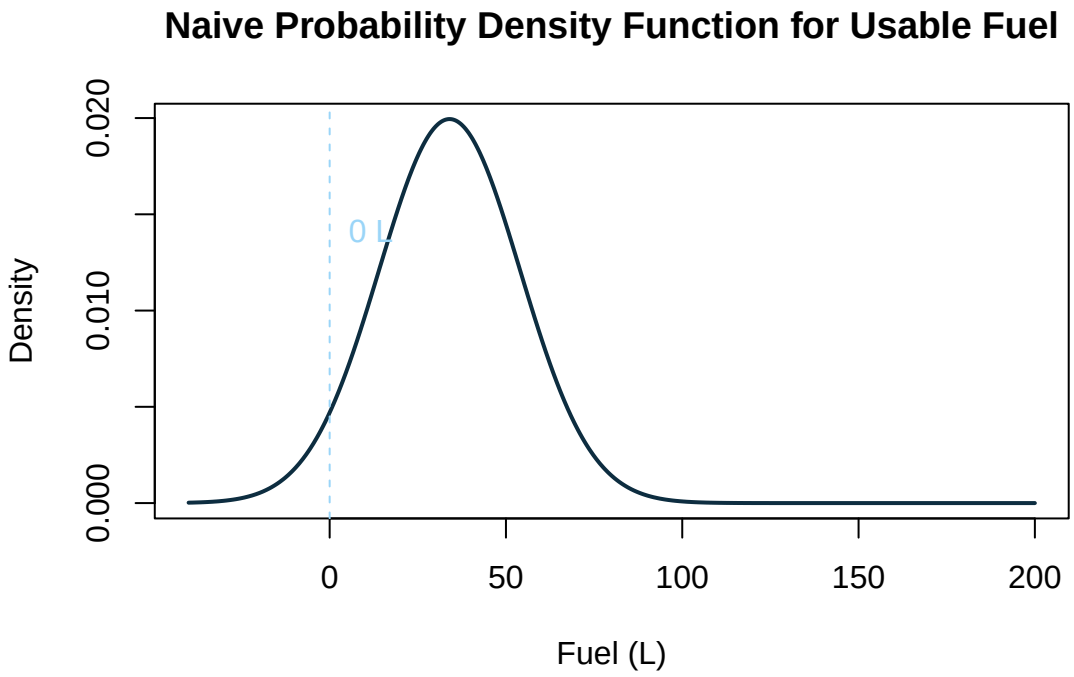
This gives the following information:

- **Expected value:** 34 liters.
- **Most likely value (mode):** 34 liters.
- **Probability of negative fuel:**

$$\mathbb{P}(F < 0) = \Phi\left(\frac{0 - 34}{20}\right) = \Phi(-1.7) \approx 4.46\%.$$

Negative fuel is impossible, hence, assigning a positive probability to negative fuel is unrealistic. We can further visualize this in the plot.

```r
# Set possible axis value to visualize unrealistic estimate
fuel <- seq(-40, 200, length.out = 1000)
likelihood <- dnorm(fuel, mean = 34, sd = 20)

plot(fuel, likelihood, type = "l", lwd = 2, col = "#0D2D40",
     xlab = "Fuel (L)", ylab = "Density",
     main = "Naive Probability Density Function for Usable Fuel")
abline(v = 0, col = "#9AD5F8", lty = 2)
text(0, max(likelihood)*0.7, "0 L", pos = 4, col = "#9AD5F8")
```



---

3. Introducing a Physically Based Prior

We want to make sure our PDF does not take in account of negative fuel estimate and not exceed the tank capacity. Then, we can choose a uniform prior over the physically admissible range:

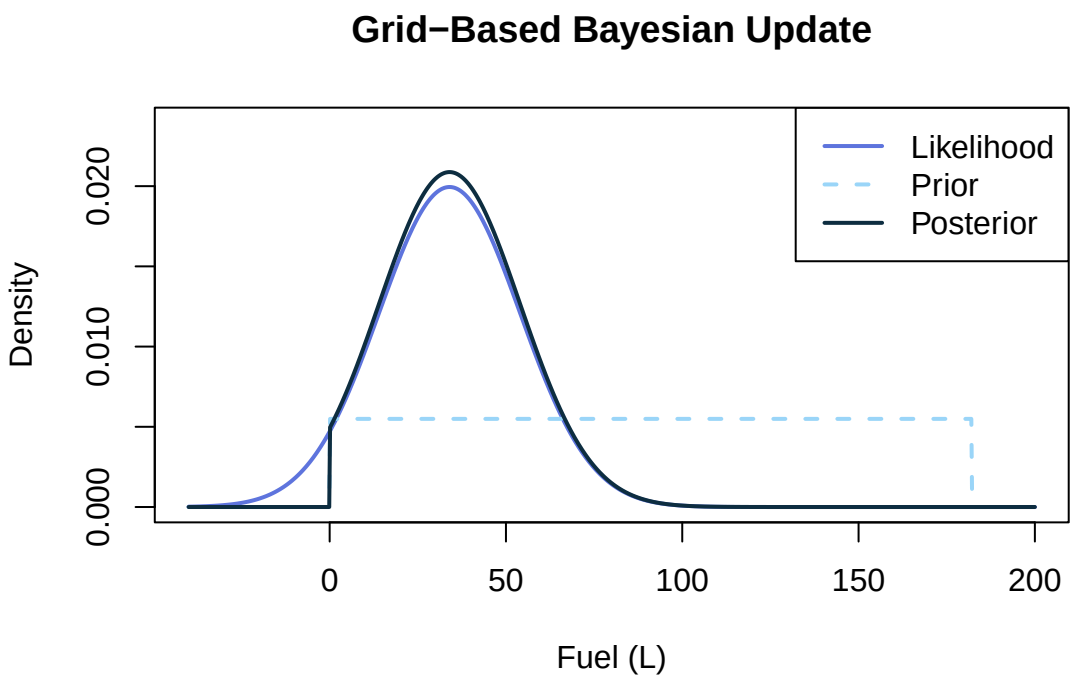
$$\mathbb{P}(F) = \begin{cases} \frac{1}{182} & \text{if } 0 \leq F \leq 182, \\ 0 & \text{otherwise.} \end{cases}$$

This is our subjective prior based on the airplane system's physical limits.

---

4. Bayesian Update using Grid-based Method

```r
# Define prior from section 3
prior <- ifelse(fuel >= 0 & fuel <= 182, 1/182, 0)

# Unnormalized posterior
unnorm_post <- likelihood * prior

# Define a normalization function to normalize the posterior
normalization <- function(x, y) {
  sum((diff(x) * (head(y, -1) + tail(y, -1))) / 2)
}
posterior <- unnorm_post / normalization(fuel, unnorm_post)

# Plot Bayesian Update using Grid-based Method
plot(fuel, likelihood, type = "l", lwd = 2, col = "#5E74DD",
     xlab = "Fuel (L)", ylab = "Density", ylim = c(0, max(likelihood)*1.2),
     main = "Grid-Based Bayesian Update")
lines(fuel, prior, col = "#9AD5F8", lwd = 2, lty = 2)
lines(fuel, posterior, col = "#0D2D40", lwd = 2)
legend("topright", legend = c("Likelihood", "Prior", "Posterior"),
       col = c("#5E74DD", "#9AD5F8", "#0D2D40"), lty = c(1,2,1), lwd = 2)
```



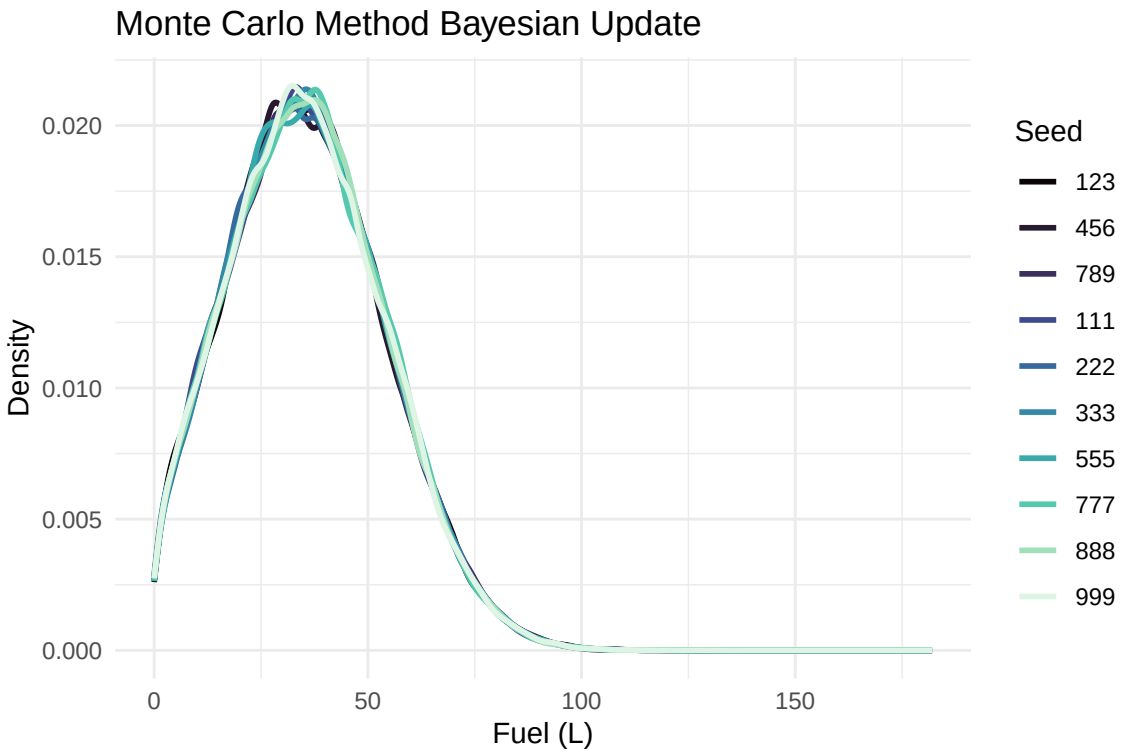

```r
# Find the probability of negative fuel again
p_neg <- sum(posterior[fuel < 0]) * (fuel[2] - fuel[1])
p_neg
```

```
## [1] 0
```

With our subjective prior, the posterior assigns zero probability to negative fuel, hence, fix the issue we had in section 2 by using Grid-based Bayesian update.

5. Bayesian Update using Monte Carlo Method

```r
library(ggplot2)

n <- 100000 # Number of runs for MC
seeds <- c(123, 456, 789, 111, 222, 333, 555, 777, 888, 999)

# Initialize data frame for density estimates for each seed
density_df <- data.frame()
for (s in seeds) {
  # Sample from the uniform prior between 0 and 182 liters
  set.seed(s)
  prior_sample <- runif(n, min = 0, max = 182)

  # Compute the likelihood weights for each sample
  w <- dnorm(prior_sample, mean = 34, sd = 20)
  w <- w / sum(w)

  # Resample according to the likelihood weights to obtain posterior samples
  posterior_sample <- sample(prior_sample, size = n, replace = TRUE, prob = w)

  # Compute the density estimate of the posterior samples and add to dataframe
  d <- density(posterior_sample, from = 0, to = 182)
  temp <- data.frame(fuel = d$x, density = d$y, seed = factor(s))
  density_df <- rbind(density_df, temp)
}

# Plot Bayesian Update using Monte Carlo Method for 10 different seeds
ggplot(density_df, aes(x = fuel, y = density, color = seed)) +
  geom_line(size = 1) +
  labs(title = "Monte Carlo Method Bayesian Update",
       x = "Fuel (L)",
       y = "Density",
       color = "Seed") +
  scale_color_viridis_d(option = "mako") +
  theme_minimal()
```



Similarly, the posterior assigns zero probability to negative fuel by sampling our subjective prior. Therefore, the Monte Carlo Bayesian update confirms the Grid-based Bayesian update.

---

6. Simple Analytical Kalman Filter Assumption

Assumptions are listed as the following:

1. Fuel level must be linear with its measurements and fuel sensor error and variations in fuel consumption are Gaussian.
2. The state space is assumed to be unbounded i.e. unconstrained.
3. Known uncertainties in the airplane system and constant over time.

Point by point we will check if these assumptions are realistic or not:

1. While the sensor reading may be approximately linear with Gaussian errors, the true fuel level is bounded.
2. Kalman filter could allow fuel to be negative or past the tank capacity, which is impossible.
3. We are concerning with airplane system that is critical in real life, hence uncertainties will be more complex than constant over time.

Thus, although a Kalman filter is oftenly used for its analytical simplicity, its assumptions are not realistic for our concern with the airplane fuel system.

---

7. Estimated Available Flight Time Visualization

In this section, we assume that fuel consumption is normally distributed and independent of the fuel level. While in practical scenarios, we need to take in consideration of non-linear effects or correlations.

With that being said, let's define T be the available flight time in hours such that

$$T = \frac{F}{c},$$

where F is the usable fuel and c is the fuel consumption rate:

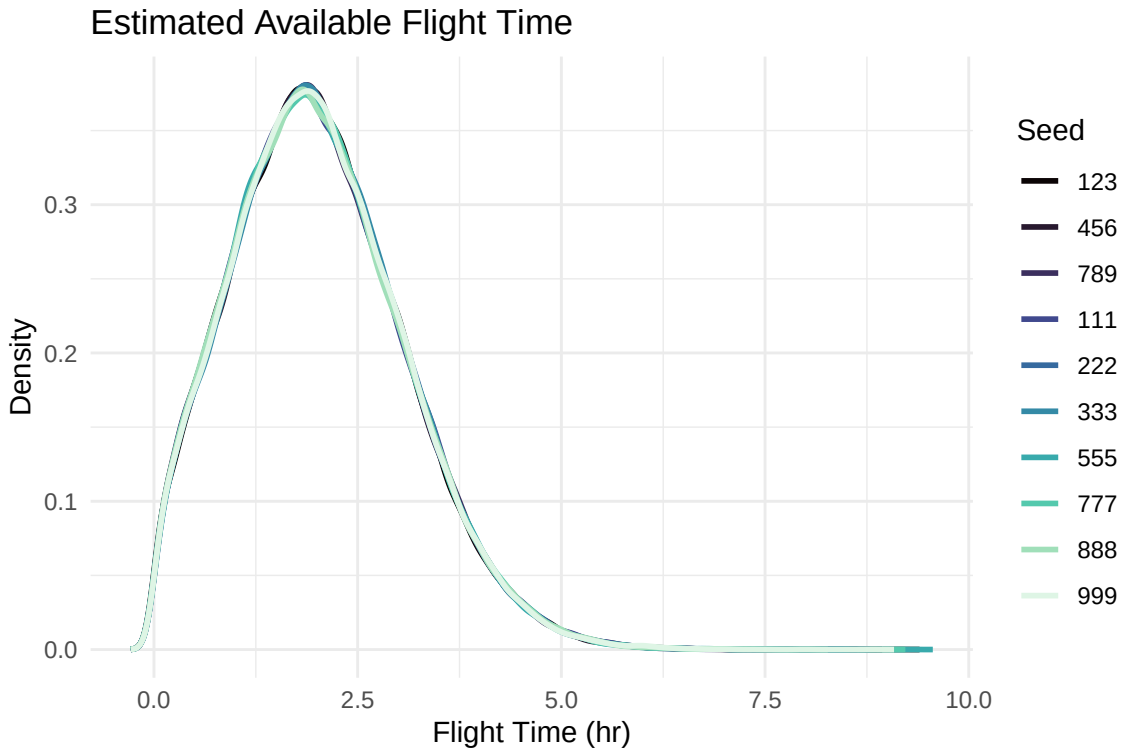
- F follows the posterior distribution.
- $c \sim \mathcal{N}(18, 2^2)$.

Given that with 100 minutes flight time away from an airport with at least 30 min reserve fuel required by regulations, we want to find:

- The probability of making it to the airport.
- The probability of running out of fuel.

```
# Initialize data frame to store probability results for each seed
results <- data.frame(seed = seeds, p_yes = NA, p_no = NA)

dens_list <- list()
i <- 1
for (s in seeds) {
  set.seed(s)

  # Resample fuel from the posterior
  fuel_MC <- sample(posterior_sample, size = n, replace = TRUE)
  consumption <- rnorm(n, mean = 18, sd = 2)
  # Ensure consumption is positive
  consumption[consumption <= 0] <- 0

  # Compute flight time in hours
  time <- fuel_MC / consumption
  dens_list[[i]] <- data.frame(time = density(time)$x, density = density(time)$y,
                               seed = as.factor(s))

  # Compute the two probabilities
  # (note that by regulation requirement, these 2 probabilities won't add up to 1)
  p_yes <- mean(time >= (100/60) + 0.5)
  p_no <- mean(time < (100/60))
  results$p_yes[[i]] <- p_yes
  results$p_no[[i]] <- p_no
  i <- i+1
}

# Combine density estimates for plotting
densities <- do.call(rbind, dens_list)
ggplot(densities, aes(x = time, y = density, color = seed)) +
  geom_line(size = 1) +
  labs(title = "Estimated Available Flight Time",
```

```
       x = "Flight Time (hr)",
       y = "Density",
       color = "Seed") +
  scale_color_viridis_d(option = "mako") +
  theme_minimal()
```




```
results_summary <- rbind(results, data.frame(seed = "Mean",
                                             p_yes = mean(results$p_yes),
                                             p_no = mean(results$p_no)))
colnames(results_summary) <- c("Seed", "Probability_of_Making_It",
                               "Probability_of_Runing_Out_Fuel")
kable(results_summary, digits = 6)
```

Seed	Probability_of_Making_It	Probability_of_Runing_Out_Fuel
123	0.419150	0.394300
456	0.418240	0.394550
789	0.417360	0.396000
111	0.420850	0.394940
222	0.420120	0.394500
333	0.421700	0.390390
555	0.417900	0.396810
777	0.420540	0.395290
888	0.418800	0.396460
999	0.419580	0.393600
Mean	0.419424	0.394684

8. Summary

1. **Choice of Prior**
   - A uniform prior is the simplest noninformative choice given known bounds
   - Alternatively, if we had more prior knowledge regarding the fuel consumption, we could choose a Beta distribution with $\alpha = 0, \beta = 182$.
2. **Uncertainties**
   - To simulate real world scenario, we used 10 different seed to observe the convergence when needed sampling.
   - Since our analysis provides a first-order approximation, there are some neglected deep uncertainties:
     - Time-varying sensor errors.
     - Possible correlations between the fuel sensor reading and the fuel consumption rate.
     - If consumption depends on flight conditions, fuel consumption would be non-Gaussian.
3. **Reproducibility**
   - All code and random seeds are included in this document.
   - By setting `seed`, anyone can reproduce the same numeric results.
   - This R Markdown file can be knitted to produce the same figures and estimates.
4. **Citations**
   - Qian, S. S., Stow, C. A., & Borsuk, M. E. (2003). On Monte Carlo methods for Bayesian inference. Ecological Modelling, 159(2-3), 269–277.
   - D'Agostini, G. (2003). Bayesian reasoning in data analysis: A critical introduction. Singapore: World Scientific Publishing. (Chapter 6)
   - Ruckert, K. L., Guan, Y., Bakker, A. M. R., Forest, C. E., & Keller, K. (2017). The effects of time-varying observation errors on semi-empirical sea-level projections. Climatic Change, 140(3-4), 349–360. https://doi.org/10.1007/s10584-016-1858-z
5. **License**
   - GNU general public license v3.0.
6. **Copyright**
   -

Appendix: Code

You can also access the repository using the following URL: https://github.com/easoncai999/bayesian-ps3

```
### PART 2 ###
fuel <- seq(-40, 200, length.out = 1000)
likelihood <- dnorm(fuel, mean = 34, sd = 20)

plot(fuel, likelihood, type = "l", lwd = 2, col = "#0D2D40",
     xlab = "Fuel (L)", ylab = "Density",
     main = "Naive Probability Density Function for Usable Fuel")
abline(v = 0, col = "#9AD5F8", lty = 2)
text(0, max(likelihood)*0.7, "0 L", pos = 4, col = "#9AD5F8")

### PART 4 ###
prior <- ifelse(fuel >= 0 & fuel <= 182, 1/182, 0)
unnorm_post <- likelihood * prior
normalization <- function(x, y) {
  sum((diff(x) * (head(y, -1) + tail(y, -1))) / 2)
}
posterior <- unnorm_post / normalization(fuel, unnorm_post)

plot(fuel, likelihood, type = "l", lwd = 2, col = "#5E74DD",
     xlab = "Fuel (L)", ylab = "Density", ylim = c(0, max(likelihood)*1.2),
     main = "Grid-Based Bayesian Update")
lines(fuel, prior, col = "#9AD5F8", lwd = 2, lty = 2)
lines(fuel, posterior, col = "#0D2D40", lwd = 2)
legend("topright", legend = c("Likelihood", "Prior", "Posterior"),
       col = c("#5E74DD", "#9AD5F8", "#0D2D40"), lty = c(1,2,1), lwd = 2)

p_neg <- sum(posterior[fuel < 0]) * (fuel[2] - fuel[1])
p_neg

### PART 5 ###
library(ggplot2)
n <- 100000
seeds <- c(123, 456, 789, 111, 222, 333, 555, 777, 888, 999)

density_df <- data.frame()
for (s in seeds) {
  set.seed(s)
  prior_sample <- runif(n, min = 0, max = 182)

  w <- dnorm(prior_sample, mean = 34, sd = 20)
  w <- w / sum(w)
  posterior_sample <- sample(prior_sample, size = n, replace = TRUE, prob = w)

  d <- density(posterior_sample, from = 0, to = 182)
  temp <- data.frame(fuel = d$x, density = d$y, seed = factor(s))
  density_df <- rbind(density_df, temp)
}

ggplot(density_df, aes(x = fuel, y = density, color = seed)) +
  geom_line(size = 1) +
```

```
  labs(title = "Monte Carlo Method Bayesian Update",
       x = "Fuel (L)",
       y = "Density",
       color = "Seed") +
  scale_color_viridis_d(option = "mako") +
  theme_minimal()

### PART 7 ###
results <- data.frame(seed = seeds, p_yes = NA, p_no = NA)
dens_list <- list()
i <- 1
for (s in seeds) {
  set.seed(s)

  fuel_MC <- sample(posterior_sample, size = n, replace = TRUE)
  consumption <- rnorm(n, mean = 18, sd = 2)
  consumption[consumption <= 0] <- 0

  time <- fuel_MC / consumption
  dens_list[[i]] <- data.frame(time = density(time)$x, density = density(time)$y,
                               seed = as.factor(s))

  p_yes <- mean(time >= (100/60) + 0.5)
  p_no <- mean(time < (100/60))
  results$p_yes[[i]] <- p_yes
  results$p_no[[i]] <- p_no
  i <- i+1
}

densities <- do.call(rbind, dens_list)
ggplot(densities, aes(x = time, y = density, color = seed)) +
  geom_line(size = 1) +
  labs(title = "Estimated Available Flight Time",
       x = "Flight Time (hr)",
       y = "Density",
       color = "Seed") +
  scale_color_viridis_d(option = "mako") +
  theme_minimal()

results_summary <- rbind(results, data.frame(seed = "Mean",
                                             p_yes = mean(results$p_yes),
                                             p_no = mean(results$p_no)))
colnames(results_summary) <- c("Seed", "Probability_of_Making_It",
                               "Probability_of_Runing_Out_Fuel")
kable(results_summary, digits = 6)
```