Linear regression for Mauna Loa CO2 data

Mauna Loa CO2 data

This is an example of linear regression and we will analyse the Mauna Loa CO2 data[1]. The data contains monthly concentrations adjusted to represent the 15th day of each month. Units are parts per million by volume (ppmv) expressed in the 2003A SIO manometric mole fraction scale. The "annual average" is the arithmetic mean of the twelve monthly values where no monthly values are missing.

We want to construct and infer with Stan the following model: %

$$\begin{aligned} y_i &= \mu(x_i) + \epsilon_i \\ \epsilon_i &\sim N(0, \sigma^2) \\ \mu(x_i) &= a + bx_i \\ p(a) &= p(b) \propto 1 \\ \sigma^2 &\sim \text{Inv-Gamma}(0.001, 0.001) \end{aligned}$$

% where $y_i, i = 1, \cdots, n$ are the reported CO2 values, x_i is time, measured as months from the first observation, a is an intercept, b is the linear weight (slope) and σ^2 is the variance of the "error" terms, ϵ_i, around the linear mean function.

In practice, it is typically advisable to construct the model for standardized observations $\dot{y}_i = (y_i - \text{mean}(y))/\text{std}(y)$ where mean(y)) and std(y) are the sample mean and standard deviations of y_i values. Similar transformation should be done also for covariates x. You should then sample from the posterior of the parameters $(\dot{a}, \dot{b}, \dot{\sigma}^2)$ corresponding to the standardized data $\dot{y}_i$ and $\dot{x}_i$. After this you have to transform the samples of $\dot{a}, \dot{b}, \dot{\sigma}^2$ to the original scale.

Your tasks are the following:

1. Solve the equations to transform samples of $\dot{a}, \dot{b}, \dot{\sigma}^2$ to the original scale a, b, σ^2.
2. Sample from the posterior of the parameters of the above model using the Maunaloa CO2 data. (You can do this either with transformed or original data so if you didn't get step 1 right you can still proceed with this.) Check the convergence of model parameters and report the results of convergence tests. Visualize the marginal posterior distribution of the model parameters and report their posterior mean and 2.5% and 97.5% posterior quantiles.
3. Discuss how you would intepret the linear mean function $\mu(x)$ and how you would intepret the error terms ϵ_i.
4. Plot a figure where you visualize
    - The posterior mean and 95% central posterior interval of the mean function $\mu(x)$ as a function of months from January 1958 to December 2027.
    - The posterior mean and 95% central posterior interval of observations y_i as a function of months from January 1958 to December 2027. In case of historical years, consider the distribution of potential replicate observations that have not been measured but could have been measured.
    - plot also the measured observations to the same figure
5. Visualize

[1] http://cdiac.esd.ornl.gov/ftp/trends/co2/maunaloa.co2

  - the Posterior predictive distribution of the mean function, $\mu(x)$ in January 2025 and in January 1958 and the difference between these.
  - the Posterior predictive distribution of observations, y_i in January 2025 and in January 1958 and the difference between these.

- Discuss why the distributions of $\mu(x_i)$ and y_i differ

See the R-template for additional instructions.

Grading: This exercise is worth 20 points so that each step gives 4 points.

Answers

1. variable transformations

The original normal linear regression model looks like:

$$y_i = a + bx_i + \epsilon_i$$

Donate the means of x and y as $\overline{x}$ and $\overline{y}$; Donate the standard deviations of x and y as σ_x and σ_y;

We have the following about standardized observations and co-variance:

$$\dot{y} = \frac{y_i - \overline{y}}{\sigma_y}$$

$$\dot{x} = \frac{x_i - \overline{x}}{\sigma_x}$$

We can derive original parameters with following calculations;

$$\dot{y}\sigma_y + \overline{y} = a + b(\dot{x}\sigma_x + \overline{x}) + \epsilon_i$$
$$\dot{y} = \frac{a + b\overline{x} + b\sigma_x\dot{x} - \overline{y} + \epsilon_i}{\sigma_y}$$
$$\dot{y} = \frac{a + b\overline{x} - \overline{y}}{\sigma_y} + \frac{b\sigma_x}{\sigma_y}\dot{x} + \frac{\epsilon_i}{\sigma_y}$$

We also have

$$\dot{a} = \frac{a + b\overline{x} - \overline{y}}{\sigma_y}$$
$$a = \sigma_y\dot{a} - b\overline{x} + \overline{y}$$

$$\dot{b} = \frac{b\sigma_x}{\sigma_y}$$
$$b = \frac{\dot{b}\sigma_x}{\sigma_y}$$

Then we can get the equations to transform samples of $\dot{a}, \dot{b}, \dot{\sigma}^2$ to the original scale a, b, σ^2.

$\dot{a}$ to a

$$a = \sigma_y\left(\dot{a} - \dot{b}\frac{\overline{x}}{\sigma_x}\right) + \overline{y}$$

$\dot{b}$ to b

$$b = \frac{\sigma_y}{\sigma_x}\dot{b}$$

$\dot{\sigma}^2$ to σ^2

$$\dot{\sigma}^2 = \text{Variance}(\frac{\epsilon_i}{\sigma_y})$$

$$\dot{\sigma}^2 = \frac{\text{Variance}(\epsilon_i)}{\sigma_y^2}$$

And therefore

$$\sigma^2 = \sigma_y^2\dot{\sigma}^2$$

2. Build and analyze Stan model

Load the needed libraries.

```r
library(ggplot2)
library(StanHeaders)
library(rstan)
```

```
## rstan (Version 2.21.2, GitRev: 2e1f913d3ca3)
```

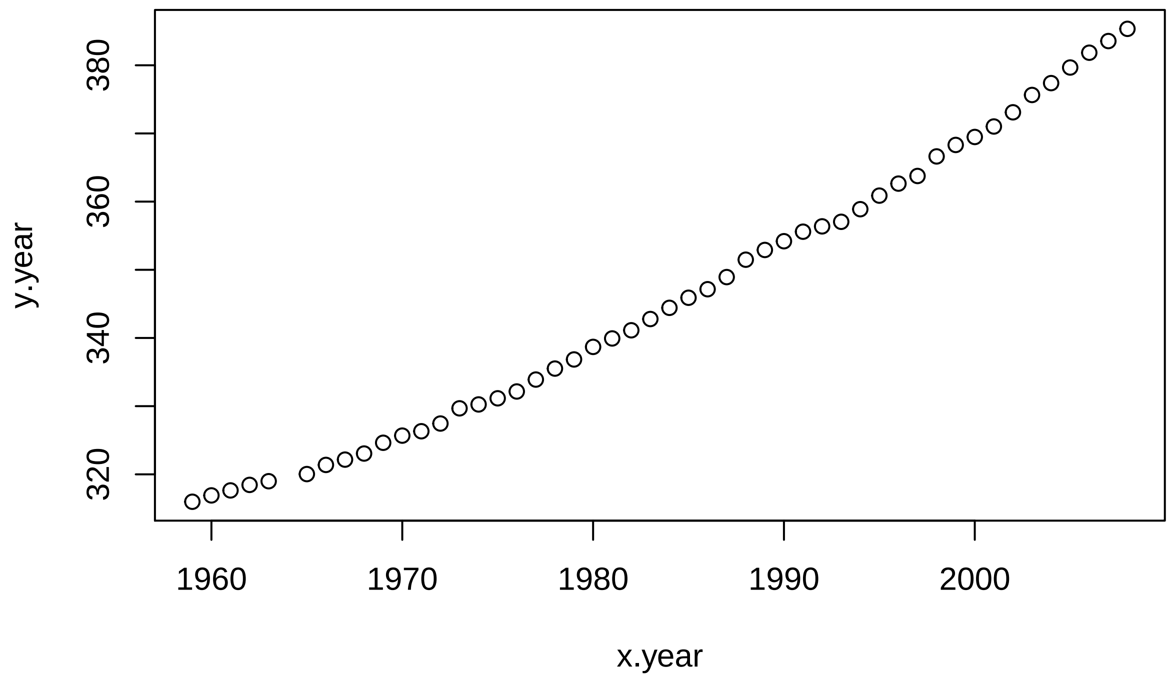
```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file
```

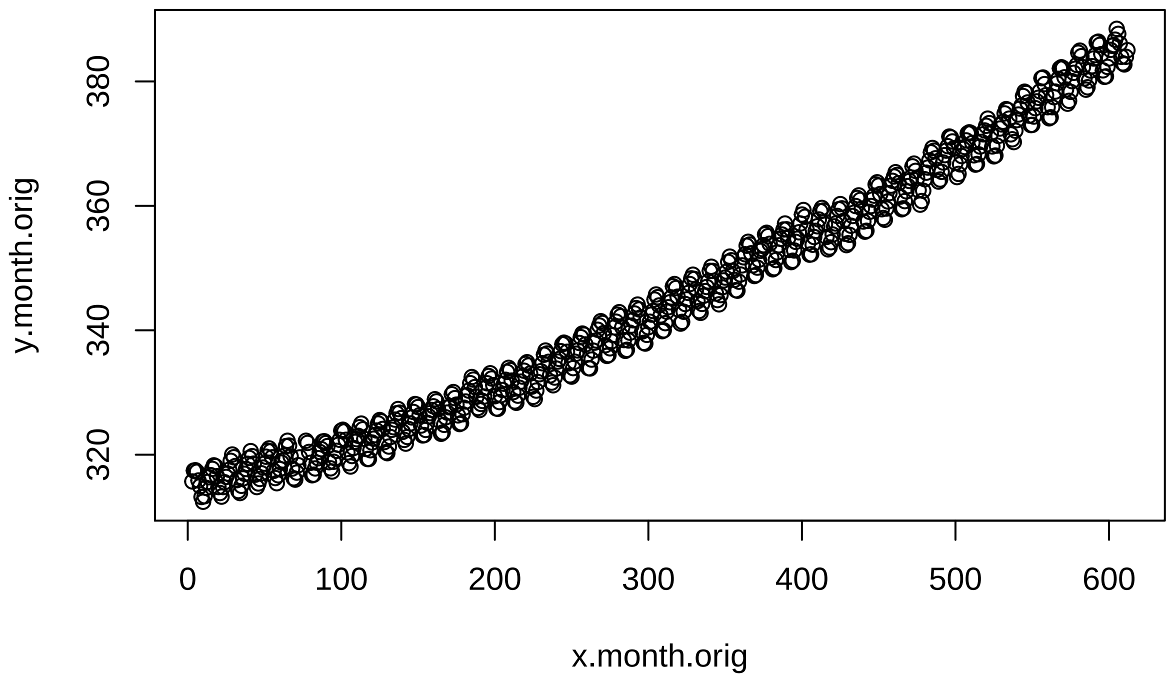
```r
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
```

Load the data and explore its properties

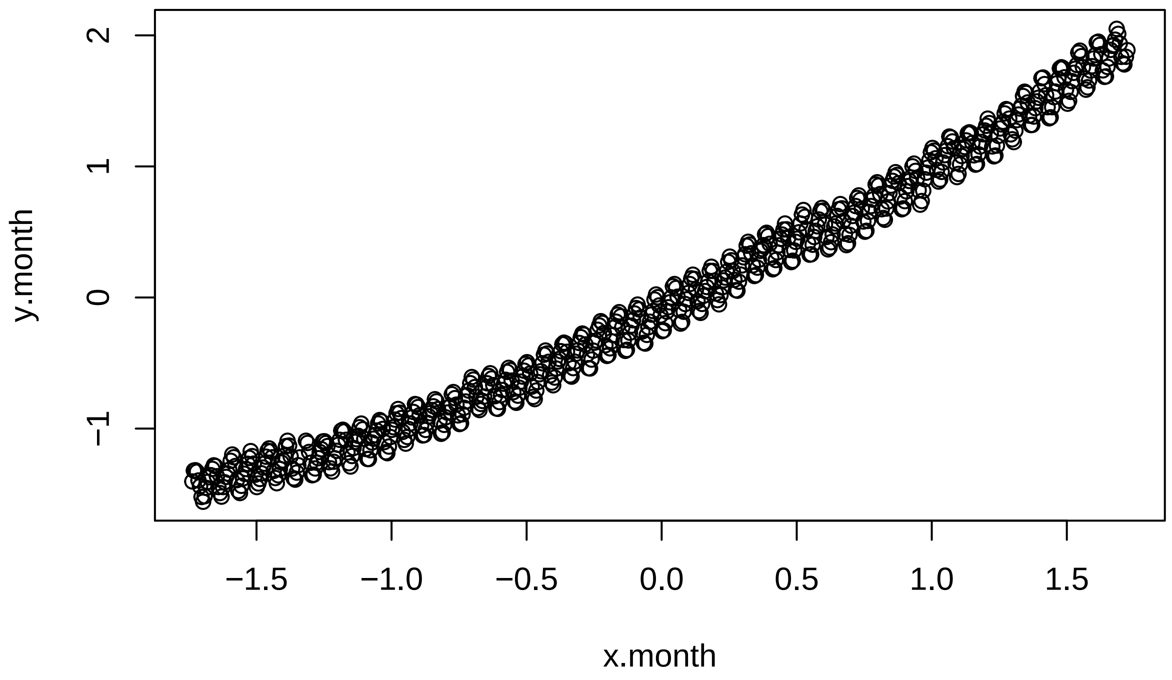
```r
# Load the data and explore it visually
maunaloa.dat = read.table("maunaloa_data.txt", header=FALSE, sep="\t")
# The columns are
# Year January February ... December Annual average

#  Notice! values -99.99 denote NA

# Let's take the yearly averages and plot them
x.year = as.vector(t(maunaloa.dat[,1]))
y.year = as.vector(t(maunaloa.dat[,14]))
# remove NA rows
x.year = x.year[y.year>0]
y.year = y.year[y.year>0]
plot(x.year,y.year)
```



```
# Let's take the monthy values and construct a "running month" vector
y.month.orig = as.vector(t(maunaloa.dat[,2:13]))
x.month.orig = as.vector(seq(1,length(y.month.orig),1))

# remove NA rows
x.month.orig = x.month.orig[y.month.orig>0]
y.month.orig = y.month.orig[y.month.orig>0]
plot(x.month.orig,y.month.orig)
```



```
# standardize y and x
my = mean(y.month.orig)             # mean of y values
stdy = sd(y.month.orig)             # std of y values
y.month = (y.month.orig-my)/stdy    # standardized y values

mx = mean(x.month.orig)             # mean of x values
stdx = sd(x.month.orig)             # std of x values
x.month = (x.month.orig-mx)/stdx    # standardized x values

plot(x.month,y.month)
```



Write the model description and set data into list

```
mauna_loa_c02_model = "
data{
  int<lower=0> n;
  real x[n];
  real y[n];
}
parameters{
  real a;
  real b;
  real<lower=0> sigma2;
}

model{
  sigma2 ~ inv_gamma(0.001, 0.001);
  for (i in 1:n){
    y[i] ~ normal(a+b*x[i], sqrt(sigma2));
  }
}"

# data list
data <- list ("n"=length(x.month), "x"=x.month, "y"=y.month)

#if you want to see what you get with the original data (not transformed)
# data <- list (N=length(x.month.orig), y=y.month.orig, x=x.month.orig)
# data
```

Now we will start the analysis. Define parameters and set initial values for them. We are going to sample four chains so we need four starting points. It is good practice to set them far apart from each others. We build linear regression model on data in order to get some reasonable initial values for our model parameters. Examine the convergence.

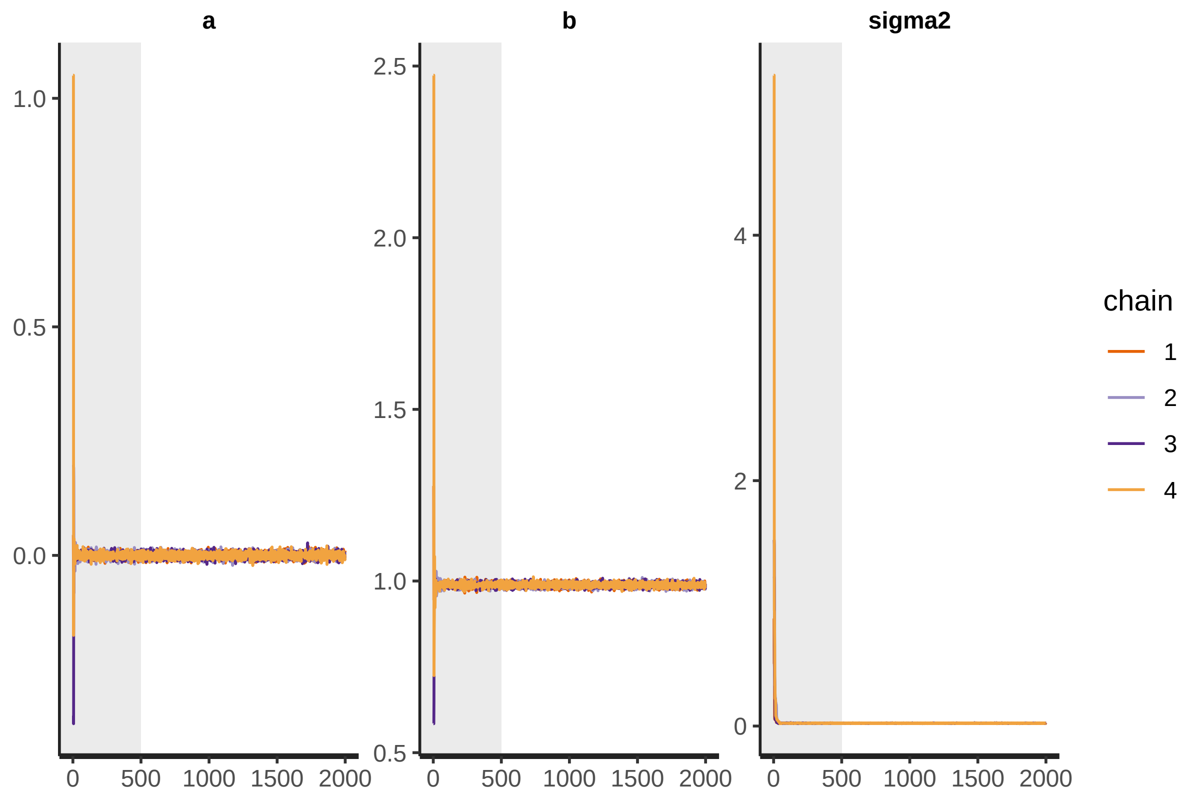
```
# Initial values
init1 <- list(a = 0, b = 0, sigma2 = 1)
init2 <- list(a = 0.5, b = 1, sigma2 = 2)
init3 <- list(a = 1, b = 2, sigma2 = 3)
init4 <- list(a = 1.5, b = 3, sigma2 = 4)
inits <- inits <- list(init1, init2, init3, init4)

## Run the Markov chain sampling with Stan:
post=stan(model_code=mauna_loa_c02_model,data=data,warmup=500,iter=2000,chains=4,thin=1,init=inits,cont

# Check for convergence, see PSRF (Rhat in Stan)
print(post,pars=c("a","b","sigma2"))
```

```
## Inference for Stan model: 4c46a2f8da9ecc8aafbedd6aa96fc659.
## 4 chains, each with iter=2000; warmup=500; thin=1;
## post-warmup draws per chain=1500, total post-warmup draws=6000.
##
##        mean se_mean   sd  2.5%  25%  50%  75% 97.5% n_eff Rhat
## a      0.00       0 0.01 -0.01 0.00 0.00 0.00  0.01  6247    1
## b      0.99       0 0.01  0.98 0.98 0.99 0.99  1.00  7150    1
## sigma2 0.02       0 0.00  0.02 0.02 0.02 0.02  0.03  3908    1
##
## Samples were drawn using NUTS(diag_e) at Tue Dec 01 13:49:34 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

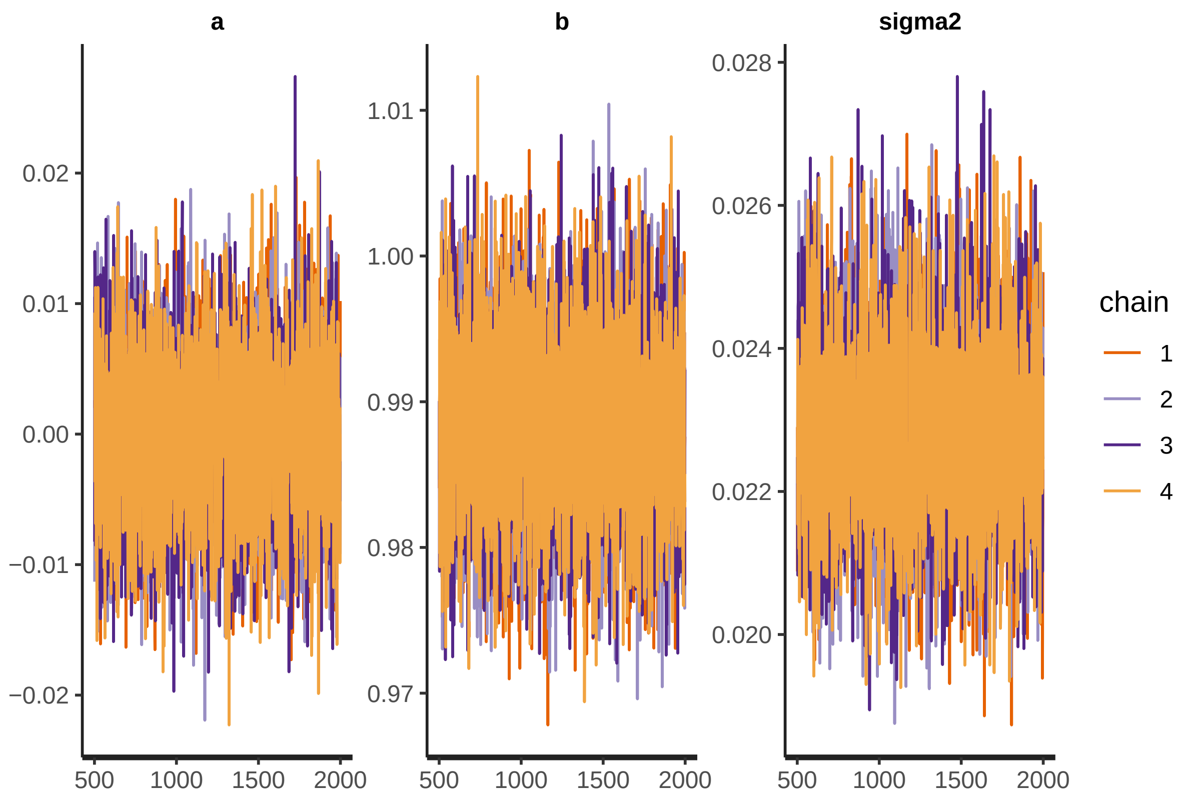
```
print(post)
```

```
## Inference for Stan model: 4c46a2f8da9ecc8aafbedd6aa96fc659.
## 4 chains, each with iter=2000; warmup=500; thin=1;
## post-warmup draws per chain=1500, total post-warmup draws=6000.
##
##          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## a        0.00    0.00 0.01  -0.01   0.00   0.00   0.00   0.01  6247    1
## b        0.99    0.00 0.01   0.98   0.98   0.99   0.99   1.00  7150    1
## sigma2   0.02    0.00 0.00   0.02   0.02   0.02   0.02   0.03  3908    1
## lp__   844.01    0.02 1.18 841.05 843.47 844.31 844.87 845.37  3000    1
##
## Samples were drawn using NUTS(diag_e) at Tue Dec 01 13:49:34 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
plot(post, pars=c("a","b","sigma2"),plotfun= "trace", inc_warmup = TRUE)
```



```r
plot(post, pars=c("a","b","sigma2"), plotfun= "trace", inc_warmup = FALSE)
```



```
#plot(post, pars=c("mu"), plotfun= "trace", inc_warmup = FALSE)
```

Visualize and summarize parameter posteriors in original scale. Extract the posterior samples as a matrix. NOTE THAT ABOVE YOU OBTAINED THE PARAMETERS $\dot{a}$, $\dot{b}$ and $\dot{\sigma}^2$. You should continue with the original parameters $\dot{a}$, $\dot{b}$ and $\dot{\sigma}^2$ from now on. So make the needed transformations. If you have not solved the transformations, you should draw histograms with samples in variable post.

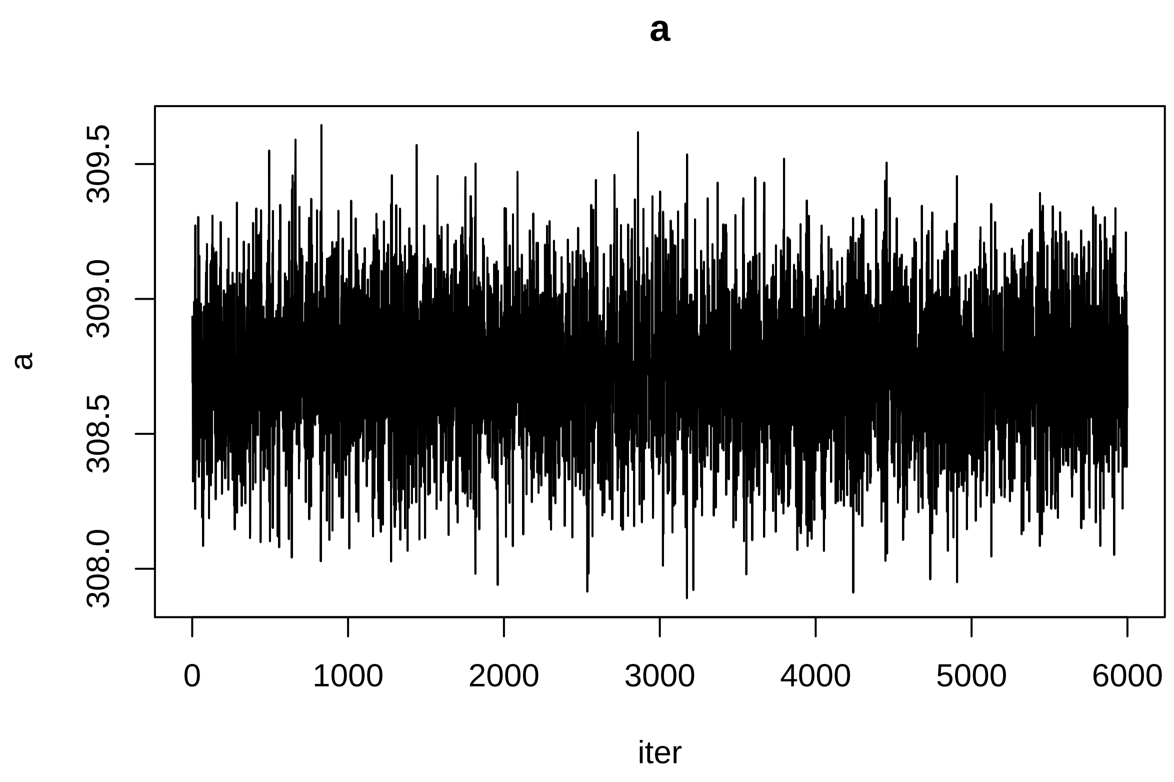
```
post_sample=as.matrix(post, pars =c("a","b","sigma2"))
dim(post_sample)
```

```
## [1] 6000    3
```

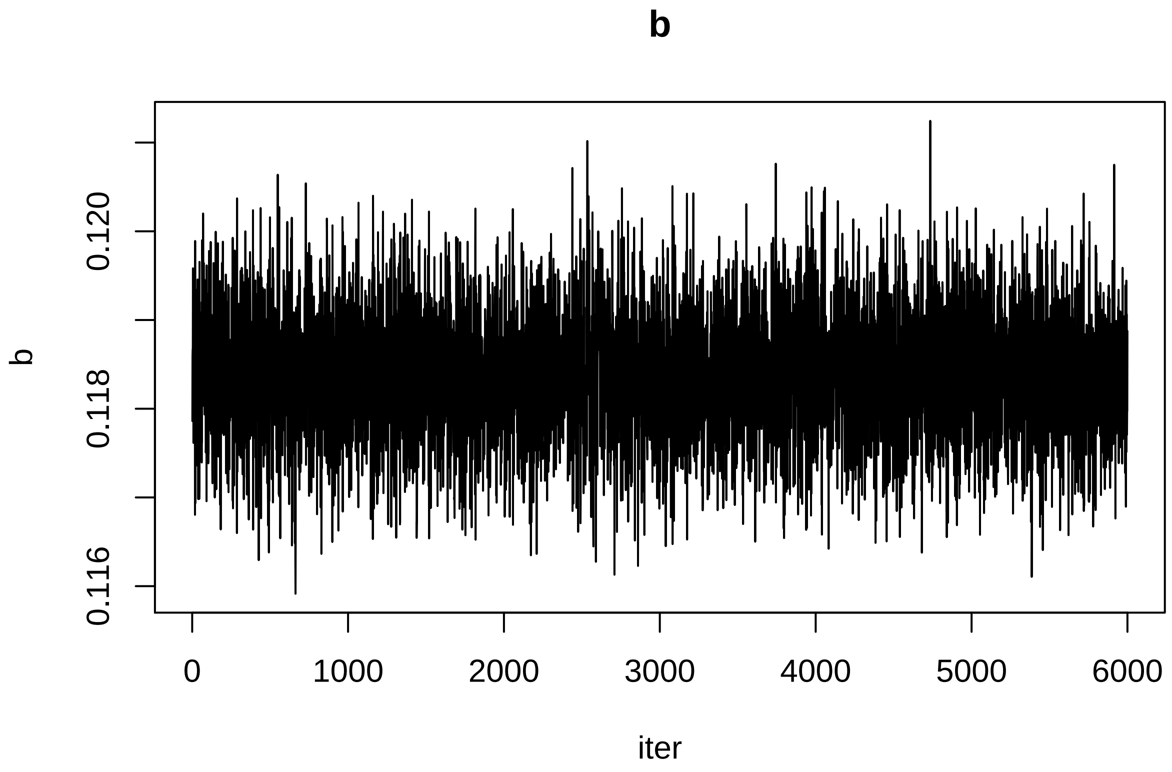
```
#one column contains posterior samples for one variable
a_dot=post_sample[,1]
b_dot=post_sample[,2]
sigma2_dot=post_sample[,3]

a = stdy*(a_dot - b_dot * mx / stdx) + my;
b = b_dot*stdy/stdx
sigma2 = stdy^2*sigma2_dot
```

Now parameter a contains a sample from the posterior $p(a|y, x, n)$ and parameter b contains sample from the posterior $p(b|y, x, n)$. We can now plot sample chains and histograms of them and do the required summaries.

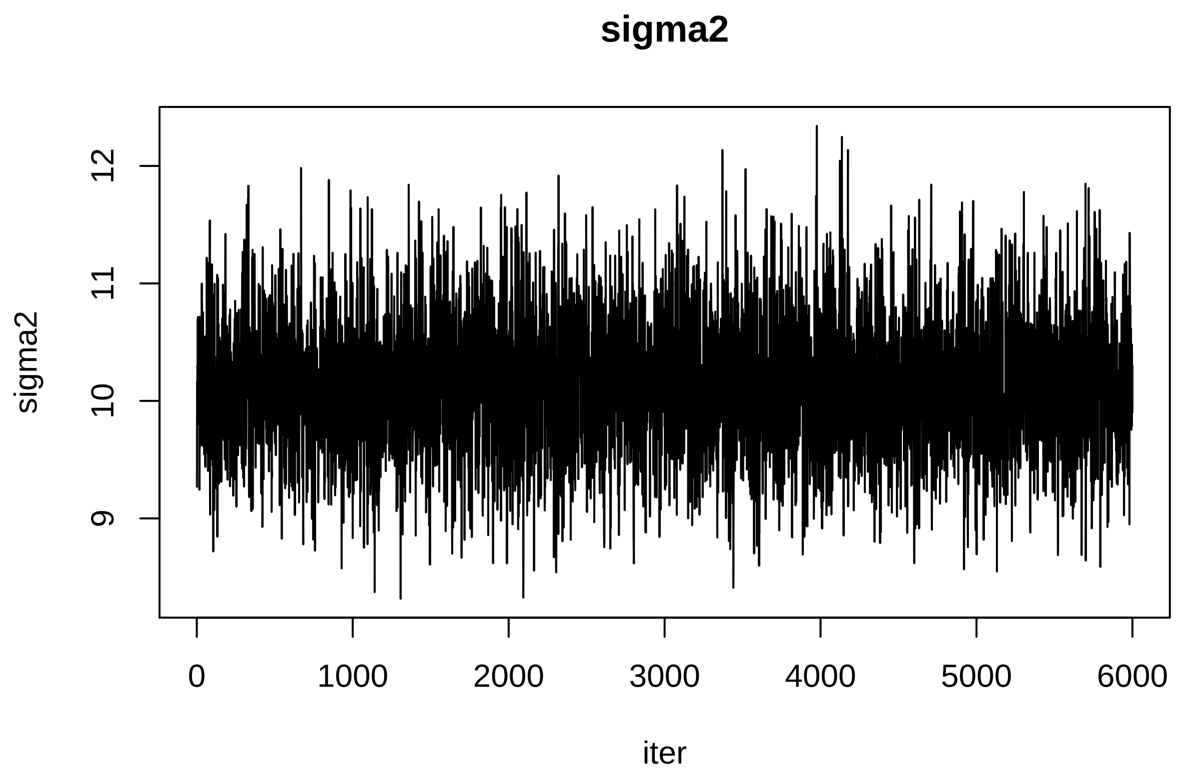
```r
#Trace plot of MCMC output to see if the chains have converged for the original parameters
plot(a, main="a", xlab="iter",type="l")
```



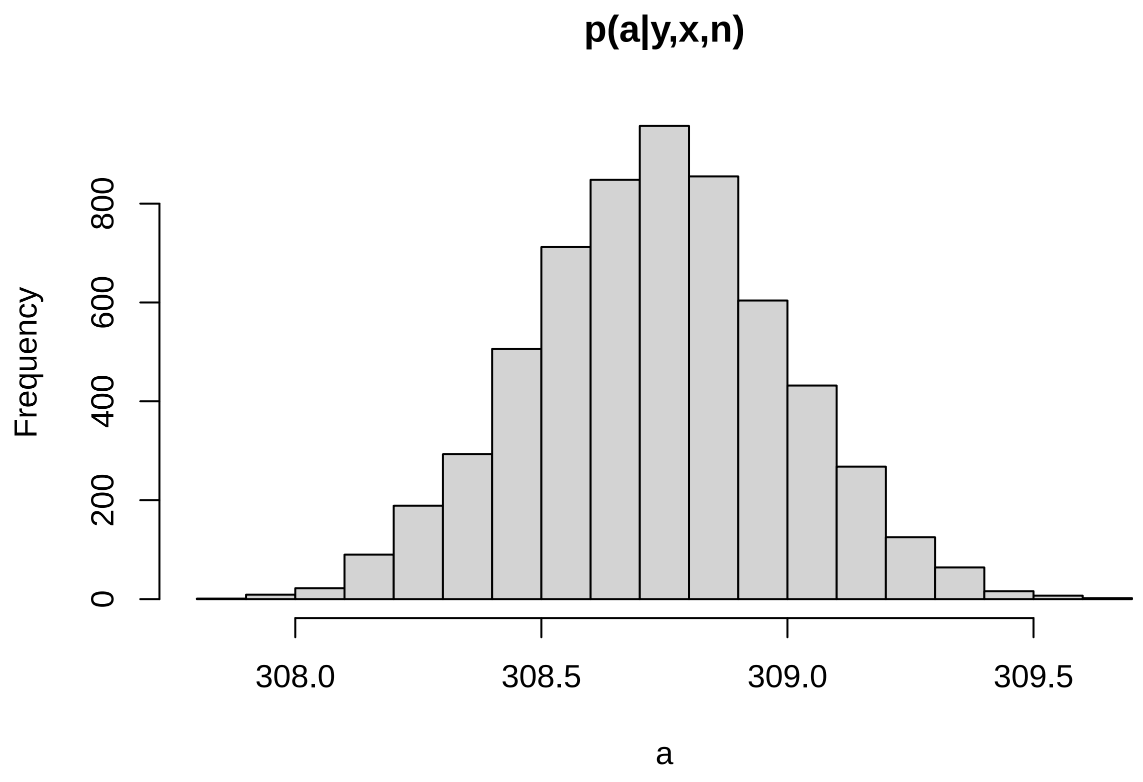
```r
plot(b, main="b", xlab="iter",type="l")
```



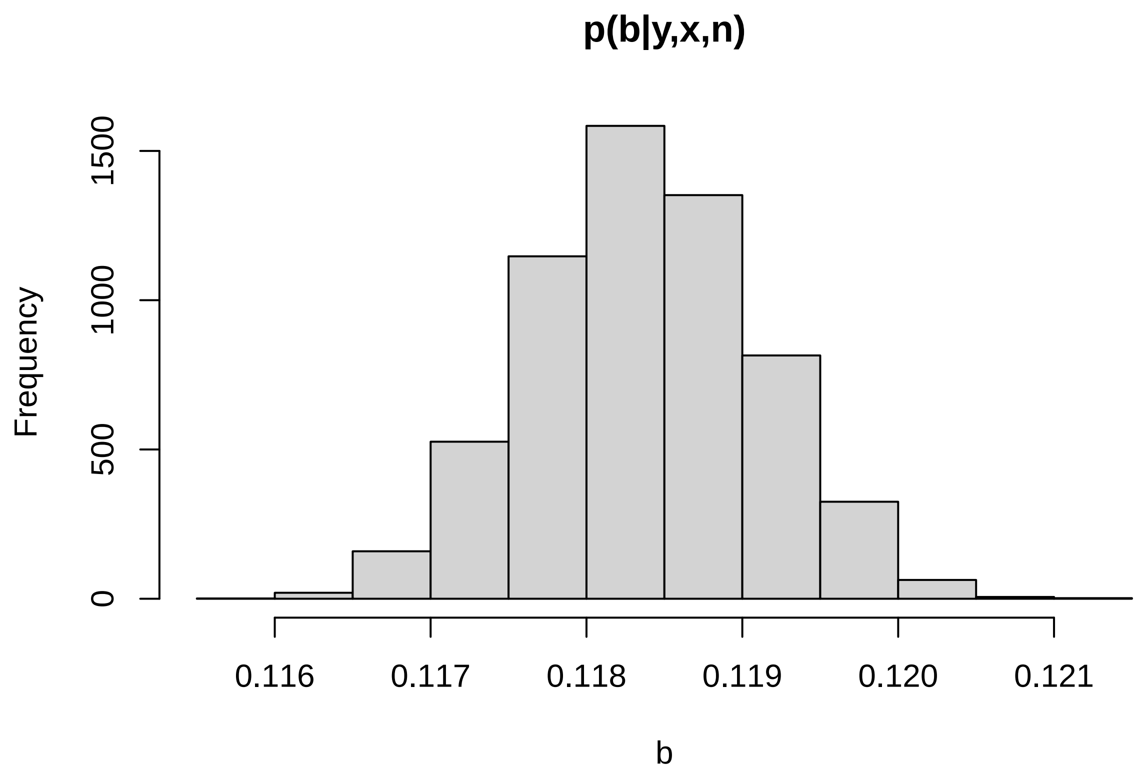
```r
plot(sigma2, main="sigma2", xlab="iter",type="l")
```



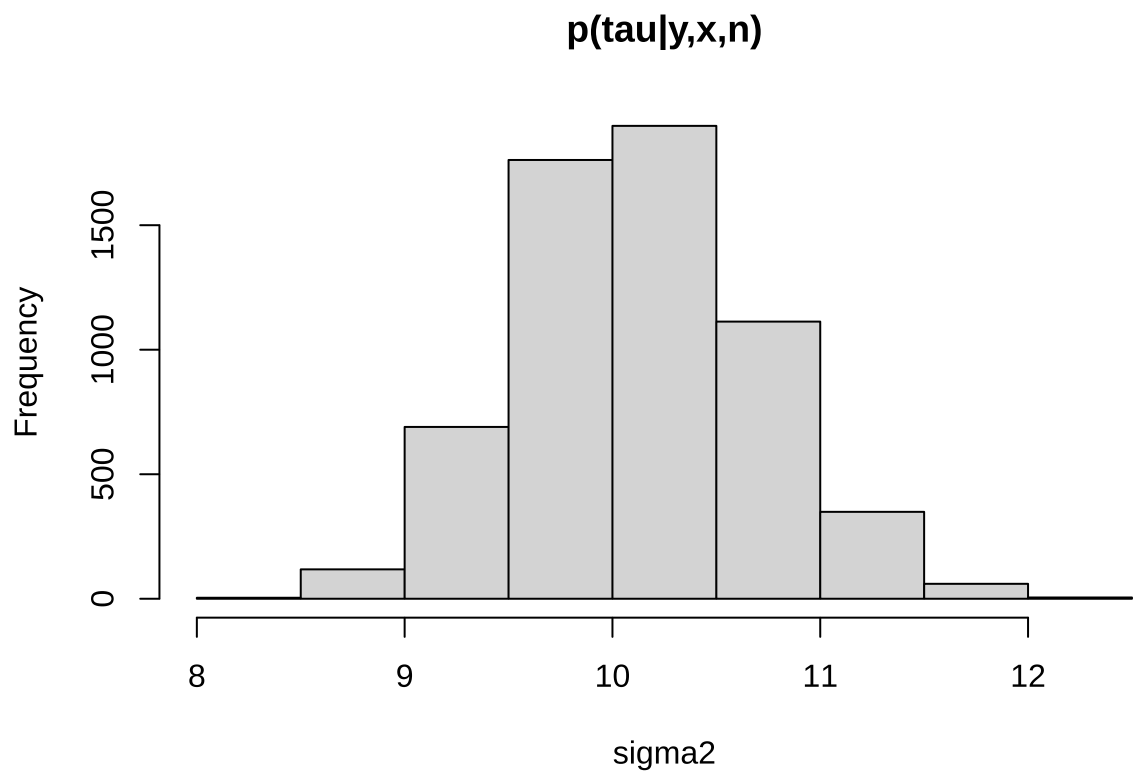
```
#Note, if the chains do not look converged see what is the problem and rerun the model

hist(a, main="p(a|y,x,n)", xlab="a")
```



```r
hist(b, main="p(b|y,x,n)", xlab="b")
```



```r
hist(sigma2, main="p(tau|y,x,n)", xlab="sigma2")
```



```r
#calculate the required summaries
paste("The mean of a:", mean(a))
```

```
## [1] "The mean of a: 308.731802592176"
```

```r
paste("The 2.5% and 97.5% posterior quantiles of a:")
```

```
## [1] "The 2.5% and 97.5% posterior quantiles of a:"
```

```r
quantile(a, probs=c(0.025, 0.975))
```

```
##     2.5%    97.5% 
## 308.2223 309.2361
```

```r
paste("The mean of b:", mean(b))
```

```
## [1] "The mean of b: 0.118379291531113"
```

```r
paste("The 2.5% and 97.5% posterior quantiles of b:")
```

```
## [1] "The 2.5% and 97.5% posterior quantiles of b:"
```

```
quantile(b, probs=c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.1169571 0.1198109
```

```
paste("The mean of sigma2::", mean(sigma2))
```

```
## [1] "The mean of sigma2:: 10.1241455781527"
```

```
paste("The 2.5% and 97.5% posterior quantiles of sigma2:")
```

```
## [1] "The 2.5% and 97.5% posterior quantiles of sigma2:"
```

```
quantile(sigma2, probs=c(0.025, 0.975))
```

```
##     2.5%    97.5%
##  9.05114 11.28537
```

3. Interpretation of $\mu(x)$ and ϵ_i

Linear mean function $\mu(x)$ is the function of the data we are interested in linearly changing with the covariate, which contains an intercept and a slope. The slope is a parameter that changes with the rate of change of the covariate. The larger the slope, the more obvious the linear change of the data we are interested in with the covariate. The intercept is a fixed value. For different covariates, the results we want have the same intercept.

The error terms ϵ_i are the measurement errors in simulated reality. It is uncertain, that is, each measurement will have different errors. Here, a normal distribution is used to simulate, that is, small errors will be very frequent but large errors will be very rare.

4. visualization of the regression curve

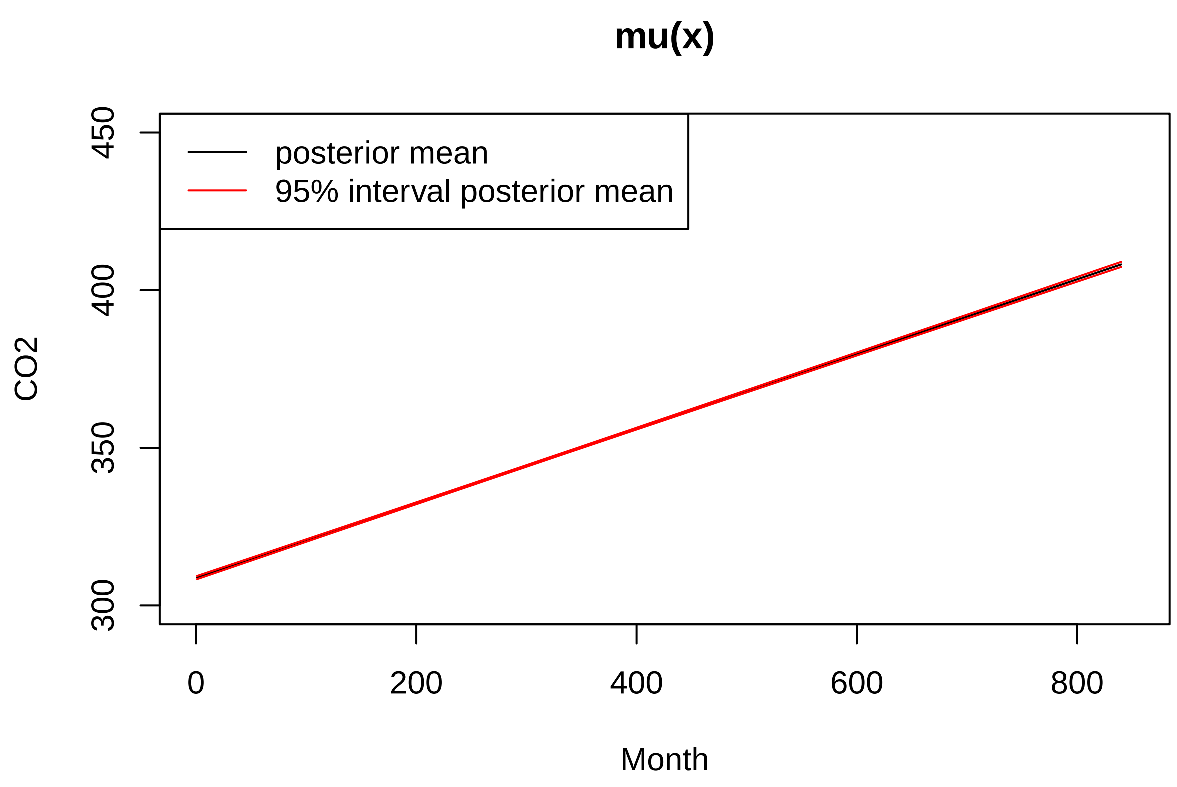
Data covers years from 1958 to 2008. Therefore, we need to construct prediction points and predict the historical and future next 20 years of CO2 concentrations

```
x.pred= seq(1,70*12,length=70*12)

mu = matrix(NA,length(x.pred),length(b))      # matrix of posterior samples of mu
y.tilde = matrix(NA,length(x.pred),length(b)) # matrix of posterior samples of y.tilde

mean_mu=rep(NA, length(x.pred))              # posterior mean of mu
int_mu = matrix(NA,length(x.pred),2)         # posterior 95% interval of mu

mean_y=rep(NA, length(x.pred))              # posterior mean of y.tilde
int_y = matrix(NA,length(x.pred),2)         # posterior 95% interval of y.tilde

for (i in 1:length(x.pred)) {
  #remember mu_i = a + b*x_i
  mu[i,] = a + b * x.pred[i]
  mean_mu[i] = mean(mu[i,])
  int_mu[i,] = quantile(mu[i,], probs=c(0.025, 0.975))
  #y_i = mu_i + e_i and e_i ~ N(0,sigma2)
```

```r
  y.tilde[i,] = mu[i,] + rnorm(length(sigma2), 0, sqrt(sigma2))
  mean_y[i] = mean(y.tilde[i,])
  int_y[i,] = quantile(y.tilde[i,], probs=c(0.025, 0.975))
}

# plot the mean and quantiles for mean function and (replicate) observations and the real observations
# Note! plot these in the original scale
#fill in here and at the end plot the real observations
```

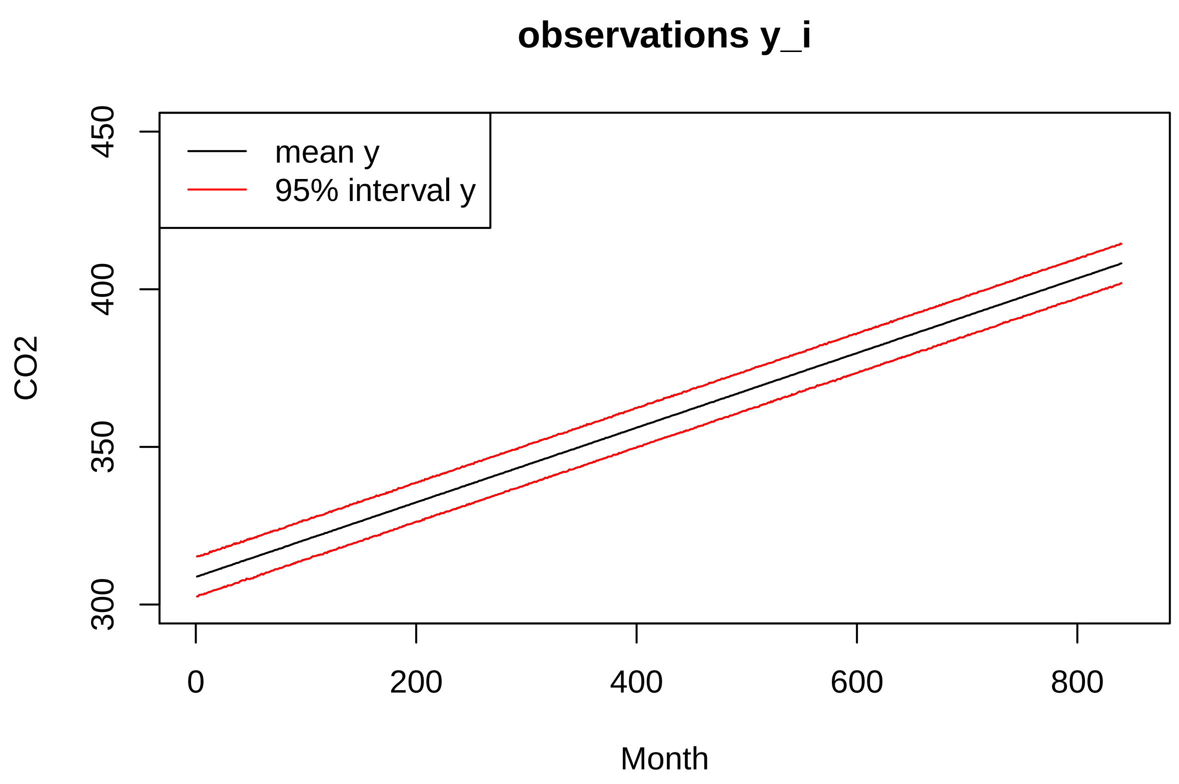
The posterior mean and 95% central posterior interval of the mean function $\mu(x)$ as a function of months from January 1958 to December 2027.

```r
plot(NA, xlim=c(1,850), ylim=c(300,450), xlab="Month", ylab="CO2", main="mu(x)")
lines(x.pred, mean_mu, col="black", type="l")
lines(x.pred, int_mu[,1], col="red")
lines(x.pred, int_mu[,2], col="red")
legend("topleft", legend=c("posterior mean", "95% interval posterior mean"),col=c("black", "red"), lty=
```



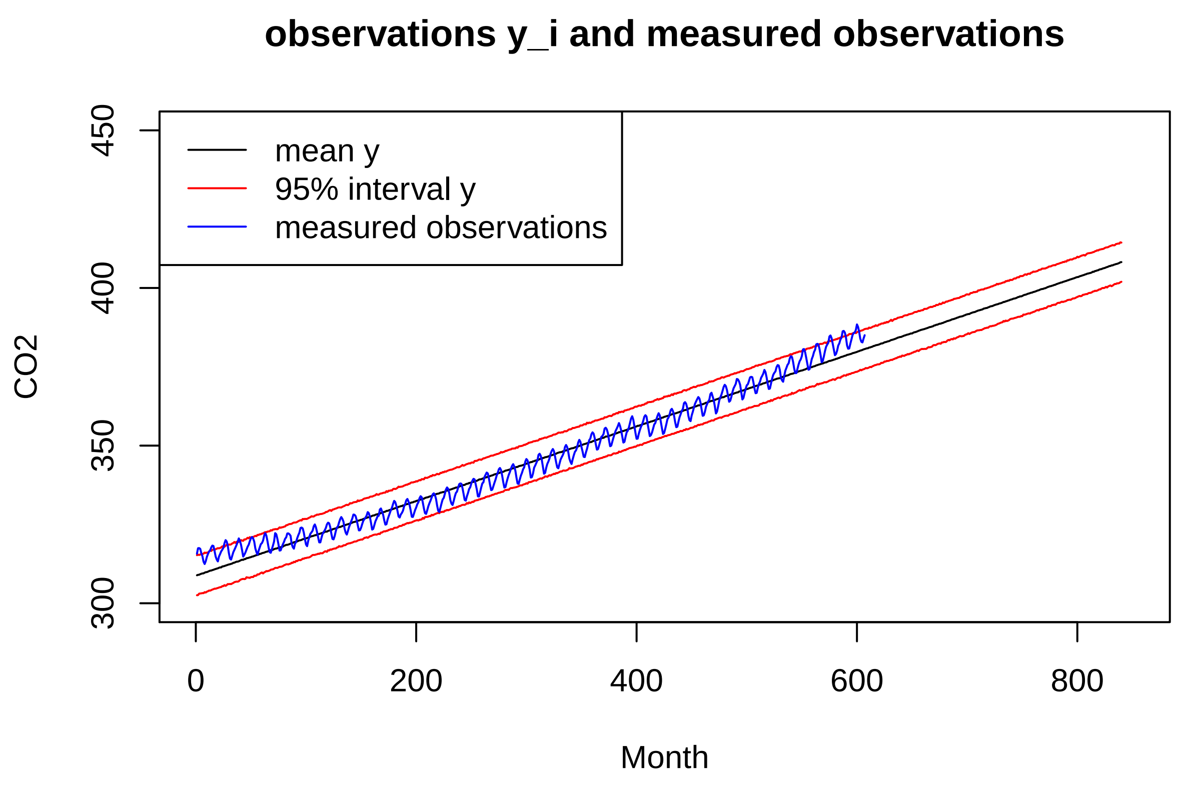
The posterior mean and 95% central posterior interval of observations y_i as a function of months from January 1958 to December 2027. In case of historical years, consider the distribution of potential replicate observations that have not been measured but could have been measured.

```
plot(NA, xlim=c(1,850), ylim=c(300,450), xlab="Month", ylab="CO2", main="observations y_i")
lines(x.pred, mean_y, col="black", type="l")
lines(x.pred, int_y[,1], col="red")
lines(x.pred, int_y[,2], col="red")
legend("topleft", legend=c("mean y", "95% interval y"), col=c("black", "red"), lty=1)
```



Plot also the measured observations to the same figure.

```
plot(NA, xlim=c(1,850), ylim=c(300,450), xlab="Month", ylab="CO2", main="observations y_i and measured o
lines(x.pred, mean_y, col="black", type="l")
lines(x.pred, int_y[,1], col="red")
lines(x.pred, int_y[,2], col="red")
lines(x.pred[1:607], y.month.orig, col="blue")
legend("topleft", legend=c("mean y", "95% interval y","measured observations"), col=c("black", "red","bl
```



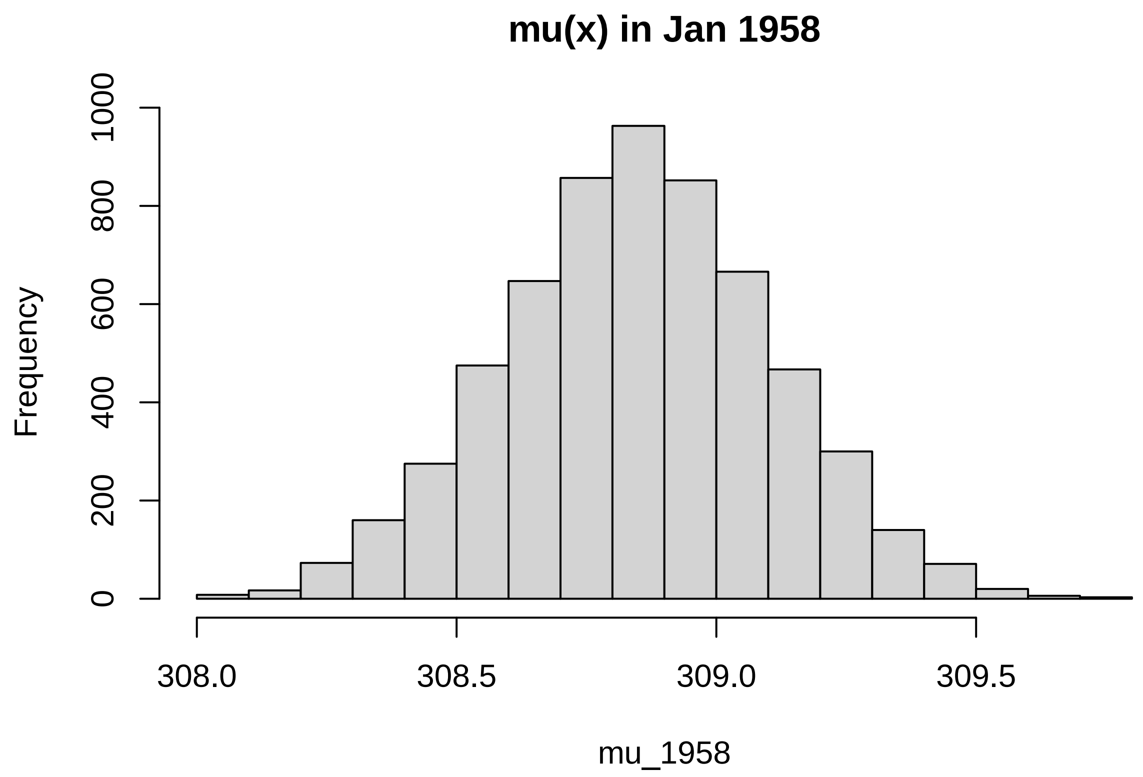
5. CO2 concentration in January 2025 and 1958

Posterior predictive distribution of the mean function in January 2025, January 1958 and the difference between these. Notice! x=1 corresponds to January 1958

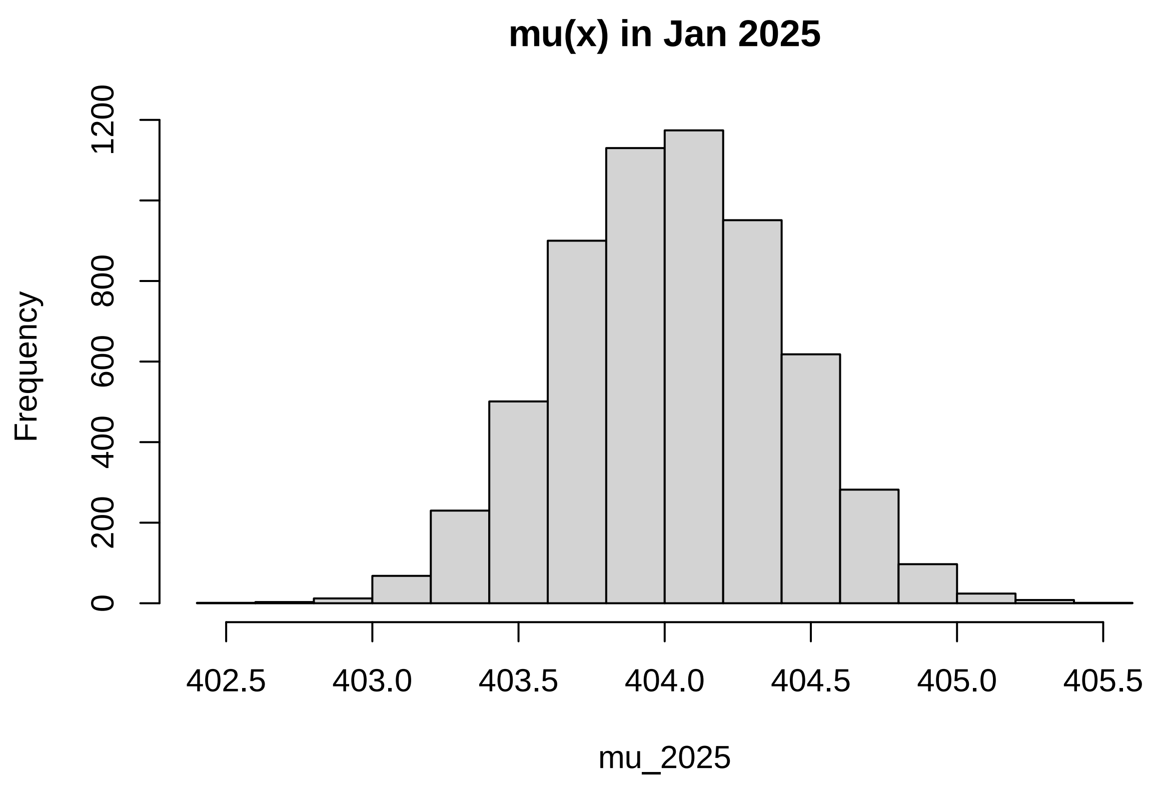
```
# Jan 1958
mu_1958 = mu[x.pred[1],]
# Jan 2025
mu_2025 = mu[x.pred[1 + (2025-1958) * 12],]
```

The Posterior predictive distribution of the mean function, $\mu(x)$ in January 2025 and in January 1958 and the difference between these.

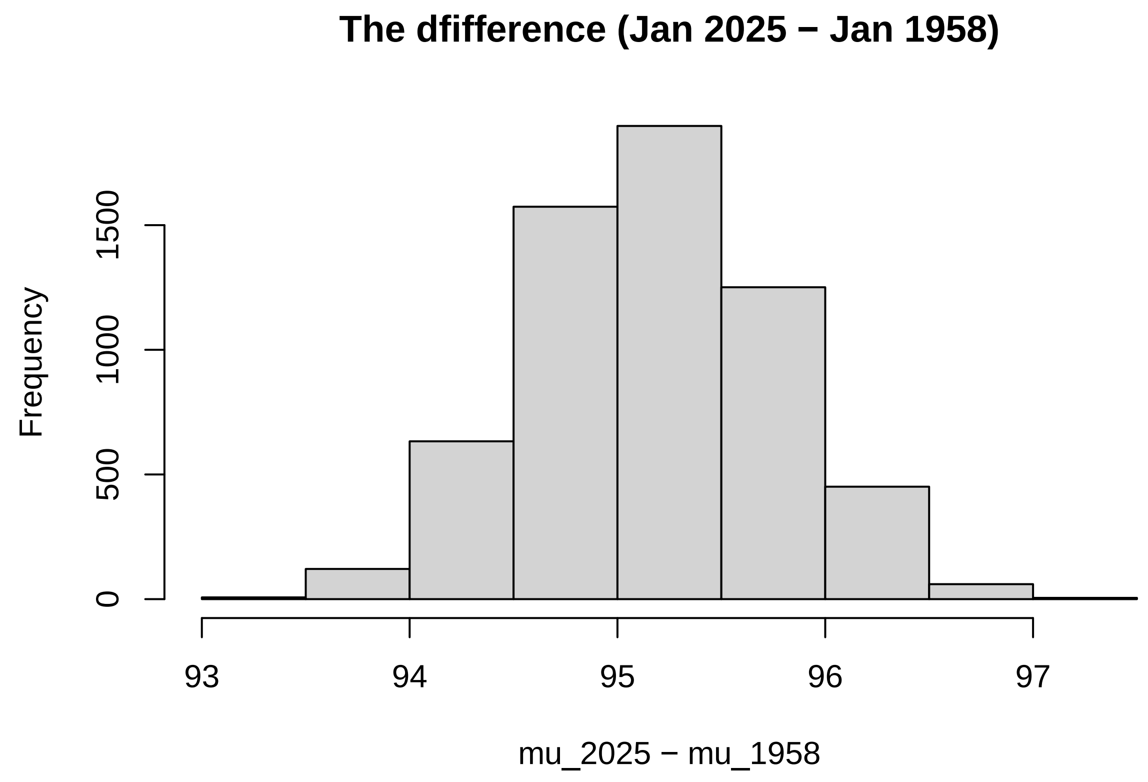
```
hist(mu_1958, main="mu(x) in Jan 1958")
```



```
hist(mu_2025, main="mu(x) in Jan 2025")
```

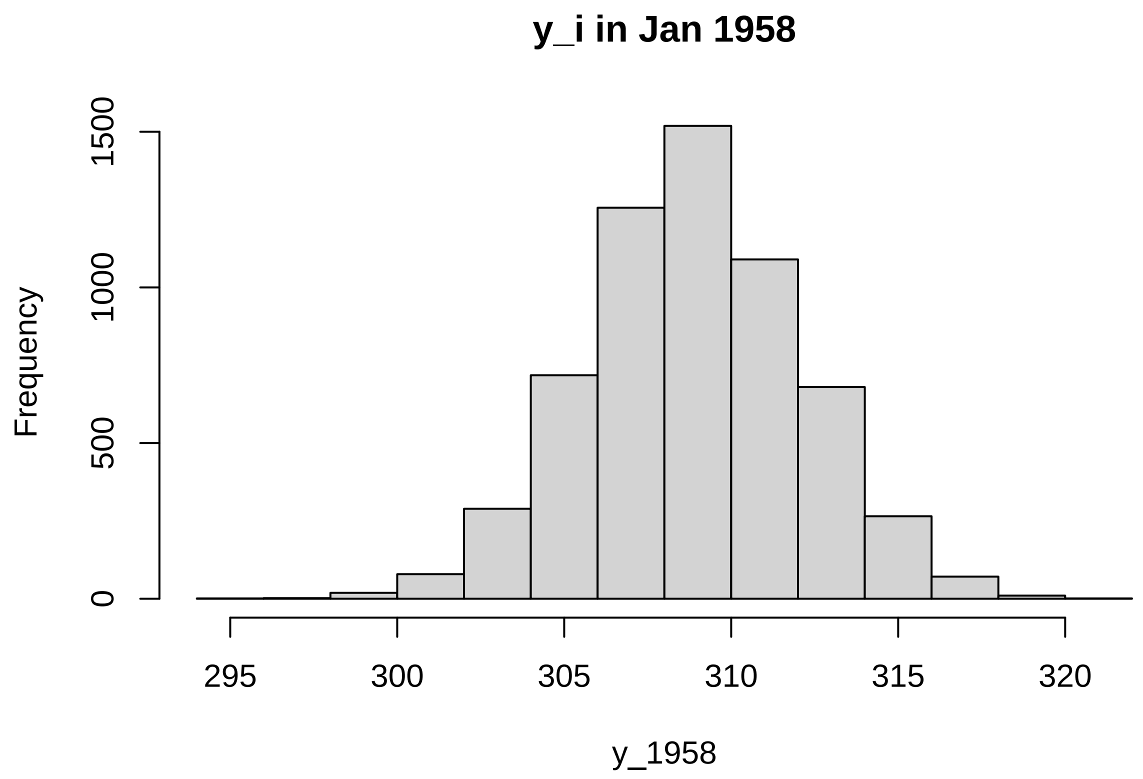


```r
hist(mu_2025 - mu_1958, main ="The dfifference (Jan 2025 - Jan 1958)")
```

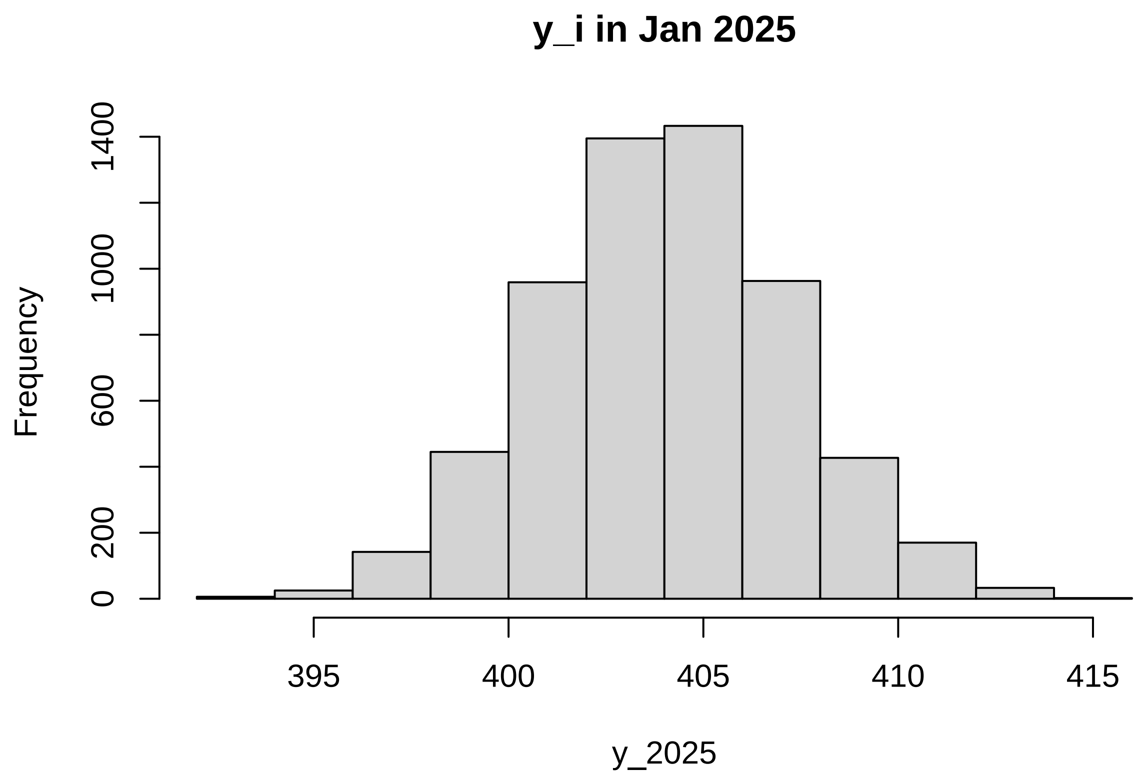


The Posterior predictive distribution of observations, y_i in January 2025 and in January 1958 and the difference between these.

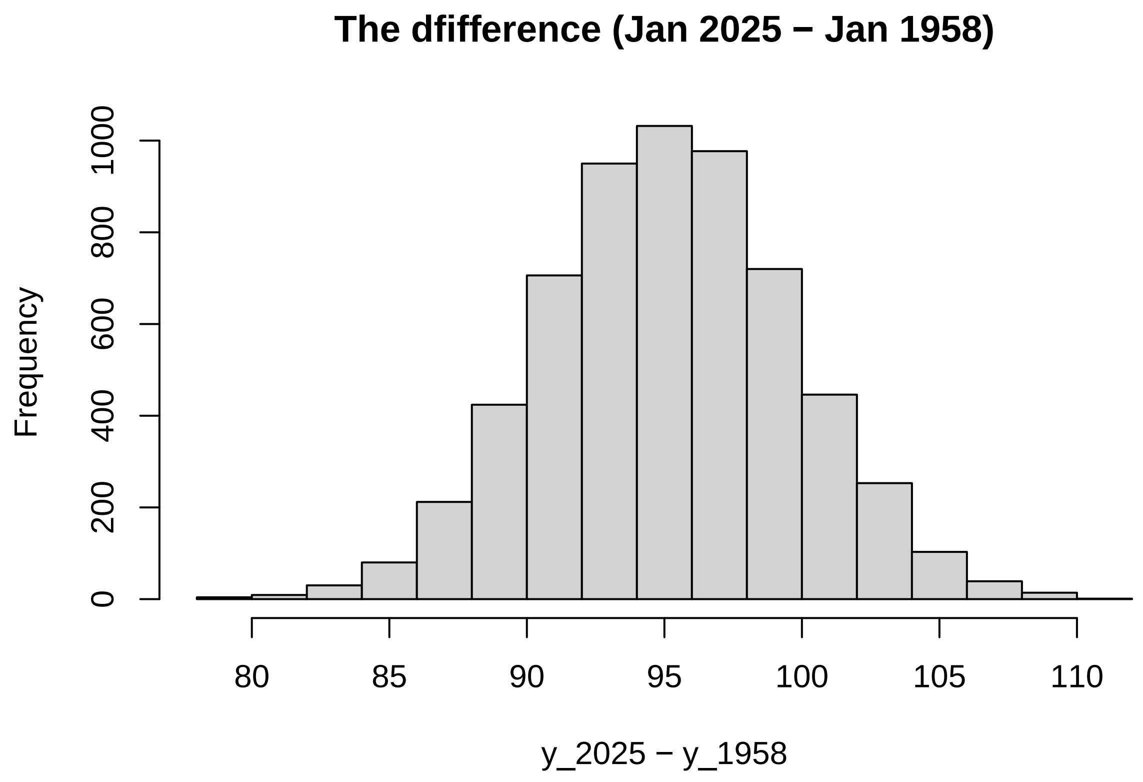
```
y_1958 = y.tilde[x.pred[1],]
y_2025 = y.tilde[x.pred[1 + (2025-1958) * 12],]

hist(y_1958, main="y_i in Jan 1958")
```



```r
hist(y_2025, main="y_i in Jan 2025")
```



```r
hist(y_2025 - y_1958, main = "The dfifference (Jan 2025 - Jan 1958)")
```



Discuss why the distributions of $\mu(x_i)$ and y_i differ.

Because $\mu(x_i)$ does not take into account the error terms, it is a linear regression result under perfect conditions, and y_i takes into account the error terms such as measurement errors, so it will be different from the theoretical value.