Denis Ostroushko - PUBH 7440 - HW5

Part A

Figure 1 shows that a linear trend is a reasonable assumption for this model.

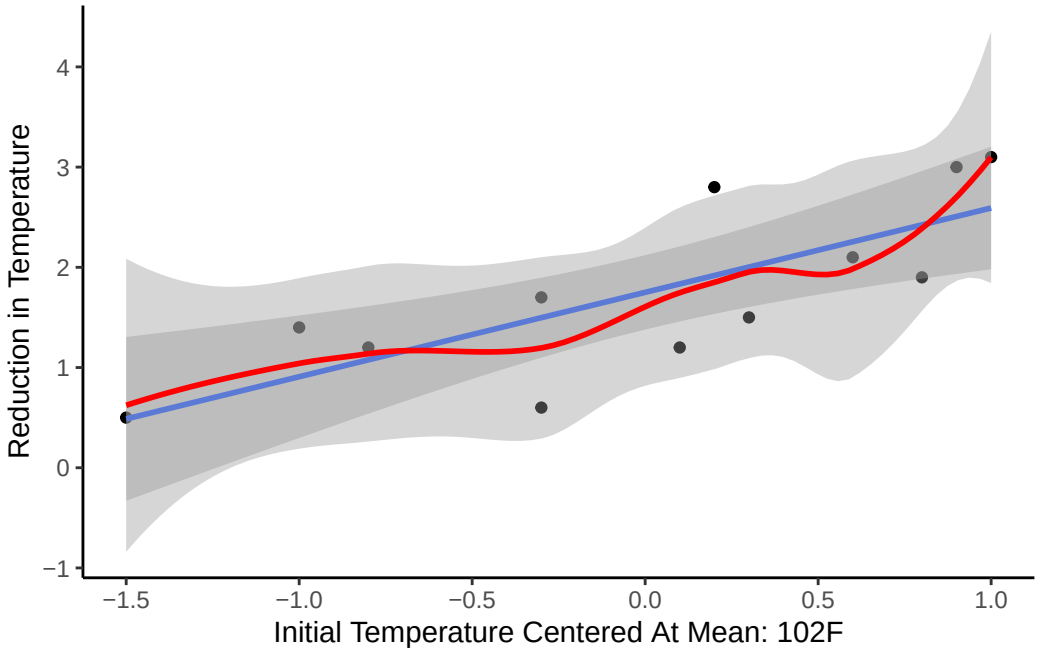

Figure 1: Reduction in termperature follows a linear trend agaisnt iniital temperature

Part B

For this problem we consider that a linear change in temperature as a function of initial temperature. Define a change in temperature as Z_i, and initial temperature as x_i. As a predictor, we consider $x_i^* = (x_i - \bar{x})$, which is a covariate centered at its mean.

$Z_i = \beta_0 + \beta_1 \times x_i^* + \epsilon_i$. According to the linear model assumptions, we consider that residuals ϵ_i are distributed according to the normal distribution, $N(0, \sigma^2)$.

We treat $\beta_0, \beta_1, \sigma^2$ as random variables, and therefore impose prior distributions on them. β_0, β_1 will have improper flat priors, and variance has $IG(0.001, 0.001)$ as its prior. Using this setup, we use `nimble` package to run MCMC and get posterior estimates.

Results

Figure 2 shows posterior samples for β_1. Approximately 0.1% of posterior samples are below 0, which given us strong evidence that children with higher initial temperatures experienced greater reduction in temperature after taking aspirin.

95% Credible Interval covers values from 0.36 to 1.33

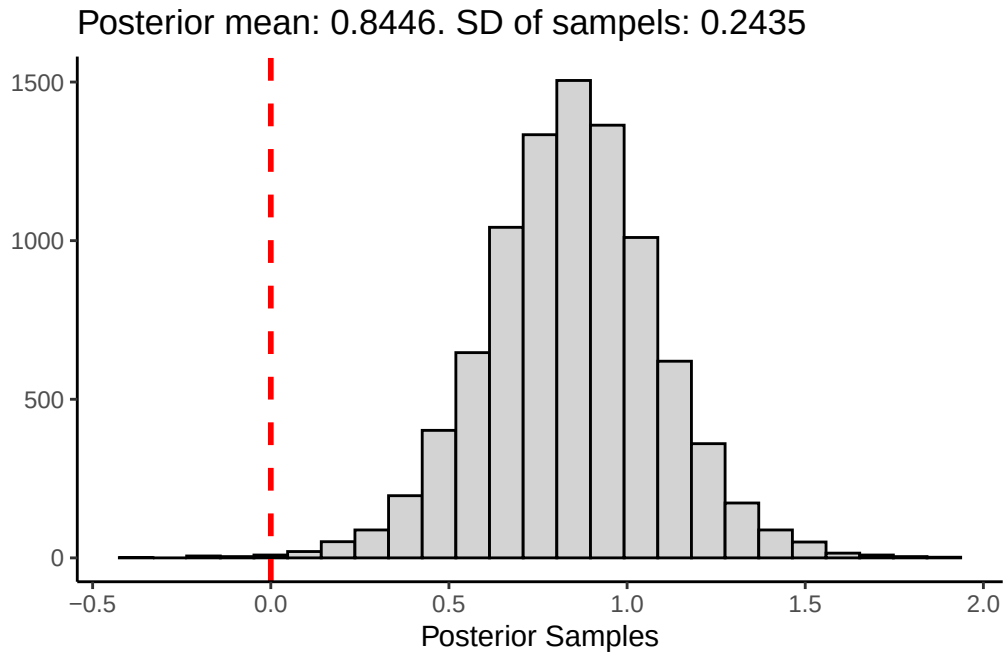

Figure 2: Pr(Beta1 < 0) is approxiamtely 0.1%

To calculate p_D and DIC, I used the following steps:

1. Obtain final estimate of σ, which I took as the mean of posterior samples. I estimated that $\hat{\sigma} = 0.6256$
2. For each observation Z_i, get mean, that is $E[Z_i]$ as the mean of posterior samples.
3. Evaluate log-likelihood of observing each Z_i under mean $E[Z_i]$ and standard deviation $\hat{\sigma}$. Sum these up for all 12 observations to obtain a deviance statistic using means of posterior samples, call this $D(\bar{\theta})$. This is quantity (1)

4. Evaluate log-likelihood of observing each Z_i at each iteration of sampling $j = 1, 2, \ldots, 9,000$ after discarding 1,000 samples as burnin period. Then, take the average of deviance statistics over 9,000 values of mean and sigma at each corresponding sampling iteration. This is quantity (2)
5. Calculate effective number of parameters $p_D = E_{\theta|y}[D] - D[E_{\theta|y}(\theta)] = (2) - (1)$
6. Calculate $DIC = E_{\theta|y}[D] + p_D$

For this model, $p_D = 2.09$ and $DIC = 23.5$. The WAIC is 24.31 and pWAIC is 1.93. WAIC and pWAIC are given as output of the `nimble` MCMC software, while DIC and p_D were calculated using posterior samples and base R code.

We have 3 model parameters, two model coefficients and estimate for residual variance. Both p_D and pWAIC are below that number, which follows the general rule fo thumb. Additionally, pWAIC and DIC both present approximately equal values, which also reinforces confidence in the calculation accuracy,

Part C

For this problem I considered two approaches:

1. Using Nimble model with a supplied value of x_{13} and a missing value of Z_{13}, track values of fitted mean and sampled values of Z_{13}. Evaluate posterior distribution of Z_{13}, treat it as the predictive posterior density
2. Use average values of posteriors for β_0, β_1, and σ to estimate mean for Z_{13} and sample from a normal distribution with mean $E[Z_i]$ and variance $\bar{\sigma^2}$.

Summary statistics for each method:

1. Using method 1 (Nimble), the average value, or predicted value, is -0.0977. Standard deviation of the fitted distribution was 0.8711
2. Using method 2, the average value, or predicted value, is -0.1089. Standard deviation of the fitted distribution was 0.6283

Figure 3 visualizes two posterior densities. Both distributions are approximately centered at 0 and appear normal. According to method 1, probability of reduction in temperature, $P(Z_{13} > 0)$, is 45.1% according to method 1, and about 42.59% according to method 2.

Overall the two methods produce similar results, while estimated posterior using nimble software is wider. I assume that this is due to the fact that:

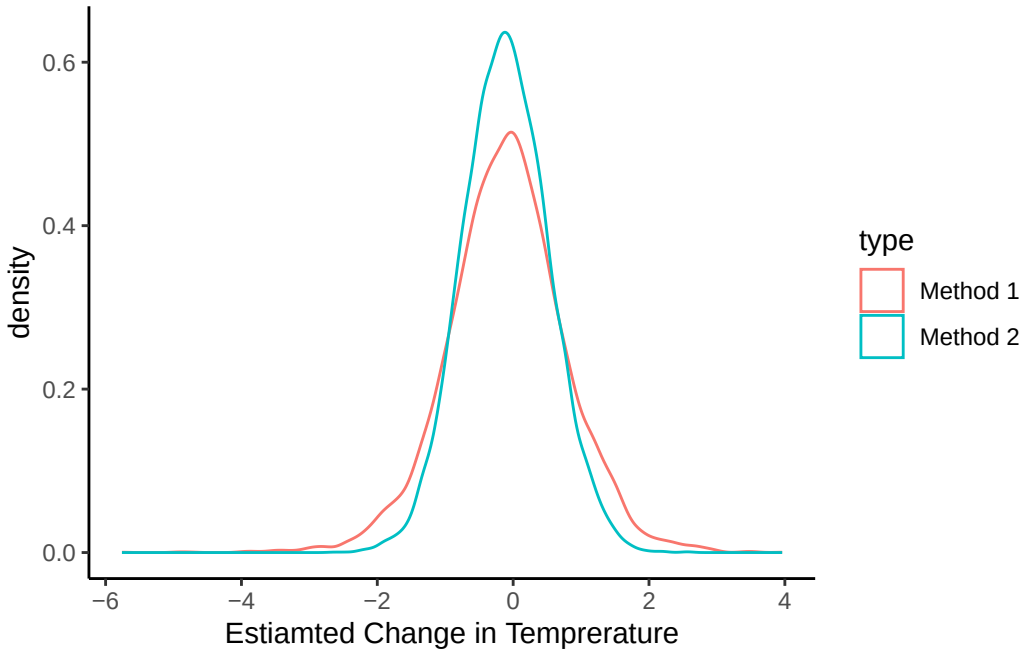

Figure 3: The two methods produce visually similar posterior predictive densities

1. Nimble possibly treated Z_{13} as a missing value, therefore there might be an extra penalty for imputing a missing value. Penalty in this case is higher variance for the posterior distribution.
2. The starting temperature of 100°F represents an extreme value towards the lower end of the x value spectrum. In my observation, confidence in predicting response Z tends to decrease, or confidence intervals widen, when examining values that align with the extremes of the predictor variable x in a linear predictor $\mathbf{x}$.

Part D

Residuals

Figure 4 visualizes studentized residuals with centered observed values of temperature change. Temperature change was centered at mean to align more closely with the residuals on the plot. There seems to be no issues with outliers. Residuals seems to be scattered randomly around the zero line.

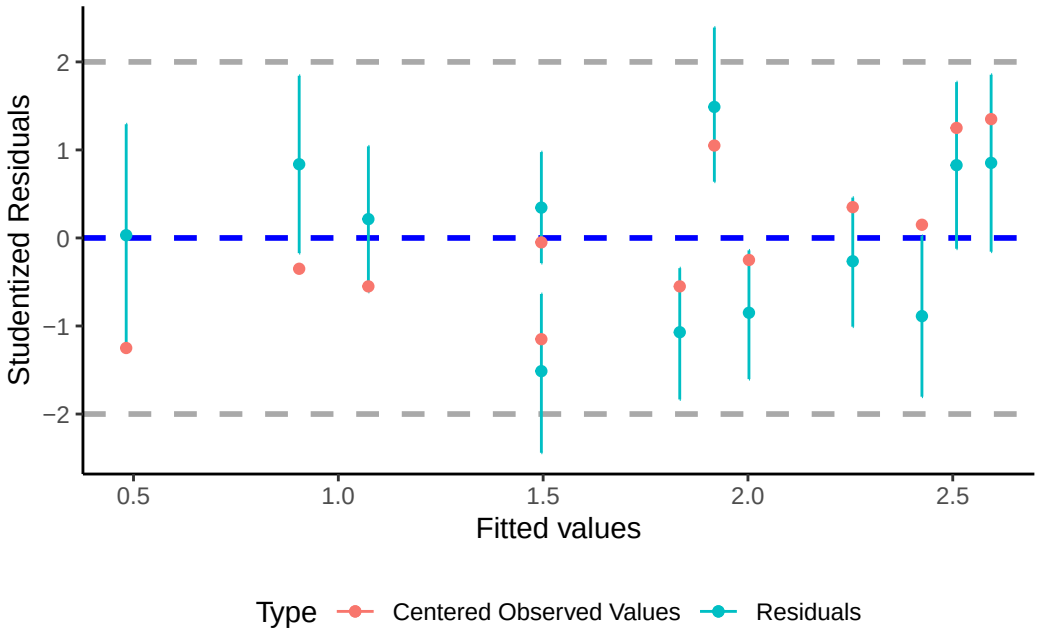

Figure 4: Residual plot does not show any evidence with model fit issues

CPO values

Figure 5 shows that CPO values for each observation is about the same. Overall, this shows no evidence of outliers in the data.

Part E

Recall previous model definition. In this section, we extend the model to include a quadratic term to allow for a non-linear trend in temperature change against initial temperature. The new proposed model is $Z_i = \beta_0 + \beta_1 \times x_i^* + \beta_2 \times (x_*)_i^2 + \epsilon_i$

Figure 6 presents posterior distribution for the coefficient of quadratic term in the new version of the linear model. 95% credible interval covers values from -0.4105 to 0.969. This does not provide enough statistical evidence to conclude that the coefficient for the quadratic term is diferent from 0.

For this model, $p_D = 3.73$ and $DIC = 24.55$. The WAIC is 25.96 and pWAIC is 2.92.

Figure 7 compares posterior predictive densities for Z_{13} using a nimble approach, and two types of linear model considered thus far.

First model with linear trend estimates mean for Z_{13} at -0.0977 with standard deviation of predictive posterior at 0.8711.

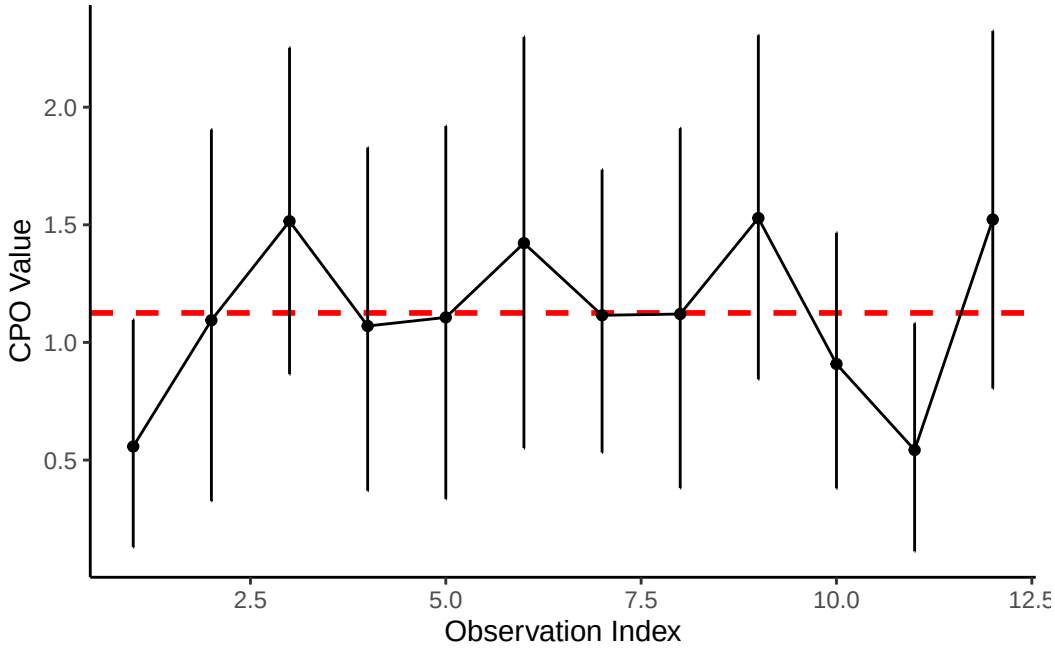

Figure 5: CPO plot does not show any evidence of outliers

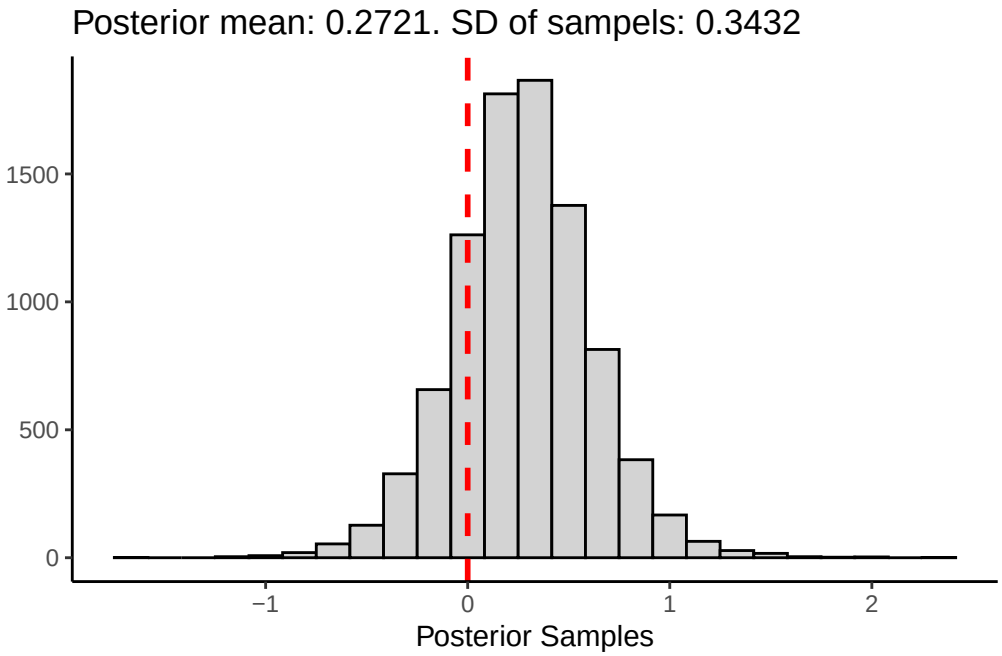

Figure 6: caption

Meanwhile, the second model with the non-linear trend estimated the mean of Z_{13} at 0.9319 and standard deviation of predictive posterior at 1.4715.

Figure 8 illuminates such drastic difference between the two predictions. As I assured earlier, the value $x_{13} = 100$ is the lower extreme of the x distribution. Centered value of $x^*_{13} = -2.2$. If we extrapolate the two fitted cureves, we can see that why predicted value using a non-linear trend model is much greater than the linear trend model.

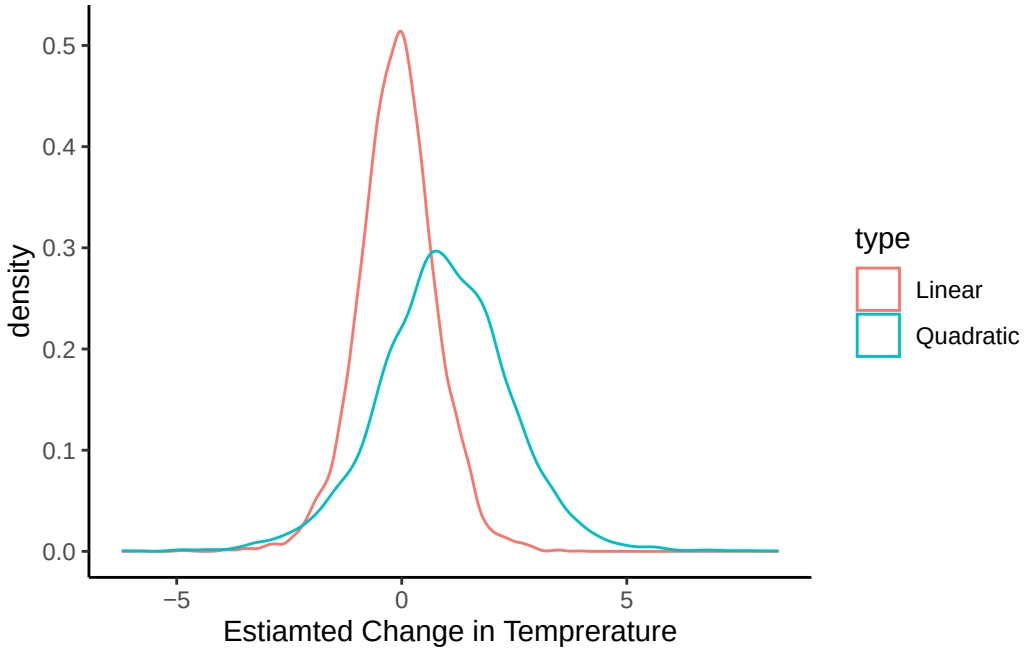

Figure 7: The two methods produce different posterior predictive densities

Overall, the two models produce similar results. Inclusion of a non-linear predictor does not significantly improve model fit to the data.

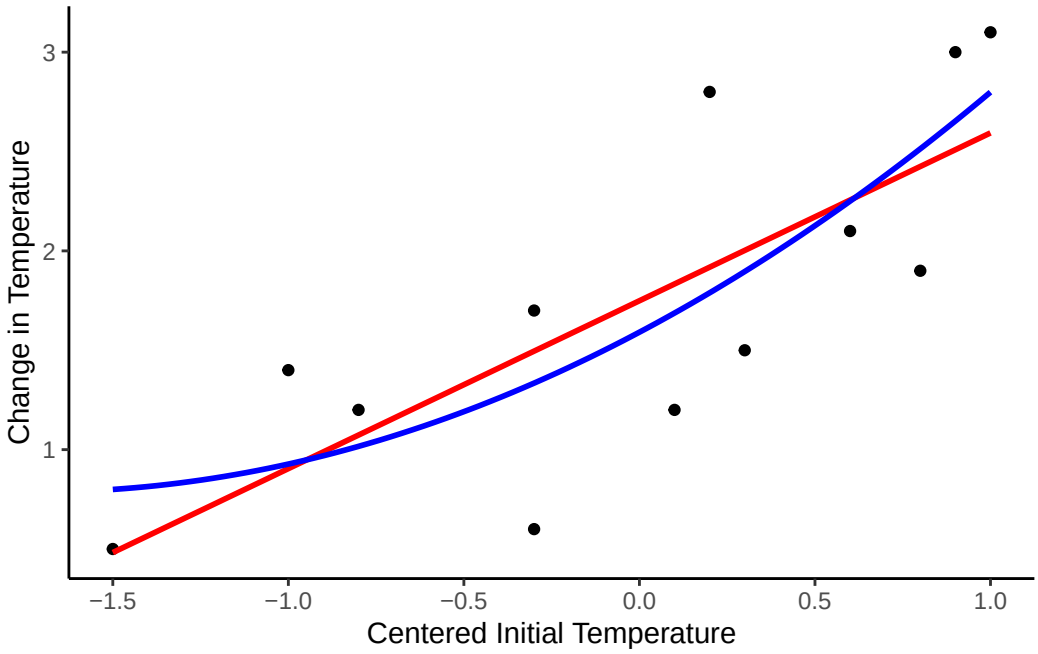

Figure 8: Comaprion of Linear and Non-linear trend models fit to the data

Appendix

```
X = c(102.4, 103.2, 101.9, 103.0, 101.2, 100.7, 102.5, 103.1, 102.8, 102.3, 101.9, 101.4)
Y = c(99.6, 100.1, 100.2, 101.1, 99.8, 100.2, 101.0, 100.1, 100.7, 101.1, 101.3, 100.2)
Z = X - Y

stored_mean = mean(X)

X_center = X - stored_mean

data_hw5 <-
  list(Z = Z,
       X = X_center)

#| label: fig-plot-partA
#| fig-cap: "Reduction in termperature follows a linear trend agaisnt iniital temperature"


ggplot(data = data.frame(x = X_center,
                         y = Z),
```

```
       aes(x = x, y = y)) +
  theme_classic() +
  geom_point() +
  stat_smooth(method = "lm") +
  stat_smooth(color = "red") +
  labs(x = "Initial Temperature Centered At Mean: 102F",
       y = "Reduction in Temperature")


N = length(data_hw5$X)

model_code <-
  nimbleCode({

    beta0 ~ dflat()
    beta1 ~ dflat()
    tau ~ dgamma(0.001, 0.001)
    sigma <- 1/sqrt(tau)

    ### this is a CHAT GPT suggestion


    for(i in 1:N){

      mu[i] <- beta0 + beta1 * X[i]
      Z[i] ~ dnorm(mu[i], tau)

      sresid_apr[i] <- (Z[i]-mu[i])/sigma
      outlier_apr[i] <- step(sresid_apr[i]-1.5) + step(-(sresid_apr[i]+1.5))
      CPO_apr[i] <-sqrt(tau)* exp(-tau/2*(Z[i]-mu[i])*(Z[i]-mu[i]))

      deviance[i] <- dnorm(Z[i], mu[i], tau, log = TRUE)

    }

  })

Initialize <- list(beta0 = 0,
                   beta1 = 1,
                   tau = 1)
```

```r
nimble_model <- nimbleModel(code = model_code, data = data_hw5)

mcmc.out <- nimbleMCMC(code = model_code,
                       model = nimble_model,
                       data = data_hw5,
                       inits = Initialize,
                       monitors = c("beta0", "beta1", "sigma", "tau", "Z", "mu", "sresid_a
                                    "outlier_apr", "CPO_apr", "deviance"),
                       thin = 1,
                       niter = 10000,
                       nburnin = 1000,
                       nchains = 1,
                       summary = T,
                       setSeed = T,
                       WAIC = T)

mcmc.out$summary

write_rds(mcmc.out, "./rds/hw5 res.rds")

#| label: fig-bayes-beta1
#| fig-cap: "Pr(Beta1 < 0) is approxiamtely 0.1% "


ggplot(
  data = data.frame(x = mcmc.out$samples[,"beta1"]),
  aes(x = x)
) +
  theme_classic() +
  geom_histogram(bins = 25, fill = "lightgrey", color = "black") +
  geom_vline(xintercept = 0, color = "red", linewidth = 1, linetype = "dashed") +

  labs(
    x = "Posterior Samples",
    y = " ",
    colour = "Value Type",
    title = paste0("Posterior mean: ", round( mean(mcmc.out$samples[,"beta1"]), 4), ". SD
  )


observed_data <- data_hw5$Z
```

```r
sigma_samples <- mcmc.out$samples[,"sigma"]

mu_cols <- which(
  substr(
    colnames(mcmc.out$samples), 1, nchar("mu")
    ) == "mu"
  )

mu_samples <- mcmc.out$samples[,mu_cols]

#### expectation over deviance from samples
###     since we apply a deviance function, it is always -2 times sum [ log  (stuff)  ]


#### This is D_bar
deviances <- rep(NA, nrow(mu_samples))

for(i in 1:length(deviances)){
  deviances[i] <- -2 * sum(dnorm(observed_data, mu_samples[i,], sigma_samples[i], log = TR
}

median(deviances) -> d_bar

#### this is D(theta bar)
-2 * sum(dnorm(observed_data,
               apply(mu_samples, 2, mean),
               mean(sigma_samples),
               log = TRUE)
         )  -> d_theta_bar

p_D_1 <- d_bar - d_theta_bar

DIC_1 <- d_bar + p_D_1

new_X = 100 - stored_mean

data_hw5_appraoch_1 <- data_hw5

data_hw5_appraoch_1$Z <- c(data_hw5_appraoch_1$Z, NA)
data_hw5_appraoch_1$X <- c(data_hw5_appraoch_1$X, new_X)
```

```r
N = length(data_hw5_appraoch_1$X)

model_code <-
  nimbleCode({

    beta0 ~ dflat()
    beta1 ~ dflat()
    tau ~ dgamma(0.001, 0.001)
    sigma <- 1/sqrt(tau)

    for(i in 1:N){

      mu[i] <- beta0 + beta1 * X[i]
      Z[i] ~ dnorm(mu[i], tau)

    }

  })

Initialize <- list(beta0 = 0,
                   beta1 = 1,
                   tau = 1)

nimble_model <- nimbleModel(code = model_code, data = data_hw5_appraoch_1)

mcmc.out_app1 <- nimbleMCMC(code = model_code,
                        model = nimble_model,
                        data = data_hw5_appraoch_1,
                        inits = Initialize,
                        monitors = c("beta0", "beta1", "sigma", "tau", "Z", "mu"),
                        thin = 1,
                        niter = 10000,
                        nburnin = 1000,
                        nchains = 1,
                        summary = T,
                        setSeed = T,
                        WAIC = T)

mcmc.out_app1$summary

write_rds(mcmc.out_app1, "./rds/hw5 predict nimble.rds")
```

```r
new_X = 100 - stored_mean

expected_value <- mean(mcmc.out$samples[,"beta0"]) + mean(mcmc.out$samples[,"beta1"] * new
sigma_ <- mean(mcmc.out$samples[,"sigma"])

predictive_posterior_samples <- rnorm(n = 10000, mean = expected_value, sd = sigma_)

# hist(predictive_posterior_samples)

### compare MUs and approach 3
  summary(mcmc.out_app1$samples[,"mu[13]"])
  summary(predictive_posterior_samples)

  mean(mcmc.out_app1$samples[,"mu[13]"])
  mean(predictive_posterior_samples)

  sd(mcmc.out_app1$samples[,"mu[13]"])
  sd(predictive_posterior_samples)

### compare Ys and approach 3
  summary(mcmc.out_app1$samples[,"Z[13]"])
  summary(predictive_posterior_samples)

  mean(mcmc.out_app1$samples[,"Z[13]"])
  mean(predictive_posterior_samples)

  sd(mcmc.out_app1$samples[,"Z[13]"])
  sd(predictive_posterior_samples)

#### probability of temperature reduction = P(Z > 0)
  length(which(mcmc.out_app1$samples[,"Z[13]"] > 0))/length(mcmc.out_app1$samples[,"Z[13]"
  length(which(predictive_posterior_samples > 0))/length(predictive_posterior_samples)

  #### So, basically, my approach is the same as more close to predicting MU using nimble,
  #### although with slightly higher variance
  #### will include two methods, and comapre the results that way

comp_plot_df <-
  data.frame(
    val = c(mcmc.out_app1$samples[,"Z[13]"],
            predictive_posterior_samples),
```

```
    type = c(rep("Nimble Z", length(mcmc.out_app1$samples[,"Z[13]"])),
             rep("Sampling Z", length(predictive_posterior_samples)))
  )


ggplot(data  = comp_plot_df,
       aes(x = val, color = type, group = type)) +
  theme_classic() +
  geom_density()



comp_plot_df <-
  data.frame(
    val = c(mcmc.out_app1$samples[,"mu[13]"],
            predictive_posterior_samples),

    type = c(rep("Nimble mu", length(mcmc.out_app1$samples[,"mu[13]"])),
             rep("Sampling mu", length(predictive_posterior_samples)))
  )


ggplot(data  = comp_plot_df,
       aes(x = val, color = type, group = type)) +
  theme_classic() +
  geom_density()


#| label: fig-comp-dens
#| fig-cap: "The two methods produce visually similar posterior predictive densities"


comp_plot_df <-
  data.frame(
    val = c(mcmc.out_app1$samples[,"Z[13]"],
            predictive_posterior_samples),

    type = c(rep("Method 1", length(mcmc.out_app1$samples[,"Z[13]"])),
             rep("Method 2", length(predictive_posterior_samples)))
  )
```

```r
ggplot(data  = comp_plot_df,
       aes(x = val, color = type, group = type)) +
  theme_classic() +
  geom_density() +
  labs(x = 'Estiamted Change in Temprerature')

#| label: fig-resid
#| fig-cap: "Residual plot does not show any evidence with model fit issues"


resid_cols <- which(
  substr(
    colnames(mcmc.out$samples), 1, nchar("sresid_apr")
    ) == "sresid_apr"
  )

fitted_val_cols <- which(
  substr(
    colnames(mcmc.out$samples), 1, nchar("mu")
    ) == "mu"
  )

mcmc.out$samples[, fitted_val_cols] %>% apply(., 2, mean)-> fitted_val
mcmc.out$samples[, resid_cols] %>% apply(., 2, mean)-> residual
mcmc.out$samples[, resid_cols] %>% apply(., 2, quantile, probs = 0.025)-> residual_lb
mcmc.out$samples[, resid_cols] %>% apply(., 2, quantile, probs = 0.975)-> residual_ub

resid_plot_df <-
  data.frame(fitted_val,
             residual,
             residual_lb,
             residual_ub
             )

ggplot(data = resid_plot_df,
       aes(x = fitted_val, y = residual)) +
  theme_classic() +
  geom_hline(yintercept = 0, color = "blue", linewidth = 1, linetype = "dashed") +
  geom_hline(yintercept = c(-2,2), color = "darkgrey", linewidth = 1, linetype = "dashed") +
  geom_errorbar(aes(ymin = residual_lb, ymax = residual_ub, width = 0, color = "Residuals"
  geom_point(aes(color = "Residuals")) +
```

```
  geom_point(data = data.frame(residual = data_hw5$Z - mean(data_hw5$Z), fitted_val),
             aes(fitted_val, residual, color = "Centered Observed Values")
             ) +
  labs(x = "Fitted values",
       y = "Studentized Residuals",
       color = "Type") +
  theme(legend.position = "bottom")

#| label: fig-cpo
#| fig-cap: "CPO plot does not show any evidence of outliers"



cpo_cols <- which(
  substr(
    colnames(mcmc.out$samples), 1, nchar("CPO")
    ) == "CPO"
  )

fitted_val_cols <- which(
  substr(
    colnames(mcmc.out$samples), 1, nchar("mu")
    ) == "mu"
  )

mcmc.out$samples[, fitted_val_cols] %>% apply(., 2, mean)-> fitted_val
mcmc.out$samples[, cpo_cols] %>% apply(., 2, mean)-> cpo
mcmc.out$samples[, cpo_cols] %>% apply(., 2, quantile, probs = 0.025)-> cpo_lb
mcmc.out$samples[, cpo_cols] %>% apply(., 2, quantile, probs = 0.975)-> cpo_ub

resid_plot_df <-
  data.frame(i = 1:length(cpo),
             cpo,
             cpo_lb,
             cpo_ub
             )

ggplot(data = resid_plot_df,
       aes(x = i, y = cpo)) +
  theme_classic() +
  geom_hline(color = "red", linewidth = 1, linetype = "dashed", yintercept = mean(cpo)) +
```

```
  geom_errorbar(aes(ymin = cpo_lb, ymax = cpo_ub, width = 0)) +
  geom_point() +
  geom_line() +
  labs(x = "Observation Index",
       y = "CPO Value")

data_hw5_appraoch_2 <- data_hw5

all_x <- c(X)

data_hw5_appraoch_2$Z <- c(data_hw5_appraoch_2$Z)
data_hw5_appraoch_2$X <- all_x - mean(all_x)
data_hw5_appraoch_2$X2 <- (data_hw5_appraoch_2$X)^2

N = length(data_hw5_appraoch_2$X)

model_code <-
  nimbleCode({

    beta0 ~ dflat()
    beta1 ~ dflat()
    beta2 ~ dflat()

    tau ~ dgamma(0.001, 0.001)
    sigma <- 1/sqrt(tau)

    for(i in 1:N){

      mu[i] <- beta0 + beta1 * X[i] + beta2 *X2[i]
      Z[i] ~ dnorm(mu[i], tau)

    }

  })

Initialize <- list(beta0 = 0,
                   beta1 = 1,
                   beta2 = 1,
                   tau = 1)

nimble_model <- nimbleModel(code = model_code, data = data_hw5_appraoch_2)
```

```r
mcmc.out_quadratic <- nimbleMCMC(code = model_code,
                         model = nimble_model,
                         data = data_hw5_appraoch_2,
                         inits = Initialize,
                         monitors = c("beta0", "beta1", "beta2", "sigma", "tau", "Z", "mu"),
                         thin = 1,
                         niter = 10000,
                         nburnin = 1000,
                         nchains = 1,
                         summary = T,
                         setSeed = T,
                         WAIC = T)

mcmc.out_quadratic$summary

write_rds(mcmc.out_quadratic, "./rds/hw5 quardatic nimble.rds")


mcmc.out_quadratic <- read_rds("./rds/hw5 quardatic nimble.rds")

data_hw5_appraoch_2 <- data_hw5

all_x <- c(X, 100)

data_hw5_appraoch_2$Z <- c(data_hw5_appraoch_2$Z, NA)
data_hw5_appraoch_2$X <- all_x - mean(all_x)
data_hw5_appraoch_2$X2 <- (data_hw5_appraoch_2$X)^2

N = length(data_hw5_appraoch_2$X)

model_code <-
  nimbleCode({

    beta0 ~ dflat()
    beta1 ~ dflat()
    beta2 ~ dflat()

    tau ~ dgamma(0.001, 0.001)
    sigma <- 1/sqrt(tau)

    for(i in 1:N){
```

```
      mu[i] <- beta0 + beta1 * X[i] + beta2 *X2[i]
      Z[i] ~ dnorm(mu[i], tau)

    }

  })

Initialize <- list(beta0 = 0,
                   beta1 = 1,
                   beta2 = 1,
                   tau = 1)

nimble_model <- nimbleModel(code = model_code, data = data_hw5_appraoch_2)

mcmc.out_quadratic2 <- nimbleMCMC(code = model_code,
                        model = nimble_model,
                        data = data_hw5_appraoch_2,
                        inits = Initialize,
                        monitors = c("beta0", "beta1", "beta2", "sigma", "tau", "Z", "mu"),
                        thin = 1,
                        niter = 10000,
                        nburnin = 1000,
                        nchains = 1,
                        summary = T,
                        setSeed = T,
                        WAIC = T)

mcmc.out_quadratic2$summary

write_rds(mcmc.out_quadratic2, "./rds/hw5 quardatic nimble 2.rds")

#| label: fig-bayes-beta2
#| fig-cap: "caption"

ggplot(
  data = data.frame(x = mcmc.out_quadratic$samples[,"beta2"]),
  aes(x = x)
) +
  theme_classic() +
  geom_histogram(bins = 25, fill = "lightgrey", color = "black") +
  geom_vline(xintercept = 0, color = "red", linewidth = 1, linetype = "dashed") +
```

```
  labs(
    x = "Posterior Samples",
    y = " ",
    colour = "Value Type",
    title = paste0("Posterior mean: ", round( mean(mcmc.out_quadratic$samples[,"beta2"]),
  )

observed_data <- data_hw5$Z

sigma_samples <- mcmc.out_quadratic$samples[,"sigma"]

mu_cols <- which(
  substr(
    colnames(mcmc.out_quadratic$samples), 1, nchar("mu")
    ) == "mu"
  )

mu_samples <- mcmc.out_quadratic$samples[,mu_cols]

#### expectation over deviance from samples
###     since we apply a deviance function, it is always -2 times sum [ log  (stuff)  ]


#### This is D_bar
deviances <- rep(NA, nrow(mu_samples))

for(i in 1:length(deviances)){
  deviances[i] <- -2 * sum(dnorm(observed_data, mu_samples[i,], sigma_samples[i], log = TR
}

mean(deviances) -> d_bar

#### this is D(theta bar)
-2 * sum(dnorm(observed_data,
               apply(mu_samples, 2, mean),
               mean(sigma_samples),
               log = TRUE)
         )  -> d_theta_bar

p_D_2 <- d_bar - d_theta_bar
```

```r
DIC_2 <- d_bar + p_D_1

mcmc.out_quadratic2 <- read_rds("./rds/hw5 quardatic nimble 2.rds")

new_X = 100 - stored_mean

expected_value <- mean(mcmc.out_quadratic$samples[,"beta0"]) + mean(mcmc.out_quadratic$sam
sigma_ <- mean(mcmc.out_quadratic$samples[,"sigma"])

predictive_posterior_samples_q <- rnorm(n = 10000, mean = expected_value, sd = sigma_)

#| label: fig-comp-dens-2
#| fig-cap: "The two methods produce different posterior predictive densities"

ggplot(data = data.frame(values = c(mcmc.out_app1$samples[,"Z[13]"],
                                    mcmc.out_quadratic2$samples[,"Z[13]"]),
                         type = c(rep("Linear", length(mcmc.out_app1$samples[,"Z[13]"])),
                                  rep("Quadratic", length(mcmc.out_quadratic2$samples[,"Z[
                                  )
                         ),
       aes(x = values, color = type, group = type)
       ) +
  theme_classic() +
  geom_density() +
  labs(x = 'Estiamted Change in Temprerature')

#| label: fig-comparison
#| fig-cap: "Comaprion of Linear and Non-linear trend models fit to the data"


ggplot(
  data = data.frame(x = data_hw5$X,
                    y = data_hw5$Z),
  aes(x = x, y = y)
) +
  theme_classic() +
  geom_point() +
  geom_smooth(data = data.frame(x = data_hw5$X, y = fitted_val),
              aes(x = x, y = y), color = "red") +

  geom_smooth(data = data.frame(x = data_hw5$X, y = apply(mu_samples, 2, mean)[-13]),
```

```
          aes(x = x, y = y), color = "blue") +
labs(x = "Centered Initial Temperature",
     y = "Change in Temperature")
```