SFS 2023 Short Course – Bayesian Applications in Environmental and Ecological Studies with R and Stan

Song S. Qian

6/3/2023

Hierarchical Structure and Big Data

If we define "big data" as data obtained from multiple sources and representing multiple levels of aggregation, it becomes evident that most of the data we utilize in our work falls into the category of big data. In our context, the era of big data coincides with the era of hierarchical modeling. Failing to appropriately address the hierarchical structure inherent in the data can often result in misleading conclusions when working with big data. ### The US National Lake Assessment data Qian et al (2019) examined various studies that utilized data from the US EPA's National Lakes Assessment program (NLA). The NLA program involved surveying over 3000 lakes across the contiguous 48 states in 2007 and 2012, collecting a wide range of variables to assess the ecological status of the nation's lakes. Each lake was visited a maximum of two times during the survey.

EPA researchers have published numerous papers utilizing the NLA data to establish national nutrient criteria. Typically, they employ lake mean values of relevant variables to establish empirical relationships between ecological response indicators (such as chlorophyll a and microcystin concentrations) and variables indicating nutrient enrichment (such as TP and TN concentrations). However, Qian et al (2019) cautioned that this approach is susceptible to Simpson's paradox.

Simpson's paradox arises when correlations established at one level of aggregation differ significantly from those at a different level of aggregation. In the context of establishing nutrient criteria, Simpson's paradox becomes relevant because the criteria are determined at a national aggregated level (spatially), whereas the resulting criteria must be implemented at individual lakes over time.

This highlights the potential pitfalls of solely relying on aggregated correlations when establishing nutrient criteria. The complex dynamics and interactions within individual lakes can lead to different relationships between ecological responses and nutrient enrichment compared to the overall aggregated level. Therefore, careful consideration is needed to account for the nuances and potential biases introduced by Simpson's paradox when establishing and implementing nutrient criteria at different levels of spatial and temporal aggregation.

From a statistical perspective, it is crucial to accurately model the hierarchical structure inherent in the data. This hierarchical structure refers to the organization of observation values and their corresponding attributes. In a typical dataset, such as one presented in an Excel spreadsheet format, data is arranged in a two-dimensional array. The rows represent individual observations, while the columns represent different variables. In the popular statistical programming language R, data is often organized using data frames, which is the commonly used format for storing and manipulating data.

In this hierarchical structure, variables can be categorized into two main types: measured variables and identification variables. Measured variables typically consist of numerical values that represent the observed measurements, while identification variables are categorical in nature and serve to identify or group the measured variables. For instance, in our dataset, variables such as chla, tp, and tn would be considered measured variables, while the variable id would be an identification variable. The concept of measured and identification variables is explicitly utilized in packages from the tidyverse family, which provide a set of

powerful tools for data manipulation and analysis.

By incorporating the hierarchical structure of the data, we can effectively group the measured variables into parallel units using the identification variable. This hierarchical modeling approach allows us to capture the dependencies and relationships within the data, leading to more accurate and meaningful statistical analyses.

Suppose we only have measurements of chla from these lakes, and the lake identifiers do not provide specific information about each lake. If we wish to determine the average chla values for these lakes, we can approximate the logarithm of chla values, denoted as log_chla, using a normal distribution. To make statistical inferences about chla for lake j, we can employ the following model:

$$\log(chla_{ij}) \sim N(\mu_j, \sigma_j^2)$$

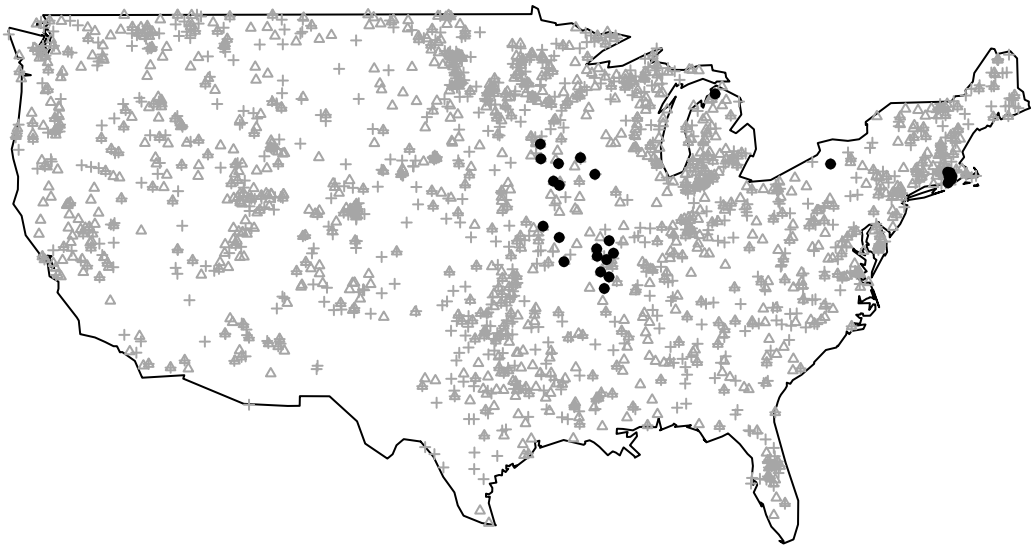
The parameters μ_j and σ_j^2 will be estimated once we have the data. In Bayesian statistics, it is necessary to assign prior distributions to these unknown parameters. In most cases, our primary interest lies in the mean parameters. As per the central limit theorem, it is reasonable to assume a normal distribution as the prior for μ_j:

$$\mu_j \sim N(\theta, \tau^2)$$

Consequently, we need to establish prior distributions for all 27 lakes involved in this problem. However, since we lack specific information about these lakes, we are unable to determine the relative magnitudes of chla values across the lakes. In other words, we cannot assign a higher or lower prior mean to lake 1 compared to lake 2. Therefore, to account for our lack of knowledge regarding the relative magnitudes among the lakes, we must assign a common prior to all 27 lakes. The equation above represents our ignorance: while we acknowledge that the mean chla values for the lakes are likely to differ, we do not possess any information about the specific differences. In the absence of further information, we employ non-informative priors for θ and τ^2. This hierarchical modeling approach is a generalization of Stein's paradox (and the James-Stein estimator) in classical statistics. By imposing this common prior, we observe the shrinkage effect demonstrated in the Everglades example. The lakes are considered exchangeable with respect to their lake-specific μ_j values.

To illustrate this issue, Qian et al. (2019) utilized data from lakes that were included in both the NLA and another extensive lake database known as LAGOS. They aimed to compare how Simpson's paradox can manifest itself in studies examining eutrophication in lakes.

The data: USA-NLA and LAGOS. We pick lakes shared in the two data bases.



We selected lakes from LAGOS with at least 10 observations for this analysis. By comparing the lake-specific models fitted using hierarchical modeling to the common practice of either combining data from all lakes

or fitting a model using lake means, we aim to highlight the significance of accounting for the hierarchical structure of the data. The selected set of 27 lakes in this example each have at least 27 observations.

In our analysis, we employed the typical log-log linear model to predict chlorophyll-a ($chla$) based on total nitrogen (tn), total phosphorous (tp), and their interaction. The decision to include the TP:TN interaction was inspired by Qian (2016), who suggested that the slope of the interaction can indicate a lake's trophic status: a negative (0, positive) interaction slope suggests eutrophic (mesotrophic, oligotrophic) conditions in the lake.

When we have observations for both `tp` and `tn` from these lakes, we can no longer assume ignorance since TP and TN are generally positively correlated with `chla`. However, in modeling the relationship between $chla$ and TP and TN using a log-log linear model:

$$\log(chla_{ij}) = \beta_{0j} + \beta_{1j}\log(TP) + \beta_{2j}\log(TP) + \beta_{3j}\log(TP)\log(TN) + \epsilon_{ij}$$

we can still be uncertain about how the regression coefficients vary among lakes. Hence, we can impose a common prior for these coefficients:

$$\begin{pmatrix}\beta_{0j}\\ \beta_{1j}\\ \beta_{2j}\\ \beta_{3j}\end{pmatrix} \sim MVN\left[\begin{pmatrix}\mu_0\\ \mu_1\\ \mu_2\\ \mu_3\end{pmatrix}, \Sigma\right]$$

Now we say that these lakes are exchangeable with respect to model coefficients.

The R package `lme4` offers efficient algorithms for estimating these parameters using the restricted maximum likelihood method. While these algorithms may not be as effective in estimating the variance parameters, they are fast and often provide satisfactory approximations. We can leverage the capabilities of `lme4` to explore various model forms and determine the most suitable approach for modeling the data. Once we have identified the preferred model form, we can then transition to using Stan for precise quantification and analysis.

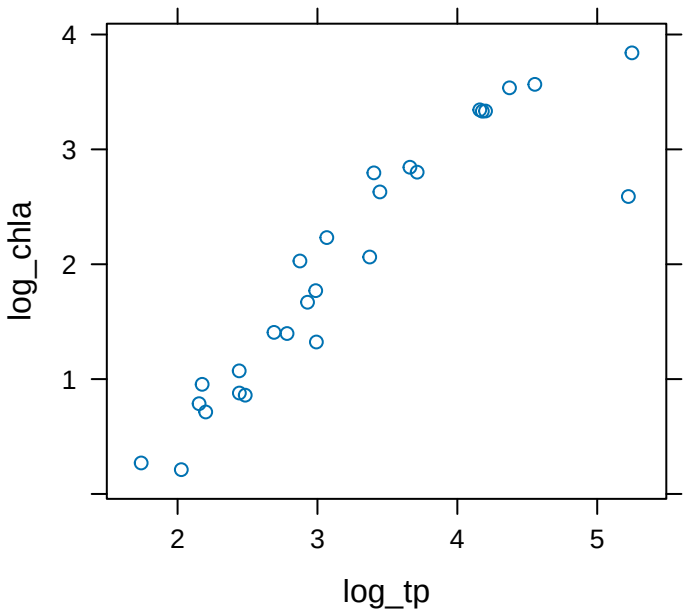
- Comparing different spatial aggregations

```
## fitting hierarchical model for each lake
log_tp_mu <- mean(log(lg_lakes$tp + 0.1), na.rm = T)
log_tn_mu <- mean(log(lg_lakes$tn + 1), na.rm = T)

lg_lakes_cen <- data.frame(log_chla = log(lg_lakes$chla), log_tp_c = log(lg_lakes$tp +
    0.1) - log_tp_mu, log_tn_c = log(lg_lakes$tn + 1) - log_tn_mu, id = lg_lakes$lagoslakeid)
lg_mlm <- lmer(log_chla ~ log_tp_c + log_tn_c + log_tp_c:log_tn_c + (1 + log_tp_c +
    log_tn_c + log_tp_c:log_tn_c | id), data = lg_lakes_cen)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## unable to evaluate scaled gradient
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge: degenerate Hessian with 1 negative eigenvalues
```

```
## Fitting a single linear regression model using lake means US EPA approach
lg_lakes_means <- data.frame(log_chla = tapply(log(lg_lakes$chla), lg_lakes$lagoslakeid,
    mean, na.rm = T), log_tp = tapply(log(lg_lakes$tp + 0.1), lg_lakes$lagoslakeid,
    mean, na.rm = T), log_tn = tapply(log(lg_lakes$tn + 1), lg_lakes$lagoslakeid,
    mean, na.rm = T))
xyplot(log_chla ~ log_tp, data = lg_lakes_means)
```

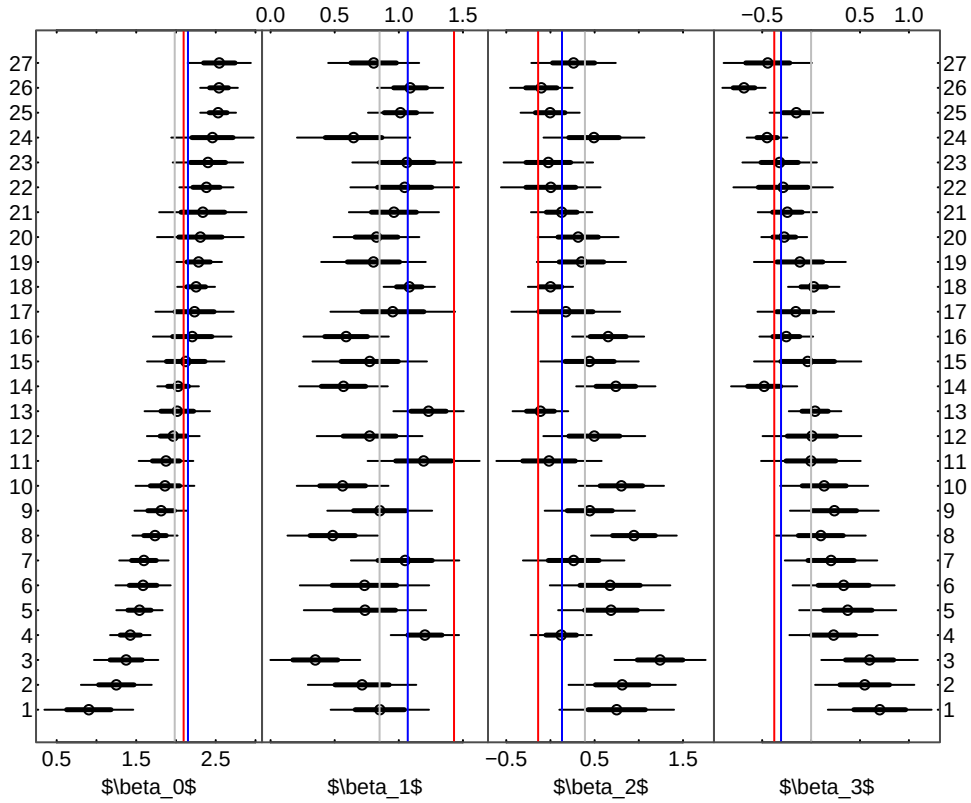



```
lg_lakes_means_lm <- lm(log_chla ~ I(log_tp - log_tp_mu) + I(log_tn - log_tn_mu) +
    I(log_tp - log_tp_mu):I(log_tn - log_tn_mu), data = lg_lakes_means)
lg_mean_lm_coef <- coef(lg_lakes_means_lm)

## fitting using all observations (complete mixing)
lg_lakes_cen <- data.frame(log_chla = log(lg_lakes$chla), log_tp_c = log(lg_lakes$tp +
    0.1) - log_tp_mu, log_tn_c = log(lg_lakes$tn + 1) - log_tn_mu, id = lg_lakes$lagoslakeid)
lg_lakes_lm <- lm(log_chla ~ log_tp_c + log_tn_c + log_tp_c:log_tn_c, data = lg_lakes_cen)
lg_lm_coef <- coef(lg_lakes_lm)
```

Now we compare the estimated coefficients:

```
line.plots <- function(est, se, yaxis = NULL, hline = 0, HL = T, oo = NULL, Outer = F,
    xloc = 1, yaxisLab = NULL, ...) {
    n <- length(est)
    if (!is.null(oo)) {
        est <- est[oo]
        se <- se[oo]
    }
    if (n != length(se))
        stop("lengths not match")
    plot(1:n, 1:n, xlim = range(c(est + 2 * se, est - 2 * se)), ylim = c(0.75, n +
        0.25), type = "n", axes = F, ...)
    axis(xloc)
    axis(side = c(1, 3)[c(1, 3) != xloc], labels = F)
    if (!is.null(yaxis))
        axis(yaxis, at = 1:n, labels = yaxisLab, las = 1, outer = Outer)
    segments(y0 = 1:n, y1 = 1:n, x0 = est - 2 * se, x1 = est + 2 * se)
    segments(y0 = 1:n, y1 = 1:n, x0 = est - 1 * se, x1 = est + 1 * se, lwd = 2.5)
    points(est, 1:n)
    if (HL)
        abline(v = hline, col = "gray")
    invisible()
}
```

```
## all lakes, by lake
est <- t(fixef(lg_mlm) + t(as.matrix(ranef(lg_mlm)[["id"]])))
se <- sqrt(t(se.fixef(lg_mlm)^2 + t(as.matrix(se.ranef(lg_mlm)[["id"]]))^2))
oo <- order(est[, 1])

par(mfrow = c(1, 4), mgp = c(1.25, 0.125, 0), oma = c(0, 3, 0, 3), tck = 0.01, las = 1,
    mar = c(3, 0, 3, 0))
line.plots(est[oo, 1], se[oo, 1], yaxis = 2, hline = fixef(lg_mlm)[1], yaxisLab = 1:27,
    xlab = "$\\beta_0$")
abline(v = lg_mean_lm_coef[1], col = "red")
abline(v = lg_lm_coef[1], col = "blue")
box(col = grey(0.3))
line.plots(est[oo, 2], se[oo, 2], yaxisLab = 1:27, hline = fixef(lg_mlm)[2], xloc = 3,
    xlab = "$\\beta_1$")
abline(v = lg_mean_lm_coef[2], col = "red")
abline(v = lg_lm_coef[2], col = "blue")
box(col = grey(0.3))
line.plots(est[oo, 3], se[oo, 3], yaxisLab = 1:27, hline = fixef(lg_mlm)[3], xlab = "$\\beta_2$")
abline(v = lg_mean_lm_coef[3], col = "red")
abline(v = lg_lm_coef[3], col = "blue")
box(col = grey(0.3))
line.plots(est[oo, 4], se[oo, 4], yaxisLab = 1:27, xlab = "$\\beta_3$", yaxis = 4,
    xloc = 3)
abline(v = lg_mean_lm_coef[4], col = "red")
abline(v = lg_lm_coef[4], col = "blue")
box(col = grey(0.3))
```



- Comparing teporal aggregations Examining the temporal scale aggregation of the three lake with long

time series

```
lg_lakes_long <- sharedLakes$lagoslakeid[sharedLakes$lg_n > 100]
lg_lakes_long <- lg_nutr[is.element(lg_nutr$lagoslakeid, lg_lakes_long), ]
lg_lakes_long$date <- as.Date(lg_lakes_long$sampledate, format = "%m/%d/%Y")


lake1 <- lg_lakes_long[lg_lakes_long$lagoslakeid == unique(lg_lakes_long$lagoslakeid)[1],
    ]
lake1$log_chla <- log(lake1$chla)
lake1$log_tp_c <- log(lake1$tp + 0.1) - log_tp_mu
lake1$log_tn_c <- log(lake1$tn + 1) - log_tn_mu

lake2 <- lg_lakes_long[lg_lakes_long$lagoslakeid == unique(lg_lakes_long$lagoslakeid)[2],
    ]
lake2$log_chla <- log(lake2$chla)
lake2$log_tp_c <- log(lake2$tp + 0.1) - log_tp_mu
lake2$log_tn_c <- log(lake2$tn + 1) - log_tn_mu

lake3 <- lg_lakes_long[lg_lakes_long$lagoslakeid == unique(lg_lakes_long$lagoslakeid)[3],
    ]
lake3$log_chla <- log(lake3$chla)
lake3$log_tp_c <- log(lake3$tp + 0.1) - log_tp_mu
lake3$log_tn_c <- log(lake3$tn + 1) - log_tn_mu

lake1_mlm <- lmer(log_chla ~ log_tp_c + log_tn_c + log_tp_c:log_tn_c + (1 + log_tp_c +
    log_tn_c + log_tp_c:log_tn_c | sampleyear), data = lake1)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
lake2_mlm <- lmer(log_chla ~ log_tp_c + log_tn_c + log_tp_c:log_tn_c + (1 + log_tp_c +
    log_tn_c + log_tp_c:log_tn_c | sampleyear), data = lake2)
```

```
## boundary (singular) fit: see help('isSingular')
```

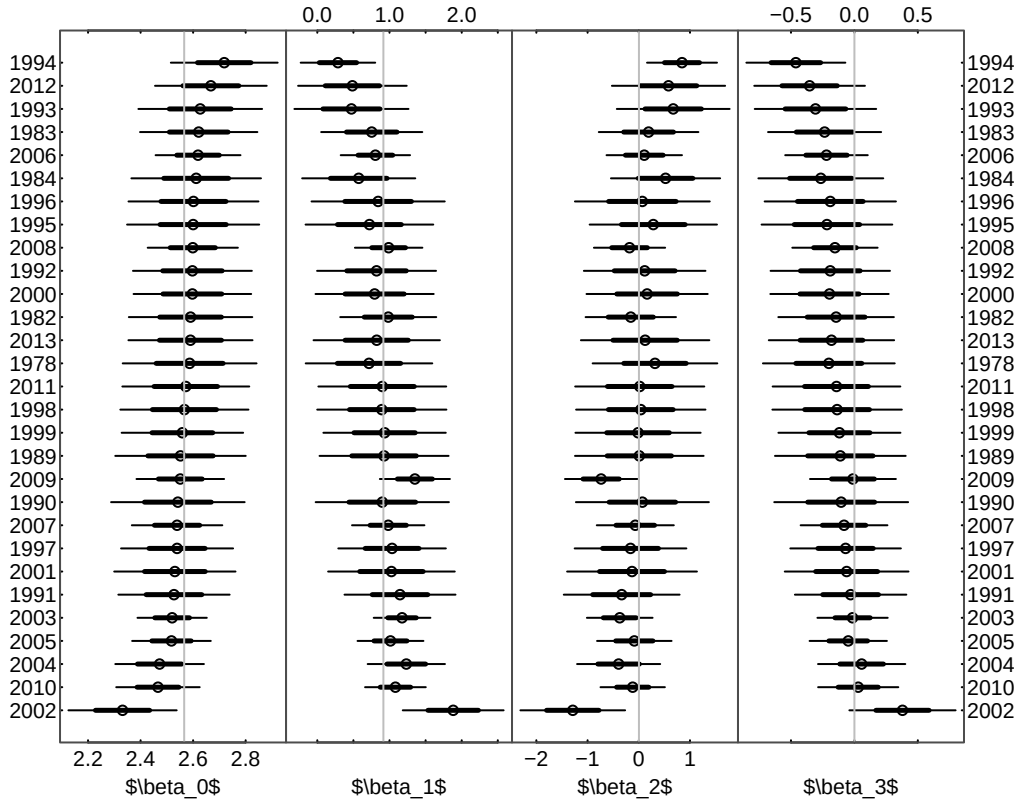
```
lake3_mlm <- lmer(log_chla ~ log_tp_c + log_tn_c + log_tp_c:log_tn_c + (1 + log_tp_c +
    log_tn_c + log_tp_c:log_tn_c | sampleyear), data = lake3)
```

```
## boundary (singular) fit: see help('isSingular')
```

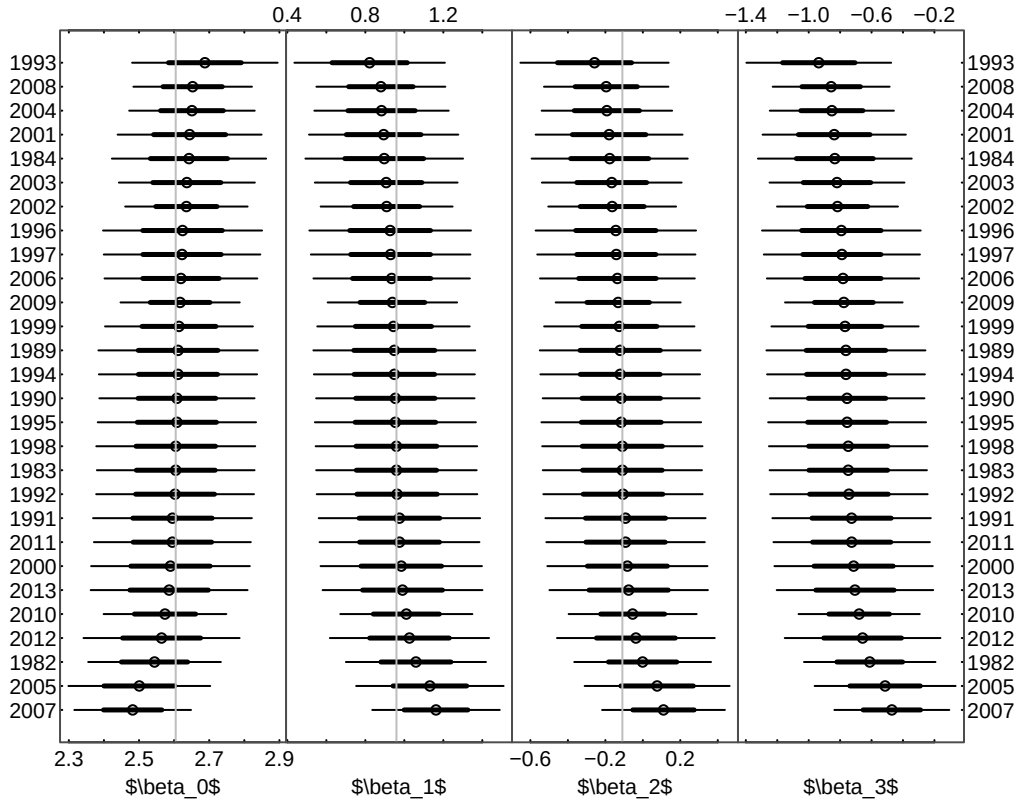
Graphical comparisons

```
## lake 1 by year
est <- t(fixef(lake1_mlm) + t(as.matrix(ranef(lake1_mlm)[["sampleyear"]])))
se <- sqrt(t(se.fixef(lake1_mlm)^2 + t(as.matrix(se.ranef(lake1_mlm)[["sampleyear"]]))^2))
oo <- order(est[, 1])
ylb <- row.names(ranef(lake1_mlm)[["sampleyear"]])

par(mfrow = c(1, 4), mgp = c(1.25, 0.125, 0), oma = c(0, 3, 0, 3), tck = 0.01, las = 1,
    mar = c(3, 0, 3, 0))
line.plots(est[oo, 1], se[oo, 1], yaxisLab = ylb[oo], yaxis = 2, hline = fixef(lake1_mlm)[1],
    xlab = "$\\beta_0$")
box(col = grey(0.3))
line.plots(est[oo, 2], se[oo, 2], yaxisLab = ylb[oo], xlab = "$\\beta_1$", hline = fixef(lake1_mlm)[2],
    xloc = 3)
box(col = grey(0.3))
line.plots(est[oo, 3], se[oo, 3], yaxisLab = ylb[oo], xlab = "$\\beta_2$", hline = fixef(lake1_mlm)[3])
```

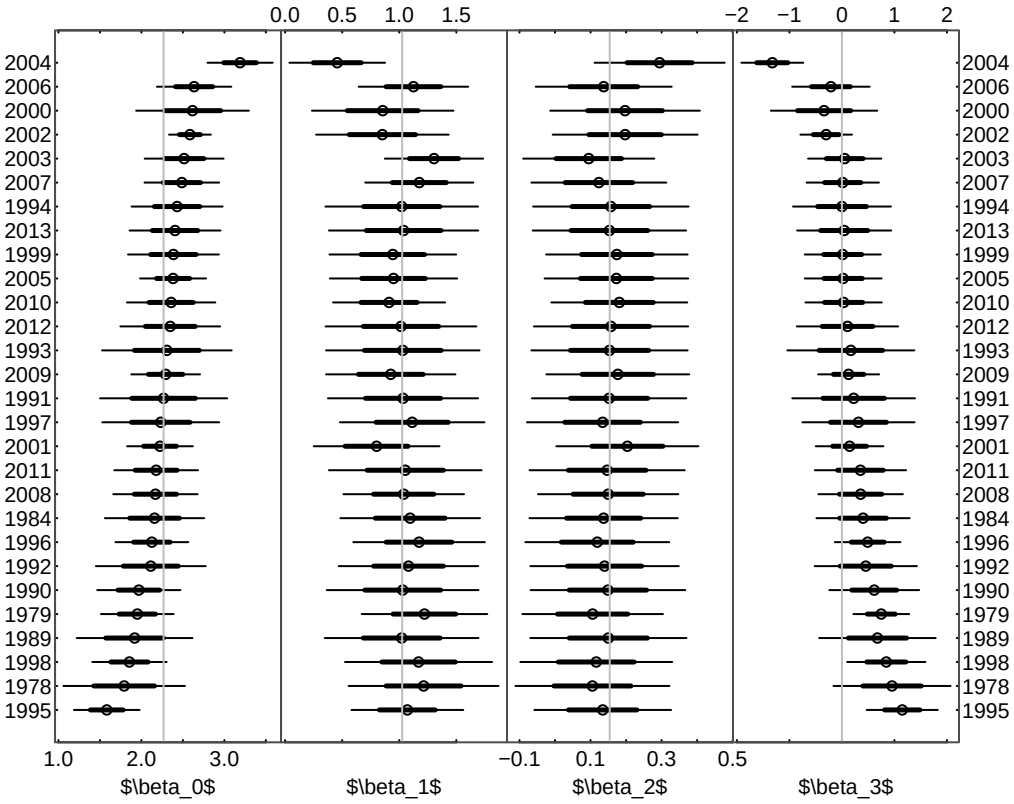
```
box(col = grey(0.3))
line.plots(est[oo, 4], se[oo, 4], yaxisLab = ylb[oo], xlab = "$\\beta_3$", yaxis = 4,
    xloc = 3)
box(col = grey(0.3))
```




```
## lake 2 by year
est <- t(fixef(lake2_mlm) + t(as.matrix(ranef(lake2_mlm)[["sampleyear"]])))
se <- sqrt(t(se.fixef(lake2_mlm)^2 + t(as.matrix(se.ranef(lake2_mlm)[["sampleyear"]]))^2))
oo <- order(est[, 1])
ylb <- row.names(ranef(lake2_mlm)[["sampleyear"]])

par(mfrow = c(1, 4), mgp = c(1.25, 0.125, 0), oma = c(0, 3, 0, 3), tck = 0.01, las = 1,
    mar = c(3, 0, 3, 0))
line.plots(est[oo, 1], se[oo, 1], yaxisLab = ylb[oo], xlab = "$\\beta_0$", yaxis = 2,
    hline = fixef(lake2_mlm)[1])
box(col = grey(0.3))
line.plots(est[oo, 2], se[oo, 2], yaxisLab = ylb[oo], xlab = "$\\beta_1$", hline = fixef(lake2_mlm)[2],
    xloc = 3)
box(col = grey(0.3))
line.plots(est[oo, 3], se[oo, 3], yaxisLab = ylb[oo], xlab = "$\\beta_2$", hline = fixef(lake2_mlm)[3])
box(col = grey(0.3))
line.plots(est[oo, 4], se[oo, 4], yaxisLab = ylb[oo], xlab = "$\\beta_3$", yaxis = 4,
    xloc = 3)
box(col = grey(0.3))
```




```
## lake 3 by year
est <- t(fixef(lake3_mlm) + t(as.matrix(ranef(lake3_mlm)[["sampleyear"]])))
se <- sqrt(t(se.fixef(lake3_mlm)^2 + t(as.matrix(se.ranef(lake3_mlm)[["sampleyear"]]))^2))
oo <- order(est[, 1])
ylb <- row.names(ranef(lake3_mlm)[["sampleyear"]])

par(mfrow = c(1, 4), mgp = c(1.25, 0.125, 0), oma = c(0, 3, 0, 3), tck = 0.01, las = 1,
    mar = c(3, 0, 3, 0))
line.plots(est[oo, 1], se[oo, 1], yaxisLab = ylb[oo], xlab = "$\\beta_0$", yaxis = 2,
    hline = fixef(lake3_mlm)[1])
box(col = grey(0.3))
line.plots(est[oo, 2], se[oo, 2], yaxisLab = ylb[oo], xlab = "$\\beta_1$", hline = fixef(lake3_mlm)[2],
    xloc = 3)
box(col = grey(0.3))
line.plots(est[oo, 3], se[oo, 3], yaxisLab = ylb[oo], xlab = "$\\beta_2$", hline = fixef(lake3_mlm)[3])
box(col = grey(0.3))
line.plots(est[oo, 4], se[oo, 4], yaxisLab = ylb[oo], xlab = "$\\beta_3$", yaxis = 4,
    xloc = 3)
box(col = grey(0.3))
```



See Section 6.4.3 of Qian et al (2022) for details on programming multilevel models in Stan.