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

Song S. Qian

6/3/2023

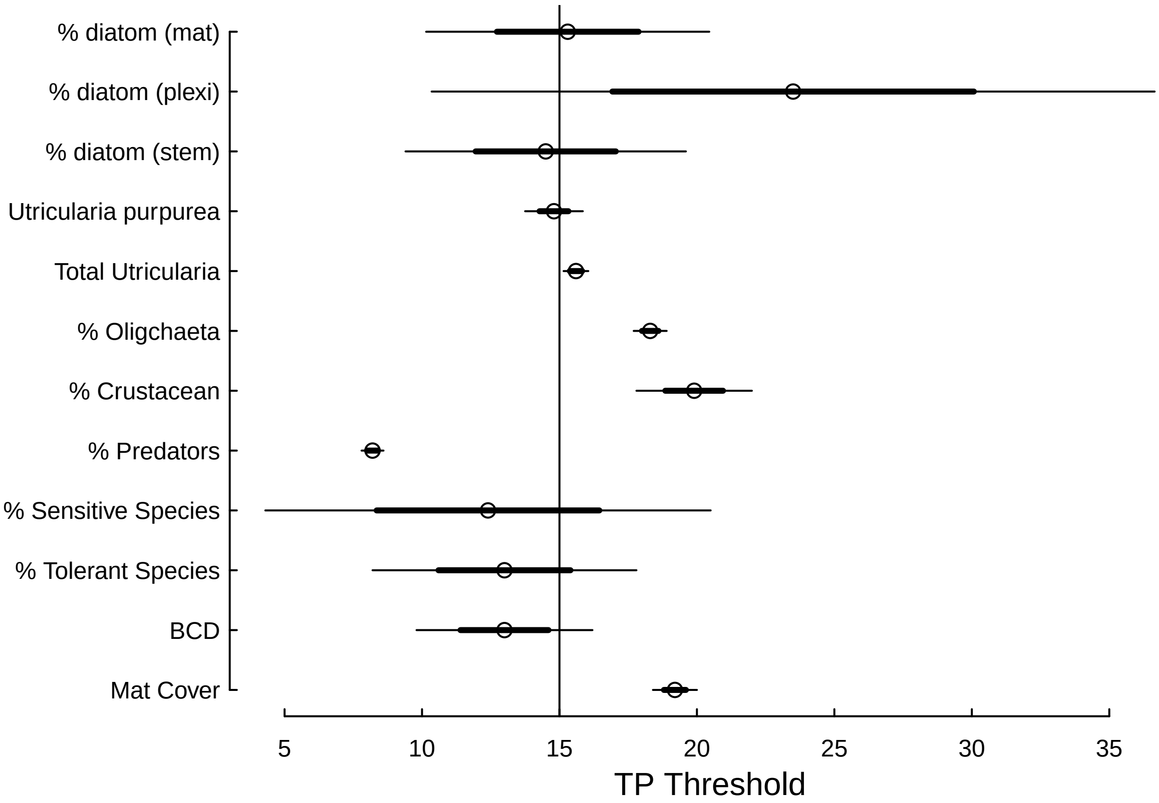
Hierarchical Models

A simple start – setting environmental standard in the Everglades

Richardson et al. (2007) conducted a mesocosm study in the Everglades of South Florida to investigate the response of wetland ecosystems to elevated phosphorus (P) input. They created a phosphorus gradient using artificial flumes and observed changes in the mesocosm ecosystem. The researchers determined the thresholds of total phosphorus (TP) concentrations that would lead to significant changes in algae, macroinvertebrates, and macrophytes communities. To quantify these thresholds, they utilized 12 biological indicators that represented how quickly the indicators would respond to changes in TP concentrations. The study reported the means of the 12 thresholds along with their corresponding 95% confidence intervals. In our analysis, we employed the 95% confidence intervals (typically calculated as the mean plus/minus 2 standard errors) to estimate the standard deviation of the mean thresholds.

```
y.hat <- c(19.2,  13,  13, 12.4, 8.2, 19.9, 18.3,
           15.6, 14.8, 14.5, 23.5, 15.3)
sigma.hat <- c( 1.6, 6.4, 9.6, 16.2, 0.8,  4.2,  1.2,
                0.9,  2.1, 10.2, 26.3, 10.3)/4

metrics <- c("Mat Cover","BCD","% Tolerant Species","% Sensitive Species",
             "% Predators","% Crustacean","% Oligchaeta","Total Utricularia",
             "Utricularia purpurea","% diatom (stem)","% diatom (plexi)",
             "% diatom (mat)")
metricsTEX <- c("Mat Cover","BCD","\\% Tol Sp",
                "\\% Sen Sp", "\\% Pred","\\% Crust",
                "\\% Oligchaeta","Tot Utr","Utr P.",
                "\\% diatom (stem)","\\% diatom (plexi)",
                "\\% diatom (mat)")

par(mar=c(3, 7, 1, 0.5), mgp=c(1.25,0.25,0),tck=0.01)
plot(range(y.hat-2*sigma.hat, y.hat+2*sigma.hat),
     c(1,length(y.hat)), type="n",
     xlab="TP Threshold", ylab=" ", axes=F)
axis(1, cex.axis=0.75)
axis(2, at=1:length(y.hat), labels=metrics, las=1, cex.axis=0.75)
segments(x0=y.hat+sigma.hat, x1=y.hat-sigma.hat,
         y0=1:length(y.hat), y1=1:length(y.hat), lwd=3)#,
#         col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4))
segments(x0=y.hat+2*sigma.hat, x1=y.hat-2*sigma.hat,
         y0=1:length(y.hat), y1=1:length(y.hat), lwd=1)#,
#         col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4))
points(x=y.hat, y=1:length(y.hat))
```

```
abline(v=15)
```



Richardson et al. (2007) recommended that the total phosphorus (TP) concentration standard should be 15 μg/L, which is close to the average of the 12 means and the mean of Utricularia purpurea, a keystone species in the Everglades wetland ecosystem. However, setting a TP standard solely based on one species is less convincing due to the scientifically vague legal requirement of protecting the natural balance of flora and fauna in the Everglades. Although the value is close to the average of all examined metrics, each metric represents a specific aspect of the ecosystem and cannot alone describe the natural balance at the ecosystem level.

Each metric represents a specific aspect of the ecosystem (individual species or species groups). These species-specific indicators by themselves cannot describe the natural balance at the ecosystem level. Suppose that there are a total of n indicators to represent the Everglades wetland ecosystem and we have estimates of thresholds of these indicators ($\phi_j, j = 1, \cdots, n$). Although each individual threshold cannot adequately represent the natural imbalance, the distribution of all thresholds should provide a quantitative summary of how TP concentration levels would affect the ecosystem as a whole. Because the 12 metrics were carefully selected to represent the Everglades wetland ecosystems, ecosystem-level threshold distribution can be estimated from these individual-level thresholds. Instead of using the average of the estimated change points of the 12 metrics, we integrate these estimates using a hierarchical model to properly represent the estimation uncertainty we have about these estimates.

The available data are the estimated change point $\hat{\phi}_j$ and its standard deviation $\hat{\sigma}_j$. These two numbers form an indicator-level model:

$$\hat{\phi}_j \sim N(\theta_j, \hat{\sigma}_j^2).$$

The estimated mean and standard deviation, $\hat{\phi}_j$ and $\hat{\sigma}_j$, summarize the information in the data. In the absence of additional information to determine the relative magnitude of thresholds for different metrics, we assume that the θ_j's can be modeled as follows:

$$\theta_j \sim N(\mu, \tau^2),$$

which represents a common prior distribution for θ_j. This hierarchical model extends the concept of Stein's paradox from the 1960s. With 12 thresholds to estimate simultaneously, Stein's paradox informs us that estimating them individually is mathematically inadmissible. By shrinking the individually estimated means towards the overall mean, we can improve the accuracy of the overall estimation. Hierarchical modeling has shown its value in applied fields since Stein's paradox, particularly in addressing environmental and ecological data analysis problems that involve variables representing different levels of spatial, temporal, and organizational aggregations. Failing to properly address data hierarchy can lead to Simpson's paradox.

The common prior distribution used in the Everglades problem reflects two key considerations: (1) our understanding that the θ_j values are likely to differ across different metrics, and (2) our lack of understanding regarding the specific differences between the θ_j values. The variance parameter τ^2 represents the between-metric variance. In this context, we expanded the meaning of τ^2 to encompass the variance among all possible metric means, not just the 12 metrics represented in the available data. This hierarchical model connects all metrics together through the common prior distribution $N(\mu, \tau^2)$.

Since we have no prior knowledge about the values of μ and τ^2, we will utilize Stan default weakly informative priors. From the perspective of modeling individual metrics, each time we model a metric mean (i.e., $\hat{\phi}_j \sim N(\theta_j, \hat{\sigma}_j^2)$), we employ Bayesian estimation and assign a prior distribution to the unknown metric mean θ_j. In this case, the prior distribution parameters are estimated based on data from other metrics. If we have an additional metric, the hierarchical model for the 13th metric can be seen as a Bayesian estimation utilizing an informative prior. This informative prior is derived from other similar quantities. This interpretation leads to the concept of treating a prior as the distribution of similar quantities, known mathematically as exchangeable units. Studies involving similar quantities from different contexts, such as eutrophication studies in various lakes, are often referred to as parallel studies. Exchangeable units can pertain to spatial aspects (e.g., different lakes, distinct eco-regions when studying climate change impacts), temporal aspects (observations from the same location over different seasons or years), and, as illustrated in this example, organizational aspects (different metrics representing various aspects of an ecosystem). I believe that any environmental and ecological data analysis problem can be approached as a hierarchical modeling problem, considering the inherent hierarchical structure of the data.

Returning to the Everglades example, we can observe the well-known shrinkage effect of hierarchical modeling, which is responsible for enhancing overall estimation accuracy. Let's provide an intuitive explanation of why shrinking estimates towards the overall mean leads to improved accuracy. When we say that an estimate has an error, we mean that the estimated value is either too high or too low. However, when we have only one parameter to estimate, we have no basis to believe that the estimate is biased towards being too high or too low. Thus, an unbiased estimator is preferred, as, on average, it tends to be correct.

When we have estimates of the same parameter from multiple exchangeable units, the overall mean of these estimates serves as a reasonable reference to gauge whether an individual estimate is likely to be too high or too low. Consequently, shrinking these estimates towards the overall mean is more likely to improve their accuracy.

One common computational issue in hierarchical models arises from the potential strong correlation among the multiple means (i.e., θ_j values), especially when there are only a small number of exchangeable units. This correlation often stems from the difficulty in accurately quantifying the hyperparameters (μ, σ^2). The phenomenon known as Neal's funnel is a typical manifestation of this correlation.

As we discussed earlier, we can address this challenge through reparameterization of the model. Instead of directly sampling θ_j as random variables, we can utilize the relationship between a normally distributed random variable with mean μ and standard deviation τ, and a standard normal random variable $z \sim N(0, 1)$:

$$\theta_j = \mu + \tau \times z_j.$$

By defining θ_j as a transformed variable, we can improve the Stan model's computational performance by avoiding directly sampling from θ_j:

```
everg_stan <- "
data {
  int<lower=0> J; // number of schools
  real y[J]; // estimated treatment effects
  real<lower=0> sigma[J]; // s.e. of effect estimates
}
parameters {
  real mu;
  real<lower=0> tau;
  real eta[J];
}
transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] = mu + tau * eta[j];
}
model {
  eta ~ normal(0, 1);
  y ~ normal(theta, sigma);
}
"

fit1 <- stan_model(model_code = everg_stan)
```

As usual, we first organize input data and initial values

```
everg_in <- function(y=y.hat, sig=sigma.hat, n.chains=nchains){
  J <- length(y)
  data <- list(y=y, sigma=sig, J=J)
  inits<-list()
  for (i in 1:n.chains)
    inits[[i]] <- list(eta=rnorm(J), mu=rnorm(1), tau=runif(1))
  pars <- c("theta", "mu", "eta", "tau")
  return(list(data=data, inits=inits, pars=pars, chains=n.chains))
}

input.to.stan <- everg_in()
fit2keep <- sampling(fit1, data=input.to.stan$data,
                     init=input.to.stan$inits,
                     pars=input.to.stan$pars,
                     iter=niters,thin=nthin,
                     chains=input.to.stan$chains,
                     control=list(max_treedepth=25))
```

```
## Warning: There were 3 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

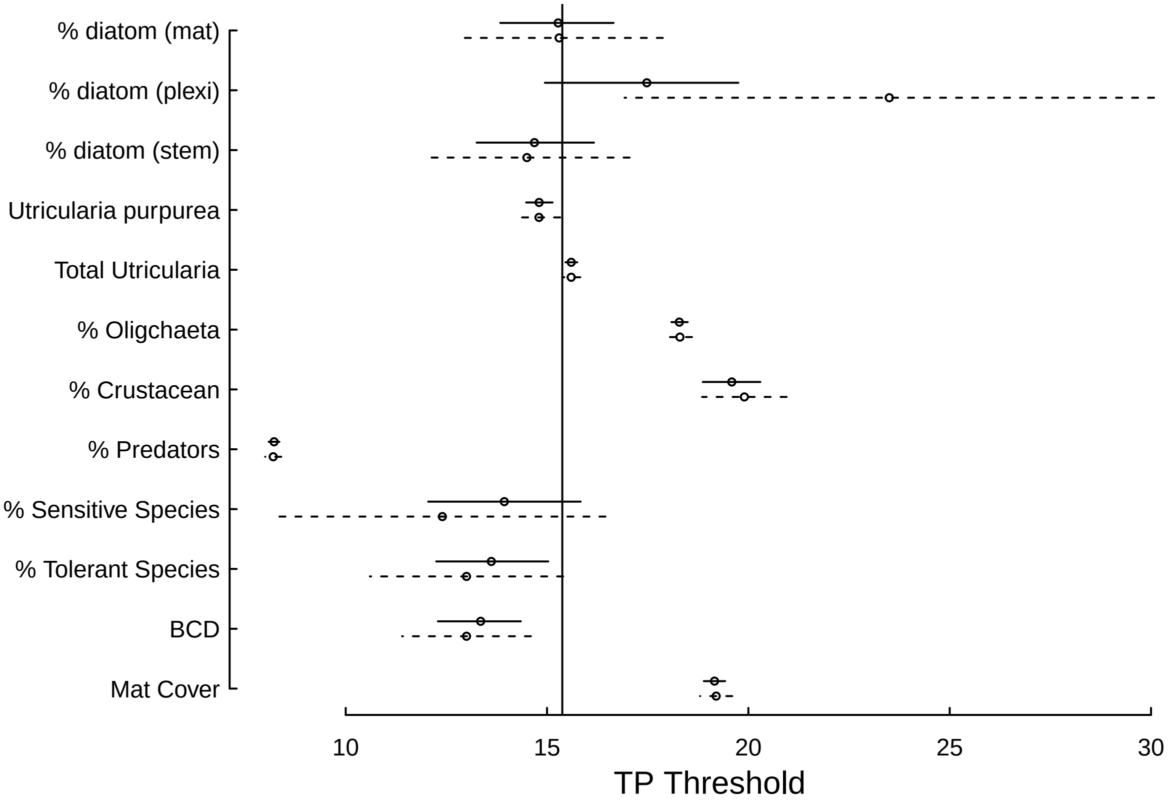
```
print(fit2keep)
```

```
## Inference for Stan model: anon_model.
## 8 chains, each with iter=5000; warmup=2500; thin=8;
## post-warmup draws per chain=313, total post-warmup draws=2504.
##
##             mean se_mean   sd   2.5%    25%   50%   75% 97.5% n_eff Rhat
## theta[1]  19.16    0.01 0.40  18.40  18.89 19.17 19.43 19.92  2507    1
## theta[2]  13.36    0.03 1.51  10.34  12.30 13.39 14.36 16.33  2588    1
## theta[3]  13.63    0.04 2.06   9.58  12.26 13.63 15.04 17.55  2642    1
## theta[4]  13.92    0.06 2.87   8.03  12.06 13.96 15.80 19.51  2667    1
## theta[5]   8.22    0.00 0.20   7.83   8.09  8.22  8.35  8.62  2402    1
## theta[6]  19.59    0.02 1.03  17.64  18.87 19.58 20.31 21.62  2605    1
## theta[7]  18.29    0.01 0.30  17.70  18.09 18.29 18.50 18.87  2500    1
## theta[8]  15.60    0.00 0.22  15.16  15.46 15.60 15.75 16.04  2375    1
## theta[9]  14.80    0.01 0.51  13.80  14.48 14.80 15.14 15.85  2624    1
## theta[10] 14.70    0.04 2.15  10.49  13.26 14.76 16.20 18.73  2499    1
## theta[11] 17.44    0.07 3.72  10.23  14.95 17.21 19.74 25.12  2512    1
## theta[12] 15.26    0.04 2.14  11.17  13.84 15.23 16.64 19.42  2471    1
## mu        15.37    0.03 1.41  12.68  14.49 15.37 16.23 18.13  2134    1
## eta[1]     1.00    0.01 0.43   0.19   0.71  0.99  1.29  1.86  2178    1
## eta[2]    -0.51    0.01 0.50  -1.48  -0.84 -0.50 -0.18  0.44  2429    1
## eta[3]    -0.43    0.01 0.58  -1.58  -0.83 -0.43 -0.03  0.70  2562    1
## eta[4]    -0.36    0.01 0.73  -1.88  -0.82 -0.35  0.13  1.04  2727    1
## eta[5]    -1.87    0.01 0.57  -3.06  -2.26 -1.84 -1.46 -0.83  2239    1
## eta[6]     1.11    0.01 0.49   0.22   0.76  1.10  1.42  2.11  2284    1
## eta[7]     0.77    0.01 0.40   0.02   0.50  0.76  1.05  1.58  2251    1
## eta[8]     0.07    0.01 0.34  -0.60  -0.16  0.07  0.30  0.72  2406    1
## eta[9]    -0.14    0.01 0.35  -0.83  -0.37 -0.14  0.09  0.58  2513    1
## eta[10]   -0.17    0.01 0.59  -1.32  -0.56 -0.17  0.23  0.99  2449    1
## eta[11]    0.49    0.02 0.87  -1.28  -0.11  0.48  1.09  2.21  2529    1
## eta[12]   -0.02    0.01 0.59  -1.19  -0.41 -0.03  0.35  1.15  2525    1
## tau        4.08    0.03 1.16   2.46   3.25  3.87  4.66  7.04  1969    1
## lp__      -9.59    0.07 3.39 -16.86 -11.76 -9.20 -7.14 -4.04  2331    1
##
## Samples were drawn using NUTS(diag_e) at Tue May 23 10:12:48 2023.
## 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).
```

Now processing Stan results

```
everg_fit1 <- rvsims(as.matrix(
    as.data.frame(rstan::extract(fit2keep, permuted=T))))

## shrinkage effect
everg_theta <- rvsims(as.matrix(as.data.frame(rstan::extract(fit2keep,
                                                             permuted=T,
                                                             pars="theta"))))
everg_mu <- rvsims(as.matrix(as.data.frame(rstan::extract(fit2keep,
                                                          permuted=T,
                                                          pars="mu"))))
everg_tau <- rvsims(as.matrix(as.data.frame(rstan::extract(fit2keep,
                                                           permuted=T,
                                                           pars="tau"))))
theta <- summary(everg_theta)
mu <- summary(everg_mu)
tau <- summary(everg_tau)

par(mar=c(3, 7, 1, 0.5), mgp=c(1.25,0.25,0),tck=0.01)
plot(range(y.hat-1*sigma.hat, y.hat+1*sigma.hat),
     c(1,length(y.hat)), type="n",
     xlab="TP Threshold", ylab=" ", axes=F)
```

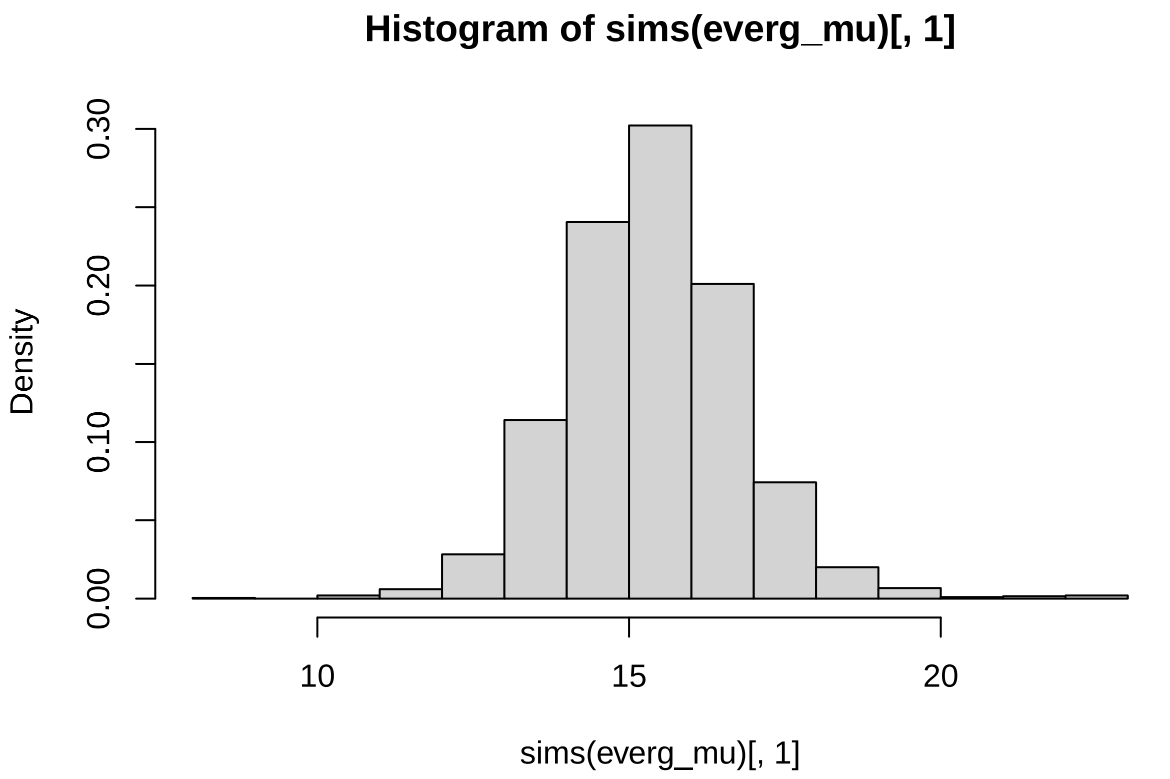
```
axis(1, cex.axis=0.75)
axis(2, at=seq(1,length(y.hat)), labels=metrics, las=1, cex.axis=0.75)
segments(x0=y.hat+sigma.hat, x1=y.hat-sigma.hat,
         y0=seq(1,length(y.hat))-0.125,
         y1=seq(1,length(y.hat))-0.125,
         lwd=1, lty=2)
## col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4),
segments(x0=theta$"25%", x1=theta$"75%",
         y0=(seq(1,length(y.hat)))+0.125,
         y1=(seq(1,length(y.hat)))+0.125)
##         col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4))
points(x=y.hat, y=seq(1,length(y.hat))-0.125, cex=0.5)
##       col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4),
points(x=theta$mean, y=0.125+(seq(1,length(y.hat))), cex=0.5)
##       pch=16,col=c(1,1, 2,2,2,2,2, 3,3, 4, 4,4))
abline(v=mu$mean)
```



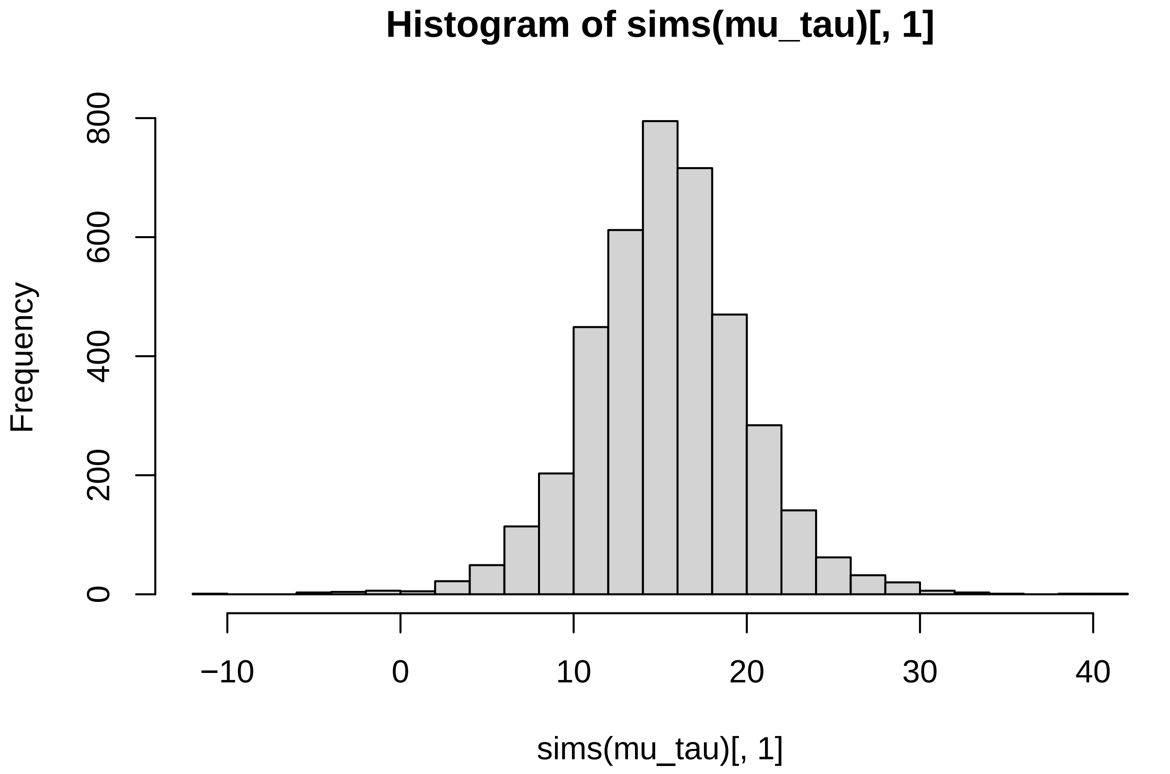
How should we determine the TP concentration standard?

Determining the TP (total phosphorus) concentration standard is primarily an ecological and environmental management decision. From a statistical perspective, the question is whether we should derive the standard based on the overall mean (μ) or consider the distribution of all metrics: between the posterior distribution of μ and the hyper-distribution.

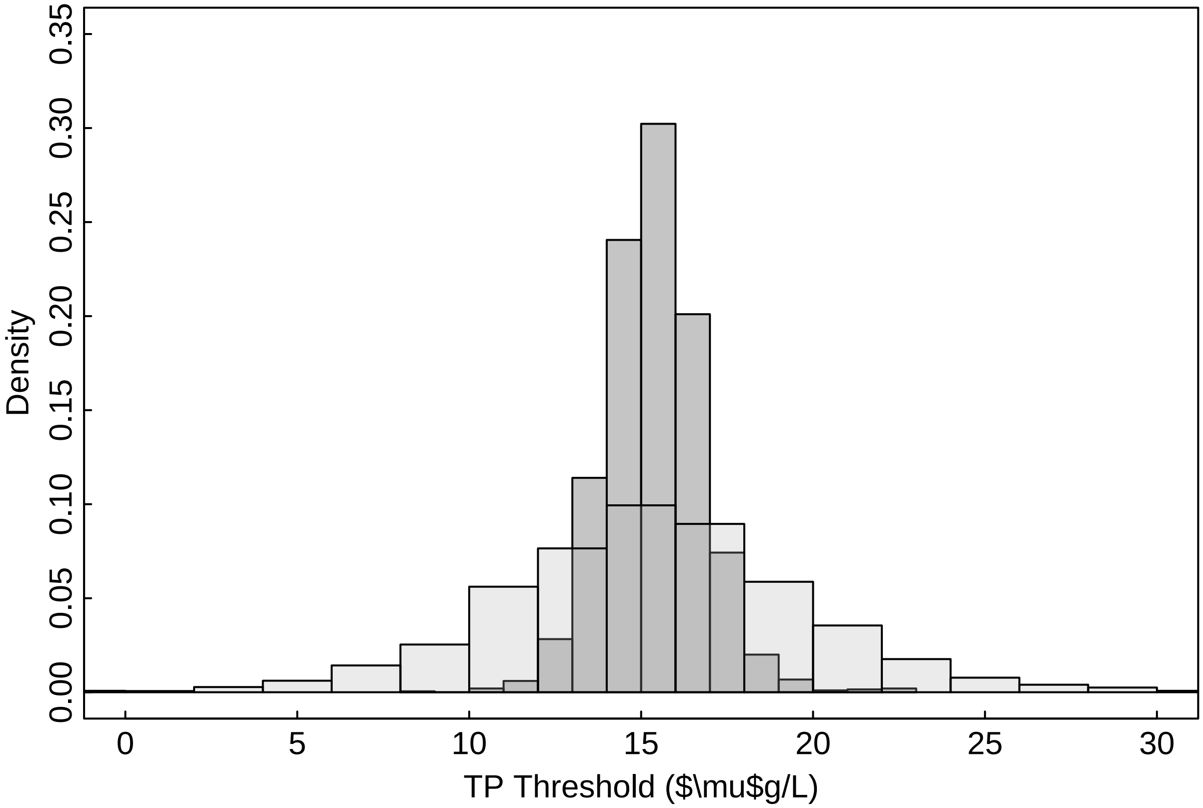
```
## mu versus N(mu, tau)
mu_tau <- rvnorm(1, everg_mu, everg_tau)
p1 <- hist(sims(everg_mu)[,1], freq=F)
```



```
p2 <- hist(sims(mu_tau)[,1], nclass=35)
```



```
par(mar=c(3, 3, 1, 0.5), mgp=c(1.25,0.125,0), tck=0.01)
plot(p1, col=rgb(0.1,0.1,.1,1/4),
     xlim=c(0,30), ylim=c(0,0.35), freq=F,
     xlab="TP Threshold ($\\mu$g/L)", main="")  # first histogram
plot(p2, col=rgb(.7,.7,.7,1/4),
     xlim=c(0,30), ylim=c(0,0.35), freq=F, add=T)  # second
box()
```



```
c(quantile(sims(everg_mu), prob=0.05), quantile(sims(mu_tau), prob=0.05))
```

```
##        5%        5%
## 13.189617  7.963412
```

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:

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

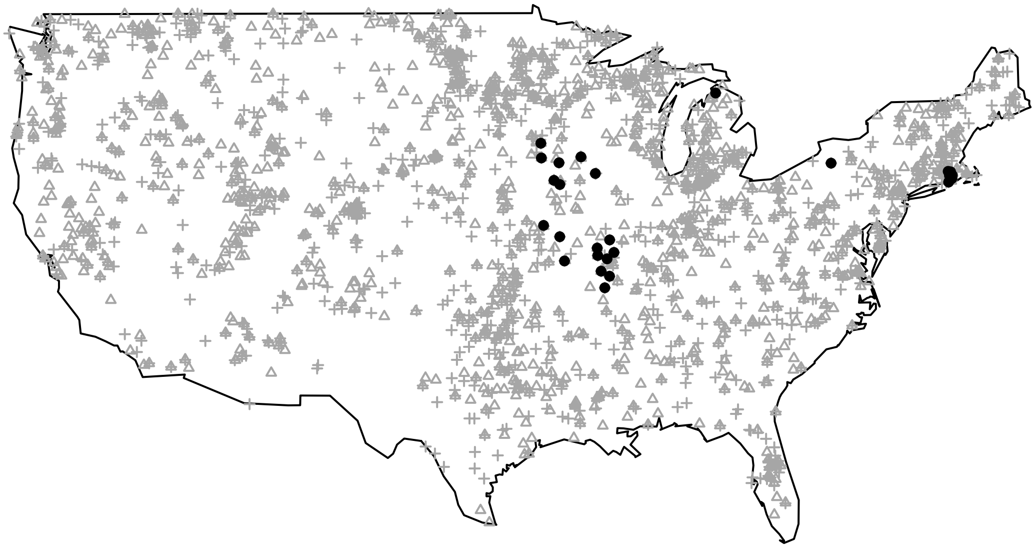
The parameters μ_j and σ_j^2 will be estimated once we have the data. In Bayesian statistics, it is necessary to assign prior distributions to these unknown parameters. In most cases, our primary interest lies in the mean parameters. As per the central limit theorem, it is reasonable to assume a normal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

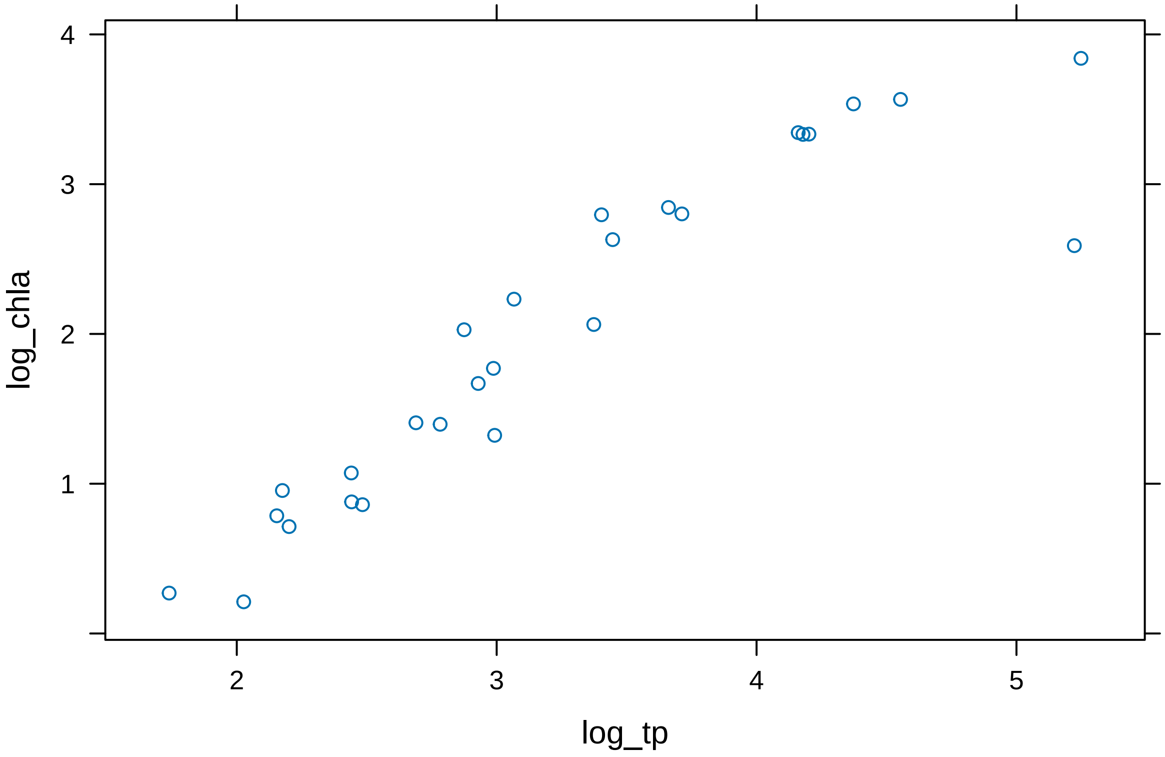
```
## fitting hierarchical model for each lake
log_tp_mu <- mean(log(lg_lakes$tp+0.1), na.rm=T)
```

```
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)
```

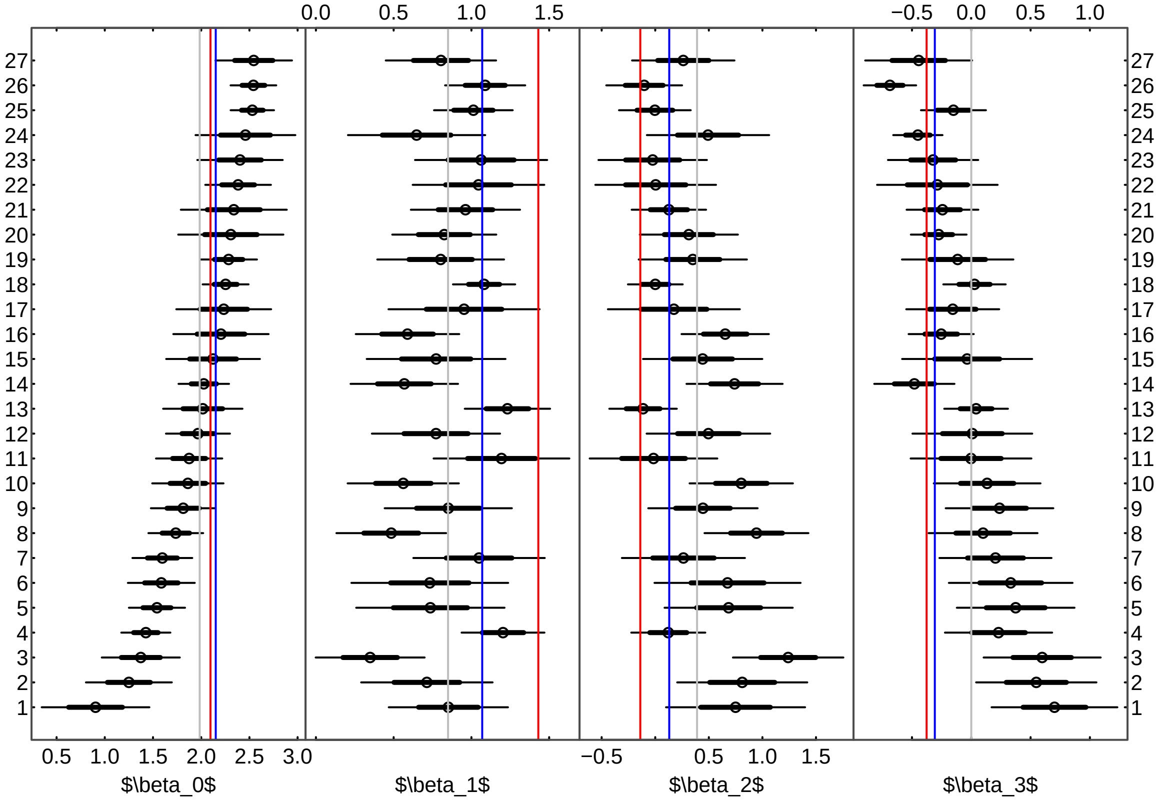


```
lg_lakes_means_lm <- lm(log_chla ~ I(log_tp-log_tp_mu)+I(log_tn-log_tn_mu)+
                          I(log_tp-log_tp_mu):I(log_tn-log_tn_mu), data=lg_lakes_means)
lg_mean_lm_coef <- coef(lg_lakes_means_lm)

## 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_
```

```
## 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_
```

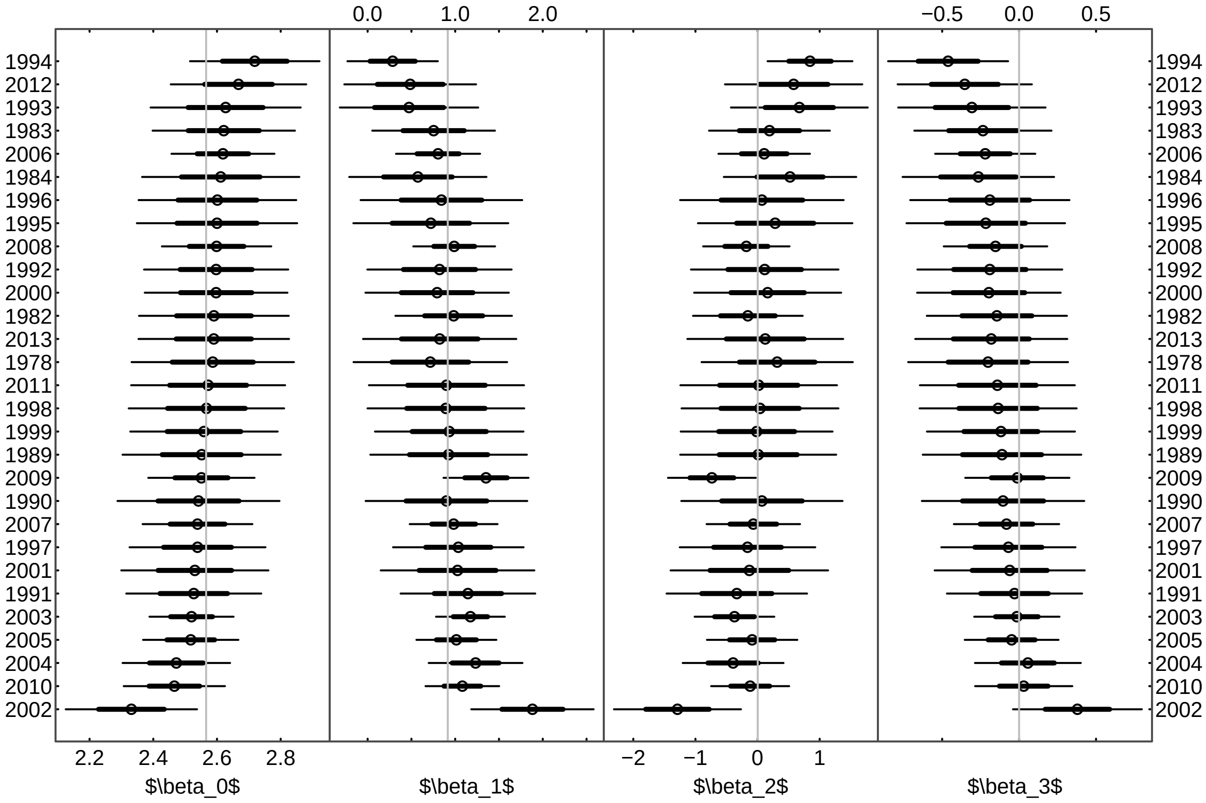
```
## boundary (singular) fit: see help('isSingular')
```

```
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_
```

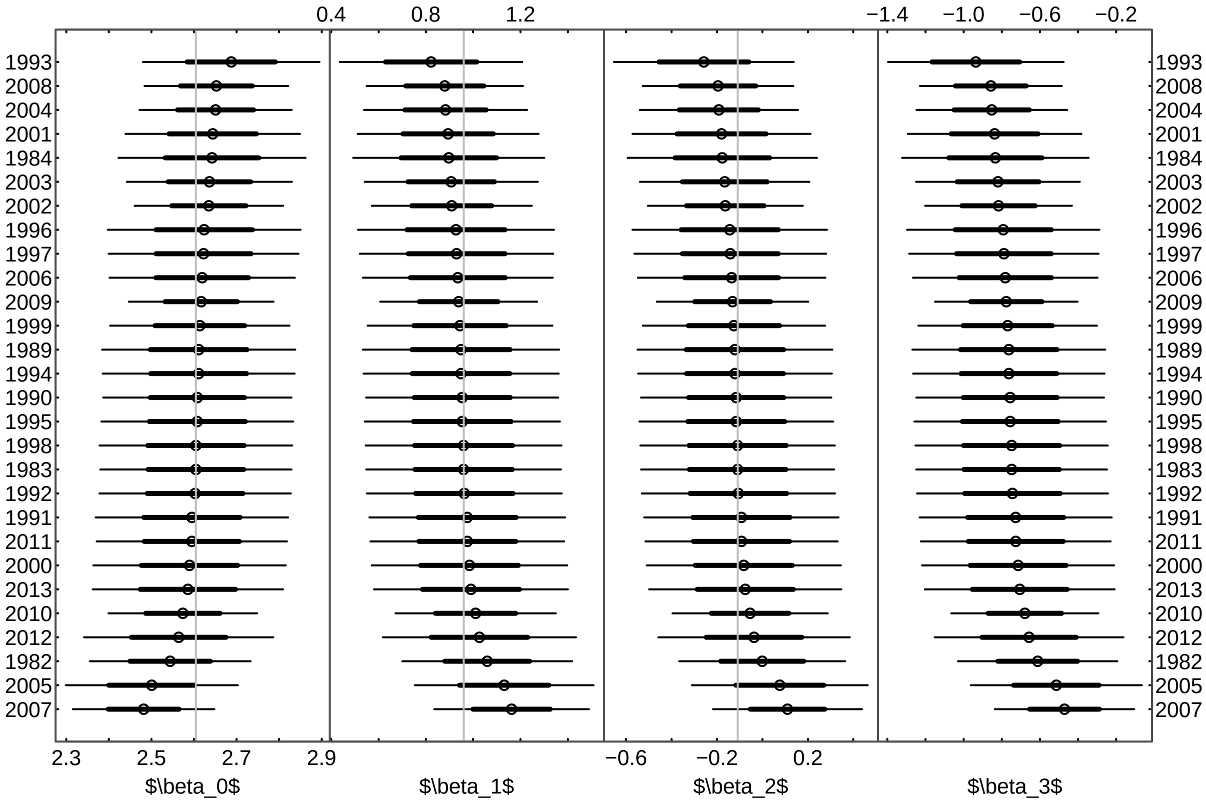
```
## boundary (singular) fit: see help('isSingular')
```

Graphical comparisons

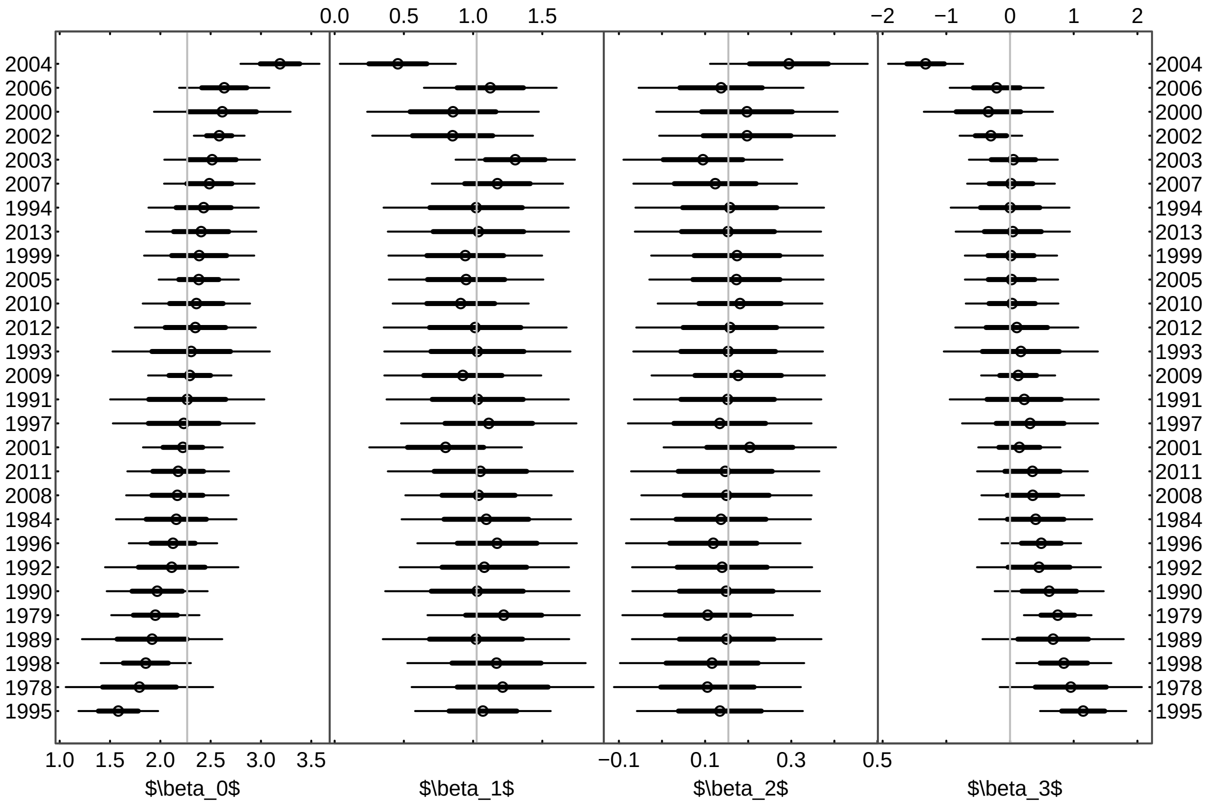
```
## lake 1 by year
est <- t(fixef(lake1_mlm) + t(as.matrix(ranef(lake1_mlm)[["sampleyear"]])))
se <- sqrt(t(se.fixef(lake1_mlm)^2+t(as.matrix(se.ranef(lake1_mlm)[["sampleyear"]]))^2))
oo <- order(est[,1])
ylb <- row.names(ranef(lake1_mlm)[["sampleyear"]])

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.

Why Simpson's paradox

What is the cause of Simpson's paradox?

The cause of Simpson's paradox has been extensively discussed, along with strategies to avoid it. We will highlight two main lines of argumentation that shed light on this phenomenon. Lindley and Novick (1981) focused on the concept of exchangeable units and emphasized that the fallacy arises when applying model results to subjects that are not exchangeable with the data used to develop the model. On the other hand, Pearl et al. (2016) emphasized the significance of accurately delineating the causal structure of the problem, particularly in identifying hidden causes. To illustrate the importance of these two lines of arguments, we will refer to the study conducted by Cheng and Basu (2017).

Cheng and Basu (2017) compiled a dataset consisting of 600 lentic water bodies, including lakes, reservoirs, and wetlands, from various studies, such as the North American Treatment Database (NATD) v2.0 for constructed wetlands. In NATD, wetlands were often represented by a small number of records, typically the temporal (e.g., annual) and spatial (e.g., segments) averages of key factors like flow, hydraulic residence time, and nutrient loading. Using this data, they calculated the nutrient retention for each water body as the ratio of the retained nutrient mass to the input loading:

$$R = \frac{M_{in} - M_{out}}{M_{in}}$$

where M_{in} is the input mass loading and M_{out} is the output loading. Additionally, they estimated two parameters commonly used in water quality models to simulate the fate and transport of contaminants, specifically for phosphorus retention in wetlands. These parameters are the effective removal rate constant k and the hydraulic residence time τ. In a simplified water quality model based on the first-order reaction mechanism, these parameters are used to estimate nutrient retention. - Assuming the water is well mixed, the continuously stirred tank reactor (CSTR) model is used:

$$k = \frac{R}{1-R}\left(\frac{1}{\tau}\right).$$

- Assuming the water flows from inlet to outlet without longitudinal diffusion and dispersion, the plug-flow reactor (PFR) model is used:

$$k = \log(1-R)\left(\frac{1}{\tau}\right).$$

Once k and τ were estimated separately for each wetland, lake, and reservoir, Cheng and Basu (2017) fit a regression model using τ as the predictor variable and k as the response variable:

$$\log(k_j) = \beta_0 + \beta_1 \log(\tau_j) + \epsilon_j$$

where j represents individual waters. They found that the estimated slope β_1 was negative, indicating that as the hydraulic residence time (τ) decreases, the phosphorus effective removal rate constant (k) increases. Based on the positive correlation between a wetland's τ and its surface area, Cheng and Basu (2017) concluded that small wetlands are more effective at removing phosphorus per unit area compared to large wetlands.

```
CB_data <- read.csv(paste(dataDIR, "ChengBasu.csv", sep="/"))
CB_data$k_TP_CSTR <- as.numeric(as.character(CB_data$k_TP_CSTR))
```

```
## Warning: NAs introduced by coercion
```

```
CB_data$k_TP_PFR <- as.numeric(as.character(CB_data$k_TP_PFR))

CB_data$Wetland <- 1
CB_data$Wetland[substring(CB_data$Type, 2, 2)!="W"] <- 0
```

First, we examine the interpretation of model coefficients using the concept of exchangeability. The model described in the previous equation is inherently a model for individual water bodies. Therefore, when fitting the model using combined data from lakes, reservoirs, and wetlands, we are combining nonexchangeable units and exposing ourselves to Simpson's paradox. To illustrate this, we fit the same model using the combined data from lakes, reservoirs, and wetlands, and compare the resulting model coefficients with the coefficients obtained from fitting the same model to the data from lakes, reservoirs, and wetlands separately. Interestingly, the slope estimated using the combined data is significantly lower than the slopes estimated using the data from the three types of water bodies separately:

```
## all data
lm1_all_CSTR <- lm(log(k_TP_CSTR) ~ log(HRT_tau), data=CB_data,
                   subset=k_TP_CSTR>0)
```

```
## Warning in log(k_TP_CSTR): NaNs produced
```

```r
lm1_all_PFR <- lm(log(k_TP_PFR) ~ log(HRT_tau), data=CB_data,
              subset=k_TP_PFR>0)
```

```
## Warning in log(k_TP_PFR): NaNs produced
```

```r
lm1_lake_CSTR <- lm(log(k_TP_CSTR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_CSTR>0 & Type=="Lake")
```

```
## Warning in log(k_TP_CSTR): NaNs produced
```

```r
lm1_lake_PFR <- lm(log(k_TP_PFR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_PFR>0 & Type=="Lake")
```

```
## Warning in log(k_TP_PFR): NaNs produced
```

```r
lm1_res_CSTR <- lm(log(k_TP_CSTR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_CSTR>0 & Type=="Reservoir")
```

```
## Warning in log(k_TP_CSTR): NaNs produced
```

```r
lm1_res_PFR <- lm(log(k_TP_PFR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_PFR>0 & Type=="Reservoir")
```

```
## Warning in log(k_TP_PFR): NaNs produced
```

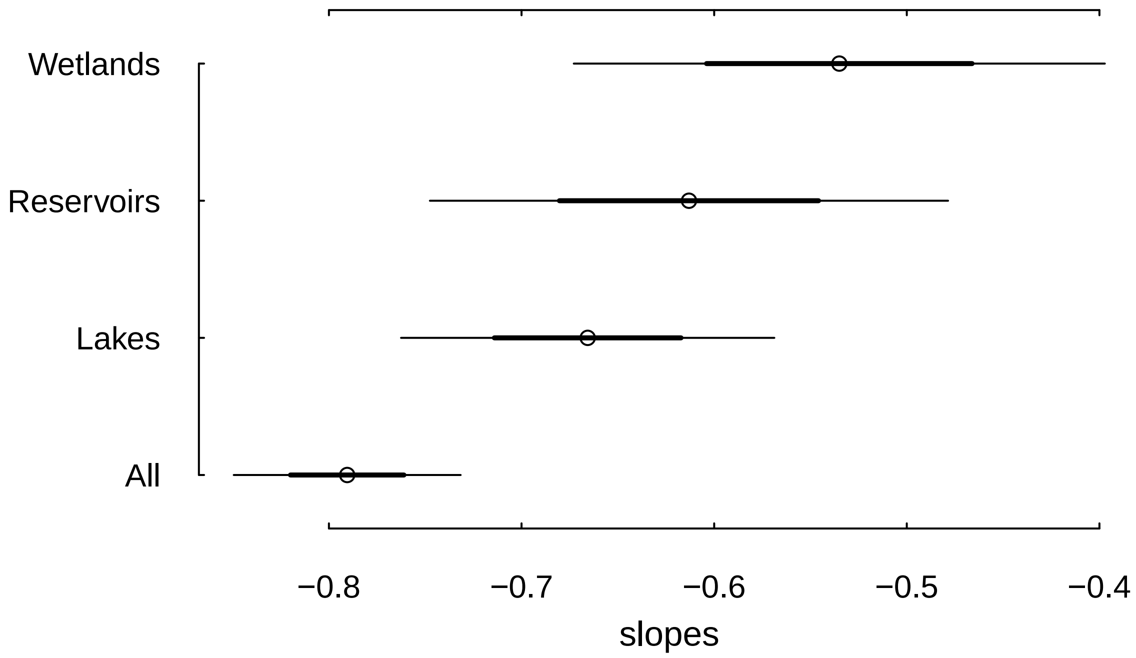
```r
lm1_wet_CSTR <- lm(log(k_TP_CSTR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_CSTR>0 & Wetland==1)
```

```
## Warning in log(k_TP_CSTR): NaNs produced
```

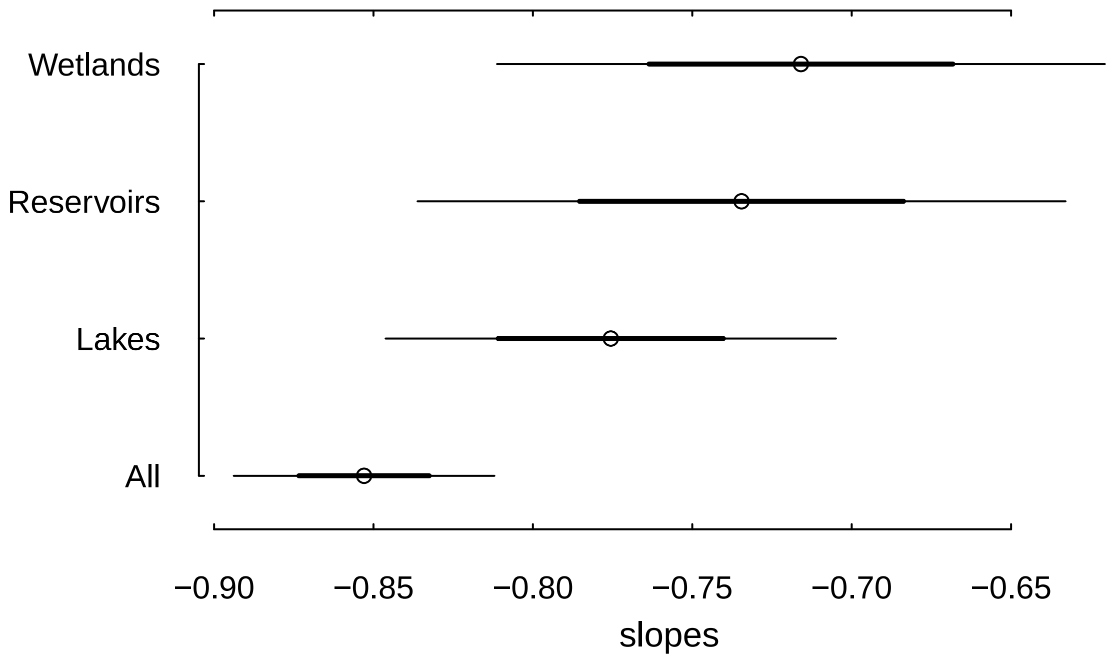
```r
lm1_wet_PFR <- lm(log(k_TP_PFR) ~ log(HRT_tau), data=CB_data,
                     subset=k_TP_PFR>0 & Wetland==1)
```

```
## Warning in log(k_TP_PFR): NaNs produced
```

```r
## Figure 1 -- aggregated slopes
slp_cstr <- rbind(
    summary(lm1_all_CSTR)$coef[2,1:2],
    summary(lm1_lake_CSTR)$coef[2,1:2],
    summary(lm1_res_CSTR)$coef[2,1:2],
    summary(lm1_wet_CSTR)$coef[2,1:2])
row.names(slp_cstr)  <- c("All", "Lakes", "Reservoirs","Wetlands")

slp_pfr <- rbind(
    summary(lm1_all_PFR)$coef[2,1:2],
    summary(lm1_lake_PFR)$coef[2,1:2],
    summary(lm1_res_PFR)$coef[2,1:2],
    summary(lm1_wet_PFR)$coef[2,1:2])
row.names(slp_pfr)  <- c("All", "Lakes", "Reservoirs","Wetlands")

par(mar=c(5,6,4, 2), mgp=c(2.25,1,0), tck=0.01, cex.main=1.3, cex.lab=1.1)
line.plots(slp_cstr[,1], slp_cstr[,2], yaxis=2, ylab="", xlab="slopes",
           yaxisLab =  rownames(slp_cstr))
```



```
par(mar=c(5,6,4, 2), mgp=c(2.25,1,0), tck=0.01, cex.main=1.3, cex.lab=1.1)
line.plots(slp_pfr[,1],slp_pfr[,2], yaxis=2, xlab="slopes", ylab="",
           yaxisLab =  rownames(slp_pfr))
```



To gain further insights, we proceed to fit the model using data from individual wetlands. We select six wetlands from the database that have more than 10 observations and estimate wetland-specific slopes using a hierarchical model, assuming that the regression coefficients for each wetland are exchangeable. These six wetlands vary greatly in size, with mean volumes ranging from 5.24 to 47,585.27 m^3.

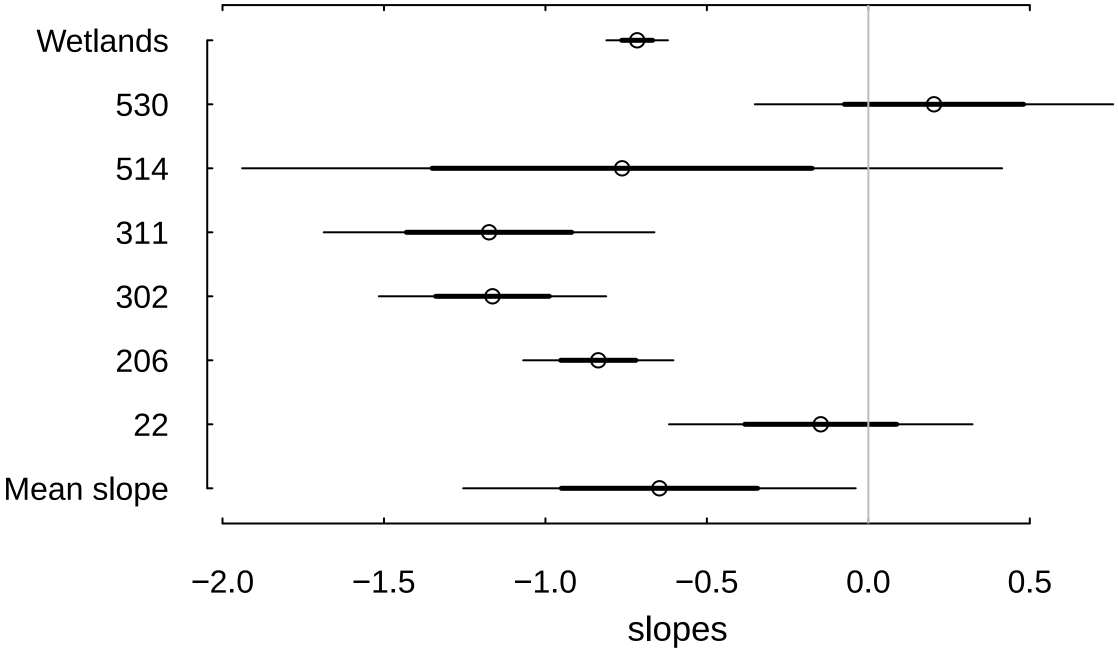
By considering the concept of exchangeable units, we acknowledge that observations from the wetland with an average volume of 5.24 m^3 cannot be treated as exchangeable with observations from the wetland with an average volume of 47,585.27 m^3. Therefore, direct combination of data from these wetlands is not appropriate. However, by assuming exchangeability of individual wetlands with respect to the regression model coefficients, we can partially pool the data from multiple wetlands using a hierarchical model.

The results reveal interesting patterns. The slopes estimated for the smallest and largest wetlands are not significantly different from 0 (although larger than the slope estimated using the combined wetland data). In contrast, the slopes of the four intermediate-sized wetlands either exhibit high uncertainty (e.g., wetland 514) or are notably smaller than the slope estimated using the combined wetland data.

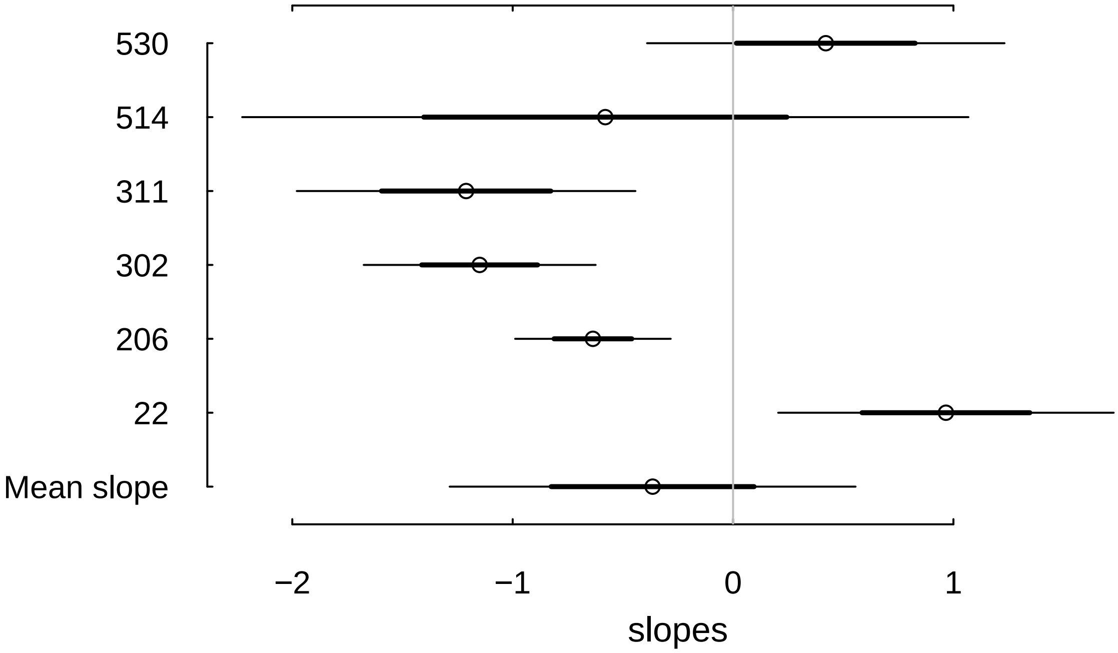
```
CB_wetland <- CB_data[CB_data$Wetland==1,]
CB_wetland$sites <- as.numeric(CB_wetland$Year)
## we determined that Year was mislabeled

CB_NADB <- CB_wetland[CB_wetland$sites<1000,] ## small wetlands
CB_NADB2 <- CB_NADB[CB_NADB$sites %in%
              names(table(CB_NADB$sites)[table(CB_NADB$sites)>10]),
                c("sites", "HRT_tau", "k_TP_PFR","k_TP_CSTR")]
## more than 10 observations
CB_NADB2 <- CB_NADB2[!is.na(CB_NADB2$k_TP_CSTR)&CB_NADB2$k_TP_CSTR>0,]
table(CB_NADB2$sites)
```

```
##
##  22 206 302 311 514 530
##  16  34  31  26  12  27
```

```
lmer_nadb_PFR <- lmer(log(k_TP_PFR) ~ log(HRT_tau) + (1+log(HRT_tau)|sites), data=CB_NADB2)

lmer_nadb_CSTR <- lmer(log(k_TP_CSTR) ~ log(HRT_tau) + (1+log(HRT_tau)|sites), data=CB_NADB2)

lmer_cstr_slp <- fixef(lmer_nadb_CSTR)[2] + ranef(lmer_nadb_CSTR)$sites[,2]
lmer_cstr_slp_se <- se.ranef(lmer_nadb_CSTR)$sites[,2]
lmer_slp_cstr <- data.frame(Estimate=c(fixef(lmer_nadb_CSTR)[2],
                                       lmer_cstr_slp),
                            se.estimate=c(se.fixef(lmer_nadb_CSTR)[2],
                                          lmer_cstr_slp_se))
rownames(lmer_slp_cstr)[1] <- "Mean slope"

lmer_pfr_slp <- fixef(lmer_nadb_PFR)[2] + ranef(lmer_nadb_PFR)$sites[,2]
lmer_pfr_slp_se <- se.ranef(lmer_nadb_PFR)$sites[,2]
lmer_slp_pfr <- data.frame(Estimate=c(fixef(lmer_nadb_PFR)[2],
                                      lmer_pfr_slp),
                           se.estimate=c(se.fixef(lmer_nadb_PFR)[2],
                                         lmer_pfr_slp_se))
rownames(lmer_slp_pfr)[1] <- "Mean slope"

lmer_slp_pfr_plot <- rbind(lmer_slp_pfr, slp_pfr[4,])
rownames(lmer_slp_pfr_plot)[8] <- "Wetlands"

par(mar=c(5,6,4, 2), mgp=c(2.25,1,0), tck=0.01, cex.main=1.3, cex.lab=1.1)
line.plots(lmer_slp_pfr_plot[,1],lmer_slp_pfr_plot[,2], yaxis=2, ylab="", xlab="slopes",
           yaxisLab=rownames(lmer_slp_pfr_plot))
```



```
par(mar=c(5,6,4, 2), mgp=c(2.25,1,0), tck=0.01, cex.main=1.3, cex.lab=1.1)
line.plots(lmer_slp_cstr[,1],lmer_slp_cstr[,2], yaxis=2, ylab="", xlab="slopes",
           yaxisLab=rownames(lmer_slp_cstr))
```



At this point, it becomes evident that the variation in estimated slopes is a clear manifestation of Simpson's paradox. The average model coefficients for all wetlands, denoted as "Mean slope," are parameterized by the hyper-distribution model, in which the wetland-specific coefficients (β_j) are assumed to follow a normal distribution ($N(\mu_\beta, \sigma^2_\beta)$). It is highly likely that the hyper-distribution mean (μ_β) differs from the slope estimated using the combined wetland data.

The hyper-distribution mean holds a significant physical interpretation as it represents the average of the wetland-specific coefficients. In contrast, the slope estimated using the combined data lacks a clear and meaningful interpretation. Therefore, the hyper-distribution mean provides a more reliable and meaningful estimate of the true underlying relationship between the variables.

By acknowledging Simpson's paradox and employing a hierarchical modeling approach, we can better account for the hierarchical structure of the data and obtain more accurate and interpretable estimates of the model coefficients.

– spurious correlation

Secondly, we can explore the differences in estimated slopes at various levels of aggregation using causal inference, as demonstrated in Tang et al. (2019). Although we lack additional variables for such an analysis, we can approach the linear regression model from a causal perspective. The two parameters of interest, k and τ, represent distinct aspects of a wetland and are likely independent of each other (Carleton and Montas, 2007; Hejzlar et al., 2007; Vollenweider, 1975).

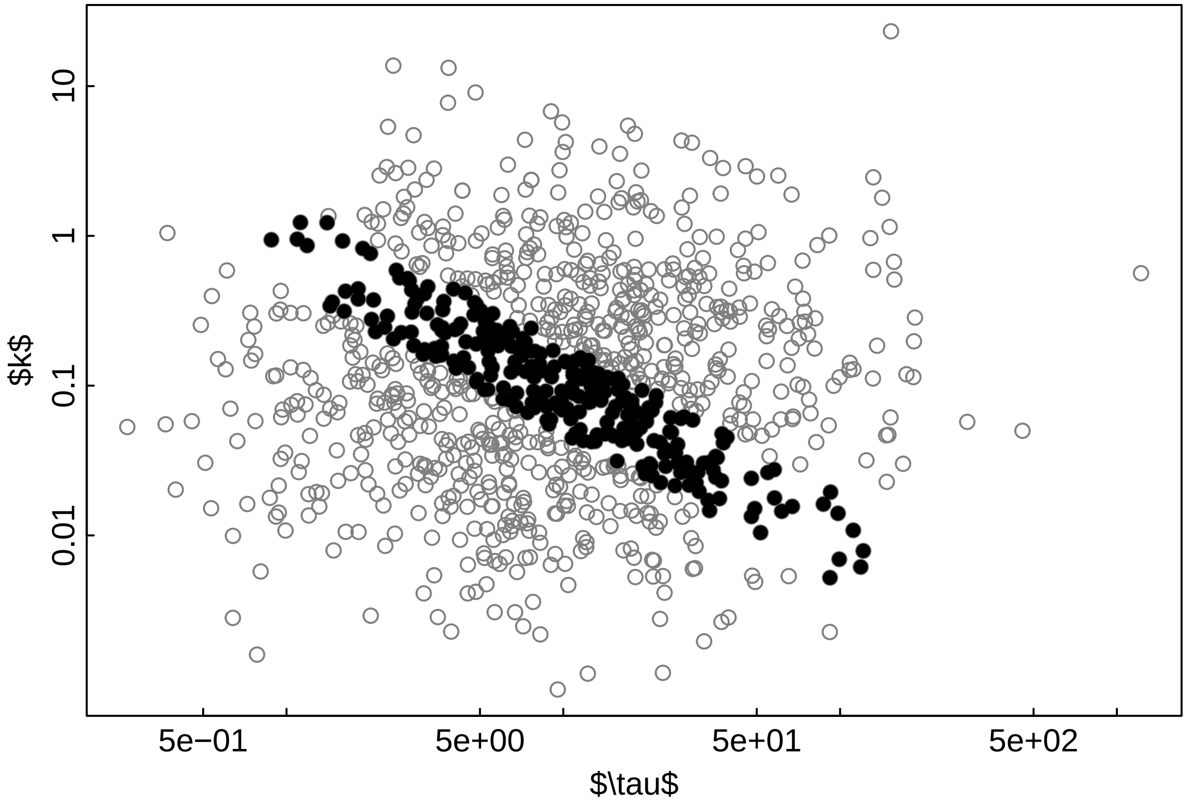
In this context, the connection between these parameters is established through the percent removal (R) in the CSTR or PFR models. The parameter k reflects the inherent characteristics of a wetland, while τ represents a parameter determined by the external input of water relative to the wetland's size. The percent removal is influenced by both k and τ, as it approximates the duration the nutrient mass remains within the system. In other words, the causal diagram for wetland phosphorus removal can be represented as $k \rightarrow\ R \leftarrow\ \tau$, indicating that k and τ jointly determine \$R” but they are independent of each other.

However, a spurious correlation between k and τ can emerge when R exhibits narrow variation. In computer science literature, k and τ are known to be "d-separated" by R. When two variables are d-separated, any apparent correlation between them is likely to be spurious. To illustrate this effect, we can employ a simulation. We randomly generate values of k and τ and calculate R using the CSTR model, introducing random noise ($R_i = k_i\tau_i/(1 + k_i\tau_i) + \epsilon_i$).

In this simulation, the parameters k_i and τ_i are independently drawn from log-normal distributions with log means ($\mu_k = -2.726$ and $\mu_\tau = 1.914$) and log standard deviations ($\sigma_k = 1.371$ and $\sigma_\tau = 1.269$), derived from the log values of k and τ for TP in the data analyzed by Cheng and Basu (2017). We then create a scatter plot of the randomly generated k and τ values, highlighting the data points where the corresponding R falls between 32% and 65%. Cheng and Basu (2017) indicated that wetlands with a percent removal (R) in this range exhibit "no significant differences between systems and across constituents."

The highlighted data points in the scatter plot demonstrate the (spurious) negative correlation between k and τ, which strikingly resembles the pattern observed in Cheng and Basu (2017) during their data analysis. This similarity suggests that the conclusion stating small wetlands are more effective in phosphorus retention on a per unit area basis may result from the spurious correlation induced by the d-separated relationship between k and τ.

```
mu_k <- mean(log(CB_data$k_TP_CSTR[CB_data$Wetland==1 & CB_data$k_TP_CSTR>0]),
             na.rm=TRUE)
s_k <- sd(log(CB_data$k_TP_CSTR[CB_data$Wetland==1 & CB_data$k_TP_CSTR>0]),
          na.rm=TRUE)

mu_tau <- mean(log(CB_data$HRT_tau[CB_data$Wetland==1 & CB_data$k_TP_CSTR>0]),
               na.rm=TRUE)
s_tau <- sd(log(CB_data$HRT_tau[CB_data$Wetland==1 & CB_data$k_TP_CSTR>0]),
            na.rm=TRUE)

ln_K_TP <- exp(rnorm(1000,mu_k,s_k))
ln_T_TP <- exp(rnorm(1000,mu_tau,s_tau))
R1 <- (ln_K_TP * ln_T_TP)/(1+(ln_K_TP * ln_T_TP))

par(mar=c(3,3,1,1),mgp=c(1.25,0.125,0),tck=0.01)
plot(ln_T_TP,ln_K_TP, log="xy",
     ylab="$k$",
     xlab="$\\tau$",
     col=gray(0.5), axes=F)
axis(1)
axis(2, at=c(0.01,0.1,1,10), labels=c("0.01","0.1","1","10"))
box()
points(ln_T_TP[R1>0.32 & R1<0.65],
       ln_K_TP[R1>0.32 & R1<0.65], pch=16)
```


k
5e−01
5e+00
5e+01
5e+02
τ