Supplementary information II

Johannes Rusch, David Strand, Tom Andersen

1 6 2021

Data preprocessing

All data from the field observations is contained in a single Excel file which we read with the `read_xl` package. The data we will use are in the second sheet of this file. First 3 lines are blank and should be removed. We do this by keeping only the rows that contain no missing values (`NA`s) with the `complete.cases` function.

```
library(readxl)

d <- read_excel("PaperV_field data_TA&JRedit_new.xlsx", sheet="Ark2")
d <- subset(d, complete.cases(d))
```

Sample identification information is contained in the first 7 columns. We convert the subset `samp` from `tibble` to `data.frame` and also shorten the names of this object.

```
samp <- as.data.frame(d[, 1:7])
names(samp) <- c("samp.ID", "date", "loc.ID", "repl.ID", "vol.filt", "Lake", "CPUE")
```

We convert `samp.ID`, `loc.ID`, `repl.ID` and `Lake` to factors before we proceed. We also convert `date` from POSIX format to just simple dates.

```
samp$samp.ID <- factor(samp$samp.ID)
samp$date <- as.Date(samp$date)
samp$loc.ID <- factor(samp$loc.ID)
samp$repl.ID <- factor(samp$repl.ID)
samp$Lake <- factor(samp$Lake)
```

There were 15 samples from Stora Le and 18 from Øymarkssjøen. Most of the samples had at least 2 replicates, and *Pacifastacus* catch per unit effort (CPUE) varied from <1 to >25 with a mean of 11. There were no samples with zero catch in any of the lakes.

```
summary(samp)
```

```
##     samp.ID        date                loc.ID   repl.ID    vol.filt
##  OK102  : 1   Min.   :2016-06-08   SL1    : 5   1:12    Min.   :2.500
##  OK14   : 1   1st Qu.:2016-06-08   SL2    : 5   2:11    1st Qu.:5.000
##  OK15   : 1   Median :2016-08-10   SL3    : 5   3: 4    Median :5.000
##  OK28   : 1   Mean   :2016-08-11   OE8    : 3   4: 3    Mean   :4.742
##  OK29   : 1   3rd Qu.:2016-09-20   OE1    : 2   5: 3    3rd Qu.:5.000
```

```
##  OK30    : 1   Max.   :2016-09-20   OE2    : 2           Max.   :5.000
##  (Other):27                        (Other):11
##           Lake         CPUE
##  Stora Le   :15   Min.   : 0.60
##  Øymarksjøen:18   1st Qu.: 3.60
##                   Median :12.20
##                   Mean   :10.98
##                   3rd Qu.:17.60
##                   Max.   :25.80
##
```

The sampling is a mixture of "classical" pseudoreplicates at the same station at the same date in Stora Le, while most of the samples from Øymarksjøen are replicates from the same location at different dates within the same month (with 1 exception: OE8).

We choose to ignore the time effect assuming that variation between sites is larger than between times at the same site (especially since time will be confounded with site anyway, since samples were taken at different times of year). We thus use `loc.ID` as grouping variable for sites.

We then extract the columns for total and positive ddPCR droplets. There are droplet counts for both species (*Aphanomyces* and *Pacifastacus*) and for individual filter halves (A and B). We also shorten the names in this step.

```
posDRP <- d[, c("posDRP_Aph_A", "posDRP_Aph_B", "posDRP_Pac_A", "posDRP_Pac_B")]
names(posDRP) <- c("Aph.A", "Aph.B","Pac.A", "Pac.B")

totDRP <- d[, c("totDRP_Aph_A", "totDRP_Aph_B", "totDRP_Pac_A", "totDRP_Pac_B")]
names(totDRP) <- c("Aph.A", "Aph.B","Pac.A", "Pac.B")
```

We define detection as 3 or more positive droplets and discard samples with less than 8000 droplets

```
detect <- as.data.frame(posDRP > 2)
discard <- (totDRP < 8000)
detect[discard] <- NA
detect <- as.data.frame(detect)
```

Consistency check between A/B filter halves:

```
(Aph.AB <- with(detect, table(Aph.A, Aph.B)))
```

```
##        Aph.B
## Aph.A   FALSE TRUE
##   FALSE    10    6
##   TRUE      3   12
```

There were 9 inconsistent samples for *Aphanomyces*, 0 of these had both replicates discarded, while 2 of these had at least 1 discarded.

```
(Pac.AB <- with(detect, table(Pac.A, Pac.B)))
```

```
##        Pac.B
## Pac.A   FALSE TRUE
##   FALSE    25    0
##   TRUE      1    5
```

There were 1 inconsistent sample for *Pacifastacus*, 0 with both replicates discarded, while 2 of these had at least 1 discarded.

We define a site information data frame that will be used for both species, with only one row per site. Since all sites had at least a replicate number 1, we extract only these rows.

```
site <- samp[samp$repl.ID == 1, c("loc.ID", "Lake", "CPUE")]
names(site) <- c("site", "Lake", "CPUE")
```

We then define two data frames, one for each species, containing the PCR detection data as logical variables.

```
Pac.pcr <- data.frame(site=samp$loc.ID, sample=samp$repl.ID,
                      pcr1=detect$Pac.A, pcr2=detect$Pac.B)
Aph.pcr <- data.frame(site=samp$loc.ID, sample=samp$repl.ID,
                      pcr1=detect$Aph.A, pcr2=detect$Aph.B)
```

Modelling strategy

We use a hierarchical Bayesian occupancy modelling package from Stratton, Sepulveda, and Hoegh (2020) "msocc: Fit and analyse computationally efficient multi-scale occupancy models in r." Methods in Ecology and Evolution 11.9: 1113-1120.(https://github.com/StrattonCh/msocc). Since some of the MCMC runs seem to have high autocorrelation we run 11000 samples instead of the default, discard first 1000, and thin remainder by 10 (run times are still about 0.1 minutes).

We investigate 3 model structures for each species: a null model with no effect of `Lake` or `CPUE`, and two models with additive or interactive effects of the two covariates. We compare model performances with the Watanabe–Akaike information criterion (WAIC), which is a generalized version of the Akaike information criterion (AIC).

Pacifastacus **models**

```
library(msocc)

Pac.mod.0 <- msocc_mod(wide_data = Pac.pcr,
                       site = list(model = ~ 1, cov_tbl = site),
                       sample = list(model = ~1, cov_tbl = Pac.pcr),
                       rep = list(model = ~1, cov_tbl = Pac.pcr),
                       num.mcmc = 11000, progress=FALSE)

Pac.mod.1 <- msocc_mod(wide_data = Pac.pcr,
                       site = list(model = ~ Lake + CPUE, cov_tbl = site),
                       sample = list(model = ~1, cov_tbl = Pac.pcr),
                       rep = list(model = ~1, cov_tbl = Pac.pcr),
                       num.mcmc = 11000, progress=FALSE)

Pac.mod.2 <- msocc_mod(wide_data = Pac.pcr,
                       site = list(model = ~ Lake * CPUE, cov_tbl = site),
                       sample = list(model = ~1, cov_tbl = Pac.pcr),
                       rep = list(model = ~1, cov_tbl = Pac.pcr),
                       num.mcmc = 11000, progress=FALSE)
```

We choose `Pac.mod.2` with interactive effects between `Lake` and `CPUE` since it has the lowest WAIC value.

```
waic(Pac.mod.0) # 148.6
```

```
## [1] 149.1462
```

```
waic(Pac.mod.1) # 126.8
```

```
## [1] 119.4682
```

```
waic(Pac.mod.2) #  63.2
```

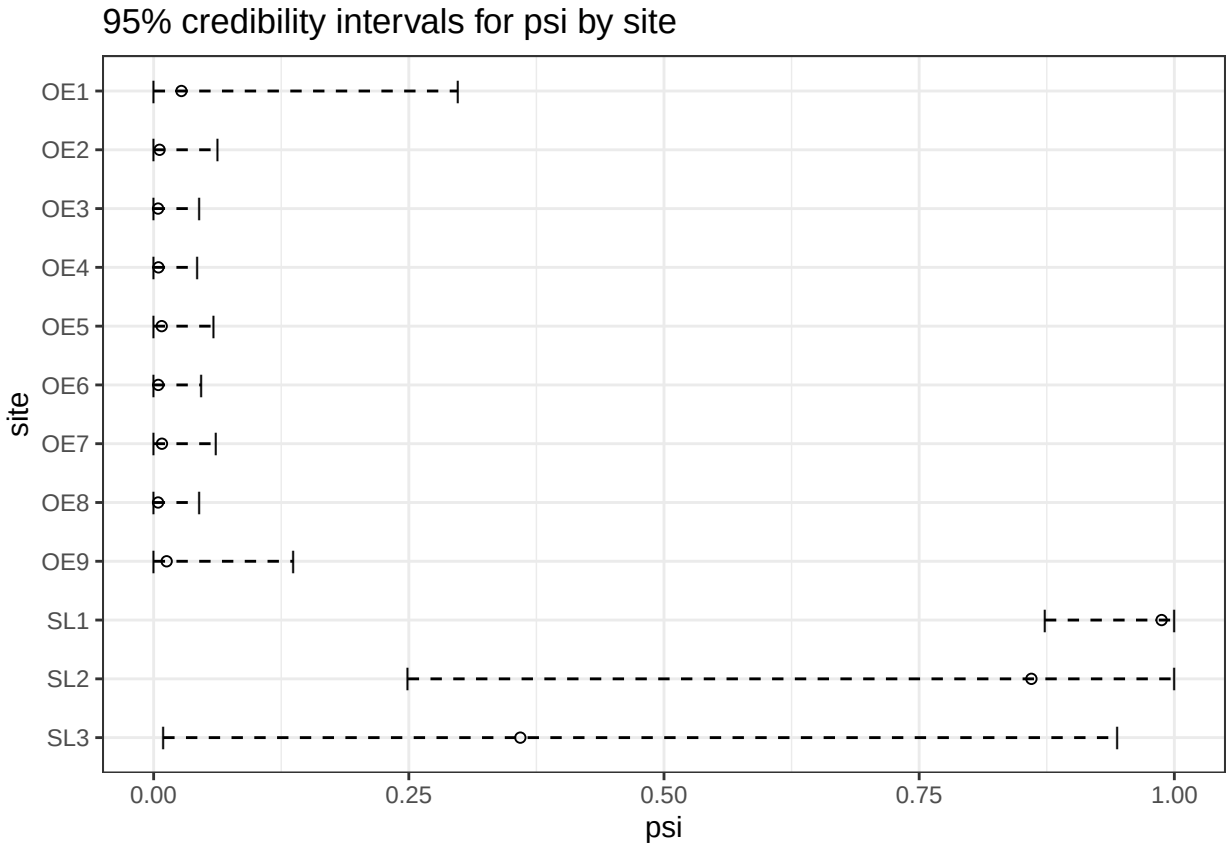
```
## [1] 64.5775
```

```
posterior_summary(Pac.mod.2)
```

```
##    site sample rep          psi     theta         p
## 1   OE1      1   1 2.874792e-04 0.6923569 0.8621211
## 2   OE2      1   1 1.418206e-07 0.6923569 0.8621211
## 3   OE3      1   1 8.008278e-10 0.6923569 0.8621211
## 4   OE4      1   1 1.580124e-12 0.6923569 0.8621211
## 5   OE5      1   1 2.307447e-17 0.6923569 0.8621211
## 6   OE6      1   1 3.325712e-09 0.6923569 0.8621211
## 7   OE7      1   1 1.292189e-17 0.6923569 0.8621211
## 8   OE8      1   1 8.008278e-10 0.6923569 0.8621211
## 9   OE9      1   1 2.006719e-05 0.6923569 0.8621211
## 10  SL1      1   1 1.000000e+00 0.6923569 0.8621211
## 11  SL2      1   1 9.649060e-01 0.6923569 0.8621211
## 12  SL3      1   1 2.926579e-01 0.6923569 0.8621211
```

The probability of *Pacifastacus* presence at the site (psi) varied from 0.00 to 1.00 with a strong lake effect. Probability of occurrence in the sample conditional on presence at the site (theta) was around 70%, while probability of detection in the replicate conditional on occurrence in the sample (p) was around 86%.

```
cred_plot(Pac.mod.2)
```



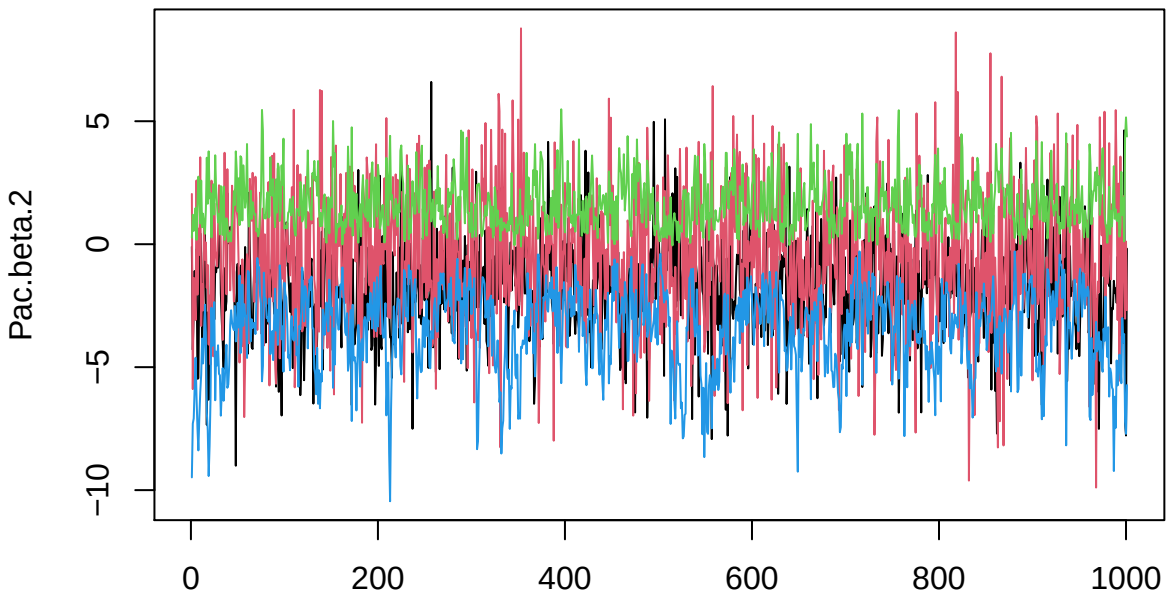
This plot shows the probability of *Pacifastacus* presence (psi) grouped by sites, illustrating the dramatically higher occupancy probability in Stora Le (sites SL1-3).

We then extract the MCMC samples from `Pac.mod.2`, discard the warm-up and retain only every tenth samples (thinning).

```
Pac.beta.2 <- Pac.mod.2$beta
Pac.beta.2 <- Pac.beta.2[seq(1000, 11000, 10), ]
```

The MCMC chains for the 4 fixed effect parameters show generally good mixing.

```
matplot(Pac.beta.2, type="l", lty=1)
```

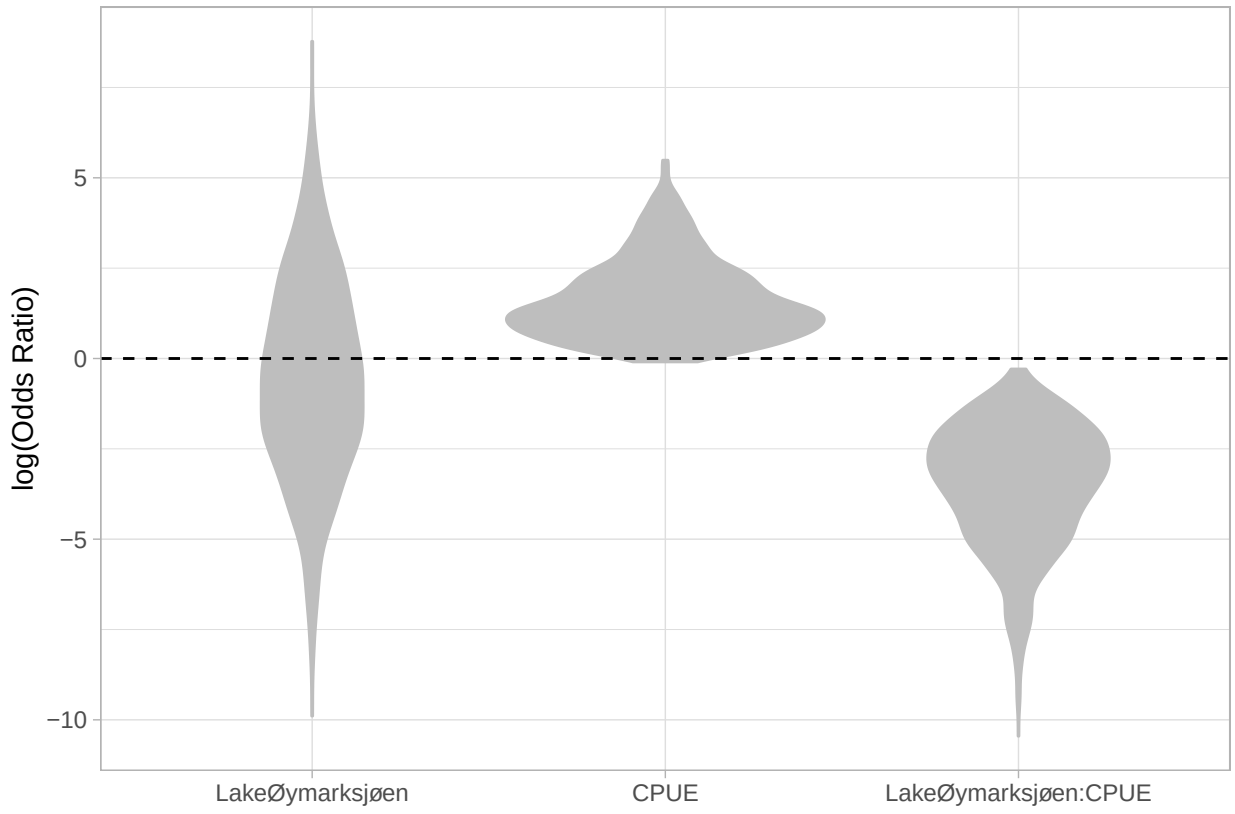


We focus on the 3 non-intercept parameters since these can be interpreted log(odds ratios) between lakes and by a unit increase in CPUE, so we make a new data frame by stacking the last 3 columns of the fixed effect parameters samples.

```
Pac.odds.2 <- stack(as.data.frame(Pac.beta.2[, -1]))
```

```
library(ggplot2)

ggplot(Pac.odds.2, aes(x=ind, y=values)) +
  geom_violin(fill="gray", color="gray") +
  geom_hline(yintercept=0, linetype=2) +
  ylab("log(Odds Ratio)") + xlab("") +
  theme_light()
```



The figure shows that there is practically no intercept difference between lakes (i.e., the probability of presence at CPUE = 0). This is among other things, probably a result of that the data set unfortunately did not contain any observations from sites with CPUE = 0. There is a strong effect of CPUE for the reference level lake (Stora Le), which becomes zero or negative for Øymarkssjøen.

We make predictions for probabilities of presence as function of CPUE and between lakes by making a new data frame with CPUE values from 0 to 20 in all combinations with the two lakes.

```
Pac.pred <- expand.grid(CPUE=seq(0, 20, 0.1), Lake=levels(site$Lake))
Pac.pred$Lake <- factor(Pac.pred$Lake)
```

Since `msocc` has no `predict` method, we make predictions by first creating a model matrix `X.Pac` corresponding to `Pac.mod.2` (`~ Lake * CPUE`), and then matrix multiply this by each of the MCMC samples of the model coefficient.

```
X.Pac <- model.matrix( ~ Lake * CPUE, data=Pac.pred)

Pac.z <- matrix(NA, nrow=nrow(X.Pac), ncol=nrow(Pac.beta.2))
for (i in 1:nrow(Pac.beta.2)) {
  Pac.z[, i] <- X.Pac %*% Pac.beta.2[i, ]
}
```

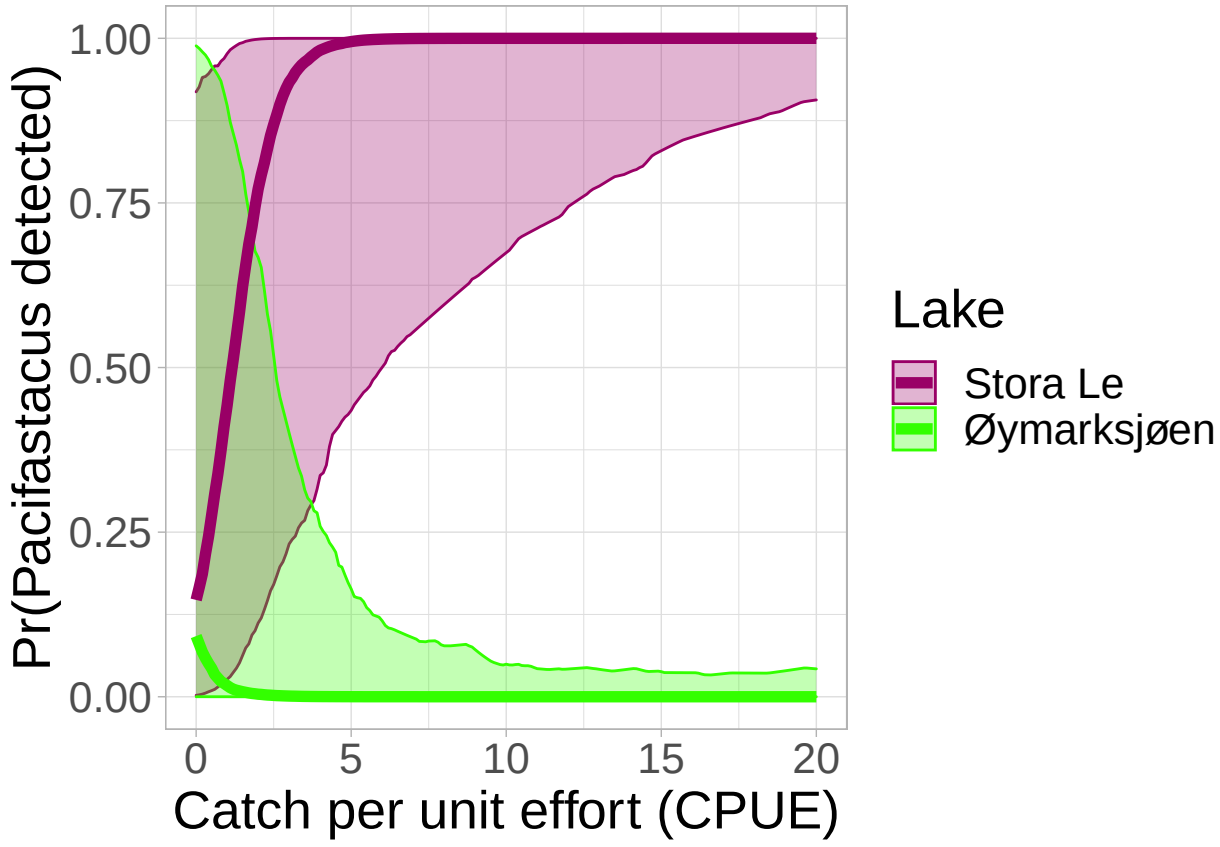
We can then extract medians and 95% percentiles from the resulting z-score matrix (`Pac.z`), and append these columns to the prediction data frame (`Pac.pred`)

```
Pac.pred$z.med <- apply(Pac.z, 1, median)
Pac.pred$z.lcl <- apply(Pac.z, 1, quantile, p=0.025)
Pac.pred$z.ucl <- apply(Pac.z, 1, quantile, p=0.975)
```

We then inverse logit-transform the log(odds) columns to get probabilities

```
Pac.pred$p.med <- 1 / (1 + exp(-Pac.pred$z.med))
Pac.pred$p.lcl <- 1 / (1 + exp(-Pac.pred$z.lcl))
Pac.pred$p.ucl <- 1 / (1 + exp(-Pac.pred$z.ucl))
```

```
ggplot(data=Pac.pred, aes(x=CPUE, y=p.med, group=Lake, color=Lake)) +
  geom_ribbon(data=Pac.pred, aes(ymin=p.lcl, ymax=p.ucl, fill=Lake), alpha=0.3) +
  geom_line(size=2) +
  scale_color_manual(values=c('#990066','#33FF00')) +
  ylab("Pr(Pacifastacus detected)") +
  xlab("Catch per unit effort (CPUE)") +
  theme_light() +
  scale_fill_manual(values = c("#990066", "#33FF00")) +
  theme(text = element_text(size = 20))
```



Aphanomyces models

```
Aph.mod.0 <- msocc_mod(wide_data = Aph.pcr,
                       site = list(model = ~ 1, cov_tbl = site),
                       sample = list(model = ~1, cov_tbl = Aph.pcr),
```

```
                        rep = list(model = ~1, cov_tbl = Aph.pcr),
                        num.mcmc = 11000, progress=FALSE)

Aph.mod.1 <- msocc_mod(wide_data = Aph.pcr,
                        site = list(model = ~ Lake + CPUE, cov_tbl = site),
                        sample = list(model = ~1, cov_tbl = Aph.pcr),
                        rep = list(model = ~1, cov_tbl = Aph.pcr),
                        num.mcmc = 11000, progress=FALSE)

Aph.mod.2 <- msocc_mod(wide_data = Aph.pcr,
                        site = list(model = ~ Lake * CPUE, cov_tbl = site),
                        sample = list(model = ~1, cov_tbl = Aph.pcr),
                        rep = list(model = ~1, cov_tbl = Aph.pcr),
                        num.mcmc = 11000, progress=FALSE)
```

We choose `Aph.mod.1` with additive effects between `Lake` and `CPUE` since it has only barely higher WAIC value than the model with interactions.

```
waic(Aph.mod.0)
```

```
## [1] 167.4855
```

```
waic(Aph.mod.1)
```

```
## [1] 146.1792
```

```
waic(Aph.mod.2)
```

```
## [1] 144.3071
```

```
posterior_summary(Aph.mod.1)
```

```
##    site sample rep       psi     theta         p
## 1   OE1      1   1 0.9999512 0.7237393 0.6972348
## 2   OE2      1   1 1.0000000 0.7237393 0.6972348
## 3   OE3      1   1 1.0000000 0.7237393 0.6972348
## 4   OE4      1   1 1.0000000 0.7237393 0.6972348
## 5   OE5      1   1 1.0000000 0.7237393 0.6972348
## 6   OE6      1   1 1.0000000 0.7237393 0.6972348
## 7   OE7      1   1 1.0000000 0.7237393 0.6972348
## 8   OE8      1   1 1.0000000 0.7237393 0.6972348
## 9   OE9      1   1 0.9999991 0.7237393 0.6972348
## 10  SL1      1   1 1.0000000 0.7237393 0.6972348
## 11  SL2      1   1 0.9998660 0.7237393 0.6972348
## 12  SL3      1   1 0.9449296 0.7237393 0.6972348
```

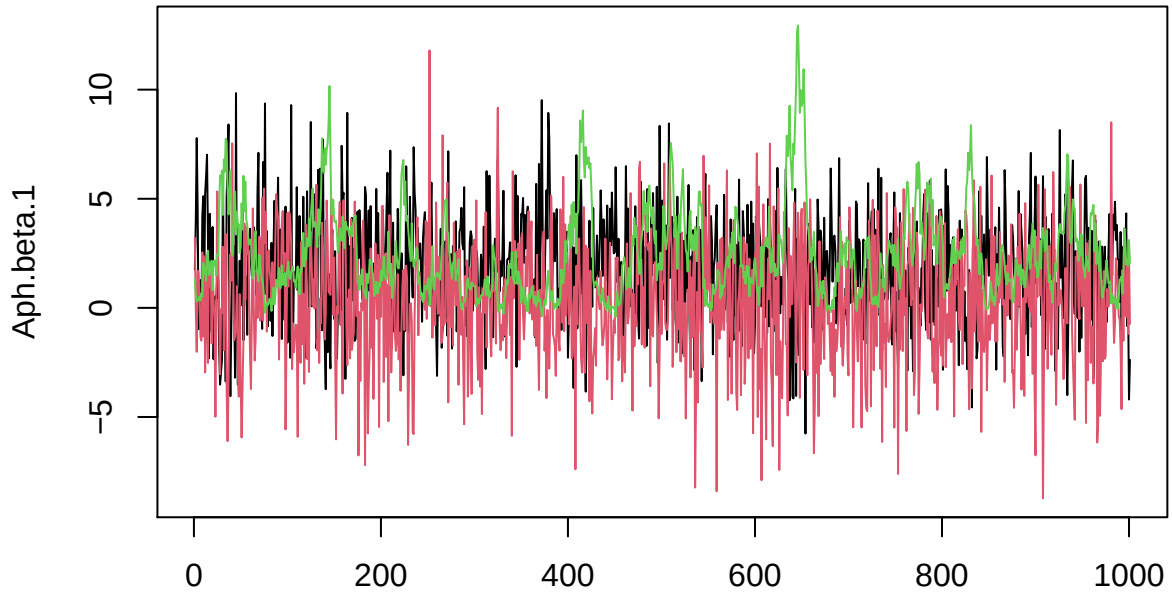
The probability of *Aphanomyces* presence at the site (psi) was around 0.99 to 1.00 for all sites. Probability of occurrence in the sample conditional on presence at the site (theta) was around 73%, while probability of detection in the replicate conditional on occurrence in the sample (p) was around 70%.

We then extract the MCMC samples from `Aph.mod.1`, discard the warm-up and retain only every tenth samples (thinning).

```
Aph.beta.1 <- Aph.mod.1$beta
Aph.beta.1 <- Aph.beta.1[seq(1000, 11000, 10), ]
```

The MCMC chains for the 3 fixed effect parameters show not as good mixing as for the *Pacifastacus* model but still quite acceptable

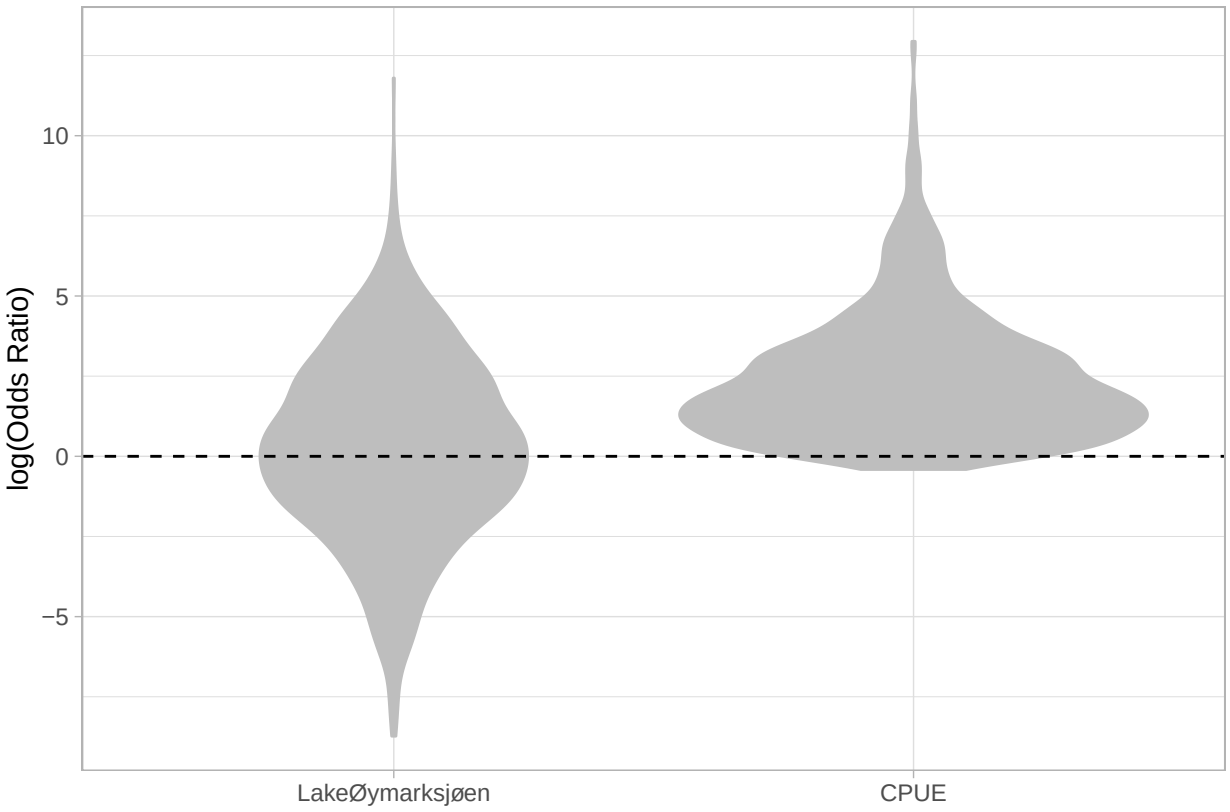
```
matplot(Aph.beta.1, type="l", lty=1)
```



We focus on the 2 non-intercept parameters since these can be interpreted log(odds ratios) between lakes an by a unit increase in CPUE, so we make a new data frame by stacking the last 3 columns of the fixed effect parameters samples.

```
Aph.odds.1 <- stack(as.data.frame(Aph.beta.1[, -1]))
```

```
ggplot(Aph.odds.1, aes(x=ind, y=values)) +
  geom_violin(fill="gray", color="gray") +
  geom_hline(yintercept=0, linetype=2) +
  ylab("log(Odds Ratio)") + xlab("") +
  theme_light()
```



The figure shows that there is practically no intercept difference between lakes (i.e., the probability of presence at CPUE = 0). This is among other things, probably a result of that the data set unfortunately did not contain any observations from sites with CPUE = 0. There is a strong positive effect of CPUE for both lakes.

We make predictions for probabilities of presence as function of CPUE and between lakes by making a new data frame with CPUE values from 0 to 20 in all combinations with the two lakes.

```
Aph.pred <- expand.grid(CPUE=seq(0, 20, 0.1), Lake=levels(site$Lake))
Aph.pred$Lake <- factor(Aph.pred$Lake)
```

Since `msocc` has no `predict` method, we make predictions by first creating a model matrix `X.Aph` corresponding to `Aph.mod.1` (`~ Lake + CPUE`), and then matrix multiply this by each of the MCMC samples of the model coefficient.

```
X.Aph <- model.matrix( ~ Lake + CPUE, data=Aph.pred)

Aph.z <- matrix(NA, nrow=nrow(X.Aph), ncol=nrow(Aph.beta.1))
for (i in 1:nrow(Aph.beta.1)) {
  Aph.z[, i] <- X.Aph %*% Aph.beta.1[i, ]
}
```

We can then extract medians and 95% percentiles from the resulting z-score matrix (`Aph.z`), and append these columns to the prediction data frame (`Aph.pred`)

```
Aph.pred$z.med <- apply(Aph.z, 1, median)
Aph.pred$z.lcl <- apply(Aph.z, 1, quantile, p=0.025)
Aph.pred$z.ucl <- apply(Aph.z, 1, quantile, p=0.975)
```

We then inverse logit-transform the log(odds) columns to get probabilities

```
Aph.pred$p.med <- 1 / (1 + exp(-Aph.pred$z.med))
Aph.pred$p.lcl <- 1 / (1 + exp(-Aph.pred$z.lcl))
Aph.pred$p.ucl <- 1 / (1 + exp(-Aph.pred$z.ucl))
```

```
ggplot(data=Aph.pred, aes(x=CPUE, y=p.med, group=Lake, color=Lake)) +
  geom_ribbon(data=Aph.pred, aes(ymin=p.lcl, ymax=p.ucl, fill=Lake), alpha=0.3) +
  geom_line(size=2) +
  scale_color_manual(values=c('#990066','#33FF00')) +
  ylab("Pr(Aphanomyces detected)") +
  xlab("Catch per unit effort (CPUE)") +
  theme_light() +
  scale_fill_manual(values = c("#990066", "#33FF00")) +
  theme(text = element_text(size = 20))
```