Case Study 1

This case study was developed primarily by D.Loughnan, X.Wang and E.Wolkovich with the input of the full manuscript author team.

We simulate data using a simple linear function with noise, assuming the sampling of five populations within its distributional range, from north to south for 10 years. The following code performs data analysis and visualization using both the traditional Fisherian approach and the Bayesian approach. For the Fisherian approach, we use an `lm` model and plot the p-values for different populations, indicating whether the increase or decrease is statistically significant. For the Bayesian approach, we use Stan code with priors and plot the posterior distribution.

```
# Use all cores to run the stan model
options(mc.cores = parallel::detectCores())

# Install dependencies
require(rstan)
require(lme4)
require(shinystan)
require(viridis)
```

```
# We set the seed so the results will be consistent
set.seed(1546)

# To simulate data that fits our needs, we use a simple linear function and
# add noise.
simulate_population <- function(a, b, t, noise_sd) {
  y <- numeric(t)
  time <- 1:t
  y <- a*time + b + rnorm(t, 0, noise_sd)
    # y is the population at a certain time
    # a is the increasing/decreasing rate
    # b is the starting population size
```

```
    # time is the x variable
    # noise_sd
  model <- lm(y ~ time)
  # Extract the estimated slope (coefficient of time)
  estimated_slope <- coef(model)["time"]
  pval <- coef(summary(lm(y ~ time)))[["time", "Pr(>|t|)"]]
  return(c(estimated_slope, pval))
}

a <- c(-2000, -1400,  600, 1400 , 2000)
t <- 10
time <- 1:t
b <- seq(40000, 101000, by = 10000)
noise_sd <- c(7000, 6600, 6200, 5800, 5400, 5000,4600)
```

Traditional Fisherian approach

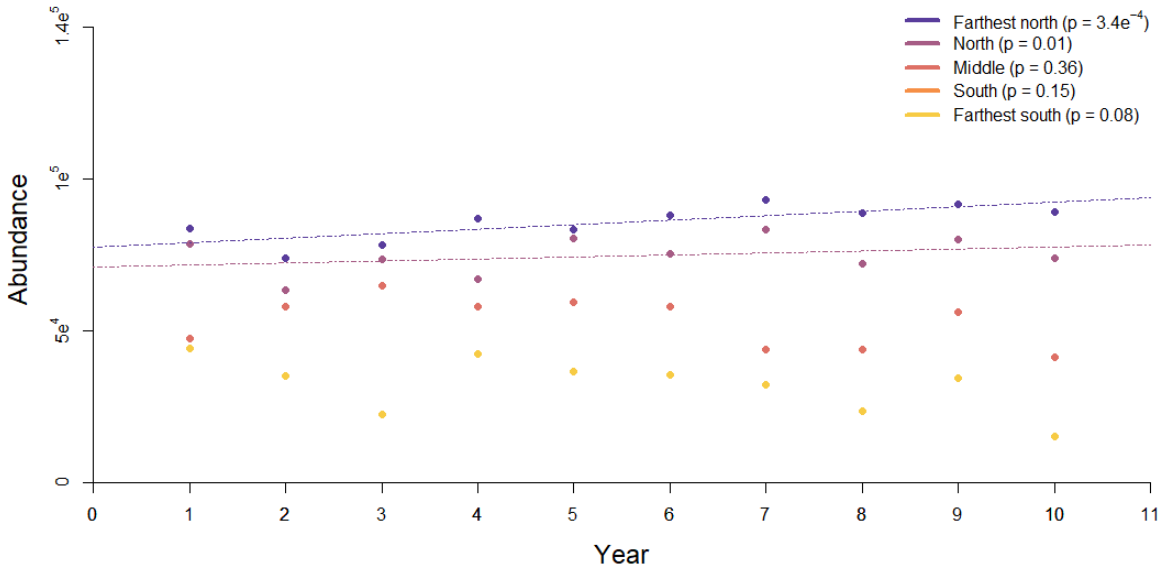
With traditional Fisherian approach, we use lm model to plot

```
output <- data.frame(iter = seq(1:(length(a)*length(time))),
                     pop = rep(1:length(a), each = length(time)),
                     year = rep(1:t, times = length(a)))
output$pred <- NA

abund <- vector()
for(i in 1:length(a)){
  y <- numeric(t)
  time <- 1:t
  y <- a[i]*time + b[i] + rnorm(t, 0, noise_sd[i])
  abund <- rbind(abund, y)
}

abund <- data.frame(reshape2::melt(t(abund)))
output$pred <- abund$value
```

Visualization



Bayesian approach

With Bayesian approach, we run the stan code:

```
data {
  int<lower=0> N; //No. obs
  int<lower=0> Ngrp; //No. in group---population or species
  int group[N]; // Group type
  vector[N] year;
  real ypred[N]; //response
}

parameters {
  real a[Ngrp] ;
  real b[Ngrp];
  real mu_a;
  real<lower=0> sigma_a;
  real mu_b;
  real<lower=0> sigma_b;
```

```
  real<lower=0> sigma_y;

}

model {

real mu_y[N];

for(i in 1:N){
     mu_y[i] = a[group[i]] + b[group[i]] * year[i];
  }

a ~ normal(mu_a, sigma_a);
b ~ normal(mu_b, sigma_b);

//Priors
mu_a ~ normal(188, 50);
sigma_a ~ normal(0,50);
mu_b ~ normal(0,10);
sigma_b ~ normal(0,10);
sigma_y ~ normal(0,10);

ypred ~ normal(mu_y, sigma_y);
}
```

```
str(output)
output$pop <- as.factor(output$pop)

datalistGrp <- with(output,
                    list( N = nrow(output),
                          Ngrp = length(unique(output$pop)),
                          group = as.numeric(as.factor(output$pop)),
                          ypred = output$pred,
                          year = output$year ))
```

```
mdlPop <- stan("partialPoolSimMdl.stan",
               data = datalistGrp)

sum <- summary(mdlPop)$summary

intercept <- sum[grep("a\\[", rownames(sum)), "mean"]
slopes <- sum[grep("b\\[", rownames(sum)), "mean"]
```

```
# Posterior distribution
post <- rstan::extract(mdlPop)
```

Visualization