Using simulated data for FISH 572

FISH 572, Winter 2026

2025-12-22

Introduction

The mid-course project for Fish 572 is to analyze survey data, which may be a data set chosen by the student or simulated data provided by the instructors. Format of the presentation will be open and can include analysis description and graphs or tables. This analysis will be graded independently of the class project, but can be included as part of the final paper. Data analysis will be due at the end of week 6 of the course. Note that you will need to install the R package sdmTMB to execute the model-based analysis portion of this example.

This document describes the expectations of the assignment and how to use one of the simulated datasets. The methods described here may be extended to a real survey dataset.

Assignment expectations

The purpose of this assignment is to become familiar with analysis of survey data. It is expected that standard design-based and model-based estimates will be presented. The paper should provide a brief description of the data used, assumptions used in the analyses, justification of the methods, and results. The assignment is due end of the day **February 16**.

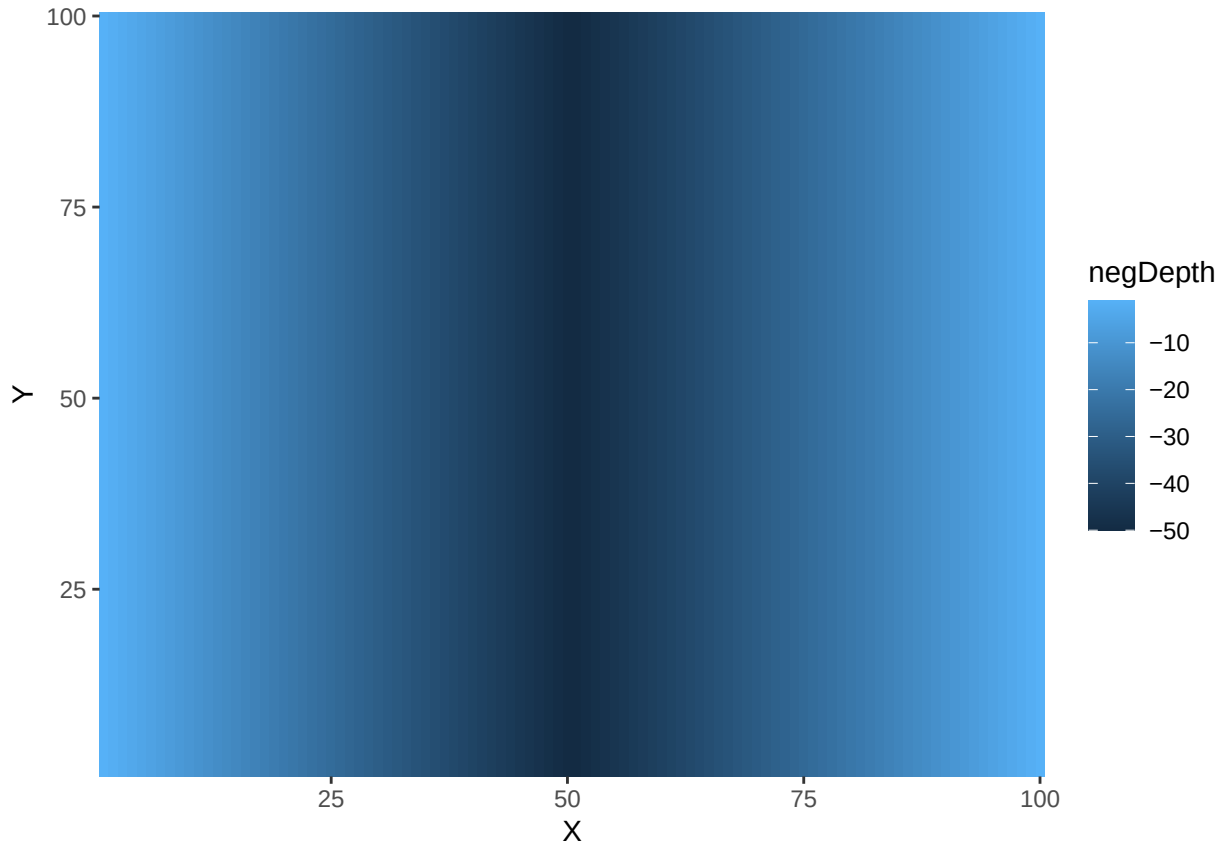
Furthermore, sensitivities to choices made for the analysis are encouraged and can be used for the final project. In particular, the effect of changing observation error and catchability parameters should be evaluated. Consider how you will evaluate the performance of different designs and estimators (e.g., comparing estimates to true values, precision and accuracy of estimators). Examples of analyses may include:

- a detailed comparison of design-based and model-based estimates, highlighting where and why differences occur, and which method is preferable for the data used;
- comparing sampling designs;
  - stratification: a comparison of sampling assumptions including pre-stratification when sampling simulated data, and/or post-stratification for sampled simulated or real data;
  - allocation: a comparison of random and systematic sampling, or other techniques, with or without stratification;
- an investigation of the effect of different assumptions or model settings used in model-based analyses;
- determining how sample size influences results, in conjunction with observation error and variation in catchability;
- other interesting topics from your own creativity.

Simulated data

Data were simulated across a square grid of 100 X 100 cells with depth contours. Here is a plot of the locations on the grid with deeper depths shown in darker blue.

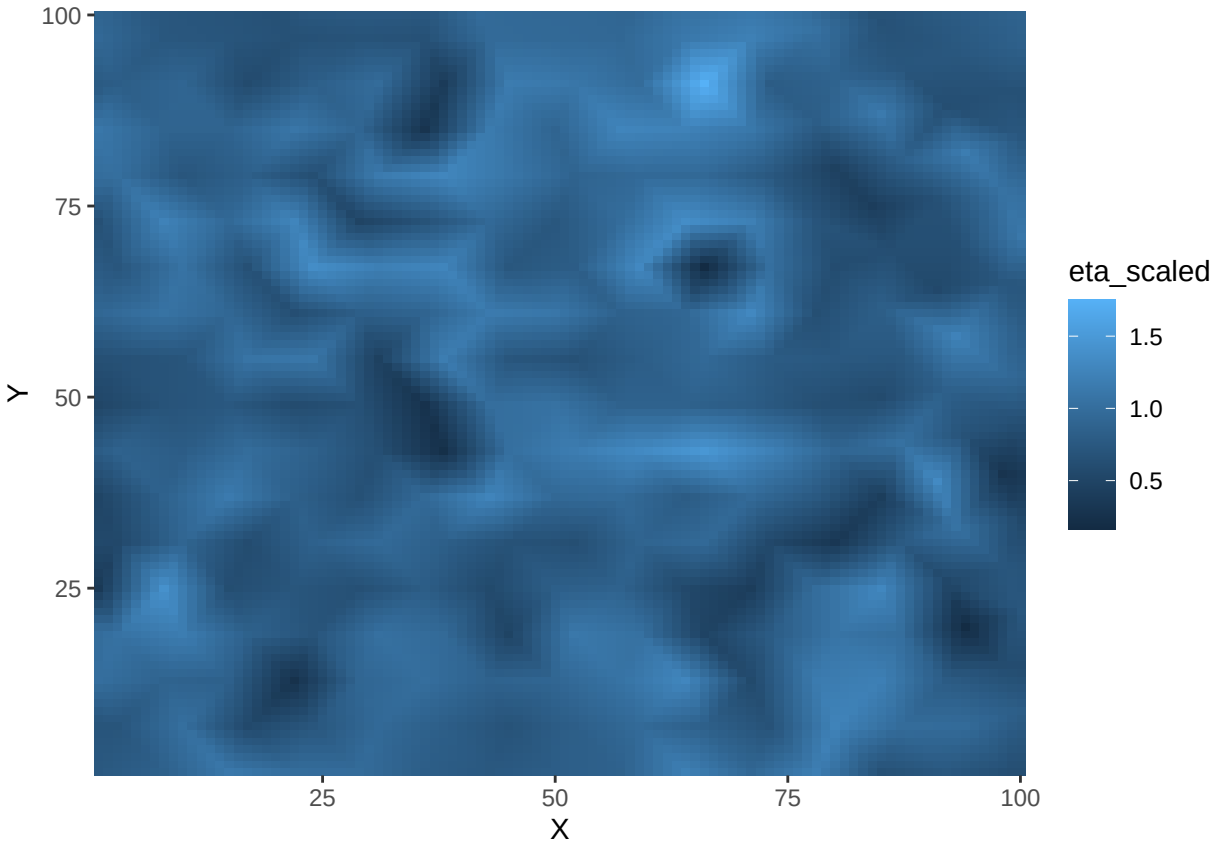
```
# make sure in the correct directory and using the dataset number assigned to you
dat <- readRDS("coursework/simulations/sim_data/sim_dat_1.RDS")
predictor_dat <- readRDS("coursework/simulations/sim_data/grid.RDS")
predictor_dat$negDepth <- -1 * predictor_dat$depth

ggplot(dplyr::filter(predictor_dat, year == 1), aes(X, Y)) +
  geom_raster(aes(fill = negDepth)) +
  scale_color_gradient2() +
  scale_x_continuous(expand = c(0, 0)) + # Remove expansion on x-axis
  scale_y_continuous(expand = c(0, 0)) # Remove expansion on x-axis
```



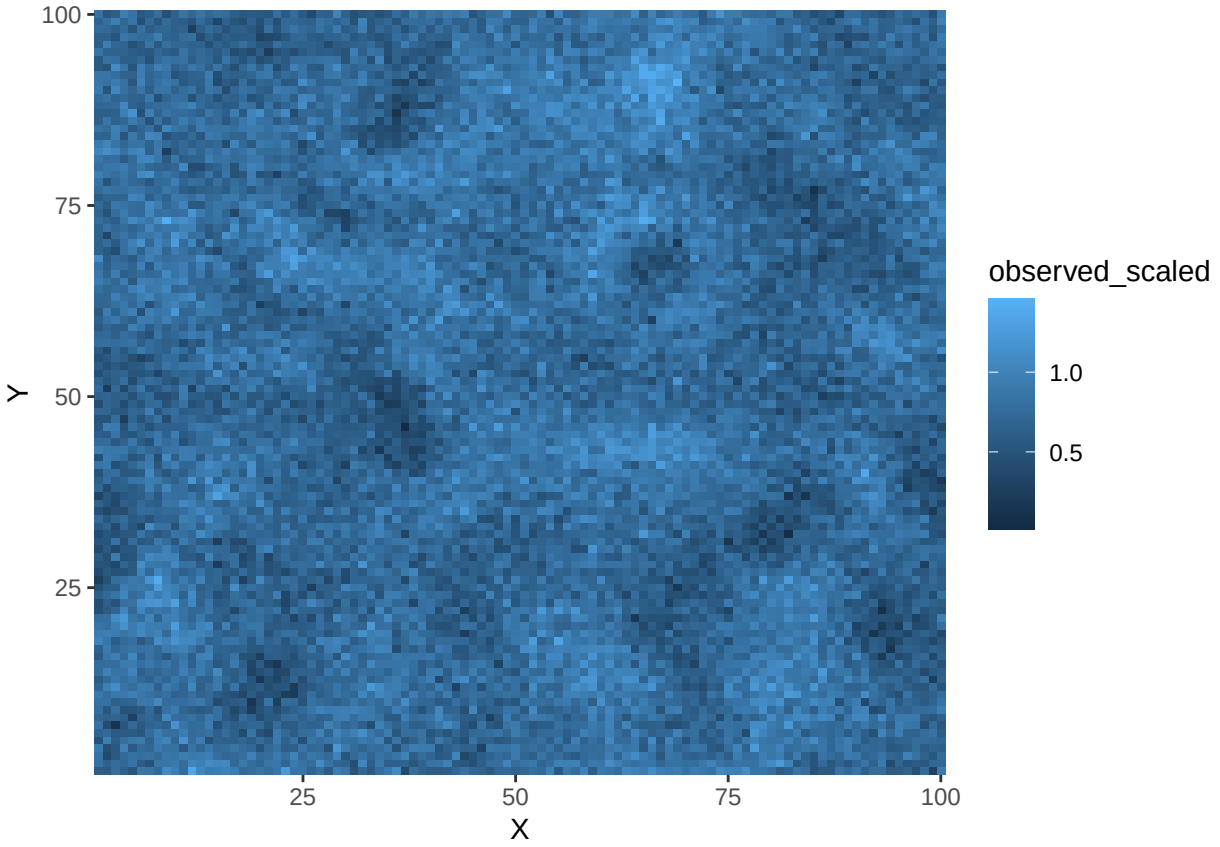
There are eight (8) columns in a simulated dataset: - year: the year, being 1:6; - X: the location along the x-axis; - Y: the location along the y-axis; - eta: the simulated values; - observed: the observed value of the simulated data (i.e. observation error applied to eta); - eta_scaled: the simulated data scaled by the mean of simulated data in a reference simulated dataset; - observed_scaled: the observed data scaled by the mean of observations in a reference simulated dataset; - depth_scaled: the depth scaled by subtracting the mean and dividing by the standard deviation.

```
#plot eta
ggplot(dplyr::filter(dat, year == 1), aes(X, Y)) +
  geom_raster(aes(fill = eta_scaled)) +
```

```
  scale_color_gradient2() +
  scale_x_continuous(expand = c(0, 0)) + # Remove expansion on x-axis
  scale_y_continuous(expand = c(0, 0)) # Remove expansion on x-axis
```




```
#plot observed
ggplot(dplyr::filter(dat, year == 1), aes(X, Y)) +
  geom_raster(aes(fill = observed_scaled)) +
  scale_color_gradient2() +
  scale_x_continuous(expand = c(0, 0)) + # Remove expansion on x-axis
  scale_y_continuous(expand = c(0, 0)) # Remove expansion on x-axis
```

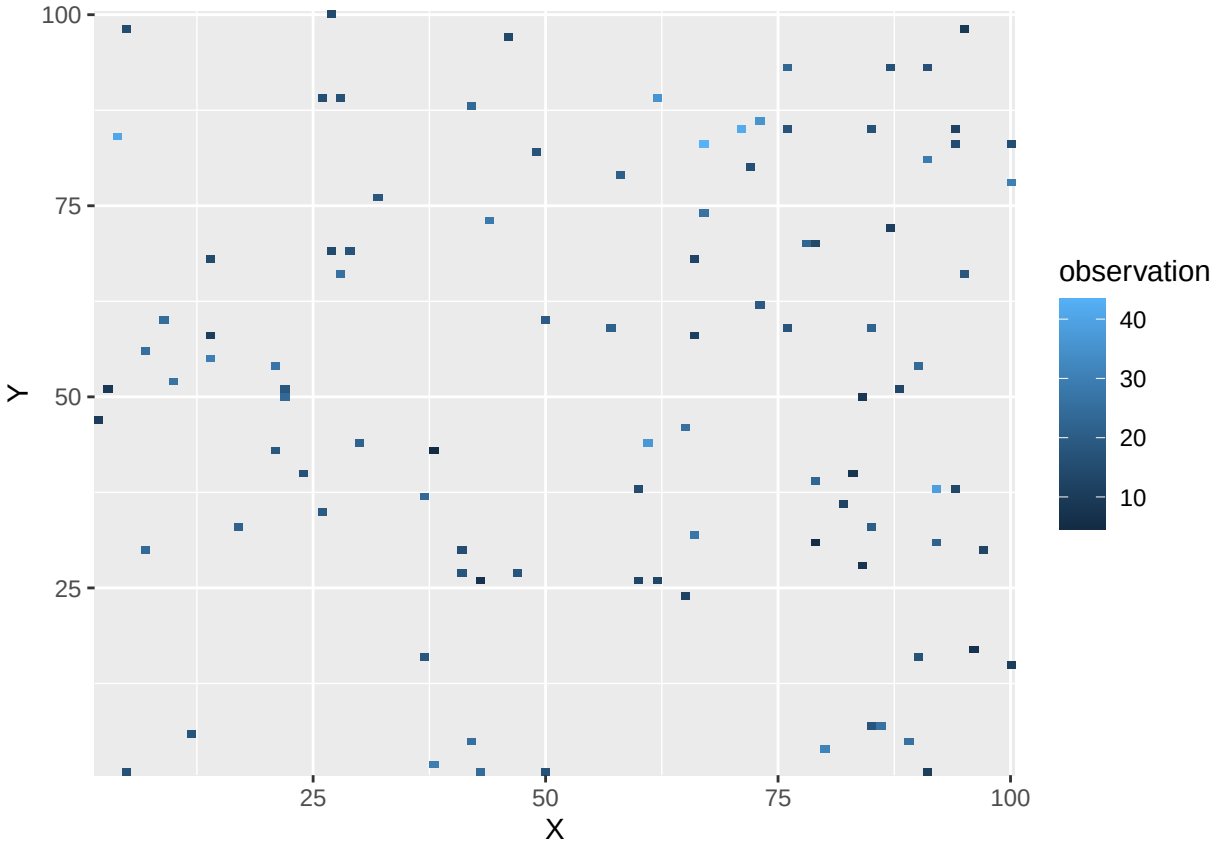


Sampling from the simulated data

One potential use of simulated data is to examine alternative sampling designs. In this example, we sample from the 100 X 100 grid in year 1 using simple random sampling (SRS).

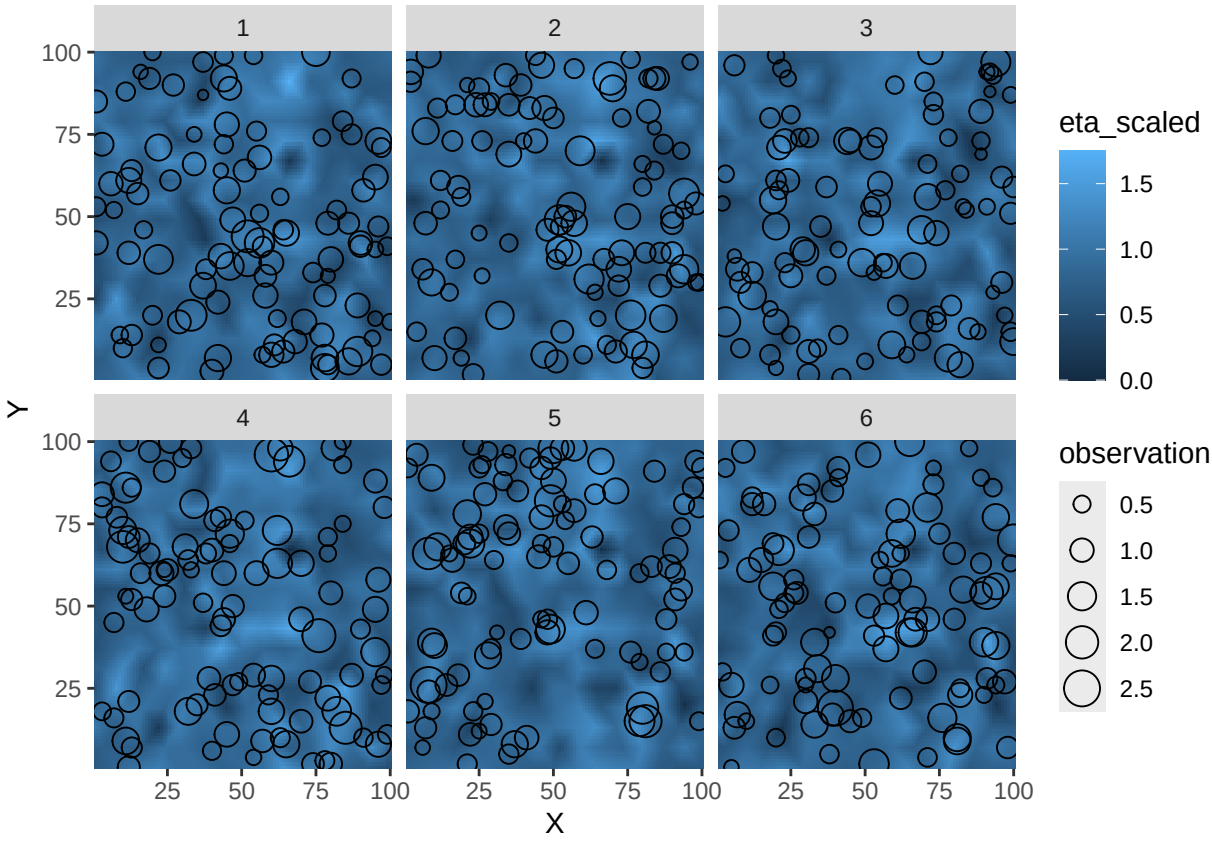
```
source("coursework/simulations/functions_sampling.R")
set.seed(245)
N <- 100 * 100
n <- 100
year <- 1
dat.yr <- dplyr::filter(dat, year == 1)
sampleRowNum <- sample(1:nrow(dat.yr),n,replace=F)
sampleDat <- dat.yr[sampleRowNum,]
# create sample (as if you were a boat sampling once in each sampled grid cell)
obsCV <- 0.2  # if using "observed" set this to zero (observation error already applied)
catchabilityPars <- list(mean=1,var=0.1) #related to gear selectivity (and availability to net)
sample <- sampleGrid.fn(sampleDat, obsCV, catchabilityPars,varName="eta")

#plot sample
ggplot(sample, aes(X, Y)) +
  geom_raster(aes(fill = observation)) +
  scale_color_gradient2() +
  scale_x_continuous(expand = c(0, 0)) + # Remove expansion on x-axis
  scale_y_continuous(expand = c(0, 0)) # Remove expansion on x-axis
```

```
## Warning: Raster pixels are placed at uneven horizontal intervals and will be shifted
## i Consider using `geom_tile()` instead.
## Raster pixels are placed at uneven horizontal intervals and will be shifted
## i Consider using `geom_tile()` instead.
```



Now let's sample all years, while providing a different example of how to sample the data object and view true values and observations.

```
# sample from each year of survey data
sample_dat <- dat |>
  dplyr::group_by(year) |>
  dplyr::slice_sample(n = 100)

samples <- sampleGrid.fn(as.data.frame(sample_dat), obsCV, catchabilityPars, varName="eta_scaled")

# plot samples over surface of mean without observation error (taken as "true")
ggplot2::ggplot(dat, aes(X, Y)) +
  ggplot2::geom_raster(aes(fill = eta_scaled)) +
  ggplot2::geom_point(aes(size = observation), data = samples, pch = 21) +
  ggplot2::facet_wrap(~year) +
  scale_color_gradient2() +
  ggplot2::scale_size_area() +
  ggplot2::coord_cartesian(expand = FALSE)
```



Analysis

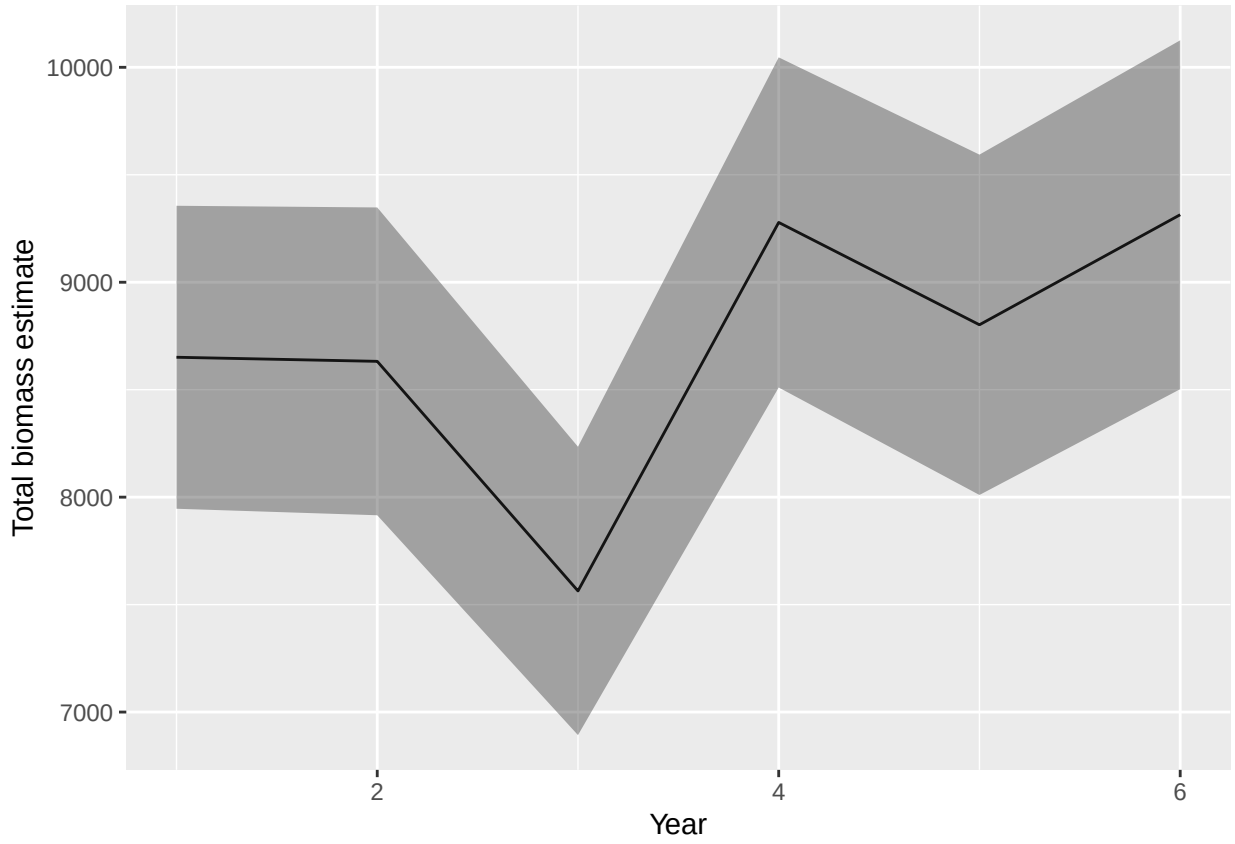
Now that you have your samples, the goal is to estimate the population in all grid cells across the entire sampling area (100 X 100 grid for the simulated data), in all years. In other words, you want to create the full simulated map but you have not sampled all cells, and those that you did sample are subject to uncertainty. There are numerous ways to analyze the data, which can be lumped into `design-based` and `model-based`.

Design-based estimate of abundance

Assuming that the area of each grid cell is 1, under simple random sampling the design-based estimate is the product of the mean of observations and the total area (1 * the number of grid cells in the domain). We then calculate the variance, standard error and 95% confidence interval.

```
index_db <- samples |>
  dplyr::group_by(year) |>
  dplyr::summarize(index = mean(observation) * N,
                   variance = N^2 * (1-(n/N)) * (var(observation)/n)) |>
  dplyr::mutate(se = sqrt(variance),
                lwr = index - qt(0.975, df = n - 1) * se,
                upr = index + qt(0.975, df = n - 1) * se)
```

```
# plot index with 95% CI
ggplot2::ggplot(index_db, aes(year, index)) +
  ggplot2::geom_line() +
  ggplot2::geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.4) +
  ggplot2::xlab('Year') +
  ggplot2::ylab('Total biomass estimate')
```



Model-based estimate of abundance

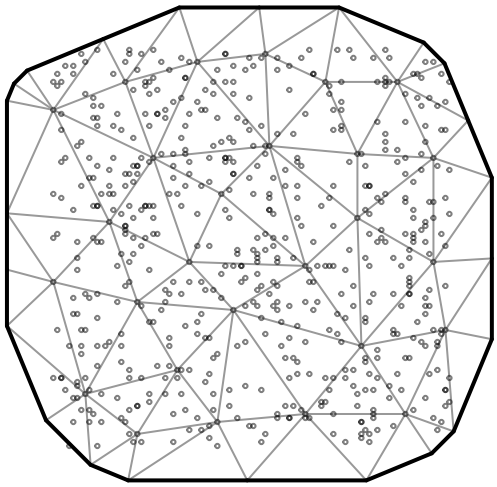
Fit model and check convergence and parameter values

```
library(sdmTMB) # install.packages("sdmTMB", dependencies = TRUE)

mesh <- sdmTMB::make_mesh(samples,
                          xy_cols = c("X", "Y"),
                          type = "cutoff_search",
                          n_knots = 50)
```

```
## cutoff = 1.00 | knots = 1029 | v
## cutoff = 100.00 | knots = 24 | ^
## cutoff = 10.00 | knots = 91 | v
## cutoff = 31.62 | knots = 34 | ^
```

```
## cutoff = 17.78 | knots = 46 | ^
## cutoff = 13.34 | knots = 61 | v
## cutoff = 15.40 | knots = 53 | v
## cutoff = 16.55 | knots = 49 | ^
## cutoff = 15.96 | knots = 50 | v
```

```
plot(mesh)
```



```
fit <- sdmTMB::sdmTMB(
  formula = observation ~ 0 + as.factor(year),
  data = samples,
  mesh = mesh,
  time = "year",
  family = sdmTMB::lognormal(),
  spatial = "on", # c("on", "off")
  spatiotemporal = "iid", # c("iid", "ar1", "rw", "off")
)

fit
```

```
## Spatiotemporal model fit by ML ['sdmTMB']
## Formula: observation ~ 0 + as.factor(year)
## Mesh: mesh (isotropic covariance)
## Time column: year
## Data: samples
```

```
## Family: lognormal(link = 'log')
##
## Conditional model:
##                  coef.est coef.se
## as.factor(year)1    -0.18    0.05
## as.factor(year)2    -0.16    0.05
## as.factor(year)3    -0.29    0.05
## as.factor(year)4    -0.08    0.05
## as.factor(year)5    -0.14    0.05
## as.factor(year)6    -0.11    0.05
##
## Dispersion parameter: 0.42
## Matérn range: 3.29
## Spatial SD: 0.85
## Spatiotemporal IID SD: 0.00
## ML criterion at convergence: 202.272
##
## See ?tidy.sdmTMB to extract these values as a data frame.
##
## **Possible issues detected! Check output of sanity().**
```

```
sanity(fit) # model checking
```

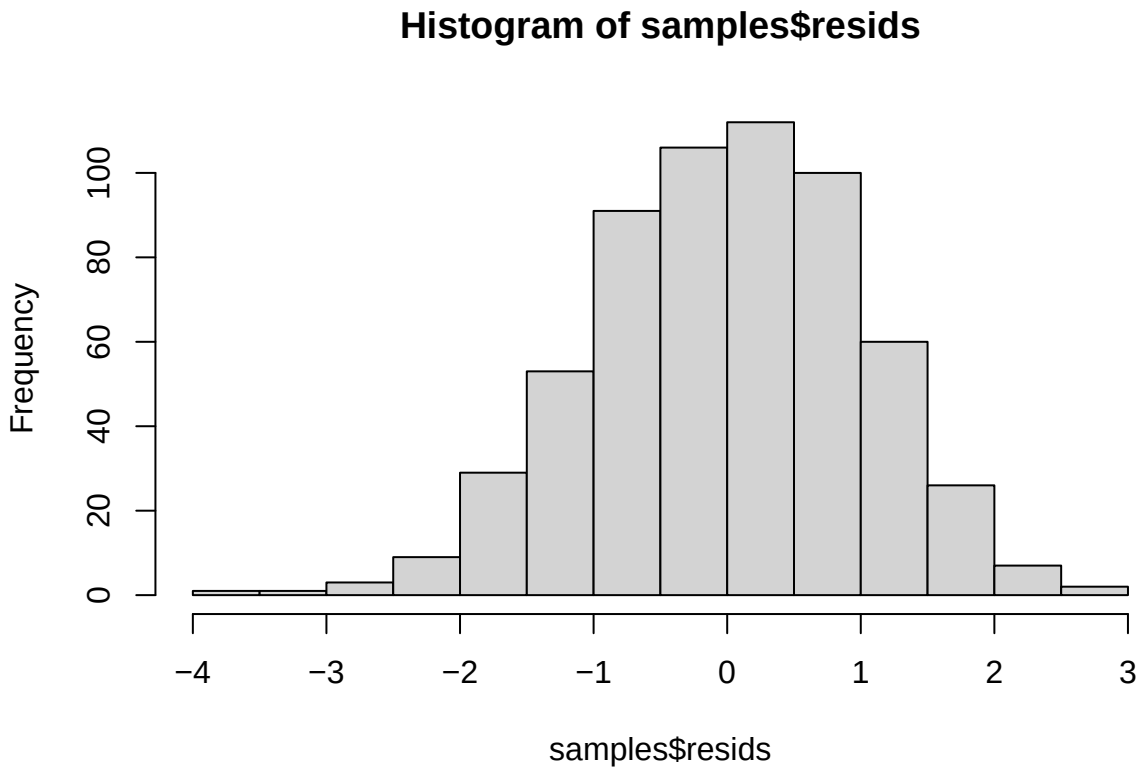
```
## v Non-linear minimizer suggests successful convergence

## v Hessian matrix is positive definite

## v No extreme or very small eigenvalues detected

## v No gradients with respect to fixed effects are >= 0.001

## v No fixed-effect standard errors are NA

## x `ln_tau_E` standard error may be large

## i `ln_tau_E` is an internal parameter affecting `sigma_E`

## i `sigma_E` is the spatiotemporal standard deviation

## i Try simplifying the model, adjusting the mesh, or adding priors

## x `sigma_E` is smaller than 0.01

## i Consider omitting this part of the model

## v Range parameter doesn't look unreasonably large
```

Inspect model for goodness of fit

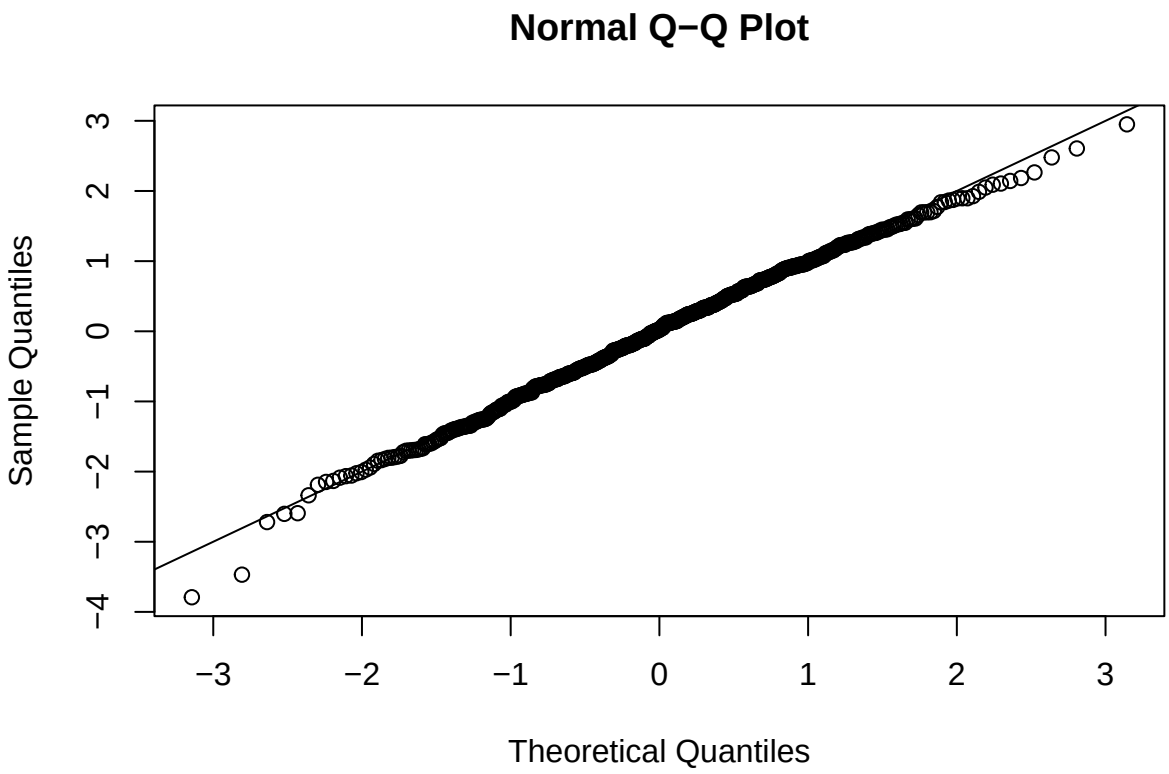
```r
# randomized quantile residuals
samples$resids <- residuals(fit)
```

```
## Note what used to be the default sdmTMB residuals (before version 0.4.3.9005)
## are now 'type = 'mle-eb''. We recommend using the current default ''mle-mvn'',
## which takes one sample from the approximate posterior of the random effects or
## 'dharma_residuals()' using a similar approach.
```

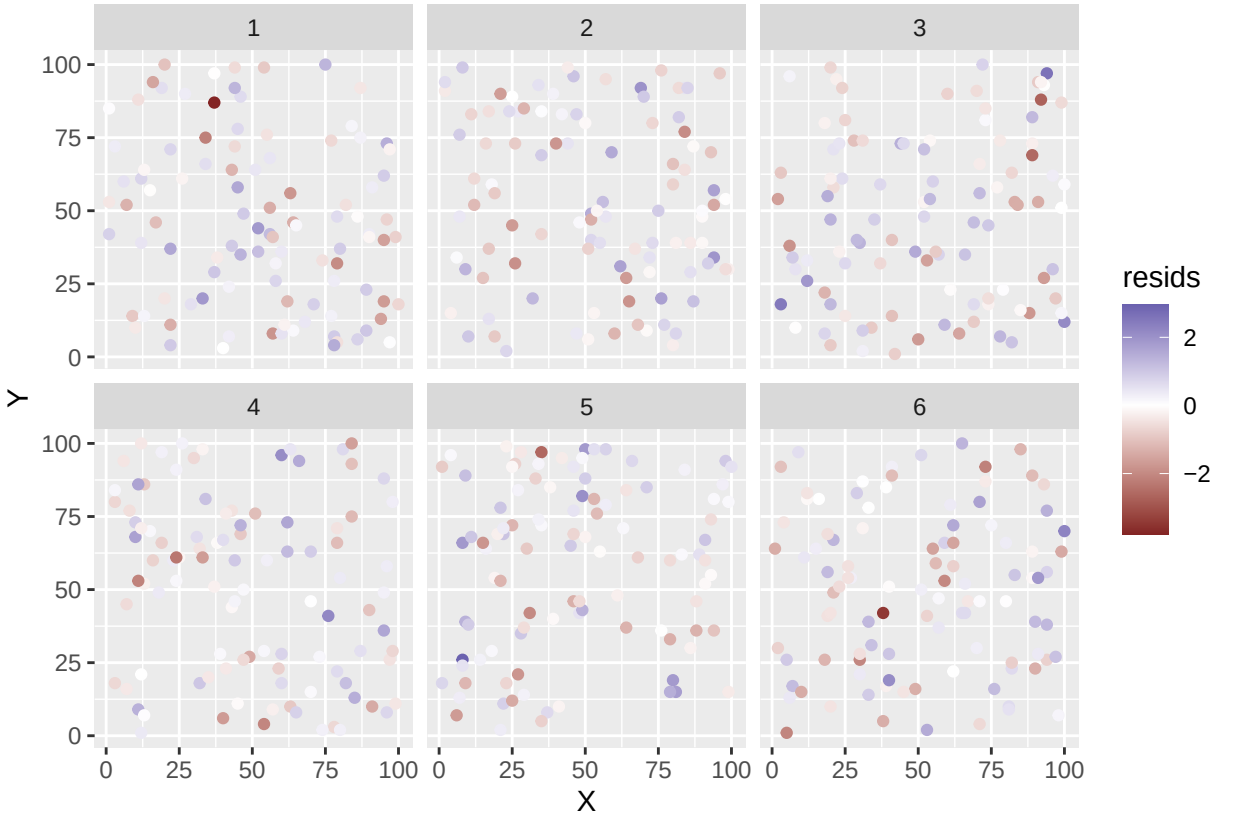
```r
hist(samples$resids)
```




```r
# qq plot
qqnorm(samples$resids)
abline(a = 0, b = 1)
```



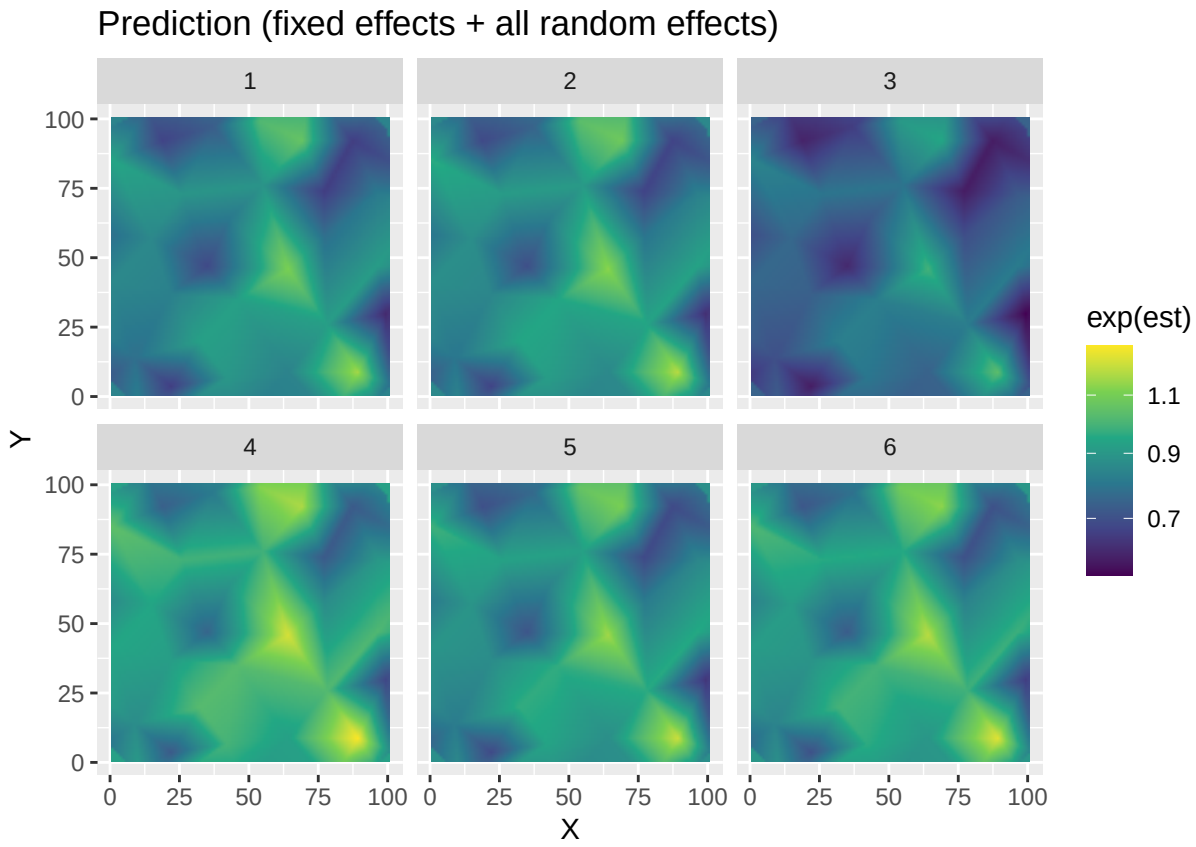

```
# spatial residuals
ggplot(samples, aes(X, Y, col = resids)) + scale_colour_gradient2() +
  geom_point() + facet_wrap(~year) + coord_fixed()
```



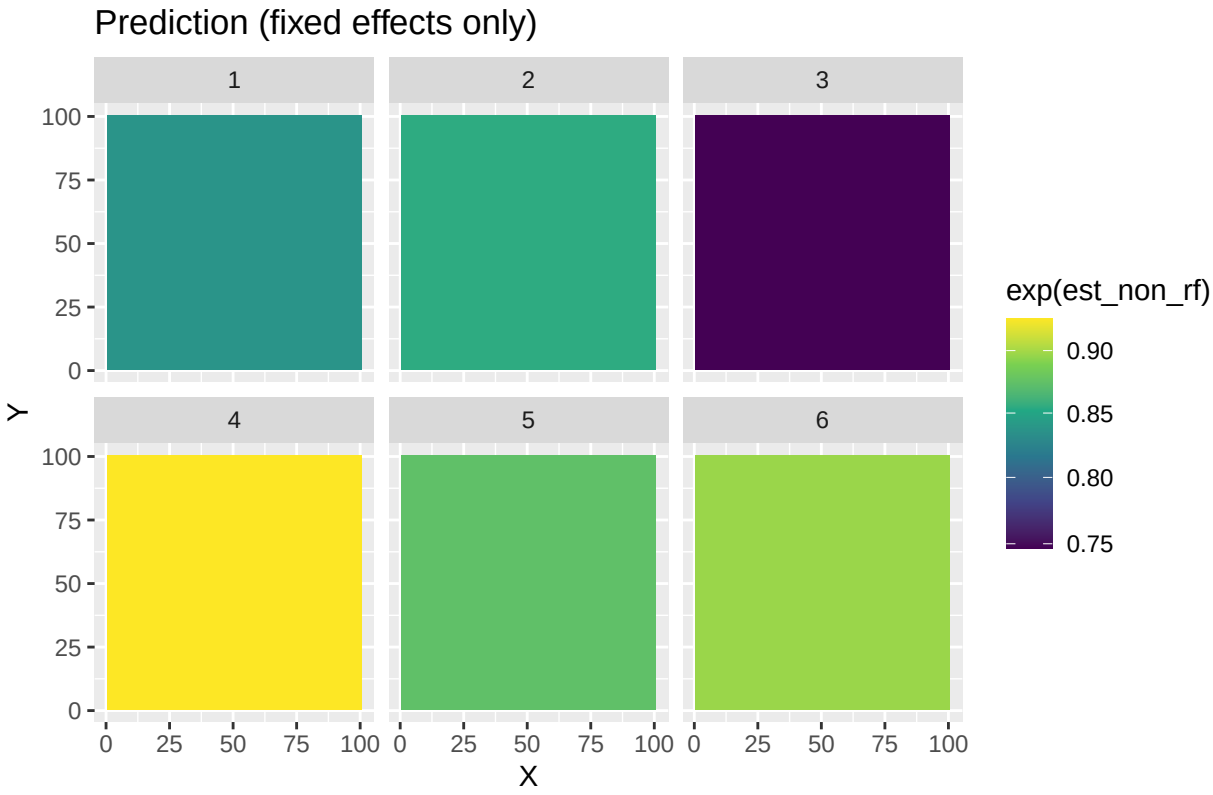
Predict from model to all locations and years

Predict to grid of full survey domain and map predictions and each component of the predictions (fixed and random effects) to see contributions of each to the full prediction.

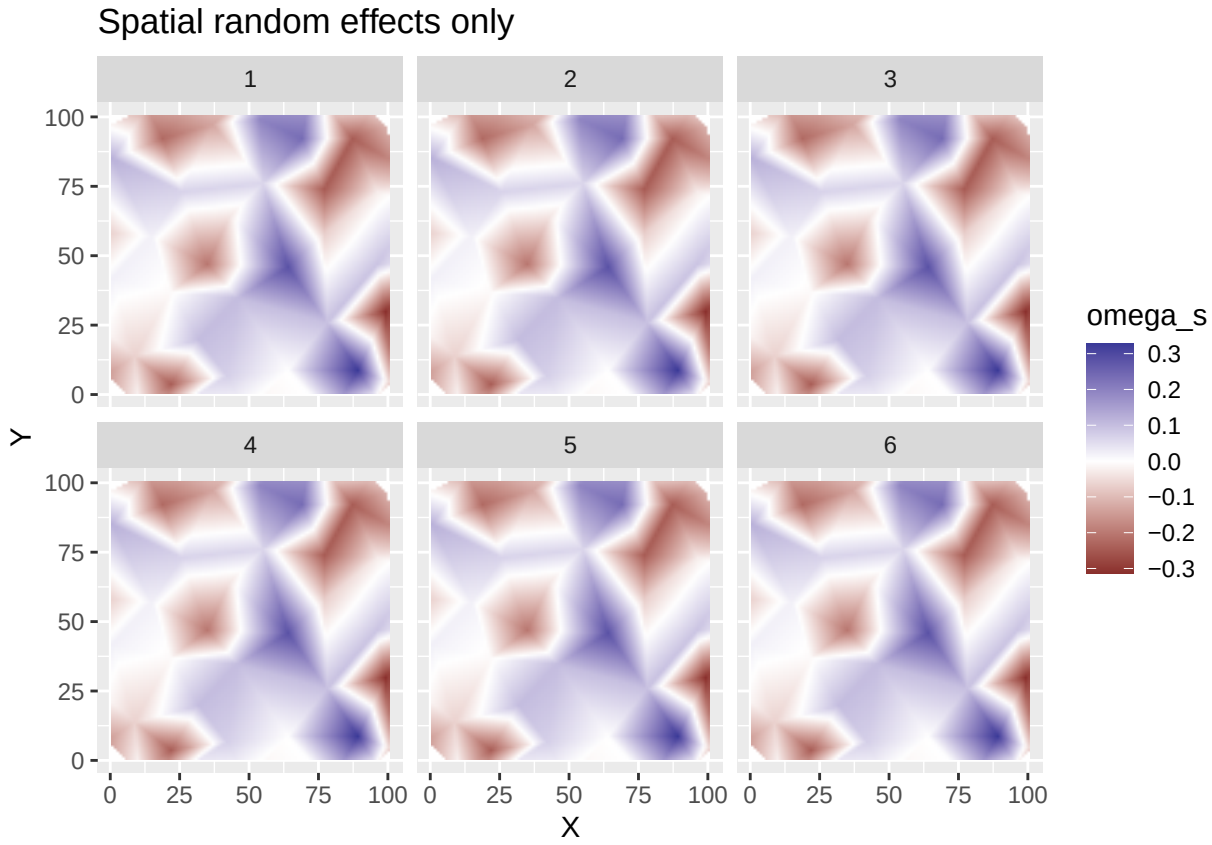
```
# predict
predictions <- stats::predict(fit, newdata = predictor_dat,
                              return_tmb_object = TRUE)

# visualize predictions, then how fixed and random effects contribute
plot_map <- function(dat, column) { ggplot(dat, aes(X, Y, fill = {{ column }})) +
    geom_raster() + facet_wrap(~year) + coord_fixed() }

# full prediction
plot_map(predictions$data, exp(est)) +
  scale_fill_viridis_c(trans = "sqrt") +
  ggtitle("Prediction (fixed effects + all random effects)")
```



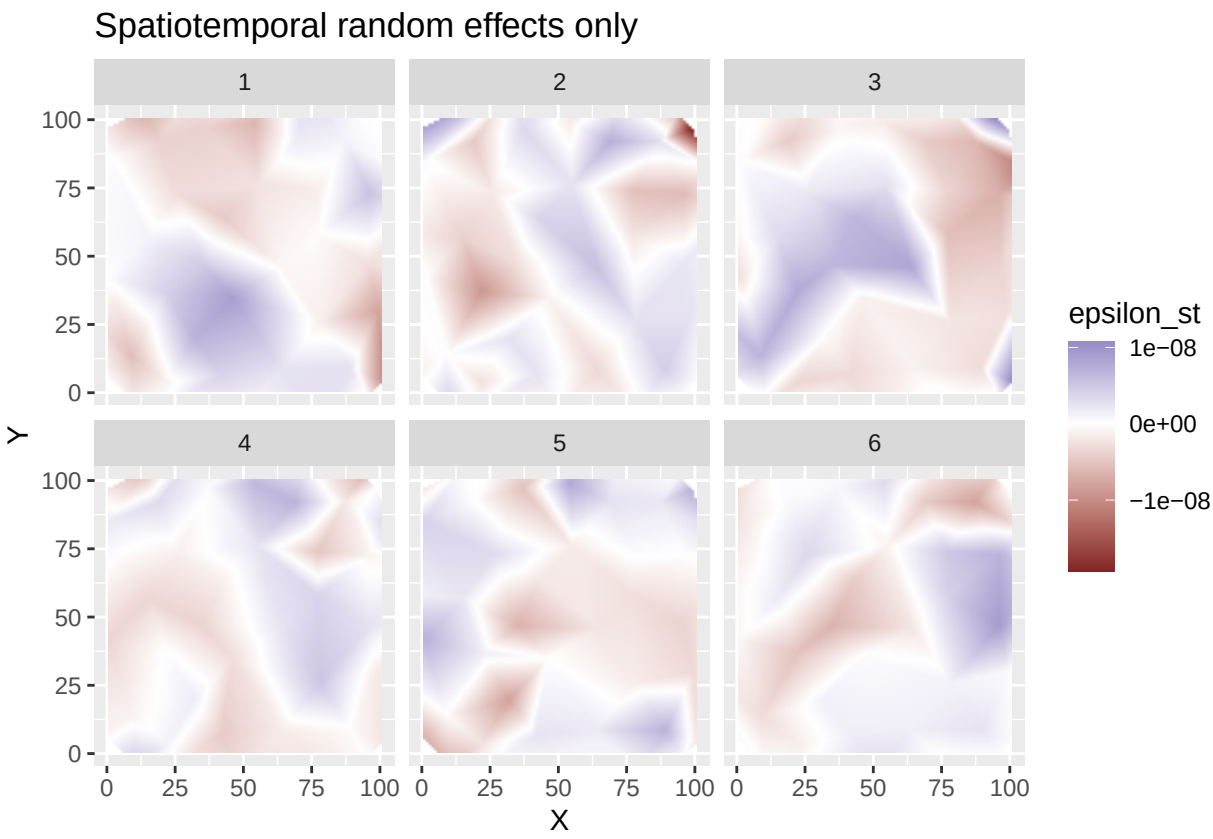

```
# fixed effects only (year)
plot_map(predictions$data, exp(est_non_rf)) +
  ggtitle("Prediction (fixed effects only)") +
  scale_fill_viridis_c(trans = "sqrt")
```




```
# spatial random effects
plot_map(predictions$data, omega_s) +
  ggtitle("Spatial random effects only") +
  scale_fill_gradient2()
```

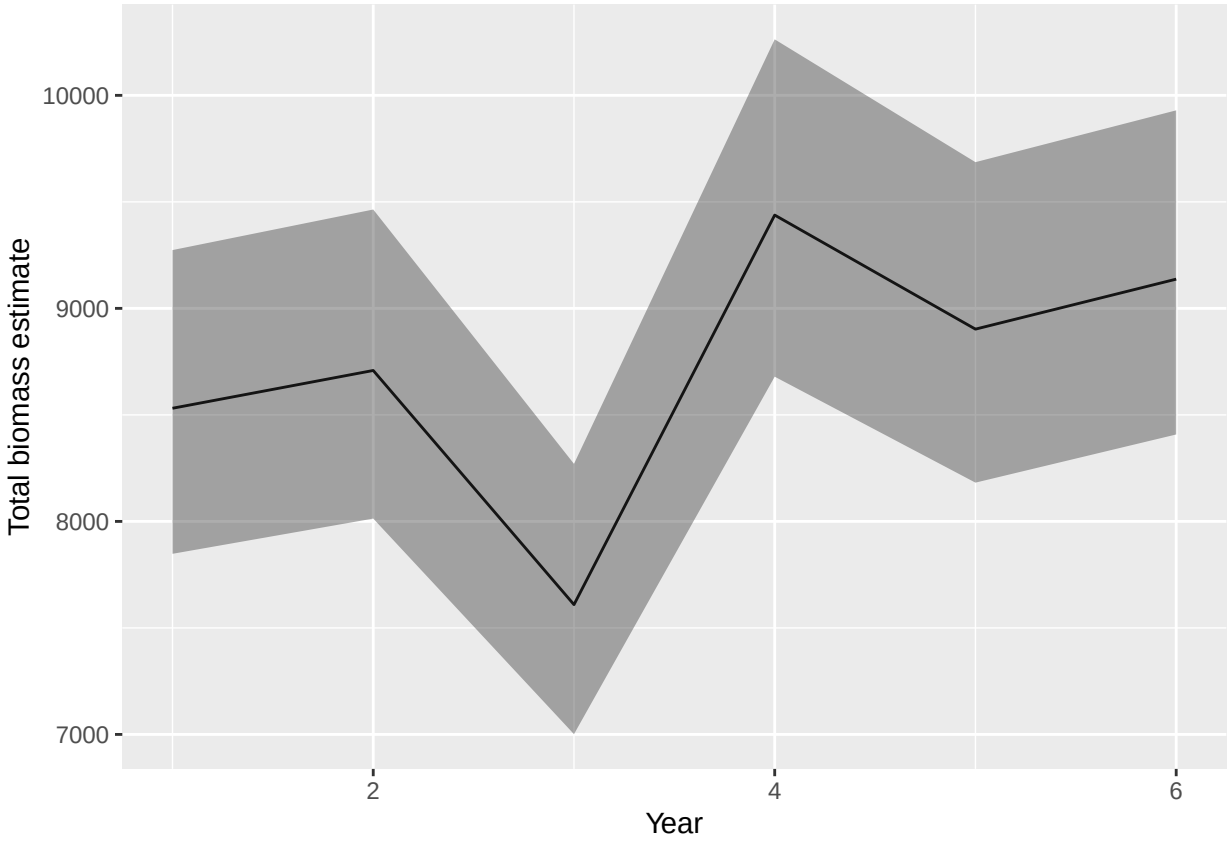



```
# spatiotemporal random effects
plot_map(predictions$data, epsilon_st) +
  ggtitle("Spatiotemporal random effects only") +
  scale_fill_gradient2()
```



Compute abundance index

```
# we will assume that the area of each grid cell is 1
index <- sdmTMB::get_index(predictions, area = 1, bias_correct = TRUE,
                           level = 0.95) # desired confidence interval here is 95%

# plot index with 95% CI
ggplot2::ggplot(index, aes(year, est)) +
  ggplot2::geom_line() +
  ggplot2::geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.4) +
  ggplot2::xlab('Year') +
  ggplot2::ylab('Total biomass estimate')
```


10000
9000
8000
7000
Total biomass estimate
2
4
6
Year