Comparing VAST and sdmTMB GOA indices

Contents

```r
#remotes::install_github("pbs-assess/sdmTMB", dependencies = TRUE)
library(VAST)
library(sp)
library(sdmTMB)
library(dplyr)
library(ggplot2)
library(here)

species <- "Sebastes_polyspinis"
#Gadus_macrocephalus Sebastes_alutus Sebastes_polyspinis Sebastes_variabilis
```

We will fit geostatistical spatiotemporal models with VAST and sdmTMB for the purposes of index standardization and compare the outputs given the same data. We will use data from the GOA AFSC GAP bottom trawl survey for the species specified above. The density units are kg/km^2.

```r
dat_ll <- readRDS(here("species_specific_code", "GOA", species, "production",
                       "data", paste0("Data_Geostat_", species, ".rds")))

dat_ll <-  dplyr::transmute(dat_ll,
                        cpue_kg_km2 = Catch_KG / AreaSwept_km2, #or Catch_KG
                        year = as.integer(Year),
                        vessel = "missing",
                        effort = 1, #AreaSwept_km2 if modeling Catch_KG
                        lat = Lat,
                        lon = Lon,
                        pass = 0) %>%
                as.data.frame() # ensure not a tibble
```

We begin by specifying the VAST model. To specify the mesh used to approximate the spatial process, which is used in the SPDE calculations, we use the k-means method in VAST. Rather than specifying the cutoff distance, meshes in VAST are typically generated by specifying only the number of knots, which we will later pass, along with other model settings to the function make_settings. We will use 750 knots, the same number in the mesh created in the existing production VAST index for this stock and region.

We will include a factor predictor that represents the mean estimate for each time slice. Settings used for index standardization are applied by specifying `purpose = "index2"`.

Unlike in sdmTMB, the fitting and predicting steps are all accomplished with the function `fit_model()` and thus we need to specify the prediction grid (referred to as the "extrapolation grid" in VAST). Here, X and Y are coordinates in UTM zone 5.

```r
GOAgrid <- read.csv(here("extrapolation_grids", "GOAThorsonGrid_Less700m.csv"))
input_grid <- cbind(Lat=GOAgrid$Latitude,
```

```r
                     Lon=GOAgrid$Longitude,
                     Area_km2=GOAgrid$Shape_Area/1000000)

settings <- make_settings(
  n_x = 750, # number of vertices in the SPDE mesh
  Region = "user",
  purpose = "index2", # index of abundance with Gamma for positive catches
  fine_scale = TRUE, # use bilinear interpolation from the INLA 'A' matrix
  zone = NA, # detects automatically
  Options = c("Calculate_Range" = TRUE, "Calculate_effective_area" = TRUE,
              "treat_nonencounter_as_zero" = FALSE),
  ObsModel = c(2, 1), # conventional logit-linked delta-Gamma; (2,4) if there are years with 100% encou
  bias.correct = TRUE,
  use_anisotropy = TRUE,
  max_cells = Inf, # use all grid cells from the extrapolation grid, production model used 2000
  knot_method = "grid", # or "samples"
  strata.limits = data.frame(STRATA = as.factor('All_areas')) # customize to sp.
)
```

Next we will fit a GLMM (generalized linear mixed effects model).

```r
# create folder for saved output:
dir.create(paste0(here("species_specific_code", "GOA", species,
                       "index_comparison")), showWarnings = FALSE)

f <- here("species_specific_code", "GOA", species, "index_comparison", "VASTfit.RDS")
if (!file.exists(f)) {
  fit <- fit_model(
    settings = settings,
    Lat_i = dat_ll[, "lat"],
    Lon_i = dat_ll[, "lon"],
    t_i = dat_ll[, "year"],
    b_i = dat_ll[, "cpue_kg_km2"],
    a_i = dat_ll[, "effort"],
    input_grid = input_grid,
    working_dir = paste0(here("species_specific_code", "GOA",
                              species, "index_comparison"), "/")
  )
  saveRDS(fit, file = f)
} else {
  fit <- readRDS(f)
  fit <- reload_model(fit)
}
#> Maximum absolute gradient of 2.66e-06: No evidence of non-convergence
```

We can look at parameter estimates. First we see estimates from the binomial component and second we see estimates from the positive Gamma component.

```r
fit$parameter_estimates$diagnostics
#>          Param starting_value     Lower        MLE     Upper final_gradient
#> 1    ln_H_input      0.4174934 -5.000000  0.4174947  5.000000   5.504432e-10
#> 2    ln_H_input      0.1748220 -5.000000  0.1748334  5.000000   2.271670e-09
#> 3      beta1_ft     -3.7578257      -Inf -3.7576014       Inf   2.967232e-08
```

```
#> 4       beta1_ft     -3.6977887      -Inf -3.6975664       Inf   1.672424e-08
#> 5       beta1_ft     -4.0050664      -Inf -4.0048440       Inf   1.429038e-08
#> 6       beta1_ft     -3.8623219      -Inf -3.8621354       Inf   1.046794e-08
#> 7       beta1_ft     -3.9447549      -Inf -3.9445511       Inf  -1.868967e-09
#> 8       beta1_ft     -4.0203246      -Inf -4.0200939       Inf   2.077973e-08
#> 9       beta1_ft     -3.8996096      -Inf -3.8994057       Inf   1.478962e-09
#> 10      beta1_ft     -4.1443365      -Inf -4.1441146       Inf   1.470996e-09
#> 11      beta1_ft     -4.2088324      -Inf -4.2086238       Inf  -5.648474e-09
#> 12      beta1_ft     -4.3412286      -Inf -4.3409803       Inf   1.525753e-08
#> 13      beta1_ft     -4.2586403      -Inf -4.2584297       Inf  -5.253091e-09
#> 14      beta1_ft     -4.5058591      -Inf -4.5056558       Inf  -1.279422e-08
#> 15      beta1_ft     -4.0518418      -Inf -4.0515957       Inf   1.692350e-08
#> 16      beta1_ft     -4.3233418      -Inf -4.3231844       Inf  -2.291975e-08
#> 17      beta1_ft     -4.3674685      -Inf -4.3672578       Inf   4.814220e-10
#> 18      beta1_ft     -4.6899421      -Inf -4.6897285       Inf  -5.034466e-09
#> 19    L_omega1_z      2.5637220      -Inf  2.5635199       Inf  -4.449757e-07
#> 20 L_epsilon1_z      0.4346413      -Inf  0.4346509       Inf  -2.702242e-09
#> 21     logkappa1     -3.7111098 -6.765487 -3.7110271 -1.659642  -4.428867e-08
#> 22      beta2_ft      5.0681523      -Inf  5.0681148       Inf  -4.972414e-10
#> 23      beta2_ft      4.5199044      -Inf  4.5199022       Inf   3.712550e-10
#> 24      beta2_ft      5.5513791      -Inf  5.5513691       Inf   1.212904e-09
#> 25      beta2_ft      5.0009813      -Inf  5.0009935       Inf  -1.258165e-09
#> 26      beta2_ft      5.8387362      -Inf  5.8387141       Inf  -4.200622e-10
#> 27      beta2_ft      5.0961360      -Inf  5.0961104       Inf   2.200906e-11
#> 28      beta2_ft      5.5564377      -Inf  5.5564149       Inf  -1.846338e-09
#> 29      beta2_ft      5.5500343      -Inf  5.5500226       Inf   1.406914e-09
#> 30      beta2_ft      5.1191376      -Inf  5.1191161       Inf   3.262457e-10
#> 31      beta2_ft      5.3324650      -Inf  5.3324400       Inf   1.134715e-09
#> 32      beta2_ft      5.7962440      -Inf  5.7962282       Inf   1.181757e-10
#> 33      beta2_ft      5.1732344      -Inf  5.1732063       Inf  -6.645564e-10
#> 34      beta2_ft      5.3091109      -Inf  5.3090777       Inf   1.971685e-10
#> 35      beta2_ft      5.3920591      -Inf  5.3920381       Inf  -1.931621e-09
#> 36      beta2_ft      5.2244991      -Inf  5.2244260       Inf  -2.577886e-09
#> 37      beta2_ft      4.8342989      -Inf  4.8341840       Inf  -5.607680e-09
#> 38    L_omega2_z      2.1889218      -Inf  2.1889141       Inf  -2.255979e-08
#> 39 L_epsilon2_z      1.8126569      -Inf  1.8126632       Inf  -2.420039e-08
#> 40     logkappa2     -2.5207541 -6.765487 -2.5207509 -1.659642   4.866340e-08
#> 41     logSigmaM      0.1941700      -Inf  0.1941823 10.000000  -2.801588e-08
```

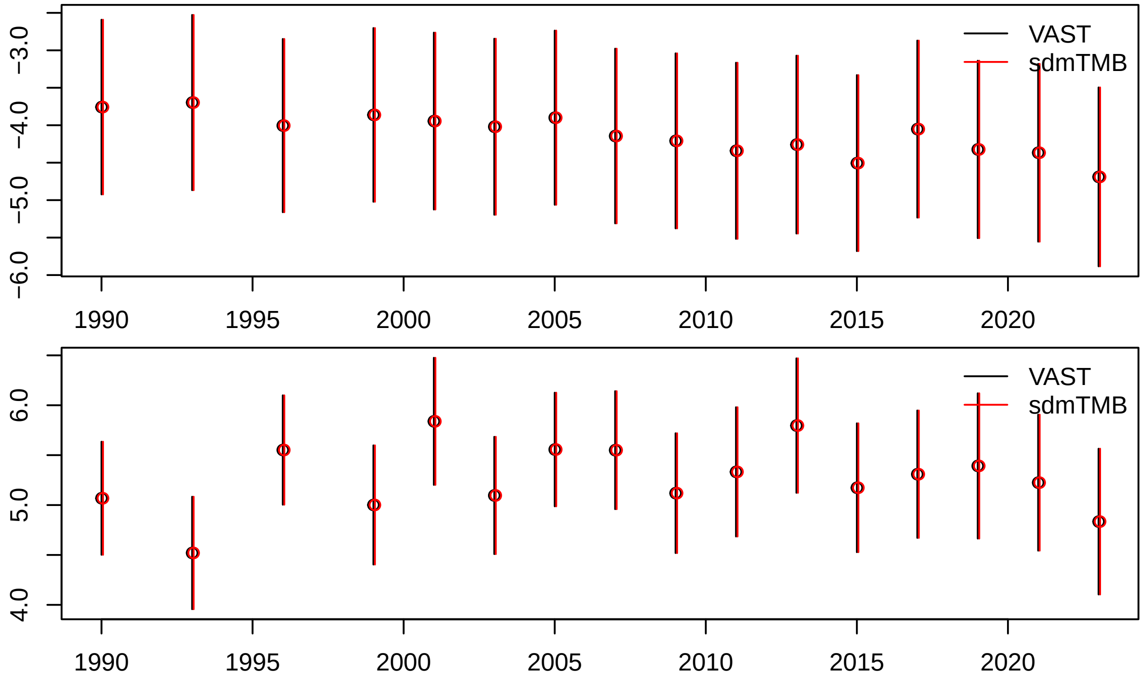
Now we fit the same model in sdmTMB:

```
dat <- dat_ll %>%
  rename(X = lon, Y = lat)

dat$year_f <- as.factor(dat$year)

coordinates(dat) <- ~ X + Y
proj4string(dat) <- CRS("+proj=longlat +datum=WGS84")
dat <- as.data.frame(spTransform(dat, CRS("+proj=utm +zone=5")))
# scale to km so values don't get too large
dat$X <- dat$coords.x1 / 1000
dat$Y <- dat$coords.x2 / 1000

f1 <- here("species_specific_code", "GOA", species,
```

```r
          "index_comparison", "fit_sdmTMB.RDS")
if (!file.exists(f1)) {
# make mesh and fit model
mesh <-  make_mesh(dat, xy_cols = c("X", "Y"), mesh = fit$spatial_list$MeshList$anisotropic_mesh) #pass
#mesh <-  make_mesh(dat, xy_cols = c("X", "Y"), n_knots = 50, type = "kmeans") #coarser mesh for experi

fit_sdmTMB <- sdmTMB(
  cpue_kg_km2 ~ 0 + year_f,
  data = dat,
  mesh = mesh,
  family = delta_gamma(type = "poisson-link"),
  time = "year",
  spatial = "on",
  spatiotemporal = "iid",
  silent = FALSE,
  anisotropy = TRUE,
  do_fit = TRUE
  #, do_index = TRUE (to compute index at same time, requires passing args)
)
fit_sdmTMB
saveRDS(fit_sdmTMB, file = here("species_specific_code", "GOA",
                                species, "index_comparison",
                                "fit_sdmTMB.RDS"))
} else {
fit_sdmTMB <- readRDS(f1)
}

# diagnose estimation issues due to model structure
#TMBhelper::check_estimability(fit_sdmTMB$tmb_obj)
```

We wrote some custom code to extract comparable parameters (not shown above). Here are the annual mean estimates in link space with 95% confidence intervals for the two components to the delta model:

```r
par(mfrow = c(2, 1), cex = 0.8, mar = c(1.5, 1, 1, 1), oma = c(2, 3, 1, 1))
plot_betas(fit, fit_sdmTMB, "beta1_ft", sdmTMB_pars = 1)
plot_betas(fit, fit_sdmTMB, "beta2_ft", sdmTMB_pars = 2)
```



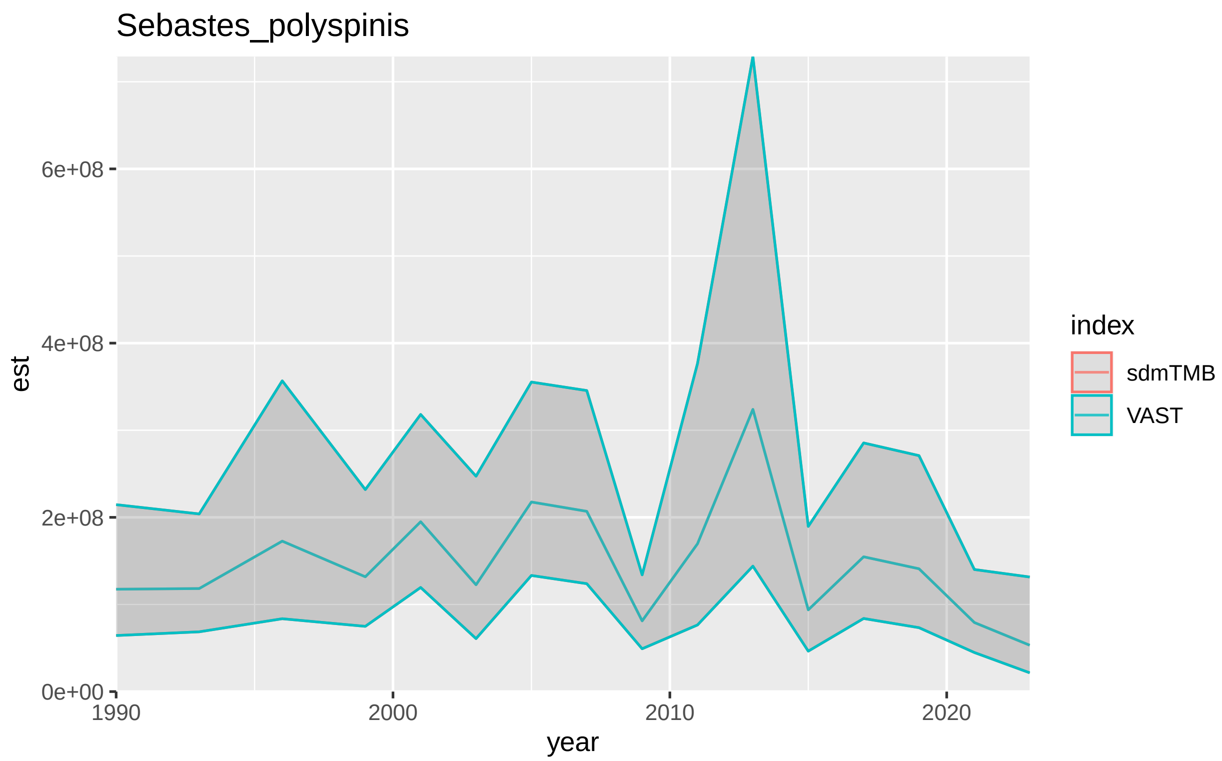
We can compare the index we would get using sdmTMB.

```r
# prep prediction grid and transform to UTM projection
grid_ll <- as.data.frame(input_grid)
names(grid_ll) <- tolower(names(grid_ll))
coordinates(grid_ll) <- ~ lon + lat
proj4string(grid_ll) <- CRS("+proj=longlat +datum=WGS84")
grid <- as.data.frame(spTransform(grid_ll, CRS("+proj=utm +zone=5")))

# rename and scale to km so values don't get too large
grid$X <- grid$coords.x1 / 1000
grid$Y <- grid$coords.x2 / 1000

# or with sf:
# grid_ll <- sf::st_as_sf(
#   x = grid_ll,
#   coords = c("lon", "lat"),
#   crs = "+proj=longlat +datum=WGS84"
# )
# grid <- sf::st_transform(grid_ll, crs = "+proj=utm +zone=5")

# replicate extrapolation grid for each year in data
pred_grid <- replicate_df(grid, "year_f", unique(dat$year_f))
pred_grid$year <- as.integer(as.character(factor(pred_grid$year_f)))

# make predictions and get index
f2 <- here("species_specific_code", "GOA", species,
           "index_comparison", "predictions.RDS")
if (!file.exists(f2)) {
```

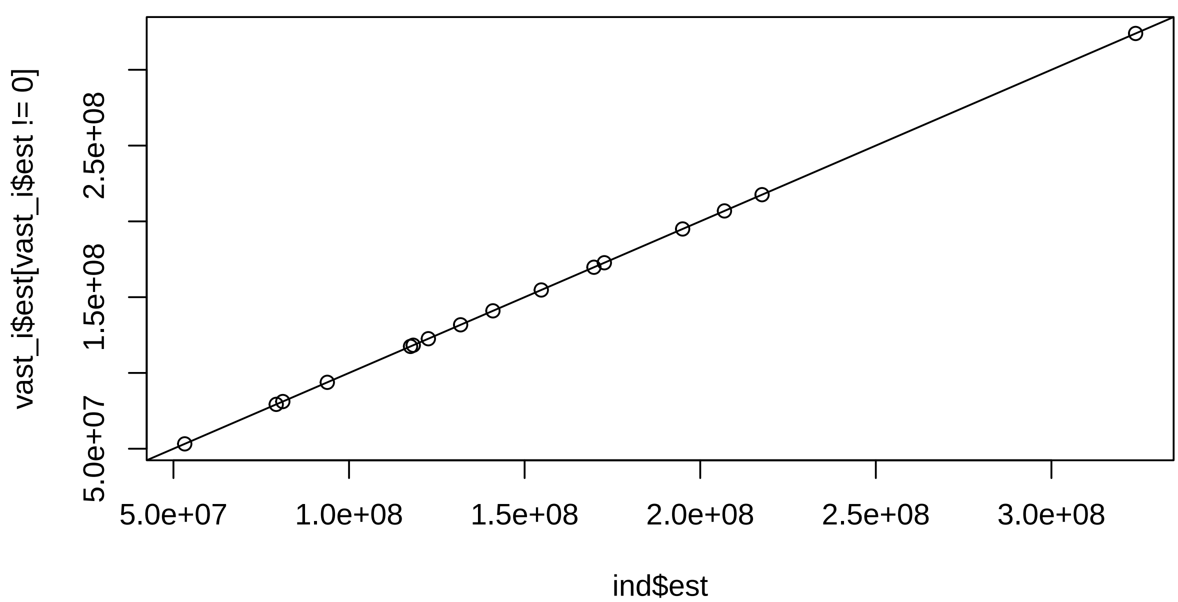
```
p <- predict(fit_sdmTMB, newdata = pred_grid, return_tmb_object = TRUE)
saveRDS(p, file = here("species_specific_code", "GOA", species, "index_comparison", "predictions.RDS"))
} else {
p <- readRDS(f2)
}

f3 <- here("species_specific_code", "GOA", species,
           "index_comparison", "index.RDS")
if (!file.exists(f3)) {
ind <- get_index(p, bias_correct = TRUE, area = p$data$area_km2)
saveRDS(ind, file = here("species_specific_code", "GOA", species, "index_comparison", "index.RDS"))
} else {
ind <- readRDS(f3)
}
```

Now, we can compare the indices.

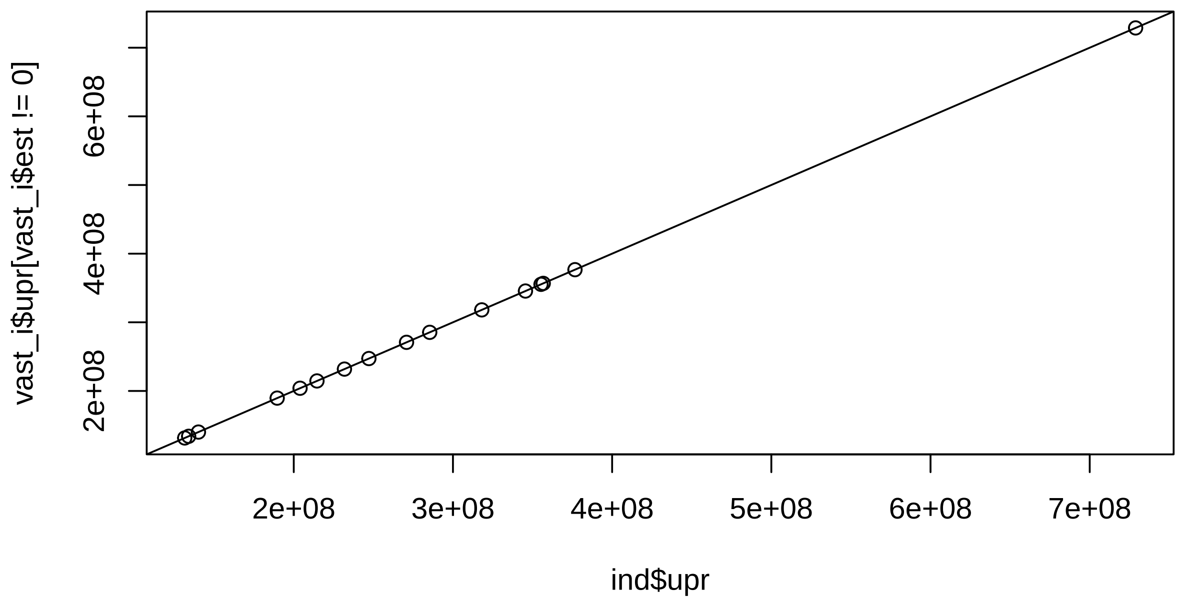
```
sdm_i <- ind %>% mutate(index = "sdmTMB")
vast_i <- read.csv(here("species_specific_code", "GOA", species, "index_comparison", "Index.csv")) %>%
  mutate(index = "VAST", year = as.numeric(Time), est = Estimate,
    se = Std..Error.for.ln.Estimate.) %>%
  select(index, year, est, se) %>%
  filter(year %in% unique(sdm_i$year)) %>%
  mutate(lwr = exp(log(est) + qnorm(0.025) * se)) %>%
  mutate(upr = exp(log(est) + qnorm(0.975) * se))
both_i <- bind_rows(sdm_i, vast_i) %>% filter(est > 0)

ggplot(both_i, aes(x = year, y = est, ymin = lwr, ymax = upr, colour = index)) +
  geom_ribbon(alpha = 0.1) +
  geom_line(alpha = 0.8) +
  ylim(0, max(both_i$upr)) +
  ggtitle(species) +
  coord_cartesian(expand = FALSE)
```



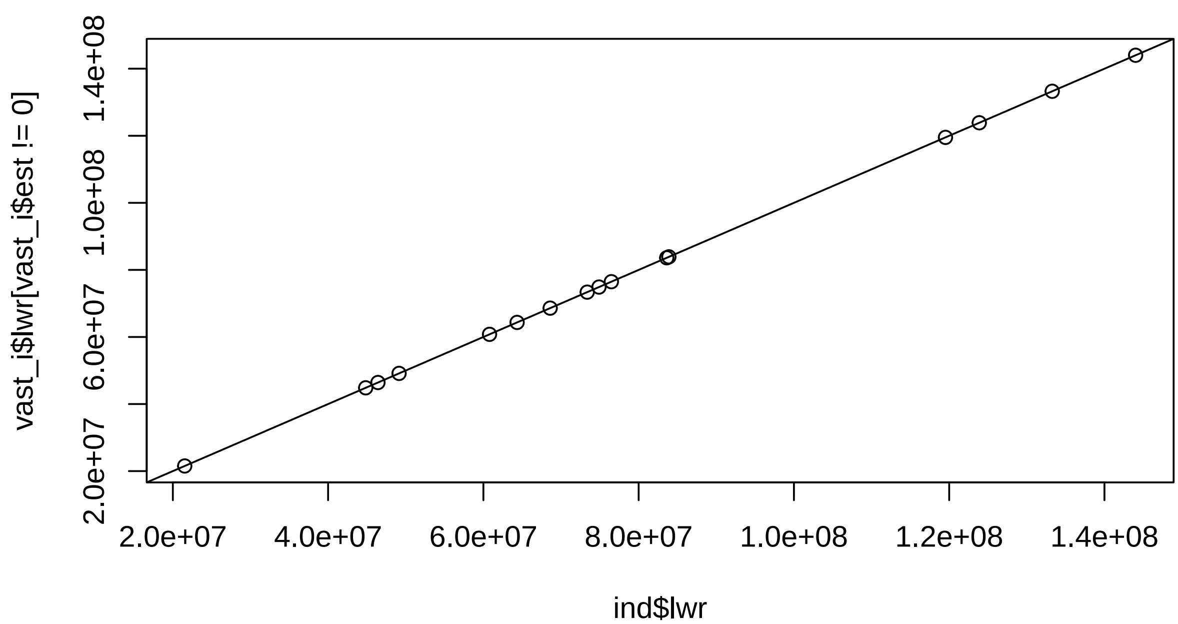

```
plot(ind$est, vast_i$est[vast_i$est != 0]);abline(0, 1)
```




```
plot(ind$upr, vast_i$upr[vast_i$est != 0]);abline(0, 1)
```




```r
plot(ind$lwr, vast_i$lwr[vast_i$est != 0]);abline(0, 1)
```




```r
(ind$est - vast_i$est[vast_i$est != 0]) / vast_i$est[vast_i$est != 0]
```

```
#>  [1] -1.797774e-09 -9.538800e-10 -9.431323e-10 -3.985331e-10 -6.178280e-10
#>  [6] -6.134534e-11  2.444702e-10 -9.091896e-10  3.101261e-10 -2.890436e-09
#> [11] -1.381692e-09 -1.010469e-10 -1.436599e-09  4.443212e-10 -3.682838e-10
#> [16] -4.687039e-10
(ind$upr - vast_i$upr[vast_i$est != 0]) / vast_i$upr[vast_i$est != 0]
#>  [1] -7.195528e-09 -1.892737e-09  5.679613e-09 -5.621654e-09 -3.398824e-09
#>  [6]  4.130735e-11  3.622632e-10 -2.272607e-09 -6.243457e-09 -1.050545e-08
#> [11] -6.362022e-09 -2.393151e-09 -4.636348e-09 -1.201062e-09 -3.519151e-09
#> [16] -1.978164e-08
(ind$lwr - vast_i$lwr[vast_i$est != 0]) / vast_i$lwr[vast_i$est != 0]
#>  [1]  3.599982e-09 -1.502082e-11 -7.565877e-09  4.824589e-09  2.163169e-09
#>  [6] -1.639969e-10  1.266826e-10  4.542287e-10  6.863704e-09  4.724576e-09
#> [11]  3.598636e-09  2.191058e-09  1.763148e-09  2.089703e-09  2.782581e-09
#> [16]  1.884424e-08
```

This document was built using:

```
R.Version()$version.string
#> [1] "R version 4.3.0 (2023-04-21 ucrt)"
packageVersion("VAST")
#> [1] '3.11.2'
packageVersion("FishStatsUtils")
#> [1] '2.13.1'
```