Comparing VAST and sdmTMB GOA indices

Contents

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

species <- "Sebastes_alutus"
#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.

```
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.

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

```
                    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).

```
# 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 1.75e-07: 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.

```
fit$parameter_estimates$diagnostics
#>           Param starting_value     Lower        MLE     Upper final_gradient
#> 1    ln_H_input      0.3172816 -5.000000  0.3172864  5.000000  -5.817036e-11
#> 2    ln_H_input      0.2622739 -5.000000  0.2622772  5.000000  -1.539249e-09
#> 3      beta1_ft     -5.5039285      -Inf -5.5039384       Inf  -7.421860e-09
```

```
#> 4      beta1_ft     -5.5357762      -Inf -5.5357244       Inf   3.904395e-09
#> 5      beta1_ft     -5.5470263      -Inf -5.5470157       Inf  -5.309680e-09
#> 6      beta1_ft     -5.2543223      -Inf -5.2542526       Inf   7.766865e-09
#> 7      beta1_ft     -5.1994637      -Inf -5.1995538       Inf  -2.765188e-08
#> 8      beta1_ft     -5.1986556      -Inf -5.1985915       Inf   7.793700e-09
#> 9      beta1_ft     -5.0057844      -Inf -5.0057291       Inf   5.970142e-09
#> 10     beta1_ft     -5.3109535      -Inf -5.3109601       Inf  -8.015114e-09
#> 11     beta1_ft     -4.9331593      -Inf -4.9331345       Inf  -1.663381e-11
#> 12     beta1_ft     -4.9296782      -Inf -4.9296123       Inf   9.731629e-09
#> 13     beta1_ft     -5.1427643      -Inf -5.1427233       Inf   1.362526e-09
#> 14     beta1_ft     -5.1076812      -Inf -5.1076563       Inf  -2.236112e-09
#> 15     beta1_ft     -4.7954332      -Inf -4.7954096       Inf  -5.559961e-10
#> 16     beta1_ft     -4.5498138      -Inf -4.5497695       Inf   4.158128e-09
#> 17     beta1_ft     -4.7958334      -Inf -4.7957981       Inf   1.577803e-09
#> 18     beta1_ft     -4.7357109      -Inf -4.7356372       Inf   8.775263e-09
#> 19   L_omega1_z      7.2572792      -Inf  7.2573201       Inf  -3.276881e-09
#> 20 L_epsilon1_z      0.3586301      -Inf  0.3586291       Inf  -2.124104e-07
#> 21    logkappa1     -4.6718188 -6.765487 -4.6718199 -1.659642   4.063463e-08
#> 22     beta2_ft      5.1028725      -Inf  5.1029239       Inf   2.366107e-11
#> 23     beta2_ft      5.5424978      -Inf  5.5425450       Inf   8.321592e-10
#> 24     beta2_ft      5.8261872      -Inf  5.8262992       Inf  -8.720562e-10
#> 25     beta2_ft      5.0598253      -Inf  5.0598062       Inf   8.002488e-10
#> 26     beta2_ft      5.8305679      -Inf  5.8305170       Inf  -2.582439e-09
#> 27     beta2_ft      5.4053552      -Inf  5.4052385       Inf  -1.049465e-09
#> 28     beta2_ft      5.6939084      -Inf  5.6938739       Inf   8.129817e-10
#> 29     beta2_ft      5.7830237      -Inf  5.7831180       Inf  -6.842029e-10
#> 30     beta2_ft      5.3532573      -Inf  5.3532570       Inf   3.142659e-10
#> 31     beta2_ft      5.7327540      -Inf  5.7326111       Inf  -1.939220e-09
#> 32     beta2_ft      6.2416401      -Inf  6.2416490       Inf   5.649703e-10
#> 33     beta2_ft      6.4977326      -Inf  6.4978028       Inf   3.182450e-10
#> 34     beta2_ft      6.2992360      -Inf  6.2991993       Inf   3.876011e-11
#> 35     beta2_ft      6.3595385      -Inf  6.3595072       Inf   7.820802e-10
#> 36     beta2_ft      6.2402684      -Inf  6.2402702       Inf   5.661533e-10
#> 37     beta2_ft      6.4689613      -Inf  6.4689364       Inf   1.034053e-09
#> 38   L_omega2_z      2.2009707      -Inf  2.2009727       Inf  -3.611547e-10
#> 39 L_epsilon2_z      1.4886851      -Inf  1.4886809       Inf  -1.098662e-07
#> 40    logkappa2     -2.8537476 -6.765487 -2.8537513 -1.659642   1.108557e-07
#> 41    logSigmaM      0.3356156      -Inf  0.3356099 10.000000  -1.195028e-07
```

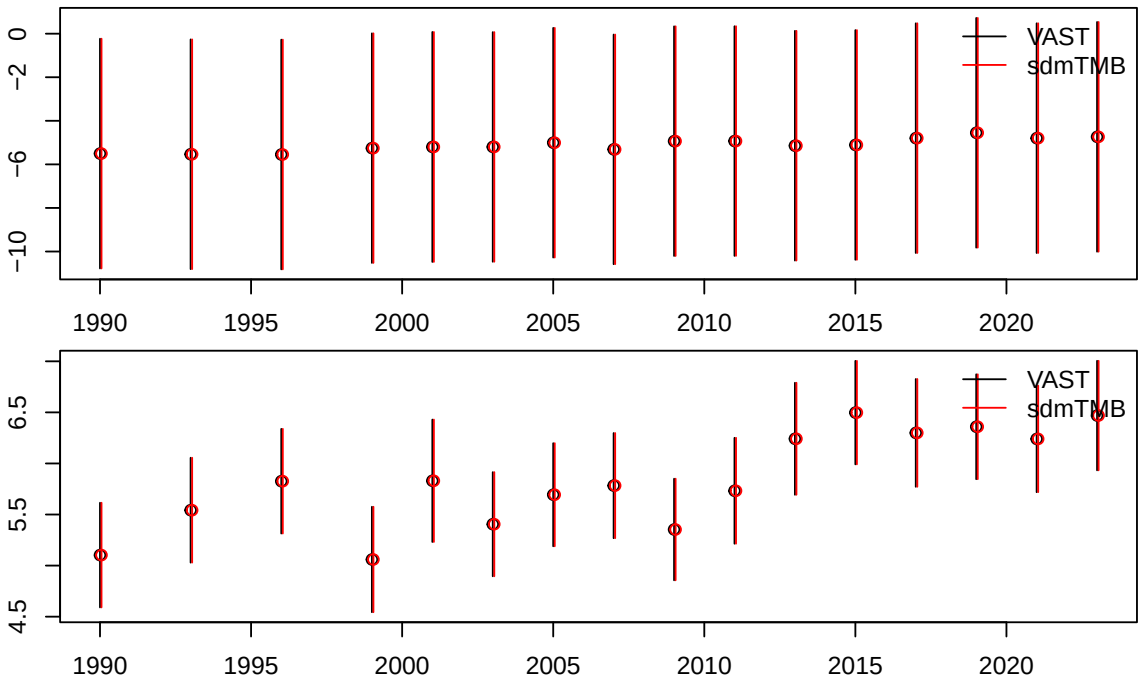
Now we fit the same model in sdmTMB:

```
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,
```

```
          "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:

```
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)
```



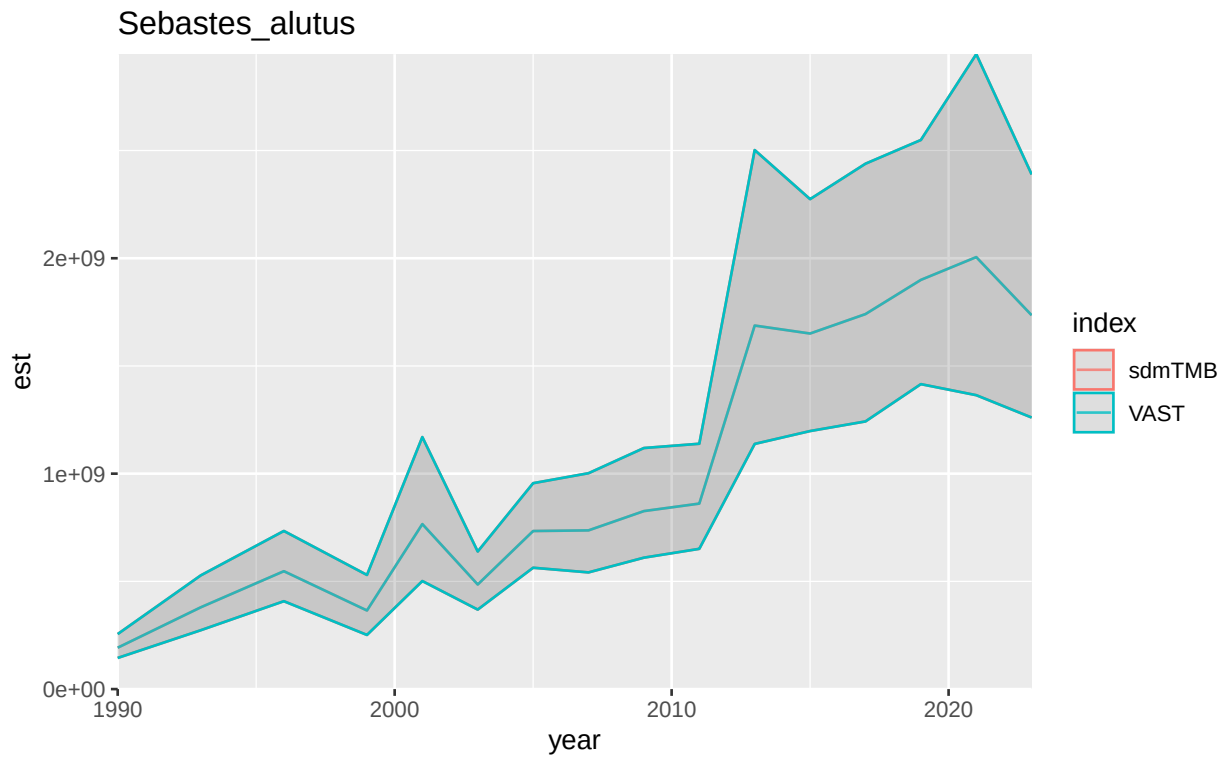
We can compare the index we would get using sdmTMB.

```
# 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)) {
```

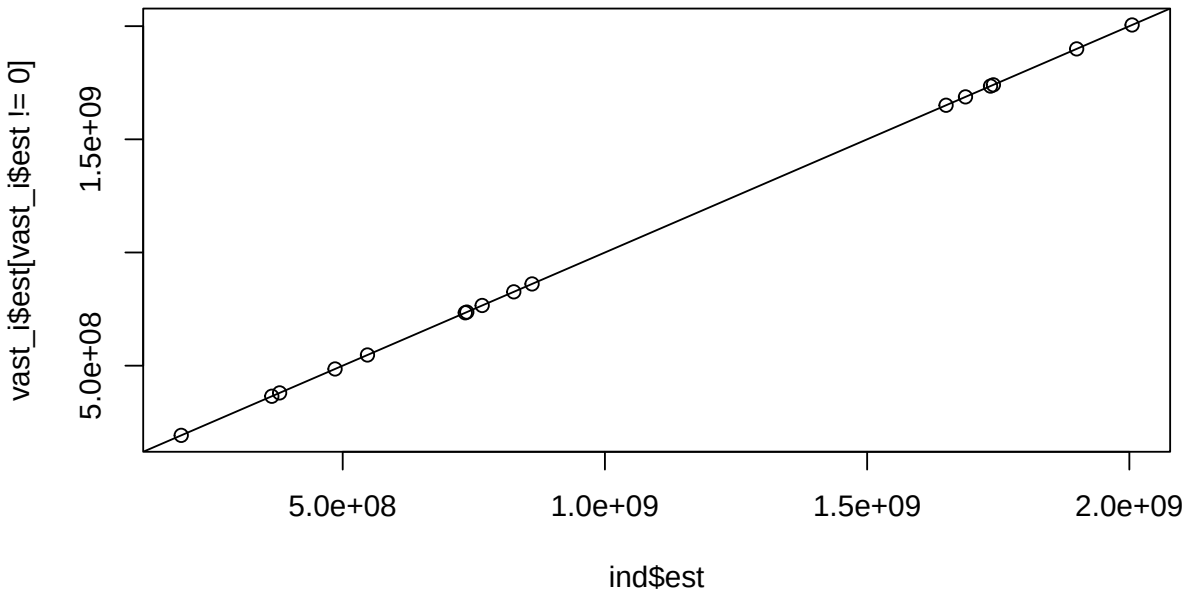
```
p <- predict(fit_sdmTMB, newdata = pred_grid, return_tmb_object = TRUE)
saveRDS(p, file = here("species_specific_code", "GOA", species, "index_comparison", "predictions.RDS"))
} else {
p <- readRDS(f2)
}

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.

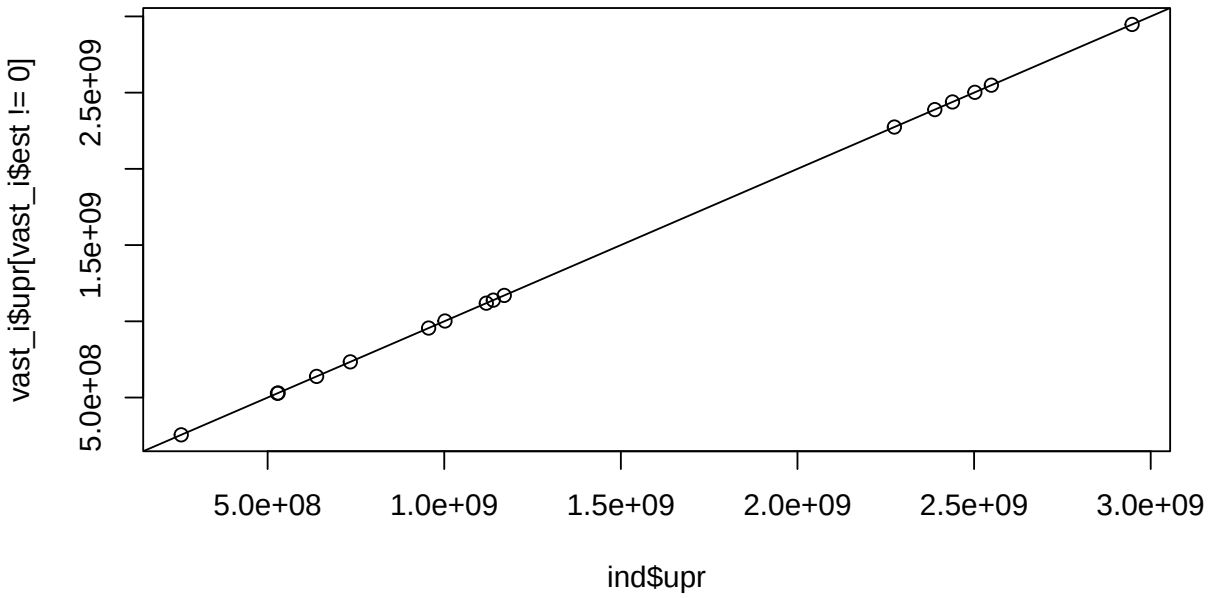
```
sdm_i <- ind %>% mutate(index = "sdmTMB")
vast_i <- read.csv(here("species_specific_code", "GOA", species, "index_comparison", "Index.csv")) %>%
  mutate(index = "VAST", year = as.numeric(Time), est = Estimate,
    se = Std..Error.for.ln.Estimate.) %>%
  select(index, year, est, se) %>%
  filter(year %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)
```



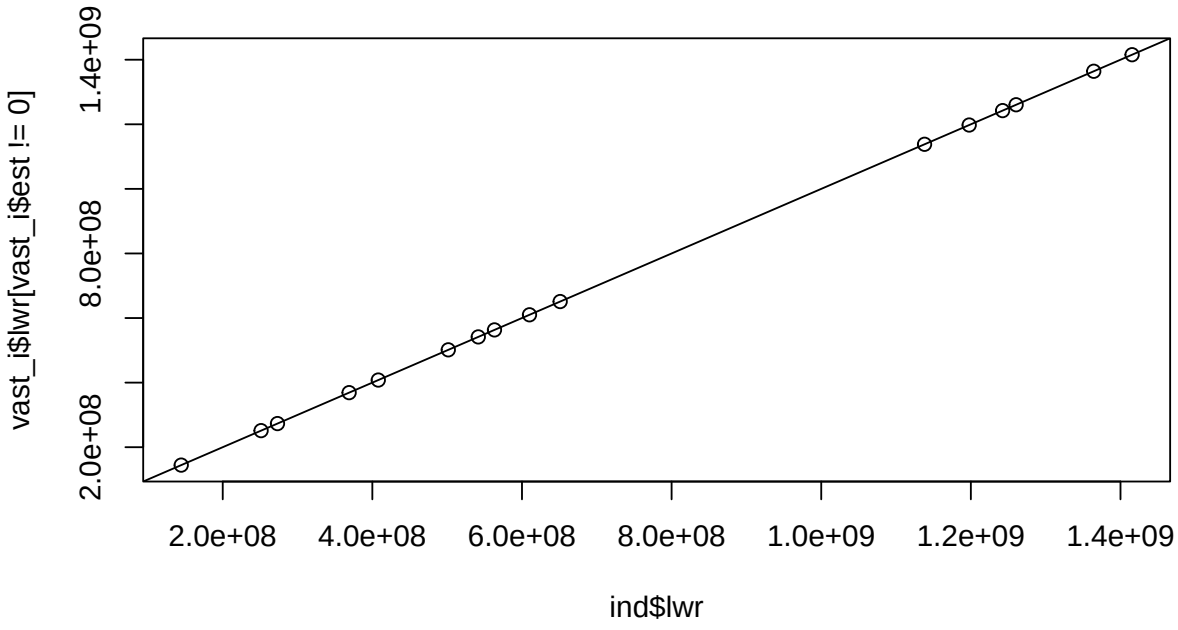

```
plot(ind$est, vast_i$est[vast_i$est != 0]);abline(0, 1)
```




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




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




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

```
#>  [1] -5.330383e-11  2.313494e-10 -1.373967e-10  8.523031e-11  2.177721e-10
#>  [6]  2.876344e-10 -1.742826e-10  1.632228e-10  1.021521e-10  3.000802e-10
#> [11] -1.126203e-11 -1.323368e-10  3.203509e-10  5.696926e-11  1.583539e-10
#> [16]  1.473826e-10
(ind$upr - vast_i$upr[vast_i$est != 0]) / vast_i$upr[vast_i$est != 0]
#>  [1] -6.371188e-10  5.620748e-10 -8.188579e-10 -1.166374e-09 -6.385232e-09
#>  [6] -3.266543e-10 -1.363798e-09  9.640857e-10 -4.513475e-10 -3.209781e-09
#> [11] -3.932566e-09 -2.504464e-09 -2.616702e-09 -2.383288e-09  1.161388e-11
#> [16] -3.983846e-09
(ind$lwr - vast_i$lwr[vast_i$est != 0]) / vast_i$lwr[vast_i$est != 0]
#>  [1]  5.305089e-10 -9.937659e-11  5.440626e-10  1.336833e-09  6.820780e-09
#>  [6]  9.019238e-10  1.015234e-09 -6.376412e-10  6.556498e-10  3.809941e-09
#> [11]  3.910042e-09  2.239787e-09  3.257405e-09  2.497224e-09  3.050894e-10
#> [16]  4.278611e-09
```

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'
```