

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_variabilis"
#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/km².

```
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% encounter
  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 2.8e-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    0.22431634 -5.000000  0.22431782  5.000000 -5.376359e-09
#> 2  ln_H_input    0.13141468 -5.000000  0.13141574  5.000000 -7.357404e-09
#> 3   beta1_ft    -3.50992317  -Inf -3.50991755      Inf -1.238694e-09

```

```

#> 4      beta1_ft      -2.53414281      -Inf -2.53414460      Inf -1.161650e-09
#> 5      beta1_ft      -2.86122138      -Inf -2.86122149      Inf -8.500614e-10
#> 6      beta1_ft      -2.92061916      -Inf -2.92060913      Inf -8.290613e-10
#> 7      beta1_ft      -2.84134737      -Inf -2.84130929      Inf  6.675836e-09
#> 8      beta1_ft      -2.67699818      -Inf -2.67698582      Inf -2.407088e-10
#> 9      beta1_ft      -2.60573764      -Inf -2.60573457      Inf -1.039933e-09
#> 10     beta1_ft      -2.68948717      -Inf -2.68949463      Inf -6.489778e-10
#> 11     beta1_ft      -2.85404062      -Inf -2.85403671      Inf -1.266466e-09
#> 12     beta1_ft      -2.91090288      -Inf -2.91090356      Inf -5.331646e-10
#> 13     beta1_ft      -2.72227845      -Inf -2.72225608      Inf  1.047644e-09
#> 14     beta1_ft      -2.74259387      -Inf -2.74258882      Inf -6.351435e-10
#> 15     beta1_ft      -2.45426843      -Inf -2.45429340      Inf -1.752198e-11
#> 16     beta1_ft      -2.18951239      -Inf -2.18950986      Inf -6.409753e-10
#> 17     beta1_ft      -2.49502666      -Inf -2.49498281      Inf  3.166615e-09
#> 18     beta1_ft      -2.45344190      -Inf -2.45344826      Inf -5.118466e-10
#> 19     L_omega1_z      1.69122361      -Inf  1.69121068      Inf -3.191666e-09
#> 20     L_epsilon1_z      0.44268706      -Inf  0.44268935      Inf -2.326690e-08
#> 21     logkappa1      -3.14173940 -6.765487 -3.14173040 -1.659642  3.699228e-09
#> 22     beta2_ft      5.19539975      -Inf  5.19550592      Inf  6.635052e-09
#> 23     beta2_ft      5.11226637      -Inf  5.11233579      Inf  4.082985e-09
#> 24     beta2_ft      5.66667169      -Inf  5.66680022      Inf  6.663633e-09
#> 25     beta2_ft      5.34051015      -Inf  5.34059811      Inf  3.406569e-09
#> 26     beta2_ft      5.85663403      -Inf  5.85673609      Inf  4.101942e-09
#> 27     beta2_ft      5.77305189      -Inf  5.77318374      Inf  9.887014e-09
#> 28     beta2_ft      5.98076258      -Inf  5.98088718      Inf  6.606189e-09
#> 29     beta2_ft      5.67361609      -Inf  5.67373826      Inf  6.725838e-09
#> 30     beta2_ft      5.43681805      -Inf  5.43690668      Inf  3.609181e-09
#> 31     beta2_ft      5.49391404      -Inf  5.49401794      Inf  4.343857e-09
#> 32     beta2_ft      5.82868288      -Inf  5.82880495      Inf  5.012975e-09
#> 33     beta2_ft      5.62396814      -Inf  5.62407540      Inf  4.460620e-09
#> 34     beta2_ft      5.47400675      -Inf  5.47409154      Inf  2.885571e-09
#> 35     beta2_ft      5.43856397      -Inf  5.43865850      Inf  4.211280e-09
#> 36     beta2_ft      5.59496343      -Inf  5.59505576      Inf  4.433950e-09
#> 37     beta2_ft      5.37130570      -Inf  5.37139262      Inf  3.954288e-09
#> 38     L_omega2_z      1.27968716      -Inf  1.27969953      Inf -1.647631e-07
#> 39     L_epsilon2_z      1.64733848      -Inf  1.64736284      Inf -1.264471e-07
#> 40     logkappa2      -2.47470640 -6.765487 -2.47465890 -1.659642  1.257782e-07
#> 41     logSigmaM      0.06254715      -Inf  0.06255359 10.000000 -2.821395e-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$tmobj)

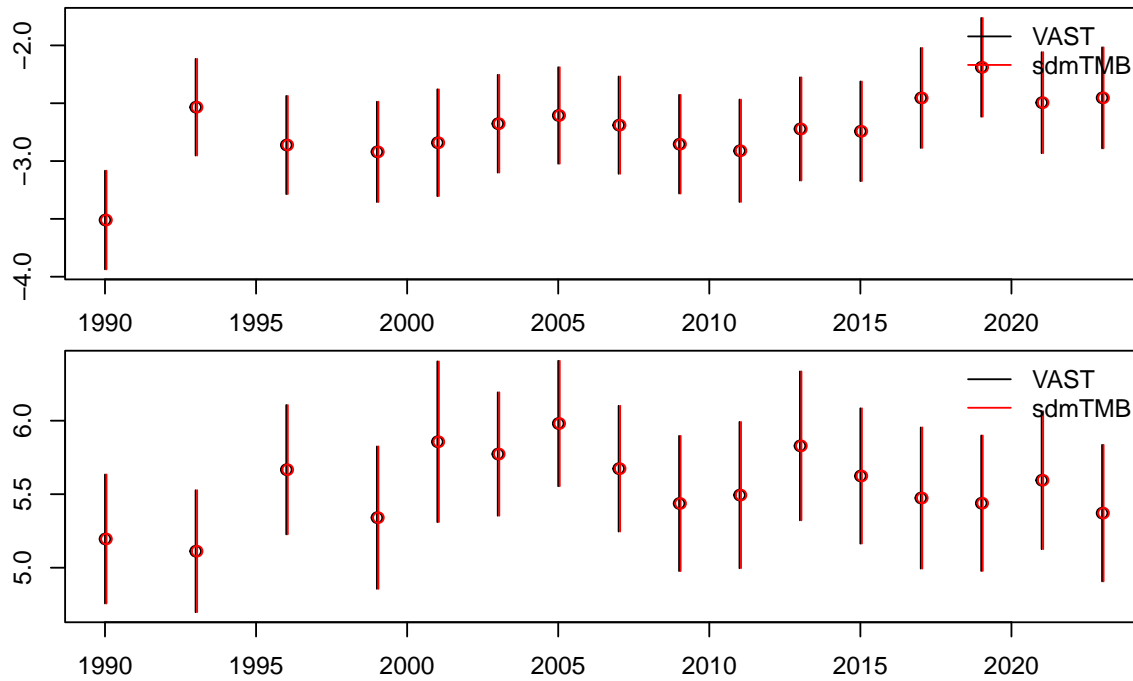
```

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)
}

```

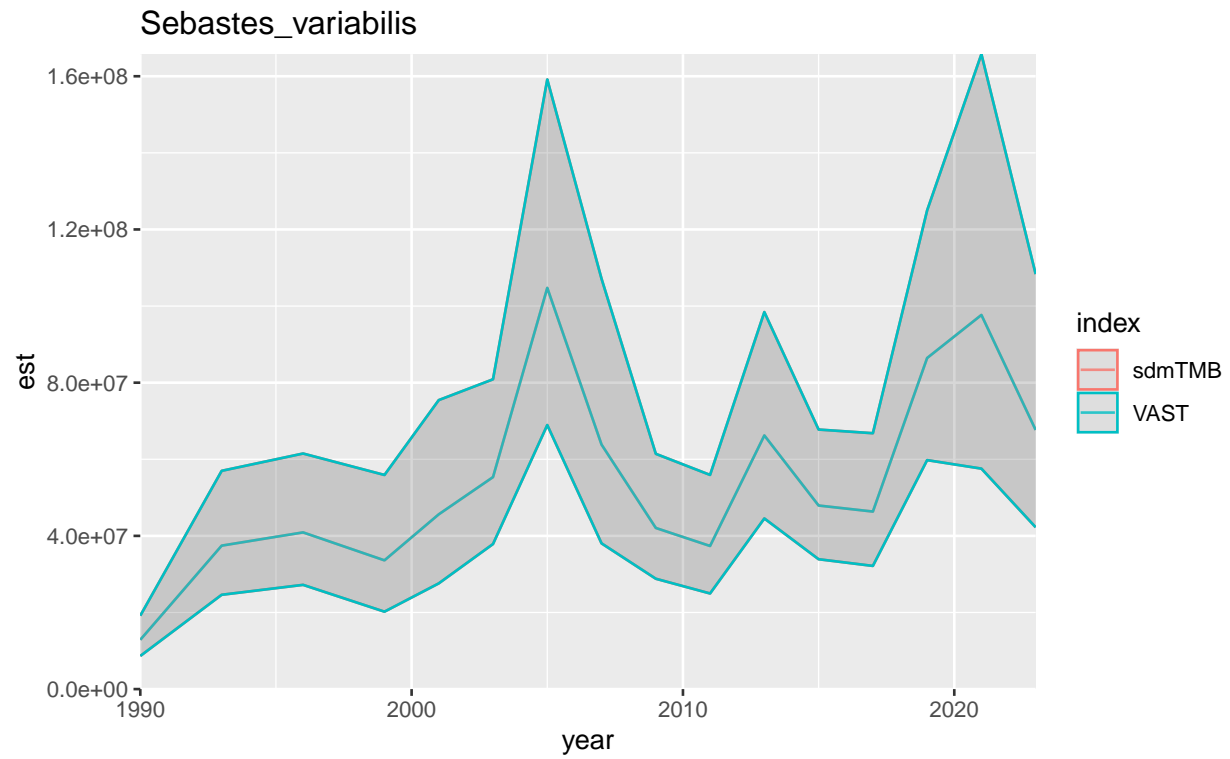
Now, we can compare the indices.

```

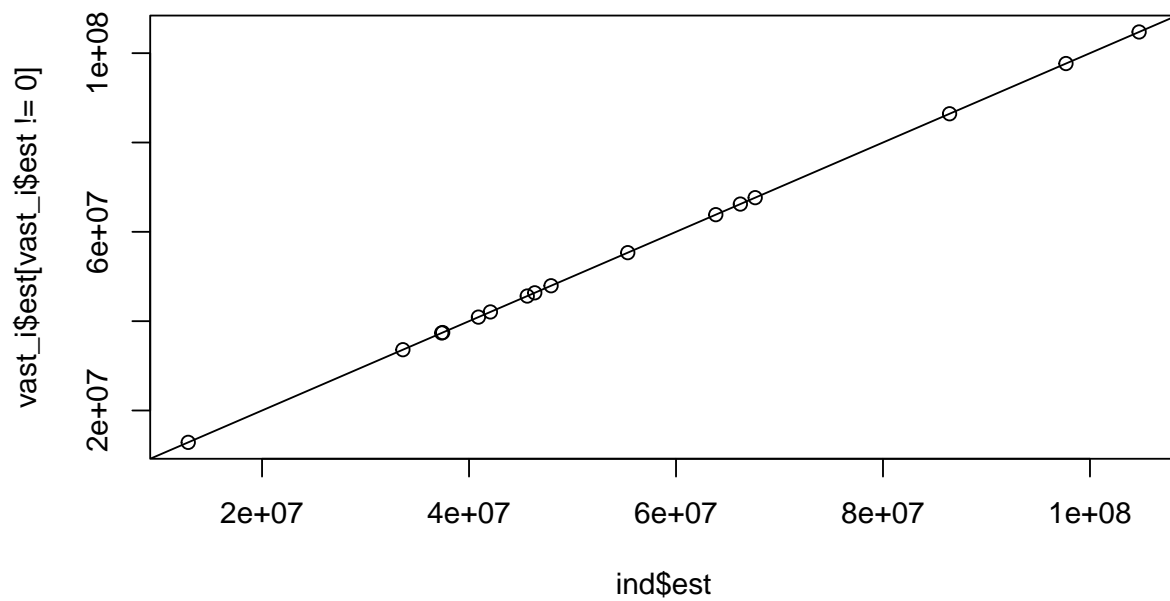
sdm_i <- ind %>% mutate(index = "sdmTMB")
vast_i <- read.csv(here("species_specific_code", "GOA", species, "index_comparison", "Index.csv")) %>%
  mutate(index = "VAST", year = as.numeric(Time), est = Estimate,
         se = Std..Error.for.ln.Estimate.) %>%
  select(index, year, est, se) %>%
  filter(year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

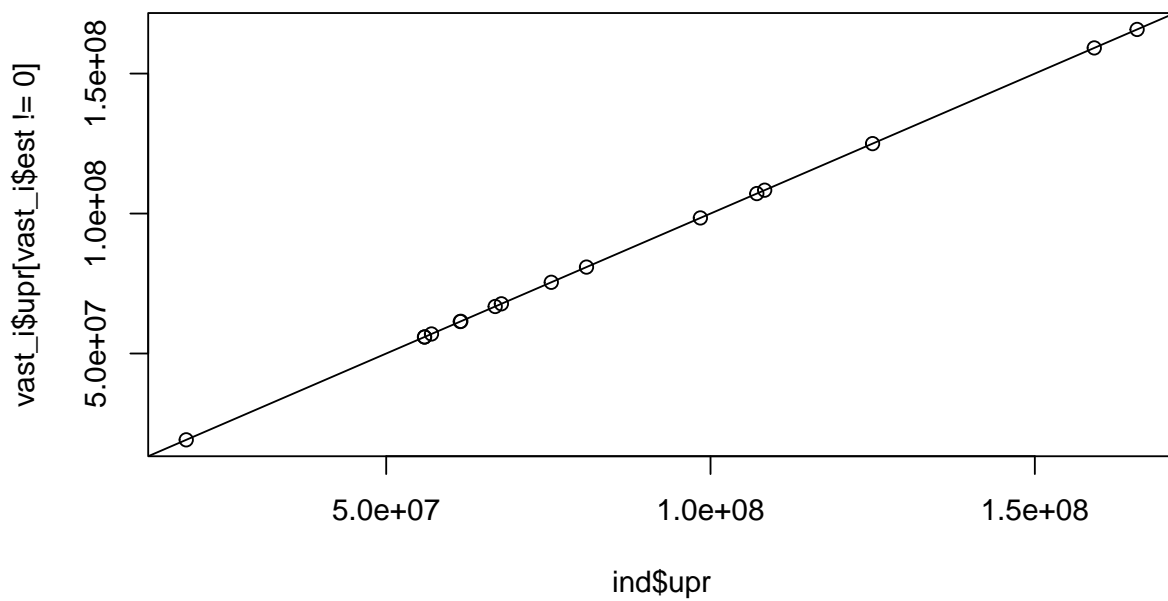
```



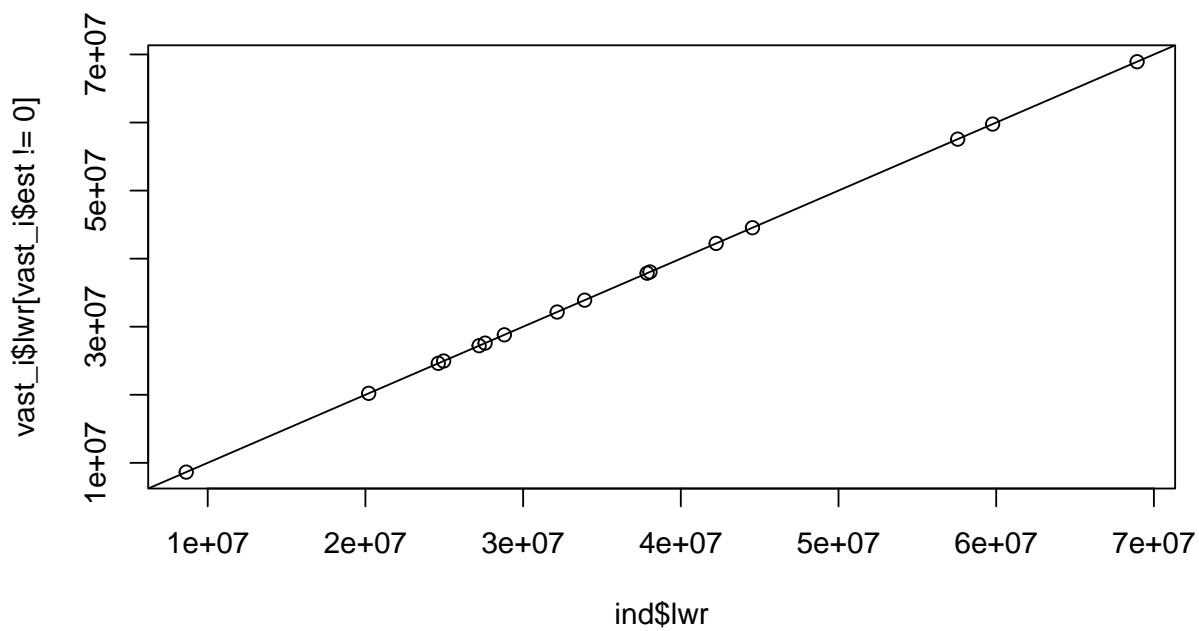
```
plot(ind$est, vast_i$est[vast_i$est != 0]);abline(0, 1)
```



```
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] 2.235602e-10 -2.013961e-10 8.356056e-11 6.498517e-11 2.297939e-10
#> [6] -3.586832e-10 2.302957e-11 -1.321207e-11 6.068031e-11 1.465415e-10
#> [11] 8.573068e-11 3.002697e-10 8.905364e-11 1.218069e-10 1.378772e-10
#> [16] 2.756719e-10
(ind$supr - vast_i$supr[vast_i$est != 0]) / vast_i$supr[vast_i$est != 0]
#> [1] -2.000714e-09 -1.285894e-08 -1.799783e-09 -1.156852e-08 1.263452e-09
#> [6] -1.222102e-09 -1.362218e-10 -1.471655e-09 -4.532946e-09 -3.547601e-09
#> [11] -5.464074e-09 2.424194e-09 -3.615099e-10 -9.088871e-10 -4.215991e-09
#> [16] -3.848658e-09
(ind$lwr - vast_i$lwr[vast_i$est != 0]) / vast_i$lwr[vast_i$est != 0]
#> [1] 2.447838e-09 1.245615e-08 1.966910e-09 1.169849e-08 -8.038582e-10
#> [6] 5.047376e-10 1.822792e-10 1.445230e-09 4.654307e-09 3.840686e-09
#> [11] 5.635535e-09 -1.823658e-09 5.396145e-10 1.152497e-09 4.491746e-09
#> [16] 4.400004e-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'

```