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 <- "Gadus_macrocephalus"
#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,
```

```
                    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 2.3e-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    0.284748154 -5.000000  0.284577229  5.000000   6.711671e-08
#> 2     ln_H_input   -0.109983499 -5.000000 -0.109928395  5.000000   1.381891e-08
#> 3       beta1_ft    0.195212567      -Inf  0.195166009       Inf  -8.392703e-11
```

```
#> 4        beta1_ft   0.171336433      -Inf  0.171372644        Inf   8.311095e-09
#> 5        beta1_ft  -0.002916705      -Inf -0.002900598        Inf   4.141840e-09
#> 6        beta1_ft  -0.327705123      -Inf -0.327649553        Inf  -1.236582e-09
#> 7        beta1_ft  -0.582152067      -Inf -0.582112061        Inf   3.522767e-09
#> 8        beta1_ft  -0.412166845      -Inf -0.412183366        Inf  -6.092453e-09
#> 9        beta1_ft  -0.388786040      -Inf -0.388814536        Inf  -2.881159e-09
#> 10       beta1_ft  -0.342445273      -Inf -0.342418168        Inf  -1.064150e-09
#> 11       beta1_ft  -0.061281039      -Inf -0.061320616        Inf  -2.574438e-09
#> 12       beta1_ft  -0.028192674      -Inf -0.028235564        Inf  -6.089888e-09
#> 13       beta1_ft  -0.083099340      -Inf -0.083087273        Inf  -1.653512e-09
#> 14       beta1_ft  -0.113169222      -Inf -0.113124098        Inf  -6.834916e-11
#> 15       beta1_ft  -0.625864836      -Inf -0.625821390        Inf   5.094845e-09
#> 16       beta1_ft  -0.363171954      -Inf -0.363178999        Inf  -3.668527e-09
#> 17       beta1_ft  -0.235271867      -Inf -0.235300328        Inf   1.309729e-09
#> 18       beta1_ft  -0.084350872      -Inf -0.084374939        Inf   3.721144e-09
#> 19     L_omega1_z   2.240463825      -Inf  2.240683709        Inf  -8.779238e-08
#> 20 L_epsilon1_z   0.517131693      -Inf  0.517142219        Inf  -6.118089e-08
#> 21      logkappa1  -3.763835187 -6.775053 -3.763776176 -1.659693   1.941856e-08
#> 22       beta2_ft   6.562098026      -Inf  6.562006725        Inf  -7.101534e-09
#> 23       beta2_ft   6.481707818      -Inf  6.481710104        Inf   1.230900e-08
#> 24       beta2_ft   6.695851940      -Inf  6.695804368        Inf   1.235978e-09
#> 25       beta2_ft   6.667421120      -Inf  6.667398042        Inf   6.224163e-09
#> 26       beta2_ft   7.016113144      -Inf  7.016001358        Inf  -1.021149e-08
#> 27       beta2_ft   6.825944291      -Inf  6.825925993        Inf   6.137248e-09
#> 28       beta2_ft   6.714614625      -Inf  6.714507026        Inf  -1.116533e-08
#> 29       beta2_ft   6.459899384      -Inf  6.459844366        Inf  -1.233103e-10
#> 30       beta2_ft   6.782682756      -Inf  6.782530055        Inf  -2.222228e-08
#> 31       beta2_ft   6.727735802      -Inf  6.727547225        Inf  -2.730335e-08
#> 32       beta2_ft   6.825294681      -Inf  6.825214310        Inf  -6.259421e-09
#> 33       beta2_ft   6.464120834      -Inf  6.464059425        Inf  -1.065009e-09
#> 34       beta2_ft   6.172553675      -Inf  6.172622557        Inf   2.546204e-08
#> 35       beta2_ft   6.269442050      -Inf  6.269389038        Inf   1.704376e-09
#> 36       beta2_ft   6.305092039      -Inf  6.305113020        Inf   1.587775e-08
#> 37       beta2_ft   6.257056953      -Inf  6.257098843        Inf   1.877512e-08
#> 38     L_omega2_z   0.990111386      -Inf  0.990193930        Inf  -7.955983e-08
#> 39 L_epsilon2_z   1.358181353      -Inf  1.358153607        Inf  -1.900610e-07
#> 40      logkappa2  -3.986428697 -6.775053 -3.986360948 -1.659693   2.303347e-07
#> 41      logSigmaM   0.254237914      -Inf  0.254231545 10.000000  -2.443863e-07
```

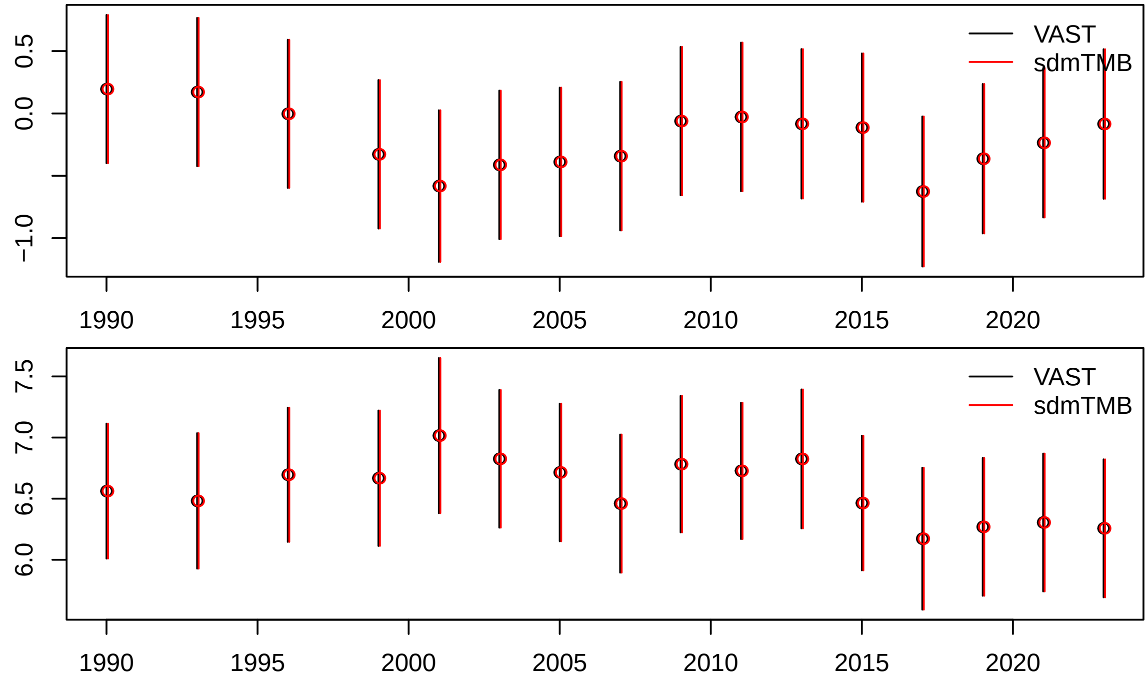
Now we fit the same model in sdmTMB:

```
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)
```



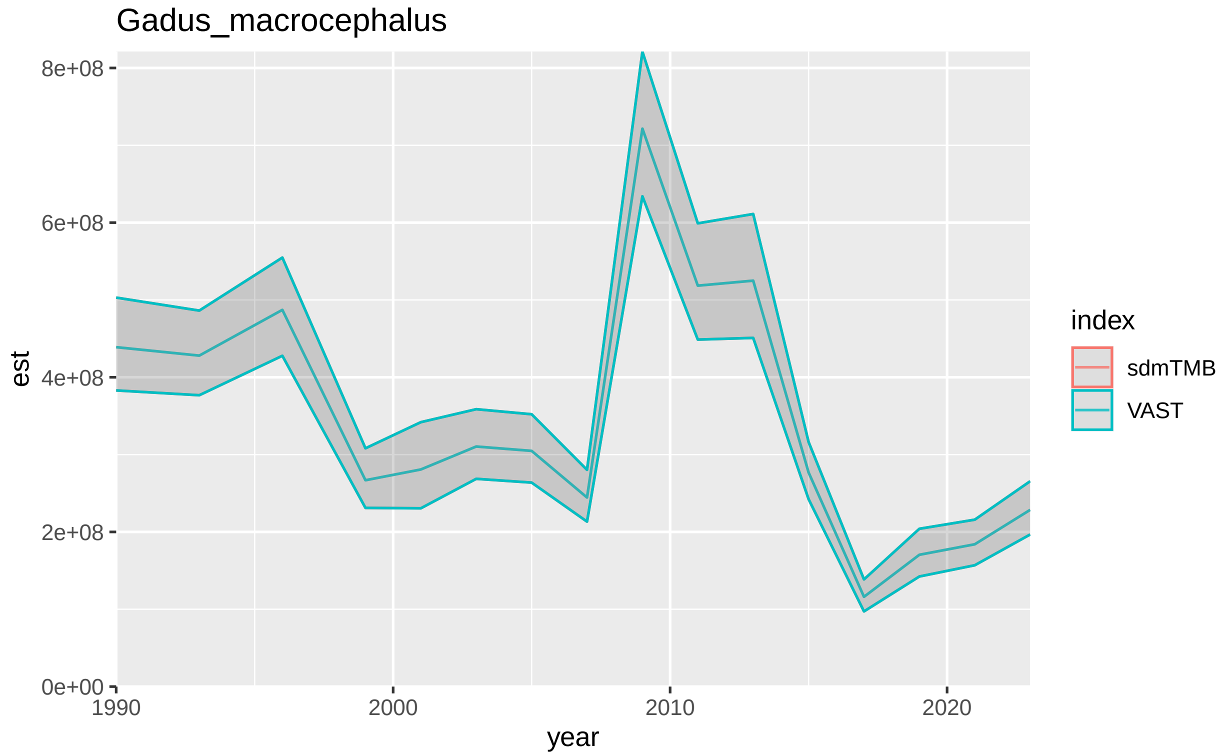
We can compare the index we would get using sdmTMB.

```
# 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)) {
```

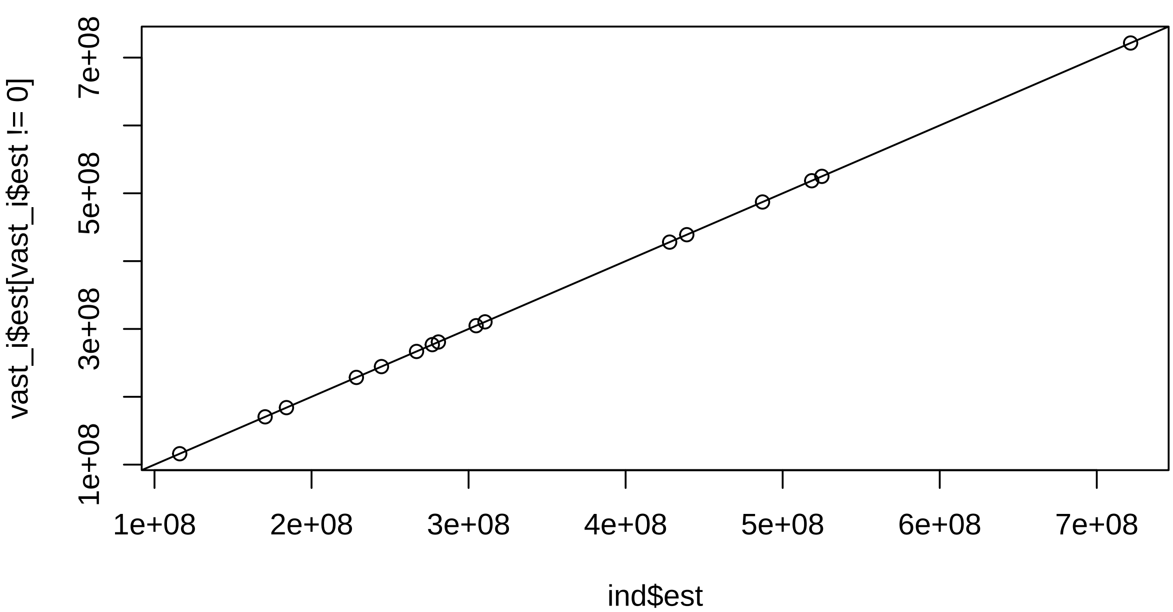
```
p <- predict(fit_sdmTMB, newdata = pred_grid, return_tmb_object = TRUE)
saveRDS(p, file = here("species_specific_code", "GOA", species, "index_comparison", "predictions.RDS"))
} else {
p <- readRDS(f2)
}

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.

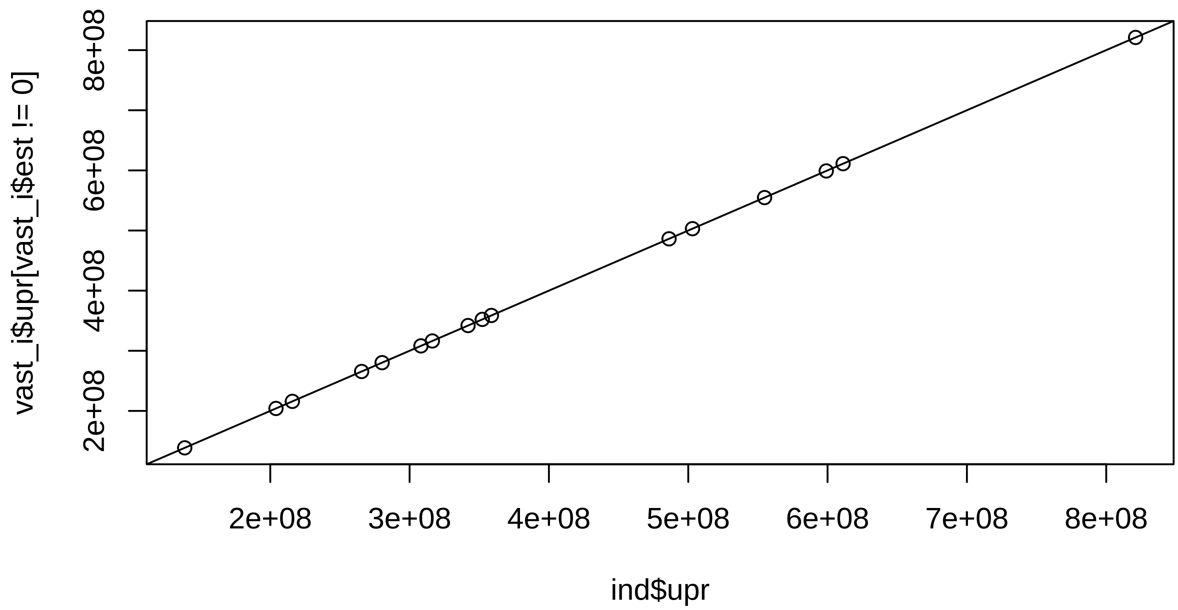
```
sdm_i <- ind %>% mutate(index = "sdmTMB")
vast_i <- read.csv(here("species_specific_code", "GOA", species, "index_comparison", "Index.csv")) %>%
  mutate(index = "VAST", year = as.numeric(Time), est = Estimate,
    se = Std..Error.for.ln.Estimate.) %>%
  select(index, year, est, se) %>%
  filter(year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



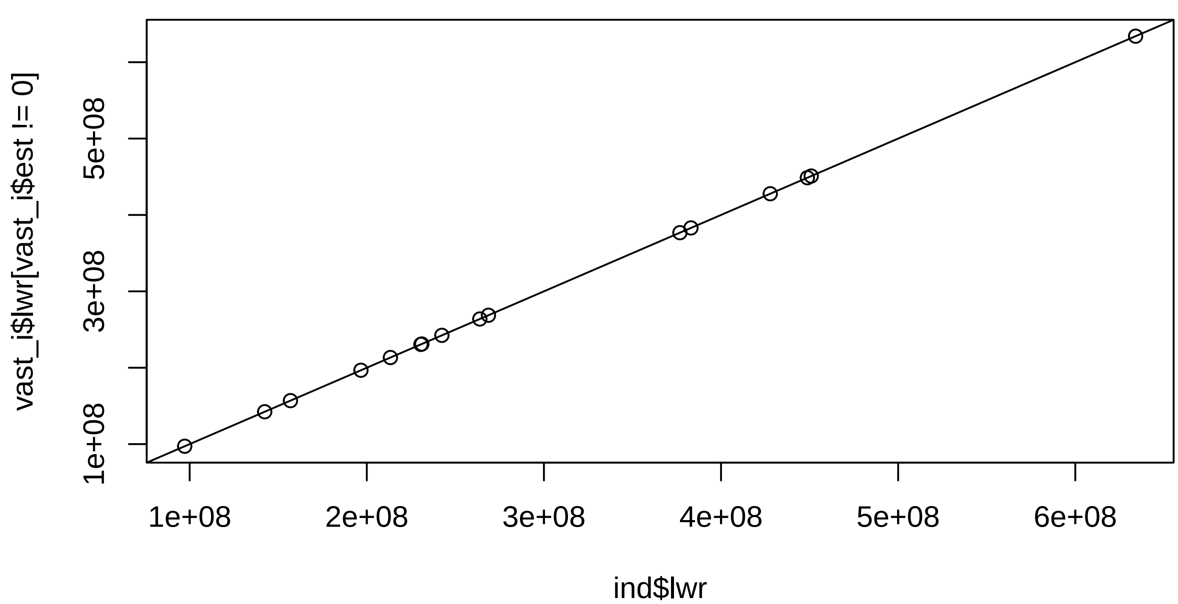

```
plot(ind$est, vast_i$est[vast_i$est != 0]);abline(0, 1)
```




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




```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]  7.729191e-11  8.647747e-13  2.907960e-11 -7.192213e-11  6.054096e-11
#>  [6]  6.744030e-11 -3.831928e-11 -1.328996e-11  6.135731e-11  8.876706e-11
#> [11]  1.888432e-12  4.538551e-11 -1.501838e-10 -4.266144e-11 -5.042058e-11
#> [16] -1.166830e-10
(ind$upr - vast_i$upr[vast_i$est != 0]) / vast_i$upr[vast_i$est != 0]
#>  [1]  1.043929e-10  1.163336e-10  7.117150e-11  2.948787e-10  1.065150e-09
#>  [6]  5.742601e-11  6.985702e-10  5.833599e-12  2.339451e-11  1.346229e-10
#> [11] -3.716509e-11  1.588170e-10 -3.453940e-11  3.589164e-10 -1.157047e-10
#> [16]  5.908056e-10
(ind$lwr - vast_i$lwr[vast_i$est != 0]) / vast_i$lwr[vast_i$est != 0]
#>  [1]  5.019277e-11 -1.146071e-10 -1.300644e-11 -4.387211e-10 -9.440731e-10
#>  [6]  7.745630e-11 -7.752092e-10 -3.241492e-11  9.931614e-11  4.291326e-11
#> [11]  4.094497e-11 -6.804505e-11 -2.658282e-10 -4.442385e-10  1.486456e-11
#> [16] -8.241762e-10
```

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'
```