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_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.

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

dat_ll <-  dplyr::transmute(dat_ll,
                       catch_kg = Catch_KG,
                       year = as.integer(Year),
                       vessel = "missing",
                       effort = AreaSwept_km2,
                       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_catch_effort_offset.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[, "catch_kg"],
    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 1.61e-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.

```
fit$parameter_estimates$diagnostics
#>          Param starting_value      Lower       MLE    Upper final_gradient
#> 1   ln_H_input      0.3210640 -5.000000 0.3210771 5.000000  -7.970873e-10
#> 2   ln_H_input      0.2616741 -5.000000 0.2616796 5.000000  -6.795969e-09
```

```
#> 3      beta1_ft    -2.3906380      -Inf -2.3902195      Inf  9.213279e-10
#> 4      beta1_ft    -2.4512048      -Inf -2.4507333      Inf -1.327608e-09
#> 5      beta1_ft    -1.9778802      -Inf -1.9774472      Inf  3.051337e-11
#> 6      beta1_ft    -1.6425668      -Inf -1.6420976      Inf -7.145715e-10
#> 7      beta1_ft    -1.6160390      -Inf -1.6156382      Inf  2.148502e-09
#> 8      beta1_ft    -1.5777705      -Inf -1.5773357      Inf  2.518696e-10
#> 9      beta1_ft    -1.3662943      -Inf -1.3658665      Inf  1.089671e-09
#> 10     beta1_ft    -1.7509292      -Inf -1.7505093      Inf  1.111537e-09
#> 11     beta1_ft    -1.3663341      -Inf -1.3659006      Inf  5.602825e-10
#> 12     beta1_ft    -1.3177830      -Inf -1.3172927      Inf -1.470619e-09
#> 13     beta1_ft    -1.5495949      -Inf -1.5491398      Inf -1.065370e-09
#> 14     beta1_ft    -1.5352830      -Inf -1.5348487      Inf  4.481926e-11
#> 15     beta1_ft    -1.2234577      -Inf -1.2229935      Inf -9.496386e-10
#> 16     beta1_ft    -0.9384373      -Inf -0.9380745      Inf  4.061519e-09
#> 17     beta1_ft    -1.2320043      -Inf -1.2315269      Inf -1.525231e-09
#> 18     beta1_ft    -1.1466206      -Inf -1.1461577      Inf -7.821725e-10
#> 19  L_omega1_z     7.5000170      -Inf  7.4999297      Inf -8.734418e-09
#> 20 L_epsilon1_z    0.3688638      -Inf  0.3688707      Inf -3.945366e-07
#> 21    logkappa1   -4.6866147 -6.765487 -4.6865996 -1.659642  1.121330e-07
#> 22     beta2_ft     1.8220260      -Inf  1.8220034      Inf  7.041621e-10
#> 23     beta2_ft     2.3175663      -Inf  2.3175567      Inf  4.816982e-10
#> 24     beta2_ft     2.1082308      -Inf  2.1082065      Inf  3.879777e-10
#> 25     beta2_ft     1.3131405      -Inf  1.3131141      Inf  2.193090e-10
#> 26     beta2_ft     2.0897872      -Inf  2.0897392      Inf  4.980905e-12
#> 27     beta2_ft     1.6338001      -Inf  1.6337857      Inf  4.155964e-10
#> 28     beta2_ft     1.9105308      -Inf  1.9105129      Inf  3.586251e-10
#> 29     beta2_ft     2.0885364      -Inf  2.0884917      Inf  1.334328e-10
#> 30     beta2_ft     1.6441588      -Inf  1.6441419      Inf  2.028813e-10
#> 31     beta2_ft     1.9781073      -Inf  1.9781127      Inf -3.630518e-10
#> 32     beta2_ft     2.4912904      -Inf  2.4913055      Inf -4.953584e-10
#> 33     beta2_ft     2.7784873      -Inf  2.7784598      Inf  3.741718e-10
#> 34     beta2_ft     2.5744374      -Inf  2.5743824      Inf -3.410676e-10
#> 35     beta2_ft     2.6073084      -Inf  2.6072592      Inf -1.668568e-10
#> 36     beta2_ft     2.5265970      -Inf  2.5266217      Inf -8.895285e-10
#> 37     beta2_ft     2.7320202      -Inf  2.7319932      Inf  1.285443e-10
#> 38  L_omega2_z     2.2008979      -Inf  2.2008979      Inf -9.232295e-09
#> 39 L_epsilon2_z    1.4843805      -Inf  1.4843901      Inf -4.477340e-08
#> 40    logkappa2   -2.8444914 -6.765487 -2.8445046 -1.659642  5.074131e-08
#> 41    logSigmaM    0.3340395      -Inf  0.3340399 10.000000 -5.874145e-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
```

```r
f1 <- here("species_specific_code", "GOA", species,
           "index_comparison", "fit_sdmTMB_catch_effort_offset.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(
  catch_kg ~ 0 + year_f,
  data = dat,
  mesh = mesh,
  family = delta_gamma(type = "poisson-link"),
  time = "year",
  spatial = "on",
  spatiotemporal = "iid",
  offset = log(dat$effort),
  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 = f1)
} else {
fit_sdmTMB <- readRDS(f1)
}
#> attempting to improve convergence with optimHess
#> running TMB sdreport

# diagnose estimation issues due to model structure
#TMBhelper::check_estimability(fit_sdmTMB$tmb_obj)
```
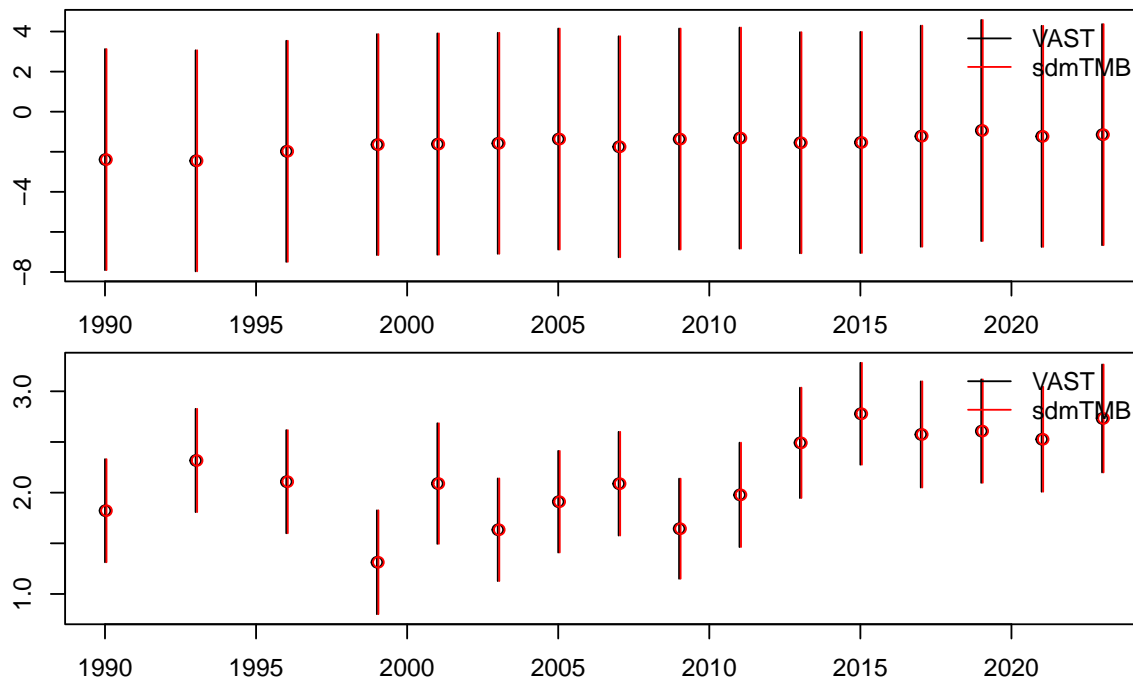
We wrote some custom code to extract comparable parameters (not shown above). Here are the annual mean estimates in link space with 95% confidence intervals for the two components to the delta model:

```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_catch_effort_offset.RDS")
if (!file.exists(f2)) {
```

```r
p <- predict(fit_sdmTMB, newdata = pred_grid, return_tmb_object = TRUE)
saveRDS(p, file = f2)
} else {
p <- readRDS(f2)
}

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

Now, we can compare the indices.

```r
sdm_i <- ind %>% mutate(index = "sdmTMB")
vast_i <- read.csv(here("species_specific_code", "GOA", species,
                        "index_comparison", "Index_catch_effort_offset.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$upr)) +
  ggtitle(species) +
  coord_cartesian(expand = FALSE)
```
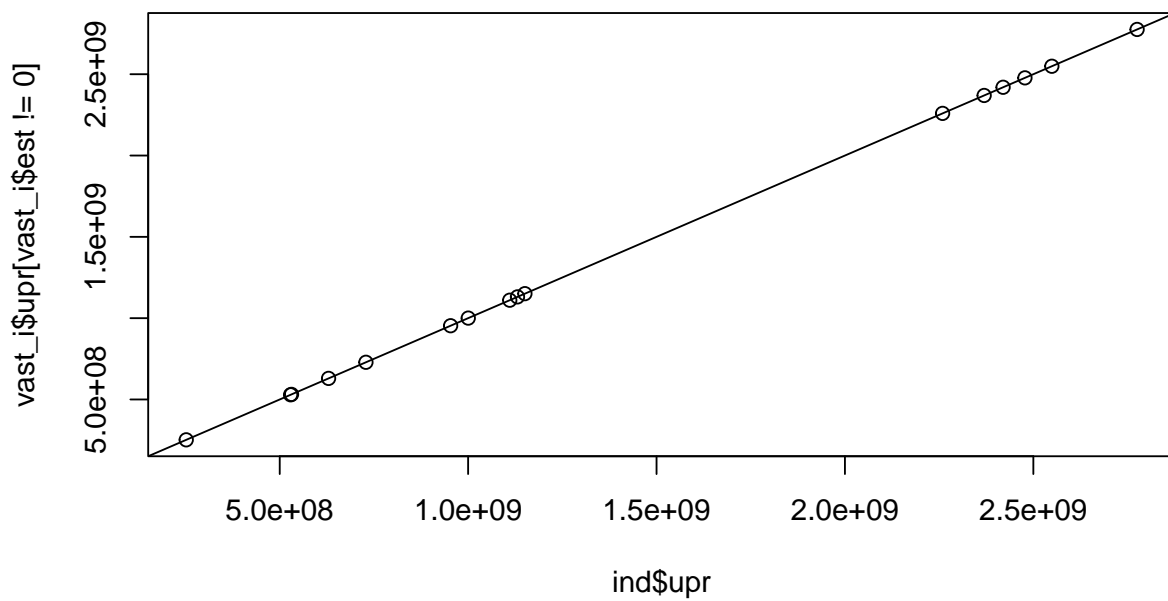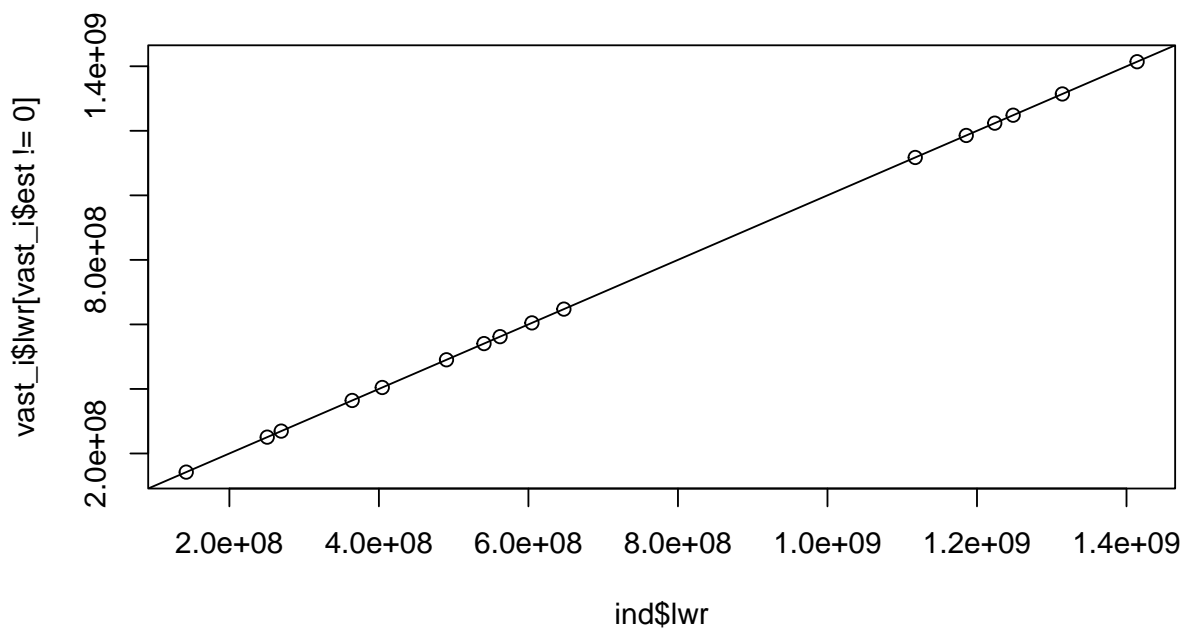
Sebastes_alutus
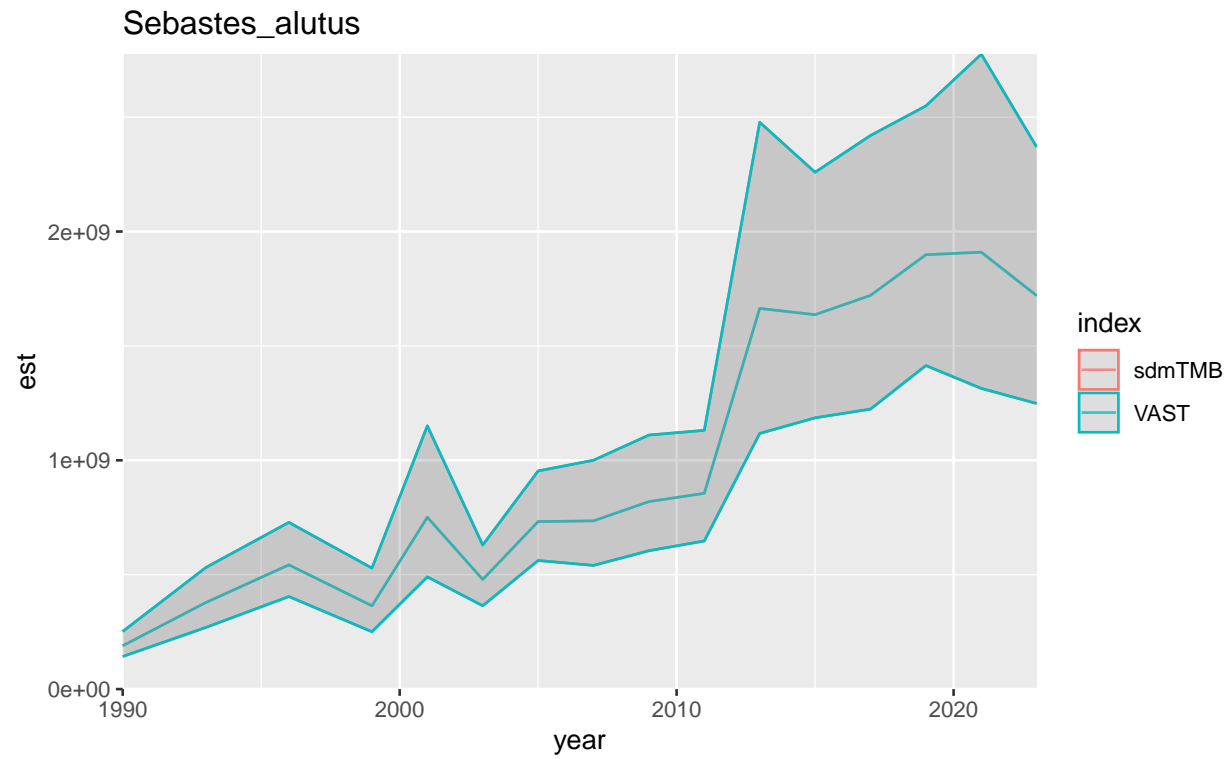
```
plot(ind$est, vast_i$est[vast_i$est != 0]);abline(0, 1)
```
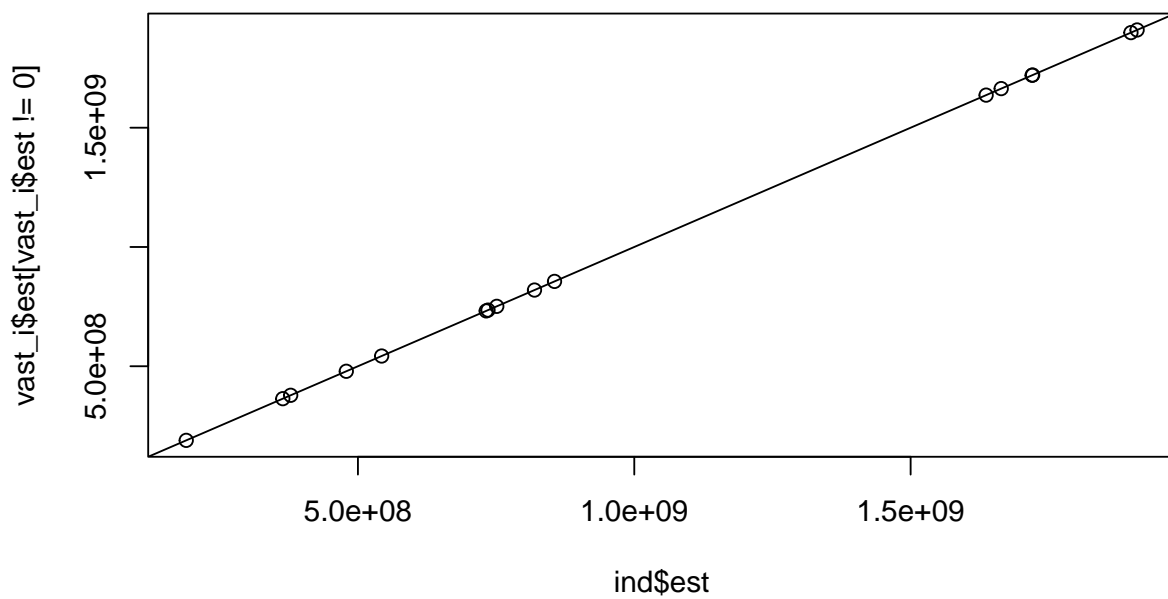


```
plot(ind$upr, vast_i$upr[vast_i$est != 0]);abline(0, 1)
```

```
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]  1.602429e-10 -7.361174e-10 -1.660134e-10 -3.196986e-10 -5.708784e-10
#>  [6] -2.380097e-10 -1.173417e-10 -1.280400e-10 -1.600541e-10 -2.154832e-10
#> [11] -4.500626e-10 -2.993315e-10 -2.246282e-10 -8.809154e-11 -1.979570e-10
#> [16]  1.467326e-10
(ind$upr - vast_i$upr[vast_i$est != 0]) / vast_i$upr[vast_i$est != 0]
#>  [1] -5.415934e-10  6.336970e-11 -9.210340e-10 -1.798561e-09 -7.614243e-09
#>  [6] -9.186003e-10 -1.385178e-09  4.988437e-10 -9.095906e-10 -3.959855e-09
#> [11] -4.790135e-09 -2.817192e-09 -2.997854e-09 -2.593286e-09 -3.619931e-10
#> [16] -3.908646e-09
(ind$lwr - vast_i$lwr[vast_i$est != 0]) / vast_i$lwr[vast_i$est != 0]
#>  [1]  8.620766e-10 -1.535607e-09  5.890044e-10  1.159165e-09  6.472487e-09
#>  [6]  4.425793e-10  1.150493e-09 -7.549234e-10  5.894840e-10  3.528889e-09
#> [11]  3.890012e-09  2.218524e-09  2.548600e-09  2.417106e-09 -3.392134e-11
#> [16]  4.202114e-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'
```