sdmTMB + DCRAB

2023-01-20

Data loading and cleaning

```
library(sdmTMB)
library(mgcv)
```

```
Loading required package: nlme
```

```
This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

```
library(ggeffects)
library(ggplot2)

d = readRDS("df_full_with_dist_to_closed_areas_not_final_20230120.rds")

# Filter out NAs
d <- dplyr::filter(d,
                   !is.na(weighted_fuel_pricegal),
                   !is.na(weighted_crab_ppp),
                   open_closed == "open",
                   season %in% c("2007-2008","2008-2009") == FALSE)
d$yearn <- as.numeric(substr(d$season,1,4))
d$yearf <- as.factor(d$yearn)
winter <- dplyr::filter(d, winter_summer == "Winter")

# try smooth over months
winter$month_n <- 1
winter$month_n[which(winter$month_name=="January")] = 2
winter$month_n[which(winter$month_name=="February")] = 3
```

```r
winter$month_n[which(winter$month_name=="March")] = 4
winter$month_n[which(winter$month_name=="April")] = 5

# Add UTM columns (zone 10)
winter = add_utm_columns(winter, ll_names = c("grd_x", "grd_y"))
```

```
Detected UTM zone 10N; CRS = 32610.
```

```
Visit https://epsg.io/32610 to verify.
```

Initial model: covariates only

```r
mesh <- make_mesh(winter, xy_cols = c("X","Y"), cutoff = 10)
mesh$mesh$n

fit0 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                 season +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
              family = tweedie(),
              mesh = mesh,
              spatial = "off",
              spatiotemporal = "off",
              data = winter,
              time = "yearf")

#sanity(fit0)
```

```r
AIC(fit0)
```

```
[1] 798492.8
```

```
#sanity(fit0)
```

Adding spatial and spatiotemporal fields (seasons)

```
fit1 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                 season +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
              family = tweedie(),
              mesh = mesh,
              spatial = "on",
              spatiotemporal = "off",
              data = winter,
              time = "yearf")
#sanity(fit1)

fit2 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                 season +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
              family = tweedie(),
              mesh = mesh,
```

```
                spatial = "on",
                spatiotemporal = "iid",
                data = winter,
                time = "yearf")
#sanity(fit2)
```

Model 1 seems to have some convergence issues – but Model 2 seems to do a lot better (adding spatiotemporal fields)

```
AIC(fit1, fit2)
```

```
     df      AIC
fit1 38 740665.4
fit2 39 730252.2
```

Just as a test, we can see if changing month to a smooth improves the fit

```
fit3 <- sdmTMB(tottraps ~ 0 + s(month_n, k = 3) + # <- new
                 OR_WA_waters +
                 season +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
              family = tweedie(),
              mesh = mesh,
              spatial = "on",
              spatiotemporal = "iid",
              data = winter,
              time = "yearf")
#sanity(fit3)
```

Model 3 also has no convergence issues –

```
AIC(fit3)
```

```
[1] 730257.6
```

```
# Just as a test, we can see if changing year/season to a smooth improves the fit
fit4 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                 s(yearn) + # <- new
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
              family = tweedie(),
              mesh = mesh,
              spatial = "on",
              spatiotemporal = "iid",
              data = winter,
              time = "yearf")
#sanity(fit4)
```

That model results in lots of convergence issues / warnings.

Similarly, we can change the IID spatiotemporal fields in model 2 to "AR1" to test the autoregressive structure

```
# Is there support for AR1 spatiotemporal fields?
# Using the best model (fit2)
fit5 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                 season +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
```

```
                  poly(weighted_fuel_pricegal,2) +
                  poly(weighted_crab_ppp,2) +
                   poly(dist_to_closed_km,2),
                family = tweedie(),
                mesh = mesh,
                spatial = "on",
                spatiotemporal = "ar1",# <- new
                data = winter,
                time = "yearf")
#sanity(fit5)
```

Model 5 seems to generally converge and has lower AIC than model 2, with rho being estimated ~ 0.51

```
AIC(fit2, fit5)
```

```
     df      AIC
fit2 39 730252.2
fit5 40 730120.3
```

Adding spatial and spatiotemporal fields (months)

Here we switch indexing of spatiotemporal fields to "month_name" and again can try the spatiotemporal fields as IID or AR1. Model 6 seems to generally converge – but model 7 struggles a bit more with variance parameter `thetaf`

```
# Other questions are whether it makes more sense to switch the spatiotemporal fields to m
fit6 <- update(fit2,
               time = "month_n")

fit7 <- update(fit2,
               time = "month_n",
               spatiotemporal = "ar1")
```

Adding fancier smooths

This was my attempt at some models with fancier smooths – but unfortunately, they don't converge well

```
fit8 <- update(fit2,
               formula = tottraps ~ 0 + month_name + OR_WA_waters +
                 yearf +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2),
               time = "month_n",
               spatiotemporal = "ar1")

fit9 <- update(fit2, formula = tottraps ~ 0 + month_name + OR_WA_waters +
                 yearf +
                SST_avg +
                wind_avg +
                poly(depth_zonal_mean,2) +
                poly(depth_zonal_sd,2) +
                poly(faults_km,2) +
                poly(dist_canyon_km,2) +
                poly(weighted_dist,2) +
                poly(weighted_fuel_pricegal,2) +
                poly(weighted_crab_ppp,2) +
                 poly(dist_to_closed_km,2))
```

Comparing delta- models to the Tweedie distribution

Maybe should have done this first, but one thing that's worth doing is also swapping in a delta-Gamma hurdle or delta-model for the Tweedie, which should increase flexibility (particularly for lots of 0s).

This model is a lot slower than the straight Tweedie above...

```
# tic()
# fit10 <- update(fit5,
#                 family = delta_gamma())
# toc()
```

```
fit10 <- sdmTMB(tottraps ~ 0 + month_name + OR_WA_waters +
                  yearf +
                 SST_avg +
                 wind_avg +
                 poly(depth_zonal_mean,2) +
                 poly(depth_zonal_sd,2) +
                 poly(faults_km,2) +
                 poly(dist_canyon_km,2) +
                 poly(weighted_dist,2) +
                 poly(weighted_fuel_pricegal,2) +
                 poly(weighted_crab_ppp,2) +
                  poly(dist_to_closed_km,2),
               family = delta_gamma(),
               mesh = mesh,
               spatial = "on",
               spatiotemporal = "ar1",
               data = winter,
               time = "yearf")
```

The delta-model is quite a bit better than the Tweedie – but takes longer to run (80 minutes on my slow computer)

```
AIC(fit5, fit10)
```

```
      df      AIC
fit5  40 730120.3
fit10 77 714771.4
```

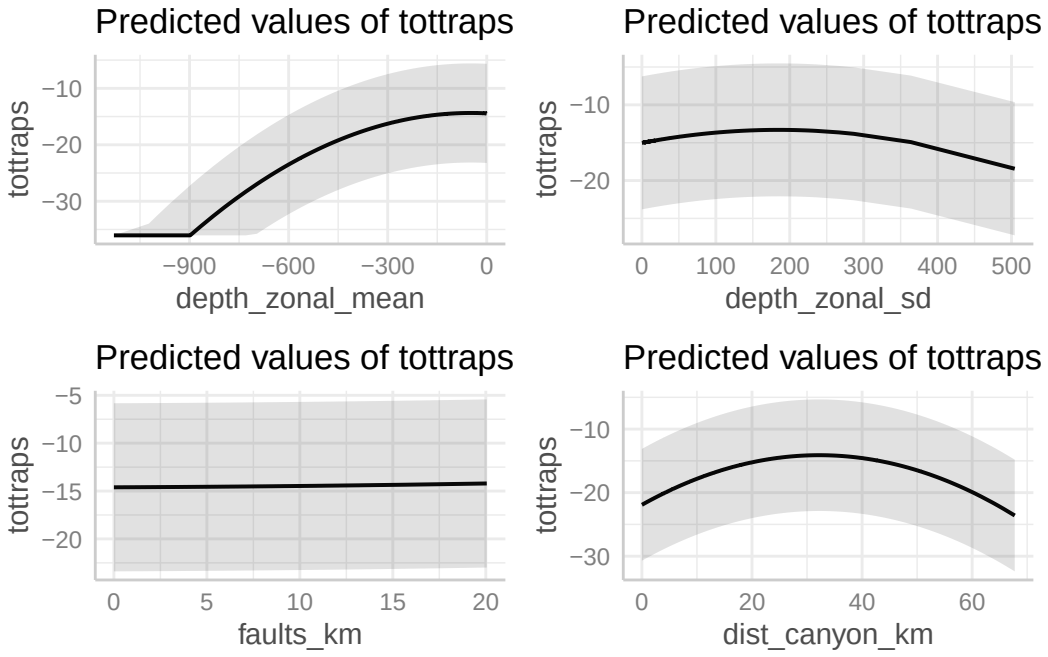
Gut check: do estimated relationships make sense?

This little function is just for helping to display the effects on log scale

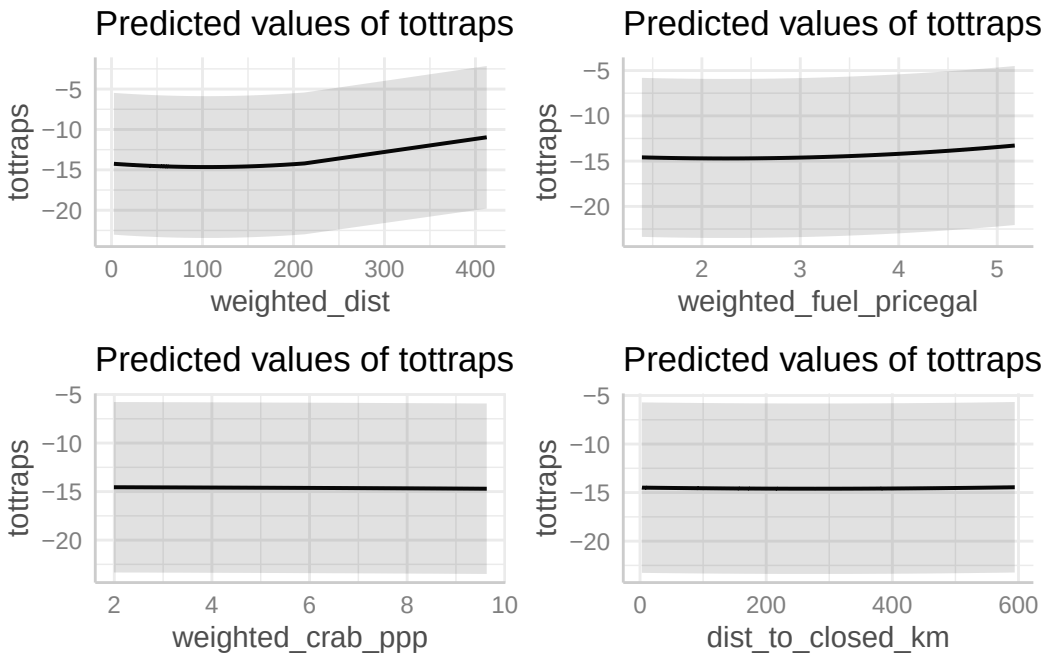
```
plot_log = function(object, term) {
  g <- ggeffect(object, term, back.transform = FALSE)
  g$conf.low <- log(g$conf.low)
  g$conf.high <- log(g$conf.high)
  g$predicted <- log(g$predicted)
  plot(g)
}
```

```
p1 <- plot_log(fit5, "depth_zonal_mean [all]")
p2 <- plot_log(fit5, "depth_zonal_sd [all]")
p3 <- plot_log(fit5, "faults_km [all]")
p4 <- plot_log(fit5, "dist_canyon_km [all]")

gridExtra::grid.arrange(p1,p2,p3,p4,nrow=2)
```

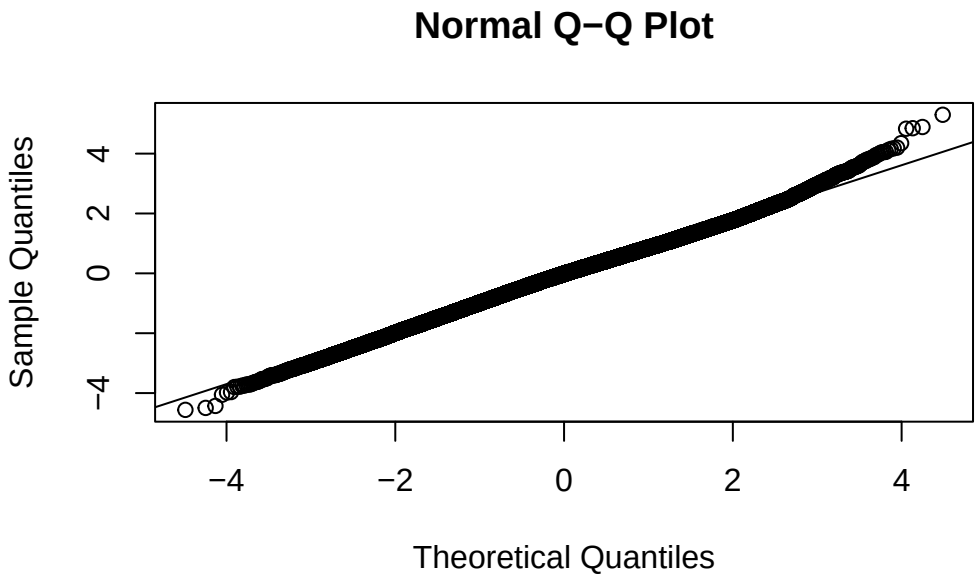



```
p1 <- plot_log(fit5, "weighted_dist [all]")
p2 <- plot_log(fit5, "weighted_fuel_pricegal [all]")
p3 <- plot_log(fit5, "weighted_crab_ppp [all]")
p4 <- plot_log(fit5, "dist_to_closed_km [all]")
gridExtra::grid.arrange(p1,p2,p3,p4,nrow=2)
```



Residuals – do qqplots look ok?

```
res <- residuals(fit5)
qqnorm(res)
qqline(res)
```



The slower and more robust check is:

```
mcmc_res <- residuals(fit5, type = "mle-mcmc", mcmc_iter = 101, mcmc_warmup = 100)
qqnorm(mcmc_res)
qqline(mcmc_res)
```

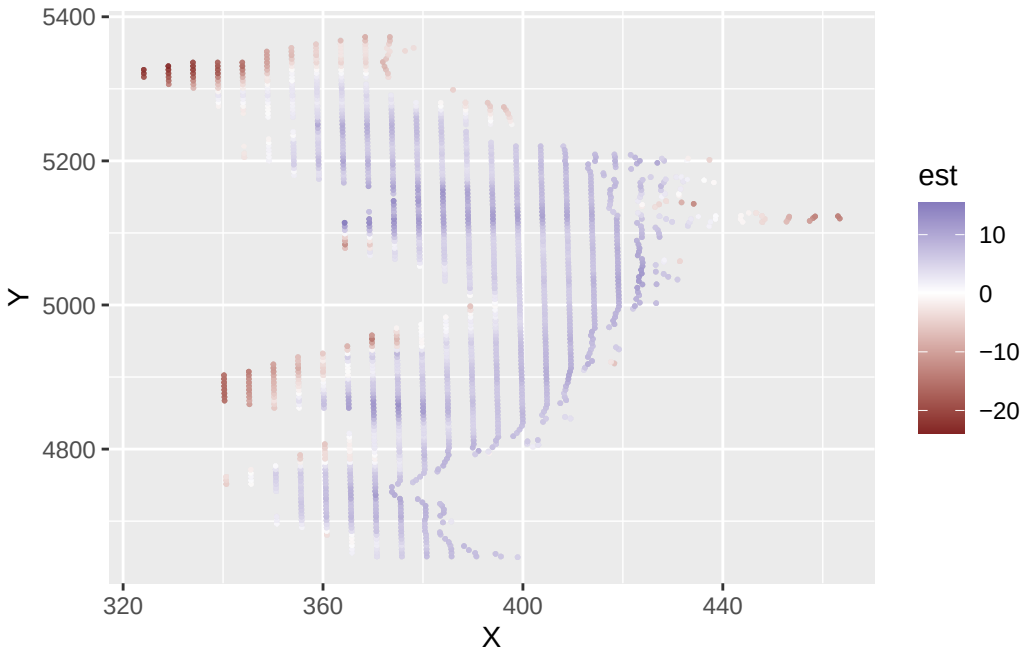
Spatial predictions – make sense?

```
pred <- winter
pred$OR_WA_waters <- winter$OR_WA_waters[1]
pred$season <- "2018-2019"
pred$SST_avg <- mean(winter$SST_avg)
pred$wind_avg <- mean(winter$wind_avg)
pred$depth_zonal_mean <- mean(winter$depth_zonal_mean)
pred$depth_zonal_sd <- mean(winter$depth_zonal_sd)
pred$faults_km <- mean(winter$faults_km)
pred$dist_canyon_km <- mean(winter$dist_canyon_km)
pred$weighted_dist <- mean(winter$weighted_dist)
pred$weighted_fuel_pricegal <- mean(winter$weighted_fuel_pricegal)
pred$weighted_crab_ppp <- mean(winter$weighted_crab_ppp)
pred$dist_to_closed_km <- mean(winter$dist_to_closed_km)

pred <- predict(fit5, pred)
```

```
as(<dgCMatrix>, "dgTMatrix") is deprecated since Matrix 1.5-0; do as(., "TsparseMatrix") inst
```

```
p <- ggplot(dplyr::filter(pred,yearf=="2019"), aes(X,Y, col = est)) +
  geom_point(size=0.3,alpha=0.3) +
  scale_color_gradient2()
p
```



TODOs

1. Transformation of data: there's a number of variables (distances and depths) that could be log-transformed and then standardized, but many contain 0s. So my cheat above was to make them all quadratic. Would be good to get thoughts on this – it effectively allows things to not be perfectly linear, and avoids wonky transformations
2. Delta-gamma vs Tweedie: some more tests could be done with this comparison
3. From best model, (1) do some residual diagnostics with QQ plots, (2) Plot predictions of spatial and spatiotemporal fields spatially
4. From best model, use visreg or ggeffects to visualize effects of quadratic predictors – do these make sense?
5. Think about Model 2 vs Model 5 above – do we want spatiotemporal fields indexed by year or month? If the latter, do we also want a spatial field (for Model 5, spatiotemporal sigma » spatial sigma, but opposite is true for Model 2)