Random Grid Results

Alex Tidd

16 November, 2018

```
library(plyr)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.3.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(reshape)
```

```
##
## Attaching package: 'reshape'
```

```
## The following object is masked from 'package:dplyr':
##
##     rename
```

```
## The following objects are masked from 'package:plyr':
##
##     rename, round_any
```

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 3.3.2
```

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: magrittr
```

```
##
## Attaching package: 'ggpubr'
```

```
## The following object is masked from 'package:plyr':
##
##     mutate
```

```
library(ggpmisc)
```

```
## Warning: package 'ggpmisc' was built under R version 3.3.2
```

```
library(mgcv)
```

```
## Warning: package 'mgcv' was built under R version 3.3.2
```

```
## Loading required package: nlme
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
## This is mgcv 1.8-22. For overview type 'help("mgcv-package")'.
```

```
library(RColorBrewer)
library(RPostgreSQL)
```

```
## Loading required package: DBI
```

```
## Warning: package 'DBI' was built under R version 3.3.2
```

```
drv =dbDriver("PostgreSQL")
#laurie db
con=dbConnect(drv,
              host ='wklife.csrzweaa3tbm.eu-west-2.rds.amazonaws.com',
              dbname  ='wklife',
              port    =5432,
              user    ='mydas',
              password='Yes_Meski')

empd_pm=dbGetQuery(con, "select * from randgridpm")
```

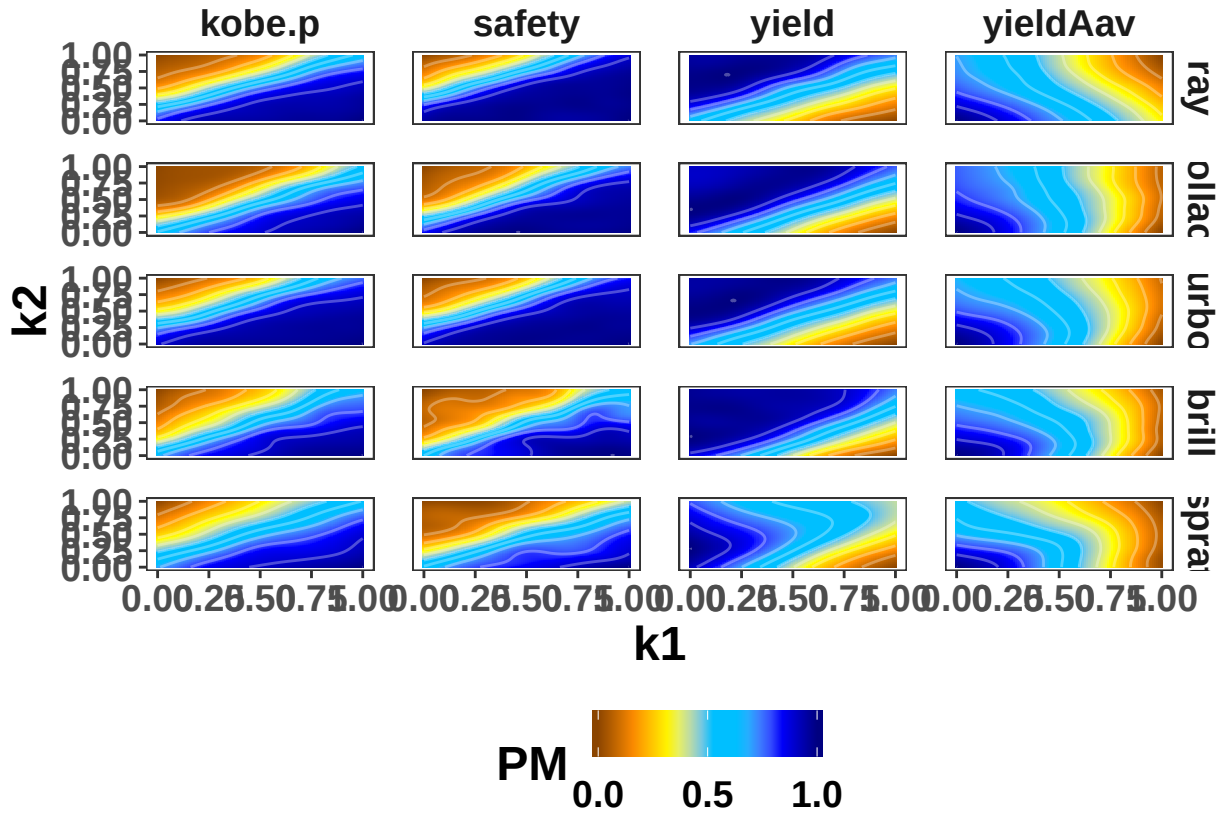
```
empd_pm$kobe.p=empd_pm$kobe.n/45
empd_pm$yieldAav = pmin(0.5,empd_pm$yieldAav)
empd_pm$yieldAav = 1 - empd_pm$yieldAav
test=melt(empd_pm, id.vars=c("spp","k1","k2"),measure.vars=c("safety","kobe.p","yield","yieldAav"))
```

modelled display using GAM

```
pairwise = list()
for (i in c("safety","kobe.p","yield","yieldAav")){
  for (j in c("brill","turbot","ray","pollack","sprat")){
    a    = subset(test, spp==j & variable %in% c(i))
    spl1 <- gam(value ~ s(k1, k2, bs = 'sos'), data = a)
    # fine grid, coarser is faster
    datmat2 <- data.frame(expand.grid(k1 = seq(0, 1, 0.01), k2= seq(0, 1, 0.01)))
    resp <- predict(spl1, datmat2, type = "response")
    datmat2$value <- resp
    datmat2$spp = j
    datmat2$objective = i
    pairwise=rbind(pairwise, datmat2)
  }
}
```

```
minMax=function(x,na.rm=TRUE) (x-min(x,na.rm=na.rm))/diff(range(x,na.rm=na.rm))
pairwise2=ddply(pairwise,.(spp,objective),transform, var2=minMax(value))
```

```
pairwise2$spp = factor(pairwise2$spp, levels=c("ray","pollack","turbot","brill", "sprat"))
mycol = rev(c("navy", "blue", "deepskyblue1", "deepskyblue", "yellow", "darkorange", "darkorange4"))
pairwise2=pairwise2[!is.na(pairwise2$spp), ]
ggplot(pairwise2) +
  aes(x = k1, y = k2, z = var2, fill = var2) +
  geom_tile() +
  #coord_equal() +
  #geom_contour(color = "white", alpha = 0.5) +
  #scale_fill_distiller("",palette="Spectral", na.value="white", direction=1) +
  #geom_contour(color = "white", alpha = 0.1) +
  scale_fill_gradientn("PM",colours=mycol, breaks=c(0, 0.5, 1))+
  geom_contour(color = "white", alpha = 0.3) +
  #scale_fill_distiller("",palette="Spectral", na.value="white", direction=1, breaks=c(0, 0.5, 1))+
  theme_bw()+facet_grid(spp~objective) +
  theme(text = element_text(size=18, face="bold"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank(),
        legend.position="bottom",panel.spacing = unit(1, "lines")
  ) + ylab("k2")+xlab("k1")
```



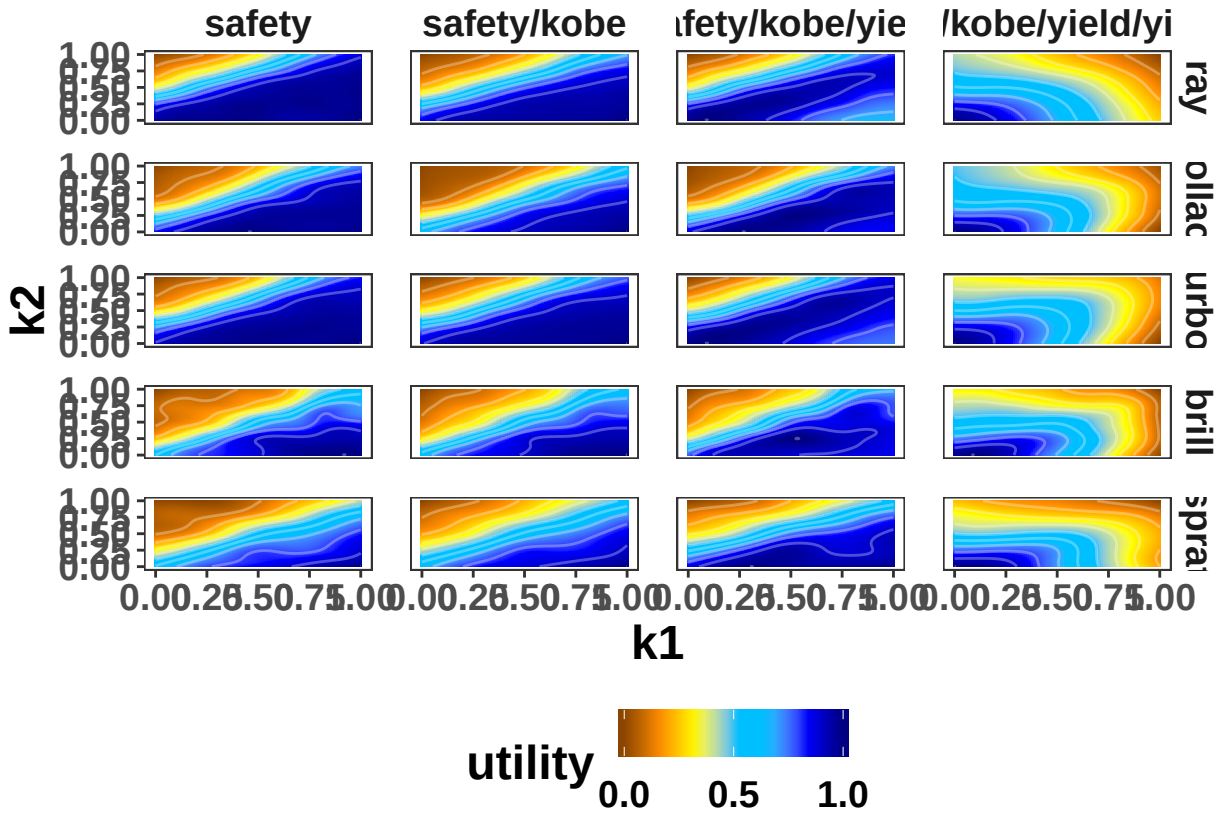
utilities

```
pair=reshape( pairwise, idvar = c("spp","k1","k2"), v.names = "value", timevar = "objective",
              direction = "wide")
colnames(pair)[4:7] = c("safety","kobe.p","yield","yieldAav")
```

```
pair$yieldAav=pair$yieldAav*10
pair$s_k=apply(pair[c(4,5)],c(1),sum,na.rm=t)/2
#pair$s_y=apply(pair[c(4,6)],c(1),sum,na.rm=t)/2
pair$s_k_y=apply(pair[c(4:6)],c(1),sum,na.rm=t)/3
pair$s_k_y_y=apply(pair[c(4:7)],c(1),sum,na.rm=t)/4


pair2=melt(pair, id.vars=c("spp","k1","k2"), measure.vars=c("safety","s_k","s_k_y","s_k_y_y"))
pair2$variable=as.character(pair2$variable)

pair2$variable[pair2$variable=="s_k"] = "safety/kobe"
pair2$variable[pair2$variable=="s_k_y"] = "safety/kobe/yield"
pair2$variable[pair2$variable=="s_k_y_y"] = "safety/kobe/yield/yieldvar"

te=ddply(na.omit(pair2),.(spp,variable),transform, var2=minMax(value))

te$variable = factor(te$variable, levels=c("safety","safety/kobe","safety/kobe/yield","safety/kobe/yiel
te$spp = factor(te$spp, levels=c("ray","pollack","turbot","brill", "sprat"))

te<- te[!is.na(te$spp), ]
```

utilities plots

```
ggplot(subset(te, !(spp %in% c("lobster", "razor")))) +
  aes(x = k1, y = k2, z = var2, fill = var2) +
  geom_tile() +
  #coord_equal() +
  #geom_contour(color = "white", alpha = 0.5) +
  #scale_fill_distiller("",palette="Spectral", na.value="white", direction=1) +
  #geom_contour(color = "white", alpha = 0.1) +
  scale_fill_gradientn("utility",colours=mycol, breaks=c(0, 0.5, 1))+
  geom_contour(color = "white", alpha = 0.3) +
  #scale_fill_distiller("",palette="Spectral", na.value="white", direction=1, breaks=c(0, 0.5, 1))+
  theme_bw()+facet_grid(spp~variable) +
  theme(text = element_text(size=18, face="bold"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank(),
        legend.position="bottom",panel.spacing = unit(1, "lines")
        ) + ylab("k2")+xlab("k1")
```




```
#ggsave(filename='/Users/alextidd/Documents/fig6.png',last_plot(),dpi=300, units='in',width=12,height=1
```

Author information

Alex Tidd. emperorfish@gmail.com

Acknowledgements

This vignette and many of the methods documented in it were developed under the MyDas project funded by the Irish exchequer and EMFF 2014-2020. The overall aim of MyDas is to develop and test a range of assessment models and methods to establish Maximum Sustainable Yield (MSY) reference points (or proxy MSY reference points) across the spectrum of data-limited stocks.