Simulate from a fitted sam model

2017-08-25

The most recent version of TMB facilitates simulating from a fitted model, so we have implemented that ability in `stockassessment`. By default, these simulations are conditional on the estimated values of F and N but it is possible to also simulate F and N forward from the initial time point using the `sim.condRE` argument to the `sam.fit` function.

Read in and organize the data using fuctions specific to the `stockassessment` package.

```
cn <- read.ices("nsher/cn.dat")
cw <- read.ices("nsher/cw.dat")
dw <- read.ices("nsher/dw.dat")
lf <- read.ices("nsher/lf.dat")
lw <- read.ices("nsher/lw.dat")
mo <- read.ices("nsher/mo.dat")
nm <- read.ices("nsher/nm.dat")
pf <- read.ices("nsher/pf.dat")
pm <- read.ices("nsher/pm.dat")
sw <- read.ices("nsher/sw.dat")
surveys <- read.ices("nsher/survey.dat")
dat <- setup.sam.data(surveys=surveys,
                      residual.fleet=cn,
                      prop.mature=mo,
                      stock.mean.weight=sw,
                      catch.mean.weight=cw,
                      dis.mean.weight=dw,
                      land.mean.weight=lw,
                      prop.f=pf,
                      prop.m=pm,
                      natural.mortality=nm,
                      land.frac=lf)
```

Set up the model's configuration.

```
conf <- defcon(dat)
conf$fbarRange <- c(2,6)
conf$corFlag <- 1
conf$keyLogFpar <- matrix(c(
-1,   -1,   -1,   -1,   -1,   -1,   -1,   -1,   -1,
-1,    0,    1,    2,    3,    4,    5,    6,   -1,
-1,    7,   -1,   -1,   -1,   -1,   -1,   -1,   -1,
 8,   -1,   -1,   -1,   -1,   -1,   -1,   -1,   -1), nrow=4, byrow=TRUE)
```

Set up initial values for the parameters based on the data and configuration.

```
par <- defpar(dat,conf)
par$logFpar <- rep(0,9)
```
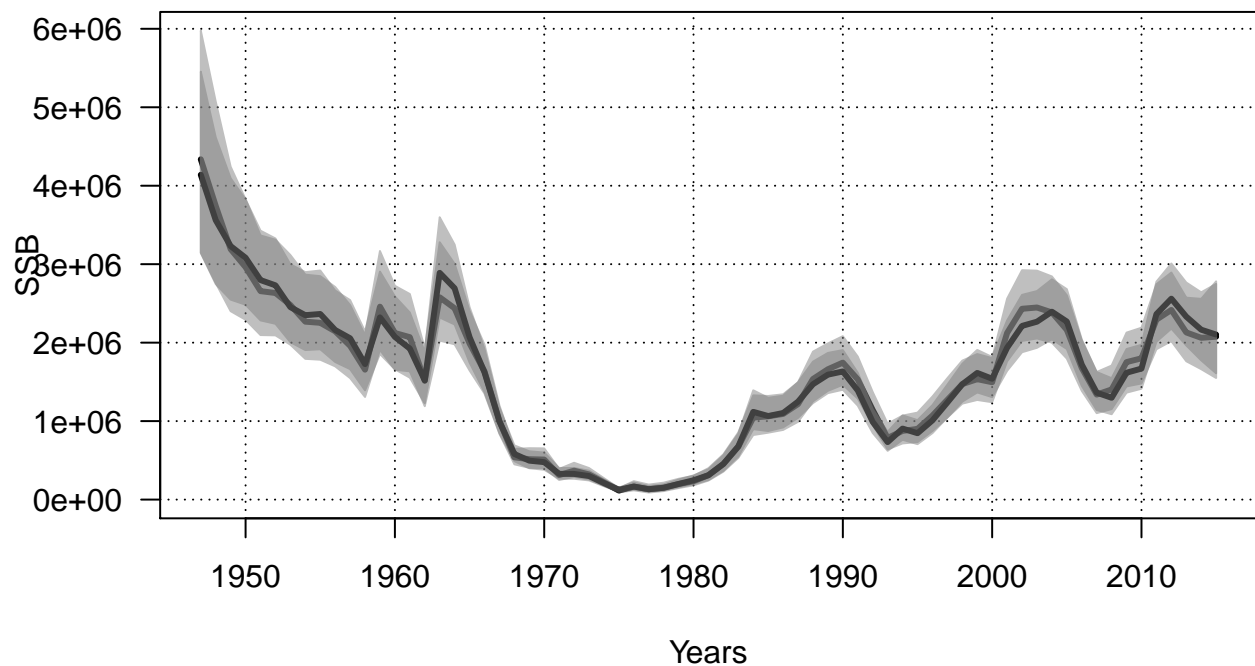
Fit the model.

```
fit <- sam.fit(dat, conf, par)
```

Simulate from the fitted model and store the simulated observations in a new data set `simdat` that has the same structure as the original data.
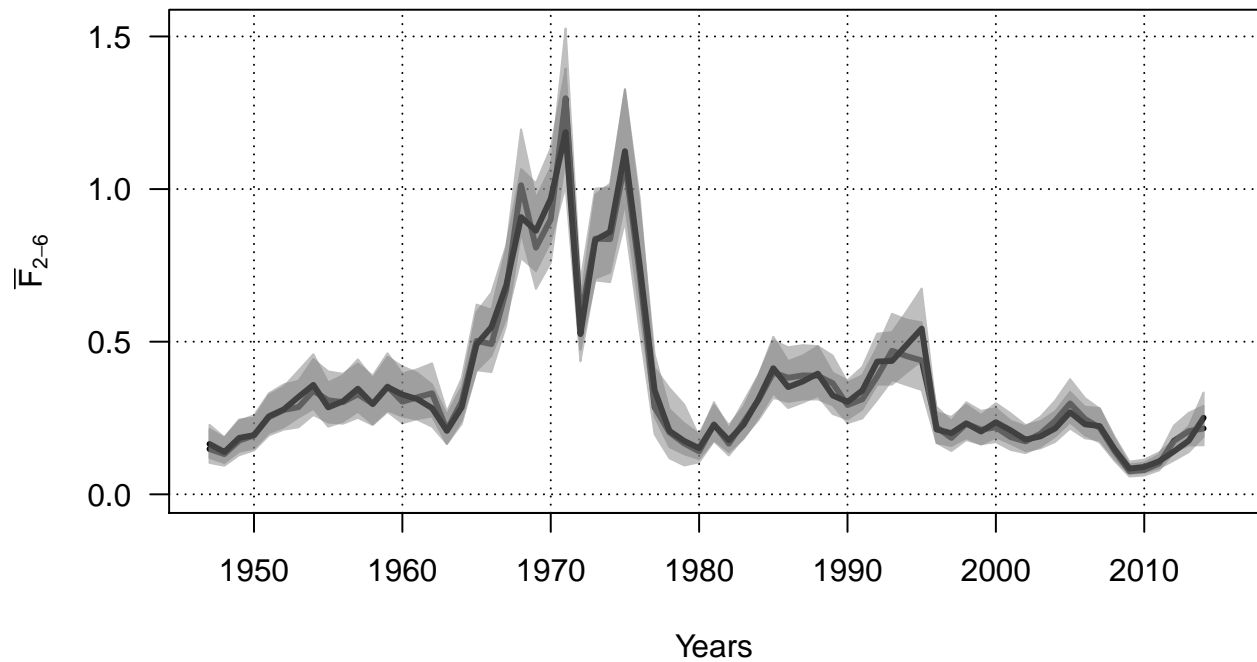
```
simdat <- simulate(fit, seed=1, nsim=1)[[1]]
```

Fit the same model to the simulated data and compare the plot to the original
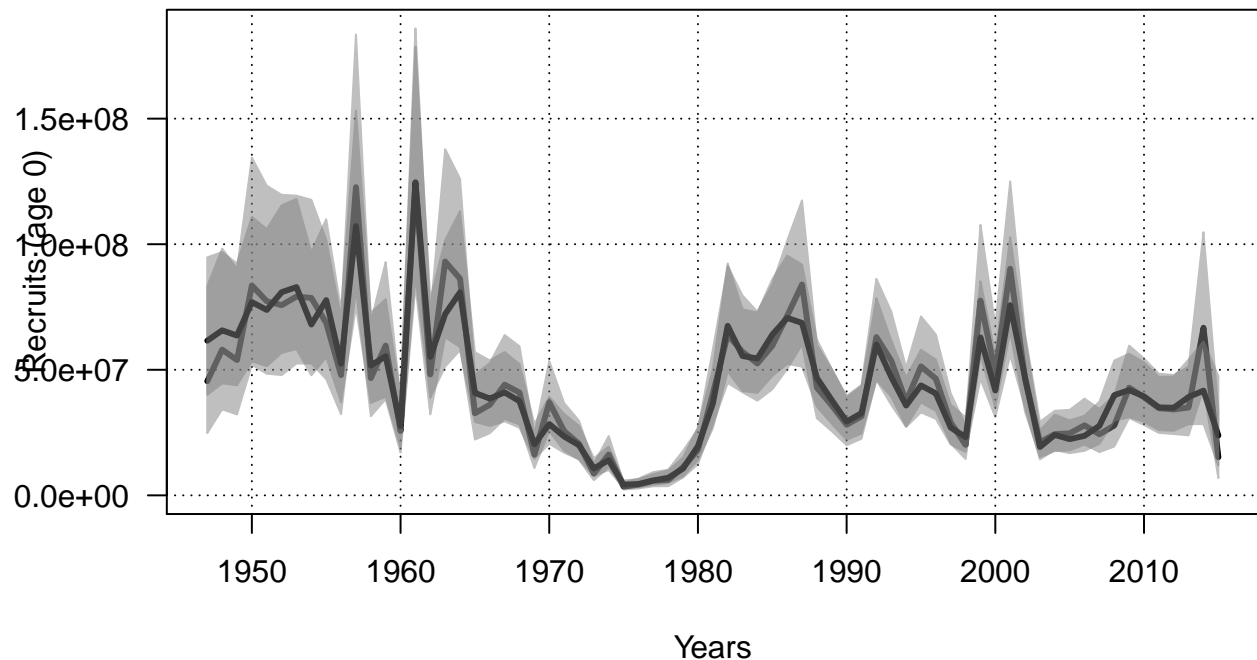
```
simfit <- sam.fit(simdat, conf, par)
ssbplot(fit)
ssbplot(simfit, add=TRUE)
```



```
fbarplot(fit, partial=FALSE)
fbarplot(simfit, add=TRUE, partial=FALSE)
```

```
recplot(fit)
recplot(simfit, add=TRUE)
```


We can also simulate multiple data sets and fit models to the new observations. For this it is best to use the `parLapply` function so that the fits are run in parallel. Fitting them in parallel just requires a few extra steps to set up the cluster, but it will save a lot of time.

```
simlist <- simulate(fit, seed=1, nsim=3)
no_cores <- detectCores() - 1 #how many cores can we use
if( no_cores>2 ) no_cores <- 2 # Cran check does not allow us to use more than two
cl <- makeCluster(no_cores) #set up some number of nodes
```

```
clusterExport(cl, c("conf", "par")) #send these objects to each node
clusterEvalQ(cl, {library(stockassessment)}) #load the package to each node
simfitslist <- parLapply(cl, simlist, function(x){sam.fit(x, conf, par)}) #do sam.fit to each element of
stopCluster(cl) #shut it down
```

```
ssbplot(fit, cicol="red")#the original data
trash <- lapply(simfitslist, function(x){ssbplot(x, add=TRUE)})
trash <- lapply(simfitslist, function(x){ssbplot(x, ci=FALSE, add=TRUE)})
```