Basic run example

LB

18 October 2018

A short vignette showing how to run the basic length frequency analysis method on an example dataset. The example dataset in this case is a cleaned up version of white-bellied anglerfish data from the surveys that cover ICES subarea VII and divisions VIII a,b,d stock.

This method allows multiple surveys from different times of year.

Loading data and formatting

Objective functions need to be compiled. Set root directory for "LFEM" folder so the function can find the .dll files and other files aswell.

```
dllroot<-"C:/Users/LukeB/Documents/LFEM/"
```

```
#install.packages(c("Matrix","data.table","plyr","reshape","TMB","ggplot2"))
library(TMB)
library(plyr)
library(reshape)
library(ggplot2)
library(data.table)
load(paste(dllroot,"data/lfdat_MON.RData",sep=""))
```

The dataframe should be formatted as below for the function, with four columns: Survey, Year, Length and RF (raising factor). The data has been aggregated by year, survey and length so as to compact the dataframe for speed. Observed log-likelihood is the same if individual fish lengths for each haul were used.

```
##    Survey Year Length RF
## 1 SP-PORC 2005      3  1
## 2 IE-IGFS 2007      4  1
## 3   EVHOE 2011      4  1
## 4 IE-IGFS 2012      4  1
## 5 SP-PORC 2004      5  1
## 6 IE-IGFS 2005      5  1
```

```
compile(paste(dllroot,"tmb/constantSD.cpp",sep=""))
compile(paste(dllroot,"tmb/linearSD.cpp",sep=""))
compile(paste(dllroot,"tmb/constantSD_OBSLL.cpp",sep=""))
compile(paste(dllroot,"tmb/linearSD_OBSLL.cpp",sep=""))
```

Source the function

```
source(paste(dllroot,"R/basic_fun.R",sep=""))
```

Function arguments and starting parameters

Survey information

Values should be entered in alphabetical order of surveys in each case.

How many years of data in each survey? What year does data for each survey start?

```r
no.years<-c(14,14,14)
year0<-c(2003,2003,2003)
```

age1 is the assumed age of the first component in each of the surveys. In this case we assume that for the first two surveys alphabetically (EVHOE and IE-IGFS) that the first component observed in the length frequency data is approximately 0.875 years old, using the common assumption fish are born on the 1st of Jan (i.e. the midpoint of the fourth quarter of the year when these surveys are conducted). SP-PORC is mainly conducted over september so we set the age1 at 0.73

```r
age1<-c(0.875,0.875,0.73)
```

Starting parameters

- *L* is the mean of the final component
- *l* is the mean of the first component
- *k.reparam* is the starting growth parameter
- sigma.start is the starting standard deviation parameter(s)
- SD.type =
  - 3 -> linear SD relative to means, needs two sigma.start parameters e.g. c(5,10)
  - 4 -> constant SD

Run the function

If rel.tolerance is set at 1e-8 as is standard then this may take a while to converge

```r
test<-basic.LFEM(year0=year0,no.years=no.years,age1=age1,L=(130),l=16,k.reparam=0.83,
sigma.start=c(6,10),No.comp=9,Lengths=lfdat,niter=10000,
SD.type=3,rel.tolerance=1e-8,dllroot=dllroot,sub.Obs.lim = 100)
```

Load the test object if you haven't run the model

```r
load(paste(dllroot,"data/test.RData",sep=""))
```

Results

There are many results from the model

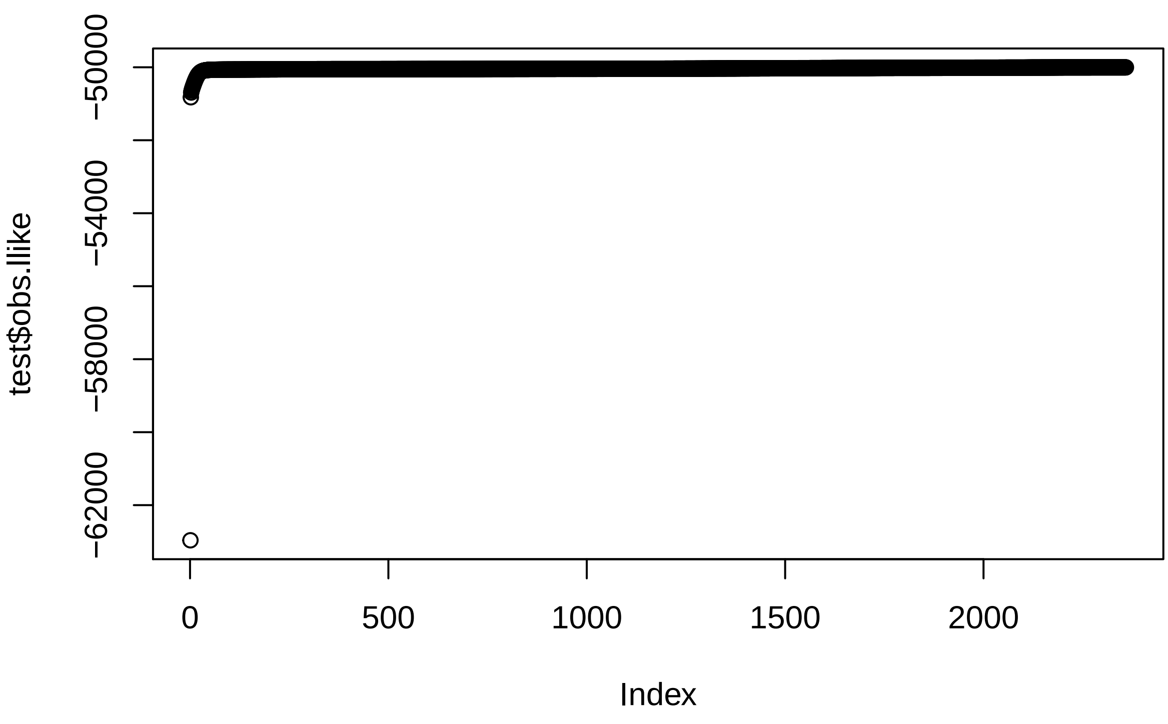
```r
str(test)
```

```
## List of 21
##  $ obs.llike    : num [1:2359] -62963 -50823 -50693 -50640 -50602 ...
##  $ k.reparam.obs: num [1:10000] 0.83 0.831 0.842 0.851 0.858 ...
##  $ L.obs        : num [1:10000] 130 120 122 124 125 ...
##  $ K.obs        : num [1:10000] 0.186 0.185 0.172 0.162 0.153 ...
##  $ Linf.obs     : num [1:10000] 163 150 157 164 170 ...
##  $ Mu           : num [1, 1:9, 1:3] 16.3 29.3 41.8 53.9 65.6 ...
##  $ Sd           : num [1:9] 3.83 4.98 6.09 7.16 8.2 ...
##  $ Lambda       : num [1:14, 1:9, 1:3] 0.253 0.63 0.164 0.31 0.271 ...
##  $ k.reparam    : Named num 0.967
##   ..- attr(*, "names")= chr "logit_k_reparam"
##  $ K            : Named num 0.0332
##   ..- attr(*, "names")= chr "logit_k_reparam"
```

```
##  $ Linf         : Named num 412
##   ..- attr(*, "names")= chr "log_L"
##  $ tzero        : Named num -0.342
##   ..- attr(*, "names")= chr "logit_k_reparam"
##  $ l            : Named num 16.3
##   ..- attr(*, "names")= chr "log_l"
##  $ L            : Named num 109
##   ..- attr(*, "names")= chr "log_L"
##  $ sigma        : num [1:2] 3.83 12.03
##  $ Lambda.params: num 336
##  $ sample.size  : num 12060
##  $ Entropy      : num 5261
##  $ age1         : num [1:3] 0.875 0.875 0.73
##  $ srep         : num [1:10, 1:2] 2.79 4.69 3.39 1.34 2.49 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:10] "log_l" "log_L" "logit_k_reparam" "log_s" ...
##   .. ..$ : chr [1:2] "Estimate" "Std. Error"
##  $ sub.Obs      : num -49210
```

First thing to check is the convergence

```r
plot(test$obs.llike)
```



Then explore the rest of the results

Mu and Lambda are arrays. For example: below shows the nine component means estimated for the first survey

```
test$Mu[,,1]
```

```
## [1]  16.32870  29.26251  41.77402  53.87700  65.58479  76.91031  87.86604
## [8]  98.46404 108.71600
```

model selection criteria

```
source(paste(dllroot,"R/BIC_AIC.R",sep=""))
aic(test)
```

```
## [1] 100689
```

```
bic(test)
```

```
## [1] 103211.6
```

```
icl_bic(test)
```

```
## [1] 113733.3
```

```
sub_aic(test)
```

```
## [1] 99102.49
```

Please see plot_example.R for plotting this model result over the length frequency distributions

Further analysis and sensitivity

Here's some inelegant code to conduct a sensitivity analysis on some of the starting parameters. You may want to leave this running over night as it is over 1400 model runs!!

```
df<-expand.grid(k = seq(0.7,0.99,length=30),L=seq(100,130,length=4),
No.comp=seq(6,14,length=9),krep.final=NA,K.final=NA,Linf.final=NA,
tzero.final=NA,l.final=NA,obs.ll=NA,iclbic=NA,aic=NA,bic=NA,
Entropy=NA,sub.aic=NA)

mu.lst<-list(list())
sd.lst<-list(list())
lambda.lst<-list(list())


for(i in 1:dim(df)[1]){

result_TMB<-basic(year0=year0,no.years=no.years,age1=age1,L=df[i,2],l=16,k.reparam=df[i,1],
sigma.start=c(10),No.comp=df[i,3],Lengths=lfdat,niter=10000,SD.type=4,rel.tolerance=1e-8)

  df$K.final[i]<-result_TMB$K
  df$l.final[i]<-result_TMB$l
  df$L.final[i]<-result_TMB$L
  df$krep.final[i] <-result_TMB$k.reparam
  df$Linf.final[i]<-result_TMB$Linf
  df$tzero.final[i]<-result_TMB$tzero
  df$obs.ll[i]<-max(result_TMB$obs.llike)
  df$iclbic[i]<-icl_bic(result_TMB)
  df$aic[i]<-aic(result_TMB)
  df$bic[i]<-bic(result_TMB)
```

```
  df$Entropy[i]<-result_TMB$Entropy
 df$sub.aic[i]<-sub_aic(result_TMB)

    mu.lst[[i]]<-result_TMB$Mu
  sd.lst[[i]]<-result_TMB$Sd
  lambda.lst[[i]]<-result_TMB$Lambda
}


#ave(ls(),file="constantSDsensruns_MON.RData")
```