Rgadget

Bjarki Þór Elvarsson

þri 16.sep 2014

Interacting with Gadget from R

- Rgadget (still in development) is a collection of functions that aid in the development of and interact with Gadget models
- Its core functions are:
  - gadget.iterative: an implementation of the iterative refweighting algorithm in R.
  - gadget.fit: collects output from Gadget and compares to data.
  - gadget.ypr: calculates Yield-per-recruit curves for a modeled stock
  - gadget.forward: projects the state of the modeled ecosystems based on fixed recruitment or with process error.
  - gadget.boot* : bootstrap variants of functions above
  - gadget.simulate : independent implementation of Gadget in R (not discussed here)

Obtaining Rgadget

- Still under active development and therefor it does not exist in a packaged form.
- Can be obtained from:
  - `https://r-forge.r-project.org/projects/rgadget/`
- The package depends on a number of other packages:

```
install.packages(c('plyr','dplyr','data.table',
                   'parallel','lubridate',
                   'stringr','ggplot2',
                   'gridExtra','reshape2'))
```

gadget.iterative

The gadget.iterative functions implements iterative reweighting heuristic for the likelihood components.

The general idea behind the iterative re-weighing is to assign the inverse variance of the fitted residuals as component weights. The variances, and hence the final weights, are calculated according the following algorithm:

- Calculate the initial SS given the initial parametrization. Assign the inverse SS as the initial weight for all likelihood components.
- For each likelihood component, do an optimization run with the initial score for that component set to 10000. Then estimate the residual variance using the resulting SS of that component divided by the degrees of freedom (df^*), i.e. $\hat{\sigma}^2 = \frac{SS}{df^*}$.
- After the optimization set the final weight for that all components as the inverse of the estimated variance from the step above (weight $= 1/\hat{\sigma}^2$).

gadget.iterative (cont.)

The effective number of data-points (df^*) is used as a proxy for the degrees of freedom determined from the number of non-zero data-points.

- Viewed as satisfactory proxy when the data-set is large
- For smaller data-sets this could be a gross overestimate.

In particular, if the survey indices are weighed on their own while the yearly recruitment is estimated they could be over-fitted. In general problem such as these can be solved with component grouping, that is in step 2 the likelihood components that should behave similarly, such as survey indices, should be heavily weighted and optimized together.

Tusk example

```
tmp <-
  gadget.iterative(rew.sI = TRUE,
     grouping = list(sind = c('si2039','si4069',
                              'si70110'),
                     survey = c('ldist.survey',
                                'alkeys.survey'),
                     catch = c('ldist.catch',
                               'alkeys.catch')),
     params.file = 'params.base',
     optinfofile = 'optinfofile',
     wgts = 'WGTS')
```

gadget.fit

Gadget-models can produce output by a number of dimensions.

- Specialised "printers" are implemented (see chapter 9 of the userguide)
- Output can be aggregated using agg files
- Digesting the output can be an excercise in bookkeeping.
- Enter 'gadget.fit':
  - collects all observation data referred to in the likelihood file
  - requests predictions/model fit from Gadget based on the dimensions in of the observations
  - merges the observations and model fit

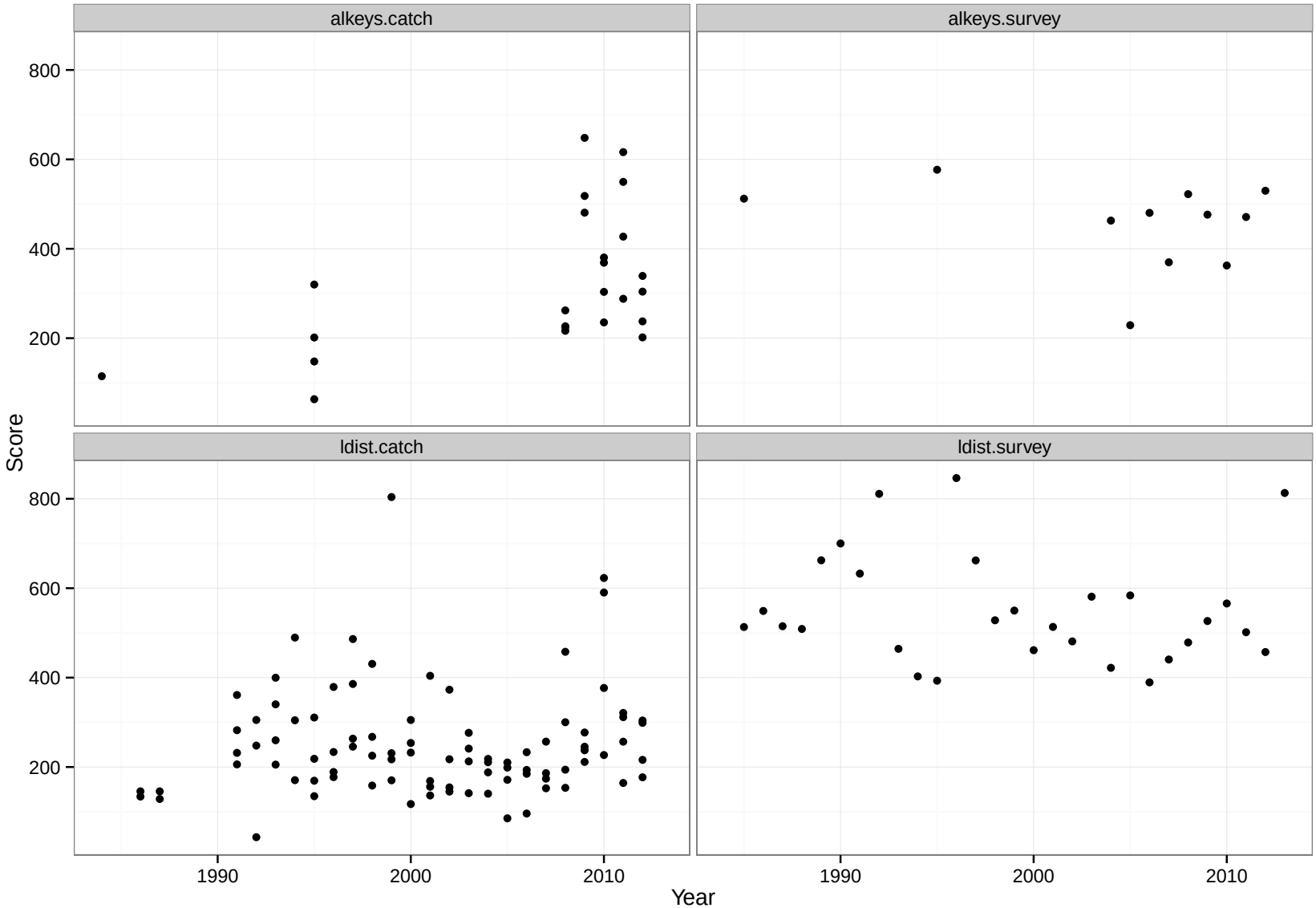
Tusk example (cont.)

```
setwd('~/Dropbox/gadget-models/08-tusk')
fit <- gadget.fit(mat.par = c(-8.6512,0.1403))
```

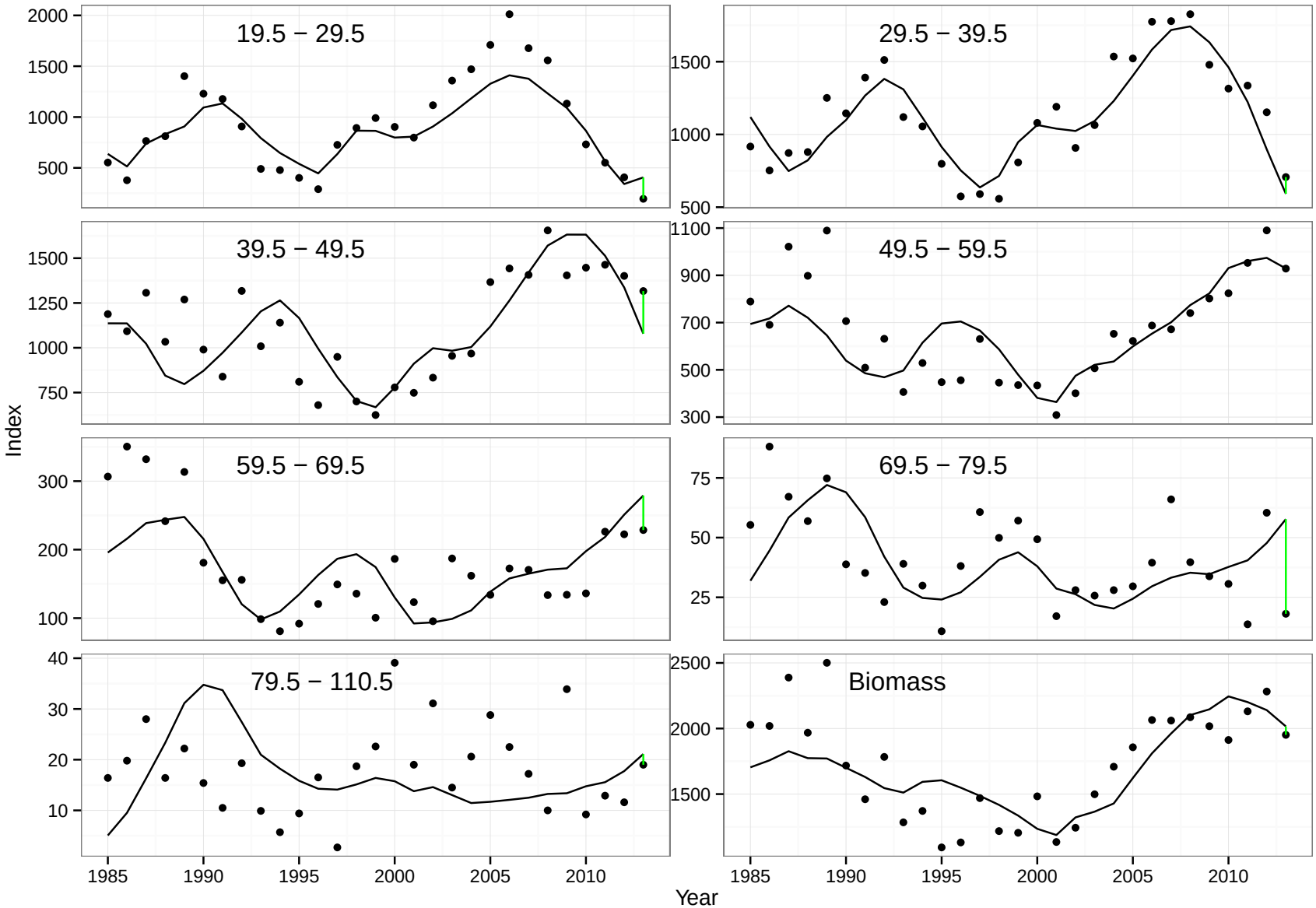
Tusk – fit statistics

	ALKc	ALKs	LDc	LDs	SI 20-39	SI 40-69	SI 70-110
Catch	7787	9158	26720	50630	15.91	36.19	83.25
Survey	8706	4944	154900	23030	14.46	36.22	99.91
Sind	11990	10780	113400	74260	1.797	4.335	17.75
Final	7676	4993	22480	15950	2.641	4.817	21.03

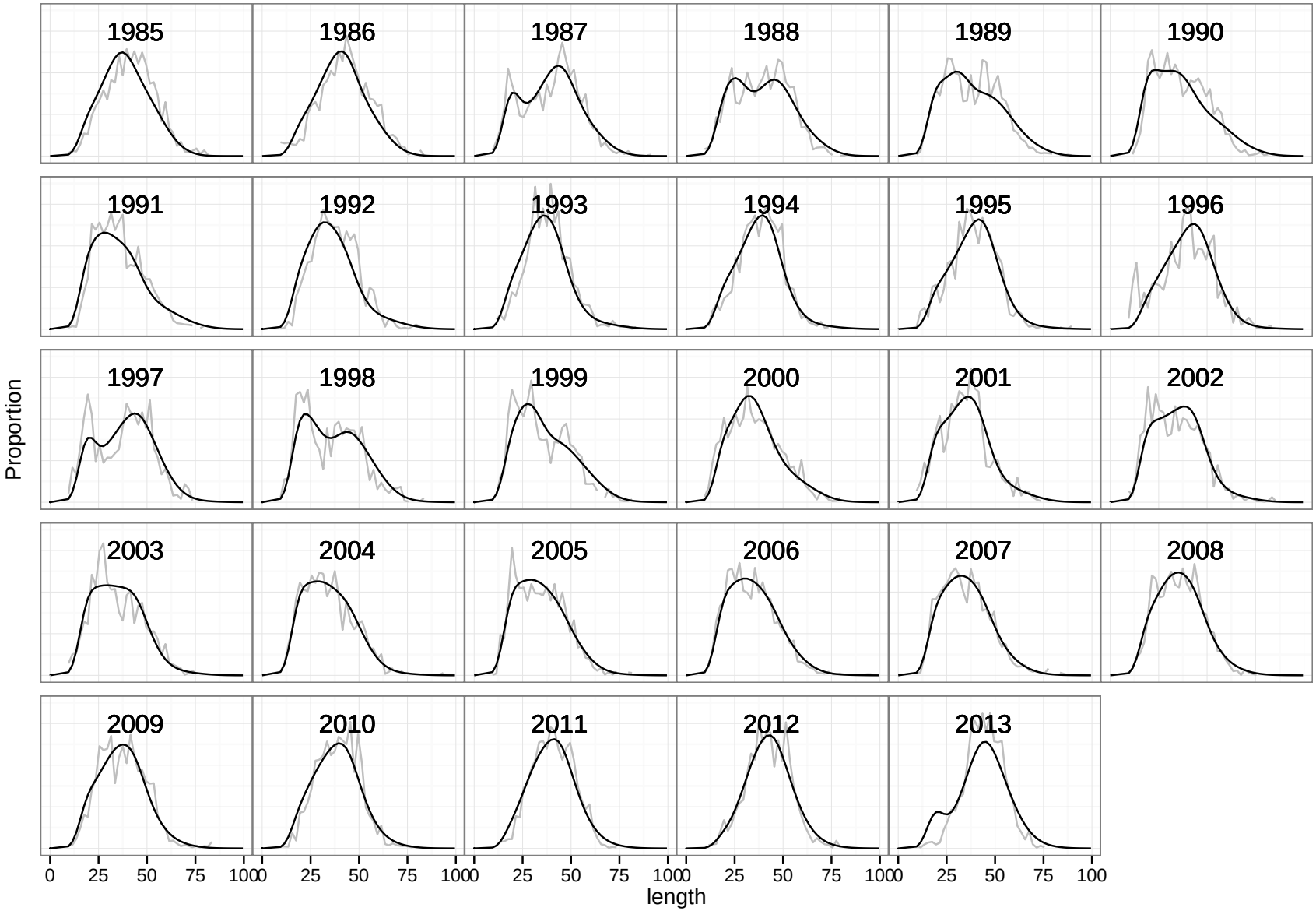
Tusk – likelihood summary



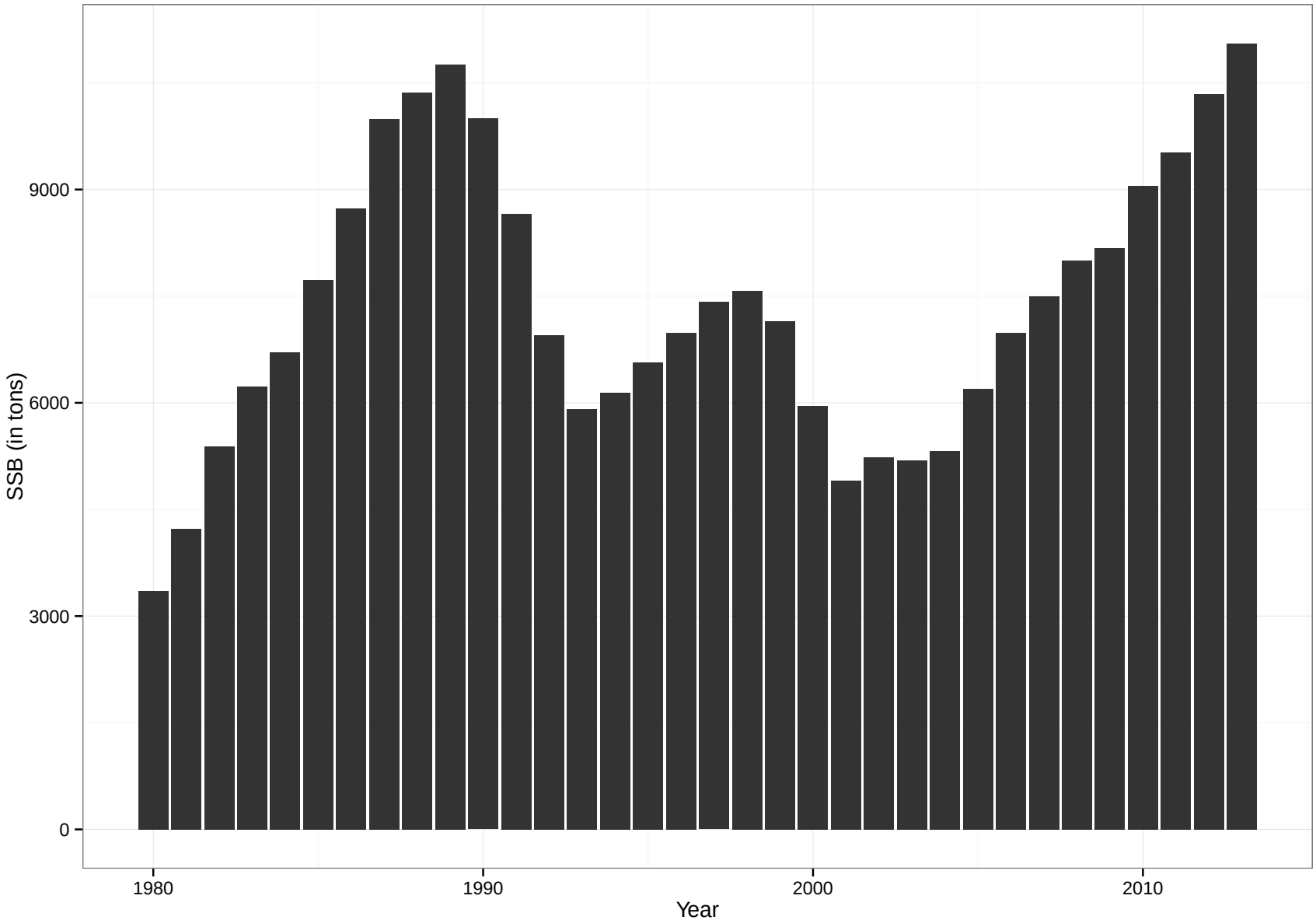
Tusk – indices



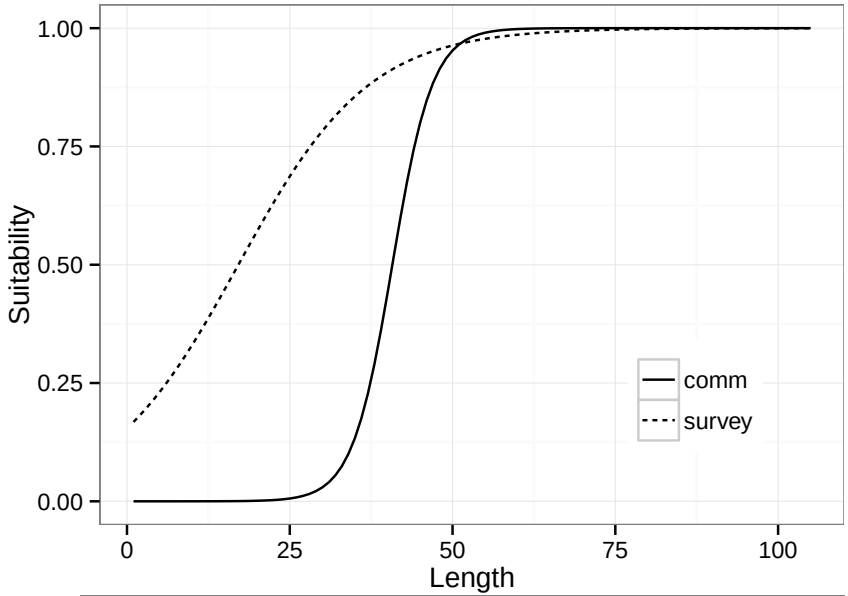
Tusk – Length distributions

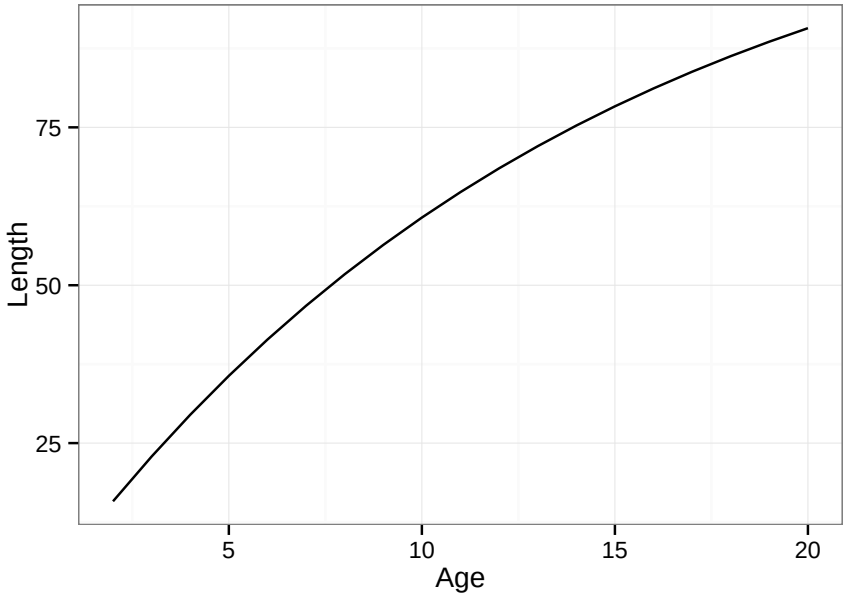


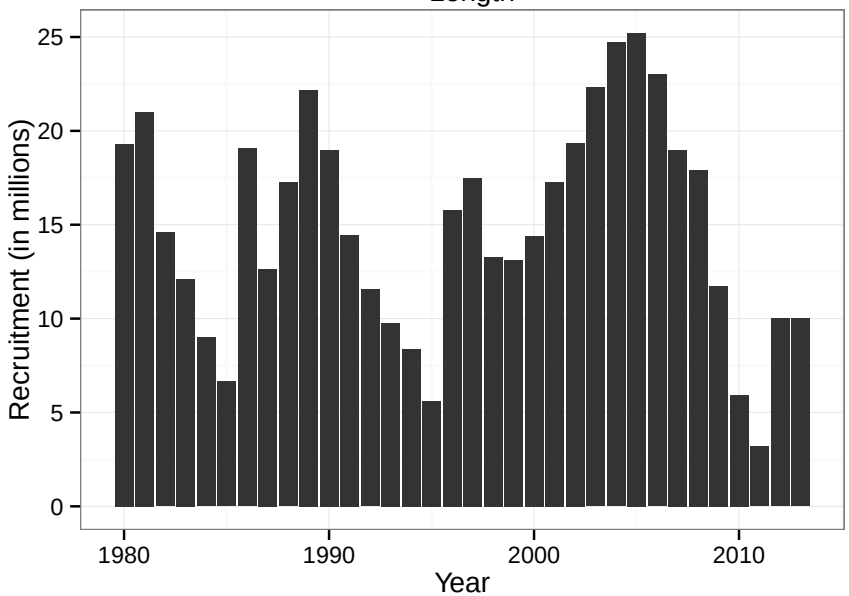
Tusk – Biomass estimates



Tusk – parameter estimates







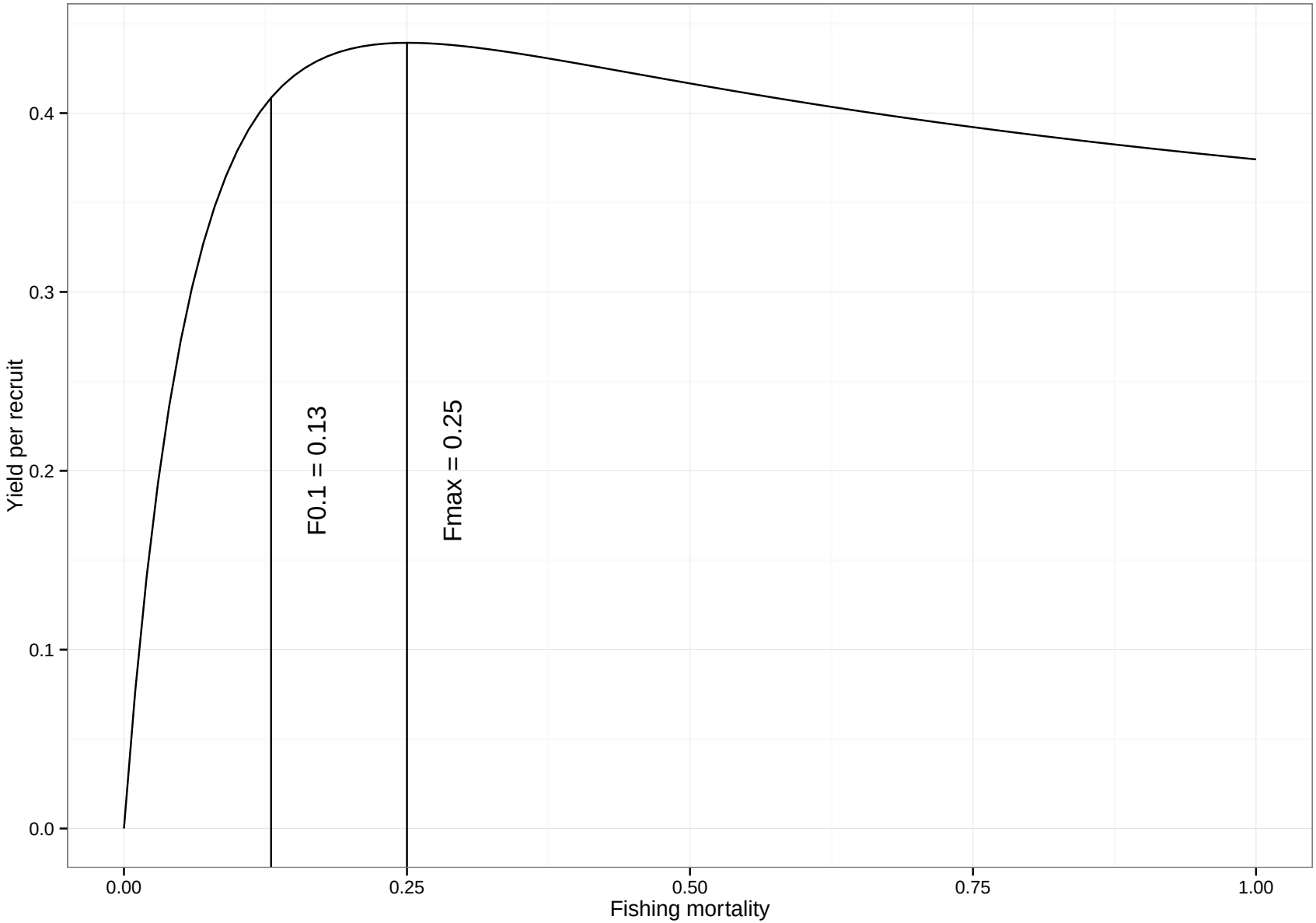
gadget.ypr

Calculates Yield-per-recruit within the model.

- Tracks a single year class for a set time period
- Fishing mortality is varied and the biomass caught is measured
- As this is a length-based model the yield is more conservative

```
setwd('~/Dropbox/gadget-models/08-tusk')
ypr <- gadget.ypr(params.file='WGTS/params.final',
                  ypr='WGTS/YPR')
```

Tusk example (cont.)



gadget.forward

- Projects the stock status forward based on assumptions for recruitment
  - Recruitment can be fixed to an average of the last years prior
  - or stochastic where the recruitment is an auto-regressive process
- Provides projections of the stock by age and length, and total catches
- Can project based on a range of F

Tusk - projections

```
setwd('~/Dropbox/gadget-models/08-tusk')

prognFmax <-
  gadget.forward(years=6,
                 params.file='WGTS/params.final',
                 pre='WGTS/PROGN',
                 stochastic=FALSE,
                 num.trials=1,
                 effort=ypr$fmax)
```

Tusk – projected yield