

# Plotting FLR objects with ggplotFL and ggplot2

Iago Mosqueira, EC JRC<sup>1</sup> - FLR Project

August 2013

## Using ggplot2 with FLR objects

The ggplot2 package provides a powerful alternative paradigm for creating simple and complex plots in R using the ideas of Wilkinson's *Grammar of Graphics*<sup>2</sup>

To facilitate the use of ggplot2 methods in FLR, the ggplotFL package has been created. The main resources on offer in this package are overloaded versions of the ggplot() method that take directly certain FLR classes, a new set of basic plots for some FLR classes, based on ggplot2 instead of lattice, and some examples and documentation on how best make use of ggplot2's powerful paradigm and implementation to obtain high quality plots for even fairly complex data structures.

## New plot() methods for FLR classes

The ggplotFL package also provides new versions of the plot method for a number of FLR classes. Each S4 class defined in any FLR package should have a plot() method defined that provides a visual summary of the contents of the object.

### FLStock

#### The overloaded 'ggplot' method

The standard ggplot functions expects a data.frame for its first argument, data. If ggplot is called on an FLR object, a conversion to data.frame takes place before the result, plus any other arguments provided, get passed to the original ggplot(). Conversion makes use of as.data.frame<sup>3</sup> methods defined in FLCore, with the cohort argument set to TRUE.

### FLQuant

Passing an FLQuant object to ggplot, we can specify the names of the dimensions as variables in the plot, where data refers to the column storing the actual numeric values. For example, to plot data (the catch slot from ple4 in this case) against year, we could use

<sup>1</sup> European Commission  
Joint Research Center  
IPSC - Maritime Affairs Unit  
Ispra, Italy  
<https://fishreg.jrc.ec.europa.eu/>

<sup>2</sup> Wilkinson, L. 1999. *The Grammar of Graphics*, Springer. ISBN 0-387-98774-6.

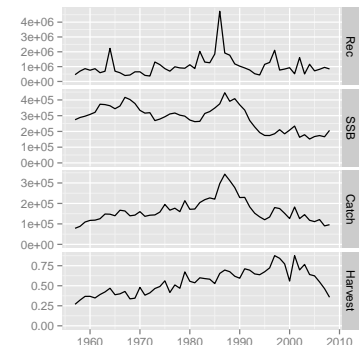


Figure 1: ggplot2 version of the standard plot() for FLStock, as applied to ple4

<sup>3</sup> method?as.data.frame('FLQuant')

```
ggplot(data = catch(ple4), aes(year, data)) + geom_point() +
  geom_line() + ylab("Catch (t)") + xlab("")
```

### FLQuants

Similarly, we can pass on to ggplot an object of class FLQuants, and the conversion to data.frame will make use of the corresponding method <sup>4</sup>. A new column gives the name of each FLQuant in the list, called qname. When can then use it to, for example, define a call to facet\_wrap() to obtain a separate subplot per element.

```
ggplot(data = FLQuants(Yield = catch(ple4), SSB = ssb(ple4),
  F = fbar(ple4)), aes(year, data)) + geom_line() +
  facet_wrap(~qname, scales = "free", nrow = 3) +
  labs(x = "", y = "")
```

This procedure is particularly useful when plotting information from objects with multiple FLQuant slots, as a subset can be selected for plotting, and transformations or computations can be carried out in the call to the FLQuants() creator.

### FLStock

#### Using ggplot2 by converting to data.frame

The methods shown above simply depend on conversion of FLR objects into data.frame, which can then be passed to ggplot(). Calling ggplot on an FLR object takes care of this conversion behind the scenes, but to obtain certain plots, it is best to directly convert the FLR objects into a data.frame.

### Some examples

#### Example: plot quantiles of a simulation

To have full control over a plot of the median (or mean) and the confidence or probability intervals of a simulated or randomized time series, i.e. an FLQuant object with iters, we need to arrange the different values computed from the object in separate columns of a data.frame.

If we start with some random FLQuant object, such as

```
f1a <- rlnorm(100, FLQuant(exp(cumsum(rnorm(25, 0,
  0.1))))), 0.1)
ggplot(f1a, aes(factor(year), data)) + geom_boxplot() +
  xlab("")
```

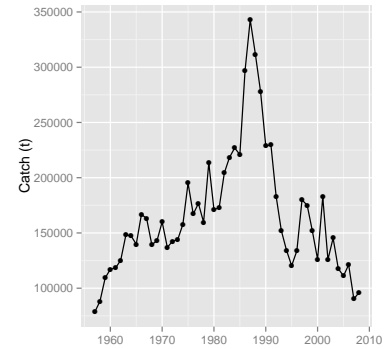


Figure 2: Combined line and point plot of a time series from an FLQuant object.  
<sup>4</sup> method?as.data.frame('FLQuants')

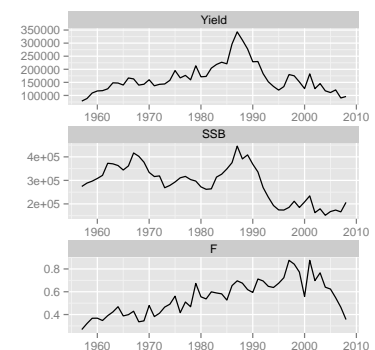


Figure 3: Facet wrap line plot of some time series from an FLQuants object.



Figure 4: Overall ggplot of an FLStock object, faceted by slot.

## PLOTTING FLR OBJECTS WI

we can first compute the necessary statistics on the object itself, as these operations are very efficient on an array. `quantile()` on an `FLQuant` will return the specified quantiles along the iter dimension. Let's extract the 10th, 25th, 50th, 75th and 90th quantiles.

```
flq <- quantile(fla, c(0.1, 0.25, 0.5, 0.75, 0.9))
```

The object can now be coerced to a `data.frame`

```
fdf <- as.data.frame(flq)
```

and inspected to see how the 100 iters have been now turned into the five requested quantiles in the iter column

```
##  quant year  unit season  area iter  data
## 1  all    1 unique  all unique 10% 2.307
## 2  all    2 unique  all unique 10% 2.462
## 3  all    3 unique  all unique 10% 2.159
```

The long format `data.frame` can be reshaped into a wide format one so that we can instruct `ggplot` to use the quantiles, now in separate columns, to provide limits for the shaded areas in `geom_ribbon`. To do this we can use `cast`, as follows

```
fdw <- cast(fdf, quant + year + unit + season + area ~
  iter, value = "data")
```

This creates a wide `data.frame` in which the iter column is spread into five columns named as the levels of its conversion into factor

```
levels(fdw[, "iter"])
```

```
## [1] "10%" "25%" "50%" "75%" "90%"
```

We can now use those five quantile columns when plotting shaded areas using `geom_ribbon`. Please note that the column names returned by `quantile()` need to be quoted using backticks.

```
p <- ggplot(data = fdw, aes(x = year, y = '50%')) +
  geom_line() + geom_ribbon(aes(x = year, ymin = '10%',
    ymax = '90%'), fill = "red", alpha = 0.15) + geom_ribbon(aes(x =
    year, ymin = '25%', ymax = '75%'), fill = "red", alpha = 0.25) +
  ylab("data")
print(p)
```

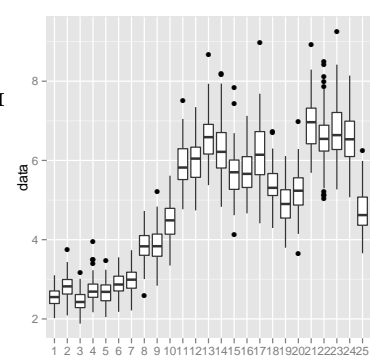


Figure 5:

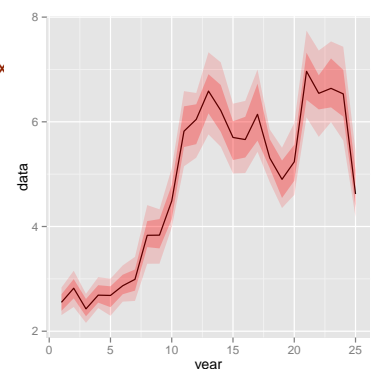


Figure 6:

*Example: Using FLQuants*

Coercion using `as.data.frame`, combined with the use of `cast` and `melt` (from the `reshape` package), provides the *FLR* user with most tools required to create a large range of `ggplots` out of any *FLR* object.

*Example: Simulation trajectories plot*

```
fds <- cast(as.data.frame(iter(fla, c(1, 4, 23, 56))),
           quant + year + unit + season + area ~ iter, value = "data")

p + geom_line(data = fds, aes(year, c('1')), colour = "red",
             size = 1)

fds <- as.data.frame(iter(fla, c(1, 4, 23, 56)))

p + geom_line(data = fds, aes(year, data, colour = iter),
             size = 1) + theme(legend.position = "none")
```

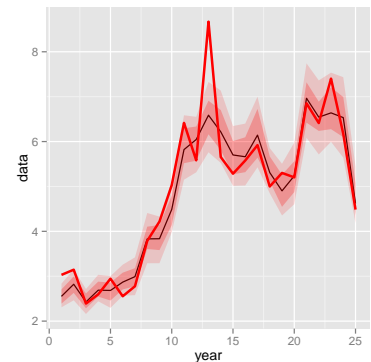


Figure 7:

*Example: Bubble plots*

```
ggplot(catch.n(ple4), aes(year, as.factor(age), size = data)) +
  geom_point(shape = 21) + scale_size(range = c(1,
  20)) + ylab("age") + theme(legend.position = "none")
```

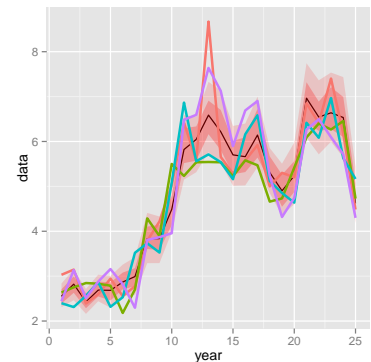


Figure 8:

*Example: Residual plots*

```
dat <- as.data.frame(catch.n(ple4))
dat$resid <- dat$data - mean(dat$data)

ggplot(dat, aes(year, as.factor(age), size = resid)) +
  geom_point(shape = 21, aes(colour = factor(sign(resid)),
    fill = factor(sign(resid)))) + scale_size(range = c(1,
  20)) + scale_colour_manual(values = c("black",
  "white")) + scale_fill_manual(values = c("lightgray",
  "black")) + theme(legend.position = "none") + ylab("age")
```

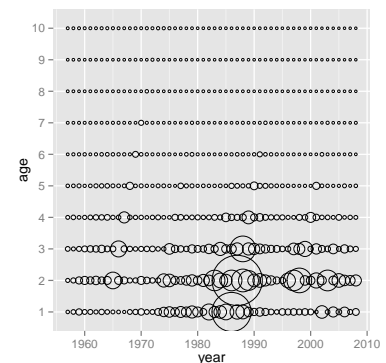
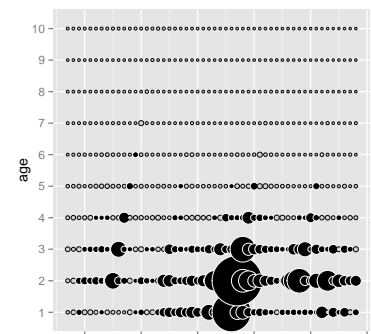


Figure 9:

*Example: Cohort*

```
ggplot(catch.n(ple4), aes(cohort, data)) + geom_line(aes(group = age,
  colour = factor(age))) + scale_colour_hue()
```



*More information*

- You can submit bug reports, questions or suggestions on ggplotFL at the ggplotFL issue page <sup>5</sup>, or on the FLR mailing list.
- Or send a pull request to <https://github.com/flr/ggplotFL/>
- For more information on the FLR Project for Quantitative Fisheries Science in R, visit the FLR webpage <sup>6</sup>.
- To learn more about ggplot2, visit the ggplot2 website <sup>7</sup>, or look at the ggplot2 book.<sup>8</sup>
- The latest version of ggplotFL can always be installed using the devtools package, by calling

<sup>5</sup> <https://github.com/flr/ggplotFL/issues>

<sup>6</sup> <http://flr-project.org>

<sup>7</sup> <http://ggplot2.org/>

<sup>8</sup> Wickham, H. 2009. *ggplot2: Elegant Graphics for Data Analysis*. Springer, Use R! Series. ISBN 978-0-387-98140-6

```
library(devtools)
install_github("ggplotFL", "flr")
```

*Package versions*

- R: R version 3.0.1 (2013-05-16)
- ggplot2: 0.9.3.1
- ggplotFL: 2.15.20130823
- FLCore: 2.5.20130820