Stickleback example

This is an example of how to use KRMr.

This is not an official release and is still in experimental phase.

Reading the shape information

Stickleback are swimbladdered fish. We have to shape files, one for the fish body and one for the swimbladder. The shape file for the fish body contains one column named `x_fb` the fishbody (fb) along the x axis (Length), `w_fb` the width (seen from top) of the fish body at each position x (along the fish body length), `z_fbL` the lower height of the fish body along the x axis (lower (L) fish body (fb) extend in direction z) and `z_fbU` the upper height along the x axis (upper (U) fish body (fb) extend in direction z).

Analoguously, the swimbadder (sb) shape file contains `x_sb` - x axis (Length direction), `w_sb` - Width of the swimbladder along the x axis, `z_sbL` - lower swimbladder extend and `z_sbU` - upper swimbladder extend.

In our special case the csv files contain two different fish body and swimbladder shapes, identified by an additional `ind` column.

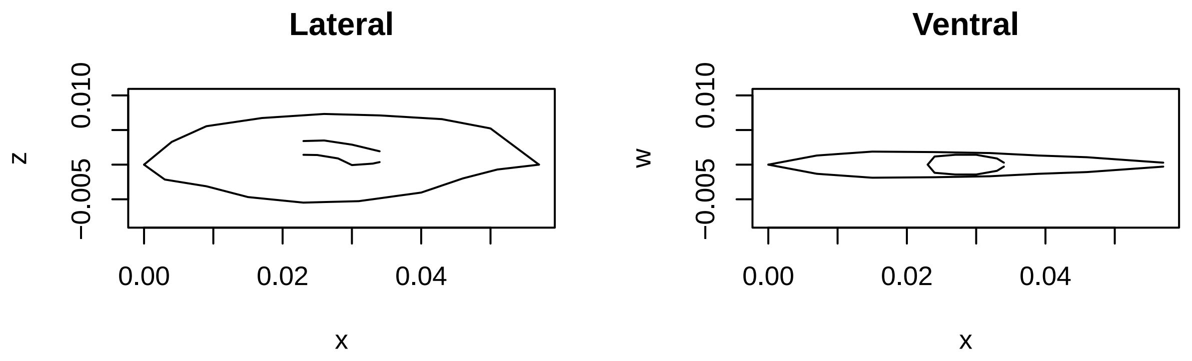
```
library(dplyr)
```

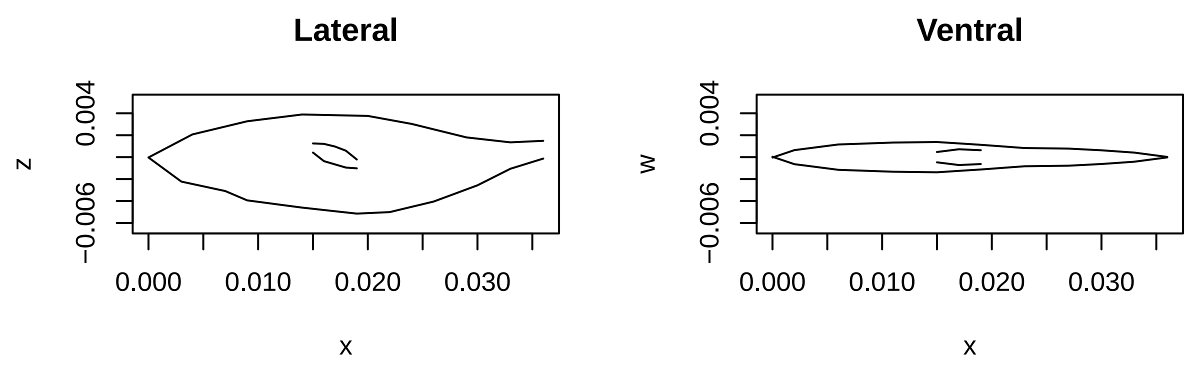
```
##
## Attaching package: 'dplyr'

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

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

```
library(KRMr)
stb_fb <- read.csv(paste0(dirname(getwd()),'/data/Sticklebacks_FB.csv'))
stb_sb <- read.csv(paste0(dirname(getwd()),'/data/Sticklebacks_SB.csv'))
```

Let's have a look at the shapes we loaded.
To plot the loaded shapes, we can use the `shplot(x_fb, w_fb, x_sb, w_sb, z_fbU, z_fbL, z_sbU, z_sbL)` function:

```
for (i in unique(stb_fb$ind)){
  fb =stb_fb%>%filter(ind==i)
  sb =stb_sb%>%filter(ind==i)
  layout(matrix(c(1,2,3,3), 2, 2, byrow = TRUE))
  KRMr::shplot(x_fb = fb$x_fb, w_fb = fb$w_fb,
              x_sb = sb$x_sb, w_sb = sb$w_sb,
              z_fbU = fb$z_fbU, z_fbL = fb$z_fbL,
              z_sbU = sb$z_sbU, z_sbL = sb$z_sbL)
}
```





For now, lets just include one shape example.

```
id = unique(stb_fb$ind)[1]
fb =stb_fb%>%filter(ind==i)
sb =stb_sb%>%filter(ind==i)
```

Running a KRM simulation

Now that we have our shapes sorted, let's have a go at a KRM simulation.
We want to get TS at 38, 70, 120 and 200 kHz, with the below defined settings:

- c.w = ambient water sound speed
- rho.w = density ambient water

- theta = orientation in degrees
- c.fb = sound speed inside fish body
- c.sb = sound speed inside swimbladder
- rho.sb = density inside swimbladder
- rho.fb = density inside fish body = L = length of the fishbody in m
- x_fb = Fish body coordinates along x axis
- x_sb = Swimbladder coordinates along x axis
- w_fb = Fish body width along x axis
- w_sb = Swimbladder width coordinates along x axis
- z_fbU = Upper height of the fish body along the x axis
- z_fbL = Lower height of the fish body along the x axis

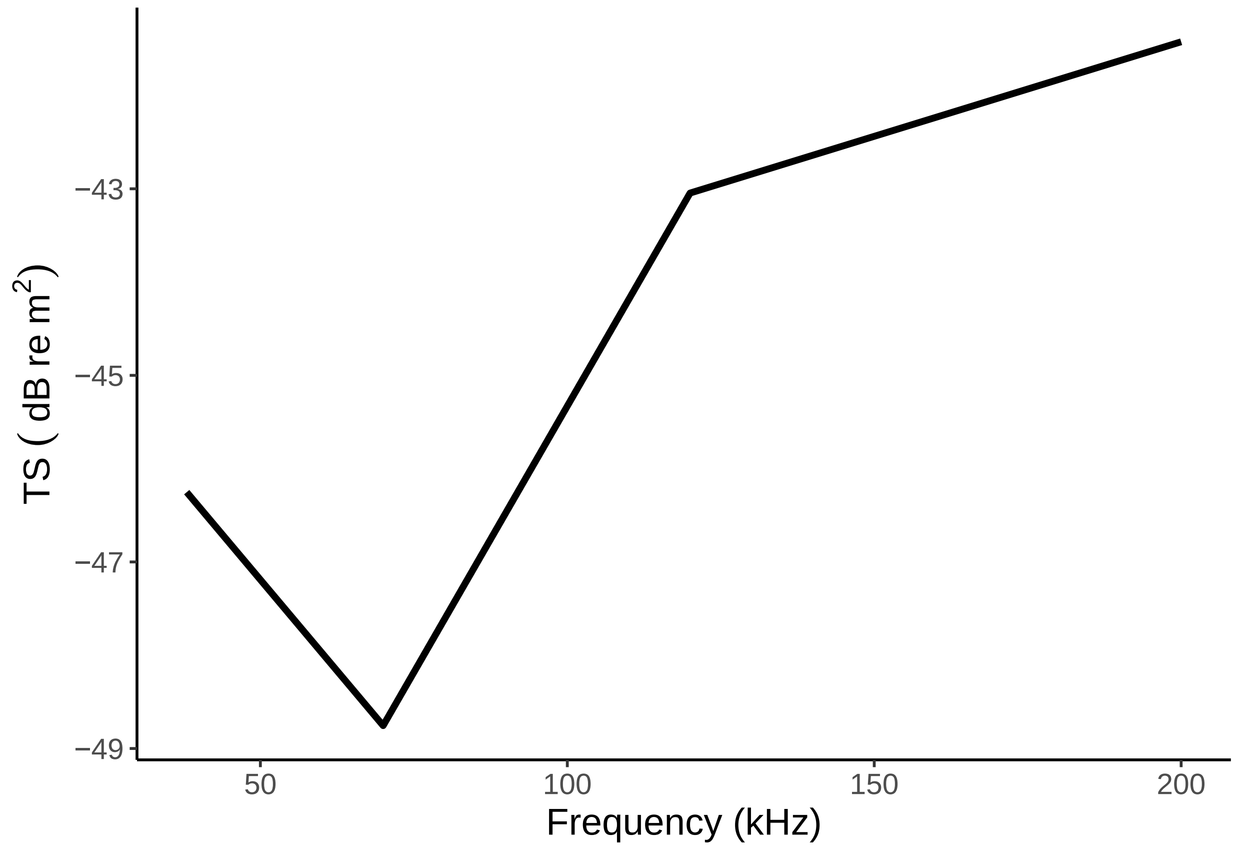
- z__fbU = Upper height of the swimbladder along the x axis
- z__fbL = Lower height of the swimbladder along the x axis

A description of the parameters is also available in the help files `?KRMr::krm` (runs a single krm simulation) or `?KRMr::krm.sim` (runs multiple simulations if any input parameter contains more than one value (except for coordinates)

```
TS = KRMr::krm.sim(frequency =c(38,70,120,200) * 1000,
                   c.w = 1490,
                   rho.w = 1030,
                   theta=90,
                   c.fb = 1570,
                   c.sb = 345,
                   rho.sb = 1.24,
                   rho.fb = 1070,
                   L=0.25,
                   x_fb = fb$x_fb,
                   x_sb = sb$x_sb,
                   w_fb = fb$w_fb,
                   w_sb = sb$w_sb,
                   z_fbU = fb$z_fbU,
                   z_fbL = fb$z_fbL,
                   z_sbU = sb$z_sbU,
                   z_sbL = sb$z_sbL)
```

Let's have a look at the outcome:

```
library(ggplot2)
ggplot(data=TS,
       aes(x=frequency/1000, y=TS))+
  geom_line(size=1.2)+
  ylab(expression(TS~(~dB~re~m^2)))+
  xlab('Frequency (kHz)')+
  theme_classic()+
  theme(text=element_text(size=14),
        legend.position='top')
```

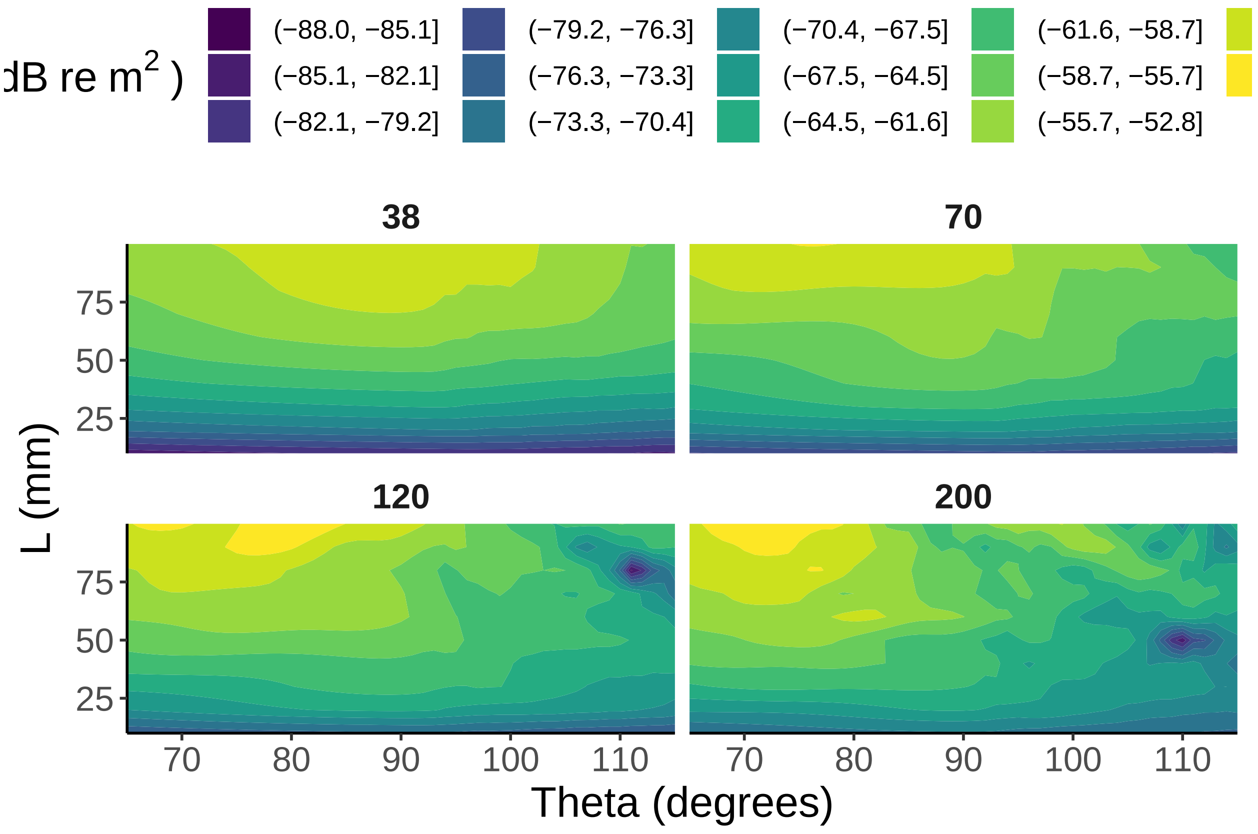


We can also compute slightly more sophisticated simulations of a range of frequencies, lengths and orientations (remermbe the KRM model is only valid for orientations, close to 90 degrees ~65-115 degrees):

```
TS = krm.sim(frequency =c(38,70,120,200) * 1000,
                   c.w = 1490,
                   rho.w = 1030,
                   theta=seq(65,115),
                   c.fb = 1570,
                   c.sb = 345,
                   rho.sb = 1.24,
                   rho.fb = 1070,
                   L=seq(0.01,0.1,by=0.01),
                   x_fb = fb$x_fb,
                   x_sb = sb$x_sb,
                   w_fb = fb$w_fb,
                   w_sb =  sb$w_sb,
                   z_fbU = fb$z_fbU,
                   z_fbL = fb$z_fbL,
                   z_sbU = sb$z_sbU,
                   z_sbL = sb$z_sbL)
```

Let’s plot the results:

```
ggplot2::ggplot(data=TS, ggplot2::aes(x= theta, y=L*1000 , z=TS))+
  facet_wrap(.~frequency/1000)+
  geom_contour_filled(bins=15)+
  ggplot2::scale_fill_viridis_d(expression(TS~'('~dB~re~ m^2~')'))+
  ggplot2::xlab('Theta (degrees)')+
  ggplot2::ylab(expression(L~'(mm)'))+
  ggplot2::scale_x_continuous(expand=c(0,0))+
  ggplot2::scale_y_continuous(expand=c(0,0))+
  ggplot2::theme_classic()+
  ggplot2::theme(legend.position='top',
                 legend.text=element_text(size=10),
                 text=element_text(size=16),
                 strip.background = element_blank(),
                 strip.text = element_text(face='bold'))
```

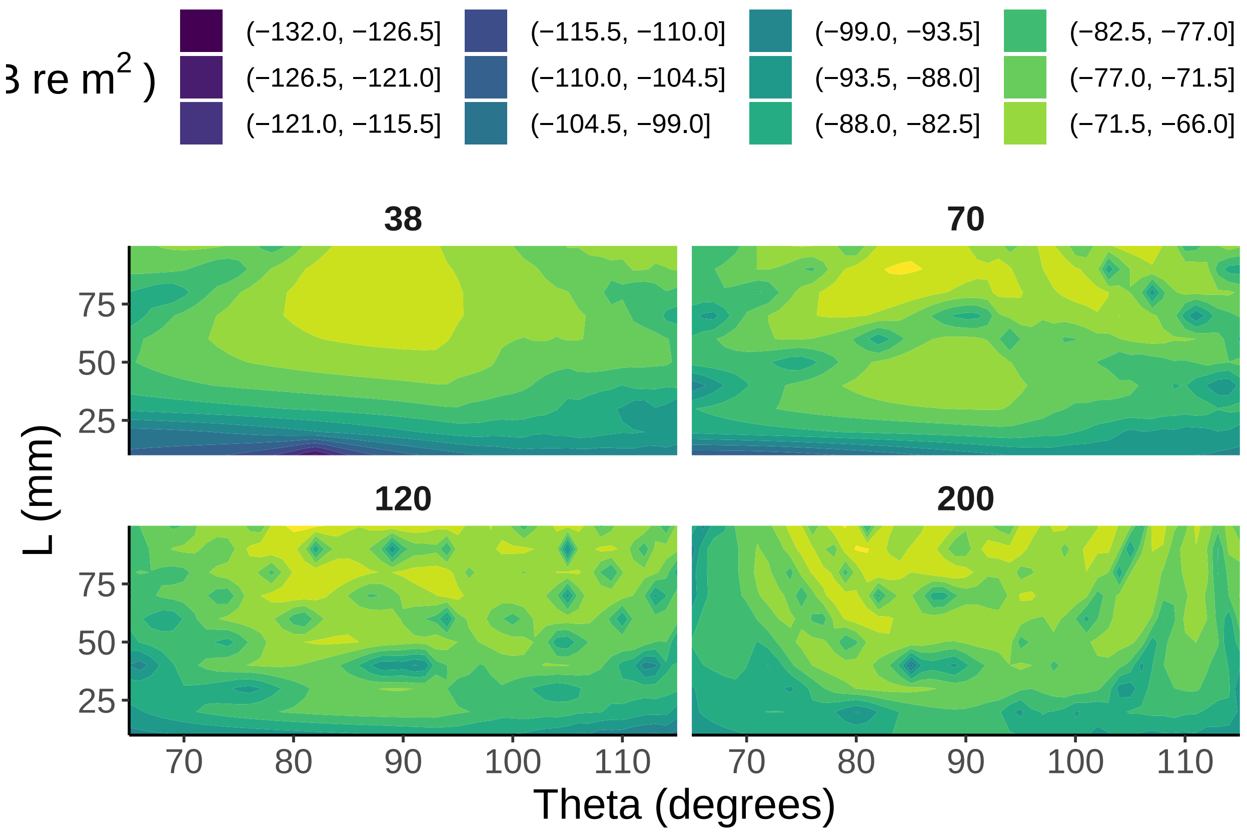


Let's do the same without a swimbladder for a hypothetical fish without swimbladder (we simply assign the swimbladder coordinates NULL) for fish sizes of 1 cm to 10 cm without swimbladder:

```
TS = krm.sim(frequency =c(38,70,120,200) * 1000,
                   c.w = 1490,
                   rho.w = 1030,
                   theta=seq(65,115),
                   c.fb = 1570,
                   c.sb = 345,
                   rho.sb = 1.24,
                   rho.fb = 1070,
                   L=seq(0.01,0.1,by=0.01),
                   x_fb = fb$x_fb,
                   x_sb = NULL,
                    w_fb = fb$w_fb,
                    w_sb =  NULL,
                    z_fbU = fb$z_fbU,
                    z_fbL = fb$z_fbL,
                    z_sbU = NULL,
                    z_sbL = NULL)
```

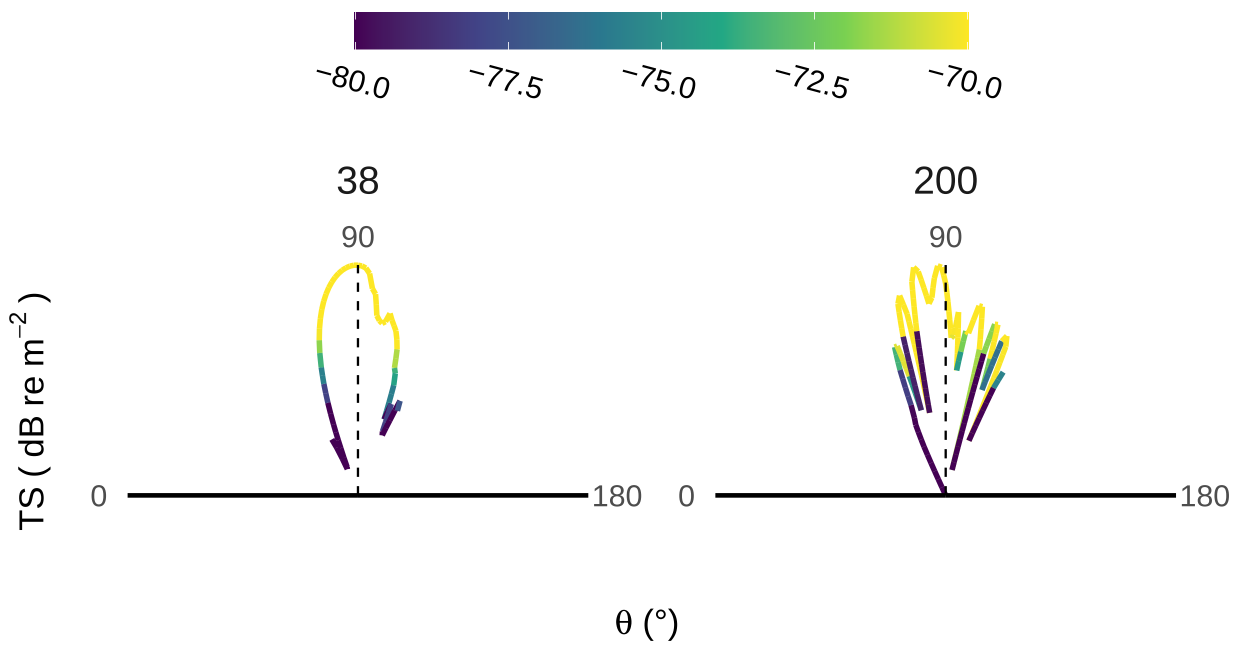
Let's plot the results:

```
ggplot2::ggplot(data=TS, ggplot2::aes(x= theta, y=L*1000 , z=TS))+
  facet_wrap(.~frequency/1000)+
  geom_contour_filled(bins=15)+
  ggplot2::scale_fill_viridis_d(expression(TS~'('~dB~re~ m^2~')'))+
  ggplot2::xlab('Theta (degrees)')+
  ggplot2::ylab(expression(L~'(mm)'))+
  ggplot2::scale_x_continuous(expand=c(0,0))+
  ggplot2::scale_y_continuous(expand=c(0,0))+
  ggplot2::theme_classic()+
  ggplot2::theme(legend.position='top',
               legend.text=element_text(size=10),
               text=element_text(size=16),
               strip.background = element_blank(),
               strip.text = element_text(face='bold'))
```



We can also look at orientation plots for a fish of 5 cm without a swimbladder at 38 kHz:

```
TSrot = krm.sim(frequency =c(38,200) * 1000,
                   c.w = 1490,
                   rho.w = 1030,
                   theta=65:115,
                   c.fb = 1570,
                   c.sb = 345,
                   rho.sb = 1.24,
                   rho.fb = 1070,
                   L=0.08,
                   x_fb = fb$x_fb,
                   x_sb = NA,#sb$x_sb,
                   w_fb = fb$w_fb,
                   w_sb = NA,#sb$w_sb,
                   z_fbU = fb$z_fbU,
                   z_fbL = fb$z_fbL,
                   z_sbU = NA,#sb$z_sbU,
                   z_sbL = NA)#sb$z_sbL)
ggplot(TSrot, aes(x = theta, y = TS, group=frequency/1000, col=TS)) +
  geom_path(size=1.2) +
  facet_wrap(.~frequency/1000)+
  scale_x_continuous(limits=c(0,360), breaks=c(0,90,180))+
  coord_polar(start=-pi/2,direction=1)+
  scale_colour_viridis_c(name='', limits=c(-80,-70), oob=scales::squish)+

  ylab(expression(TS~'('~dB~re~m^-2~')' ))+
  xlab(expression(paste(theta,' (\u00B0) ')))+

  geom_vline(xintercept=90, lty=2)+
  geom_vline(xintercept=0,size=1)+
  geom_vline(xintercept=180,size=1)+

  theme_classic()+
  theme(strip.background = element_blank(),
        strip.text=element_text(size=18),
        legend.position='top',
        legend.text=element_text(angle=-15),
        legend.key.width = unit(2,'cm'),
        axis.line.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.line.x = element_blank(),
        panel.spacing.y = unit(-5,'lines'),
        axis.title.x = element_text(vjust=27, size=16),
        axis.title.y = element_text(hjust=0.75, size=16),
        text=element_text(size=18))
```



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.