Rmarkdown and Shiny

Lei mingri

2021-12-13

Contents

Chapter 1

Markdown rmarkdown shiny leaflet plotly Github COVID-19 R R

1.1

bookdown **.Rmd**

`index.Rmd`

1.2

HTML :

RStudio IDE **Build** **Build Book**

R console

```
bookdown::render_book()
```

PDF `bookdown::pdf_book` XeLaTeX TinyTeXhttps://yihui.org/tinytex/

1.3

HTML .Rmd

Chapter 2

2.1

2.1.1 Github

- **GitHub https://github.com**
- **R RStudio**
- Git **Git(windows)https://gitforwindows.org/**
- Git bash Git

```
git config --global user.name 'leimingri'
git config --global user.email 'lmr18845128812@163.com'
git config --global --list
```

GitHub

- GitHub /

2.1.2 Git GitHub,RStudio

RStudio GitHub Git Rstudio Git *Tools* > *global options* > *Git/SVM* GitHub repository RStudio GitHub

- RStudio Git GitHub

GitHub , RStudio GitHub Rstudio RStudio *File > New Project > Version Control > Git*

- R “ ” —— R *Tools>Version Control>Commit* GitHub
- GitHub

  Commit GitHub “Push”

R Markdown http://rmarkdown.rstudio.com
happy-git-with-R

2.1.3

1. : LaTeX failed to compile R.tex.

   https://yihui.org/tinytex/r/#debugging for debugging tips.—

2. GitHub URL ‘…Connection was reset, errno 10054’

   `git config --global http.sslVerify` “false” ssl git

3. Knit PDF Knit HTML) Knit PDF : pandoc document conversion failed with error 43

   github
```
install.packages("devtools")
```
   devtools
```
devtools::install_github("rstudio/rmarkdown")
```

4. PDF

latex_engine: xelatex

5. GitHub URL ‘…Connection was reset, errno 10054’

   git bash `git config --global http.sslVerify "false"` ssl git

```
git config --global --add core.compression -1
```

.
524288000 500 1048576000 1G
```
git config --global http.postBuffer 524288000
```

ssh
```
git clone git://github.com/XX/XXXX.git
```

```
git config http.sslVerify "false"
```

2.2 Rmarkdown Rmarkdown

1	https://github.com/geekcompany/DeerResume
2	https://github.com/geekcompany/ResumeSample
MarkDown	http://link.ftqq.com/0rsRL
	http://link.ftqq.com/KWkVX
	https://github.com/resumejob/awesome-resume
	https://osjobs.net/topk/
	https://wowchemy.com/hugo-themes/

2.3 ggplot2

2.3.1

- Layer
- Scale X
- Coordinate
- Facet

2.3.2

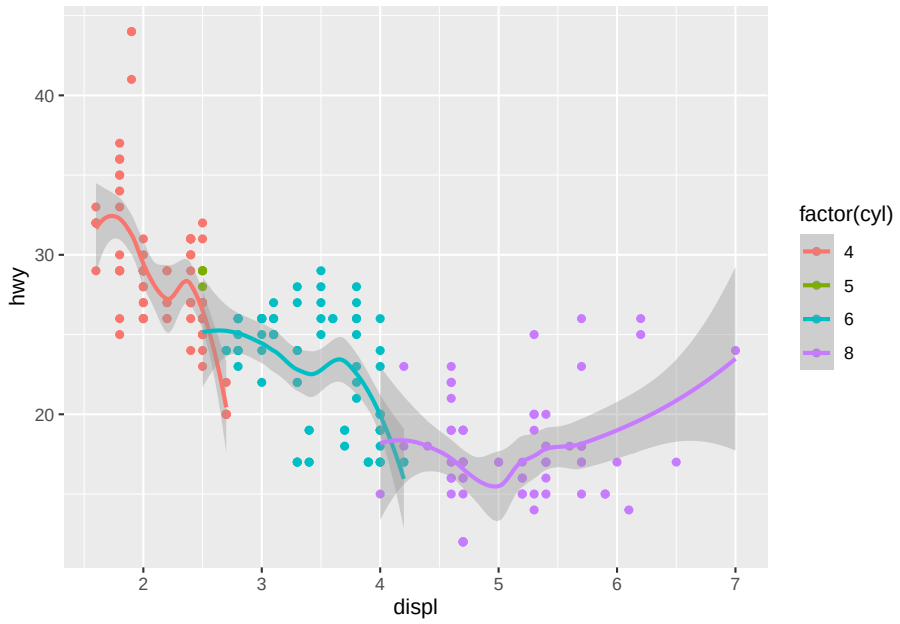
ggplot2 mpg (displ) (hwy) (cyl) ggplot2 ggplot aes displ X hwy

```
install.packages("ggplot2")
```

```
## Installing package into '/home/leimingri/R/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
library(ggplot2)
p <- ggplot(data=mpg,aes(x=displ,y=hwy,colour=factor(cyl)))
p + geom_point() + geom_smooth()
```

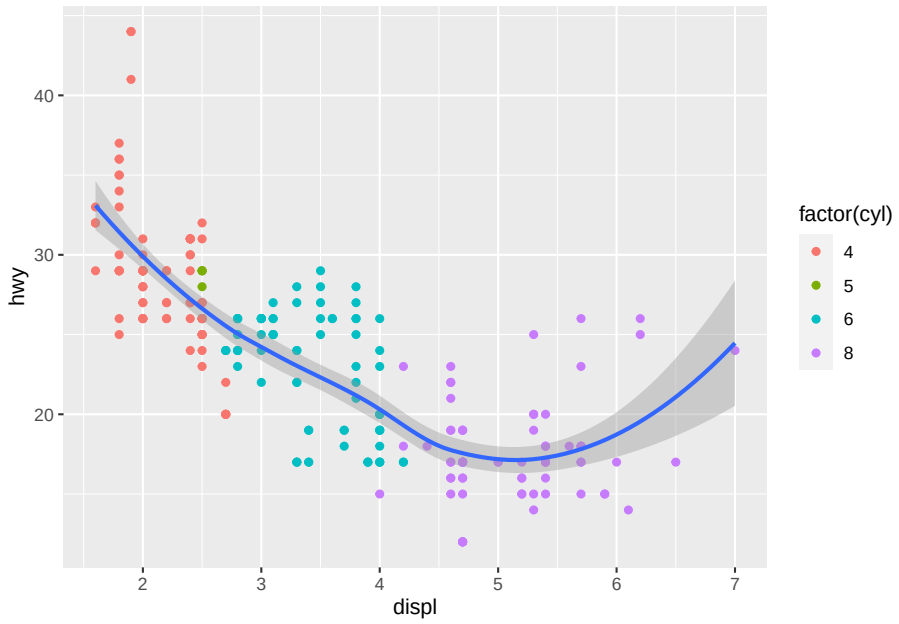
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



colour colour

```
p <- ggplot(mpg,aes(x=displ,y=hwy))
p + geom_point(aes(colour=factor(cyl))) + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



2.3.3

2.3.3.1

data aes aes Aesthetic
mpg hwy ggplot hwy X + p summary print(p)

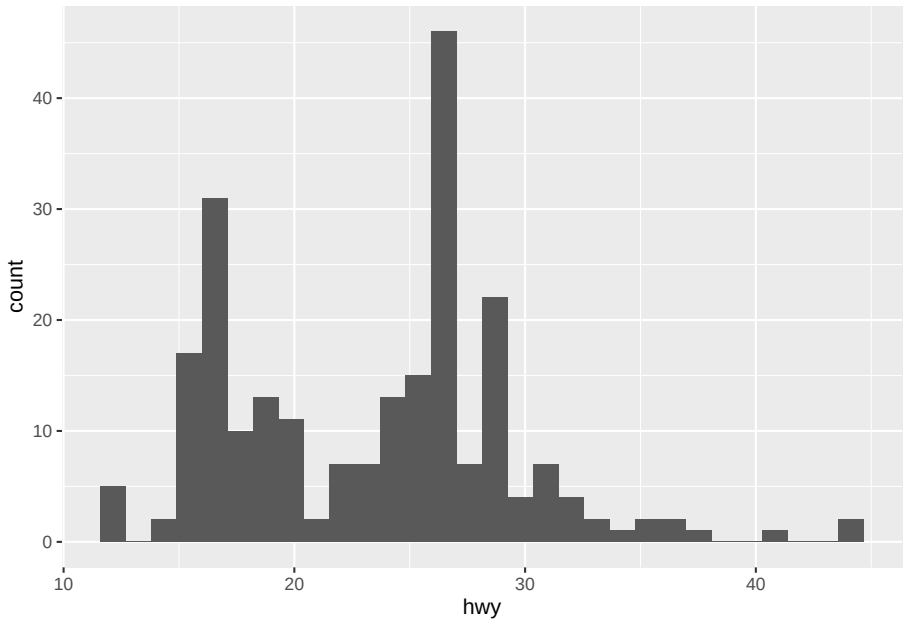
```
library(ggplot2)
p <- ggplot(data = mpg,aes(x = hwy))
p <- p + geom_histogram()
summary(p)
```

```
## data: manufacturer, model, displ, year, cyl, trans, drv, cty, hwy, fl,
##   class [234x11]
## mapping:  x = ~hwy
## faceting: <ggproto object: Class FacetNull, Facet, gg>
##     compute_layout: function
##     draw_back: function
##     draw_front: function
##     draw_labels: function
##     draw_panels: function
##     finish_data: function
##     init_scales: function
```

```
##     map_data: function
##     params: list
##     setup_data: function
##     setup_params: function
##     shrink: TRUE
##     train_scales: function
##     vars: function
##     super:  <ggproto object: Class FacetNull, Facet, gg>
## ----------------------------------
## geom_bar: na.rm = FALSE, orientation = NA
## stat_bin: binwidth = NULL, bins = NULL, na.rm = FALSE, orientation = NA, pad = FALSE
## position_stack
```

```
print(p)
```

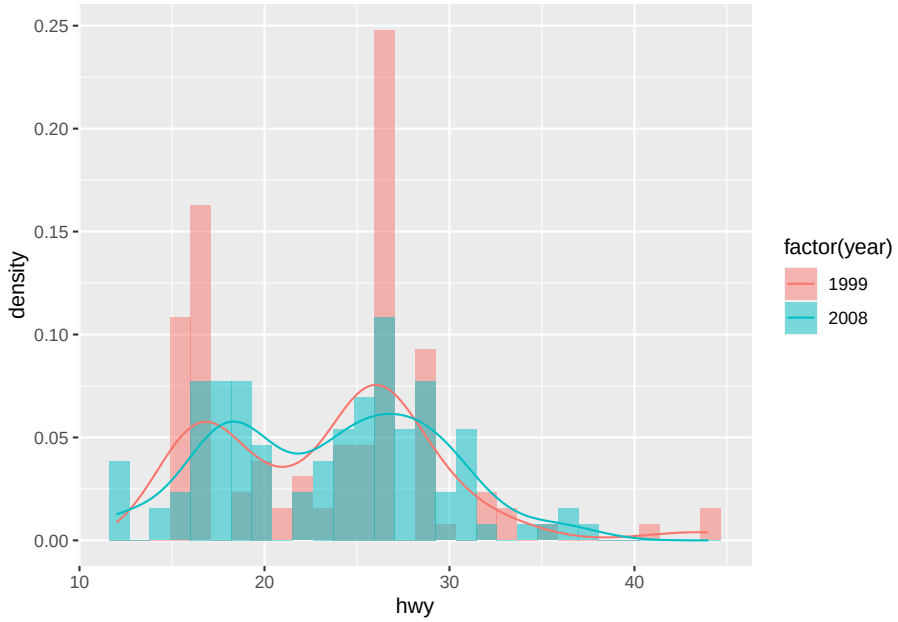
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



p geom_histogram geom ggplot geom_histogram geom aes geom stat geom_histogram stat_bin year Y

```
p <- ggplot(mpg,aes(hwy))
p + geom_histogram(position = 'identity',
alpha=0.5,
aes(y = ..density..,
fill = factor(year))) +
stat_density(geom = 'line',
position = 'identity',
aes(colour = factor(year)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2.3.3.2

Position adjustments stack dodge fill identity jitter

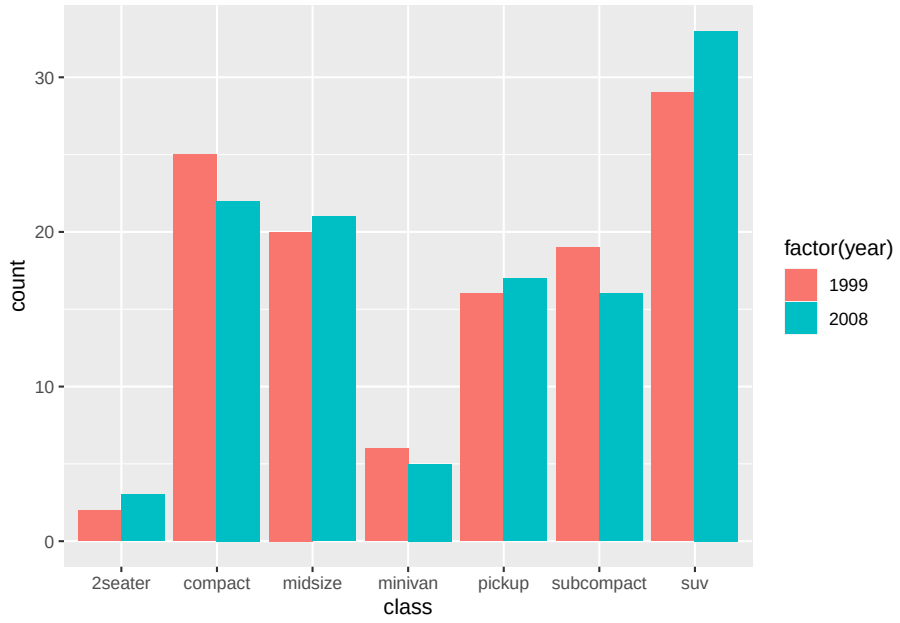
mpg class year

```
library(ggplot2)
with(mpg,table(class,year))
```

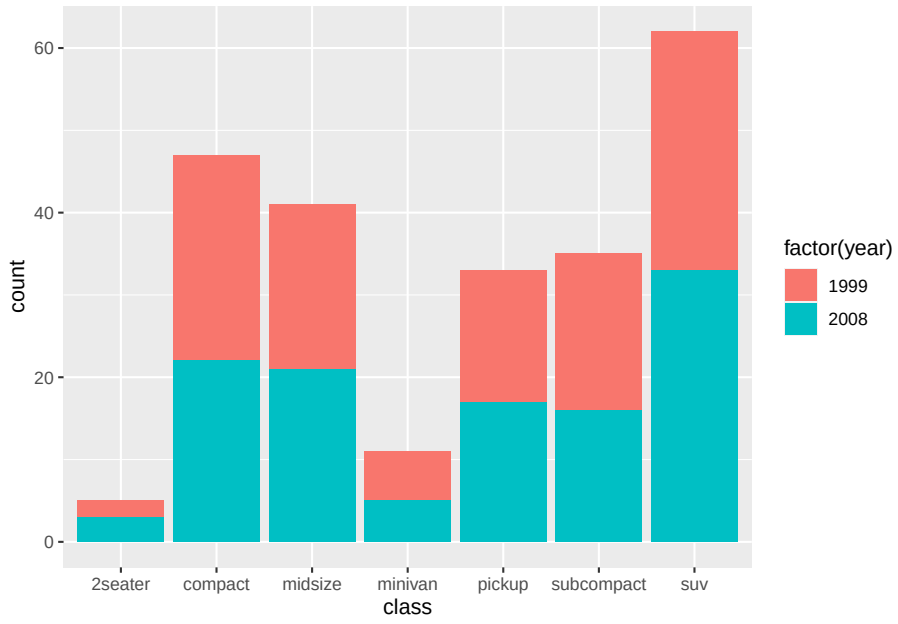
```
##             year
## class        1999 2008
```

```
##   2seater       2    3
##   compact      25   22
##   midsize      20   21
##   minivan       6    5
##   pickup       16   17
##   subcompact   19   16
##   suv          29   33
```

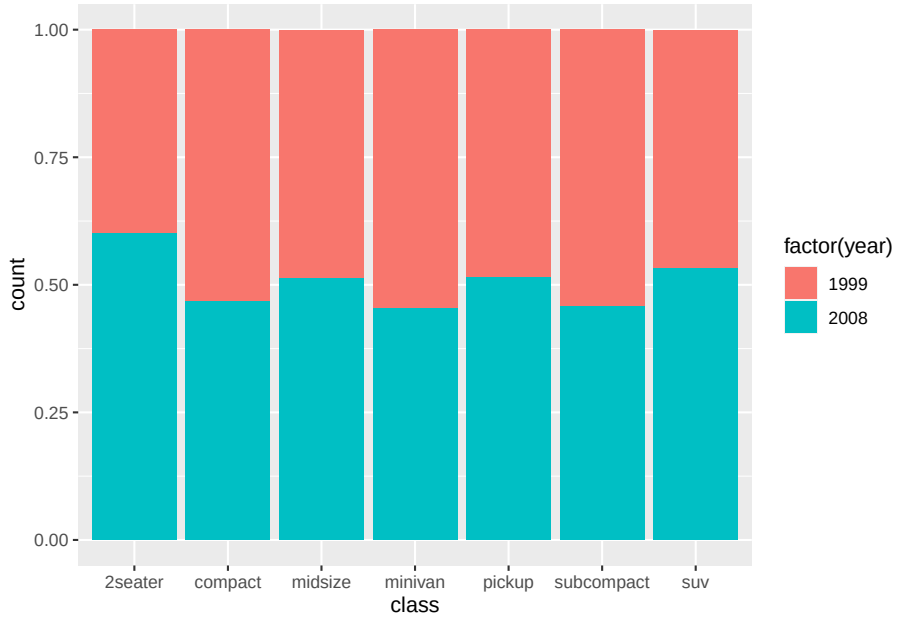
```
p <- ggplot(data=mpg,aes(x=class,fill=factor(year)))
p + geom_bar(position='dodge')
```



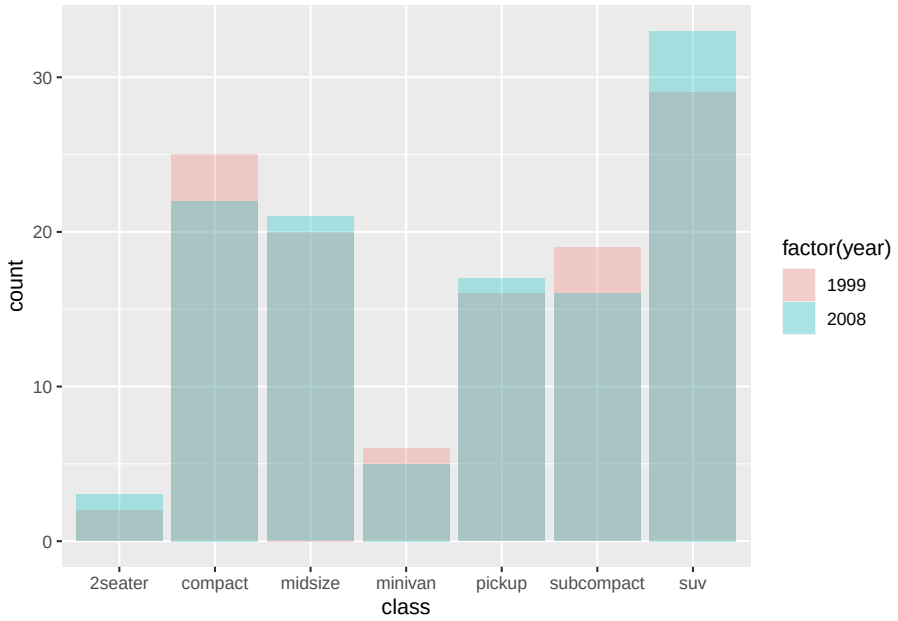

```
p + geom_bar(position='stack')
```




```
p + geom_bar(position='fill')
```



```
p + geom_bar(position='identity',alpha=0.3)
```



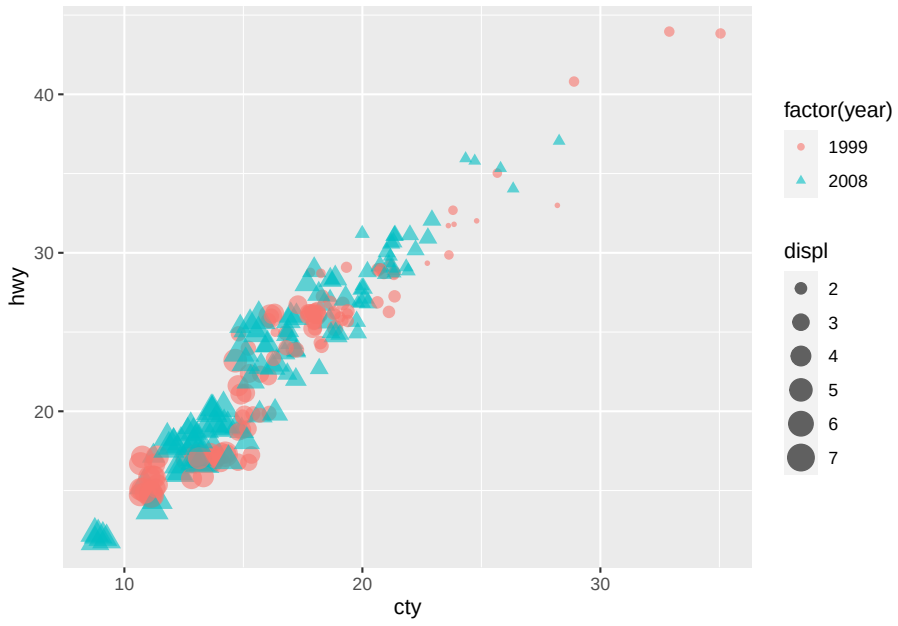
dodge stack geom_bar fill stack Y identity

2.3.3.3

-

mpg cty ,hwy ,displ ,year

```
library(ggplot2)
p <- ggplot(mpg, aes(cty, hwy))
p1 <- p + geom_point(aes(colour = factor(year),shape = factor(year), size = displ), al
print(p1)
```

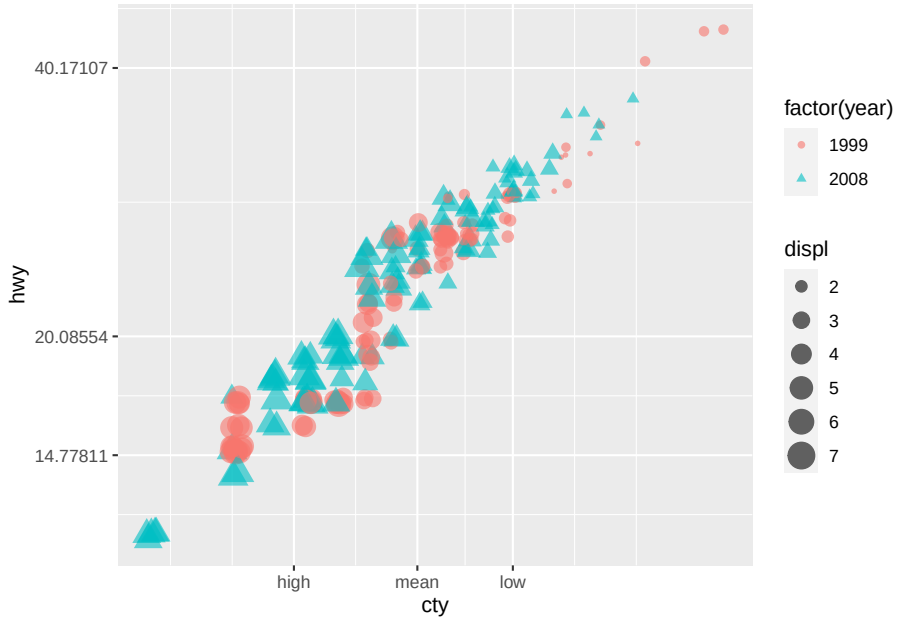


1999 2008 jitter 2008

•

ggplot2 X Y X X

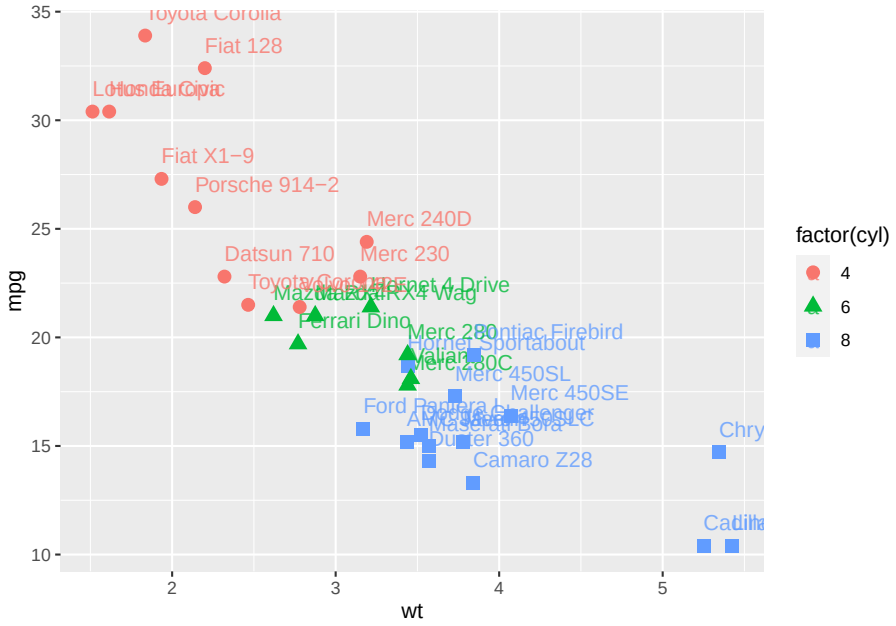
```
cty.mean=with(mpg,mean(cty))
cty.sd=with(mpg,sd(cty))
p1 + scale_x_continuous(trans='log',breaks=c(cty.mean-cty.sd,cty.mean,cty.mean+cty.sd), labels=c(
```



-

geom_text

```
p <- ggplot(mtcars, aes(x=wt, y=mpg,colour=factor(cyl),label=rownames(mtcars)))
p + geom_text(hjust=0,vjust=-1,alpha=0.8)+ geom_point(size=3,aes(shape=factor(cyl)))
```



2.3.4 ggplot2

ggplot2 data.frame ggplot()

2021 https://www.sodic.com.cn/datasets

Chapter 3

Shiny APP

Mastering Shiny

3.1

Shiny APP app.R app.R

`library(shiny)`(Shiny)

```
ui <- fluidPage()
```

```
server <- function(input, output, session) {}
```

`shinyApp(ui, server)` ui server shiny

3.1.1

```
library(shiny)
ui <- fluidPage(
  selectInput("dataset", label = "Dataset", choices = ls("package:datasets")),
  verbatimTextOutput("summary"),
  tableOutput("table")
)
```

fluidPage()

selectInput()

verbatimTextOutput() tableOutput() Shiny

verbatimTextOutput() tableOutput()

3.1.2

Shiny

```
server <- function(input, output, session) {
  output$summary <- renderPrint({
    dataset <- get(input$dataset, "package:datasets")
    summary(dataset)
  })

  output$table <- renderTable({
    dataset <- get(input$dataset, "package:datasets")
    dataset
  })
}
```

render

renderPrint() verbatimTextOutput() ( ) renderTable() tableOutput()

3.1.3 Shiny

```
shinyApp(ui=ui, server=server)
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is
```

3.2 UI

3.2.1 inputs

sliderInput

textInput

passwordInput

textAreaInput

numericInput

dateInput

dateRangeInput

selectInput

radioButtons

checkboxGroupInput

checkboxInput

fileInput

actionButton

[https://shiny.rstudio.com/gallery/widget-gallery.html]https://shiny.rstudio.com/gallery/widget-gallery.html

3.2.2 outputs

3.2.2.1 Text

textOutput()

verbatimTextOutput()

renderText() textOutput() renderPrint() verbatimTextOutput()

```
ui <- fluidPage(
  textOutput("text"),
  verbatimTextOutput("code")
)
server <- function(input, output, session) {
  output$text <- renderText({
    "Hello friend!"
  })
  output$code <- renderPrint({
    summary(1:10)
  })
}
```

3.2.2.2 Tables

tableOutput()

dataTableOutput()

tableOutput() renderTable() dataTableOutput() renderDataTable()

```
ui <- fluidPage(
  tableOutput("static"),
  dataTableOutput("dynamic")
)
server <- function(input, output, session) {
  output$static <- renderTable(head(mtcars))
  output$dynamic <- renderDataTable(mtcars, options = list(pageLength = 5))
}
```

3.2.2.3 Plots

plotOutput()

plotOutput() renderPlot()

```
ui <- fluidPage(
  plotOutput("plot", width = "400px")
)
server <- function(input, output, session) {
  output$plot <- renderPlot(plot(1:5), res = 96)
}
```

3.2.2.4 Downloads

downloadButton()

downloadLink()

3.3

3.3.1 inputs

1.

2. render...() reactive()

3.3.2 output

: ID

3.3.3 Reactive programming

input output

Shiny ui server

reactive()

```
server <- function(input, output, session) {
  x1 <- reactive(rnorm(input$n1, input$mean1, input$sd1))
  x2 <- reactive(rnorm(input$n2, input$mean2, input$sd2))

  output$hist <- renderPlot({
    freqpoly(x1(), x2(), binwidth = input$binwidth, xlim = input$range)
  }, res = 96)

  output$ttest <- renderText({
    t_test(x1(), x2())
  })
}
```

:reactiveTimer()

```
server <- function(input, output, session) {
  timer <- reactiveTimer(500)

  x1 <- reactive({
    timer()
    rpois(input$n, input$lambda1)
  })
  x2 <- reactive({
    timer()
    rpois(input$n, input$lambda2)
  })

  output$hist <- renderPlot({
    freqpoly(x1(), x2(), binwidth = 1, xlim = c(0, 40))
  }, res = 96)
}
```

actionButton():

```
ui <- fluidPage(
  fluidRow(
```

```
    column(3,
      numericInput("lambda1", label = "lambda1", value = 3),
      numericInput("lambda2", label = "lambda2", value = 5),
      numericInput("n", label = "n", value = 1e4, min = 0),
      actionButton("simulate", "Simulate!")
    ),
    column(9, plotOutput("hist"))
  )
)
server <- function(input, output, session) {
  x1 <- eventReactive(input$simulate, {
    rpois(input$n, input$lambda1)
  })
  x2 <- eventReactive(input$simulate, {
    rpois(input$n, input$lambda2)
  })

  output$hist <- renderPlot({
    freqpoly(x1(), x2(), binwidth = 1, xlim = c(0, 40))
  }, res = 96)
}
```

eventReactive() : x1() x2()

Chapter 4

Shiny feedback

4.1 Validation

4.1.1 Validating input

:req()

```
library(shiny)

ui <- fluidPage(
  shinyFeedback::useShinyFeedback(),
  numericInput("n","n",value=10),
  textOutput("half")

)

server <- function(input, output, session) {
  half<-reactive({
    even<-input$n %% 2==0
    shinyFeedback::feedbackWarning("n",!even,"please select an even number!")
    #req(even)
    input$n /2

  })
    output$half<-renderText(half())

}
```

4.1.2 Cancelling execution with req()

```
library(shiny)

ui <- fluidPage(
  selectInput("language","Language",choices = c("","English","Maori")),
  textInput("name","Name"),
  textOutput("greeting")

)

server <- function(input, output, session) {
  greetings<-c(
    Engilsh="Hello",
    Maori="Kia ora"
  )
  output$greeting<-renderText({
    #req(input$language,input$name)
    paste0(greetings[[input$language]],"",input$name,"!")
  })
}
```

4.1.3 req() and validation

req() shinyFeedback

cancelOutput = TRUE : cancelOutput = TRUE good value

```
library(shiny)

ui <- fluidPage(
  shinyFeedback::useShinyFeedback(),
   textInput("dataset", "Dataset name"),
  tableOutput("data")
)

server <- function(input, output, session) {
  data<-reactive({
    req(input$dataset)

    exists<-exists(input$dataset,"package:datasets")
    shinyFeedback::feedbackDanger("dataset",!exists,"Unknown dataset")
    req(exists,cancelOutput = TRUE)
```

```
    get(input$dataset, "package:datasets")

  })

  output$data<-renderTable({
    head(data())
  })
}
```

4.1.4 Validate output

shiny: validate() validate(message)

```
library(shiny)

ui <- fluidPage(
  numericInput("x","x",value=0),
  selectInput("trans","transformation",choices=c("square", "log", "square-root")),
  textOutput("out")
)

server <- function(input, output, session) {

 output$out <- renderText({
    if (input$x < 0 && input$trans %in% c("log", "square-root")) {
      validate("x can not be negative for this transformation")
    }

   switch(input$trans,
      square = input$x ^ 2,
      "square-root" = sqrt(input$x),
      log = log(input$x)
    )
  })
}
```

4.2 Notifications

4.2.1 Transient notification

4.2.2 Removing on completion

(duration = NULL) closeButton = FALSE
on.exit() ( )
on.exit:ensures that the notification is removed

Chapter 5

Shiny uploads and downloads

5.1 upload

```
library(shiny)

ui <- fluidPage(
  fileInput("upload",NULL,buttonLabel="Upload...", multiple = TRUE),
  tableOutput("files")
)

server <- function(input, output, session) {
  output$files<-renderTable(input$upload)
}
```

input*upload* *req*(*input*upload)

accept accept = ".csv" accept

R tools::file_ext()

```
library(shiny)

ui <- fluidPage(
  fileInput("upload",NULL,accept = c(".csv",".tsv")),
  numericInput("n","Rows",value=5, min = 1, step = 1),
  tableOutput("head")
)
```

```
server <- function(input, output, session) {
  data<-reactive({
    req(input$pload)

    ext<-tools::file_ext()
    switch(ext,
           csv=vroom::vroom(input$upload$datapath,delim=","),
             tsv=vroom::vroom(input$upload$datapath,delim="\t"),
             validate("Invalid file; Please upload a .csv or .tsv file"))
  })
  output$head<-renderTable({
    head(data(),input$n)
  })
}
```

5.2 Download

: downloadButton(id) downloadLink(id)

,downloadButton() , downloadHandler()

downloadHandler() , :filename , ( ), content file ,file

```
library(shiny)

ui <- fluidPage(
  selectInput("dataset", "Pick a dataset", ls("package:datasets")),
  tableOutput("preview"),
  downloadButton("download","Download.tsv")
)

server <- function(input, output, session) {
   data <- reactive({
    out <- get(input$dataset, "package:datasets")
    if (!is.data.frame(out)) {
      validate(paste0("'", input$dataset, "' is not a data frame"))
    }
    out
  })

  output$preview <- renderTable({
    head(data())
  })
```

```
  output$download <- downloadHandler(
    filename = function() {
      paste0(input$dataset, ".tsv")
    },
    content = function(file) {
      vroom::vroom_write(data(), file)
    }
  )
}
```

5.3 Downloading reports

RMarkdown RMarkdown YAML

```
library(shiny)

ui <- fluidPage(
  sliderInput("n", "Number of points", 1, 100, 50),
  downloadButton("report", "Generate report")
)

server <- function(input, output, session) {
  output$report <- downloadHandler(
    filename = "report.html",
    content = function(file) {
      params <- list(n = input$n)

      id <- showNotification(
        "Rendering report...",
        duration = NULL,
        closeButton = FALSE
      )
      on.exit(removeNotification(id), add = TRUE)

      rmarkdown::render("report.Rmd",
        output_file = file,
        params = params,
        envir = new.env(parent = globalenv())
      )
    }
  )
}
```

Chapter 6

Dynamic UI

:

update

tabsetPanel()

uiOutput() renderUI()

6.1 Updating inputs

```
library(shiny)

ui <- fluidPage(
  numericInput("n","Simulations",10),
  actionButton("simulate","Simulate")
)

server <- function(input, output, session) {
  observeEvent(input$n, {
    label<-paste0("Simulate",input$n,"times")
    updateActionButton(inputId = "simulate",label=label)
  })
}
```

```
library(shiny)

ui <- fluidPage(
  selectInput("dataset","Choose a dataset",c("pressure","cars")),
```

```
  selectInput("column","Choose column",character(0)),
  verbatimTextOutput("summary")
)

server <- function(input, output, session) {
  # freezeReactiveValue(input,"column")
  dataset<-reactive(get(input$dataset,"package:datasets"))

  observeEvent(input$dataset,{
    updateSelectInput(inputId = "column",choices = names(dataset()))
  })
  output$summary<-renderPrint({
    summary(dataset()[[input$column]])
  })
}
```

6.2 Dynamic visibility

```
library(shiny)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      selectInput("controller","Show",choices = paste0("panel",1:3))
    ),
    mainPanel(
      tabsetPanel(
        id="switcher",
        type="hidden",
        tabPanelBody("panel1","Panel 1 content"),
        tabPanelBody("panel2","Panel 2 content"),
        tabPanelBody("panel3","Panel 3 content")
      )
    )
  )
)

server <- function(input, output, session) {
  observeEvent(input$controller,{
    updateTabsetPanel(inputId = "switcher",selected=input$controller)
  })
}
```

```
library(shiny)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      selectInput("dist","Distribution",
                  choices=c("normal","uniform","exponential")),
      numericInput("n","Number of samples",value=100),
      parameter_tabs<-tabsetPanel(
        id="params",
        type="hidden",
        tabPanel("normal",
                 numericInput("mean","mean",value=1),
                 numericInput("sd","standard deviation",min=0,value=1)),
        tabPanel("uniform",
                 numericInput("min","min",value=0),
                 numericInput("max","max",value=1)),
        tabPanel("exponential",
                 numericInput("rate","rate",value=1,min=0))
      )
    ),
    mainPanel(
      plotOutput("hist")
    )
  )
)

server <- function(input, output, session) {
  observeEvent(input$dist,{
    updateTabsetPanel(inputId="params",selected = input$dist)
  })
  sample<-reactive({
    switch(input$dist,
           normal=rnorm(input$n,input$mean,input$sd),
           uniform=runif(input$n,input$min,input$max),
           exponential=rexp(input$n,input $ rate))
  })
  output$hist<-renderPlot(hist(sample()),res=96)
}
```

```
library(shiny)

ui <- fluidPage(
  tabsetPanel(
    id="wizard",
```

```
    type="hidden",
    tabPanel("page_1",
             "Welcome!",
             actionButton("page_12","next")),
    tabPanel("page_2","Only one page to go",
             actionButton("page_21","prev"),
             actionButton("page_23","next")),
    tabPanel("page_3","You're done!",
             actionButton("page_32","prev"))
  )
)

server <- function(input, output, session) {
  switch_page<-function(i){
    updateTabsetPanel(inputId="wizard",selected=paste0("page_",i))
  }
  observeEvent(input$page_12,switch_page(2))
  observeEvent(input$page_21,switch_page(1))
  observeEvent(input$page_23,switch_page(3))
  observeEvent(input$page_32,switch_page(2))

}
```

6.3 Creating UI with code

```
library(shiny)

ui <- fluidPage(
  textInput("label","label"),
  selectInput("type","type",c("slider","numeric")),
  uiOutput("numeric")
)
server <- function(input, output, session) {
  output$numeric<-renderUI({
    #value<-isolate(input$dynamic)
    if(input$type=="slider"){
      sliderInput("dynamic",input$label,value=0,min=0,max=10)
    }else{
      numericInput("dynamic",input$label,value=0,min=0,max=10)
    }
  })
}
```

```
library(purrr)
library(shiny)

ui <- fluidPage(
  numericInput("n", "Number of colours", value = 5, min = 1),
  uiOutput("col"),
  textOutput("palette")
)


server <- function(input, output, session) {
  col_names <- reactive(paste0("col", seq_len(input$n)))

  output$col <- renderUI({
    map(col_names(), ~ textInput(.x, NULL))
  })

  output$palette <- renderText({
    map_chr(col_names(), ~ input[[.x]] %||% "")
  })
}
```

```
library(shiny)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      numericInput("n", "Number of colours", value = 5, min = 1),
      uiOutput("col"),
    ),
    mainPanel(
      plotOutput("plot")
    )
  )
)

server <- function(input, output, session) {
  col_names <- reactive(paste0("col", seq_len(input$n)))

  output$col <- renderUI({
    map(col_names(), ~ textInput(.x, NULL, value = isolate(input[[.x]])))
  })

  output$plot <- renderPlot({
    cols <- map_chr(col_names(), ~ input[[.x]] %||% "")
```

```
    # convert empty inputs to transparent
    cols[cols == ""] <- NA

    barplot(
      rep(1, length(cols)),
      col = cols,
      space = 0,
      axes = FALSE
    )
  }, res = 96)
}
```

Chapter 7

Bookmarking

7.1 Basic

```
library(shiny)

ui <- fluidPage(
   sidebarLayout(
     sidebarPanel(
       sliderInput("omega", "omega", value = 1, min = -2, max = 2, step = 0.01),
      sliderInput("delta", "delta", value = 1, min = 0, max = 2, step = 0.01),
      sliderInput("damping", "damping", value = 1, min = 0.9, max = 1, step = 0.001),
      numericInput("length", "length", value = 100)
     ),
     mainPanel(
       plotOutput("fig")
     )
  )
)

server <- function(input, output, session) {
   t <- reactive(seq(0, input$length, length.out = input$length * 100))
  x <- reactive(sin(input$omega * t() + input$delta) * input$damping ^ t())
  y <- reactive(sin(t()) * input$damping ^ t())

  output$fig<-renderPlot({
    plot(x(), y(), axes = FALSE, xlab = "", ylab = "", type = "l", lwd = 2)
  }, res = 96
  )
}
```

: 1. bookmarkButton() 2. ui function 3. enableBookmarking = “url” shinyApp()

```
library(shiny)

ui <- function(request){
  fluidPage(
    sidebarLayout(
      sidebarPanel(
        sliderInput("omega", "omega", value = 1, min = -2, max = 2, step = 0.01),
        sliderInput("delta", "delta", value = 1, min = 0, max = 2, step = 0.01),
        sliderInput("damping", "damping", value = 1, min = 0.9, max = 1, step = 0.001)
        numericInput("length", "length", value = 100),
        bookmarkButton()

      ),
      mainPanel(
        plotOutput("fig")
      )
    )
  )
}

#shinyApp(ui, server,enableBookmarking = "url ")
```

:

```
# # Automatically bookmark every time an input changes
# observe({
#   reactiveValuesToList(input)
#   session$doBookmark()
# })
# # Update the query string
# onBookmarked(updateQueryString)
```

:

```
server <- function(input, output, session) {
  t <- reactive(seq(0, input$length, length = input$length * 100))
  x <- reactive(sin(input$omega * t() + input$delta) * input$damping ^ t())
  y <- reactive(sin(t()) * input$damping ^ t())

  output$fig <- renderPlot({
    plot(x(), y(), axes = FALSE, xlab = "", ylab = "", type = "l", lwd = 2)
  }, res = 96)
```

```
  observe({
    reactiveValuesToList(input)
    session$doBookmark()
  })
  onBookmarked(updateQueryString)
}
```

bookmark

```
library(shiny)

ui <- function(request){
  fluidPage(
    sidebarLayout(
      sidebarPanel(
        sliderInput("omega", "omega", value = 1, min = -2, max = 2, step = 0.01),
        sliderInput("delta", "delta", value = 1, min = 0, max = 2, step = 0.01),
        sliderInput("damping", "damping", value = 1, min = 0.9, max = 1, step = 0.001),
        numericInput("length", "length", value = 100),
        bookmarkButton()

      ),
      mainPanel(
        plotOutput("fig")
      )
    )
  )
}

server <- function(input, output, session) {
  t <- reactive(seq(0, input$length, length = input$length * 100))
  x <- reactive(sin(input$omega * t() + input$delta) * input$damping ^ t())
  y <- reactive(sin(t()) * input$damping ^ t())

  output$fig <- renderPlot({
    plot(x(), y(), axes = FALSE, xlab = "", ylab = "", type = "l", lwd = 2)
  }, res = 96)

  observe({
    reactiveValuesToList(input)
    session$doBookmark()
  })
  onBookmarked(updateQueryString)
}
```

7.2 Storing richer state

enableBookmarking="server" server rds URL

```
#shinyApp(ui, server, enableBookmarking = "server")
```