Interactive Graphics in R

Wen-Jie Tseng

05.09.2017 @ NCKU

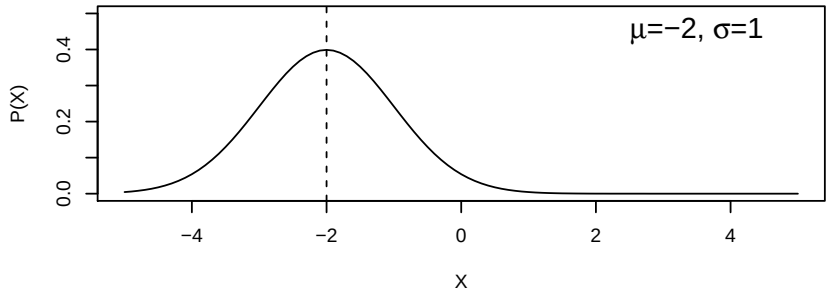
Overview

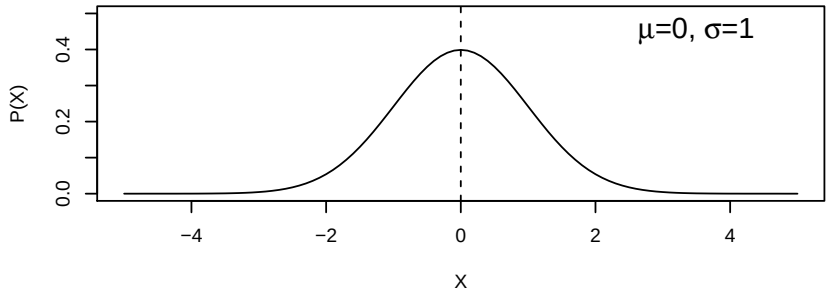
- Why interactive?
- Tools
  - shiny
  - ggvis
- Demo
  - Exploring data
  - Introducing concept
  - Building experiment
- Discussion
- References

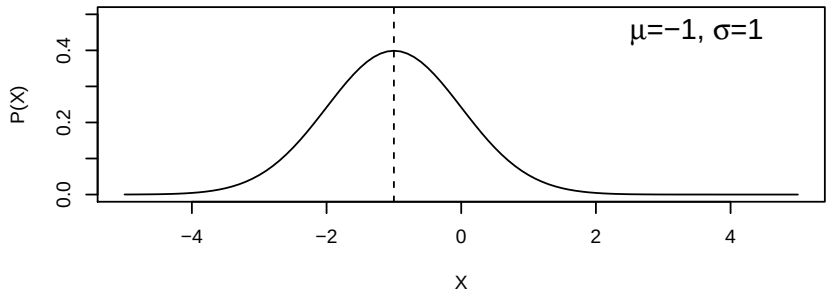
Why interactive?

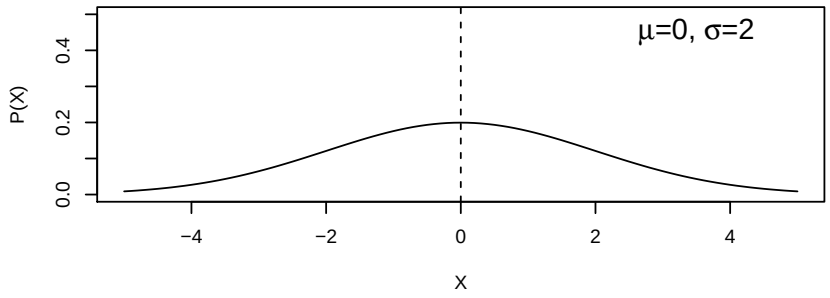
- Static vs. Dynamic
- Instantly Feedback: Interactivity
- Easy to implement

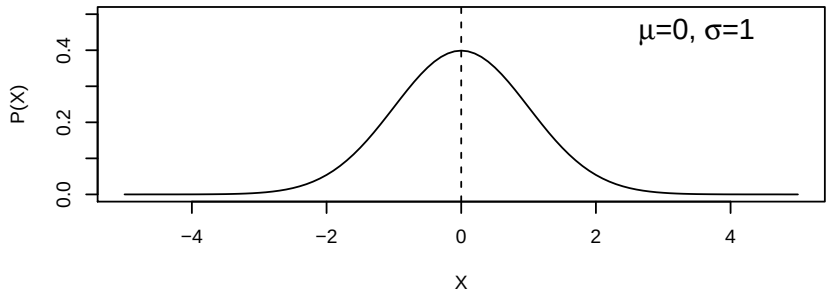
Graph: normal distribution

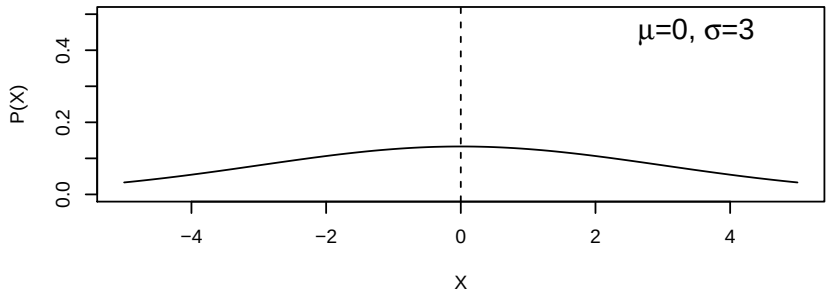












Animation: normal distribution

Fantastic interactive graphics - D3.js

- Homepage
- One has to learn JavaScript!

Tools

- R packages
  1. ggvis
  2. shiny
- Others: shinyjs, animation, ggplot2, dplyr, magrittr

Data - UCLA hs0.txt

```
dta <- read.table("hs0.txt", sep = "", header = TRUE)
names(dta)[2] <- c("gender")
str(dta)
```

```
## 'data.frame':    200 obs. of  11 variables:
##  $ id     : int  70 121 86 141 172 113 50 11 84 48 ...
##  $ gender : Factor w/ 2 levels "female","male": 2 1 2 2
##  $ race   : Factor w/ 4 levels "african-amer",..: 4 4 4
##  $ ses    : Factor w/ 3 levels "high","low","middle": 2
##  $ schtyp : Factor w/ 2 levels "private","public": 2 2 2
##  $ prog   : Factor w/ 3 levels "academic","general",..:
##  $ read   : int  57 68 44 63 47 44 50 34 63 57 ...
##  $ write  : int  52 59 33 44 52 52 59 46 57 55 ...
##  $ math   : int  41 53 54 47 57 51 42 45 54 52 ...
##  $ science: int  47 63 58 53 53 63 53 39 58 NA ...
##  $ socst  : int  57 61 31 56 61 61 61 36 51 51 ...
```

1. ggvis

- Web graphic which is implemented by shiny
- The syntax is similiar to ggplot2
- Incorporates shiny and dplyr (%>%)

ggvis - %>% (pipeline)

```
library(ggvis)
library(magrittr) # %>%
library(dplyr)

ggvis(data = dta, x = ~math, y = ~science) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")
```

ggvis - ~

```
dta %>%
  ggvis(~math, ~science, fill = ~ses) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")
```

ggvis - scale_ordinal

```
dta$ses <- factor(dta$ses,
  levels = c("low", "middle", "high"))

ggvis(dta, ~math, ~science, fill = ~ses) %>%
  scale_ordinal("fill",
    range = c("lightblue","steelblue","blue")) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")
```

ggvis - :=

```
ggvis(data = dta, x = ~math, y = ~science,
      fill = ~ses, stroke := "black") %>%
  scale_ordinal("fill",
      range = c("lightblue","steelblue","blue")) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")
```

ggvis - scale_numeric

```
ggvis(dta, ~math, ~science, fill = ~read) %>%
  scale_numeric("fill", range = c("pink","blue")) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")
```

ggvis - layer

```
p <- ggvis(data = dta, x = ~math, y = ~science,
      fill = ~read, opacity := 0.5) %>%
  scale_numeric("fill", range = c("pink","blue")) %>%
  layer_points() %>%
  add_axis("x", title = "Math") %>%
  add_axis("y", title = "Science")

# add new layer
p %>% layer_model_predictions(model = "lm")
p %>% layer_smooths()
```

ggvis - interactivity

```
# add_tooltip
p %>% add_tooltip(function(dta) dta$read)
```

ggvis - interactivity

```
# input_select
ggvis(dta, ~read, ~math, opacity := 0.6) %>%
  layer_points(
    fill = input_select(
      choices = c("Gender" = "gender",
                  "Race" = "race",
                  "Social Economic Status" = "ses",
                  "School Type" = "schtyp",
                  "Program" = "prog"),
      label = "Fill by",
      selected = "gender",
      map = as.name))
```

2. shiny

- web application
- Inherits syntaxes from HTML and JavaScript
- ui.R and server.R

shiny - ui.R

ui (an object or a script) is user interface of shiny application, which receives inputs from user and display the outputs (e.g., text).

```
ui_0 <- shinyUI(fluidPage(
  h3("A simple user interface example"),
  sidebarLayout(
    sidebarPanel(
      textInput("text",label="Just type some texts :)")
    ),
    mainPanel(
      h1(textOutput("out.text"))
    )
  )
))
```

shiny - server.R

server is a function with arguments (input and output, list), which runs code as what we do in console.

```
server_0 <- shinyServer(function(input, output){
  output$out.text <- renderText({
    print(input$text)
  })
})
```

shiny - execute shiny App

1. For running the app, one can do it by shinyApp with objects.
2. Or save ui and server as two .R scripts in same directory, use runApp().

```
shinyApp(ui.0, server.0)
# runApp("your_dir")
```

shiny - an example

- Exploring UCLA data, hs0.txt.

Demo - Exploring data

- Balance of Foreign trade of Taiwan (2015)

Balance of Foreign trade of Taiwan (2015)

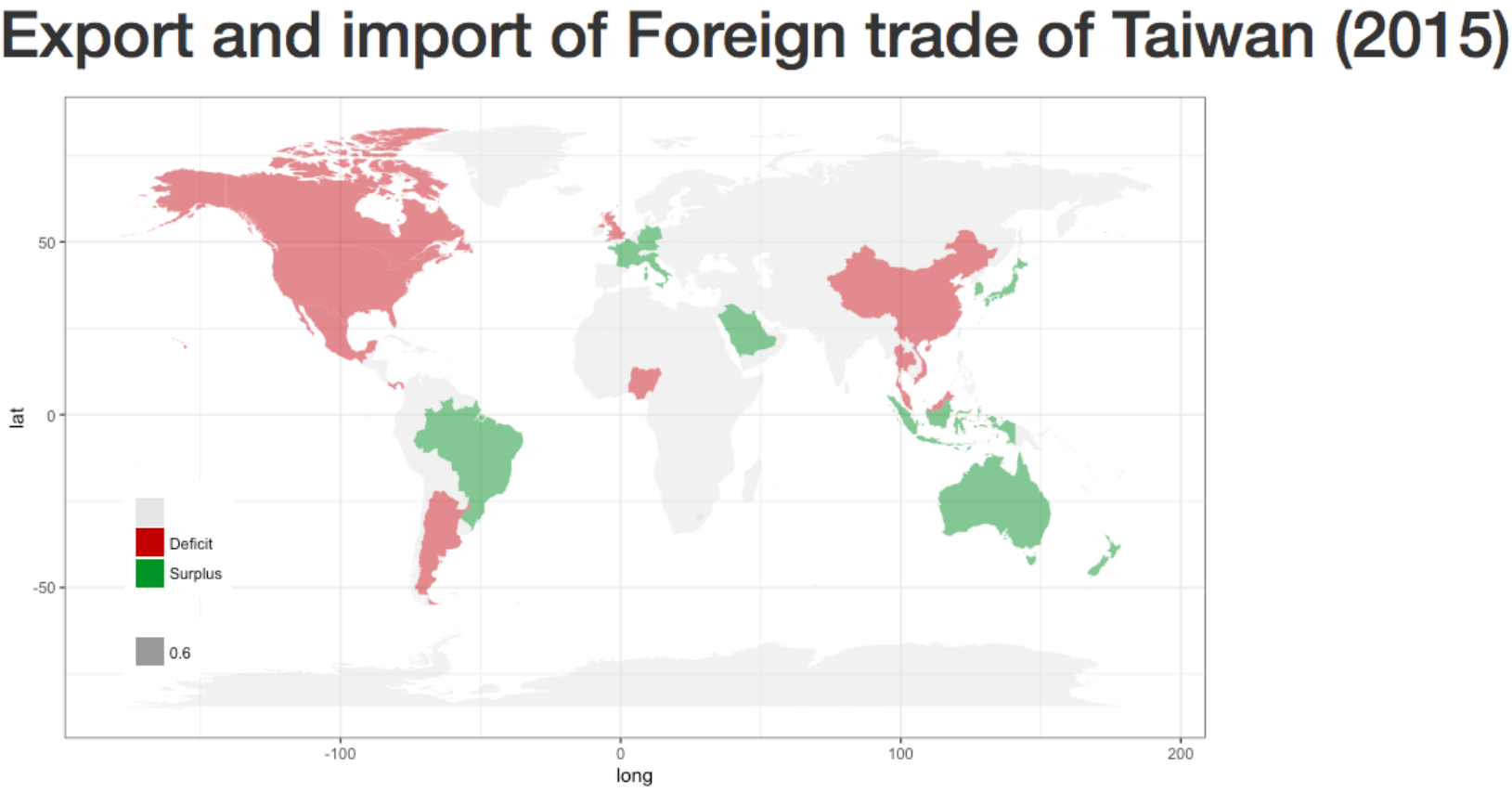


Figure 1

Demo - Introducing concept

- Demonstrate statistical concepts
  1. Optimization of simple regression
  2. Signal detection theory
  3. Rowing dice, sampling distribution, and expectation

Demo - Building experiment

1. Hot-hand
2. Recognition memory

Some more issues before jumping into experiment examples

- I/O
- observe, observeEvent, reactive, <<-

Discussion

1. The reason of rejection:
   - No emprical data support
   - Learning statistics is too complicate to measure
2. About experiment:
   - Reaction time
   - Latency
3. About user interface:
   - Keyboard input
   - Neat layout

Compare ggvis and shiny

Properties	Order
Convenient	ggvis > shiny
Difficulty	shiny > ggvis
Flexibility	shiny > ggvis

References

1. URLs

- `http://ggvis.rstudio.com/`
- `https://shiny.rstudio.com/`
- National Statistics
- Bureau of Foreign Trade

2. Books

- D. A. Norman (1988) The Design of Everyday Things.

Thank You for Your Attention!