Pivotal

Pivotal Beamer Template

Example Presentation

Ronert Obst, Data Scientist

April 23, 2014

Outline

1. First Section

2. Second Section

3. Some Code

First Section

Buzzwords

- Big data
  1. Next generation data science
  2. Elastic deep learning
- Industry 8.0
- IoT 5.0
- More cloud → more winning

Two columns

- Considering the aspect ratio is 16:9
- It is probably a good idea to use two columns
- To avoid really long lines
- Typesetting just looks nicer than in PowerPoint
- And it can be versioned!

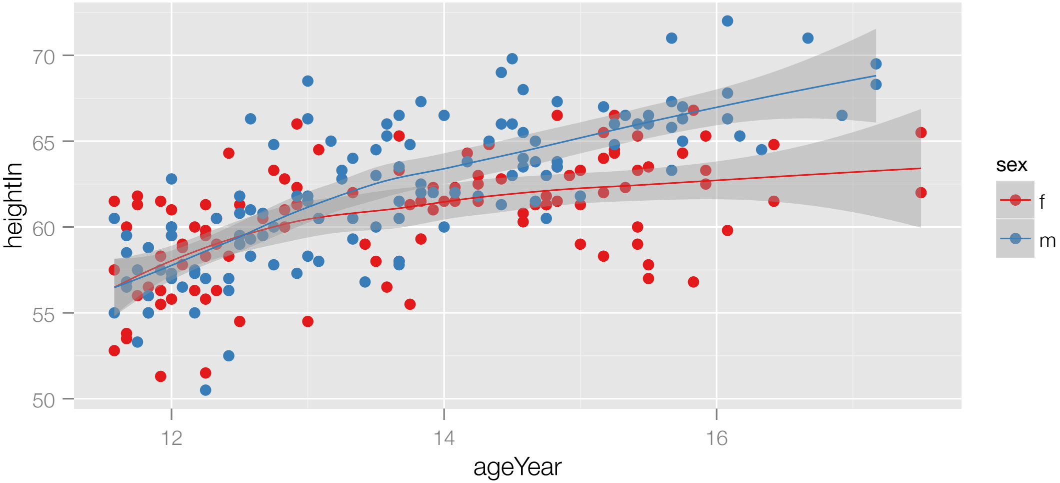
- Some other content on this side
- Big data
- More big data
- Hadoop
- GPDB
- MADlib
- $x + \mathrm{y} = z$

Let us create a plot with `ggplot2`

- `ggplot2` is a really nice plotting library
- It can export graphics using TikZ
- Which look really nice in LaTeX

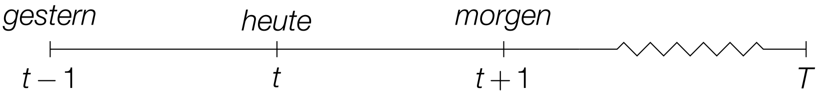
```r
library(ggplot2)
library(gcookbook)
sps <- ggplot(heightweight, aes(x=ageYear, y=heightIn, colour=sex))
+ geom_point()
+ scale_colour_brewer(palette=”Set1”)
sps + geom_smooth()
```

The plot using TikZ



Second Section

A timeline using TikZ



Explaining equations (TEXample.net)

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^{b}d^2}{dt^2}\vec{r} + 2\vec{w}_{ib} \times \frac{{}^{b}d}{dt}\vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{w}_{ib} \times (\vec{w}_{ib} \times \vec{r})$$

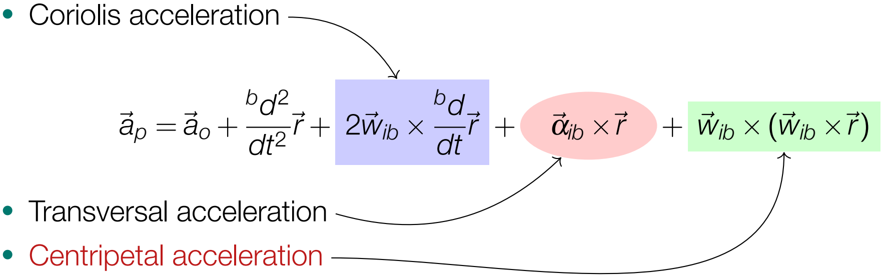
Explaining equations (TEXample.net)

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + 2\vec{w}_{ib} \times \frac{{}^b d}{dt}\vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{w}_{ib} \times (\vec{w}_{ib} \times \vec{r})$$

- Transversal acceleration

Explaining equations (TEXample.net)

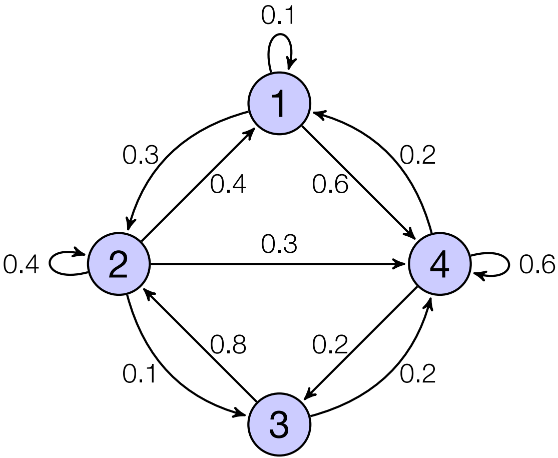


- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + 2\vec{w}_{ib} \times \frac{{}^b d}{dt}\vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{w}_{ib} \times (\vec{w}_{ib} \times \vec{r})$$

- Transversal acceleration
- Centripetal acceleration

A graph (TEXample.net)



Some Code

Python

- Pygments via `minted` has syntax highlighting for all major languages

```python
# List comprehensions
num = [1, 4, -5, 10, -7, 2, 3, -1]
filtered_and_squared = [ x**2 for x in num if x > 0]
print filtered_and_squared

# Generators
num = [1, 4, -5, 10, -7, 2, 3, -1]
filtered_and_squared = ( x**2 for x in num if x > 0 )
print filtered_and_squared

for item in filtered_and_squared:
    print item
```

New dplyr package

```r
library(dplyr)
# Built in data frame
head(hflights)

# Coerce to data table
hflights_dt <- tbl_dt(hflights)

# Caches data in local SQLite db
hflights_db1 <- tbl(hflights_sqlite(), ”hflights”)

# Caches data in local postgres db
hflights_db2 <- tbl(hflights_postgres(), ”hflights”)

carriers_df  <- group_by(hflights, UniqueCarrier)
carriers_dt  <- group_by(hflights_dt, UniqueCarrier)
carriers_db1 <- group_by(hflights_db1, UniqueCarrier)
carriers_db2 <- group_by(hflights_db2, UniqueCarrier)
```

SQL

```sql
select public.diabetes.diabetes,
    madlib.logregr_predict( array[1, times_pregnant,
            plasma,
            diastolic_blood_pressure,
            triceps_skin_fold_thickness,
            serum_insulin,
            bmi,
            diabetes_pedigree,
            age],
        public.diabetes_log_reg_ronert.coef
        )::integer as predict
from public.diabetes, public.diabetes_log_reg_ronert limit 100;
```