Introduction of R and Traffic Safety Analysis

CEE 412/ CET 522

Transportation Data Management and Visualization

WINTER 2020

Before moving to R

Transportation Data

- Process → Management → $\boxed{\text{Analysis}}$ → Visualization

Analysis and Visualization: R and Python has similar functionalities

- R
  - Analysis: various packages
  - Visualization: ggplot2, Shinny
- Python
  - Analysis: Various packages, including NumPy, Pandas, SciPy, scikit-learning, TensorFlow, PyTorch
  - Visualization: matplotlib, Streamlit

We choose R, but it does not imply that R is better than Python in terms of analyzing and visualizing transportation data.

Introduction to R

R Introduction

- R is an open source programming language and software environment for statistical computing and graphics.
- R is based on the S language originally developed by John Chambers and colleagues at AT&T Bell labs in the late 1970s and early 1980s.
- Many classical and modern statistical techniques have been implemented in R in the base environment or as packages.
- Volunteers continue to create and update the software packages.

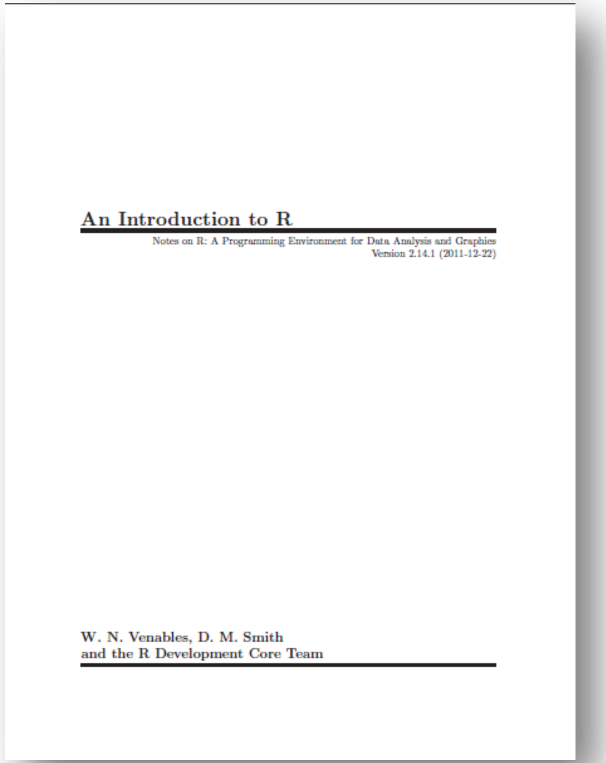
Advantages of R

- FREE!
- Abundant Web Resources and User Network
- Large number of packages to support a diverse range of applications
- Data structure, manipulation and analysis
- Statistical modeling
- Data visualization

Web Resources

An Introduction to R

http://cran.r-project.org/doc/manuals/R-intro.pdf



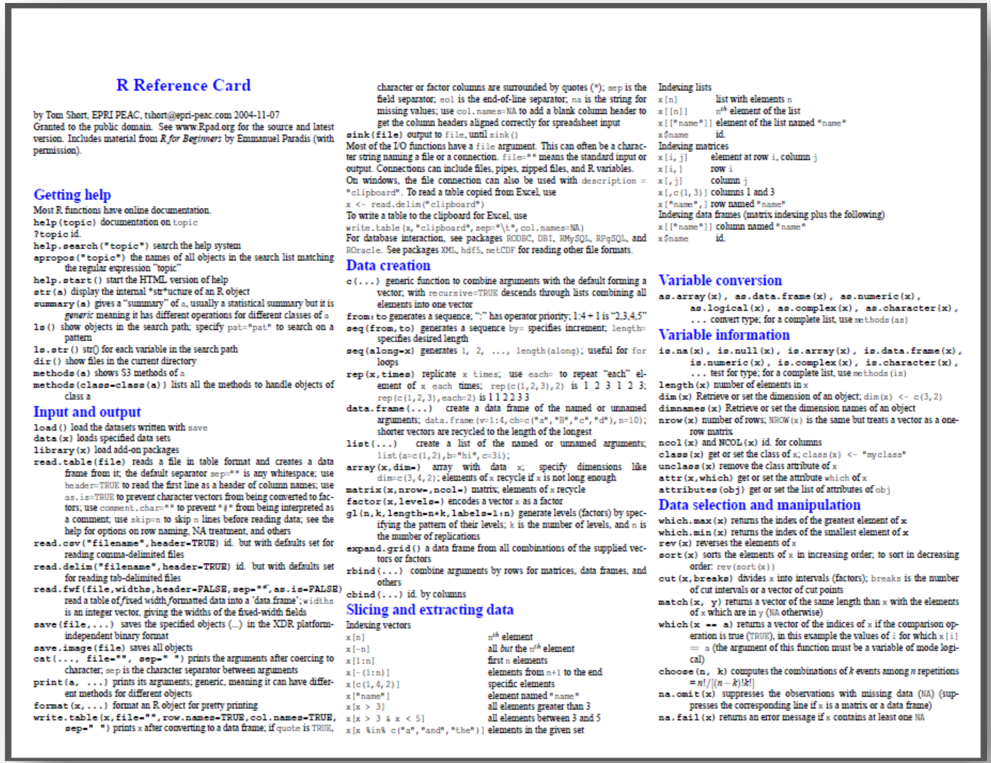
An Introduction to R

Notes on R: A Programming Environment for Data Analysis and Graphics
Version 2.14.1 (2011-12-22)

W. N. Venables, D. M. Smith
and the R Development Core Team

R Reference Card (Tom Short)

http://cran.r-project.org/doc/contrib/Short-refcard.pdf



R Reference Card

by Tom Short, EPRI PEAC, tshort@epri-peac.com 2004-11-07
Granted to the public domain. See www.Rpad.org for the source and latest version. Includes material from *R for Beginners* by Emmanuel Paradis (with permission).

Getting help

Input and output

Data creation

Slicing and extracting data

Variable conversion

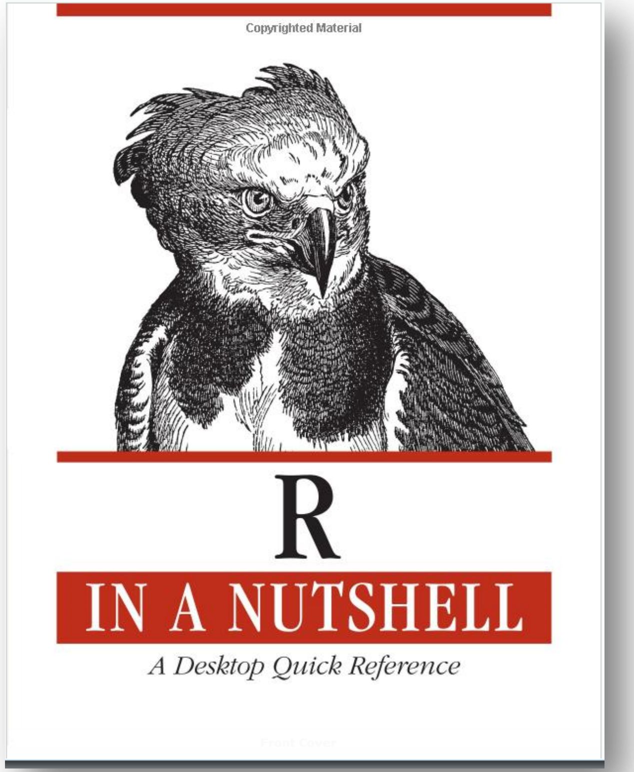
Variable information

Data selection and manipulation

Textbooks for Quick Reference

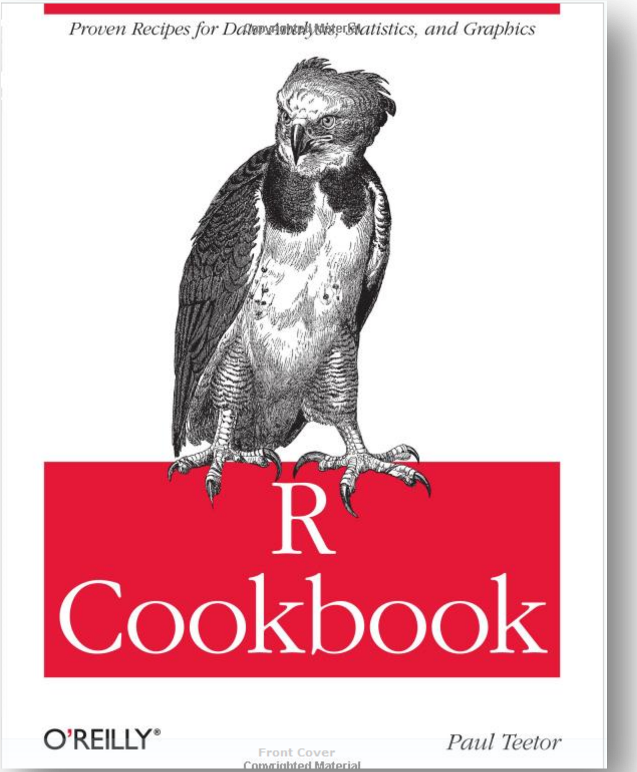
R in a Nutshell, 2nd Edition

BY Joseph Adler



R Cookbook

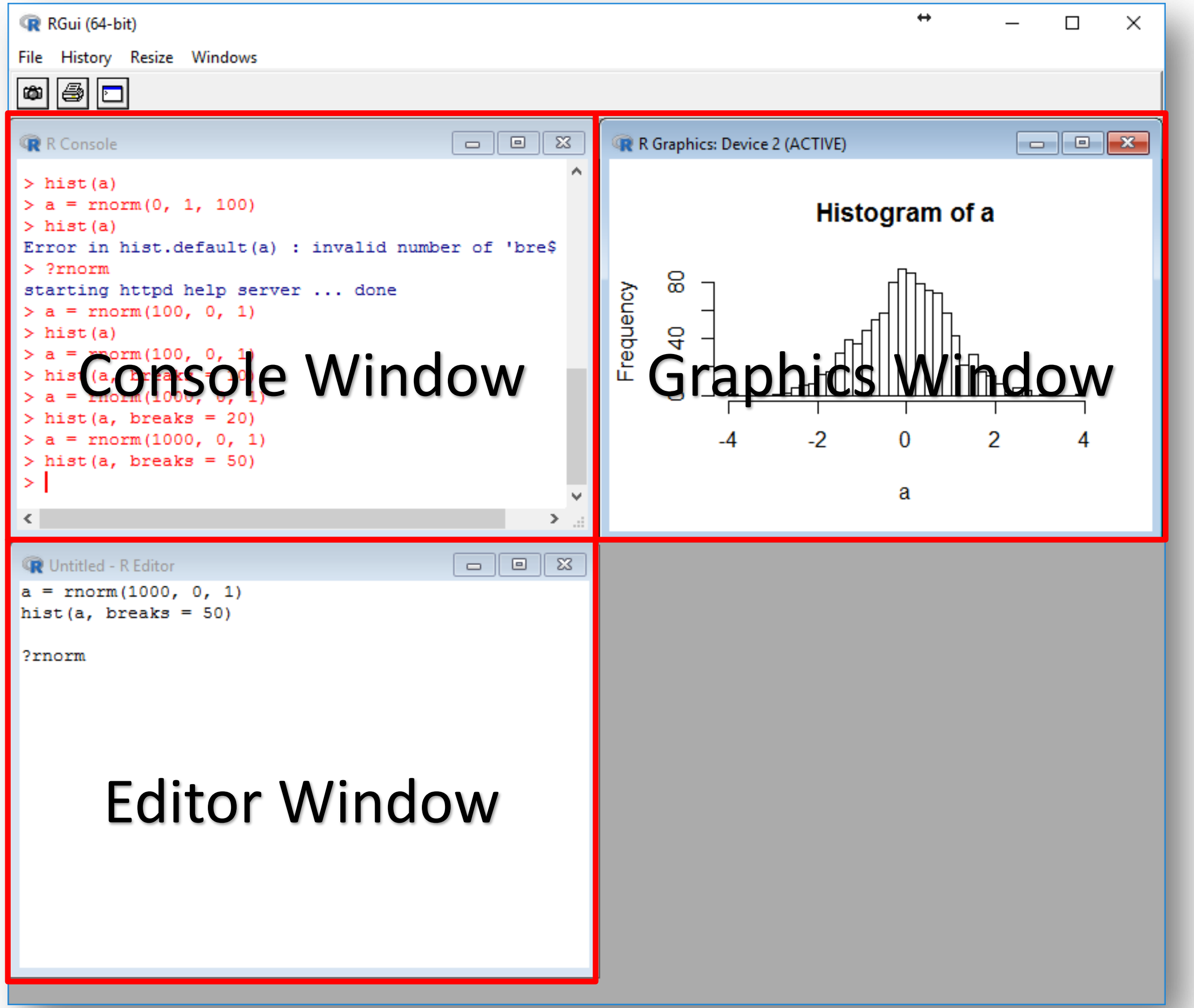
BY Paul Teetor



R Environment

R

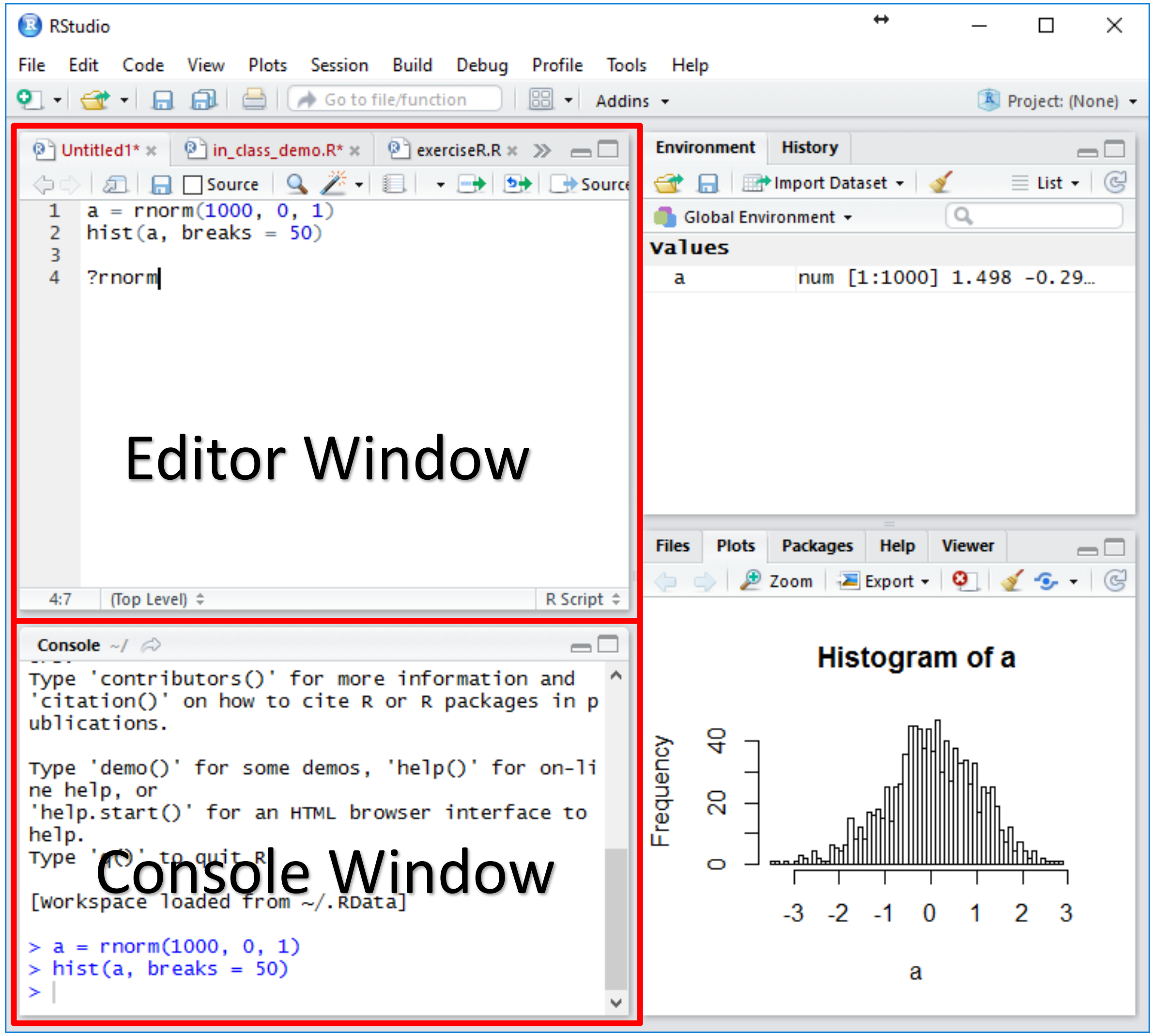
- Develop and edit scripts in Editor Window.
- Ctrl + R to execute the command in the Console.
- You can move and resize windows to change layout.
- Download: http://cran.rstudio.com/



R Environment

R Studio

- IDE for R with better editing interface and more GUI options
- Runs on Windows, Mac, Linux, and even in the web browser using R Studio Server
- To work with R Studio, you must also have installed R.
- Free to download: http://www.rstudio.com/products/rstudio/download/



Executing code in R

Type commands in Console Window and hit enter:

- Code is instantly run.
- You can fill in a line with a previously entered expression by pressing the up arrow on your keyboard.
- No editing, except by navigating with arrow keys and typing.

Write script in Editor Window and execute (recommended):

- Edit and save the code into a R file.
- Highlight the part you want to run and hit Ctrl + R.
- Very little help from R in code writing (use R studio instead)

Basic Math in R

Variable assignment

Output values

Variable assignment

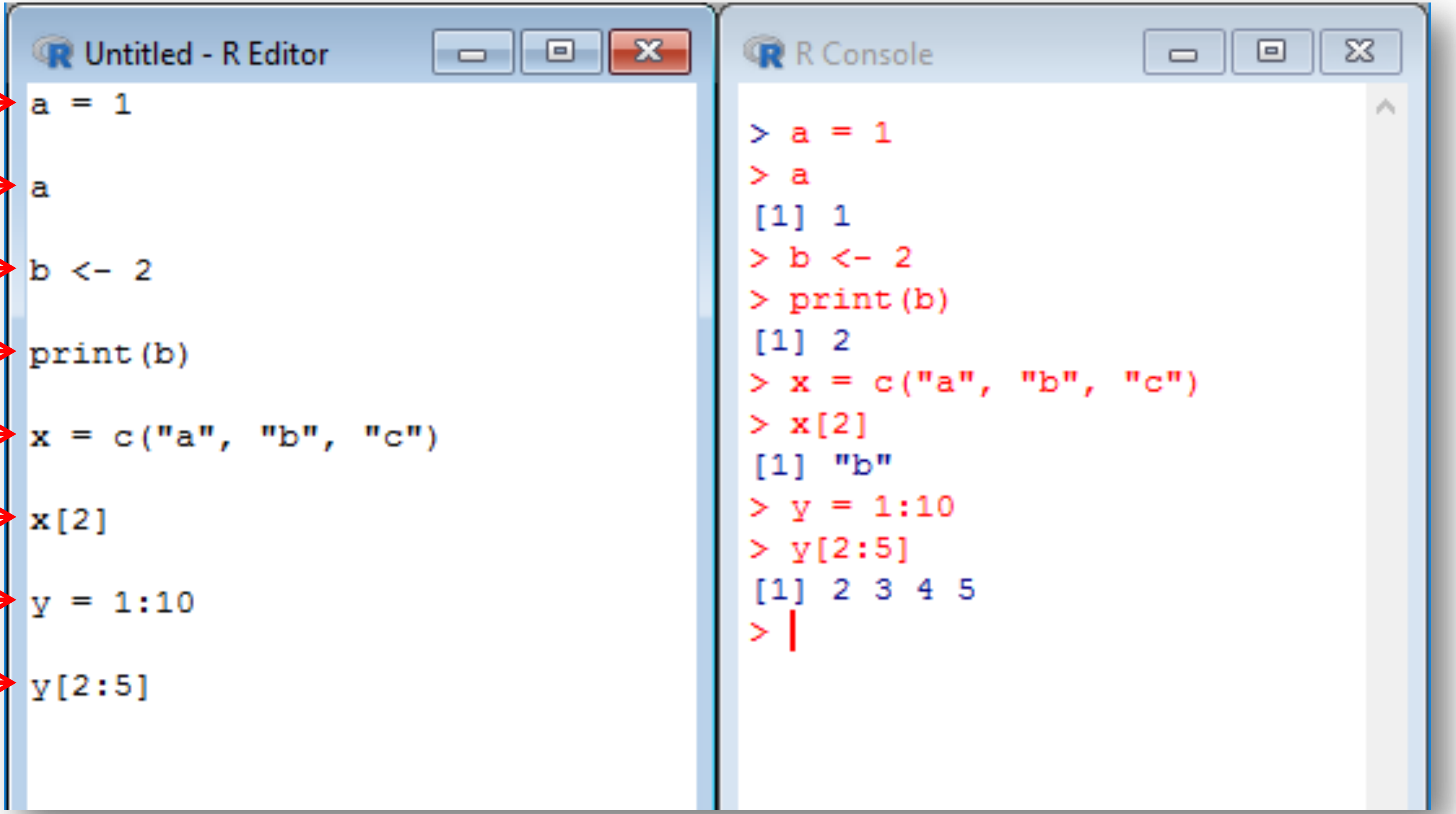
Output values

Vector

Find an element in vector

Colon: from…to…

Subset in a vector



Data Structures in R

Basic data types:

- Numeric, logical, character, factor, etc.

Matrices:

- All columns have the same data type
- All columns/rows the same length

Dataframes:

- Conceptually similar to data tables (database relation, Excel sheet)
- Columns can be of different data types (but same length)

Lists:

- Ordered collection of objects
- List of numbers, list of dataframes, list of matrices, list of lists, etc.

class(object) → returns the object type

str(object) → returns the structure of an object

Dataframes in R

Useful functions:

- data.frame(object) --- create a dataframe from compatible object(s)
- nrow(dataframe), ncol(dataframe) --- return number of rows/columns
- names(dataframe) --- return the column headings of dataframe
- dataframe[1,2] --- return the value of the 1st row, 2nd column in dataframe
- Dataframe$columnname – return one column as a vector

Dataframes in R

Create vectors (columns)

Create a dataframe

Select a column

Select an element

Select a row

Name	Age
Peter	24
Mark	19
Ellen	22

Untitled - R Editor

```
name = c("Peter","Mark","Ellen")
age = c(24,19,22)

student = data.frame(name,age)

student$age

student[2,2]

student$age[2]

student[1,]
```

R Console

```
> name = c("Peter","Mark","Elle$
> age = c(24,19,22)
> student = data.frame(name,age)
> student$age
[1] 24 19 22
> student[2,2]
[1] 19
> student$age[2]
[1] 19
> student[1,]
   name age
1 Peter  24
> 
```

Dataframes in R

Filtering rows and columns in a dataframe.

- Find students who are 20 or older

```
> student[student$age >= 20,]
   name age
1 Peter  24
3 Ellen  22
```

This statement does a pairwise value comparison, and returns a logical vector i.e. {FALSE, FALSE, TRUE…}

- Or

```
> select = which(student$age >= 20)
> student[select,]
   name age
1 Peter  24
3 Ellen  22
```

The “which” function returns the vector indices where the statement evaluates to TRUE

R vs. SQL

Syntax:

- R is case sensitive; SQL is not case sensitive.
- R is sensitive to line feed; SQL is not sensitive to line feed.
- Both are not sensitive to indentation.

Data structure:

- In SQL, tuples in a relation are unordered.
- In R, rows in a dataframe are ordered.

Operators and functions:

- A lot of differences, below are some common used ones:

	R	SQL
Comparison operator	==	=
Logical operator	!, &, |	NOT, AND, OR
Function	mean()	AVG()

Specify Working Directory

R reads and writes files to the working directory unless otherwise specified.

There are two possible methods to specify the working directory.

1. Run the setwd() function:

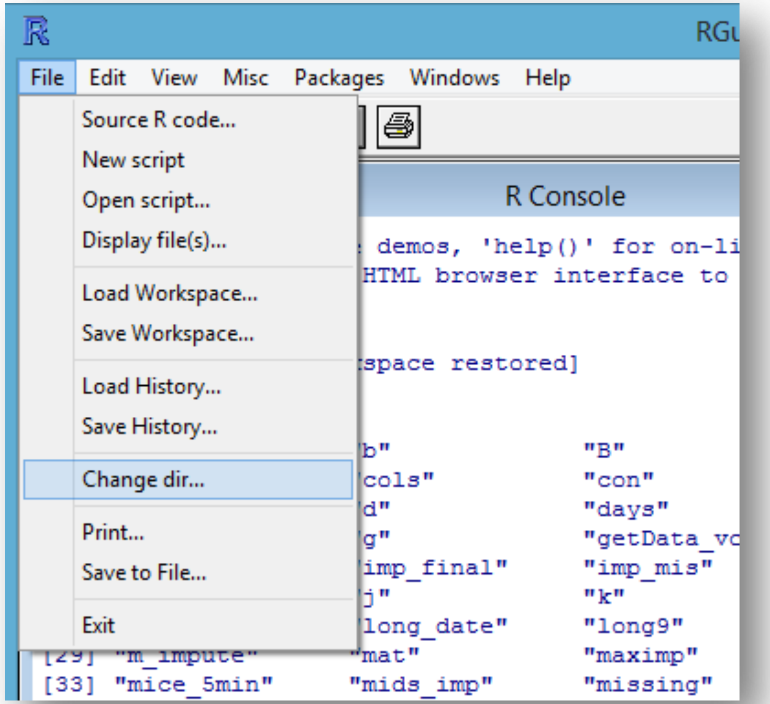
```
> setwd("C:/Users/Zhiyong Cui/CEE412_CET522")
```

- All file path names need to use the forward slash / not the backward slash \
- R is **case sensitive**. Check your folder names.

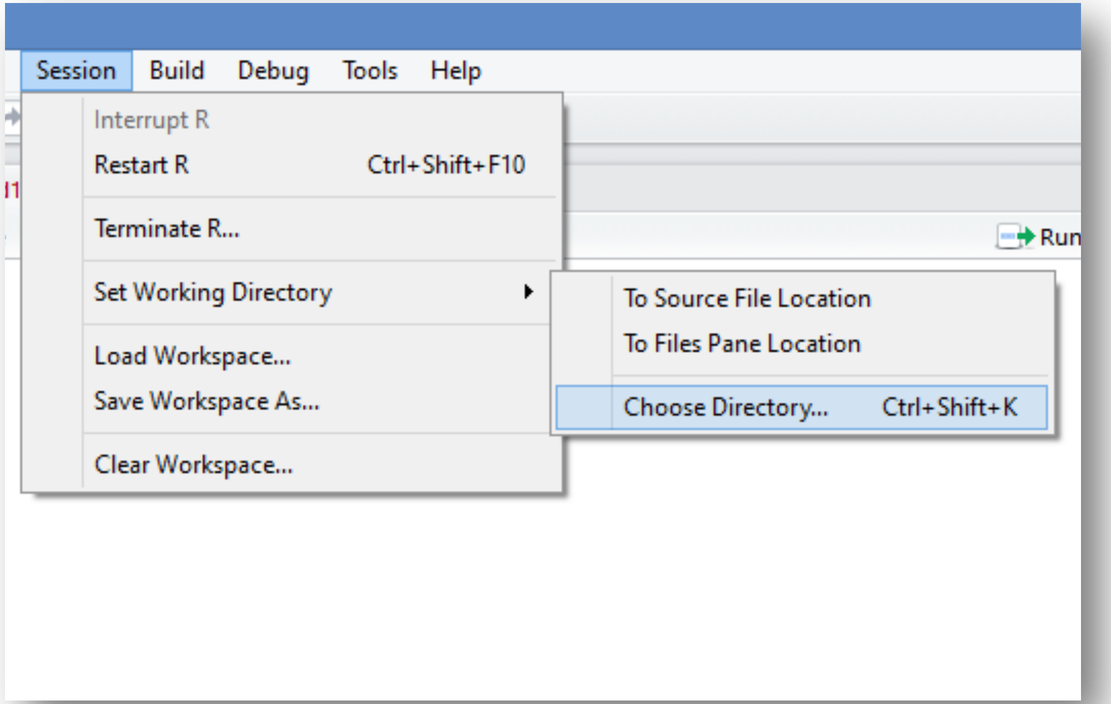
Specify Working Directory

2. Specify the working directory in the GUI

R:



R Studio:



Data Import

Data files supported in R

- Text files (CSV is often used)
- Web URLs
- Excel, Minitab, SPSS, etc. (supported by functions in different packages)
- Database connection

Load data from a CSV file using read.csv() function:

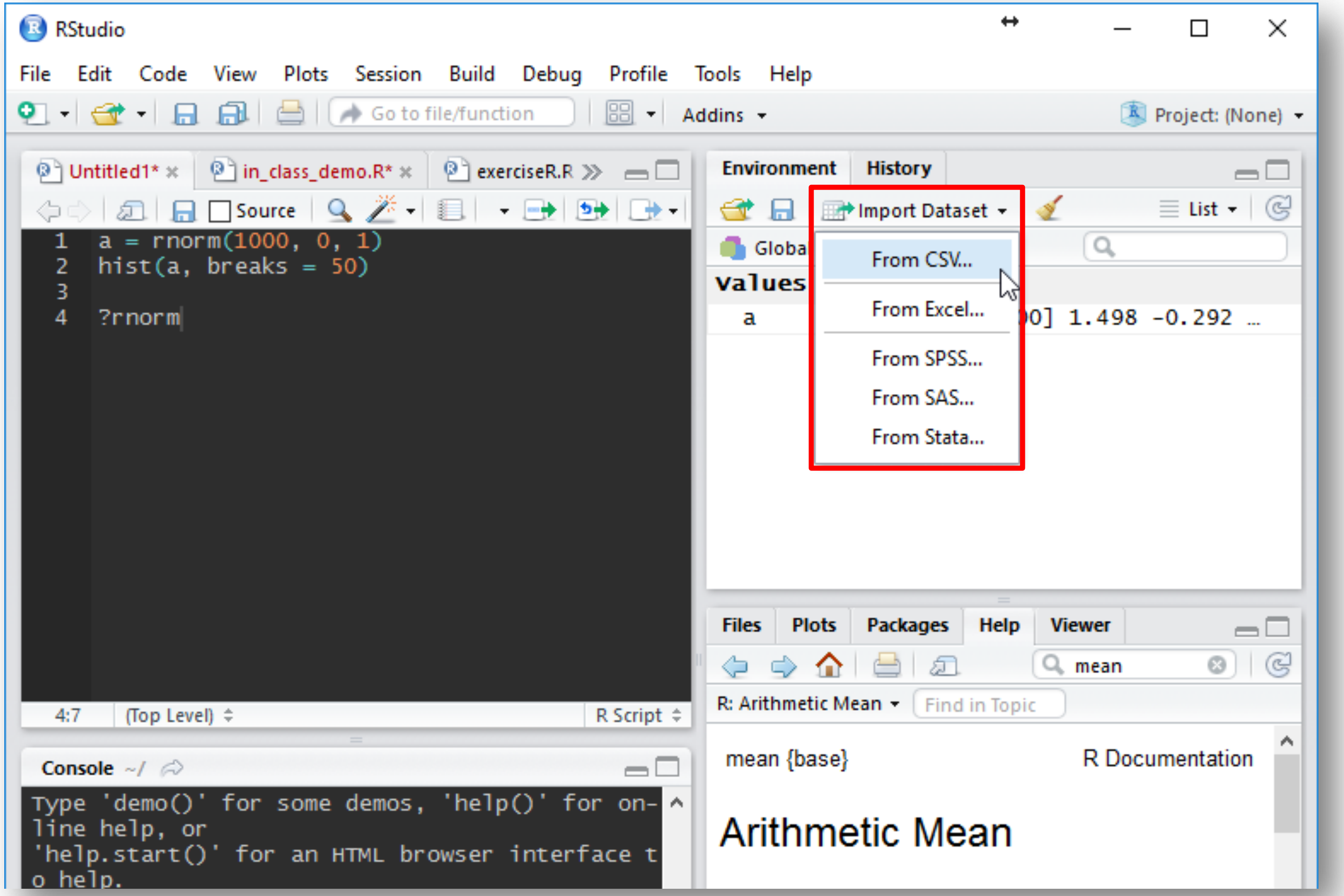
```
accident = read.csv("wa11acc.csv")
```

- In this case, the CSV file is my working directory, otherwise I need to specify the file path relative to my working directory.

Data Import

Load data using R Studio GUI:

Import Dataset → From CSV…



Packages in R

All R functions and datasets are stored in packages.

- Packages must first be installed, and then loaded for use in any given R session.
- The standard (base) packages that contain the basic R functions are automatically available in R installation.

Examples of R packages:

- caret – tools for regression and classification models
- ggplot2 – graphics and plots
- data.table – tools for working with large datasets
- randomForest – tools for creating and training random forest models

Packages in R

Install a package:

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

Quotes required

- When you run the code to install a package, you will be prompted to select a mirror. In general, choose the one closest to you.

Then, load the package:

```
> library(ggplot2)
```

Quotes optional

- Every time you reopen a R session, you need to load the packages, but no need to reinstall the package.

Database Connection

The RODBC Package

- Allow connections to a database and a variety of database operations (query, insert, update, etc.)
- Embed queries in R code
- Just another package in R
- General in structure, can connect to a variety of DBMSs

Install the package:

```
> install.packages("RODBC")
```

Database Connection

Two groups of functions in the RODBC package:

- Low level – not often used
- SQL – high level, SQL functionality

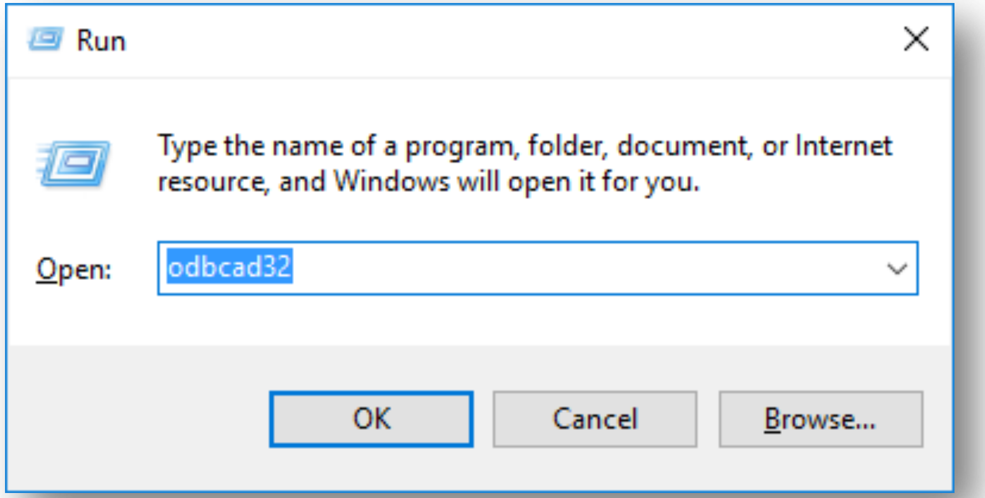
Relies on Open Database Connectivity (ODBC)

- Programing API for accessing DBMSs
- Requires an ODBC driver, which is a middleware between the DBMS and software
- Less important as time goes on, some newer software development platforms do not require it

Database Connection

Create a Data Source Name (DSN) in windows

- This is a file that defines the connection to a particular database
- Includes connection string: user, password, database name
- Click **Start** → **Run** or press Win+R and type “odbcad32”



- Create a SQL server connection with login information.

Database Connection

Now, you can connect to your database from R and start running queries!

- Load RODBC package:

```
> library(RODBC)
```

- Create connection to database:

```
> conn <- odbcConnect("CEE412_CET522", “Username", “Password")
```

DSN

SQL Server login

Database Connection

Useful functions in RODBC:

- Find out what tables are available:

```
> sqlTables(conn)
```

- Fetch a table as a dataframe:

```
> table = sqlFetch(conn, "table_name")
```

- Fetch data using arbitrary query (returns dataframe):

```
> table = sqlQuery(conn, "SELECT * FROM table_name")
```

Database Connection

Useful functions in RODBC:

- Save data to existing or new table in SQL Server:

If true, values will be inserted into existing table. If false, will try to create a new table and fail if table already exists.

```
> sqlSave(conn, dataframe, “table_name”, append=TRUE)
```

Dataframe in R

Table name in SQL

Database Connection

The process of accessing a database as a data source is as follows:

1. Install and load necessary packages (RODBC in this case)
2. Create a connection object (in R) with information including name of the connection, user credentials for accessing database, and location of the data source (ip address, etc.)
3. Run queries – update, fetch, save, etc. – using connection
4. Close connection

This process is very similar for other programing languages: python, C#, etc.

The connection information will be saved in your code, so don't share it in a form that would allow someone to get access to your credentials.

Traffic Safety Analysis

Traffic Safety Analysis

- Safety analysis might seem like a very conventional topic in transportation, but still very important today.
- Traffic accident rate in the US is still high compared with many developed countries.
- Crashes are the leading cause of deaths in the U.S. for ages 15-24.
- Road traffic accidents put a heavy financial burden on our society, both in direct costs and terms of congestion, pollution, and policing/medical costs.
- Thus, the cost of safety analysis and improvements is a good investment.

Impact of Traffic Accidents

In the world:

1,250,000

road traffic deaths each year.

In the US:

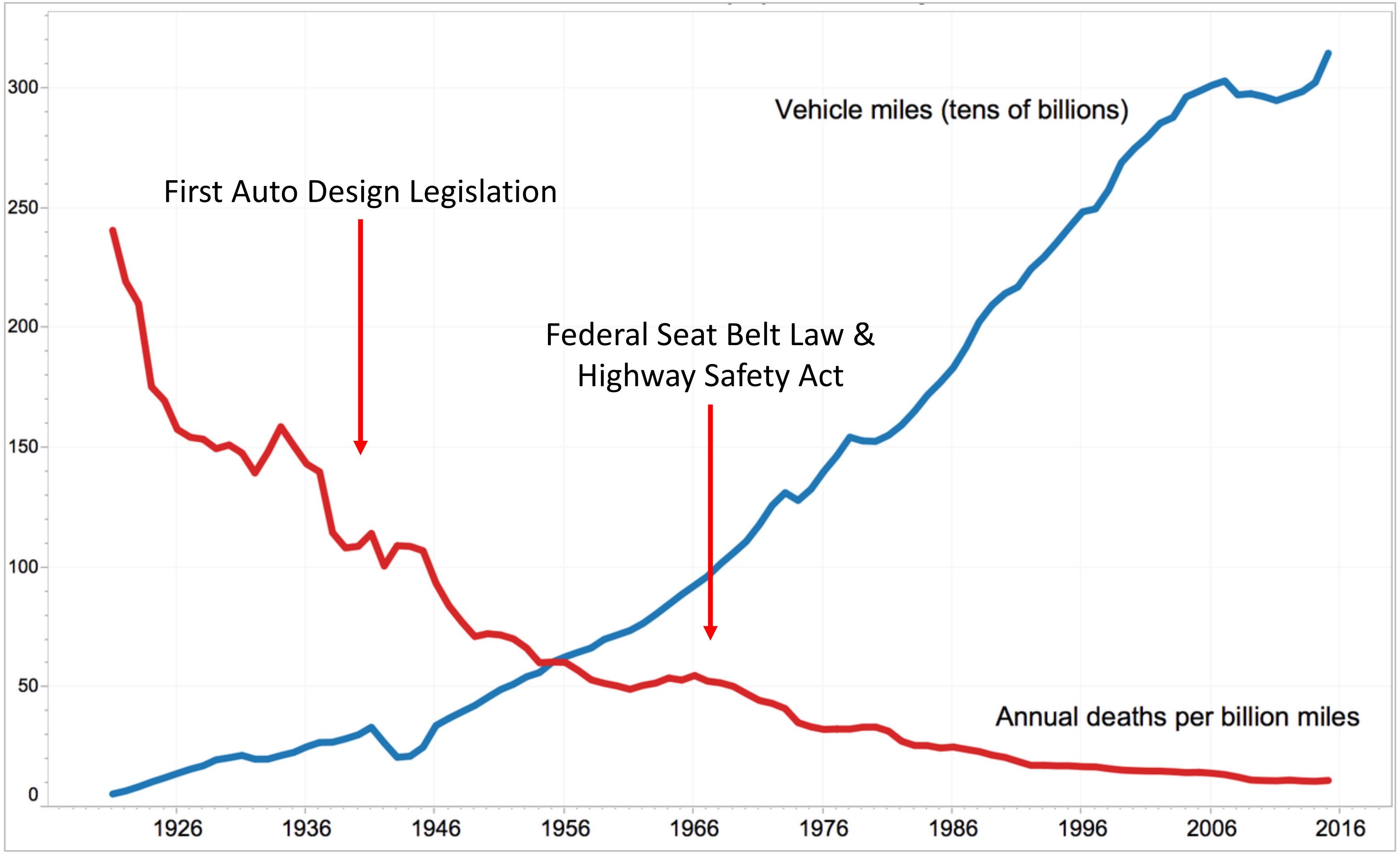
- 36,000+ people died in traffic accidents in 2018.
- 2.3 million injured in traffic accidents in the US.
- Total loss: $267.5 billion in the US.
- Cost of accidents is 2.2 times higher than congestion cost.

Sources:
WHO. http://www.who.int/violence_injury_prevention/road_safety_status/2015/en/
NHTSA. https://www-fars.nhtsa.dot.gov/Main/index.aspx

How Much does a Collision Cost?

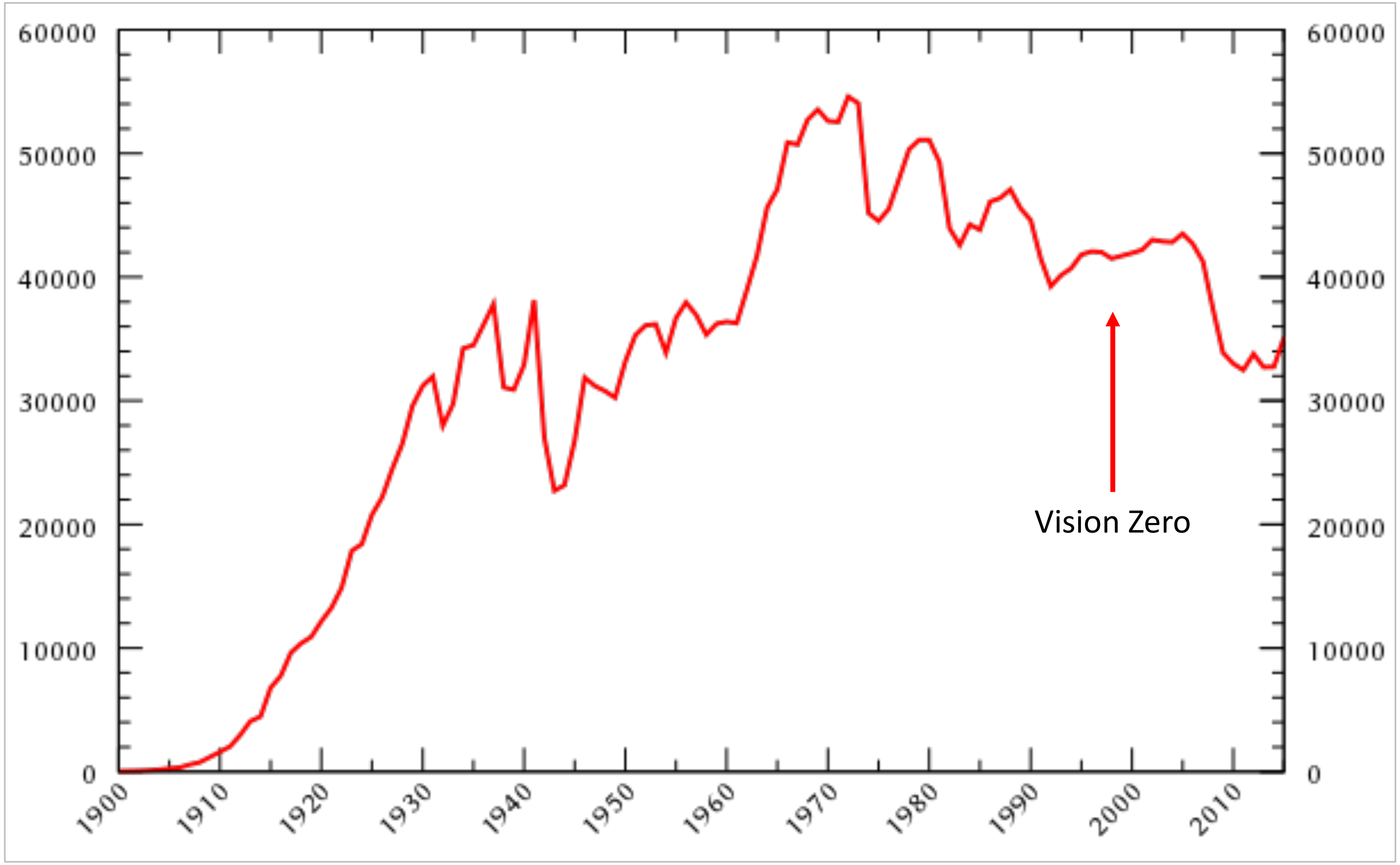
Sources:
2004 Annual State Highway Collision Data Summary, WSDOT, 2006

Vehicle Mile Traveled and Fatality Rate in the US



Sources: FARS, NHTSA. https://upload.wikimedia.org/wikipedia/commons/d/d8/USA_annual_VMT_vs_deaths_per_VMT.png

Annual Crash Fatalities in the US



Sources: FARS, NHTSA. https://upload.wikimedia.org/wikipedia/commons/d/d1/Motor_vehicle_deaths_in_the_US.svg

Traffic Safety Analysis

Common topics:

- Risk analysis based on different factors (e.g., weather, driver, etc.)
- Impact analysis (e.g., accident induced delay, clearance time, etc.)
- **Hotspot identification (HSID)** for safety treatments

A crash hotspot is generally defined as a location that has elevated risk of accidents, and should receive priority consideration for future safety treatments.

How is it done?

- Observed accident count, frequency, property damage, etc.
- Geospatial analysis: kernel density estimation, cluster analysis, etc.
- Modeling: Empirical Bayes, Potential for improvement, others

Accident Hotspot Identification

Why model accidents rather than just fix those locations with high accident counts?

- Crashes are random events, and do not give a stable estimate of risk.
- Some facility classes are inherently higher risk (higher traffic volumes, longer road segment length, etc.).
- We can use statistical models to compare facilities at the same level.

Though more advanced/accurate methods have been developed, Empirical Bayes (EB) is generally considered to be the standard.

Poisson Process

Poisson process is one of the most important models used in queueing theory.

- Time-dependent, event-based count data (e.g., number of customers arrived in a time period)
- Counts in non-concurrent time periods are independent

Distributions

- The event count in a time period follows a **Poisson Distribution**
- The time interval between two consecutive events follows an **Exponential Distribution**

Poisson Distribution

The Poisson distribution describes the data that is generated from a Poisson process.

- Discrete probability distribution, assuming rate (i.e. expected value) is constant in time
- Non-negative
- Expected value is equal to variance
- Widely used to model fixed-time event count data → model accident counts

Probability mass function (pmf):

$$P(Y = y) = \frac{\lambda^{y} e^{-\lambda}}{y!}$$

where:

$\lambda = \mathrm{E}(Y) = \mathrm{Var}(Y)$ → How realistic is this for accident data?

Poisson Regression for Accident Count

Suppose Y is the accident count, which can be influenced by a number of factors, $\{X_1, X_2, \ldots, X_n\}$. $Y \sim Poisson(\lambda)$.

Then, $$P(Y = y) = \frac{\lambda^y e^{-\lambda}}{y!}$$

How to build a connection between X and Y?

- We can model $\lambda = \mathrm{E}(Y)$ as a function of X.

Is the following general linear model a good form?

$$\lambda = X\beta = \beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n$$

- Not really. This simple model could predict negative values, but λ must be positive.

How about this: $$\log(\lambda) = X\beta$$

So that, $$\lambda = e^{\beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n}$$

This do guarantee that λ is positive.

Maximum Likelihood Estimation

- From the previous model, we have:

$$\mathrm{E}(Y) = \lambda = e^{\beta X}$$

$$\mathrm{P}(Y = y) = \frac{(e^{\beta X})^{y} e^{-e^{\beta X}}}{y!}$$

- Assume we have collected some data, including the accident count *Y* and predictor set *X* (e.g., speed limit, traffic volume, weather, etc.).
- The likelihood that the observed data will occur is calculated as the product of probabilities for all observations:

$$L(\beta | X, Y) = \prod_{i=1}^{m} \frac{(e^{\beta X_i})^{y_i} e^{-e^{\beta X_i}}}{y_i!}$$

Maximum Likelihood Estimation (MLE):

- Estimate model parameters by maximizing the likelihood of observations.

Maximum Likelihood Estimation

- Convert to Log likelihood function for computational convenience:

$$LogL(\beta|X,Y) = \sum_{i=1}^{m}(y_i\beta X_i - e^{\beta X_i} - \log(y_i!))$$

This term is a constant, and can be dropped.

- The final objective function:

$$LogL(\beta|X,Y) = \sum_{i=1}^{m}(y_i\beta X_i - e^{\beta X_i})$$

- Goal: find β that results in the highest (log) likelihood for observed data.
- No closed form solution. The software use numerical optimization to find the optimal solution for β.

Negative Binomial Regression

We have done the Poisson regression, but there is still room to further improve the model.

- Consider the assumption of a Poisson process: $\mathrm{E}(Y) = \mathrm{Var}(Y)$. This may not hold true in reality.
- A dispersion term can be introduced to allow $\mathrm{E}(Y) < \mathrm{Var}(Y)$
- This changes into the **negative binomial regression**.

Negative Binomial Regression:

- Now we define:

$$E(y) = \mu \qquad \text{and} \qquad Var(y) = \mu + k\mu^2$$

- Where k is the dispersion parameter, so pmf can be shown as:

$$P(Y = y) = \frac{\Gamma(k^{-1} + y)}{\Gamma(k^{-1})y!}\left(\frac{k\mu}{1 + k\mu}\right)^y \left(\frac{1}{1 + k\mu}\right)^{1/k}$$

Negative Binomial Regression

Where did this come from?

- The outside is still a Poisson model, where $Y \sim Poisson(\lambda)$
- But, we are now allowing λ to be itself a random variable with gamma distribution so $\lambda \sim Gamma(\alpha, \beta)$ with the result that:

$$E(\lambda) = E(Y) = \mu = \alpha\beta$$

- And variance of Y is (law of total variance):

$$Var(Y) = E\big(Var(Y|\lambda)\big) + Var\big(E(Y|\lambda)\big) = E(\lambda) + Var(\lambda)$$
$$= \alpha\beta + \alpha\beta^2 = \mu + k\mu^2$$

- Defining the dispersion parameter k as:

$$k = 1/\alpha$$

Negative Binomial Regression

Now we need to find both regression parameters (the coefficient vector β) and k. Again, this is solved with the maximum likelihood estimation.

Notes about the negative binomial regression:

- As $k \rightarrow 0$, this becomes a Poisson regression.
- Negative binomial regression can handle over-dispersion, not under-dispersion (i.e. variance is less than the mean).

Empirical Bayes Method

Standard statistical inference is based on the likelihood function, with the goal of obtaining maximum likelihood estimates for parameters and standard errors.

Bayesian approach: there is a prior distribution describing knowledge about parameters, prior to considering any data.

- There is a need to define **hyperparameters**, which describe the prior distribution of model parameters.
- In the Empirical Bayes (EB) method, hyperparameters are estimated from observed data.
- In hierarchical Bayes method, the hyperparameters are themselves given a prior distribution (hyperpriors).

Empirical Bayes Method

In the accident HSID analysis, the idea of EB is to combine two pieces of information:

1. The estimated crash count (from SPF).
2. The actual crash count at the location of interest (Observed).

Safety performance function (SPF):

$$SPF = f(\beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n)$$

- Where $X_1, \ldots, X_n$ are traffic and roadway characteristics (e.g., Lane width, Traffic volumes, etc.)
- Multiple model forms can be used, and the **negative binomial regression** is quite universal.

Empirical Bayes Method

- Combine the predicted crash mean with observed count as weighted average:

Expected safety of site i

$$\pi_i = \alpha_i SPF_i + (1 - \alpha_i) K_i$$

K_i is the observed crash count for segment i

α_i is a weighting factor (between 0 and 1)

SPF_i is the NB estimated crash count for segment i

Where does α_i come from?

- Calculated from the dispersion factor, which represents the variance of the SPF estimate:

$$\alpha_i = \frac{1}{1 + SPF_i/(kL_i^{\gamma})}$$

γ is a constant between 0 and 1 (we will use 0 in this class)

k is the dispersion parameter from the NB model

L_i is the length of segment i

Empirical Bayes Method

To finish, rank sites according to expected safety or compute Accident Reduction Potential (ARP) as another measure for prioritizing safety treatments:

$$ARP = (1 - \alpha_i)(K_i - SPF_i)$$

Interpreting ARP:

- If K_i is much larger than SPF_i (e.g., observed crash count large than estimation), this means larger accident reduction potential (and higher priority for safety treatments).
- If α_i is large (i.e. close to 1), the variance in the SPF estimate is higher, so lower ARP (and lower priority for safety treatments).

Considerations for EB Method

1. You will need at minimum 2-3 years of accident data.
2. We considered multiple years of data as one interval, other methods would be needed to consider longer (likely time varying) periods.
3. There is often a need to consider accident type and/or severity levels separately (not covered here).
4. The way we define "similar" road segments could break down considering between-site heterogeneity.
5. If there are a large number of segments with zero accidents, we might want to use a zero inflated negative binomial model.

Source:

Powers, M., and J. Carson. Before-After Crash Analysis: A Primer for Using the Empirical Bayes Method. Tutorial. No. FHWA/MT-04-002-8117-21,. 2004.

Create the Data Table in SQL

Assume we have a database containing, at minimum, an accidents table and roads table

Create a data table for HSID analysis using a SQL statement:

- SELECT all of the road and accident attributes that will be used as predictors;
- Use the COUNT() aggregation function to count the number of accidents associated with each combination of predictor variables; and
- GROUP BY predictor variables (usually fields describing road segment characteristics).

Create the Data Table in SQL

Result table:

Predictors / Independent variables

Response / Dependent variables

RouteNo	BeginMP	EndMP	AADT	Length	SpeedLMT	TruckRate	AccCnt
5	0.00	0.27	123000	0.27	50	9	14
5	0.27	0.28	123000	0.01	50	0	1
5	0.28	0.29	123000	0.01	50	0	3
5	0.29	0.32	123000	0.03	50	0	1
5	0.32	0.39	123000	0.07	50	0	6
5	0.39	0.50	123000	0.11	50	0	3
5	0.50	0.59	123000	0.09	50	0	2
5	0.59	0.60	123000	0.01	50	0	0
5	0.60	0.68	123000	0.08	50	0	1
…	…	…	…	…	…	…	…

Build a Model in R

In R, connect to the SQL database and get the table as a dataframe (below shows the top 15 rows):

Each row represents a road segment with some AADT, speed limit, and truck rate. The accident count is the number of accidents that occurred on that segment.

```
> AccCnt_Table[1:15,]
   RouteNo BeginMP EndMP   AADT Length SpeedLMT TruckRate AccCnt
1        5    0.00  0.27 123000   0.27       50         9     14
2        5    0.27  0.28 123000   0.01       50         0      1
3        5    0.28  0.29 123000   0.01       50         0      3
4        5    0.29  0.32 123000   0.03       50         0      1
5        5    0.32  0.39 123000   0.07       50         0      6
6        5    0.39  0.50 123000   0.11       50         0      3
7        5    0.50  0.59 123000   0.09       50         0      2
8        5    0.59  0.60 123000   0.01       50         0      0
9        5    0.60  0.68 123000   0.08       50         0      1
10       5    0.68  0.69 123000   0.01       50         0      0
11       5    0.69  0.78 123000   0.09       50         8      1
12       5    0.78  0.79 123000   0.01       50         8      0
13       5    0.79  0.82 115958   0.03       50         0      2
14       5    0.82  0.87 115958   0.05       50         0      1
15       5    0.87  1.05 115958   0.18       50         0      2
```

Build a Model in R

Fit a Poisson regression model:

```
> model.pois = glm(AccCnt~TruckRate+SpeedLMT+AADT, family="poisson", data=AccCnt_Table)
```

Function to fit a generalize linear model

Model form

Model type

Data table

- A Poisson regression model is fitted here as an example. Functions to fit a negative binomial model is available in the “MASS” package.

Summarize the model:

```
> summary(model.pois)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.0563  -0.9761  -0.6442  -0.3810  10.4869

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.421e+00  2.246e-01  10.779  < 2e-16 ***
TruckRate   -8.526e-03  2.518e-03  -3.386 0.000708 ***
SpeedLMT    -6.273e-02  3.663e-03 -17.124  < 2e-16 ***
AADT         1.264e-05  2.616e-07  48.317  < 2e-16 ***
```

Build a Model in R

Some other useful functions:

- Access specific model attributes

```
> SPF = model.pois$fitted.values     # model predictions (fitted values)
> AIC = model.pois#aic               # AIC (Akaike Information Criterion) of the model
```

- Plot residuals, normal Q-Q, and other diagnostics plots

```
> plot(model.pois)
```

Recall that in EB method, expected safety is a weighted average of prediction and observation:

$$\pi_i = \alpha_i SPF_i + (1 - \alpha_i) K_i$$

- In R, this can be calculated using (assuming we already have alpha):

```
> pi = alpha*SPF + (1-alpha)*AccCnt_Table$AccCnt
```

- The point: using a vector in some arithmetic operation (e.g., alpha*SPF) will repeat the calculation for each element in the vector. The result will be a vector of the same length.