Text Analysis in R

Lecture 1: Introduction

Helge Liebert

Introduction

- Text data is now commonly used as a modeling input in both research and industry.
- Often involves substantial amounts of preparation.
- Text processing is a common task in IT - many helpful tools available.
- Working with text requires understanding regular expressions.

Basics

Which language to choose?

- *It depends.* Choice is use-case- and taste-specific.
- *Anything* can be done in *any* language. Convenience varies.
- Concepts and toolkits transfer easily most of the time.
- Prior knowledge vs. task suitability.
- Ease of exploratory analysis vs. ease of deployment in production.

Possible options

- Specialized languages: R, Julia, MATLAB/Octave, Stata, Gauss, ...
- General-purpose languages: Python, Perl, Ruby, C, ...
- Choose a high-level, dynamic, interpreted language unless you are sure you require the extra speed of a compiled language.
- Ideally free and open source. Popular is typically better.
- Research ex ante which libraries are mature and best for solving your specific problem.

R

- R is the major statistical programming language.
- It is free, used in many sciences and in industry. Good documentation.
- New models are frequently published and implemented first in R.
- Having data processing and analysis in the same language is nice.
- Good library support for common tools (e.g. databases, regular expressions).
- Specific tasks for which high-level wrapper functions are not available may be very cumbersome.
- In recent years, R development has been very active and libraries exist for almost anything.

Python

- General-purpose programming language, supports object-oriented programming.
- Reads like english. Explicit and clear. Whitespace matters, no braces. (*“There should be one obvious way to do it”*.)
- Used extensively in industry and sciences. Good documentation.
- Libraries for almost anything.
- Many science-related libraries exist for other languages, but rarely are they as mature.
- Good and growing support for statistical modeling.
- A bit less suited for interactive data work (but more so for deployment in production).

This lecture

- The lab sessions utilize R.
- Any task covered by this lecture can be accomplished using R or Python (augmented by shell programs).
- R, Python, SQL and knowing your way around a terminal are highly valued skills on the job market.
- Rule-of-thumb recommendation:
  - Simple data analysis/small text corpora:
    Stick with R. Augment with other tools where required.
  - More involved data processing/larger text corpora:
    Go with Python. You can still analyze data in R.
  - … and whatever program your colleagues are using.

Survey

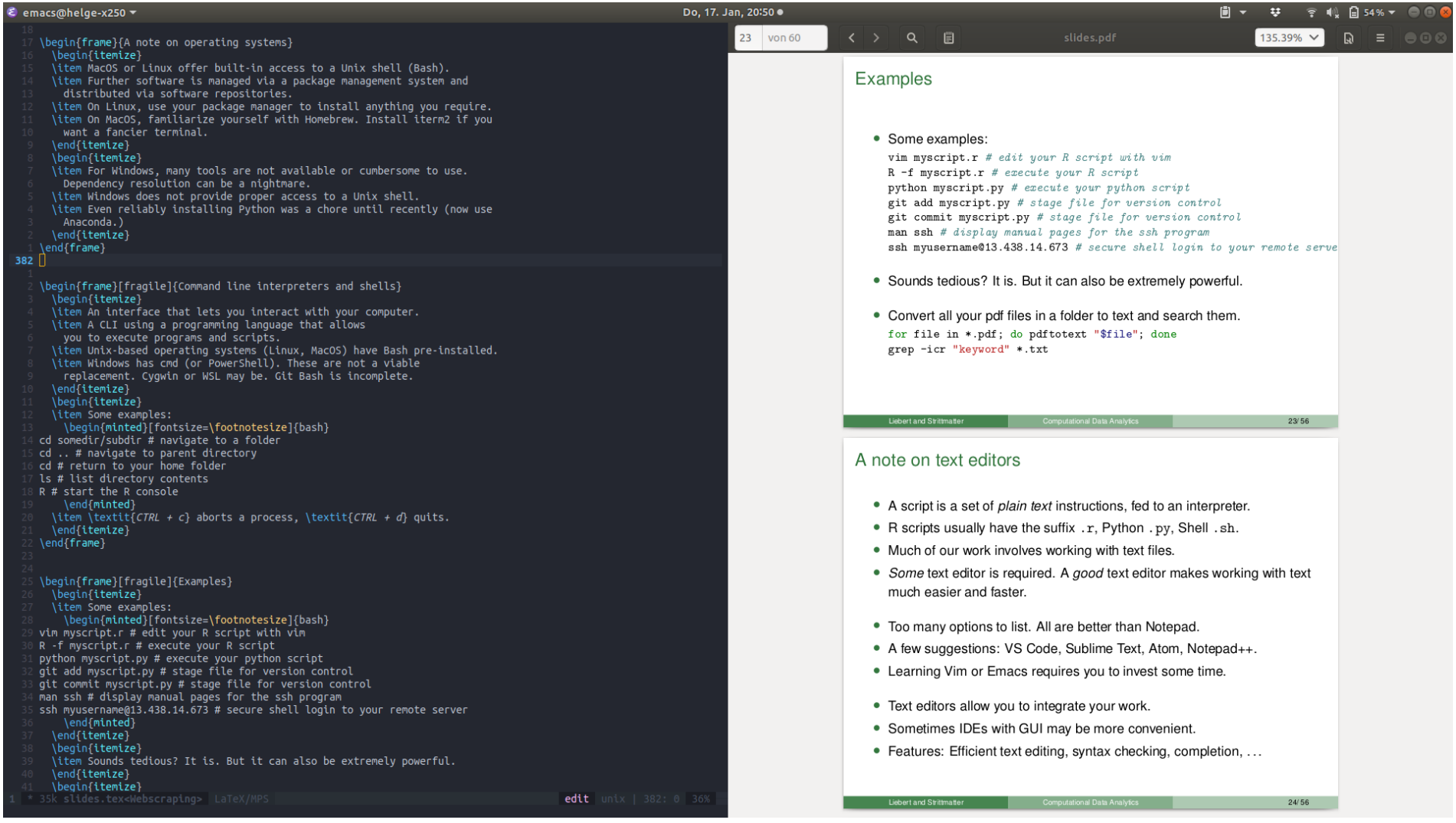
- Operating systems - Windows, MacOS, Linux?
- Scientific programming - Stata, R, Python, Matlab/Octave, Julia, ...?
- General purpose programming languages?
- CLI, servers, shell scripting?
- (Relational) databases - SQL?
- Version control - Git?

A note on text editing

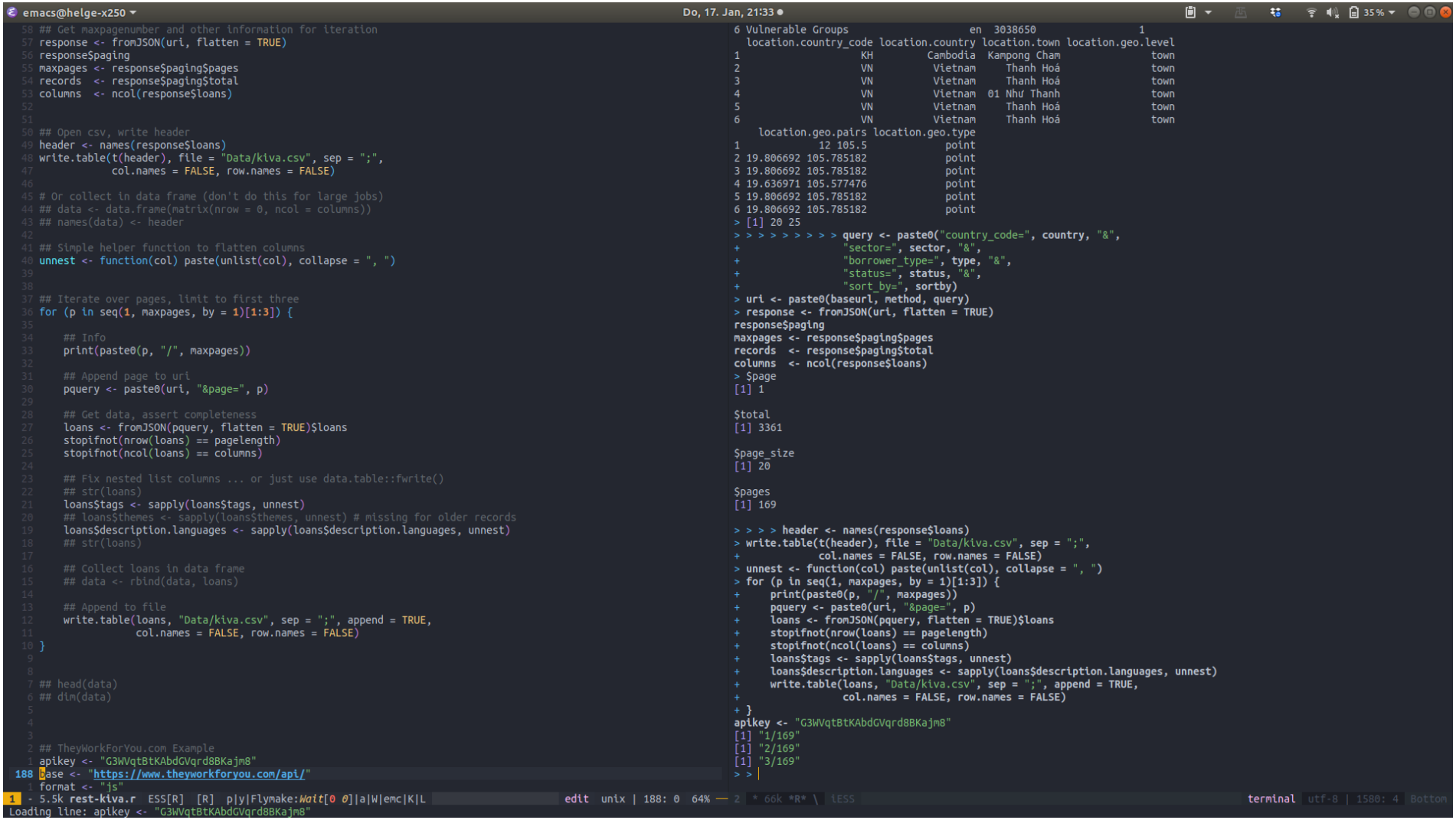
- A script is a set of *plain text* instructions, fed to an interpreter.
- Editing is independent from running code.
- R scripts usually have the suffix `.r`, Python `.py`, Shell `.sh`.
- Proficiency in a text editor makes working with text easier and faster.

- Too many options to list. All are better than Notepad.
- A few options: VS Code, Sublime Text, Atom, Notepad++, ...
- Learning Vim or Emacs requires you to invest some time.

A possible setup



… that is universal



… that is universal

```
each school-track, classroom formation is conditionally ignorable with respect to SN
status, as students from different primary school districts are mixed and their SN status
is not observed by secondary school administrators.\footnote{Using PISA data from
secondary schools in Switzerland, \citet{Vardardottir2015} shows that track-by-school
fixed effects render peer group composition conditionally uncorrelated with a large set
of students' characteristics, while track fixed effects and school fixed effects do not.}
Neither primary schools nor the SPS share information with secondary schools for equity
reasons and to avoid stigma when transitioning between schools.
174
175 % NEU: BEu bitte lesen
176 % changed some small things. ok for me.
177 Beyond this anecdotal evidence, we formally test the validity of the identification
strategy with four balancing tests, which are presented and discussed extensively in
Appendix B. First, we examine whether the proportion of SN peers predicts individual
baseline characteristics (gender, native speaker, and age). The aim of this test is to
detect potential selection into classrooms. We also conduct this test separately for SN
and non-SN students. None of the baseline characteristics are statistically significant
at conventional levels, either considered individually or jointly. Second, we regress the
indicator for SN status on class fixed effects, which should be jointly insignificant if
assignment to classrooms is ignorable with respect to SN status \citep{chettyEtal2011}.
We also conduct this test to check for ignorable assignment of SN students to teachers.
We find no evidence for systematic assignment of SN students to either classes or
teachers. Third, we conduct a simulation exercise in the spirit of \citet{carrell2010}.
We re-sample classes and thereby assign SN students randomly to classrooms, and test
whether the observed distribution of SN students differs from the simulated one. In
addition, we compute the interquartile range of the proportion SN students across classes
for each simulation, and compare these simulated interquartile ranges with the one we
observe in the data. Neither simulation procedure uncovers any worrisome pattern in the
assignment of SN students to classes. Fourth, we decompose the variation in the fraction
of SN peers across and within schools. To do so, we examine the residual variation in the
proportion of SN peers after partialling out the school-track-year fixed effects. We find
that the residual distribution in the proportion of  SN peers is consistent with
variation from a random process. Overall, the balancing tests we performed indicate that
the key identification assumption of (conditionally) ignorable assignment of SN students
to classes is plausible.
178
179 Our identification relies on variation between classes within school-track-years.
Although families can potentially choose their district of residence and thereby
influence schooling options for their children, possible selection into schools does not
confound our results.\footnote{Endogenous class formation could still occur if parents
request to transfer their children to a class with a lower SN fraction. To investigate
this potential threat, we acquired the official education statistics from the Swiss
Federal Statistical Office (SDL, \textit{Statistik der Lernenden} in German) for the
years 2012-2015. Importantly, the SDL has a classroom ID which allows us to reconstruct
the classes within each school-year-track. For the state of St.\ Gallen, we find that on
A:  ●  104k  xSource/manuscript_R1_v3.tex  ??:0 37%        LF UTF-8  LaTeX/FPS  ⊘ 1

different education tracks, and classes are strictly separated between tracks. Within
each school-track, classroom formation is conditionally ignorable with respect to SN
status, as students from different primary school districts are mixed and their SN status
is not observed by secondary school administrators.\footnote{Using PISA data from
secondary schools in Switzerland, \citet{Vardardottir2015} shows that track-by-school
fixed effects render peer group composition conditionally uncorrelated with a large set
of students' characteristics, while track fixed effects and school fixed effects do not.}
Neither primary schools nor the SPS share information with secondary schools for equity
reasons and to avoid stigma when transitioning between schools.
172
173 % NEU: BEu bitte lesen
174 Beyond this anecdotal evidence, we formally test the validity of the identification
strategy with four balancing tests, which are presented and discussed extensively in
Appendix B. First, we examine whether the proportion of SN peers predicts individual
baseline characteristics (gender, native speaker, and age). The aim of this test is to
detect potential selection into classrooms. We also conduct this test separately for SN
and non-SN students. None of the baseline characteristics are statistically significant
at conventional levels, either considered individually or jointly. Second, we regress the
indicator for SN status on class fixed effects, which should be jointly insignificant if
assignment to classroom is ignorable with respect to SN status \citep{chettyEtal2011}. We
also conduct this test to check for ignorable assignment of SN students to specific
teachers. We find no evidence for systematic assignment of SN students to either classes
or teachers. Third, we conduct a simulation exercise in the spirit of
\citet{carrell2010}. We re-sample classes, randomly assign SN students to classes, and
test whether the observed distribution of SN students differs from the simulated one. In
addition, we simulate random classroom assignment within schools, compute the
interquartile range of the proportion SN across classes, and compare the simulated
interquartile range with the one we observe in the data. Neither simulation procedure
uncovers any worrisome pattern in the assignment of SN students to classes. Fourth, we
decompose the variation in the fraction of SN peers across and within schools. To do so,
we examine the residual variation in the proportion of SN peer after partialling out the
school-track-year fixed effects. We find that the residual distribution in the proportion
of  SN peers is consistent with variation from a random process. Overall, the balancing
tests we performed indicate that the key identification assumption of (conditionally)
ignorable assignment of SN students to classes is plausible.
175
176 Our identification relies on variation between classes within school-track-years.
Although families can potentially choose their district of residence and thereby
influence schooling options for their children, possible selection into schools does not
confound our results.\footnote{Endogenous class formation could still occur if parents
request to transfer their children to a class with a lower SN fraction. To investigate
this potential threat, we acquired the official education statistics from the Swiss
Federal Statistical Office (SDL, \textit{Statistik der Lernenden} in German) for the
years 2012-2015. Importantly, the SDL has a classroom ID which allows us to reconstruct
the classes within each school-year-track. For the state of St.\ Gallen, we find that on
B:  ●  104k  xOld/manuscript_R1_v2.tex  ??:0 37%        LF UTF-8  LaTeX/FPS  ⊘ 11

Type ? for help
-- *Ediff Control Panel*   diff 4 of 15        Quick Help
Auto-refining is ON
```

Rstudio

RStudio

Haiyan-13.R

```
1  library(XML)
2  url <- "http://www.gdacs.org/Cyclones/report.aspx?eventid=41058&episodeid=28&
3  dat <- readHTMLTable(readLines(url), which=5)
4  dat$latlon <- dat[,8]
5  levels(dat$latlon) <- sapply(
6    strsplit(levels(dat[,8]), ",\n        "),
7         function(x) paste(x[2], x[1], sep=":")
8  )
9  dat$Category <- factor(dat$Category, levels=levels(dat$Category)[c(6,7,1:5)],
10                        ordered=TRUE)
11 dat$cat <- as.numeric(dat$Category)
12 dat$Gust_kmh <- dat[,6]
13 levels(dat$Gust_kmh) <- sapply(strsplit(levels(dat[,6]), "km"),
14                                function(x) gsub(" ", "",x[1]))
15 dat$Gust_kmh <- as.numeric(as.character(dat$Gust_kmh))
16 M <- gvisGeoChart(dat, "latlon", sizevar="cat",
17                   colorvar="Gust_kmh",
18                   options=list(region='035',
19                                backgroundColor="lightblue",
20                                datalessRegionColor="grey"))
21 plot(M)
22
```

Console ~/Downloads/IdR-dev/IdR/

```
> dat$latlon <- dat[,8]
> levels(dat$latlon) <- sapply(
+   strsplit(levels(dat[,8]), ",\n        "),
+         function(x) paste(x[2], x[1], sep=":")
+ )
> dat$Category <- factor(dat$Category, levels=levels(dat$Category)[c(6,7
,1:5)],
+                        ordered=TRUE)
> dat$cat <- as.numeric(dat$Category)
> dat$Gust_kmh <- dat[,6]
> levels(dat$Gust_kmh) <- sapply(strsplit(levels(dat[,6]), "km"),
+                                function(x) gsub(" ", "",x[1]))
> dat$Gust_kmh <- as.numeric(as.character(dat$Gust_kmh))
> M <- gvisGeoChart(dat, "latlon", sizevar="cat",
+                   colorvar="Gust_kmh",
+                   options=list(region='035',
+                                backgroundColor="lightblue",
+                                datalessRegionColor="grey"))
> plot(M)
>
```

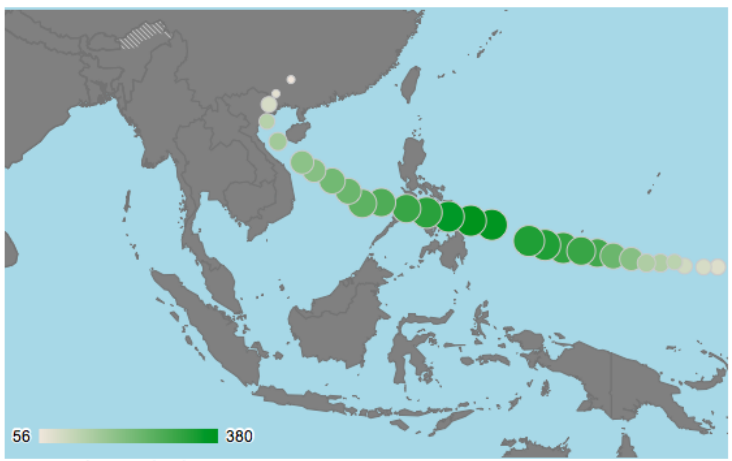
Environment | History | Presentation

Import Dataset | Clear | List

Global Environment

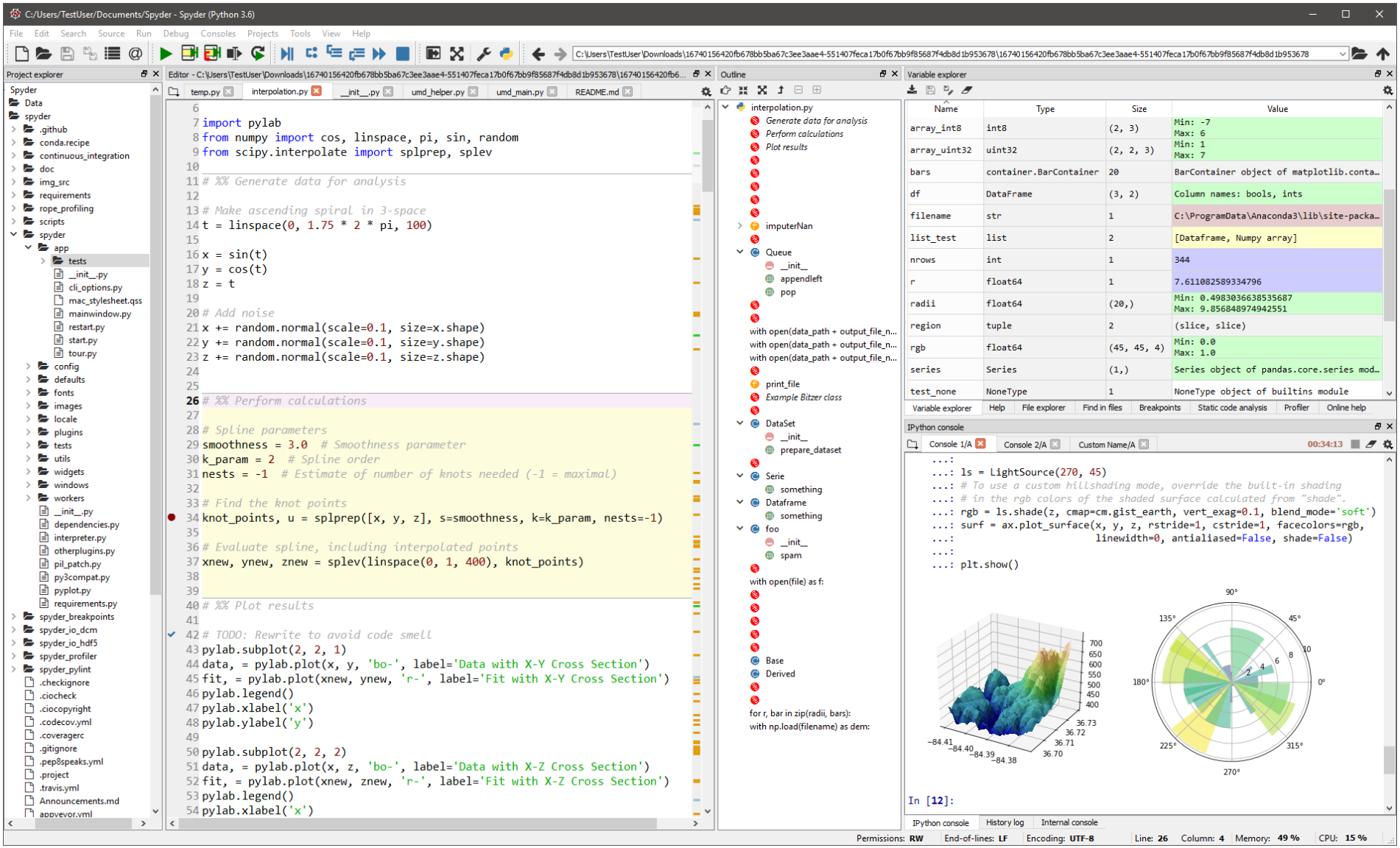
Data	
dat	33 obs. of 11 variables
Values	
M	List of 3
url	"http://www.gdacs.org/Cyclones/report.aspx?eventid=41058&e…

Files | Plots | Packages | Help | Viewer



Data: dat • Chart ID: GeoChartID926a2a743168
R version 3.0.2 (2013-09-25) • googleVis-0.4.7 • Google Terms of Use • Data Policy

Spyder


C:/Users/TestUser/Documents/Spyder - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Project explorer
Editor - C:\Users\TestUser\Downloads\16740156420fb678bb5ba67c3ee3aae4-551407feca17b0f67bb9f85687f4db8d1b953678\16740156420fb6...
temp.py
interpolation.py
__init__.py
umd_helper.py
umd_main.py
README.md
import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev
# %% Generate data for analysis
# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)
x = sin(t)
y = cos(t)
z = t
# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)
# %% Perform calculations
# Spline parameters
smoothness = 3.0 # Smoothness parameter
k_param = 2 # Spline order
nests = -1 # Estimate of number of knots needed (-1 = maximal)
# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)
Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)
# %% Plot results
# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
pylab.subplot(2, 2, 2)
data, = pylab.plot(x, z, 'bo-', label='Data with X-Z Cross Section')
fit, = pylab.plot(xnew, znew, 'r-', label='Fit with X-Z Cross Section')
pylab.legend()
pylab.xlabel('x')
Outline
Variable explorer
Name Type Size Value
array_int8 int8 (2, 3) Min: -7 Max: 6
array_uint32 uint32 (2, 2, 3) Min: 1 Max: 7
bars container.BarContainer 20 BarContainer object of matplotlib.conta...
df DataFrame (3, 2) Column names: bools, ints
filename str 1 C:\ProgramData\Anaconda3\lib\site-packa...
list_test list 2 [Dataframe, Numpy array]
nrows int 1 344
r float64 1 7.611082589334796
radii float64 (20,) Min: 0.4983036638535687 Max: 9.856848974942551
region tuple 2 (slice, slice)
rgb float64 (45, 45, 4) Min: 0.0 Max: 1.0
series Series (1,) Series object of pandas.core.series mod...
test_none NoneType 1 NoneType object of builtins module
Variable explorer Help File explorer Find in files Breakpoints Static code analysis Profiler Online help
IPython console
Console 1/A Console 2/A Custom Name/A 00:34:13
...: ls = LightSource(270, 45)
...: # To use a custom hillshading mode, override the built-in shading
...: # in the rgb colors of the shaded surface calculated from "shade".
...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
...:   linewidth=0, antialiased=False, shade=False)
...: plt.show()
In [12]:
IPython console History log Internal console
Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 26 Column: 4 Memory: 49 % CPU: 15 %

A note on text editing

- Text editors allow you to integrate your work and edit text efficiently.
- Sometimes IDEs with GUI may be more convenient.
- Features: Regex search and replace, diff, syntax checking, formatting, completion, documentation lookup, version control support, ...
- Consider using version control. Git is the predominant version control software used today. ProGit is a good and free resource.

Regular expressions

Regular expressions

- A regular expression is a character sequence that describes a set of strings.
- Regular expressions are constructed analogously to arithmetic expressions, by using various operators to combine smaller expressions.
- Usually used for find/replace operations on strings, or for validation.
- Pervasive in Unix text processing programs (`grep` was originally written by Ken Thompson).
- `grep`: **g**lobally search a **r**egular **e**xpression and **p**rint.
- Not limited to command line search tools.

Regular expressions

- *Pattern matching*: Find one of a specified set of strings in text.
- Examples:
  - Diagnoses in medical records.
  - Addresses or zip codes in concatenated admin records.
  - Sequences within a genome, e.g. a virus signature.
  - Validate data-entry fields (URL, date, email, credit card #).
  - Example using a regex tester.

Obligatory xkcd

Examples

AHVN 13: `756\.[0-9]{4}\.[0-9]{4}\.[0-9]{4}\.[0-9]{2}`
Matches: `756.1234.5678.90`
Does not match: `123.45.678.675`

US-SSN: `[0-9]{3}-[0-9]{2}-[0-9]{4}`
Matches: `166-11-4433`
Does not match: `11-55555555`

Email addresses: `[a-z]+@([a-z]+\.)+(ch|edu|com)`
Matches: `someone@unibas.ch`
Does not match: `someone@invalid.domain`

Screening job candidates

```
“ [First name]! and pre/2 [last name] w/7
   bush or gore or republican! or democrat! or charg!
   or accus! or criticiz! or blam! or defend! or iran contra
   or clinton or spotted owl or florida recount or sex!
   or controvers! or fraud! or investigat! or bankrupt!
   or layoff! or downsiz! or PNTR or NAFTA or outsourc!
   or indict! or enron or kerry or iraq or wmd! or arrest!
   or intox! or fired or racis! or intox! or slur!
   or controvers! or abortion! or gay! or homosexual!
   or gun! or firearm! ”
```

— LexisNexis search string used by Monica Goodling to illegally screen candidates for DOJ positions

LexisNexis™

http://www.justice.gov/oig/special/s0807/final.pdf

Regular expressions

- Characters in a regular expression are either regular characters (literal meaning) or metacharacters (special meaning).
- Generally, letters and numbers match themselves.
- Normally case sensitive, but can be set to ignore case.
- Careful with puntuation, most of it has special meanings.
- To match metacharacters literally, they need to be *escaped*, i.e. preceded by a backslash `\`.

Matching string literals

Regular expression	Input string
`input`	`This input string is short.`
`15`	`The due date is 15.12.`

Matching string literals

Regular expression	Input string
`input`	`This input string is short.`
`15`	`The due date is 15.12.`
`15.12`	`The due date is 15.12.`

Matching string literals

Regular expression	Input string
`input`	`This input string is short.`
`15`	`The due date is 15.12.`
`15.12`	`The due date is 15.12.`

but:

`15.12`	`The due date is 15712.`

match `.` literal:

`15\.12`	`The due date is 15712.`
`15\.12`	`The due date is 15.12.`

Regex basics

Operation	Regular expression	Input string
concatenation	`foobar`	`Matches foobar but not foo or bar.`
disjunction	`this|that`	`Matches this or that.`
closure	`like.* apples`	`I like apples, Peter likes apples.`
		`Mary also likesALKFHEDL apples.`
	`like. apples`	`Mary also likesALKFHEDL apples.`
parentheses	`(He|She) likes`	`She likes apples. He likes apples.`
	`(He|She).*(very )*much\.`	`She likes apples very very much.`

- More complicated patterns can be expressed via concatenation, disjunction, repetition and scope.
- Precedence in descending order.

Quantifiers

Character	Matches
`*`	0 or more instances of preceding char
`+`	1 or more instances of preceding char
`?`	0 or one instance of preceding char
`{m}`	exactly `m` instances of preceding char
`{m,n}`	`m` through `n` instances of preceding char
`{m,}`	`m` or more instances of preceding char
`{,n}`	up to `n` instances of preceding char
`?`	add to a quantifier to match ungreedy

- Quantifiers match greedily by default (i.e. the longest string possible).
  - Ex: `^begin.*end` will match 'begin bla bla end bla end'.
    `^begin.*?end` will match 'begin bla bla end bla end'.

Groups, ranges and character classes

Character	Matches	Example RE	Matches
`.`	Any character, except `\n`	`like.`	`likes like! like  like`
`(a|b)`	a or b	`(you|me)`	`you or me`
`[ab]`	Character range	`202[01]`	`2019 2020 2021 2022`
`[a-z]`	Character range	`[A-Z][a-z]*`	Capitalized words
`[0-9]`	Digit range	`20[0-9]{2}`	Years in the 21st century
`[^ab]`	Any character but (negation)	`20[^0][01]`	`2000 2010 2020 2025 2031`

- Quantifiers, ranges and other shortcuts improve expressiveness.
  Ex: `[A-E]+` is shorthand for `(A|B|C|D|E)(A|B|C|D|E)*`.
- More character classes (sometimes) available.
  Ex: `\w` for words (`[A-Za-z0-9_]`), or `\d` for digits (`[0-9]`), `\a`, `\s`, ...

Anchors and other special characters

Character	Matches	Example RE	Matches
`^`	Beginning of a string	`^New`	New research in this field
`$`	End of a string	`[A-Za-z]+!$`	A breakthrough! Finally!
`\n`	Newline		
`\t`	Tab		
…	…		

- Strings can stretch multiple lines.
- Character encodings can sometimes cause problems. Stick to UTF-8.

Regex syntaxes

- More elaborate regex syntaxes also support positive and negative lookahead/lookbehind, conditionals and group references.
  Ex: `^(?!.*word).*$` matches lines not containing a word.
- Different syntaxes (basic, extended, perl, vim, …) mostly similar with regard to basic features.
- Perl-compatible regular expressions (PCRE) is the de-facto standard.
- Most regex implementations feature switches to invert the search pattern, to ignore case, and more.

Example: Valid RFC-822 email addresses

http://www.ex-parrot.com/ pdw/Mail-RFC822-Address.html

Remarks

- Writing a regular expression is like writing a program.
- Requires understanding the programming model.
- Can be easier to write than read.
- Can be difficult to debug.

➥ Break up problems into smaller pieces. Try not to do everything in one large regex. Comment liberally.

➥ Regular expression tools help (e.g. regex101.com).

➥ Pin a cheatsheet to your office wall.

Remarks

- Regexes are a powerful tool.
- Easy to grasp, complex to master.

- Using them in applications can be complex and error-prone.
- Regular expressions are not parsers.

> *“Some people, when confronted with a problem, think ‘I know, I’ll use regular expressions.’ Now they have two problems.”*
>
> *Emacs newsgroup*

Next lecture: Representing text as data

References

References

- Chacon, S. and B. Straub (2014). *Pro Git*. Apress.
- Fitzgerald, M. (2012). *Introducing Regular Expressions: Unraveling Regular Expressions, Step-by-Step*. 1. ed. Beijing: O'Reilly.
- Matloff, N. (2011). *The Art of R Programming: A Tour of Statistical Software Design*. No Starch Press.
- Shotts, W. E. (2019). *The Linux Command Line: A Complete Introduction*. Second edition. San Francisco: No Starch Press.