Introduction to R

Mandy Vogel

University Leipzig

September 28, 2015

107

Overview

- The big plan
- R Intro
- Packages
- First Session
- Citation/License
- Objects
- Combining Data Frames
- Reading Data
- The Apply Family apply() tapply() Putting it all together

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together

today and next week

- Today
 - preliminaries
 - objects/data structures
 - combining data frames
 - indexing
 - reading data
 - apply family
 - reading the data files
- next time
 - data manipulation with dplyr
 - dates and times with lubridate
 - stringr/tidyr
 - graphics with ggplot2
 - classical tests
 - functions simulations understanding classical tests

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together



What's R?

- R is a high-level language and an environment for data analysis and graphics
- influenced by S (Becker, Chamber, Wilks) and Scheme (Sussman)
- and created by Ross Ihaka and Robert Gentleman at the university of Auckland
- R is free.
- R is open source.
- R is a dialect of S system.

What R can do...

R provides a wide variety of statistical and graphical techniques including

- linear and nonlinear modelling
- classical statistical tests
- time-series analysis
- classification
- clustering and many more

What R can do...

R provides a wide variety of statistical and graphical techniques including

- linear and nonlinear modelling
- classical statistical tests
- time-series analysis
- classification
- clustering and many more

R is easily extensible, can produce publication-quality graphs including mathematical symbols; dynamic and interactive graphics are available through additional packages.

Pros

- R is free and R is open source
- there is a lot of material and books available
- there is a lot of help on the web, including developers who are active in mailing lists
- most of your problems are already solved and with a high probability the solution is available from one of the repositories (as package)
- there are a lot of intuitive GUIs
- the language is easy to learn and also intuitive
- the graphics capabilities are impressive

R is Different

- R is a little different from other packages for statistical analysis
- these differences make R very powerful, but for new users they can sometimes be confusing that's normal!

Nothing is lost or hidden

- statistical packages provide canned procedures to address common statistical problems
- canned procedures are useful for routine analysis, but they are also limiting - you can only do what the programmer lets you do
- in R, the result of statistical calculation are always accessible, so
 - $\circ\;$ you can use them for further calculations
 - you can always see how calculations were done

Cons

- there is a LOT of help on the web
- with a high probability there is more than one solution for your problem
- there are a lot of intuitive GUIs so you have to decide what you want (so first you have to know what you want)
- the real power of R (i.e. high flexibility) is not entirely available through GUIs
- and therefore the learning curve can be lengthy in the beginning (but soon accelerating ;)

The number of analytics jobs for the more popular software (250 jobs or more, 2/2014).



¹²/107

O'Reilly Data Science Survey results for 2012 and 2013 combined.



107

Impact of data analysis software on academic publications as measured by hits on Google Scholar



⁺/107

Impact of data analysis software on academic publications as measured by hits on Google Scholar



Mandy Vogel mandy.vogel@googlemail.com

The best way to learn R is to use it!



Where can I get it?

For the basic installation CRAN is a good place to start

- CRAN stands for Comprehensive R Archive Network
- http://cran.r-project.org
 - Microsoft Windows: :: http://cran.r-project.org/bin/windows/base/
 - MacOS: :: http://cran.r-project.org/bin/macosx/
 - o Linux: :: http://cran.r-project.org/bin/linux/
- for mac and pc users: just download and install the precompiled binaries
- for ubuntu users: add deb http://ftp5.gwdg.de/pub/misc/cran/bin/linux/ubuntu vivid/

to

/etc/apt/sources.list;

detailed howto:

http://cran.r-project.org/bin/linux/ubuntu/

- https://cran.r-project.org/mirrors.html
- https://cran.r-project.org/mirmon_report.html

The R-Commander

The R commander, developed by John Fox is a complete GUI for R. It is implemented in the package Rcmdr:

- Rcmdr has a comprehensive menu, which includes data reading, summaries, statistical analyses, etc.
- When the menu is activated, the Rcmdr will generate an R script. This script can be used as a log for documentation or for self learning.
- It has excellent graphical tools.

The R-Commander

The southern the second s	coment Statistik Grafike	n Modelle Verteilunge	n Extrac I	ditte.	
Determatrix Prestige	Datenmatrix bearbeiten	Datenmatrix betrachten	Nodell:	«Kein aktuelles Modell	>
kristfeoster					
ibrary(pplot2) ata(Freatige)					
ibrary(relimp, pos-d)					
nowsstaturestige, piscen	ent="-10+200", fost-get	twendre togront), na	swidts-so,	. maximught=30, suppre	ess.xtt.warni
u santefearter					Defabl surfice
library(ggplot2)					
data(Freatige)					
> data(Freatige) > library(relimp, pom-4)					
<pre>data(Freatige) library(relimp, pos-d) showbata(Frentiar, plan</pre>	ments'-20+200', faster	entimete (* Looffont * Lo		0. maalumishtaj0. summ	
<pre>data(Freatige) library(relimp, pos-d) obseleta(Freatige, place)</pre>	ement="-20+220", fast-q	petRonde (*Logfont*) ,	maneidt b-S	0, maabeight-30, sogg	press.X11.wars
 data(Frentige) library(relimp, pos-d) showData(Frentige, place) 	ement="-20+210", foot-	petRonds (*Logfont *),	maanai di b-d	07, maaloright-30, sopy	press.X11.wars
 data(Freatige) library(relimp, pos-d) showData(Freatige, place) 	ements"-20+210", foots	petRende (*Logfont *L,	maasa i di b-si	00, mamberight=30, sopp	press.Xll.wars
data(Freatige) library(relimp, pos-d) showdata(Freatige, plac	ement-"-20+220", foot-q	yetRondr (*LoyPont*) ,	maanei.dt.h.=9	0), mamberight—30, nopy	press (X11.wars
- data(Freatige) - likrasy(relimp, pos=d) - skowData(Freatige, plac	emento"-20+220', fantoș	getRonde (*Logfoet *) ,	manna i dit hoof	80, maabelght-30, soqq	press All war
> data(Prestige) > likeasy(celiep, pas=d) > showSate(Prestige, plac	ement+"-20+250", feat-p	getRonde (*logfoot*),	nancidth-S	80, meshright—30, nopy	press All wars
> data(Prestige) > likrary(chimp, yos=d) > showdata(Prestige, plac	mmeryt+'-20+210', fant-q	petRonde("logTest"),	maane i dt h-sk	80, maabelght—30, sugg	press.Xll.war
> data(Prestige) > library(selimp, posed) > abselata(Prestige, plac	emerat="-20+210", feek-q	petRonde (*LogPoot*) ,	manori di bos	00, maabeight=30, sogg	press.X11.wars
<pre>> data(Prestige) > library(reliep, pan=d) > showdata(Prestigr, place)</pre>	mments"-20+220', (asta	petRomde (*SoyPoot*),	manos i. dt. hoof	07, mamberight=30, nogg	press.Xil.war
> data(Prestign) 11hrary(reling, point) > showData(Prestign, place) *showData(Prestign, place)	ement-'-20+220', feat-p	getRonde (*Logfont*) ,	maana i dit k8	00. mamberight—31. supp	press.311.war

Auswahl der aktiven Datenmatrix Aktualisiere aktive Datenmatrix Hilfe zur aktiven Datenmatrix (falls vorhanden) Variablen in aktiver Datenmatrix Fallbezeichnungen setzen ... Teilmenge der aktiven Datenmatrix ... Aggregate variables in active data set... Remove row(s) from active data set... Variablen übereinander plazieren ... Fälle mit fehlenden Werten entfernen ... Speichere aktive Datendatei ... Exportiere aktive Datenmatrix ...

🙉 🔿 💿 Prestige							
	education	income	women	prestige	census	type	
gov.administrators	13.11	12351	11,16	68,8	1113	prof	13
	12.26	25879	4.02	69.1	1130	prof	
	12.77	9271	15.70	63.4	1171	prof	111
	11.42	8865	9.11	56.8	1175	prof	
	14.62	8403	11.68	73.5	2111	prof	
	15.64	11030	5.13	77.6	2113	prof	
	15.09	8258	25.65	72.6	2133	prof	
	15.44	14163	2.69	78.1	2141	prof	
	14.52	11377	1.03	73.1	2143	prof	
mining.engineers	14.64	11023	0.94	68.8	2153	prof	m
surveyors	12.39	5902	1.91	62.0	2161	prof	
draughtamen	12.30	7059	7.83	60.0	2163	prof	
computer.programers	13.83	8425	15.33	53.8	2183	prof	
economists	14.44	8049	57.31	62.2	2311	prof	
psychologists	14.36	7405	48.28	74.9	2315	prof	
	14.21	6336	54.77	55.1	2331	prof	
	15,77	19263	5,13	82.3	2343	prof	
	14.15	6112	77.10	58.1	2351	prof	
	15.22	9593	34.89	58.3	2391	prof	
	14.50	4686	4.14	72.8	2511	prof	
	15.97	12480	19.59	84.6	2711	prof	
	13.62	5648	83.78	59.6	2731	prof	
	15.08	8034	46.80	66.1	2733	prof	
	15.96	25308	10.56	87.2	3111	prof	
	15.94	14558	4.32	66.7	3115	prof	
	14.71	17498	6.91	68.4	3117	prof	
	12.46	4614	96.12	64.7	3131	prof	
	9.45	3485	76.14	34.9	3135	bc	
	13.62	5092	82.66	72.1	3137	prof	
pharmacists	15.21	10432	24.71	69.3	3151	prof	12

Farbpalette ... Index-Plot ... Histogramm ... "Stamm und Blatt" Abbildung Boxplot ... Quantile-comparison plot... Streudiagramm Matrix ... Liniengrafik ... XY conditioning plot... Plot für arithmetische Mittel ... Strip chart... Balkendiagramm ... Kreisdiagramm ... Speichere Abbildung in Datei

The R-Commander

To install Rcmdr go to Packages \rightarrow Install package(s) (or simply type install.packages("Rcmdr")), then choose a CRAN mirror close to you, than OK. A window with a list of packages will pop-up, on this list choose Rcmdr and OK. A bundle of packages will be automatically installed. To run the "R Commander" GUI type at the prompt line:

> library(Rcmdr)

This will start a GUI similar to other statistical software. Therefore, any typical process, like read data, produce plots, make statistical analyses, etc. will be made by clicking the appropriate menu.



Getting R-Studio

RStudio is a free and open source integrated development environment (IDE) for R. You can run it on your desktop (Windows, Mac, or Linux) or even over the web using RStudio Server. Available at http://rstudio.org/ (install R first)

Johans 1 \$2.00								
	and + 0,000m2 + 0,000	ODredeskellend i 🖉 sullpalen. End i	\$3980°	a 🗅	Entro	net thisy		
87	5 - 4 000 + 0		JAN S	CUR-	a 8	Prest Menty & One 10		
1000					4 Cat	i Endostani-		
1 1231 01	1.40							
a weather pro-	10.000							
A Distant	Include Access Includes 1	a single formation mater for an	And an all states of	of K and				
deciments.	or more details on unless 8 Ref.	dan an diffe-Visablen relate	100.					
2								
8. Wen yes cl.	de the "Notif" bottom a decay	nt will be precided that includes	with centert, is will	10.94				
oract of a	y waxabid it sole churks v255	the docenit, for car whed as it is	ode churk Ulie th's					
2								
M(0								
11 summy/or								
10.1								
20 C								
- C					100	Box Balance Box House		
18 The case of the	etel data. for exercise							
17					Q.764	Nobr 🍳 Della 🔌 Resure 🖉 Nov-		
Mr. TTr. edu					C23	Dris .		
18 (\$16)(011)						- faite	509	Ridfiel
S						diamon and	110	lie 14 2015 2 30 5
8					12.2			
12 NOS 0411	e soo i kkoa paraebi ka	NORE 21 TH COR CURL 10 PREMIT	bound on an en	201.114		Roday	12.8	36 IL 201, 840
111 B Flatter				I Building 7		2000/EET	155 B	NY 16 21 26 20 20 20 20 20 20 20 20 20 20 20 20 20
Jamatia - C. Al-				-0		000		
Genete - C.A. Millio Partner	84 142136 2012			-0	0.0	Dakumede		
General III A attick Platboo besile Gettinets	164 1023-06 2012 184 1083-25 2013			-0		Dáunede Dovrbab		
Gemain - ()) attick Plations Jopin Continents Vysiler Special	9.4 8 02.8 36 2.91 1.2 9.4 8 00.8 25 3.01 5.4 16.7 8 00.8 20 3.21 5.3			-0		Dakumede Downbath		
Gemaie ad Lisz Praetaan Jestin Gentleent Vysiler teperfal Jet 10	364 8 823 36 231 12 124 8 883 25 38 56 367 8 883 28 31 13 324 4 32 88 68 23	2 2 30 1 4 1 1 1 9 7 42 1 4 1 4 9 7 42 1 4 1 4 9 7 42 1 4 1 1		-0		own Disumetrie Counteals emiCSIC5		
benah	364 8 473 8 36 230 13 1364 8 473 8 36 230 13 167 8 468 23 340 14 167 8 468 23 340 14 164 8 167 16 460 12 164 8 157 16 440 14	17.00 0 0 1 0 17.00 0 0 1 4 17.42 0 0 1 4 19.42 1 1 4 1 19.52 1 1 4 1		-0		olar Odunete Oserbati enfCIIS eungiscietze	110	Nov 25, 2024, 5137
Genete - A atilik Flatboo beste Getteets vyster beetal lat 19 eek Ovic geta fantta geta fantta	B.4 1 07.8 2.01 1.01 1.0.4 4 60.0 2.55 3.04 5.66 1.4.7 4 60.0 2.55 3.04 5.66 1.4.7 4 60.0 2.55 3.04 5.66 2.4.4 4 7.97 5.66 1.20 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 1.01 3.04 3.04 1.01 3.04 1.01 3.04 3	8 27.00 8 8 3 8 8 27.40 8 9 3 4 8 27.42 8 9 3 4 9 37.42 8 9 3 4 9 39.42 1 4 1 9 39.52 1 1 4 1 6 39.00 1 1 4 1		-D		own Dakunste Dowlaats enfots sangeschektep Water	110	Nov 25, 2024, 5537 Nov 26, 2024, 5537
Genete - A attick Platboo broth Ontherb hyster Deptal lat 19 eets Dvik gets Ionita sets Ionita sets Ionita sets Ionita	8.4 8 87.3 86.4 1.0 1.9.4 8 80.8 2.5 3.0 1.6 1.9.4 8 80.8 2.5 3.0 1.6 1.9 8 8.2 3.21 3.0 1.6 3.14 8 8.6 9.2 3.21 1.6 3.14 4 1.7 5 4.0 1.6 3.14 4 1.7 5 4.0 1.6 3.14 1.1 1.6 6.2 1.6 1.6 1.2 3.14 1.1 1.6 6.2 1.6 </td <td>37.00 8 3 4 87.26 6 3 4 87.26 8 3 4 93.62 1 4 1 93.62 1 4 1 63.93 1 4 1 63.93 1 4 1 63.93 1 4 1 63.93 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1</td> <td></td> <td>-0</td> <td></td> <td>oler Däunete Overbah eriCHS exopic.tiettap macag</td> <td>110</td> <td>NV11,2131,537 NV21,2131,238</td>	37.00 8 3 4 87.26 6 3 4 87.26 8 3 4 93.62 1 4 1 93.62 1 4 1 63.93 1 4 1 63.93 1 4 1 63.93 1 4 1 63.93 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1 63.94 1 4 1		-0		oler Däunete Overbah eriCHS exopic.tiettap macag	110	NV11,2131,537 NV21,2131,238
Genete - A attic Ration Justi Gotiano hysio spotal lat 59 mii Gvic sota Grufa pota Grufa pota Grufa sota Grufa sota Grufa sota Grufa	8.4 8 87.8 2.8 1.8 8.4 8 8.6 2.5 3.8 5.4 1.8 8 8.25 3.8 5.4 5.6 1.7 8 8.3 3.9 3.9 3.8 3.8 3.8 3.8 3.8 3.8 3.9 3.9 3.4 3.8 3.9 <t< td=""><td>8 27.00 8 3 1 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 1 4 1 8 36.72 1 4 1 5 35.42 1 4 1 6 35.40 1 4 1 6 35.42 1 4 1 6 35.42 1 4 1 6 35.42 1 4 1 8 36.72 1 4 1 9 36.72 8 4 3 9 36.73 8 6 3 9 37.98 6 3 2</td><td></td><td>-0</td><td></td><td>omi Dalamen Oselosto entoto entoto interg interg</td><td>1100 124 139</td><td>8v23,2234,537 8v23,2234,238 8v23,2234,239</td></t<>	8 27.00 8 3 1 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 8 3 4 8 27.40 1 4 1 8 36.72 1 4 1 5 35.42 1 4 1 6 35.40 1 4 1 6 35.42 1 4 1 6 35.42 1 4 1 6 35.42 1 4 1 8 36.72 1 4 1 9 36.72 8 4 3 9 36.73 8 6 3 9 37.98 6 3 2		-0		omi Dalamen Oselosto entoto entoto interg interg	1100 124 139	8v23,2234,537 8v23,2234,238 8v23,2234,239
Genete	B.4 8 87.3 87.4 1.0 <th1.0< th=""> <th1.0< th=""> <th1.0< th=""></th1.0<></th1.0<></th1.0<>	8 7.70 8 8 3 4 8 7.24 8 3 4 1 8 7.74 8 6 3 4 1 9.74 1 1 4 1 1 4 1 9.74 1 1 4 1 1 4 1 9.75 1 1 4 1 1 4 1 9.76 1 1 4 1		-0		perior Dakumeter Constate entrototo entrototo interrigi interrigi interrigi	11 GI 12 2 13 5	No. 21, 2124, 537 No. 21, 2124, 228 No. 21, 2124, 228
Genete - Catalog	36.4 8 87.3 96.6 2.01 1.07 10.4 6 60.6 2.05 3.01 5.0 10.7 6 60.2 2.0 1.07 5.0 10.7 6 60.2 2.0 1.07 5.0 1.07 10.4 4 1.07 26 60.2 1.07 5.0 1.07 10.4 4 1.07 26 60.2 1.07 5.0 1.07 10.4 4 1.07 26 60.2 1.07 <td>8 7.00 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 8 3 4 8 7.20 8 8 3 4 9.30 1 4 1 4 1 9.30 1 1 4 1 1 9.30 1 4 2 1 4 1 9.30 1 4 2 3 1 <t< td=""><td></td><td>-0</td><td></td><td>omin Osensete Osensete entgeschetzp interry interry interry</td><td>11 01 12 1 13 3</td><td>No. 21, 2124, 557 No. 21, 2124, 213 No. 21, 2124, 223 No. 21, 2124, 223</td></t<></td>	8 7.00 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 6 3 4 8 7.20 8 8 3 4 8 7.20 8 8 3 4 9.30 1 4 1 4 1 9.30 1 1 4 1 1 9.30 1 4 2 1 4 1 9.30 1 4 2 3 1 <t< td=""><td></td><td>-0</td><td></td><td>omin Osensete Osensete entgeschetzp interry interry interry</td><td>11 01 12 1 13 3</td><td>No. 21, 2124, 557 No. 21, 2124, 213 No. 21, 2124, 223 No. 21, 2124, 223</td></t<>		-0		omin Osensete Osensete entgeschetzp interry interry interry	11 01 12 1 13 3	No. 21, 2124, 557 No. 21, 2124, 213 No. 21, 2124, 223 No. 21, 2124, 223
Generale	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	8 2 8 4 3 4 8 2 2 8 4 3 4 8 2 2 8 4 3 4 8 2 2 8 4 3 4 8 2 2 8 4 3 4 5 3 2 1 4 2 5 5 3 1 1 4 2 5 6 3 1 4 2 5 5 6 3 1 4 2 5 5 6 3 1 4 2 2 2 6 3 1 4 2 2 2 6 3 2 2 2 2 2 6 7 6 3 2 2 2 6 7 6 4		-0		o over O desensete I Desensete I enklostis I enklostis I enklostis I enklostististe I enklostistege	8.8 KB 12.8 13.9 22.96	Nov 23, 2014, 5137 Nov 28, 2014, 2014 Nov 28, 2014, 2019 Nov 28, 2014, 2019
Genete	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		-0		ome Osamente Downbath enf0005 enf0005 Informy informy informy informy informy informatory Network by Network by	11 G 12 2 13 3 13 4 13 6	Nov 21, 2024, 5537 Nov 24, 2024, 2028 Nov 24, 2024, 2025 Nov 24, 2024, 2021 Jan 24, 2024, 2021
Generale - A att tax Flamboo Incoln Gertinets System Special Bet Sti enk Drin Spect Santta Bet Sinne Star Sinklar Star Sinklar Star Sinklar Internet Star Sinklar Internet Star Sinklar	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$			-D		o over Downeatr ereCCS compact down interry interry interry interformatory interformatory interformatory interformatory Versene y	1140 103 153 1246 1340	8w 21, 2014, 553 8w 21, 2014, 2016 8w 21, 2014, 2016 8w 21, 2014, 223 9w 21, 2014, 223 9w 21, 2012, 223
Genate < // att the Kartako Inchi Gottawo Nyike taya fal lat 59 web Dviz gota Santta keta Dviz apita Farafor atta: Farafor lat 30 f sotta: Farafor sotta: Farafor lat 30 f sotta: Farafor sotta: Farafor lat 30 f sotta: Farafor sotta: Farafor	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$		1		o over O causande enkCRES O canapacaloritap enhang I mbang i m	11.00 133 133 1340 1340 1140	Nov 24, 2024, 5537 Nov 24, 2024, 2028 Nov 24, 2024, 2029 Nov 24, 2024, 2029 Nov 24, 2024, 2029 Jun 14, 2022, 2029 Jun 14, 2022, 2029
Generale	$\begin{array}{c c c c c c c c c c c c c c c c c c c $	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		C		o over O commente D'oministeri emilicatio indicational in	11.00 103 153 12.00 13.00 13.00 13.00 13.00 13.00	Nov 11, 2014, 553 Nov 24, 2014, 2018 Nov 24, 2014, 2019 Nov 24, 2014, 2021 Jan 24, 2014, 2021 Jan 24, 2014, 2021 Jan 24, 2014, 2021 Jan 24, 2014, 2021
Generale	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		10		per Deneta Deneta Deneta Deneta enception indexes indexes indexes indexes basereng Vacareng Vacareng Vacareng Vacareng Vacareng d	11.00 133 153 153 150 1100 1100 1100 1100	Nov 21, 2024, 5537 Nov 21, 2024, 2537 Nov 21, 2024, 2525 Nov 22, 2024, 2525 Jan 34, 2022, 2539 Jan 34, 2022, 2539 Jan 34, 2022, 2539 Jan 34, 2022, 2539

Getting Deducer

Deducer is designed to be a free easy-to-use alternative to proprietary data analysis software such as SPSS and Minitab. It has a menu system to perform common data manipulation and analysis tasks, and an excel-like spreadsheet in which to view and edit data frames. Available at http://www.deducer.org; for Windows there is a all-in-one installer (incl. R)



Deducer Ubuntu

There were some problems with color setting in the plot builder when they are defined via the GUI. They are fixed in the recent version (0.6-3)

- if you haven't installed R yet, first install r-base-dev r-recommended (sudo apt-get install r-base-dev r-recommended)
- Open R and at the prompt enter: install.packages(c("JGR", "Deducer"))
- Run JGR() to open JGR, and library(Deducer) to load Deducer



Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together



Packages

The capabilities of R are extended through user-created packages, which allow specialized statistical techniques, graphical devices, import/export capabilities, reporting tools, etc.

These packages are developed primarily in R, and sometimes in Java, C and Fortran. A core set of packages is included with the installation of R, 4300 (as of March 2011) with more than 7100 + 1000 (BioC) + 1100 (BioC Annotation/Exp.) (as of September 2015) available at the Comprehensive R Archive Network (CRAN), Bioconductor, and other repositories.



R taskviews

- you can google your problem or you use http://www.rseek.org/ or http://www.rdocumentation.org/ instead of www.google.com
- http://cran.r-project.org/web/views/
- before you install a new package: help.search() allows for searching the help system for documentation matching a given character string in the (file) name, alias, title, concept or keyword entries (or any combination thereof), using either fuzzy matching or regular expression matching.(installed help system)
- there are many blogs or forums, a very popular is http://stackoverflow.com/ or http://stackexchange.com/ (they are helpful for all other statistical packages as well)

Packages

- An R installation contains a library of packages. Some of these packages are part of the basic installation. These packages have the recommended status.
- others can be downloaded from CRAN or BioConductor or from other repositories
- A package is loaded into R using the library() or the require() command. For example to load the survival package you should enter
 - > library(survival)
- you need to reload a package when you start a new R session

Getting Packages

- You can download a package from CRAN and install it by using the package menu.
- Another effective way to download and install a package is by command line. For example the following line install the R commander package with all its dependencies:
 - > install.packages("Rcmdr", dependencies=TRUE)
- bioconductor packages have their own installation routine, which is documented on the website http://www.bioconductor.org/

Software Container

- if you know about docker: there are R and BioConductor Docker container available:
 - o https://github.com/rocker-org/rstudio-daily
 - o https://www.bioconductor.org/help/docker/

Table of Contents I

The big pla R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together

First Session I

- start RStudio
- choose your working directory (via a menu or by typing setwd('/your/Rworkdirectory/'))
- R works fundamentally by the question-and-answer model: you enter a line with a command and press Enter (←). Then the program dœs something and prints or stores the results. Then it asks for more input. When R is ready for input, it prints out its prompt, a ">". if you see a "+" R waits for you to end to line; with ESC you can go back to ">"
- It is possible to use R as a text-only application, and also in batch mode (Rscript skript.r arg1 arg2)

The Workspace

- During a session you create a workspace. The workspace contains all variables created during the session
- for example typing

> x <- rnorm(100, mean=2, sd=4)

creates a variable \mathbf{x} containing a vector with 100 random numbers normal distributed with mean 2 and standard deviation 4

• to see the contents of a variable just type its name

Showing an Object

- more sophisticated R objects have print methods which do not show you the object itself but a kind of summary
 - > xx <- density(x)

> xx

```
Call:
density.default(x = x)
```

```
Data: x (100 obs.); Bandwidth 'bw' = 1.465
```

Х		У	
Min.	:-9.711	Min.	:0.00034
1st Qu.	:-2.976	1st Qu.	:0.005759
Median	: 3.759	Median	:0.030555
Mean	: 3.759	Mean	:0.037080
3rd Qu.	:10.495	3rd Qu.	:0.068267
Max.	:17.230	Max.	:0.088606

First Plot

- $\bullet\,$ To plot the values contained in x type
 - > plot(x)



107

ls()/objects()

- All variables, functions and diverse objects can be seen by typing ls() and the newer, more verbose version of it objects() function. Thus in our example we will have > ls()
 - [1] "x" "xx"
- in R studio you see the content of the workspace in the Environment tab
library()/require()

- we have seen the use of library() to load a package
- typing library() without argument gives you a list of installed packages per installation path
- a more detailed list of installed packages including path as well but also a lot of additional information can be achieved by installed.packages()

Quitting

- quitting R is done with the q() function

> q()

at the command prompt. You will be asked to save your "workspace image".

• if you save the work space, all R objects can be reloaded in a new session, but be careful. It is recommended to rather save data explicitly

Help

- Entering the command
 - > help.start()

at the command line, will launch an extensive online help that can be read using a Web browser such as Firefox or Internet Explorer. Another way to access to these "help" pages is by the menu bar on Windows. Notice that the HTML version of the help system has a very useful "Search Engine and Keywords".

typing ?command gives the help for a specific command

First Session



http://127.0.0.1:10137/doc/manual/R-exts.html

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects Combining Data Frame: Reading Data The Apply Family apply() tapply() Putting it all together

Citation

Input > citation()

To cite R in publications use:

```
R Development Core Team (2012). R: A language and environment for
statistical computing. R Foundation for Statistical Computing,
Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.
```

```
A BibTeX entry for LaTeX users is
```

```
@Manual,
  title = R: A Language and Environment for Statistical Computing,
  author = R Development Core Team,
  organization = R Foundation for Statistical Computing,
  address = Vienna, Austria,
  year = 2012,
  note = ISBN 3-900051-07-0,
  url = http://www.R-project.org/,
```

We have invested a lot of time and effort in creating R, please cite it when using it for data analysis. See also 'citation("pkgname")' for citing R packages.

Licence

Licence

R is mainly distributed under the terms of the GNU General Public License, either Version 2, June 1991 or Version 3, June 2007. Core Bioconductor packages are typically licensed under Artistic-2.0. You get detailed information with: license(), RShowDoc("COPYING"),

packageDescription("packagename")\$License

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames Reading Data The Apply Family apply() tapply() Putting it all together



Data Structures

- R's base data structures can be organized by
 - their dimensionality and
 - $\circ\;$ whether they are homogeneous or heterogeneous

	Homogeneous	Heterogeneous			
1d	Atomic Vector	List			
2d	Matrix	Data frame			
nd	Array				





- given an object, the best way to understand its structure it's the str() command
- str() is short for structure and gives a compact, human readable description of any R data structure

str()

Input/Output

<pre>> example(scale_colour_brewer) > str(dsamp) 'data.frame': 1000 obs. of 10 variables: \$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74 \$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74</pre>	
 > str(dsamp) 'data.frame': 1000 obs. of 10 variables: \$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74	
<pre>> str(dsamp) 'data.frame': 1000 obs. of 10 variables: \$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74</pre>	
'data.frame': 1000 obs. of 10 variables: \$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74	
\$ carat : num 2.02 1.01 0.52 1.57 0.82 1.17 0.78 1.05 0.28 0.74	
<u> </u>	
\$ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<: 5 4 4 6 1 4 6 1 2	2.
<pre>\$ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<: 3 3 2 4 3 2 4 3</pre>	54
\$ depth : num 61.1 63 59.7 62.5 61.6 61.2 62.7 60.5 62.4 61.8	
\$ table : num 58 57 56 58 57 58 58 60 56 58	
\$ price : int 18236 5555 1042 9847 4135 5595 2646 5762 828 3180	
\$ x : num 8.07 6.37 5.28 7.5 6.02 6.84 5.89 6.62 4.2 5.79	
\$ y : num 8.16 6.4 5.31 7.42 6.06 6.86 5.82 6.53 4.16 5.82	
\$ z : num 4.96 4.02 3.16 4.66 3.72 4.19 3.68 3.98 2.61 3.59	

107

str()

Input/Output

```
> str(d)
List of 9
$ data :'data.frame': 1000 obs. of 10 variables:
  ..$ carat : num [1:1000] 2.02 1.01 0.52 1.57 0.82 1.17 0.78
$ mapping :List of 3
  ..$ colour: symbol clarity
  ..$ x : symbol <u>carat</u>
  ..$ y : symbol price
$ coordinates:List of 1
  ..$ limits:List of 2
  ....$ x: NULL
  ....$ y: NULL
  ..- attr(*, "class")= chr [1:2] "cartesian" "coord"
```

Vectors

- R's basic data structure
- three common properties:
 - o type(typeof())
 - o lenth (length())
 - attributes (attributes() optional)
- there are four common types of atomic vectors:
 - logical
 - integer
 - double (numeric)
 - character
- usually created using c() (for concatenate/combine)
- flat structure

Types/Cœrcion

- while combining two or more vectors of different types, the will be corced to the most flexible type
- types from least to most flexible are: logical, integer, double, character

Cœrcion - Exercises

• Test your understanding of vector cœrcion rules by predicting the output of the following uses of c():



Why is 1 == "1" true? Why is -1 < FALSE true? Why is "one" < 2 false?

•

• should be self explanatory

[′]107

attributes

The most important are:

- names
- dimensions
- class

107

factors

A factor is a vector that can contain only predefined values (categorical variable).

- built on top of integer values with the following attributes:
 - class: factor
 - levels: set of allowed values
 - labels

factors - Exercise

• What happens to a factor when you modify its levels? (Hint: use summary() and as.numeric())

Input

- > f1 <- factor(letters)</pre>
- > levels(f1) <- rev(levels(f1))</pre>
- > f2 <- rev(factor(letters))</pre>
- > f3 <- factor(letters, levels = rev(letters))</pre>
- > f4 <- c(f2,f3)

- lists are different from atomic vectors because their elements can be of any type
- constructing lists is done by list()
- they can be recursive, i.e. lists can contain lists can contain lists etc
- they can be combined by c()

Data Frames

- most common way to store data in R
- under the hood a data frame is a list of equal-length vectors
- so it is a 2-dimensional structure
- the length() of a data frame is the length of the underlying list (i.e. the number of columns)
- ncol(), nrow(), dim(), names(), rownames(), colnames()
- constructing is done by data.frame()

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together

rbind()

 rbind() can be used to combine two dataframes (or matrices) in the sense of adding rows, the column names and types must be the same for the two objects

Input/Output

- > x <- data.frame(id=1:3,score=rnorm(3))</pre>
- > y <- data.frame(id=13:15,score=rnorm(3))</pre>
- > rbind(x,y)

id score

- 1 1 0.71121163
- 2 2 -0.62973249
- 3 3 1.17737595
- 4 13 -0.45074940
- 5 14 -0.01044197
- 6 15 -1.05217176

cbind()

 cbind() can be used to combine two dataframes (or matrices) in the sense of adding columns, the number of rows must be the same for the two objects

In	Input/Output								
>	> cbind(x,y)								
	id	score1	score2	score3					
1	1	0.11440705	0.14536778	-1.1773241					
2	2	-1.62862651	0.02020604	0.5686415					
3	3	0.05335811	0.25462270	0.8844987					
4	4	-0.19931734	0.15625511	0.9287316					
5	5	-1.15217836	-1.79804503	-0.7550234					

• it is not recommended to use cbind() to combining data frames



- merge() is the command of choice for merging or joining data frames
- it is the equivalent of join in sql
- there are four cases
 - inner join
 - \circ left outer join
 - right outer join
 - \circ full outer join



merge() ||

In	iput	/Out	:put
>	(d1	<- ċ	<pre>lata.frame(id=LETTERS[c(1,2,3)],day1=sample(10,3</pre>
	id (day1	
1	A	3	
2	В	4	
3	С	5	
>	(d2	<- ċ	lata.frame(id=LETTERS[c(1,3,5,6)],day2=sample(10
	id d	day2	
1	А	7	
2	C	10	
3	E	3	
4	F	6	

⁶¹/107

inner join

• inner join means: keep only the cases present in both of the data frames

Input/Output						
>	> merge(d1,d2)					
	id	day1	day2			
1	A	3	7			
2	С	5	10			

left outer join

 left outer join means: keep all cases of the left data frame no matter if they are present in the right data frame (all.x=T)

Input/Output

>	<pre>merge(d1,d2,all.x = T)</pre>								
	id	day1	day2						
1	А	3	7						
2	В	4	NA						
3	С	5	10						

right outer join

 right outer join means: keep all cases of the right data frame no matter if they are present in the left data frame (all.y=T)

In	Input/Output								
>	> merge(d1,d2,all.y = T)								
	id	day1	day2						
1	A	3	7						
2	С	5	10						
3	Е	NA	3						
4	F	NA	6						
								IL	

full outer join

 full outer join means: keep all cases of both data frames (all=T)

In	Input/Output									
>	> merge(d1,d2,all = T)									
	id	day1	day2							
1	A	3	7							
2	В	4	NA							
3	С	5	10							
4	Е	NA	3							
5	F	NA	6							





- if not stated otherwise R uses the intersect of the names of both data frames, in our case only id
- you can specify these columns directly by by=c("colname1", "colname2") if the columns are named identical or
- using

```
by.x=c("colname1.x","colname2.x"),
by.y=c("colname1.y","colname2.y") if they have
different names in the data frames
```





- now read in the file personendaten.txt using the appropriate command
- join the demographics with our prel data frame (even though it does not make sense now)

Reduce()

- is a higher order function (functional)
- Reduce() uses a binary function (like rbind() or merge()) to combine successively the elements of a given list
- it can be used if you have not only two but many data frames



• first we make up 4 artifical data frames



Reduce()

Input/Output

```
> (d1 <- data.frame(id=LETTERS[c(1,2,3)],day1=sample(10,3)))</pre>
  id day1
   A
  (d2 <- data.frame(id=LETTERS[c(1,3,5,6)],day2=sample(10,4)))</pre>
  id day2
   A
4
      <- data.frame(id=LETTERS[c(2,4:6)],day3=sample(10,4)))
  id day3
4
      <- data.frame(id=LETTERS[c(1:5)],day4=sample(10,5)))
  id day4
```



Reduce()

now we use Reduce() in combination with merge()

Input/Output

> Reduce(merge,list(d1,d2,d3,d4))
[1] id day1 day2 day3 day4
<0 Zeilen> (oder row.names mit Länge 0)

- and what we get is an empty data frame
- well this isn't exactly what we wanted, so why?
- it is because the default behavior of merge() is set all=F, so we get only complete lines which is in this case - none
- so we have to define a wrapper function which only change this argument to all=T
Reduce()

• now we use Reduce() in combination with merge()

lr	ipu	t/Ou	tput			
>	Rec	duce(1	functi	lon(x	,y) {	<pre>merge(x,y, all=T) },</pre>
+			list(d	d1,d2	,d3,d4	L))
	id	day1	day2	day3	day4	
1	А	3	8	NA	2	
2	В	1	NA	8	7	
3	С	7	2	NA	8	
4	Е	NA	5	4	1	
5	F	NA	3	10	NA	
6	D	NA	NA	3	9	

which is exactly what we want

Reduce()

• a second example in combination with rbind()

Input/Output

```
> d4$day <- names(d4)[2]</pre>
> names(d4)[2] <- "score"</pre>
> Reduce(function(x,y) { y$day <- names(y)[2]</p>
                           names(y)[2] <- "score"</pre>
+
                           rbind(x,y) } ,
          list(d1, d2, d3), init = d4)
   id score day
 A 2 day4
2 B 7 day4
3 C 8 day4
 D 9 day4
```

which is exactly what we want

Excercises - Merging

- in the rstuff.r file you find the code to create three data frames: m1, m2 and subjdata
- combine these three data sets using merge()

Indexing with Positive Integers

- there are circumstances where we want to select only some of the elements of a vector/array/dataframe/list
- this selection is done using subscripts (also known as indices)
- subscripts have square brackets [2] while functions have round brackets (2)
- Subscripts on vectors, matrices, arrays and dataframes have one set of square brackets [6], [3,4] or [2,3,2,1]
- while subscripts on lists have double square brackets [[2]] or [[i,j]]
- when a subscript appears as a blank it is understood to mean all of thus
 - [,4] means all rows in column 4 of an object
 - [2,] means all columns in row 2 of an object.

Indexing with Positive Integers

• A vector of positive integers as index:The index vector can be of any length and the result is of the same length as the index vector. For example,

Input/Output

```
> letters[1:3]
[1] "a" "b" "c"
> letters[c(1:3,1:3)]
[1] "a" "b" "c" "a" "b" "c"
>
```

 A logical vector as index: Values corresponding to T values in the index vector are selected and those corresponding to F or NA are omitted. For example,

Input/Output

```
> x<-c(1,2,3,NA)
> x[!is.na(x)]
[1] 1 2 2
```

```
[1] 1 2 3
```

creates a vector without missing values. Also

```
> x[is.na(x)] <- 0
```

```
> x
[1] 1 2 3 0
```

replaces the missing value by zeros.

A common operation is to select rows or columns of data frame that meet some criteria. For example, to select those rows of painters data frame with Colour \geq 17:

Input/Output						
> library(MASS)						
> attach(painters)					
> painter	s[Colour >=:	17,]				
	Composition	Drawing	Colour	Expression	School	
Bassano	6	8	17	0	D	
Giorgione	8	9	18	4	D	
Pordenone	8	14	17	5	D	

We may want to select on more than one criterion. We can combine logical indices by the 'and', 'or' and 'not' operators &, | and !. For example,

Input/Output > painters [Colour >=17 & Composition > 10, c(1,2,3)] Composition Drawing Colour Titian 12 18 15 Rembrandt 6 17 18 13 17 Rubens 17 Van Dyck 15 10

List of Logical Operations

Operation	Description
!	logical NOT
&	logical AND
	logical OR
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	logical equals (double =)
! =	not equal
&&	AND with IF
	OR with IF
<pre>xor(x,y)</pre>	exclusive OR
isTRUE(x)	an abbreviation of identical(TRUE,x)



If we want to select a subgroup, for example those with schools A, B, and D. We can generate a logical vector using the %in% operator as follows:

Input/Output						
> painters[School	%in%	c("A","C",	'D"),	.]		
Da Udine	10	8	16	3	А	
Da Vinci	15	16	4	14	А	
Del Piombo	8	13	16	7	А	

Indexing

A vector character strings with variable names can be used to extract those variables relevant for analysis. This is very useful when we have a large number of variables and we need to work with a few ones. For example,

Input/Output

> names(pai	nters)				
[1] "Compos	sition"	"Drawing"	"Colour"	"Expression"	"School
> painters[[1:3,c("	Drawing","	Expression	on")]	
	Drawing	Expression	n		
Da Udine	8		3		
Da Vinci	16	1	.4		
Del Piombo	13		7		

Indexing with Characters

• a vector of character strings could a index on a vector when the vector has names:

```
Input/Output
> x <- c(1:3,NA)
> names(x)<-letters[1:4]
> x
a b c d
1 2 3 NA
> x[c("a","c")]
a c
1 3
```



Trimming Vectors Using Negative Indices

- an extremely useful facility is to use negative indices to drop terms from a vector
- suppose we wanted a new vector, z, to contain everything but the first element of x

```
Input/Output
> x<- c(5,8,6,7,1,5,3)
> (z <- x[-1])
[1] 8 6 7 1 5 3</pre>
```



Indexing - Exercises

First try to understand the following commands:

Input
> x <- c(2, 7, 0, 9, 10, 23, 11, 4, 7, 8, 6, 0)
> x[4]
> x[3:5]
> x[c(1, 5, 8)]
> x[x > 10]
> x[(1:6) * 2]
> x[x == 0] <- 1
> x
> ifelse(round($x/2$) == $x/2$, "even", "odd")

Indexing - Exercises

Now try the following:

- 1. Display every third element in x
- 2. Display elements that are less than 10, but greater than 4
- 3. Modify the vector x, replacing by 10 all values that are greater than 10
- 4. Modify the vector x, multiplying by 2 all elements that are smaller than 5
- Create a new vector y with elements 0,1,0,1, . . . (12 elements) and a vector z that equals x when y=0 and 3x when y=1. (You can do it using ifelse, but there are other possibilities)

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply() Putting it all togethe

Reading Data

The most convenient way of reading data into R is via the function called read.table(). It requires that the data is in "ASCII format", or a "flat file" as created with Windows' NotePad or any plain-text editor. The result of read.table() is a data frame.

It is expected that each line of the data file corresponds to a subject information, that the variables are separated by blanks or any other separator symbol (e.g., ",", ";"). The first line of the file can contain a header (header=T) giving the names of the variables, which is highly recommended!

read.table()

As an example we read in the data contained in the file fishercats.txt

Input/Output

> read.table("week1/data/fishercats.txt", + sep=" ",header=T) Sex Bwt Hwt 1 F 2.0 7.0 2 F 2.0 7.4 3 F 2.0 9.5 4 F 2.1 7.2 5 F 2.1 7.3

These data correspond to the heart and body weights of samples of male and female cats (R. A. Fisher, 1947).

read.table()

The first argument corresponds to the data file, the second to the fields separator and the third header=T specifies that the first line is a header with variable names. Important: the character variables will be automatically read as factors. There is a variant for reading data from an url:

Input/Output

- > winer <- read.table(</pre>
- + "http://socserv.socsci.mcmaster.ca/jfox/Courses/R/ICPSR/Wine
- + header=T)

read.table()

There are other variants of read.table function alike :

- read.csv() this function assumes that fields are separated by a comma instead of whites spaces
- read.csv2() this function assumes that the separate symbol is the semicolon, but use a comma as the decimal point (some programs, e.g., Microsoft Excel, generate this format when running in European systems)
- the function scan() is a powerful, but less friendly, way to read data in R; you may need it, if you want to read files with different numbers ov values per line

Reading data from the clipboard

With the function read.delim() or also read.table() it is possible to read data directly from the clipboard. For example mark and copy some columns from an Excel spreadsheet and transfer this content to an R by

Input/Output

> mydata <- read.delim("clipboard",na.strings=".")
> str(mydata) # structure of the data

The Data Editor

To interactively edit a data frame in R you can use the edit function. For example:

Input/Output

- > data(airquality)
- > aq <-edit(airquality)</pre>

This brings up a spreadsheet-like editor with a column for each variable in the data frame. See help(airquality) for the contents of this data set. The function edit() leaves the original data frame unchanged, the changed data frame is assigned to aq. The function fix(x) invokes the function edit(x) on x and assign the new (edited) version of x to x



Reading Data from Other Programs

You can always use the export function from other (statistical) software to export data from other statistical systems to a tab or comma-delimited file and use the read.table(). However, R has some direct methods.

The foreign package is one of the "recommended" packages in R. It contains routines to read files from SPSS (.sav format), SAS (export libraries), Epilnfo (.rec), Stata, Minitab, and some S-PLUS version 3 dump files. For example

Input/Output

- > library(foreign)
- > mydata <- read.spss("test.sav", to.data.frame=T)</pre>

read the test.sav SPSS data set and convert it to a data.frame.

Reading Data from Excel Files

Input/Output

- > library(XLConnect)
- > setwd("/media/TRANSCEND/mpicbs/data/")
- > my.wb <- loadWorkbook("Duncan.xls")</pre>
- > sheets <- getSheets(my.wb)</pre>
- > content <- readWorksheet(my.wb, sheet=1)</pre>
- > head(content)

ColO	type	income	education	prestige
accountant	prof	62	86	82
pilot	prof	72	76	83
architect	prof	75	92	90
author	prof	55	90	76
chemist	prof	64	86	90
minister	prof	21	84	87
	Col0 accountant pilot architect author chemist minister	ColO type accountant prof pilot prof architect prof author prof chemist prof minister prof	Col0typeincomeaccountantprof62pilotprof72architectprof75authorprof55chemistprof64ministerprof21	Col0typeincomeeducationaccountantprof6286pilotprof7276architectprof7592authorprof5590chemistprof6486ministerprof2184

Reading Data from Excel Files

- whereas XLConnect is the most sophisticated R package to read and write Excel files it depends on java and is therefore a bit clumsy
- the relatively new package readx1 does not depend on java nor on perl
- up to now two commands: excelsheets(), read_excel()

Input/Output

>	<pre>> require(readx1)</pre>					
La	ade nötiges	Paket	c: read	cl		
>	x <- read_e	excel	("week1,	/data/Dunca	an.xls")	
>	head(x)					
	NA	type	income	education	prestige	
1	accountant	prof	62	86	82	
2	pilot	prof	72	76	83	
3	architect	prof	75	92	90	
4	author	prof	55	90	76	

Reading Data from Excel Files

If someone is really fond of Excel, RExcel (http://rcom.univie.ac.at/download.html) is really worth the effort. There is also a function reading MSAccess files (mdb.get() from the Hmisc package)

Something on Connections

The function read.table() opens a connection to a file, read the file, and close the connection. However, for data stored in databases, there exists a number of interface packages on CRAN.

The RODBC package can set up ODBC connections to data stored by common applications including Excel and Access (for Excel and Access RODBC dœsn't work on Unix but it is great for data base connections). There are also more general ways to build connections to data bases.

For up-to-date information on these matters, consult the "R Data Import/Export" manual that comes with the system.

Table of Contents I

The big plan

R Intro

Packages

First Session

Citation/License

Objects

Combining Data Frames

Reading Data

The Apply Family apply() tapply()

Putting it all together

Implicit Loops

A common application of loops is to apply a function to each element of a set of values and collect the results in a single structure.

In R this is done by the functions (but there is of course also for()):

- lapply()
- sapply()
- apply()
- tapply()

lapply()

Input/Output

> lapply(mtcars,mean)
\$mpg
[1] 20.09062

\$cyl [1] 6.1875

\$disp [1] 230.7219

\$hp [1] 146.6875

107

lapply()

Input/Outp	but				
> sapply(mto	cars,mean)				
mpg	cyl	disp	hp	drat	
20.090625	6.187500	230.721875	146.687500	3.596563	3.217
vs	am	gear	carb		
0.437500	0.406250	3.687500	2.812500		

⁰²/107

apply()

 apply() this function can be applied to an array. Its argument is the array, the second the dimension/s where we want to apply a function and the third is the function. For example

Input/Output

- > x <- 1:12
- > dim(x) < -c(2,2,3)
- > apply(x,3,quantile) #calculate the quantiles
 - [,1] [,2] [,3] #for each 2x2 matrix
- 0% 1.00 5.00 9.00 25% 1.75 5.75 9.75 50% 2.50 6.50 10.50 75% 3.25 7.25 11.25

100% 4.00 8.00 12.00

tapply()

 The function tapply() allows you to create tables (hence the "t") of the value of a function on subgroups defined by its second argument, which can be a factor or a list of factors. For example in the quine data frame, we can summarize Days classify by Eth and Lrn as follows:

Input/Output



Table of Contents I

- The big plan
- R Intro
- Packages
- First Session
- Citation/License
- Objects
- Combining Data Frames
- Reading Data
- The Apply Family apply() tapply()
- Putting it all together

Reading the Data

Input/Output
<pre>> tmp <- lapply(files,function(filename){</pre>
+ xx <- readLines(filename)
+ d1 <- read.table(text = xx[4:length(xx)],fill = T,hea
+ d2 <- read.table(text = xx[1:2],fill = T,header=T)
+ d1\$subject <- rownames(d2)
+ d1\$timepoint <- d2\$subject
+ d1\$date <- d2\$timepoint
+ d1\$time <- d2\$date
+ d1\$no.trials <- d2\$no_trials
+ return(d1)
+ })
>
> system.time(result <- Reduce(rbind,tmp))
Fehler in match.names(clabs, names(xi)) :
Namen passen nicht zu den vorhergehenden Namen
Timing stopped at: 0.01 0 0.01

06

Reading the Data

Input/Output						
<pre>> tmp <- lapply(files,function(filename){</pre>						
xx <- readLines(filename)						
+ d1 <- read.table(text = xx[4:length(xx)],fill = T,head.table(text = xx[4:length(xx)],fill = T,head.table(tex						
+ names(d1)[3] <- "trial"						
+ d2 <- read.table(text = xx[1:2],fill = T,header=T)						
+ d1\$subject <- rownames(d2)						
+ d1\$timepoint <- d2\$subject						
+ d1\$date <- d2\$timepoint						
+ d1\$time <- d2\$date						
+ d1\$no.trials <- d2\$no_trials						
+ return(d1)						
+ })						
>						
> system.time(result <- Reduce(rbind,tmp))						
User System verstrichen						
57.812 0.017 57.765						

107

