

O'REILLY®

Wydanie II

Język R Receptury

Analiza danych, statystyka
i przetwarzanie grafiki



Helion 

J.D. Long
Paul Teetor

Tytuł oryginału: R Cookbook: Proven Recipes for Data Analysis, Statistics, and Graphics, 2nd Edition

Tłumaczenie: Krzysztof Sawka

ISBN: 978-83-283-6288-8

© 2020 Helion SA

Authorized Polish translation of the English edition of R Cookbook, 2nd Edition
ISBN 9781492040682 © 2019 J.D. Long and Paul Teetor

This translation is published and sold by permission of O'Reilly Media, Inc.,
which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means,
electronic or mechanical, including photocopying, recording or by any information storage retrieval system,
without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej
publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną,
fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje
naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich
właścicieli.

Autor oraz Helion SA dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne
i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym
ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Helion SA nie ponoszą również
żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
<ftp://ftp.helion.pl/przyklady/jezrr2.zip>

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/jezrr2>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- [Lubię to!](#) » [Nasza społeczność](#)

Spis treści

Witaj w książce *Język R. Receptury. Analiza danych, statystyka i przetwarzanie grafiki. Wydanie II* 11

1. Pierwsze kroki i uzyskiwanie pomocy 17

- 1.1. Pobranie i instalacja R 18
- 1.2. Instalacja środowiska RStudio 20
- 1.3. Uruchamianie środowiska RStudio 21
- 1.4. Wprowadzanie poleceń 23
- 1.5. Wyjście ze środowiska RStudio 24
- 1.6. Przerwanie realizacji kodu R 26
- 1.7. Przeglądanie dołączonej dokumentacji 27
- 1.8. Uzyskiwanie pomocy na temat funkcji 28
- 1.9. Wyszukiwanie dodatkowej dokumentacji 30
- 1.10. Uzyskiwanie pomocy na temat pakietu 31
- 1.11. Wyszukiwanie pomocy w internecie 32
- 1.12. Wyszukiwanie przydatnych funkcji i pakietów 35
- 1.13. Przeszukiwanie list dyskusyjnych 36
- 1.14. Przesyłanie pytań do serwisu Stack Overflow lub innego 37

2. Garść podstaw 41

- 2.1. Wyświetlanie interesujących nas danych na ekranie 41
- 2.2. Wyznaczanie zmiennych 43
- 2.3. Tworzenie listy zmiennych 44
- 2.4. Usuwanie zmiennych 46
- 2.5. Tworzenie wektorów 47
- 2.6. Obliczanie podstawowych statystyk 49
- 2.7. Tworzenie sekwencji 51
- 2.8. Porównywanie wektorów 52
- 2.9. Wybieranie elementów wektora 54
- 2.10. Wykonywanie obliczeń wektorowych 57
- 2.11. Ustalanie pierwszeństwa operatorów 59

2.12. Osiąganie więcej przy mniejszej liczbie znaków	61
2.13. Tworzenie strumienia wywołań funkcji	62
2.14. Unikanie najpowszechniejszych pomyłek	65
3. Korzystanie z oprogramowania	71
3.1. Sprawdzanie i wyznaczanie katalogu roboczego	71
3.2. Tworzenie nowego projektu RStudio	72
3.3. Zapisywanie przestrzeni roboczej	74
3.4. Przeglądanie historii wpisanych poleceń	75
3.5. Zapisywanie wyniku wcześniejszego polecenia	76
3.6. Wyświetlanie załadowanych pakietów poprzez ścieżkę wyszukiwania	77
3.7. Przeglądanie listy zainstalowanych pakietów	79
3.8. Uzyskiwanie dostępu do funkcji zawartych w pakiecie	80
3.9. Uzyskiwanie dostępu do wbudowanych zestawów danych	81
3.10. Instalowanie pakietów z repozytorium CRAN	82
3.11. Instalowanie pakietu z serwisu GitHub	84
3.12. Wyznaczanie lub zmiana domyślnego serwera CRAN	85
3.13. Uruchamianie skryptu	86
3.14. Uruchamianie skryptu wsadowego	87
3.15. Wyszukiwanie katalogu domowego R	89
3.16. Personalizowanie rozruchu R	91
3.17. Korzystanie z R i RStudio w chmurze	94
4. Dane wejściowe i wyjściowe	97
4.1. Wprowadzanie danych za pomocą klawiatury	97
4.2. Wyświetlanie mniejszej (lub większej) liczby znaków	98
4.3. Przekierowywanie wyników do pliku	100
4.4. Wyświetlanie listy plików	101
4.5. Problem z otwieraniem pliku w systemie Windows	103
4.6. Odczytywanie rekordów o stałej szerokości	104
4.7. Odczytywanie plików danych tabelarycznych	107
4.8. Odczytywanie plików CSV	110
4.9. Zapisywanie danych w pliku CSV	112
4.10. Odczytywanie danych tabelarycznych lub CSV z internetu	113
4.11. Odczytywanie danych z arkuszy Excel	114
4.12. Zapisywanie ramki danych w pliku Excel	116
4.13. Odczytywanie danych z pliku SAS	118
4.14. Odczytywanie danych z tabel HTML	120
4.15. Odczytywanie plików o skomplikowanej strukturze	122
4.16. Odczyt baz danych MySQL	126
4.17. Uzyskiwanie dostępu do bazy danych za pomocą pakietu dbplyr	129
4.18. Zapisywanie i transportowanie obiektów	131

5. Struktury danych	135
5.1. Dodawanie danych do wektora	142
5.2. Wstawianie danych do wektora	144
5.3. Reguła zawijania	144
5.4. Tworzenie wektora czynnikowego (zmiennej kategoryjnej)	146
5.5. Łączenie wielu wektorów w jeden wektor i wektor czynnikowy	147
5.6. Tworzenie listy	149
5.7. Wybieranie elementów listy za pomocą ich pozycji	150
5.8. Wybieranie elementów listy po nazwie	152
5.9. Tworzenie listy asocjacyjnej nazwa/wartość	153
5.10. Usuwanie elementu z listy	155
5.11. Spłaszczanie listy do postaci wektora	156
5.12. Usuwanie elementów o wartości NULL z listy	157
5.13. Warunkowe usuwanie elementów listy	158
5.14. Inicjowanie macierzy	159
5.15. Wykonywanie operacji macierzowych	161
5.16. Nadawanie nazw opisowych rzędom i kolumnom macierzy	162
5.17. Wybór jednego rzędu/kolumny macierzy	163
5.18. Inicjowanie ramki danych z danymi kolumny	164
5.19. Inicjowanie ramki danych z danymi rzędu	165
5.20. Dołączanie rzędów do ramki danych	168
5.21. Wybór kolumn ramki danych za pomocą pozycji	170
5.22. Wybór kolumn ramki danych za pomocą nazwy	174
5.23. Zmianianie nazw kolumn w ramce danych	175
5.24. Usuwanie wartości NA z ramki danych	176
5.25. Wykluczanie kolumn za pomocą nazwy	177
5.26. Łączenie dwóch ramek danych	178
5.27. Scalanie kolumn dwóch ramek danych	179
5.28. Przekształcanie jednej wartości atomowej w inną	181
5.29. Przekształcanie jednego ustrukturyzowanego typu danych w inny	183
6. Przekształcenia danych	187
6.1. Stosowanie funkcji wobec każdego elementu listy	187
6.2. Stosowanie funkcji wobec każdego rzędu ramki danych	190
6.3. Stosowanie funkcji wobec każdego rzędu macierzy	191
6.4. Stosowanie funkcji wobec każdej kolumny	192
6.5. Stosowanie funkcji wobec wektorów równoległych lub list	194
6.6. Stosowanie funkcji wobec grup danych	196
6.7. Tworzenie nowej kolumny na podstawie jakiegoś warunku	197

7. Łańcuchy znaków i daty	199
7.1. Uzyskiwanie długości łańcucha znaków	201
7.2. Łączenie łańcuchów znaków	202
7.3. Wydobywanie fragmentów łańcuchów znaków	203
7.4. Rozdzielanie łańcucha znaków zgodnie z rozgranicznikiem	204
7.5. Zastępowanie fragmentów łańcuchów znaków	205
7.6. Tworzenie wszystkich kombinacji par łańcuchów znaków	206
7.7. Uzyskiwanie bieżącej daty	208
7.8. Przekształcanie łańcucha znaków w obiekt Date	208
7.9. Przekształcanie obiektu Date w łańcuch znaków	209
7.10. Przekształcanie roku, miesiąca i dnia w obiekt Date	210
7.11. Uzyskiwanie daty juliańskiej	211
7.12. Wydobywanie elementów składowych daty	212
7.13. Tworzenie sekwencji dat	213
8. Prawdopodobieństwo	215
8.1. Wyznaczanie liczby kombinacji	217
8.2. Generowanie kombinacji	218
8.3. Generowanie liczb losowych	219
8.4. Generowanie odtwarzalnych liczb losowych	220
8.5. Generowanie próby losowej	222
8.6. Generowanie sekwencji losowych	223
8.7. Losowe permutacje wektora	224
8.8. Obliczanie prawdopodobieństwa rozkładów dyskretnych	225
8.9. Obliczanie prawdopodobieństwa rozkładów ciągłych	226
8.10. Przekształcanie prawdopodobieństw w kwantyle	228
8.11. Tworzenie wykresu funkcji gęstości	229
9. Statystyka ogólna	235
9.1. Podsumowywanie danych	237
9.2. Obliczanie częstości względnych	239
9.3. Zestawianie wektorów czynnikowych w tabeli i tworzenie tablic wielodzzielczych	240
9.4. Sprawdzanie niezależności zmiennych kategoryjnych	241
9.5. Obliczanie kwantylów (i kwartyłów) zestawu danych	242
9.6. Uzyskiwanie odwrotności kwantylu	243
9.7. Normalizowanie danych	244
9.8. Testowanie średniej próby (test t)	244
9.9. Kształtowanie przedziału ufności dla średniej	246
9.10. Kształtowanie przedziału ufności dla mediany	247
9.11. Testowanie proporcji próby	248
9.12. Kształtowanie przedziału ufności dla proporcji	249

9.13. Testowanie pod względem rozkładu normalnego	250
9.14. Testowanie przebiegów	251
9.15. Porównywanie średnich dwóch prób	252
9.16. Nieparametryczne porównywanie położenia dwóch prób	254
9.17. Testowanie korelacji pod względem istotności	256
9.18. Testowanie grup pod względem równych proporcji	257
9.19. Porównywanie parami średnich poszczególnych grup	259
9.20. Testowanie dwóch prób w kontekście tego samego rozkładu	260
10. Grafika	263
10.1. Tworzenie wykresu punktowego	267
10.2. Wstawianie tytułu i etykiet	267
10.3. Dodawanie (lub usuwanie) siatki	269
10.4. Stosowanie motywu wobec wykresu ggplot	272
10.5. Tworzenie wielogrupowego wykresu punktowego	277
10.6. Dodawanie (lub usuwanie) legendy	278
10.7. Rysowanie linii regresji na wykresie punktowym	282
10.8. Tworzenie wykresów par zmiennych	285
10.9. Tworzenie wykresów punktowych dla poszczególnych grup danych	287
10.10. Tworzenie wykresu kolumnowego	289
10.11. Umieszczanie przedziałów ufności na wykresie kolumnowym	292
10.12. Wprowadzanie kolorów na wykresie kolumnowym	295
10.13. Rysowanie linii łączącej pary punktów x i y	297
10.14. Zmiana rodzaju, szerokości i koloru linii	297
10.15. Tworzenie wykresu zawierającego wiele zestawów danych	301
10.16. Dodawanie linii pionowych lub poziomych	302
10.17. Tworzenie wykresu pudełkowego	304
10.18. Tworzenie po jednym wykresie pudełkowym na każdy poziom wektora czynnikowego	306
10.19. Tworzenie histogramu	308
10.20. Dodawanie oszacowania gęstości do histogramu	310
10.21. Tworzenie standardowego wykresu kwantyl-kwantyl	311
10.22. Tworzenie innych wykresów kwantyl-kwantyl	314
10.23. Rysowanie zmiennej w różnych kolorach	316
10.24. Tworzenie wykresu funkcji	319
10.25. Wyświetlanie wielu wykresów na jednej stronie	321
10.26. Zapisywanie wykresu do pliku	324
11. Regresja liniowa i analiza ANOVA	327
11.1. Przeprowadzanie prostej analizy liniowej	329
11.2. Przeprowadzanie wielorakiej regresji liniowej	331
11.3. Uzyskiwanie statystyk regresji	332

11.4. Omówienie podsumowania regresji	335
11.5. Przeprowadzanie regresji liniowej bez użycia punktu przecięcia z osią współrzędnych	338
11.6. Przeprowadzanie regresji wyłącznie przy użyciu zmiennych ściśle skorelowanych ze zmienną objaśnianą	339
11.7. Przeprowadzanie regresji liniowej z członami interakcyjnymi	342
11.8. Wybór najlepszych zmiennych regresji	344
11.9. Przeprowadzanie regresji na podzbiorze danych	349
11.10. Korzystanie ze wzorów w równaniu regresji	350
11.11. Przeprowadzanie regresji względem wielomianu	351
11.12. Regresja względem przekształconych danych	353
11.13. Wyszukiwanie najlepszego przekształcenia potęgowego (procedura Boxa-Coxa)	355
11.14. Kształtowanie przedziałów ufności dla współczynników regresji	359
11.15. Tworzenie wykresu elementów resztowych regresji	360
11.16. Diagnostowanie regresji liniowej	361
11.17. Wykrywanie najbardziej znaczących obserwacji	364
11.18. Testowanie wartości resztowych pod względem autokorelacji (test Durбина-Watsona)	366
11.19. Przewidywanie nowych wartości	367
11.20. Kształtowanie przedziałów predykcji	368
11.21. Przeprowadzanie jednoczynnikowej analizy ANOVA	369
11.22. Tworzenie wykresu interakcji	371
11.23. Wyszukiwanie różnic pomiędzy średnimi grup	372
11.24. Przeprowadzanie odpornej analizy ANOVA (test Kruskala-Wallis)	375
11.25. Porównywanie modeli za pomocą analizy ANOVA	376
12. Przydatne sztuczki	379
12.1. Zagląwanie do danych	379
12.2. Wyświetlanie rezultatu przypisania	380
12.3. Sumowanie rzędów lub kolumn	382
12.4. Wyświetlanie danych w kolumnach	383
12.5. Grupowanie danych w przedziały	384
12.6. Określanie położenia danej wartości	385
12.7. Wybieranie co n-tego elementu wektora	385
12.8. Określanie minimów i maksimów	386
12.9. Tworzenie wszystkich kombinacji kilku zmiennych	388
12.10. Spłaszczanie ramki danych	389
12.11. Sortowanie ramki danych	390
12.12. Usuwanie atrybutów ze zmiennej	391
12.13. Odkrywanie struktury obiektu	392

12.14. Obliczanie czasu potrzebnego na realizację kodu	395
12.15. Wstrzymywanie ostrzeżeń i komunikatów o błędach	396
12.16. Pobieranie argumentów funkcji z listy	397
12.17. Definiowanie własnych operatorów binarnych	399
12.18. Blokowanie komunikatu rozruchowego	401
12.19. Przeglądanie i wyznaczanie zmiennych środowiskowych	401
12.20. Używanie sekcji kodu	402
12.21. Równoległe przetwarzanie kodu R na komputerze lokalnym	403
12.22. Równoległe przetwarzanie kodu R w sposób zdalny	406
13. Zaawansowane obliczenia numeryczne i statystyczne	411
13.1. Minimalizowanie lub maksymalizowanie funkcji jednoparametrowej	411
13.2. Minimalizowanie lub maksymalizowanie funkcji wieloparametrowej	412
13.3. Obliczanie wartości własnych i wektorów własnych	414
13.4. Przeprowadzanie analizy głównych składowych	415
13.5. Przeprowadzanie prostej regresji ortogonalnej	416
13.6. Wyszukiwanie skupień w danych	418
13.7. Przewidywanie zmiennej binarnej (regresja logistyczna)	421
13.8. Metody samowsporne	423
13.9. Analiza czynnikowa	425
14. Analiza szeregów czasowych	431
14.1. Reprezentowanie danych szeregów czasowych	433
14.2. Tworzenie wykresów danych szeregów czasowych	436
14.3. Wydobywanie najstarszych lub najnowszych obserwacji	437
14.4. Tworzenie podzbiorów z szeregów czasowych	439
14.5. Scalanie kilku szeregów czasowych	441
14.6. Uzupełnianie brakujących obserwacji w szeregach czasowych	443
14.7. Opóźnianie lub przyspieszanie szeregu czasowego	446
14.8. Obliczanie kolejnych różnic	447
14.9. Wykonywanie obliczeń na szeregu czasowym	449
14.10. Obliczanie średniej kroczącej	450
14.11. Stosowanie funkcji przy uwzględnieniu okresu kalendarzowego	451
14.12. Stosowanie funkcji rozwijającej	453
14.13. Tworzenie wykresu funkcji autokorelacji	455
14.14. Testowanie szeregów czasowych pod kątem autokorelacji	456
14.15. Tworzenie wykresu funkcji autokorelacji cząstkowej	457
14.16. Wyszukiwanie korelacji opóźnionych pomiędzy dwoma szeregami czasowymi	459
14.17. Usuwanie trendów z szeregów czasowych	461
14.18. Dopasowywanie modelu ARIMA	463
14.19. Usuwanie nieistotnych współczynników z modelu ARIMA	466

14.20. Diagnozowanie modelu ARIMA	468
14.21. Uzyskiwanie prognoz z modelu ARIMA	470
14.22. Tworzenie wykresu prognoz	471
14.23. Sprawdzanie występowania zjawiska równania do średniej w szeregu czasowym	472
14.24. Wygładzanie szeregu czasowego	475
15. Elementy prostego programowania	479
15.1. Wybór pomiędzy dwiema alternatywnymi opcjami: if/else	480
15.2. Przetwarzanie w pętli	482
15.3. Definiowanie funkcji	483
15.4. Tworzenie zmiennej lokalnej	485
15.5. Wybór pomiędzy wieloma alternatywnymi ścieżkami: funkcja switch	485
15.6. Definiowanie wartości domyślnych parametrów	487
15.7. Sygnalizowanie błędów	488
15.8. Ochrona przed błędami	489
15.9. Tworzenie funkcji anonimowej	490
15.10. Tworzenie zbioru funkcji wielokrotnego użytku	491
15.11. Automatyczne formatowanie kodu	492
16. Środowisko R Markdown i publikowanie	495
16.1. Tworzenie nowego dokumentu	496
16.2. Dodawanie tytułu, danych autora i daty	498
16.3. Formatowanie dokumentu tekstowego	499
16.4. Wstawianie nagłówków dokumentu	500
16.5. Wstawianie listy	500
16.6. Prezentowanie wyników kodu R	502
16.7. Kontrolowanie wyświetlania kodu i wyników	503
16.8. Wstawianie wykresu	505
16.9. Wstawianie tabeli	507
16.10. Wstawianie wygenerowanej tabeli	509
16.11. Wstawianie równań matematycznych	512
16.12. Generowanie wyniku w formacie HTML	513
16.13. Generowanie wyniku w formacie PDF	514
16.14. Generowanie wyników w formacie Microsoft Word	516
16.15. Generowanie pliku prezentacji	522
16.16. Tworzenie parametryzowanego raportu	524
16.17. Organizowanie pracy z dokumentami R Markdown	527

Pierwsze kroki i uzyskiwanie pomocy

W tym rozdziale wyznaczamy podwaliny pod pozostałą część książki. Wyjaśniamy w nim sposób pobrania, instalacji i uruchamiania R.

Co jednak ważniejsze, nauczysz się tu uzyskiwać odpowiedzi na pytania. Społeczność R gwarantuje obfitość dokumentacji i pełne wsparcie. Nie jesteś sam. Poniżej przedstawiamy najpopularniejsze źródła pomocy:

Dokumentacja zainstalowana lokalnie

Wraz z instalacją R na komputerze uzyskujesz dostęp do rozbudowanej dokumentacji. Możesz przeglądać dokumentację lokalną (receptura 1.7) i przeszukiwać ją (receptura 1.9). Jest zdumiewające, jak często poszukujemy odpowiedzi w internecie, gdy okazuje się, że od samego początku można ją znaleźć w zainstalowanej dokumentacji.

Widoki zadań

Widok zadań (<https://cran.r-project.org/web/views/>; ang. *task view*) opisuje pakiety specyficzne dla określonego działu statystyki, np. ekonometrii, obrazowania medycznego, psychometrii czy statystyki przestrzennej. Każdy widok zadań został utworzony i jest utrzymywany przez specjalistę w danej dziedzinie. Istnieje ponad 35 widoków zadań, zatem prawdopodobnie przynajmniej jeden mieści się w obszarze Twojego zainteresowania. Zalecamy, aby każda osoba początkująca znalazła i przeczytała przynajmniej jeden widok zadań, aby poznać zakres możliwości R (receptura 1.12).

Dokumentacja pakietu

Większość pakietów zawiera przydatną dokumentację. Wiele z nich zawiera również przeglądy i samouczki, zwane w społeczności R **ulotkami** (ang. *vignettes*). Dokumentacja jest przechowywana z pakietami w takich repozytoriach jak CRAN (<https://cran.r-project.org/>) i zostaje automatycznie zainstalowana na komputerze wraz z danym pakietem.

Strony z pytaniami i odpowiedziami (Q&A)

Na stronie z pytaniami i odpowiedziami (ang. *Questions and Answers — Q&A*) każdy może umieścić pytanie, a kompetentne osoby mają możliwość udzielenia odpowiedzi. Czytelnicy oceniają odpowiedzi, zatem po pewnym czasie wyłaniają się najlepsze z nich. Wszystkie te informacje zostają oznakowane i zarchiwizowane. Tego typu witryny stanowią etap pośredni pomiędzy listą dyskusyjną a portalem społecznościowym; doskonałym przykładem jest serwis *Stack Overflow* (<https://stackoverflow.com/>).

Internet

Internet jest wypełniony informacjami na temat R, a do tego dostępne jest specyficzne narzędzie do ich wyszukiwania (receptura 1.11). Internet zmienia się przez cały czas, dlatego zawsze warto sprawdzać go pod względem nowych, lepszych sposobów organizowania i wyszukiwania informacji na temat R.

Listy dyskusyjne

Ochotnicy hojnie poświęcają swój czas na odpowiadanie na pytania nowicjuszy zamieszczone na listach dyskusyjnych społeczności R. Listy te są archiwizowane, zatem możesz bez problemu przeglądać archiwa w poszukiwaniu rozwiązania trapiącego Cię problemu (receptura 1.13).

1.1. Pobranie i instalacja R

Problem

Chcesz zainstalować R na swoim komputerze.

Rozwiązanie

Użytkownicy systemów Windows i macOS mogą pobrać R z serwisu CRAN (ang. *Comprehensive R Archive Network* — kompleksowe archiwum sieciowe R). W przypadku Linuksa i Uniksa wystarczy zainstalować pakiety R za pomocą menedżera pakietów.

System Windows

1. Otwórz stronę <https://www.r-project.org/> w swojej przeglądarce.
2. Kliknij odnośnik CRAN. Zostanie wyświetlona lista serwerów uporządkowana ze względu na kraj.
3. Wybierz serwer umieszczony w pobliżu Ciebie lub widoczny na samej górze serwer *0-Cloud*, gdyż dobrze działa w większości rejonów (<https://cloud.r-project.org/>).
4. W ramce *Download and Install R* kliknij odnośnik *Download R for Windows*.
5. Kliknij odnośnik *base*.
6. Kliknij odnośnik do pobrania najnowszej wersji R (pliku *.exe*).
7. Po zakończeniu pobierania kliknij dwukrotnie plik *.exe*, odpowiedz na pytania i postępuj zgodnie z instrukcjami.

System macOS

1. Otwórz stronę <https://www.r-project.org/> w swojej przeglądarce.
2. Kliknij odnośnik CRAN. Zostanie wyświetlona lista serwerów uporządkowana ze względu na kraj.
3. Wybierz serwer umieszczony w pobliżu Ciebie lub widoczny na samej górze serwer *0-Cloud*, gdyż dobrze działa w większości rejonów.
4. Kliknij odnośnik *Download R for (Mac) OS X*.

5. W sekcji *Latest release* kliknij plik *.pkg* w celu pobrania najnowszej wersji R
6. Po zakończeniu pobierania kliknij dwukrotnie plik *.pkg*, odpowiedz na pytania i postępuj zgodnie z instrukcjami.

Dystrybucje Linuksa lub Uniksa

Główne dystrybucje Linuksa zawierają pakiety służące do instalacji R. Kilka przykładów zostało zaprezentowanych w tabeli 1.1.

Tabela 1.1. Pakiety R w dystrybucjach Linuksa

Dystrybucja	Pakiet
Ubuntu lub Debian	<i>r-base</i>
Red Hat lub Fedora	<i>R.i386</i>
SUSE	<i>R-base</i>

Do pobrania i zainstalowania odpowiedniego pakietu użyj menedżera pakietów. Zazwyczaj konieczna jest znajomość hasła administratora *root* lub uprawnień *sudo*; jeżeli nie masz do nich dostępu, poproś administratora o zainstalowanie pakietu.

Dyskusja

Instalacja R w systemach Windows i macOS jest prosta, ponieważ na te platformy przygotowano pliki binarne (skompilowane programy). Wystarczy postępować zgodnie z powyższymi instrukcjami. Serwis CRAN zawiera również odnośniki do zasobów związanych z procesem instalacji, np. do często zadawanych pytań (ang. *frequently asked questions* — FAQ), a także wskazówek do nietypowych sytuacji („Czy można zainstalować R w systemie Windows Vista/7/8/Server 2008?”).

Najlepszym sposobem instalacji R w Linuksie lub Uniksie jest zainstalowanie go jako pakietu z poziomu menedżera pakietów. Rozwiązanie to znacznie ułatwia zarówno proces instalacji, jak i późniejsze aktualizowanie języka.

W dystrybucjach Ubuntu i Debian do pobrania i instalacji R użyj menedżera *apt-get*. Aby uzyskać potrzebne uprawnienia, skorzystaj z konta *sudo*:

```
$ sudo apt-get install r-base
```

Z kolei w dystrybucjach Red Hat i Fedora możesz wykorzystać menedżer pakietów *yum*:

```
$ sudo yum install R.i386
```

Większość dystrybucji Linuksa zawiera również graficzne menedżery pakietów, które dla niektórych osób są wygodniejsze w użyciu.

Oprócz tych pakietów bazowych zalecamy instalację pakietów dokumentacji. Chcemy zainstalować pakiet *r-base-html* (ponieważ lubimy przeglądać dokumentację zawierającą hiperłącza), a także *r-doc-html*, umieszczający na dysku ważne instrukcje obsługi R:

```
$ sudo apt-get install r-base-html r-doc-html
```

Niektóre linuksowe repozytoria zawierają również kopie pakietów R dostępnych w serwisie CRAN. Nie będziemy z nich jednak korzystać, ponieważ wolimy pobierać je bezpośrednio z serwisu CRAN, gdzie są zazwyczaj dostępne ich najbardziej aktualne wersje.

W rzadkich przypadkach może być konieczna kompilacja środowiska R od podstaw. Być może korzystasz z nietypowej, nieobsługiwanej wersji Uniksa lub musisz brać pod uwagę specyficzne kwestie związane z wydajnością albo konfiguracją. Procedura kompilacji w Linuksie i Uniksie jest standardowa. Wystarczy pobrać skompresowane archiwum *.tar* z wybranego serwera CRAN; jego nazwa wygląda mniej więcej tak: *R-3.5.1.tar.gz*, gdzie wyrażenie *3.5.1* jest zastępowane bieżącą wersją. Rozpakuj to archiwum, poszukaj pliku o nazwie *INSTALL* i postępuj zgodnie z umieszczonymi w nim instrukcjami.

Zobacz również

W książce *R in a Nutshell Josepha Adlera* (O'Reilly) zawarto szczegółowy opis pobierania i instalowania R, w tym instrukcje jego kompilowania w systemach Windows i macOS. Być może kompletnym przewodnikiem jest *R Installation and Administration* (<https://cran.r-project.org/doc/manuals/R-admin.html>), dostępny w serwisie CRAN, opisujący kompilowanie i instalowanie R na różnych platformach.

Niniejsza receptura opisuje instalację pakietu bazowego. Jeżeli chcesz zainstalować dodatkowe pakiety z poziomu serwisu CRAN, zajrzyj do receptury 3.10.

1.2. Instalacja środowiska RStudio

Problem

Potrzebujesz bardziej kompleksowego **zintegrowanego środowiska programistycznego** (ang. *integrated development environment — IDE*) od standardowego interfejsu R. Innymi słowy, chcesz zainstalować program RStudio Desktop.

Rozwiązanie

W ciągu kilku ostatnich lat aplikacja RStudio stała się najczęściej stosowanym środowiskiem R. Uważamy, że niemal wszystkie projekty związane z R powinny być realizowane w programie RStudio Desktop, chyba że istnieje dobry powód, by tego nie robić. Na środowisko RStudio składa się kilka produktów, w tym takie, jak RStudio Desktop, RStudio Server czy RStudio Shiny Server. W niniejszej książce, pisząc *RStudio*, mamy na myśli aplikację RStudio Desktop, chociaż większość związanych z nim pojęć znajduje również zastosowanie w programie RStudio Server.

Aby zainstalować środowisko RStudio, pobierz z oficjalnej strony (<https://www.rstudio.com/products/rstudio/download/>) najnowszy instalator na używaną przez Ciebie platformę.

Wersja *RStudio Desktop Open Source License* jest bezpłatna do pobrania i użytkowania.

Dyskusja

Podczas pisania książki korzystaliśmy z wersji 1.2x środowiska RStudio i wersji 3.5.x R. Nowe wydania środowiska RStudio są udostępniane co kilka miesięcy, dlatego warto regularnie je aktualizować. Program RStudio działa z dowolną zainstalowaną wersją R, dlatego aktualizacja do najnowszej wersji środowiska RStudio *nie* aktualizuje jednocześnie R; należy przeprowadzić tę czynność oddzielnie.

Oddziaływanie z R poprzez środowisko RStudio różni się nieco od korzystania z interfejsu domyślnego. Na potrzeby niniejszej książki zdecydowaliśmy się realizować wszystkie przykłady z poziomu aplikacji RStudio.

1.3. Uruchamianie środowiska RStudio

Problem

Chcesz uruchomić program RStudio na swoim komputerze.

Rozwiązanie

Powszechnym błędem popełnianym przez nowych użytkowników R i środowiska RStudio jest mylenie obydwu ikon. Najpewniejszym sposobem uruchomienia aplikacji RStudio jest jej wyszukanie na komputerze, a następnie umieszczenie jej ikony w jakimś łatwym do odnalezienia miejscu:

System Windows

Kliknij menu *Start* w lewym dolnym rogu ekranu, a następnie w polu wyszukiwania wpisz RStudio.

System macOS

Wyszukaj aplikację RStudio w pasku startowym lub wciśnij kombinację klawiszy *Cmd+Spacja* (*Cmd* to skrót od nazwy *Command*, jest symbolizowany znakiem ⌘) i wpisz RStudio, aby wyszukać program za pomocą funkcji Spotlight.

Ubuntu

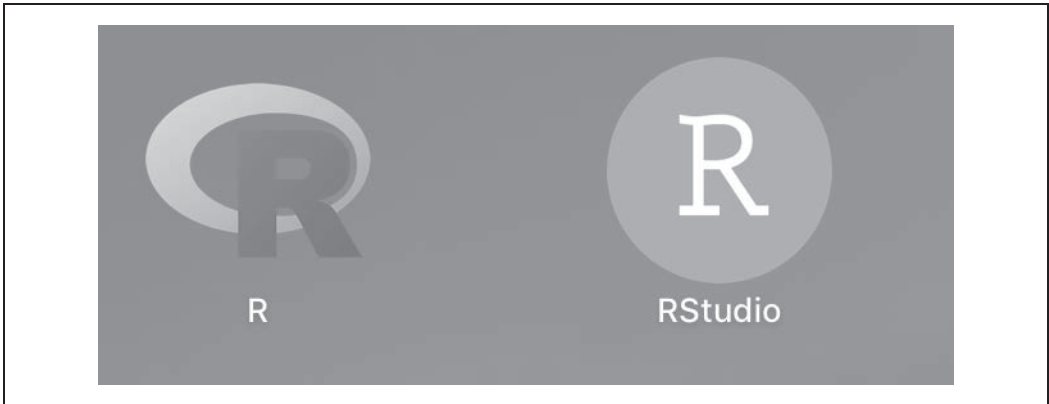
Wciśnij kombinację klawiszy *Alt+F1* i wpisz RStudio.

Dyskusja

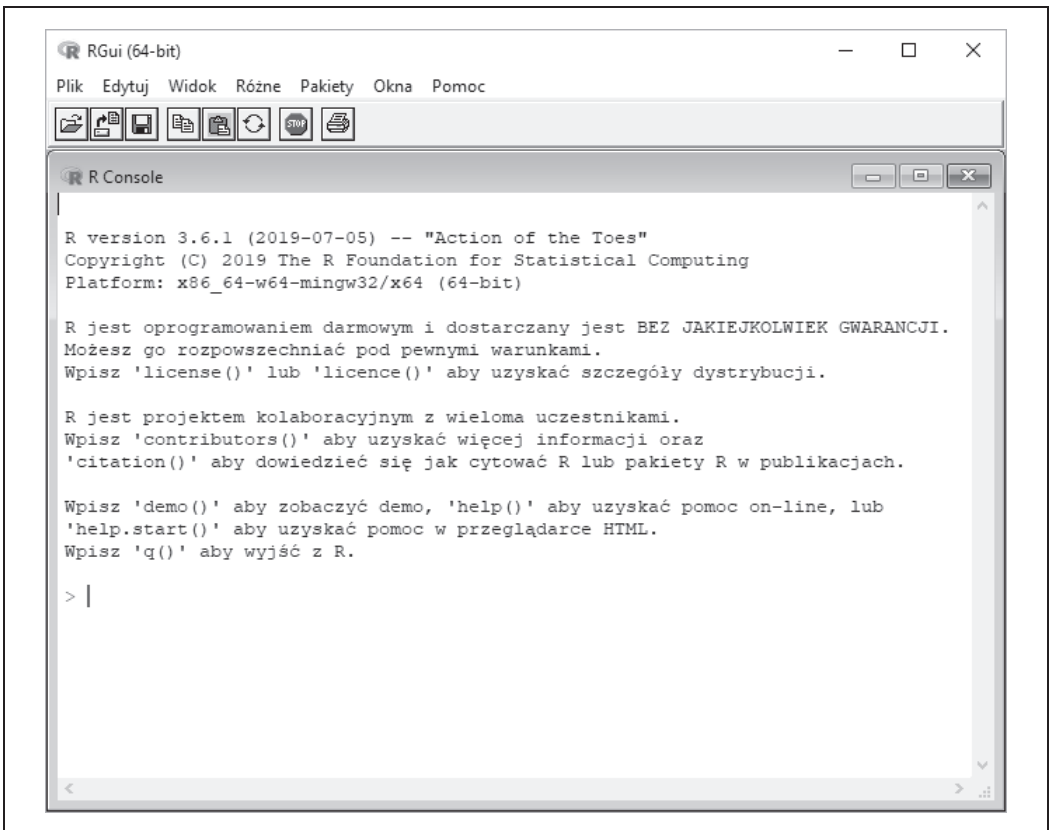
Łatwo się pomylić, ponieważ, jak widać na rysunku 1.1, ikony R i środowiska RStudio wyglądają podobnie.

Jeżeli klikniesz ikonę R, Twoim oczom ukaże się okno zaprezentowane na rysunku 1.2, czyli podstawowy interfejs R w systemie Windows, ale z pewnością niebędący częścią środowiska RStudio.

Podczas korzystania ze środowiska RStudio po uruchomieniu jest domyślnie otwierany ostatnio tworzony projekt.



Rysunek 1.1. Ikony R i aplikacji RStudio w systemie Windows



Rysunek 1.2. Konsola R w systemie Windows

1.4. Wprowadzanie poleceń

Problem

Uruchomiłeś środowisko RStudio. Co teraz?

Rozwiązanie

Po uruchomieniu aplikacji RStudio głównym ekranem po lewej stronie jest okno sesji R. To tutaj właśnie możesz w interaktywny sposób wprowadzać bezpośrednio polecenia.

Dyskusja

W R znak `>` jest symbolem zachęty. Na początku wystarczy traktować środowisko R jak rozbudowany kalkulator: wprowadź jakieś wyrażenie, a R przeprowadzi obliczenia i wyświetli wynik:

```
> 1 + 1
[1] 2
>
```

Komputer sumuje wartości 1 i 1, po czym wyświetla wynik, czyli 2.

Zapis `[1]` na początku wyniku może budzić zdziwienie. W R wynik jest wektorem, nawet jeśli składa się tylko z jednego elementu. Przez umieszczenie etykiety `[1]` dla otrzymanej wartości R podkreśla, że mamy do czynienia z pierwszym elementem wektora... co nie powinno nas dziwić, gdyż jest on *jedynym* elementem tego wektora.

R będzie czekał, dopóki nie zostanie wpisane pełne wyrażenie. Takim wyrażeniem jest zapis `max(1, 3, 5)`, zatem R przestaje odczytywać dane wejściowe i oblicza wynik z tego, co już otrzymał:

```
> max(1, 3, 5)
[1] 5
>
```

Z kolei `max(1, 3,` stanowi przykład wyrażenia niepełnego, zatem R będzie zachęcał Cię do wprowadzania kolejnych danych wejściowych. Symbol zachęty zmienia się ze znaku „większy od” (`>`) na znak „plus” (`+`), co oznacza, że R czeka na więcej danych:

```
> max(1, 3,
+ 5)
[1] 5
>
```

Łatwo się pomylić podczas wpisywania poleceń, a ich poprawianie jest męczące i frustrujące, zatem w celu ułatwienia życia została udostępniona funkcja edycji na poziomie wiersza polecenia. Mamy do dyspozycji zdefiniowane skróty klawiszowe pozwalające szybko wywoływać, poprawiać i ponownie wykonywać polecenia. Typowe oddziaływanie z wierszem polecenia wygląda następująco:

1. Nieprawidłowo wpisujesz wyrażenie R.
2. Środowisko R narzeka z powodu pomyłki.
3. Wciskasz klawisz `↵` (strzałka w górę), aby ponownie wyświetlić nieprawidłowo zapisany wiersz.

4. Używasz klawiszy ← (strzałka w lewo) i → (strzałka w prawo) do umieszczenia kursora przy błędnym zapisie.
5. Za pomocą klawisza *Delete* usuwasz nieprawidłowe znaki.
6. Wpisujesz prawidłowe znaki, dzięki czemu zostają umieszczone w wierszu polecenia.
7. Wciskasz klawisz *Enter*, aby ponownie wykonać poprawione polecenie.

Tak wyglądają podstawy. R obsługuje standardowe skróty klawiszowe służące do wyświetlania i edytowania poleceń, co zostało ukazane w tabeli 1.2.

Tabela 1.2. Skróty klawiszowe poleceń

Oznakowany klawisz	Skrót z użyciem klawisza Ctrl	Skutek
Strzałka w górę	<i>Ctrl+P</i>	Wyświetla uprzednio wpisywane polecenia w kolejności od najnowszego.
Strzałka w dół	<i>Ctrl+N</i>	Wyświetla polecenia w kolejności od najstarszego.
<i>Backspace</i>	<i>Ctrl+H</i>	Usuwa znak po lewej stronie kursora.
<i>Delete (Del)</i>	<i>Ctrl+D</i>	Usuwa znak po prawej stronie kursora.
<i>Home</i>	<i>Ctrl+A</i>	Przenosi kursor na początek wiersza.
<i>End</i>	<i>Ctrl+E</i>	Przenosi kursor na koniec wiersza.
Strzałka w prawo	<i>Ctrl+F</i>	Przesuwa kursor o jeden znak w prawo (do przodu).
Strzałka w lewo	<i>Ctrl+B</i>	Przesuwa kursor o jeden znak w lewo (do tyłu).
	<i>Ctrl+K</i>	Usuwa treść od pozycji kursora do końca wiersza.
	<i>Ctrl+U</i>	Usuwa cały wiersz.
<i>Tab</i>		Uzupełnia nazwę (na niektórych platformach).

W większości systemów operacyjnych możesz także zaznaczać polecenia za pomocą myszy, jak również stosować standardowe polecenia kopiowania/wklejania w celu umieszczania tekstu w nowym wierszu.

Zobacz również

Patrz również receptura 2.12. W aplikacji RStudio dla systemu Windows wybierz *Help/Keyboard Shortcuts Help*, aby wyświetlić pełną listę wszystkich skrótów klawiszowych pomocnych w edytowaniu wiersza polecenia.

1.5. Wyjście ze środowiska RStudio

Problem

Chcesz wyjść z aplikacji RStudio.

Rozwiązanie

System Windows i większość dystrybucji Linuksa

W menu głównym kliknij *File/Quit Session* lub ikonę *X* w prawym górnym rogu okna.

System macOS

W menu głównym kliknij *File/Quit Session* lub wciśnij kombinację klawiszy *Cmd+Q*, ewentualnie kliknij czerwone kółko w lewym górnym rogu okna.

Na wszystkich platformach do przerywania działania R i środowiska RStudio możesz również wykorzystać funkcję *q* (skrót od angielskiego słowa *quit* — wyjście):

```
q()
```

Pusty nawias jest niezbędny do wywołania tej funkcji.

Dyskusja

Za każdym razem gdy chcesz wyjść ze środowiska R, będziesz pytany o to, czy chcesz zapisać swoją przestrzeń roboczą (ang. *workspace*). Masz do wyboru trzy możliwości:

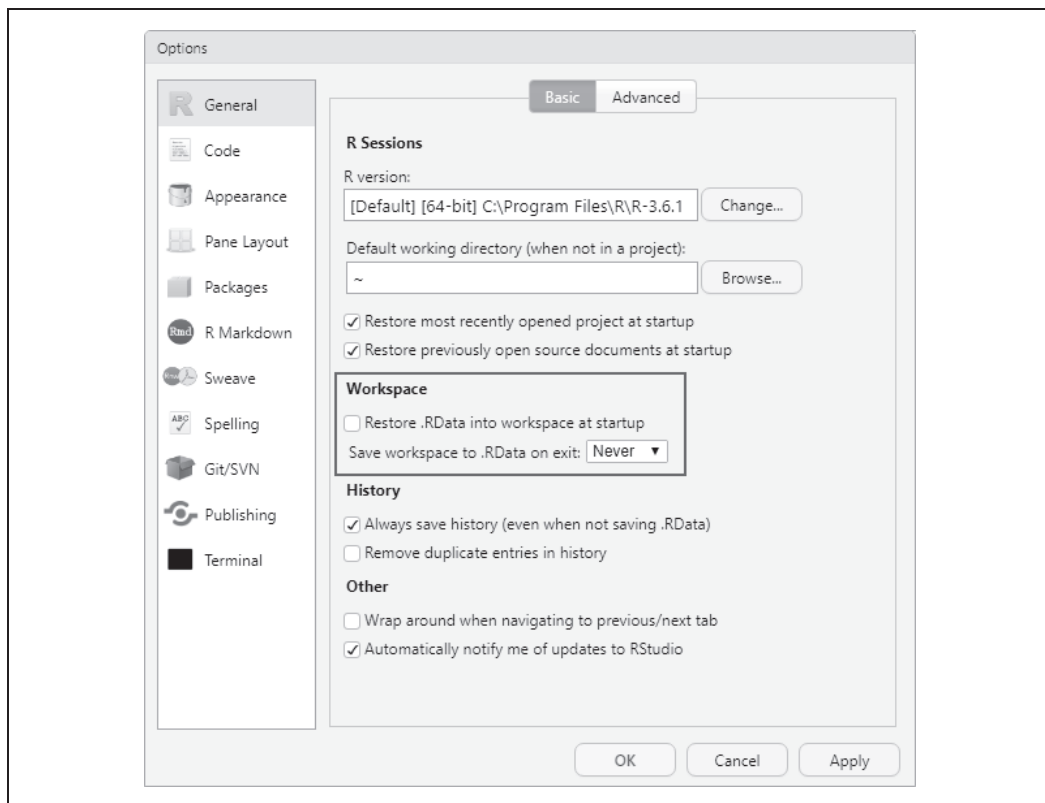
- zapisanie przestrzeni roboczej i wyjście,
- wyjście z programu bez zapisywania przestrzeni roboczej,
- zrezygnowanie z wyjścia i powrót do wiersza polecenia.

W przypadku zapisywania przestrzeni roboczej zostaje utworzony plik *.RData* w bieżącym katalogu roboczym. W ten sposób zachowujemy wszelkie utworzone przez siebie obiekty R. Przy następnym uruchomieniu środowiska R z poziomu tego katalogu roboczego nasza przestrzeń robocza zostanie automatycznie wczytana. Zapisanie nowszej przestrzeni roboczej spowoduje nadpisanie jej starszej wersji (jeśli istnieje), dlatego nie zapisuj jej, jeżeli nie jesteś zadowolony z wprowadzonych zmian (np. jeśli przypadkowo usunąłeś ważne dane z przestrzeni roboczej).

Zalecamy, abyś nigdy nie zapisywał przestrzeni roboczej w momencie wyjścia z programu, a zamiast tego zawsze jawnie zapisywał projekty, skrypty i dane. Proponujemy również, abyś wyłączył propozycję zapisywania i automatycznego wczytywania przestrzeni roboczej w środowisku RStudio przez modyfikację opcji w menu *Tools/Global Options* w sposób zaprezentowany na rysunku 1.3. W ten sposób w momencie wyjścia ze środowiska R i aplikacji RStudio nie będziesz pytany o zapisanie przestrzeni roboczej. Pamiętaj jednak, że obiekty utworzone, ale nie zapisane na dysku zostaną utracone!

Zobacz również

Receptura 3.1 przybliży Ci informacje na temat bieżącego katalogu roboczego, a z receptury 3.3 dowiesz się więcej o zapisywaniu przestrzeni roboczej. Warto również zapoznać się z rozdziałem 2. książki *R in a Nutshell*.



Rysunek 1.3. Opcje zapisywania przestrzeni roboczej

1.6. Przerwanie realizacji kodu R

Problem

Chcesz przerwać długotrwałe obliczenia i wrócić do wiersza polecenia bez potrzeby wychodzenia ze środowiska RStudio.

Rozwiązanie

Wciśnij klawisz *Esc* lub kliknij menu *Session* w aplikacji RStudio i wybierz opcję *Interrupt R*. Możesz również kliknąć ikonę ze znakiem „Stop” w oknie konsoli kodu.

Dyskusja

Przerwanie realizacji kodu R oznacza zaprzestanie wykonywania danego polecenia przez R, ale bez potrzeby usuwania zmiennych z pamięci i całkowitego zamknięcia środowiska RStudio. Oznacza to jednak, że zmienne mogą znajdować się w stanie nieokreślonym, w zależności od postępów przeprowadzonych obliczeń, dlatego po przerwaniu należy sprawdzić przestrzeń roboczą.

Zobacz również

Patrz receptura 1.5.

1.7. Przeglądanie dołączonej dokumentacji

Problem

Chcesz zapoznać się z dokumentacją dołączoną do R.

Rozwiązanie

Aby uzyskać dostęp do spisu treści dokumentacji, użyj funkcji `help.start`:

```
help.start()
```

W ten sposób zostanie otwarty panel z odnośnikami do zainstalowanej dokumentacji. W środowisku RStudio pomoc jest dostępna w zakładce *Help*, umieszczonej domyślnie w prawym dolnym panelu.

Możesz również w aplikacji RStudio kliknąć menu *Help/R Help*, aby otworzyć spis opcji pomocy dla zarówno R, jak i środowiska RStudio.

Dyskusja

Dystrybucja bazowa R zawiera bogatą dokumentację, dosłownie tysiące stron. Z kolei dokumentacja dostępna na komputerze lokalnym jest instalowana wraz z dodatkowymi pakietami.

Możesz z łatwością przeglądać tę dokumentację za pomocą funkcji `help.start`, służącej do otwierania głównego spisu treści. Rysunek 1.4 przedstawia wynik funkcji `help.start` wewnątrz panelu pomocy w środowisku RStudio.

Dwa odnośniki w sekcji *Reference* są wyjątkowo przydatne:

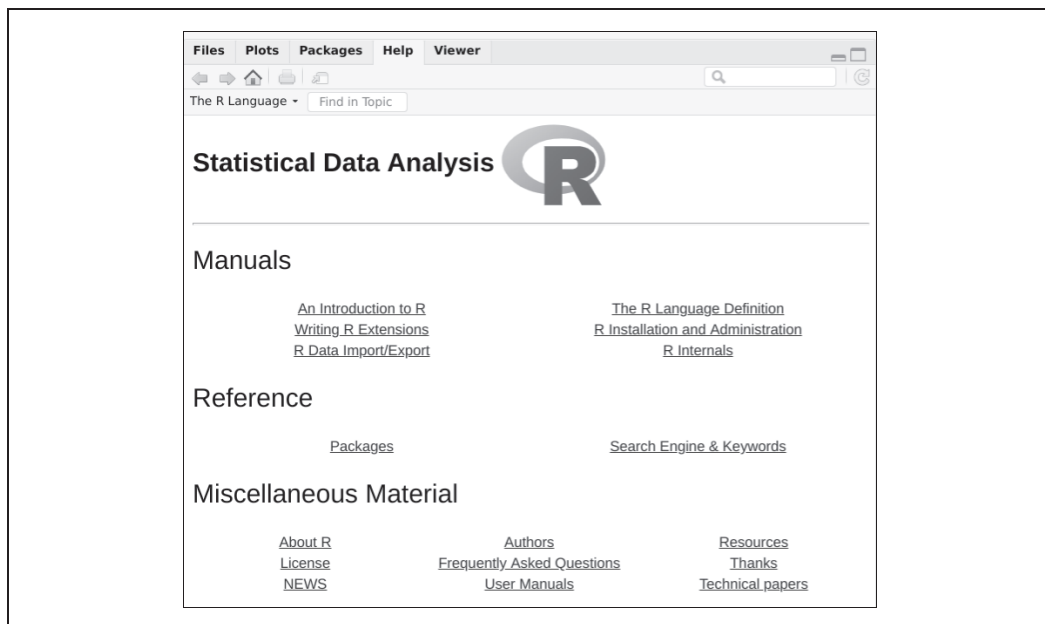
Packages

Kliknij tu, aby przejrzeć listę wszystkich zainstalowanych pakietów, zarówno bazowych, jak i dodatkowych. Po kliknięciu nazwy pakietu wyświetlona zostanie lista jego funkcji i zestawów danych.

Search Engine & Keywords

Kliknij tu, aby uzyskać dostęp do prostej wyszukiwarki umożliwiającej szukanie dokumentacji za pomocą słów kluczowych lub wyrażeń. Znajdziesz tu także uporządkowaną tematycznie listę najpopularniejszych słów kluczowych; po kliknięciu jednego z nich zostaną wyświetlone powiązane z nim strony.

Dokumentacja bazowej wersji R, do której uzyskujemy dostęp za pomocą funkcji `help.start`, zostaje umieszczona na dysku w czasie instalacji środowiska R. Pomoc aplikacji RStudio, uruchamiana przez kliknięcie opcji *Help/R Help*, zostaje wyświetlona w postaci strony z odnośnikami do serwisu pakietu RStudio, zatem do jej przeglądania wymagany jest dostęp do internetu.



Rysunek 1.4. Wynik funkcji `help.start` w aplikacji RStudio

Zobacz również

Dokumentacja lokalna stanowi kopię dokumentacji z witryny projektu R (<https://www.r-project.org/>), gdzie mogą znajdować się uaktualnione dokumenty.

1.8. Uzyskiwanie pomocy na temat funkcji

Problem

Chcesz dowiedzieć się więcej na temat zainstalowanej funkcji.

Rozwiązanie

Za pomocą funkcji `help` możesz wyświetlić dokumentację danej funkcji:

```
help(nazwa_funkcji)
```

Funkcja `args` wyświetla przypomnienie argumentów danej funkcji:

```
args(nazwa_funkcji)
```

Do prezentacji przykładów wykorzystania danej funkcji służy funkcja `example`:

```
example(nazwa_funkcji)
```


Dyskusja

W niniejszej książce prezentujemy mnóstwo funkcji. Każda z nich zawiera więcej niuansów, niż jesteśmy w stanie tu opisać. Jeżeli jakaś funkcja wzbudzi Twoje zainteresowanie, sugerujemy zapoznanie się z jej dokumentacją. Któryś z niuansów może okazać się bardzo przydatny.

Założmy, że interesuje Cię funkcja `mean`. Możesz użyć funkcji `help` w następujący sposób:

```
help(mean)
```

W ten sposób zostanie otwarta strona pomocy funkcji `mean` w panelu pomocy środowiska RStudio. Zamiast wpisywać polecenie `help`, możesz wpisać nazwę funkcji poprzedzoną symbolem `?`:

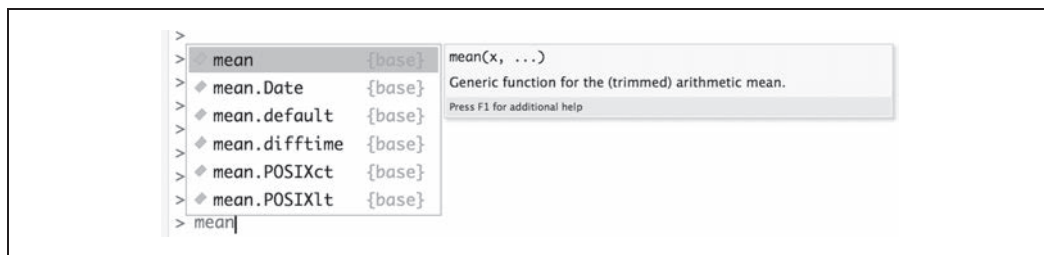
```
?mean
```

Czasami potrzebne jest nam tylko szybkie przypomnienie argumentów danej funkcji: czym są i w jakiej występują kolejności. W takich sytuacjach przydaje się funkcja `args`:

```
args(mean)
#> function (x, ...)
#> NULL
```

```
args(sd)
#> function (x, na.rm = FALSE)
#> NULL
```

Pierwszy wiersz wyniku funkcji `args` stanowi streszczenie wywołania danej funkcji. W przypadku funkcji `mean` widzimy jeden argument, `x`, będący wektorem liczbowym. Z kolei dla funkcji `sd` widzimy ten sam wektor `x`, a także dodatkowy argument `na.rm`; możesz zignorować drugi wiersz wyniku, gdyż bardzo często ma on wartość `NULL`. W aplikacji RStudio wynik funkcji `args` jest wyświetlany jako podpowiedź po wpisaniu nazwy funkcji, co zostało zaprezentowane na rysunku 1.5.



Rysunek 1.5. Podpowiedź w środowisku RStudio

Większość dokumentacji zawiera przykładowy kod pod koniec dokumentu. Przydatną własnością R jest możliwość realizacji tych przykładów, dzięki czemu możesz poznać próbkę możliwości danej funkcji. Możemy choćby sprawdzić możliwości funkcji `mean` bez konieczności samodzielnego przepisywania przykładowego kodu. Wystarczy użyć funkcji `example`:

```
example(mean)
#>
#> mean> x <- c(0:10, 50)
#>
#> mean> xm <- mean(x)
```

```
#>
#> mean> c(xm, mean(x, trim = 0.10))
#> [1] 8.75 5.50
```

Wszystkie wiersze po funkcji `example(mean)` zostały wygenerowane przez R; użyto tu przykładów z dokumentacji i wyświetlono rezultat.

Zobacz również

Receptura 1.9 ukazuje sposób wyszukiwania funkcji, a w recepturze 3.9 zajmujemy się ścieżką wyszukiwania.

1.9. Wyszukiwanie dodatkowej dokumentacji

Problem

Chcesz uzyskać więcej informacji na temat funkcji zainstalowanej na komputerze, ale funkcja `help` nie jest w stanie znaleźć dokumentacji.

Ewentualnie chcesz wyszukać zainstalowaną dokumentację zawierającą dane słowo kluczowe.

Do przeszukiwania dokumentacji R na komputerze użyj funkcji `help.search`:

```
help.search("wzorzec")
```

Typowym wzorcem jest nazwa funkcji lub słowo kluczowe. Zwróć uwagę, że należy umieścić je w cudzysłowie.

Znacznym ułatwieniem jest możliwość wywołania funkcji wyszukiwania przez wpisanie dwóch znaków zapytania (w tym przypadku cudzysłów staje się niepotrzebny). Należy zauważyć, że do wyszukiwania funkcji używamy jednego pytajnika, natomiast do wyszukiwania wzorca tekstowego potrzebne są dwa znaki zapytania:

```
> ??wzorzec
```

Dyskusja

Od czasu do czasu możesz szukać pomocy na temat jakiejś funkcji tylko po to, aby się dowiedzieć, że środowisko R nie zawiera żadnych informacji na jej temat:

```
help(adf.test)
#> No documentation for 'adf.test' in specified packages and libraries:
#> you could try '??adf.test'
```

Bywa to frustrujące, gdy *wiesz*, że dana funkcja jest zainstalowana na komputerze. W tym przypadku problem polega na tym, że pakiet funkcji nie jest w danym momencie wczytany, a Ty nie wiesz, w którym pakiecie znajduje się ta funkcja. Jest to swoiste błędne koło (zgodnie z komunikatem błędu pakiet nie znajduje się obecnie w ścieżce wyszukiwania, dlatego R nie może znaleźć jego pliku pomocy; więcej informacji znajdziesz w recepturze 3.6).

Rozwiązaniem okazuje się przeszukanie wszystkich zainstalowanych pakietów. Wystarczy skorzystać z funkcji `help.search` tak, jak zostało zasugerowane w komunikacie o błędzie:

```
help.search("adf.test")
```

W rezultacie zostanie wyświetlona lista wszystkich pakietów zawierających daną funkcję:

```
Help files with alias or concept or title matching 'adf.test' using
regular expression matching:
```

```
tseries::adf.test      Augmented Dickey-Fuller Test
Type '?PKG::FOO' to inspect entry 'PKG::FOO TITLE'.
```

Z powyższego wyniku dowiadujemy się, że funkcję `adf.test` zawiera pakiet `tseries`. Możesz bezpośrednio przejrzeć jego dokumentację z wykorzystaniem funkcji `help`:

```
help(adf.test, package = "tseries")
```

Możesz ewentualnie użyć operatora `::`, dzięki któremu R sprawdzi wyznaczony pakiet:

```
?tseries::adf.test
```

Istnieje możliwość poszerzenia zakresu przeszukiwania za pomocą słów kluczowych. R wyszuka w ten sposób zainstalowaną dokumentację zawierającą dany termin. Załóżmy, że interesują nas wszystkie funkcje mające jakiś związek z rozszerzonym testem Dickeya-Fullera (ang. *augmented Dickey-Fuller test* — *ADF*). W tym celu możemy wykorzystać następujący wzorzec:

```
help.search("dickey-fuller")
```

Zobacz również

Masz również do dyspozycji lokalną wyszukiwarkę z poziomu przeglądarki dokumentacji; sposób jej użycia został opisany w recepturze 1.7. Więcej informacji na temat ścieżki przeszukiwania znajdziesz w recepturze 3.7, a receptura 1.8 omawia uzyskiwanie pomocy na temat funkcji.

1.10. Uzyskiwanie pomocy na temat pakietu

Problem

Chcesz zdobyć więcej informacji na temat zainstalowanego pakietu.

Rozwiązanie

Użyj funkcji `help`, wewnątrz której możesz podać nazwę pakietu:

```
help(package = "nazwa_pakietu")
```

Dyskusja

Czasami możesz chcieć sprawdzić zawartość pakietu (funkcje i zestawy danych). Przykładowo dotyczy to sytuacji, gdy pobierasz i instalujesz nowe pakiety. Funkcja `help` pozwala sprawdzić zawartość pakietu, a także inne informacje na jego temat.

Przez poniższe wywołanie funkcji `help` zostanie wyświetlona zawartość pakietu `tseries`, czyli jednego ze standardowych pakietów w dystrybucji bazowej:

```
help(package = "tseries")
```

Najpierw zostaje wyświetlony opis pakietu, a po nim indeks funkcji i zestawów danych. W aplikacji RStudio następuje otwarcie strony pomocy sformatowanej z użyciem kodu HTML wewnątrz okna pomocy środowiska IDE.

Niektóre pakiety zawierają również ulotki, czyli dodatkowe dokumenty, takie jak wprowadzenia, samouczki czy karty referencyjne. Zostają one zainstalowane jako część dokumentacji wraz z pakietem. Na stronie pomocy danego pakietu lista ulotek jest zlokalizowana pod koniec dokumentu.

Możesz przejrzeć listę wszystkich ulotek znajdujących się na dysku za pomocą funkcji `vignette`:

```
vignette()
```

W aplikacji RStudio zostanie otwarta nowa zakładka, która ukazuje listę wszystkich zainstalowanych pakietów zawierających ulotki wraz z nazwami i opisami tych ulotek.

Przez dołączenie nazwy pakietu możesz wyświetlić listę wszystkich jego ulotek:

```
vignette(package = "nazwa_pakietu")
```

Każda ulotka ma wyznaczoną nazwę, za pomocą której może zostać wyszukana:

```
vignette("nazwa_ulotki")
```

Zobacz również

Więcej informacji na temat uzyskania pomocy co do funkcji stanowiącej część danego pakietu znajdziesz w recepturze 1.8.

1.11. Wyszukiwanie pomocy w internecie

Problem

Chcesz znaleźć w internecie informacje i odpowiedzi związane z R.

Rozwiązanie

Aby wyszukać słowo kluczowe albo wyrażenie, użyj funkcji `RSiteSearch` w środowisku R:

```
RSiteSearch("wyrażenie kluczowe")
```

Warto też przeglądać następujące serwisy:

RSeek (<https://rseek.org/>)

Jest to niestandardowa wyszukiwarka skupiająca się na witrynach poświęconych R.

Stack Overflow (<https://stackoverflow.com/>)

Serwis Stack Overflow to przeszukiwalna witryna Q&A, stanowiąca część sieci Stack Exchange, zorientowana na kwestie programistyczne, takie jak struktury danych, kodowanie i grafika. Serwis ten jest znakomitym „pierwszym przystankiem” w poszukiwaniu odpowiedzi na wszelkie pytania dotyczące składni.

Cross Validated (<https://stats.stackexchange.com/>)

Serwis Cross Validated stanowi część sieci Stack Exchange. Skupia się on na statystyce, uczeniu maszynowym i analizie danych. Warto pytać tu o wybór metody statystycznej.

RStudio Community (<https://community.rstudio.com/>)

Witryna RStudio Community to forum dyskusyjne zarządzane przez twórców środowiska RStudio. Wśród przedmiotów zainteresowania znajdziemy tu R, środowisko RStudio i powiązane z nimi technologie. Ze względu na przynależność serwis ten jest często odwiedzany przez personel RStudio, a także stałych użytkowników oprogramowania. Jest to dobre miejsce na zadawanie ogólnych pytań, jak również takich, które nie pasują zbyt do formatu stosowanego w witrynie Stack Overflow.

Dyskusja

Funkcja `RSiteSearch` otwiera okno przeglądarki i przekierowuje ją do wyszukiwarki na stronie witryny projektu R (<http://search.r-project.org/>). Zostaną wyświetlone wstępne wyniki wyszukiwania, które możesz dalej uściślać. Na przykład poniższe wywołanie tej funkcji wyświetli rezultaty dla wyrażenia `canonical correlation`:

```
RSiteSearch("canonical correlation")
```

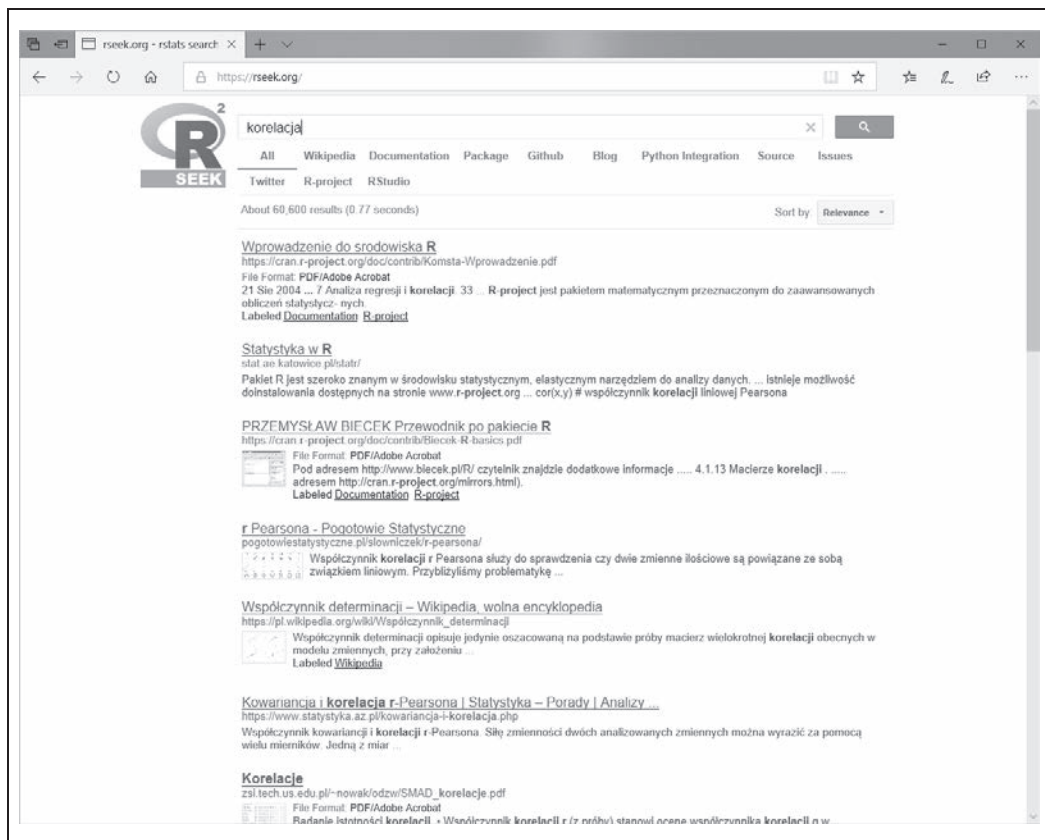
Rozwiązanie to przydaje się do szybkiego przeszukiwania internetu bez wychodzenia ze środowiska R, ale zakres przeszukiwania ogranicza się do dokumentacji R i archiwów list dyskusyjnych¹.

Serwis *RSeek* (<https://rseek.org/>) zapewnia większe pole manewru. Jego zaletą jest wykorzystanie możliwości wyszukiwarki Google przy jednoczesnym koncentrowaniu się na stronach poświęconych R. W ten sposób unikamy nieistotnych wyników wyszukiwania standardowej wyszukiwarki Google. Piękno serwisu *RSeek* polega na tym, że organizuje wyniki wyszukiwania w przydatny sposób.

Rysunek 1.6 przedstawia wyniki wyszukania rezultatów dla słowa `korelacja` w serwisie *RSeek*. Zakładki w górnej części strony pozwalają filtrować wyniki wyszukiwania pod względem różnych typów treści:

- *All* (wszystkie),
- *Wikipedia*,
- *Documentation* (dokumentacja),
- *Package* (pakiet),
- *Github*,
- *Blog*,

¹ Zatem funkcja ta nie rozpoznaje słów i wyrażeń kluczowych w języku polskim — *przyp. tłum.*



Rysunek 1.6. Wyszukiwarka RSeek

- *Python Integration* (integracja z językiem Python),
- *Source* (źródło),
- *Issues* (problemy),
- *Twitter*,
- *R-project*,
- *RStudio*.

Serwis *Stack Overflow* (<https://stackoverflow.com/>) jest stroną typu Q&A, co oznacza, że każdy może zamieścić tam pytanie, a doświadczeni użytkownicy udzielą odpowiedzi; często jedno pytanie dostaje wiele odpowiedzi. Czytelnicy głosują na odpowiedzi, zatem te najlepsze wyświetlane są w pierwszej kolejności. Powstaje w ten sposób bogata baza pytań i odpowiedzi, którą możesz swobodnie przeglądać. Serwis *Stack Overflow* jest ściśle zorientowany na rozwiązywanie problemów, a zakres tematyki wiąże się głównie z zagadnieniami programistycznymi R.

Witryna *Stack Overflow* poświęcona jest wielu językom programowania, dlatego podczas wprowadzania w polu wyszukiwania jakiegoś terminu warto umieścić na początku wyrażenie [r], przez co wyszukiwarka skupi się na pytaniach związanych z R. Na przykład po wpisaniu [r] standard

error będą wyszukiwane wyłącznie zapytania przeznaczone dla R, natomiast zostaną pominięte pytania związane z językami Python czy C++.

W serwisie Stack Overflow znajduje się również *encyklopedia wiki poświęcona R* (<https://stackoverflow.com/tags/r/info>), zawierająca stworzoną przez społeczność znakomitą listę zasobów internetowych.

Sieć Stack Exchange (do której należy serwis Stack Overflow) zawiera również sekcję Q&A poświęconą analizie statystycznej o nazwie *Cross Validated* (<https://stats.stackexchange.com/>). Witryna ta jest bardziej zorientowana na kwestie statystyczne niż programistyczne, zatem należy tam zaglądać raczej w poszukiwaniu odpowiedzi dotyczących ogólnie pojętej statystyki niż szczegółowych zagadnień z zakresu R.

Strona Rstudio zawiera także własny *panel dyskusyjny* (<https://community.rstudio.com/>). Jest to idealne miejsce do zadawania ogólnych, a także koncepcyjnych pytań, które mogą nie nadawać się do serwisu Stack Overflow.

Zobacz również

Jeżeli w wyniku wyszukiwania znajdziesz jakiś przydatny pakiet, instrukcję jego instalacji znajdziesz w recepturze 3.10.

1.12. Wyszukiwanie przydatnych funkcji i pakietów

Problem

Trudno stwierdzić, które z ponad 10 000 pakietów dostępnych dla R będą przydatne dla Ciebie.

Rozwiązanie

- Aby odkryć pakiety przeznaczone dla określonego działu statystyki, odwiedź *listę widoków zadań* (<https://cran.r-project.org/web/views/>) w serwisie CRAN. Wybierz interesujący Cię widok zadania, co spowoduje wyświetlenie odnośników do odpowiednich pakietów wraz z opisami. Ewentualnie otwórz wyszukiwarkę *RSeek* (<https://rseek.org/>), wpisz słowo kluczowe, przejdź do zakładki *Task Views* i wybierz odpowiedni widok zadania.
- Odwiedź serwis *crantastic* (<https://crantastic.org/>) i wyszukaj pakiet za pomocą słowa kluczowego.
- Aby znaleźć istotne funkcje, otwórz wyszukiwarkę *RSeek* (<https://rseek.org/>), wpisz nazwę lub słowo kluczowe i przejdź do zakładki *Functions*.

Dyskusja

Problem ten jest szczególnie dokuczliwy dla początkujących. Uważasz, że R może rozwiązać Twoje problemy, ale nie masz pojęcia, które pakiety i funkcje będą przydatne. Popularnym pytaniem umieszczanym na listach dyskusyjnych jest: „Czy istnieje pakiet służący do rozwiązania problemu X?”. Jest to niemy krzyk osoby tonącej w R.

W czasie pisania książki dostępnych do pobrania z serwisu CRAN jest ponad 10 000 bezpłatnych pakietów. Każdy z nich zawiera stronę z podsumowaniem, krótkim opisem i odnośnikami do dokumentacji. Po znalezieniu potencjalnie przydatnego pakietu należy zazwyczaj kliknąć odnośnik *Reference manual*, aby przejrzeć jego szczegółową dokumentację w formacie PDF (na stronie podsumowania umieszczone są również odnośniki do pobrania pakietu, pakiety są jednak rzadko instalowane w ten sposób; patrz receptura 3.10).

Czasami interesują nas jedynie zagadnienia ogólne, takie jak analiza bayesowska, ekonometria, optymalizacja czy grafika. W serwisie CRAN znajdziemy zestaw stron z widokami zadań zawierającymi opis pakietów, które mogą być użyteczne w danym kontekście. Warto rozpocząć poszukiwania od widoku zadań, ponieważ znajdziesz tam przegląd dostępnych pakietów. Możesz przejrzeć listę widoków zadań na stronie *CRAN Task Views* (<https://cran.r-project.org/web/views/>) lub wyszukać je tak, jak opisaliśmy w punkcie „Rozwiązanie”. Na widoki zadań składa się pokazna lista dziedzin, do których są przypisane odpowiednie pakiety. Mamy np. do dyspozycji widoki zadań dla obliczeń wysokowydajnych, genetyki, szeregów czasowych, nauk społecznych itd.

Załóżmy, że akurat znasz nazwę jakiejś przydatnej biblioteki; ktoś mógł wspomnieć tę nazwę na forum dyskusyjnym. W serwisie CRAN (<https://cran.r-project.org/web/packages/>) jest dostępna kompletna alfabetyczna lista pakietów z odnośnikami do stron podsumowania pakietu.

Zobacz również

Możesz pobrać i zainstalować pakiet *sos*, umożliwiający wyszukiwanie pakietów na wiele innych sposobów; patrz *ulotka pakietu sos* (<https://cran.r-project.org/web/packages/sos/vignettes/sos.pdf>).

1.13. Przeszukiwanie list dyskusyjnych

Problem

Masz jakieś pytanie i chcesz sprawdzić w archiwach list dyskusyjnych, czy ktoś już kiedyś na nie odpowiedział.

Rozwiązanie

Otwórz listę dyskusyjną *Nabble* (<http://r.789695.n4.nabble.com/>). Wpisz słowo kluczowe lub inne wyrażenie stanowiące część Twojego pytania. Zostaną wyświetlone wyniki z list dyskusyjnych pomocy technicznej.

Dyskusja

Ta receptura to w zasadzie jedno z zastosowań receptury 1.11. Jest to jednak ważne zastosowanie, ponieważ przed zamieszczeniem pytania na liście należy przejrzeć jej archiwa. Najprawdopodobniej ktoś już wcześniej odpowiedział na Twoje pytanie.

Zobacz również

Serwis CRAN zawiera listę dodatkowych zasobów internetowych; odwiedź adres *CRAN Search* (<https://cran.r-project.org/search.html>).

1.14. Przesyłanie pytań do serwisu Stack Overflow lub innego

Problem

Masz jakieś pytanie, na które nie możesz znaleźć odpowiedzi w internecie, zatem chcesz je umieścić do wglądu społeczności R.

Rozwiązanie

Pierwszym etapem zadawania pytania w internecie jest utworzenie odtwarzalnego przykładu. Najważniejszym elementem poszukiwań pomocy w sieci jest przygotowanie przykładowego kodu, który inne osoby mogą uruchomić, by dokładnie przyrzeć się Twojemu problemowi. Prawidłowe pytanie zawierające odtwarzalny przykład składa się z trzech części:

Przykładowe dane

Możesz dostarczyć dane symulowane lub rzeczywiste.

Przykładowy kod

Zadaniem tego kodu jest ukazanie czynności, którą próbowałeś zrealizować, lub wyświetlanego błędu.

Część opisowa

To tutaj opisujesz, co osiągnąłeś, co chcesz osiągnąć oraz jakie metody okazały się nieskuteczne.

Szczegóły przygotowania odtwarzalnego przykładu zostały omówione w punkcie „Dyskusja”. Po jego przygotowaniu możesz zamieścić pytanie w serwisie *Stack Overflow* (https://stackoverflow.com/users/login?ssrc=anon_ask&returnurl=https%3a%2f%2fstackoverflow.com%2fquestions%2fask). Nie zapomnij umieścić znacznika *r* w sekcji *Tags* na stronie zapytania.

Jeżeli pytanie jest ogólne lub związane bardziej z pojęciami niż ze składnią, warto pamiętać, że twórcy środowiska RStudio udostępnili *forum dyskusyjne społeczności RStudio* (<https://community.rstudio.com/>). Jest ono podzielone na wiele różnych działów, zatem wybierz ten, który jest najściślej związany z pytaniem.

Ewentualnie możesz umieścić pytanie na listach dyskusyjnych społeczności R (ograniczaj się jednak do jednego miejsca, gdyż zasypywanie różnych witryn, forów i list dyskusyjnych tym samym pytaniem uznawane jest za duży nietakt).

Strona list dyskusyjnych (<https://www.r-project.org/mail.html>) zawiera informacje i instrukcje dotyczące korzystania z tej formy kontaktu. Poniżej przedstawiamy najważniejsze wytyczne:

1. Zarejestruj się na głównej liście dyskusyjnej R, *R-help* (<https://stat.ethz.ch/mailman/listinfo/r-help>)².
2. Ostrożnie i prawidłowo sformułuj swoje pytanie i dołącz do niego odtwarzalny przykład.
3. Wyślij pytanie na adres email *r-help@r-project.org*.

Dyskusja

Lista dyskusyjna R-help, serwis Stack Overflow i witryna RStudio Community stanowią doskonałe źródła zasobów, należy jednak korzystać z nich jedynie w ostateczności. Czytaj strony pomocy, zapoznaj się z dokumentacją, przeszukuj archiwa pomocy i internet. Najprawdopodobniej ktoś już kiedyś udzielił odpowiedzi na Twoje pytanie. Bądźmy szczerzy: bardzo mało pytań jest niepowtarzalnych. Jeżeli wyczerpałeś wszystkie pozostałe możliwości, to może jednak rzeczywiście czas sformułować dobre pytanie.

Odtwarzalny przykład stanowi sedno prawidłowo przygotowanej prośby o pomoc. Jego pierwszym składnikiem są przykładowe dane. Dobrym sposobem jest ich symulowanie za pomocą kilku funkcji R. W poniższym przykładzie tworzymy ramkę danych o nazwie `example_df` zawierającą trzy kolumny, z których każda ma zdefiniowany inny typ danych:

```
set.seed(42)
n <- 4
example_df <- data.frame(
  wartości_rzeczywiste = rnorm(n),
  jakieś_litery = sample(LETTERS, n, replace = TRUE),
  liczby_całkowite = sample(1:10, n, replace = TRUE)
)

example_df
#>   wartości_rzeczywiste   jakieś_litery   liczby_całkowite
#> 1          1.371          R              10
#> 2         -0.565          S              3
#> 3          0.363          L              5
#> 4          0.633          S              10
```

Zauważ, że na początku kodu wykorzystujemy polecenie `set.seed`. W ten sposób po każdym uruchomieniu kodu będziemy otrzymywać zawsze te same wyniki. Wartość `n` oznacza liczbę rzędów przykładowych danych, które chcemy wygenerować. W celu zilustrowania problemu twórz zawsze jak najprostsze przykładowe dane.

Zamiast generowania danych możesz ewentualnie wykorzystać przykładowe dane dołączone do R. Na przykład zestaw danych `mtcars` zawiera ramkę danych przechowującą 32 rekordy z opisem różnych modeli samochodów:

```
data(mtcars)
head(mtcars)
#>   mpg   cyl  disp    hp  drat   wt  qsec vs  am  gear  carb
#> Mazda RX4      21.0   6  160   110   3.90  2.62 16.5  0   1    4     4
#> Mazda RX4 Wag  21.0   6  160   110   3.90  2.88 17.0  0   1    4     4
#> Datsun 710     22.8   4  108   93    3.85  2.32 18.6  1   1    4     1
```

² W menu rozwijalnym w prawym górnym rogu możesz zmienić język na polski — *przyp. tłum.*

```
#> Hornet 4 Drive      21.4      6      258  110      3.08  3.21  19.4  1  0      3      1
#> Hornet Sportabout  18.7      8      360  175      3.15  3.44  17.0  0  0      3      2
#> Valiant             18.1      6      225  105      2.76  3.46  20.2  1  0      3      1
```

Jeżeli przykład jest odtwarzalny wyłącznie przy użyciu Twoich danych, możesz wykorzystać funkcję `dput` do wstawienia tych danych w łańcuch znaków, który może być użyty w przykładowym kodzie. Zaprezentujemy to rozwiązanie na dwóch rzędach z zestawu danych `mtcars`:

```
dput(head(mtcars, 2))
#> structure(list(mpg = c(21, 21), cyl = c(6, 6), disp = c(160,
#> 160), hp = c(110, 110), drat = c(3.9, 3.9), wt = c(2.62, 2.875
#> ), qsec = c(16.46, 17.02), vs = c(0, 0), am = c(1, 1), gear = c(4,
#> 4), carb = c(4, 4)), row.names = c("Mazda RX4", "Mazda RX4 Wag"
#> ), class = "data.frame")
```

Możesz umieścić otrzymany obiekt `structure` bezpośrednio w zapytaniu:

```
example_df <- structure(list(mpg = c(21, 21), cyl = c(6, 6), disp = c(160,
160), hp = c(110, 110), drat = c(3.9, 3.9), wt = c(2.62, 2.875
), qsec = c(16.46, 17.02), vs = c(0, 0), am = c(1, 1), gear = c(4,
4), carb = c(4, 4)), row.names = c("Mazda RX4", "Mazda RX4 Wag"
), class = "data.frame")
```

```
example_df
#>           mpg      cyl  disp  hp   drat   wt  qsec  vs  am  gear  carb
#> Mazda RX4    21.0      6   160  110   3.90  2.62  16.5  0  1    4    4
#> Mazda RX4 Wag 21.0      6   160  110   3.90  2.88  17.0  0  1    4    4
```

Drugim składnikiem dobrego, odtwarzalnego przykładu jest przykładowy kod. Powinien być on możliwie prosty i ukazywać to, co próbujesz osiągnąć. *Unikaj* rozbudowanych bloków kodu zawierających najróżniejsze mechanizmy. Ogranicz przykład wyłącznie do niezbędnej partii kodu. Jeżeli korzystasz z jakichś pakietów, nie zapomnij dołączyć wywołania library na początku kodu. Ponadto nie dołączaj potencjalnie szkodliwych elementów, takich jak wyrażenie `rm(list=ls())`, powodujące usunięcie wszystkich obiektów R z pamięci. Wejść w skórę osoby próbującej Ci pomóc i zrozum, że dobrowolnie poświęca ona swój czas oraz być może uruchamia Twój kod na swoim komputerze roboczym.

W celu przetestowania przykładowego kodu otwórz nową sesję R i spróbuj go uruchomić. Po przygotowaniu kodu nadchodzi czas udzielenia dodatkowych informacji. Zwykłymi słowami opisz, co starasz się osiągnąć, jakie wypróbowałeś sposoby, a także zamieść swoje pytanie. Staraj się pozostać maksymalnie zwięzły. Podobnie jak w przypadku przykładowego kodu Twoim zadaniem jest jak najskuteczniejsza komunikacja z osobą, która może Ci pomóc. Być może pomocne okaże się podanie używanej wersji R, a także platformy (Windows, macOS, Linux). Informacje te możesz z łatwością uzyskać za pomocą polecenia `sessionInfo`.

Jeżeli planujesz zamieścić pytanie na liście dyskusyjnej, pamiętaj, że w rzeczywistości istnieje ich kilka rodzajów. Główną listą dyskusyjną przeznaczoną dla pytań ogólnych jest `R-help`. Dostępne są również listy dyskusyjne dla **grup użytkowników o podobnych zainteresowaniach** (ang. *special interest groups* — *SIG*), przeznaczone dla określonych dziedzin, takich jak genetyka, finanse, rozwijanie R, a nawet poszukiwanie pracy. Pełną listę grup `SIG` znajdziesz pod adresem <https://stat.ethz.ch/mailman/listinfo>. Jeżeli Twoje pytanie dotyczy określonego działu, uzyskasz lepszą odpowiedź, jeśli zamieścisz pytanie w odpowiedniej grupie. Podobnie jednak jak w przypadku listy `R-help` przejrzyj archiwum przed zamieszczeniem pytania.

Zobacz również

Zalecamy, by przed zamieszczeniem jakiegokolwiek pytania zapoznać się ze znakomitym artykułem *How to Ask Questions the Smart Way* (<http://www.catb.org/~esr/faqs/smart-questions.html>) autorstwa Erica Raymonda i Ricka Moena. Poważnie, przeczytaj go.

W serwisie Stack Overflow znajdziesz świetny wpis opisujący między innymi proces przygotowywania odtwarzalnego przykładu. Jest dostępny na stronie <https://stackoverflow.com/questions/5963269/how-to-make-a-great-r-reproducible-example>.

Jenny Bryan stworzyła doskonały pakiet R o nazwie *reprex*, ułatwiający stworzenie dobrego odtwarzalnego przykładu, a także zawierający dodatkowe funkcje pomocne w tworzeniu tekstu formatowanego w języku Markdown dla takich serwisów, jak Stack Overflow. Pakiet ten znajdziesz w serwisie *GitHub* (<https://github.com/tidyverse/reprex>).

PROGRAM PARTNERSKI

— GRUPY HELION —

- 
1. ZAREJESTRUJ SIĘ
 2. PREZENTUJ KSIĄŻKI
 3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

R: błyskawicznie osiągniesz znakomite wyniki!

Język R jest potężnym narzędziem używanym w statystyce, przetwarzaniu grafiki i programowaniu statystycznym; stanowi konkurencję dla komercyjnych systemów do obliczeń tego rodzaju. Zawiera wszystkie narzędzia, których potrzebują statystycy. Równocześnie jest to specyficzny język, przez co jego użytkowanie może sprawiać problemy. Zarówno proste, jak i złożone zadania są łatwe do wykonania, jeśli tylko wiadomo, w jaki sposób je zrobić. Jeżeli jednak trzeba stopniowo dochodzić do właściwego rozwiązania, może to kosztować sporo cierpliwości i zniechęcać.

Oto zbiór 275 receptur instruktażowych, z których każda pomaga w rozwiązaniu konkretnego problemu. Wszystkie zostały starannie przetestowane i wielokrotnie dowodziły swojej przydatności. Każda receptura została poprzedzona krótkim wprowadzeniem i omówieniem zastosowanych mechanizmów działania. Nie jest to klasyczny podręcznik programowania, jednak z pewnością przyspieszy naukę praktycznego wykorzystania możliwości R. Jeśli masz już pewne doświadczenie z tym językiem, odświeżysz swoją wiedzę i uzyskasz szerszą perspektywę. Wśród receptur znajdziesz obejmujące szeroki zakres zadania — od podstawowych operacji na danych wejściowych i wyjściowych, poprzez statystykę ogólną, aż po grafikę i regresję liniową. Dowiesz się również, jak wykorzystać język R do wizualizacji danych za pomocą ciekawych wykresów graficznych.

W książce między innymi:

- przygotowywanie danych wejściowych i upraszczanie danych wyjściowych
- macierze, listy, wektory czynniki, ramki danych
- testy statystyczne, przedziały ufności, prawdopodobieństwa
- modele statystyczne z wykorzystaniem regresji liniowej i analizy wariancji
- stosowanie zaawansowanych technik statystycznych

J.D. Long jest programistą i założycielem stowarzyszenia Chicago R User Group. Pasjonuje go tworzenie kodu w językach Python i R oraz usługi AWS. Jest znany z obrazowych metafor, którymi okrasza swoje wystąpienia na konferencjach poświęconych językowi R.

Paul Teetor jest programistą statystycznym. Specjalizuje się w analizie i inżynierii oprogramowania w sektorach zarządzania inwestycyjnego, handlu papierami wartościowymi i zarządzania ryzykiem. Współpracuje z funduszami hedgingowymi, maklerami i zarządcami portfeli na terenie Chicago.

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

Sprawdź nasze szkolenia!
SZKOLENIA
AKADEMIA IT & BUSINESS
HELIONSZKOLENIA.PL

KOD KORZYŚCI
Sięgnij po więcej! ▶



ISBN 978-83-283-6288-8



9 788328 362888

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 89,00 zł