STATISTICAL TESTING OF ACTIVITY CLIFFS

by

Sarah Josephine Aurit

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Doctor of Philosophy

Major: Statistics

Under the Supervision of Souparno Ghosh

Lincoln, Nebraska

December, 2023

STATISTICAL TESTING OF ACTIVITY CLIFFS

Sarah Josephine Aurit, Ph.D.

University of Nebraska, 2023

Adviser: Souparno Ghosh

Here is my abstract. *(350 word limit)*

DEDICATION

Dedicated to...

ACKNOWLEDGMENTS

Thank you to all my people!

Table of Contents

List of Figures

List of Tables

Chapter 1

Literature Review

1.1 Motivation and Background

The journey of discovering effective cancer drugs can be long and expensive. The elements of time and cost weigh against the race to improve survival and the quality of life for cancer patients. Researchers typically start testing drug compounds on cancer cell lines within a laboratory setting first before testing on animals or humans. To narrow the search field, drugs are sometimes pursued that are similar to ones previously found to be successful in cancer treatment. That said, it is possible for the new drug to be much less effective, which is defined as an activity cliff (Cruz-Monteagudo et al., 2014).

According to HU et al. (2017), compounds that are structurally similar yet have large potency differences are identified as producing activity cliffs upon an activity landscape. Additionally, an activity cliff is potentially produced within the context of the following four key aspects: 1) we are considering and comparing a pair of compounds, 2) both compounds are active against the same target, 3) the two compounds are similar in chemical structure, and 4) there is a difference in potency.

Focusing on the aspect of compound similarity via cheminformatics, formal investigation is completed by looking at similarity measurements. There are multiple approaches regarding the quantification of similarity between two compounds. One popular selection is the Tanimoto coefficient.

The Tanimoto coefficient, a measure of similarity, is determined by the quantification of molecular attributes that are similar between two molecular compounds. Essentially, it is a ratio of the number of similar entities divided by the total number of molecular attributes. "Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?"

Analogously, matched molecular pairs (MMP) analysis encompasses another measurement of similarity. Essentially, two molecules differ by one "well-defined, structural" chemical transformation. https://pubs-acs-org.libproxy.unl.edu/doi/pdf/10.1021/jm200452d. MMPs were initially used as a special case of a quantitative structure-activity relationship (QSAR), which relates molecular compounds to the response of potency or other biochemical properties.

QSAR methodology can be essential in determining the relationships between chemical structures and biochemical activities, and it is typical that the methodology incorporates supervised machine learning techniques. Compounds are first represented by a numerical entity such as a two-dimensional structural formula that promulgates atom connectivity via chemical bonds. The structural formula then is related to bioactivities. It is not uncommon for there to be violations of QSAR methodology given the presence of an activity cliff (AC). Reference: QSAR modeling where have you been where are you going

1.2 GDSC Dataset

The Genomics of Drug Sensitivity in Cancer (GDSC) database was instituted by the UK's Wellcome Sanger Institute's Cancer Genome Project and Massachusetts General Hospital's Center for Molecular Therapeutics. It is a repository and compilation of anti-cancer drug response information linked to cancer cell lines of multiple primary cancer sites. The response is quantified by IC_{50}, a metric that reveals how much substance or dose is necessary to inhibit a biological component by 50%. Also, AUC values were calculated from dose-response curves attributed to nine dose levels where drug response is measured 72 hours post-administration of drug to cancer cell lines.

We focus on breast cancer data, which is comprised by 208 drugs. Disregarding missing data, approximately 900 drug characteristics were identified for each drug alongside IC_{50} values.

1.3 Isomap

Dimension reduction is a critical element in the investigation of breast cancer drugs because of the high dimensionality of drug characteristics relative to the number of drugs. Classical techniques such as principal component analysis (PCA) and multidimensional scaling (MDS) rely on the assumptions that fail within the context of non-linear structures that are intrinsic to some datasets. An alternative dimension reduction technique is that of isometric feature mapping (isomap). https://wearables.cc.gatech.edu/paper_of_week/isomap.pdf The algorithm is comprised of three steps. First, Euclidean distances are calculated at each point for all other points that are either within a specific radius or are considered a K nearest neighbors (kNN), which are utilized to construct a distance matrix. Next, geodesic distances for all pairwise distances using an algorithm that calculates the shortest path, which includes Dijkstra's (shortest pathway from one node to another while navigating intermediary nodes) or Floyd-Warshall algorithm. Finally, MDS is applied to these geodesic distances.

LMNN

Plot of distances before and after LMNN

1.4 Stationarity for Irregularly Spaced Data

We can consider the investigation of drug efficacy over molecular space within the context of spatial statistics. Commonly, assumptions surrounding spatial data include that of second-order stationarity to make accurate predictions. We consider drug efficacy as a random variable for each known position within molecular space. The first requirement for second-order stationarity is that the expectation and variance of drug efficacy is constant over the entire domain of molecular space. The second requirement is that the covariance between any two observations only relies on the distance between the two observations.

$$\mu = E(Z(s)) = \sum P_i x_i \tag{1.1}$$

Chapter 2

R Markdown Basics

Here is a brief introduction into using *R Markdown*. *Markdown* is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. *R Markdown* provides the flexibility of *Markdown* with the implementation of **R** input and output. For more details on using *R Markdown* see `http://rmarkdown.rstudio.com`.

Be careful with your spacing in *Markdown* documents. While whitespace largely is ignored, it does at times give *Markdown* signals as to how to proceed. As a habit, try to keep everything left aligned whenever possible, especially as you type a new paragraph. In other words, there is no need to indent basic text in the Rmd document (in fact, it might cause your text to do funny things if you do).

2.1 Lists

It's easy to create a list. It can be unordered like

- Item 1
- Item 2

or it can be ordered like

1. Item 1
2. Item 2

Notice that I intentionally mislabeled Item 2 as number 4. *Markdown* automatically figures this out! You can put any numbers in the list and it will create the list. Check it out below.

To create a sublist, just indent the values a bit (at least four spaces or a tab). (Here's one case where indentation is key!)

1. Item 1
2. Item 2
3. Item 3
    - Item 3a
    - Item 3b

2.2 Line breaks

Make sure to add white space between lines if you'd like to start a new paragraph. Look at what happens below in the outputted document if you don't:

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph. This should be a new paragraph.

Now for the correct way:

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph.

This should be a new paragraph.

2.3 R chunks

When you click the **Knit** button above a document will be generated that includes both content as well as the output of any embedded **R** code chunks within the document. You can embed an **R** code chunk like this (`cars` is a built-in **R** dataset):

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

2.4 Inline code

If you'd like to put the results of your analysis directly into your discussion, add inline code like this:

> The `cos` of 2π is 1.

Another example would be the direct calculation of the standard deviation:

> The standard deviation of `speed` in `cars` is 5.2876444.

One last neat feature is the use of the `ifelse` conditional statement which can be used to output text depending on the result of an **R** calculation:

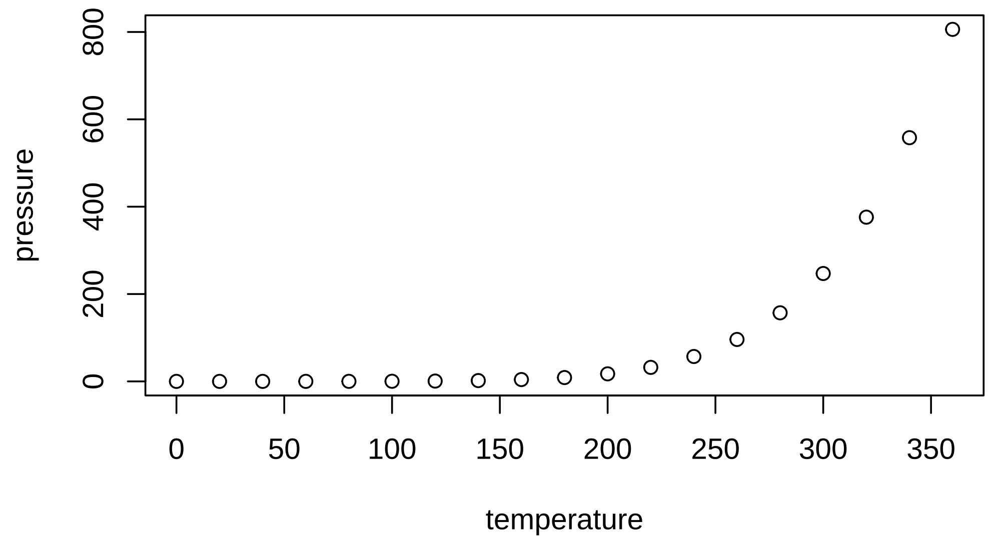
> The standard deviation is less than 6.

Note the use of `>` here, which signifies a quotation environment that will be indented.

As you see with `$2 \pi$` above, mathematics can be added by surrounding the mathematical text with dollar signs. More examples of this are in Mathematics and Science if you uncomment the code in Math.

2.5 Including plots

You can also embed plots. For example, here is a way to use the base **R** graphics package to produce a plot using the built-in `pressure` dataset:



Note that the echo=FALSE parameter was added to the code chunk to prevent printing of the **R** code that generated the plot. There are plenty of other ways to add chunk options. More information is available at http://yihui.name/knitr/options/.

Another useful chunk option is the setting of cache=TRUE as you see here. If document rendering becomes time consuming due to long computations or plots that are expensive to generate you can use knitr caching to improve performance. Later in this file, you'll see a way to reference plots created in **R** or external figures.

2.6 Loading and exploring data

Included in this template is a file called flights.csv. This file includes a subset of the larger dataset of information about all flights that departed from Seattle and Portland in 2014. More information about this dataset and its **R** package is available at http://github.com/ismayc/pnwflights14. This subset includes only Portland flights and only rows that were complete with no missing values. Merges were also done with the airports and airlines data sets in the pnwflights14 package to get more descriptive airport and airline names.

We can load in this data set using the following command:

The data is now stored in the data frame called flights in **R**. To get a better feel for the variables included in this dataset we can use a variety of functions. Here we can see the dimensions (rows by columns) and also the names of the columns.

```
## [1] 52808    16
```

```
##  [1] "month"        "day"          "dep_time"
##  [4] "dep_delay"    "arr_time"     "arr_delay"
##  [7] "carrier"      "tailnum"      "flight"
## [10] "dest"         "air_time"     "distance"
## [13] "hour"         "minute"       "carrier_name"
## [16] "dest_name"
```

Another good idea is to take a look at the dataset in table form. With this dataset having more than 50,000 rows, we won't explicitly show the results of the command here. I recommend you enter the command into the Console ***after*** you have run the **R** chunks above to load the data into **R**.

While not required, it is highly recommended you use the `dplyr` package to manipulate and summarize your data set as needed. It uses a syntax that is easy to understand using chaining operations. Below I've created a few examples of using `dplyr` to get information about the Portland flights in 2014. You will also see the use of the `ggplot2` package, which produces beautiful, high-quality academic visuals.

We begin by checking to ensure that needed packages are installed and then we load them into our current working environment:

The example we show here does the following:

- Selects only the `carrier_name` and `arr_delay` from the `flights` dataset and then assigns this subset to a new variable called `flights2`.
- Using `flights2`, we determine the largest arrival delay for each of the carriers.

A useful function in the `knitr` package for making nice tables in *R Markdown* is called `kable`. It is much easier to use than manually entering values into a table by copying and pasting values into Excel or LaTeX. This again goes to show how nice reproducible documents can be! (Note the use of `results="asis"`, which will produce the table instead of the code to create the table.) The `caption.short` argument is used to include a shorter title to appear in the List of Tables.

Table 2.1: Maximum Delays by Airline

Airline	Max Arrival Delay
Alaska Airlines Inc.	338
American Airlines Inc.	1539
Delta Air Lines Inc.	651
Frontier Airlines Inc.	575
Hawaiian Airlines Inc.	407
JetBlue Airways	273
SkyWest Airlines Inc.	421
Southwest Airlines Co.	694
United Air Lines Inc.	472

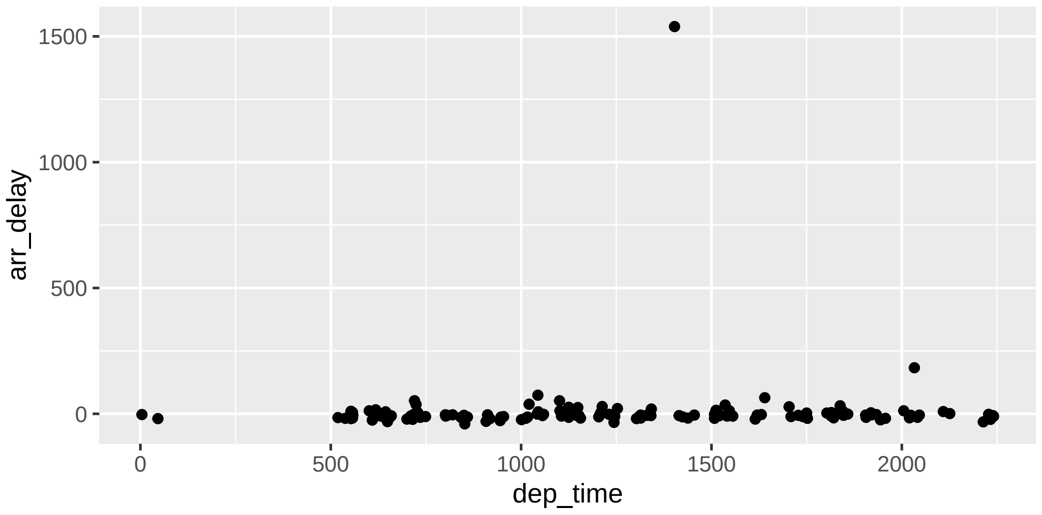
US Airways Inc.	347
Virgin America	366

The last two options make the table a little easier-to-read.

We can further look into the properties of the largest value here for American Airlines Inc. To do so, we can isolate the row corresponding to the arrival delay of 1539 minutes for American in our original `flights` dataset.

```
##   dep_time dep_delay arr_time tailnum flight dest air_time
## 1     1403      1553     1934  N595AA   1568  DFW      182
##   distance
## 1     1616
```

We see that the flight occurred on March 3rd and departed a little after 2 PM on its way to Dallas/Fort Worth. Lastly, we show how we can visualize the arrival delay of all departing flights from Portland on March 3rd against time of departure.



2.7 Additional resources

- *Markdown* Cheatsheet - https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet
- *R Markdown* Reference Guide - https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf
- Introduction to `dplyr` - https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html
- `ggplot2` Documentation - http://docs.ggplot2.org/current/

Chapter 3

Mathematics and Science

3.1 Math

TEX is the best way to typeset mathematics. Donald Knuth designed TEX when he got frustrated at how long it was taking the typesetters to finish his book, which contained a lot of mathematics. One nice feature of *R Markdown* is its ability to read LaTeX code directly.

If you are doing a thesis that will involve lots of math, you will want to read the following section which has been commented out. If you're not going to use math, skip over or delete this next commented section.

Exponent or Superscript: O^{-}

Subscript: CH_4

To stack numbers or letters as in Fe_2^{2+}, the subscript is defined first, and then the superscript is defined.

Bullet: $\mathrm{CuCl} \bullet 7\mathrm{H}_2\mathrm{O}$

Delta: Δ

Reaction Arrows: $\longrightarrow$ or $\xrightarrow{solution}$

Resonance Arrows: $\leftrightarrow$

Reversible Reaction Arrows: $\rightleftharpoons$

3.1.1 Typesetting reactions

You may wish to put your reaction in an equation environment, which means that LaTeX will place the reaction where it fits and will number the equations for you.

$$C_6H_{12}O_6 + 6O_2 \longrightarrow 6CO_2 + 6H_2O \tag{3.1}$$

We can reference this combustion of glucose reaction via Equation (3.1).

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In the main Rmd file

```
# This chunk ensures that the huskydown package is
# installed and loaded. This huskydown package includes
# the template files for the thesis.
if(!require(devtools))
  install.packages("devtools",
                   repos = "http://cran.rstudio.com")
if(!require(huskydown))
  devtools::install_github(
    "benmarwick/huskydown"
  )
library(huskydown)
library(tidyverse)
library(knitr)
library(ggplot2)
```

In Chapter ??:

Appendix B

The Second Appendix, for Fun

Colophon

This document is set in EB Garamond, Source Code Pro and Lato. The body text is set at 11pt with *lmr*.

It was written in R Markdown and $\LaTeX$, and rendered into PDF using huskydown and bookdown.

This document was typeset using the XeTeX typesetting system, and the University of Washington Thesis class class created by Jim Fox. Under the hood, the University of Washington Thesis LaTeX template is used to ensure that documents conform precisely to submission standards. Other elements of the document formatting source code have been taken from the Latex, Knitr, and RMarkdown templates for UC Berkeley's graduate thesis, and Dissertate: a LaTeX dissertation template to support the production and typesetting of a PhD dissertation at Harvard, Princeton, and NYU

The source files for this thesis, along with all the data files, have been organised into an R package, xxx, which is available at `https://github.com/xxx/xxx`. A hard copy of the thesis can be found in the University of Washington library.

This version of the thesis was generated on 2023-02-13 10:22:54. The repository is currently at this commit:

The computational environment that was used to generate this version is as follows:

```
## - Session info ----------------------------------------------
##  setting  value
##  version  R version 4.2.1 (2022-06-23 ucrt)
##  os       Windows 10 x64 (build 22000)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.utf8
##  ctype    English_United States.utf8
##  tz       America/Chicago
##  date     2023-02-13
##  pandoc   2.18 @ C:/Program Files/RStudio/bin/quarto/bin/tools/ (via rmarkdown)
##
## - Packages ----------------------------------------------------
##  package      * version date (UTC) lib source
##  assertthat     0.2.1   2019-03-21 [1] CRAN (R 4.2.2)
##  backports      1.4.1   2021-12-13 [1] CRAN (R 4.2.0)
##  bookdown       0.31.2  2023-01-09 [1] Github (rstudio/bookdown@604939c)
##  broom          1.0.2   2022-12-15 [1] CRAN (R 4.2.2)
##  cachem         1.0.6   2021-08-19 [1] CRAN (R 4.2.2)
##  callr          3.7.3   2022-11-02 [1] CRAN (R 4.2.2)
##  cellranger     1.1.0   2016-07-27 [1] CRAN (R 4.2.2)
##  cli            3.4.1   2022-09-23 [1] CRAN (R 4.2.2)
##  colorspace     2.0-3   2022-02-21 [1] CRAN (R 4.2.2)
##  crayon         1.5.2   2022-09-29 [1] CRAN (R 4.2.2)
##  DBI            1.1.3   2022-06-18 [1] CRAN (R 4.2.2)
##  dbplyr         2.2.1   2022-06-27 [1] CRAN (R 4.2.2)
##  devtools     * 2.4.5   2022-10-11 [1] CRAN (R 4.2.2)
##  digest          0.6.31  2022-12-11 [1] CRAN (R 4.2.2)
##  dplyr         * 1.0.10  2022-09-01 [1] CRAN (R 4.2.2)
##  ellipsis        0.3.2   2021-04-29 [1] CRAN (R 4.2.2)
##  evaluate        0.19    2022-12-13 [1] CRAN (R 4.2.2)
##  fansi           1.0.3   2022-03-24 [1] CRAN (R 4.2.2)
##  farver          2.1.1   2022-07-06 [1] CRAN (R 4.2.2)
##  fastmap         1.1.0   2021-01-25 [1] CRAN (R 4.2.2)
##  forcats       * 0.5.2   2022-08-19 [1] CRAN (R 4.2.2)
##  fs              1.5.2   2021-12-08 [1] CRAN (R 4.2.2)
##  gargle          1.2.1   2022-09-08 [1] CRAN (R 4.2.2)
##  generics        0.1.3   2022-07-05 [1] CRAN (R 4.2.2)
##  ggplot2       * 3.4.0   2022-11-04 [1] CRAN (R 4.2.2)
##  git2r           0.30.1  2022-03-16 [1] CRAN (R 4.2.2)
##  glue            1.6.2   2022-02-24 [1] CRAN (R 4.2.2)
##  googledrive     2.0.0   2021-07-08 [1] CRAN (R 4.2.2)
##  googlesheets4   1.0.1   2022-08-13 [1] CRAN (R 4.2.2)
##  gtable          0.3.1   2022-09-01 [1] CRAN (R 4.2.2)
##  haven           2.5.1   2022-08-22 [1] CRAN (R 4.2.2)
##  hms             1.1.2   2022-08-19 [1] CRAN (R 4.2.2)
##  htmltools       0.5.4   2022-12-07 [1] CRAN (R 4.2.2)
##  htmlwidgets     1.6.1   2023-01-07 [1] CRAN (R 4.2.2)
##  httpuv          1.6.7   2022-12-14 [1] CRAN (R 4.2.2)
##  httr            1.4.4   2022-08-17 [1] CRAN (R 4.2.2)
##  huskydown     * 0.0.5   2022-12-20 [1] Github (benmarwick/huskydown@addb48e)
##  jsonlite        1.8.4   2022-12-06 [1] CRAN (R 4.2.2)
##  knitr         * 1.41    2022-11-18 [1] CRAN (R 4.2.2)
##  labeling        0.4.2   2020-10-20 [1] CRAN (R 4.2.0)
##  later           1.3.0   2021-08-18 [1] CRAN (R 4.2.2)
##  lifecycle       1.0.3   2022-10-07 [1] CRAN (R 4.2.2)
##  lubridate       1.9.0   2022-11-06 [1] CRAN (R 4.2.2)
##  magrittr        2.0.3   2022-03-30 [1] CRAN (R 4.2.2)
## memoise         2.0.1   2021-11-26 [1] CRAN (R 4.2.2)
## mime            0.12    2021-09-28 [1] CRAN (R 4.2.0)
## miniUI          0.1.1.1 2018-05-18 [1] CRAN (R 4.2.2)
## modelr          0.1.10  2022-11-11 [1] CRAN (R 4.2.2)
## munsell         0.5.0   2018-06-12 [1] CRAN (R 4.2.2)
## pillar          1.8.1   2022-08-19 [1] CRAN (R 4.2.2)
## pkgbuild        1.4.0   2022-11-27 [1] CRAN (R 4.2.2)
## pkgconfig       2.0.3   2019-09-22 [1] CRAN (R 4.2.2)
## pkgload         1.3.2   2022-11-16 [1] CRAN (R 4.2.2)
## prettyunits     1.1.1   2020-01-24 [1] CRAN (R 4.2.2)
## processx        3.8.0   2022-10-26 [1] CRAN (R 4.2.2)
## profvis         0.3.7   2020-11-02 [1] CRAN (R 4.2.2)
## promises        1.2.0.1 2021-02-11 [1] CRAN (R 4.2.2)
## ps              1.7.2   2022-10-26 [1] CRAN (R 4.2.2)
## purrr         * 1.0.0   2022-12-20 [1] CRAN (R 4.2.2)
## R6              2.5.1   2021-08-19 [1] CRAN (R 4.2.2)
## Rcpp            1.0.9   2022-07-08 [1] CRAN (R 4.2.2)
## readr         * 2.1.3   2022-10-01 [1] CRAN (R 4.2.2)
## readxl          1.4.1   2022-08-17 [1] CRAN (R 4.2.2)
## remotes         2.4.2   2021-11-30 [1] CRAN (R 4.2.2)
## reprex          2.0.2   2022-08-17 [1] CRAN (R 4.2.2)
## rlang           1.0.6   2022-09-24 [1] CRAN (R 4.2.2)
## rmarkdown       2.19    2022-12-15 [1] CRAN (R 4.2.2)
## rstudioapi      0.14    2022-08-22 [1] CRAN (R 4.2.2)
## rvest           1.0.3   2022-08-19 [1] CRAN (R 4.2.2)
## scales          1.2.1   2022-08-20 [1] CRAN (R 4.2.2)
## sessioninfo     1.2.2   2021-12-06 [1] CRAN (R 4.2.2)
## shiny           1.7.4   2022-12-15 [1] CRAN (R 4.2.2)
## stringi         1.7.8   2022-07-11 [1] CRAN (R 4.2.1)
## stringr       * 1.5.0   2022-12-02 [1] CRAN (R 4.2.2)
## tibble        * 3.1.8   2022-07-22 [1] CRAN (R 4.2.2)
##  tidyr        * 1.2.1   2022-09-08 [1] CRAN (R 4.2.2)
##  tidyselect     1.2.0   2022-10-10 [1] CRAN (R 4.2.2)
##  tidyverse    * 1.3.2   2022-07-18 [1] CRAN (R 4.2.2)
##  timechange     0.2.0   2023-01-11 [1] CRAN (R 4.2.2)
##  tzdb           0.3.0   2022-03-28 [1] CRAN (R 4.2.2)
##  urlchecker     1.0.1   2021-11-30 [1] CRAN (R 4.2.2)
##  usethis      * 2.1.6   2022-05-25 [1] CRAN (R 4.2.2)
##  utf8           1.2.2   2021-07-24 [1] CRAN (R 4.2.2)
##  vctrs          0.5.1   2022-11-16 [1] CRAN (R 4.2.2)
##  withr          2.5.0   2022-03-03 [1] CRAN (R 4.2.2)
##  xfun           0.35    2022-11-16 [1] CRAN (R 4.2.2)
##  xml2           1.3.3   2021-11-30 [1] CRAN (R 4.2.2)
##  xtable         1.8-4   2019-04-21 [1] CRAN (R 4.2.2)
##  yaml           2.3.6   2022-10-18 [1] CRAN (R 4.2.2)
##
##  [1] C:/Program Files/R/R-4.2.1/library
##
## ----------------------------------------------------------
```

References

Angel, E. (2000). *Interactive computer graphics : A top-down approach with OpenGL*. Boston, MA: Addison Wesley Longman.

Angel, E. (2001a). *Batch-file computer graphics : A bottom-up approach with QuickTime*. Boston, MA: Wesley Addison Longman.

Angel, E. (2001b). *Test second book by angel*. Boston, MA: Wesley Addison Longman.