APSTA-GE-2352 Practicum in Applied Statistics: Statistical Computing

Klint Kanopka

Fall 2024-2025

1 Course Description

This course will introduce the student to statistical programming and simulation using R. Students will first understand variables, data structures, program flow (e.g., conditional execution, looping) and functional programming, then apply these skills to answer interesting statistical questions involving the comparison of groups. Most statistical analysis will be motivated via simulations, rather than mathematical theory. The course content (programming and data analysis) requires significant outside reading and programming.

1.1 Additional Notes

This course is designed to treat `R` *as a programming language*[2]. Programming skills are taught through the analysis, design, and implementation of algorithms, with special attention paid to modern statistical algorithms (e.g., gradient descent, k-Means clustering, Markov Chain Monte Carlo). Time will also be spent on optimization techniques (e.g., vectorization and parallelization) useful for statistical analysis. Everything done in this course will be implemented in base `R`. Notable exceptions include the use of the `ggplot2` library for visualization and libraries like `MASS` for sampling from distributions[4, 3]. Students must implement algorithms, write unit tests, and debug from scratch using the tools available in base `R`.

1.2 Prerequisites

This course assumes some experience with the `R` programming language and probability. You may find prior experience with computer science fundamentals and `ggplot2` to be helpful. No previous exposure to the design and analysis of algorithms is assumed.

2 Student Learning Outcomes

During the course of the semester:

1. Students will implement literate programming to produce coherent and reproducible code.
2. Students will verify code function through the implementation of unit tests.
3. Students will write more efficient code by applying optimization techniques (e.g., vectorization, parallelization).
4. Students will solve problems by implementing and modifying algorithms.
5. Students will answer statistical questions by implementing Monte Carlo simulations.

3 Course Logistics

3.1 Instructors

- **Klint Kanopka** - Instructor, Lecture
  - klint.kanopka@nyu.edu
  - Office Hours: Tuesdays, 11a-12p
  - Appointments also available
- **Sophia Deng** - TA, Lab Section 1
  - sd5718@nyu.edu
  - Office Hours: Mondays, 1-2p
- **Victoria Zhang** - TA, Lab Section 2
  - xz2661@nyu.edu
  - Office Hours: Tuesdays, 2:30p-3:30p

3.2 Meeting Times and Locations

Attendance at in-person lectures is required. Students are to attend their assigned lab section unless prior permission is received from the teaching team. Attendance is taken at lab.

- **Lecture:** Thursdays, 4.55p-6.35p @ 194 Mercer St, Rm 208
- **Lab Section 1:** Wednesdays, 3.45p-4.35p @ Bobst Library, Rm LL143
- **Lab Section 2:** Wednesdays, 5.55p-6.45p @ Bobst Library, Rm LL147

3.3 Required and Recommended Reading

- Required: *Introduction to Algorithms, Fourth Edition* [1]
- Recommended: *Comparing Groups: Randomization and Bootstrap Methods Using R* [5]

Both texts are available online at no cost through the NYU library.

4 Grading

This is a three-credit class. In exceptional circumstances with instructor approval, it may be taken for one credit. Your graded output for this course consists of eight (8) equally weighted problem sets and your (physical) attendance and participation in your weekly lab section. Each problem set is typically worth 100 points. You will have five (5) participation points. Each unexcused lab absence results in a deduction of one (1) point. Note that this amounts to 20% of your available lab participation points. Table 1 contains the proportion of the final grade coming from each assignment category. Table 2 contains the transformation from the weighted proportion of earned points to letter grades. When transforming to a letter grade category, weighted proportions of earned points are **not** rounded.

Category	p
Problem Sets	0.9
Participation (Lab)	0.1

Table 1: Final grade weighting scheme

	G^-	G	G^+
A	$[.895, .945)$	$[.945, 1]$	—
B	$[.795, .825)$	$[.825, .865)$	$[.865, .895)$
C	$[.695, .825)$	$[.725, .765)$	$[.765, .795)$
D	$[.600, .640)$	$[.640, .670)$	$[.670, .695)$
F	—	$[0, .600)$	—

Table 2: Grading Scale

4.1 Problem Sets

The course contains eight equally weighted problem sets (PS0-PS7).

4.1.1 Submission Guidelines

All problem sets should be submitted on Brightspace using Quarto (preferred) or Rmarkdown and adhere to the following conventions:

1. **Submit assignments by the deadline.** Assignment deadlines are typically **Thursdays before lecture begins** (i.e., 4.54p). Late assignments will receive a 10% penalty per day.

2. **Name your files correctly.** For each assignment, name your files using the convention: `LastName_FirstInitial_PS#.ext`

3. **Put your name and the assignment in text of the file.** These should occupy the author and title fields, respectively.

4. **Submit both a source file and a knitted file.** Source files have the extension `.qmd` or `.Rmd`. Files should be knitted to `.pdf`. As an example for the first homework, I would submit two files:

   - `Kanopka_K_PS0.qmd`
   - `Kanopka_K_PS0.pdf`

   Note that you will need a function LaTeXinstallation on your machine in order to knit to `.pdf`.

5. **Your source file should run without modification.** If you load a file, please use relative paths. Do not load libraries unless explicitly requested in an assignment.

6. **Do not use `install.packages()` calls in your assignments.** Please install packages locally and then load them in your assignment source code.

4.2 Attendance and Participation

In-person attendance at lecture and lab is a core requirement of the course, and thus mandatory. Satisfactory participation in lab looks like arriving on time and honestly engaging with planned lab activities. Each unexcused absence or failure to participate in lab will result in a deduction of 20% of available participation points. To receive an excused absence, please contact me in advance and cc your lab instructor on any communication.

4.3 Extra Credit

This course allows extra credit to be earned in any of the following three ways:[1]

1. The first person to report typographical errors or mistakes in any of my course materials will receive one (1) extra credit point per error (I expect there to be a lot of them). These should be reported in your course section's Slack channel

2. "Valuable suggestions" will also result in one (1) extra credit point and acknowledgment in a footnote

[1] This policy is inspired by the Knuth Reward Check

3. Reporting a coding error with a solution or a valuable suggestion that requires additional labor (code snippets, a figure, a dataset, etc.) will be worth more, up to a maximum of 2^8 points. This amount will be scaled based on the nature and duration of required labor

5 Course Policies

5.1 Academic Integrity

I take academic integrity incredibly seriously. Please review NYU Steinhardt's Academic Policies and Procedures for more information on specific policies, the disciplinary process, and sanctions.

5.2 Collaboration

I **strongly** encourage students to form study groups. Students may discuss and work on problem sets in groups. Each student must (1) report at the top of each question what other students they consulted with, (2) write their code and solution independently, and (3) understand their work well enough in order to reconstruct it entirely on their own.

5.3 AI Tool Policy

All assignments should be your own original work. The use of generative AI tools in this course is explicitly not allowed. Not all AI tools are generative (and thus banned), however. Here are some guiding examples to consider when using various AI-based tools. If you have a question about a specific tool or use case, please reach out to me.

Examples of AI that are okay to use in the course include:

- Grammar and spelling checkers (e.g., Grammarly)
- Transcription or translation tools (e.g., OtterAI)

Examples of AI that are **not** okay to use in this course:

- Text generators (e.g., Chat GPT, Bard, Ernie Bot, LLaMa, Bing chat)
- Audio, image, or video generators where the output is meant to represent research, data, or code
- Data analyzers (e.g., Chartify, Rows.ai)
- Code generators (e.g., Copilot)

5.4 Students with Disabilities

Students with physical or learning disabilities are required to register with the Moses Center for Student Accessibility, 726 Broadway, 2nd Floor, (212-998-4980 and online at `http://www.nyu.edu/csd`). They must present a letter from the Center to the instructor at the start of the semester to be considered for appropriate accommodation.

5.5 Mental Health Statement

If you are experiencing undue personal and/or academic stress during the semester that may be interfering with your ability to perform academically, the NYU Wellness Exchange (212-443-9999) offers a range of services to assist and support you. I am available to speak with you about stresses related to your work in my course, and I can assist you in connecting with the Wellness Exchange. Additionally, if you anticipate any challenges with completing the assignments, readings, exams and other work required in this course, I encourage you to register with the Moses Center (212 998 4980) in advance so that you may be granted the proper academic accommodations.

5.6 Inclusion

NYU values an inclusive and equitable environment for all our students. I hope to foster a sense of community in this class and consider it a place where individuals of all backgrounds, beliefs, ethnicities, national origins, gender identities, sexual orientations, religious and political affiliations, and abilities will be treated with respect. I intend that all students' learning needs be addressed in and out of class and that the diversity students bring to this class be viewed as a resource and strength. Please feel free to speak with me if this standard is not being upheld.

6 Course Calendar

Table 3 contains topics, assignments, and deadlines. Note that this is subject to change.

Week	Lab (Wednesday)	Lecture (Thursday)	Assignments
1: 9.5	Topics: • Introduction to R and RStudio • Working with Quarto documents • Literate programming	Topics: • Data types • Vector and matrix arithmetic • Random number generation • Environments • Visualizing distributions	Released: • PS0 • PS1
2: 9.12	Topics: • Visualization of two or more variables • `ggplot2` practice	Topics: • More environments • Writing functions • Distances • (Dis)Similarity Metrics • Unit Testing	Due: • PS0
3: 9.19	Topics: • More distances • Writing test functions • Introduction to logical expressions	Topics: • Logical expressions • Conditional statements • `for` loops • Indexing • Sorting algorithms	Due: • PS1 Released: • PS2
4: 9.26	Topics: • `for` loop practice • More sorting algorithms • Comparing runtimes	Topics: • `while` loops • The k-Means clustering algorithm • Debugging	
5: 10.3	Topics: • Extending k-Means	Topics: • Randomization • Monte Carlo Methods • Writing simulations	Due: • PS2 Released: • PS3

6: 10.10	Topics: • Solving problems using Monte Carlo methods	Topics: • Advanced matrix computation • Dimensionality reduction • Principal component analysis	
7: 10.17	Topics: • Visualization with PCA	Topics: • Numerical optimization • Writing loss functions • Gradient descent • Regularization • Using `optim()`	Due: • PS3 Released: • PS4
8: 10.24	Topics: • Write your own regularized regression function • Debugging	Topics: • Recursive algorithms • Divide-and-conquer strategies • Even more sorting	
9: 10.31	Topics: • Applications of recursion	Topics: • Randomized algorithms • Searching and sorting • Markov Chain Monte Carlo	Due: • PS4 Released: • PS5
10: 11.7	Topics: • MCMC and stationary distributions • Problem solving with MCMC	Topics: • Advanced linear algebra techniques for computation • Eigenvalues, eigenvectors, and eigendecomposition • A little more PCA • The Singular Value Decomposition	

11: 11.14	Topics: • SVD for matrix completion • Imputation	Topics: • Greedy Algorithms • Scheduling problems • The knapsack problem • Dynamic Programming	Due: • PS5 Released: • PS6
12: 11.21	Topics: • Applying dynamic programming • Comparing approaches	Topics: • Linear Programming • Convex Programming	
13: 12.5	Topics: • Solving problems with linear programming • Convex optimization	Topics: • Techniques for efficient numeric computation • Parallelization	Due: • PS6 Released: • PS7
14: 12.12	Topics: • Course review • Parallelization practice	Topics: • Review • Looking forward	Due: • PS7 (12.19 @11.59p)

Table 3: Course Calendar

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT press, 2022.

[2] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2023. URL `https://www.R-project.org/`.

[3] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL `https://www.stats.ox.ac.uk/pub/MASS4/`. ISBN 0-387-95457-0.

[4] H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL `https://ggplot2.tidyverse.org`.

[5] A. S. Zieffler, J. R. Harring, and J. D. Long. *Comparing groups: Randomization and bootstrap methods using R*. John Wiley & Sons, 2011.