WIP: Identifying Tutorial Affordances for Interdisciplinary Learning Environments

Hannah Kim
Department of Biology
Temple University
Philadelphia, PA, USA
hannah.kim0007@temple.edu

Sergei L. Kosakovsky Pond
Department of Biology
Temple University
Philadelphia, PA, USA
spond@temple.edu

Stephen MacNeil
Dept. of Computer & Information Sciences
Temple University
Philadelphia, PA, USA
stephen.macneil@temple.edu

***Abstract*—This work-in-progress research paper explores the effectiveness of tutorials in interdisciplinary learning environments, specifically focusing on bioinformatics. Tutorials are typically designed for a single audience, but our study aims to uncover how they function in contexts where learners have diverse backgrounds. With the rise of interdisciplinary learning, the importance of learning materials that accommodate diverse learner needs has become evident. We chose bioinformatics as our context because it involves at least two distinct user groups: those with computational backgrounds and those with biological backgrounds. The goal of our research is to better understand current bioinformatics software tutorial designs and assess them in the conceptual framework of interdisciplinarity. We conducted a content analysis of 22 representative bioinformatics software tutorials to identify design patterns and understand their strengths and limitations. We found common codes in the representative tutorials and synthesized them into ten themes. Our assessment shows degrees to which current bioinformatics software tutorials fulfill interdisciplinarity.**

I. Introduction

Interdisciplinarity involves solving problems by integrating ideas and approaches from multiple disciplines [1], [2]. Although the cost of merging ideas from more than one discipline is steep, it is justified by the significant impact of technological or socio-technological problems being addressed (e.g. climate change and public health) [2]. A key challenge in interdisciplinary research is the establishment of common ground between disciplines. The challenge entails the necessity of exploring another discipline for concepts and methods when required by the problem.

Bioinformatics is an interdisciplinary field in which the goal of research is to find important biological patterns using computational and statistical methods. Bioinformatics software users can range from those in experimental biology [3], [4] to those in software development [5]. Bioinformatics-oriented courses are offered across many disciplines, including biology, biotechnology, life sciences, pharmacy, computer science, and information technology [6]. Naturally, users come from vastly different backgrounds. However, the time allotted for bioinformatics coursework in a formal training is often limited [5]. Hence, there is a constant need for users to learn outside formal learning environments.

Informal learning can supplement formal learning [7]. After completing formal education, individuals continue to gain knowledge through informal learning for the rest of their lives. Afforded by the prevalence of e-learning, researchers routinely encounter tutorials online. In bioinformatics, many tutorials focus on providing information about bioinformatics software.

Developing tutorials in bioinformatics presents unique challenges due to its interdisciplinary nature. Most software and corresponding tutorials are built on the assumption that users have basic scripting or data-cleaning skills. In reality, users vary widely, ranging from those without much computational experience to those capable of developing their own software. Hence, there must be an indication of design deficiencies for specific user groups.

To understand the affordances of bioinformatics software tutorials, we used the conceptual framework of interdisciplinarity proposed by Borrego and Newswander [2]. Their five criteria used to assess interdisciplinary work are: (1) grounding in traditional disciplines; (2) synergistic integration of disciplinary insights; (3) communication and translation (C&T) across boundaries to reconcile disciplinary differences; (4) critical awareness of achievable scopes and limitations; and (5) teamwork among researchers.

Previously, users of bioinformatics tool documentation were categorized as new, experienced, power, and all [8]. Bioinformatics software documentation for new users encompasses the manuscript, readme, quickstart, reference manual, FAQ, and searchable forum or mailing list, with the first three exclusively reserved for new users. However, this approach has limitations. First, the taxonomy above ignores the diverse user disciplinary backgrounds in bioinformatics [2]. Second, the static nature of academic publications does not reflect the dynamic nature of software development expected in successful tools [9]. Third, academic publications are not always open access and therefore not always accessible to users. Hence, we defined a bioinformatics software tutorial as a standalone set of documentation containing information about the biological and methodological significance and design that supports new users. With this definition of a tutorial, we asked the following research questions:

RQ 1 What features do bioinformatics software tutorials have?
RQ 2 What do the tutorial designs achieve in the interdisciplinary learning environment?

II. Content Analysis Methodology

We adopted a content analysis approach using bioinformatics software tutorials to systematically explore design patterns.

A. Sampling Methods

We obtained a list of 22 representative bioinformatics software tools from a 2018 study [8]. We identified their tutorial websites by querying the software names with the term "bioinformatics" on Google. We gathered tutorial materials for each software from the corresponding website. The samples were collected between May 1, 2024, and May 12, 2024.

Software List: BLAST [10], MEGA [11], PLINK [12], SWiss-PdbViewer [13], SAMtools [14], BWA [15], EMBOSS [16], Bowtie 2 [17], DESeq 2 [18], Cufflinks [19], GATK [20], limma [21], edgeR [22], MACS [23], Bedtools [24], Clustal Omega [25], Meme Suite [26], Trimmomatic [27], STAR [28], Segway [29], Bioconductor [30], and Picard Tools [31].

B. Coder

The main coder was a bioinformatics researcher with a background in chemistry, computational biology, and higher education teaching. Given the coder's hybrid academic background and a decade of experience in the field, they were considered well-qualified to extract themes comprehensively.

C. Open Coding Process

The coder used open coding [32] inductively to capture emerging themes in tutorials. Open coding was accompanied by memoing to extract meanings effectively [33]. After identifying open codes from all tutorials, we iterated through the tutorials again deductively to record the presence of each code in each tutorial. We organized the records in the form of tables. The table was filled primarily using Boolean values (e.g., command line tool availability), but also including numeric (e.g., year of first publication) and categorical (e.g., Python or R package manager) values. We also collected codes describing new user materials (e.g., from Software A, "Help; Getting Started; Manual; Guide; Handbook"). Throughout this process, the codes were discussed with the research team and adjusted as needed through a mediation process.

D. Analyzing Codes

1) Quantitative analysis: We converted each code row into binary or numeric values as appropriate. Based on this data, we examined the frequency, mean, and standard deviation of each open code. We also investigated the correlations between the codes. Preliminary statistical analyses were performed using the R programming language.

2) Qualitative analysis: We implemented an axial coding approach to identify relationships among open codes [32]. We focused on outcomes achieved by each code.

3) Evaluation: We calculated the arithmetic mean of code frequencies to assess the interdisciplinarity of bioinformatics software tutorials. Codes were assigned to each interdisciplinarity criterion with redundancy: if a code supported the criterion, it was given a (+) value; if it contradicted the criterion, it was given a (-) value.

III. Preliminary Results

A. Open Coding

1) At a Glance: From the open coding of tutorial materials of the bioinformatics software (N=22), we obtained 56 open codes. We verified that the software list was representative, covering bioinformatics topics extensively, from phylogenetics to transcriptomics to repository management. The year of first software availability ranged from 1990 to 2015, with a median of 2010. These tools were each maintained for 4 to 33 years (mean=14.2, sd=6.8), based on their first availability and last release dates. All tutorials had the following codes: general software significance explained, download/installation information found easily, command-line tools, current version information and instructions written by the developers.

2) Tutorials Come in Various Names: Unlike what the study in 2018 [8] suggested, the vocabulary around new user materials was not straightforward. In fact, materials for new users were frequently found in different documents of the same software. There were between 2 and 16 different words per software signaling new user materials (mode=6.0, sd=3.7). Out of 48 words describing new user materials, 18 of them occurred more than twice across different software.

Top 18 words: (manual,12), (documentation,11), (FAQ,11), (help,11), (tutorial,9), (gettingstarted,8), (guide,8), (quickstart,8), (workflow,7), (reference,5), (resource,4), (vignette,4), (description,3), (glossary,3), (how,3), (note,3), (tips&tricks,3), and (troubleshooting,3).

3) Correlations: The amount of tutorial material was expected to increase with the length of maintenance. We observed that the years of software maintenance and the count of codes describing new user materials showed a moderate correlation (Pearson's r=0.45). When we limited the same analysis to tools that have basic tutorials across more than one platform, we found the correlation to be negligible (r=0.06; N=10). On the other hand, when we limited the analysis to the tools that have basic tutorials on only one platform, there was a strong correlation (r=0.63; N=12). Based on our memo, most tutorials using more than one platform utilized platforms with infrastructure such as GitHub [34], Readthedocs [35], SourceForge [36], and Bioconductor [30]. Each platform had a set of default functionalities that can aid in tutorial development. We counted any personal website as one platform. In summary, the volume of new user materials increased over time when tutorials were maintained on a single platform.

We investigated the presence of discussion group promotions in tutorials over time. We found a moderate correlation between developer-promoted discussion groups and the year of first availability (r=0.52). Seventeen out of nineteen tools that were first available after the year 2000 had a discussion group link in their tutorials. Other time-related codes, such as the years of software maintenance (r=-0.37) and the year of latest software release (r=0.29), did not show a strong correlation with the discussion group code.

B. Axial Coding

By grouping open codes, we aimed to understand recurring themes. Ten axial codes have been synthesized from the 51 open codes. Each axial code had 3 to 8 open codes involved. From this point forward, codes will be referenced by their numerical identifiers as detailed in Table I. Note that some open codes in X1, X2, and X4 are likely reserved for a specific group of new users.

C. Evaluation of Interdisciplinarity

We assigned codes to the criteria previously proposed [2].

+ Disciplinary grounding: {C3, C7-C9, C18-C19}
− Disciplinary grounding: ∅
+ Integration: ∅
− Integration: ∅
+ C&T: {C1-C2, C4-C6, C10-C13, C14-C17, C20-C22, C23-C24, C26, C37-C42, C48-C51}
− C&T: {C27-C28}
+ Critical Awareness: {C29-C32}
− Critical Awareness: {C34-36}
+ Teamwork: {C43-47}
− Teamwork: ∅

We assessed each criterion by taking the arithmetic mean.

- Disciplinary grounding: `73%`
- Integration: `NA`
- C&T: `58%`
- Critical Awareness: `17%`
- Teamwork: `54%`

IV. Interpretation

A. RQ1 → Features in Bioinformatics Software Tutorials

We captured ten themes reflecting trends in tutorial features (See Table I). Tutorials were developed for various types of user interfaces and typically included features related to installation, data specification, format, and relevance. We observed that features related to tutorial maintenance varied, with some indicating effectiveness and others indicating ineffectiveness. Similar mixed signals were noted in features related to resilience. Furthermore, features aimed at promoting community engagement, collaboration, and marketing were commonly found across tutorials.

B. RQ2 → Design Achievements by Themes

We examined the achievements of the tutorial features based on the identified themes. The availability of X1 determined the accessibility of the software and their tutorials. X2-X5 included features dedicated to providing structures. Users with no prior knowledge of a bioinformatics tool can benefit from having structures [37]. For example, providing clear installation and data specification information can directly influence user engagement with the software. Descriptions of general software significance (C21) and method (C22) can motivate new users to further explore the software. Incorporating multimedia presentations in the form of videos (C14) can also improve new user learning experience [38]. X6-X10 were

Codes	%	% Visualization
X1. User Interface		
C1. Web application	41%	●●●●○○○○○○
C2. Graphical user interface on local machine	27%	●●●○○○○○○○
C3. Command-line tool	100%	●●●●●●●●●●
X2. Installation		
C4. Download/install information found easily	100%	●●●●●●●●●●
C5. Additional installation instruction	91%	●●●●●●●●●○
C6. Operating system requirements	86%	●●●●●●●●●○
C7. Package managers, esp. Python- or R-based	41%	●●●●○○○○○○
C8. Pre-compiled binaries	86%	●●●●●●●●●○
C9. Docker image available	27%	●●●○○○○○○○
X3. Data Specification		
C10. Sample data	82%	●●●●●●●●○○
C11. Raw sample data with reference	55%	●●●●●●○○○○
C12. Input data format	91%	●●●●●●●●●○
C13. Output description	68%	●●●●●●●○○○
X4. Format		
C14. Link to video tutorials	18%	●●○○○○○○○○
C15. Clear list of tasks & options in tutorial	95%	●●●●●●●●●●
C16. Tutorial curated by topic	82%	●●●●●●●●○○
C17. Step-by-step instruction	73%	●●●●●●●○○○
C18. Github or source code	86%	●●●●●●●●●○
C19. Software release notes	95%	●●●●●●●●●●
X5. Relevance		
C20. Software name explained	77%	●●●●●●●●○○
C21. General software significance explained	100%	●●●●●●●●●●
C22. General software method explained	82%	●●●●●●●●○○
X6. Maintenance		
C23. Instructions written by the developers	100%	●●●●●●●●●●
C24. Current version information	100%	●●●●●●●●●●
C25. Years of software maintenance (mean=14.18)	NA	NA
C26. Citation information conspicuous	91%	●●●●●●●●●○
C27. Citation info less accessible e.g., in FAQ	18%	●●○○○○○○○○
C28. Page link not working	41%	●●●●○○○○○○
X7. Resilience		
C29. Last tutorial update date	59%	●●●●●○○○○○
C30. Software limitations	59%	●●●●●○○○○○
C31. Statement of error or retraction	5%	●○○○○○○○○○
C32. Deprecation information	41%	●●●●○○○○○○
C33. Latest software release (mean=2021.59)	NA	NA
C34. Multiple software versions on the main page	9%	●○○○○○○○○○
C35. Tutorials for different major software versions	27%	●●●○○○○○○○
C36. Guide to landing pages itself	9%	●○○○○○○○○○
X8. Promotion of Community		
C37. Instructions written by community members	32%	●●●○○○○○○○
C38. User rating	14%	●○○○○○○○○○
C39. User feedback form	77%	●●●●●●●●○○
C40. Use cases via featuring user work	41%	●●●●○○○○○○
C41. Developer-organized social events	14%	●○○○○○○○○○
C42. Developer-promoted discussion group	77%	●●●●●●●●○○
X9. Willingness to Collaborate		
C43. Software dependency on other tools	36%	●●●●○○○○○○
C44. Link to references other than its own	91%	●●●●●●●●●○
C45. Instructions for other developers	36%	●●●●○○○○○○
C46. Instructions for administrators	9%	●○○○○○○○○○
C47. Copyright Information	100%	●●●●●●●●●●
X10. Marketing		
C48. Ad/news	41%	●●●●○○○○○○
C49. Logo	41%	●●●●○○○○○○
C50. Performance comparison with competitors	18%	●●○○○○○○○○
C51. Summary statistics of download	32%	●●●○○○○○○○

TABLE I
BIOINFORMATICS SOFTWARE TUTORIAL OPEN CODES BY THEMES

themes that required time to establish. Specifically, X6-X7 captured how the software and their tutorials accommodated changes or imperfections. Good maintenance of tutorials can serve as an indicator of software integrity. Resilience showed how tutorials have managed the accumulation of knowledge over time. X8-X10 were environmentally conscious themes. These features demonstrated an understanding of the user community and collaborators. The effort to distinguish the software in the market was a natural outcome of this awareness.

Not all features have uniformly achieved usefulness for all users. There were three different types of user interfaces: two graphical user interfaces (C1-C2) and one command-line interface (C3). The availability of graphical user interfaces varied dramatically across tutorials (41% and 27%), while the command-line interface was universally available. Even the union of graphical user interface features was only 55%. Developing command-line tools is less expensive in many ways. Nevertheless, using command-line tools requires additional steps for users compared to using tools with graphical user interfaces. This apparent trend in tools, which have been maintained for an average of 14.18 years (C25), reflects unrealistic developer expectations about the majority of potential users. The learning cost associated with these expectations is likely prohibitive to many users in the interdisciplinary environment.

Having a large volume of tutorial content may not necessarily improve learning outcomes. Because we sampled tools by representativeness, their tutorials showed a significant correlation between the duration of maintenance (C25, mean=14.18) and the number of words used to describe new user materials (not shown in the table, mode=6). While a rich collection of learning materials can benefit users from different disciplines, using six different words to indicate new user materials is excessive [39], especially in e-learning environments where personalized guidance is often lacking. This highlights the need for greater tutorial resilience to changes (X7), as having too many pointers can undermine efforts to provide structures.

Promotion of community engagement can address knowledge gaps. Many recent bioinformatics tutorials have directed users towards a community (r=0.52, 17 out of 19 after 2010). Developer promotion of community (X8) aligns with a constructivist approach [40]. While a structured approach is effective for beginners [37], it becomes less effective when developer expectations of users inadvertently diverge from reality. By promoting knowledge sharing and increasing interaction opportunities in a community, developers can accommodate diverse user needs in interdisciplinary learning environments.

Some bioinformatics software tutorials draw advantages from leveraging default infrastructures. For example, Bioconductor [30] requires developers to maintain tutorial features such as "vignettes" and keep track of the date of the last tutorial update (C29). Conda [41], [42] not only simplifies software dependency management but also makes installation (C7) straightforward. Additionally, package management platforms often require features such as the latest software version (C24) and release (C33). Overall, default infrastructures significantly facilitate the management of new user materials.

C. RQ2 → Design Achievements by Interdisciplinarity

We assessed the interdisciplinarity of bioinformatics software tutorials based on the five criteria proposed by Borrego and Newswander. We found a strong indication of disciplinary grounding (73%). Communication and translation across disciplinary boundaries, as well as teamwork, were moderately implemented (58% and 54%). Critical awareness was rarely observed (17%). To understand the synergy of integrating multiple disciplines, it is necessary to evaluate outcomes achieved beyond traditional disciplinary boundaries; however, this aspect was outside the scope of our content analysis.

Interdisciplinarity evaluation supplemented the thematic analysis. Each theme could be better understood through the lens of interdisciplinarity. As observed across X1, X2, and X4, many features were grounded in computational disciplines. Information related to other disciplines (e.g., biology) was present throughout the tutorials, but no features could be specifically assigned to these disciplines. Features related to communication and translation were observed across almost all themes, although a couple of them were counterproductive. This trend indicates the widespread, though moderate, presence of such efforts. Features related to critical awareness were identical to those of resilience but could be further categorized into favorable and unfavorable sets. All features related to willingness to collaborate were interpreted as teamwork-related.

V. Conclusion

We have identified design features in bioinformatics software tutorials and assessed their achievements. Integrating ideas and approaches from multiple disciplines is challenging yet highly rewarding. By analyzing current affordances, we can detect strengths and weaknesses to leverage in interdisciplinary learning environments. In bioinformatics, there is currently a noticeable disciplinary skew in tutorials intended for learners from diverse backgrounds. Recognizing such design deficiencies will improve the development of learning materials in environments beyond traditional learning settings.

VI. Potential Limitations

Although we have systematically studied tutorial affordances, some limitations may affect our interpretation. First, only a single coder coded the data. Second, the sample size was small (N=22). Third, our representative samples came from an older study in 2018. Because bioinformatics is a fast-moving field, the tutorial landscape may have shifted since then. Fourth, our findings assume the popularity of the tools. What we found may not apply to less popular tools. Nevertheless, our work-in-progress study provides sufficient insights into the affordances of tutorials in interdisciplinary learning environments for future works.

Acknowledgment

The authors would like to thank H. Ji and Dr. M. Yeon for providing feedback during the conception of the work. The authors would also like to thank C. Zastudil, S. Bernstein, K. Angelikas, and C. Kapp for stimulating discussions around the topic during the weekly HCI Reading Group meetings.

REFERENCES

[1] J. T. Klein, *Interdisciplinarity: History, theory, and practice*. Wayne state university press, 1990.

[2] M. Borrego and L. K. Newswander, "Definitions of interdisciplinary research: Toward graduate-level interdisciplinary learning outcomes," *The Review of Higher Education*, vol. 34, no. 1, pp. 61–84, 2010.

[3] B. Giardine, C. Riemer, R. C. Hardison, R. Burhans, L. Elnitski, P. Shah, Y. Zhang, D. Blankenberg, I. Albert, J. Taylor *et al.*, "Galaxy: a platform for interactive large-scale genome analysis," *Genome research*, vol. 15, no. 10, pp. 1451–1455, 2005.

[4] V. Jalili, E. Afgan, Q. Gu, D. Clements, D. Blankenberg, J. Goecks, J. Taylor, and A. Nekrutenko, "The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2020 update," *Nucleic acids research*, vol. 48, no. W1, pp. W395–W402, 2020.

[5] N. Mulder, R. Schwartz, M. D. Brazas, C. Brooksbank, B. Gaeta, S. L. Morgan, M. A. Pauley, A. Rosenwald, G. Rustici, M. Sierk *et al.*, "The development and application of bioinformatics core competencies to improve bioinformatics training and education," *PLoS computational biology*, vol. 14, no. 2, p. e1005772, 2018.

[6] A. J. Magana, M. Taleyarkhan, D. R. Alvarado, M. Kane, J. Springer, and K. Clase, "A survey of scholarly literature describing the field of bioinformatics education and bioinformatics educational research," *CBE—Life Sciences Education*, vol. 13, no. 4, pp. 607–623, 2014.

[7] S. Carliner, "How have concepts of informal learning developed over time?" *Performance Improvement*, vol. 52, no. 3, pp. 5–11, 2013.

[8] M. Karimzadeh and M. M. Hoffman, "Top considerations for creating bioinformatics software documentation," *Briefings in Bioinformatics*, vol. 19, no. 4, pp. 693–699, 2018.

[9] P. P. Gardner, J. M. Paterson, S. McGimpsey, F. Ashari-Ghomi, S. U. Umu, A. Pawlik, A. Gavryushkin, and M. A. Black, "Sustained software development, not number of citations or journal choice, is indicative of accurate bioinformatic software," *Genome biology*, vol. 23, no. 1, p. 56, 2022.

[10] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of molecular biology*, vol. 215, no. 3, pp. 403–410, 1990.

[11] S. Kumar, K. Tamura, and M. Nei, "Mega: molecular evolutionary genetics analysis software for microcomputers," *Bioinformatics*, vol. 10, no. 2, pp. 189–191, 1994.

[12] S. Purcell, B. Neale, K. Todd-Brown, L. Thomas, M. A. Ferreira, D. Bender, J. Maller, P. Sklar, P. I. De Bakker, M. J. Daly *et al.*, "Plink: a tool set for whole-genome association and population-based linkage analyses," *The American journal of human genetics*, vol. 81, no. 3, pp. 559–575, 2007.

[13] N. Guex, "Swiss-pdbviewer: a fast and easy-to-use pdb viewer for macintosh and pc," *Protein Data Bank Quarterly Newsletter*, vol. 77, p. 7, 1996.

[14] H. Li, "A statistical framework for snp calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data," *Bioinformatics*, vol. 27, no. 21, pp. 2987–2993, 2011.

[15] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[16] P. Rice, I. Longden, and A. Bleasby, "Emboss: the european molecular biology open software suite," *Trends in genetics*, vol. 16, no. 6, pp. 276–277, 2000.

[17] B. Langmead and S. L. Salzberg, "Fast gapped-read alignment with bowtie 2," *Nature methods*, vol. 9, no. 4, pp. 357–359, 2012.

[18] M. I. Love, W. Huber, and S. Anders, "Moderated estimation of fold change and dispersion for rna-seq data with deseq2," *Genome biology*, vol. 15, pp. 1–21, 2014.

[19] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. Van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter, "Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation," *Nature biotechnology*, vol. 28, no. 5, pp. 511–515, 2010.

[20] A. McKenna, M. Hanna, E. Banks, A. Sivachenko, K. Cibulskis, A. Kernytsky, K. Garimella, D. Altshuler, S. Gabriel, M. Daly *et al.*, "The genome analysis toolkit: a mapreduce framework for analyzing next-generation dna sequencing data," *Genome research*, vol. 20, no. 9, pp. 1297–1303, 2010.

[21] M. E. Ritchie, B. Phipson, D. Wu, Y. Hu, C. W. Law, W. Shi, and G. K. Smyth, "limma powers differential expression analyses for rna-sequencing and microarray studies," *Nucleic acids research*, vol. 43, no. 7, pp. e47–e47, 2015.

[22] M. D. Robinson, D. J. McCarthy, and G. K. Smyth, "edger: a bioconductor package for differential expression analysis of digital gene expression data," *bioinformatics*, vol. 26, no. 1, pp. 139–140, 2010.

[23] Y. Zhang, T. Liu, C. A. Meyer, J. Eeckhoute, D. S. Johnson, B. E. Bernstein, C. Nusbaum, R. M. Myers, M. Brown, W. Li *et al.*, "Model-based analysis of chip-seq (macs)," *Genome biology*, vol. 9, pp. 1–9, 2008.

[24] A. R. Quinlan and I. M. Hall, "Bedtools: a flexible suite of utilities for comparing genomic features," *Bioinformatics*, vol. 26, no. 6, pp. 841–842, 2010.

[25] F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Söding *et al.*, "Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega," *Molecular systems biology*, vol. 7, no. 1, p. 539, 2011.

[26] T. L. Bailey, M. Boden, F. A. Buske, M. Frith, C. E. Grant, L. Clementi, J. Ren, W. W. Li, and W. S. Noble, "Meme suite: tools for motif discovery and searching," *Nucleic acids research*, vol. 37, no. suppl_2, pp. W202–W208, 2009.

[27] A. M. Bolger, M. Lohse, and B. Usadel, "Trimmomatic: a flexible trimmer for illumina sequence data," *Bioinformatics*, vol. 30, no. 15, pp. 2114–2120, 2014.

[28] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras, "Star: ultrafast universal rna-seq aligner," *Bioinformatics*, vol. 29, no. 1, pp. 15–21, 2013.

[29] M. M. Hoffman, O. J. Buske, J. Wang, Z. Weng, J. A. Bilmes, and W. S. Noble, "Unsupervised pattern discovery in human chromatin structure through genomic segmentation," *Nature Methods*, vol. 9, pp. 473–476, 2012. [Online]. Available: http://www.nature.com/nmeth/journal/vaop/ncurrent/full/nmeth.1937.html

[30] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry *et al.*, "Bioconductor: open software development for computational biology and bioinformatics," *Genome biology*, vol. 5, pp. 1–16, 2004.

[31] "Picard toolkit," https://broadinstitute.github.io/picard/, 2019.

[32] M. Williams and T. Moser, "The art of coding and thematic exploration in qualitative research," *International management review*, vol. 15, no. 1, pp. 45–55, 2019.

[33] M. Birks, Y. Chapman, and K. Francis, "Memoing in qualitative research: Probing data and processes," *Journal of research in nursing*, vol. 13, no. 1, pp. 68–75, 2008.

[34] "Github," https://github.com/.

[35] "Read the docs," https://about.readthedocs.com/?ref=readthedocs.org.

[36] "Sourceforge," https://sourceforge.net/.

[37] R. D. Tennyson and C. A. Bagley, "Structured versus constructed instructional strategies for improving concept acquisition by domain-experienced and domain-novice learners." 1991.

[38] M. Tani, M. Manuguerra, and S. Khan, "Can videos affect learning outcomes? evidence from an actual learning environment," *Educational technology research and development*, vol. 70, no. 5, pp. 1675–1693, 2022.

[39] S. E. DiCarlo, "Too much content, not enough thinking, and too little fun!" *Advances in physiology education*, 2009.

[40] S. O. Bada and S. Olusegun, "Constructivism learning theory: A paradigm for teaching and learning," *Journal of Research & Method in Education*, vol. 5, no. 6, pp. 66–70, 2015.

[41] A. S. Distribution, "Conda," https://www.anaconda.com/.

[42] P. J. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski *et al.*, "Biopython: freely available python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, p. 1422, 2009.