Notes on the First Release of the Genetics10 educational software.

Joan Alcaide Núñez

Deutsche Schule Barcelona, Av. Jacint Esteva i Fontanet, 105 08950 Esplugues de Llobregat Barcelona, Spain
joanalnu@outlook.com, joan.alcaide@dsbarcelona.com

July 16, 2024

We present an educational tool to learn about general genomics using an easy-to-use software. It enables students to play with various functions and learn about genomics and biology. There is no requirement for previous knowledge of coding. The program enables students to translate DNA strings into RNA and amino acid sequences, following the biological procedures. It also has several additional functions such as comparison between sequences, generation of mutation, or protein structure visualization using AlphaFold. We believe this is a powerful tool for students to play, learn, and solve challenges creating their code.

1 Introduction

This is the paper for the release of the Genetics10 software program. In the following sections, we will define the creator's aims and the terms of usage of the product. Further, we will teach and show how to properly use the tool and how to implement it in science class.

1.1 Definition

`Genetics10` is a software program written in `Python` and based on `Jupyter Notebook`. This enables the code to be easily accessed via the web, without the need to install anything or powerful computing performance. This makes it perfect for school-managed devices. The program has several functions that one can call to perform simulations of various biological processes around genomics (see section 4.1). Further, no previous knowledge of coding or computer science is needed to use this program.

1.2 Aim

This project aims to provide a powerful and flexible tool for teachers and students to learn about genetics in biology or science classes. It is made in mind of the European curriculum for biology class of year 10. However, this program can be used in a variety of contexts. The requirements in terms of coding are zero.

The goal is to enable students to play with the different functions and let them learn through challenges and hands-on activities. This provides them with real experiences they can remember and learn from better than text-book-reading. Additionally, it is also easy to expand, which means that students with less coding knowledge can learn the basics and start coding right away. More advanced students can use this base to power up the functions to expand the possibilities and use their full potential. The fact that this program has Object Oriented Programming (OOP) structure makes it easy to adapt to different students, making it easy for beginners and scaleable for advanced.

2 Who Can Use It? - Everyone!

2.1 It's free

This is a free-of-charge resource. We want to provide useful tools to learners to be able to showcase their potential regardless of economic situation. Therefore, we have uploaded the program, guidelines, and further files to a `GitHub Repository` which is public and free to download.

2.2 Open Source

We want the code to be open source, which means that everyone can look into the code and make their changes for a more personal use. We are also open to feedback via email notifications or using GitHub's issues or pull requests.

2.3 MIT License

This program and the complete `GitHub Repository` is protected under the MIT license with Copyright (c) to Joanalnu 2024.

Finally, we also don't want to set language barriers, therefore we try to translate the code into as many languages as possible. If your language is not available yet, reach out to us and we can work on a translation. Please, note that this can take time.

3 How To Access It? - GitHub!

3.1 Getting software from GitHub

First, go to the project's `GitHub Repository` and download the files (you can clone the repository using git, or download a ZIP folder) by clicking the code button. If you download the ZIP, don't forget to decompress it.

3.2 Opening Jupyter Lab

Next open Jupyter Lab online (this is a platform from Jupyter that enables you to run Notebook in your browser) with this link: https://jupyter.org/try-jupyter/lab/.

Once you have opened the tab, go to the left-handed panel, where the files in the directory can be seen. Click on the upload button in the panel's toolbar to upload the file you want from the previously downloaded Genetics10 folder (wherever you have downloaded it on your device). The file with the extension `ipynb` is the file for the code, first 2 letters in the file's name indicate the language of that file. Select the preferred language and upload the file. Additionally, you can also upload the `README.md` or `License` files. Note that if you upload the `README.md` file, you must overwrite `Jupyter Lab's README.md` file.

Once you have uploaded your files, you only need to open them by double-clicking on them and you are ready to start.

4 How To Use It? - Easy!

In the beginning, you must run the first cell one time to initialize the `biogen` class. You can do that by pressing `shift + enter` or pressing the run button in the upper toolbar. After you have run that cell you can collapse it by clicking the blue bar to the left.

Below the first cell, you can find explanations and examples of how to use different functions. In a few words, type `biogen.fucntionname(argument)` to call any function of the `biogen` class. Remember to provide the appropriate argument inside the brackets. The code snippet below calls the `dna2rna()` function. It gives a string `dna` as input, and the functions returns the output called `rna`.

```
rna = dna2rna(dna)
```

4.1 Code's Features

The available functions are listed here:

1. `dna2rna()`
   Transcribes the provided DNA string into an RNA string by changing the bases.
   Argument: `string`
   Output: `string`

2. `rna2dna()`
   Transcribes the provided DNA string into an amino acid string by reading codons (3x bases) and using the catalog.
   Argument: `string`
   Output: `string`

3. `dna2amino()`
   Transcribes DNA strings directly into amino acid strings, it's a merge of the dna2rna and rna2amino methods.
   Argument: `string`
   Output: `string`

4. `compare()`
   Compares the strings (regardless if DNA, RNA, or amino acids), it always returns a boolean and a string. True if both strings are identical, or False and where the string differs.
   Argument: `string1, string2`
   Output: `boolean, string`

5. `check()`
   It checks if the provided string is a valid DNA or RNA string. It does not check for amino acid strings.
   Argument: `string`
   Output: `string`

6. `read_input()`
   Used to open files. The full path to the file must be saved in the same folder as this file and can have only 1 sequence.

Argument: string
Output: string

7. createmutation()
Returns a new string with a mutation (only 1 per run). The mutation can change a base, erase a base, or add a new one in any position.
Argument: string
Output: string

8. iterate()
By inputting a list of inputs and a list of functions it returns a table with all the results for each function and input. Argument: list, list
Output: dataframe (table)

9. tosingle()
Transcribes an amino acid string from three-letter code to single-letter code.
Argument: string
Output: string

10. alphafold_prediction()
By inputting a UniProt ID, it returns a URL to the PBD file of the predicted protein's structure.
Argument: string
Output: dictionary

11. download_pdb()
Internal function which enables generate_protein to work. It retrieves the structure data from the URL.

12. generate_protein()
By inputting the resulting dictionary of alphafold_prediction() it returns a visualization of the predicted protein's structure.
Argument: dictionary
Output: None

Note that the AlphaFold API (alphafold_prediction() function) only admits UniProt IDs as input. You can find the UniPort ID of a protein or gene on the internet. We recommend the following databases.

- Official UniProt website[1]
- For genes: Ensembl[2]
- AlphaFold website[3]

5 Coding Tips for beginners

As stated before, no previous knowledge of coding is required. Therefore, we will explain some basic concepts here. We use Jupyter Notebook to deliver this software to everybody. This is a platform that enables running code online without installing or downloading anything. You only need to open the link, upload the files, and start playing.

[1]https://www.uniprot.org
[2]https://www.ensembl.org/Multi/Tools/Blast
[3]https://alphafold.ebi.ac.uk

The program is coded in Python (a language). Let's dive into an example of using the code. First, we want to provide a DNA string to be translated. We'll call it my_dna:

```
my_dna = 'TACACTTGACTTATCATT' # string must be between
    this ones ''
```

Next, we will translate this DNA string into RNA. Biologically this process is conducted inside the cell's nucleus. To simulate that we'll use the dna2rna() function as shown next:

```
my_rna = dna2rna(my_dna)
```

Let's divide this line of code into parts. First, we declare the new variable my_rna, which is the result of calling the function dna2rna(). Inside the bracket of the function we insert the input (argument), this is the information that the function requires, in this case, the DNA string we created before called my_dna.

Now that we have the RNA string, let's transcribe it into an amino acid sequence, which in a cell would configure a protein. To do so we call the rna2amino() function as follows:

```
my_aminoacids = rna2amino(my_rna)
```

Finally, we want to see our results. Therefore, we will type our variable to be shown:

```
my_aminoacids
```

If you run this code on the Jupyter Notebook, you will obtain the amino acid sequence for the DNA string you provided. Now that you know about the basics of coding with biogen, you can start to play with these and the other functions. If you have further doubts or questions ask your teacher for help.

6 Info for educators

6.1 How can I use this in my class?

First, identify in your curriculum where you can integrate the software, which is already built and aligned with the general education guidelines. Then you should start by explaining the fundamental concepts of genomics in your biology or science class, as you would do normally. Then you can introduce this tool to students and explain how to use it.

You can use the software to design problem-solving challenges that require students to use critical thinking and coding skills. For example, a scenario where a gene mutation causes a disease, and ask students to write code that identifies and corrects the mutation. This type of activity fosters creativity and problem-solving skills and leads further to more science like CRISPR Cas-9.

Also, perform planned activities where students apply what they've learned in real life. Create assignments where students write simple code using the pre-established functions to emulate genetic processes such as transcription and translation.

By providing step-by-step instructions students will have better chances of understanding the biological content and a better usage of the full potential of this tool. Moreover, providing by integrating real-world examples and applications in genomics and biotechnology can increase student motivation and interest, and show and discuss modern research tools.

Finally, you can also adopt a flipped classroom approach by assigning software tutorials as homework and using class time for interactive and applied learning. This allows for maximized classroom engagement and allows for more personalized instruction.

Encouraging collaboration by planning group projects, students can work together to solve more complex problems. Collaborative projects foster teamwork and allow students to learn from each other.

By incorporating these strategies, you can effectively use this software to enhance your biology curriculum, engage students, and foster a deeper understanding of both genomics and coding.

6.2 Why should I use this in my class?

This is a useful resource for students to learn both genomics and basic coding. On the one hand, this is a powerful tool that enables students to apply what they have learned regarding biology. It is made to be interactive and customizable and anyone can run their code without knowledge of coding. On the other hand, students will learn and get first-hand experience with bioinformatics and computation. Coding is an essential skill for future workers, regardless of their field.

Further, the fact that it is web-based and does not need any installation makes it perfect for school-managed devices and enables usage regardless of the operating system. It also fosters teamwork and communication skills, as projects can be done in collaboration.

Additionally, the features of the software are aligned with the scholar curriculum and it shows practical applications of classroom content right away. It also promotes critical thinking by allowing students to write their code to solve problems and engage actively. Prior knowledge of coding is not required at all, as students will use the pre-established functions that enable a wide range of possibilities. Further, students can adapt their code to their problems or write new functions. The code is easily scalable and has endless possibilities!

Acknowledgements

We acknowledge the use of Python [VRD09] and additional libraries: Numpy [HMvdW+20], Random [VR20], Pandas [pdt20], os-sys requests, and py3Dmol [RK15]. 3DMol.js is funded through R35GM140753 from the National Institute of General Medical Sciences. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institute of General Medical Sciences or the National Institutes of Health. Further, we make use of the public AlphaFold API.

Genetics10 is a project developed voluntarily without the aim of lucre and any type of funding.

I want to thank Ania Alvarez for the beneficial feedback and the great support, as well as the German School's Barcelona Biology Department for accepting using this software.

I also want to thank to all contributors and testers who returned useful feedback during the development and testing of the program, and those who supported this project with enthusiastic motivation.

I also want to thank Prof Sonia Casillas Viladerrams et al.(Universitat Autónoma de Barcelona) and the Institut de Ciències de l'Educació for organizing the course "Adapta't: La nostra història evolutiva Lleida en el genoma" in 2022, which first introduced me to genomics and bioinformatics, as well as laboratory activity.

Finally, I want to express a bold thanks to my family, who have provided support and motivation despite not testing the software.

7 About the Creator

At the time of this release I'm a 12th-grade student at the Deutsche Schule Barcelona (German School of Barcelona) based in Esplugues de Llobregat. I'm an independent researcher primarily in physics, astronomy & astrophysics, and extragalactic astronomy.

However, I've applied my coding knowledge in a variety of challenges and projects ranging from statistical predictions of World Cup winners to this genetics project.

Happy to collaborate, feel free to reach out!

References

[HMvdW+20] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[pdt20] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[RK15] Nicholas Rego and David Koes. 3dmol. js: molecular visualization with webgl. *Bioinformatics*, 31(8):1322–1324, 2015.

[VR20] Guido Van Rossum. *The Python Library Reference, release 3.8.2*. Python Software Foundation, 2020.

[VRD09] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.