

Development of Node- RED web-based, open- source visual workflow tool for bioinformatics

Ferran Fàbregas

Area 4

Master's degree in bioinformatics and biostatistics
(UOC/UB)

Elisabeth Ortega, PhD
(Antoni Pérez Navarro, PhD)

2nd June 2022



This work is under the license Attribution-NonCommercial
<https://creativecommons.org/licenses/by-nc/3.0>

FINAL WORK CARD

Title:	Development of Node-RED web-based, open-source visual workflow tool for bioinformatics
Author:	Ferran Fàbregas
Tutor:	Elisabeth Ortega, PhD
SRP:	Antoni Pérez Navarro, PhD
Date of delivery:	2 nd June 2022
Studies:	Master's degree in bioinformatics and biostatistics (UOC/UB)
Area:	Area 4
Language:	English
Number of credits:	15
Keywords:	Node-RED, javascript, workflow, visual toolkit, integration, genomics, programming

Abstract

Bioinformatics tools have become a key element in the current development of genomics, largely thanks to the involvement of public institutions, research centers, private companies, and individuals alike, which has led to the appearance of a large quantity and variety of applications.

In this MTP, we developed a bioinformatics software for the visual design of genomic data workflows in a web environment based on Node-RED [1], incorporating functionalities related to genomics as the reading and writing of genomic data files, sequence management, treatment, and analysis, with the aim of designing an open, flexible, and interdisciplinary tool with a moderate learning curve that takes advantage of the native functionalities of the Node-RED platform and, above all, is useful not only for the bioinformatics community but also for the educational community too.

For this reason, not only the main software tool has been developed, but also a set of utilities has been published, such as an issue management platform, a mailing list with an open forum, the publication of the source code as open-source, and the packaged software itself, in addition to its corresponding documentation.

In conclusion, we endeavored to provide the bioinformatics community with a new useful integrative tool that provides a new approach, both at a technical and conceptual level, thanks to its innovative architecture, the implementation of workflows, the simplicity of use, and its integration with other non-bioinformatics services, with the addition of to be an open-source-based tool, thus allowing its constant evolution by any member of the bioinformatics community.

Contents

1	Abstract	9
2	Introduction	10
2.1	Background and justification	10
2.2	General description	11
2.3	Objectives	11
2.3.1	General objectives	11
2.3.2	Specific objectives	12
2.4	Approach and methodology	13
2.5	Planning	14
2.5.1	Main Tasks	14
2.5.2	Resources	15
2.5.3	Calendar	15
2.5.4	Milestones	17
2.5.5	Risk analysis	17
2.6	Expected results	18
2.7	Structure of the MTP	19
3	State of the art	20
3.1	Different types of bioinformatic tools	20
3.1.1	Data sources	20
3.1.1.1	Primary sources	20
3.1.1.2	Secondary sources	21
3.1.2	Specialized software	22
3.1.3	Integrative software	22
3.1.3.1	Comparison with the Galaxy project	22
4	Methodology	24

5	Results	26
5.1	Study cases	29
5.1.1	Biological study case	29
5.1.2	Educational study case	30
5.1.3	Practical example	32
5.1.4	Integrative study case	33
6	Discussion	35
7	Economic assessment	39
8	Conclusions	41
8.1	Further actions	42
8.2	Planning follow-up	43
9	Annexes	46
9.1	Software documentation	46
9.1.1	Description of the biotools Node-RED packages	46
9.1.2	Biotools: Preprocessing Raw data node	54
9.1.3	Biotools: Processing FASTA file node	55
9.1.4	Biotools: Processing JASPAR file node	56
9.1.5	Biotools: FASTA file output	57
9.1.6	Biotools: Processing GenBank file node	58
9.1.7	Biotools-sequence: Reverse node	59
9.1.8	Biotools-sequence: Complement node	59
9.1.9	Biotools-sequence: Transcription node	60
9.1.10	Biotools-sequence: Translation node	60
9.1.11	Biotools-sequence: Is degenerate node	61
9.1.12	Biotools-sequence: Get subsequence	61
9.1.13	Biotools-sequence: Sequence type node	62
9.1.14	Biotools-analysis: Sequence count node	62
9.1.15	Biotools-analysis: Hamming distance	63
9.1.16	Biotools-analysis: Transition / transversion rate node	63
9.1.17	Biotools-analysis: Alignment analysis node	64
9.1.18	Biotools-analysis: Motif PFM score node	65
9.1.19	Notes	66
9.2	Software source code	66
9.3	Tutorials	66

10 Glossary	67
11 Bibliography	70

List of Figures

2.1	Node-RED main screen	12
2.2	Counting nucleotides webapp using Node-RED dashboard nodes	14
2.3	Gantt diagram of project tasks and milestones	16
2.4	Detail of tasks and milestones schedule	16
3.1	Gene Expression Omnibus main website	21
3.2	Galaxy project main page	23
4.1	Iterative and incremental process diagram (<i>by Krupadeluxe - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=104924876</i>)	25
5.1	JASPAR PFM format from https://testjaspar.uio.no/docs/#raw-pfm-format	27
5.2	Project documentation website	28
5.3	GitHub repo of the project	29
5.4	Practical example of the biological study case with alignment analysis and motif score functionalities	31
5.5	Practical example of basic genomic sequence management using Node-RED and the developed module	33
5.6	Practical example of access to ENSEMBL RESTful API using Node-RED	34
6.1	List of available open-source bioinformatics software	36
6.2	List of available Node-RED nodes ordered by number of downloads	38
7.1	Preamble of the GNU3.0 licence from https://www.gnu.org/licenses/gpl-3.0.html	40
8.1	GitHub project issue tracker	43
8.2	NPM repository with the published biotools modules developed in this MTP	45
9.1	Node.js installers depending on OS platform	47
9.2	Node-RED main menu	48
9.3	Instalation of biotools nodes using Manage Palette	49

9.4 Filtered biotools nodes from Node-RED main screen 50

9.5 Node-RED deploy options 50

Chapter 1

Abstract

Bioinformatics tools have become a key element in the current development of genomics, largely thanks to the involvement of public institutions, research centers, private companies, and individuals alike, which has led to the appearance of a large quantity and variety of applications.

In this MTP, we developed a bioinformatics software for the visual design of genomic data workflows in a web environment based on Node-RED [1], incorporating functionalities related to genomics as the reading and writing of genomic data files, sequence management, treatment, and analysis, with the aim of designing an open, flexible, and interdisciplinary tool with a moderate learning curve that takes advantage of the native functionalities of the Node-RED platform and, above all, is useful not only for the bioinformatics community but also for the educational community too.

For this reason, not only the main software tool has been developed, but also a set of utilities has been published, such as an issue management platform, a mailing list with an open forum, the publication of the source code as open-source, and the packaged software itself, in addition to its corresponding documentation.

In conclusion, we endeavored to provide the bioinformatics community with a new useful integrative tool that provides a new approach, both at a technical and conceptual level, thanks to its innovative architecture, the implementation of workflows, the simplicity of use, and its integration with other non-bioinformatics services, with the addition of to be an open-source-based tool, thus allowing its constant evolution by any member of the bioinformatics community.

Chapter 2

Introduction

2.1 Background and justification

In recent decades, the evolution of technologies associated with genomics has led to an enormous increase in the amount of data available to the scientific community. This fact has made bioinformatics applications a key element in the scientific analysis of information and in the current biotechnological field.

This need has led to the continuous and powerful development of a huge set of applications designed to enable and facilitate these analyses and procedures.

Because of the very heterogeneous needs and different professional profiles in the bioinformatics field, there are a large number and variety of tools and applications, from those that fall within a more general scope to others that are much more specialized.

Particularly, the currently available bioinformatics integrative tools are extremely powerful, but also has some drawbacks like:

- Often are very complex.
- Steep learning curve.
- Often needs advanced programming skills.
- Complex integration with external non-biological services and resources.

In this sense, after the analysis of different sets of existing bioinformatics utilities, we have concluded that there is room for the implementation of a new integrative tool that:

- Softens the learning curve with a visual toolkit.
- Allows being used not only for the bioinformatics professionals but also for the educative community.

- Improves the integration to the non-biological IT resources without the need for programming.

2.2 General description

This master's degree final project is developed by Ferran Fàbregas Carreté, a student of the UOC/UB bioinformatics master's degree under the direction of Dr. Elisabeth Ortega.

The project focuses on the development of a new integrative bioinformatic toolkit based on the creation of workflows using Node-RED, a node.js [2], web-based visual environment for developers.

The project planning is based on:

- Development of bioinformatics software modules.
- Testing of the modules.
- Documentation of modules features.
- Setting up an online platform, allowing to share the source code, documentation and interaction with the users.
- Release a public version of the bioinformatics software available to the bioinformatics and educational communities.

Through all these elements, in this MTP project, we will create an easy-to-use, accessible, flexible, powerful, functional, and transversal integrative bioinformatics tool.

2.3 Objectives

2.3.1 General objectives

The main goals of this project are:

- To provide to the bioinformatics community a new visual integrative and dynamic workflow tool through the implementation of a set of node.js and Node-RED web platform (Figure 2.1) software modules.
- To give access to the different bioinformatics tools to Node-RED's pre-existing set of modules and functionalities [3].
- To develop a modular bioinformatics tool that brings a moderate learning curve, making the software useful not only in professional environments but also in educational ones.



Figure 2.1: Node-RED main screen

2.3.2 Specific objectives

The main goals defined in the previous section can be split into several specific objectives:

- Provide to the bioinformatics community a new visual integrative and dynamic workflow tool through the implementation of a set of node.js and Node-RED web platform (Figure 1.1) software modules.
 - Build three Node-RED modules related to bioinformatics.
 - Bring to the community the documentation about the developed modules.
- Give access to the different bioinformatics tools to Node-RED's pre-existing set of modules and functionalities.
 - Integrate new modules to Node-RED modules repository. Node-RED has more than 3000 modules available called flows, enabling the user to process, transform and interconnect data with thousands of useful services for the bioinformatician like:
 - * Input/output data from almost any existing database engine, relational or not.
 - * Integration of a huge range of APIs and web services.
 - * Data transformations.
 - * Creation of flexible web-based interactive dashboards.
 - * Integration of non-bioinformatic IT services.

- Develop a modular bioinformatics tool that brings a moderate learning curve, making the software useful not only in professional environments but also in educational ones.
 - Facilitate the collaboration between the bioinformatics and the educational community to improve the application.
 - Bring to the educational community a multilayered easy-to-learn but powerful tool related to genomics.

2.4 Approach and methodology

In order to develop this project, we will carry out research on the bioinformatics tools for the analysis of genomic data currently available.

Since bioinformatics is a highly specialized field, there is a huge amount of software available, some open-source, and some proprietary and linked to specialized hardware, but as the proposed project responds to an integrative development related to genomics the software tools analyzed have been focused on this applications profile.

After performing the investigation, it was concluded that despite the impressive development of some existing applications, there was a space for the implementation of a new visual integrative tool on a slightly different approach than the existing ones.

Because of that, the application development on this MTP will be focused not only on the bioinformatics functionalities but also on the transversality between the bioinformatics and non-bioinformatics features, such as general database systems, the generation of interactive dashboards (Figure 2.2), the implementation or use of RESTful APIs for web services, ... So we have opted for the development from scratch of three bioinformatics modules based on the Node-RED platform.

The Node-RED platform has been developed in node.js, a multiplatform Javascript [4] runtime environment, which gives us the possibility to create data workflows visually and flexibly since it gives us access to thousands of modules developed by the community thanks to its open-source architecture.

The methodology that will be used for the development of these Node-RED modules will be based on the coding of several sets of bioinformatics Javascript scripts. Each Node-RED module is based on a subset of nodes, and each node involves the development of one script that implements one specific functionality. These functionalities will be implemented natively with the Javascript programming language or by interfacing with other bioinformatics libraries, whether implemented in Javascript or Python [5].

Instead of coding all the modules at the same time, and later testing and debugging the whole application, we decided to choose an iterative and incremental methodology, implementing, testing, and documenting each module (and each node inside the modules) individually,

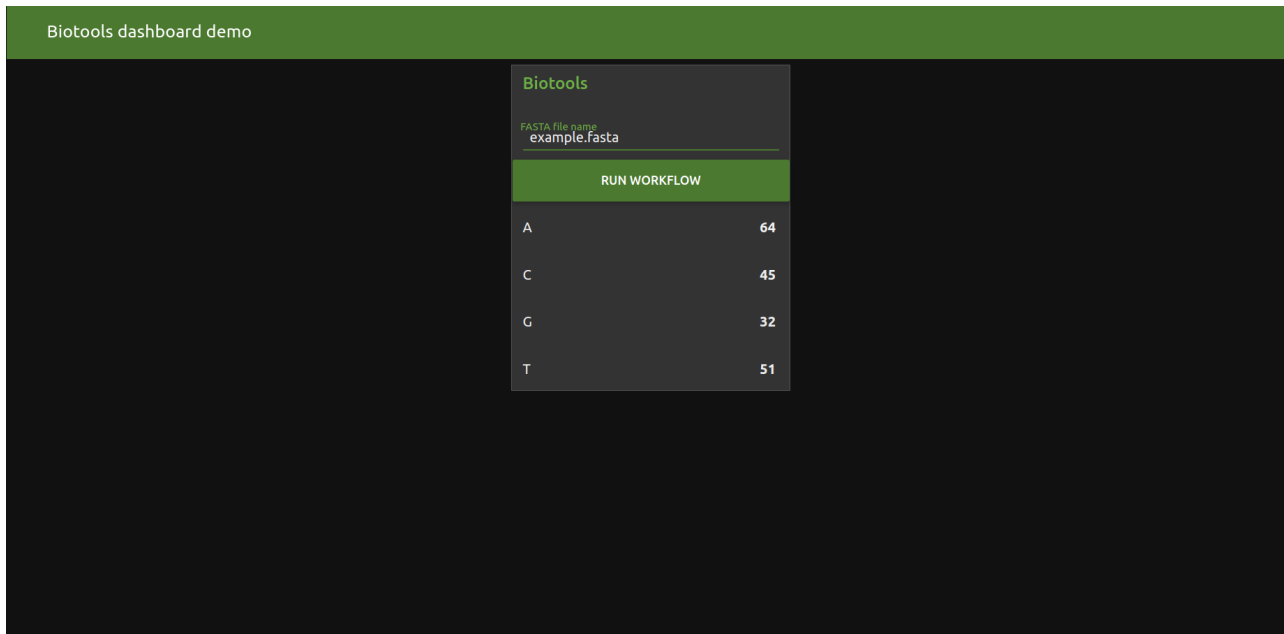


Figure 2.2: Counting nucleotides webapp using Node-RED dashboard nodes

iterating the procedure if required, and ending the process with an integration testing of all the modules to check if the application is working as expected.

Finally, the finished tool will be publicly available to the community using online open resources.

The multiplatform and open-source architecture of those nodes and modules are key elements concerning the power, flexibility, and possibilities of this software tool, offering to us the possibility to develop bioinformatic functionalities and taking advantage of all the power that the Node-RED tool delivers natively to us.

2.5 Planning

2.5.1 Main Tasks

To achieve the objectives defined in the previous sections, the project has been divided into different, sequentially ordered tasks:

- Analysis of the available IT resources (5 days).
- Development of the software (45 days).
 - First module development (basic sequence reading and sequence processing) (14 days).

- Unit testing. Risk analysis breakpoint (5 days).
- Development of the rest of the modules and unit testing. (20 days)
- Software integration tests. (6 days)
- Software documentation. (45 days)
- Setup and configuration of the online resources, allowing appropriate interaction with the educational and professional communities. (10 days)
- Preparation of the MTP report. (60 days)
- Preparation of the virtual presentation. (5 days)

It's important to remark that the first version of this project's source code and documentation has been released publicly through the node.js package manager, and has been available to the community as an open-source software.

2.5.2 Resources

As the MTP deliverable is an open-source software, the needed resources are freely available over the internet as they are also open-source tools and software development tools. For the project development we needed:

- An Internet connection.
- Access to programming software like Javascript and Python (open-source).
- Access to Node-RED software (open-source)
- Access to node.js runtime environment and node.js package manager (open-source).
- A local or remote server. A regular laptop or PC is enough.
- Access to Git infrastructure for version control and interaction with the final users like GitHub [22].

From a philosophical and ethical perspective, open-source software is a kind of software that respects the freedom of users and the community, meaning that the users are free to run, copy, distribute, study, modify, and improve the software.

2.5.3 Calendar

In order to schedule the different project tasks and milestones, a Gantt chart (Figure 2.3) and a detail from tasks schedule (Figure 2.4) has been developed using the open-source Planner [23] application.

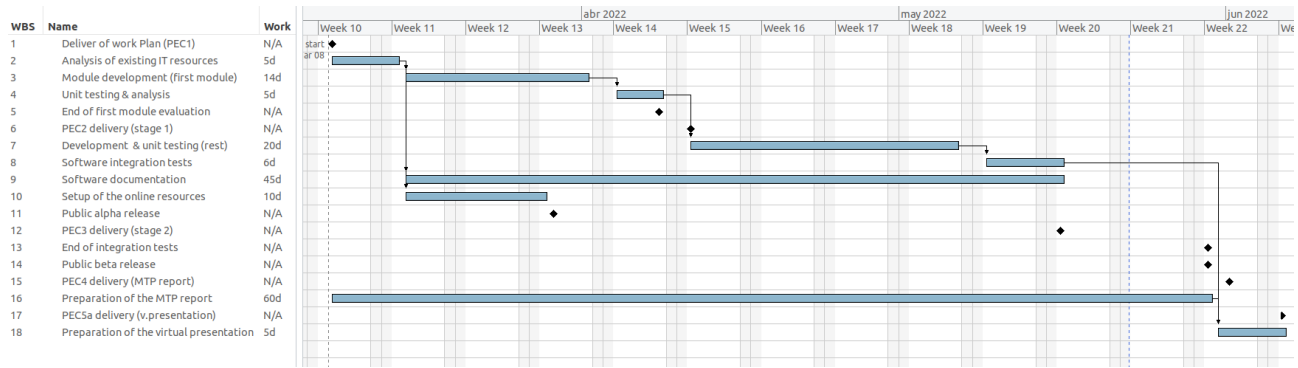


Figure 2.3: Gantt diagram of project tasks and milestones

WBS	Name	Work
1	Deliver of work Plan (PEC1)	N/A
2	Analysis of existing IT resources	5d
3	Module development (first module)	14d
4	Unit testing & analysis	5d
5	End of first module evaluation	N/A
6	PEC2 delivery (stage 1)	N/A
7	Development & unit testing (rest)	20d
8	Software integration tests	6d
9	Software documentation	45d
10	Setup of the online resources	10d
11	Public alpha release	N/A
12	PEC3 delivery (stage 2)	N/A
13	End of integration tests	N/A
14	Public beta release	N/A
15	PEC4 delivery (MTP report)	N/A
16	Preparation of the MTP report	60d
17	PEC5a delivery (v.presentation)	N/A
18	Preparation of the virtual presentation	5d

Figure 2.4: Detail of tasks and milestones schedule

2.5.4 Milestones

This project has several milestones based on the teaching plan of the MTP and the scheduled deliveries.

- Delivery of the work plan (PEC1)
- End of first module development (*see risk analysis subsection*)
- Delivery of the first stage of the project, including the first modules and their associated documentation (PEC2)
- Public alpha release.
- Delivery of the second stage of the project, including the rest of the modules and their associated documentation (PEC3)
- End of integration tests. Beta public release and public testing.
- Public beta release
- End of MTP report (PEC4)
- End of virtual presentation (PEC5a)

These items are also marked as milestones in the previous section Gantt chart (Figure 2.3) and the detailed view of tasks and milestones (Figure 2.4).

2.5.5 Risk analysis

In the development process of a new tool from scratch within a new developer's knowledge area, there is always a certain level of uncertainty about the risks and problems that may appear, like:

- Development problems related to the initial chosen framework.
- Interconnection problems with third party services.
- Module integration problems.

Consequently, several elements have been foreseen to minimize the impact of these potential problems:

- An early analysis of the first module development, allowing the early assessment of the potential issues, and if needed, make some modifications to the original work plan when there is still time to redefine the course of the project.
- Keep the project within the defined objectives, but with a certain level of adaptability in the development, considering that it's an open-source software, accessible to the bioinformatics community, and it's intended to go beyond the end of this MTP.

2.6 Expected results

At the end of the project, we obtained different elements as part of this Master/Thesis Project (MTP), on the one hand, we delivered:

- The work plan.
- The memory of the MTP.
- A virtual presentation for its defense.
- The final software product and its associated documentation.

The final product deliverable is based on a series of bioinformatics modules of a general range developed in Javascript and HTML that will be integrated transparently with the Node-RED visual programming platform.

These modules integrate different functionalities related to the field of genetics and bioinformatics, grouped in different categories:

- The reading of different kinds and sources of bioinformatics data, like Fasta, GFF, Clustal, genebank, or UniProt file formats.
- Conversions between different data types.
- Sequence and string manipulation.
- Sequence management (complement, reverse complement, transcription, translation, counting, motif localization, ...).
- Sequence alignment (hamming distance, calculation of the relationship between transitions and transversions, ...).

It is important to emphasize that the different functionalities of the modules are not limited to the features mentioned in this section, since the software is freely available and publicly accessible from the official node.js and Node-RED module repository, we want to encourage the collaboration of the bioinformatics community to improve and extend the functionalities and features of the project progressively.

For this reason, a GitHub repository <https://github.com/ferrithemaker/biotools> is fully available to the community with all the project source code, technical documentation, a communication system with the users, and an issues tracking system.

Detailed information about the final results is available in chapter 5. *Results*.

2.7 Structure of the MTP

The following chapters of this report detail the development process followed to create the software described in this chapter of the MTP.

In particular, we will detail the analysis of the existing resources and services, the methodology used and the results of the different phases of the project development, including several real study cases of the implemented software tool.

We will also discuss and analyze the results of this project, talking about the strengths and weaknesses of the final software from the technical and biological point of view, to the economic assessment of the project.

Finally, we will add a concluding chapter that will contain the main reflections, the lessons learned, and problems encountered, as well as a commentary on the most relevant or noteworthy elements in the development of this project.

The conclusions chapter will also point out the possible lines of future development of the project, as the software is intended to be modular and expandable through the implementation of new future functionalities. This will allow that the software development does not remain stagnant or closed at the end of this MTP but can continue under development over time, evolving to remain useful in the bioinformatics and educational sectors.

Chapter 3

State of the art

3.1 Different types of bioinformatic tools

There are currently a large number of programs, tools, and utilities used by the bioinformatics community for the study and analysis of genomic data.

There are different selection criterias For this MTP we used a criteria based on the main functionality of the too, and we have divided them into three major groups, data sources, specialized software, and integration services.

Each of these categories offers a wide variety of applications and services that often include a large number of tools, usually including an interconnection system with third party services and specialized features, because of this, it's impossible to make a very compartmentalized and strict division of those tools, however, we tried to make an indicative grouping on the basis of their main functionalities.

3.1.1 Data sources

3.1.1.1 Primary sources

The primary sources are databases repositories of nucleotide sequences from all organisms with original sequence data defined by INSDC (International Nucleotide Sequence Database Collaboration) [31].

This databases are:

- DNA Bank of Japan (National Institute of Genetics) [32]
- EMBL (European Bioinformatics Institute) [33]
- GenBank (National Center for Biotechnology Information)[34]

3.1.1.2 Secondary sources

The secondary services and data sources are mostly available through web pages or web services and as the primary sources, are also focused - although not exclusively - on providing primary, raw, or base information to the bioinformatic users.

Those services make available a huge amount of information in the form of structured and processed data, using different kinds of databases, file formats, and information exchange protocols to facilitate their use among the bioinformatics community.

But these kinds of tools are not only focused on providing raw information to the user, but also allowing can have a strong focus on the representation, analysis, and processing of the data. Usually, these applications also allow some kind of interaction and information sharing features between different services, but those tasks usually have to be performed in a non-automated or semi-automated way, which means that they can sometimes be slow and complex.

We could define as secondary data sources, platforms like the UCSC genome browser [6], the ENCODE project [7], the ENSEMBL genomic browser [8], JASPAR [9], or the NCBI Gene Expression Omnibus (GEO) (Figure 3.1) [10].

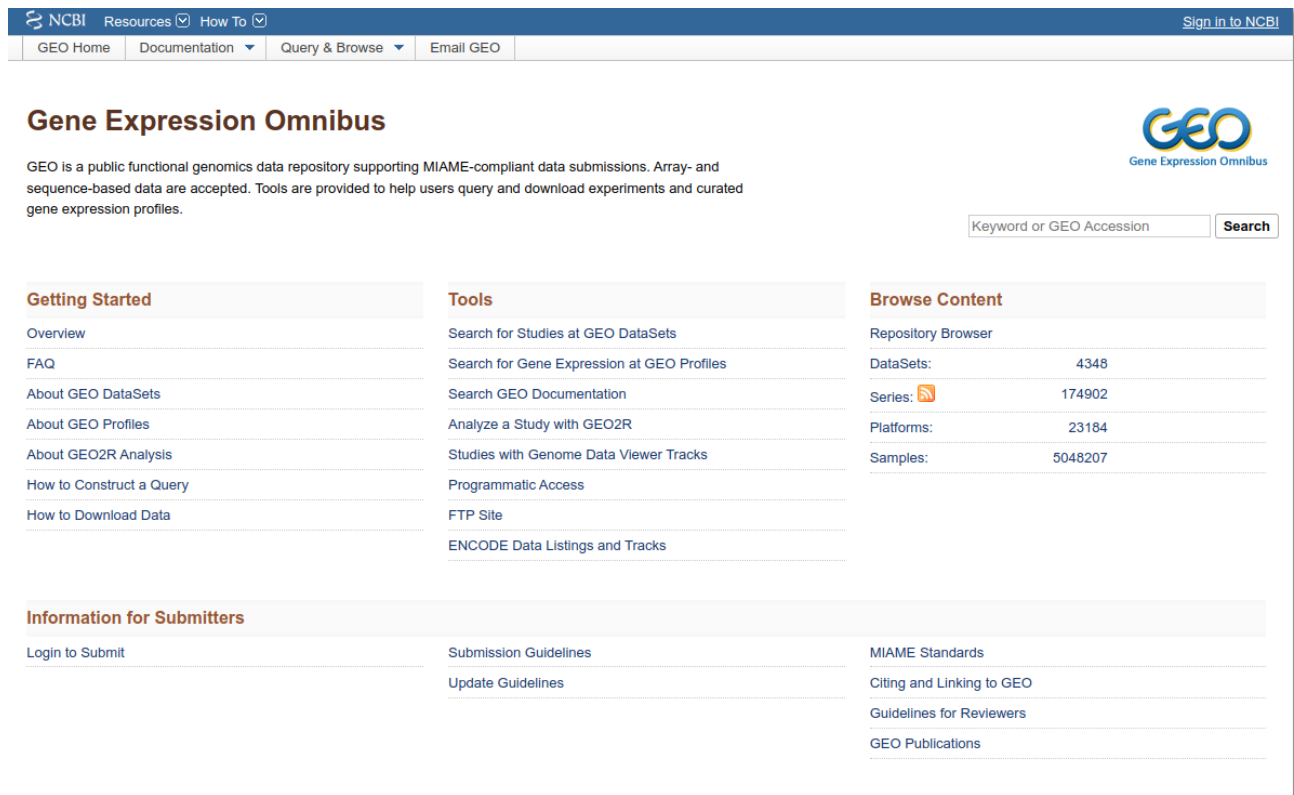


Figure 3.1: Gene Expression Omnibus main website

3.1.2 Specialized software

In the field of genetics and bioinformatics, we can find different sets of tools that are responsible for performing specific and specialized tasks, usually based on data extracted from the available primary data sources.

Among those that we could classify as specialized tools, we can find applications like the GENEID prediction software [11], the BLAST alignment tool [12], the MEME motif finder [13], or different existing proprietary tools related to commercial genomic sequencing devices, among many others.

3.1.3 Integrative software

We could define the integrative software as the one whose main functionality is the automation of the processing, treatment, integration, and interaction of data between different primary services, data sources, and specialized software, through the creation of information workflows and processes between all of them, allowing to generate complex outputs with new information.

In this regard, we can highlight the Galaxy project [14] as one of the most complete existing platforms for the integration of different bioinformatic tools without the need for programming. Although, there are also other integration tools more oriented to software development, such as R [15] and BioConductor [16] or Python and BioPython [17].

3.1.3.1 Comparison with the Galaxy project

One of the most powerful integrative applications in the field of bioinformatics is the Galaxy project (Figure 3.2). The Galaxy project is currently a reference in the field of bioinformatics, the project is developed and maintained by a powerful community and is supported by relevant international institutions like the National Institutes of Health, NSF International [18], the German Federal Ministry of Education and Research [19], Bioplatforms Australia [20] and Australian Research Data Commons [21].

In a certain sense, the project developed in this MTP, as it is also a visual integrative tool through the use of workflows, could be interpreted as a simplified revision of the Galaxy project, but there are several aspects with relevant differences.

First of all, the Galaxy project is a massive project with years of history, developed by professionals with extensive experience and background in the world of bioinformatics, and supported by first-rate world institutions, something that cannot be possibly compared to this MTP project: a few months project from only one student; although, it is important to emphasize that the open, modular and flexible architecture of this project would allow achieving a similar development in the future, in case an active community of developers grows around it.

Secondly, the very characteristics of this MTP project have a clearly differentiated orientation from the Galaxy project. Since it uses the Node-RED as a development platform, a wide variety of easily accessible modules that implement a relevant set of IT functionalities related to information technology, data processing, data storage, and exchange of data are available, and a lot of these features do not exist or are not so easy to implement with the Galaxy project.

In addition, the project of this MTP is also developed from a pedagogical and educational point of view, allowing a moderate learning curve, not only for the bioinformatics professional use but also for use in the education field, either the university or secondary schools.

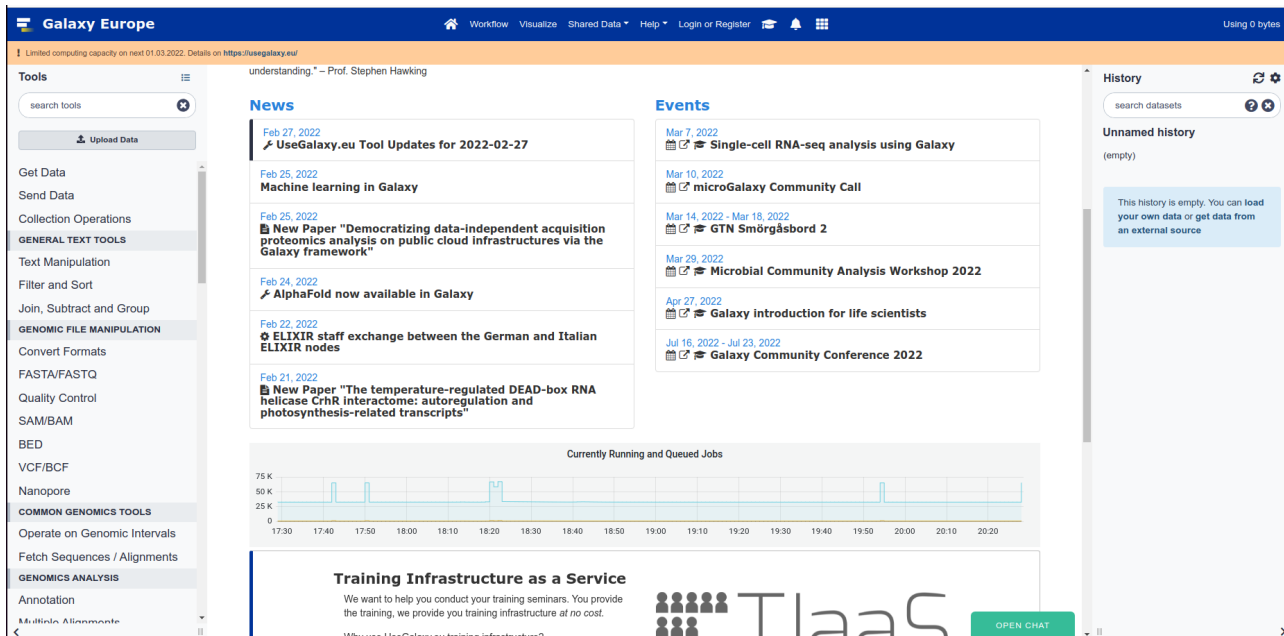


Figure 3.2: Galaxy project main page

Chapter 4

Methodology

As was briefly mentioned in section 2.4 *Approach and methodology* we chose an iterative and incremental methodology (Figure 4.1) for this project, those are a classic software development methodologies and are the inception of the popular Agile software development methodology, which is a combination an evolution of both iterative and incremental methodologies.

We choose the integration of iterative and incremental methodologies because it's very effective, flexible, and easier to manage, update and change the requirements of the project. These characteristics make this kind of methodology adequate, simple, and suitable for this kind of short-term, one-person, software development.

The main concept behind the iterative and incremental methodology for software development is iteration.

As you can find in *Ghahrai, Amir (2016). Software Development Methodologies* [26] each iteration is composed of 4 stages:

- Planning and requirements phase: In the requirements phase, we need to study and analyze all the requirements of the project that we defined in the planning.
- Analysis and design phase: In the analysis and design phase, we will define and design a technical solution that fits the requirements.
- Implementation and testing phase: In the testing phase we want to implement and test the source code resulting from the implementation process.
- Evaluation phase: In the evaluation phase we want to check if the code implemented meets the previously defined requirements and analyze if there are new elements that require some planning reshaping.

As we are using an incremental and iterative approximation, the idea is not to develop all the planned project as a whole, but divide it into functional units (that we call nodes) that can be developed independently.

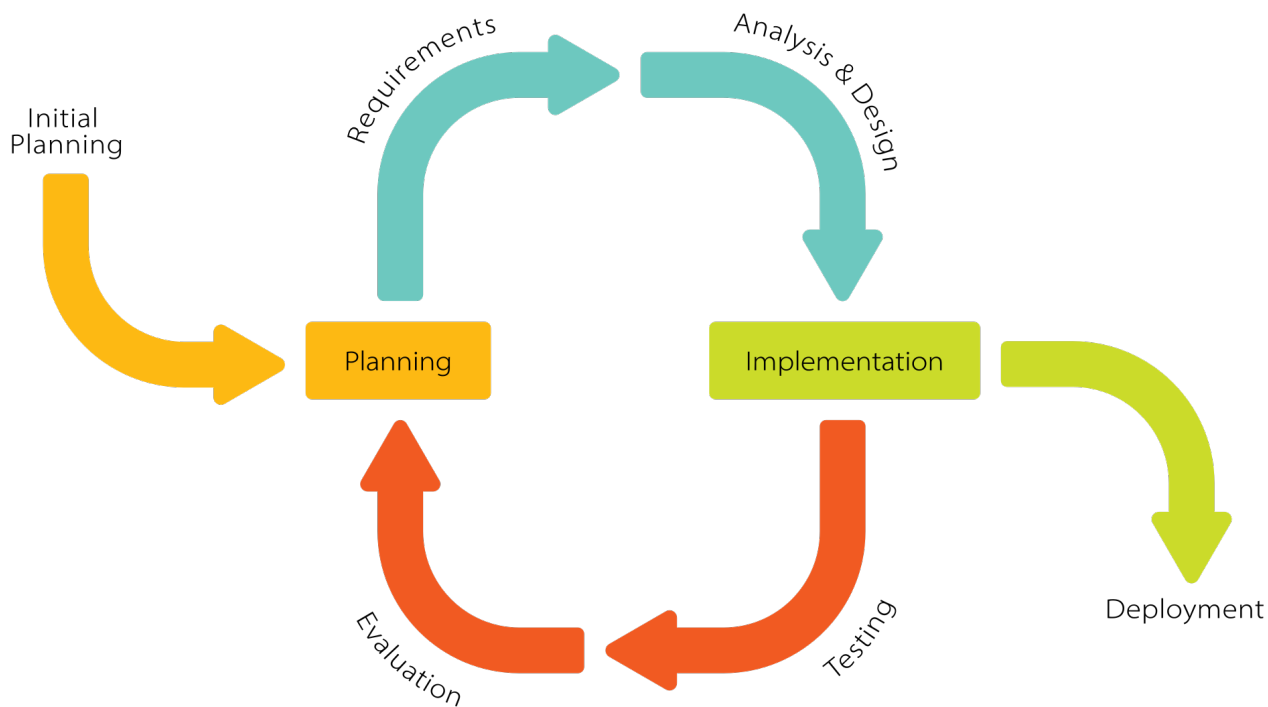


Figure 4.1: Iterative and incremental process diagram (by Krupadeluxe - Own work, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=104924876>)

After each iteration, we can reevaluate the requirements, updating the previous releases if required and incorporating new functional units incrementally.

This process will continue until all functionalities are tested and evaluated and the final software deployment can be released.

It's important to notice that this is an open, ongoing project, and this iterative and incremental process is not closed. At the time of finishing this MTP, the first beta version of the software has been released, but we are still in continuous active development, solving bugs and adding functionalities beyond the scope and the defined features of this document.

Chapter 5

Results

Based on the defined objectives (check section 3. *Objectives*) and the methodology that has been used (check chapter 4. *Methodology*) it has been possible to generate a set of bioinformatics software modules based on the dynamic workflow design platform Node-RED, in addition to all its documentation and a supportive environment designed to facilitate its use and enhance communication between users.

In this sense, the results of the project are:

- Source code of the project modules published on GitHub and available at:

<https://github.com/ferrithemaker/biotools>

- Ready to use modules and nodes published on the NPM repository:

<https://www.npmjs.com/search?q=biotools>

and the Node-RED flows library:

<https://flows.nodered.org/search?term=biotools> containing:

– Biotools module:

- * Preprocessing Raw data node: This node reads and parses a unformatted nucleotide or amino acid sequence.
- * Processing FASTA file node: This node reads and parses a FASTA format file.
- * Processing JASPAR file node: This node reads and parses a nucleotide JASPAR PFM (Position Frequency Matrix) file [27] (Figure 5.1).
- * FASTA file output: This node parses the msg.payload object to create a FASTA format structure, ready to be written by the *write file* Node-RED node.
- * Processing GenBank file node: This node reads and parses a GenBank format file [28].

– Biotools-sequence module:

- * Reverse node: This node returns a reverse sequence of the original nucleotide sequence.
- * Complement node: This node returns the complementary sequence of the original nucleotide sequence.
- * Transcription node: This node returns the mRNA transcription sequence of the original DNA sequence.
- * Translation node: This node returns de amino acid translation of the original mRNA sequence.
- * Get subsequence node: This node returns a partial subsequence of the original input nucleotide sequence.
- * Check degenerate symbols node: This node checks if the original sequence of nucleotides contains degenerate base symbols.
- * Sequence type node: This node returns the nucleotide sequence type.

JASPAR Matrix formats

JASPAR stores transcription factor binding profiles in four formats. Following is more information on formats and the DNA binding profile for GATA6 transcription factor (JASPAR ID MA1104.2) as an example:

Raw PFM

Each matrix is separated by a FASTA-like header starting with the > symbol and then a matrix ID. The count for each base (ACGT) is specified on its own space separated line where each element corresponds to one column. The order of the lines for the bases is A, C, G and finally T.

```

>MA1104.2 GATA6
22320 20858 35360 5912 4535 2560 5044 76686 1507 1096 13149 18911 22172
16229 14161 13347 11831 62936 1439 1393 815 852 75930 3228 19054 17969
13432 11894 10394 7066 6459 580 615 819 456 712 1810 18153 11605
27463 32531 20343 54635 5514 74865 72392 1124 76629 1706 61257 23326 27698
    
```

[Back to top](#)

JASPAR

This is similar to the raw format, having an identical header. The lines for each base however start with a label for the nucleotide (A, C, G or T) and then the columns follow enclosed in brackets: [].

```

>MA1104.2 GATA6
A [ 22320 20858 35360 5912 4535 2560 5044 76686 1507 1096 13149 18911 22172 ]
C [ 16229 14161 13347 11831 62936 1439 1393 815 852 75930 3228 19054 17969 ]
G [ 13432 11894 10394 7066 6459 580 615 819 456 712 1810 18153 11605 ]
T [ 27463 32531 20343 54635 5514 74865 72392 1124 76629 1706 61257 23326 27698 ]
    
```

Figure 5.1: JASPAR PFM format from <https://testjaspar.uio.no/docs/#raw-pfm-format>

- Biotools-analysis module:
 - * Sequence count node: This node counts the number of nucleotides of the sequence [Adenine / Cytosine / Guanine / Thymine / Uracil / All] (degenerated or not degenerated).
 - * Hamming distance node: This node calculates the hamming distance between two not degenerated nucleotide sequences of the same size.

- * Transition transversion ratio node: This node calculates the transition / transversion rate between two not degenerated nucleotide sequences.
 - * Alignment analysis node: This node checks the alignment between two nucleotide sequences using the needleman-wunsch algorithm [30].
 - * Motif PFM score node: This node calculates the score of binding sites between a JASPAR PFM (Position Frequency Matrix) and a nucleotide sequence.
- Documentation (Figure 5.2): available at <https://github.com/ferrithemaker/biotools/wiki>
 - Mailing list and the associated online message board, for discussion, updates and information about the project at biotools-node-red@googlegroups.com.
 - Beta version of the software (Figure 5.3): available at <https://www.npmjs.com/package/node-red-contrib-biotools>
 - Issue tracker: available at <https://github.com/ferrithemaker/biotools/issues>

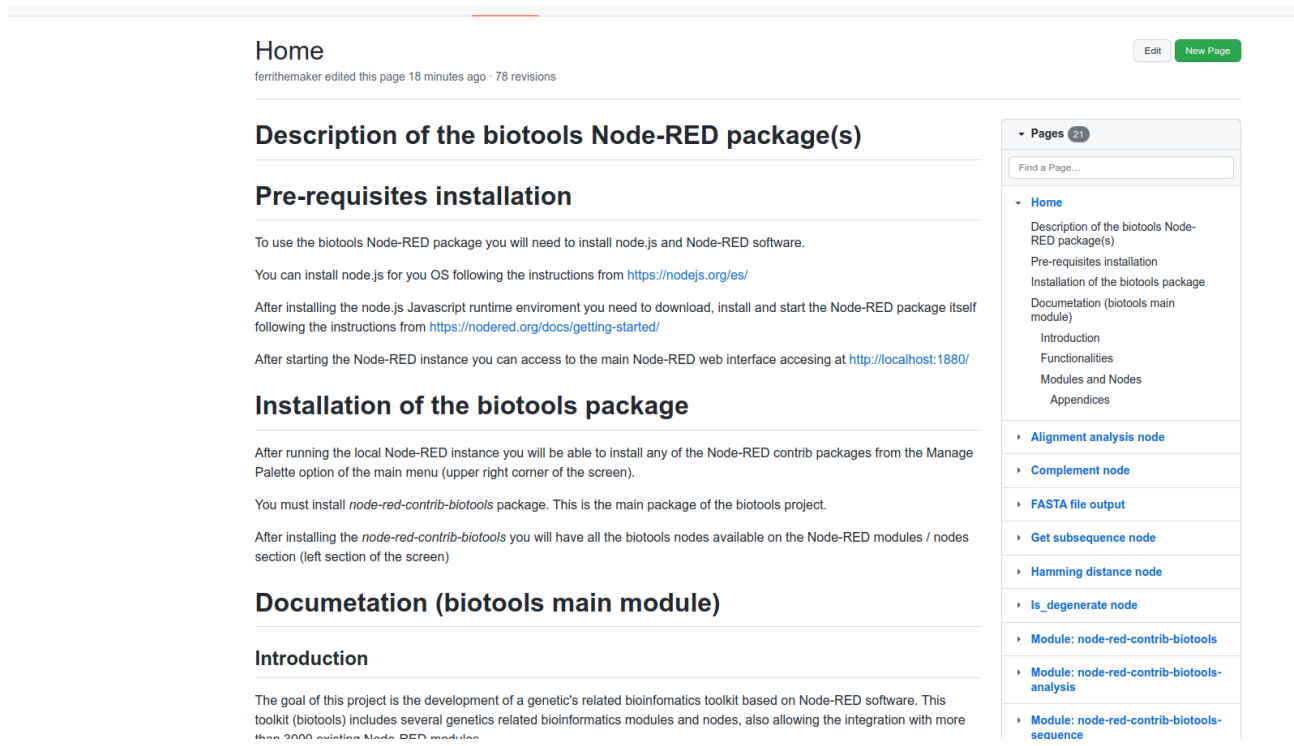


Figure 5.2: Project documentation website

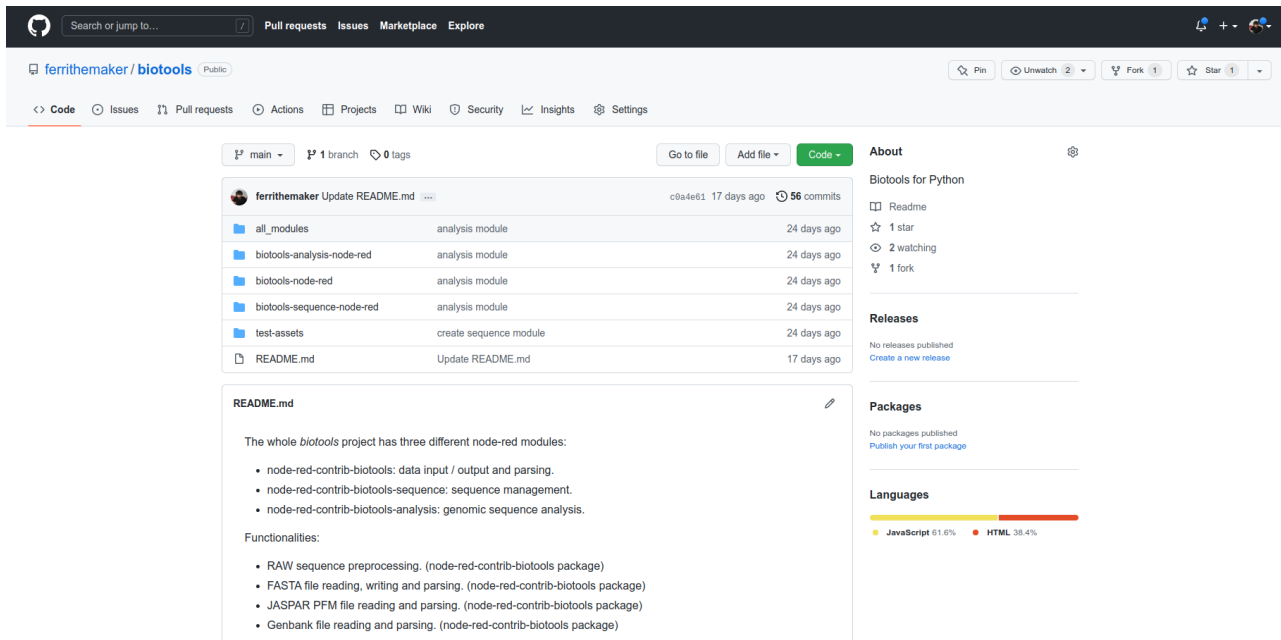


Figure 5.3: GitHub repo of the project

5.1 Study cases

With the objective to show some practical applications of this project and its associated software, a set of case studies have been proposed, from the biological field and also from the educational field.

5.1.1 Biological study case

A biotechnology laboratory requires the rapid development of a visual tool for the comparative analysis of genomic sequence alignment and motifs score calculation based on information stored in an external relational database.

The basic requirements of this development will be:

- The development time must be short and the tool must be flexible and adaptable to later modifications.
- Adapted for use by non-bioinformaticians.
- The base information will be stored in a relational database.
- The developed tool must be able to perform a comparative analysis of genomic sequences and search for motifs.

- There must be a friendly graphical visual environment (GUI) both for its use and for its future development adaptations or modifications (it should not require complex programming knowledge)
- The results of the comparative analysis will be stored by default in a public MongoDB server (a noSQL external database).
- It should be possible to download the information in plain text files like FASTA.

Through the use of the software developed in this MTP, we can implement a bioinformatics tool that would meet the stated requirements, allowing us to:

- Use a visual bioinformatics development environment (Figure 5.4), with no programming knowledge required (although it's recommended).
- Easily integrates the Node-RED SQL and noSQL database modules into the tool, which will allow us to access a huge range of relational and no relational databases, both for reading and writing information.
- Retrieve and parse FASTA formatted data.
- Perform a sequence alignment analysis using the using the needleman-wunsch algorithm.
- Retrieve and parse JASPAR PCM formatted data.
- Convert frequency data from PCM to PWM.
- Calculate PWM/PCM motif score.
- Create a visual environment for the management and user experience of the developed tool, by using the *dashboard node* available in Node-RED.
- Import and export data in different formats based on user requirements through the features developed in this MTP software (i.e., export to FASTA) or the use of native Node-RED modules (i.e., export to EXCEL)

5.1.2 Educational study case

In an educational center, a biology teacher wants to carry out some computer practices with his students about protein synthesis.

The requirements would be:

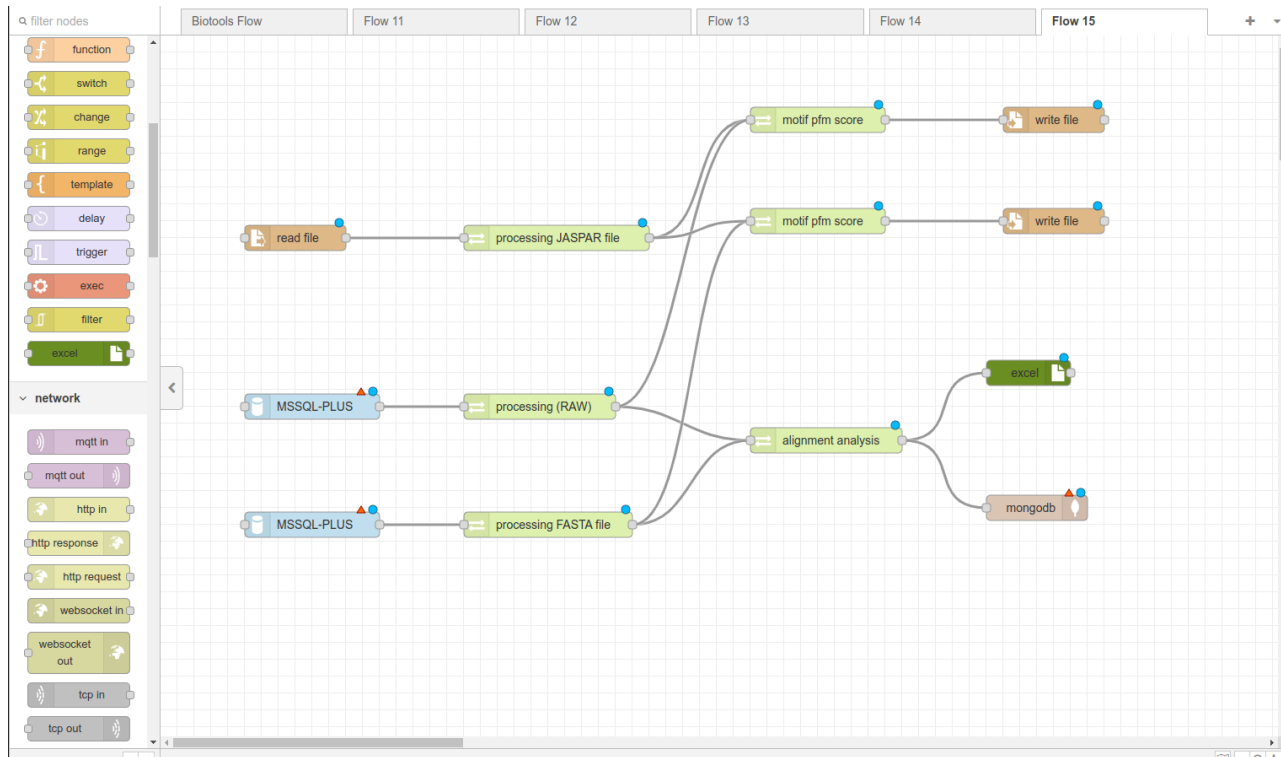


Figure 5.4: Practical example of the biological study case with alignment analysis and motif score functionalities

- Rapid and visual development of a simple tool for the development of computer practices related to protein synthesis (complement, reverse complement, transcription, translation)
- Ability to easily share the tool with students without the need to install complex software.
- Flexibility of the tool, that must be specifically adapted to the needs of the teacher.
- Students should be able to share results easily.

Thanks to the software implemented in this MTP, the teacher, and the students could:

- Use Node-RED’s native development environment to develop your tool visually, quickly, and without programming.
- Thanks to Node-RED’s ability to export workflows in JSON, it is possible to easily share the program and data between teacher and students.
- Integrate the different processes in a single workflow, as opposed to the use of different online tools that are not natively interconnected and without requiring the use of highly complex online and offline bioinformatics tools.

- Use the functionalities developed in this MTP through the implementation of various nodes/modules to carry out the required bioinformatic processes associated with protein synthesis (complement, reverse complement, transcription, translation, ...)
- View the entire data transformation process of the nucleotides and amino acid data associated with the protein synthesis step by step, thanks to the Node-RED workflow architecture, boosting the educational use of the tool.

5.1.3 Practical example

With the aim of showing a practical example of the software, we set up a simple case based on the educational study case.

The question to solve is:

”We have a FASTA formatted file with two nucleotide sequences, the second one has 9 nucleotides of the positive strand of a genomic sequence, we know that the gene is on the negative strand of the sequence and we want to know the amino acids that will be coded by the codons of the gene sequence.”

The Figure 5.5 shows the implementation of a simple workflow using this MTP software that implements an elegant solution to the previous question.

The workflow does:

- File reading using the native Node-RED *read file* node.
- Parse the requested FASTA information from multiFASTA file and process de nucleotide information. The sequence number is defined in the node’s *sequence number* parameter of the *Processing FASTA file* node. (check <https://github.com/ferrithemaker/biotools/wiki/processing-FASTA-file-node>). The reading is the short nucleotide sequence *TGAACCAGA*.
- As the gene is on the negative strand, we need to reverse and complement the original sequence. (*TGAACCAGA* \gg *AGACCAAGT* \gg *TCTGGTTCA*)
- Transcribe the sequence from DNA to mRNA (*TCTGGTTCA* \gg *UCUGGUUCA*).
- Translate the sequence from mRNA to an amino acid sequence (*UCUGGUUCA* \gg *SGS*).
- Outputs the final and partial results on the screen.
- Finally, the information has been saved to an output FASTA formatted file using the *FASTA file output* node and the native Node-RED *write file* node.

Information about nucleotide and amino acid sequences can be found at:

<https://github.com/ferrithemaker/biotools/wiki#appendices>

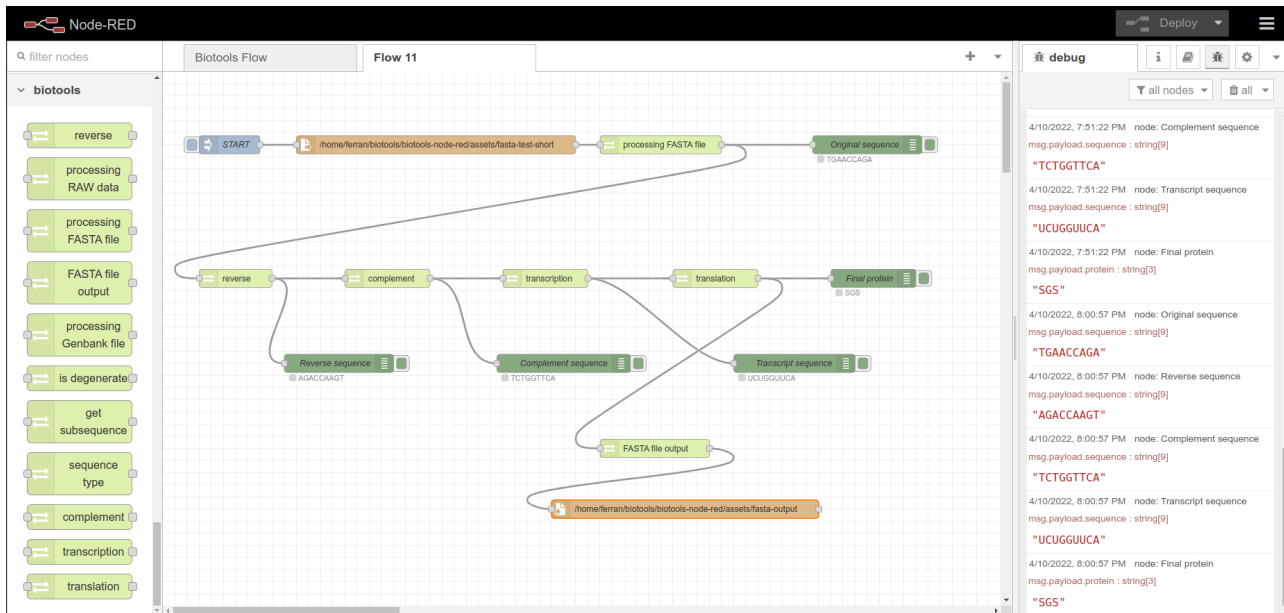


Figure 5.5: Practical example of basic genomic sequence management using Node-RED and the developed module

5.1.4 Integrative study case

One of the objectives of the project was the integration of the native Node-RED tools with the newly developed bioinformatics nodes. This integration can be noted in the subsection *1.6.1 Biological study case*, allowing to read and write several file formats like excel or accessing a different kind of database data servers.

But in this study case, we want to analyze how to use the native Node-RED modules to access a very popular bioinformatic service like ENSEMBL [29], using their available RESTful API, which provides a programmatic way to access the information on the ENSEMBL database (Figure 5.6).

The goals of this study case are:

- Get data and associated information from the GenBank file.
- Use identifiers from the GenBank file to find the ENSEMBL database information related to them (gene, transcript, protein, ...)

Using the software implemented in this MTP, we could:

- Retrieve the information from the GenBank file using the GenBank processing node.
- Read the requested id from data using the *split node* from Node-RED available nodes.

- Sending an HTTP Request to the official ENSEMBL RESTful API using the Node-RED native *HTTP Request* node.
- Save the data to a custom file using the Node-RED *write file* node.

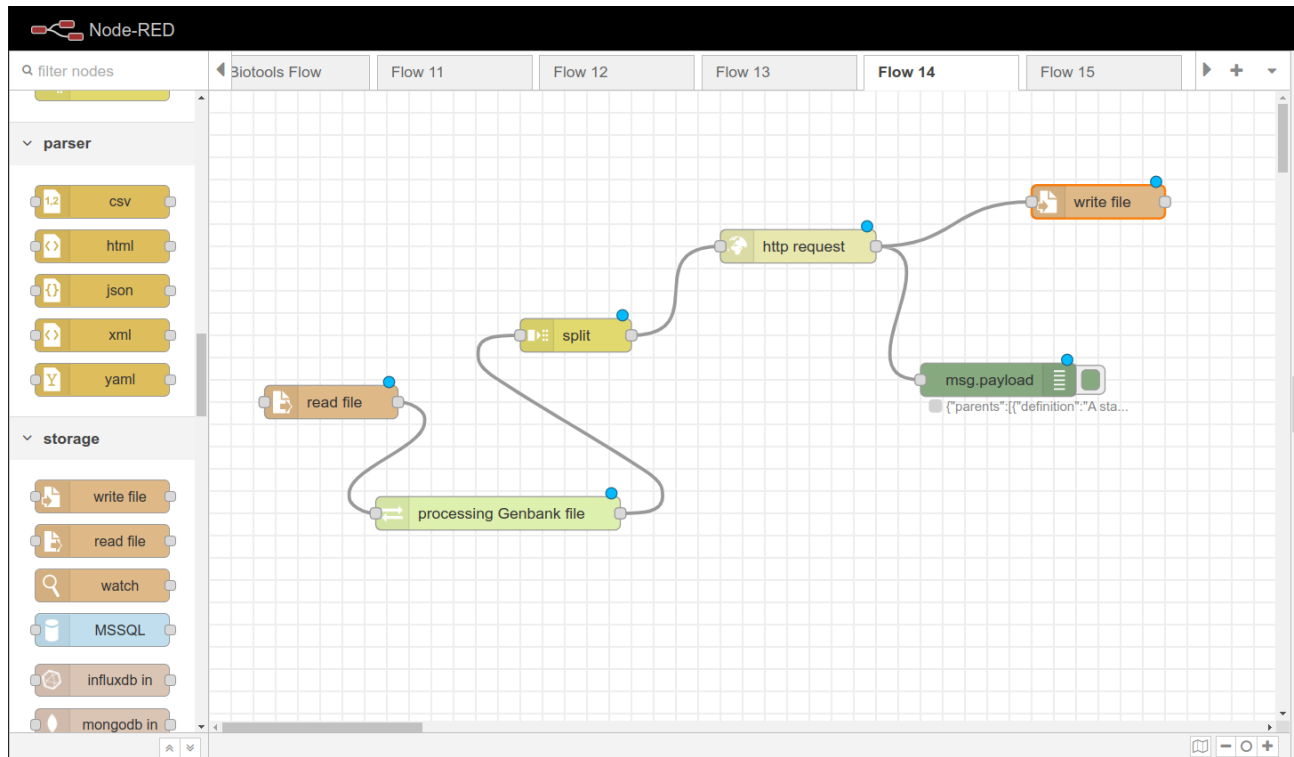


Figure 5.6: Practical example of access to ENSEMBL RESTful API using Node-RED

Chapter 6

Discussion

During the development of the project of this MTP, we have tried to achieve the purpose of implementing a limited but coherent set of bioinformatics functionalities, built on Node-RED, a tool for creating dynamic information visual workflows based on a web environment.

The objective of the project has never been to exhaustively cover and implement all or most of the functionalities that are used in the current bioinformatics background (check a consistent list of open-source bioinformatics software at https://en.wikipedia.org/wiki/List_of_open-source_bioinformatics_software) (Figure 6.1), since it would be impossible to make any kind of tool similar to the Galaxy Project (check *Subsection 3.1.3.1 Comparison with the Galaxyproject*). In this sense, we tried to:

- Provide a different approach to the design of visual workflows.
- Promote the interaction between bioinformatic tools and non-bioinformatic tools.
- Create a tool with smooth learning curve.
- Enhance the interaction between the bioinformatics and educative communities with a transversal-oriented application.
- Take advantage of the synergies with the native functionalities offered by Node-RED (check *subsection 5.1.3 Integrative study case*).

For this reason, in the initial requirements definition of the project, we specified a limited but consistent set of tools, leaving aside others important tools in the bioinformatic field, but not required to meet the goals defined in this project, with the aim to carry out a coherent, useful, reasonable, and functional development based on the human resources and time availability.

To bypass this limitation, and thanks to the open-source architecture of the project, we allow any bioinformatics development or research group that finds useful the use of the advantages

List of open-source bioinformatics software

From Wikipedia, the free encyclopedia

This is a list of computer software which is made for bioinformatics and released under open-source software licenses with articles in Wikipedia.

Software	Description	Platform	License	Developer
.NET Bio	Language-neutral toolkit built using the Microsoft 4.0 .NET Framework to help developers, researchers, and scientists	.NET Framework	Apache	Collaborative project
AMPHORA	Metagenomics analysis software	Linux	GPL	?
Anduril	Component-based workflow framework for data analysis	Linux, macOS, Windows	GPL	University of Helsinki
Ascalaph Designer	Computer program for general purpose molecular modelling for molecular design and simulations.	?	GPLv2	Agile Molecule
AutoDock	Suite of automated docking tools	?	GPL	?
Avogadro	C++ (Qt) based molecule editor and visualizer for in computational chemistry, molecular modeling, bioinformatics, materials science, and related areas.	?	GPL	?
BEDtools	"Genome arithmetic" -- manipulation of coordinate sets and the extraction of sequences from a BED file.	Linux	MIT	QuintanLab [®] , University of Utah
Bioclipse	Visual platform for chemo- and bioinformatics based on the Eclipse Rich Client Platform (RCP)	?	Eclipse Public	The Bioclipse Project
Bioconductor	R (programming language) language toolkit	Linux, macOS, Windows	Artistic 2.0	Fred Hutchinson Cancer Research Center
BioJava	Java library functions for manipulating sequences, protein structures, file parsers, CORBA interoperability, Distributed Annotation System (DAS), access to AceDB, dynamic programming, and simple statistical routines	Linux, macOS, Windows	LGPL v2.1	Open Bioinformatics Foundation
BioJS	JavaScript library of components to visualize biological data	Web browser	Apache	?
BioMOBY	Registry of web services	Web browser	Artistic	Open Bioinformatics Foundation
BioPerl	Perl language toolkit	Cross-platform	Artistic, GPL	Open Bioinformatics Foundation
BioPHP	PHP language toolkit with classes for DNA and protein sequence analysis, alignment, database parsing, and other bioinformatics tools	Cross-platform	GPL v2	Open Bioinformatics Foundation
Biopython	Python language toolkit	Cross-platform	Biopython ^[1]	Open Bioinformatics Foundation
BioRuby	Ruby language toolkit	?	GPL v2 or Ruby	Open Bioinformatics Foundation
BLAST	Algorithm and program for comparing primary biological sequence information, including DNA and protein sequences.	Cross-platform	Public domain	National Center for Biotechnology Information
CP2K	Perform atomistic simulations of solid state, liquid, molecular and biological systems, written in Fortran 2003.	?	GPL and LGPL	Free open source GNU GPLv2 or later
EMBOSS	Suite of packages for sequencing, searching, etc. written in C	?	GPL and LGPL	Collaborative project
Galaxy	Scientific workflow and data integration system	Unix-like	Academic Free	Collaborative project
GenePattern	Scientific workflow system that provides access to hundreds of genomic analysis tools	Unix-like (public server); Linux, macOS, Windows	MIT	Broad Institute, UC San Diego

Figure 6.1: List of available open-source bioinformatics software

of the visual workflow design architecture through Node-RED, to expand the functionalities based on their needs and interests, allowing further tool improvements.

Since the software has been released under GPL license (check *Chapter 7 Economic assessment*), any development that uses the same software must be released with the same license, and this allows the project to be enriched with the collaboration of third parties, allowing that these new functionalities can be also available to everyone.

Consequently, it should not be surprising that the implementation of relevant and popular functionalities in the bioinformatic environment (for example, the reading and writing of BED formatted files) have not been planned. It is important to insist that in these cases the tools have been excluded for reasons of coherence in the development of the rest of the functionalities and they can always be added later if they are relevant in the future.

There are also other highly complex functionalities, which have not been planned in the initial requirements of the project, in this case, the main reasons are:

- Temporal limitations.
- Human resources available.
- Technical aspects:
 - High complexity.

- Extremely consuming resources (CPU / memory) of some functionalities.

One of the purposes of the project was the development of a set of bioinformatics tools that, although limited compared to other cutting-edge tools, were coherent with the previously defined goals.

Let us not forget that one of the initial objectives of the MTP (check *section 2.3 Objectives*) was the implementation of a bioinformatics tool that would not only be useful for the bioinformatics professional sector but would also be useful in the educational community too, so we tried to develop a tool that was relatively simple to use, with a learning curve that was not a barrier to entry for non-experts.

Another relevant aspect to be remarked on is the integration of the development of this software with other tools of a non-bioinformatic type and more related to information technology (IT), data storage, and data management. Many of the current bioinformatics tools do not have integration features with IT tools, certainly caused by the great heterogeneity and variability of those tools, relying on a mostly manual or semi-manual integration.

Thanks to the native tools of Node-RED, the software developed in this MTP allows us to efficiently integrate the bioinformatic functionalities developed with a set of more than 3000 nodes compatible with Node-RED visual workflows (Figure 6.2), which simplifies the possibility of carrying out cross-cutting projects that require not only the use of bioinformatics tools but also information technology resources and services (check *section 5.1 Study cases*).

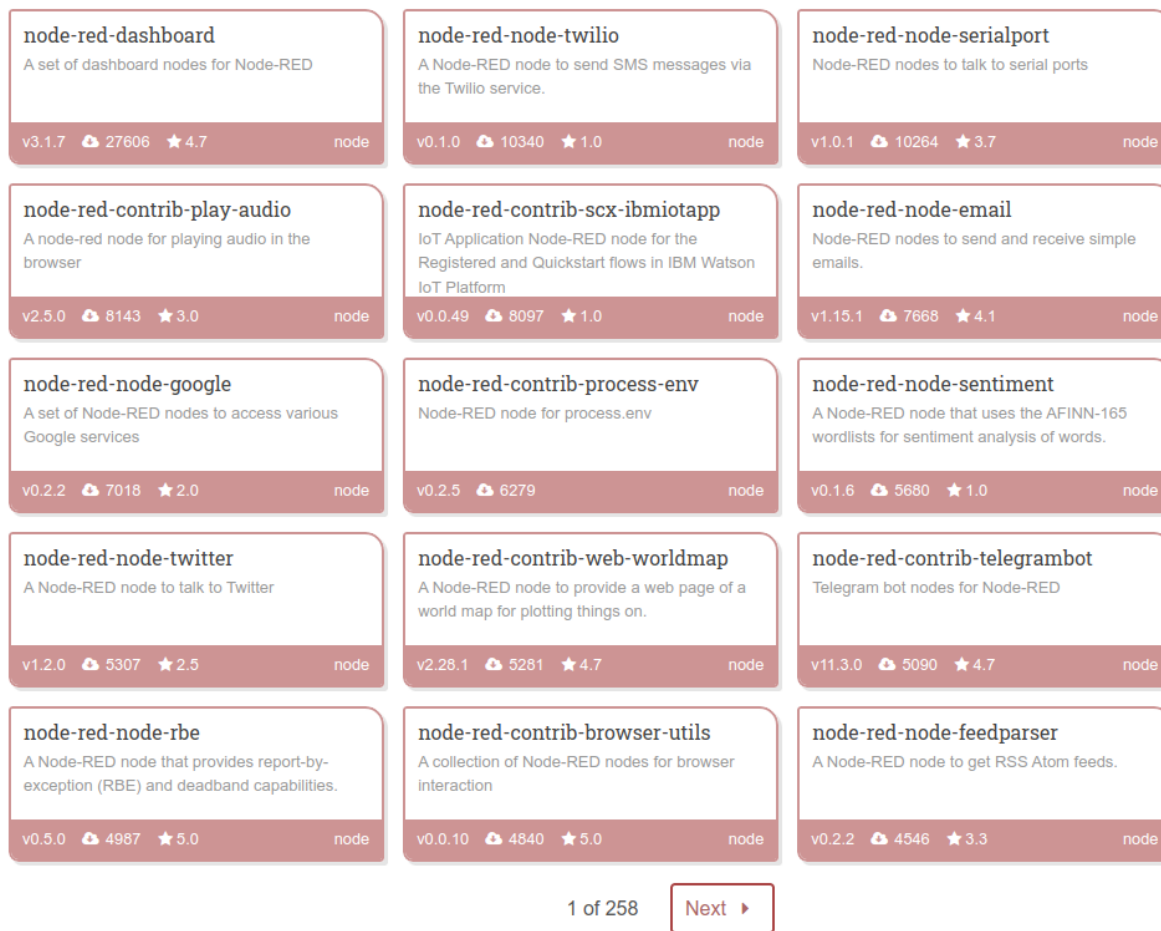


Figure 6.2: List of available Node-RED nodes ordered by number of downloads

Chapter 7

Economic assessment

One of the most important aspects of the development of this software project is that it is part of what is known as "free software".

In this sense, all the software has been released under the GNU 3.0 license [24]. The GNU GPL (general public licence) license (Figure 7.1) allows software users to have a series of freedoms related to the software [25]:

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

(List of freedoms from Free Software Foundation, Inc. 2021 <https://www.gnu.org/philosophy/free-sw.en.html#four-freedoms>)

Thanks to the specific characteristics of this license, this software can be freely used and adapted by both individual users and companies, with the consequent potential advantages, not only at an economic level but also at a software management and development level.

This fact means that the viability of the product does not depend on purely commercial or economic aspects, but only on the impact and relevance that it may have in the global bioinformatics community.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

Figure 7.1: Preamble of the GNU3.0 licence from <https://www.gnu.org/licenses/gpl-3.0.html>

Chapter 8

Conclusions

This MTP project has been based on the development of an integrative software bioinformatics tool, which would allow the execution of bioinformatics data workflows through the implementation of a set of genomics and bioinformatics-related functionalities (check *Chapter 5. Results*).

The software cannot be compared to other integrative software like the Galaxy Project, but we thought that has the potential and flexibility to become a useful tool for bioinformaticians and the educative community.

In the initial inception of this MTP, the software project had as objectives:

- Provide to the bioinformatics community a new visual integrative and dynamic workflow tool through the implementation of a set of node.js and Node-RED web platform software modules.
 - Build three Node-RED modules related to bioinformatics. The modules will be grouped by the functionality of their nodes (reading data from different sources & format conversions, sequences manipulation & management, and sequence analysis) to improve the application structure and coherence. Based on this, the implementation of 3 different modules has been planned (biotools main module, biotools-sequence, and biotools-analysis).

Degree of fulfillment: The planned modules and nodes development has been achieved. Some nodes have been reassigned and/or modified.
 - Bring to the community the documentation about the developed modules.

Degree of fulfillment: The documentation of all three modules is available (check *Chapter 5. Results* section) and the objective has been achieved.
- Improve bioinformatics tools, with the integration of Node-RED's pre-existing set of modules and functionalities.

- Integrate new modules to Node-RED modules repository. Node-RED has more than 3000 modules available, enabling the user to process, transform and interconnect data with thousands of useful services for the bioinformatician.

Degree of fulfillment: The integration of the Node-RED native modules with our bioinformatics software has been achieved.

- Develop a modular bioinformatics tool that brings a moderate learning curve, making the software useful not only in professional environments but also in educational ones.
 - Facilitate the collaboration between bioinformatics and the educational community to improve the application.

Degree of fulfillment: Thanks to the issue tracker, the documentation wiki, mailing list, and online discussion board (check *Chapter 5. Results* section), the objective related to the implementation of the tools, allowing the collaboration between the bioinformatics and educational community has been achieved.

- Bring to the educational community a multilayered easy-to-learn but powerful tool related to genomics.

Degree of fulfillment: This objective has been achieved

Although there have been some modifications and adaptations to the initial planning, we can say that the initial objectives have been achieved. During development, it might have been interesting to add some new features through the implementation of new nodes, beyond those originally contemplated in the planning, but for temporary reasons, it has not been possible. In any case, the open-source architecture of the project allows future extensions, either by the author of the project or by third parties.

From a non-informatics point of view, the development of this genomic bioinformatics tool allowed me to delve into the knowledge of relevant genomics concepts and learn the detailed behavior and operation of several bioinformatics tools related to genomics, helping me deepen my knowledge in aspects of biology and statistics that I did not know when I started this master degree.

8.1 Further actions

In the development of this project, a set of relevant software bioinformatics utilities for professional and educational use have been implemented, those utilities are in accordance with the original objectives of the MTP, but the bioinformatics field is so huge that there are uncountable functionalities that can be further implemented inside the scope of this project.

Therefore, the open-source philosophy of the implementation and the original conception of the project also implies the possibility that the functionalities of the software can be expanded

and improved based on the needs of the bioinformatics community, in this sense it is important to emphasize that the different functionalities of the modules are not limited to the implemented features, since the software:

- Is freely available and publicly accessible from the official GitHub repository of the project, the NPM node.js site and the Node-RED module repository.
- Has an online public forum and an open mailing list allowing the communication between the interested communities.
- Has a wiki website with all the documentation available.
- Has an issue tracker to help users to solve their problems and issues with the software (Figure 8.1).

We also aim to encourage the further collaboration of the bioinformatics community to improve and extend the functionalities and features of the project progressively.

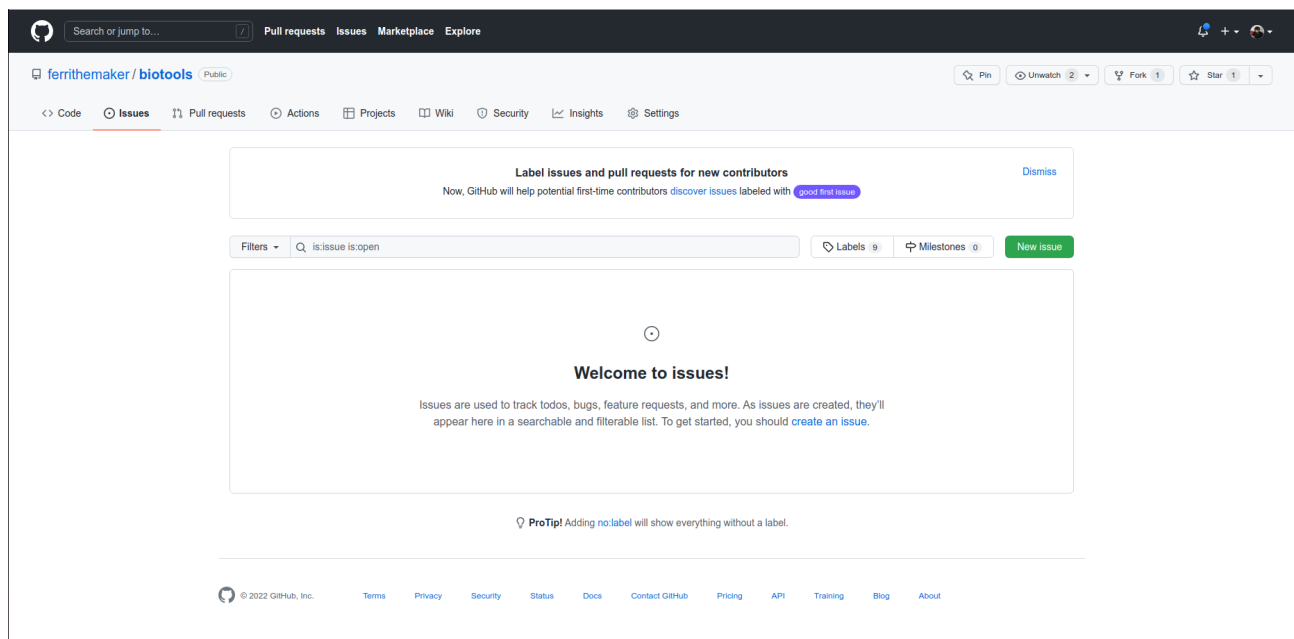


Figure 8.1: GitHub project issue tracker

8.2 Planning follow-up

The project development has generally been quite consistent with the original planning, allowing the successful implementation of the functionalities initially defined and the accomplishment of the objectives of this MTP.

In this project, we used an iterative and incremental methodology (check *Chapter 4. Methodology*). This methodology has been proved adequate for this project since it has allowed us a simple way to define a methodical process for the implementation of the various nodes and modules.

In any case, during the development of the project we introduced several modifications in the implementation of the different nodes and the module organization (Figure 8.2):

- Reassignment of nodes between modules. These modules have been removed from biotools module and added to biotools-sequence module:
 - Reverse: This node returns a reverse sequence of the original nucleotide sequence.
 - Complement: This node returns the complementary sequence of the original nucleotide sequence.
 - Transcription: This node returns the mRNA transcription sequence of the original DNA sequence.
 - Translation: This node returns de amino acid translation of the original mRNA sequence.
- Implementation or redefinition of new modules:
 - Get subsequence: This node returns a partial subsequence of the original input nucleotide sequence.
 - Check degenerate symbols: This node checks if the original sequence of nucleotides contains degenerate base symbols.
 - JASPAR file processing: This node reads JASPAR PCM file format data.
- Removal of nodes planned to be implemented:
 - Clustal file reading: This node has been substituted for the new JASPAR file processing node.
 - GFF file reading: This node has been set temporarily on hold because of the predicted lack of use within the project. The GFF file format is mainly used to locate genomic features, and is not expected to implement this type of functionality yet.
- Creation of a mailing list and the associated online message board, for discussion, updates and information about the project at *biotools-node-red@googlegroups.com*.

As previously mentioned, thanks to the iterative and incremental methodology, it has been possible to progressively redefine the requirements and functionalities to meet the objectives, introducing the necessary changes during the development process.

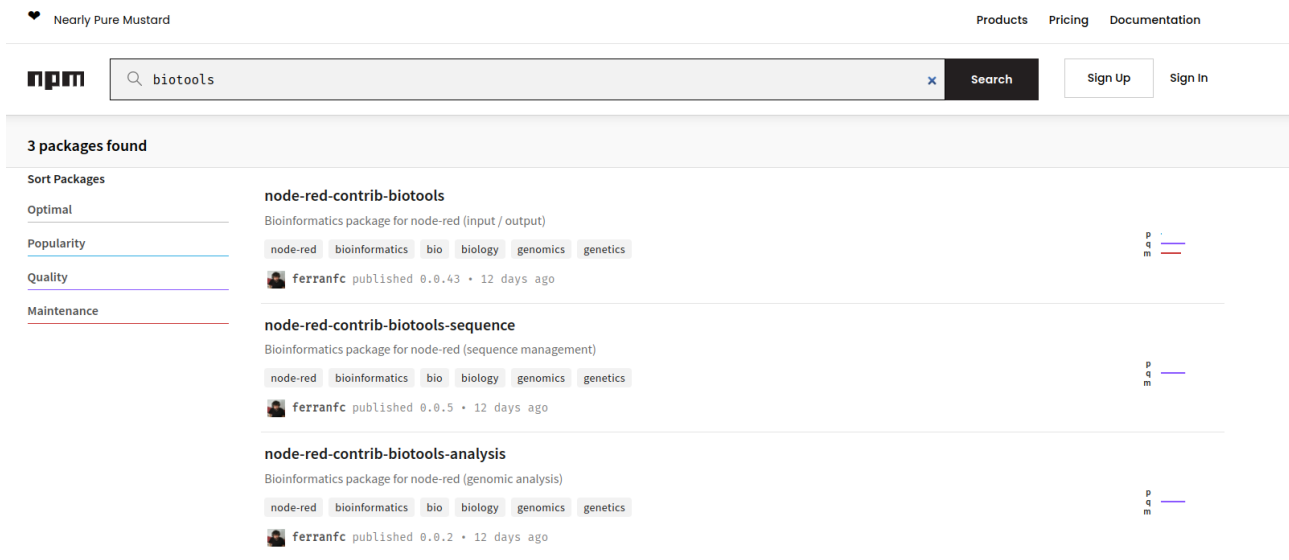


Figure 8.2: NPM repository with the published biotools modules developed in this MTP

Chapter 9

Annexes

9.1 Software documentation

The documentation of all the implemented nodes within the project has been included in the annexes because can be helpful for a better understanding of each node's procedure and behavior.

9.1.1 Description of the biotools Node-RED packages

9.1.1.1 Pre-requisites installation

To use the biotools Node-RED package you will need to install node.js and Node-RED software.

You can install node.js for you OS downloading the software from <https://nodejs.org/> (Figure 9.1)

If you are using some GNU/Linux distribution you can also use their own package managers like *yum*, *RPM* or *apt* ,but note that maybe the versions available on those repositories are not up-to-date.

After installing the node.js Javascript runtime enviroment, that includes the node.js package manager (NPM), you will need to download, install and run the Node-RED package itself executing those comands from Windows Powershell, Windows CMD, GNU/Linux bash shell or MacOS terminal:




- `[sudo] npm install -g -unsafe-perm node-red`
- `node-red`

For more information for you specific platform, check the detailed instructions from: <https://nodered.org/docs/getting-started/>

Downloads

Latest LTS Version: 16.15.0 (includes npm 8.5.5)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

	LTS Recommended For Most Users	Current Latest Features
	Windows Installer node-v16.15.0-x86.msi	
	macOS Installer node-v16.15.0.pkg	
		Source Code node-v16.15.0.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v16.15.0.tar.gz	

Additional Platforms

Docker Image	Official Node.js Docker Image
Linux on Power LE Systems	64-bit
Linux on System z	64-bit
AIX on Power Systems	64-bit

Figure 9.1: Node.js installers depending on OS platform

After starting the Node-RED instance you can access to the main Node-RED web interface accessing at <http://localhost:1880/>

9.1.1.2 Installation of the biotools packages

After running the local Node-RED instance you will be able to install any of the Node-RED contrib packages from the *Manage Palette* option of the main menu (upper right corner of the screen) (Figure 9.2).

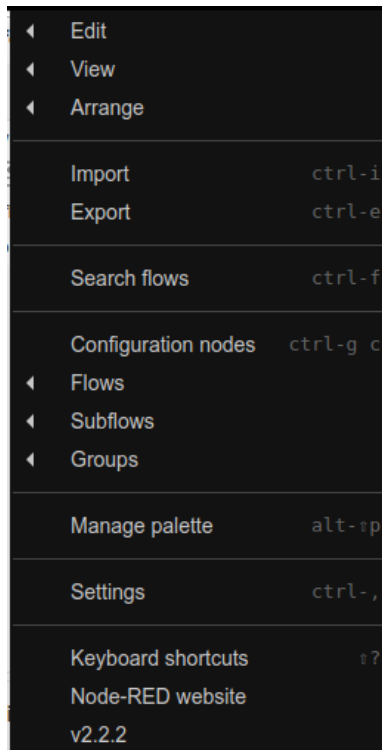


Figure 9.2: Node-RED main menu

You can install any of the available packages (Figure 9.3):

- *node-red-contrib-biotools*
- *node-red-contrib-biotools-sequence*
- *node-red-contrib-biotools-analysis*

The installation of all three packages is recommended.

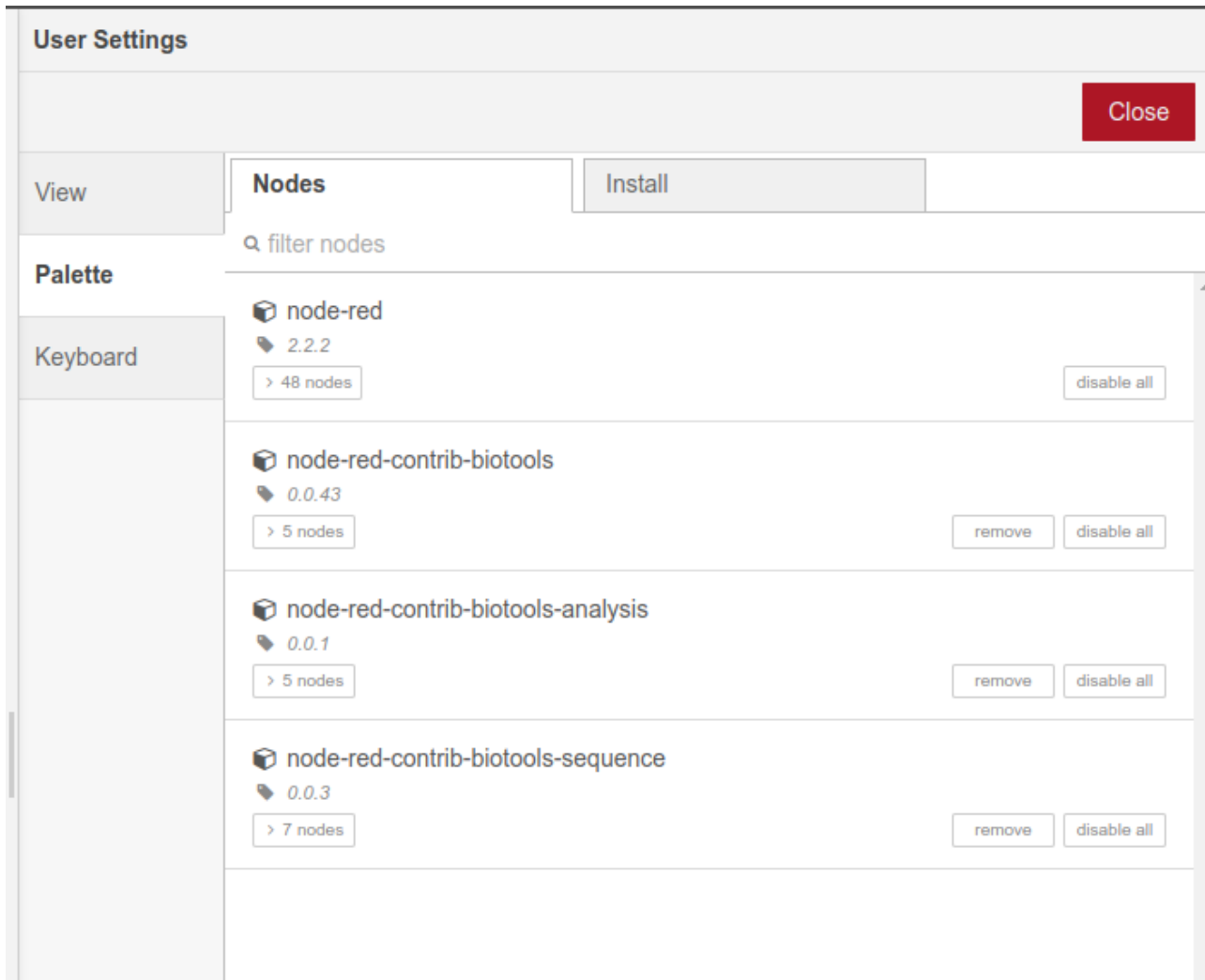


Figure 9.3: Instalation of biotools nodes using Manage Palette

9.1.1.3 Executing flows

After installing the package(s) you will have all the biotools nodes available on the Node-RED modules / nodes section (left section of the screen) (Figure 9.4)

It's important to keep in mind that after every change or update of the flow, it's required to *deploy* (Figure 9.5) the flow to apply the changes.

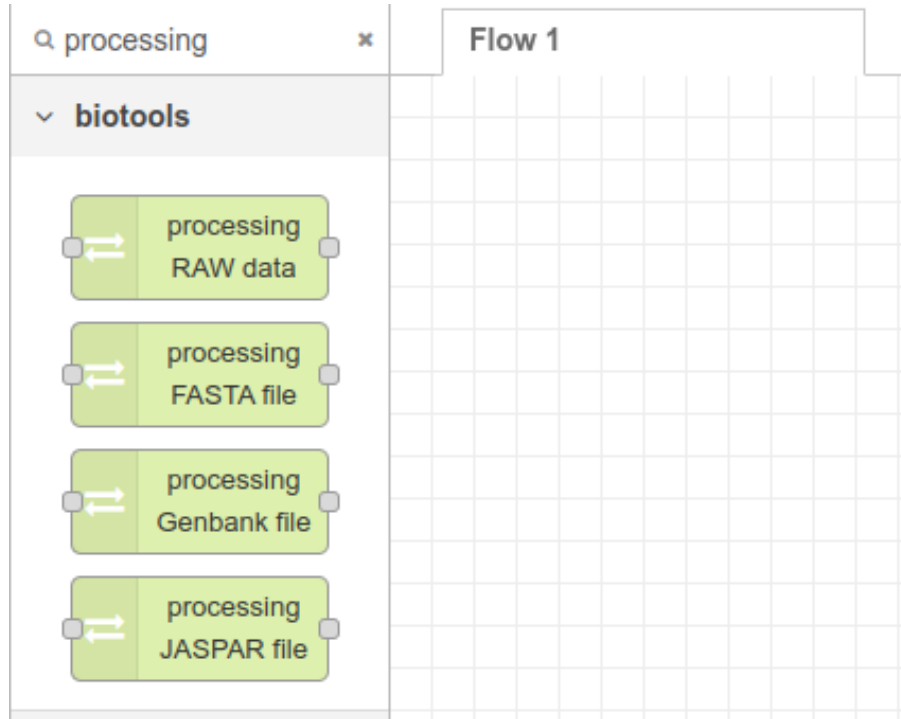


Figure 9.4: Filtered biotools nodes from Node-RED main screen

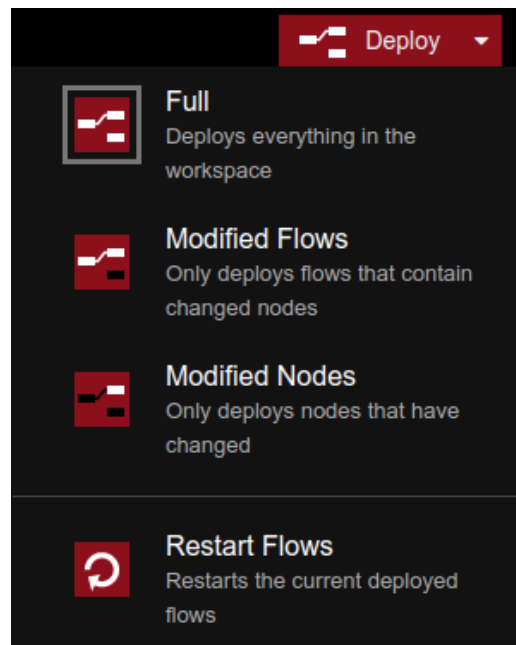


Figure 9.5: Node-RED deploy options

9.1.1.4 Introduction

The goal of this project is the development of a genetic's related bioinformatics toolkit based on Node-RED software. This toolkit (biotools) includes several genetics related bioinformatics modules and nodes, also allowing the integration with more than 3000 existing Node-RED modules.

9.1.1.5 Functionalities

This package is a node-red module implementing bioinformatic tools such as:

- RAW sequence preprocessing.
- FASTA file reading and parsing.
- GenBank file reading and parsing.
- JASPAR file reading and parsing.
- Nucleotide counting.
- Reverse sequencing.
- Complementary sequence.
- Transcription.
- Translation.
- Sequence management.
- Hamming distance.
- Transition / transversion ratio.
- Alignment analysis (needleman-wunsch algorithm).
- Motif analysis.

This package is a beta version for testing only purposes and there is no guarantee that the results will be accurate.

It may contain errors, the author assumes no responsibility or liability for any errors or omissions in the content of this project.

The information contained in this project is provided on an "as is" basis with no guarantees of completeness, accuracy, usefulness or timeliness.

<https://github.com/ferrithemaker/biotools/issues>

9.1.1.6 Modules and Nodes

The biotools modules available are:

- Module name on npm or Node-RED module repository:

node-red-contrib-biotools

Main functionalities: *Data input / output and parsing.*

- Module name on npm or Node-RED module repository:
node-red-contrib-biotools-sequence
 Main functionalities: *Sequence management.*
- Module name on npm or Node-RED module repository:
node-red-contrib-biotools-analysis
 Main functionalities: *Genomic sequence analysis.*

9.1.1.7 FASTAs NCBI supported types

Type	Format
local (i.e. no database reference)	lcl integer lcl string
GenInfo backbone seqid	bbs integer
GenInfo backbone moltype	bbm integer
GenInfo import ID	gim integer
GenBank	gb accession locus
EMBL	emb accession locus
PIR	pir accession name
SWISS-PROT	sp accession name
patent	pat country patent sequence-number
pre-grant patent	pgp country application-number sequence-number
RefSeq	ref accession name
general database reference	gnl database integer gnl database string
GenInfo integrated database	gi integer
DDBJ	dbj accession locus
PRF	prf accession name
PDB	pdb entry chain
third-party GenBank	tpg accession name
third-party EMBL	tpe accession name
third-party DDBJ	tpd accession name
TrEMBL	tr accession name

9.1.1.8 Nucleic acid notation by IUPAC/IUB

Nucleic Acid Code	Meaning	Mnemonic	Degenerate
A	A	Adenine	No

Nucleic Acid Code	Meaning	Mnemonic	Degenerate
C	C	Cytosine	No
G	G	Guanine	No
T	T	Thymine	No
U	U	Uracil	No
R	A or G (I)	puRine	Yes
Y	C, T or U	pYrimidines	Yes
K	G, T or U	bases which are Ketones	Yes
M	A or C	bases with aMino groups	Yes
S	C or G	Strong interaction	Yes
W	A, T or U	Weak interaction	Yes
B	not A (i.e. C, G, T or U)	B comes after A	Yes
D	not C (i.e. A, G, T or U)	D comes after C	Yes
H	not G (i.e., A, C, T or U)	H comes after G	Yes
V	neither T nor U (i.e. A, C or G)	V comes after U	Yes
N	A C G T U	Nucleic acid	Yes
-	gap of indeterminate length		

9.1.1.9 Amino acid notation by IUPAC/IUB

Amino Acid Code	Meaning
A	Alanine
B	Aspartic acid (D) or Asparagine (N)
C	Cysteine
D	Aspartic acid
E	Glutamic acid
F	Phenylalanine
G	Glycine
H	Histidine
I	Isoleucine
J	Leucine (L) or Isoleucine (I)
K	Lysine
L	Leucine
M	Methionine/Start codon
N	Asparagine
O	Pyrrolysine (rare)
P	Proline

Amino Acid Code	Meaning
Q	Glutamine
R	Arginine
S	Serine
T	Threonine
U	Selenocysteine (rare)
V	Valine
W	Tryptophan
Y	Tyrosine
Z	Glutamic acid (E) or Glutamine (Q)
X	any
	translation stop
-	gap of indeterminate length

9.1.1.10 Recommended Node-RED complementary nodes

- storage: `read file` (core)

Reads the contents of a file as either a string or binary buffer.

- storage: `write file` (core)

Writes *msg.payload* to a file, either adding to the end or replacing the existing content. Alternatively, it can delete the file.

- sequence: `split` (core)

Splits a message into a sequence of messages.

Check the **help node** option from Node-RED user interface for more information about the Node-RED nodes.

9.1.2 Biotools: Preprocessing Raw data node

9.1.2.1 Description

This node reads and parses a unformatted nucleotide or amino acid sequence.

9.1.2.2 Input

- *msg.payload* : An unformatted string sequence.

9.1.2.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Input* : Defines the type of data that would be parsed. [*nucleotide / amino acid*]
- *Allow degenerate base symbols* : Defines if the degenerate nucleotide base symbols defined by IUPAC/IUB will be parsed. [*yes / no*]
- *Id* : Defines a custom *Id* for the node (this parameter is not related with *Name* parameter). This *Id* is required if this node is used as an input for a node with two inputs.

9.1.2.4 Outputs

- *msg.payload.id* : the id defined at the *Id* parameter.
- *msg.payload.sequence* : String of parsed nucleotide sequence.
- *msg.payload.protein* : String of parsed amino acid sequence.

9.1.3 Biotools: Processing FASTA file node

9.1.3.1 Description

This node reads and parses a FASTA format file.

9.1.3.2 Input

- *msg.payload* : string from text file with compatible FASTA format: (<https://blast.ncbi.nlm.nih.gov/Blast.cgi>)

The nucleotide and amino acid symbols are defined by IUB/IUPAC.

9.1.3.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Input* : Type of data from FASTA file. [*nucleotide / protein*]
- *Seq. number* : In case of multiFASTA file, the number of the FASTA data structure to be parsed. (*default: 1*)
- *Id* : Defines a custom *Id* for the node (this parameter is not related with *Name* parameter). This *Id* is required if this node is used as an input for a node with two inputs.

9.1.3.4 Outputs

The output Node-RED `msg.payload` property includes an object with 4 properties:

- `msg.payload.id` : the id defined at the *Id* parameter.
- `msg.payload.sequence` : this string type property can include a nucleotide sequence (ACTGUNKSYMWRBDHV-) in uppercase.
- `msg.payload.protein` : this string type property can include an amino acid sequence (ADERNCFGQHILKMPSYTWVU*-) in uppercase.
- `msg.payload.information` : this property includes an structured object with data about the sequence information.
 - `msg.payload.information.db` : Original information about the database.
 - `msg.payload.information.id` : Original information about the id of the data.
 - `msg.payload.information.infodata` : Relevant information data depending on the db if any.
 - `msg.payload.information.extradata`: Any extra data from the information header of the FASTA file if any.

9.1.4 Biotools: Processing JASPAR file node

9.1.4.1 Description

This node reads and parses a nucleotide JASPAR PFM (Position Frequency Matrix) file. (<https://jaspar.genereg.net/docs/#raw-pfm-format>)

9.1.4.2 Input

- `msg.payload` : string from text file with compatible JASPAR PFM format: (<https://jaspar.genereg.net/docs/#raw-pfm-format>)

9.1.4.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Id* : Defines a custom *Id* for the node (this parameter is not related with *Name* parameter). This *Id* is required if this node is used as an input for a node with two inputs.

9.1.4.4 Outputs

The output Node-RED `msg.payload` property includes an object with 3 properties:

- `msg.payload.id` : the id defined at the *Id* parameter.
- `msg.payload.information` : the information header from the JASPAR PFM file.
- `msg.payload.PFMArray` : the PFM array from the JASPAR PFM file.

9.1.5 Biotools: FASTA file output

9.1.5.1 Description

This node parses the `msg.payload` object to create a FASTA format structure, ready to be written by the `write file` Node-RED node.

9.1.5.2 Input

The input Node-RED `msg.payload` property must include an object with 3 properties:

- `msg.payload.sequence` : this string type property can include a nucleotide sequence (ACTGUNKSYMWRBDHV-) in uppercase.
- `msg.payload.protein` : this string type property can include an amino acid sequence (ADERNCFGQHILKMPSYTWVU*-) in uppercase.
- `msg.payload.information` : this property includes an structured object with data about the sequence information.
 - `msg.payload.information.db` : Original information about the database.
 - `msg.payload.information.id` : Original information about the id of the data.
 - `msg.payload.information.infodata` : Relevant information data depending on the db if any.
 - `msg.payload.information.extradata`: Any extra data from the information header of the FASTA file if any.

The nucleotide and amino acid symbols are defined by IUB/IUPAC.

9.1.5.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Input* : Type of data to be parsed to FASTA file. [*nucleotide* / *protein*]

9.1.5.4 Outputs

- *msg.payload* : This object contains the postprocessed FASTA formatted data that will be injected to `write file` native Node-RED module.

9.1.6 Biotools: Processing GenBank file node

9.1.6.1 Description

This node reads and parses a GenBank format file.

9.1.6.2 Input

- *msg.payload* : string from text file with compatible GenBank format:
<http://quma.cdb.riken.jp/help/gbHelp.html>

9.1.6.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Input* : Type of data from read from GenBank file. [*origin* / *features*]
- *Seq. number* : In case of feature input, the number of the CDS data structure to be parsed. (*default: 1*)
- *Id* : Defines a custom *Id* for the node (this parameter is not related with *Name* parameter). This *Id* is required if this node is used as an input for a node with two inputs.

9.1.6.4 Outputs

The output Node-RED *msg.payload* property includes an object with 4 properties:

- *msg.payload.id* : the id defined at the *Id* parameter.
- *msg.payload.sequence* : this string type property can include a nucleotide sequence (ACTGUNKSYMWRBDHV-) in uppercase (if origin input has been selected).
- *msg.payload.protein* : this string type property can include an amino acid sequence (ADERNCFGQHILKMPSYTWVU*-) in uppercase (if features input has been selected).
- *msg.payload.information* : this property includes an structured object with data about the sequence information.

- if *origin* input is selected : Information gets the LOCUS string from Genbank File.
- if *features* input is selected : Information gets the *protein_id* property from the specific feature requested.

9.1.7 Biotools-sequence: Reverse node

9.1.7.1 Description

This node returns a reverse sequence of the original nucleotide sequence.

9.1.7.2 Input

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.7.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.7.4 Outputs

- *msg.payload.sequence* : Reverse nucleotide sequence string.

9.1.8 Biotools-sequence: Complement node

9.1.8.1 Description

This node returns the complementary sequence of the original nucleotide sequence.

9.1.8.2 Input

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.8.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Output* : Defines the output type of the sequence string [*DNA* / *RNA*].

9.1.8.4 Outputs

- *msg.payload.sequence* : Complementary nucleotide sequence string (DNA or RNA).

9.1.9 Biotools-sequence: Transcription node

9.1.9.1 Description

This node returns the RNA transcription sequence of the original DNA sequence.

9.1.9.2 Input

- *msg.payload.sequence* : DNA original nucleotide string.

9.1.9.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.9.4 Outputs

- *msg.payload.sequence* : RNA transcribed nucleotide string.

9.1.10 Biotools-sequence: Translation node

9.1.10.1 Description

This node returns de amino acid translation of the original RNAm sequence.

The traslation procedure will silenty ignore degenerate base symobols to avoid a multiple amino acid sequences output.

9.1.10.2 Input

- *msg.payload.sequence* : Original RNA nucleotide sequence string.

9.1.10.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.10.4 Outputs

- *msg.payload.protein* : Translated amino acid sequence string.

9.1.11 Biotools-sequence: Is degenerate node

9.1.11.1 Description

This node checks if the original sequence of nucleotides contains degenerate base symbols.

9.1.11.2 Input

- *msg.payload.sequence* : Original nucleotide sequence.

9.1.11.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.11.4 Outputs

This node has two outputs:

- *First output* : if sequence contains degenerate base symbols, the original input *msg.payload.sequence* is redirected to the first output.
- *Second output* : if sequence **not** contains degenerate base symbols, the original input *msg.payload.sequence* is redirected to the second output.

9.1.12 Biotools-sequence: Get subsequence

9.1.12.1 Description

Returns a partial subsequence of the original input nucleotide sequence.

9.1.12.2 Input

- *msg.payload.sequence* : Original DNA / RNA nucleotide sequence string.

9.1.12.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *From* : The initial position of the subsequence. The first position of the sequence is not the 0 index position but 1 to comply with the GenBank file format.
- *To* : The final position of the subsequence. The final position of the sequence is also included on the output subsequence to comply with the gGenBank file format.

9.1.12.4 Outputs

- *msg.payload.sequence* : Parsed subsequence of DNA / RNA nucleotide string.

9.1.13 Biotools-sequence: Sequence type node

9.1.13.1 Description

This node returns the nucleotide sequence type.

9.1.13.2 Input

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.13.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Allow degenerate base symbols* : Defines if the degenerate nucleotide base symbols defined by IUPAC/IUB will be parsed. [*yes* / *no*]

9.1.13.4 Outputs

- *msg.payload* : The function returns the type of the nucleotide sequence [*RNA* / *DNA* / *wrong sequence* / *undefined*]

9.1.14 Biotools-analysis: Sequence count node

9.1.14.1 Description

This node counts the number of nucleotides of the sequence [*Adenine* / *Cytosine* / *Guanine* / *Thymine* / *Uracil* / *All*] (degenerated or not degenerated).

9.1.14.2 Input

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.14.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

- *Allow degenerate base symbols* : Defines if the degenerate nucleotide base symbols defined by IUPAC/IUB will be parsed. [*yes / no*]
- *Data type* : Defines de requested nucleotide information [*Adenine / Cytosine / Guanine / Thymine / Uracil / All*]

9.1.14.4 Outputs

- *msg.payload* : The function returns the number of the requested nucleotides of the sequence.

9.1.15 Biotools-analysis: Hamming distance

9.1.15.1 Description

This node calculates the hamming distance between two not degenerated nucleotide sequences of the same size.

9.1.15.2 Inputs

This node requires two different input nodes. Each input node requires a diferent *Id* parameter.

Each input node requires a data string of not degenerated nucleotide sequence at *msg.payload.sequence*

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.15.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.15.4 Outputs

- *msg.payload* : The hamming distance value.

9.1.16 Biotools-analysis: Transition / transversion rate node

9.1.16.1 Description

This node calculates the transition / transversion rate between two not degenerated nucleotide sequences.

9.1.16.2 Inputs

This node requires two different input nodes. Each input node requires a different *Id* parameter.

Each input node requires a data string of not degenerated nucleotide sequence at *msg.payload.sequence*

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.16.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.

9.1.16.4 Outputs

- *msg.payload.transitions* : The number of transitions.
- *msg.payload.transversions* : The number of transversions.
- *msg.payload.ratio* : The ratio between the number of transitions and the number of transversions.

9.1.17 Biotools-analysis: Alignment analysis node

9.1.17.1 Description

This node checks the alignment between two nucleotide sequences using the needleman-wunsch algorithm.

9.1.17.2 Inputs

This node requires two different input nodes. Each input node requires a different *Id* parameter.

Each input node requires a data string of not degenerated nucleotide sequence at *msg.payload.sequence*

- *msg.payload.sequence* : Original nucleotide sequence string.

9.1.17.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *Gap* : Defines the *gap* weight for alignment calculation.

- *Miss* : Defines the *miss* weight for alignment calculation.
- *Match* : Defines the *match* weight for alignment calculation.

9.1.17.4 Outputs

- *msg.payload* : The needleman-wunsch algorithm returns a two-dimensional array as [*headA*, *outA*], [*headB*, *outB*], [*aln*, *alignment*], [*score*, *z*]

9.1.18 Biotools-analysis: Motif PFM score node

9.1.18.1 Description

This node calculates the score of binding sites between a JASPAR PFM (Position Frequency Matrix) and a nucleotide sequence

9.1.18.2 Input

This node requires two different input nodes. Each input node requires a different Id parameter.

- *msg.payload.PFMArray* : The PFM array from the JASPAR PFM file:
(<https://jaspar.genereg.net/docs/#raw-pfm-format>)
- *msg.payload.sequence* : Nucleotide sequence string.

9.1.18.3 Parameters

- *Name* : Defines node custom name. Blank field means that the node has the default name.
- *MinScore* : Defines a minimum score threshold.

9.1.18.4 Outputs

The output Node-RED *msg.payload* property includes an object with 3 properties:

- *msg.payload.sequence* : the original nucleotide sequence.
- *msg.payload.PFMArray* : the original PFM array from the JASPAR PFM file.
- *msg.payload.score* : the motif score using the PWM (Position Weight Matrix) method.

9.1.19 Notes

- To get the latest version of the documentation check the online updated version at: <https://github.com/ferrithemaker/biotools/wiki>
- This documentation has been transformed from Github Markdown to $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ using the <https://alldocs.app/convert-markdown-to-latex> online service.

9.2 Software source code

For reasons of space optimization, the entire software source code is not included in the annexes. The latest version can be checked in the GitHub repository of this MTP: <https://github.com/ferrithemaker/biotools>

9.3 Tutorials

There are some video tutorials available to help install and using biotools Node-RED nodes at: https://youtube.com/playlist?list=PLmbQdBhC77OvUgysrm675Ru7wq93_c6k5

Chapter 10

Glossary

alignment tool (bioinformatics) A way of arranging the sequences of DNA, RNA, or protein to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences.

API Application Programming Interface.

apt Advanced Package Tool.

clustal (bioinformatics) A series of widely used computer programs used in bioinformatics for multiple sequence alignment.

complement (bioinformatics) Nucleic acid sequence of bases that can form a double-stranded structure by matching base pairs.

database A set of data belonging to the same context and systematically stored for later use.

data type A particular kind of data item, as defined by the values it can take, the programming language used, or the operations that can be performed on it.

dashboard A visual interface where the user can manage hardware and/or software components.

fasta (bioinformatics) Text-based format for representing either nucleotide sequences or amino acid (protein) sequence.

framework A layered structure indicating what kind of programs can or should be built and how they would interrelate.

FSF Free Software Foundation.

gantt chart A chart in which a series of horizontal lines shows the amount of work done or production completed in certain periods of time in relation to the amount planned for those periods.

genbank (bioinformatics) A comprehensive database that contains publicly available nucleotide sequences.

GFF (bioinformatics) A file format used for describing genes and other features of DNA, RNA and protein sequences.

git Software for tracking changes in any set of files.

GNU GNU's Not Unix.

GPL General Public License (GPL) is a free, copyleft license used primarily for software.

hamming distance (bioinformatics) The number of positions that two codewords of the same length differ.

integration tests A type of testing where software modules are integrated logically and tested as a group.

integrative tool A tool designed to integrate third-party services using different kinds of data processing and communication methods.

IT Information technologies.

MTP Master/Thesis Project.

noSQL No Structured Query Language / not only SQL.

NPM Node.js Package Manager.

module Each of a set of standardized parts or independent units that can be used to construct a more complex structure.

motif (bioinformatics) A sequence motif is a nucleotide or amino-acid sequence pattern.

open source software Software for which the original source code is made freely available and may be redistributed and modified.

package manager Software used to automate the process of installing, upgrading, configuring, and removing programs.

prediction software (bioinformatics) Software that do the process of identifying the regions of genomic DNA that encode genes.

proprietary software Software that legally remains the property of the organisation, group, or individual who created it.

raw data Data collected from an original source.

RESTful API Architectural style for an application program interface (API) that uses HTTP requests to access and use data.

RPM RedHat Package Manager.

runtime environment Environment in which a program or application is executed.

sequence (bioinformatics) A one-dimensional ordering of monomers, covalently linked within a biopolymer.

shell A computer program which exposes an operating system's services to a human user or other programs.

SQL Structured Query Language.

transcription (bioinformatics) A process of making an RNA strand from a DNA tem-

plate.

translation (bioinformatics) Convert a DNA sequence into a protein.

transition (bioinformatics) A point mutation in DNA that changes a purine nucleotide to another purine, or a pyrimidine nucleotide to another pyrimidine.

transversion (bioinformatics) A point mutation in DNA in which a single purine is changed for a pyrimidine, or vice versa.

uniprot (bioinformatics) A freely accessible database of protein sequence and functional information.

unit testing A software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

web service Technology that uses a set of protocols and standards to exchange data between applications.

YUM Yellowdog updater modified (package installer).

Chapter 11

Bibliography

- 1 OpenJS Foundation. (2022). *Node-RED*. <https://nodered.org/>
- 2 OpenJS Foundation. (2022). *Node.js*. <https://nodejs.org/es/>
- 3 OpenJS Foundation. (2022). *Library - Node-RED*. <https://flows.nodered.org/>
- 4 ECMA International (2021). *ECMA-262-Ecma International*. <https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>
- 5 Python Software Foundation (2022). *Welcome to Python.org*. <https://www.python.org/>
- 6 University of California Santa Cruz Genomics Institute (2022). *UCSC Genome Browser*. <https://genome.ucsc.edu/>
- 7 Stanford University (2022). *ENCODE*. <https://www.encodeproject.org/>
- 8 EMBL-EBI (2021). *Ensembl release 105*. <https://www.ensembl.org/index.html>
- 9 Elixir Norway — London Institute of Medical Science — University of Copenhagen — Centre for Molecular Medicine and Therapeutics — Center of Molecular Medicine Norway (2022). *JASPAR - A database of transcription factor binding profiles*. <https://jaspar.genereg.net/>
- 10 National Center for Biotechnology Information (2022). *Home - GEO - NCBI*. <https://www.ncbi.nlm.nih.gov/geo/>
- 11 Genome Bioinformatics Research Laboratory (2019). *geneid homepage*. <https://genome.crg.es/software/geneid/>

- 12 National Center for Biotechnology (2022) *BLAST: Basic Local Alignment Search Tool*. <https://blast.ncbi.nlm.nih.gov/Blast.cgi>
- 13 National Institutes of Health (-) *Introduction - MEME Suite*. <https://meme-suite.org/meme/>
- 14 German Federal Ministry of Education and Research (2022). *Galaxy*. <https://usegalaxy.eu/>
- 15 The R Foundation (2022). *R: The R Project for Statistical Computing*. <https://www.r-project.org/>
- 16 Bioconductor (2022). *Bioconductor - Home*. <https://www.bioconductor.org/>
- 17 Open Bioinformatics Foundation (2021). *Biopython*. <https://biopython.org/>
- 18 NSF International (2022). *NSF International*. <https://es.nsf.org/es>
- 19 Federal Ministry of Education and Research (2022). *Homepage - BMBF*. <https://www.bmbf.de/>
- 20 Macquarie University (2022). *Welcome to Bioplatforms Australia*. <https://bioplatforms.com/>
- 21 National Research Infrastructure for Australia (2022). *Home - ARDC*. <https://ardc.edu.au/>
- 22 GitHub, Inc. (2022). *GitHub: Where the world builds software · GitHub*. <https://github.com/>
- 23 The GNOME Project (2022). *Apps/Planner*. <https://wiki.gnome.org/Apps/Planner>
- 24 Free Software Foundation, Inc. (2021). *What is Free Software? - GNU Project - Free Software Foundation* <https://www.gnu.org/licenses/gpl-3.0.html.en>
- 25 Free Software Foundation, Inc. (2021). *The GNU General Public License v3.0 - GNU Project - Free Software Foundation* <https://www.gnu.org/philosophy/free-sw.en.html#four-freedoms>
- 26 Amir Ghahrai (2016). *Software Development Methodologies* <https://devqa.io/software-development-methodologies/>

- 27 Elixir Norway — London Institute of Medical Science — University of Copenhagen — Centre for Molecular Medicine and Therapeutics — Center of Molecular Medicine Norway (2022). *JASPAR - Documentation*. <https://jaspar.genereg.net/docs/#raw-pfm-format>
- 28 RIKEN Japan (2019). *GeneBank format*. <http://quma.cdb.riken.jp/help/gbHelp.html>
- 29 EMBL-EBI (2021). *Ensembl release 105*. <https://www.ensembl.org/index.html>
- 30 Massachusetts Institute of Technology (2021). *2.5: The Needleman-Wunsch Algorithm*. [https://bio.libretexts.org/Bookshelves/Computational_Biology/Book%3A_Computational_Biology_-_Genomes_Networks_and_Evolution_\(Kellis_et_al.\)/02%3A_Sequence_Alignment_and_Dynamic_Programming/2.05%3A_The_Needleman-Wunsch_Algorithm](https://bio.libretexts.org/Bookshelves/Computational_Biology/Book%3A_Computational_Biology_-_Genomes_Networks_and_Evolution_(Kellis_et_al.)/02%3A_Sequence_Alignment_and_Dynamic_Programming/2.05%3A_The_Needleman-Wunsch_Algorithm)
- 31 EMBL-EBI (2022). *International Nucleotide Sequence Database Collaboration — INSDC*. <https://www.insdc.org/>
- 32 DDBJ Center (2022). *DDBJ*. <https://www.ddbj.nig.ac.jp/index-e.html>
- 33 EMBL (2022). *ENA Browser*. <https://www.ebi.ac.uk/ena/browser/home>
- 34 National Library of Medicine (2021). *GenBank Overview*. <https://www.ncbi.nlm.nih.gov/genbank/>