Identification and analysis of copper resistance and homeostasis genes in a new copper-tolerant *Alteromonas macleodii* bacterial strain

Ane Iturbide Martinez de Albeniz
Máster en Bioinformática y Bioestadística
Genómica Comparativa

Ivan Erill Sagales
Carles Ventura Royo

5 de Junio de 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Identification and analysis of copper resistance and homeostasis genes in a new copper-tolerant* Alteromonas macleodii *bacterial strain*
Nombre del autor:	*Ane Iturbide Martinez de Albeniz*
Nombre del consultor/a:	*Ivan Erill Sagales*
Nombre del PRA:	*Carles Ventura Royo*
Fecha de entrega (mm/aaaa):	06/2019
Titulación::	*Máster en Bioinformática y Bioestadística*
Área del Trabajo Final:	*Genómica Comparativa*
Idioma del trabajo:	*Inglés*
Palabras clave	*Copper-tolerance, marine bacteria, comparative genomics*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

Finalidad: Identificar el mecanismo responsable de la resistencia al cobre de las cepas bacterianas *Alteromonas macleodii* resistentes mediante el análisis de la presencia de genes de resistencia al cobre en su genoma. Estudiar la regulación transcripcional de genes de resistencia al cobre en *Alteromonas* macleodii en particular y en el orden Alteromonadales en general mediante la búsqueda de motivos de secuencias de ADN conservadas en sus promotores.

Contexto: El uso del cobre como agente antimicrobiano está presente en un gran número de sectores, como el de la salud y la agricultura. En respuesta a concentraciones tóxicas de cobre en su ambiente, muchas bacterias han desarrollado sistemas de tolerancia al cobre. El surgimiento de estas bacterias resistentes tiene un impacto en la salud y la economía de nuestra sociedad. La caracterización de estos sistemas y su conservación en diferentes especies bacterianas profundiza nuestro conocimiento en la resistencia bacteriana al cobre y puede ayudar a desarrollar en un futuro nuevas estrategias antimicrobianas basadas en el cobre.

Metodología: Se han utilizado métodos de genómica comparativa. Estos incluyen el desarrollo de scripts de python para automatizar la búsqueda de proteínas homólogas y la obtención de secuencias promotoras de genes. Las herramientas BLAST y MEME se han utilizado para encontrar las proteínas homologas y la presencia de motivos de secuencias de ADN conservadas respectivamente. R ha sido utilizado para realizar las representaciones gráficas y el análisis de los datos.

Resultados: Una lista de posibles proteínas ortólogas ha sido generada y la representación de genes entre diferentes sistemas de resistencia al cobre y bacterias marinas ha sido analizada. La búsqueda del motivo de secuencia de ADN en los genes de interés ha encontrado un motivo conservado relacionado con el factor de transcripción CusR. La conservación de este motivo ha sido estudiado en especies de la orden de Alteromonadales.

Conclusiones: El proyecto presentado muestra una primera aproximación para automatizar la búsqueda de proteínas ortólogas. También se ha analizado la primera lista de candidatos obtenida resultando en observaciones sobre la representación de los sistemas de resistencia a cobre en diferentes cepas bacterianas. Las dos cepas resistentes de *Alteromonas macleodii* analizadas muestran un número mayor de genes relacionados con resistencia al cobre en su genoma. La búsqueda del motivo de secuencia de ADN conservado a dado lugar a nuevas perspectivas en los mecanismos de regulación transcripcional en la especie *Alteromonas macleodii*, el cual ha mostrado ser diferente de otras especies que también pertenecen a las *Alteromonas*.

Abstract (in English, 250 words or less):

Aim: Identify the mechanism behind copper-resistance of *Alteromonas macleodii* resistant bacterial strains by analysing copper resistance genes present in its genome. Study the transcriptional regulation of copper-resistance genes in *Alteromonas macleodii* in particular and Alteromonadales order in general by looking for conserved DNA sequence motifs in its promoters.

Context: The usage of copper as antimicrobial agent is present in a wide range of sectors, such as healthcare and agriculture. In response to toxic copper concentrations in their environment, several bacteria have developed copper-tolerance systems. The emergence of these resistant bacteria has a health and economical impact in our society. The characterization of these systems and their conservation in different bacterial species deepens our knowledge on bacterial copper-resistance and could help develop in the future new copper-based antimicrobial strategies

Methodology: Comparative genomics approaches have been applied. These include the development of python scripts to automatize protein homology search and collection of gene promoter sequences. BLAST and MEME tools have been used to find homologous proteints and presence of conserved DNA sequence motifs respectively. R has been used for graphic representation and analysis of the data.

Results: a list of putative protein orthologs has been generated and the representation of these genes among different copper-resistance systems and marine bacteria has been analyzed. The DNA sequence motif search on genes of interest has found a conserved motif related to CusR transcription factor. The conservation of this motif has been studied in Alteromonadales order species.

Conclusions: the presented project shows a first approach to automatize the search of ortholog proteins and has analysed first obtained list of candidates generating interesting observations regarding copper-resistance systems representation in different bacterial strains. Both resistant strains of *Alteromonas macleodii* analyzed show higher number of copper-resistance related genes in their genome. The search for a conserved DNA sequence motif has given new insights into the transcriptional regulation mechanisms in *Alteromonas macleodii* species, which have been shown to be different from its *Alteromonas* counterparts.

Index

Figure list

Table list

1. Introduction

1.1. Context and project grounds

Copper is an essential metal ion involved in aerobic metabolism as donator or acceptor of electrons in redox-active enzymes[1]. Although being essential for mammalian metabolism, it is also highly toxic for prokaryotes and it has been used as a powerful bactericidal by humans for a long time in history[2].

The role for copper as antimicrobial agent has expanded to different sectors in our society. In health care for instance, copper surfaces are being considered as a way to prevent Healthcare-Associated Infections (HAI), usually caused by contact with contaminated healthcare equipment and facilities[3,4]. Copper antimicrobial effect also has applications in agriculture, where copper-based antimicrobial compounds have been developed, mainly during the twentieth century, for crop protection[5]. It is also widely used to prevent fouling on vessels caused by microorganisms. Using paints containing copper to coat the underwater part of the vessel, the released copper acts as antimicrobial preventing the attachment and growth of marine bacteria[6]. This late approach can actually have a strong economical impact on this sector[7].

Importantly, even though copper has been used for centuries now as an antimicrobial agent, the research on its mechanism of action is still ongoing today, as well as the development of new technologies to apply it[8]. However, in response to toxic copper concentrations in their environment, several bacteria developed copper-tolerance systems.

Copper-resistance systems are based on three main strategies: copper efflux, sequestration and oxidation[9]. The Cue system participates in the copper efflux strategy. When Cu^+ is sensed in the cytosol by CueR protein, it activates the transcription of *copA*. This gene encodes for a copper exporting P_{1B}-type ATPase that exports Cu^+ from the cytosol to the periplasm. The Cus system represents an independent copper efflux system, responsible for exporting Cu^+ out of the periplasm. Other systems such as Pco and Cop, which are contained in plasmids, also contain genes that encode for copper pumps.

Cus and Pco systems also count with proteins involved in copper sequestration. These include CusF, which binds copper and delivers it to Cus exporters; and PcoE, which functions in the periplasm as a soluble copper binder. Regarding copper oxidation, this strategy is based on the fact that Cu^+ is more toxic than Cu^{2+} in anoxic conditions. Cue system counts with a copper oxidase, CueO. PcoA from Pco system is also suspected to have this enzymatical activity.

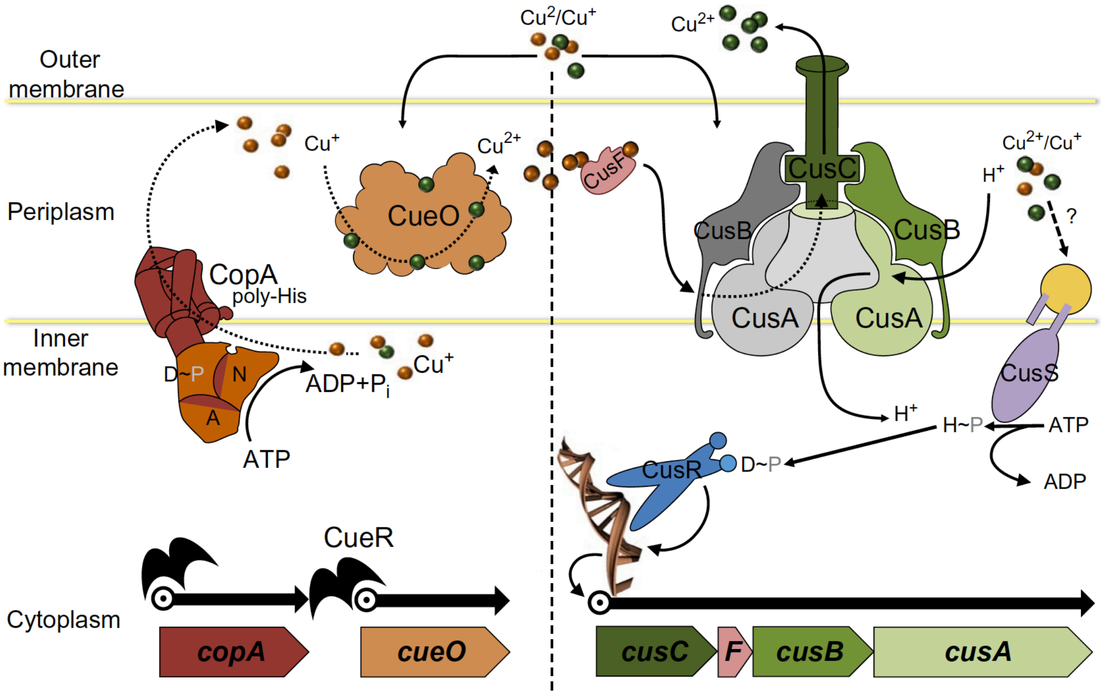


Figure 1: Representation of Cue and Cus systems (adapted from Pal et al., 2017)[10]

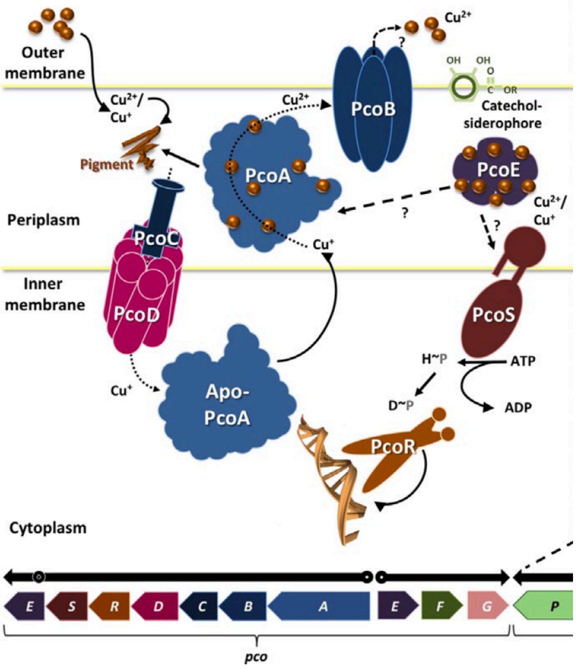


Figure 2: Representation of Pco system (adapted from Pal et al., 2017)[10]

Recently, a highly copper-tolerant strain of *Alteromonas macleodii* has been isolated from copper coupons in Key West (Florida, USA) by Kathleen Cusick. This newly discovered strain was named *Alteromonas macleodii CUKW*. This species of bacteria is a member of proteobacteria genus *Alteromonas* and, as a marine proteobacterium, it can be found in surface waters around the world[11].

Since this strain showed possible copper-tolerance capacity, it was cultured for 6 months in high copper concentration conditions (3mM). The resulting strain was also isolated and named *Alteromonas macleodii KCCO2*.

In order to better understand how this strain acquired high copper-tolerance, its genome has been sequenced before and after the culturing in high copper concentration media. In this project, computational tools for comparative genomics have been used in order to identify the possible mechanisms behind this feature by analysing copper resistance and homeostasis genes present in its genome.

Moreover, to get more insight on how these systems are regulated, a study on the transcriptional regulation of copper resistance genes has also been performed in Alteromonadales order (which contains *Alteromonas macleodii* species) by exploring putative sequence motifs present in its promoters.

The characterization of these systems and their conservation in different bacterial species deepens our knowledge on bacterial copper-resistance. Taking into account the wide usage of copper as antimicrobial in several fields and the linkage between copper resistance and virulence[9], the research on this area might be critical to better understand copper tolerance systems and could help develop in the future new copper-based antimicrobial strategies.

1.2. Objectives

The objectives pursued in this project are the following.

General objectives (GO)

1. Identify genes coding for copper tolerance/resistance in *Alteromonas macleodii CUKW* and *KCC02*, in *Alteromonas* at large, and several other marine bacteria.

2. Identify possible common regulatory elements upstream of identified copper tolerance genes.

Specific objectives (SO)

1. Identification of copper resistance genes.

    1.1 Development of automatized tools for protein homology search.
    1.2 Identify copper resistance related homologous proteins in target species using *Escherichia coli* and *Pseudomonas syringae* as models.
    1.3 Analyse the presence of different resistance systems in target species and look for possible representation enrichment in resistant strains.

2. Identification of transcriptional regulators of copper resistance systems

2.1 Development of automatized tools for specific genomic sequence acquisition.
2.2 Homolog motif discovery among identified proteins promoter regions.
2.3 Study conservation of these motifs among target species.
2.4 Identify putative motifs binding transcription factors.

1.3. Approach and methods

In order to achieve abovementioned goals, a combination of a bioinformatic and experimental approach are required. This work is part of a collaboration project between Erill and Cusick labs and each of them will take care over one of this aspects. Erill lab's role in the project is to employ computational tools on comparative genomics in order to give context and have a general picture of the mechanisms involved in the resistance of these bacterial strains. As well as to indicate putative regulation mechanisms involved. Bioinformatics allows a much faster way of gathering information around this biological question, saving time and money on experimental research. These results will hopefully work as starting point for several functional experiments in the Cusick lab, where a deeper molecular characterization can be performed on the involved identified resistance mechanisms.

1.4. Work planning

Hereunder an overview of the tasks planed and the timelines to do so will be detailed. Moreover, milestones and corresponding PECs will be specified.

Tasks

To fulfil the project objectives, the following tasks have been defined.

Phase I - Identification of copper resistance genes across species (GO 1) - 5 weeks

- Generate computational tools for automated protein homology search (SO 1.1) - 2 weeks
- Literature search for copper resistance genes (SO 1.2) - 1 week
- Systematic search of homologous proteins in target genomes (SO 1.2) - 1 week
- Data visualisation and analysis (SO 1.3) - 1 week
- Statistical analysis (SO 1.3) - 3 days
- Results discussion, consulting literature and writing report - Throughout the 5 weeks

Phase II - Identification of common regulatory elements (GO 2) - 3.5 weeks

- Generate computational tools to obtain genomic sequences upstream of genes (SO 2.1) - 1 week
- Obtain genomic data from identified homologues (SO 2.2) - 2 days

- DNA sequence motif discovery (SO 2.2 and SO 2.4) - 4 days
- Study conservation of motifs across-species with comparative genomics (SO 2.3) - 1 week
- System network regulation conservation data visualisation and analysis (SO 2.3) - 1 week
- Statistical analysis (if needed) (SO 2.3) - 3 days
- Results discussion, consulting literature and writing report - Throughout the 3.5 weeks

Calendar

A calendar was specified at the beginning of the project to implement the tasks, as well as taking into account the time needed for thesis planning, writing, and defence preparation. This planning can be seen represented in a Gantt chart in Figure 3.

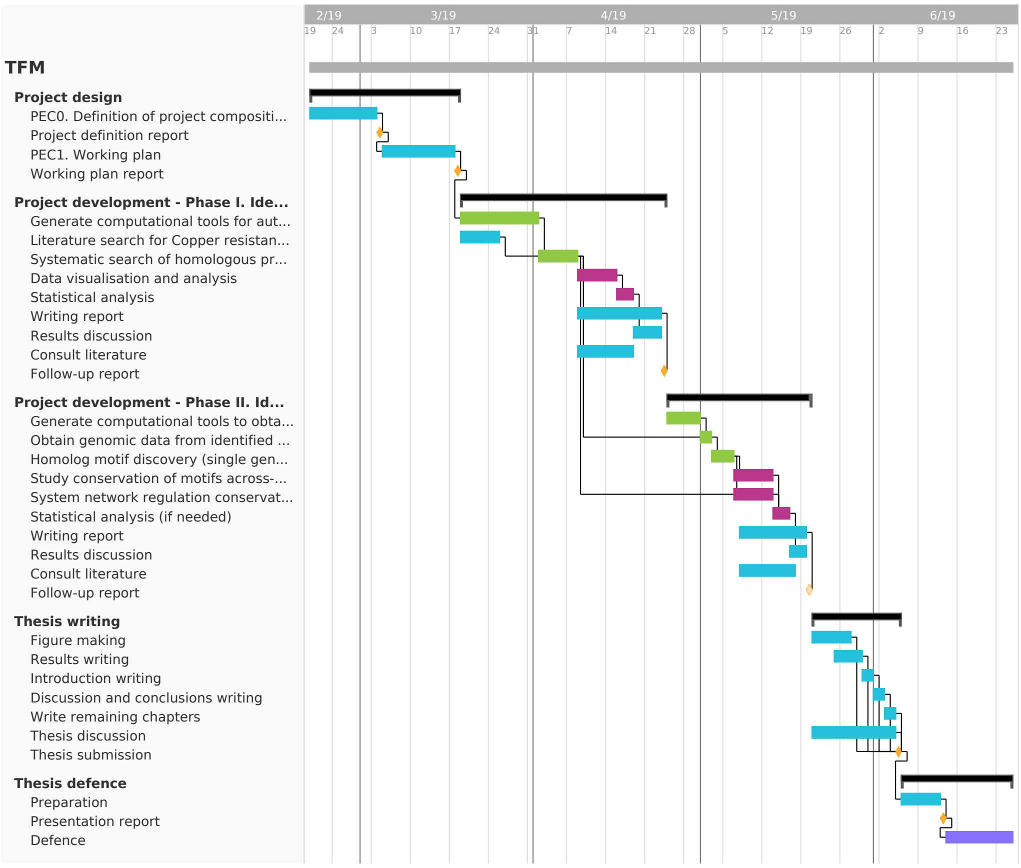


Figure 3: Project Gantt Chart

As we can see in here, the first month was dedicated to project discussion and planning with the supervisor, together with report writing for overview of the project (PEC0) and detailed approach planned to carry it out (PEC1). From mid of March on, hands on started for the specific defined tasks. These tasks show connection lines showing dependency and are colour coded based on their nature. Green, red and blue colours refer to script writing and execution, data analysis and reading/discussion/writing tasks respectively. Lastly, yellow diamonds represent milestones, which will be discussed later in this section.

First phase of the project is the one for which more time was assigned, as it was expected to require accustom to the computational tools as well as first script templates writing. The timings for each of the tasks carried out during this phase are also specified. Second phase on the other hand was expected to require less time, since there is already a familiarization with the tools and previously written scripts can be used as templates for the new ones. The specific duration of each of the tasks has also been indicated.

After project development, time for thesis writing and defence preparation was also planned. This includes tasks regarding figure making and final report writing and discussion among others.

Milestones

Taking into account project development phases and task dependency, milestones were defined. These represent key progress points in the project necessary for its success and are represented in the Gantt chart (Figure 3) as yellow diamonds. Detailed information about the milestones, including connection to each of the PECs, is depicted in Table 1.

Milestone	PEC	Deadline
Project definition report	PEC0	04/03/2019
Working plan report	PEC1	18/03/2019
Follow-up report I	PEC2	24/04/2019
Follow-up report II	PEC3	20/05/2019
Thesis submission	PEC4	05/06/2019
Thesis defence preparation	PEC5a	13/06/2019
Thesis defence	PEC5b	26/06/2019

Table 1: Project milestones

1.5. Brief summary of obtained products

As a result of this work, two types of products were obtained. First, scripts were developed to use for comparative genomics. The first script, named `ortholog_search_genomic.py`, takes protein accession numbers as input and makes a blast on defined species. It returns the information of found hits in `json` and `csv` format files. The second script, named `biosample_id.py`, gets corresponding biosample ids when provided with accession numbers. The third script, named `prot2proms_ai.py`, takes protein accessions as input, makes a blast with them on indicated species, gets promoter sequences from the obtained hits and returns them in a fasta file.

Secondly, data has been generated using these tools. From the combination of running first and second script on our proteins of interest list, information of hits was stored in a csv file named `results_tblastn_genomic.csv`. After using local BLAST to add the information of the *Alteromonas macleodii CUKW* and *KCC02* strains hits, the previous csv was complemented with this data resulting in `results_tblastn_genomic_CUKW_KCC02.csv`.

After using the `prot2proms_ai.py`, a fasta file was generated containing promoter sequences named `output_226.fas`, `output_72275.fas` and `output_135622.fas` (depending on the taxid used for the search). These files were used for MEME search, which returned the results as reports in html files (`Alteromonas_pal.html`, `Alteromonas_no_pal.html`, `Alteromonadaceae_pal.html`, `Alteromonadaceae_no_pal.html`, `Alteromonadales_pal.html`, `Alteromonadales_no_pal.html`). Finally, after using the CGB[12] program with the sequence motif of interest, the returned output was a heatmap in svg format file (`heatmap_ai.svg`).

1.6. Brief description of memorandum chapters

Introduction

This section starts explaining the background of the project and its objectives. Continues specifying the working plan of the project, were the required tasks and the time assigned to perform them are defined. Moreover, milestones are detailed and the products generated from this project listed. This section helps to explain overall what are the grounds of the project, the aim of it and how that is going to be achieved.

Materials and Methods

In this section the methods used to fulfil the tasks are described. Including what tools where used to develop the scripts and carry out the analysis (e.g. programming language, packages), as well as the technical details of the parameters used for each of the searches. This section describes how the work was done from a technical point of view and can be used as guideline for reproducing the results and for other usages of the scripts.

Results

This chapter shows the obtained results from implemented approaches. These are visualized with different kinds of graphical representations. How these results were obtained is briefly described in order to make it clearer to follow. The observations made out of these figures are also commented. This section puts together everything that was achieved during the project and answers at least partially the biological questions raised in the introduction.

Discussion

The obtained results are discussed, together with commenting of possible future steps to complete the work. It helps evaluate the work done based on obtained results, identifies points that could be improved and suggests how to do so.

Conclusions

It summarizes the conclusions of the project as a whole. It evaluates how well the timing of the working plan was followed and the level of objectives fulfilment. It ends with future perspectives of the work. In general, this chapters wraps up the work performed in all its aspects.

Glossary

Lists the definitions of the most relevant terms and acronyms used throughout this report.

Bibliography

Enumerated list of used bibliographical references throughout this report.

Annex

Extended data too lengthy to be included in the main section. To be used for consulting if wanted for more specific details on the results and used code.

2. Materials and Methods

The mains tasks developed during this project are the following:

Phase I - Identification of copper resistance genes across species (GO 1)

- Generation of computational tools for automated protein homology search (SO 1.1)
- Data visualisation and analysis (SO 1.3)

Phase II - Identification of common regulatory elements (GO 2) - 3.5 weeks

- Generation of computational tools to obtain genomic sequences upstream of genes (SO 2.1)
- DNA sequence motif discovery (SO 2.2 and SO 2.4)
- Study conservation of motifs across-species with comparative genomics. System network regulation conservation data visualisation and analysis (SO 2.3)

The general computational tools to perform these tasks were based on Python and R. The Python programming language was used for script writing, allowing automatization of homology search and genomic sequence acquisition. These scripts were developed in Spyder[13], an open source cross-platform integrated development environment (IDE), designed for scientific programming. Besides basic libraries for working with files and python objects, the use of Biopython library was central to these tools[14,15]. This library comprises a set of computational biology tools.

Once the information was gathered, exploratory and statistical analysis were performed in R language. In this case, RStudio[16] was used as IDE. Graphs for visualizing the data analysis were generated using the ggplot2 package[17].

Hereafter, how these tasks were performed will be detailed.

2.1. Identification of copper resistance genes across species

Generation of computational tools for automated protein homology search

In order to find orthologous proteins in the different bacterial species and strains, a python script has been written. This script uses biopython tools, including NCBIWWW[18] module from Blast package[19] (from Bio[15]) in order to call the NCBI BLAST server, as well as NCBIXML[20] to parse the obtained data and SeqIO[21] and Entrez[22] packages to get and handle key data for the search (also from Bio package). For more details, complete code with comments has been attached in the annex. Briefly, protein of interest is searched in RefSeq genomic database using tblastn blast tool. tblastn takes protein sequence, translates it to

DNA and looks for homologue sequences in the indicated target genome and gives obtained hit information as output. In this case, accession number, description, E-value and protein coverage among others are saved as relevant hit information.

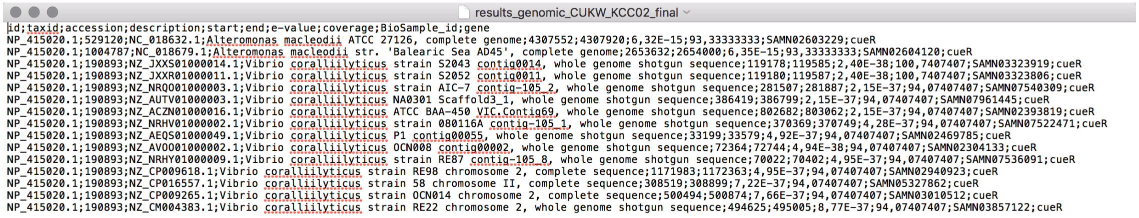
For the search on *Alteromonas macleodii CUKW* and *KCC02*. BLAST+[23,24] tool was locally installed and databases generated from the sequenced genome. Each of the proteins were searched for manually, an example command line is depicted below:

```
tblastn  –db  CUKW  –word_size  7  –query  NP_415102.1.fasta  –outfmt
‘7delim=,’    –max_target_seqs    20    e-value    10e-20    –out
NP_415102.1_CUKW.csv
```

All searches could have been done with BLAST+ in the local server. However, when starting the project, all target genomes were expected to be uploaded to NCBI (including *Alteromonas macleodii CUKW* and *KCC02*) and therefore the first script was thought to be only used. The decision to use the NCBIWWW module instead of the BLAST+ was made because it was thought to be faster, since the download of databases and protein fasta sequences would be avoided.

From this approach, a list of protein hits was obtained and its information stored as a dictionary in a JavaScript Object Notation (json) file and in a comma separated value (csv) format file. This file includes the protein accession, taxonomy id (taxid) of the targeted species for the search, accession of the genome were the hit was found, description associated to this accession, start and end positions of the hit in the genome, hit E-value and protein coverage.

Afterwards, another script was written to find the BioSample Id corresponding to each of the hits to have a clear idea of the species in which the hit was found and be able to make numbers with the number of hits per species in later analysis. This script uses the information on this file as input, and searches for the corresponding BioSample Id and stores this information in csv format. This information was then added to the previous csv file, together with the gene name associated to each of the protein accessions in order to make future analysis easier. The head of this file is shown in the following figure (Figure 4).



```
id;taxid;accession;description;start;end;e-value;coverage;BioSample_id;gene
NP_415020.1;529120;NC_018632.1;Alteromonas macleodii ATCC 27126, complete genome;4307552;4307920;6,32E-15;93,33333333;SAMN02603229;cueR
NP_415020.1;1004787;NC_018679.1;Alteromonas macleodii str. 'Balearic Sea AD45', complete genome;2653632;2654000;6,35E-15;93,33333333;SAMN02604120;cueR
NP_415020.1;190893;NZ_JXXS01000014.1;Vibrio coralliilyticus strain S2043 contig0014, whole genome shotgun sequence;119178;119585;2,40E-38;100,7407407;SAMN03323919;cueR
NP_415020.1;190893;NZ_JXXR01000011.1;Vibrio coralliilyticus strain S2052 contig0011, whole genome shotgun sequence;119180;119587;2,40E-38;100,7407407;SAMN03323806;cueR
NP_415020.1;190893;NZ_NRQO01000003.1;Vibrio coralliilyticus strain AIC-7 contig-105_2, whole genome shotgun sequence;281507;281887;2,15E-37;94,07407407;SAMN07540309;cueR
NP_415020.1;190893;NZ_AUTV01000003.1;Vibrio coralliilyticus NA0301 Scaffold3_1, whole genome shotgun sequence;386419;386799;2,15E-37;94,07407407;SAMN07961445;cueR
NP_415020.1;190893;NZ_ACZN01000016.1;Vibrio coralliilyticus ATCC BAA-450 VIC.Contig69, whole genome shotgun sequence;802682;803062;2,15E-37;94,07407407;SAMN02393819;cueR
NP_415020.1;190893;NZ_NRHV01000002.1;Vibrio coralliilyticus strain 080116A contig-105_1, whole genome shotgun sequence;370369;370749;4,28E-37;94,07407407;SAMN07522471;cueR
NP_415020.1;190893;NZ_AEQS01000049.1;Vibrio coralliilyticus P1 contig00055, whole genome shotgun sequence;33199;33579;4,92E-37;94,07407407;SAMN02469785;cueR
NP_415020.1;190893;NZ_AVOO01000002.1;Vibrio coralliilyticus OCN008 contig00002, whole genome shotgun sequence;72364;72744;4,94E-38;94,07407407;SAMN02304133;cueR
NP_415020.1;190893;NZ_NRHY01000009.1;Vibrio coralliilyticus strain RE87 contig-105_8, whole genome shotgun sequence;70022;70402;4,95E-37;94,07407407;SAMN07536091;cueR
NP_415020.1;190893;NZ_CP009618.1;Vibrio coralliilyticus strain RE98 chromosome 2, complete sequence;1171983;1172363;4,95E-37;94,07407407;SAMN02940923;cueR
NP_415020.1;190893;NZ_CP016557.1;Vibrio coralliilyticus strain 58 chromosome II, complete sequence;308519;308899;7,22E-37;94,07407407;SAMN05327862;cueR
NP_415020.1;190893;NZ_CP009265.1;Vibrio coralliilyticus strain OCN014 chromosome 2, complete sequence;500494;500874;7,66E-37;94,07407407;SAMN03010512;cueR
NP_415020.1;190893;NZ_CM004383.1;Vibrio coralliilyticus strain RE22 chromosome 2, whole genome shotgun sequence;494625;495005;8,77E-37;94,07407407;SAMN03857122;cueR
```

Figure 4: Preview of hits information stored

Data visualisation and analysis

For visualization of the data, exploratory and statistical analysis RStudio was used. This could have been also done with python programming language, mainly using Numpy[25], SciPy[26] and Matplotlib[27] libraries. However, R programming language and Rstudio were chosen instead based on the student's expertise and knowledge.

The csv file with hits information was loaded into RStudio. Number of hits per protein and per species were counted using `table` function, grouping each time for category of interest. Data visualization was made using ggplot2 library using `geom_boxplot`, `geom_line` and `geom_point` functions. For statistical analysis, dunn.test package[28] was used.

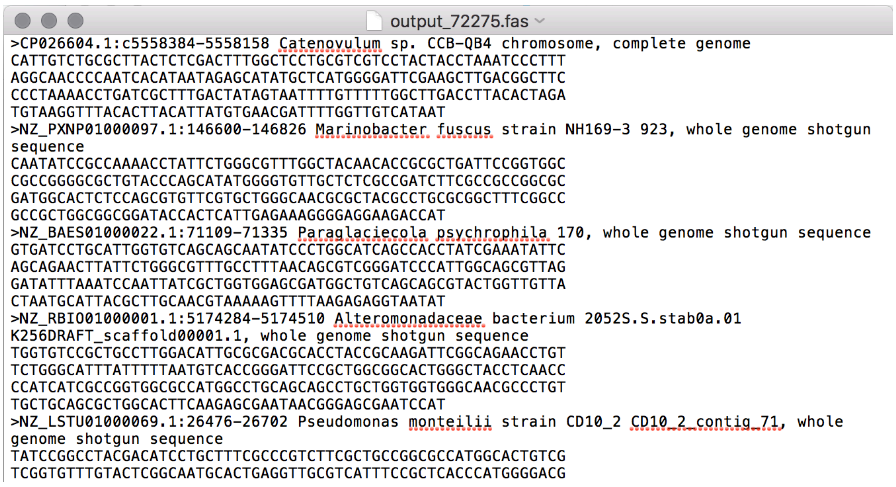
2.2. Identification of common regulatory elements

Generation of computational tools to obtain genomic sequences upstream of genes

For this specific task, the script was not written from scratch. There was already one written in the host laboratory. However, it was updated in some of the code lines, since some of the outputs generated by biopython Entrez tools had changed since this was written.

The main features of the used script are the following. It first does a BLAST search of putative ortholog proteins in selected taxonomy group, *Alteromonas* in this case. The accession numbers of the found hits are used then to obtain genomic data of the genes that encode for them (start and end point of the coding sequence, strand localization). Moreover, from the hits found per protein, most complete genome records are reached using a score system to prioritize best genome entries available for each of them. Once the genomic information is gathered, the promoter sequence is obtained, by choosing the positions 200bp upstream to and the coding sequence start site. Once the promoter sequences are obtained, a similarity filter is applied, and for sequences showing a higher than 80% similarity, the record is removed. This is an important step, since too similar sequences can lead to the finding of false sequence motifs, generated because of sequence similarity and not because of a conserved sequence due to functionality. Finally, filtered sequences are returned in an output fasta file.

After running this script using the list of proteins of interest shown in Table 6, an output fasta file was returned containing the list of sequences. A preview of the file can be seen in Figure 5.



```
>CP026604.1:c5558384-5558158 Catenovulum sp. CCB-QB4 chromosome, complete genome
CATTGTCTGCGCTTACTCTCGACTTTGGCTCCTGCGTCGTCCTACTACCTAAATCCCTTT
AGGCAACCCCAATCACATAATAGAGCATATGCTCATGGGGATTCGAAGCTTGACGGCTTC
CCCTAAAACCTGATCGCTTTGACTATAGTAATTTTGTTTTTGGCTTGACCTTACACTAGA
TGTAAGGTTTACACTTACATTATGTGAACGATTTTGGTTGTCATAAT
>NZ_PXNP01000097.1:146600-146826 Marinobacter fuscus strain NH169-3 923, whole genome shotgun
sequence
CAATATCCGCCAAAACCTATTCTGGGCGTTTGGCTACAACACCGCGCTGATTCCGGTGGC
CGCCGGGGCGCTGTACCCAGCATATGGGGTGTTGCTCTCGCCGATCTTCGCCGCCGGCGC
GATGGCACTCTCCAGCGTGTTCGTGCTGGGCAACGCGCTACGCCTGCGCGGCTTTCGGCC
GCCGCTGGCGGCGGATACCACTCATTGAGAAAGGGGAGGAAGACCAT
>NZ_BAES01000022.1:71109-71335 Paraglaciecola psychrophila 170, whole genome shotgun sequence
GTGATCCTGCATTGGTGTCAGCAGCAATATCCCTGGCATCAGCCACCTATCGAAATATTC
AGCAGAACTTATTCTGGGCGTTTGCCTTTAACAGCGTCGGGATCCCATTGGCAGCGTTAG
GATATTTAAATCCAATTATCGCTGGTGGAGCGATGGCTGTCAGCAGCGTACTGGTTGTTA
CTAATGCATTACGCTTGCAACGTAAAAAGTTTTAAGAGAGGTAATAT
>NZ_RBIO01000001.1:5174284-5174510 Alteromonadaceae bacterium 2052S.S.stab0a.01
K256DRAFT_scaffold00001.1, whole genome shotgun sequence
TGGTGTCCGCTGCCTTGGACATTGCGCGACGCACCTACCGCAAGATTCGGCAGAACCTGT
TCTGGGCATTTATTTTTAATGTCACCGGGATTCCGCTGGCGGCACTGGGCTACCTCAACC
CCATCATCGCCGGTGGCGCCATGGCCTGCAGCAGCCTGCTGGTGGTGGGCAACGCCCTGT
TGCTGCAGCGCTGGCACTTCAAGAGCGAATAACGGGAGCGAATCCAT
>NZ_LSTU01000069.1:26476-26702 Pseudomonas monteilii strain CD10_2 CD10_2_contig_71, whole
genome shotgun sequence
TATCCGGCCTACGACATCCTGCTTTCGCCCGTCTTCGCTGCCGGCGCCATGGCACTGTCG
TCGGTGTTTGTACTCGGCAATGCACTGAGGTTGCGTCATTTCCGCTCACCCATGGGGACG
```

Figure 5: Preview of the fasta fail returned by the script

This same search was repeated for the *Alteromonadaceae* family and the Alteromonadales order. Same type of output files as the one for *Alteromonas* were generated.

Homolog motif discovery

In order to find conserved motifs among the sequences, MEME[29,30] was used, which was locally installed in the server. MEME could have been used online to do the same search from the MEME Suite website[31]. However, taking into account the number of sequences for which the search had to be performed, it was decided to use the tool locally, as the analysis would run much faster.

The already mentioned fasta files were used as input for MEME tool and searched for both palindromic and non-palindromic and repeated sequences. The sequence length limit was set from 8 to 24 base pairs with a maximum of 20 hits to be returned. An example command line is depicted below:

```
meme –dna –o poi_pal –pal –mod anr –nmotifs 20 –minw 8 –maxw 24
output_72275.fas
```

The search was performed for the three fasta files generated in the previous approach. Generated report example can be found in the annex (Figure 20).

Study conservation of motifs across-species. System network regulation conservation data visualisation and analysis

For this step, a program already developed by the host laboratory was used, CGB[12]. This program takes an input sequence motif sequence and the protein to it. Then, it looks for the motif in the selected species genomes and using a scoring system, returns the probability that the same sequence motif is present in the promoter of that gene (taking into account variability in the sequence). Applying a threshold, this is discretized to present/not present. This data is then returned visually as a heatmap. On the upper part of the heatmap, the species are clustered depending on the sequence similarity of the motif binding protein. On the right, genes in which the motif has been found are listed. The colour code of the heatmap corresponds to the following: blue means no orthologs for the genes evaluated in that line were found for the species of that column; green means that the sequence motif is present in the promoter gene of that line and red that it is not. In this particular case, CusR binding motif was searched using this approach.

3. Results

3.1. Identification of copper-resistance genes

As described in the objectives section, the first goal was to identify genes coding for copper tolerance/resistance in *Alteromonas macleodii CUKW* and *KCC02*, in *Alteromonas* at large, and several other marine bacteria. On this regard, a thorough search was made through previously published work in order to identify known copper-resistance systems and components across different bacterial species.

Two bacterial species were used as models for the orthologs search. *Escherichia coli k-12 mg1655* for gram negative bacteria and *Pseudomonas syringae* for gram positive bacteria. For each of the found proteins, accession number in the corresponding model species was stored, together with information on gene name, protein name and others. This information was saved in a csv file for later use in the developed scripts. The data can be seen in Table 2.

Accession	Protein	Gene name	Location	System	Species
NP_415020.1	CueR	cueR	Chromosome	Cue	E. coli k-12 mg1655
NP_414665.1	CueO	cueO	Chromosome	Cue	E. coli k-12 mg1655
NP_415017.1	CopA	copA	Plasmid	Cue	E. coli k-12 mg1655
NP_415102.1	CusS	cusS	Chromosome	Cus	E. coli k-12 mg1655
NP_415103.1	CusR	cusR	Chromosome	Cus	E. coli k-12 mg1655
NP_415107.1	CusA	cusA	Chromosome	Cus	E. coli k-12 mg1655
NP_415106.1	CusB	cusB	Chromosome	Cus	E. coli k-12 mg1655
NP_415104.1	CusC	cusC	Chromosome	Cus	E. coli k-12 mg1655
NP_415105.1	CusF	cusF	Chromosome	Cus	E. coli k-12 mg1655
ANH09828.1	PcoA	pcoA	Plasmid	Pco	E. coli
ANH09778.1	PcoB	pcoB	Plasmid	Pco	E. coli
ANH09779.1	PcoC	pcoC	Plasmid	Pco	E. coli
ANH09780.1	PcoD	pcoD	Plasmid	Pco	E. coli
ANH09781.1	PcoR	pcoR	Plasmid	Pco	E. coli
ANH09782.1	PcoS	pcoS	Plasmid	Pco	E. coli
ANH09783.1	PcoE	pcoE	Plasmid	Pco	E. coli
AFX60851.1	PcoF	pcoF	Plasmid	Pco	E. coli
AZZ87773.1	PcoG	pcoG	Plasmid	Pco	E. coli
AQX42270.1	CopA	copA	Plasmid	Cop	P. syringae
AQX42189.1	CopB	copB	Plasmid	Cop	P. syringae
AQX42188.1	CopC	copC	Plasmid	Cop	P. syringae
AQX42268.1	CopD	copD	Plasmid	Cop	P. syringae
AQX42267.1	CopR	copR	Plasmid	Cop	P. syringae

AQX42266.1	CopS	copS	Plasmid	Cop	P. syringae
AQX41994.1	CopZ	copZ	Plasmid	Cop	P. syringae
AQX42189.1	CopB	copB	Plasmid	Cop	P. syringae
AAG10085.1	CopY	copY	Plasmid	Cop	S. mutans
AMP34391.1	CsoR	csoR	Plasmid	Cop	S. haemolyticus

Table 2: Copper resistance genes

First interest lies on searching for these genes in the genomes of recently sequenced *Alteromonas macleodii CUKW* and *KCC02* strains. However, to study the conservation of the genes in other *Alteromonas* and marine bacteria, a target group of species was selected representing these species (Table 3). Representing *Alteromonas macleodii*, there are *Alteromonas macleodii ATCC 27126* and *Alteromonas macleodii str. 'Balearic Sea AD45'* strains. They were chosen because these are the strains for which functional experiments for copper resistance are being carried out in the Cusick laboratory, allowing a more comprehensive analysis complementing bioinformatic with biological data. The rest of species were chosen as broad representatives of marine bacteria.

Species	**TaxID**
Alteromonas macleodii ATCC 27126	529120
Alteromonas macleodii str. 'Balearic Sea AD45'	1004787
Vibrio coralyticus	190893
Vibrio alginolyticus	663
Vibrio harveyi	669
Roseobacter denitrificans	2434
Marinovum algicola	42444
Pseudoalteromonas atlantica T6c	342610
Ruegeria TM1040	292414

Table 3: Target species

Next, the proteins of interest were searched against the genomes of named targets species using tblastn tool from BLAST[32,33]. This tool takes protein sequence as input, translates it to genomic sequence, and searches for similar sequences in genomes of interest. As a result, it returns the genomes were it was found, at which positions, percentage of similarity and E-value of the comparison. The E-value is a parameter that describes the number of hits one can "expect" to see by chance when searching a database of a particular size. Therefore, the lower the E-value, or the closer it is to zero, the more "significant" the match is considered.

Since a long list of proteins and targets had to be searched, a script was written in python, using biopython tools[18,19], to automatize this process. How this script works is explained in the materials and methods section. The complete code with detailed comments can be consulted in the annex. For the search in *Alteromonas macleodii CUKW* and *KCC02* strains, since they do not have a taxid, BLAST+[23,24] was installed locally and a database was created with their sequenced genomes.

As a result, 2003 putative protein orthologs were obtained for all species, using a threshold of an E-value < 10E-20. How this number is reduced while increasing the threshold can bee seen in Figure 6.

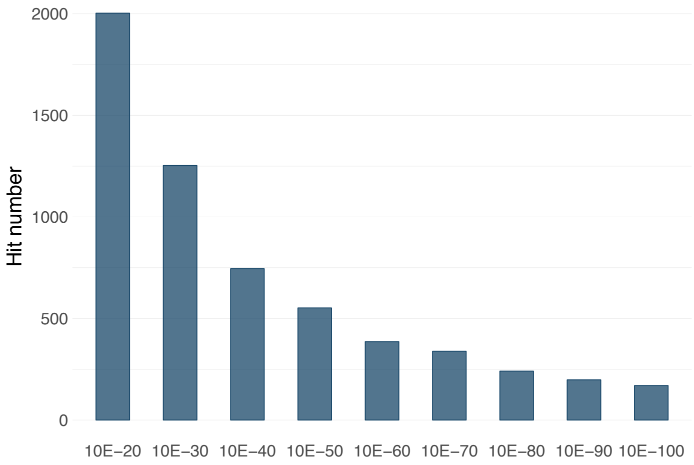


Figure 6: Total number of hits per E-value

This data was then analysed for representation of the differences between the species. In Figure 7 the number of hits per species has been plotted at different threshold E-values in order to visualize the main differences.

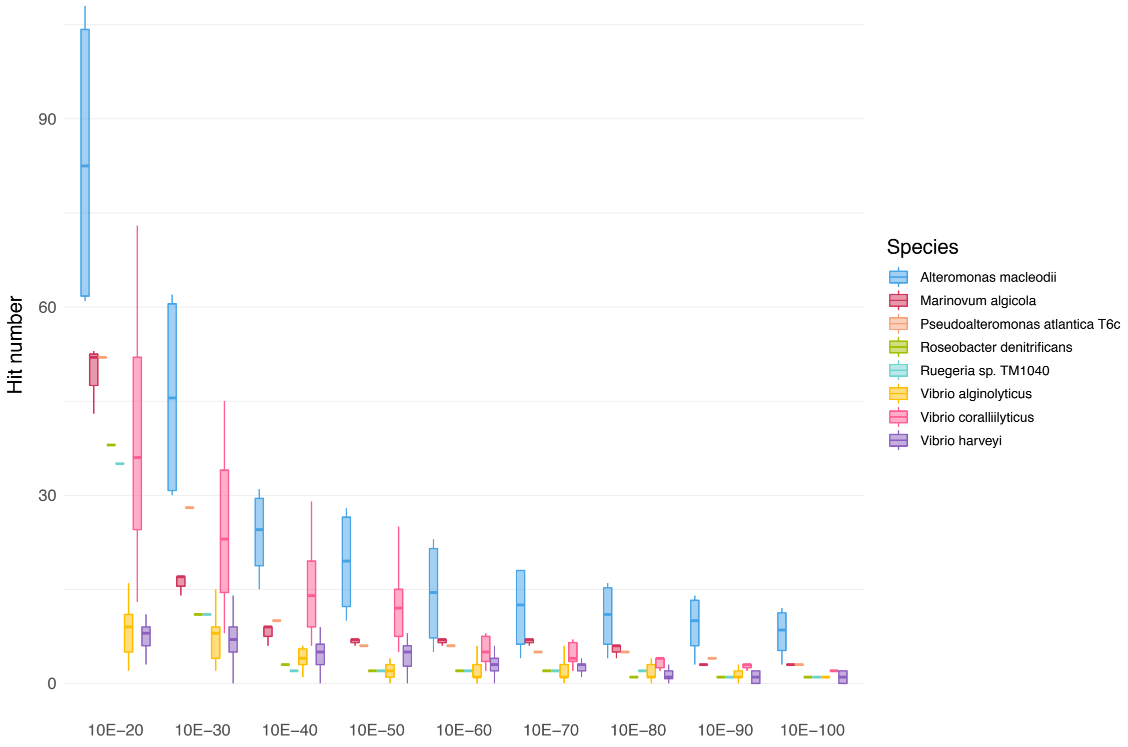


Figure 7: Found hits grouped by species

In order to know if the difference between species observed is significant, a statistical analysis was performed. Since these values do not follow a normal distribution (confirmed by Shapiro-Wilk normality test), Dunn's test was performed to obtain results among multiple pairwise comparisons. The null hypothesis for each pairwise comparison is that the probability of observing a randomly selected value from the first group that is larger than a randomly selected value from the second group equals one half; this null hypothesis corresponds to that of the Wilcoxon-Mann-Whitney rank-sum test. In this case, the adjusted p-value for multiple comparisons was obtained using Bonferroni adjustment. Since the first E-value used as threshold (10E-20) is quite permissive and a high number of found orthologs might be false positives, this point was removed from the data. Generated results can be seen in Table 4.

List of pairwise comparisons	Z statistic	Adjusted p-value
Alteromonas macleodii - Marinovum algicola	1.722192	1.0000
Alteromonas macleodii - Pseudoalteromonas atlantica T6c	1.144350	1.0000
Marinovum algicola - Pseudoalteromonas atlantica T6c	-0.031110	1.0000
Alteromonas macleodii - Roseobacter denitrificans	6.314494	0.0000 *
Marinovum algicola - Roseobacter denitrificans	4.295705	0.0002 *
Pseudoalteromonas atlantica T6c - Roseobacter denitrificans	3.068633	0.0301
Alteromonas macleodii - Ruegeria sp. TM1040	4.275194	0.0003 *
Marinovum algicola - Ruegeria sp. TM1040	3.000315	0.0378
Pseudoalteromonas atlantica T6c - Ruegeria sp. TM1040	2.475149	0.1865
Roseobacter denitrificans - Ruegeria sp. TM1040	-0.037207	1.0000
Alteromonas macleodii - Vibrio alginolyticus	8.901675	0.0000 *
Marinovum algicola - Vibrio alginolyticus	5.604938	0.0000 *
Pseudoalteromonas atlantica T6c - Vibrio alginolyticus	3.351398	0.0113 *
Roseobacter denitrificans - Vibrio alginolyticus	-0.249120	1.0000
Ruegeria sp. TM1040 - Vibrio alginolyticus	-0.104959	1.0000
Alteromonas macleodii - Vibrio coralliilyticus	3.175525	0.0209 *
Marinovum algicola - Vibrio coralliilyticus	0.745699	1.0000
Pseudoalteromonas atlantica T6c - Vibrio coralliilyticus	0.491428	1.0000
Roseobacter denitrificans - Vibrio coralliilyticus	-4.800032	0.0000 *
Ruegeria sp. TM1040 - Vibrio coralliilyticus	-2.897808	0.0526
Vibrio alginolyticus - Vibrio coralliilyticus	-9.500791	0.0000 *
Alteromonas macleodii - Vibrio harveyi	8.166610	0.0000 *
Marinovum algicola - Vibrio harveyi	4.956921	0.0000 *
Pseudoalteromonas atlantica T6c - Vibrio harveyi	2.966333	0.0422

Roseobacter denitrificans - Vibrio harveyi	-0.902371	1.0000
Ruegeria sp. TM1040 - Vibrio harveyi	-0.491104	1.0000
Vibrio alginolyticus - Vibrio harveyi	-1.737097	1.0000
Vibrio coralliilyticus - Vibrio harveyi	8.242823	0.0000 *

Table 4: Species pairwise comparisons

As we can see, in most of comparisons *Alteromonas macleodii* shows a significant difference. Therefore, we can say that this species is significantly enriched in copper resistance genes compared to others. Interestingly, for some of the species the boxplots show wider distributions, indicating a higher variability between strains for those species. This is clear for *Alteromonas macleodii*, specially when E-value decreases and stringency increases in considering the proteins as orthologs. This could be due to the variability added by the copper resistant strains *CUKW* and *KCC02*. In order to have a clearer comparison of these strains with the average of all the different species, the data has been plotted again with *Alteromonas macleodii* boxplot divided into the strains that it represented (Figure 8).

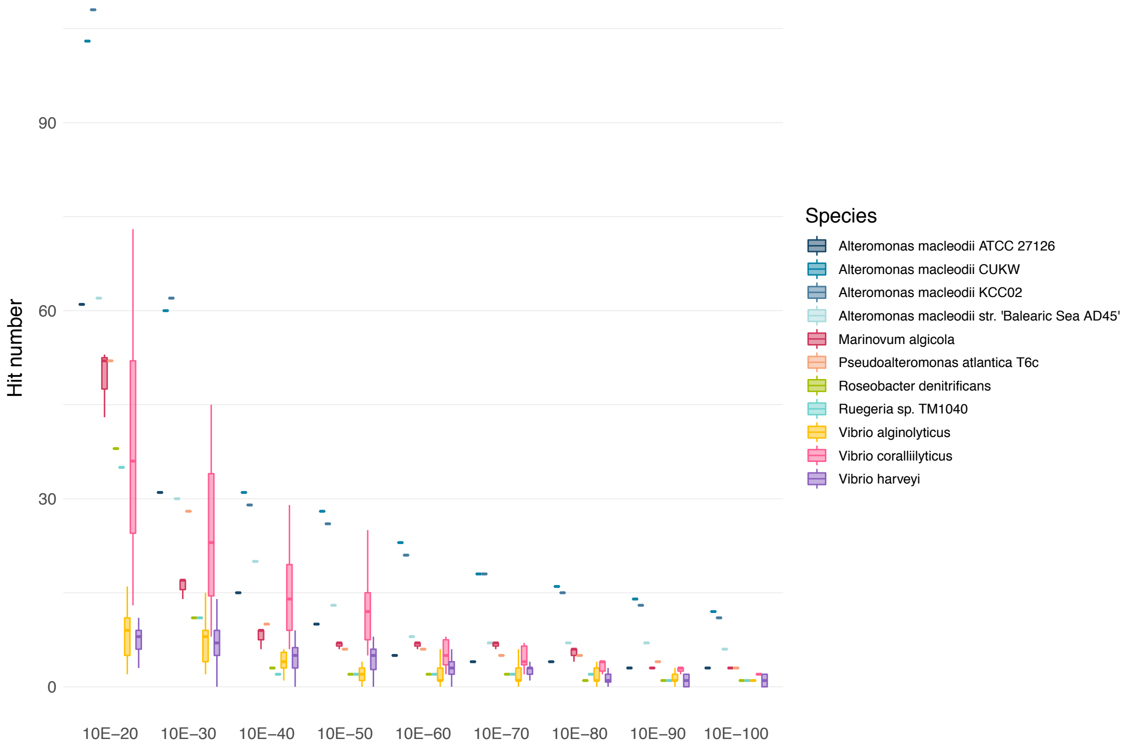


Figure 8: Found hits grouped by species compared to Alteromonas macleodii strains

The plot shows that this was actually the case, *Alteromonas macleodii CUKW* and *KCC02* show very different values from their *Alteromonas macleodii* counterparts.

To have more insight into this observation, another plot with only *Alteromonas macleodii* strains data is shown in Figure 9. Here *Alteromonas macleodii CUKW* and *Alteromonas macleodii KCC02* strains can be compared in detail within the *Alteromonas macleodii* species.

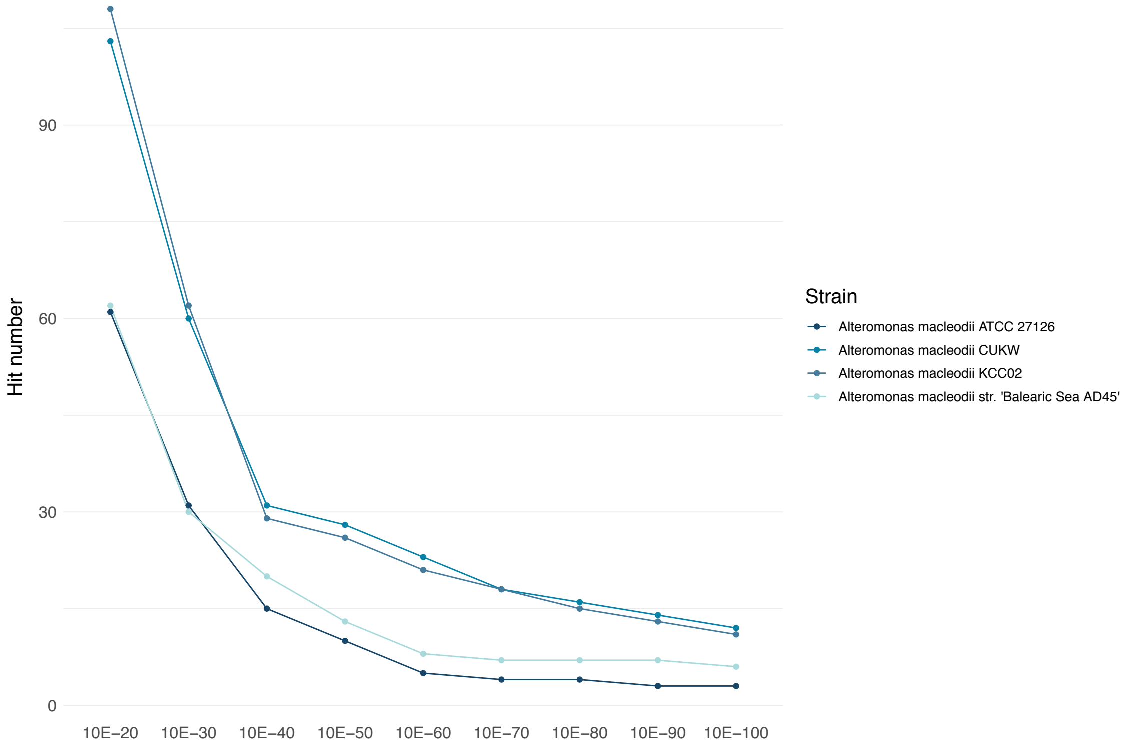


Figure 9: Found hits in Alteromonas macleodii strains

There seems to be no difference between *Alteromonas macleodii ATCC 27126* and *Alteromonas macleodii ‘Balearic Sea AD45’*, as well as between *Alteromonas macleodii CUKW and Alteromonas macleodii KCC02*. However, these two groups are quite distant from each other at all E-values.

For the purpose of testing if this difference is significant, Dunn’s test was again performed. Results can be seen in Table 5.

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii CUKW	-2.827501	0.0141 *
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii KCC02	-2.627442	0.0258
Alteromonas macleodii CUKW - Alteromonas macleodii KCC02	0.200059	1.0000
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii str. 'Balearic Sea AD45'	-0.840248	1.0000

Alteromonas macleodii CUKW - Alteromonas macleodii str. 'Balearic Sea AD45'	1.987253	0.1407
Alteromonas macleodii KCC02 - Alteromonas macleodii str. 'Balearic Sea AD45'	1.787194	0.2217

Table 5: Alteromonas macleodii strains pairwise comparisons

The previous data shows differences between the strains taking into account the whole set of hits found. Although important information to have an overall overview, these plots do not give insight into the different mechanisms of copper resistance that the different bacteria might have. On this regard, the data represented in the previous figures was divided into plots by copper resistance system in Figures 10, 11 and 12.

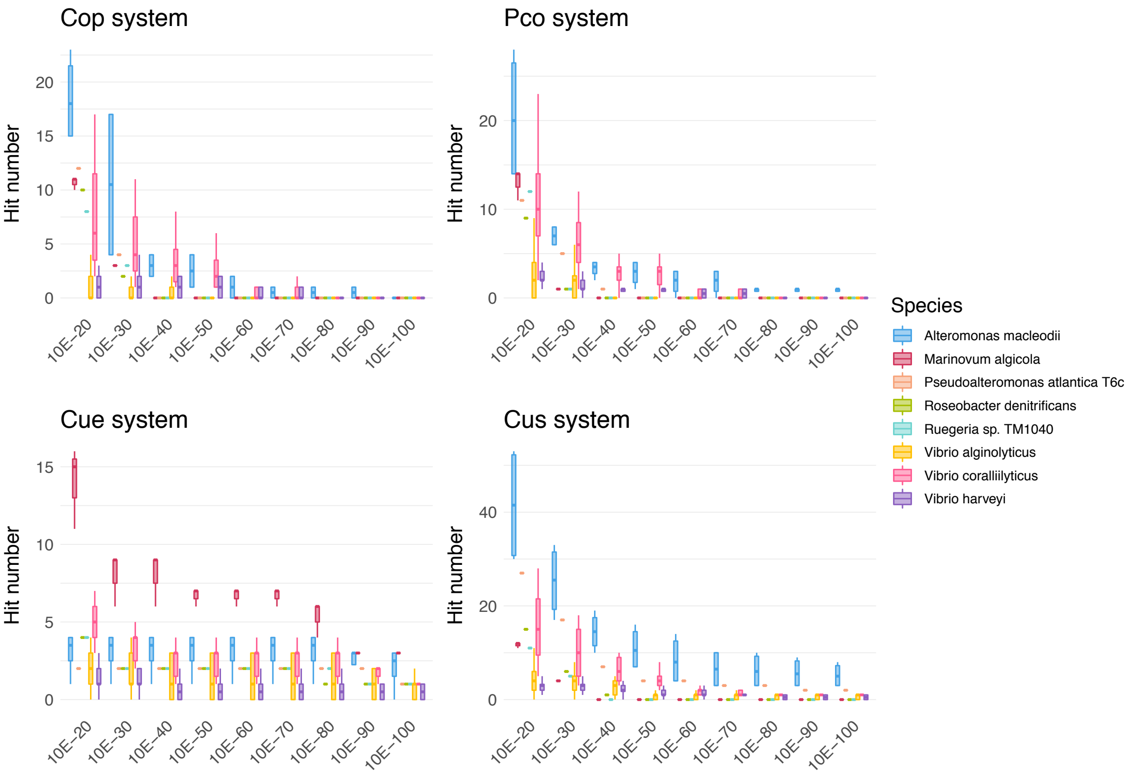


Figure 10: Found hits grouped by species per system

Globally, we can see how all systems follow similar dynamics as in previous figures, with the exception of the Cue system. In this case, *Marinovum algicola* species seems to be the one enriched in this system and both *Alteromonas macleodii* and *Vibrio coralliilyticus* are less represented in strike difference to the other systems.

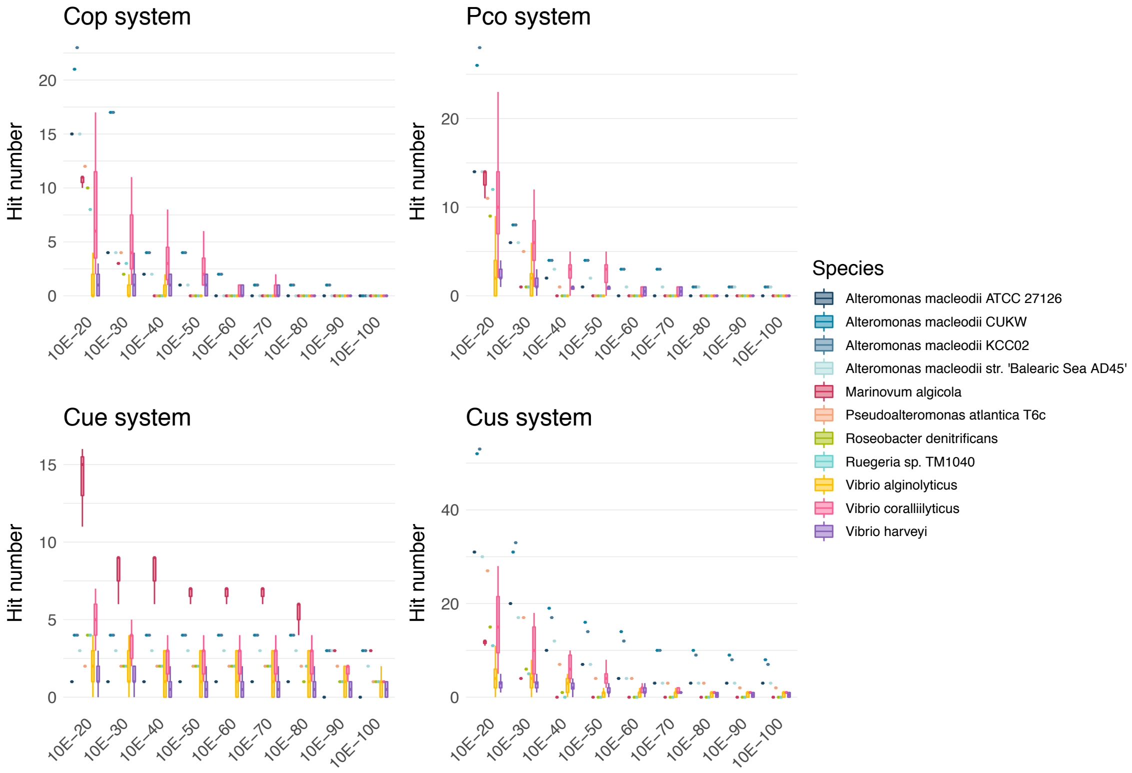


Figure 11: Found hits grouped by species compared to Alteromonas macleodii strains per system

For all systems with the exception of Cue, *Alteromonas macleodii CUKW* and *Alteromonas macleodii KCC02* show higher hit number than the rest of the species, mostly at lower E-values. This result suggests, as we hypothesized, that these two strains count on copper-resistance systems that allowed them to survive high copper concentrations.

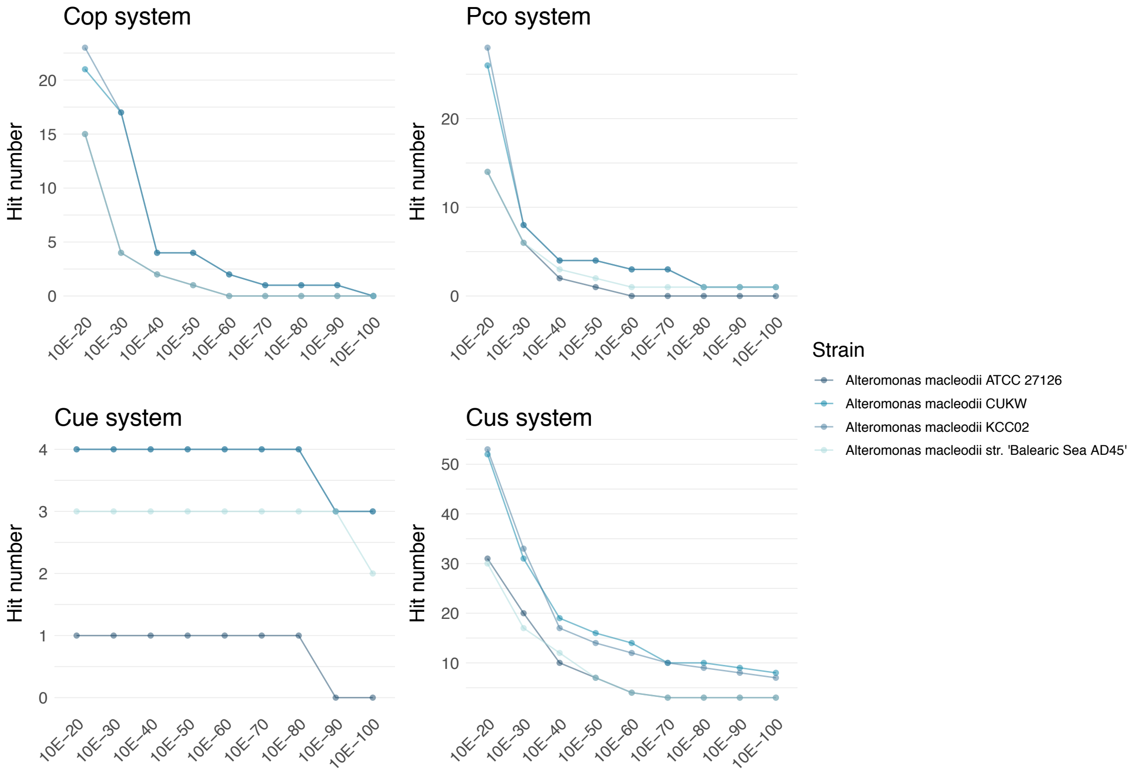


Figure 12: Found hits in Alteromonas macleodii strains per systems

When comparing between *Alteromonas macleodii* strains, all four systems are mainly represented in *Alteromonas macleodii CUKW* and *Alteromonas macleodii KCC02*, with no difference between the last two. In line with what has been shown in Figure 10 and Figure 11, Cue system is the less abundant in hit numbers among the copper-resistance systems.

Statistical analyses per system were also performed and can be consulted in the annex (Tables 7, 8, 9 and 10). In all cases we find significant differences. However, depending on the system the species involved are different. Importantly, we can see that the system that shows highest variety is Cus. Regarding comparison among *Alteromonas macelodii* strains, the results are shown in Tables 11, 12, 13 and 14 included in the annex. Interestingly, even if the plots show a higher number of hits for both *CUKW* and *KCC02* strains in all systems, this enrichment only shows a statistical significance for the Cue system, and only when compared to *ATCC 27126* strain.

Next, the representation of the different genes per system was explored. For this, the number of hits per gene was plotted per *Alteromonas* strain together with the average number of hits found in the other species for comparison at different E-value thresholds (Figures 13, 14, 15, 16 and 17).

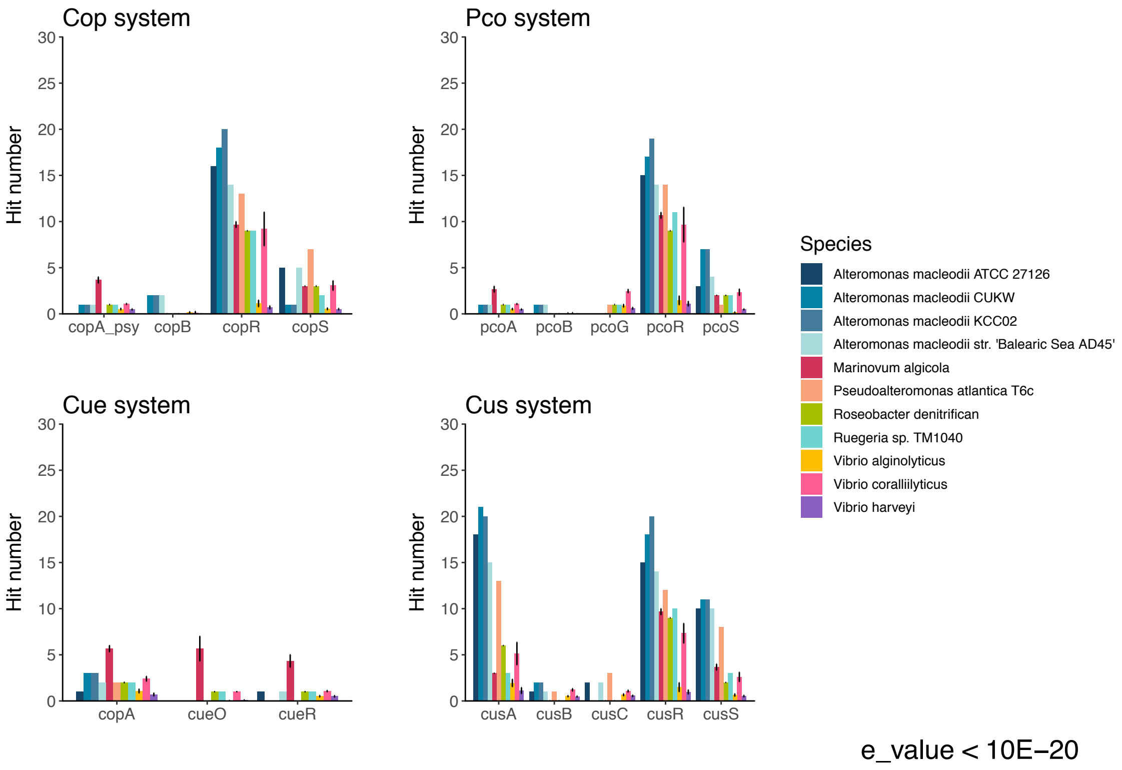


Figure 13: Average hits found per gene per species compared to Alteromonas macleodii strains at E-value <10E-20

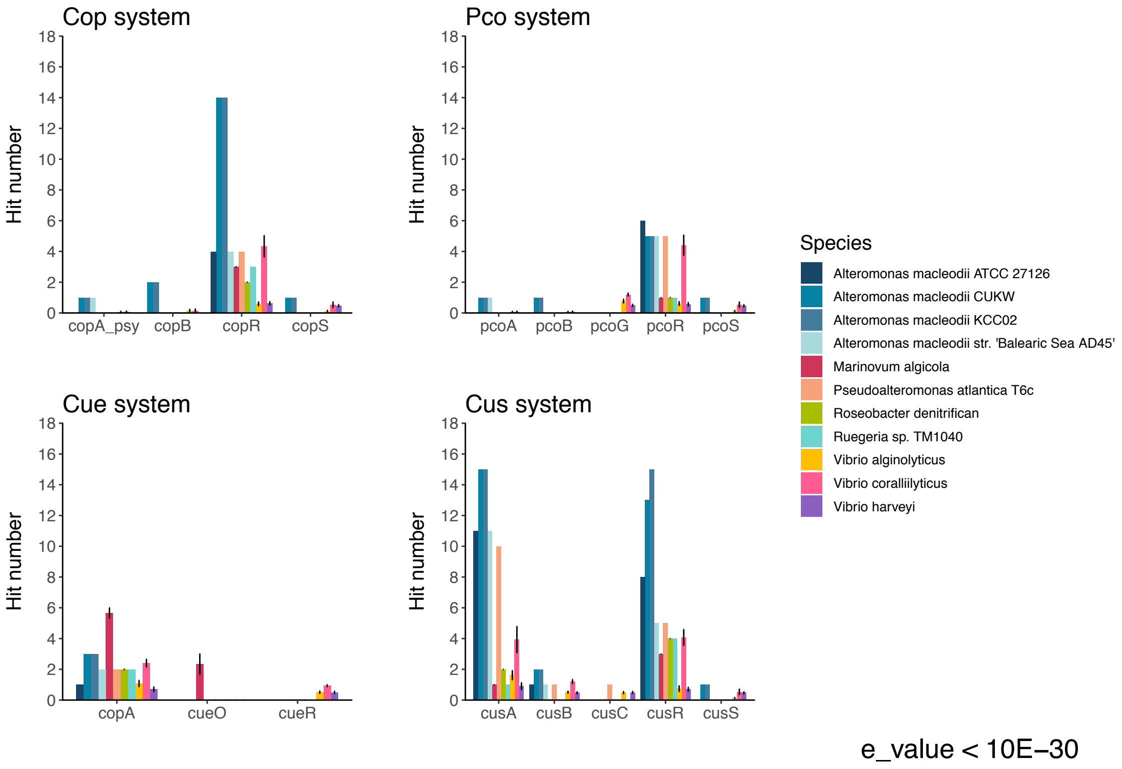


Figure 14: Average hits found per gene per species compared to Alteromonas macleodii strains at E-value <10E-30

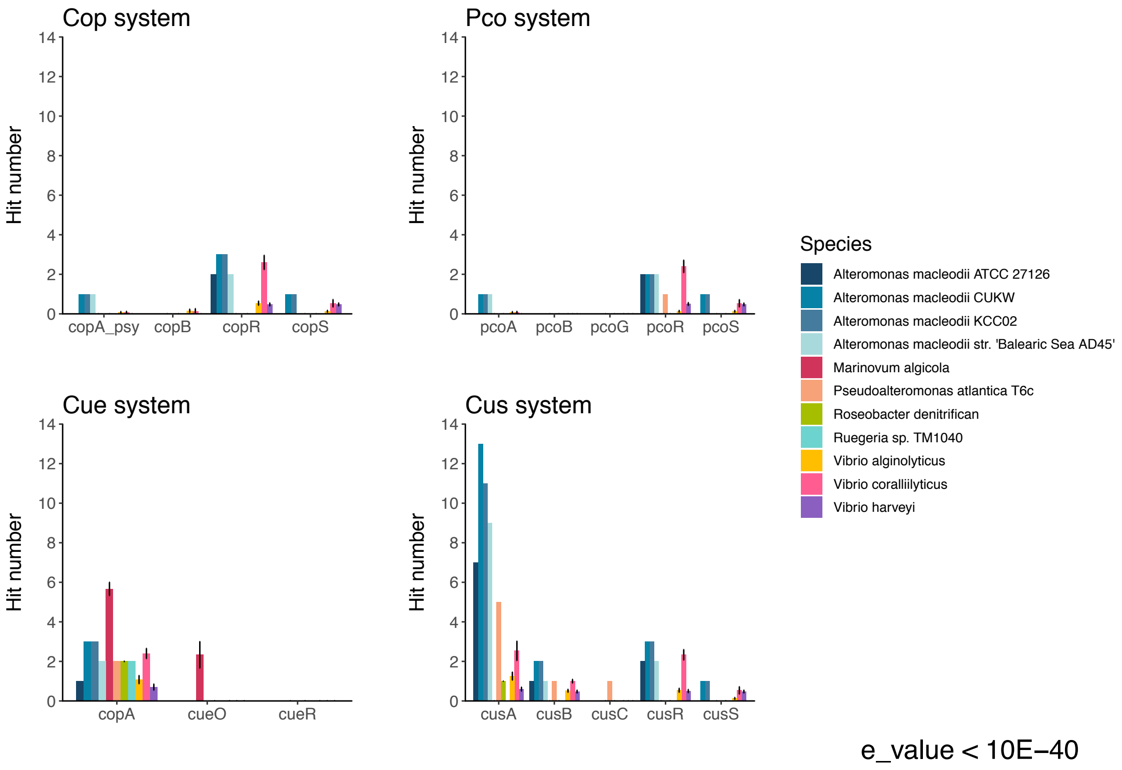


Figure 15: Average hits found per gene per species compared to Alteromonas macleodii strains at E-value <10E-40

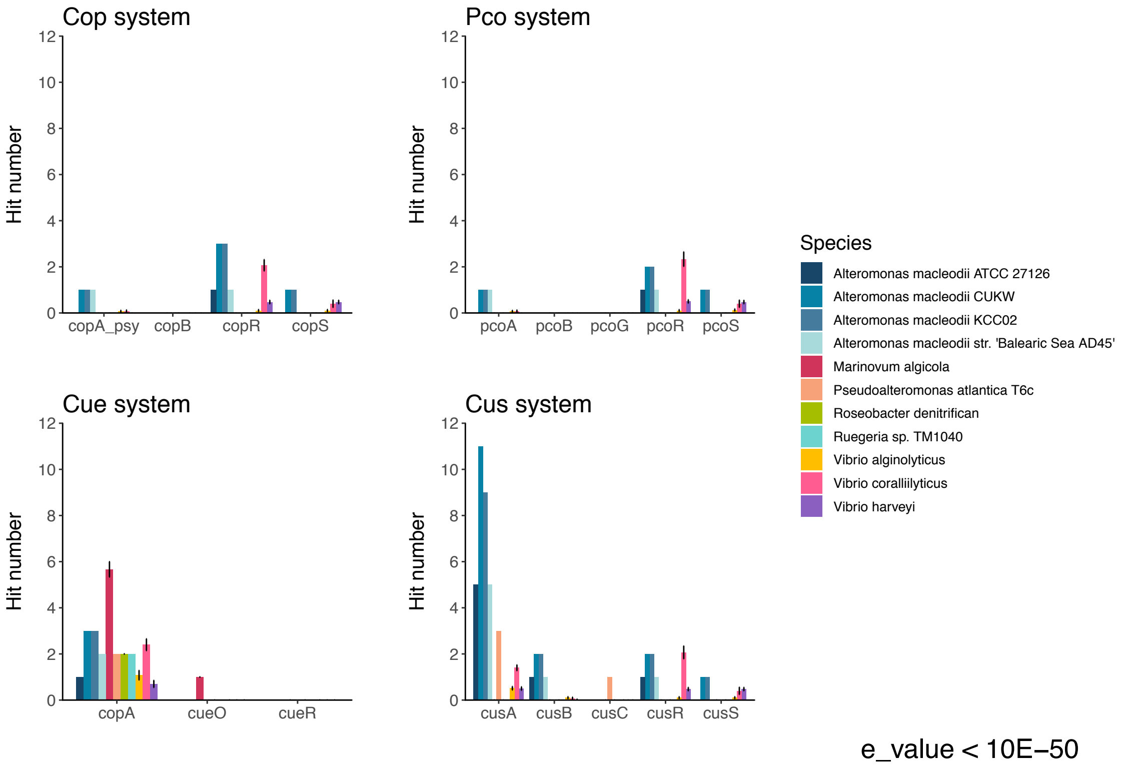


Figure 16: Average hits found per gene per species compared to Alteromonas macleodii strains at E-value <10E-50

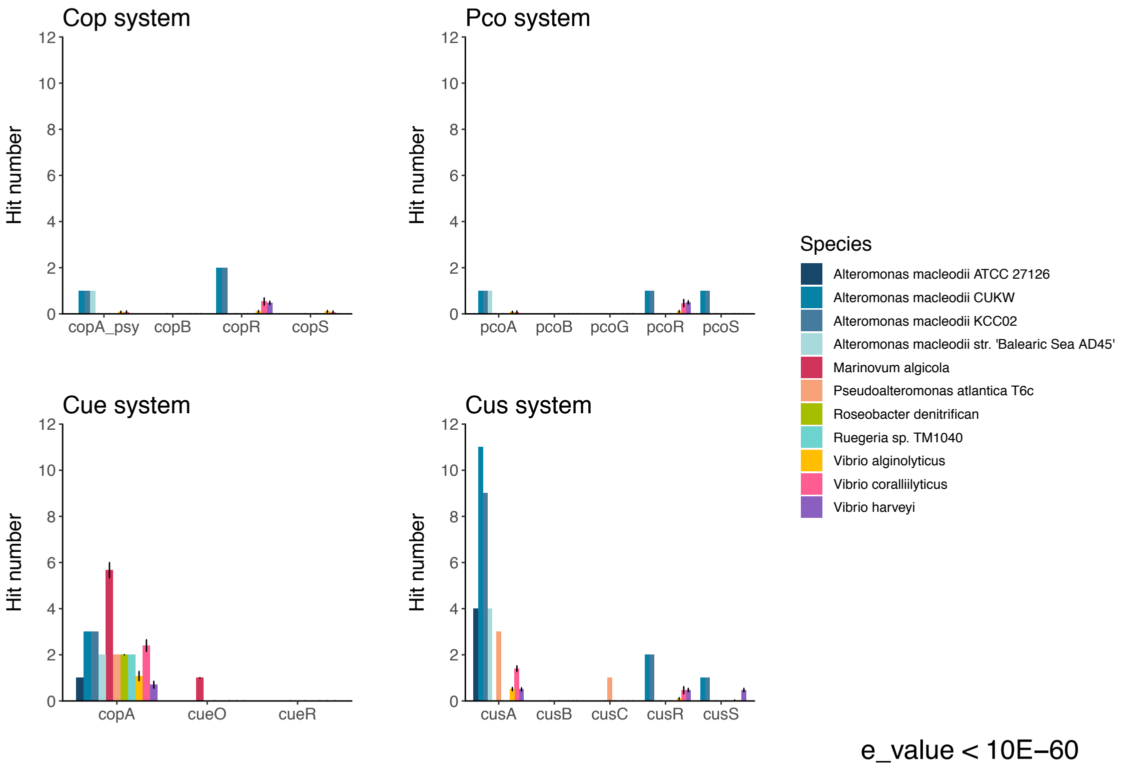


Figure 17: Average hits found per gene per species compared to Alteromonas macleodii strains at E-value <10E-60

Several interesting observations can be made from these figures. First, we can see copB and copR enriched for the Cop system in the resistant strains; pcoR and pcoS in the Pco system; copA in Cue and cusA, cusR and cusS in the Cus system. However, as E-value decreases, copA, copR and copS are the mostly represented for Cop system; pcoA, pcoR and pcoS for Pco; copA for Cue and cusA, cusR and cusS for Cus. Therefore, for most systems we have the representation of the Cu^+ exporter protein (the A elements), the activator transcription factor (the R elements) and the Cu+ sensor and transcription factor activator (the S elements). This could represent a basic working network for the systems to be functional. In the case of the Cue system, we only see the Cu^+ exporter represented. Even if not activated by its own system's transcriptional activator, and in absence of other system components, this protein can be a support to the other systems on copper export, as it can bind to the ion and export it out of the cell without the help of any of its system's other factors. Regarding the Pco system, apart from the sensor and transcription factor, the putative Cu^+ oxidase is conserved.

3.2. Identification of transcriptional regulators of copper-resistance systems

With the aim of finding transcriptional regulators of copper-resistance systems first in *Alteromonas* genus and then up to Alteromonadales full order, a search for conserved sequence motifs in the gene promoters was performed.

First, promoter sequences of genes of interest were gathered. Since these genes organize in operons, the structure of those was analysed to see which gene comes first in the transcription of the operon and therefore would have more chances of containing the putative regulating transcription factor motif on its promoter. Basic schemes of these operons are shown in Figures 21 to 26 in the annex.

After looking at the structures of the operons for all the copper-resistance systems, the list of genes of interest was narrowed to six (Table 6).

Accession	**Gene name**
NP_415020.1	cueR
NP_414665.1	cueO
NP_415103.1	cusR
AFX60851.1	pcoF
ANH09828.1	pcoA
AQX42270.1	copA

Table 6: Genes selected for gathering promoter sequences

Since the interest is to find common regulators conserved in *Alteromonas* first, the search was narrowed down to orthologs found in this genus. This process was also automatized with a python script using blastp tool from BLAST. How this script works is explained in materials and methods section. The complete code with detailed comments can be consulted in the annex.

In order to find conserved motifs among the sequences, MEME was locally installed and used[29,30]. This tool provides a group of algorithms to discover novel motifs in collections of unaligned nucleotide or protein sequences. The promoter sequences were used as input for MEME tool and searched for both palindromic and non-palindromic sequences. More details into the search parameters can be read in materials and methods.

Starting with the sequences obtained from *Alteromonas*, the MEME search returned several motifs discovered. The search was then expanded to *Alteromonadaceae* family and the Alteromonadales order. Interestingly, one of the hits was found as one of the best regarding E-value in all three searches, suggesting a conservation of this sequence in the studied order, family and genus (Figure 18). An example of the reports of these searches can be consulted in the annex.

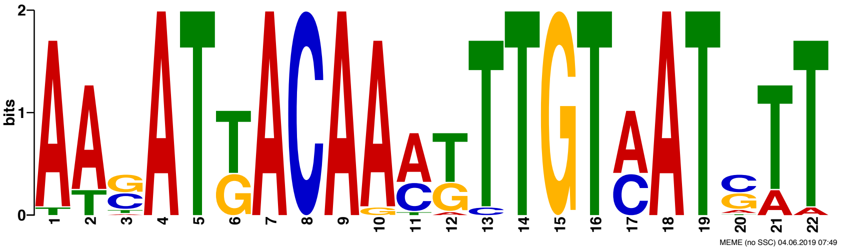
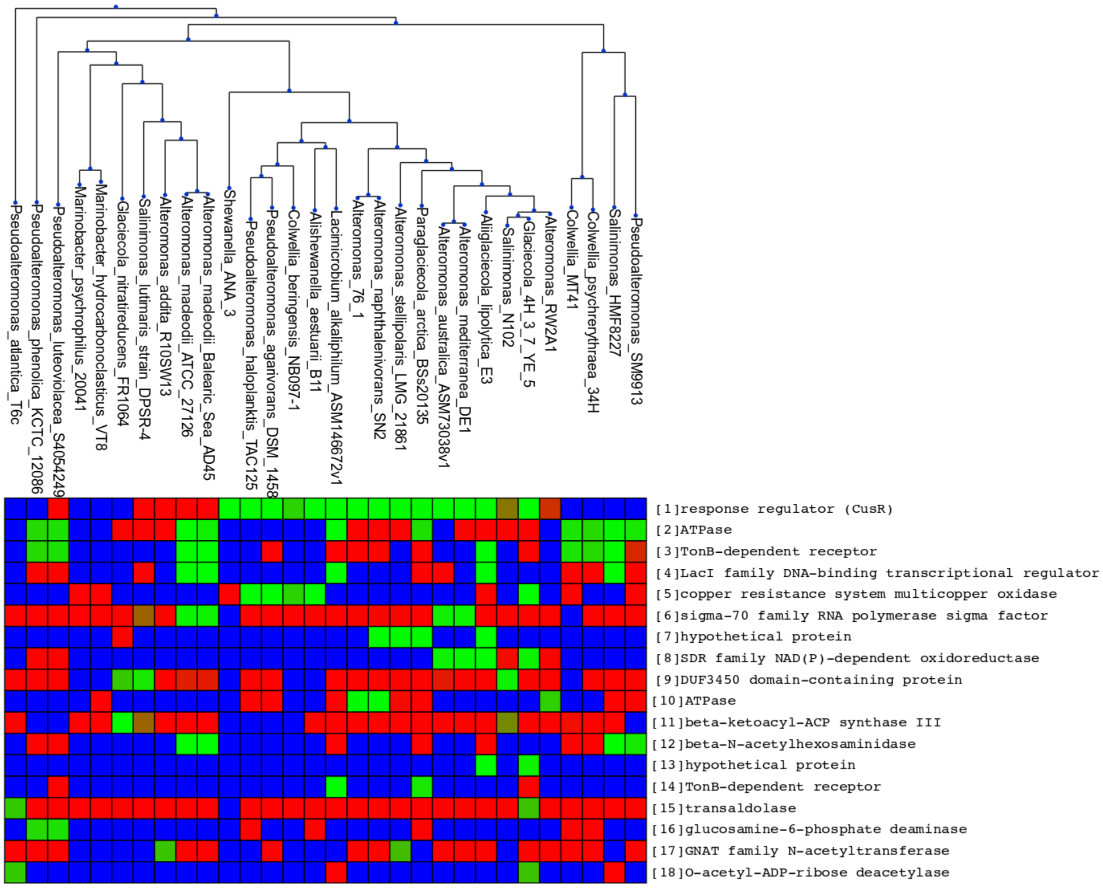


Figure 18: Found motif logo

This sequence matches the described sequence motif for CusR in *E. coli*[34], indicating that CusR is probably regulating itself and other copper-resistance genes as it happens in other species.

Next, how well this regulation by CusR is conserved in Alteromonadales order was studied. On this regard, a program developed by the host laboratory called CGB was used[12]. Using this motif as input, it searched for the presence of CusR orthologs in the different species and looked for the presence of the binding motif in the promoters of annotated genes. The results can be seen edited in Figure 19 (whole figure is shown in the annex as Figure 27).



(···)

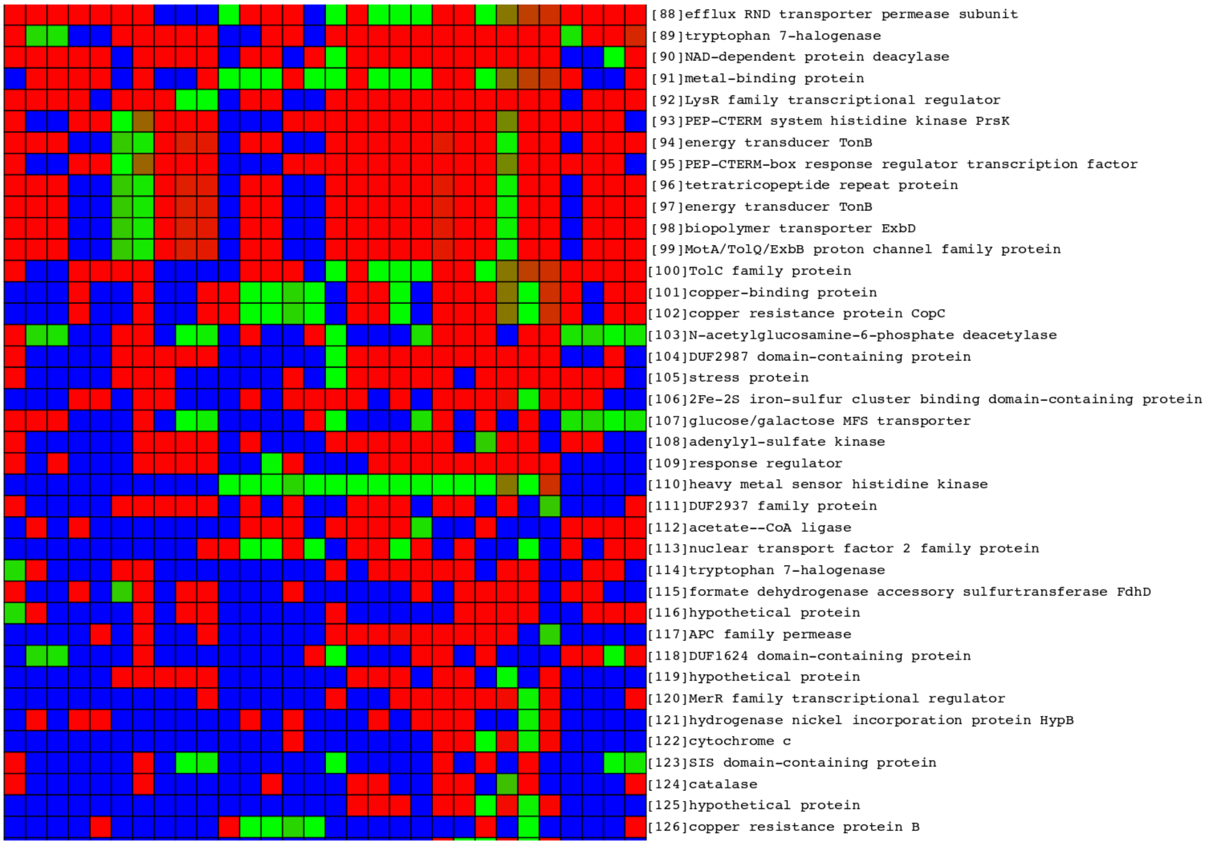


Figure 19: heatmap containing CGB results (edited)

The top part of the heatmap shows the species trees based on the CusR protein sequence conservation. On the right side, the name of the genes in which the motif presence is evaluated is shown. Regarding the boxes, blue ones mean that no ortholog for those genes was found in that species. When red, the gene is found and there is no presence of the motif on its promoter; when green, the motif has been found in that gene (above a delimited threshold).

At first sight we can see that CusR regulates itself in most of the species where it has been identified, including most *Alteromonas* (clustered green boxes in first line). Moreover, when following the presence of the CusR motif in these species, we can see green boxes related to copper-resistance (or metal ion resistance) related genes. This is the case for example of:

- [5] copper resistance system multicopper oxidase
- [81] efflux RND transporter periplasmic adaptor subunit
- [88] efflux RND transporter permease subunit
- [91] metal-binding protein
- [101] copper-binding protein
- [102] copper resistance protein CopC
- [110] heavy metal sensor histidine kinase
- [126] copper resistance protein B

Interestingly, *Alteromonas macleodii* is not included in this group of species. It clusters further away from the other *Alteromonas species* indicating that the protein sequence differs in this species. Moreover, its CusR binding motif is not enriched in these genes, but others. This suggests that CusR protein in *Alteromonas macleodii* is regulating itself and other copper-resistance related genes binding to a different motif sequence or maybe through another transcription factor. This change in functionality might be related to the change in protein sequence.

4. Discussion

The results obtained in this project started to be shaped since the moment the proteins of interest to be used for the search were selected. These proteins represented the most well known factors in copper resistance systems. However, it cannot be discarded that other unknown factors can also be present in our species of interest, and therefore the analysis is far from being complete. Regardless of this limitation, taking into account that the selected proteins were based not only in literature search but also in discussion with Cusick lab, a good representation is expected to have been chosen and therefore a good general picture of the copper resistance systems presence has been hopefully achieved.

In the beginning of the project, BLAST search was performed to have a general idea of putative protein orthologs that could be found in *Alteromonas macleodii* and others. This decision was made thinking about having a first general idea of the representation of these proteins and to use as a basis to build up all the analysis pipeline, including the script writing and graphic generation in R. However, the use of BLAST this way also has a disadvantage. When a hit is found, there is a calculation measuring of how likely it is that this hit is an homologous sequence, however, it cannot assure this. Another approach that was planned to be implemented but was not possible yet due to time limitations was looking for the reciprocal hits of the BLAST results. This method is named Reciprocal Best Hits (RBH) and is based on the idea that two genes from different species are considered orthologs if when performing BLAST (or another alignment approach) with each of them, they both find each other as the best scoring match[35].

Nevertheless, with applied methods, interesting data has been obtained. When first looking at the main differences in number of copper-resistance proteins in all species (Figure 7), it is clear that all of them contain different numbers. These differences can be due to the different environments where these bacteria leave. It would be reasonable to think that bacteria living in areas were no copper is found would not need this kind of systems. On the other hand, between the ones that might live in copper containing waters, the representation of copper-resistance genes might be correlated with the concentrations of copper they have to deal with, as well as how often is that copper present. In this sense, it has been shown that *Alteromonas macleodii CUKW* and *KCC02* show the greatest representation of these systems, what correlates with the fact that they were isolated from a piece of copper and therefore in constant contact with high copper concentrations.

Regarding the differences observed between systems, *Alteromonas macleodii CUKW* and *KCC02* have the most representation in all of them but Cue system, where *Marinovum algicola* is the species showing highest number of hits. This could be because Cue system does not add too much effectiveness on resistance when the bacteria already has the other systems. For *Marinovum algicola* might be the only system present in its genome (Figures 11 and 13). This could be due to the species handling a low copper concentration in its environment, where some efflux is needed to maintain homeostasis but it is not really a threatening concentration. Regarding the comparison of strains inside *Alteromonas macleodii*, *CUKW* and *KCC02* strains are dominant in all systems (Figure 12).

When the different components of these systems were looked into in detail, it became clear that not all components of all the systems are conserved, only a fraction of them (Figures 13 to 17). This suggests that a set of copper-resistance genes has been enriched more than a specific system. Importantly, the conserved ones could be the minimum functional system, as they have the functionality of sensing copper and activating a transcription regulator that will in turn activate transcription of an effector protein that will export or oxidase the copper.

Once this data was analysed, the possible transcriptional regulatory mechanisms of these system were studied. When looking for a sequence motif conserved in the genes of interest in *Alteromonas*, *Alteromonadaceae* and Alteromonadales, a sequence having a biological relevance was expected to be found, since the more conserved, the more relevant the functionality of the sequences tend to be. After the search, a conserved sequence motif was found indeed, which turned out to be the motif known to be recognized by CusR in *E. coli* (Figure 18). This result indicates that the regulation mechanism going on in *E. coli* is also probably going on in most species belonging to Alteromonadales order.

In order to have more insight in the regulatory system by CusR, the presence of its binding motif in Alteromonadales order species was studied using the CGB program. This showed a correlation between the presence of the sequence motif bound by CusR and copper-resistance related genes. However, interestingly, this was not the case for *Alteromonas macleodii* species, which diverge not only from other different species but also from other *Alteromonas* species too (Figure 19). This divergence can be observed from CusR protein sequence point of view as well as from regulated genes cluster.

This can result from different situations. One possibility is that CusR is binding a different motif in copper resistance genes in this species, idea reinforced by the fact that the protein sequence is also different in this species. It could be that the system had switched to another one where there is higher or lower affinity between the transcription factor and its binding motif in order to keep levels of expression of these genes higher or lower, or even constant or completely silenced. Biological experiments are being performed regarding copper resistance of these strains that might shed light on this idea. A second possibility could be that the regulation mechanism has added another player in this species. This other protein could be a transcriptional regulator regulated by CueR that then would activate the copper-resistance genes.

In order to better understand what is the system in *Alteromonas macleodii*, a new sequence motif search exclusive for *Alteromonas macleodii* would be useful. If any good candidate is found, and with the complementation of biological experiments, this regulation system might be characterized.

5. Conclusion

Throughout this project, comparative genomics tools have been used in order to answer biological questions. This has led to the development of useful scripts, a deepening in knowledge on both python language and biopython tools as well as on the fields of comparative genomics and resistance systems in bacteria. Importantly, data has been generated that will help answer raised biological questions and will hopefully in the future be incorporated into a scientific publication to be shared with the community.

Looking back at the set goal in the beginning of the project, it can be said that all of them have been worked on and at least partially fulfilled. However, in order to fully complete them, approaches discussed in the discussion should be implemented, and the research on the regulatory system should also continue. In this sense, a fully completed project where all questions have been answered and more bioinformatical approached have been implemented in order to reinforce the preliminary results would require much more time than the one available for this project. Around 6 months would probably be a good time assignment for it.

As it was explained in previous reports during the development of this project, the timing worked well for the first part of the project. However, some problems delayed the advancement to the second part. Although they were solved, more time for the second half of the project would have been beneficial and some more searches could have been run. Problems are of course always expected, but the time assigned to have fulfilled each specific case, taking into account possible setbacks was maybe underestimated. The project could have focused only on the first objective and have completed it more thoroughly. Even so, in that case none of the tasks of the second objective would have taken place, and they have been profitable not only in found results but mainly in developed skills and learned methods. Therefore, the decision made to try to complete both might have been the most beneficial one.

Future perspectives for the project include the already described ones in the discussion section. These include a more robust approach to search for ortholog proteins and further analysis in the transcriptional regulation of the copper-resistance genes.

6. Glossary

Cu^+: copper(1+), a copper ion. It has a strong reductase property, what leads to its oxidation to Cu^{2+}.

Cu^{2+}: copper(2+), a copper ion. The most common oxidized level of copper.

Homology: existence of shared ancestry between two genes (or characters).

Orthology: subtype of homology. Used when homologous genes are generated through speciation.

DNA sequence motif: nucleotide pattern that is widespread and has biological significance. For example, a transcription factor can recognize it and activate transcription of adjacent coding sequence.

Script: a file containing orders, used to automate the execution of tasks. Might be considered as a simple program.

json: JavaScript Object Notation. Simple text format used for data exchange.

csv: Comma Separated Value. File format used to represent data in tables.

fasta: text format used in bioinformatics to represent sequence data from nucleotide or amino acids.

CGB: a complete comparative genomics platform built previously in the host lab by Sefa Kılıç. Its aim is to analyse transcriptional regulation on any annotated bacterial genome.

BLAST: Basic Local Alignment Search Tool. An algorithm for comparing nucleotide and amino acid sequence information. Aligns sequences looking for resemblance and calculates the significance of the result. The comparison can be made against a vast database of annotated sequences.

7. Bibliography

1. Solioz, M., Abicht, H. K., Mermod, M. & Mancini, S. Response of gram-positive bacteria to copper stress. *Journal of biological and inorganic chemistry JBIC* **15**, 3–14 (2010).
2. Borkow, G. & Gabbay, J. Copper, An Ancient Remedy Returning to Fight Microbial, Fungal and Viral Infections. (2009). doi:info:doi/10.2174/1872313097890548 87
3. Leite, G. & Padoveze, M. Copper as an antimicrobial agent in healthcare: an integrative literature review. *J Infect Cont* **1**, (2012).
4. Grass, G., Rensing, C. & Solioz, M. Metallic Copper as an Antimicrobial Surface. *Appl. Environ. Microbiol.* **77**, 1541–1547 (2011).
5. Lamichhane, J. R. *et al.* Thirteen decades of antimicrobial copper compounds applied in agriculture. A review. *Agron. Sustain. Dev.* **38**, 28 (2018).
6. Earley, P. J. *et al.* Life cycle contributions of copper from vessel painting and maintenance activities. *Biofouling* **30**, 51–68 (2014).
7. Schultz, M. P., Bendick, J. A., Holm, E. R. & Hertel, W. M. Economic impact of biofouling on a naval surface ship. *Biofouling* **27**, 87–98 (2011).
8. Ingle, A. P., Duran, N. & Rai, M. Bioactivity, mechanism of action, and cytotoxicity of copper-based nanoparticles: A review. *Appl Microbiol Biotechnol* **98**, 1001–1009 (2014).
9. Chaturvedi, K. S. & Henderson, J. P. Pathogenic adaptations to host-derived antibacterial copper. *Front. Cell. Infect. Microbiol.* **4**, (2014).
10. Pal, C. *et al.* Chapter Seven - Metal Resistance and Its Association With Antibiotic Resistance. in *Advances in Microbial Physiology* (ed. Poole, R. K.) **70**, 261–313 (Academic Press, 2017).
11. López-López, A., Bartual, S. G., Stal, L., Onyshchenko, O. & Rodríguez-Valera, F. Genetic analysis of housekeeping genes reveals a deep-sea ecotype of Alteromonas macleodii in the Mediterranean Sea. *Environmental Microbiology* **7**, 649–659 (2005).
12. Kilic, S. *Enhancing comparative genomics of transcriptional regulatory networks through data collection, transfer and integration*. (University of Maryland, Baltimore County, 2016).
13. Spyder Website. Available at: https://www.spyder-ide.org/. (Accessed: 4th June 2019)
14. Cock, P. J. A. *et al.* Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009).
15. Bio. Available at: https://biopython.org/DIST/docs/api/Bio-module.html. (Accessed: 4th June 2019)
16. RStudio Team. *RStudio: Integrated Development Environment for R*. (RStudio, Inc., 2015).
17. Wickham, H. *ggplot2: Elegant Graphics for Data Analysis*. (Springer-Verlag New York, 2016).
18. Bio.Blast.NCBIWWW. Available at: https://biopython.org/DIST/docs/api/Bio.Blast.NCBIWWW-module.html. (Accessed: 2nd June 2019)
19. Bio.Blast. Available at: https://biopython.org/DIST/docs/api/Bio.Blast-module.html. (Accessed: 2nd June 2019)
20. Bio.Blast.NCBIXML. Available at:

https://biopython.org/DIST/docs/api/Bio.Blast.NCBIXML-module.html. (Accessed: 5th June 2019)

21. Bio.SeqIO. Available at: https://biopython.org/DIST/docs/api/Bio.SeqIO-module.html. (Accessed: 5th June 2019)

22. Bio.Entrez. Available at: https://biopython.org/DIST/docs/api/Bio.Entrez-module.html. (Accessed: 5th June 2019)

23. Camacho, C. *et al.* BLAST+: architecture and applications. *BMC Bioinformatics* **10**, 421 (2009).

24. National Center for Biotechnology Information (US) & Camacho, C. *BLAST (r) Command Line Applications User Manual*. (National Center for Biotechnology Information (US), 2008).

25. Van Der Walt, S., Colbert, S. C. & Varoquaux, G. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering* **13**, 22 (2011).

26. Jones, E., Oliphant, T., Peterson, P. & others. *SciPy: Open source scientific tools for Python*. (2001).

27. Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in science & engineering* **9**, 90 (2007).

28. Dinno, A. & Dinno, M. A. Package ‘dunn. test’. (2017).

29. Bailey, T. L. & Elkan, C. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *Proc Int Conf Intell Syst Mol Biol* **2**, 28–36 (1994).

30. MEME - MEME Suite. Available at: http://meme-suite.org/doc/meme.html?man_type=web. (Accessed: 2nd June 2019)

31. MEME - Submission form. Available at: http://meme-suite.org/tools/meme. (Accessed: 4th June 2019)

32. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *Journal of Molecular Biology* **215**, 403–410 (1990).

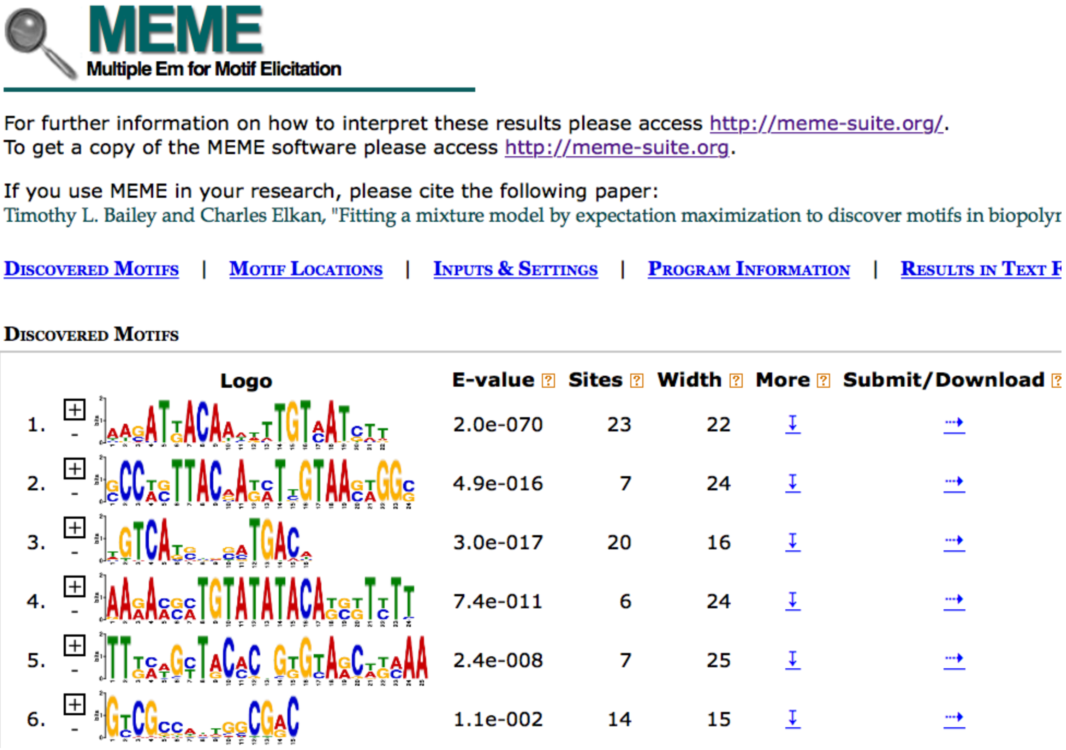
33. Altschul, S. F. *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* **25**, 3389–3402 (1997).

34. Urano, H., Umezawa, Y., Yamamoto, K., Ishihama, A. & Ogasawara, H. Cooperative regulation of the common target genes between H2O2-sensing YedVW and Cu2+-sensing CusSR in Escherichia coli. *Microbiology* **161**, 729–738 (2015).

35. Ward, N. & Moreno-Hagelsieb, G. Quickly Finding Orthologs as Reciprocal Best Hits with BLAT, LAST, and UBLAST: How Much Do We Miss? *PLOS ONE* **9**, e101850 (2014).

8. Annex

8.1. MEME report example



For further information on how to interpret these results please access http://meme-suite.org/.
To get a copy of the MEME software please access http://meme-suite.org.

If you use MEME in your research, please cite the following paper:
Timothy L. Bailey and Charles Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolyr

DISCOVERED MOTIFS | MOTIF LOCATIONS | INPUTS & SETTINGS | PROGRAM INFORMATION | RESULTS IN TEXT F

DISCOVERED MOTIFS

	Logo	E-value	Sites	Width	More	Submit/Download
1.		2.0e-070	23	22	↧	⇢
2.		4.9e-016	7	24	↧	⇢
3.		3.0e-017	20	16	↧	⇢
4.		7.4e-011	6	24	↧	⇢
5.		2.4e-008	7	25	↧	⇢
6.		1.1e-002	14	15	↧	⇢

Figure 20: Preview of a section of a MEME results report

8.2. Extended statistical analysis from first section of the results

List of pairwise comparisons	Z statistic	Adjusted p-value
Alteromonas macleodii - Marinovum algicola	4.119190	0.0005 *
Alteromonas macleodii - Pseudoalteromonas atlantica T6c	2.784050	0.0752
Marinovum algicola - Pseudoalteromonas atlantica T6c	-0.028943	1.0000
Alteromonas macleodii - Roseobacter denitrificans	4.218533	0.0003 *
Marinovum algicola - Roseobacter denitrificans	0.092926	1.0000
Pseudoalteromonas atlantica T6c - Roseobacter denitrificans	0.094652	1.0000
Alteromonas macleodii - Ruegeria sp. TM1040	2.813942	0.0685
Marinovum algicola - Ruegeria sp. TM1040	0.000000	1.0000
Pseudoalteromonas atlantica T6c - Ruegeria sp. TM1040	0.023632	1.0000
Roseobacter denitrificans - Ruegeria sp. TM1040	-0.065709	1.0000
Alteromonas macleodii - Vibrio alginolyticus	6.023687	0.0000 *

Marinovum algicola - Vibrio alginolyticus	0.027445	1.0000
Pseudoalteromonas atlantica T6c - Vibrio alginolyticus	0.049237	1.0000
Roseobacter denitrificans - Vibrio alginolyticus	-0.099192	1.0000
Ruegeria sp. TM1040 - Vibrio alginolyticus	0.016237	1.0000
Alteromonas macleodii - Vibrio coralliilyticus	1.562453	1.0000
Marinovum algicola - Vibrio coralliilyticus	-3.584192	0.0047 *
Pseudoalteromonas atlantica T6c - Vibrio coralliilyticus	-2.162501	0.4281
Roseobacter denitrificans - Vibrio coralliilyticus	-3.704160	0.0030 *
Ruegeria sp. TM1040 - Vibrio coralliilyticus	-2.194860	0.3944
Vibrio alginolyticus - Vibrio coralliilyticus	-7.515215	0.0000 *
Alteromonas macleodii - Vibrio harveyi	3.784815	0.0022 *
Marinovum algicola - Vibrio harveyi	-1.940011	0.7333
Pseudoalteromonas atlantica T6c - Vibrio harveyi	-1.114049	1.0000
Roseobacter denitrificans - Vibrio harveyi	-2.066763	0.5426
Ruegeria sp. TM1040 - Vibrio harveyi	-1.147059	1.0000
Vibrio alginolyticus - Vibrio harveyi	-5.233632	0.0000 *
Vibrio coralliilyticus - Vibrio harveyi	3.651454	0.0037 *

Table 7: Cop system species pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii - Marinovum algicola	6.098236	0.0000 *
Alteromonas macleodii - Pseudoalteromonas atlantica T6c	3.322900	0.0125 *
Marinovum algicola - Pseudoalteromonas atlantica T6c	-0.816219	1.0000
Alteromonas macleodii - Roseobacter denitrificans	6.098236	0.0000 *
Marinovum algicola - Roseobacter denitrificans	0.000000	1.0000
Pseudoalteromonas atlantica T6c - Roseobacter denitrificans	0.816219	1.0000
Alteromonas macleodii - Ruegeria sp. TM1040	4.165888	0.0004 *
Marinovum algicola - Ruegeria sp. TM1040	0.000000	1.0000
Pseudoalteromonas atlantica T6c - Ruegeria sp. TM1040	0.666440	1.0000
Roseobacter denitrificans - Ruegeria sp. TM1040	0.000000	1.0000
Alteromonas macleodii - Vibrio alginolyticus	8.116430	0.0000 *
Marinovum algicola - Vibrio alginolyticus	-0.661535	1.0000
Pseudoalteromonas atlantica T6c - Vibrio alginolyticus	0.539263	1.0000
Roseobacter denitrificans - Vibrio alginolyticus	-0.661535	1.0000
Ruegeria sp. TM1040 - Vibrio alginolyticus	-0.391369	1.0000

Alteromonas macleodii - Vibrio coralliilyticus	3.852328	0.0016 *
Marinovum algicola - Vibrio coralliilyticus	-3.936685	0.0012 *
Pseudoalteromonas atlantica T6c - Vibrio coralliilyticus	-1.498156	1.0000
Roseobacter denitrificans - Vibrio coralliilyticus	-3.936685	0.0012 *
Ruegeria sp. TM1040 - Vibrio coralliilyticus	-2.410717	0.2229
Vibrio alginolyticus - Vibrio coralliilyticus	-6.890299	0.0000 *
Alteromonas macleodii - Vibrio harveyi	5.694137	0.0000 *
Marinovum algicola - Vibrio harveyi	-2.792429	0.0732
Pseudoalteromonas atlantica T6c - Vibrio harveyi	-0.720139	1.0000
Roseobacter denitrificans - Vibrio harveyi	-2.792429	0.0732
Ruegeria sp. TM1040 - Vibrio harveyi	-1.651063	1.0000
Vibrio alginolyticus - Vibrio harveyi	-5.666743	0.0000 *
Vibrio coralliilyticus - Vibrio harveyi	2.702439	0.0964

Table 8: Pco system species pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii - Marinovum algicola	-2.356893	0.2580
Alteromonas macleodii - Pseudoalteromonas atlantica T6c	0.797447	1.0000
Marinovum algicola - Pseudoalteromonas atlantica T6c	2.331064	0.2765
Alteromonas macleodii - Roseobacter denitrificans	1.910669	0.7847
Marinovum algicola - Roseobacter denitrificans	3.991939	0.0009 *
Pseudoalteromonas atlantica T6c - Roseobacter denitrificans	0.491663	1.0000
Alteromonas macleodii - Ruegeria sp. TM1040	1.051341	1.0000
Marinovum algicola - Ruegeria sp. TM1040	2.576895	0.1396
Pseudoalteromonas atlantica T6c - Ruegeria sp. TM1040	0.200720	1.0000
Roseobacter denitrificans - Ruegeria sp. TM1040	-0.245831	1.0000
Alteromonas macleodii - Vibrio alginolyticus	6.240144	0.0000 *
Marinovum algicola - Vibrio alginolyticus	8.472541	0.0000 *
Pseudoalteromonas atlantica T6c - Vibrio alginolyticus	2.354601	0.2596
Roseobacter denitrificans - Vibrio alginolyticus	3.032445	0.0340
Ruegeria sp. TM1040 - Vibrio alginolyticus	2.074310	0.5327
Alteromonas macleodii - Vibrio coralliilyticus	0.813569	1.0000
Marinovum algicola - Vibrio coralliilyticus	3.570097	0.0050 *
Pseudoalteromonas atlantica T6c - Vibrio coralliilyticus	-0.419979	1.0000
Roseobacter denitrificans - Vibrio coralliilyticus	-1.583473	1.0000

Ruegeria sp. TM1040 - Vibrio coralliilyticus	-0.694827	1.0000
Vibrio alginolyticus - Vibrio coralliilyticus	-9.276325	0.0000 *
Alteromonas macleodii - Vibrio harveyi	7.865687	0.0000 *
Marinovum algicola - Vibrio harveyi	9.897785	0.0000 *
Pseudoalteromonas atlantica T6c - Vibrio harveyi	3.193554	0.0197 *
Roseobacter denitrificans - Vibrio harveyi	4.452825	0.0001 *
Ruegeria sp. TM1040 - Vibrio harveyi	2.913175	0.0501
Vibrio alginolyticus - Vibrio harveyi	3.771093	0.0023 *
Vibrio coralliilyticus - Vibrio harveyi	12.11163	0.0000 *

Table 9: Cue system species pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii - Marinovum algicola	9.297378	0.0000 *
Alteromonas macleodii - Pseudoalteromonas atlantica T6c	0.756689	1.0000
Marinovum algicola - Pseudoalteromonas atlantica T6c	-5.416976	0.0000 *
Alteromonas macleodii - Roseobacter denitrificans	8.655145	0.0000 *
Marinovum algicola - Roseobacter denitrificans	-0.600753	1.0000
Pseudoalteromonas atlantica T6c - Roseobacter denitrificans	4.992179	0.0000 *
Alteromonas macleodii - Ruegeria sp. TM1040	6.311678	0.0000 *
Marinovum algicola - Ruegeria sp. TM1040	-0.038381	1.0000
Pseudoalteromonas atlantica T6c - Ruegeria sp. TM1040	4.391604	0.0002 *
Roseobacter denitrificans - Ruegeria sp. TM1040	0.386415	1.0000
Alteromonas macleodii - Vibrio alginolyticus	9.827016	0.0000 *
Marinovum algicola - Vibrio alginolyticus	-3.240714	0.0167 *
Pseudoalteromonas atlantica T6c - Vibrio alginolyticus	4.259071	0.0003 *
Roseobacter denitrificans - Vibrio alginolyticus	-2.422024	0.2161
Ruegeria sp. TM1040 - Vibrio alginolyticus	-1.873471	0.8540
Alteromonas macleodii - Vibrio coralliilyticus	4.527120	0.0001 *
Marinovum algicola - Vibrio coralliilyticus	-7.199616	0.0000 *
Pseudoalteromonas atlantica T6c - Vibrio coralliilyticus	1.647517	1.0000
Roseobacter denitrificans - Vibrio coralliilyticus	-6.424047	0.0000 *
Ruegeria sp. TM1040 - Vibrio coralliilyticus	-4.365935	0.0002 *
Vibrio alginolyticus - Vibrio coralliilyticus	-8.596422	0.0000 *
Alteromonas macleodii - Vibrio harveyi	8.523421	0.0000 *
Marinovum algicola - Vibrio harveyi	-4.395637	0.0002 *

Pseudoalteromonas atlantica T6c - Vibrio harveyi	3.579250	0.0048 *
Roseobacter denitrificans - Vibrio harveyi	-3.576216	0.0049 *
Ruegeria sp. TM1040 - Vibrio harveyi	-2.555208	0.1486
Vibrio alginolyticus - Vibrio harveyi	-3.064465	0.0305
Vibrio coralliilyticus - Vibrio harveyi	6.348708	0.0000 *

Table 10: Cus system species pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii CUKW	-1.830056	0.2017
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii KCC02	-1.830056	0.2017
Alteromonas macleodii CUKW - Alteromonas macleodii KCC02	0.000000	1.0000
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii str. 'Balearic Sea AD45'	0.000000	1.0000
Alteromonas macleodii CUKW - Alteromonas macleodii str. 'Balearic Sea AD45'	1.830056	0.2017
Alteromonas macleodii KCC02 - Alteromonas macleodii str. 'Balearic Sea AD45'	1.830056	0.2017

Table 11: Cop system Alteromonas macleodii strains pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii CUKW	-2.435406	0.0446
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii KCC02	-2.435406	0.0446
Alteromonas macleodii CUKW - Alteromonas macleodii KCC02	0.000000	1.0000
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii str. 'Balearic Sea AD45'	-1.458491	0.4341
Alteromonas macleodii CUKW - Alteromonas macleodii str. 'Balearic Sea AD45'	0.976914	0.9858

Alteromonas macleodii KCC02 - Alteromonas macleodii str. 'Balearic Sea AD45'	0.976914	0.9858

Table 12: Pco system Alteromonas macleodii strains pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii CUKW	-4.296449	0.0001*
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii KCC02	-4.296449	0.0001*
Alteromonas macleodii CUKW - Alteromonas macleodii KCC02	0.000000	1.0000
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii str. 'Balearic Sea AD45'	-2.190346	0.0855
Alteromonas macleodii CUKW - Alteromonas macleodii str. 'Balearic Sea AD45'	2.106102	0.1056
Alteromonas macleodii KCC02 - Alteromonas macleodii str. 'Balearic Sea AD45'	2.106102	0.1056

Table 13: Cue system Alteromonas macleodii strains pairwise comparisons

List of pairwise comparisons	**Z statistic**	**Adjusted p-value**
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii CUKW	-2.462061	0.0414
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii KCC02	-2.152622	0.0940
Alteromonas macleodii CUKW - Alteromonas macleodii KCC02	0.309439	1.0000
Alteromonas macleodii ATCC 27126 - Alteromonas macleodii str. 'Balearic Sea AD45'	-0.013453	1.0000
Alteromonas macleodii CUKW - Alteromonas macleodii str. 'Balearic Sea AD45'	2.448607	0.0430
Alteromonas macleodii KCC02 - Alteromonas macleodii str. 'Balearic Sea AD45'	2.139168	0.0973

Table 14: Cus system Alteromonas macleodii strains pairwise comparisons

8.3. Operons schemes

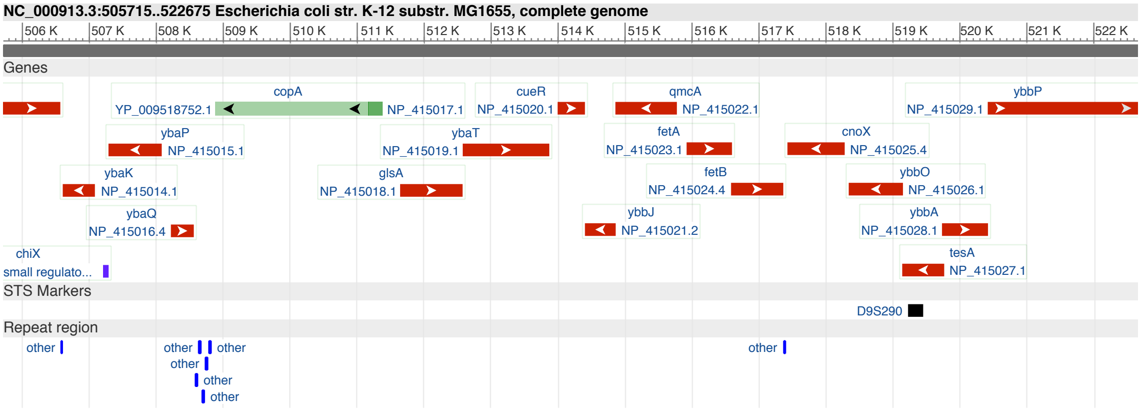


Figure 21: cueR genomic localization in *E. coli*

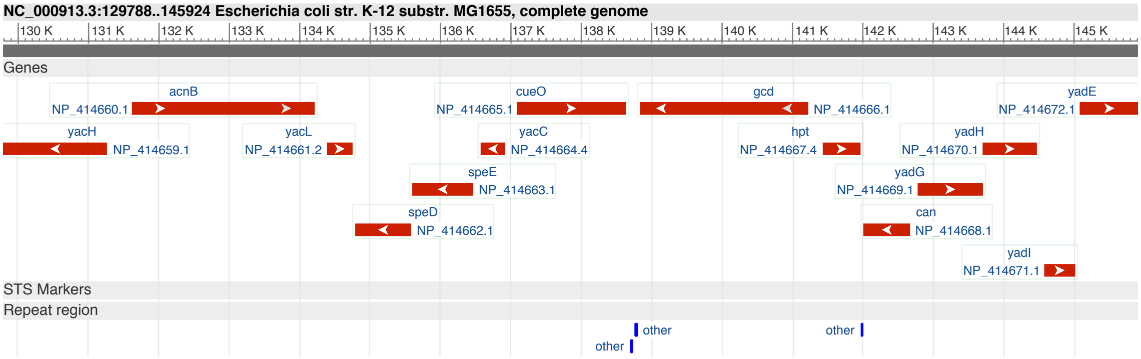


Figure 22: cueO genomic localization in *E. coli*

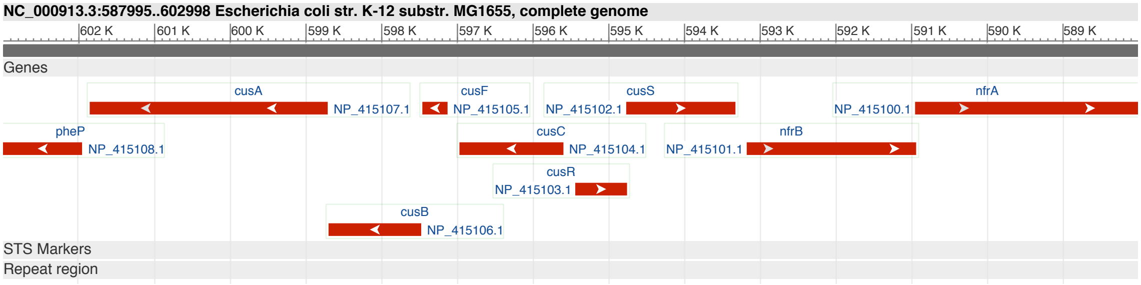


Figure 23: cus operon in *E. coli*

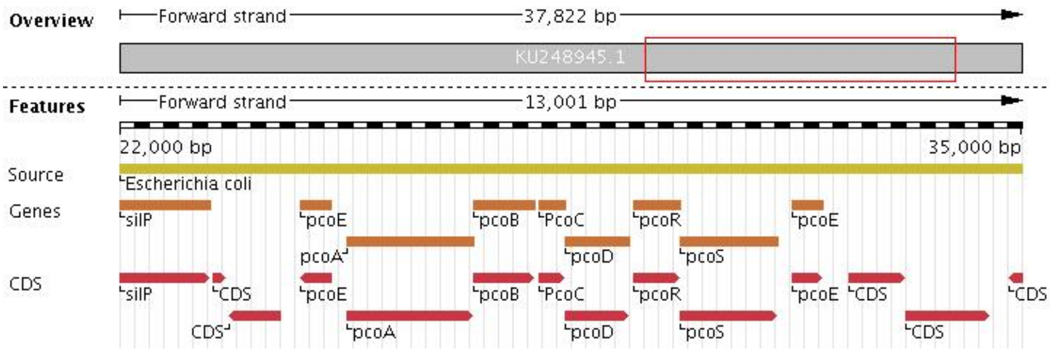


Figure 24: pco operon in *E. coli* (plasmid)

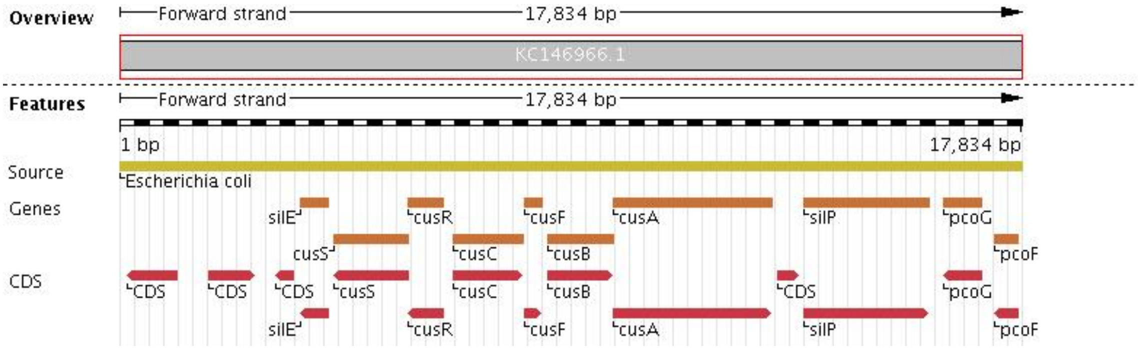


Figure 25: pcoG and pcoF in a cus operon in *E.coli* (plasmid)

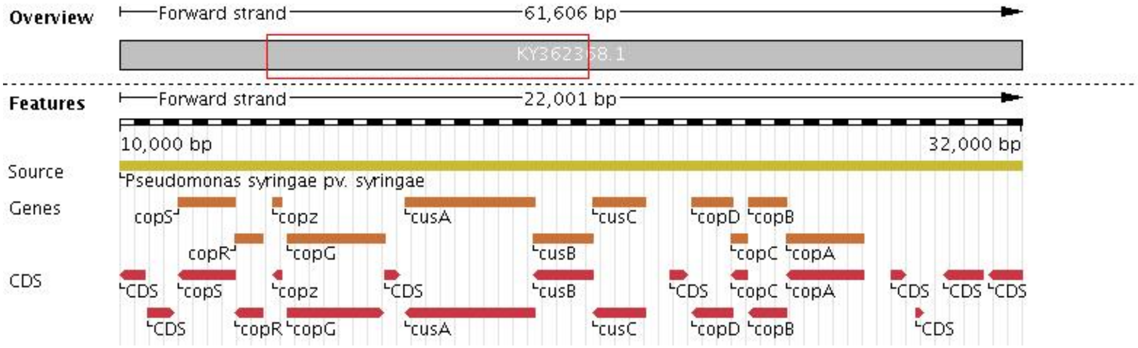
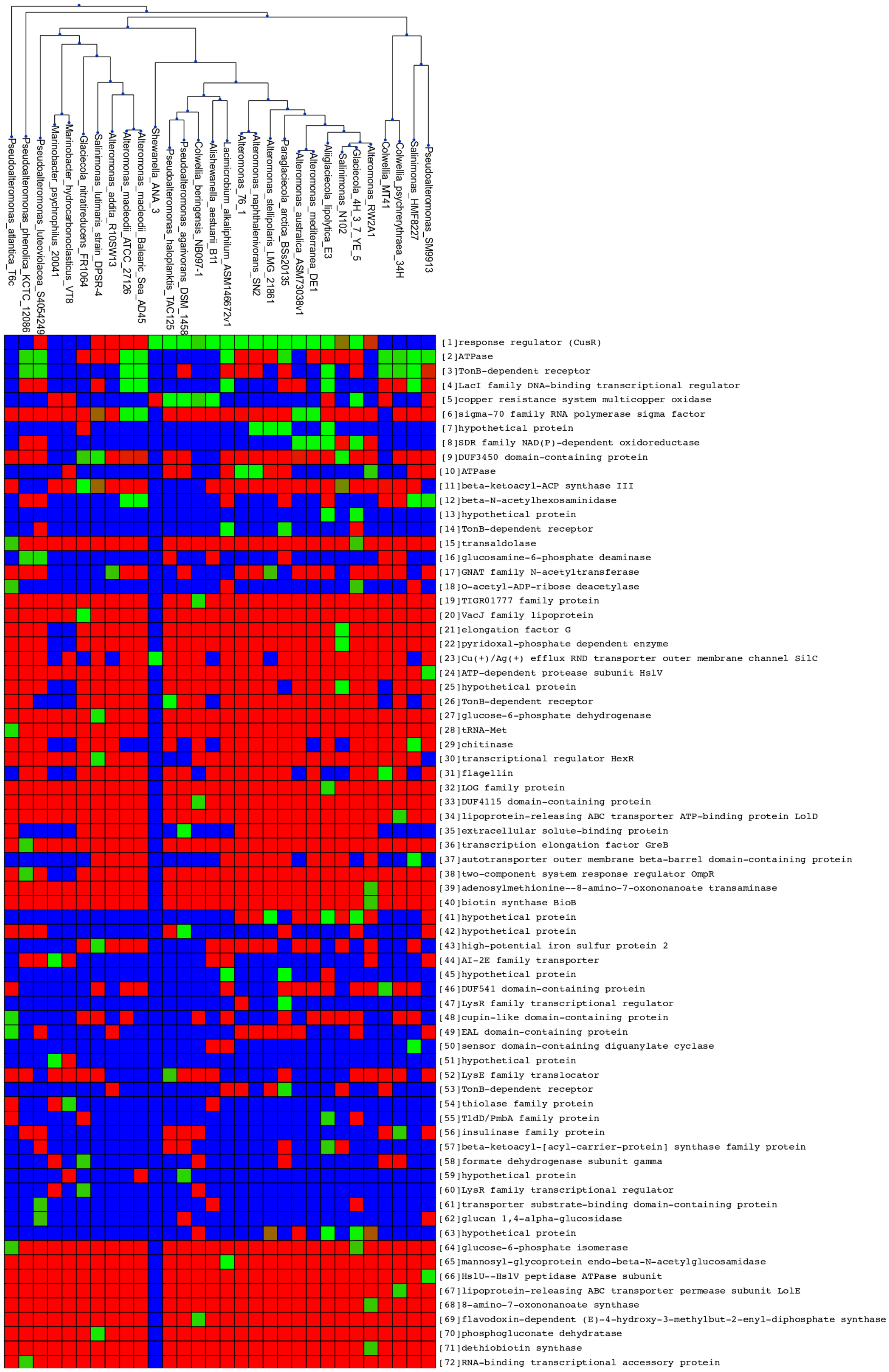


Figure 26: cop operon in *P. syringae* (plasmid)

8.4. Complete heatmap from second section of the results



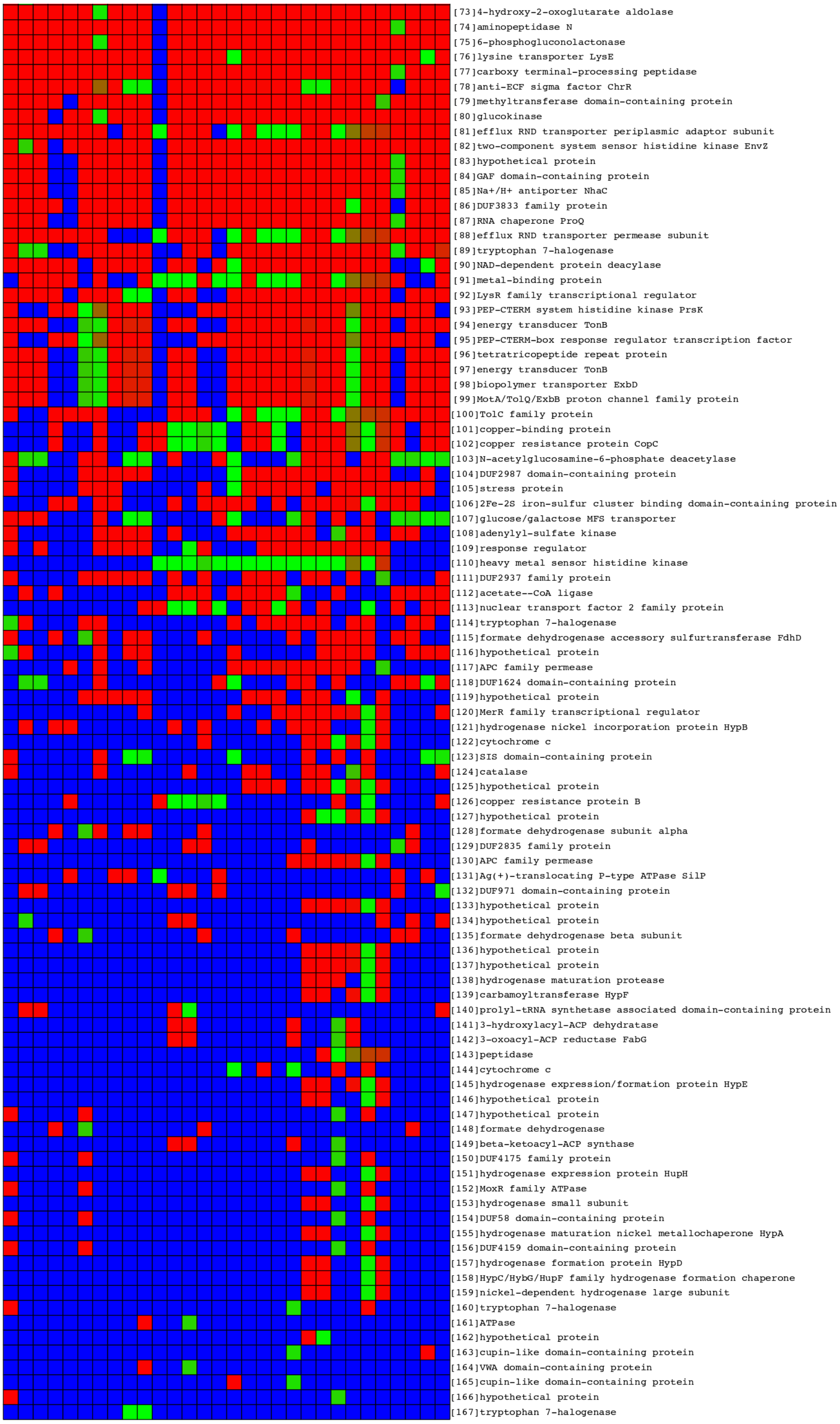


Figure 27: Heatmap of CGB results (complete)

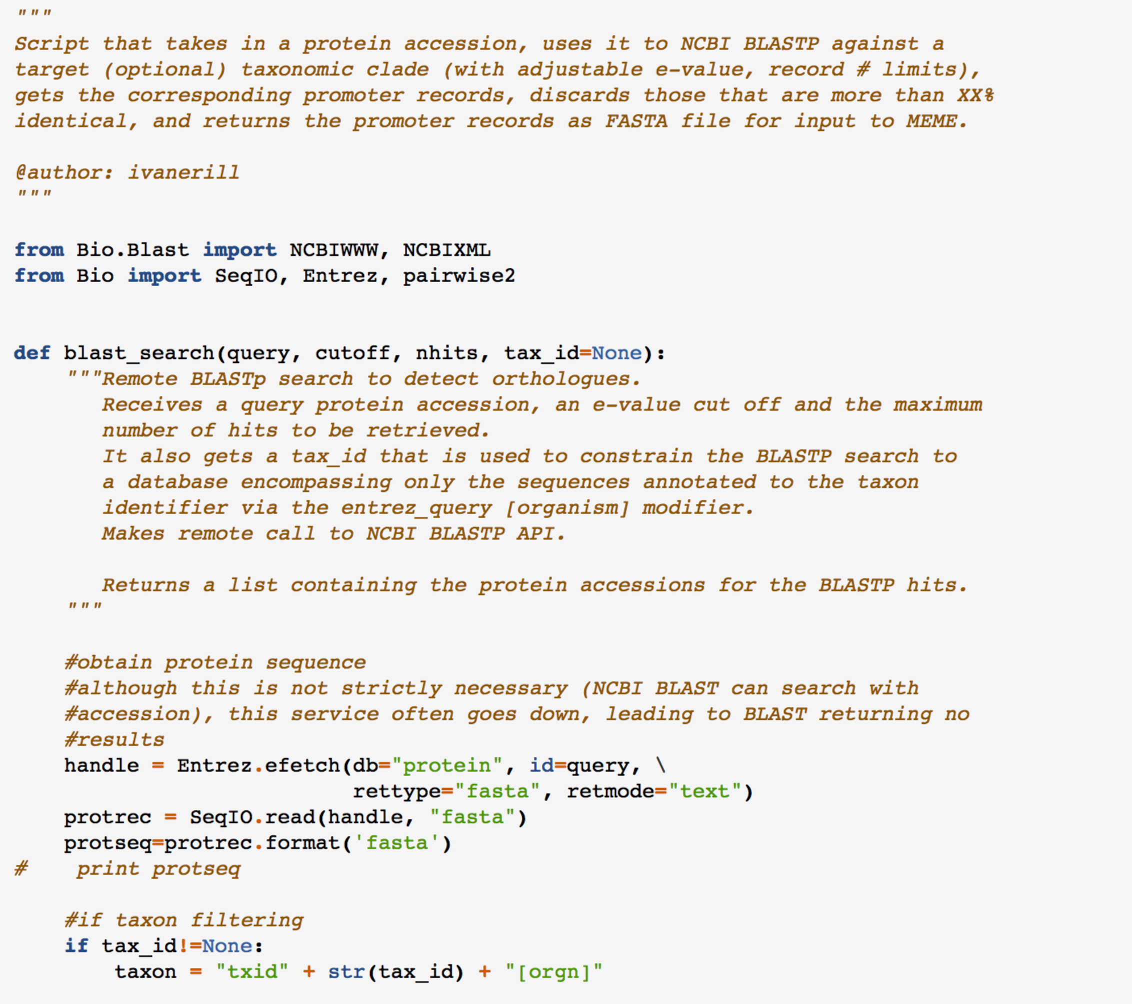
8.5. Code of script ortholog_search_genomic.py

```
""" Sript that reads two csv files, from first one selects the column
with protein accession numbers and from the second one the tax_ids. It
hen uses this information to tblastn against all specified taxids. e-
value and number of hits can be delimited. Gets the corresponding
orthologos with the following information: query protein accession,
taxid, accession of genome where the hit was found, description
related to that accession, start site, end site, query coverage and E-
value. """
from Bio.Blast import NCBIWWW, NCBIXML
from Bio import SeqIO, Entrez
import json
import csv
import time

def accession_list(csvfile):
    """Obtains protein accession numbers from input csv file and
    returns them as a list.
    """
    f=open(csvfile,"r")
    lines=f.readlines()
    protein_accession=[]
    for x in lines:
        protein_accession.append(x.split(',')[0])
    protein_accession=protein_accession[1:]

    return protein_accession

def taxons(taxcsvfile):
    """Obtains protein taxid numbers from input csv file and returns
    them as a list.
    """
    ft=open(taxcsvfile,"r")
    lines=ft.readlines()
    taxons=[]
    for x in lines:
        taxons.append((x.split(',')[1]).rstrip())
    taxons=taxons[1:]

    return taxons

def blast_search(query, Email, cutoff, nhits, tax_id=None):
    """Remote TBLASTN search to detect orthologues. Receives a query
    protein accession, an e-value cut off and the maximum number of
    hits to be retrieved. It also gets a tax_id that is used to
    constrain the TBLASTN search to a database encompassing only the
    sequences annotated to the taxon identifier via the entrez_query
    [organism] modifier. Makes remote call to NCBI TBLASTN API. Returns
    a list containing the protein accessions for the TBLASTN hits.
    """
    #obtain protein sequence
    #although this is not strictly necessary (NCBI BLAST can search
    #with accession), this service often goes down, leading to BLAST
    #returning no results
    Entrez.email = Email
    handle = Entrez.efetch(db="protein", id=query, \
                          rettype="fasta", retmode="text")

    protrec = SeqIO.read(handle, "fasta")
    protseq=protrec.format('fasta')

    #if taxon filtering
    if tax_id!=None:
    taxon = "txid" + str(tax_id) + "[orgn]"

        #perform TBLASTN search and parse results
        handleresults = NCBIWWW.qblast(program='tblastn',\
                                      database='refseq_genomic',\
                                      sequence=protseq, \
                                      entrez_query=taxon,\
                                      expect=cutoff, \
                                      hitlist_size=nhits)

    else:
        #perform TBLASTN search and parse results
        handleresults = NCBIWWW.qblast(program='tblastn',\
                                      database='refseq_genomic',\
                                      sequence=protseq, \
                                      expect=cutoff,\
                                      hitlist_size=nhits)

    blast_records = list(NCBIXML.parse(handleresults))

    return blast_records


def main_function(csvfile, targetscsvfile, json_outputfile,
                   csv_outputfile, Email, cutoff, nhits):

""" Takes the results from the TBLASTN and for each of the hits stores
the following information in a dictionary: accession, description,
start, end, E-value and coverage. This dictionary is returned as a
json fileself. A csv file is also written. This file contains for each
hit the accession of the query, tax_id, accession of the hit,
definition of the hit, start, end, E-value and coverage. """

POI = accession_list(csvfile)
taxids = taxons(targetscsvfile)

output = {}

with open(csv_outputfile, 'w') as myfile:
    wr = csv.writer(myfile)
    wr.writerow(['id', 'taxid','accession', 'description', 'start',
                  'end', 'e-value','coverage'])

    for id in POI:
    output[id] = []

        for taxid in taxids:
               blast_hits = blast_search(id, Email, cutoff,
                                             nhits, taxid)
                    a = {}
                    a[taxid] = {}
                    a[taxid]['accession'] = []
                    a[taxid]['description'] = []
                    a[taxid]['start'] = []
                    a[taxid]['end'] = []
                    a[taxid]['e-value'] = []
                    a[taxid]['coverage'] = []
                    for b in blast_hits:
                         for alignment in b.alignments:
                            a[taxid]['accession'].append(
                                   alignment.hit_id.split('|')[-2])
                            a[taxid]['description'].append(
                                   alignment.hit_def)
                            for hsp in alignment.hsps:
                                   a[taxid]['start'].append(
                                            hsp.sbjct_start)
                                   a[taxid]['end'].append(
                                            hsp.sbjct_end)
                                   a[taxid]['e-value'].append(
                                            hsp.expect)
                                   a[taxid]['coverage'].append(
                                          (float(hsp.align_length)/
                                          float(b.query_length))*100.)

                                   wr.writerow([id, taxid,
                                          alignment.hit_id.split(
                                                             '|')[-2],
                                          alignment.hit_def,
                                          hsp.sbjct_start,
                                          hsp.sbjct_end,
                                          hsp.expect,
                                          (float(hsp.align_length)/
                                          float(b.query_length))*
                                                              100.])

                    output[id].append(a)

                    time.sleep(3)

    json.dump(output, open(json_outputfile, "w"))

    return output

"""For test run"""

main_function("POI.csv",'targets_updated.csv','results_genomic.json',
'results_genomic.csv','ane.iturbide@gmail.com', 10E-20, 20)
```

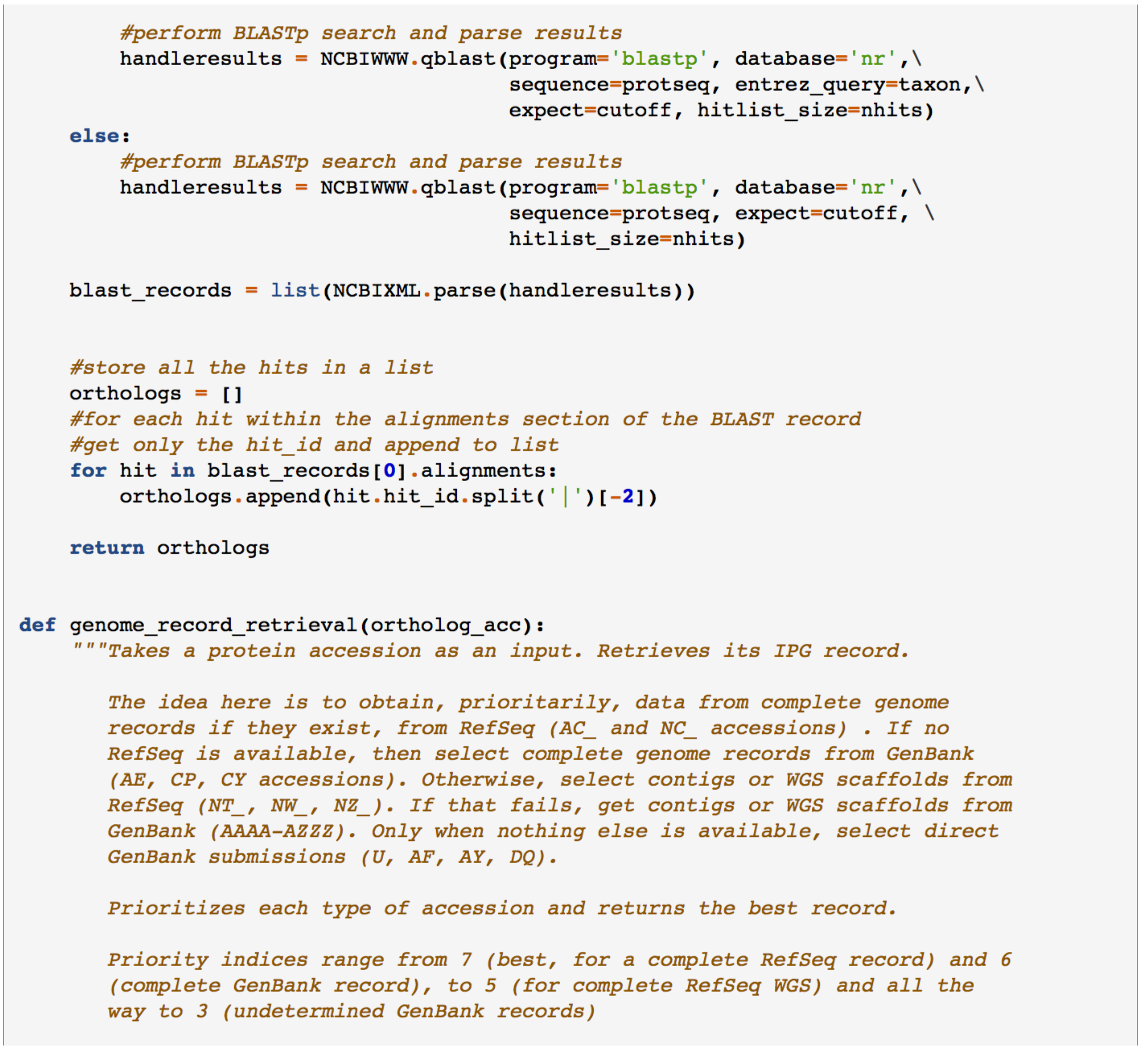
8.6. Code of script biosample_id.py

```
from Bio import SeqIO, Entrez
import csv
import pandas as pd

def hit_species(resultscsv):

    """
    Takes the results of the previous script and stores the
    accession of the hit genome in a list. Returns de list.
    """

    ft=open(resultscsv,"r")
    lines=ft.readlines()
    species=[]

    for x in lines:
        species.append((x.split(',')[9]).rstrip())
    species=species[1:]
    return species

def biosample(resultscsv, Email, outputcsv):

    """
    Takes the list from the hit_species function and does a search of
    the corresponding BioSample id in nucleotide databse. The data is
    then stored in a csv and returned.
    """

    species = hit_species(resultscsv)
    Entrez.email = Email
    with open(outputcsv, 'w') as myfile:
        wr = csv.writer(myfile)
        wr.writerow(['Species','BioSample_id'])

        for spec in species:
             handle = Entrez.efetch(db="nucleotide", id=spec,
                        rettype="gb", retmode="text")
             record = SeqIO.read(handle, "genbank")
             biosample = record.dbxrefs[0].split(':')[1]
             wr.writerow([spec, biosample])

"""Test run"""

biosample('results_tblastn_species_2.csv', 'ane.iturbide@gmail.com',
'biosamples_2.csv')
```

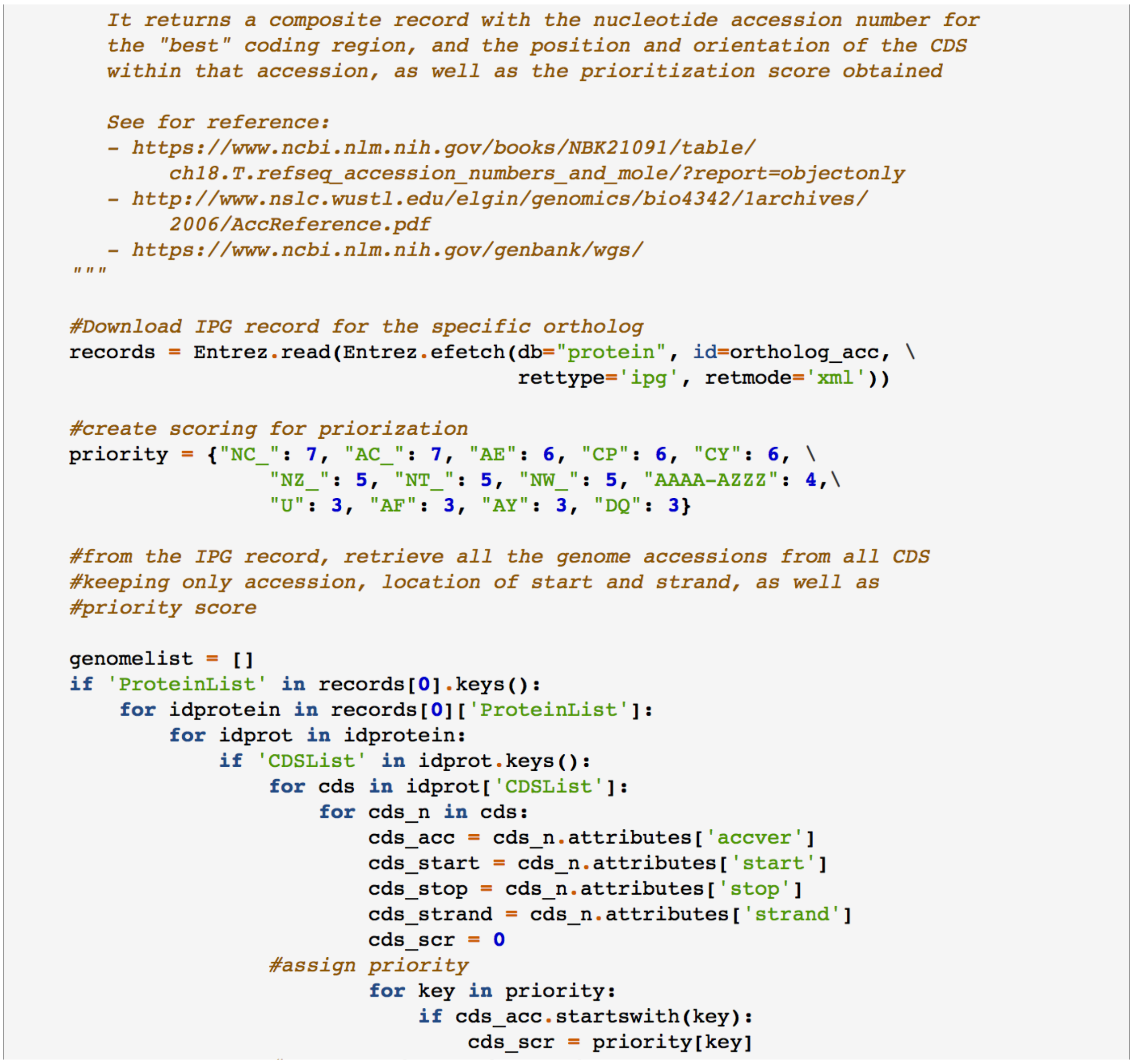
8.7. Code of script prot2proms_ai.py



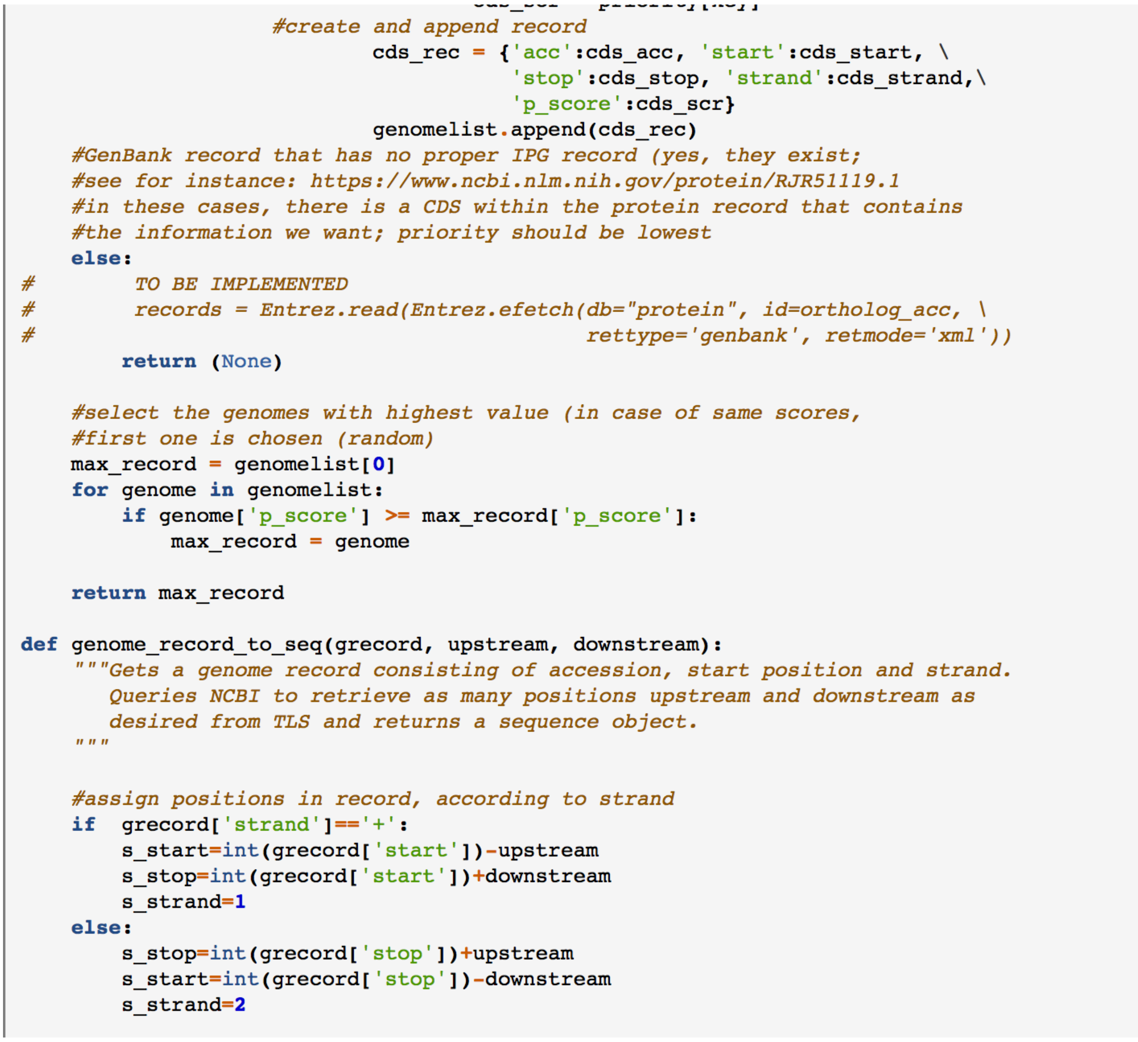
```
"""
Script that takes in a protein accession, uses it to NCBI BLASTP against a
target (optional) taxonomic clade (with adjustable e-value, record # limits),
gets the corresponding promoter records, discards those that are more than XX%
identical, and returns the promoter records as FASTA file for input to MEME.

@author: ivanerill
"""

from Bio.Blast import NCBIWWW, NCBIXML
from Bio import SeqIO, Entrez, pairwise2


def blast_search(query, cutoff, nhits, tax_id=None):
    """Remote BLASTp search to detect orthologues.
       Receives a query protein accession, an e-value cut off and the maximum
       number of hits to be retrieved.
       It also gets a tax_id that is used to constrain the BLASTP search to
       a database encompassing only the sequences annotated to the taxon
       identifier via the entrez_query [organism] modifier.
       Makes remote call to NCBI BLASTP API.

       Returns a list containing the protein accessions for the BLASTP hits.
    """

    #obtain protein sequence
    #although this is not strictly necessary (NCBI BLAST can search with
    #accession), this service often goes down, leading to BLAST returning no
    #results
    handle = Entrez.efetch(db="protein", id=query, \
                           rettype="fasta", retmode="text")
    protrec = SeqIO.read(handle, "fasta")
    protseq=protrec.format('fasta')
#    print protseq

    #if taxon filtering
    if tax_id!=None:
        taxon = "txid" + str(tax_id) + "[orgn]"
```



```python
        #perform BLASTp search and parse results
        handleresults = NCBIWWW.qblast(program='blastp', database='nr',\
                                       sequence=protseq, entrez_query=taxon,\
                                       expect=cutoff, hitlist_size=nhits)
    else:
        #perform BLASTp search and parse results
        handleresults = NCBIWWW.qblast(program='blastp', database='nr',\
                                       sequence=protseq, expect=cutoff, \
                                       hitlist_size=nhits)

    blast_records = list(NCBIXML.parse(handleresults))


    #store all the hits in a list
    orthologs = []
    #for each hit within the alignments section of the BLAST record
    #get only the hit_id and append to list
    for hit in blast_records[0].alignments:
        orthologs.append(hit.hit_id.split('|')[-2])

    return orthologs


def genome_record_retrieval(ortholog_acc):
    """Takes a protein accession as an input. Retrieves its IPG record.

       The idea here is to obtain, prioritarily, data from complete genome
       records if they exist, from RefSeq (AC_ and NC_ accessions) . If no
       RefSeq is available, then select complete genome records from GenBank
       (AE, CP, CY accessions). Otherwise, select contigs or WGS scaffolds from
       RefSeq (NT_, NW_, NZ_). If that fails, get contigs or WGS scaffolds from
       GenBank (AAAA-AZZZ). Only when nothing else is available, select direct
       GenBank submissions (U, AF, AY, DQ).

       Prioritizes each type of accession and returns the best record.

       Priority indices range from 7 (best, for a complete RefSeq record) and 6
       (complete GenBank record), to 5 (for complete RefSeq WGS) and all the
       way to 3 (undetermined GenBank records)
```



```
       It returns a composite record with the nucleotide accession number for
       the "best" coding region, and the position and orientation of the CDS
       within that accession, as well as the prioritization score obtained

       See for reference:
       - https://www.ncbi.nlm.nih.gov/books/NBK21091/table/
            ch18.T.refseq_accession_numbers_and_mole/?report=objectonly
       - http://www.nslc.wustl.edu/elgin/genomics/bio4342/1archives/
            2006/AccReference.pdf
       - https://www.ncbi.nlm.nih.gov/genbank/wgs/
    """

    #Download IPG record for the specific ortholog
    records = Entrez.read(Entrez.efetch(db="protein", id=ortholog_acc, \
                                        rettype='ipg', retmode='xml'))

    #create scoring for priorization
    priority = {"NC_": 7, "AC_": 7, "AE": 6, "CP": 6, "CY": 6, \
                "NZ_": 5, "NT_": 5, "NW_": 5, "AAAA-AZZZ": 4,\
                "U": 3, "AF": 3, "AY": 3, "DQ": 3}

    #from the IPG record, retrieve all the genome accessions from all CDS
    #keeping only accession, location of start and strand, as well as
    #priority score

    genomelist = []
    if 'ProteinList' in records[0].keys():
        for idprotein in records[0]['ProteinList']:
            for idprot in idprotein:
                if 'CDSList' in idprot.keys():
                    for cds in idprot['CDSList']:
                        for cds_n in cds:
                            cds_acc = cds_n.attributes['accver']
                            cds_start = cds_n.attributes['start']
                            cds_stop = cds_n.attributes['stop']
                            cds_strand = cds_n.attributes['strand']
                            cds_scr = 0
                    #assign priority
                            for key in priority:
                                if cds_acc.startswith(key):
                                    cds_scr = priority[key]
```



```
                                    cds_scr = priority[key]
                    #create and append record
                            cds_rec = {'acc':cds_acc, 'start':cds_start, \
                                       'stop':cds_stop, 'strand':cds_strand,\
                                       'p_score':cds_scr}
                            genomelist.append(cds_rec)
    #GenBank record that has no proper IPG record (yes, they exist;
    #see for instance: https://www.ncbi.nlm.nih.gov/protein/RJR51119.1
    #in these cases, there is a CDS within the protein record that contains
    #the information we want; priority should be lowest
    else:
#        TO BE IMPLEMENTED
#        records = Entrez.read(Entrez.efetch(db="protein", id=ortholog_acc, \
#                                            rettype='genbank', retmode='xml'))
        return (None)

    #select the genomes with highest value (in case of same scores,
    #first one is chosen (random)
    max_record = genomelist[0]
    for genome in genomelist:
        if genome['p_score'] >= max_record['p_score']:
            max_record = genome

    return max_record

def genome_record_to_seq(grecord, upstream, downstream):
    """Gets a genome record consisting of accession, start position and strand.
       Queries NCBI to retrieve as many positions upstream and downstream as
       desired from TLS and returns a sequence object.
    """

    #assign positions in record, according to strand
    if  grecord['strand']=='+':
        s_start=int(grecord['start'])-upstream
        s_stop=int(grecord['start'])+downstream
        s_strand=1
    else:
        s_stop=int(grecord['stop'])+upstream
        s_start=int(grecord['stop'])-downstream
        s_strand=2
    #Download FASTA record containing the desired upstream sequence
    net_handle = Entrez.efetch(db="nuccore",id=grecord['acc'], \
                               strand=s_strand, seq_start=s_start, \
                               seq_stop=s_stop, rettype='fasta',\
                               retmode="txt")
    gnome_record=SeqIO.read(net_handle, "fasta")

    return(gnome_record)

def id_below_maxid_perc(el1, el2, max_percent_id):
    """Aligns two sequence elements and determines whether they are
       more than %ID identical (false) or not (true)
       Scoring: Match:+2, Mismatch:-1, GapO: -2, GapE: -0.2
    """

#    print "Seq1 length: " + str(len(el1.seq)) + ' ' + el1.id
#    print "Seq2 length: " + str(len(el2.seq)) + ' ' + el2.id
    al=pairwise2.align.globalms(el1.seq, el2.seq, 2, 0, -2, -.5,\
                                one_alignment_only=True, \
                                penalize_end_gaps=False)

    #print al
    matches=0
    gapless=0
    #for each position in the alignment
    for ch_pair in zip(al[0][0],al[0][1]):
        #if this is a non-gapped position
        if '-' not in ch_pair:
            #if it's a match, count it
            if ch_pair[0]==ch_pair[1]:
                matches=matches+1
            gapless=gapless+1

    perID = float(matches)/float(gapless)

#    print "Matches: ", matches
#    print "Gapless: ", gapless
#    print "%ID: ", perID

    #return true or false depending on percent identity
    if perID*100<=float(max_percent_id):
        return(True)
    else:
        return(False)

def identity_filter_list(us_list, percent_id):
    """Gets a list of upstream sequence records and a max percent id.
       Goes through the list, removing any records with more than %ID.
       Returns the trimmed list.
    """

    filt_list=[]
    cnt=0
    while cnt < len(us_list):
        #get next first element in upstream seq list, removing it from list
        current_element=us_list.pop(0)
        #and adding it to the filtered list
        filt_list.append(current_element)

        #check against all remaining elements in up. seq list; remove them from
        #up. seq the list if they are not below threshold of identity
        #at each pass revised up. seq list hence contains only elements less
        #than %ID  identical to previously processed elements now stored in
        #filt_list
        us_list[:]=[upel for upel in us_list if \
                    id_below_maxid_perc(upel,current_element, percent_id)]

    return(filt_list)

def retrieve_orth_ups(query_acc, Eemail, outname='output.fas', Be_val=10E-10, \
                Bmax_res=50, BtaxID=None,\
                up_region=200, dw_region=25, maxID=85):
    """Note: for optional parameters, use param=value format in function call
       Makes us of library functions to:
           - take a protein accession
           - identify BLASTP hits
           - get to the (best) nucleotide records mapping to those BLASTP hits
           - get the sequence upstream of those CDS

       Takes in:
           - Query_acc: the protein accession that serves as BLASTP query
           - Entrez email
           - Output file name (output.fas by default)
           - BLAStP limit based on e-value (default: 10E-10)
           - BLASTP max results (default: 50)
           - BLASTP tax ID, used to limit BLAST DB scope (default: none)
       -- This is a tax ID from NCBI, numerals only
       - Upstream span: number of nucleotides grabbed upstream of ATG
         (default 200)
       - Downstream span: number of nucleotides grabbed downstream of ATG
         (default 25)
       - Percent ID: maximum identity allowed for filtering, so that we
         only retrieve somewhat dissimilar results for motif discovery
         (default: 80). Use 100 for no filtering.
"""

print 'Running with parameters as follows:'
print 'Protein query ID: ', query_acc
print 'Entrez email: ', Eemail
print 'Output file name: ', outname
print 'BLASTP e-value cap: ', Be_val
print 'BLASTP max results: ', Bmax_res
print 'BLASTP tax ID filter: ', BtaxID
print 'Upstream size to grab: ', up_region
print 'Downstream size to grab: ', dw_region
print 'Max percent ID to weed out: ', maxID

Entrez.email = Eemail

#if query is not a list, make it so
if not isinstance(query_acc, (list,)):
    query_acc=[query_acc]

firstquery=True
for query in query_acc:
    #BLAST
    print 'Performing BLASTP search for', query
    ot_list=blast_search(query,Be_val,Bmax_res,BtaxID)
    print 'Retrieved ' + str(len(ot_list)) + ' BLASTP hits'

    #create list of upstream sequences
    upel_list=[]
    #for every BLAST protein hit
    for prothitacc in ot_list:
        print '|-> Obtaining best nucleotide record for: ' + prothitacc
        #get the best nucleotide record from IPG list
        nucrec=genome_record_retrieval(prothitacc)
        if nucrec!=None:
            print '   |-> Grabbing upstream nucleotide sequence for: ' \
                  + prothitacc
            #grab the sequence corresponding to that record
            seq=genome_record_to_seq(nucrec, up_region, dw_region)
            upel_list.append(seq)

    #output data
    with open(outname+'.prefilter',"w" if firstquery else 'a') as out_handle:
        SeqIO.write(upel_list,out_handle,'fasta')

    print 'Filtering out list of results based on sequence similarity'
    #filter the list of upstream sequences
    filt_list=identity_filter_list(upel_list, maxID)
    print 'Number of upstream sequences left: ' + str(len(filt_list))

    print 'Writting' + ' output to file: ' + outname if firstquery \
            else 'Appending' + ' output to file: ' + outname
    #output data
    with open(outname,"w" if firstquery else 'a') as out_handle:
        SeqIO.write(filt_list,out_handle,'fasta')



    firstquery=False
```