

Universidade do Minho

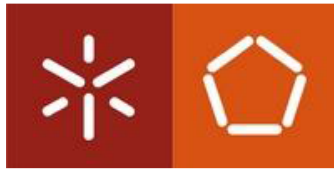
Mestrado em Bioinformática

Ano Letivo 2014/2015

Docentes: Prof. Miguel Rocha

Prof. Isabel Rocha

Prof. Óscar Dias



Análise de parte do genoma da *Neisseria gonorrhoeae*

Algoritmos para Análise de Sequências Biológicas/
Laboratórios de Bioinformática

Grupo 1:

Daniel Oliveira (PG27667)

Jorge Reis (PG26544)

Raquel Silva (PG27668)

Introdução Teórica

A *Neisseria gonorrhoeae* é uma bactéria Gram-negativa que se agrupa na forma diplococos, sendo o agente causador da gonorreia, infecção sexualmente transmitida (STI) e das mais antigas registadas no Homem [1]. De acordo com estudos recentes é a segunda maior causa de STI's de origem bacteriana, causando entre 60 a 100 milhões de novos casos por ano [1-4].

O não tratamento da infecção pode levar a infertilidade na mulher, uretrite no homem e ao aumento do risco de transmissão e aquisição do vírus da imunodeficiência humana (HIV) [1, 5].

O diagnóstico de infecção é feito, geralmente, através de uma cultura bacteriana ou ensaios de amplificação de ácidos nucleicos (NAAT). Estes testes são realizados em ambientes controlados de laboratório, com resultados geralmente disponíveis em poucos dias. Na maioria dos casos os médicos pedem então que o paciente compareça para o tratamento ser iniciado. Em clínicas de zonas urbanas a maioria dos pacientes com diagnóstico de STI's recebe tratamento atempadamente [6].

A história mostra que a *N. gonorrhoeae*, ao longo dos últimos 70 – 80 anos, foi tratada com vários tipos de fármacos diferentes que eram usados no tratamento principal na infecção. Contudo, devido ao tratamento ineficiente, a aquisição de genes de resistência e a propagação de clones de grupos isolados de linhagens resistentes, levou a problemas mais desafiantes no tratamento desta infecção e a diminuição rápida dos fármacos que se podem usar para tratar a infecção [7, 8]. Na figura 1 é possível observar alguns dos fármacos usados para tratar a infecção e a altura em que apareceram estirpes resistentes aos mesmos.

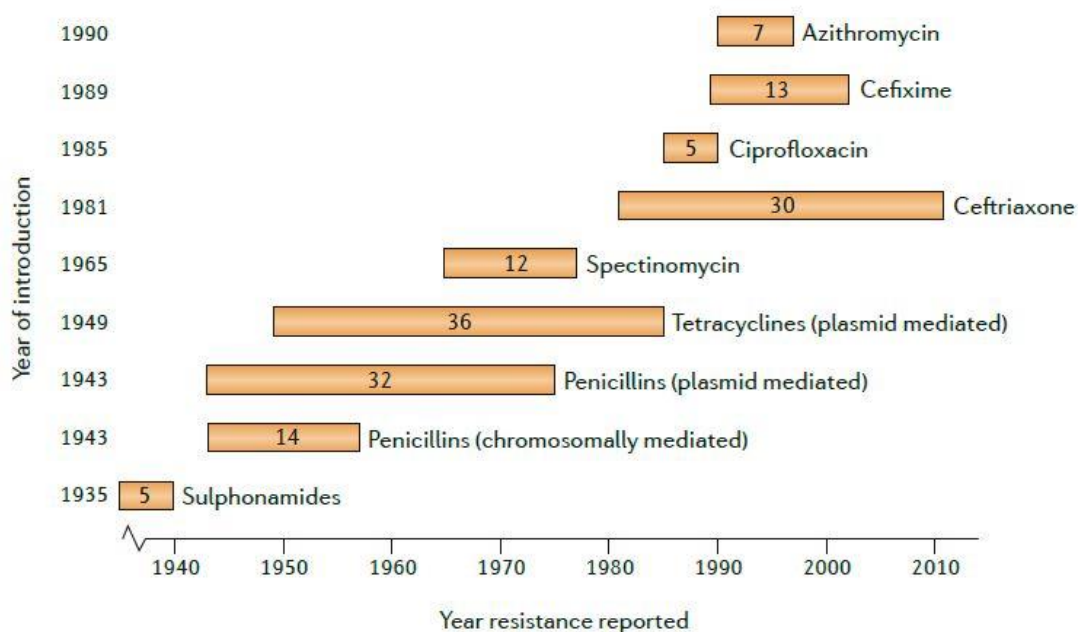


Figura 1 - Aquisição de resistências farmacológicas de *Neisseria gonorrhoeae* ao longo do tempo [9].

Existem relatos que se revelam alarmantes tendo em conta que as cefalosporinas de espectro alargado de terceira geração (ESC), ceftriaxona (injetável) e cefixima (oral) são o último tipo de antibiótico comum ainda eficiente para o tratamento da gonorreia. Contudo, a eficiência deste fármaco tem sofrido um declínio devido ao aumento de resistências a este antibiótico pela bactéria em causa [10]. Este antibiótico pertence à classe dos antibióticos β -lactâmicos, sendo que, no caso do gene *penA* que codifica a proteína PBP2, esta proteína é o alvo dos antibióticos β -lactâmicos. Existem outros genes determinantes para a *N. gonorrhoeae* adquirir resistência além do gene *penA*, como por exemplo o gene *mtrR* e *penB* [11]. Na figura 2 é apresentado um esquema ilustrativo que relaciona os antibióticos aos respetivos genes de resistência.

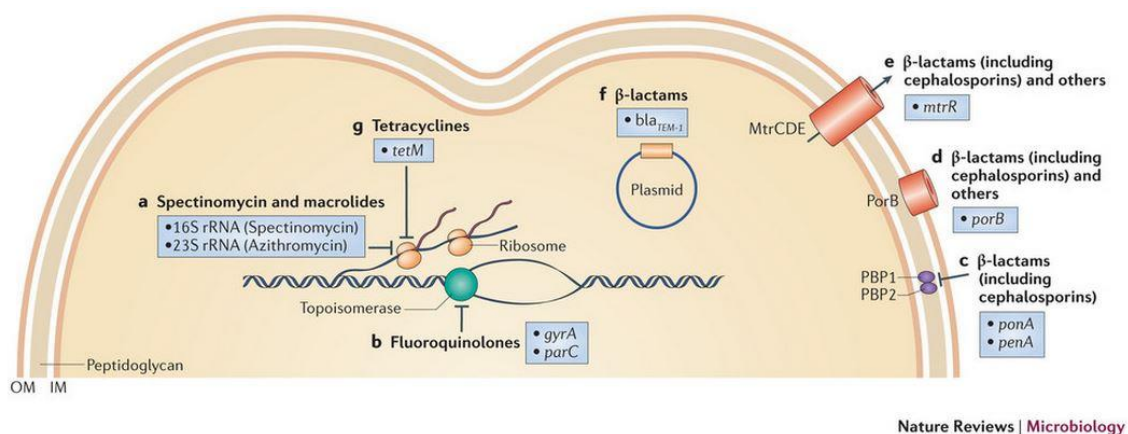


Figura 2 - Antibióticos que têm como alvo *Neisseria gonorrhoeae* e os respectivos genes de resistência [9].

Também o facto da *N. gonorrhoeae* ser uma bactéria gram-negativa indica uma maior preocupação pois, neste momento, as bactérias gram-negativas são as mais preocupantes em relação à aquisição de resistência a antibióticos. No caso da *N. gonorrhoeae*, além de algumas estirpes já serem resistentes às ESC's, também são resistentes a penicilinas, tetraciclina, fluoroquinolonas, estreptomicina, cefalosporinas de espectro estreito, combinações de sulfonamidas com trimetoprim (co-trimoxazol), macrolídeos e azitromicina [8, 12].

Uma informação precisa e atualizada da resistência antimicrobiana (AMR) da *N. gonorrhoeae* é essencial para a formulação com sucesso de uma política de controlo da gonorreia. As atividades de vigilância da doença têm que ser otimizadas dependendo do país e população alvo, mas existe uma falta de dados que permita a capacidade de controlo da AMR, tanto em países desenvolvidos como em países em desenvolvimento [13].

Existem fármacos que estão a ser desenvolvidos mas, apesar de serem necessários novos fármacos, ainda nenhum está numa fase final de testes. Entre eles estão pleuromutilinas, inibidores da topoisomerase bacteriana, e bombas de efluxo, inibidores de FavI e LpxC. Estes compostos podem potencialmente vir a ser explorados no futuro para melhorar o tratamento da gonorreia resistente a múltiplos fármacos (MDR) [14].

Os fatores que promovem um desenvolvimento da AMR incluem um uso indiscriminado de antibióticos e uma transmissão continuada da gonorreia. O controlo

da infecção poderá ser feito de melhor maneira por campanhas de saúde para melhorar o acesso a tratamentos de saúde, promoção de sexo seguro e regulação do uso de antibióticos, tanto no caso dos sistemas de saúde como fora deles. Este é o momento oportuno para abordar estas questões enquanto ainda existe a possibilidade de controlar a AMR [9].

Desenvolvimento

No início do trabalho estávamos a realizar um programa direccionado para um utilizador, em que este podia escolher a sua zona do genoma ou o ficheiro que pretendia utilizar. Mais tarde, reparamos que não era necessário um programa, mas apenas funções que nos permitissem retirar a informação requerida.

Encontramos vários impasses quando pretendíamos obter a informação de forma automática, sendo que o que nos tomou mais tempo foi a obtenção dos *Uniprot Accession Numbers* (AC) e a informação associada a cada uma das proteínas, tal como a realização de *blasts* e obtenção dos *hits*. Após uma procura mais aprofundada de forma a resolver o nosso primeiro impasse vimos alguns *packages*, mas estes apenas poderiam ser aplicados na versão 2 do *Python*, sendo que utilizamos *Python 3.4*. Portanto, tivemos que recorrer a outros métodos, nomeadamente aceder à informação do *site* da *Uniprot* utilizando o *proteinID* como *query* e colocando os *uniprotAC* numa lista para futura utilização, graças ao *urllib*.

Na elaboração de código para fazer *blast* tivemos problemas em arranjar um método de tornar o nosso código autónomo porque tínhamos cerca de 6528 proteínas para analisar o que era muito difícil recolher essa informação sem recorrer a um algoritmo que fizesse isso por ele. Tivemos também problemas de como organizar a nossa informação por isso recorremos a pastas com o nome das nossas proteínas para colocar toda a informação delas.

A maior parte da informação, sendo o *locus tag* do gene, o nome do gene, a sua localização e a *strand*, o *geneID* e o GI, os *EC numbers*, os *proteinID*, os produtos/nomes da proteína e a sua anotação, foi obtida através dos módulos e classes do BioPython nomeadamente, Entrez, UniprotIO e SeqIO, esta última permite-nos ver as *features* do ficheiro *genbank* (*gb*). A primeira é utilizada para obter o ficheiro *gb* correspondente à *Neisseria gonorrhoeae* FA 1090, Versão: NC_002946.2 GI:59800473 na zona [0:246000] os primeiros 246000 nucleótidos correspondentes a 203 genes.

Por meio do *site* da UniProt fizemos download de um ficheiro de texto e um ficheiro *xml* em que ambos continham informação de cada gene/proteína. Como apresentavam todos os genes do organismo, tivemos que criar uma função para apenas obter os resultados dos nossos genes/proteínas correspondentes à nossa zona do genoma.

Através do ficheiro de texto obtivemos os *UniprotID*, o grau de revisão, o número de aminoácidos, a localização celular, as reacções e o *pathway*. Através do ficheiro *xml* obtivemos as identificações do KEGG *orthology*, KEGG *pathway*, BioCyc AC e GeneOntology (GO) *number*.

Com as matrizes de resultados, tivemos que recorrer às bibliotecas *numpy* e *pandas*, de forma a criar um *data frame* com a informação e posterior criação de um ficheiro *csv* para conseguirmos fazer um ficheiro *excel* com as colunas de informação.

No caso da lista de termos do GO, também recorremos à informação apresentada no *site* da *ebi* no *QuickGO* através da *urllib*. Os domínios conservados foram obtidos através da informação de cada gene no *ncbi*.

Mesmo com estes processos automáticos tivemos que colmatar a nossa tabela *excel* manualmente recorrendo às bases de dados fornecidas no enunciado entre outras. Nomeadamente, utilizamos a base de dados da PSORT para completar a informação relativamente à localização celular, usamos o KEGG para procurar os identificadores do KEGG *pathway* (*ngo*) e o nome da via. Com os termos do GO obtidos, conseguimos inferir a classificação de algumas funções das proteínas, tal como a utilização dos EC *numbers* que nos permitiu conseguir os identificadores do KEGG *reactions*.

Alinhamento múltiplo e filogenia

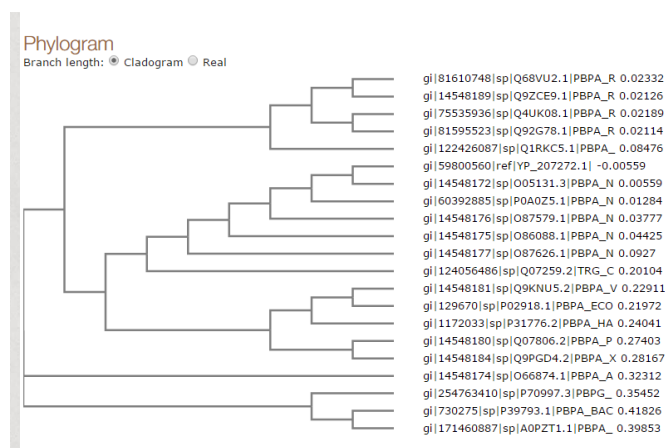


Figura 3 – Cladograma do alinhamento múltiplo da *penicillin-binding protein*

Fizemos alinhamento múltiplo com os melhores hits da *penicillin-binding protein* do nosso organismo porque é uma proteína que apresenta resistência à penicilina, por isso usamos esta ferramenta para ver a conservação desta com outros organismos.

Utilizamos a ferramenta Clustal Omega do website <http://www.ebi.ac.uk/>

Através da análise do cladograma e da matriz de identidade, podemos verificar que existem 4 proteínas muito próximas à nossa proteína *Penicillin-binding protein 1A*, estas proteínas pertencem aos organismos *Neisseria meningitidis* Z2491, *Neisseria lactamica*, *Neisseria cinerea*, *Neisseria flavescens*, com percentagem de identidades de 98.57, 93.10, 92.38, 83.77, respetivamente

	1:	gi 81610748 sp Q68VU2.1 PBPA_R	100.00	95.54	89.68	88.66	80.00	34.63	33.03	33.03	33.16	32.24	32.76	40.26	32.63	32.99	32.85	32.94	33.38	31.94
29.03	2:	gi 14548189 sp Q9ZCE9.1 PBPA_R	95.54	100.00	90.60	89.45	81.07	35.61	33.33	33.33	33.46	32.55	33.20	39.83	32.28	33.16	33.03	32.59	33.02	31.99
28.48	3:	gi 75535936 sp Q4UK08.1 PBPA_R	89.68	90.60	100.00	95.70	84.68	35.61	33.46	33.46	33.73	32.81	33.07	39.83	33.07	33.81	33.29	32.86	33.69	32.41
28.81	4:	gi 81595523 sp Q92G78.1 PBPA_R	88.66	89.45	95.70	100.00	84.18	36.10	34.12	34.12	33.20	33.73	40.69	32.41	33.68	33.16	32.86	33.69	32.41	
28.96	5:	gi 122426087 sp Q1RKC5.1 PBPA_N	80.00	81.07	84.68	84.18	100.00	37.32	34.33	34.46	34.60	34.20	34.60	41.13	31.88	33.16	31.68	34.29	34.08	32.69
29.22	6:	gi 59800560 ref YP_207272.1	34.63	35.61	35.61	36.10	37.32	100.00	100.00	98.57	93.10	92.38	83.77	53.68	39.33	40.38	37.94	40.15	34.62	35.55
33.44	7:	gi 14548172 sp O05131.3 PBPA_N	33.03	33.33	33.46	34.12	34.33	100.00	100.00	97.37	92.23	91.73	81.68	53.68	38.25	40.00	36.84	43.47	38.99	34.58
30.64	8:	gi 60392885 sp P0A0Z5.1 PBPA_N	33.03	33.33	33.46	34.12	34.46	98.57	97.37	100.00	92.86	91.98	81.93	53.68	38.25	39.87	37.24	43.08	39.12	34.44
30.64	9:	gi 14548176 sp O87579.1 PBPA_N	33.16	33.46	33.73	34.12	34.60	93.10	92.23	92.86	100.00	90.73	80.68	53.25	39.03	39.61	37.50	43.47	39.65	34.72
31.24	10:	gi 14548175 sp O86088.1 PBPA_N	32.24	32.55	32.81	33.20	34.20	92.38	91.73	91.98	90.73	100.00	82.31	52.81	38.38	39.74	37.76	42.56	39.25	34.58
30.49	11:	gi 14548177 sp O87626.1 PBPA_N	32.76	33.20	33.07	33.73	34.60	83.77	81.68	81.93	80.68	82.31	100.00	51.95	38.82	39.92	38.29	41.57	38.50	33.38
30.45	12:	gi 124056486 sp Q07259.2 TRG_C	40.26	39.83	39.83	40.69	41.13	53.68	53.68	53.68	53.25	52.81	51.95	100.00	42.17	45.65	44.29	41.67	42.42	40.97
38.96	13:	gi 14548181 sp Q9KNU5.2 PBPA_V	32.63	32.28	33.07	32.41	31.88	39.33	38.25	38.25	39.03	38.38	38.82	42.17	100.00	55.12	52.07	43.61	39.35	31.58
30.61	14:	gi 129670 sp P02918.1 PBPA_ECO	32.99	33.16	33.81	33.68	33.16	40.38	40.00	39.87	39.61	39.74	39.92	45.65	55.12	100.00	53.15	42.70	39.87	34.81
30.29	15:	gi 1172033 sp P31776.2 PBPA_HA	32.85	33.03	33.29	33.16	31.68	37.94	36.84	37.24	37.50	37.76	38.29	44.29	52.07	53.15	100.00	40.48	39.06	33.71
28.44	16:	gi 14548180 sp Q07806.2 PBPA_P	32.94	32.59	32.86	32.86	34.29	40.15	43.47	43.08	43.47	42.56	41.57	41.67	43.61	42.70	40.48	100.00	44.43	33.10
30.59	17:	gi 14548184 sp Q9PGD4.2 PBPA_X	33.38	33.02	33.69	33.69	34.08	34.62	38.99	39.12	39.65	39.25	38.50	42.42	39.35	39.87	39.06	44.43	100.00	33.15
28.84	18:	gi 14548174 sp O66874.1 PBPA_A	31.94	31.99	32.41	32.41	32.69	35.55	34.58	34.44	34.72	34.58	33.38	40.97	31.58	34.81	33.71	33.10	33.15	100.00
31.30	19:	gi 254763410 sp P70997.3 PBPA_BAC	29.03	28.48	28.81	28.96	29.22	33.44	30.64	30.64	31.24	30.49	30.45	38.96	30.61	30.29	28.44	30.59	28.84	31.30
100.00	20:	gi 730275 sp P39793.1 PBPA_BAC	18.46	19.05	18.73	18.47	18.70	27.21	18.57	18.32	18.19	18.70	18.19	38.26	18.88	19.36	18.96	17.95	19.69	20.55
19.28	21:	gi 171460887 sp A0PZT1.1 PBPA_N	20.14	19.67	19.86	19.72	19.70	24.57	22.39	21.84	21.98	22.12	21.70	31.17	23.08	22.70	22.89	20.83	21.64	27.14
22.47	22:	gi 171460887 sp A0PZT1.1 PBPA_N	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32	18.32

Figura 4 – Matriz identidade do alinhamento múltiplo da *penicillin-binding protein*

Implementação em Python

Relativamente aos ficheiros de código, o ficheiro “*Main_Program.py*” foi separado em vários ficheiros consoante a sua utilidade, mas só o ficheiro principal é que foi utilizado para obter as informações, ou seja, os outros ficheiros de código são para visualização das funções usadas na obtenção de informação específica, para uma melhor procura por parte de outros utilizadores que pretendam utilizar as funções. O ficheiro “*Uniprot_Parser.py*” foi obtido externamente isto é, devido a dificuldades acima descritas para obter informação do UniProt, tivemos que recorrer a funções provenientes de outros programadores na mesma situação, sendo que nenhum grupo de trabalho apresentava soluções para este problema.

As funções criadas para a obtenção das nossas informações serão explicadas brevemente a seguir.

Primeiramente, temos a função **create_file** que utiliza a função **get_genome_zone** caso não apresentemos ficheiro *gb* da nossa zona do genoma. Esta última irá buscar a informação ao *ncbi*, irá criar o ficheiro *gb* sendo movido (*src (source)*) para outra pasta (*res (resource)*) para posterior utilização, e irá ler o ficheiro (*record*).

O nosso *record* é usado para obter todas as *features* e respectivos *qualifiers* nelas contidas para cada um dos nossos genes. As *features* encontradas na nossa zona foram “gene”, “CDS” e “tRNA”, os *qualifiers* foram “locus_tag”, “protein_id”, “note”, “product”, “db_xref”, “pseudogene”, “gene”, “EC_number” e “translation”. Alguns genes podiam conter ou não estes *qualifiers* sendo que o “db_xref” para o “CDS” normalmente apresentava-se duas vezes, para o *GI* e para o *geneID*. Cada “CDS” apresentava a localização e a *strand*.

A função **locus_tag** irá colocar numa lista todos os locus_tag (locus) dos nossos genes, sendo utilizada para muitas outras funções de forma a obter apenas a informação dos nossos genes.

As seguintes funções irão utilizar a nossa lista locus e irão a cada “CDS” *feature* e compara o *qualifier* “locus_tag” com os nossos locus, tirando a informação apenas para este gene.

A função **protein_ID** retorna o *qualifier* “protein_id” para cada gene, caso tenha um, senão retorna “Não tem”.

A função **location** retorna a localização e a *strand* de cada gene.

A função **note** retorna o *qualifier* "note" de cada gene caso este tenha alguma, senão retorna “Não tem”.

A função **product** retorna o *qualifier* "product" de cada gene caso este tenha algum, senão retorna “Não contem produtos”.

A função **gene_ID_GI** retorna os dois *qualifiers* "db_xref", nomeadamente o GI e o *geneID*.

A função **gene_names** retorna o *qualifier* "gene" caso tenha nome do gene, senão retorna “Não tem”.

A função **EC_number** retorna o *qualifier* "EC_number" caso tenha o EC *number* do gene, senão retorna “Não contem”.

Estas funções acima irão ser utilizadas na função **info** de forma a criar uma matriz (lista de listas no caso do Python) em que cada linha contém a informação retornada para cada gene, isto é, cada linha contém informação para apenas um gene.

As seguintes funções serão utilizadas para obter a informação dos pseudogenes e dos tRNA.

A função **pseudogenes** irá buscar todos os locus dos pseudogenes e colocar numa lista para posterior utilização na obtenção das *features*.

A função **location_pseudo** vai comparar os locus dos pseudogenes para obter apenas a localização e a *strand* destes.

A função **pseudogeneID** retorna apenas os *geneID* dos pseudogenes.

As funções acima serão utilizadas na função **info_pseudogenes** para obter uma matriz com a informação para cada gene.

A função **tRNA** irá buscar todos os locus da *feature* tRNA e colocar numa lista para posterior utilização na obtenção das *features*.

A função **gene_ID_tRNA** retorna apenas os *geneID* dos tRNA.

A função **product_tRNA** retorna os produtos do tRNA.

A função **location_tRNA** vai comparar os locus dos tRNA para obter apenas a localização e a *strand* destes.

Estas funções mencionadas acima são utilizadas na função **info_tRNA** em que retorna uma matriz com a informação de cada locus de tRNA.

A função **tabela** usa a matriz retornada em **info** para criar um *array* de forma a colocar num *data frame* e criar um ficheiro *csv* com as informações obtidas na matriz.

A função **tabela_pseudogenes** usa a matriz retornada em **info_pseudogenes** para criar um *array* de forma a colocar num *data frame* e criar um ficheiro *csv* com as informações obtidas na matriz.

A função **tabela_tRNA** usa a matriz retornada em **info** para criar um *array* de forma a colocar num *data frame* e criar um ficheiro *csv* com as informações obtidas na matriz.

A função **without_note** retorna uma lista com os "protein_id" de cada gene que não contenham nota.

A função **hypoth_proteins** retorna uma lista com os "protein_id" de cada gene que tenham como produto proteínas hipotéticas.

A função **GI_number** retorna uma lista com GI *numbers* de todos os nossos genes.

A função **CDD** irá aceder ao *ncbi* com cada valor de GI, irá ler a informação obtida e às *features* de como a obter o *qualifier* "db_xref" que contém os valores de CDD, posteriormente irá coloca-los numa lista.

A função **tabela_CDD** usa a matriz retornada em **CDD** para criar um *array* de forma a colocar num *data frame* e criar um ficheiro *csv* com as informações obtidas na matriz.

A função **list_genes_names** dá-nos uma lista com todos os nomes dos genes caso haja.

A função **uniprot_ID** obtém os UniprotAC através do acesso por *url* do *site* da UniProt, utilizando o *urllib*, colocando-os numa lista.

Esta lista será usada como termo de comparação, de modo a conseguir a informação dada pelos ficheiros *xml* e *txt* de todas as proteínas do nosso organismo, nas seguintes funções, **info_uniprot** e **more_info_uniprot** em que se conseguem obter as características mencionadas no Desenvolvimento.

As funções **tabela_uniprot** e **tabela_uniprot2** usam as matrizes retornadas nas duas funções acima para criar *arrays* de forma a colocar em *data frame* e criar os ficheiros *csv* com as informações obtidas nas matrizes.

A função **tab** e **sorting** são utilizadas na **tabela_uniprot2**, a primeira é necessária pois as matrizes precisam de ter as mesmas linhas e colunas de informação adicionadas para conseguir separar por colunas de informação, a segunda função é usada para organizar por ordem de UniprotAC.

Para tirar a lista de termos do KEGG GO, foi necessário aceder a todos os GO *numbers* de cada UniprotAC dados na matriz ordenada pelo **sorting** retornada pela **tabela_uniprot2**. Após a obtenção destes números, tivemos que aceder novamente através da *urllib* de forma a retirar do *site QuickGO* os termos associados a cada GO e retornar uma lista.

Para procurar possíveis funções de proteínas que não tinham notas no NCBI e também confirmar se as funções das proteínas que tinham notas estavam corretos implementamos um código com ajuda de bibliotecas para fazer *blast* de todas as proteínas da nossa parte do genoma da *N.Gonorrhoeae*. Para isso optamos primeiro por dividir as proteínas em dois grupos, proteínas com notas e sem notas no NCBI e criamos os respetivos scripts *py*.

Os dois scripts criados têm praticamente as mesmas funções apenas muda o tipo de proteínas que fazem o *blast* e as diretorias onde guardam e vão buscar os ficheiros.

Com estas funções quisemos no final criar um ficheiro *txt* com o GI da nossa proteína e dentro todas as funções dos *hits* do *blast*.

Para isso recorreremos a algumas bibliotecas:

- SeqIO para ler ficheiros *gb*
- shutil, para mover os ficheiros criados para outra diretoria;
- os.path, para trabalhar com diretorias;
- urllib, para ir à Uniprot recolher informação das nossas possíveis proteínas homólogas;

```
7
8 from Bio import SeqIO #reading gb file
9 from Bio.Blast import NCBIWWW,NCBIXML #fetching/parsing blast
10 import shutil#moving files
11 import os.path#checking files in path
12 import urllib #getting info from site
13 from Uniprot_Parser import * #parsing uniprot text file
```

Figura 5 - Bibliotecas importadas para fazer *blast*;

Na primeira parte do código fomos buscar os GI's das proteínas que vamos fazer *blast*.

```
15 #GI numbers from genes with note
16 def GInumbers(record,locus_tag):
17     GI=[]
18     for i in range(len(record.features)):
19         my_cds = record.features[i]
20         if my_cds.type == "CDS":
21             for j in range(len(locus_tag)):
22                 if my_cds.qualifiers["locus_tag"][0]==str(locus_tag[j]):
23                     if "note" in my_cds.qualifiers:
24                         if "db_xref" in my_cds.qualifiers:
25                             x=my_cds.qualifiers["db_xref"]
26                             GI.append(x[0])
27     return GI

18 #get gi from protein without note
19 def giwithout_note(record):
20     ID=[]
21     for i in range(len(record.features)):
22         my_cds = record.features[i]
23         if my_cds.type == "CDS":
24             if "note" not in my_cds.qualifiers:
25                 x=my_cds.qualifiers["db_xref"]
26                 GI=x[0]
27                 ID.append(gi[3:])
28     return ID
```

```
Python console
159800477, '159800483', '159800485', '159800487', '159800488',
'159800489', '159800490', '159800491', '159800492', '159800494', '159800495',
'159800497', '159800499', '159800500', '159800501', '159800502', '159800504',
'159800511', '159800513', '159800517', '159800521', '159800522', '159800523',
'159800524', '159800525', '159800528', '159800529', '159800530', '159800536',
'159800537', '159800538', '159800539', '159800540', '159800541', '159800542',
'159800545', '159800548', '159800549', '159800550', '159800551', '159800552',
'159800555', '159800556', '159800557', '159800559', '159800562', '159800563',
'159800564', '159800565', '159800570', '159800571', '159800572', '159800574',
'159800579', '159800580', '159800581', '159800582', '159800583', '159800584',
'159800585', '159800589', '159800590', '159800591', '159800594',
'159800595', '159800596', '159800597', '159800598', '159800599', '159800600',
'159800601', '159800602', '159800603', '159800607', '159800609',
'159800610', '159800611', '159800612', '159800615', '159800616', '159800617',
'159800619', '159800620', '159800621', '159800622', '159800625', '159800626',
'159800631', '159800632', '159800633', '159800634', '159800636', '159800637',
'159800641', '159800642', '159800643', '159800647', '159800648', '159800649',
'159800651', '159800654', '159800655', '159800656', '159800657', '159800658',
'159800659', '159800661', '159800667', '159800668', '159800669', '159800673',
'159800675', '159800679', '159800680', '159800681', '159800683', '159800685',
'159800686', '159800688', '159800689', '159800690']
```

Figura 6 -Função que dá os GI numbers com notas no NCBI e o seu resultado na consola.

Com a lista criada pela função anterior efetuamos *blasts* para todas proteínas. Nesta parte recorreremos primeiramente a uma procura *non redundant blast* contudo pelo fato de haver muitos *hits* que não correspondiam ao que estávamos à procura passamos a usar a *swissprot* pois tinha resultados mais credíveis.

Devido ao fato de este processo ser um pouco demorado preferimos guardar a informação de cada *blast* em ficheiros *xml* para no futuro podemos recorrer a essa informação.

Com os ficheiros anteriormente criados fizemos uma análise aos mesmos filtrando-os resultados como aceitáveis ou não. Se o *hit* tivesse um *E-value* inferior a 0.05 considerávamos esse resultado como um possível homólogo e guardávamos *hits* em um ficheiro “matches.txt” ou “nomatches.txt” consoante tivéssemos *hits* ou não. Quanto mais *e-value* for próximo de zero mais significante é o resultado, contudo temos de

olhar sempre para o tamanho da sequência porque as mais pequenas tem alta probabilidade de ocorrer numa grande.

Assim conseguimos separar as proteínas que podem ter homologas e as que não tem.

```
29
30 #blast gi with note
31 def blastwithnote():
32     locus=locus_tag(record)
33     gi=GInumbers(record,locus)
34     for i in range(len(gi)):
35         gis=gi[i]
36         GI_num=gi[3:]
37         result_handle = NCBIWWW.qblast("blastp","swissprot", GI_num)
38         save_file = open(GI_num+'.xml','w')
39         save_file.write(result_handle.read())
40         save_file.close()
41         result_handle.close()
42         path=os.getcwd()
43         #moving the file to another directory
44         src = path+"/"+GI_num+'.xml' #source folder
45         dst = "../res/blast_with_note/" #destination folder
46         shutil.move(src, dst)
47
48 #blast results and analyse the hits, if e-value > 0.05 we don't consider this result
49 def blastanaliserwithnote():
50     blast=[]
51     for file in os.listdir("../res/blast_with_note"):
52         if file.endswith(".xml"):
53             blast.append(file)
54     E_VALUE_THRESH = 0.05
55     lista=[]
56     for i in range(len(blast)):
57         lista.append(blast[i])
58         lista[i].append(blast[i])
59         result_handle = open("../res/blast_with_note/"+blast[i])
60         blast_record = NCBIWWW.read(result_handle)
61         for alignment in blast_record.alignments:
62             for hsp in alignment.hsps:
63                 if hsp.expect < E_VALUE_THRESH:
64                     lista[i].append(alignment.title)
65                     lista[i].append(alignment.length)
66                     lista[i].append(hsp.expect)
67         save_file = open('nomatches.txt','w')
68         for i in range(len(lista)):
69             if len(lista[i])<2:
70                 save_file.write(str(lista[i])+"\n")
71         save_file.close()
72         #moving the file to another directory
73         path=os.getcwd()
74         src = path+"/"+nomatches.txt #source folder
75         dst = "../res/blast_with_note/nomatch/" #destination folder
76         shutil.move(src, dst)
77
78         save_file = open('matches.txt','w')
79         for i in range(len(lista)):
80             if len(lista[i])>2:
81                 save_file.write(str(lista[i])+"\n")
82         save_file.close()
83         #moving the file to another directory
84         path=os.getcwd()
85         src = path+"/"+matches.txt #source folder
86         dst = "../res/blast_with_note/match/" #destination folder
87         shutil.move(src, dst)
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2
```

Com recurso à leitura do ficheiro anterior fomos à Uniprot recolher toda a informação das possíveis proteínas homólogas. Para organizar esses ficheiros criamos pastas com o GI da nossa proteína para alojar os ficheiros de possíveis proteínas homólogas.

No total tivemos 6528 proteínas que podem ser homólogas das nossas.

```
125 #read allhits.txt, go to uniprot by hits and save txt by protein hit
126 def uniprotallhitswithnote():
127     handle = open("../res/blast_with_note/match/allhits/allhits.txt").readlines()
128
129
130
131     for n in range(len(handle)):
132         x=handle[n].split()
133         for k in range(len(x)-1,len(x)):
134             e=x[k]
135             limpo=e.replace("]", "")
136             gi=(limpo[1:])
137             #to organize results we create directory with gi of our proteins and save all txt from protein in the
138             if not os.path.exists("../res/blast_with_note/match/function/teste/"+gi):
139                 os.makedirs("../res/blast_with_note/match/function/teste/"+gi)
140             for j in range(len(x)-1):
141                 m=x[j]
142                 q=m.replace("[", "")
143                 protein=q[1:]
144                 site = urllib.request.urlopen("http://www.uniprot.org/uniprot/"+protein+".txt")
145                 data = site.readlines()
146                 file = open("../res/blast_with_note/match/function/teste/"+protein+".txt", "wb") #open file in binary
147                 file.writelines(data)
148                 file.close()
149             try:
150                 src = "../res/blast_with_note/match/function/teste/"+protein+'.txt' #source folder
151                 dst = "../res/blast_with_note/match/function/teste/"+gi #destination folder
152                 shutil.move(src, dst)
153             except:
154                 pass
155
```

Figura 9 - Programa que cria pastas com o nome do GI das nossas proteínas e guarda a informação dessas proteínas, recolhida da Uniprot , na respetiva pasta.

Para concluir fomos a cada pasta e pegamos apenas na informação que pretendíamos, a função das proteínas, posto isto criamos um ficheiro com as funções de todas as possíveis proteínas homólogas com o GI da nossa proteína. Esse ficheiro serviu para criar duas tabelas de Excel uma para as proteínas com notas e a outra para as proteínas sem notas. Para a criação das tabelas utilizamos o script “exportexcel.py”.

Estas tabelas foram utilizadas para completar a tabela principal e foram criadas para ser mais fácil trabalhar com a informação que conseguimos com os *blasts*.

	GI	Blast hit (Uniprot ID)	Description
1	59800483	O9KMA5	Antitoxin component of a toxin-antitoxin (TA) module that counteracts the effect of the HigB-2 toxin. Binds to its own promoter and regulates transcription of the higB-2/higA-2 operon. [ECO:0000269] PubMed:17020579]. INDUCTION: Induced by amino acid starvation and the protein synthesis inhibitor chloramphenicol. [ECO:0000269] PubMed:17020579]. MISCELLANEOUS: HigB-2/HigA-2 has been shown to stabilize plasmids very efficiently in E.coli. SIMILARITY: Contains 1 HTH cro/C1-type DNA-binding domain. [ECO:0000305]. Copyrighted by the UniProt Consortium, EMBL; AE003853; AAF96373.1; -; Genomic_DNA. PIR; E82455; E82455. RefSeq; NP_232861.1; NC_002506.1. ProteinModelPortal; O9KMA5; -. STRING; 243277.VCA0469; -. DNASU; 2611844; -. EnsemblBacteria; AAF96373; AAF96373; VC_A0469. GeneID; 2611844; -. KEGG; vch:VCA0469; -. PATRIC; 20085488; VBI/VibChoi83274_3095. eggNOG; COG2944; -. KO; K07726; -. OMA; NWEQGRA; -. OrthoDB; EOG6VMTSG; -. BioCyc; VCHO:VCA0469-MONOMER; -. Proteomes; UP000000584; Chromosome 2. GO; GO:0043565; F:sequence-specific DNA binding; IEA:InterPro. GO; GO:0006355; P:regulation of transcription, DNA-templated; IEA:UniProtKB-KW. GO; GO:0006351; P:transcription, DNA-templated; IEA:UniProtKB-KW. Gene3D; 1.10.260.40; -. 1. InterPro; IPR001387; Cro/C1-type_HTH. InterPro; IPR010982; Lambda_DNA-bd_dom. SMART; SM00530; HTH_XRE; 1. SUPFAM; SSF47413; SSF47413; 1. PE 2: Evidence at transcript level; KW Complete proteome; DNA-binding; Reference proteome; Transcription; KW Transcription regulation. FT CHAIN 1 104 Antitoxin iGA-2. FT /FTid=PRO_0000278767. FT DOMAIN 45 98 HTH cro/C1-type. FT DNA_BIND 56 75 H-T-H motif. [ECO:0000250]. SQ SEQUENCE 104 AA; 11696 MW; 966
2	59800485	O32233	Involved in protein export. Participates in an early event of protein translocation (By similarity). [ECO:0000250]. SUBCELLULAR LOCATION: Cell membrane [ECO:0000250]; Multi-pass membrane protein [ECO:0000250]. SIMILARITY: Belongs to the SecG family. [ECO:0000305]. Copyrighted by the UniProt Consortium,
3	59800485	O66505	Subunit of the protein translocation channel SecYEG. SUBUNIT: Component of the Sec protein translocase complex. Heterotrimer consisting of SecY, SecE and SecG subunits. The heterotrimers can form oligomers, although 1 heterotrimer is thought to be able to translocate proteins. Interacts with SecDF, and other proteins may be involved. The channel interacts with SecA via subunit SecY. [ECO:0000269] PubMed:18923516]. SUBCELLULAR LOCATION: Cell membrane [ECO:0000250]; Multi-pass membrane protein [ECO:0000250]. SIMILARITY: Belongs to the SecG family. [ECO:0000305]. Copyrighted by the UniProt Consortium,
4			Subunit of the protein translocation channel SecYEG. Overexpression of some hybrid proteins has been thought to jam the protein secretion apparatus resulting

Figura 10 -Tabela criada com a informação do Blast para as proteínas sem anotações.

Para realizar uma anotação nas proteínas que não tinham informação, foram analisados os resultados do *blast*, em comparação com outras informações retiradas das restantes bases de dados pesquisadas. Regra geral os dados possibilitavam uma análise simplificada aos resultados do *blast*, sendo assim é possível retirar uma função provável para as proteínas sem notas. Contudo existiram proteínas que não tendo dados de outras bases de dados, e sendo os resultados do *blast* pouco homogêneos, a atribuição de uma função à proteína foi consideravelmente mais problemática.

Em relação as proteínas com anotações recorremos ao *blast* apenas para confirmar as funções das proteínas para verificar se as anotações estavam corretas.

Discussão de Resultados

A tabela em ficheiro *excel* denominada “**Final_Table**” contém toda a informação recolhida durante a realização deste trabalho. Esta tabela tem 26 colunas e 215 linhas, cada linha corresponde a informações de genes/proteínas, cada coluna é uma informação obtida para os genes/proteínas.

As colunas correspondem às informações do locus, gene name, location/strand, GI number/GeneID, EC number, proteinID, Uniprot_ID, Revision, Protein length, Subcellular Location, KEGG Orthlogy, KEGG Pathway, Pathway, Product (protein name), Note, Note from blast, Classification of the protein function, BioCyc AC, Catalytic Activity, KEGG Reaction, GeneOntology Terms, GO number, CDD, Property/PATRIC ID e Prosite.

Os ficheiros que deram origem à coluna “Note from blast” foram os ficheiros *excel* “withoutnote” e “withnote”, que contêm a informação de todos os *hits* do *blast* para cada proteína. Após um profundo estudo dos *hits* foi escolhido o mais aproximado para as proteínas que não contêm nota. No caso de as proteínas conterem nota, foi comparada a função dada nos *hits* como a encontrada no *ncbi*. Nas não revistas foi obtido um *blast* muito inconclusivo.

Na nossa zona [0:246000] temos 203 genes, 7 pseudogenes, 4 tRNA, 43 proteínas hipotéticas, com 137 proteínas/genes sem nota.

Um gene/proteína que encontramos na literatura, e que não apareceu nos dados que retiramos automaticamente das bases de dados, para a nossa região do genoma, foi o *ponA*/PBP1 respectivamente. Os antibióticos β -lactamicos tem como um dos possíveis alvos a penicillin-binding protein 1 (PBP1) e PBP2, que são necessárias para a síntese do peptidoglicano. Sendo assim, uma mutação no gene *ponA* que codifica a PBP1 é uma das hipóteses que confere resistência aos antibióticos β -lactamicos.

Bibliografia

1. Jarvis, G.A. and T.L. Chang, *Modulation of HIV transmission by Neisseria gonorrhoeae: molecular and immunological aspects*. Curr HIV Res, 2012. **10**(3): p. 211-7.
2. Skerlev, M. and I. Culav-Koscak, *Gonorrhea: new challenges*. Clin Dermatol, 2014. **32**(2): p. 275-81.
3. Markowicz, S., et al., *Gonococcal aneurysm of the ascending aorta: case report and review of Neisseria gonorrhoeae endovascular infections*. Sex Transm Dis, 2014. **41**(2): p. 111-3.
4. Yu, R.X., et al., *Worldwide susceptibility rates of Neisseria gonorrhoeae isolates to cefixime and cefpodoxime: a systematic review and meta-analysis*. PLoS One, 2014. **9**(1): p. e87849.
5. Barbee, L.A. and J.C. Dombrowski, *Control of Neisseria gonorrhoeae in the era of evolving antimicrobial resistance*. Infect Dis Clin North Am, 2013. **27**(4): p. 723-37.
6. Watchirs Smith, L.A., et al., *Point-of-care tests for the diagnosis of Neisseria gonorrhoeae infection: a systematic review of operational and performance characteristics*. Sex Transm Infect, 2013. **89**(4): p. 320-6.
7. Rennie, R.P., *Current and future challenges in the development of antimicrobial agents*. Handb Exp Pharmacol, 2012(211): p. 45-65.
8. Unemo, M. and R.A. Nicholas, *Emergence of multidrug-resistant, extensively drug-resistant and untreatable gonorrhea*. Future Microbiol, 2012. **7**(12): p. 1401-22.
9. Goire, N., et al., *Molecular approaches to enhance surveillance of gonococcal antimicrobial resistance*. Nat Rev Microbiol, 2014. **12**(3): p. 223-9.
10. Buono, S.A., et al., *Stemming the tide of drug-resistant Neisseria gonorrhoeae: the need for an individualized approach to treatment*. J Antimicrob Chemother, 2014.
11. Blomquist, P.B., et al., *Is gonorrhea becoming untreatable?* Future Microbiol, 2014. **9**(2): p. 189-201.
12. Rossolini, G.M., et al., *Update on the antibiotic resistance crisis*. Curr Opin Pharmacol, 2014. **18C**: p. 56-60.
13. Whiley, D.M., et al., *The ticking time bomb: escalating antibiotic resistance in Neisseria gonorrhoeae is a public health disaster in waiting*. J Antimicrob Chemother, 2012. **67**(9): p. 2059-61.
14. Lewis, D.A., *Global resistance of Neisseria gonorrhoeae: when theory becomes reality*. Curr Opin Infect Dis, 2014. **27**(1): p. 62-7.

Chang, Jeff; Chapman, Brad; Friedberg, Iddo; Hamelryck, Thomas; de Hoon, Michiel ; Cock, Peter; Antao, Tiago ; Talevich, Eric ; Wilczyński, Bartek. **Biopython Tutorial and Cookbook**.