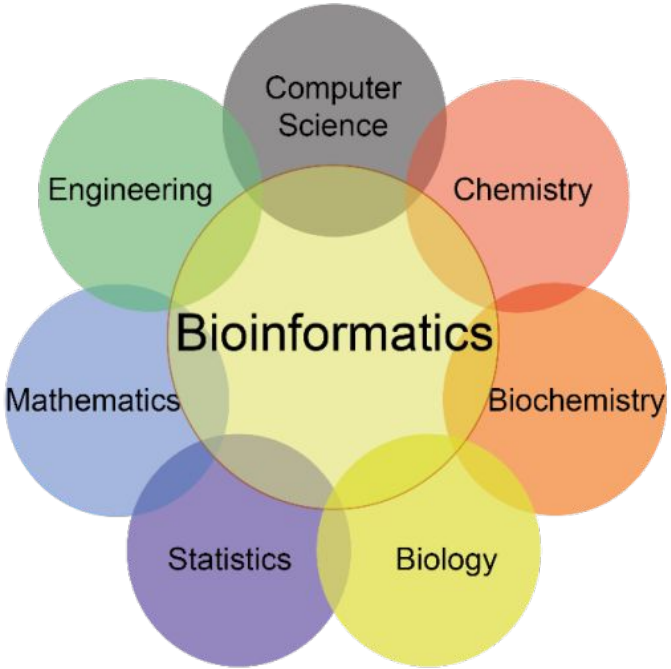
Bioinformática

#BioinformaticsGRX

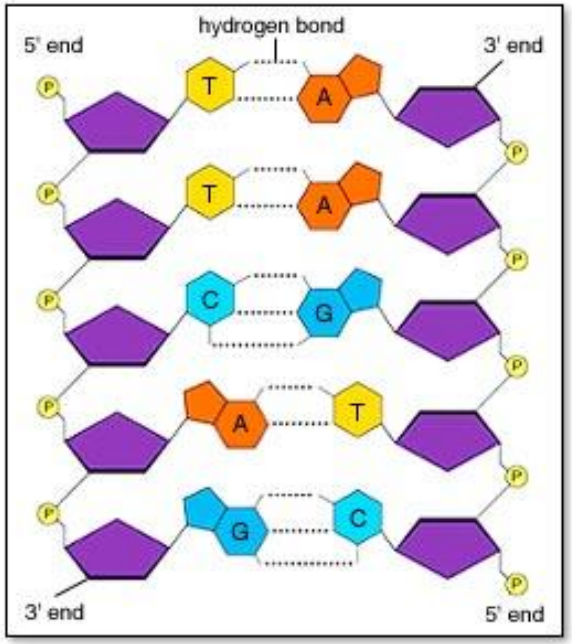
¿Qué es la bioinformática?

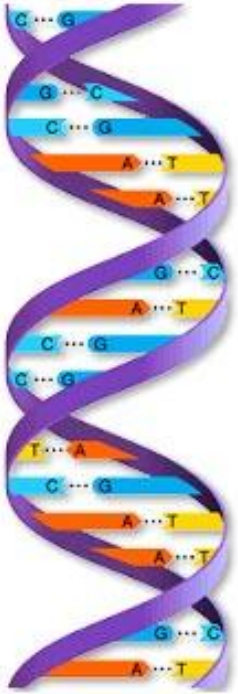
La bioinformática es la aplicación de tecnologías computacionales a la gestión y análisis de datos biológicos.

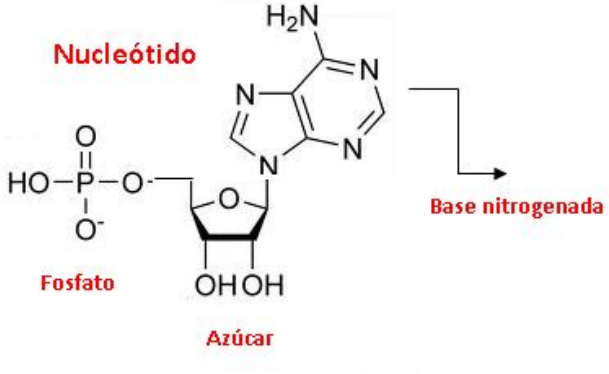
El término se acuñó en 1970 por Paulien Hogeweg y Ben Hesper



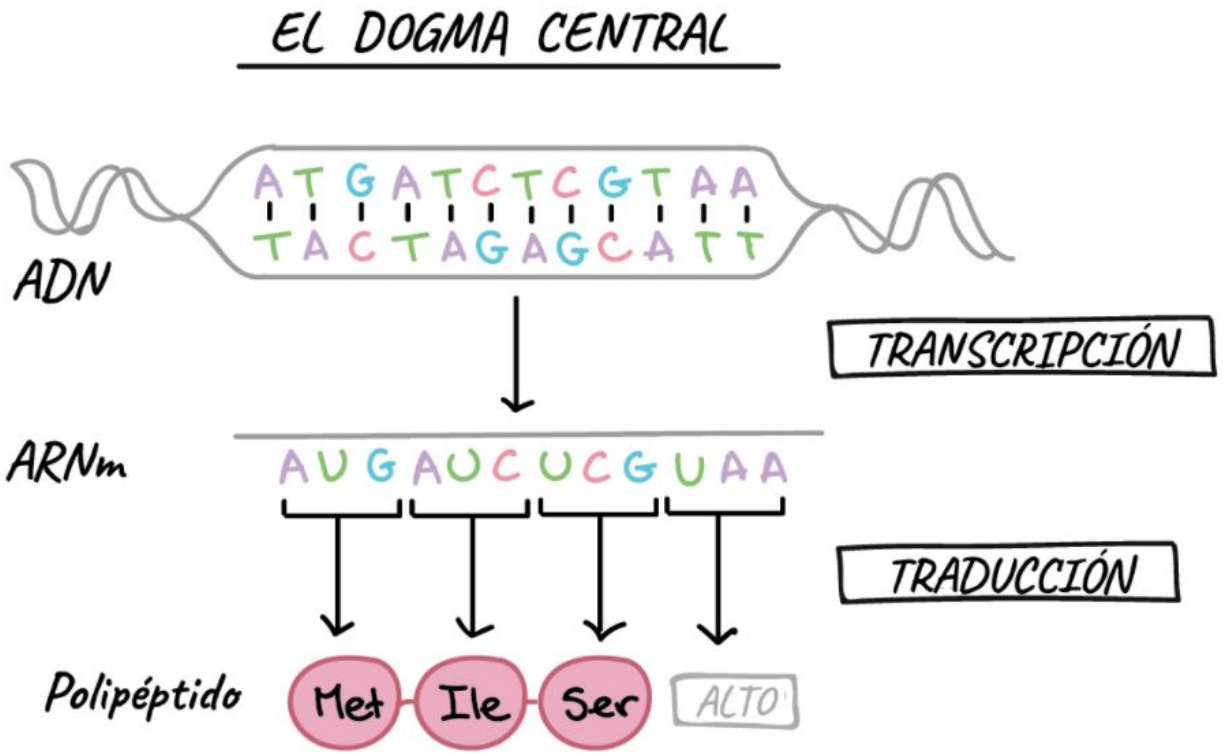
¿Qué es el ADN?


hydrogen bond
5' end
3' end
T
A
T
A
C
G
A
T
G
C
3' end
5' end

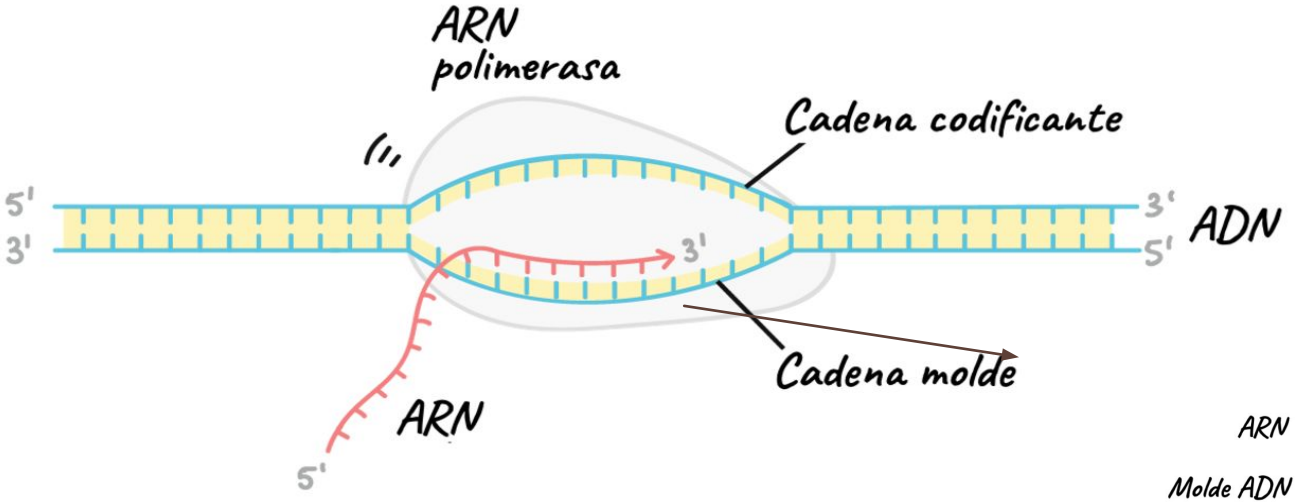


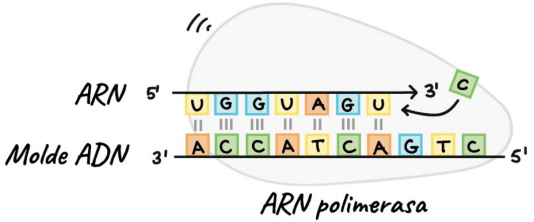

Nucleótido
H_2N
N
N
N
N
O
HO–P–O
O⁻
O
OH OH
Fosfato
Azúcar
Base nitrogenada

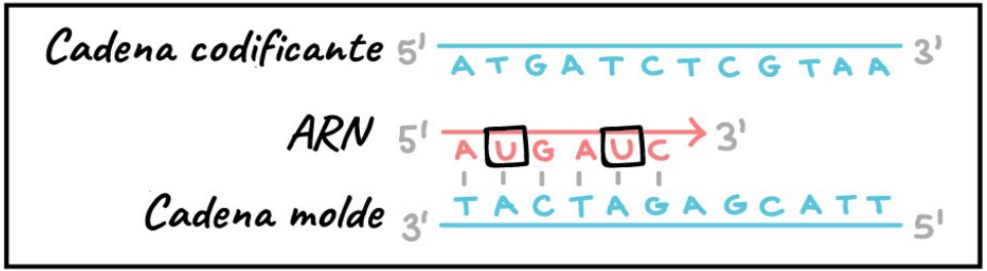
De ADN a proteína

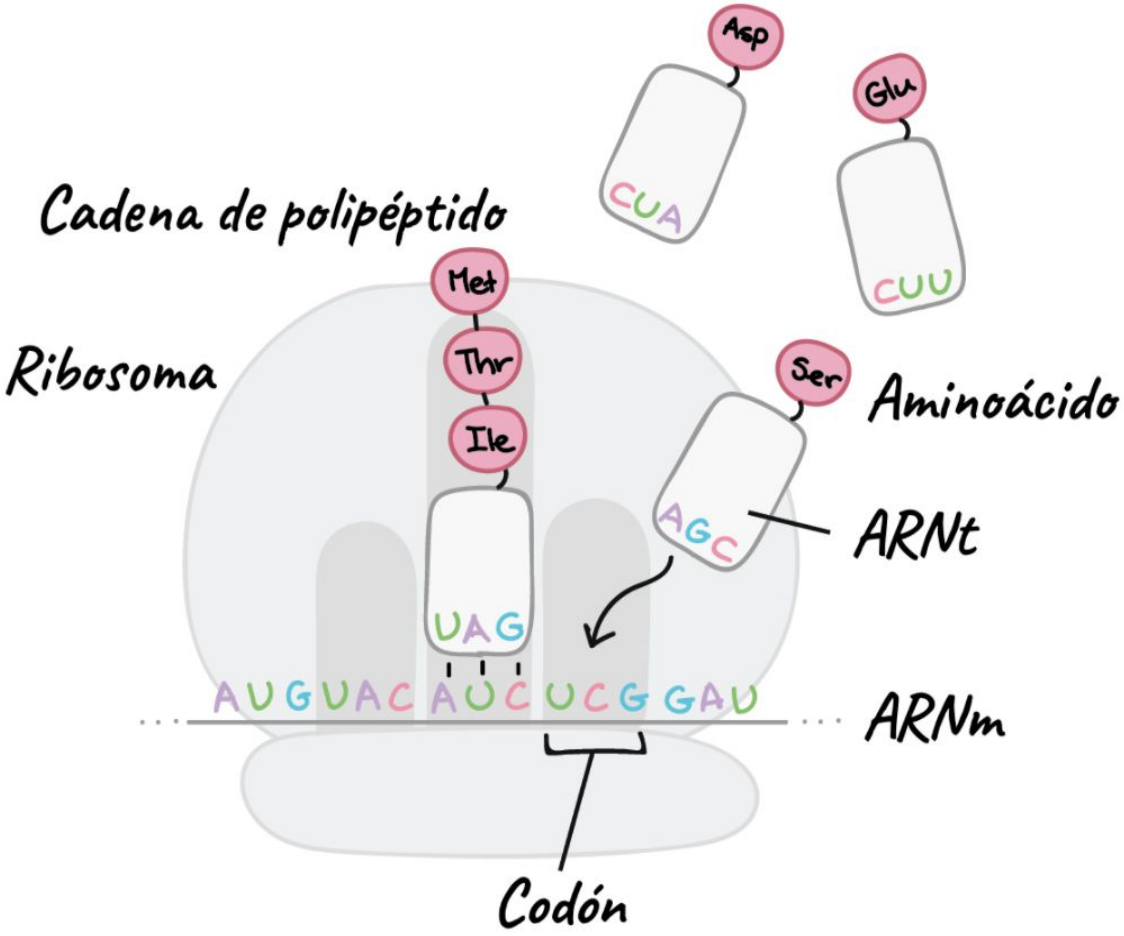


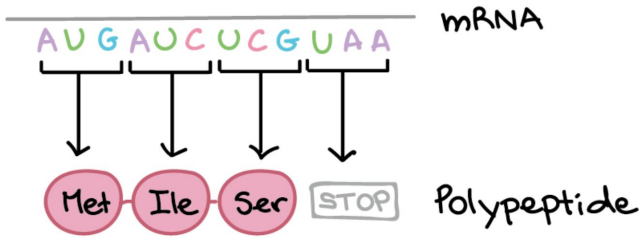
De ADN a proteína

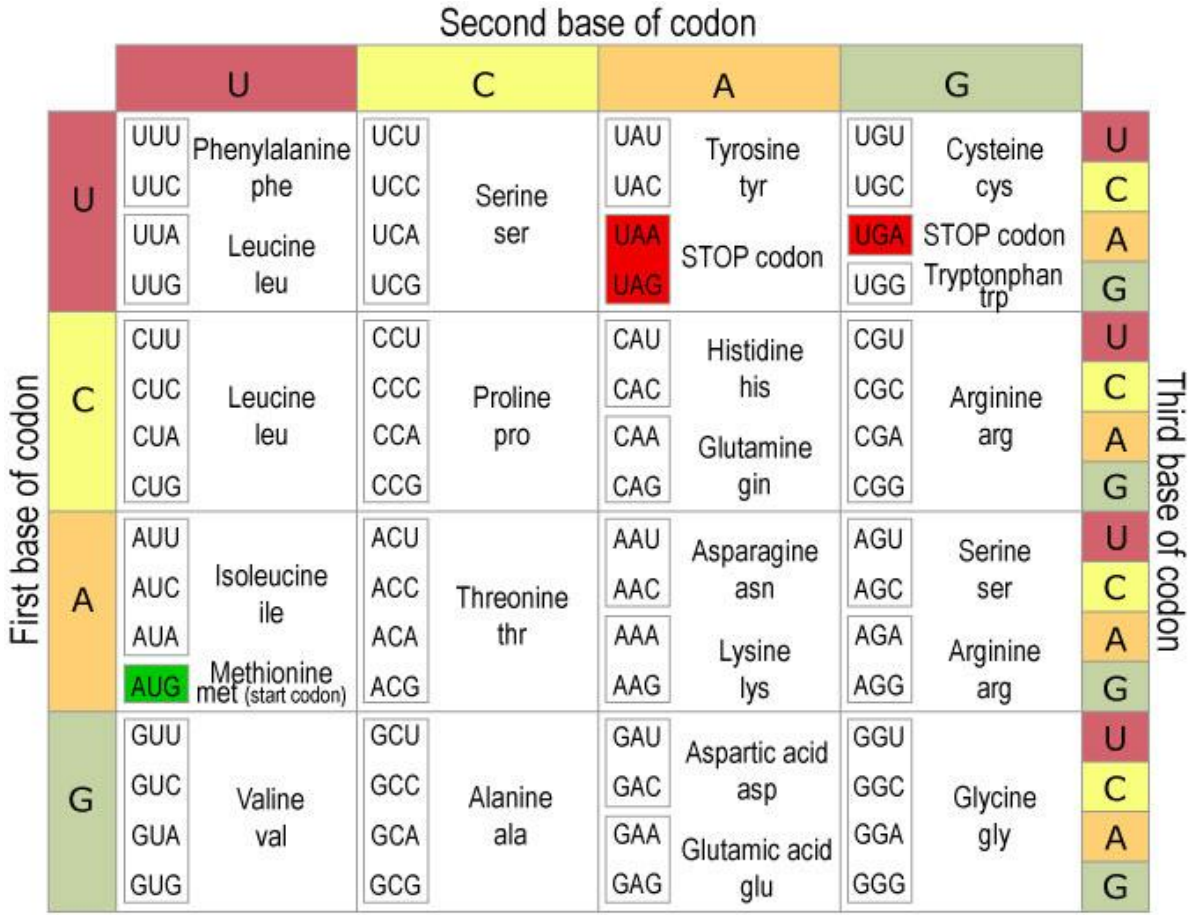




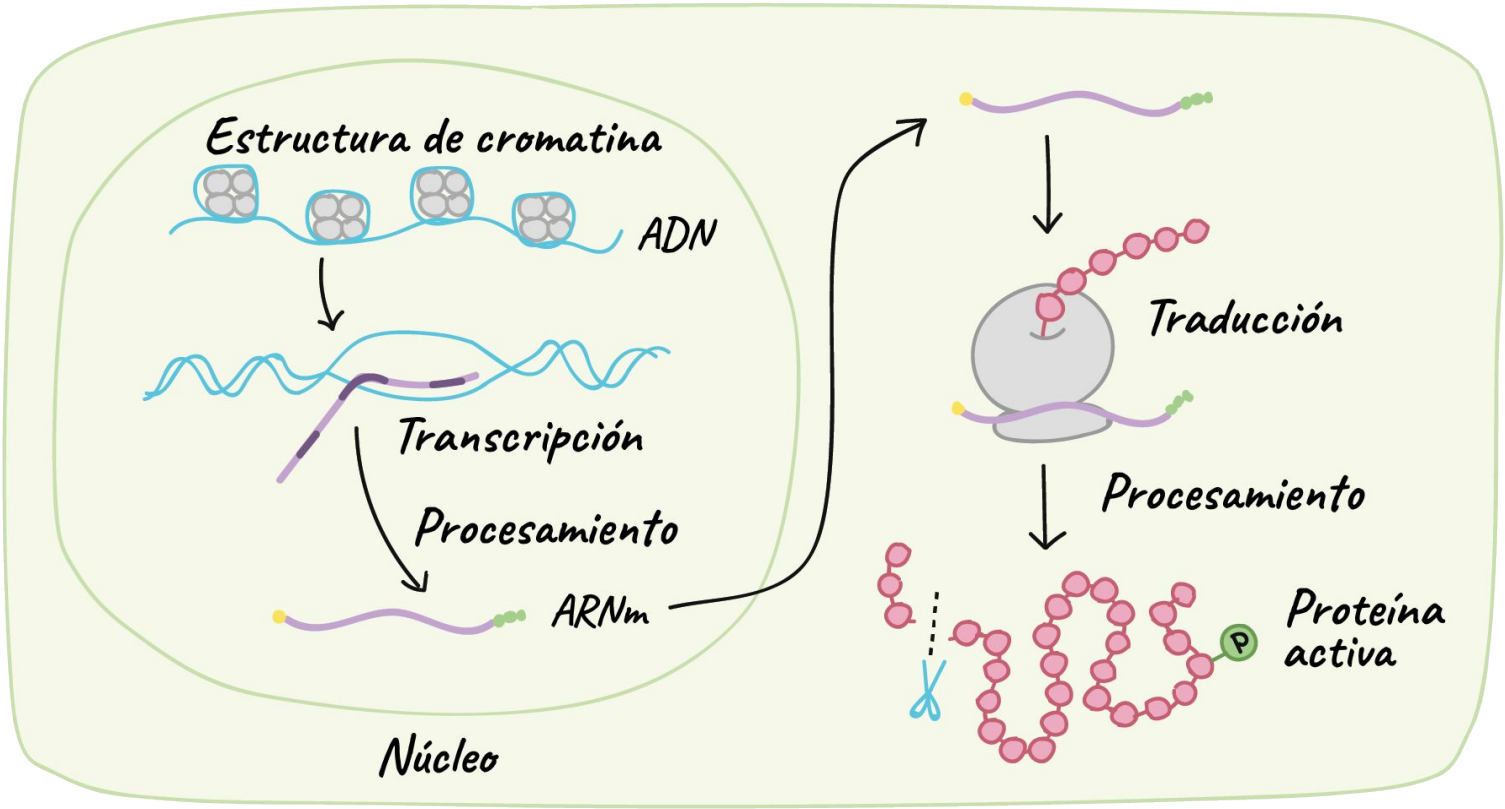



Asp
Glu
CUA
CUU
Cadena de polipéptido
Met
Thr
Ile
Ribosoma
Ser
Aminoácido
AGC
ARNt
UAG
AUGUACAUCUCGGAU
ARNm
Codón

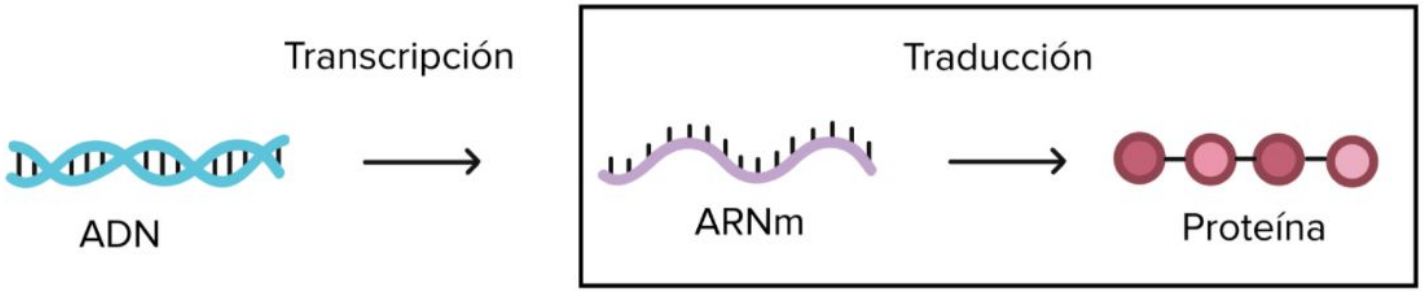

mRNA
AUGAUCUCGUAA
Met
Ile
Ser
STOP
Polypeptide



El código genético es degenerado


Estructura de cromatina
ADN
Transcripción
Procesamiento
ARNm
Núcleo
Traducción
Procesamiento
P
Proteína activa

De ADN a proteína



Campos de aplicación de la Bioinformática

- Medicina y medicina preventiva
- Microbiología
- Desarrollo de fármacos
- Resistencia antibiótica
- Biotecnología
- Agricultura
- Veterinaria
- Análisis forenses
- Muchas otros

Formas de aplicación de la Bioinformática

Genómica

Proteómica

Transcriptómica

Metabolómica

Ejemplo

Molde de ADN:

5’-ATGAAGTTTGGCACTTAA-3’
3’-TACTTCAAACCGTGAATT-5’

Ejemplo: Mioglobina

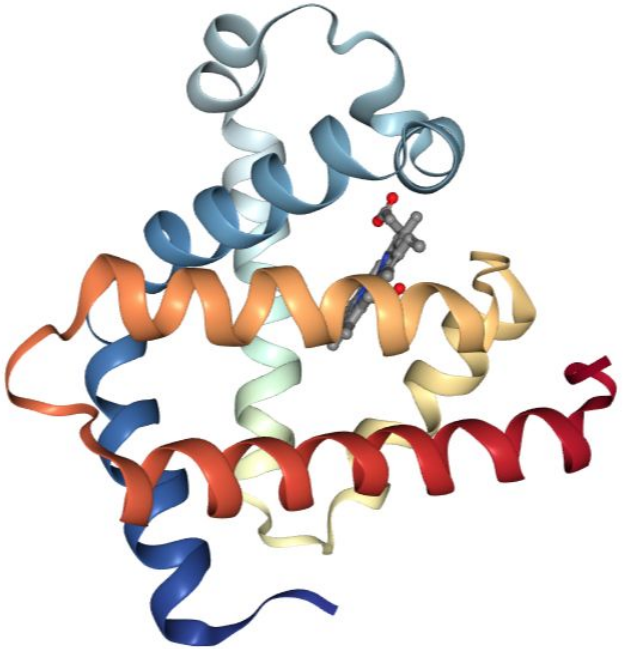
Secuencia:

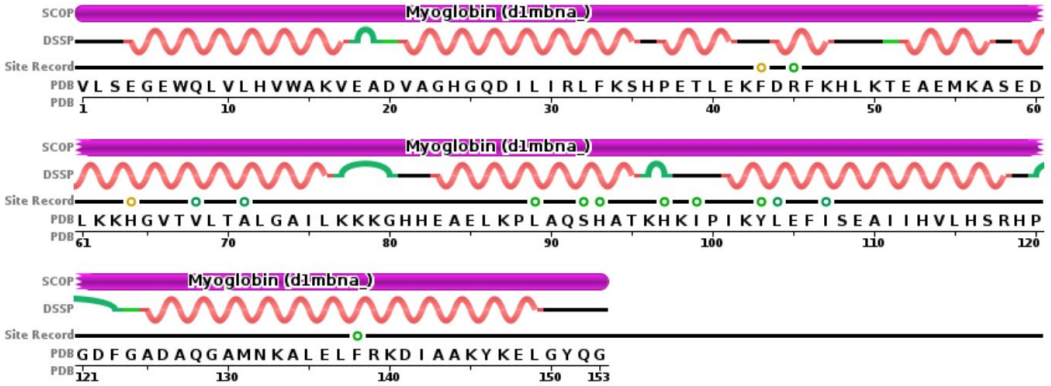
GAGCATGTTGGCCTGGTCCTTTG
CTAGGTACTGTAGAGCAGGTGAG
AGAGTGAGGGGGAAGGACTCCAA
ATTAGACCAGTTCTTAGCCATGAA
GCAGAGACTCTGAAGCCAGACTA
CCTGGGTCCCAATCTTGGGCTTG
GTATTTCCTCGCTGTGTGACTCTG
GGTAAGTTACTTAACTTCTCTGTG
CCTCAGTTCTCTCAAGTGTAAA...

Aminoácidos:

MGLSDGEWQLVLNVWGKVEADIPG
HGQEVLIRLFKGHPETLEKFDKFKHL
...

Ejemplo: Mioglobina





Matemáticas

#BioinformaticsGRX

Alguien tendrá que traducir los requerimientos biológicos en un lenguaje entendible para la máquina


I'm afraid we need to
use...MATH
ALL NEW

Multitud de herramientas matemáticas a nuestra disposición.

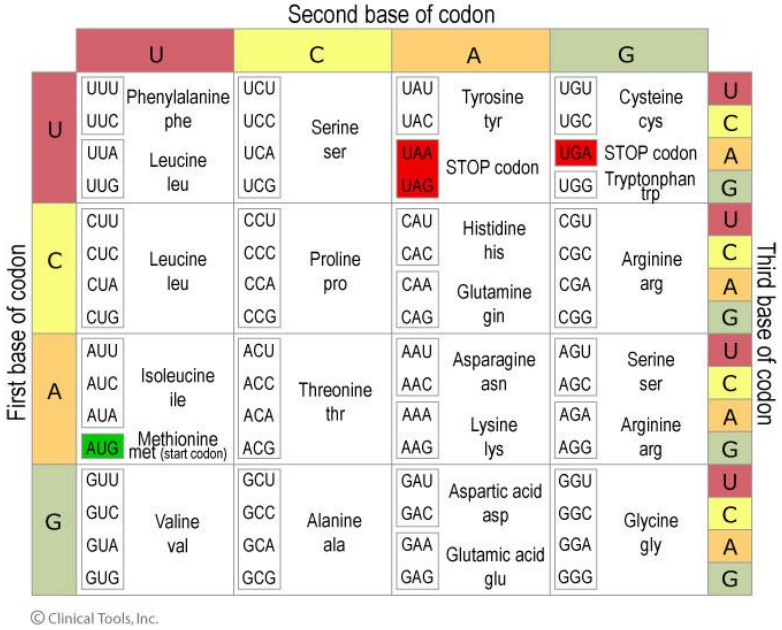
- Geometría diferencial
- Análisis
- Teoría de nudos
- Probabilidad y estadística
- Álgebra y álgebra abstracta

1.
Empecemos por lo básico.

¿Cómo representamos el código genético?

Claves

- Simetrías
- Matrices
- Código degenerado
- Inmunidad al ruido



Una posible idea:

Discernir subconjuntos dentro del código genético y utilizar con ellos lenguaje binario:

TABLE 2.2 Three Binary Subalphabets According to Three Types of Binary-Opposite Attributes in a Set of Nitrogenous Bases C, A, G, U[a]

N	Symbol of a Genetic Letter	C	A	G	U/T
1	0_1, pyrimidines (one ring in a molecule) 1_1, purines (two rings in a molecule)	0_1	1_1	1_1	0_1
2	0_2, a letter with amino-mutating property (amino) 1_2, a letter without it (keto)	0_2	0_2	1_2	1_2
3	0_3, a letter with three hydrogen bonds 1_3, a letter with two hydrogen bonds	0_3	1_3	0_3	1_3

Aprovechar los algoritmos existentes:

- Matrices de Hadamard y producto de Kronecker:

Si denominamos :

$$H_{n+1} = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}^n$$

Como el Kernel de la familia de matrices, inmediatamente podemos establecer una conexión con nuestro código genético:

$$P(n) = \begin{pmatrix} C & A \\ U & G \end{pmatrix}^n$$

$P^{(1)}=$

	0		1
		C	A
0	00	00	01
		(0)	(1)
		U	G
1	10	10	11
		(2)	(3)

;

$P^{(2)}=$

	00(0)	01(1)	10(2)	11(3)
	CC	CA	AC	AA
00	0000	0001	0010	0011
(0)	(0)	(1)	(2)	(3)
	CU	CG	AU	AG
01	0100	0101	0110	0111
(1)	(4)	(5)	(6)	(7)
	UC	UA	GC	GA
10	1000	1001	1010	1011
(2)	(8)	(9)	(10)	(11)
	UU	UG	GU	GG
11	1100	1101	1110	1111
(3)	(12)	(13)	(14)	(15)

	000 (0)	001 (1)	010 (2)	011 (3)	100 (4)	101 (5)	110 (6)	111 (7)
000 (0)	CCC 000000 (0)	CCA 000001 (1)	CAC 000010 (2)	CAA 000011 (3)	ACC 000100 (4)	ACA 000101 (5)	AAC 000110 (6)	AAA 000111 (7)
001 (1)	CCU 001000 (8)	CCG 001001 (9)	CAU 001010 (10)	CAG 001011 (11)	ACU 001100 (12)	ACG 001101 (13)	AAU 001110 (14)	AAG 001111 (15)
010 (2)	CUC 010000 (16)	CUA 010001 (17)	CGC 010010 (18)	CGA 010011 (19)	AUC 010100 (20)	AUA 010101 (21)	AGC 010110 (22)	AGA 010111 (23)
011 (3)	CUU 011000 (24)	CUG 011001 (25)	CGU 011010 (26)	CGG 011011 (27)	AUU 011100 (28)	AUG 011101 (29)	AGU 011110 (30)	AGG 011111 (31)
100 (4)	UCC 100000 (32)	UCA 100001 (33)	UAC 100010 (34)	UAA 100011 (35)	GCC 100100 (36)	GCA 100101 (37)	GAC 100110 (38)	GAA 100111 (39)
101 (5)	UCU 101000 (40)	UCG 101001 (41)	UAU 101010 (42)	UAG 101011 (43)	GCU 101100 (44)	GCG 101101 (45)	GAU 101110 (46)	GAG 101111 (47)
110 (6)	UUC 110000 (48)	UUA 110001 (49)	UGC 110010 (50)	UGA 110011 (51)	GUC 001100 (52)	GUA 110101 (53)	GGC 110110 (54)	GGA 110111 (55)
111 (7)	UUU 111000 (56)	UUG 111001 (57)	UGU 111010 (58)	UGG 111011 (59)	GUU 111100 (60)	GUG 111101 (61)	GGU 111110 (62)	GGG 111111 (63)

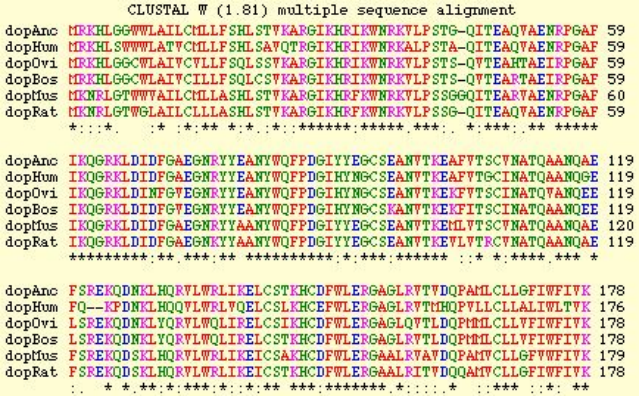
2.
Alineación

Bases

Una alineación entre dos (o más) secuencias es una comparación entre los componentes de cada secuencia.

El análisis básico de secuencias consiste en preguntar si hay dos o más secuencias relacionadas.

Una verdadera alineación de las secuencias biológicas es aquella que refleja la relación evolutiva entre dos o más secuencias que comparten un antepasado común antepasado común.



Bases

La base de la alineación y el análisis de la secuencia se basa en el hecho de que las secuencias biológicas se desarrollan a partir de secuencias preexistentes en lugar de ser inventado por la naturaleza desde el principio.

A esto debemos añadirle tres fenómenos principales:

- Mutaciones puntuales
- Inserciones
- Delecciones

Bases

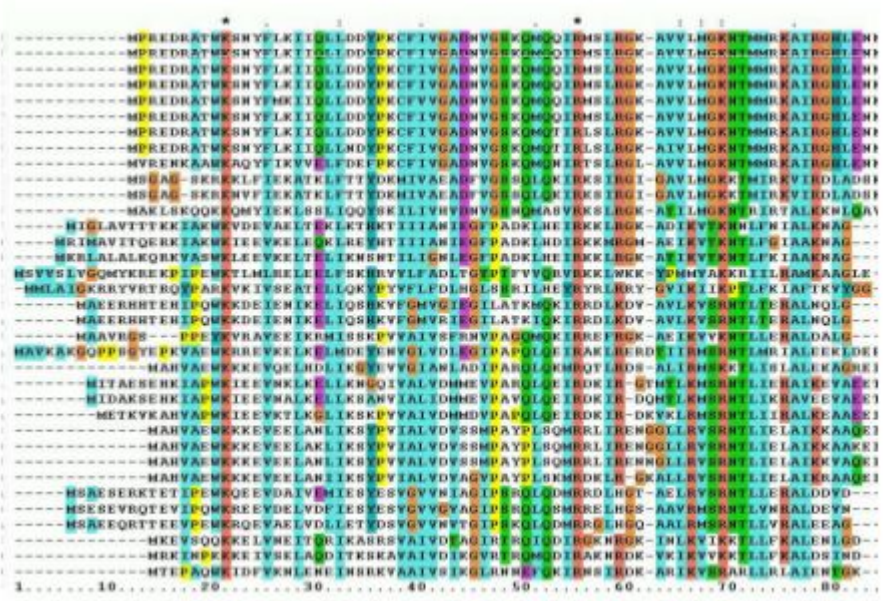
Además de los conceptos biológicos, debemos hacernos preguntas específicas con las que deberemos trabajar, y de las que dependerá nuestro método:

- ¿Qué tipo de alineamiento estamos buscando?
- ¿Qué puntuación aplicamos al alineamiento?
- ¿Qué algoritmo desarrollamos de forma óptima para llevar nuestro método a la máquina?
- ¿Cuál es la significación estadística de nuestro resultado?

Ejemplos:

- **Alineamiento global: FASTA**
- **Alineamiento local: BLAST**

Siguiente paso: Alineamiento múltiple.



3.
Análisis de componentes principales

Definición:

“Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (or sometimes, principal modes of variation)”

Formas básicas:

Existen dos formas básicas de aplicar el ACP:

1. **Método basado en la matriz de correlación, cuando los datos no son dimensionalmente homogéneos o el orden de magnitud de las variables aleatorias medidas no es el mismo.**
2. **Método basado en la matriz de covarianzas, que se usa cuando los datos son dimensionalmente homogéneos y presentan valores medios similares.**

Caso particular: Correlación

$$\mathbf{R} = [r_{ij}] \in M_{m \times m} \qquad \text{donde} \qquad r_{ij} = \frac{\operatorname{cov}(F_i, F_j)}{\sqrt{\operatorname{var}(F_i)\operatorname{var}(F_j)}}$$

http://setosa.io/ev/principal-component-analysis/

Multitud de herramientas matemáticas a nuestra disposición.

- Geometría diferencial
- Análisis
- Teoría de nudos
- Probabilidad y estadística
- Álgebra y álgebra abstracta

Informática

Bioinformática

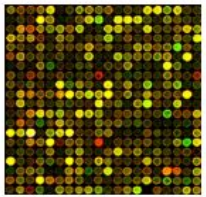
Illumina MiSeq

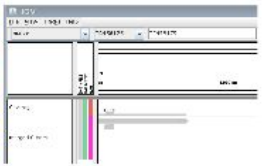
¡Ups!

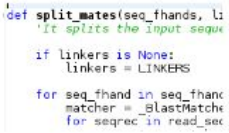

MiniSeq
MiSeq
NextSeq
HiSeq 4000
HiSeq X Ten
MAX OUTPUT
8 Gb
MAX READ NUMBER
25 million
MAX READ LENGTH
2x150 bp
MAX OUTPUT
15 Gb
MAX READ NUMBER
25 million
MAX READ LENGTH
2x300 bp
MAX OUTPUT
120 Gb
MAX READ NUMBER
400 million
MAX READ LENGTH
2x150 bp
MAX OUTPUT
1500 Gb
MAX READ NUMBER
5 billion
MAX READ LENGTH
2x150 bp
MAX OUTPUT
1800 Gb
MAX READ NUMBER
6 billion
MAX READ LENGTH
2x150 bp

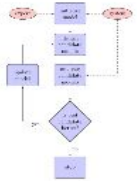
¿Qué recursos tenemos?


R45大沢(24.8K)上り









Biological Data

GUI usage

CLI usage

Ad hoc script

programming

Algorithm

Math and IT staff

Bioinformaticians

Biologists

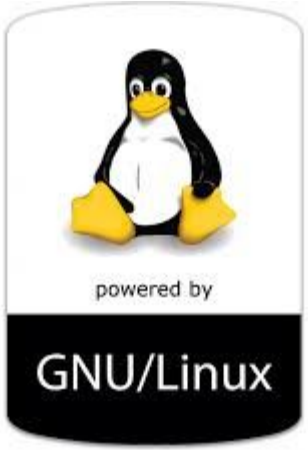
¿Qué podemos hacer?

- Programar
- Utilizar plataformas fiables y potentes

¿Qué podemos hacer?

- Aprender a dar órdenes al ordenador con total libertad
- Utilizar plataformas fiables y potentes

Otros sistemas operativos



https://es.wikipedia.org/wiki/GNU/Linux

¿Programar?


ReactionGIFS.me


SOFTPEDIA
WH
.GOV
CO
DE

¿Cuál?

JavaScript
Perl
PHP
DOSBatch
Markup
Tcl-Tk
Logic-based
LabVIEW
Logo
Visual
Haskell
Python
Scala
Functional
Simula
Miranda
C++
SQL
.NET
XML
Pascal
VisualBasic
MATLAB
Cobol
Prototype-based
DataStructured
Erlang
Declarative
Ruby
Ada
Scripting
Multiparadigm
Squeak
Java
Clojure
Lisp
Obfuscated
Scheme
Reflective
HTML
Prolog
Fortran
Assembly
Object-Oriented
Procedural
BASIC
Interpreted
Imperative
Distributed
AppleScript
C-sharp
Smalltalk
PostScript
HyperCard
Yorick
Occam
Wirth
Goedel
Algol60
Mathematica
TRAC
Befunge
Blue
Bigwig
Pike
Rigal
Frontier
Awk
ElasticC
Lua
Miva
Fortress
REBOL
Clarion
ObjectiveCaml
Database
BETA
Self
Euphoria
RPG
Cocoa
Pliant
Dylan
TeX
Maple
Oz
Tempo
S-Lang
PowerBuilder
VHDL
ML
Mumps
Verilog
CurlyBracket
Groovy
Rexx
XOTcl
Intercal
Lagoona
LaTeX
Clean
Jal
Oberon
IDL
Bistro
Caml
FP
YAFL
Limbo
VBScript
APL
TADS
SAS
Protous
Forth
Objective-C
Sather
CobolScript
Leda
ADL
Mercury
POP-11
Algol68
Dataflow
ARB
CHILL
Delphi
Eiffel
ABC
Snobol
T3X
Icon
Turing
ICI
Refal
GLSL
Curl
UML
SETL
Simkin
Moto
CLU
Sisal
Cecil
SGML

¿Python?


python™

¿Por qué Python?

- Tiene una sintaxis sencilla
- Viene con bibliotecas por defecto con muchas tareas comunes
- Es libre
- Es interpretado
- Es muy utilizado (también en la comunidad científica)

Sintaxis sencilla

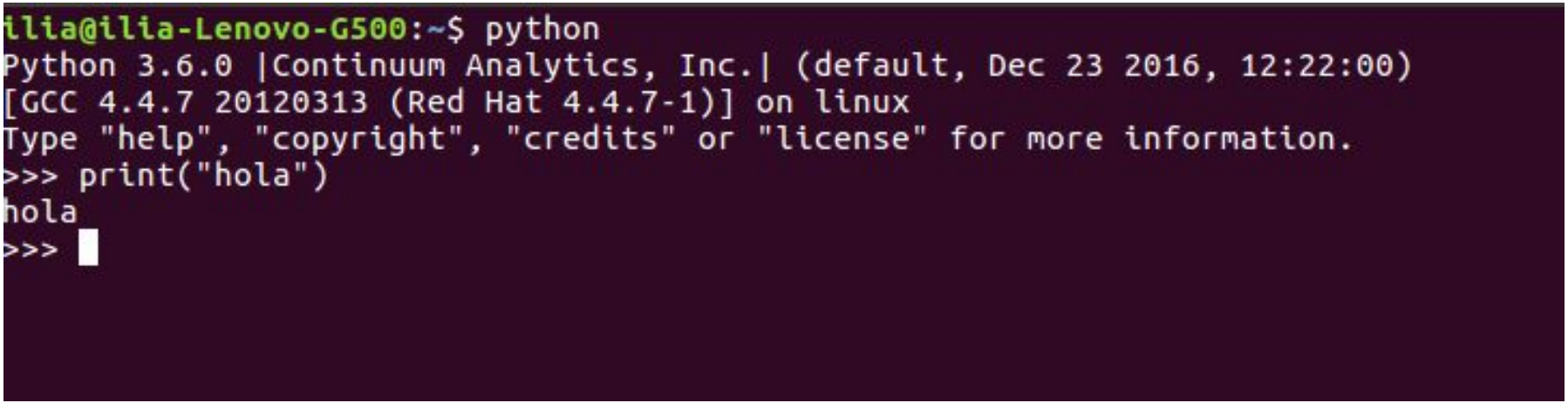
```
print("Hola mundo")
```

Código en Python

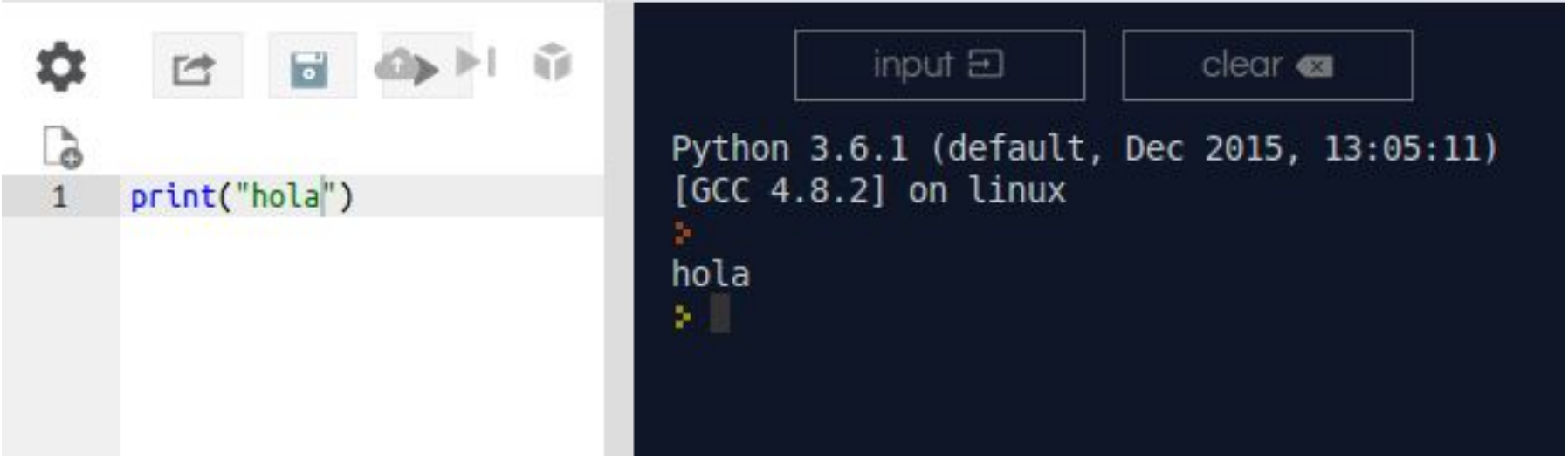
```
#include <stdio.h>

int main(void) {
    printf("Hello, World!\n");

    return 0;
```

Código en C

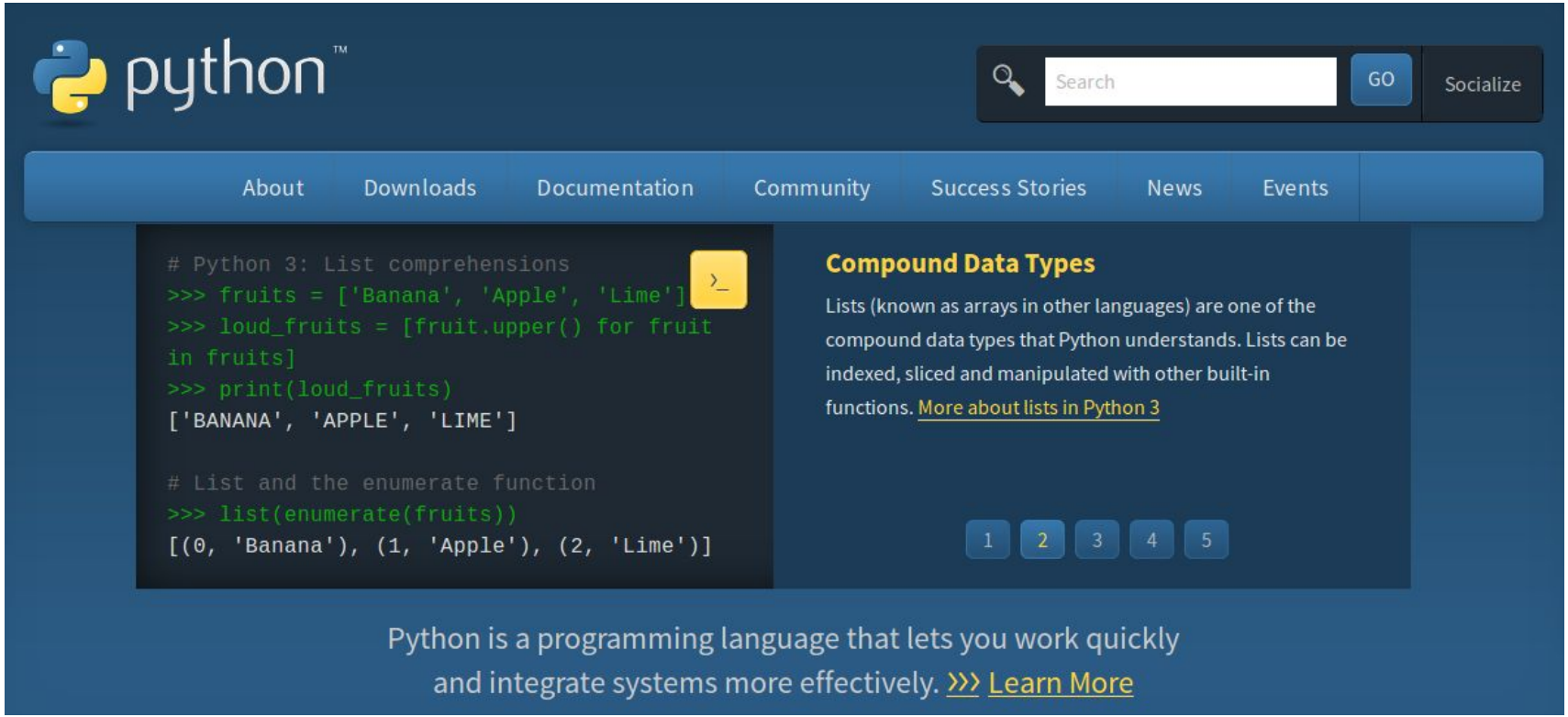
Consola o intérprete



https://repl.it/languages/python3



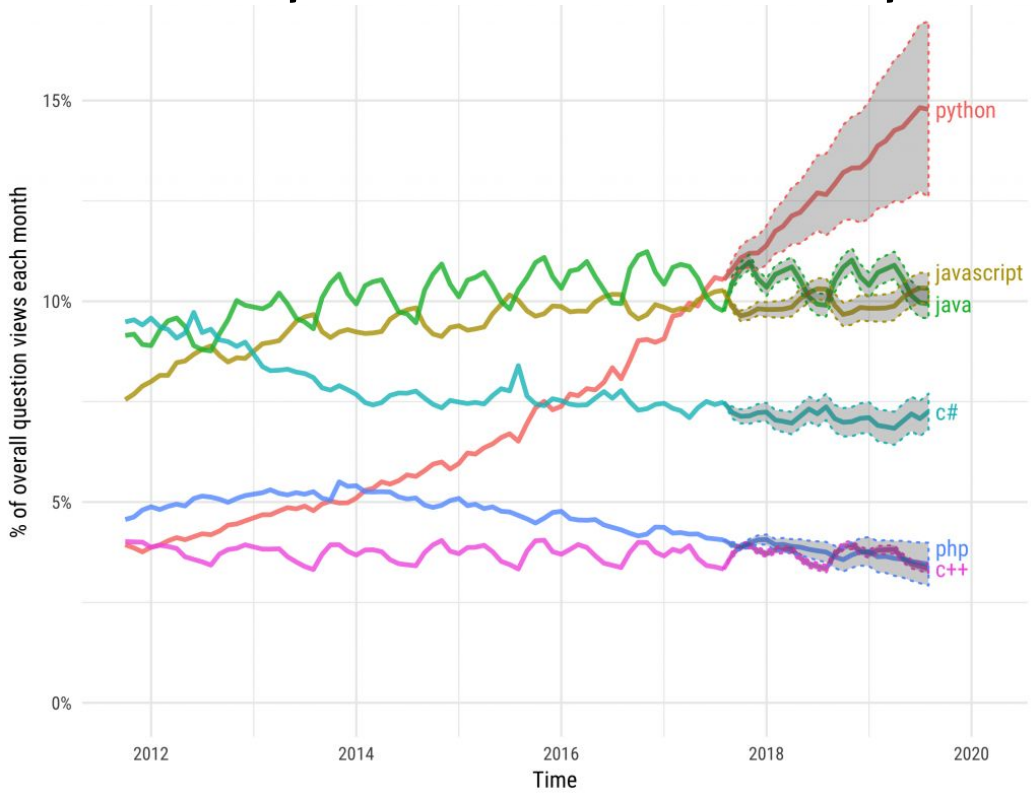
OpenSource - www.python.org



Ranking de lenguajes según IEEE en 2017

https://stackoverflow.blog/2017/09/06/incredible-growth-python/

Predicción de tráfico en stack overflow



https://stackoverflow.blog/2017/09/06/incredible-growth-python/

http://biopython.org/

Un ejemplito

```
dna = 'AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAA'

nt = ['A', 'C', 'G', 'T']

for n in nt:

    nt_total = dna.count(n)

    print('{}: {}'.format(n, nt_total))
```

¿Puedo tener más interactividad?

http://jupyter.org/

- Interactividad
- Más lenguajes (+40): R, Perl, Octave...
- Compartir *notebooks*
- Incrustar imágenes, vídeos y componentes para manipular los datos

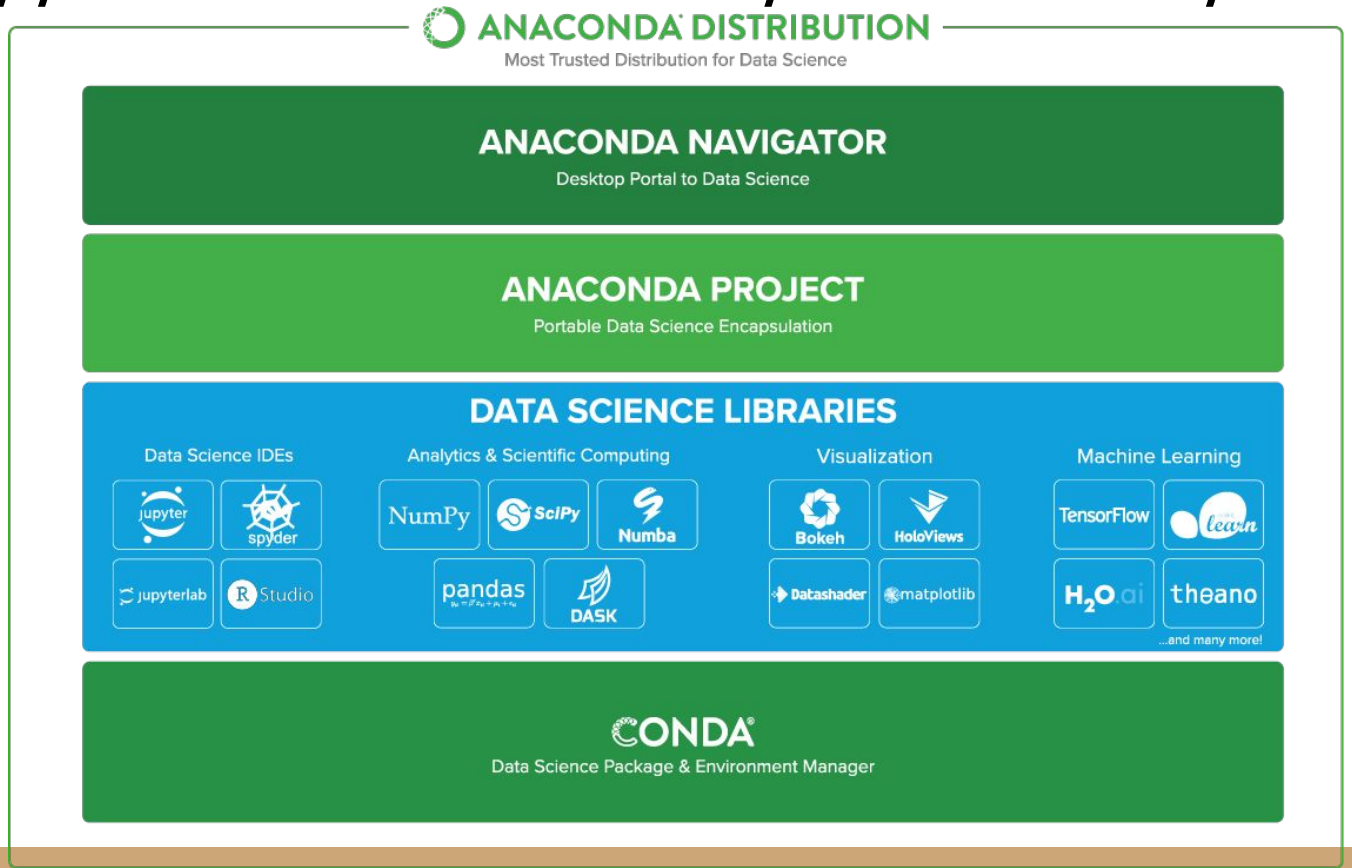
https://try.jupyter.org/

Compartir y colaborar

Aplicación

- Libros
- Cursos
- Talleres

https://www.anaconda.com/distribution/



Compartir y Colaborar

Ciencia abierta

Artículo abierto sobre RNA

jupyter nbviewer

JUPYTER FAQ

Zika-RNAseq-Pipeline / Zika.ipynb

An open RNA-Seq data analysis pipeline tutorial with an example of reprocessing data from a recent Zika virus study

Zichen Wang[1] and *Avi Ma'ayan*[1*]

[1]Department of Pharmacology and Systems Therapeutics;
BD2K-LINCS Data Coordination and Integration Center;
Mount Sinai Knowledge Management Center for Illuminating the Druggable Genome;
Icahn School of Medicine at Mount Sinai, One Gustave L. Levy Place, Box 1603, New York, NY 10029 USA

*Correspondence: avi.maayan@mssm.edu

Abstract

RNA-seq analysis is becoming a standard method for global gene expression profiling. However, open and standard pipelines to perform RNA-seq analysis by non-experts remain challenging due to the large size of the raw data files and the hardware requirements for running the alignment step. Here we introduce a reproducible open source RNA-seq pipeline delivered as an IPython notebook and a Docker image. The pipeline uses state-of-the-art tools and can run on various platforms with minimal configuration overhead. The pipeline enables the extraction of knowledge from typical RNA-seq studies by generating interactive principal component analysis (PCA) and hierarchical clustering (HC) plots, performing enrichment analyses against over 90 gene set libraries, and obtaining lists of small molecules that are predicted to either mimic or reverse the observed changes in mRNA expression. We apply the pipeline to a recently published RNA-seq dataset collected from human neuronal progenitors infected with the Zika virus (ZIKV). In addition to confirming the presence of cell cycle genes among the genes that are downregulated by ZIKV, our analysis uncovers significant overlap with upregulated genes that when knocked out in mice induce defects in brain morphology. This result potentially points to the molecular processes associated with the microcephaly phenotype observed in newborns from pregnant mothers infected with the virus. In addition, our analysis predicts small molecules that can either mimic or reverse the expression changes induced by ZIKV. The IPython notebook and Docker image are freely available at: http://nbviewer.jupyter.org/github/maayanlab/Zika-RNAseq-Pipeline/blob/master/Zika.ipynb and https://hub.docker.com/r/maayanlab/zika/

Keywords

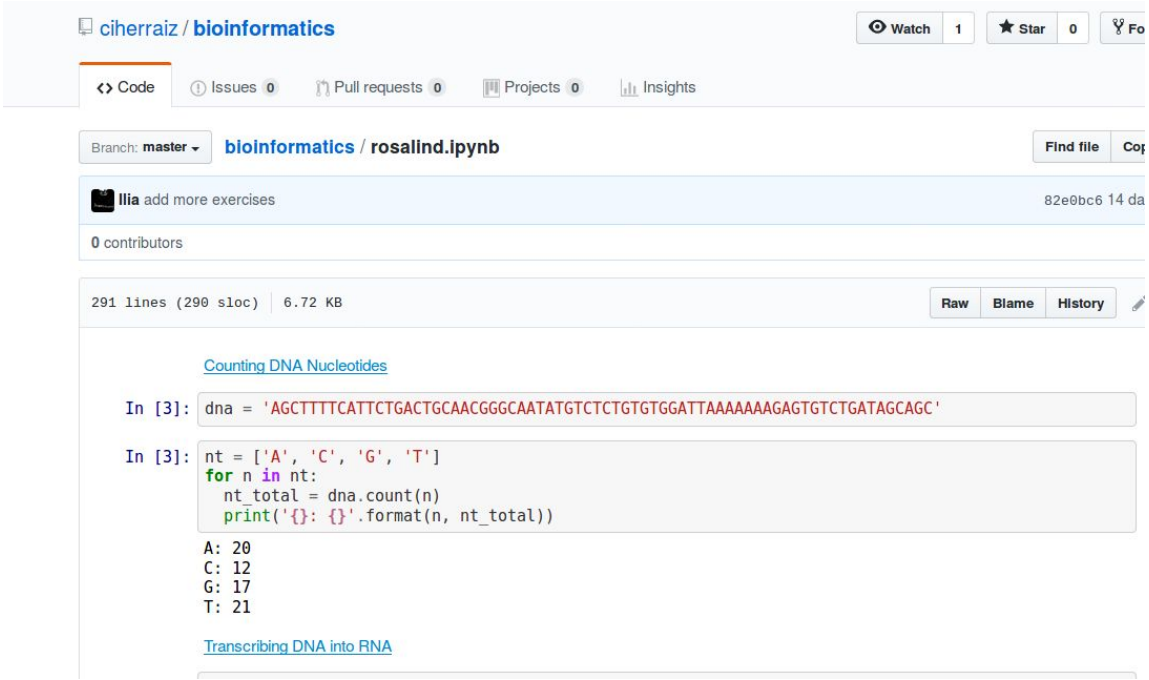
Systems biology, bioinformatics pipeline, gene expression analysis, RNA-seq

Introduction

The increase in awareness about the irreproducibility of scientific research requires the development of methods that make experimental and computational protocols easily repeatable and transparent [1]. The advent of interactive notebooks for data analysis pipelines significantly enhances the recording and sharing of data, source code, and figures [2]. In a subset of recent publications, an interactive notebook was published alongside customary manuscripts [3].

http://nbviewer.jupyter.org/github/maayanlab/Zika-RNAseq-Pipeline/blob/master/Zika.ipynb

Ejercicios resueltos de Rosalind



https://github.com/ciherraiz/bioinformatics/blob/master/rosalind.ipynb

¡Muchas gracias!

Déjanos tu opinión: https://tinyurl.com/yc58omy7

Dónde encontrarnos


t.me/bioinformaticsGRX


@bioinfGRX


https://github.com/bioinformaticsGRX


bioinformaticsgrx.slack.com
(con invitación)