

# **Introducción a la Bioinformática**

## **Experimentos *in silico***

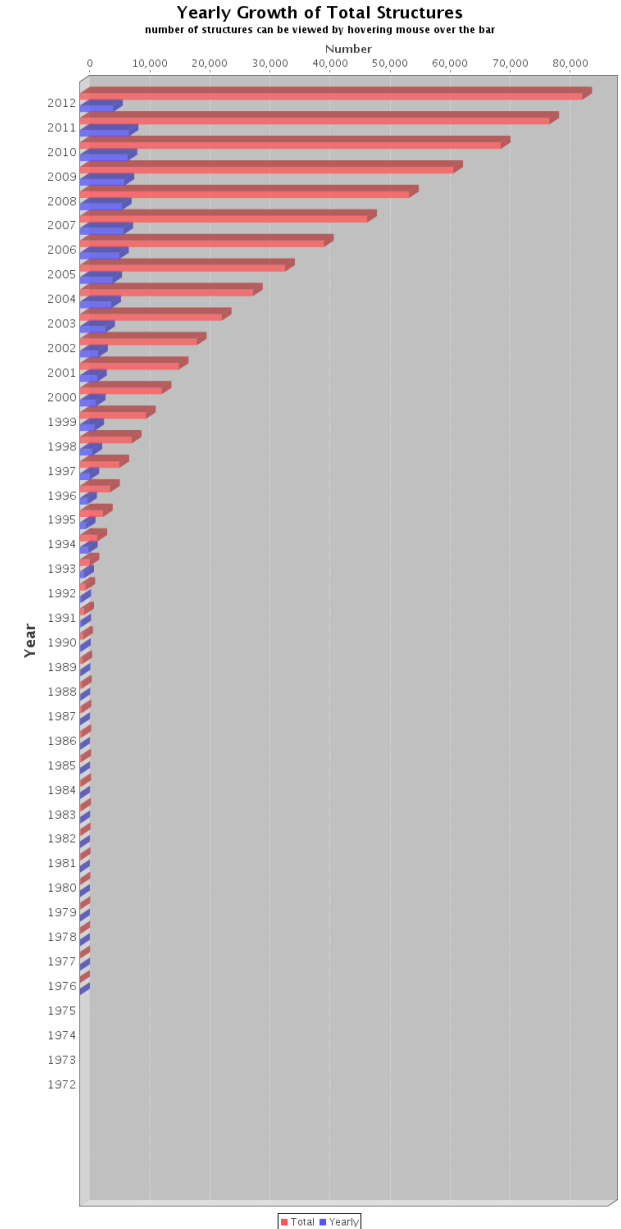
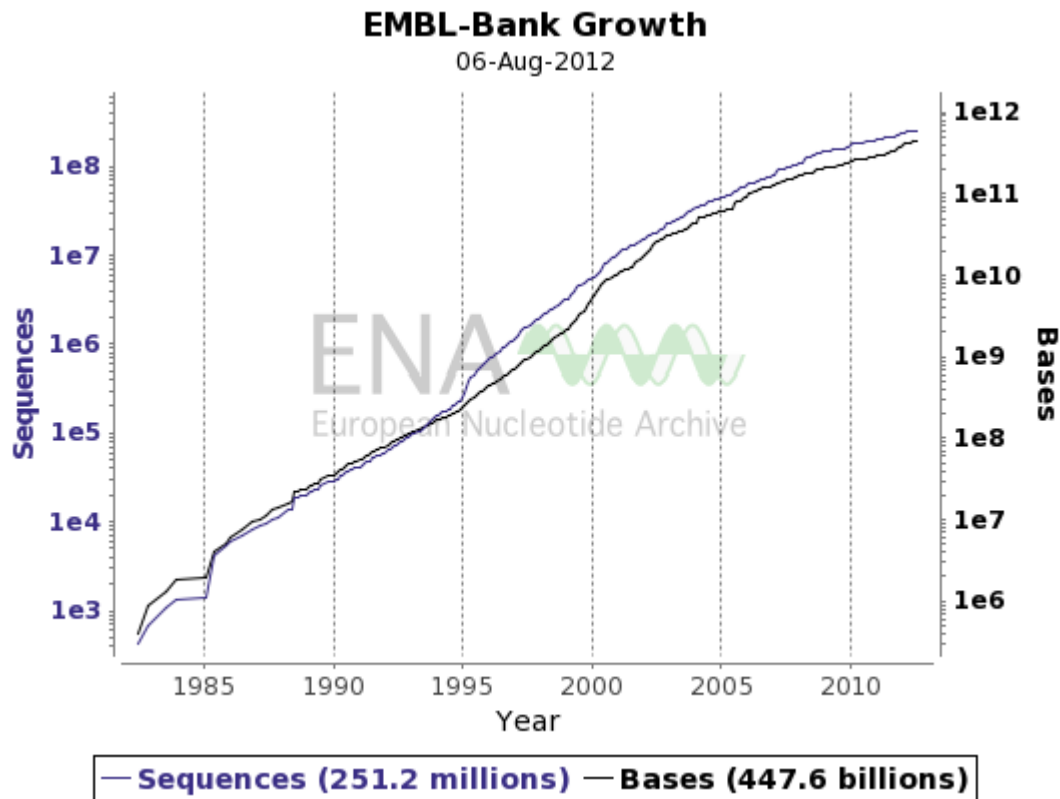
**Fernán Agüero**

**Instituto de Investigaciones Biotecnológicas**  
**Universidad Nacional de General San Martín**

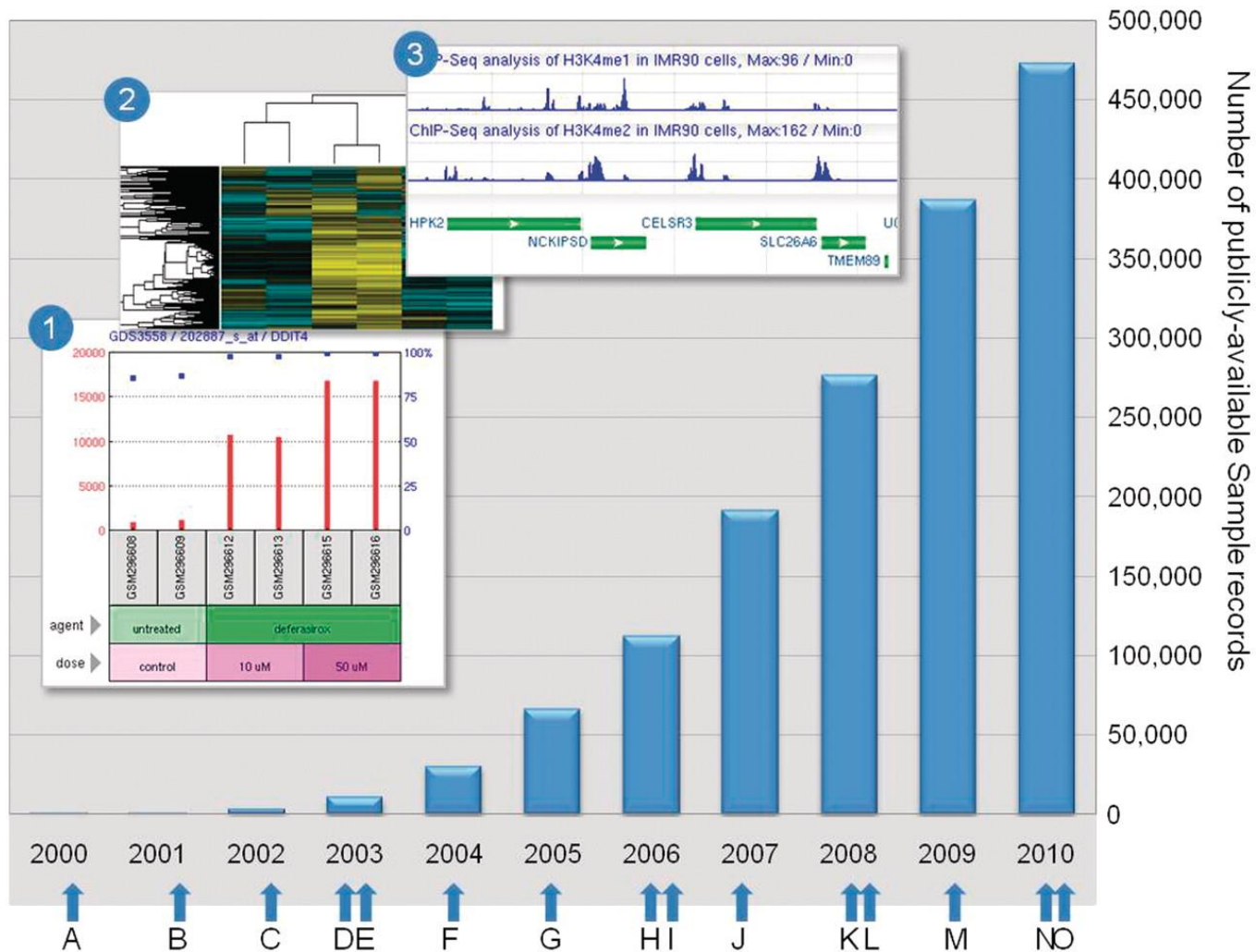
# Un breve repaso histórico

- **La aparición de las secuencias completas del genoma humano y cientos de otros genomas es el producto de un siglo de investigación dirigido a comprender la información genética.**
- **Comienzos del siglo XX: redescubrimiento de las leyes de Mendel**
- **Durante el primer cuarto de siglo, la biología descubrió que la base celular de la información eran los cromosomas**
- **Durante el segundo cuarto de siglo, se descubrió que la base molecular de la información era el DNA**
- **Durante el tercer cuarto de siglo, se definieron los mecanismos que utilizan las células para leer esta información y se desarrollaron las herramientas de DNA recombinante**
- **Durante el ultimo cuarto de siglo, los biólogos se volcaron a coleccionar información genética - primero de genes, luego de genomas completos, transcriptomas, etc.**

# Información biológica



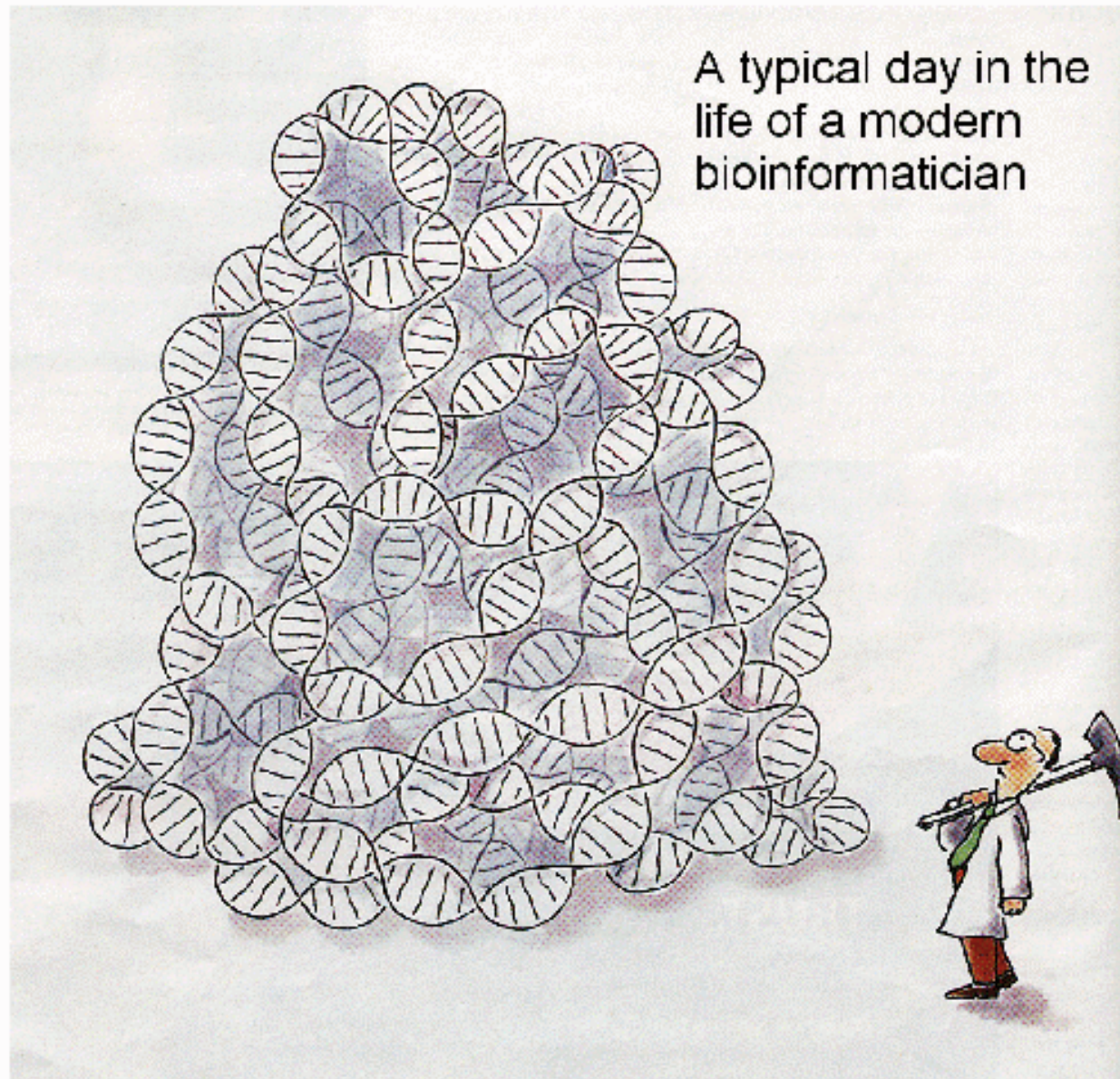
# A timeline of GEO database growth, development and events.



Barrett T et al. Nucl. Acids Res. 2011;39:D1005-D1010

Nucleic Acids Research

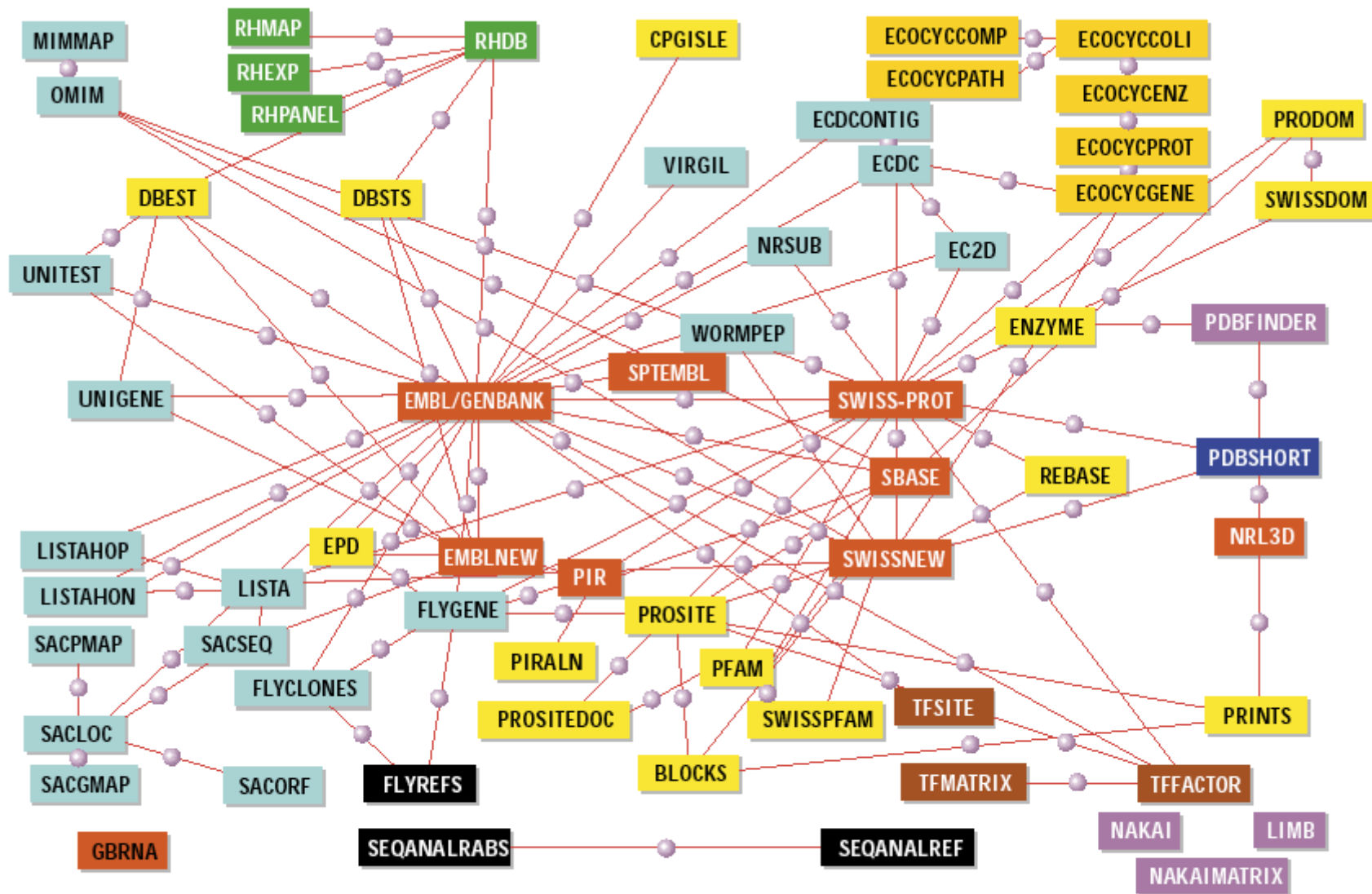
# En que estamos hoy ...





# En que estamos hoy ...

- **El resultado: de ser una ciencia puramente experimental (con base en el laboratorio) la biología está siendo transformada en una ciencia de la información**
- **La información acumulada no sólo es información genética (secuencias de DNA)**
  - **expresión de RNAs**
  - **interacción entre proteínas**
  - **estructuras tridimensionales**
  - **Anulación sistemática de genes (knockouts, RNAi) que produce información de fenotipos**
  - **...**
- **Cada vez más diversos estudios comienzan con el análisis de bases de datos para luego formular hipótesis o diseñar experimentos**
- **Cada vez más el trabajo de laboratorio termina en la acumulación de colecciones masivas de datos que deben ser luego analizados**



■ Sequence 
 ■ Protein Structure 
 ■ SeqRelated 
 ■ Genome 
 ■ Metabolic 
 ■ Literature 
 ■ Others 
 ■ Transfac 
 ■ Mapping

# Un experimento bioinformático

- **Un experimento en la computadora no es distinto de cualquier experimento en la mesada:**
  - los resultados deben contestar una pregunta concreta
  - deben ser reproducibles por otra persona que utilice el mismo método
- **Identificar el problema**
  - cuál es el mecanismo catalítico de la enzima X?
  - Qué genes ven afectada su expresión por el tratamiento con la droga Y?
- **Identificar las herramientas necesarias para resolver el problema**
  - búsquedas de secuencias similares, alineamientos múltiples, detección de perfiles y motivos, modelado de la estructura tridimensional, evaluación del modelo
  - Análisis comparativo de cambios globales de expresión usando datos de RNAseq o microarrays
- **Definir criterios de satisfacción (éxito del experimento)**
  - Prácticamente todos los métodos computacionales producen resultados. Una búsqueda utilizando BLAST casi siempre produce algún hit
  - Es necesario distinguir resultados significativos del ruido para no terminar comparando superóxido dismutasas con alcohol dehidrogenasas.
  - Hay que entender cómo funcionan los programas, en qué algoritmos están basados, que puntos débiles tienen, etc.
  - Y plantear una estrategia estadística! (hipótesis nula)



# Un experimento bioinformático ...

- **Seleccionar el set de datos apropiados**

- **En el laboratorio, los materiales y reactivos son objetos físicos necesarios para realizar un experimento. Generalmente uno sabe cuando fueron preparados, quien los preparo, como fueron preparados, etc.**
- **En bioinformática el mismo tipo de información es esencial. Las fuentes de información (bases de datos, por ej), fecha de ultima actualizacion, el criterio y el metodo utilizado para extraer los datos que van a ser utilizados en el experimento**

**El costo de un proyecto bioinformático es bajo una vez que cubierto el gasto inicial en computadoras (y eventualmente software)**

# Un ejemplo concreto

- Un investigador interesado en estudiar genes involucrados en la interacción hospedador-parásito, con especial interés en identificar aquellos productos que sean secretados
- Un sitio web reporta los resultados de un análisis sistemático de expresión (usando microarrays) de todos los genes del genoma en todos los estadios del ciclo de vida del parásito
- El investigador puede bajar un archivo con un resumen de estos experimentos
- Las secuencias de todas las proteínas codificadas por el genoma se encuentran disponibles en una base de datos.
- Lo que se necesita es contar con la capacidad de identificar genes que se expresen en los estadios del ciclo de vida que ocurren en el hospedador y extraer las secuencias de estos genes de la base de datos
- En última instancia el objetivo es analizar las secuencias de interés usando SignalP para predecir la posible presencia de un péptido señal

# Cuestiones a tener en cuenta:

- **Podemos hacer el trabajo 'a mano'**

- Abrimos el resumen con los datos de los experimentos con microarrays en un procesador de texto
- buscamos los genes que muestran expresión en el estadio de interés
- Construimos una lista de genes (accession numbers)
- Luego vamos a nuestra base de datos con secuencias genómicas y sus traducciones y buscamos una por una las secuencias
- El ultimo paso es pasar todas las secuencias a un formato que entienda SignalP y ingresarlas una por una en el formulario correspondiente.

- **Hay tres problemas evidentes:**

- Si el número de genes que se expresan en nuestro estadio de interes es más que 'unos cuantos' el trabajo se vuelve tedioso y más que nada lento por el tiempo que insume
- Peor aun, cada vez que aparezcan nuevos resultados de microarrays o se actualicen, hay que repetir todo el procedimiento
- El proceso de abrir el resumen con datos de microarrays (o la base de datos de genes) en un procesador de textos puede no ser factible si el tamaño de los archivos excede los 5 o 10 MB

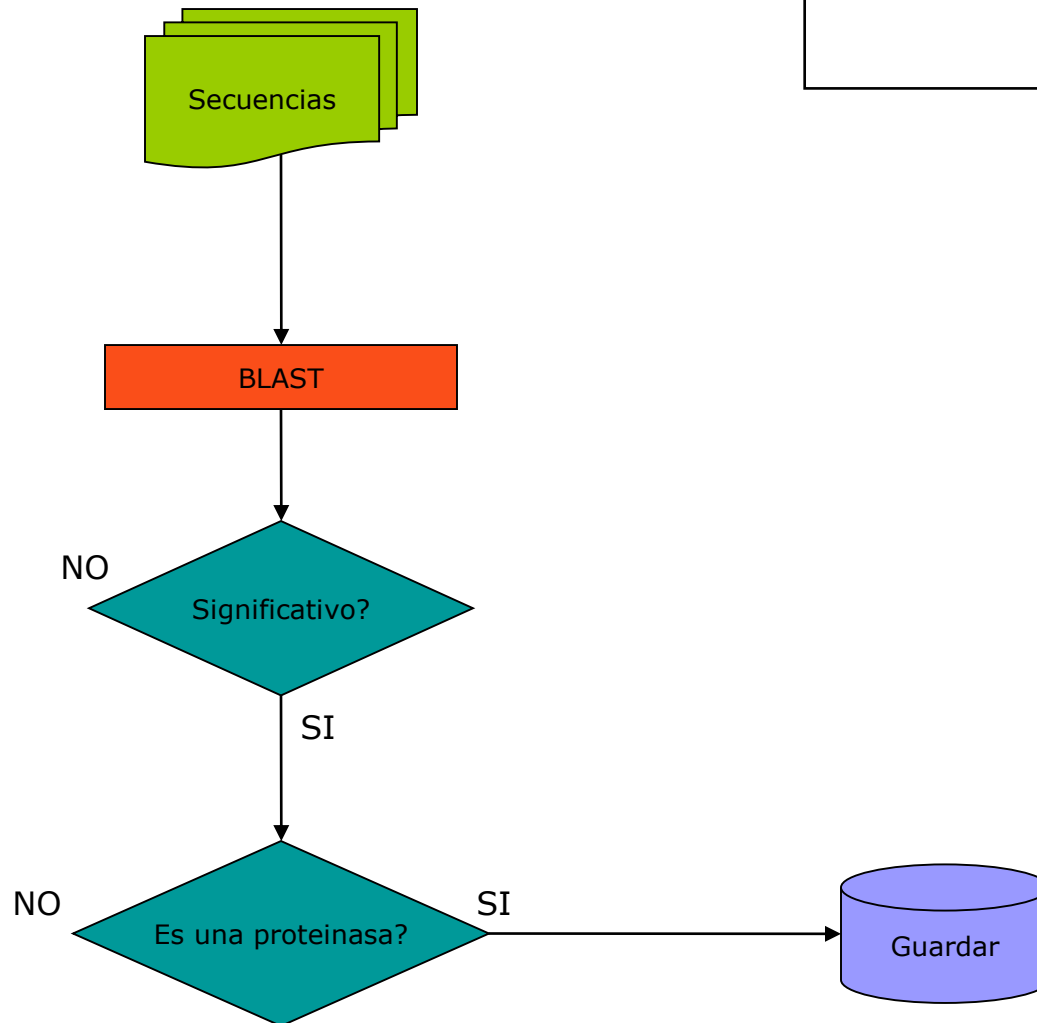
# Programación en biología

- **Cualquier persona que tenga experiencia en el diseño y llevado a cabo de experimentos para responder una pregunta puede programar una computadora**
- **Un experimento en el laboratorio comienza con una pregunta que evoluciona hacia una hipótesis testeable**
- **Finalmente el experimento sirve para afirmar o descartar una afirmación**
- **En la computadora el programa que uno escriba debe estar diseñado de manera de producir resultados que respondan a este tipo de afirmaciones**
- **Aprender un lenguaje de programación puede resultar un desafío no trivial, pero es similar a aprender a utilizar una nueva herramienta, tecnología u otro lenguaje (inglés, francés)**

# Programación en biología

- **Ejemplos simples:**
  - automatizar tareas
  - identificar una o más tareas que uno quiere realizar
  - escribir un programa que las realice en forma automática
- **Analizar todas las proteínas de un genoma y seleccionar aquellas que sean (o parezcan) proteinasas**
  - Un archivo con todas las secuencias
  - Una base de datos de proteínas (Swissprot, GenPept)
  - Un programa para buscar secuencias similares en bases de datos (BLAST)
  - Una serie de instrucciones a seguir (un protocolo)

# Automatizar búsquedas con BLAST





# Automatizar BLAST

- **Muy lindo el diagrama, pero: cómo se hace?**
- **Por cada secuencia de una lista de secuencias hay que:**
  - **correr la comparación (BLAST) contra una base de datos**
  - **analizar el reporte que genera el programa y extraer dos tipos de datos:**
    - score, expect, identidad, similitud (algún criterio cuantitativo que me sirva para tomar una decisión)
    - descripción de la secuencia obtenida de la base de datos

```
>gi|32172429|sp|P25807|CYS1_CAEEL Gut-specific cysteine proteinase precursor  
>gi|32172419|sp|P07268|PRZN_SERSP Serralysin precursor (Extracellular metalloproteinase) (Zinc proteinase)
```

# Programación

- Todo lenguaje de programación provee construcciones para tomar decisiones:
  - if **A** then do **B**, else do **C**
  - if **A > 100** then **continue** else **exit**
- Algunos lenguajes de programación proveen métodos para ejecutar otros programas
  - salir al sistema operativo, ejecutar el programa X y tomar el output
  - **blast secuencia vs swissprot**
  - **system( "blast -i secuencia -d swissprot" )**
- Lo más difícil: analizar el output y tomar los datos de interés
  - para poder tomar decisiones (hacer comparaciones) tenemos que tener los datos en variables

# Reportes de BLAST

- **Un reporte de BLAST tal como aparece en un navegador o al ejecutar el programa en la línea de comando (Unix) es básicamente un archivo de texto (un archivo plano o flatfile)**
- **Ningun reporte es igual a otro. Sin embargo hay patrones similares (la apariencia de hecho es similar). Tenemos que entrenar a nuestro programa para reconocer patrones:**
  - **la primer línea contiene información sobre el programa**
  - **la quinta línea contiene información sobre la secuencia utilizada para la búsqueda**
  - **la décima línea contiene información sobre la base de datos**
  - **la línea que comienza con '>' indica el comienzo de la descripción de un hit**
  - **etc.**

# Anatomía de un reporte de BLAST

## Header

### Programa

\$programa = "TBLASTN"  
\$version = "2.2.6"

TBLASTN 2.2.6 [Apr-09-2003]

### Query

\$id = "GROU\_DROME"  
\$accession = "P16371"  
\$descripcion = "Groucho protein ..."  
\$longitud = "719"

#### Reference:

Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997), "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs", Nucleic Acids Res. 25:3389-3402.

RID: 1056464019-01741-14921

Query= GROU\_DROME P16371 Groucho protein (Enhancer of split M9/10) (719 letters)

### Base de datos

\$database = "GenBank non-mouse ..."  
\$secuencias = "8104717"

Database: GenBank non-mouse and non-human EST entries  
8,104,717 sequences; 4,196,642,183 total letters

If you have any problems or questions with the results of this search please refer to the [BLAST FAQs](#)

[Taxonomy reports](#)

# Anatomía de un reporte de BLAST

## Hit List

| Sequences producing significant alignments:        |                               | Score<br>(bits)    | E<br>Value |
|----------------------------------------------------|-------------------------------|--------------------|------------|
| <a href="#">gi 32150256 gb CB923560.1 CB923560</a> | TcAmaP103Run01_C08 Trypa...   | <a href="#">59</a> | 7e-10      |
| <a href="#">gi 4513990 gb AI562645.1 AI562645</a>  | TENS2632 T. cruzi epimast...  | <a href="#">44</a> | 2e-05      |
| <a href="#">gi 4775755 gb AI664767.1 AI664767</a>  | TENG0727 T. Cruzi epimast...  | <a href="#">42</a> | 6e-05      |
| <a href="#">gi 3331819 gb AI057953.1 AI057953</a>  | TENU2045 T. cruzi epimast...  | <a href="#">37</a> | 0.002      |
| <a href="#">gi 32150583 gb CB923715.1 CB923715</a> | TcAmaP104Run01_A07 Trypa...   | <a href="#">36</a> | 0.005      |
| <a href="#">gi 3492632 gb AI110425.1 AI110425</a>  | TENU4028 T. cruzi epimast...  | <a href="#">36</a> | 0.006      |
| <a href="#">gi 3404513 gb AI075562.1 AI075562</a>  | TENU3139 T. cruzi epimast...  | <a href="#">36</a> | 0.006      |
| <a href="#">gi 3115610 gb AA952514.1 AA952514</a>  | TENS1449 T. cruzi epimast...  | <a href="#">35</a> | 0.013      |
| <a href="#">gi 3115566 gb AA952470.1 AA952470</a>  | TENS1408 T. cruzi epimast...  | <a href="#">35</a> | 0.013      |
| <a href="#">gi 3258752 gb AI035022.1 AI035022</a>  | TENG0082 T. cruzi epimast...  | <a href="#">29</a> | 0.72       |
| <a href="#">gi 32150506 gb CB923679.1 CB923679</a> | TcAmaP110Run01_G08 Trypa...   | <a href="#">29</a> | 0.72       |
| <a href="#">gi 3415247 gb AI078900.1 AI078900</a>  | TENU3693 T. cruzi epimast...  | <a href="#">28</a> | 0.94       |
| <a href="#">gi 11266027 gb BF317566.1 BF317566</a> | 24G-2-13 Trypanosoma cru...   | <a href="#">28</a> | 1.2        |
| <a href="#">gi 11266021 gb BF317564.1 BF317564</a> | 24G-2-11 Trypanosoma cru...   | <a href="#">28</a> | 1.2        |
| <a href="#">gi 11233231 gb BF299431.1 BF299431</a> | SC-1-9 Trypanosoma cruzi...   | <a href="#">28</a> | 1.2        |
| <a href="#">gi 11233227 gb BF299429.1 BF299429</a> | SC-1-7 Trypanosoma cruzi...   | <a href="#">28</a> | 1.2        |
| <a href="#">gi 3411477 gb AI077169.1 AI077169</a>  | TENU3391 T. cruzi epimast...  | <a href="#">28</a> | 1.2        |
| <a href="#">gi 32150600 gb CB923724.1 CB923724</a> | TcAmaP109Run01_H01 Trypa...   | <a href="#">24</a> | 1.3        |
| <a href="#">gi 32150264 gb CB923564.1 CB923564</a> | TcAmaP108Run01_F09 Trypa...   | <a href="#">27</a> | 2.1        |
| <a href="#">gi 4827391 gb AI668083.1 AI668083</a>  | TENG01005 T. Cruzi epimas...  | <a href="#">27</a> | 2.1        |
| <a href="#">gi 3340717 gb AI065310.1 AI065310</a>  | TENU2197 T. cruzi epimast...  | <a href="#">27</a> | 2.1        |
| <a href="#">gi 4521410 gb AI563028.1 AI563028</a>  | TENS2191 T. cruzi epimast...  | <a href="#">27</a> | 2.7        |
| <a href="#">gi 3392864 gb AI069889.1 AI069889</a>  | TENU2937 T. cruzi epimast...  | <a href="#">27</a> | 3.6        |
| <a href="#">gi 3331821 gb AI057955.1 AI057955</a>  | TENU2047 T. cruzi epimast...  | <a href="#">27</a> | 3.6        |
| <a href="#">gi 3115528 gb AA952432.1 AA952432</a>  | TENS1370 T. cruzi epimast...  | <a href="#">27</a> | 3.6        |
| <a href="#">gi 3340579 gb AI065172.1 AI065172</a>  | TENU2052 T. cruzi epimast...  | <a href="#">26</a> | 4.7        |
| <a href="#">gi 3299275 gb AI050158.1 AI050158</a>  | TENU1403 T. cruzi epimast...  | <a href="#">26</a> | 4.7        |
| <a href="#">gi 3294057 gb AI046159.1 AI046159</a>  | TENU1245 T. cruzi epimast...  | <a href="#">26</a> | 4.7        |
| <a href="#">gi 3340825 gb AI065418.1 AI065418</a>  | TENU2306 T. cruzi epimast...  | <a href="#">26</a> | 4.7        |
| <a href="#">gi 3321028 gb AI053149.1 AI053149</a>  | TENU1567 T. cruzi epimast...  | <a href="#">26</a> | 4.7        |
| <a href="#">gi 7327682 gb AW621093.1 AW621093</a>  | TENU5213 T. cruzi epimasti... | <a href="#">26</a> | 6.1        |
| <a href="#">gi 32150478 gb CB923665.1 CB923665</a> | TcAmaP106Run01_C09 Trypa...   | <a href="#">26</a> | 6.1        |
| <a href="#">gi 4647750 gb AI622825.1 AI622825</a>  | TENG0609 T. Cruzi epimast...  | <a href="#">26</a> | 6.1        |
| <a href="#">gi 4514073 gb AI562728.1 AI562728</a>  | TENS2715 T. cruzi epimast...  | <a href="#">26</a> | 6.1        |
| <a href="#">gi 4513975 gb AI562630.1 AI562630</a>  | TENS2616 T. cruzi epimast...  | <a href="#">26</a> | 6.1        |
| <a href="#">gi 32150480 gb CB923666.1 CB923666</a> | TcAmaP106Run01_A07 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150476 gb CB923664.1 CB923664</a> | TcAmaP104Run02_H11 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 3115441 gb AA952345.1 AA952345</a>  | TENS1283 T. cruzi epimast...  | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150482 gb CB923667.1 CB923667</a> | TcAmaP106Run01_G05 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 3115544 gb AA952448.1 AA952448</a>  | TENS1386 T. cruzi epimast...  | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150472 gb CB923662.1 CB923662</a> | TcAmaP102Run01_A11 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150468 gb CB923660.1 CB923660</a> | TcAmaP110Run01_C04 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150486 gb CB923669.1 CB923669</a> | TcAmaP108Run01_H01 Trypa...   | <a href="#">25</a> | 8.0        |
| <a href="#">gi 32150484 gb CB923668.1 CB923668</a> | TcAmaP108Run01_F07 Trypa...   | <a href="#">25</a> | 8.0        |

# Anatomía de un reporte de BLAST


## High scoring pairs (HSPs)

### Subject

\$gi = "132150256"  
\$gb = "CB923560"  
\$version = "1"  
\$desc = "TcAmaPI03Run01\_C08 ..."  
Longitud = "653"

### HSP info

\$score = "58.9"  
\$expect = "7e-10"  
\$identity = "24%"  
\$similarity = "43%"  
\$frame = "+1"

 >gi|132150256|gb|CB923560.1|CB923560 TcAmaPI03Run01\_C08 Trypanosoma cruzi  
Trypanosoma cruzi cDNA 5' similar to activated protein  
kinase C receptor homolog.  
Length = 653

Score = 58.9 bits (141), Expect = 7e-10  
Identities = 34/138 (24%), Positives = 60/138 (43%), Gaps = 7/138 (5%)  
Frame = +1

Query: 544 DGNIAVWDLHNEILVRQFQGHTDGASCIDISPDGSLWTGGLDNTVRSWDLREGRQLQQH 603  
D + VWD+ + L+ +GHT+ + +SPDGS + D R WDL +G L +  
Sbjct: 28 DNLVKVWDIASGRLLTDLKGHTNYITSVTVSPDGS LCASSDKDGVARLWDLTKGEALSEM 207

Query: 604 DFSSQIFSLGYCPTGDWLA VG MEN-----SHVEVLHASKPKDYQLHLHESCVLSLRFA 656  
+ I + + P W+ E + + V+ P+ Q +S+ ++  
Sbjct: 208 AAGAPINQICFSPNRYWMCAATEKGIRIFDLENKDVIVELAPEAQKSKKTPECMSIAWS 387

Query: 657 ACGKWFVSTGKDNLLNAW 674  
A G S DN++ W  
Sbjct: 388 ADGNTLYSGYTDNVIRVW 441

Score = 50.4 bits (119), Expect = 2e-07  
Identities = 46/167 (27%), Positives = 70/167 (41%), Gaps = 15/167 (8%)  
Frame = +1

Query: 445 TKYVYTGG-KGCVKVWDISQPGNKNPVSQLDCLQRDNVIRSVKLLPDGRTLIVGGEASNL 503  
T + +GG VKVWDI+ D NYI SV + PDG +  
Sbjct: 1 TPLIVSGGWDNLVKVWDIASGRLLT-----DLKGHTNYITSVTVSPDGS LCASSDKDGV 165

Query: 504 SIWDLASPTPRIKAELTSAAPACYALAI SPDSKVCFS-----CCSDGNIAVWDLHNEI 556  
+WDL K E S A +P +++CFS ++ I ++DL N+  
Sbjct: 166 RLWDLT-----KGEALSEMAAG-----APINQICFSPNRYWMCAATEKGIRIFDLENK 312

Query: 557 LVRQFQGHTDGAS-----CIDI--SPDGSRLWTGGLDNTVRSWDLRE 596  
++ + S C+ I S DG+ L++G DN +R W + E  
Sbjct: 313 VIVELAPEAQKSKKTPECMSIAWSADGNTLYSGYTDNVIRVWSVSE 453



# Anatomía de un reporte de BLAST

## Footer

### Estadísticas para esta corrida

Base de datos  
Parámetros estadísticos  
Matriz  
Penalties  
Detalles sobre lo que hizo el algoritmo

Database: GenBank non-mouse and non-human EST entries  
Posted date: Jun 23, 2003 8:07 PM  
Number of letters in database: 2,039,974,386  
Number of sequences in database: 4,227,249

| Lambda | K     | H     |
|--------|-------|-------|
| 0.315  | 0.134 | 0.405 |

| Gapped<br>Lambda | K      | H     |
|------------------|--------|-------|
| 0.267            | 0.0410 | 0.140 |

Matrix: BLOSUM62  
Gap Penalties: Existence: 11, Extension: 1  
Number of Hits to DB: 4,102,102  
Number of Sequences: 10582263  
Number of extensions: 69479  
Number of successful extensions: 630  
Number of sequences better than 10.0: 49  
Number of HSP's better than 10.0 without gapping: 572  
Number of HSP's successfully gapped in prelim test: 0  
Number of HSP's that attempted gapping in prelim test: 0  
Number of HSP's gapped (non-prelim): 625  
length of query: 719  
length of database: 1,456,088  
effective HSP length: 86  
effective length of query: 633  
effective length of database: 505,530  
effective search space: 320000490  
effective search space used: 320000490  
frameshift window, decay const: 50, 0.1  
T: 13  
A: 40  
X1: 16 (7.3 bits)  
X2: 38 (14.6 bits)  
X3: 64 (24.7 bits)  
S1: 41 (21.5 bits)  
S2: 53 (25.0 bits)

# Analizando un reporte de BLAST

- Nuestro programa ya leyó el reporte
- Y almacenó los valores que le pedimos en distintas variables
- Ahora podemos hacerle hacer lo que querramos:
  - (en pseudocódigo):
- **if** \$score < 100 { read next report }  
**else** { **print** \$accession }
- **if** \$description =~ "proteinase" { **print** \$accession } **else**  
{ read next report }
- **if** \$score < 100 **AND** \$description =~ "proteinase"  
{ **print** \$accession }  
**else** { read next report }

# Módulos de software reusables

- **Resumiendo:**
  1. nuestro programa tiene que poder leer el reporte **(FACIL)**
  2. identificar dentro del reporte distintos elementos y almacenarlos en variables **(MAS COMPLICADO)**
  3. tomar decisiones en base a los valores contenidos en las variables y realizar acciones (imprimir algo en pantalla, almacenar datos en un archivo, base de datos, etc.)  
**(Criterio del usuario)**
- **El criterio del usuario es lo que va a hacer que el programa sirva para un fin u otro**
- **Es evidente que los pasos 1 y 2 van a ser necesarios para cualquier programas que intenten procesar reportes de BLAST**
  - solo hay que programarlos una vez
  - modulos reusables (subrutinas)

# Bibliotecas de modulos reusables

- **Perl, Python, Java, C**
  - en general todos los lenguajes proveen bibliotecas de módulos reusables
  - el módulo contiene código que realiza ciertas operaciones
  - no es necesario saber como funciona internamente el módulo para poder usarlo
  - solo necesitamos saber que datos necesita (por ejemplo: una secuencia) y que resultados produce (un valor: 135, una respuesta: SI/NO)
- **En el caso de aplicaciones biológicas**
  - **BioPerl**
  - **BioPython**
  - **BioJava**
  - **Otros**

# Consideraciones prácticas

- La bioinformática es más barata que el trabajo en el laboratorio
- El equipamiento es significativamente más barato que el de un laboratorio de biología molecular
- Los materiales (programas) y reactivos (datos) son en general gratuitos y libremente accesibles
- **Almacenamiento**
  - La cantidad y tipos de bases de datos que se planean instalar
  - La cantidad y tipo de datos que se planean generar
- **Memoria y Procesador**
  - Los requerimientos de los distintos métodos
  - BLAST es principalmente memoria-intensivo
  - HMMER es principalmente procesador-intensivo

# Experimentos *in silico*

- **No son diferentes de experimentos en la mesada**
- Hay que definir una pregunta
- Y una hipótesis nula!
- Definir pasos a seguir y criterios de éxito/fracaso de cada paso
- **Y documentar el proceso!**



# Experimentos *in silico*

- **Particularidades**

- **Uso de cantidades masivas de datos**
- **Necesidad de entender el funcionamiento de programas (limitaciones, ventajas, compromisos)**
- **Necesidad de escribir programas**
  - O interaccionar con bioinformáticos

# Bibliografía sugerida

- **Developing Bionformatics Computer Skills**
  - **O'Reilly & Associates**
- **Bioinformatics. Sequence and genome analysis.**
  - **CSHL Press**
- **Bioinformatics, a practical guide to the analysis of genes and proteins**
  - **Wiley InterScience**