Proyecto de Sistemas Informáticos
Facultad de Informática
Curso 2005 - 2006

Desarrrollo de una plataforma de análisis de datos en Bioinformática basada en Matlab

Jonás Andradas Arias
MªPaloma de la Fuente Bahón
Carolina García Rodríguez

Dirigido por:
Alberto Pascual Montano

.

Índice

1. Autorización

Por la presente, los autores de este proyecto autorizamos a la Universidad Complutense a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

Jonás Andradas Arias

Paloma de la Fuente Bahón

Carolina García Rodríguez

2. Resumen

Resumen en Español

La Bioinformática es una disciplina que nos permite tratar de almacenar y analizar la gran cantidad de información experimental que existe en la actualidad en el campo de la Biología. Con la aparición de nuevas tecnologías, los volúmenes de información cada vez son mayores y con ello aumenta la complejidad para procesar todos estos datos de los que disponemos. La necesidad de nuevos desarrollos informáticos orientados a la comprensión de datos genéticos cada vez es mayor, ya que las técnicas de análisis existentes crecen a un ritmo mucho menor que la producción de datos.

Este proyecto consiste en el desarrollo de una plataforma de análisis de datos. Dicha plataforma tiene una serie de métodos y algoritmos que permiten el análisis y la visualización de los datos. Lo hemos desarrollado en Matlab porque nos permite que el tiempo de procesamiento de los datos sea menor que si se lleva a cabo con otros lenguajes. Nuestro proyecto va a permitir ampliar y mejorar un toolbox ya existente en Bioinformática, ya que aporta nuevas aplicaciones aún no existentes.

Resumen en Ingles

The Bio-Computer Science is a discipline that allows to try to store and to analyze the great amount us of experimental information that it exists at the present time in the field of the Biology. With the appearance of new technologies, the volumes of information every time are greater and with it it increases the complexity to process all these data which we have. The necessity of new computer science developments oriented to the understanding of genetic data every time is greater, since the techniques of existing analyses grow to a rate much smaller than the production of data.

This project consists of the development of a platform of analysis of data. This platform has a series of methods and algorithms that allow to the analysis and the visualization of the data. We have developed it in Matlab because it allows us that the time of processing of the data is minor who if it is carried out with other languages. Our project is going to allow to extend and to improve toolbox already existing in Bio-Computer Science, since it contributes new existing applications not yet.

3. Palabras clave

Palabras claves para su búsqueda bibliográfica (ordenadas por orden alfabético):

- **BIOINFORMÁTICA** :
  Es una disciplina que se encuentra a caballo entre las matemáticas, la genética y la informática. Nace por la necesidad que ha surgido en las últimas décadas de poder tratar y analizar la inmensa cantidad de datos biológicos de los que disponemos.

- **ENTRENAMIENTO** :
  Para que la reorganización de las neuronas sea visible, es necesario que los pasos realizados en los algoritmos se repitan una y otra vez. A esta repetición del proceso le llamamos entrenamiento.
  Por lo general estaremos entrenando la malla hasta que las neuronas consigan una cierta estabilidad, o bien, cuando se haya ejecutado el algoritmo hasta un máximo número de iteraciones.

- **FSOM (Fuzzy Self-Organizing Maps)** :
  También llamado Mapa Auto-Organizativo Difuso. Este tipo de mapas son una optimización de los Mapas Auto-Organizativos, donde la principal diferencia se encuentra en que los elementos ahora no se organizan por regiones, sino que se estudia la posibilidad de que cada elemento pertenezca a cada una de estas regiones.

- **KERDENSOM** :
  Es otra variante de los Mapas Auto-Organizativos, en el contexto del análisis de datos microarray. KerDenSOM se diseña especialmente para encontrar un sistema de vectores representativos del código con una densidad de la probabilidad tan similar como sea posible a la de los datos de entrada.

- **MATLAB** :
  Matlab es al mismo tiempo un entorno y un lenguaje de programación. Matlab, entendido como entorno, es una plataforma de desarrollo de aplicaciones, donde se puede desarrollar fácilmente un toolbox. Matlab como lenguaje de programción nos permite crear nuestras propias funciones (que serían archivos .m), aprovechando entre otras

cosas las propiedades de las matrices para lograr una ejecución más rápida.

- **MICROARRAY DE ADN (Microarreglo de ADN)** :
  También llamado ARRAY DE ADN (Arreglo de ADN). Es una serie ordenada de muestras de ADN que se pueden estudiar para determinar patrones en la expresión de los genes.

- **PARÁMETRO DE DIFUSIÓN** :
  El parámetro de difusión varía el porcentaje de pertenencia de un dato a cada región.
  Cuanto más pequeño es el parámetro de difusión, más probabilidad tiene el dato de pertenecer al centroide más cercano. Es decir, la relación del parámetro de difusión con el porcentaje de pertenencia de un dato a su centroide más cercano es inversamente porporcional.

- **PARÁMETRO DE SUAVIDAD** :
  El parámetro de suavidad hace que una malla se expanda de manera más o menos ordenada.
  A mayor parámetro de suavidad, más restricciones se van a seguir a la hora de ordenar la red. Pero si el parámetro de suavidad es demasiado grande, puede hacer que la malla intente ordenarse tanto que al final se distorsione.

- **REDES NEURONALES** :
  Son sistemas computacionales, de implementación en hardware o software, que imitan las habilidades computacionales del sistema nervioso biológico, usando un gran número de simples neuronas artificiales interconectadas. Primero la red aprende y clasifica y después podemos utilizarla con otros ejemplos. Las redes neuronales son sistemas muy útiles para la clasificación y el reconocimiento de patrones en grandes grupos de datos.

- **SOM (Self-Organizing Maps)** :
  También llamado Mapa Auto-Organizativo. Es uno de los tipos de redes neuronales que más se ha utilizado. La idea principal de este tipo de mapas es la de organizar los elementos de procesado en regiones, que actuarán como clasificadores de los datos de entrada. Esta organización se hace de manera automática mediante un proceso cíclico de entrenamiento.

- **TOOLBOX** :

  Es un conjunto de herramientas inteligentes para la resolución de problemas en áreas de aplicación específica. Nuestros algoritmos pretenden precisamente ampliar y mejorar un Toolbox ya existente en Bioinformática.

4. Introducción a la Bioinformática

La Biología moderna debe manejarse con una inmensa cantidad de datos que no pueden ser interpretados sin ayuda computacional. De esta manera nace la Bioinformática como la aplicación de las Matemticas y la Informática en el tratamiento de datos biológicos. Consiste en la *recogida, mantenimiento, distribución, análisis y uso* de las inmensas cantidades de información biológica disponibles. [1]

Los estudios que la Bioinformática permiten llevar a cabo, requieren además la colaboración de disciplinas lejanas a la Biología:

- La Informática, como complemento imprescindible, proporcionando y desarrollando sistemas, herramientas y algoritmos.
- La Estadística, es fundamental en el manejo de grandes números y en la elaboración de modelos de muchos parámetros.
- La Física y la Química en la elaboración de modelos y en el análisis de procesos al nivel molecular.

Figura 1: Cadena de ADN

La Bioinformática intenta simplificar y organizar el manejo de grandes volúmenes de datos biológicos, como cadenas de ADN de la figura 1, empleando conocimientos derivados de la Biología Molecular y realizando experimentos que permiten organizar los datos en información y ésta en conocimiento. [3]

En la representación que muestra la figura 2, se distinguen tres disciplinas distintas que son las Tecnologías de la información (Informática), la medicina y la Investigación genómica (Genética), que se interrelacionan dando lugar:

- La medicina molecular que es la convergencia entre la medicina y la genética. Este nuevo área promete el desarrollo de nuevas soluciones diagnósticas y terapéuticas.

- La bioinformática, disciplina que se encuentra en la intersección entre la Genética y la Informática, proporcionando herramientas y recursos para favorecer la Investigación Biomédica.
- Y la informática médica que es el campo de la informática preocupada en el análisis de datos médicos a través de aplicaciones sobre aspectos del cuidado de la salud y medicina.

Las tres disciplinas están empezando a relacionarse y darán lugar a una nueva disciplina, denominada INFORMÁTICA BIOMÉDICA, en la que se contemplen estos enfoques integrados de procesamiento de la información relacionada con las enfermedades.

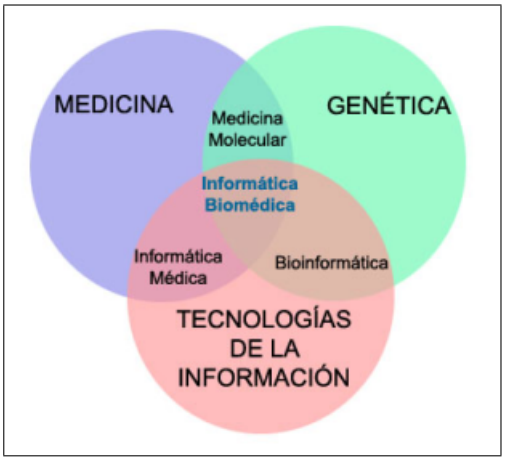


Figura 2: Las tres disciplinas [4]

Las principales áreas de la Bioinformática son:

1. El desarrollo de herramientas que permitan el acceso, uso y actualización de distintos tipos de información biológica.
2. El desarrollo de nuevos algoritmos y soluciones estadísticas para analizar grandes conjuntos de datos y resolver problemas biológicos complejos, tales como predecir la estructura de un gen en una secuencia genómica, predecir la estructura de proteínas, etc [5]

4.1. Análisis de Expresión Génica

El análisis de expresión génica es útil en el diagnóstico y tratamiento de enfermedades ligadas a patrones de expresión genética particulares.

Existen diferentes métodos para estudiar los cambios en la expresión de los genes:

- Los "microarrays"y "macroarrays"de ADN
- Hibridación sustractiva.
- La expresión diferencial de genes (Display diferencial).
- El análisis seriado de la expresión génica.
- Etiquetas de secuencia expresadas (EST).

Ninguno de estos procedimientos es óptimo para todas las aplicaciones. Crear combinaciones de varias de estas técnicas suele ser la mejor opción en muchas ocasiones.

4.2. Los arrays de ADN

Un array de ADN es un conjunto de sondas moleculares fijadas de manera ordenada sobre un soporte sólido.

Un microarray es un dispositivo de pequeño tamaño que tiene inmovilizado material biológico, que permite la automatización simultánea de miles de ensayos encaminados a conocer en profundidad la estructura y funcionamiento de nuestra dotación genética.

En ellos se integran decenas de miles de fragmentos de material genético, de secuencia conocida y de diferente tamaño. Los microarrays son leídos en un escáner, originándose un patrón de luz característico, los datos obtenidos son interpretados mediante un ordenador. Esto permite conocer la estructura y función de la dotación genética del paciente. [6]

Los arrays de ADN que se comercializan actualmente se dividen en:

- **Arrays de alta densidad** :

  Este tipo de arrays permite una gran densidad de integración y la realización de un importante número de ensayos de manera simultánea. Hasta el momento, la empresa líder en arrays de alta densidad es Affimetrix.

- **Arrays de media y baja densidad** :

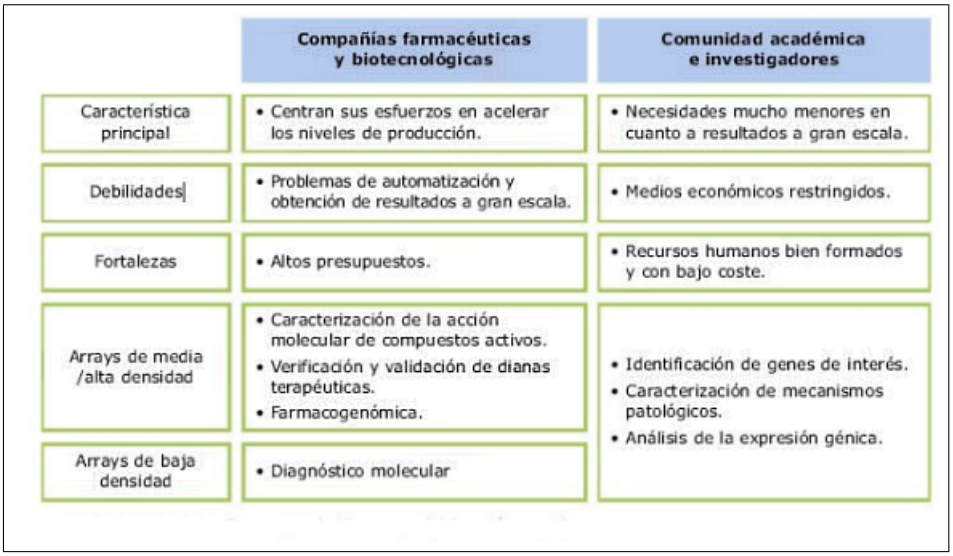
  Suelen utilizarse para trasladar a una escala más práctica los resultados obtenidos con arrays de alta densidad. Son importantes cuando lo que se busca es fiabilidad y flexibilidad.

Los arrays de ADN son especialmente útiles para la clasificación de tumores.

Consideraremos macroarrays aquellos arrays en los que las sondas se depositan e inmovilizan sobre un soporte tipo membrana, generalmente de nitrocelulosa o nailon. Hay una gran variedad y son más baratos que los microarrays. La desventaja es que los macroarrays son radiactivos y requieren mayor volúmen de muestra. [7]

La característica principal de los microarrays es la gran cantidad de sondas que pueden fijarse sobre el soporte sólido pudiéndose alcanzar hasta 60.000 sondas en un único soporte.

En la siguiente figura 3 se muestra los distintos usuarios de la Tecnologías de Microarrays:



	Compañías farmacéuticas y biotecnológicas	Comunidad académica e investigadores
Característica principal	• Centran sus esfuerzos en acelerar los niveles de producción.	• Necesidades mucho menores en cuanto a resultados a gran escala.
Debilidades	• Problemas de automatización y obtención de resultados a gran escala.	• Medios económicos restringidos.
Fortalezas	• Altos presupuestos.	• Recursos humanos bien formados y con bajo coste.
Arrays de media /alta densidad	• Caracterización de la acción molecular de compuestos activos. • Verificación y validación de dianas terapéuticas. • Farmacogenómica.	• Identificación de genes de interés. • Caracterización de mecanismos patológicos. • Análisis de la expresión génica.
Arrays de baja densidad	• Diagnóstico molecular	

Figura 3: Usuarios de la tecnología.

La alteración permanente en los patrones de expresión génica está íntimamente relacionada con mecanismos evolutivos adaptativos, que producen modificaciones de algunos rasgos fenotípicos y de rasgos de especies animales de importancia económica, así como con el desarrollo de distintos tipos de enfermedades, entre otros el cáncer, de ahí la importancia de analizar niveles de expresión génica de genes implicados en el desarrollo de estos fenotipos adaptativos y patológicos.

El uso de microchips de ADN (también denominados microarrays de ADN) constituye una auténtica revolución en el campo de la Bioinformática, ya que estos dispositivos permiten monitorizar la actividad de miles de genes en un solo ensayo, mientras que hasta

hace pocos años estaba limitado a un pequeño número de genes al mismo tiempo.

Los resultados obtenidos con microarrays de ADN se deben procesar utilizando datos almacenados, por lo que la ausencia de estandarización hace muy difcil la aplicación de programas informáticos universales.

4.2.1. Métodos de análisis no supervisados

Los métodos de aprendizaje no supervisado son conjuntos de técnicas que agrupan los datos en función de una distancia, sin utilizar ningún tipo de información externa para organizar los grupos. Dependiendo de la forma en la que los datos son agrupados, podemos distinguir dos tipos de aprendizaje:

1. **Aprendizaje jerárquico** : Está basado en una matriz de distancias. Establece pequeos grupos de genes que tienen un patrón de expresión común y posteriormente construye un árbol. El árbol (ó dendograma), establece una relación ordenada de los grupos previamente definidos y la longitud de sus ramas es una representación de la distancia entre los distintos nodos del mismo.

   La estrategia general que utiliza el aprendizaje jerárquico consiste en separar cada gen en un nodo diferente, calcular la distancia entre los dos genes mas próximos y los juntan en un cluster. Entonces se vuelve a calcular la matriz de distancias sustituyendo los dos patrones que se han unido por el promedio de ambos. En cada paso, los algoritmos son capaces de juntar los genes no sólo de dos en dos sino muchos más a la vez. La distancia del nuevo cluster formado al resto de los elementos de la matriz depende de la estrategia utilizada.

2. **Aprendizaje no jerárquico** : En este caso los algoritmos comienzan a calcular la matriz de distancias a partir de un número predefinido de clusters y van recolocando de forma iterativa los genes en los diferentes grupos hasta minimizar la dispersión interna de cada cluster. Los dos algoritmos más representativos de este tipo de aprendizaje son:

   - K-Medias: es un algoritmo que comienza con una muestra de "k"genes elegidos al azar de la matriz original de datos. Cada uno de ellos se utiliza como el centroide

inicial de los "k" clusters que se van a formar. La matriz de distancias se calcula desde dicho centroide hasta cada uno de los genes de la matriz de datos y cada uno de ellos será asignado de esta forma al centroide más cercano. Entonces la matriz de distancias se recalcula reemplazando cada centroide por la media de los genes asignados a él y el algoritmo repite entonces el proceso anterior. El mapa de clusters que ofrece este algoritmo carece de topología.

- SOM: los mapas auto-organizados (Self-Organising Maps) son redes neuronales. El algoritmo permite, de forma iterativa, que los patrones más parecidos se vayan juntando entre sí y alejándose de aquellos otros que son mas diferentes. Este tipo de algoritmos son mas fiables y robustos puesto que se basan en redes neuronales que por definición son capaces de trabajar con grandes cantidades de datos con ruido. Sin embargo, no carece de ciertos inconvenientes. SOM es una herramienta particularmente útil en el tratamiento de datos procedentes de series temporales.

El gran problema que presentan estos métodos no jerárquicos es que al no generar un árbol no permiten hacerse una idea de la representación espacial de los genes, la cual suele ofrecer un conocimiento intuitivo de cómo analizar los datos de microarrays.

4.2.2. Métodos de análisis supervisados

En este caso si que se asume cierto tipo de información previa, sabemos con cierta exactitud lo que tenemos y ahora lo que nos interesa encontrar es qué marca la diferencia entre nuestros grupos, así como la posibilidad de que con esas diferencias ó patrones podamos clasificar nuevas muestras que puedan llegar a nuestras manos.

4.3. La hibridación sustrativa

Los métodos de hibridación substractiva tienen una gran capacidad para aislar genes de función relacionada sin tener ningún tipo de conocimiento previo. Además no requieren equipos especializados de detección y análisis.

El procedimiento general consiste en la hibridación de ADNc provenientes de una muestra prueba (tester) con un exceso de ARNm

proveniente de una muestra control (driver). Los transcriptos expresados en ambas muestras (tester y driver) forman moléculas de ARNm/ADNc híbridas, mientras que las secuencias de ADNc que están presentes únicamente en la muestra tester permanecen como hebras simples. Las moléculas de hebras simples y dobles se separan usandocromatografía en hidroxilapatita. Los ADNc expresados diferencialmente pueden entonces ser recuperados y clonados o usados directamente como sondas para analizar una biblioteca.

4.4. La expresión diferencial de genes

Es una técnica simple, versátil y eficaz. Se emplea actualmente en los estudios de expresión de genes en muestras cuyo genoma no es conocido y, por lo tanto, no puede analizarse mediante "micro arrays". No obstante, presenta varias inconvenientes: el elevado número de análisis necesarios para estudiar los cambios totales en la expresión de genes de una muestra y la cantidad elevada de falsos positivos.

4.5. El análisis seriado de la expresión génica.(SAGE)

Con el empleo de esta técnica, los clones de ADNc son digeridos al azar y secuenciados sobre una mezcla de fragmentos de ADNc obtenidos a partir de diferentes transcritos. Más tarde estas secuencias se comparan con las existentes en los bancos de genes, lo que nos permite identificar secuencias desconocidas hasta ese momento. Posteriormente, mediante la determinación de la frecuencia de aparición de estos genes se pueden calcular los niveles de ARNm de cada gen.

Las aplicaciones del SAGE son múltiples. El análisis seriado de la expresión génica ha sido utilizado de forma satisfactoria para comparar cuantitativamente los perfiles de expresión génica de una célula normal y una célula cancerosa.

4.6. Etiquetas de secuencia expresadas.(EST)

Las etiquetas de secuencia expresada son pedazos pequeños de secuencia del ADN. Es un procedimiento en el que se seleccionan aleatoriamente clones de ADNc de bibliotecas obtenidas a partir de ARNm y se secuencian de forma parcial. Estos clones fueron denominados "etiquetas"de secuencias expresadas (EST).

Los clones de ADNc son seleccionados al azar y secuenciados. Seguidamente, el resultado de la secuenciación se compara con las

secuencias almacenadas en las bases de datos de genes disponibles en Internet, e inmediatamente se seleccionan las secuencias nuevas, que no han sido informadas en la literatura internacional, ni en estas bases de datos con anterioridad.

Esta tecnología permite descubrir nuevos genes y mapear sus posiciones en los cromosomas. Su aplicación nos deja determinar el perfil de expresión génica de una célula. Durante la década de 1990, el procedimiento de los EST desempeñó un papel primordial, contribuyendo en gran medida a la comprensión del genoma humano, gracias a que ha facilitado distinguir entre las secuencias genómicas que se expresan y las que no lo hacen.

Esta estrategia es una forma sumamente eficaz de encontrar genes nuevos. A continuación vamos a nombrar unas ventajas y desventajas:

- **Ventajas** :
  - Muy buenos como marcadores genéticos.
  - Codominantes.
  - Las secuencias se pueden generar rápidamente.
  - Fuente eficiente de secuencias para obtener cebadores para SSR.
- **Desventajas** :
  - El aislamiento del ARNm puede ser complicado.
  - Los intrones, que pueden contener información importante, no son parte del ADNc.
  - Necesidad de facilidades de secuenciación de ADN en gran escala, lo cual es incompatible con los pocos recursos de los pequeños laboratorios de investigación.

5. Matlab The Language of Technical Computing

5.1. Descripcion

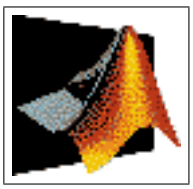


Figura 4: El Simbolo de Matlab

Los fundadores de "The MathWorks"Jack Little y Cleve Moler detectaron la necesidad de crear un nuevo lenguaje computacional más avanzado y desarollado que otros lenguajes. El nombre del programa viene de MATrix LABoratory(laboratorio de matrices), la primera versión estaba escrita en LINPACK y EISPACK. Fue MathWorks quien más adelante reescribió el programa en C. Matlab es al mismo tiempo un entorno y un lenguaje de programación. Permite construir nuestras propias herramientas reusables. Para ello crear nuestras propias funciones en código matlab y guardarlas en ficheros .m. Además los ficheros que forman una colección especializada en un problema o área se agrupan y se denominan toolbox. Si incluimos comentarios antes de la primera orden ejecutable del fichero o función .m estos se mostrarán cuando se llame al comando help con el nombre de nuestro fichero. Es una de las formas de obtener ayuda sobre un comando de Matlab. Así podemos ver a matlab o como una calculadora capaz de hacer cálculos matemáticos de forma más sencilla, o como una plataforma de desarrollo de aplicaciones.

Si nos centramos en Matlab como un entorno nos ofrece prestaciones para el cálculo numérico y visualización de análisis numérico; cálculo matricial; procesamiento de señales y gráficos. Los problemas y soluciones que se realizan en matlab son expresados de la misma manera que se escriben matemáticamente. Se puede observar en el cáculo de matrices, si tenemos definidas dos variables A y B como:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \text{ y } B = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

y queremos realizar una multiplicación de las mismas solo habría que

poner: “A * B”y matlab realizaría la operación fácilmente ya que las matrices son el elemento básico de datos y no requieren dimensionamiento. Es una forma de ahorrar recursos. Mientras que trabajamos en las ventanas Matlab recuerda todos los comandos escritos, y es capaz de salvar las variables para ser usadas posteriormente.

Otra de las características es que existe una gran comunidad de programadores detrás de todo este código, que hace que este programa este en continuo crecimiento y desarrollo. Las nuevas herramientas que van apareciendo se incluyen en los diferentes toolbox si pertenecían a un tema en concreto ya existente, o surge la creación de un nuevo toolbox. Ahora mismo se ofrecen diferentes toolbox para las siguientes áreas de conocimiento: Math and Optimization, Statistics and Data Analysis (Neural Network Toolbox), Control System Design and Analysis (Fuzzy Logic Toolbox), Signal Processing and Communications, Image Processing, Test & Measurement, Computational Biology (Bioinformatics Toolbox), Financial Modeling and Analysis, Application Deployment, Application Deployment Targets y Database Connectivity and Reporting.

Matlab es una aplicación multiplataforma ya que tiene versiones para Windows, Unix/Linux y Macintosh. Los requerimientos de sistemas para las diferentes plataformas son [8]:

- **Windows** :
    - Gráficos: Tarjeta gráfica que soporte OpenGL de 16, 24 o 32 bits
    - **Otras recomendaciones** :
        - MS Windows supported graphics accelerator card
        - MS Windows supported printer
        - MS Windows supported sound card
        - MS Word 2000, MS Word 2002 or MS Word 2003 es necesario para ejecutar MATLAB Notebook
        - MS Excel 2000, MS Excel 2002 or MS Excel 2003 es necesario para ejecutar MATLAB Builder for Excel, Excel Link

	Operating System	Processors	Disk Space	RAM
32-bit MathWorks Products	Windows XP ▪ Service Pack 1 or 2 Windows 2000 ▪ Service Pack 3 or 4 Windows Server 2003	Pentium III Pentium IV Pentium Xeon Pentium M AMD Athlon AMD Athlon MP AMD Athlon XP AMD Athlon 64 AMD Opteron	460MB (MATLAB only)*	512 MB 1024 MB (recommended)
64-bit MathWorks Products	Windows XP x64	Intel EM64T processors AMD64 processors	460MB (MATLAB only)*	512 MB 1024 MB (recommended)

Figura 5: Requisitos para el sistema con Windows

- **Unix/Linux** :

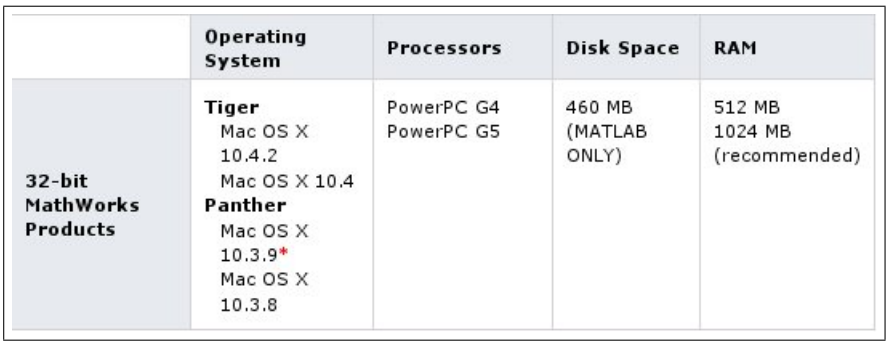
  - **Gráficos** :
    - 16-bit graphics or higher adaptor and display (24-bit recommended)
    - 24-bit graphics display for Sun Solaris
  - **Otras recomendaciones** :
    - Postscript printer
    - Graphics accelerator card

	Operating System	Processors	Disk Space	RAM
32-bit MathWorks Products	Linux - built using ▪ kernel: 2.4.x or 2.6.x ▪ glibc 2.3.2 & above	Pentium III Pentium IV Pentium Xeon Pentium M AMD Athlon AMD Athlon MP AMD Athlon XP AMD Athlon 64 AMD Opteron	460 MB (MATLAB ONLY)	512 MB 1024 MB (recommended)
64-bit MathWorks Products	Linux - built using ▪ kernel: 2.4.x or 2.6.x ▪ glibc 2.3.4 & above	Intel EM64T processors AMD64 processors	460 MB (MATLAB ONLY)	512 MB 1024 MB (recommended)

Figura 6: Requisitos para el sistema con Unix/Linux

- **Macintosh** :

  - **Gráficos** :
    - 16-bit graphics or higher adaptor and display (24-bit recommended)
    - X11 (X server) for Mac OS X
  - Otras recomendaciones: Postscript printer



	Operating System	Processors	Disk Space	RAM
32-bit MathWorks Products	**Tiger** Mac OS X 10.4.2 Mac OS X 10.4 **Panther** Mac OS X 10.3.9* Mac OS X 10.3.8	PowerPC G4 PowerPC G5	460 MB (MATLAB ONLY)	512 MB 1024 MB (recommended)

Figura 7: Requisitos para el sistema con Macintosh

5.2. Bioinformatics Toolbox

El Toolbox de Bioinformática [15]ofrece la posibilidad de explorar ideas, prototipos de nuevos algoritmos, y nuevas aplicaciones de búsqueda, ingeniería genética, y otros proyectos genéticos.

Científicos e ingenieros pueden contestar preguntas, resolver problemas, prototipar nuevos algoritmos, y construir nuevas aplicaciones para diseño y descubrimiento de fármacos, ingeniería genética, y busquedas biológicas. Se pueden usar las funciones básicas de bionformatica que el toolbox contiene o crear mas algoritmos complejos o aplicaciones.

5.2.1. Formatos de ficheros y acesso a bases de datos

Se puede acceder a muchos ficheros estandars de datos biológicos, bases de datos en web y otras fuentes de datos en línea del Bioinformatics Toolbox. Por ejemplo, se podría:

- Leer secuencias de datos de ficheros estandar incluyendo FASTA, PDB, y SCF.

- Leer array de datos desde ficheros como Affymetrix DAT, EXP, CEL, CHP, y CDF ficheros; ImaGene datos del formato de los resultados ; Agilent Feature Extraction Software ficheros; y GenePix GPR y GAL ficheros.
- Interfaz con las bases de datos Web-based databases, como GenBank, EMBL, NCBI BLAST, y PDB.
- Importar datos directamente de NCBI Gene Expression Omnibus Web site usando un simple comando.

5.2.2. Análisis de Secuencia

El Toolbox de Bioinformática proporciona las funciones para el análisis y la visualización genéticos y proteínicos de la secuencia. Los análisis pueden extenderse de alineaciones múltiples de la secuencia al edificio y a los árboles phylogenetic recíprocamente que ven y de manipulaciones.

Alineación de Secuencias El Toolbox de Bioinformática ofrece un sistema comprensivo de los métodos del análisis para realizar parejas de secuencia, perfil de la secuencia, y la alineación múltiple de la secuencia. Éstos incluyen:

- Puestas en práctica de los algoritmos estándares para la alineación local y global de la secuencia, tal como el Needleman-Wunsch, el Smith-Waterman, y los algoritmos modelo perfil-ocultados de Markov.
- Alineación múltiple progresiva de la secuencia.
- Las matrices como representación gráfica de la alineación.
- Matrices que anotan estándares, tales como familias de la matriz PAM y BLOSUM.
- Cálculo de la secuencia del consenso y exhibición de la insignia de la secuencia.

Utilidades de las Secuencias y Estadísticas Podemos manipular y analizar las secuencias para ganar una comprensión más profunda de los datos. Las rutinas de la caja de herramientas de Bioinformática nos permiten:

- Convertir las secuencias de DNA o de RNA a las secuencias del aminoácido usando el código genético.

- Realizar análisis estadístico en las secuencias y buscar patrones específicos dentro de una secuencia.

Visualización de la Secuencia La caja de herramientas de Bioinformática contiene las herramientas para visualizar secuencias y alineaciones. Podemos estudiar mapas lineales o circulares de las secuencias. Se nos permite modificar, explorar en parejas y alineaciones múltiples de la secuencia.

Análisis Genético del Árbol La toolbox de Bioinformática nos permite crear y corregir árboles genéticos. Podemos calcular distancias entre las secuencias alineadas o sin alinear del aminoácido usando una amplia gama de la métrica de la semejanza, tal como Jukes-Cantor, p-distancia, alineación-cuenta, o un método para la distancia definido por el usuario. Se construyen los árboles genéticos usando el acoplamiento jerárquico con una variedad de técnicas, incluyendo el acoplamiento solo, el acoplamiento completo y el UPGMA.

La caja de herramientas de Bioinformática incluye las herramientas para cargar las sub-estructuras calculadoras, y calcular formas canónicas de árboles. A través del interfaz gráfico de usuario (GUI), podemos podar, reordenar, y reutilizar ramas y explorar las distancias.

Análisis de la característica de la proteína El toolbox de Bioinformática proporciona varios métodos para el análisis de la proteína, así como rutinas para calcular las características de una secuencia, tales como composición atómica y peso molecular. La GUI nos deja ver las características a lo largo de la longitud de la secuencia.

5.2.3. Análisis de datos de Microarray y visualización

El análisis de datos de Microarray y la visualización nos permiten analizar y comprender datos de microarray.

Normalización de Microarrays La caja de herramientas de Bioinformática proporciona varios métodos para normalizar datos de microarray. Estos métodos se pueden aplicar al microarray entero o a regiones específicas.

Análisis de Datos y Visualización Usando rutinas del toolbox de Bioinformática podemos clasificar los resultados y representar los datos en visualizaciones estadísticas, tales como mapas del calor.

6. Objetivos del Proyecto

Este proyecto nos permite analizar el caudal de datos de secuencias con el fin de comprender la información amasada en términos de su estructura.

El principal objetivo de este proyecto es desarrollar una herramienta especializada para la consulta, visualización y anáisis de datos genéticos. Dicho objetivo surge por la necesidad de poder tratar en el menor tiempo posible, una inmensa cantidad de datos, que las nuevas tecnologías nos han proporcionado. La Biología ya no es una ciencia puramente experimental, sino que el almacenamiento y la comprensión de la información están en continuo incremento. Por ello la necesidad de optimizarlos

Los algoritmos desarrollados en este proyecto, surgen por la necesidad de nuevos métodos que se adapten a la naturaleza compleja de los sistemas biológicos estudiados en la actualidad. Se han desarrollado por lo tanto métodos que permiten el análisis masivo de datos biológicos de distintos tipos, con el objetivo de ofrecer una aportación al cuello de botella que representa el almacenamiento y procesamiento de inmensas cantidades de datos.

Es importante entender la complejidad y diversidad de los grandes volúmenes de datos de los que disponemos. Comprender dichos datos nos permite obtener información que es de gran utilidad. Así nace la necesidad de implementar métodos que, aplicados a grandes conjuntos de datos, sean capaces de resumir de manera comprensible dicha información, pero preservando las características esenciales de los datos, para poder desarrollar una estudio estructural de los mismos.

De esta manera, y aprovechando funciones de costo bien planteadas matemáticamente, hemos desarrollado la implementación en Matlab de dos métodos de exploración basados en redes neuronales auto-organizativas, que son el FSOM y el KerDensom, ya que son una herramienta muy potente para el análisis exploratorio.

7. Descripción de Algoritmos

7.1. Algoritmo SOM

Descripción

En 1990, Kohonen propuso originalmente el algoritmo SOM (Self-Organizing Maps) como un modelo para auto-organizar los dominios visuales del cerebro. El algoritmo SOM establece una correspondencia entre la información de entrada y un espacio de salida bidimensional, estableciendo relaciones, desconocidas previamente, entre los miembros del conjunto de datos. [9]

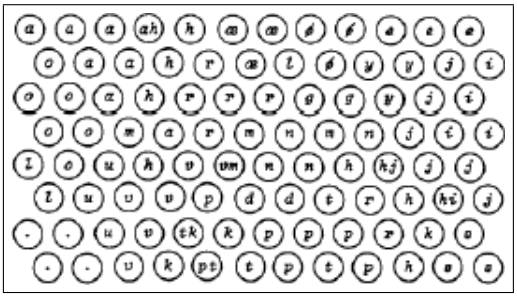


Figura 8: Para el algoritmo SOM

Como acabamos de decir, el espacio de salida suele ser de dos dimensiones, ya que precisamente uno de los objetivos principales de SOM es reducir la dimensionalidad a un espacio menor donde la visualización pueda ser directa. También es posible tener mapas en dimensiones mayora que dos, pero la visualización de los resultados resulta más difícil ó simplemente impracticable.

El principio es bastante sencillo, se recibe información de entrada y lo que sale es una matriz de puntos (neuronas) que llamaremos mapa. La información de entrada se procesa de tal forma que cada una ocupa un lugar en el mapa. Para ello la red debe realizar gran cantidad de cálculos con una serie de datos de ejemplo (entrenamiento de la red).

Las neuronas en la capa de salida están interconectadas entre sí por medio de una malla, topología puede ser: rectangular, hexagonal, toroidal, etc.

La estructura de una red neuronal tipo SOM está representada en la siguiente figura 9, siendo la topología de la malla en este caso

tipo rectangular.

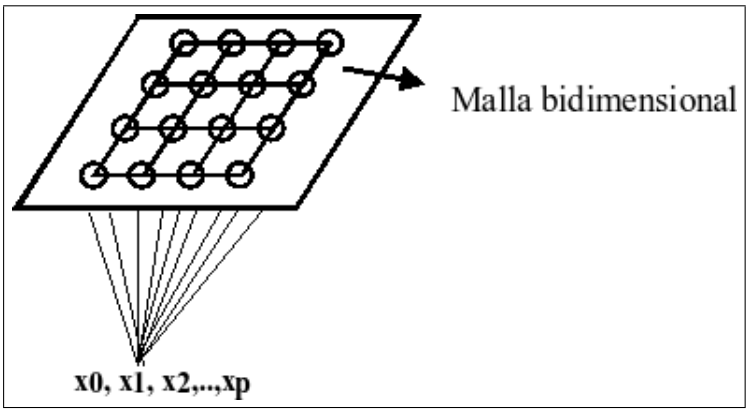


Figura 9: Para el algoritmo SOM

Cada neurona contiene un vector de coeficientes asociado (vector diccionario). El vector diccionario debe tener la misma dimensión que los datos de entrada.

El proceso de entrenamiento se hace de forma no supervisada y consta de los siguientes pasos:

1. Se escoge aleatoriamente un nuevo vector del conjunto de datos que se va a representar.

2. Las neuronas en la capa de salida compiten entre sí y tanto la neurona ganadora, cuyo valor es el más parecido al dato de entrada, como un conjunto de neuronas vecinas actualizan sus valores.

3. Después de un número predeterminado de iteraciones, ó bien cuando los valores de las neuronas se estabilizan, se detiene el proceso.

Una vez entrenada, la red puede recibir datos nuevos que irá ordenando en un determinado lugar del mapa de salida. Lo importante es que la red siempre pondrá el mismo dato en el mismo lugar del mapa, y tenderá a agrupar los datos similares en la misma zona de dicho mapa.

Los datos de entrada con características comunes activarán zonas próximas del mapa. Cuando entra un vector de datos a la red, una neurona de la capa de salida resulta activada, y será aquella cuyo

vector de pesos sea más parecido a la entrada actual (menor distancia euclídea). A esta neurona se la denomina "ganadora". Dicha neurona representa la clase a la que pertenece la entrada.

En la representación gráfica del algoritmo SOM no existen zonas én blanco', es el que mejor aprovecha el espacio visual. Por lo que en la mayoría de los casos no es necesario aplicar ninguna técnica de transformación visual.

SOM es robusto ante la presencia de puntos atípicos, que aparecen frecuentemente en cualquier proceso de toma de datos. En este algoritmo los puntos atípicos afectan únicamente a una sola neurona y en menor grado a sus vecinas. Realizando un estudio de estas neuronas afectadas es posible detectar los datos atípicos, lo que nos permite descartarlos, o analizarlos independientemente.

Limitaciones del SOM:

1. El SOM no define un modelo de densidad en el espacio de datos.
2. El algoritmo de entrenamiento no optimiza una función objetivo.
3. No hay garantía teórica de convergencia.
4. Carece de marco teórico para la selección de parámetros.

Ejemplo del algoritmo

Mediante este ejemplo [9], vamos a ir explicando las distintas fases de entrenamiento del algoritmo SOM.

PASO 1: Tenemos un conjunto de 30 caras creadas artificialmente, con distintos niveles de grises en la nariz, ojos y boca y a las que se las agregó ruido aleatorio.

PASO 2: Inicializamos de manera aleatoria los vectores diccionarios con imágenes del conjunto original de datos. Se ha optado por una red de 5x5.

PASO 3: Después de que la red haya sido entrenada 2000 veces, podemos observar que las imágenes representantes se empiezan a ordenar.

PASO 4: El mapa converge a una solución donde los conjuntos homogéneos de imágenes de entrada son claramente diferenciables, nos han hecho falta 5000 iteraciones en el entrenamiento de la red para llegar a este resultado.

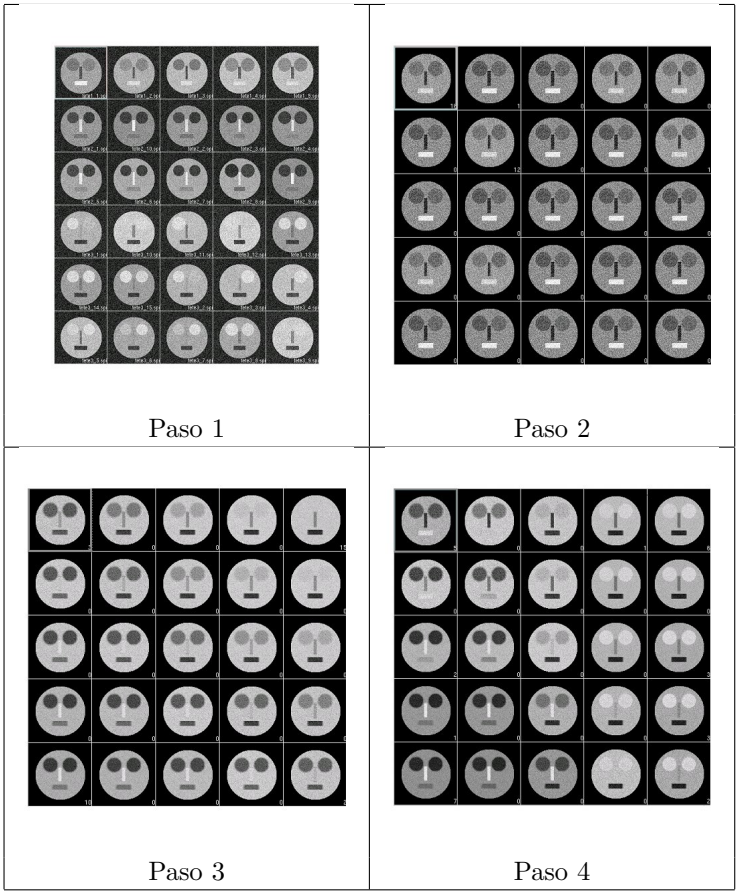


Figura 10: Ordenación con SOM del ejemplo

7.2. Algoritmo FuzzySOM

Mapa auto-organizativo difuso (Fuzzy SOM). La característica central de los sistemas difusos es que están basados en el concepto de particionar la información mediante reglas de inferencia. El estudio de estos sistemas se debe a la necesidad de disminuir el tiempo de ejecución de los procesos, al mismo tiempo que se mantiene la calidad de la representación.

El agrupamiento difuso (fuzzy clustering) se hace en dos grandes pasos:

- Obtener un primer modelo, con un número adecuado de reglas.
- Aplicar una reducción del número de reglas, ya que el primer modelo obtenido tiene cierto grado de redundancia. Dicha reducción se lleva a cabo mediante un algoritmo de transformación.

En este algoritmo los vectores representantes se encuentran distribuidos en un espacio de baja dimensionalidad.

A diferencia del algoritmo SOM, en este algoritmo lo que se estudia es la probabilidad de que los datos pertenezcan a cada una de las regiones de ordenación. En el SOM se examinaba la existencia del dato en uno o en otro grupo de ordenación.

Este algoritmo encuentra una solución que converge a un mínimo local de la función de coste.
Una posible mejora a este algoritmo es aplicarle una estrategia conocida como enfriamiento determinista (*deterministic annealing*). Esta estrategia consiste en ir variando en cada iteración al parámetro de difusión. El parámetro de difusión es el que varía el porcentaje de pertenencia de un dato a cada una de las regiones. Cuanto más pequeño sea, más probabilidad tiene el dato de pertenecer al centroide más cercano. El enfriamiento determinista comienza con valores altos del parmetro de difusión, y en cada una de las iteraciones lo haremos decrecer, hasta que alcance valores muy cercanos a 1.

Gracias a esta técnica, se puede lograr una convergencia hacia el mínimo global de la función de coste, y se pude disminuir el efecto negativo que producen ciertas inicializaciones tanto de la matriz de datos como de la matriz de centroides.

7.3. Algoritmo KerDenSOM

Mapa auto-organizativo basado en la estimación de la densidad de probabilidad (Kernel Density Estimator Self-Organizing Map).

El algoritmo KerDenSOM fue diseñado especialmente para poder encontrar un sistema de vectores representativos del código, con una densidad de la probabilidad tan similar como sea posible a la de los datos de entrada. Está basado en la estimación no paramétrica

de la función densidad de probabilidad, de manera que los vectores representantes tienden a poseer la misma distribución estadística de los datos originales. Esta es, de hecho, su principal ventaja frente a otros mapas auto-organizativos: Conocida la función de de distribución de probabilidad de la cual provienen los datos a estudiar, resulta mucho más sencillo realizar un estudio detallado de los mismos. Si utilizamos una función de núcleo Gaussiana, la función a maximizar l_s sería:

$$l_s = -\frac{np}{2} ln 2c\pi\alpha + \sum_{i=1}^{n} \left(\sum_{j=1}^{c} \exp\left(-\frac{\| X_i - V_i \|^2}{2\alpha} \right) \right) - \frac{\vartheta}{2\alpha} tr(VCV^T) \tag{1}$$

Este algoritmo es muy sensible a los datos iniciales, y por tanto, la mejor forma de maximizar (1) es emplear una técnica de enfriamiento determinista, mediante la cual variamos ϑ entre un valor ϑ_0 máximo (mucha suavidad) y otro ϑ_1 mínimo (poca suavidad), de forma que el valor final será el más adecuado para el problema, calculando el criterio generalizado de validación cruzada aleatoria de Wahba et al.

Como resultado, no sólo obtenemos la malla de los representantes de cada conjunto, sino además, la matriz U de probabilidad de pertenencia de cada uno de los datos de estudio a cada uno de los conjuntos.

8. Implementación. Descripción de los Algoritmos

8.1. FuzzySom

El algoritmo Fuzzy Som es similar al algoritmo Fuzzy c-Means(FCM). La diferencia principal es que en el calculo de los representantes de las regiones(centroides) en las que se clasifican los datos de entrada, se tiene en cuenta el porcentaje de pertenencia de los datos a cada centroide. Además los centroides forman una red neuronal que tratara de recubir los datos de entrada.

En el entrenamiento de la red neuronal se tienen en cuenta dos parámetros el de suavidad y el de difusión. El parámetro de difusion nos da el nivel de porcentaje de pertenencia que queremos que tenga cada dato con cada centroide. Y el parámetro de suavidad nos indica el grado de ordenacion que queremos en la red.

Como hemos indicado antes el algoritmo Fuzzy Som tiene una cierta similitud con el algoritmo FCM, en este proyecto hemos tratado de ampliar el toolbox de bioinformática que proporciona Mathworks para Matlab. El algoritmo es una ampliación de los mapas auto-organizativos y es muy útil para el análisis de grandes cantidades datos biológicos. Por ello a la hora de implementarlo hemos tratado de seguir la línea de implementación que siguieron los creadores del toolbox.

Como el algoritmo entrena una red neuronal se investigó el código de redes neuronales que ofrece Matlab, pero este código no se ajustaba y era difícil de modificar para poder usarlo en la implementación del Fuzzy Som. Se tomo la decisión de seguir el estilo de implementación usado para el algoritmo FCM en el que se parte de un fichero principal donde se recogen los parámetros que se necesitan para entrenar la red. Despues se tiene un fichero para la inicialización de cada variable y un fichero donde se realizan los cálculos en cada paso o iteración que tiene el algoritmo. Además ha sido necesario implementar una nueva función que pinte el resultado de lo que se ha calculado.

Antes de seguir explicando el método de implementación vamos a observar en la figura 11 el diagrama de flujo del algoritmo para poder entender mejor el código realizado:

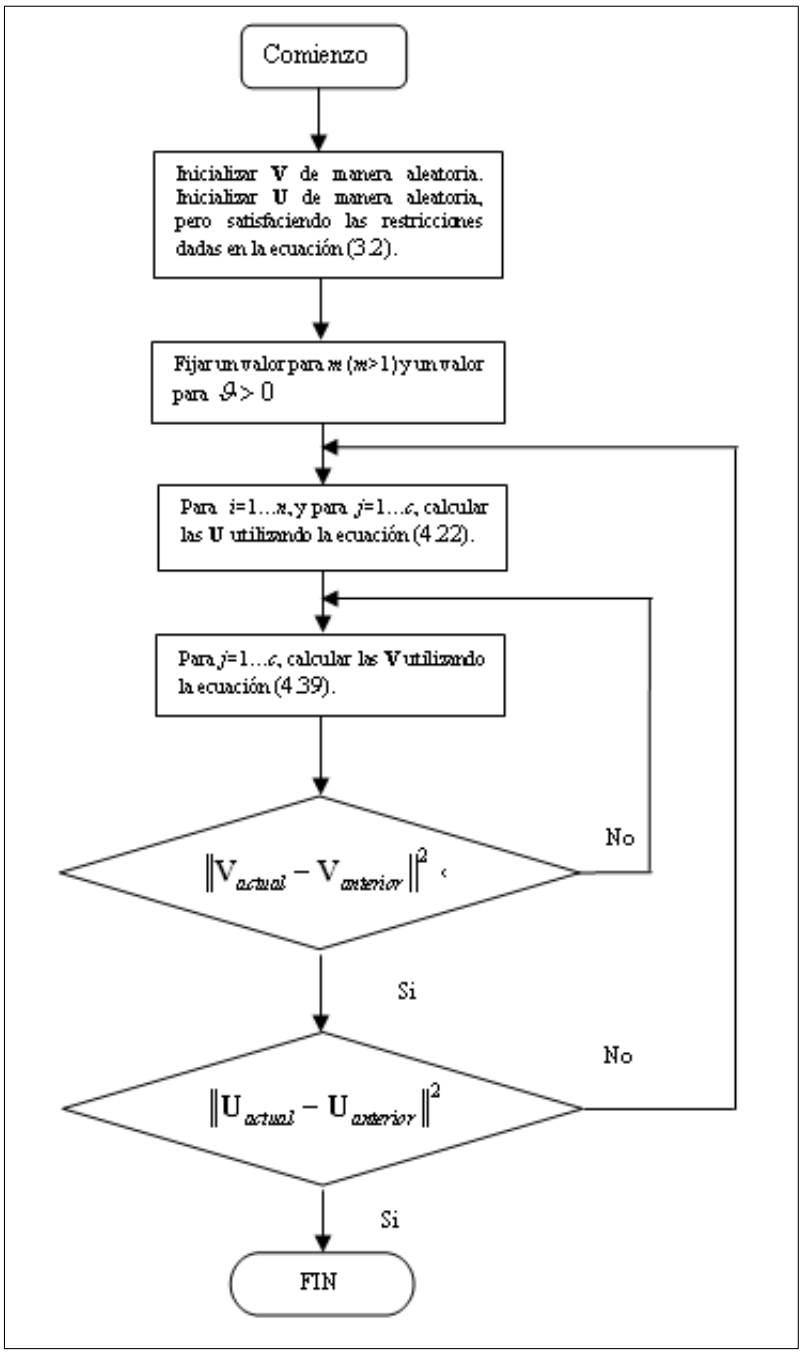


Figura 11: Diagrama de flujo del algoritmo Fuzzy Som

Como se aprecia en la figura para el funcionamiento del algoritmo es necesario fijar un parámetro de difusión (m) y el usuario también ha de dar un valor para el parámetro de suavidad (ϑ) que cumplan las condiciones que se indican. La inicialización de U, matriz donde se almacenan los porcentajes de pertenencia de cada dato a cada centroide, se hace de manera aleatoria pero ha de cumplir las condiciones que indica la siguiente ecuación:

$$\begin{cases} 0 \leq U_{ji} \leq 1 \\ \sum_{j=1}^{c} U_{ji} = 1, \forall i \end{cases}$$

Es entonces cuando desde el fichero principal : "fsom.m"se llama a otro fichero (stepfsomu.m) que se encarga del cálculo de U en cada iteración, el valor de U viene dado por la siguiente ecuación:

$$U_{ji} = \frac{1}{\sum_{k=1}^{c} \frac{\parallel X_i - V_i \parallel^{(2/m-1)}}{\parallel X_i - V_k \parallel^{(2/m-1)}}}$$

Durante el cálculo de U también ha de calcularse mediante un número de pasos la matriz de posiciones de las neuronas que forman la red, V, que se calcula en el fichero "stepfsomv.m". como se puede ver para el calculo de las posiciones de los centroides intervienen los porcentajes de pertenencia y los datos que se están analizando. El cálculo de V se aprecia en la siguiente ecuación:

$$V_j = \frac{\sum_{i=1}^{n} U_{ji}^m X_i + \vartheta \overline{V_j}}{\sum_{i=1}^{n} U_{ji}^m + \vartheta}$$

Para que los bucles paren y no se sigan recalculando U y V se establecen dos condiciones que si se cumplen se paran ambos y el algoritmo entonces finaliza. En el caso de U es:

$$\parallel U_{actual} - U_{anterior} \parallel^2 < \varepsilon$$

Y en el caso de V es:

$$\| V_{actual} - V_{anterior} \|^2 < \varepsilon$$

Además de implementar el algoritmo según el diagrama de flujo también hemos implementado una estrategia para ayudar a una convergencia hacia el mínimo global. Esta estrategia se denomina como hemos nombrado en algunas ocasiones enfriamiento determinista. Consiste en repetir el algoritmo Fuzzy Som un número de iteraciones que el usuario del algoritmo decida, además durante cada iteración se disminuye el parámetro de difusión. Se comienza con un valor elevado y terminara la ejecución con un valor mínimo que también se proporciona. Esta estrategia también ha sido implementada en el fichero: "dafsom.m"

Durante la ejecución de ambas funciones se da la posibilidad de mostrar información de como se esta realizando la ejecución y también se puede obserbar como se entrena la malla.

Ejemplo

Ahora vamos a mostrar algunas capturas de pantalla de un ejemplo en el que la muestra son 194 datos de 2 dimensiones. Los datos los podemos ver en la siguiente figura 12:

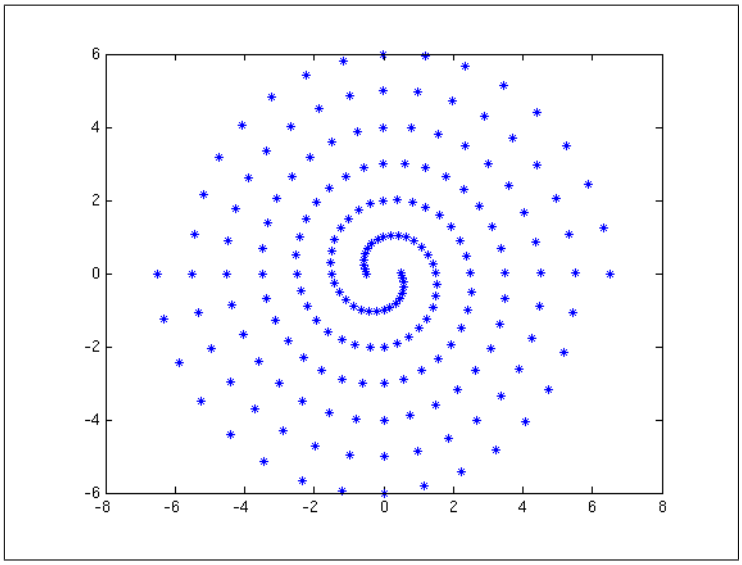


Figura 12: La espiral que usaremos

Como se puede observar la red comienza totalmente desordenada, se le ofrece al usuario inicializar la red neuronal de tres formas diferentes: de forma aleatoria pero en el rango de valores de los datos, escogiendo posiciones de los datos aleatoriamente hasta obtener todos los centroides que forman la red neuronal y se ofrece la posibilidad de que la matriz la introduzca el usuario.

Se realiza una llamada a la función fsom de la siguiente manera:

```
[center,U,obj_kdsom] = fsom(espiral, 10, 8, 0.8,1.8,400);
```

Teniendo en cuenta que espiral es la matriz de los datos, que la red neuronal es de 10x8 y que el parámetro de suavidad es 0.8 y el parámetro de difusión es 1.8 El inicio de como se situa la red teniendo en cuenta que se ha dejado la opción por defecto es esta 13:

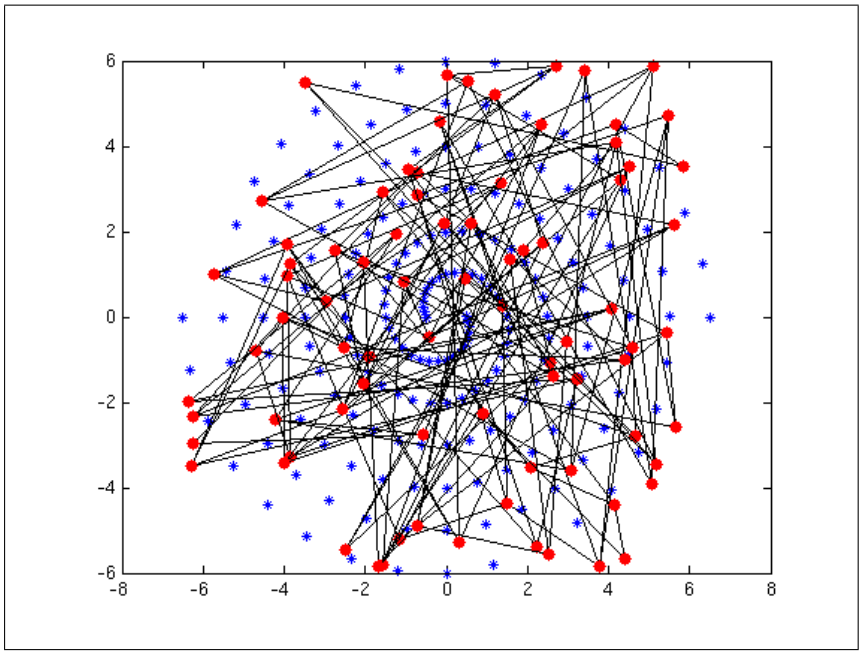


Figura 13: Captura con el inicio de ejecución de fsom

El resultado del algoritomo queda asi 14:

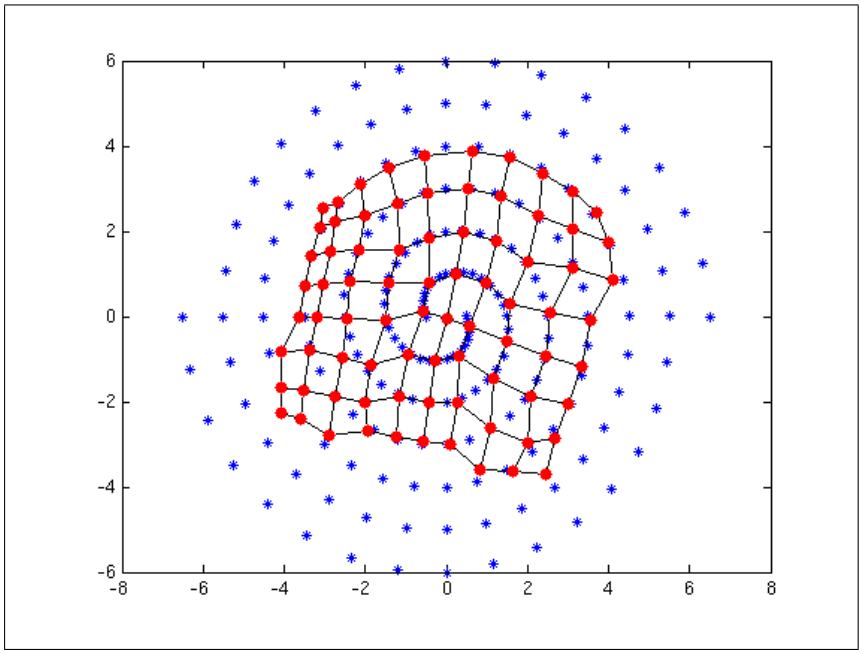


Figura 14: Resultado final del algoritmo

Ahora vamos a mostrar el resultado de aplicar un enfriamiento determinista a estos mismos datos con la siguiente llamada:

```
[center,U,obj_fsom] = dafsom(espiral, 10, 8, .8, 1.02, 2.5, 80);
```

Donde vamos a ir desde un parámetro de difusión de 2.5 a uno de 1.02 en 100 pasos y con un parámetro de suavidad de 0.8 en la siguiente figura 15:

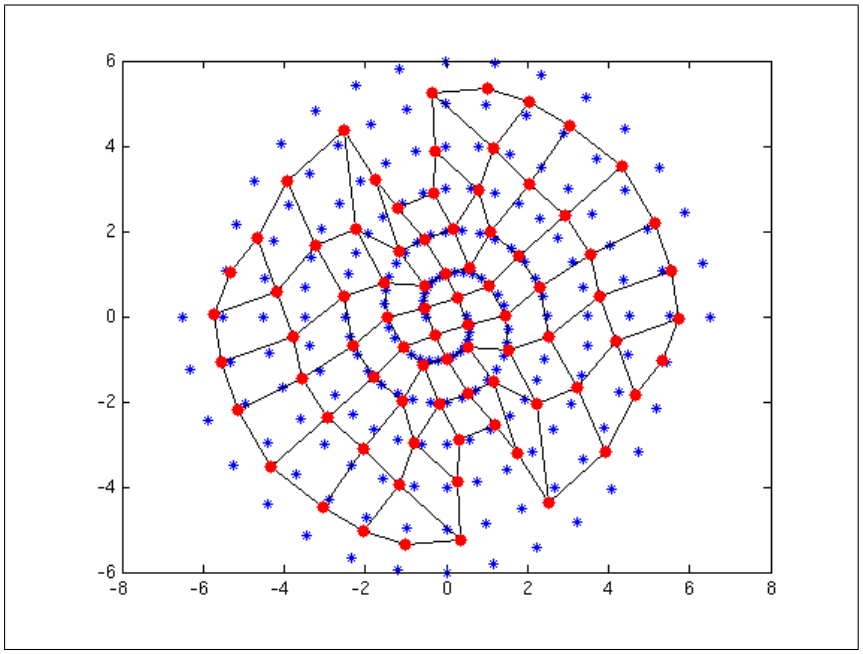


Figura 15: Resultado final del algoritmo con enfriamiento determinista

8.2. KerDenSom

Nos hemos basado en el diagrama de flujo de la tesis [9], para obtener un algoritmo KerDenSOM iterativo, que se puede resumir en 9 pasos, que describiremos a continuación:

1. Inicializamos la malla V, a partir de datos aleatorios en el rango de los datos a estudiar, o bien tomando datos del caso de estudio al azar, o bien a partir de una malla suministrada como parámetro.

2. Inicializamos la matriz de pertenencia U, asignando a cada dato una probabilidad del 100 % de pertenencia al vector V_j, siendo V_j el más similar a dicho dato. De este modo, los datos pertenecen inicialmente a un único representante, siendo la probabilidad de pertenecer a cualquier otro, de 0.

3. Si maximizamos l_s respecto a α, e igualamos a cero la derivada parcial, podemos llegar a la fórmula:

$$\alpha = \frac{1}{np}\left(\sum_{i=1}^{n}\sum_{j=1}^{c} \parallel X_i - V_j \parallel^2 U_{ji} + \vartheta \sum_{j=1}^{c}\sum_{k=1}^{c} C_{jk} V_j^T V_k\right)$$

   a través de la cual obtendremos el α inicial.

4. Repetimos los pasos (5) a (9) *MaxIter* veces, aumentando el contador *Iter* desde 0 hasta alcanzar el valor máximo.

5. Calculamos ϑ para esta iteración:

$$\vartheta = \exp\left(ln(\vartheta_1) - (ln(\vartheta_1) - ln(\vartheta_0))\frac{Iter}{MaxIter}\right)$$

6. Repetimos lo siguiente para V_j con j=1..c hasta que los vectores diccionario cambien muy poco, es decir, se cumpla que $\parallel V_{j_{actual}} - V_{j_{anterior}} \parallel^2 < \varepsilon$

$$V_j = \frac{\sum_{i=1}^{n} U_{ji} X_i + \vartheta \bar{V_j}}{\sum_{i=1}^{n} U_{ji} + \vartheta}$$

7. Calculamos α utilizando la ecuación del paso (3):

$$\alpha = \frac{1}{np}\left(\sum_{i=1}^{n}\sum_{j=1}^{c} \parallel X_i - V_j \parallel^2 U_{ji} + \vartheta \sum_{j=1}^{c}\sum_{k=1}^{c} C_{jk} V_j^T V_k\right)$$

8. Para i=1..n y $j = 1..c$, calculamos U_{ji} con la ecuación:

$$U_{ji} = \frac{K(X_i - V_j; \alpha)}{\sum_{k=1}^{c} K(X_i - Vk; \alpha)}$$

9. Volver al paso (6) mientras que no se cumpla que

$$\| U_{ji_{actual}} - U_{ji_{anterior}} \|^2 < \varepsilon$$

En la siguiente hoja podemos apreciar el flujo del programa que lo podemos ver en la siguiente figura16:

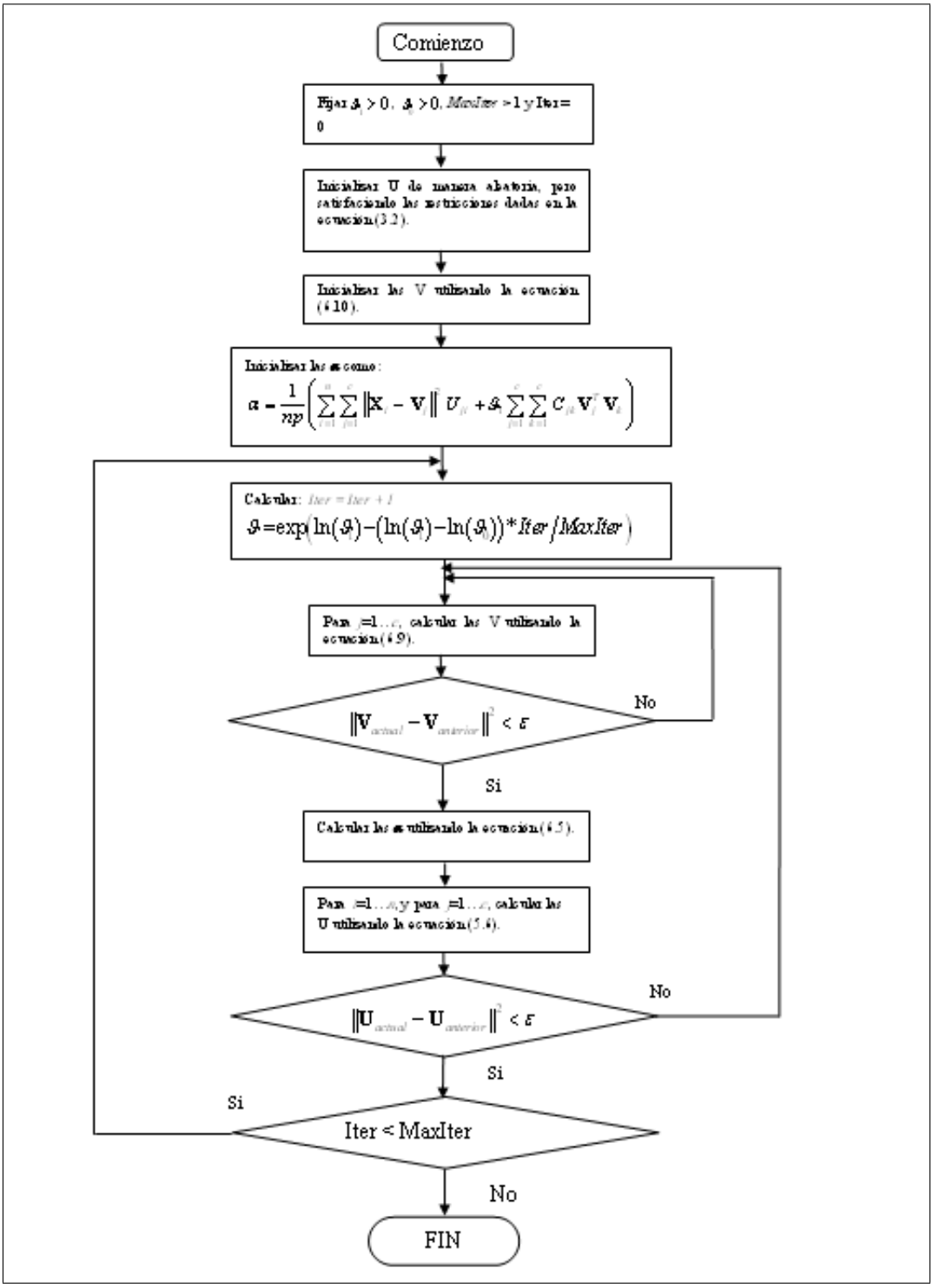


Figura 16: Diagrama de flujo del algoritmo KerDenSom

Tras la ejecución, podemos ver cómo el mapa original de 12x12, con 144 puntos queda reducido a uno 7x7

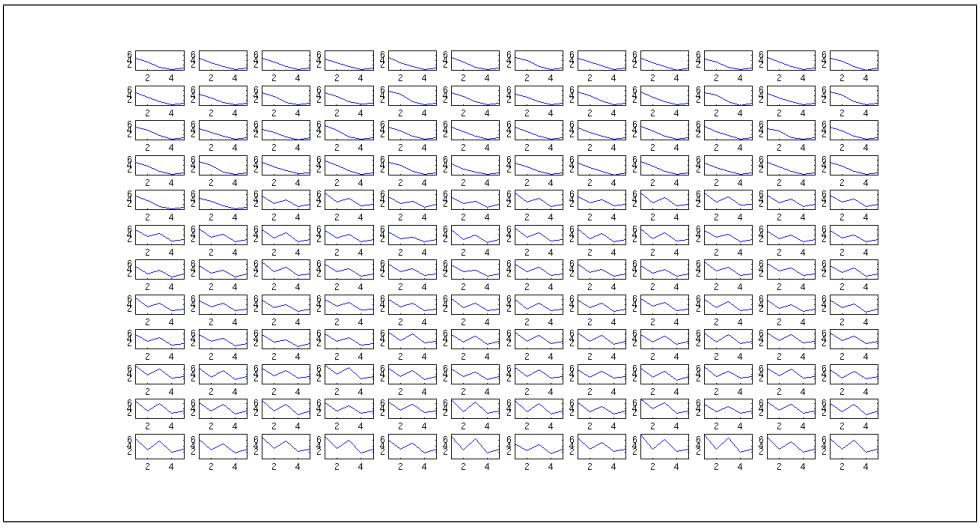


Figura 17: Conjunto de 144 datos iniciales que vamos analizar

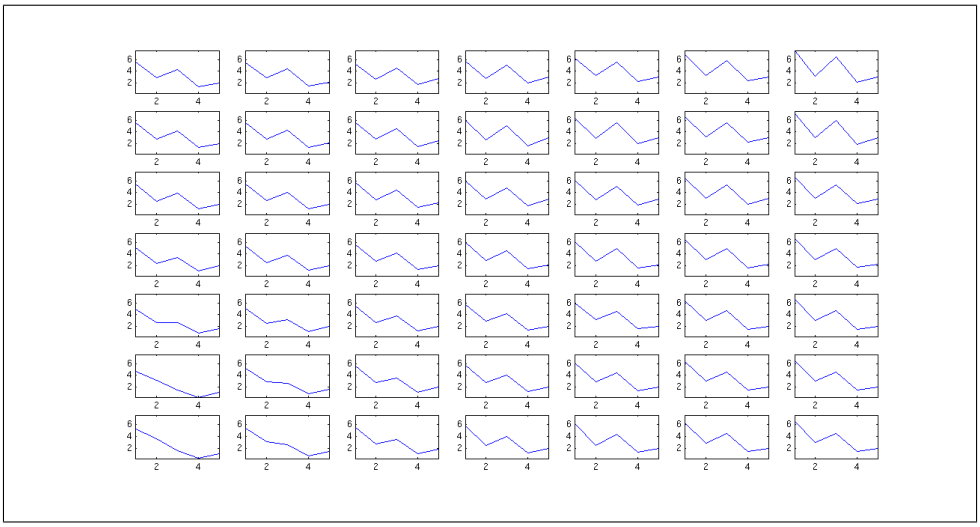


Figura 18: Malla obtenida a partir del anterior conjunto de imágenes

O este ejemplo, en el que se ve cómo la malla se ha ubicado en el espacio para asemejarse y clasificar los puntos en 3 dimensiones:

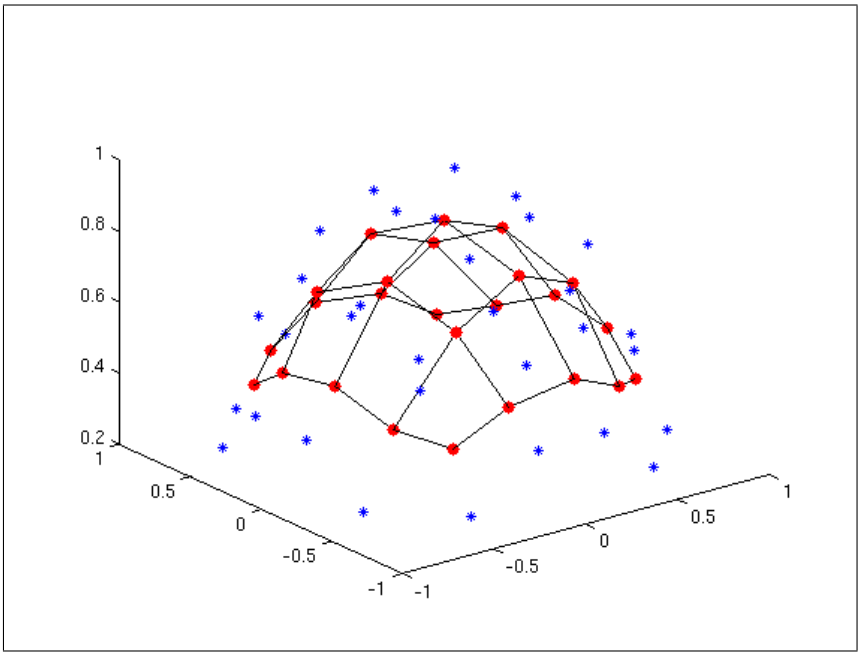


Figura 19: Malla formando una semiesfera

9. Caso de estudio real

En anteriores ocasiones hemos mencionado que la bioinformática ha surgido debido a la gran cantidad de datos biológicos. Datos que si se analizan pueden darnos respuestas a problemas. Hemos tomado de [11] que es una base de datos de la expresión génica para la farmacología molecular del cáncer los datos de un estudio, uno de los estudios que han realizado. En el que usaron 118 compuestos probados en 60 celulas. Podemos ver en la siguiente figura 20 un gráfico del estudio:

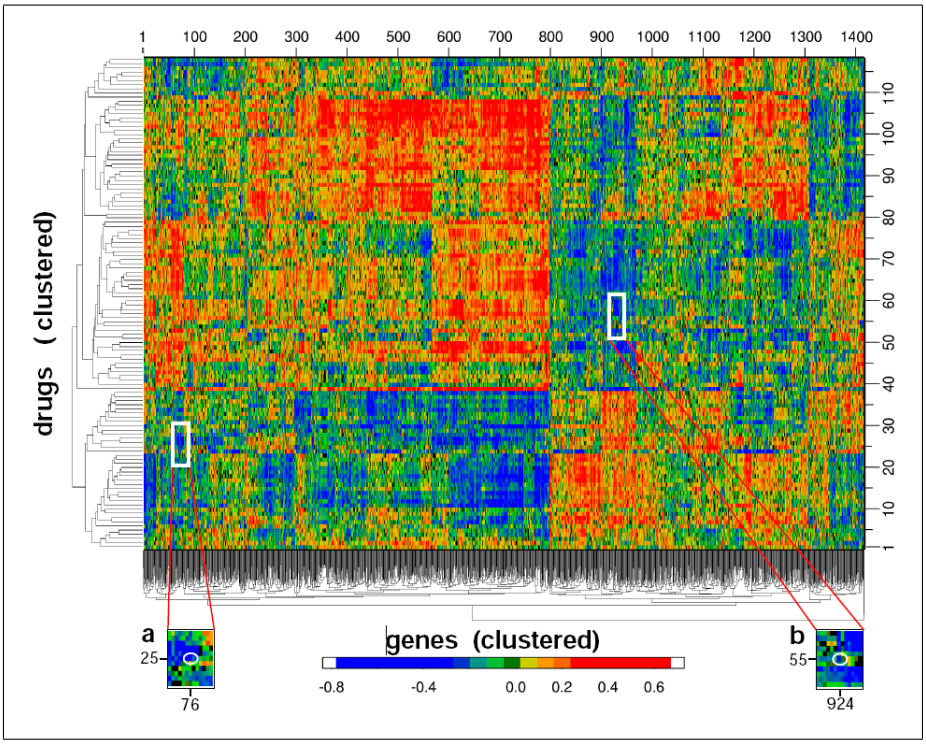


Figura 20: Mapa de imagen arracimado

Como observamos es un mapa de imagen arracimado (CIM) que relaciona los patrones de la actividad de 118 compuestos probados con los patrones de la expresión de 1.376 genes en las 60 variedades de células. Se incluyen además de los niveles de la expresión de los genes para 40 blancos moleculares que determinaron uno a la vez en las células. Un punto rojo (alto coeficiente de correlacin positivo de Pearson) indica que el agente tiende para ser más activo (en el análisis de dos días del SRB) contra las variedades de células que expresan más del gen; un punto azul (alta correlacin negativa)

indica la tendencia opuesta. Los genes racimo-fueron pedidos en base de sus correlaciones con los fármacos (significar-restadas, medio-acoplamiento arracimado con la correlación métrica); los fármacos fueron arracimados en base de sus correlaciones con los genes.

En una de las demos que hemos creado, en concreto para el algoritmo KerDenSom, hemos usado esos datos de la actividad de los 118 fármacos por lo tanto en la tabla de datos que metemos al algoritmo hay 118 fármacos x 60 células. Las columnas delimitadas nombradas en formato del Excel que le metemos y analizamos en la demo contiene:

1. A: Mecanismo de la acción, si está sabido.
2. B: Nombre del fármaco, si está sabido. El número de NSC.
3. C: Número de NSC (es decir un identificador único del fármaco).
4. D-BK: Actividades del fármaco (- Log10 Gi50)

Extraemos los datos de la tabla de Excel, y los cargamos en dos matrices. Despues, recorremos estas dos tablas, sacando los datos necesarios.

```
[numericData, textData] = xlsread('../datos/cancerdata.xls');
giValues = numericData(:,2:end);
drugMechanism = textData(2:end,1);
drugName = textData(2:end,2);
drug = strcat(drugMechanism,'-',drugName);
drugID = numericData(:,1);
cellLine = textData(1,4:end);
tumorTypes = strtok(cellLine,':');
```

Empleamos varias funciones del Toolbox de estadistica para rellenar los valores que faltan en las tablas. Estos valores, que estan marcados como "NaN"(no son valores numéricos), los calculamos empleando las funciones nanmedian, que calcula la media de los valores de las filas o las columnas, ignorando aquellos valores que no estan presentes. De esta forma, rellenamos los datos desconocidos con la media de los demas. Puesto que tenemos 118 datos, vamos a intentar dividirlos en una malla 9x9, aunque podríamos haber elegido otros valores. Usaremos una ϑ_1 de 2.6, y una ϑ_0 de 1.0, en 40 pasos:

```
[center,U,obj_kdsom] = kdsom(giValues, 9, 9, 1.0, 2.6, 40);
```

En la primera imagen mostramos los datos iniciales, al ser 60 células se muestra cada una de ellas en las “cajitas ”:

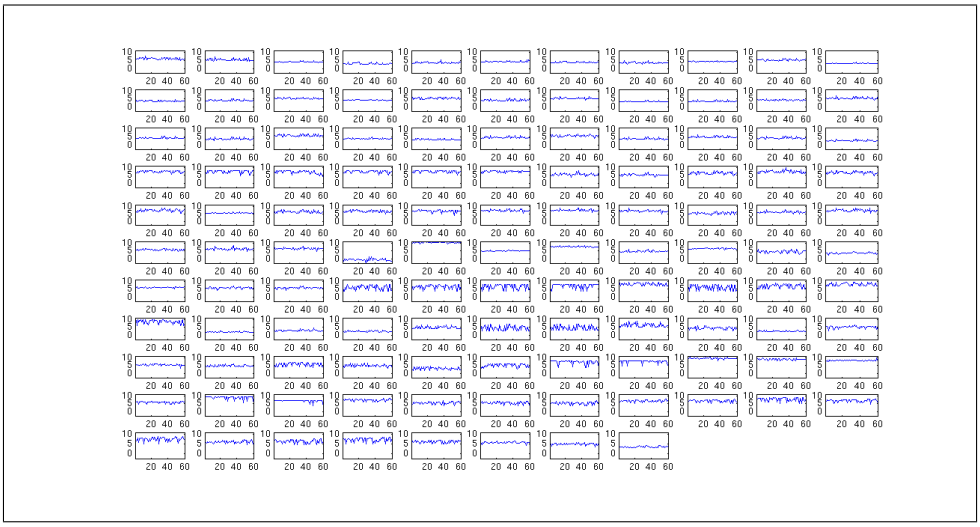


Figura 21: Muestra de los datos pintados con nuestro algoritmo

Y en la siguientes figuras se muestra como quedaría la red neuronal sobre las celulas teniendo en cuenta los datos. Primero la red sin entrenar 22 y despues con la red entrenada 23

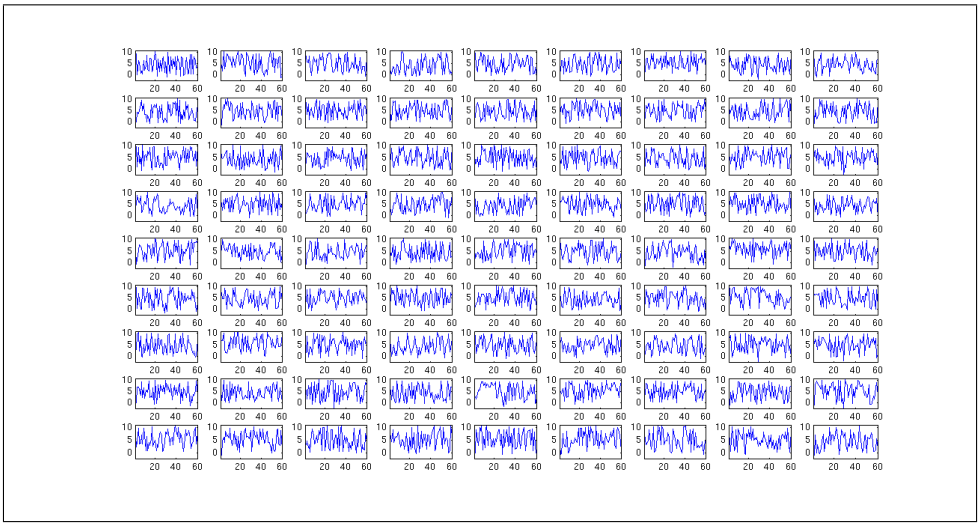


Figura 22: Red neuronal usada antes de entrenar

Como deciamos esta imagen es con la red neuronal ya entrenada.

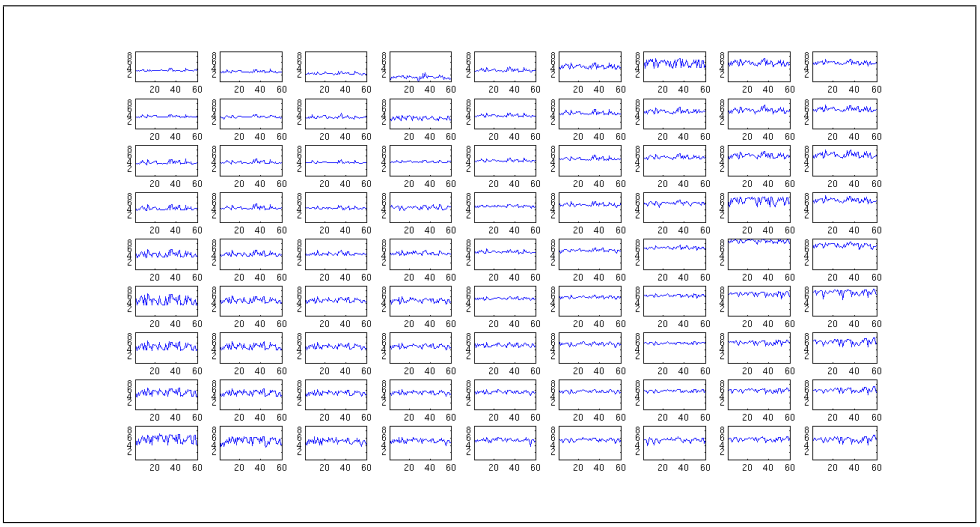


Figura 23: Resultado final despues del entrenamiento con KerDenSom

10. Conclusiones y trabajo futuro

El mundo de las Matemáticas está más unido que nunca al mundo de la Informática. Tanto es así, que el tiempo de ejecución de los algoritmos implementados en este proyecto, hubiese sido mucho mayor en el caso de no disponer de fórmulas matemticas bien planteadas.

Los avances en tecnología y las nuevas investigaciones deben ir de la mano de progresos en el campo de la programación. No sirve de nada tener una mayor cantidad de datos, si no existen los medios necesarios para poder almacenarlos, procesarlos o analizarlos.

Por lo tanto, son necesarias enversiones en nuevos desarrollos de Bioinformática, ya que gracias a este tipo de algoritmos podemos analizar cantidades de datos que de otra manera sería inviable. Con el tiempo, esta cantidad de datos seguirá aumentando, por lo que se debe disponer de medios suficientes para poder procesarlos. No sirve de nada tener una cantidad enorme de datos, si no podemos analizarlos para lograr entenderlos mejor.

Métodos que en un primer momento parece que únicamente sirven para reorganizar una gran cantidad de datos en diferentes regiones, hacen que se esté más cerca de descubrir fármacos para enfermedades tan graves como el cncer.

11. Apéndices

LA NEURONA

La neurona es la célula nerviosa, es la unidad funcional del sistema nervioso (50000 neuronas por cada milímetro del cerebro). Las neuronas pueden ser sensoriales, motoras y de asociación.

- El Soma es el cuerpo de la neurona, contiene al núcleo, y se encarga de recibir toda la información del resto de neuronas y de realizar todas las actividades metabólicas.
- Las dendritas son las entradas de la neurona.
- El axón es la salida y es el que envía impulsos ó señales a otras neuronas.

Una neurona se puede comparar con una caja negra compuesta por varias entradas y una salida. Las señales van de unas neuronas a otras por medio de un proceso químico. El efecto es elevar o disminuir el potencial eléctrico dentro del cuerpo de la neurona receptora. La neurona se dispara cuando su potencial alcanza un cierto umbral, se envía un pulso a través del axón.

LA RED NEURONAL

En el cerebro hay una gigantesca red de neuronas ocultas que realizan todos los cálculos necesarios. Por lo que una red neuronal artificial debe estar compuesta por una gran cantidad de neuronas artificiales.
Entre 1940 y 1950 los científicos comenzaron a estudiar seriamente las redes neuronales, relacionando las neuronas con interruptores digitales (on - off) . Es en 1949 cuando se expuso la idea de que las redes neuronales podían aprender. Pero sólo dos años más tarde apareció la primera Red Neuronal junto con la primera máquina de redes neuronales. Era una red de 40 neuronas artificiales que imitaban el cerebro de una rata. Cada neurona era como una posición de un laberinto y cuando se activaba daba a entender que la "rata"sabía en que punto del laberinto estaba. Las neuronas que estaban conectadas alrededor de la activada, hacían la función de alternativas que seguir por el cerebro, la activación de la siguiente neurona, es decir, la elección entre "derecha"o "izquierda"en este caso estaría dada por la fuerza de sus conexiones con la neurona activada. Pero eso en sí no era inteligencia, ya que la red neuronal nunca llegaría a trazar un plan.

LA NEURONA ARTIFICIAL

Una neurona artificial es un circuito eléctrico que realiza la suma ponderada de las diferentes señales que recibe de otras unidades iguales, produciendo un uno o un cero en la salida, dependiendo de dicha suma con respecto al umbral o nivel de disparo.
La neurona artificial es un dispositivo eléctrico que responde a señales eléctricas. La respuesta la produce el circuito activo o función de transferencia que forma parte del cuerpo de la neurona. Las "dendritas"llevan las señales eléctricas al cuerpo de la misma. Estas señales provienen de censores o son salidas de neuronas vecinas. Las señales por las dendritas pueden ser voltajes positivos o negativos; los voltajes positivos contribuyen a la excitación del cuerpo y los voltajes negativos contribuyen a inhibir la respuesta de la neurona. En un ordenador los datos se guardan en posiciones de memoria aisladas entre sí. De esta manera, cuando se quiere acceder a una posición, se obtiene el dato de esta celda. Las posiciones de memoria colindantes con ésta no aprecian dicha operación. En el cerebro el almacenamiento es muy diferente ya que cuando buscamos una información no hace falta que sepamos dónde se encuentra almacenada, ya que basta con que pensemos en el contenido o significado de la información para que un mecanismo, cuyo funcionamiento nadie conoce, nos proporcione automáticamente no solo la información deseada sino que también las informaciones vecinas, es decir, datos que de una u otra manera hacen referencia a lo buscado. Es decir, los humanos asociamos ideas, interrelacionando contenidos, significados, modelos. La cantidad de información de datos de un cerebro todavía no se ha podido superar artificialmente, tampoco la velocidad.

LA LÓGICA DIFUSA

La lógica difusa o lógica fuzzy es utilizada en algunos sistemas expertos y en otras aplicaciones de inteligencia artificial. En ella las variables pueden tener varios niveles de verdad o falsedad representados por rangos entre el 1 (verdadero) y el 0 (falso). El resultado de una operación se puede expresar como una probabilidad y no necesariamente como una certeza. Lógica difusa o Lógica fuzzy, en informática, forma de lógica utilizada en algunos sistemas expertosy en otras aplicaciones de inteligencia artificial, en la que las variables pueden tener varios niveles de verdad o falsedad representados por rangos de valores entre el 1 (verdadero) y el 0 (falso). Con la lógica fuzzy, el resultado de una operación se puede expresar como una probabilidad y no necesariamente como una certeza. Por ejemplo, además de los valores verdadero o falso, un resultado puede adoptar valores tales como probablemente verdadero, posiblemente verdadero, posiblemente falso y probablemente falso.

12. Agradecimientos

Queremos mencionar a las siguientes personas por la ayuda que nos han proporcionado durante el desarrollo de este proyecto:

- Alberto Pascual Montano, director de este proyecto. Por ayudarnos a comprender los algoritmos y por contestar a los correos de auxilio con tanta rapidez. De verdad, muchas gracias.
- Carlos Andradas Heranz. Por prestarnos su conocimiento matricial que tanta falta nos ha hecho.
- Francisco P. Rodríguez. Por sus colaboraciones con la Interfaz Gráfica.
- Familiares y amigos. Por aguantarnos en esos días en los que no veíamos la luz. Sin vosotros no habríamos llegado hasta aquí.

Referencias

[1] http://www.ub.es/stat/docencia/Biologia/introbioinformatica/Introduccion/Introduccio

[2] http://www.galeon.com/scienceducation/sld001.htm

[3] http://www.bioxeo.com/bioinfo/index.htm

[4] http://bvs.isciii.es/bib-gen/Actividades/curso_virtual/Introduccion/internet.htm

[5] http://www.unia.es/maestrias_unia/bioinformatica/presentaciongeneral.htm

[6] http://www.pdg.cnb.uam.es/cursos/Leon_2003/pages/Genomas_Anal_Anot/index.html

[7] http://external.doyma.es/espacioformacion/eimc/temas/m2t8.pdf

[8] http://www.mathworks.es

[9] Redes neuronales auto-organizativas basadas en optimización funcional. Aplicación en bioinformática y biología computacional. Alberto Domingo Pascual Montano

[10] http://gndp.cigb.edu.cu/Combinatoria%20Molecular/PDF/chapter19.pdf

[11] http://discover.nci.nih.gov/nature2000/

[12] http://www.dig.cs.gc.cuny.edu/www.cnb.uam.es/%257Ebioinfo/NewXmipp/Applicatio

[13] http://xmipp.cnb.csic.es/NewXmipp/Applications/Src/KerDenSOM/Help/kerdensom.h

[14] http://www.lce.hut.fi/publications/pdf/LampinenKostiainen_sompdf.pdf

[15] http://www.mathworks.es/products/bioinfo