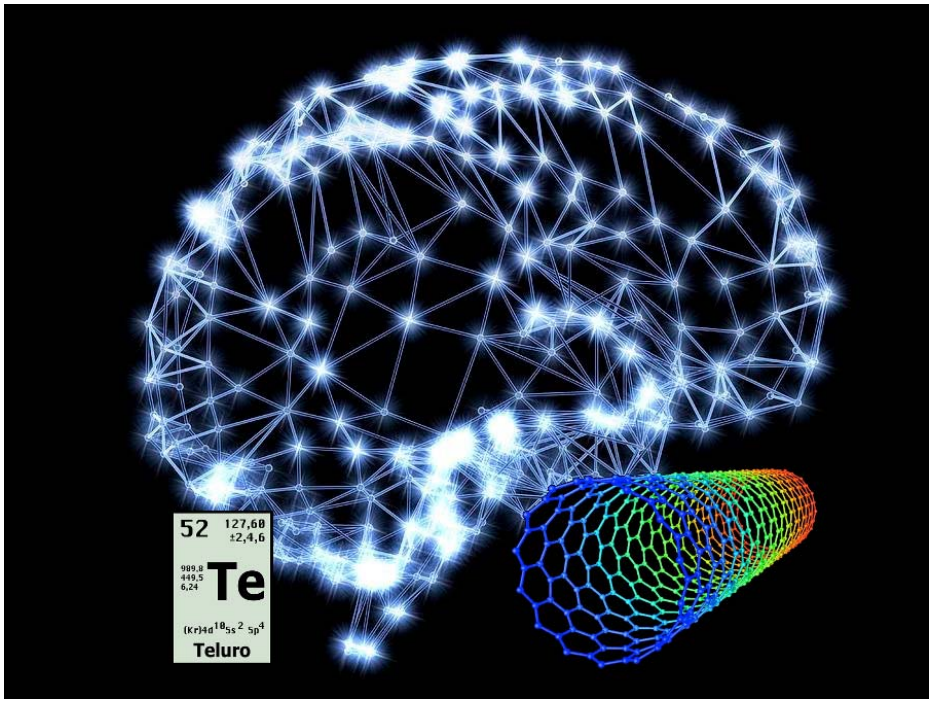


Antonio Díaz Pozuelo

Universitat Oberta de Catalunya

Enrique Lomba García

Instituto de Química Física Rocasolano, CSIC

GPUs y redes neuronales

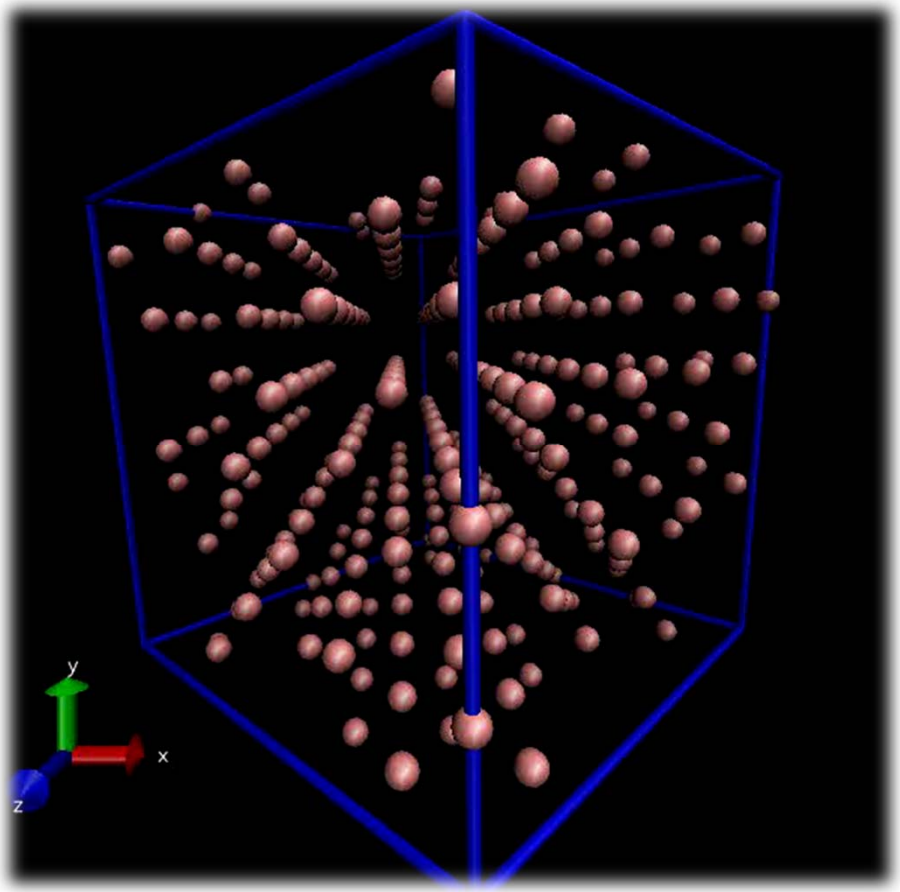
http://github.com/adpozuelo/HDNNP

¿ Cuándo y por qué ?

- En el estudio de la materia condensada puede ser necesaria la simulación de conjuntos masivos de átomos (10^4 átomos) y tiempos “largos” (varios ns)
  - Transiciones de fase: fusión, vitrificación,…
  - Sistemas biológicos
- No siempre es posible el uso de potenciales efectivos clásicos
  - Falta de transferibilidad
  - Formación/destrucción de enlaces
  - Relevancia del comportamiento electrónico, e.g. semiconductores

Density Functional Theory ?

- Alternativa ab initio más común para:
  - Determinar la energía electrónica de sistemas atómicos y moleculares
  - Determinar las fuerzas interatómicas
  - Realizar simulaciones de Dinámica Molecular *on-the-fly*



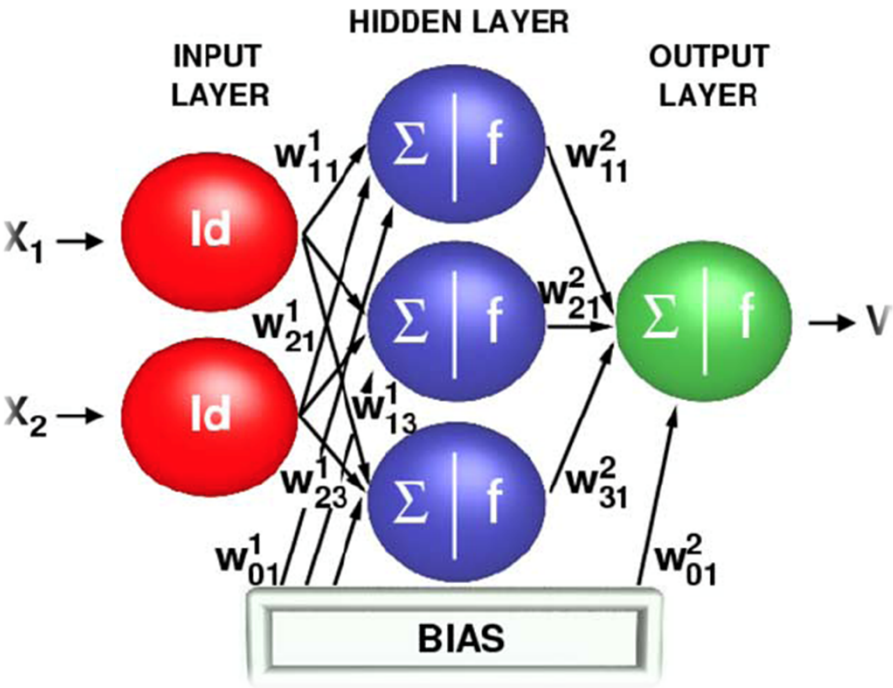
Simulación de 256 átomos de Te, realizada en un i7-3.6 GHZ. VASP version 5.4.1 (build 9.02.16) sobre una GeForce GTX 980 2048 cores Compute Capability 5.3. 1ps = 20 horas de cálculo

Problemas

- A pesar de la paralelización de los códigos (MPI, GPU), los tiempos de cálculo son prohibitivos
- El **tiempo de CPU escala con N^4** !!!

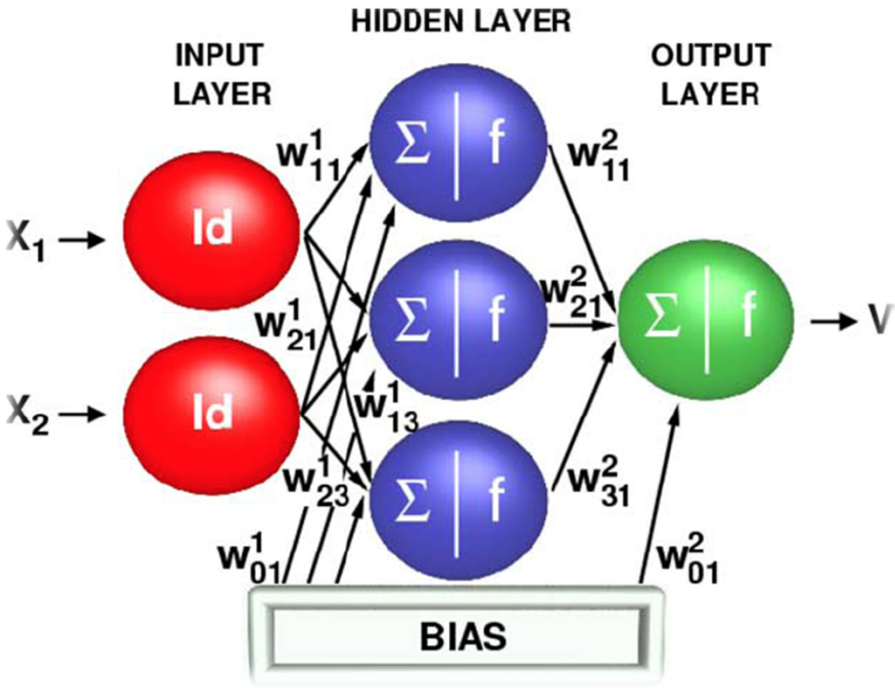
Alternativas

- Potenciales efectivos (de 2, 3, 4..cuerpos, embedded atom models, etc ..): Poco transferibles
- Aproximaciones semiempíricas (e.g. Tight Binding): aplicabilidad limitada
- **Machine learning: Redes Neuronales**



Una red neural simple: superficies de energía potencial

- Se trata de un sistema compuesto por nodos (neuronas)
- Funciones de respuesta e input acotadas (e.g. Sigmoides, tanh)
- El proceso de aprendizaje es una regresión no lineal
- Caso más simple
  - Input: coordenadas moleculares
  - Output: Energía potencial
  - Datos de aprendizaje (referencia): Energías ab initio

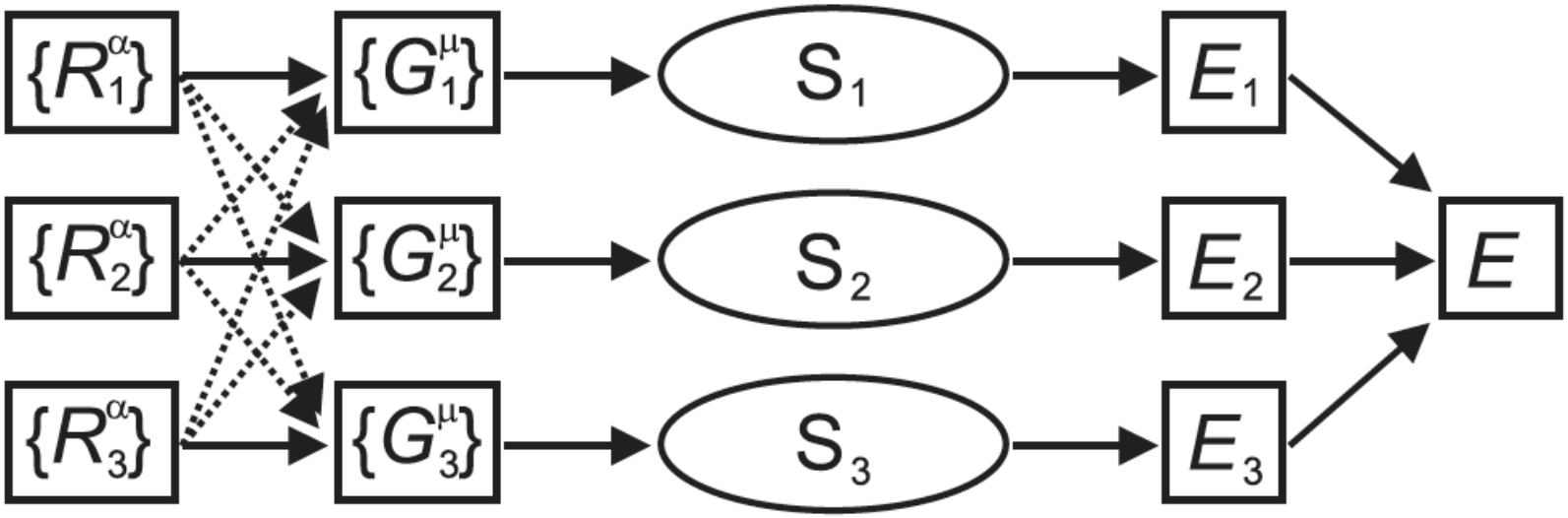


$$V_{nn}(\mathbf{x}) = f_2\left(w^2_{01} + \sum_j w^2_{j1}\, f_1\left(w^1_{0j} + \sum_{i=1}^{6} w^1_{ij} x_i\right)\right)$$

Transferibilidad y funciones de simetría

- Se requiere un nuevo aprendizaje (ajuste) cuando el número de átomos cambia: la formulación **no es transferible**
- Behler y Parrinello[3] (2007) formulan la **representación atómica en términos de funciones de simetría**

$$E = \sum_{i}^{N} E_i(\{R_i\}) \qquad \{R_i\} \rightarrow \{G_i^1, G_i^2\}$$



- Las funciones de simetría atómica describen el entorno de cada átomo
  - Correlaciones pares

$$G_i^1 = \sum_{j \neq i}^{\text{all}} e^{-\eta(R_{ij}-R_s)^2} f_c(R_{ij})$$

  - Correlaciones de tripletes

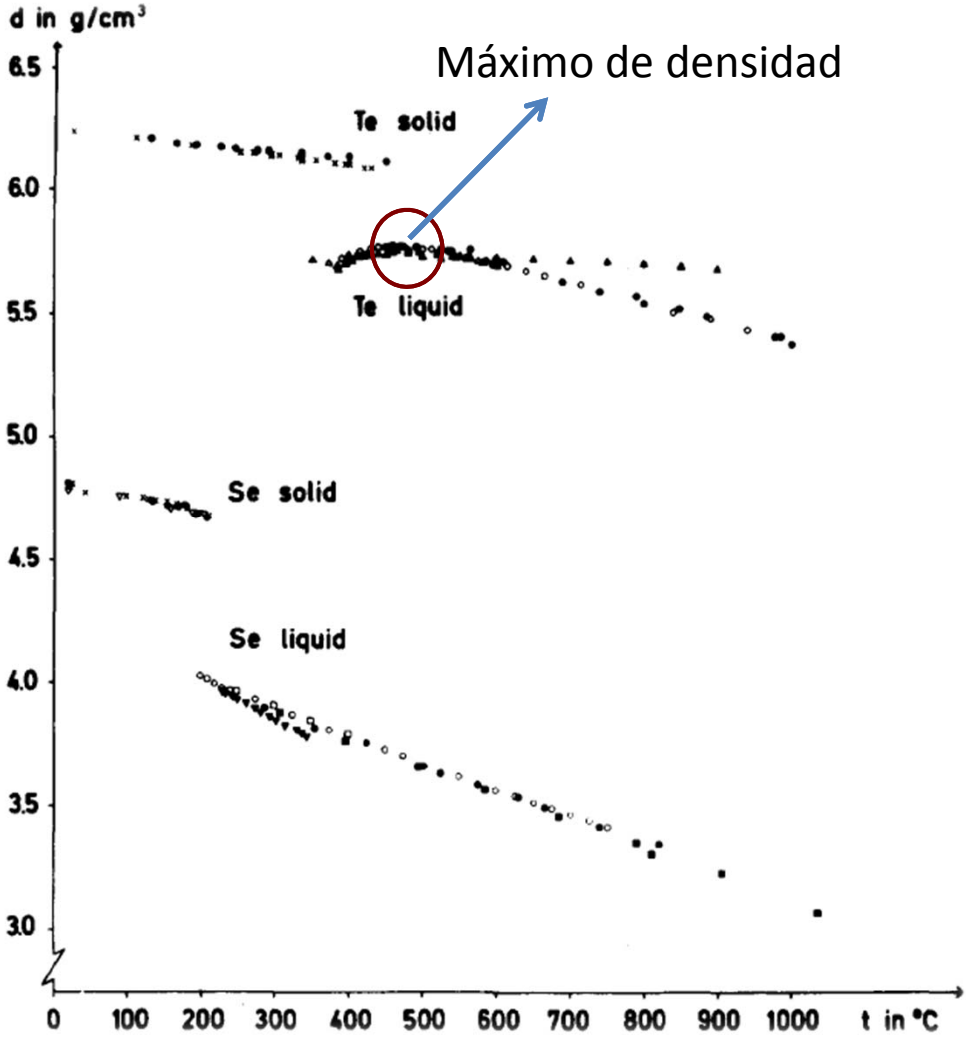
$$G_i^2 = 2^{1-\zeta} \sum_{j,k \neq i}^{\text{all}} (1 + \lambda \cos\theta_{ijk})^\zeta \times e^{-\eta(R_{ij}^2+R_{ik}^2+R_{jk}^2)} f_c(R_{ij}) f_c(R_{ik}) f_c(R_{jk})$$

Ventajas

- Los parámetros son transferibles (dentro del rango físico de aprendizaje) cuando el número de átomos aumenta
- Escalado lineal: método O(N)
- El sistema es altamente paralelizable (óptimo para GPUs)

Problema físico

- Sistema a estudiar: **Te líquido**
- Rango de aprendizaje: **600C – 800C**
- Objetivo: caracterizar el máximo de densidad



High-Dimensional Neural Network Potentials

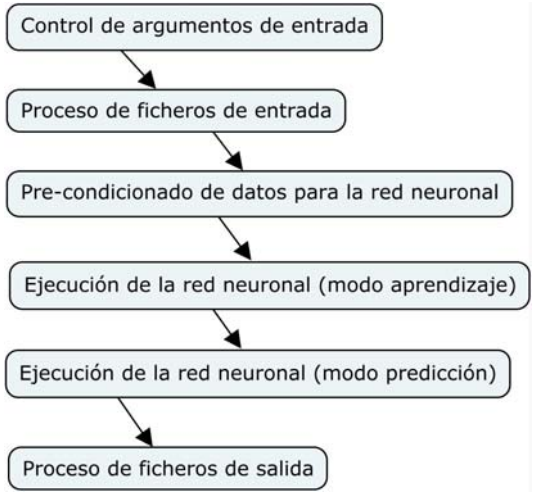
- HDNNP es una red neuronal que se ejecuta sobre una GPU (*Graphical Processor Unit*) del fabricante NVIDIA con arquitectura CUDA (*Compute Unified Device Architecture*).
- HDNNP es capaz de aprender a partir de las propiedades de determinados conjuntos de sistemas de átomos (posiciones en una secuencia temporal y energía para una temperatura y densidad dadas) y predecir las propiedades macroscópicas de otro sistema de átomos diferente, en condiciones de temperatura y/o densidad, a los utilizados para el aprendizaje.
- HDNNP tiene las siguientes características:
  - Es pre-alimentada FFNN (*Feed-Forward Neural Network*).
  - Es multi-capa.
  - Se ha diseñado para sistemas de altas dimensiones.

Análisis y propuesta de soluciones

- Exposición del problema:
  - Dado un sistema de átomos (configuración) se quieren calcular las propiedades macroscópicas (energía, presión, conductividad, etc.) del mismo.
- Posibles soluciones:
  - Utilizar interacciones efectivas entre pares/tripletes de partículas ajustadas a cálculos mecano-cuánticos y/o propiedades experimentales. Éste es un procedimiento habitual cuando no se requiere gran precisión en sistemas relativamente simples.
  - Resolver la ecuación de Schrödinger para un conjunto de átomos dado - empleando la teoría del funcional de la densidad (DFT) - y obtener las energías y las fuerzas correspondientes. Estos métodos, conocidos como *ab initio*, son computacionalmente muy costosos y sólo pueden aplicarse a unos cientos de átomos.
  - Utilizar redes neuronales NN (*Neural Networks*) dada su habilidad para representar funciones arbitrarias. Estas redes se consideran "aproximadores universales" y aprenden a partir de datos *ab initio* para conjuntos pequeños de átomos permitiendo un estudio detallado en condiciones donde el cálculo requiere muestras (número de átomos/moléculas) grandes.

Diseño

- Se realiza de forma modular por lo que cada parte de la solución se pueda diseñar e implementar en base a descomposiciones (especificaciones) de otras partes más genéricas.
- Dada la alta cantidad de información a procesar y almacenar, cada módulo deberá ser lo máximo posible independiente de los otros con objeto de optimizar los recursos (memoria RAM). Esto permitirá realizar ejecuciones independientes de cada módulo obteniendo resultados de las pruebas que retro-alimentarán el ciclo diseño → implementación → pruebas.
- Algoritmo general:



Diseño → Módulos 1 y 2

- Módulo 1: El objetivo es controlar los argumentos de entrada de la aplicación y generar los nombres de los ficheros de entrada a procesar.
- Módulo 2: El objetivo de este módulo es leer los distintos ficheros de entrada de aprendizaje y de predicción y crear un único fichero de salida compuesto por la energía de cada secuencia temporal (conjunto de átomos) junto a los datos estrictamente necesarios (selección de características) de dicha secuencia.

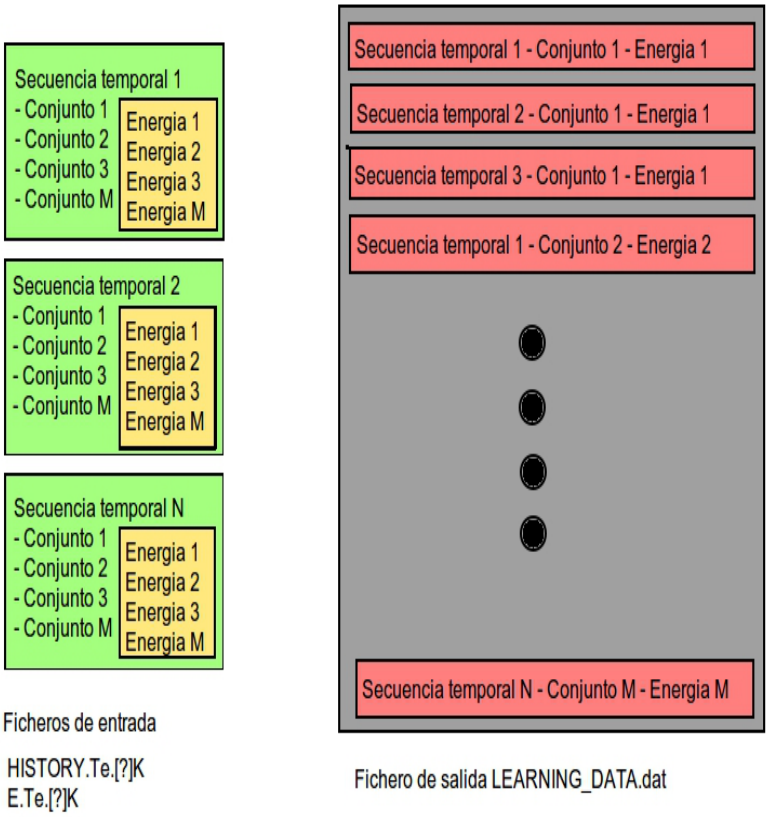
```
timestep         1        64         2        1    0.001000
   14.508000      0.000000      0.000000
    0.000000     14.508000      0.000000
    0.000000      0.000000     14.508000
      Te         1  127.600000    0.000000
    9.2039780617E+00      9.7824159389E+00      2.8602546664E+00
    6.9020853329E-01      5.7969950169E-01     -5.0218420389E+00
    2.1141000000E-01      7.3537400000E-01     -2.0765000000E-01
      Te         2  127.600000    0.000000
    4.9346167759E+00      2.1517051280E+00      4.2864128284E+00
   -6.4354145474E-01      3.1454012737E+00     -7.9618288613E-01
    1.6940500000E-01      6.5414700000E-01      2.2044600000E-01
      Te         3  127.600000    0.000000
    1.0175378176E+01      4.5515028593E+00      1.3390518543E+01
    5.0135202338E-01     -5.8780197567E-01     -1.1360254390E+00
   -2.0113400000E-01      5.2877400000E-01      5.7874100000E-01
      Te         4  127.600000    0.000000
    9.8613298836E-01      9.5423831100E+00      1.4415460722E+00
    1.0302747776E+00     -3.7611708550E+00      1.1845292580E+00
    6.0212800000E-01     -1.2605300000E-01     -2.3624700000E-01
      Te         5  127.600000    0.000000
    1.0231880453E+01      7.7367275856E-01      6.2895434144E+00
   -2.0277714305E+00     -1.4420043594E+00     -2.8902172356E+00
    5.0040300000E-01      2.7809400000E-01      1.2903300000E-01
      Te         6  127.600000    0.000000
    1.0756549941E+01      2.8265929146E+00      1.0011055345E+01
   -1.5319025498E+00     -1.4049868508E+00      2.0238728085E+00
   -3.4564200000E-01      1.6534200000E-01      4.2873300000E-01
      Te         7  127.600000    0.000000
    8.9678764656E+00      6.6238054132E+00      4.7118632652E-01
   -3.4923324006E+00      1.7661316123E-01      1.5661413212E-01
    2.8351200000E-01     -2.8377000000E-01     -3.7302300000E-01
```

HISTORY.Te.673K.

```
0.000000   -185.488060
0.001000   -185.470680
0.002000   -185.453920
0.003000   -185.437850
0.004000   -185.422500
0.005000   -185.407910
0.006000   -185.394180
0.007000   -185.381330
0.008000   -185.369410
0.009000   -185.358520
0.010000   -185.348730
0.011000   -185.340070
0.012000   -185.332590
0.013000   -185.326300
0.014000   -185.321260
0.015000   -185.317470
0.016000   -185.314960
```

E.Te.673K

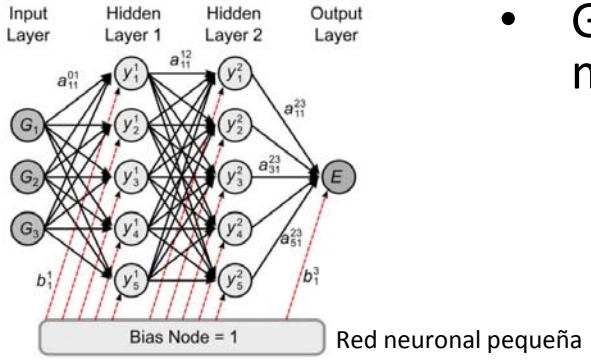
Módulo 2 → Sesgo temporal



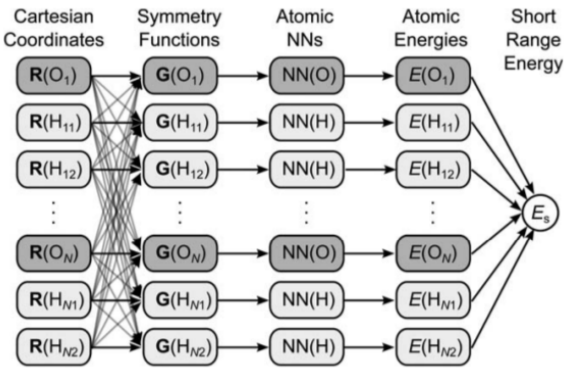
Estructura del fichero de salida del aprendizaje

- La lectura de los conjuntos de átomos de los ficheros de entrada se realiza de forma secuencial por la propia naturaleza de dichos ficheros.
- La escritura de dichos conjuntos tras la selección de características se realiza en el mismo orden secuencial de la lectura.
- Se crea un sesgo temporal relativo a las temperaturas seleccionadas.
- Con objeto de evitar el sesgo en el aprendizaje se debe realizar un intercalado, en la escritura del fichero de salida para el aprendizaje, entre los conjuntos de átomos de las distintas secuencias temporales.

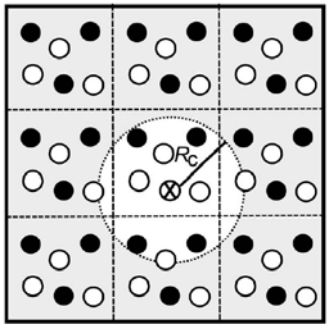
Módulo 3 → Pre-condicionado de datos para la red neuronal



Red neuronal pequeña



Red neuronal alta dimensionalidad



Cutoff del átomo

- Generar los valores de representación de cada átomo mediante funciones de simetría:
  - La necesidad de utilizar funciones de simetría emerge de la condición de alta dimensionalidad de la red neuronal.
  - La red neuronal pequeña el vector de entradas $\{G_i\}$ corresponde a las coordenadas cartesianas de los átomos y sólo existe una red neuronal para cada configuración.
  - La red neuronal de alta dimensionalidad el vector de entradas $\{G_i\}$ corresponde al conjunto de valores de representación, existiendo una red neuronal por átomo.
  - Si en lugar de utilizar los valores de representación se emplearan las coordenadas atómicas como entrada directa a la red neuronal ésta no sería "transferible" a sistemas con diferente número de partículas.
  - La operación a realizar consiste en transformar las coordenadas de un átomo en múltiples valores de representación (obtenidos de evaluar las funciones de simetría) que caracterizarán a dicho átomo dentro de un "entorno limitado esférico" (*cutoff*).
  - Dicho *cutoff* define el entorno energéticamente relevante del átomo y se establece como el radio de la esfera que limita el entorno.

Módulo 3 → Pre-condicionado de datos para la red neuronal

- Se utilizarán las siguientes funciones de simetría:
  - ✓ Función de *cutoff*: decrece cuando se incrementa la distancia (R_{ij}) entre el átomo central i y su vecino j, lo que refleja cualitativamente la reducción de las interacciones entre átomos según se distancian.

$$f_{c,1}(R_{ij}) = \begin{cases} 0.5 \cdot \left[\cos\left(\frac{\pi R_{ij}}{R_c}\right) + 1\right] & \text{for} \quad R_{ij} \leq R_c \\ 0.0 & \text{for} \quad R_{ij} > R_c \end{cases},$$

Donde:
R_{ij}: distancia entre el átomo central i y el átomo vecino j.
R_c: radio del *cutoff,* se define como 6Å pero se utilizará como valor de referencia R_c^2 = 36Å, con objeto de evitar el cálculo de raíces cuadradas en la determinación de distancias inter-atómicas.

  - ✓ Función radial: define el orden radial de los átomos dentro del *cutoff*.

$$G_i^2 = \sum_{j=1}^{N_{atom}} e^{-\eta\left(R_{ij}-R_s\right)^2} \cdot f_c\left(R_{ij}\right).$$

Donde:
η: parámetro que define el ancho de las gausianas.
R_s: parámetro que define el centro de las gausianas.

  - ✓ Función angular: define el orden angular de los átomos dentro del cutoff.

$$G_i^3 = 2^{1-\zeta} \sum_{j \neq i} \sum_{k \neq i,j} \left[\left(1 + \lambda \cdot \cos\theta_{ijk}\right)^{\zeta} \cdot e^{-\eta\left(R_{ij}^2 + R_{ik}^2 + R_{jk}^2\right)} \cdot f_c\left(R_{ij}\right) \cdot f_c\left(R_{ik}\right) \cdot f_c\left(R_{jk}\right)\right]$$

Donde:
η: parámetro que define el ancho de las gausianas.
ζ: parámetro que define la distribución de los ángulos.
λ: parámetro que cambia la forma de la función del coseno.
θ_{ijk}: ángulo, respecto al átomo central i, que forman los tres átomos i, j y k.

Módulo 3 → Pre-condicionado de datos para la red neuronal

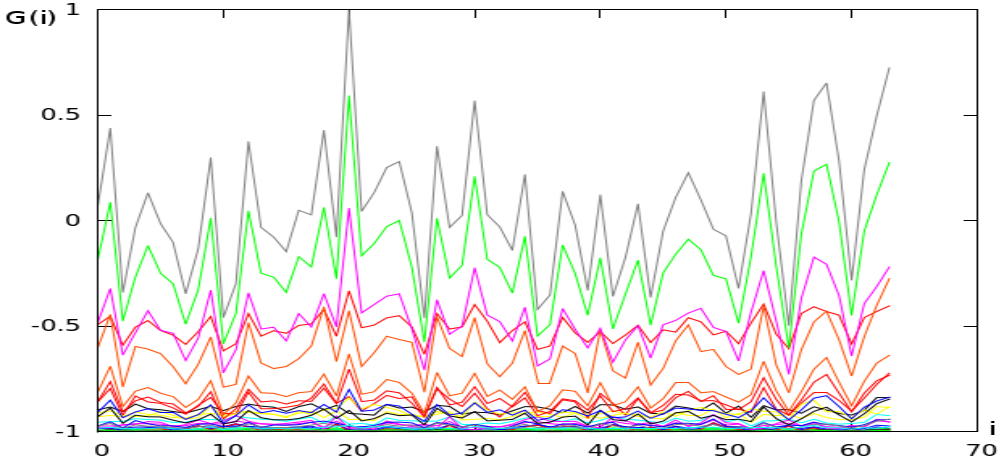
- El cálculo de funciones de simetría consistirá en evaluar cada unas de las funciones anteriores para cada átomo de la configuración y para cada combinación de parámetros posible[1].
  - $\eta = \{ 0, 0.04, 0.14, 0.32, 0.71, 1.79 \}$
  - $R_s = \{ 0, 1, 2, 3, 4, 5 \}$
  - $\zeta = \{ 1, 2, 4, 16 \}$
  - $\lambda = \{ -1, 1\}$
- Cada átomo quedará representado por el conjunto de valores obtenidos de los cálculos anteriores, que son los denominados valores de representación.
- Dichos cálculos se realizarán en paralelo utilizando la GPU.
- A los valores de representación de cada átomo se les realiza una selección de características con el objetivo de reducir la dimensionalidad de los datos. Se debe comprobar que:
  - Para el mismo átomo no existen dos valores de representación iguales.
  - Para una misma función de simetría y para todos los átomos el valor de representación es cero.

Módulo 3 → Pre-condicionado de datos para la red neuronal

- Comprobar que los valores obtenidos por las funciones de simetría representan correctamente al átomo:
  - Los valores de representación se deben poder diferenciar lo máximo posible.
  - Para cada pareja de átomos debe existir una correlación en la magnitud entre los valores de representación en términos de funciones de simetría (que definen su entorno físico) y las fuerzas a las que están sometidos dichos átomos. Si $d_{ij} \rightarrow 0$, entonces $f_{ij} \rightarrow 0$.

$$d_{ij} = \sqrt{\sum_{\alpha} (G_i^2 - G_j^2)^2}$$

$$F_{ij} = \left| \vec{F}_i - \vec{F}_j \right|$$



Valores de representación normalizados

Módulo 3 → Pre-condicionado de datos para la red neuronal

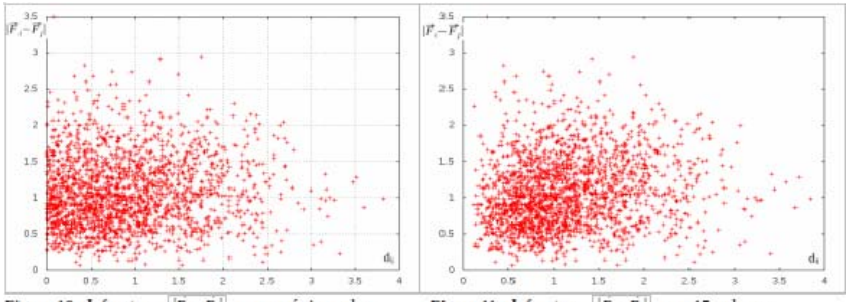


Figura 10: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con un único valor.

Figura 11: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con 15 valores.

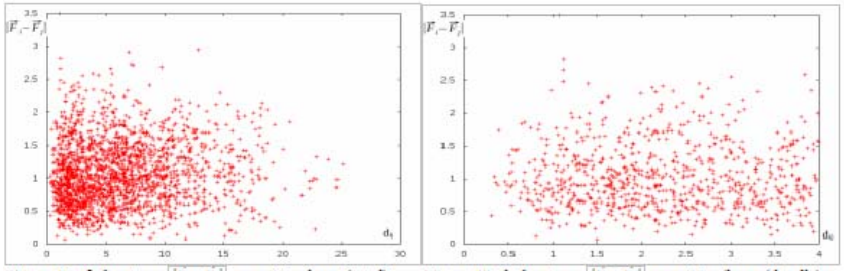


Figura 12: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con 30 valores (total).

Figura 13: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con 30 valores (detalle).

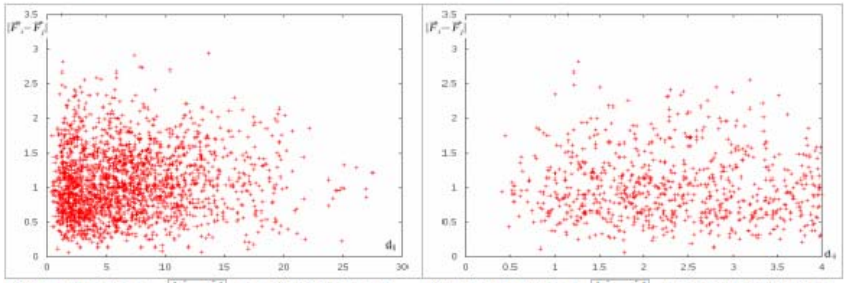


Figura 14: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con 57 valores (total).

Figura 15: d_{ij} frente a $|\vec{F}_i - \vec{F}_j|$ con 57 valores (detalle).

- Para un único valor de representación la relación de magnitud está descompensada. Se tienen muchos valores nulos o muy cercanos al cero para d_{ij} con $f_{ij} \neq 0$, las fuerzas a que están sometidos los átomos i y j son diferentes.
- Para quince valores de representación la relación empieza a ser más equitativa y los puntos se alejan del cero y se dispersan.
- A partir de treinta valores de representación se representa una relación de magnitud equitativa.

Módulo 3 → Pre-condicionado de datos para la red neuronal

- Con el objetivo de representar una configuración con los mínimos errores y obtener una capacidad de ajuste de la red neuronal lo más eficiente posible, los datos se deben normalizar.
- Se utilizará el método de *ranging* con los límites [-1 , 1], aplicando la función siguiente a todos los valores de representación:

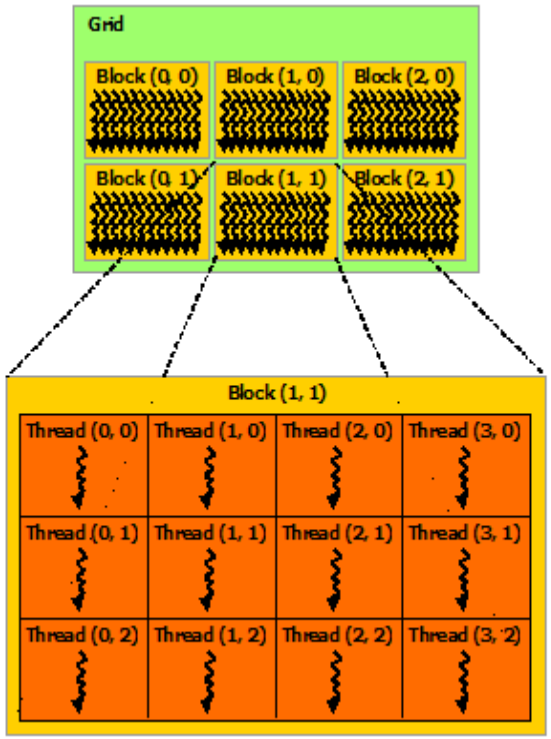
$$G_i^{\text{scaled}} = \frac{2\left(G_i - G_{i,\min}\right)}{G_{i,\max} - G_{i,\min}} - 1,$$

- La decisión relativa al método de normalización (*ranging*) viene dada porque los nodos de la red neuronal utilizarán funciones de estimulación tipo sigmoide centradas en cero, y cuya respuesta estará acotada entre -1 y 1.

$$y = \frac{1}{1 + e^{-x}}$$

Función sigmoide

Módulo 3 → Paralelización GPU



Arquitectura lógica GBT (Grid-Block-Thread) en CUDA.

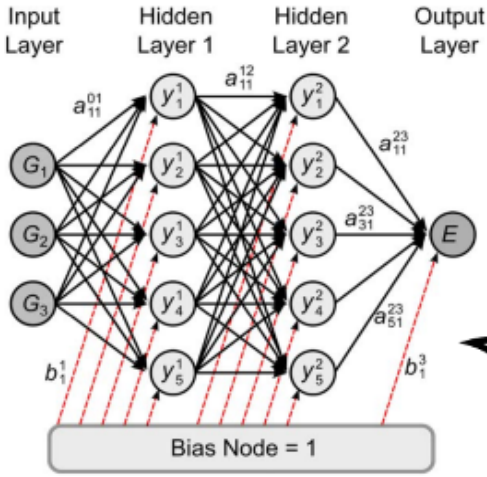
- Se decide asignar los bloques a los átomos centrales i (sobre los que se calcula las funciones de simetría) y los hilos a los átomos vecinos de primera instancia j (sobre los que se pivota en primera instancia).
- Este diseño ahorra dos bucles: el de recorrer los átomos centrales y el de pivotar sobre los vecinos de primera instancia.
- La llamada al *kernel* resultará en la siguiente forma:
  - **kernel**<<<número_de_átomos, número_de_átomos>>>(parámetro1, parámetro2, …,parámetro N);

- Si se dispone de GPUs con varios procesadores (multi-GPU) se puede paralelizar las llamadas a los *kernels*: de forma que a cada *grid* se le asigna el átomo central i, a cada bloque el átomo vecino de primera instancia j y a cada hilo el átomo vecino de segunda instancia k.

Módulo 4 → Ejecución de la red neuronal (modo aprendizaje):

- Se definen dos operaciones diferentes:
  - Época de la red neuronal: leer los datos de representación normalizados, inicializar la red neuronal, calcular la predicción y el error de predicción.
  - Optimización: optimizar los parámetros de la red neuronal de modo que se minimice el error de predicción.
- Cada átomo inicializa una red neuronal.

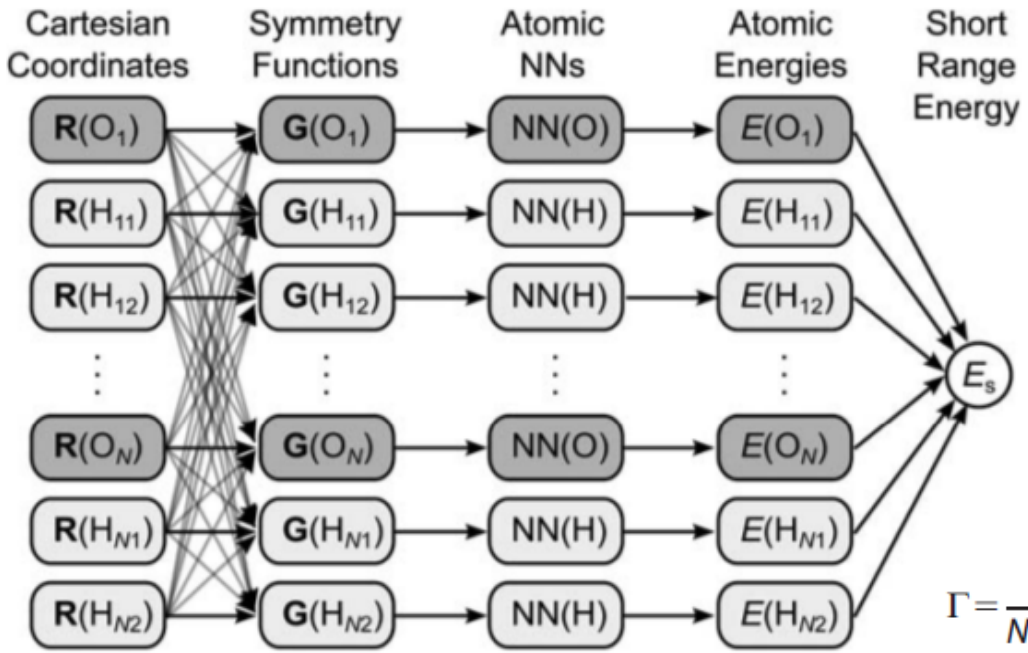
Módulo 4 → Ejecución de la red neuronal (modo aprendizaje):



$$y_i^j = f_i^j\left(b_i^j + \sum_{k=1}^{N_{j-1}} a_{k,i}^{j-1,j} \cdot y_k^{j-1}\right) \quad (5)$$

$$E = f_1^3\left(b_1^3 + \sum_{k=1}^{5} a_{k1}^{23} \cdot f_k^2\left(b_k^2 + \sum_{j=1}^{5} a_{jk}^{12} \cdot f_j^1\left(b_j^1 + \sum_{i=1}^{|G|} a_{ij}^{01} \cdot G_i\right)\right)\right) \quad (6)$$

Época de la red neuronal



$$E_s = \sum_{i=1}^{N_{atom}} E_i. \quad (7)$$

$$\Gamma = \frac{1}{N_{struct}} \sum_{i=1}^{N_{struct}} \left(E_{NN}^i - E_{Ref}^i\right)^2 \quad (8)$$

Módulo 4 → Ejecución de la red neuronal (modo aprendizaje):

- La optimización o aprendizaje se basa en ajustar, tras la ejecución de cada época, el valor de los N parámetros $a^{j-1}_{k\,i}$ de la red neuronal con el objetivo de minimizar el error de predicción.
- Se decide utilizar el optimizador COMPLEX-BCPOL de la librería IMSL.
- La ejecución de la optimización de los parámetros se realizará de forma secuencial mediante la librería IMSL.
- Se dirige la búsqueda de la optimización mediante el parámetro BIAS -> OUTPUT_NEURON.
- Se paralelizará el cálculo de la función de coste para que devuelva el error de predicción de un número determinado (NUMBER_OF_BOXES_TO_OPT) de configuraciones de átomos.
- Kernel:
  - Se crearán NUMBER_OF_BOXES_TO_OPT bloques que representarán las cajas de átomos.
  - Se crearán tantos hilos como átomos se tenga por caja por lo que cada hilo representará una red neuronal atómica.
  - **kernel**<<<NUMBER_OF_BOXES_TO_OPT, número_átomos_por_caja>>>(parámetro1, parámetro2, …,parámetro N);
- El uso de un filtro digital extendido de Kalman mejoraría la optimización de parámetros ya que se trata de un método dirigido y con memoria.
- El uso de gradientes y de las fuerzas en el proceso de optimización también mejoraría notablemente el rendimiento de la operación de optimización.

Módulo 5 → Ejecución de la red neuronal (modo predicción):

- Se obtiene la predicción de la energía de cada secuencia temporal atómica.

  1. Leer los valores de representación normalizados de cada átomo para una configuración dada, inicializar una red neuronal por átomo, obtener la predicción atómica de dicha red neuronal y sumar todas las predicciones atómicas resultantes de las redes neuronales obteniendo la predicción de la configuración (caja).
  2. Repetir el paso 1 hasta que no queden configuraciones a predecir.

- Por lo tanto, se define una única operación:
  - Época de la red neuronal: leer los datos de representación normalizados, inicializar la red neuronal y calcular la predicción. No se calcula el error de predicción dado que sólo se quiere obtener la predicción de la energía.

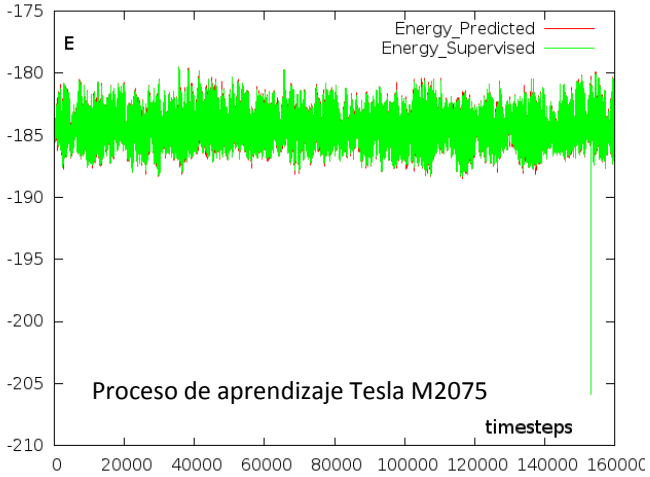
Módulo 6 → Proceso de ficheros de salida

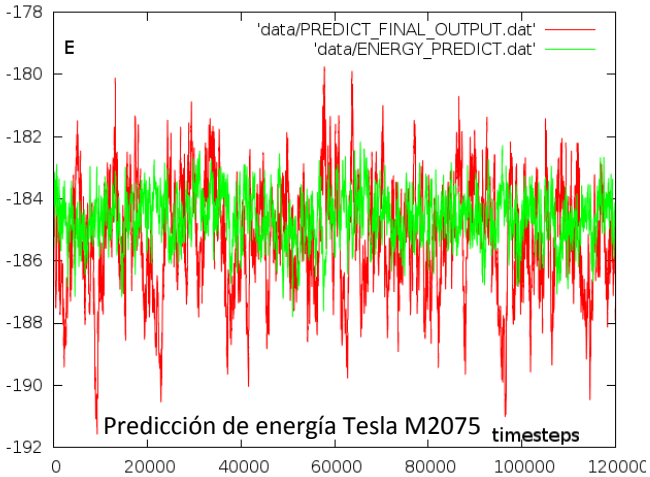
- El objetivo de este módulo es calcular el error medio de predicción y generar las gráficas que indiquen la calidad del aprendizaje y de la predicción.
- Se imprimirá por pantalla el error medio de predicción y se crearán los ficheros “data/LEARNING_PROCESS.png” y “data/FINAL_PREDICTION.png.

Pruebas y simulaciones

- Se ajusta la red neuronal y el optimizador a los siguientes parámetros :
  - Número de valores de representación por átomo = 30.
  - Número de neuronas en cada capa oculta de la red neuronal = 4.
  - Número máximo de llamadas a la función de coste = 80000.
  - Sólo se permite predecir un único fichero de configuraciones.
- Se dispone de seis ficheros de datos *ab initio* con las temperaturas 673K, 723K, 773K, 823K, 873K y 973K y de cinco GPUs modelos GTX590, GTX660, GTX960, GTX980 y TeslaM2075, se realizan las siguientes simulaciones:

Dos conjuntos de aprendizaje (673K y 973K) y uno de predicción (773K)

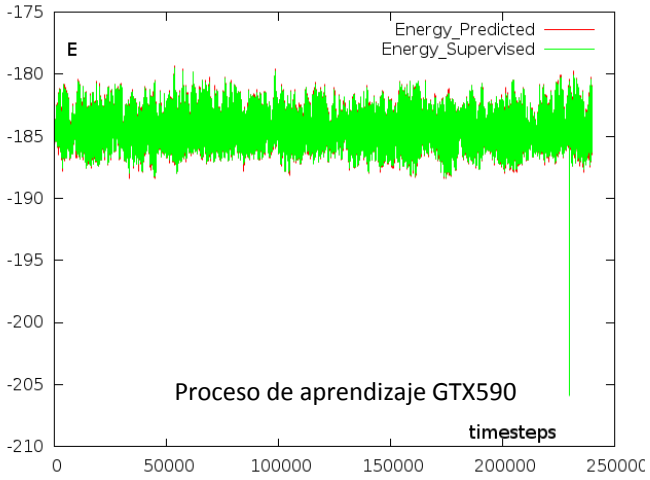


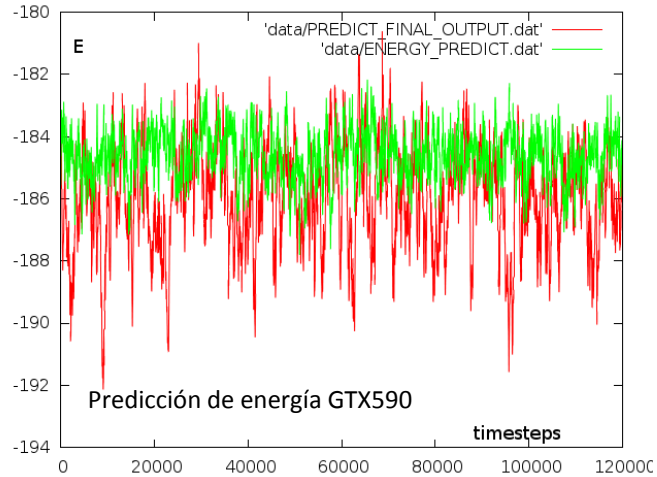


2-Aprendizaje	Tiempo (m)	Ē de predicción
GTX 590	740	0.395462
GTX 660	755	0.527924
GTX 960	383	0.408463
GTX 980	340	0.553131
Tesla M2075	435	0.720734

- El proceso de aprendizaje es correcto dado que la diferencia entre la energía predicha (rojo), realizada tras la optimización de los parámetros de la red neuronal, y la energía supervisada (verde) converge y es pequeña.
- El ajuste en la predicción de la energía no es óptimo dado que la predicción (rojo) no se asemeja a la energía supervisada (verde).
- El error medio de predicción (Ē) es alto lo que explica el mal ajuste representado en la predicción de la energía.
- Se concluye que el sistema aprende correctamente pero no predice con el ajuste deseado.
- Las GPUs más modernas y/o potentes necesitan menos tiempo para realizar la simulación y que todas ellas poseen un error medio de predicción similar (del mismo orden de magnitud).

Tres conjuntos de aprendizaje (673K, 723K y 973K) y uno de predicción (773K)

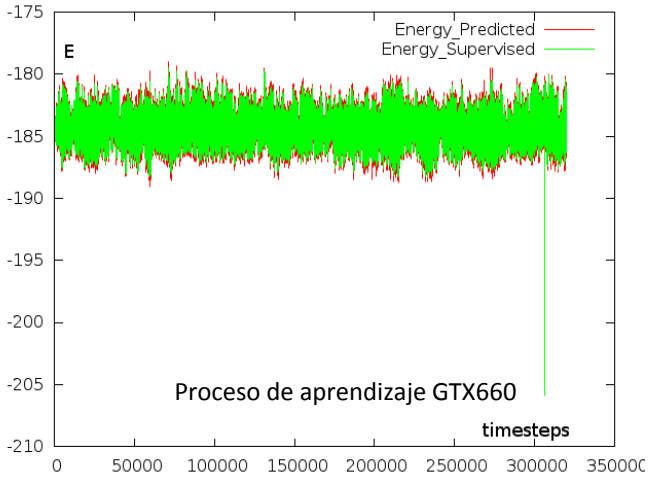


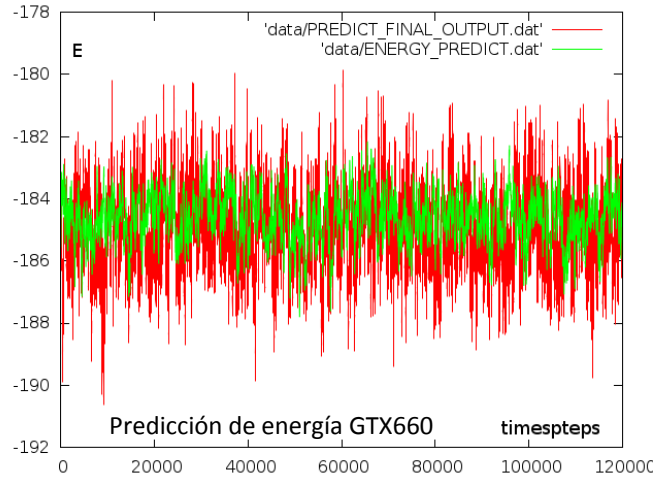


3-Aprendizaje	Tiempo (m)	Ē de predicción
GTX 590	1087	1,399277
GTX 660	1034	1,256638
GTX 960	488	1,067123
GTX 980	307	1,328995
Tesla M2075	576	0,819427

- El proceso de aprendizaje es correcto dado que la diferencia entre la energía predicha (rojo), realizada tras la optimización de los parámetros de la red neuronal, y la energía supervisada (verde) converge y es pequeña.
- El ajuste en la predicción de la energía no es óptimo dado que la predicción (rojo) no se asemeja a la energía supervisada (verde).
- El error medio de predicción (Ē) es alto (incluso más que antes) lo que explica el mal ajuste representado en la predicción de la energía.
- Se concluye que el sistema aprende correctamente pero no predice con el ajuste deseado.
- Las GPUs más modernas y/o potentes necesitan menos tiempo para realizar la simulación y que todas ellas poseen un error medio de predicción similar (del mismo orden de magnitud).

Cuatro conjuntos de aprendizaje (673K, 723K, 873K y 973K) y uno de predicción (773K)

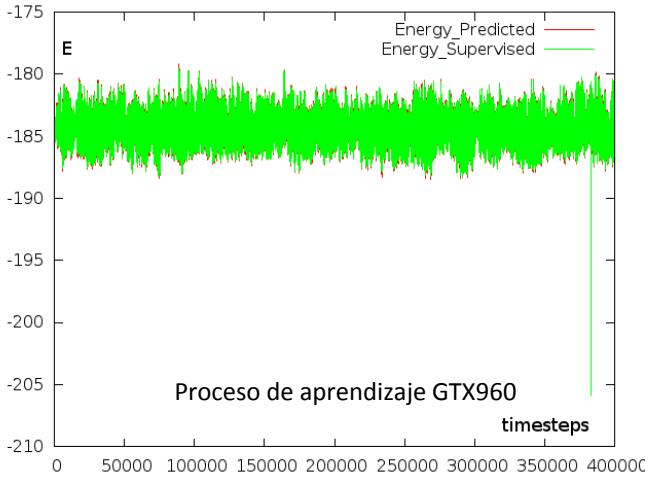


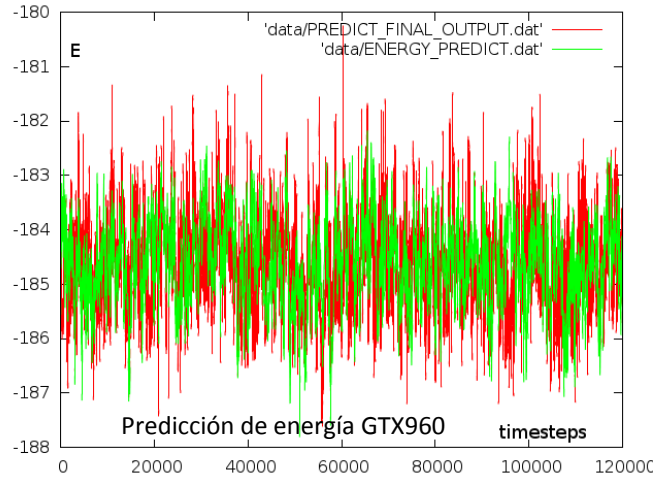


4-Aprendizaje	Tiempo (m)	Ē de predicción
GTX 590	1322	0.320465
GTX 660	1399	0.396881
GTX 960	763	0.638229
GTX 980	409	0.631714
Tesla M2075	732	0.707733

- El proceso de aprendizaje es correcto dado que la diferencia entre la energía predicha (rojo), realizada tras la optimización de los parámetros de la red neuronal, y la energía supervisada (verde) converge y es pequeña.
- El ajuste en la predicción de la energía no es óptimo dado que la predicción (rojo) no se asemeja a la energía supervisada (verde).
- El error medio de predicción (Ē) es alto lo que explica el mal ajuste representado en la predicción de la energía.
- Se concluye que el sistema aprende correctamente pero no predice con el ajuste deseado.
- Se observa una mejoría en el ajuste de la predicción y en el error medio de predicción de la energía respecto de los casos anteriores.

Cinco conjuntos de aprendizaje (673K, 723K, 823K, 873K y 973K) y uno de predicción (773K)



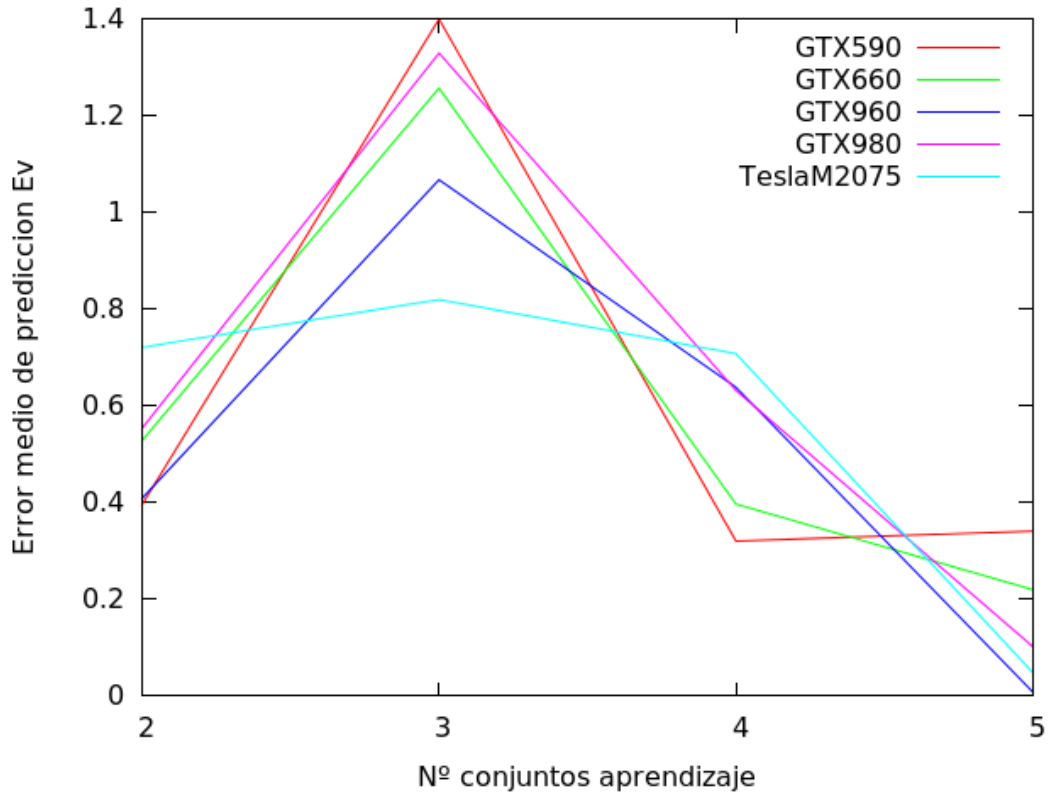


5-Aprendizaje	Tiempo (m)	Ē de predicción
GTX 590	1371	0.340820
GTX 660	1726	0.219604
GTX 960	935	0.007904
GTX 980	603	0.101883
Tesla M2075	1019	0.048141

- El proceso de aprendizaje es correcto dado que la diferencia entre la energía predicha (rojo), realizada tras la optimización de los parámetros de la red neuronal, y la energía supervisada (verde) converge y es pequeña.
- El ajuste en la predicción de la energía es más óptimo que en los casos anteriores dado que la predicción (rojo) se asemeja a la energía supervisada (verde).
- El error medio de predicción (Ē) es más bajo que en los casos anteriores, siendo realmente óptimo en los casos de la GTX960 (0.007904 eV) y de la Tesla (0.048141 eV)
- Se concluye que el sistema aprende correctamente y predice con el ajuste deseado.
- Se observa una notable mejoría en el ajuste de la predicción y en el error medio de predicción de la energía respecto de los casos anteriores.

CONCLUSIONES

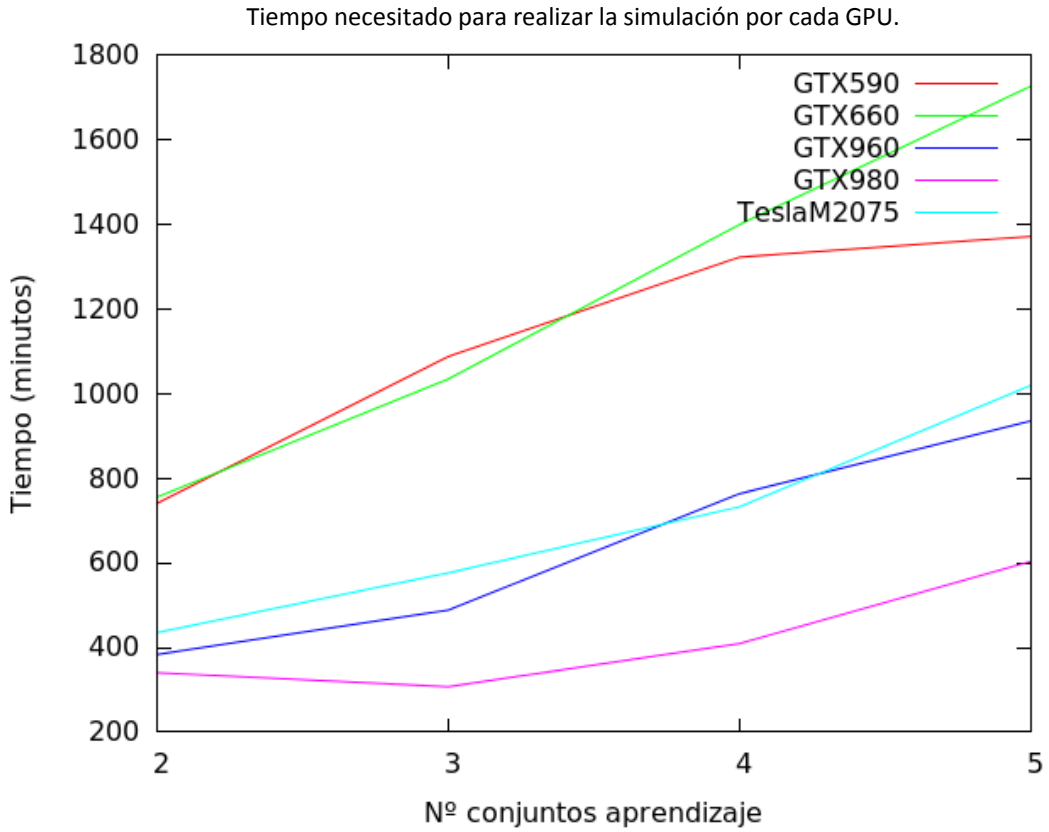
- Error medio de predicción:
  - El error medio de predicción disminuye según se incrementan los conjuntos de aprendizaje con los que se alimenta la red neuronal.
  - Los distintos errores en cada tipo de GPU pueden estar relacionados con la diferencia en el número de núcleos de cada *hardware*.



Error medio de predicción Ē (eV) por cada GPU.

CONCLUSIONES

- Tiempo:
  - El tiempo necesario para realizar la simulación aumenta de forma más o menos lineal respecto al número de conjuntos de aprendizaje utilizados.



BIBLIOGRAFÍA

1. "Constructing High-Dimensional Neural Network Potentials: A Tutorial Review", Jörg Behler. *Int. J. Quantum Chem*. **2015**, *115*, 1032-1050. DOI: 10.1002/qua.24890
2. "Neural Network Models of Potential Energy Surfaces: Prototypical Examples", James B. Witkoskie and Douglas J. Doren. *Journal of Chemical Theory and Computation.* **2005**, *1 (1)*, 14-23. DOI: 10.1021/ct049976i
3. "Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces", Behler y Parrinello, Phys. Rev. Lett. **2007**, *98*, 146401; DOI: 10.1103/PhysRevLett.98.146401
4. "Neural network models of potential energy surfaces", Thomas B. Blank, Steven D. Brown, August W. Calhoun, and Douglas J. Doren. J. Chem. Phys. **1995,** *103*, 4129; DOI: 10.1063/1.469597.
5. "Atom-centered symmetry functions for constructing high-dimensional neural network potentials", Jörg Behler. J. Chem. Phys. **2011**, *134*, 074106; DOI: 10.1063/1.3553717.
6. CUDA Programming - ISBN: 978-0-12-415933-4.
7. CUDA By Example - ISBN 978-0-13-138768-3.
8. CUDA Toolkit Documentation – http://docs.nvidia.com/cuda/index.html.