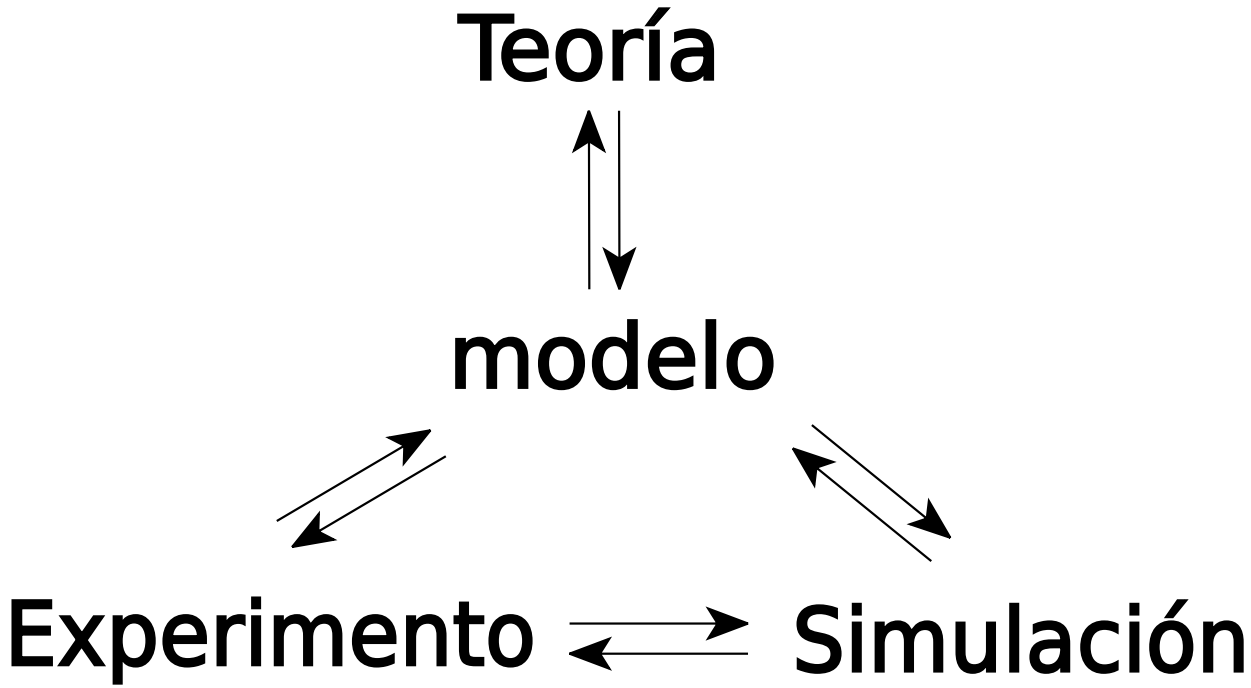
Dinámicas Híbridas QM/MM aceleradas en GPUs

Mariano C. González Lebrero, Matías A. Nitsche y Darío A. Estrin

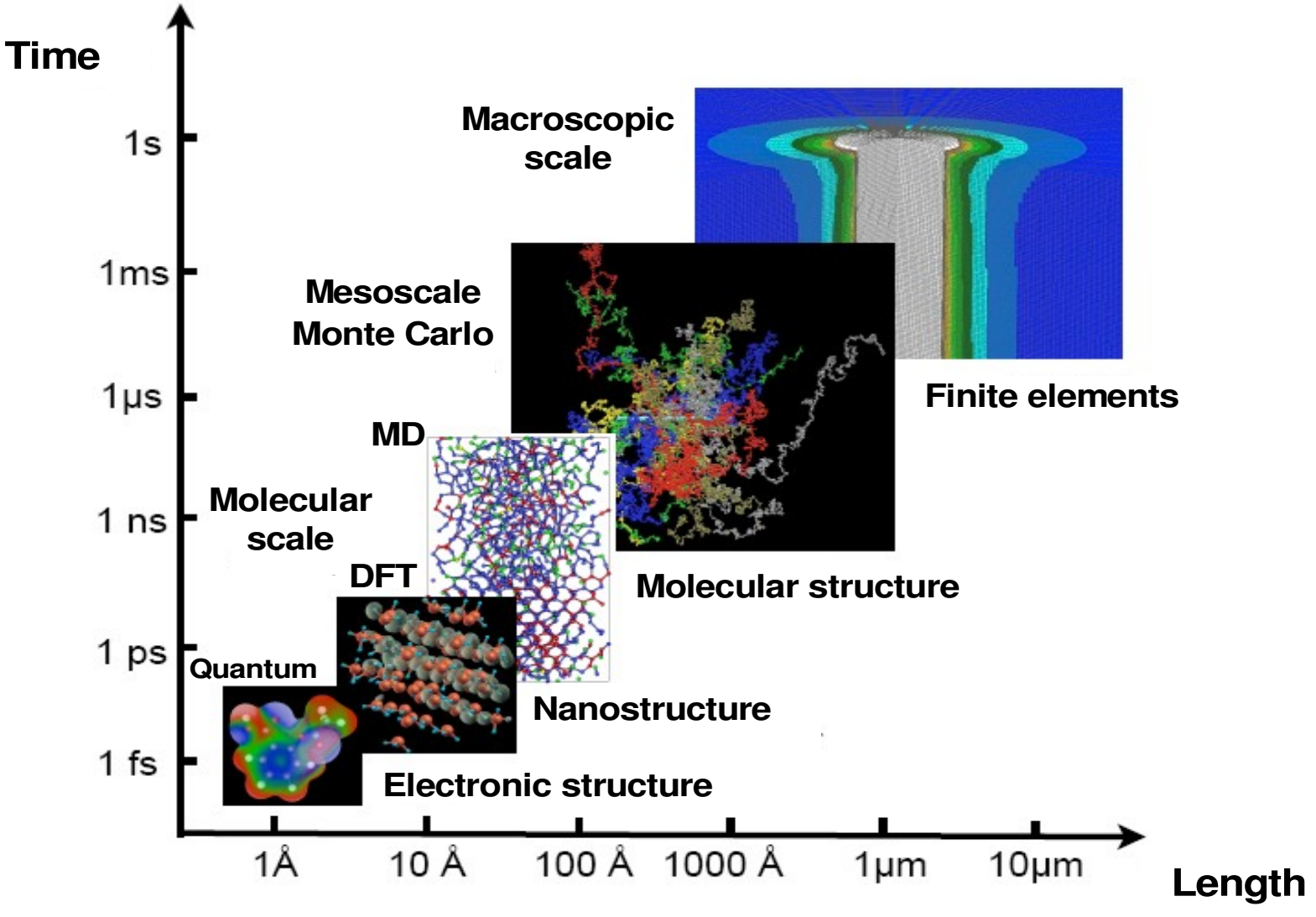
IQUIFIB-CONICET, INQUIMAE-CONICET, FCEN-UBA.

1 de agosto de 2012

Definición: Simular es experimentar con un modelo.



Modelos.


Time
1s
1ms
1µs
1 ns
1 ps
1 fs
Macroscopic scale
Finite elements
Mesoscale Monte Carlo
MD
Molecular scale
Molecular structure
DFT
Quantum
Nanostructure
Electronic structure
1Å
10 Å
100 Å
1000 Å
1µm
10µm
Length

MM

Ya lo vieron ayer...

- Sin detalle elctrónico
- Potenciales parametrizados.
- Aplicable a cientos de miles de átomos y en escalas de los nanosegundos o más.

QM

$$\hat{H}\psi = E\psi$$

Parece simple, pero...

- No es razonable, intuitiva ni fácilmente comprensible.
- No se puede resolver exactamente para más de 1 electrón.
- Soluciones autoconsistentes mediante iteraciones.
- Soluciones aproximadas, aunque pueden ser muy buenas.
- Las soluciones mejores sólo se pueden aplicar a unos pocos átomos en el vacío, para todo lo demas existe...

Métodos QM

Hartree Fock

- Semiempíricos.
- HF
- Post Hartree Fock (MP2, CC, CI, etc.)

DFT

- Semiempíricos
- Local Density Aproximation (LDA)
- Generalized Gradient Aproximation (GGAs: PBE, BP86, etc.)

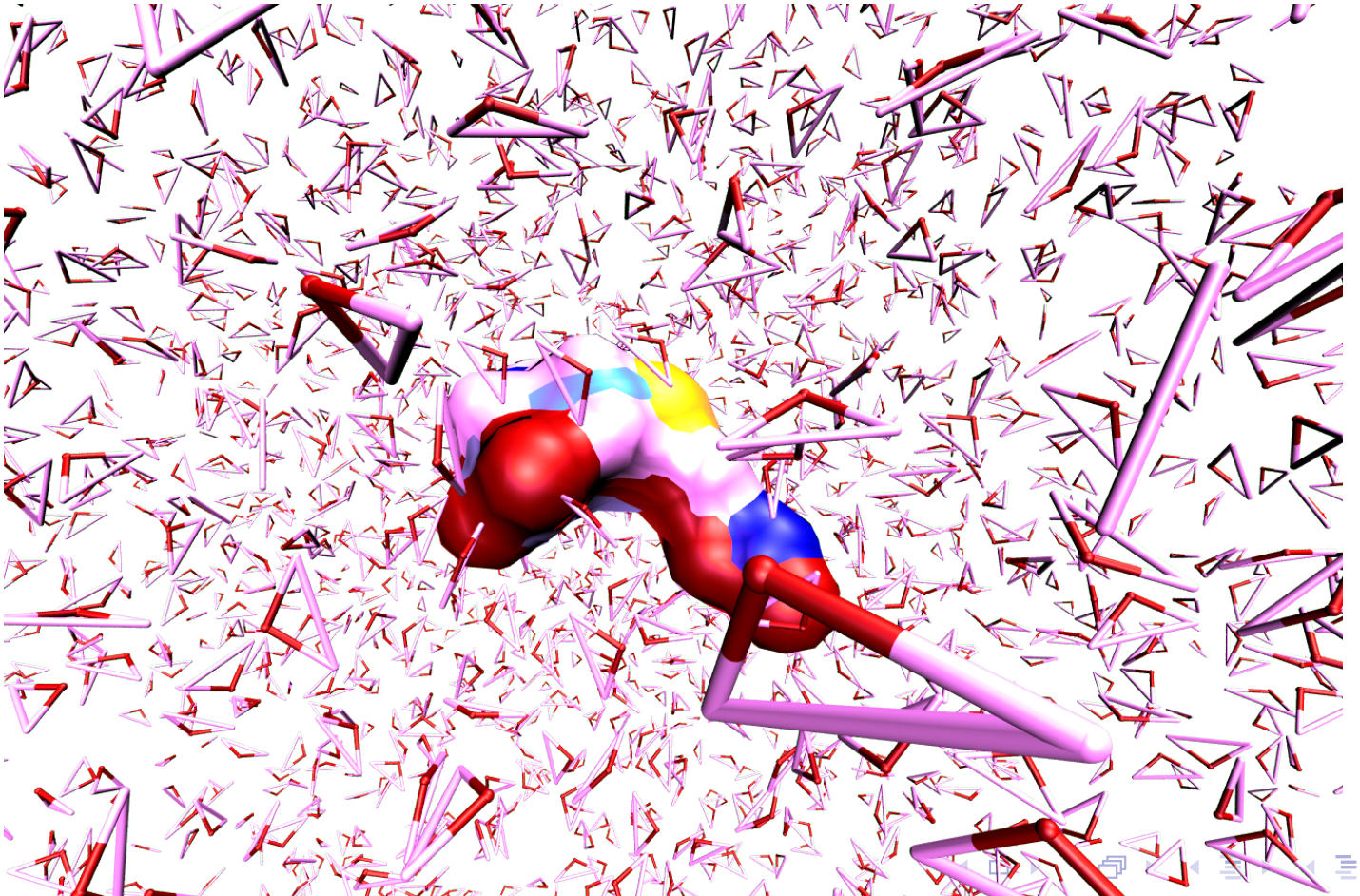
Híbridos (mezcla de DFT y HF)

B3LYP, PBE0, etc.

QM-MM

Métodos Híbridos y Dinámica Molecular

- Para sistemas complejos:
- Métodos híbridos o QM-MM: porción clásica (soluto) + porción cuántica (solvente)



QM/MM y dinámica molecular

$$E = E_{QM} + E_{MM} + E_{QM/MM}$$

$$E_{QM/MM} = \sum_{i=1}^{Ncargas} q_i \int \frac{\rho(r)}{\mid R_i - r \mid} dr + \sum_{i=1}^{Clas} \sum_{j=1}^{cuan} [V_{LJ}(R) + \frac{q_i Z_j}{\mid R_i - r_j \mid}]$$

Para hacer dinámica necesitos las fuerzas, o sea las derivadas de la energía.

$$F = -\nabla E$$

Haciendo un DFT eficiente...

Partimos de un programa existente, escrito en el año 1992 por Darío Estrin, en fortran 77.

1. **Nuevos algoritmos**, basados en el trabajo de Stratmann *et al.*
   - Menor complejidad
   - Variaciones novedosas, teniendo en cuenta: Procesador gráficos y los tipos de sistemas moleculares que se busca tratar
2. **Nueva implementación**
   - Extiende el programa existente (`Fortran77`)
     - Se modifica la porción más significativa del cálculo (XC) para hacerlo en la GPU.
   - Código para procesador gráfico
     - Particularidades del hardware y evaluación de alternativas de implementación
   - Optimizaciónes a mansalva (mayormente reordenamientos de loops) y uso de memoria dinámica.
   - Empleo de bibliotecas MKL de intel (parelelas).
   - otros.

Método de Kohn-Sham

La energía

$$E[\rho] = T_s[\rho] + V_{ne}[\rho] + \frac{1}{2}\int\int \frac{\rho(\vec{r}_1)\rho(\vec{r}_2)}{r_{12}} d\vec{r}_1 d\vec{r}_2 + E_{xc}[\rho]$$

E_{xc} : *energía de intercambio y correlación*

$$\rho(\vec{r}) = \sum_{i=1}^{N_{oc}} |\chi_i(\vec{r})|^2$$

$$\chi_i(\vec{r}) = \sum_{\mu=1}^{M} C_{i\mu}\phi_k(\vec{r})$$

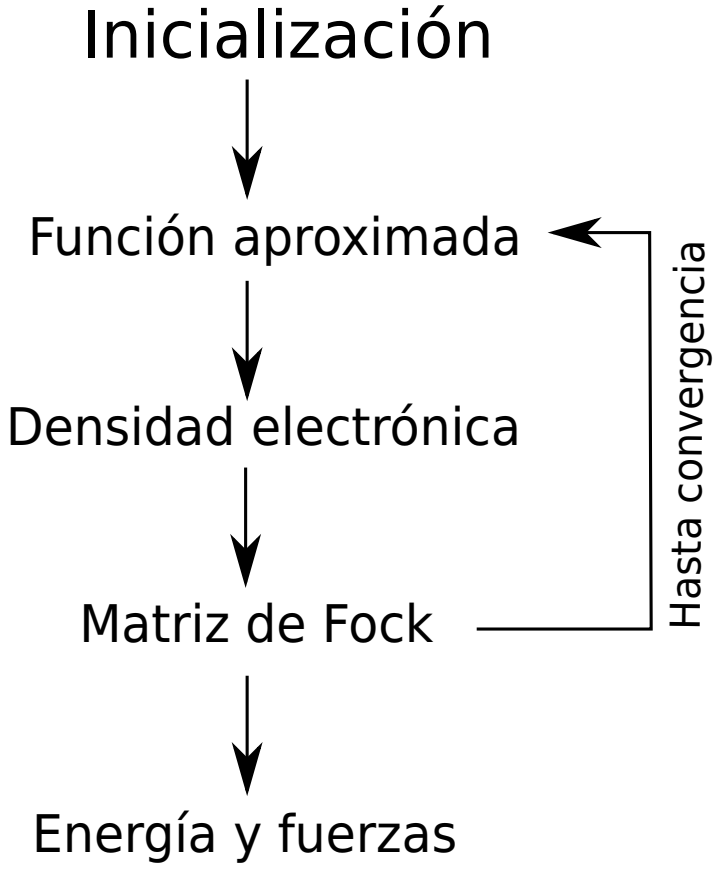
Las ϕ_k son funciones llamadas base, en nuestro caso son Gaussianas.

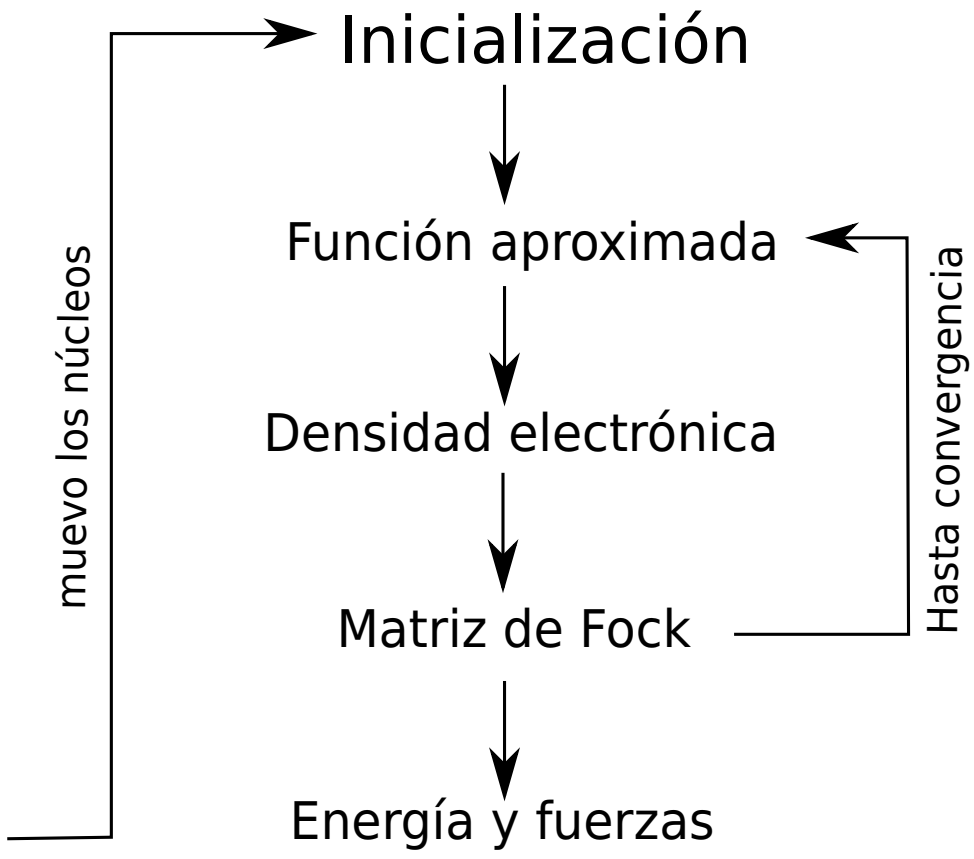
... se llega a que

$$\rightarrow \rho(\vec{r}) = \sum_{i=1}^{N_{oc}} \left| \sum_{k=1}^{M} C_{ik}\phi_k(\vec{r}) \right|^2$$

Se trata de encontrar los coeficientes C_{ik} que minimicen la energía.

Esquema de cálculo





Energía de Intercambio y Correlación

- Múltiples métodos de aproximación, el más simple es LDA:

$$E_{xc}[\rho] = \int \rho(\vec{r})\varepsilon_{xc}(\rho(\vec{r}))d\vec{r}$$

ε_{xc} : funcional de intercambio y correlación

- Si sólo depende de la densidad.
- Otros método más preciso: GGA (Global Gradient Approach).

$$E_{xc}[\rho] = \int \rho(\vec{r})\varepsilon_{xc}(\rho(\vec{r}), \nabla\rho(\vec{r}))d\vec{r}$$

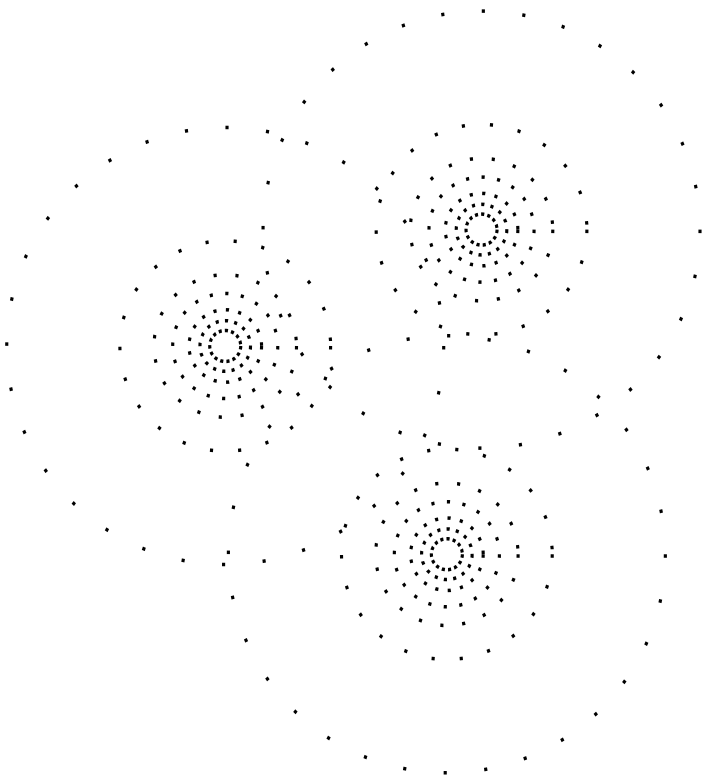
No existe solución analítica de esta integral, se integra numéricamente sobre una grilla.

$$I = \int F(\vec{r})d\vec{r}$$

$$I \simeq \sum_i \omega_i F(\vec{r}_i)$$

La Grilla

- Se toma una grilla esférica cómo base.
- Superposición de grillas base, c/escalamiento: capas concéntricas, sobre cada átomo
- En sistemas poliatómicos: unión de las grillas de c/átomo



Las funciones.

La función de onda se describe como combinación lineal de un grupo de funciones, llamado Base.
La base puede ser de funciones Gaussianas, ondas planas, etc.
En nuestro caso son Gaussianas, es decir tienen esta pinta:

$$\phi = (x - x_0)^n (y - y_0)^m (z - z_0)^l e^{\alpha |\vec{r} - \vec{r_0}|^2}$$

Donde n, m, y l son número enteros que dan el momento angular a la función.
Usar funciones Gaussianas es venajoso porque reproducen adecuadamente la distribución electrónica y son matemáticamente amigables.

La densidad.

Los electrones estan en orbitales moleculares:

$$\chi_i(\vec{r}) = \sum_{\mu=1}^{M} C_{i\mu}\phi_k(\vec{r})$$

Entonces la densidad electrónica total es la suma de las contribuciones de cada orbital:

$$\rightarrow \rho(\vec{r}) = \sum_{i=1}^{N_{oc}} \left| \sum_{k=1}^{M} C_{ik}\phi_k(\vec{r}) \right|^2$$

O dando vuelta la sumatoria se puede llegar a que:

$$\rightarrow \rho(\vec{r}) = \sum_{j=1}^{M}\sum_{k=1}^{M} P_{jk}\phi_k(\vec{r})\phi_j(\vec{r})$$

Donde P_{ij}es la llamada matriz densidad.

Empezando a pensar algoritmos eficientes.

Costo de calcular la integral de intercambio y corralacion= $\alpha M^2 \times N$ donde M es el número de funciones base y N el número de puntos de la grilla.

¿Que hacemos?

Respuesta: funciones significativas.

funciones significativas.

1. Agrupar los puntos de la grilla.
2. Encontrar las funciones significativas
   ϕ es significativa en un grupo si $\phi(r_i) \geqslant \epsilon$ para algún punto (o lugar) del grupo.

Ahora hay que definir el tamaño de los grupos y el valor de ϵ.

- Grupos muy chicos -> menos funciones x grupo pero más grupos.
- ϵ grande -> menos funciones significativas pero mayor error.

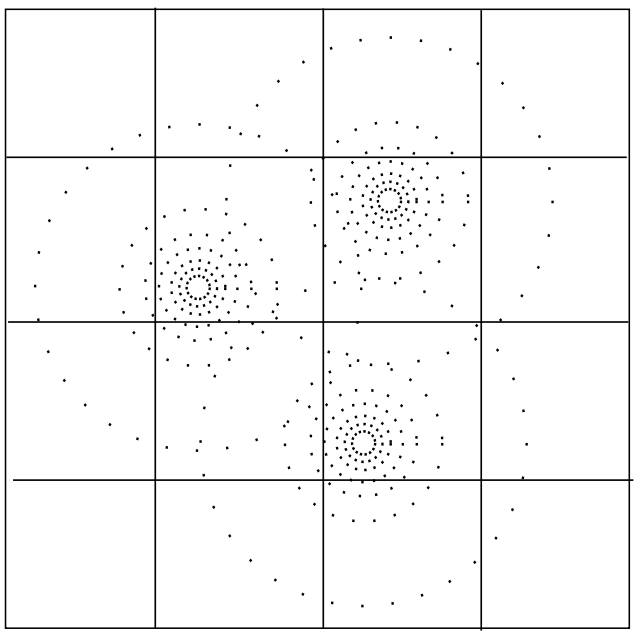
Si hacemos esto para sistemas grandes el número de funciones significativas no crece con el tamaño del mismo.

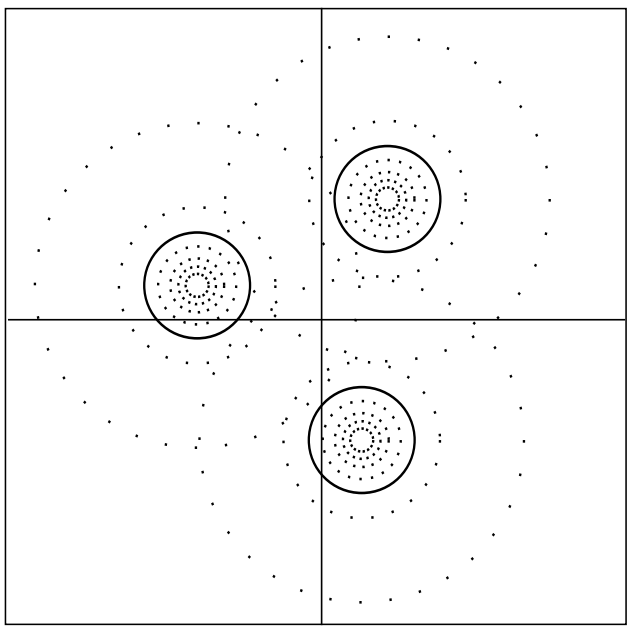
¡Escalamiento lineal!

Algoritmos eficientes.

Ej: Particionado del sistema y funciones significativas.

Algoritmo usual basado en cubos $\Rightarrow$Muchos cubos con pocos puntos (malo para GPU).
Algoritmo basado en esferas y cubos $\Rightarrow$Mucho más eficiente.





El nuevo elefante: Coulomb

$$E_{Coul} = \frac{1}{2} \int \int \frac{\rho(\vec{r}_1)\rho(\vec{r}_2)}{r_{12}} d\vec{r_1} d\vec{r_2}$$

Donde

$$\rho(r) = \sum_{i(ocup).j(Base)k(base)} \sum \sum C_{ij} C_{ik} \Phi_i \Phi_j = \sum_{i(base)j(base)} \sum P_{ij} \Phi_i \Phi_j$$

$$\rho \cong \tilde{\rho} = \sum_{Auxiliar} C_k G_k$$

$$E_{Coul} \approx \int \int \frac{\rho(\vec{r}_1)\tilde{\rho}(\vec{r}_2)}{r_{12}} d\vec{r_1} d\vec{r_2} - \frac{1}{2} \int \int \frac{\tilde{\rho}(\vec{r}_1)\tilde{\rho}(\vec{r}_2)}{r_{12}} d\vec{r_1} d\vec{r_2}$$

Coulomb

- Se pueden tirar términos basado en el teorema de prouctos de Gaussianas.

$$\int e^{-a(\vec{r}-\vec{r_a})^2} \times e^{-b(\vec{r}-\vec{r_b})^2} = \int e^{-\frac{ab}{a+b}(\vec{r_a}-\vec{r_b})^2} e^{-(a+b)(\vec{r}-\frac{a\vec{r_a}+b\vec{r_b}}{a+b})}$$

- Escala cuadráticamente.
- Se puede guardar en memoria y no recalcular en cada iteración (los productos de las Gaussianas no cambian, sólo los coeficientes P_{ij})
- El cálculo de las derivadas (para la fuerza) es lo más demandante para sistemas medianos (HEMO), junto con la integral QM/MM.

Oootro tema: Precisión.

Energía de formación de un agregado.

$$24\,H_2O \longrightarrow (H_2O)_{24}$$

$$\nabla E = E((H_2O)_{24}) - 24E(H_2O)$$

$$\nabla E = -1832{,}896H - 24 \times (-76{,}352H) = 0{,}437H$$

La presición química es del orden de 1 kcal/mol o sea 10^{-3}Hartrees!

Es indispensable usar presición alta siempre, no??

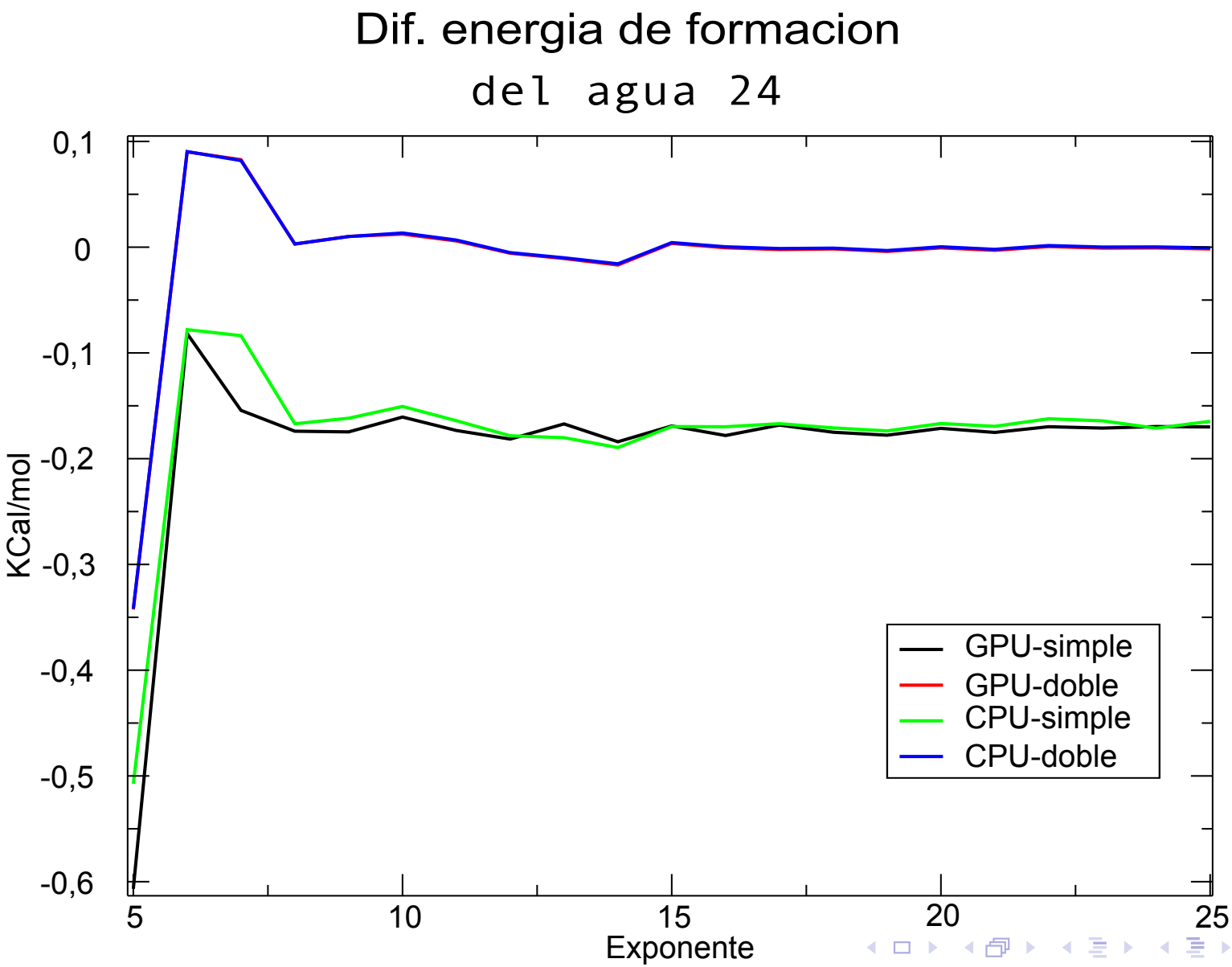
Respuesta: no

- Se puede usar precisión simple para muchas partes manteniendo la calidad del resultado.
- Para otras se puede usar precisión mixta.
- Mejora los tiempos de cálculo (¡sobre todo en GPUs!).
- Disminuye el uso de memoria (muy útil en Coulomb).

uso de presición simple.

En la integral de intercambio y correlación para todo salvo para la acumulación.
Para las integrales de Coulomb se aplica el mismo criterio basado enel producto de Gaussianas. Si son muy chicas, se tiran, si tienen un valos intermedio se hacen en simple, si tienen un valor mayor en doble.

Calidad.


Dif. energia de formacion
del agua 24
KCal/mol
0,1
0
-0,1
-0,2
-0,3
-0,4
-0,5
-0,6
5
10
15
20
25
Exponente
GPU-simple
GPU-doble
CPU-simple
CPU-doble

Calidad

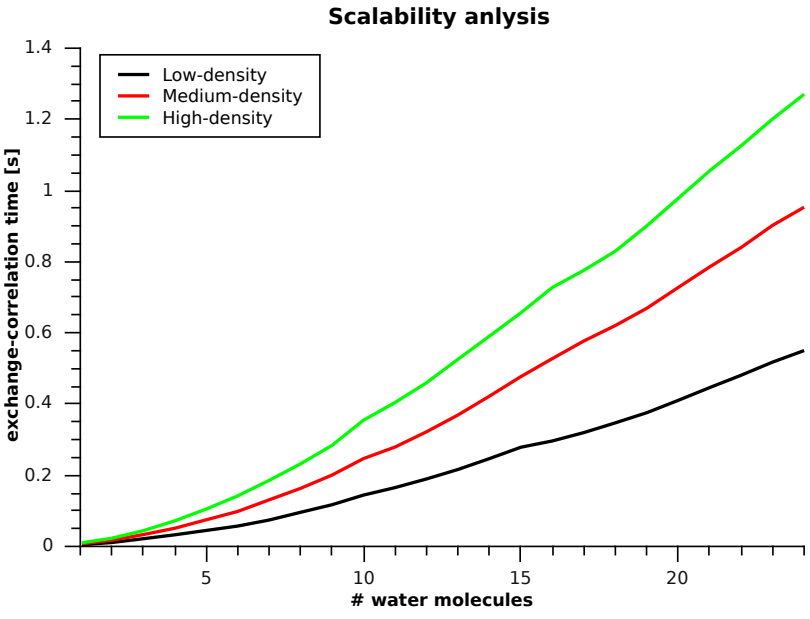
- Usando precision simple se comete un error en la energía de 0,2 Kcal/mol para los sitemas mas grandes (incluso para la valinomicina, de 168 átomos).
- Con exponentes mayores a 10 ya se obtiene un buen resultado.
- GPU en doble es equivalente a CPU en doble.

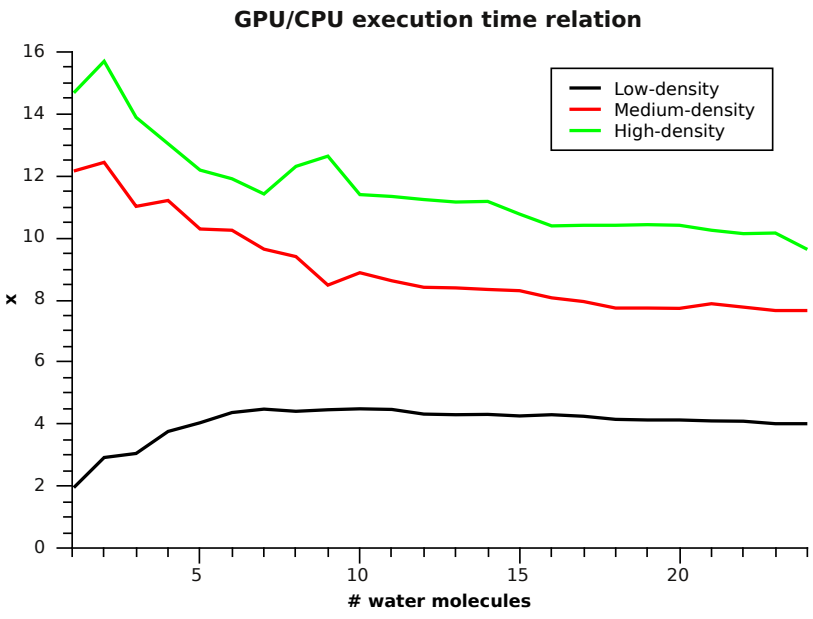
Parametros nuevos.

Tienen impacto en la performance y la calidad.

- max_function_exponent => Define el valor de las funciones que se tiran.
- Tamaño de los cubos.
- Tamaxo de las esferas (entre 0. y 1.)
- rmax => Define los pares de Gaussianas que se tiran cuando hace Coulomb.
- rmaxs => Define los pares de Gausianas que se hacen en doble.

Escalamiento y aceleración.





	Taxol-631G		Valinomicina-631G		Hemo-DZVP	
	GPU	CPU/GPU	GPU	CPU/GPU	GPU	CPU/GPU
Generacion de la grilla	0.99	19.53	1.63	18.88	0.42	18.79
Iteración SCF	3.70	8.19	6.40	6.03	2.92	9.13
Intercambio y correlación	2.41	11.96	3.5	10.09	2.3	11.26

Garchamber

Ahora lo bueno, a este código mejorado lo "pegué" al amber.

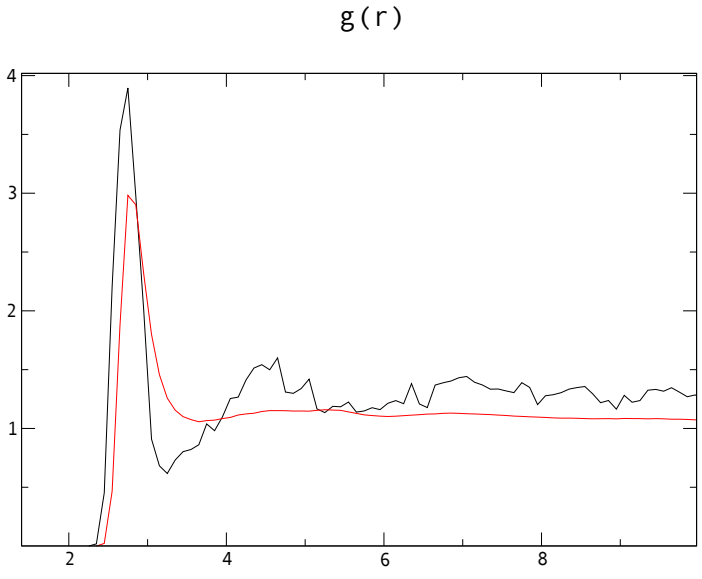
- Usé buena parte del código de QM/MM de Amber (para DFTB y PM3).
- Neceista un achivo de input extra con las bases y los parámetros del DFT. (y se tiene que llamar input)
- Pase todo a memoria dinámica (digamos un 90 %) y las variables en un módulo (no mas COMM).
- Compilado como biblioteca.
- Cut off, no Ewald.

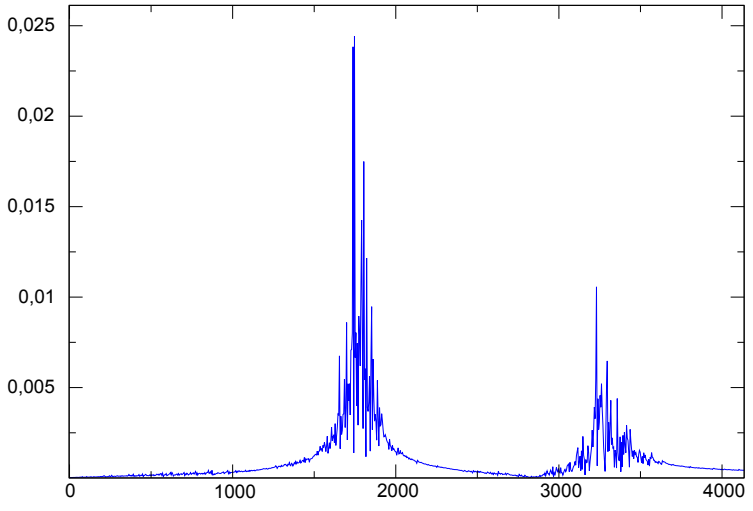
¿Cómo anda?

Sistema	segundos por paso de dinámica
agua1 en agua	0.56
agua2 en agua	1.3
agua3 en agua	2.3
Hemo O2	180
trip en agua	20

Resultados para el agua

40000 pasos de dinámica en 6 horas y media





¿Qué se logró?

1. Una implementación de DFT con variaciones novedosas:
   - Esferas $\rightarrow$ mejora desempeño en GPU
   - En sistemas relativamente pequeños (útiles para QM-MM):

- Mejor rendimiento usando GPU
  - Speedup: entre cerca de $10x$
- Poco uso de memoria
  - Se podría aplicar a sistemas más grandes
- Buena calidad numérica.
  - ¡Solo con precisión simple!
  - Mejor con presición doble.

Lo aprendido

- Límitaciones/virtudes de los GPU
  - Cómo encarar futuros problemas (*divide & conquer*)
- Aspectos de performance
  - Principal: acceso a memoria
    - Patrón de acceso eficiente, generalizable a otros problemas

Trabajo a futuro (se aceptan voluntarios)

- Hacer mas amigable el input (bases en un formato mas estandar y etc.)
- Hacer UHF (se supone que es fácil)
- Mejorar el método de convergencia.
- Integrales de Coulomb y QM/MM en GPU.
- Outputs de orbitales en un formato graficables.
- ¿Condiciones periódicas de borde?, ¿cutt off por residuo?
- ¿Método lineal para las integrales de coulomb? (para sistemas enormes, Ej: Proteínas enteras.)

Autores.

Lic. Matías Nitsche.
Dr. Darío estrin.

Gracias!