Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Física

(ANEXO)

Simulaciones *ab initio* de la adsorción de moléculas gaseosas en MOFs dopados

Autor: Luis Javier Amo Grasa
Tutor: Luis Miguel Molina Martín
Año 2022

Índice general

1. *Software* empleado **1**
1.1. GPAW 1
1.2. *makeinput* 4
1.3. recorta-MOF 5

2. Procedimiento detallado **9**

3. Figuras de los resultados **13**
3.1. Construcción de moléculas gaseosas y estructuras base 14
3.2. Adsorción de gases en átomos y moléculas diatómicas 17
3.3. Adsorción de gases en clústeres metálicos 21
3.3.1. Disociaciones de CH4 25
3.4. Adsorción de gases y átomos de metales de transición en el MOF-5 26
3.5. Adsorción de gases en MOF-5 dopado con átomos de metales de transición 30
3.6. Adsorción de gases en MOF-5 interpenetrado 34

Índice de figuras

2.1. Interfaz del *Xmakemol*. 9

3.1. Moléculas gaseosas 14
3.2. Átomos metálicos 14
3.3. Moléculas diatómicas metálicas 14
3.4. Clústeres metálicos romboédricos 14
3.5. Clústeres metálicos tetraédricos 15
3.6. MOF-5 16
3.7. MOF-5 interpenetrado 16
3.8. Adsorciones de gases en átomos metálicos 17
3.9. Adsorciones de CO2 en Cu2 18
3.10. Adsorciones de CH4 en Cu2 18
3.11. Adsorciones de CO2 en Ag2 19
3.12. Adsorciones de CH4 en Ag2 19
3.13. Adsorciones de CO2 en Au2 20
3.14. Adsorciones de CH4 en Au2 20
3.15. Adsorciones de CO2 en Cu4 21
3.16. Adsorciones de CH4 en Cu4 22
3.17. Adsorciones de CO2 en Ag4 23
3.18. Adsorciones de CH4 en Ag4 23
3.19. Adsorciones de CO2 en Au4 24
3.20. Adsorciones de CH4 en Au4 24
3.21. Disociaciones de CH4 25
3.22. Adsorciones de CO2 en MOF-5 26
3.23. Adsorciones de CH4 en MOF-5 27
3.24. Adsorciones de Cu en MOF-5 28
3.25. Adsorciones de Au en MOF-5 29
3.26. Adsorciones de CO2 en MOF-5 dopado con Cu 30
3.27. Adsorciones de CH4 en MOF-5 dopado con Cu 31
3.28. Adsorciones de CO2 en MOF-5 dopado con Au 32
3.29. Adsorciones de CH4 en MOF-5 dopado con Au 33
3.30. MOF-5 interpenetrado. Celda primitiva y celda manipulada 34
3.31. Poros en el MOF-5 interpenetrado 34
3.32. Adsorciones de CO2 en el MOF-5 interpenetrado 35
3.33. Adsorciones de CH4 en el MOF-5 interpenetrado 36

Capítulo 1

Software empleado

1.1. GPAW

A continuación vamos a mostrar un ejemplo *script* de *Python* que utiliza GPAW.
Estos archivos aparecen como `archivo.py`

```
# Librerias python para importar
from ase import Atom, Atoms
from gpaw import GPAW, FermiDirac, Mixer, MixerSum, MixerDif
from ase.optimize import BFGS
import numpy as np
from ase.io import write
from ase.constraints import FixAtoms
from ase.constraints import FixBondLength

# Introducir coordenadas atomicas y celda de simulacion
molecula = Atoms([
Atom('O',( 0.000000,  0.000000,  0.000000)),
Atom('O',( 3.000000,  0.000000,  0.000000)),
Atom('C',( 1.500000,  0.000000,  0.000000))],
cell=(12.0, 12.0, 12.0))

# Centramos el sistema
molecula.center()

# Parametros calculo
calc = GPAW(xc='PBE', nbands = 15, txt='co2-relax-nonspin-1.txt',
maxiter=250, occupations=FermiDirac(width=0.05),
convergence={'energy': 0.0005,  # eV / electron
'density': 1.0e-5,
'eigenstates': 2.0e-7,  # eV^2 / electron
'bands': 'occupied'})

# Funcion que manda hacer los calculos
molecula.set_calculator(calc)

# Si solo queremos calcular la energia sin relajacion
# estructural, usar el comando en la linea siguiente
# molecula.get_potential_energy()

```

```
# Calcular relajacion estructural
relax = BFGS(molecula, trajectory='co2-relax-nonspin-1.traj')
relax.run(fmax=0.015, steps=50) # precision de la relajacion estructural

calc.write('co2-relax-nonspin-1.gpw')
# saca eigenvalues y densities
# Si queremos analizar orbitales se cambia a:
# calc.write('archivo.gpw', mode='all')
write ('geometry-CO2-nonspin-1.xyz',molecula)
```

Vamos a explicar cada uno de los parámetros a modificar dependiendo del sistema que estemos estudiando.

- **Clúster de entrada**
  En él se incluye el símbolo químico del átomo y sus coordenadas cartesianas en ángstroms. Por ejemplo describiríamos el clúster de CO2 así:

```
molecula = Atoms([
Atom('O',( 0.000000,  0.000000,  0.000000)),
Atom('O',( 3.000000,  0.000000,  0.000000)),
Atom('C',( 1.500000,  0.000000,  0.000000))]
```

- **El tamaño de la celda**
  En el caso de querer usar una celda paralelepípeda usaremos:

```
cell=(x, y, z)
```

  Si queremos otro tipo de celda tendremos que dar la terna de vectores que conformen la celda de la forma:

```
cell=((x1, y1, z1),( x2, y2, z2),( x3, y3, z3))
```

  Si queremos crear una estructura que se replique en alguna dirección espacial tenemos que añadir el objeto *periodic bounded conditions* (`pbc`):

```
pbc=(direccion x, direccion y, direccion x)
```

  Pondremos un 1 en las direcciones que queremos que se replique y 0 en las que no. Por ejemplo, si queremos que la estructura se replique en las tres direcciones, como es el caso del MOF-5, escribiríamos:

```
molecula = Atoms([
Atom('O',( 0.000000,  0.000000,  0.000000)),
Atom('O',( 3.000000,  0.000000,  0.000000)),
Atom('C',( 1.500000,  0.000000,  0.000000))],
cell=(12.0, 12.0, 12.0), pbc=(1, 1, 1))
```

- **Parámetros de cálculo**
  Creamos una lista de parámetros necesarios para el cálculo:

```
calc = GPAW(xc='PBE', nbands = 15, txt='archivo.txt',
maxiter=250, occupations=FermiDirac(width=0.05),
convergence={'energy': 0.0005, 'density': 1.0e-5,
'eigenstates': 2.0e-7, 'bands': 'occupied'})
```

Con `xc` fijamos el funcional de intercambio-correlación que usaremos, en nuestro caso el de Perdew-Burke-Ernzerhof (PBE).
Con `nbands` fijamos el número de bandas electrónicas. El código solo hace cálculos con los electrones de valencia y en cada banda caben dos electrones debido al principio de exclusión de Pauli. Sin embargo, debemos añadir unas bandas desocupadas auxiliares (un $25\,\%$ de las de conducción aproximadamente) que ayudan con la convergencia de las funciones de onda y a analizar el GAP valencia-conducción. Calculamos el número de bandas de un compuesto A_nB_m como:

$$nbands = \frac{1}{2}\left[valencia(A)\cdot n + valencia(B)\cdot m\right] + bandas\ desocupadas \tag{1.1}$$

Con `txt` nombramos el `archivo.txt` en el que se guardaran los resultados de los cálculos.
Con `maxiter` fijamos el número máximo de iteraciones que realizará al resolver las ecuaciones de Kohn-Sham.
Con `occupations=FermiDirac(width=0.05)` fijamos el GAP entre estados ocupados y desocupados.
`width=0.05` controla lo estrecho que es el margen al aplicar la distribución de Fermi-Dirac, es decir la "distancia" entre los estados ocupados y desocupados.
Con `convergence` fijamos unos criterios para los que damos por convergidas las funciones de onda y finaliza el cálculo iterativo. Estos criterios son cambios en la energía, densidad, autovalores y bandas ocupadas respectivamente.

- **Relajación estructural**
  Fijamos la relajación estructural con:

```
relax = BFGS(molecula, trajectory='archivo.traj')
```

Donde `BFGS` es el algoritmo empleado, `molecula` el objeto sobre el que se aplica la relajación y `trajectory` el archivo donde se guardan los datos de las relajación estructural, en nuestro caso `archivo.traj`.
Fijamos unos parámetros de parada de la relajación estructural con:

```
relax.run(fmax=0.015, steps=250)
```

Donde `fmax` es la fuerza neta y, si es menor que este valor fijado, el cálculo de la relajación se da por finalizado. El parámetro `steps` es el máximo de desplazamientos atómicos que permitimos, si la fuerza tras todos lo `steps` es mayor que `fmax` tenemos que reiniciar el cálculo desde la última geometría obtenida.
Es recomendable fijar un número máximo de desplazamientos atómicos ya que podemos llegar a una situación el la que se alternen dos valores, de fuerza neta, a la salida de cada iteración y el cálculo no converja nunca. En ese caso se relanza el cálculo a partir de la última geometría obtenida.

1.2. *makeinput*

Este programa, escrito en C, se encarga de pasar los datos del clúster escritos en un formato `archivo.xyz` que usa el *Xmakemol* a un formato `output.txt` que "entiende" el código GPAW. El programa está escrito en un archivo llamado `makeinput.c`:

```
#include <stdio.h>

int main(int argc, char *argv[]) {

    if ( argc != 2 ) /* argc should be 2 for correct execution */
    {
        /* We print argv[0] assuming it is the program name */
        printf( "usage: %s filename \n", argv[0] );
    }
    else
    {
        // We assume argv[1] is a filename to open
        FILE *file = fopen( argv[1], "r" );

        /* fopen returns 0, the NULL pointer, on failure */
        if ( file == 0 )
        {
            printf( "Could not open file\n" );
        return 0;
        }


int i, N;

FILE *file2 = fopen("output.txt", "w");

char aux,letra[2];

fscanf(file,"%d",&N);
fscanf(file,"%c",&aux);
if(aux!=' ') fscanf(file, "%*[^\n]\n", NULL);


double POS[3];


fprintf(file2,"molecula = Atoms([ \n");

for (i=0;i<N;i++) {
fscanf(file," %c%c  %lf  %lf  %lf",&letra[0],&letra[1],&POS[0],&POS[1],&POS
    [2]);
if (i==N-1) {
if(letra[1]==' ')
fprintf(file2,"Atom('%c',( %lf,  %lf,  %lf))], \n",letra[0],POS[0],POS[1],
    POS[2]);
else
fprintf(file2,"Atom('%c%c',( %lf,  %lf,  %lf))], \n",letra[0],letra[1],POS
    [0],POS[1],POS[2]); }
else {
```

```
47 if(letra[1]==' ')
48 fprintf(file2,"Atom(' %c',( %lf,  %lf,  %lf)), \n",letra[0],POS[0],POS[1],
       POS[2]);
49 else
50 fprintf(file2,"Atom(' %c %c',( %lf,  %lf,  %lf)), \n",letra[0],letra[1],POS
       [0],POS[1],POS[2]); }
51 }
52
53 fclose(file);
54 fclose(file2);
55
56  }
57 }
```

Lo compilamos y después lo usamos haciendo `makeinput` seguido del `archivo.xyz`:

```
$ makeinput archivo.xyz
```

1.3. recorta-MOF

Este programa, escrito en C, añade átomos a la celda primitiva del MOF-5 para recrear su estructura y poder visualizar átomos que se encuentran en celdas vecinas.
El programa está escrito en un archivo llamado `recorta-MOF.c`:

```
1  #include <stdio.h>
2
3  int main(int argc, char *argv[]) {
4
5      if ( argc != 2 ) /* argc should be 2 for correct execution */
6      {
7          /* We print argv[0] assuming it is the program name */
8          printf( "usage: %s filename \n", argv[0] );
9      }
10     else
11     {
12         // We assume argv[1] is a filename to open
13         FILE *file = fopen( argv[1], "r" );
14
15         /* fopen returns 0, the NULL pointer, on failure */
16         if ( file == 0 )
17         {
18             printf( "Could not open file\n" );
19         return 0;
20         }
21
22
23 int i, h, k, l, N, N2=0;
24
25 FILE *file2 = fopen("output.xyz", "w");
26
27
28 char aux;
29
```

```
fscanf(file,"%d",&N);
fscanf(file,"%c",&aux);
if(aux!=' ') fscanf(file, "%*[^\n]\n", NULL);

double a = 13.15;

double red[3][3]={{a,a,0.0},{a,0.0,a},{0.0,a,a}};

double POS[N][3],POS2[40000][3],POS3[5000][3];
char letra[N][2],letra2[40000][2],letra3[5000][3];

double centro[3] = {-6.6,-6.6,0.0};
double min[3], max[3];

int zona[3] = {1,1,1};
char yes;

printf("Celda por defecto: +1, +1, +1 \n");
printf("Cambiamos celda?  s/n \n");
scanf(" %c",&yes);
if(yes== 's'||yes== 'S') {
printf("Introduzca 3 numeros de valores +1 o -1: \n");
scanf("%d %d %d",&zona[0],&zona[1],&zona[2]);
}

double margen = 2.0;
printf("Cambiamos espacio de margen?  s/n \n");
scanf(" %c",&yes);
if(yes== 's'||yes== 'S') {
printf("Introduzca valor de margen (>2.0): \n");
scanf("%lf",&margen);
}


for (i=0;i<3;i++) {
        if(zona[i]>0) {
       min[i] = centro[i] - margen;
       max[i] = centro[i] + a + margen;
        }
    else {
       min[i] = centro[i] - a - margen;
       max[i] = centro[i] + margen;
    }
}


for (i=0;i<N;i++) {
fscanf(file," %c%c  %lf  %lf  %lf",&letra[i][0],&letra[i][1],&POS[i][0],&
   POS[i][1],&POS[i][2]);
//fscanf(file, "%*[^\n]\n", NULL);
}



for(h=-2;h<=2;h++) {
```

```
84          for(k=-2;k<=2;k++) {
85                  for(l=-2;l<=2;l++) {
86                 for (i=0;i<N;i++) {
87    POS2[N2][0] = POS[i][0] + h*red[0][0] + k*red[1][0] + l*red[2][0];
88    POS2[N2][1] = POS[i][1] + h*red[0][1] + k*red[1][1] + l*red[2][1];
89    POS2[N2][2] = POS[i][2] + h*red[0][2] + k*red[1][2] + l*red[2][2];
90    letra2[N2][0]=letra[i][0];
91    letra2[N2][1]=letra[i][1];
92    N2 = N2 + 1;
93     }
94           }
95    }
96  }
97
98  int N3 = 0;
99
100 for (i=0;i<N2;i++) {
101
102 if((POS2[i][0]>=min[0])&&(POS2[i][0]<=max[0])&&(POS2[i][1]>=min[1])&&(POS2[
       i][1]<=max[1])&&
103 (POS2[i][2]>=min[2])&&(POS2[i][2]<=max[2]))
104 {
105  letra3[N3][0]=letra2[i][0];
106  letra3[N3][1]=letra2[i][1];
107  POS3[N3][0] = POS2[i][0];
108  POS3[N3][1] = POS2[i][1];
109  POS3[N3][2] = POS2[i][2];
110  N3=N3+1;
111 }
112
113 }
114
115 fprintf(file2,"%d \n",N3);
116 fprintf(file2,"\n");
117
118 for (i=0;i<N3;i++) {
119 if(letra3[i][1]==' ')
120 fprintf(file2,"%c    %lf   %lf   %lf \n",letra3[i][0],POS3[i][0],POS3[i][1],
       POS3[i][2]);
121 else
122 fprintf(file2,"%c%c   %lf   %lf   %lf \n",letra3[i][0],letra3[i][1],POS3[i
       ][0],POS3[i][1],POS3[i][2]);
123 }
124
125
126 fclose(file);
127 fclose(file2);
128
129
130  }
131 }
```

Lo compilamos y después lo usamos haciendo `recorta-MOF` seguido del `archivo.xyz` que queremos manipular:

```
$ recorta-MOF archivo.xyz
```

Capítulo 2

Procedimiento detallado

A continuación vamos a describir un procedimiento general para realizar las simulaciones. Comenzamos creando el clúster de partida con la ayuda del *Xmakemol*, para ello crearemos un `geometria-inicial.xyz` como el ejemplo que se muestra a continuación:

```
3

O     0.000000000000000     0.000000000000000     0.000000000000000
O     3.000000000000000     0.000000000000000     0.000000000000000
C     1.500000000000000     0.000000000000000     0.000000000000000
```

Donde se expresa el número de átomos en la primera línea, y en las siguientes el átomo con su símbolo químico acompañado de su posición en coordenadas cartesianas en ángstroms.
Podemos visualizar el clúster y manipular las posiciones de estos átomos usando el *Xmakemol*. En la Figura 2.1 podemos ver la interfaz del programa.

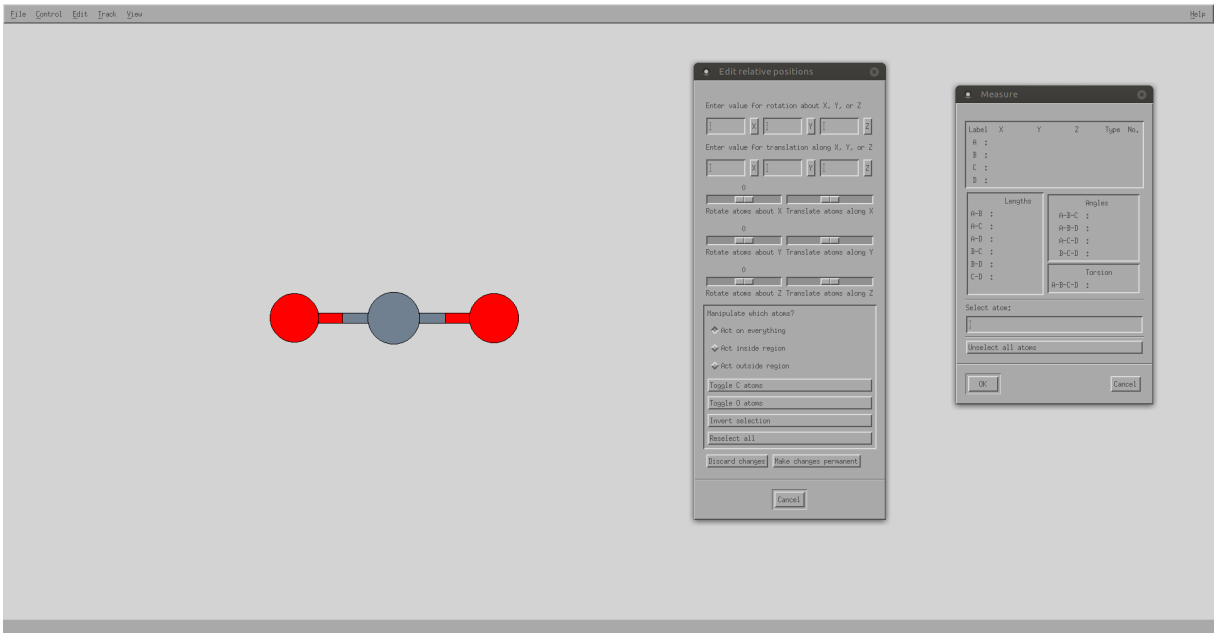

Figura 2.1: Interfaz del *Xmakemol*.

Es importante saber que, cuando estemos manipulando el MOF-5 o el MOF-5 interpenetrado, no podemos mover ni rotar la estructura del MOF; ya que esto cambiaría la base estructural y no conseguiríamos

replicar la red en todo el espacio, por tanto solo podemos mover y rotar los átomos y moléculas que queramos añadirle.
Una vez tenemos los átomos en las posiciones deseadas, queremos pasar del formato que utiliza el *Xmakemol* a uno que pueda introducirse en GPAW. Para ello usamos el programa `makeinput.c` y hacemos la siguiente operación en la terminal:

```
$ makeinput geometria-inicial.xyz
```

Esto nos creará un fichero `output.txt`, el cuál tiene la siguiente forma:

```
molecula = Atoms([
Atom('O',( 0.000000,  0.000000,  0.000000)),
Atom('O',( 3.000000,  0.000000,  0.000000)),
Atom('C',( 1.500000,  0.000000,  0.000000))],
```

Ahora nos tenemos que conectar al superordenador para realizar los cálculos. Para ello hacemos:

```
$ ssh -X direccion_del_servidor
```

Una vez dentro nos movemos a través de los directorios o carpetas y nos situamos en la que queremos usar para hacer el cálculo. Tendremos una *script* de *Python* similar a la mostrada en la sección (1.1), y en ella pegaremos el texto del *output.txt*.
Dentro de la *script* haremos los cambios en los parámetros que consideremos pertinentes dependiendo del sistema a estudiar.
En el caso de los clústeres metálicos hemos usado:

```
cell=(12.0, 12.0, 12.0)
```

En el caso de los MOFs hemos usado:

```
cell=((13.15, 13.15, 0.0),(13.15, 0.0, 13.15),(0.0, 13.15, 13.15)), pbc
    =(1,1,1)
```

Se hicieron cálculos para diferentes valores de celda para ver con cuáles se obtenía la menor energía.
Para cada clúster diferente habrá que calcular el número de bandas como hemos indicado en la sección (1.1).
En el directorio en el que está el *script* también tendremos un archivo `submit.txt` de la forma:

```
nohup mpirun -np 8 -machinefile /home/abenito/hostfiles/hostfileSXX gpaw-python
/home/abenito/directorio/geometria-inicial.py > geometria-final.out
```

En donde `/home/abenito/hostfiles/hostfileSXX` indica el nodo del superordenador en el que se realiza el cálculo, `8` indica el número de procesadores que usa en paralelo
y `geometria-final.out` es el archivo que contendrá los resultados de los cálculos (número de iteraciones, hora, energía DFT y fuerza neta).
Podemos ver un ejemplo de archivo `geometria-final.out` a continuación:

```
BFGS:   0  13:24:57      -17.982756      11.3036
BFGS:   1  13:25:09      -18.884861      11.2861
BFGS:   2  13:25:23      -19.776189      11.0420
BFGS:   3  13:25:36      -20.639279      10.4781
BFGS:   4  13:25:49      -21.449244       9.6240
BFGS:   5  13:26:01      -22.174462       8.3638
BFGS:   6  13:26:14      -22.771598       6.5229
```

```
BFGS:   7  13:26:28      -23.192699       3.9507
BFGS:   8  13:26:40      -23.373640       0.3994
BFGS:   9  13:26:50      -23.374949       0.0531
BFGS:  10  13:26:56      -23.375145       0.0023
```

Para realizar el cálculo debemos conectarnos a un nodo del superordenador, para ello haremos:

```
$ ssh compute-X-X
```

Donde `X-X` indica el número del nodo.
Antes de lanzar el cálculo es conveniente comprobar si hay otros cálculos realizándose en el nodo, para ello podemos hacer:

```
$ qstat -f
```

Y así ver la información de los nodos (cuántos procesadores se están usando).
O bien, dentro del nodo, podemos hacer:

```
$ ps -fu abenito
```

Y así vemos los procesos que se están ejecutando en el nodo.
Una vez en el nodo vamos al directorio donde está `geometria-inicial.py` haciendo:

```
$ cd /home/abenito/directorio/
```

Y para lanzar el cálculo hacemos en el directorio:

```
$ ./submit.txt &
```

El `&` es importante escribirlo para que no colapse la terminal.
Ahora tenemos que esperar a que finalicen los cálculos y estar pendientes de que no colapsen. En caso de que colapsen y no converjan relanzaríamos el cálculo desde la última geometría.
Una vez finalizado el cálculo tendremos distintos tipos de archivos. Si hacemos `ls` en el directorio veremos algo similar a esto:

```
geometria-final.gpw      geometria-final.xyz
geometria-final.traj     geometria-inicial.py
geometria-final.txt      submit.txt
geometria-final.out
```

El archivo `geometria-final.xyz` es la geometría final tras la relajación estructural y la podemos visualizar con el *Xmakemol*. También nos es de interés el archivo `geometria-final.traj`, ya que con el podemos crear un archivo `movie.xyz` en el que visualizar todos los pasos de la relajación estructural usando el *Xmakemol*. Para crear el archivo `movie.xyz` hacemos lo siguiente para los clústeres normales:

```
$ ase-gui -o movie.xyz geometria-final.traj
```

Para el caso de una estructura periódica hacemos:

```
$ ase-gui -r 1,1,1 -o movie.xyz geometria-final.traj
```

Donde el `1,1,1` indica la celda primitiva. Si vamos añadiendo +1, nos añade la celda contigua en una de las direcciones de replicación (x, y, z). Ejemplo, si ponemos `2,1,1` nos representa también la

celda contigua a la primitiva en el eje x. En el caso del MOF-5 interpentrado hemos usado el programa *recorta-MOF.c*, explicado en la sección (1.3), para poder visualizar mejor la estructura. Hacemos:

```
$ recorta-MOF geometria-final.xyz
```

Para hacer los cálculos de energías de corrección a las energías de unión hacemos lo siguiente, en el caso de los clústeres en los que estudiamos adsorciones:

```
$ run-dftd3-cluster geometria-final.xyz
```

Y hacemos lo siguiente, en el caso de los MOFs en los que estudiamos adsorciones:

```
$ run-dftd3-MOF geometria-final.xyz
```

En ambos casos tendremos que fijarnos antes en el archivo `geometria-final.xyz`, para saber como se disponen el fragmento base y el fragmento a adsorber en este archivo.
Finalmente, para traspasarnos un directorio desde el servidor del superordenador a nuestro ordenador personal, haremos el siguiente comando en nuestro ordenador personal:

```
$ scp -r direcciondelservidor:home/abenito/directorio/ .
```

Donde `direcciondelservidor:home/abenito/directorio/` es la ruta del directorio en el servidor y `.` refleja el directorio en el que estamos en nuestro ordenador personal.
En el caso de sólo querer pasarnos archivos haremos:

```
$ scp direcciondelservidor:home/abenito/directorio/archivos .
```

Capítulo 3

Figuras de los resultados

Al igual que en la memoria del Trabajo de Fin de Grado se adjunta la Tabla 3.1 para facilitar la identificación de los átomos en las figuras.
Recordemos que, en las figuras de adsorciones de moléculas gaseosas en el MOF-5 limpio y dopado, están señaladas las moléculas gaseosas con una flecha azul. Y que, en las figuras del MOF-5 interpenetrado, las moléculas gaseosas están señaladas con una flecha azul y el origen de la celda primitiva con una flecha verde.

Tabla 3.1: Guía de identificación de átomos en las figuras.

Átomo	Símbolo	Color
Hidrógeno	H	Blanco
Oxígeno	O	Rojo
Carbono	C	Gris
Zinc	Zn	Marrón
Cobre	Cu	Azul
Plata	Ag	Plata
Oro	Au	Oro

NOTA: Para poder subir el PDF del Anexo a la sede electrónica se ha tenido que bajar la resolución. Es posible que algunas figuras se vean ligeramente borrosas.

3.1. Construcción de moléculas gaseosas y estructuras base

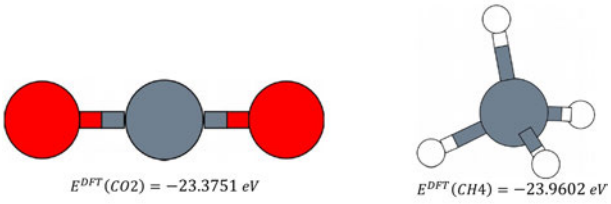

Figura 3.1: Estructuras relajadas obtenidas para CO2 y CH4 y sus energías DFT.

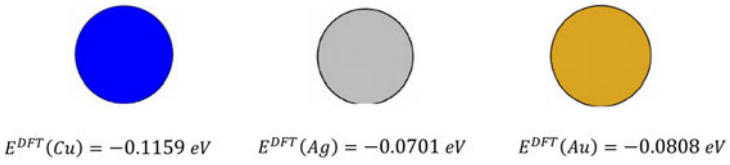

Figura 3.2: Átomos de Cu, Ag y Au con sus energías DFT.

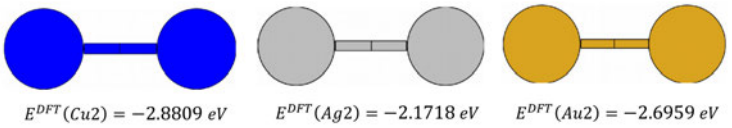

Figura 3.3: Moléculas diatómicas de Cu2, Ag2 y Au2 con sus correspondientes energías DFT.

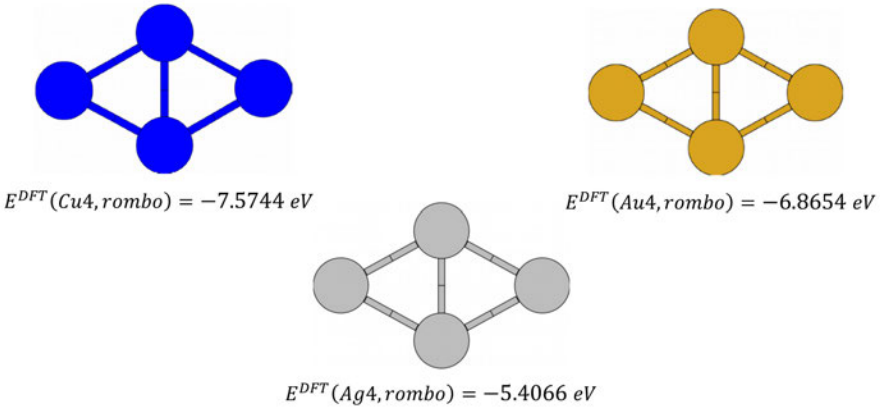

Figura 3.4: Clústeres de Cu4, Ag4 y Au4 con estructura romboédrica y sus correspondientes energías DFT.

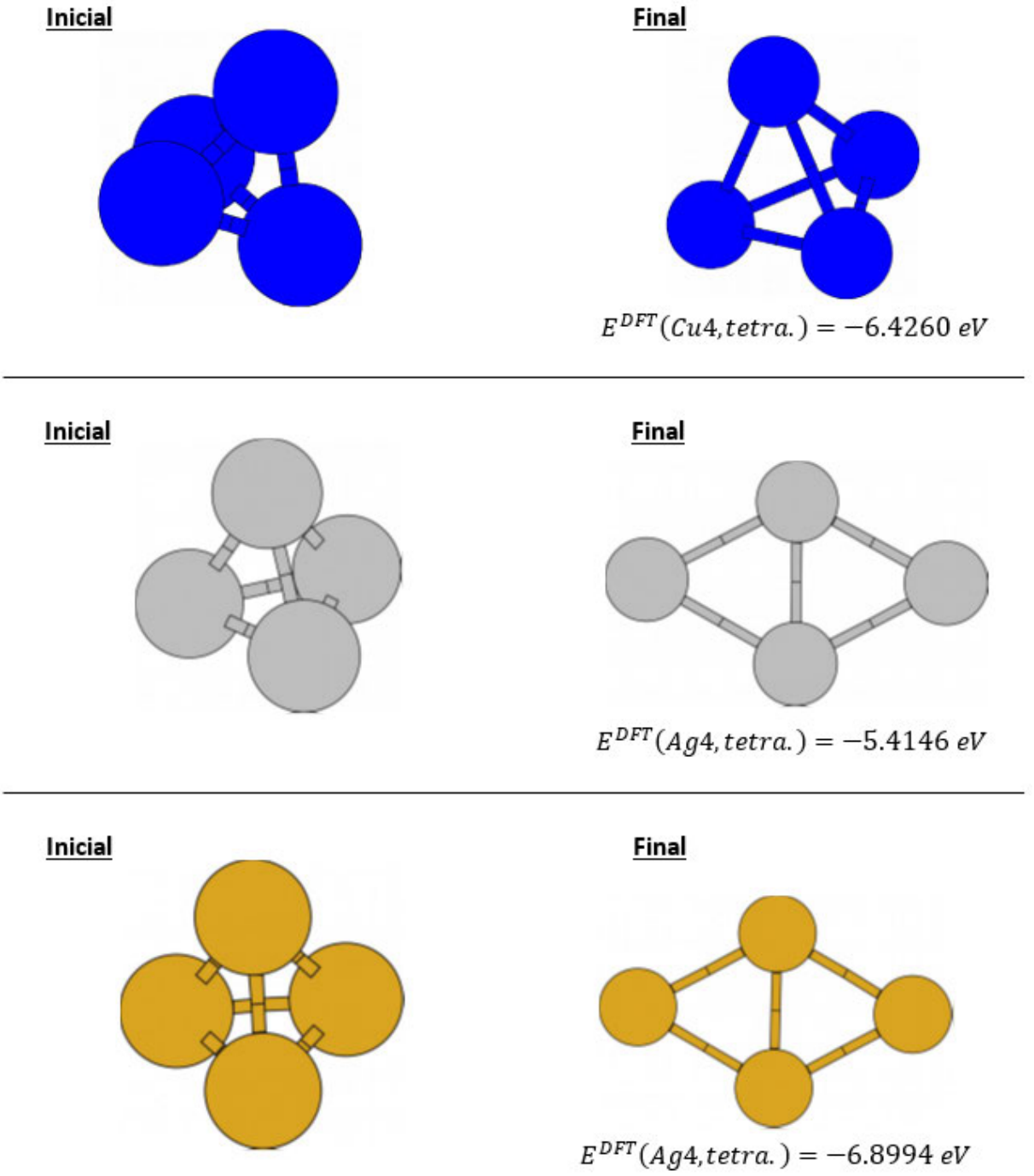

Figura 3.5: Geometrías tetraédricas antes (izq.) y después (dch.) de la relajación estructural del Cu4 (arriba), Ag4 (medio) y Au4 (abajo) junto a sus la energías DFT tras la relajación.

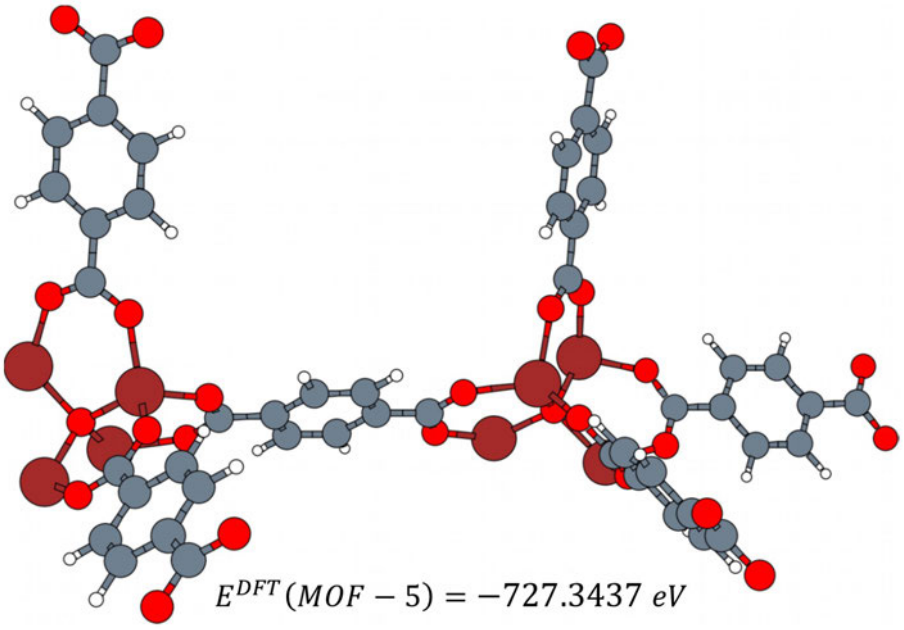

Figura 3.6: Celda primitiva del MOF-5 tras la relajación estructural y su correspondiente energía DFT.

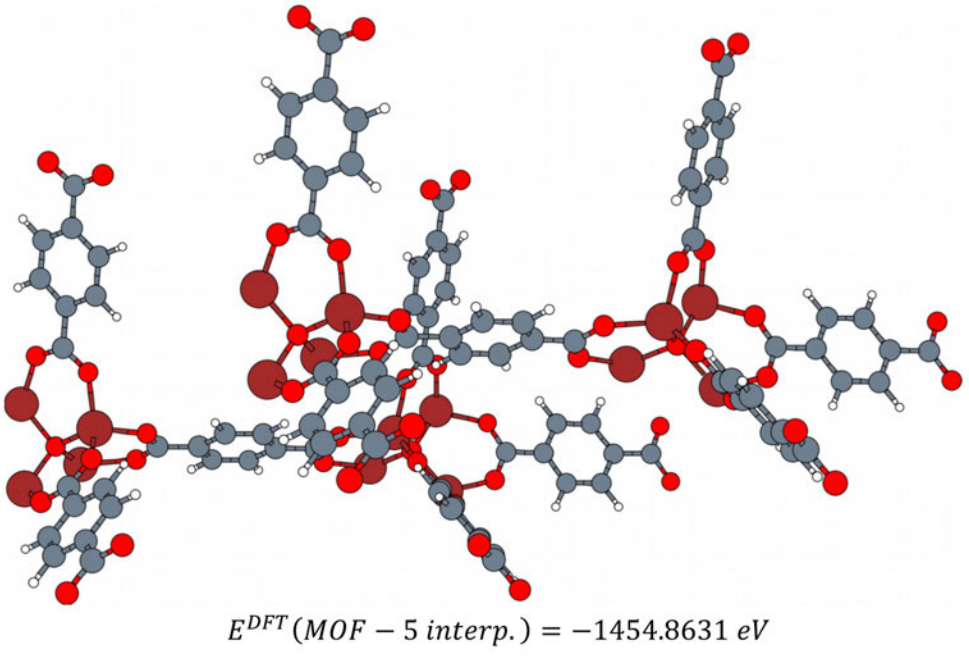

Figura 3.7: Celda primitiva del MOF-5 interpenetrado tras la relajación estructural y su correspondiente energía DFT.

3.2. Adsorción de gases en átomos y moléculas diatómicas

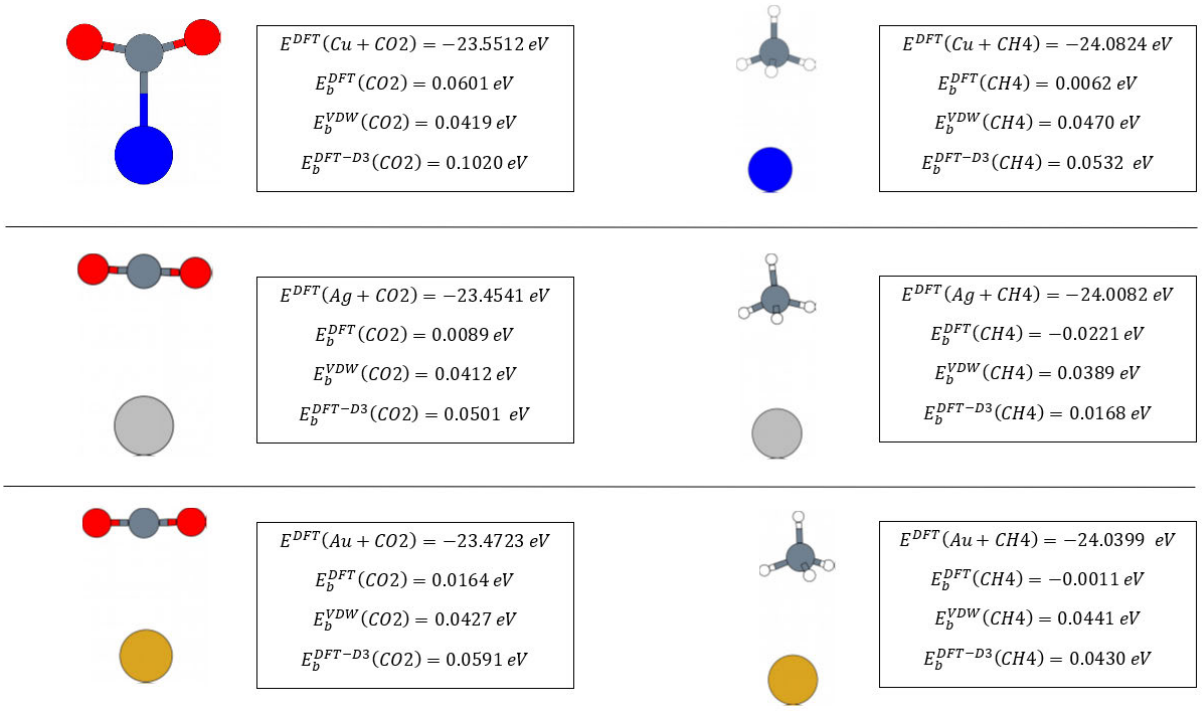

Figura 3.8: Geometrías tras la relajación, energías DFT del sistema y energías de unión del CO2 (izq.) y CH4 (dch.) para Cu (arriba), Ag (medio) y Au (abajo).

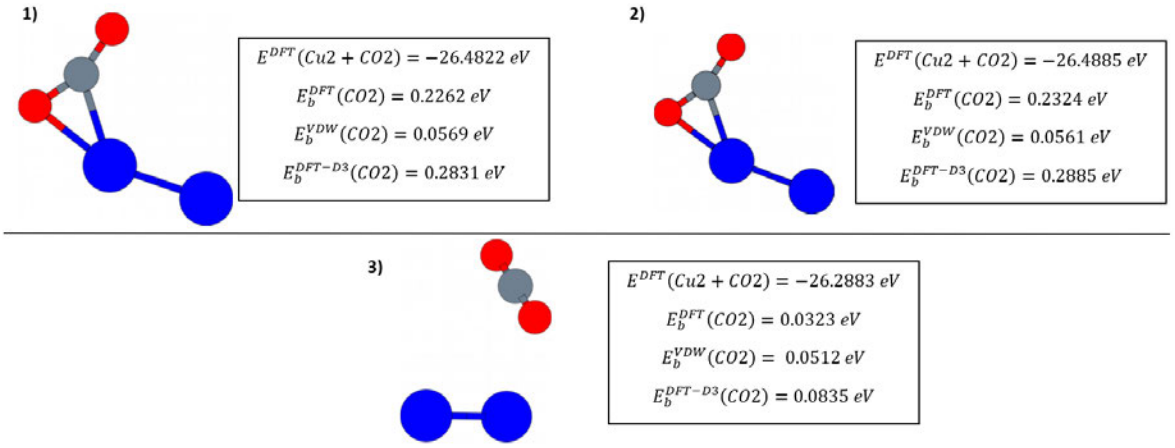

Figura 3.9: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Cu2 más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

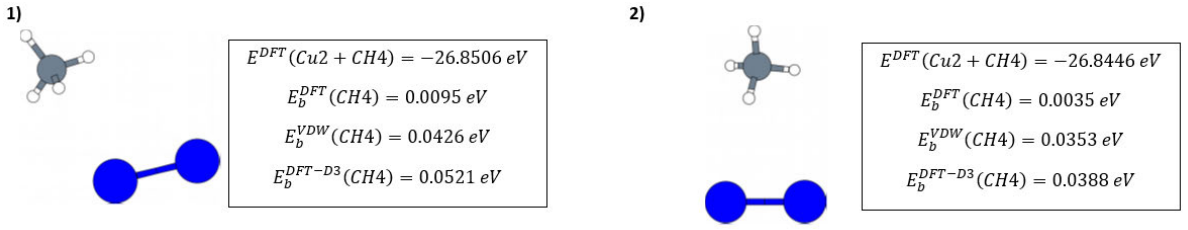

Figura 3.10: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Cu2 más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

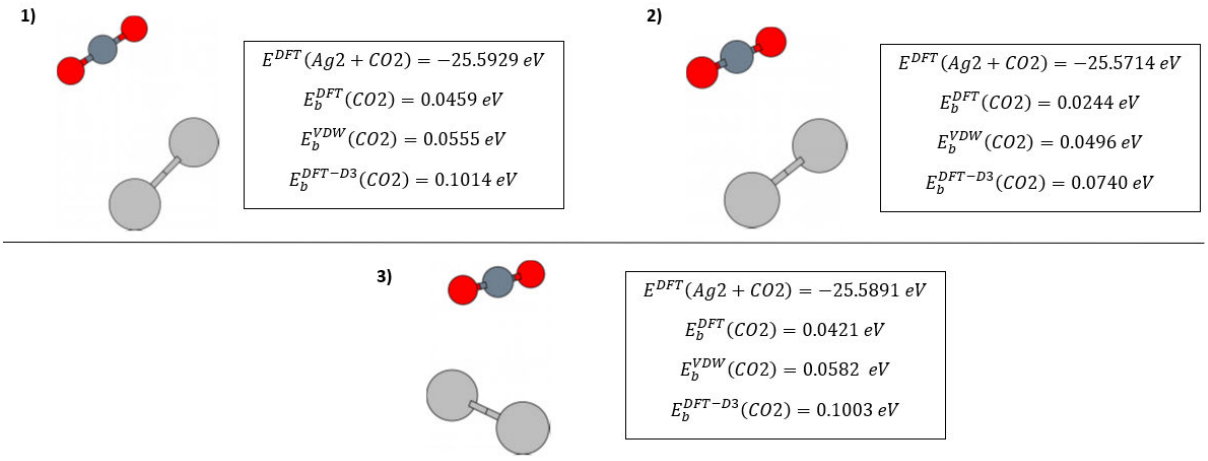

Figura 3.11: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Ag2 más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

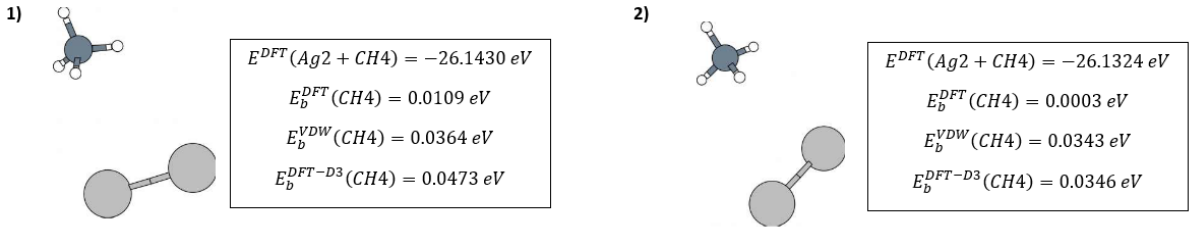

Figura 3.12: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Ag2 más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

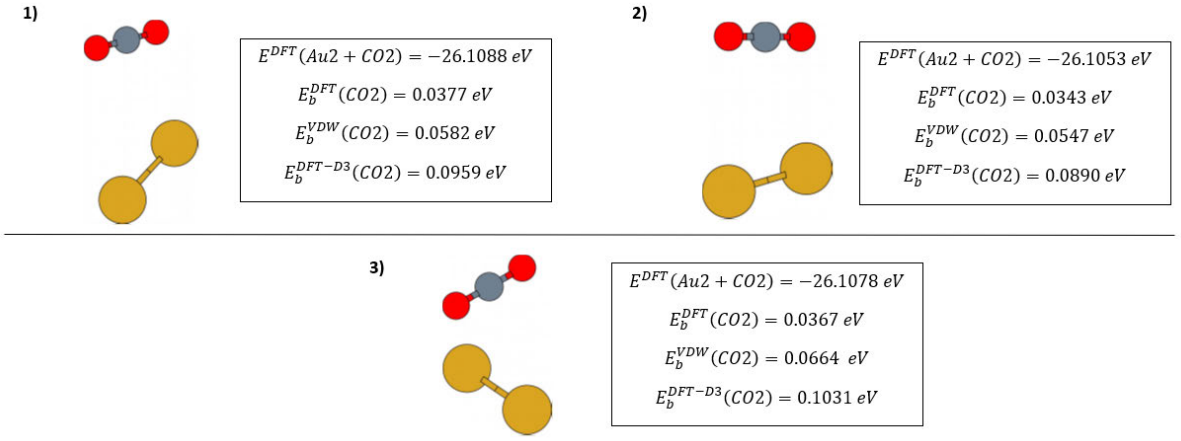

Figura 3.13: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Au2 más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

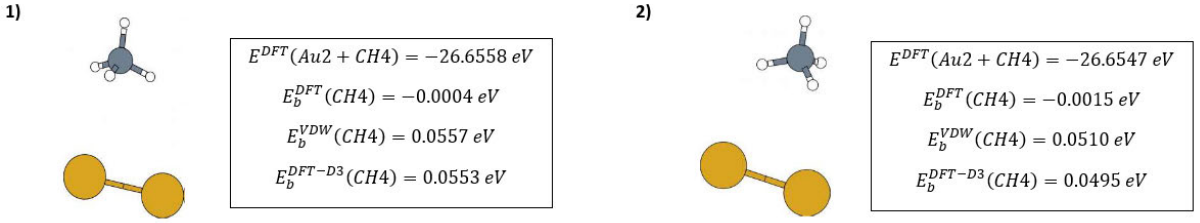

Figura 3.14: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Au2 más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

3.3. Adsorción de gases en clústeres metálicos

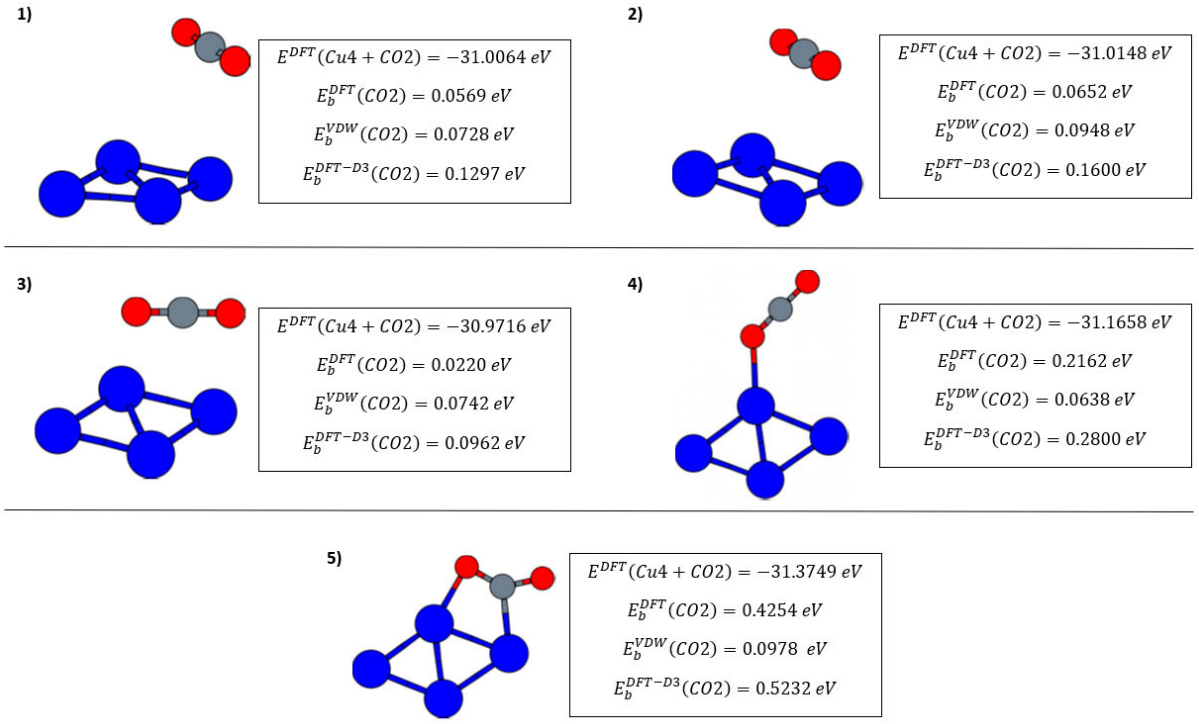

Figura 3.15: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Cu4 romboidal más CO2 junto con las energías DFT del sistema y energías de enlace del CO2.

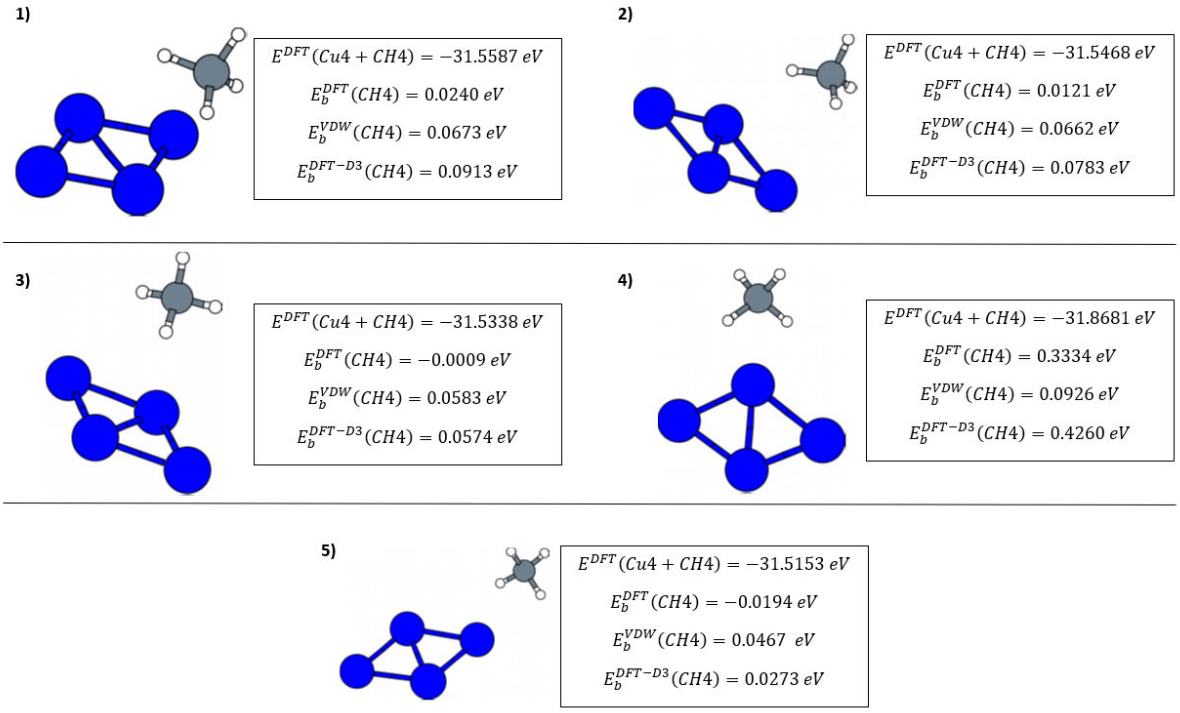

Figura 3.16: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Cu4 romboidal más CH4 junto con las energías DFT del sistema y energías de enlace del CH4.

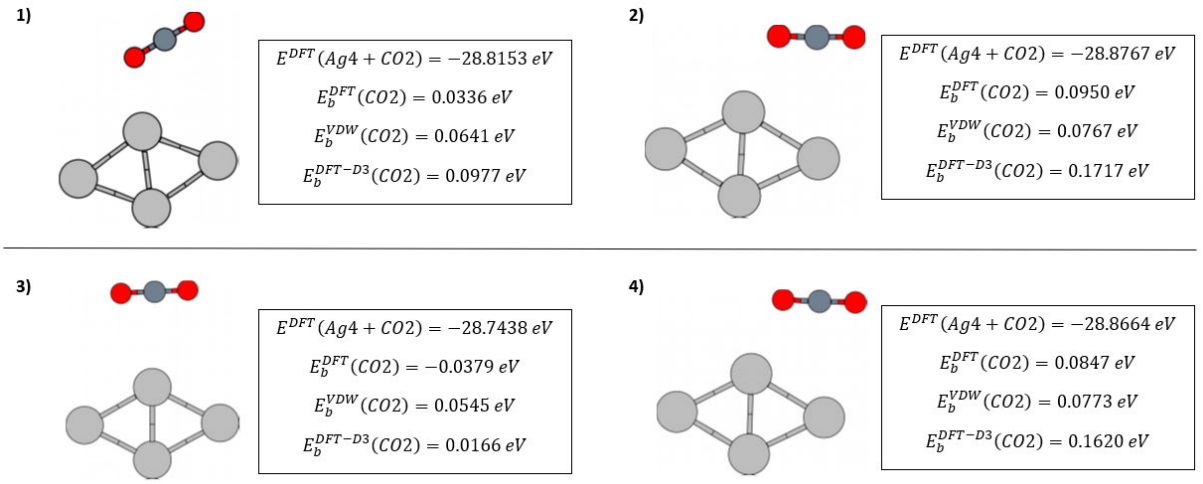

Figura 3.17: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Ag4 romboidal más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

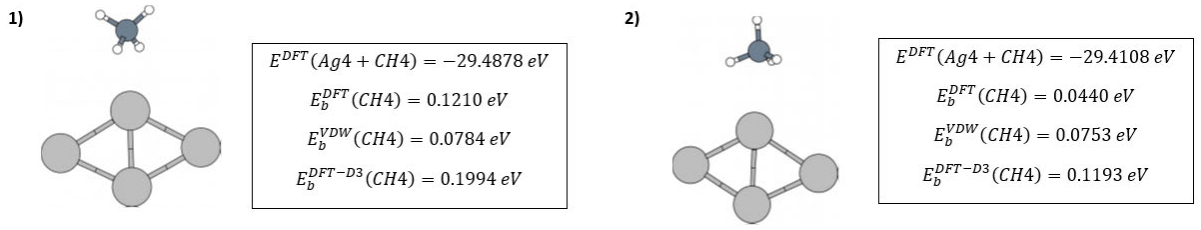

Figura 3.18: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Ag4 romboidal más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

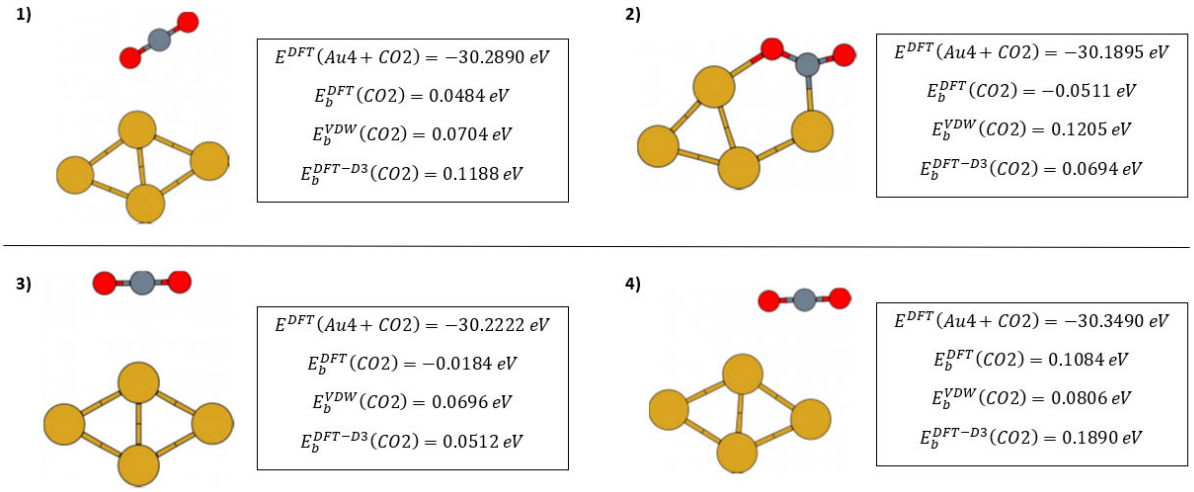

Figura 3.19: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Au4 romboidal más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

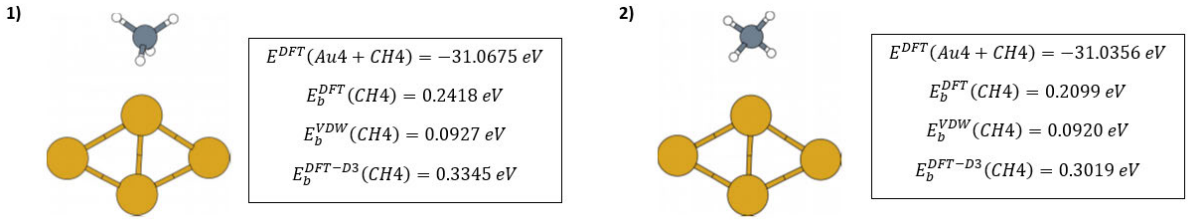

Figura 3.20: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de Au4 romboidal más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

3.3.1. Disociaciones de CH4

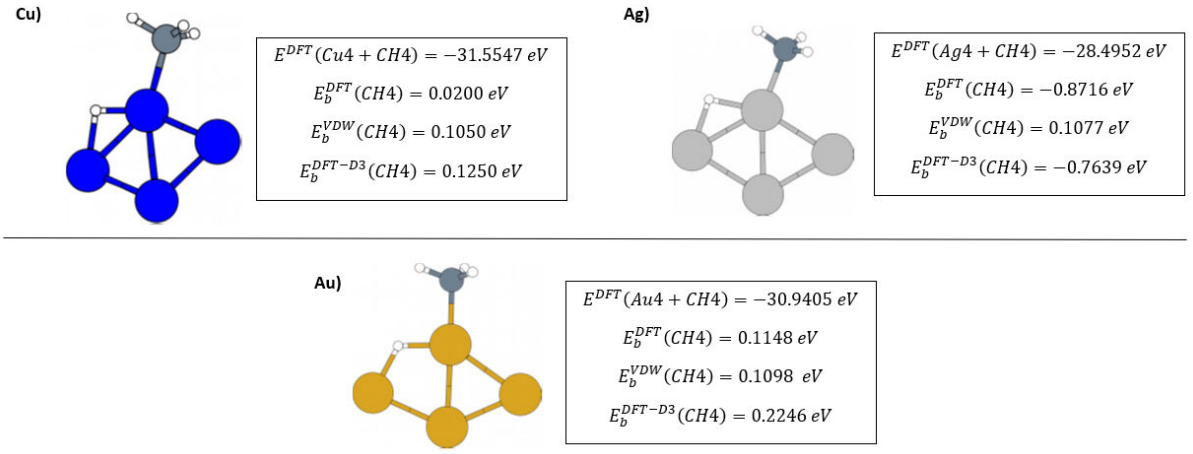

Figura 3.21: Geometrías relajadas junto con su energía DFT y energías de unión de CH4 disociado al Cu4 (izq.), Ag4 (dch.) y Au4 (abajo).

3.4. Adsorción de gases y átomos de metales de transición en el MOF-5

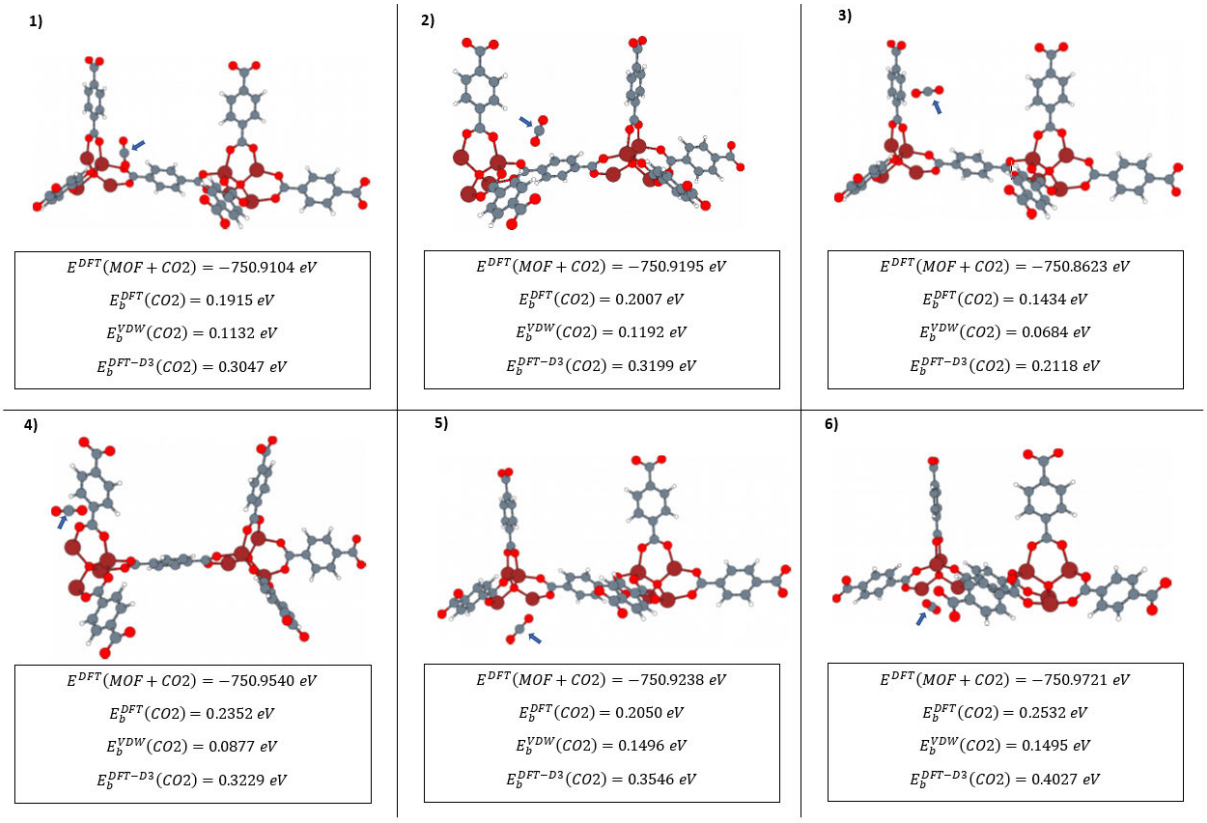

Figura 3.22: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 más CO2 junto con las energías DFT del sistema y energías de unión del CO2.

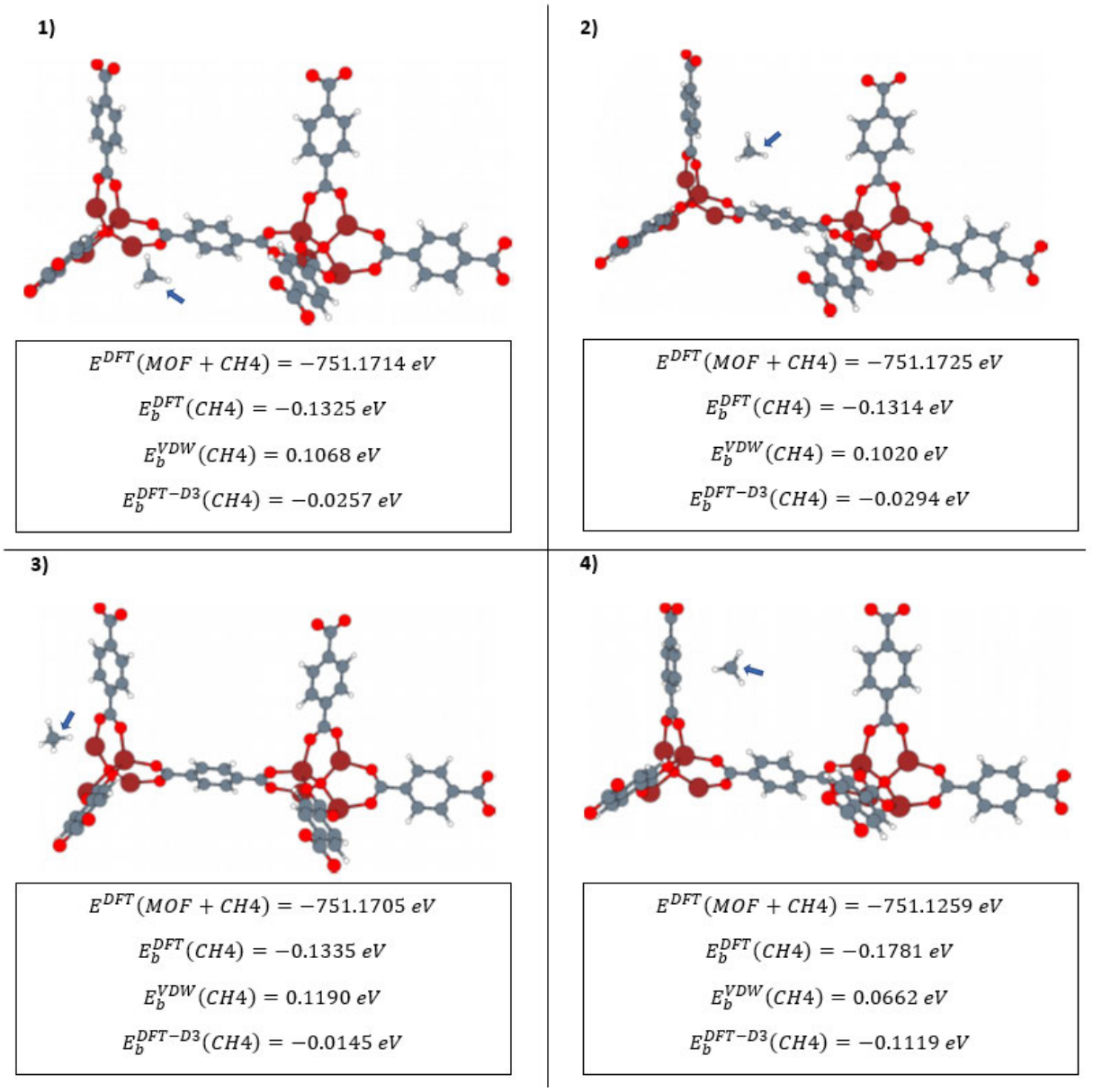

Figura 3.23: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 más CH4 junto con las energías DFT del sistema y energías de unión del CH4.

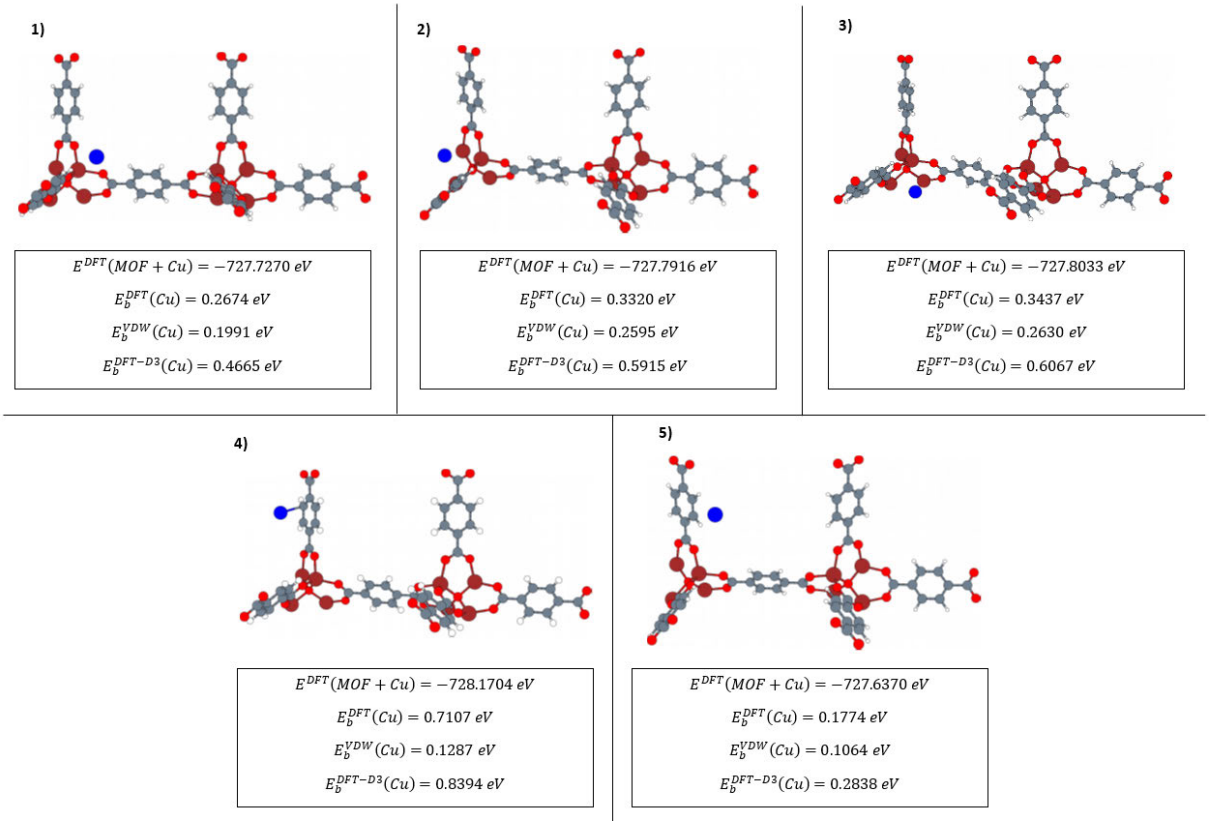

Figura 3.24: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 más Cu junto con las energías DFT del sistema y energías de unión del Cu.

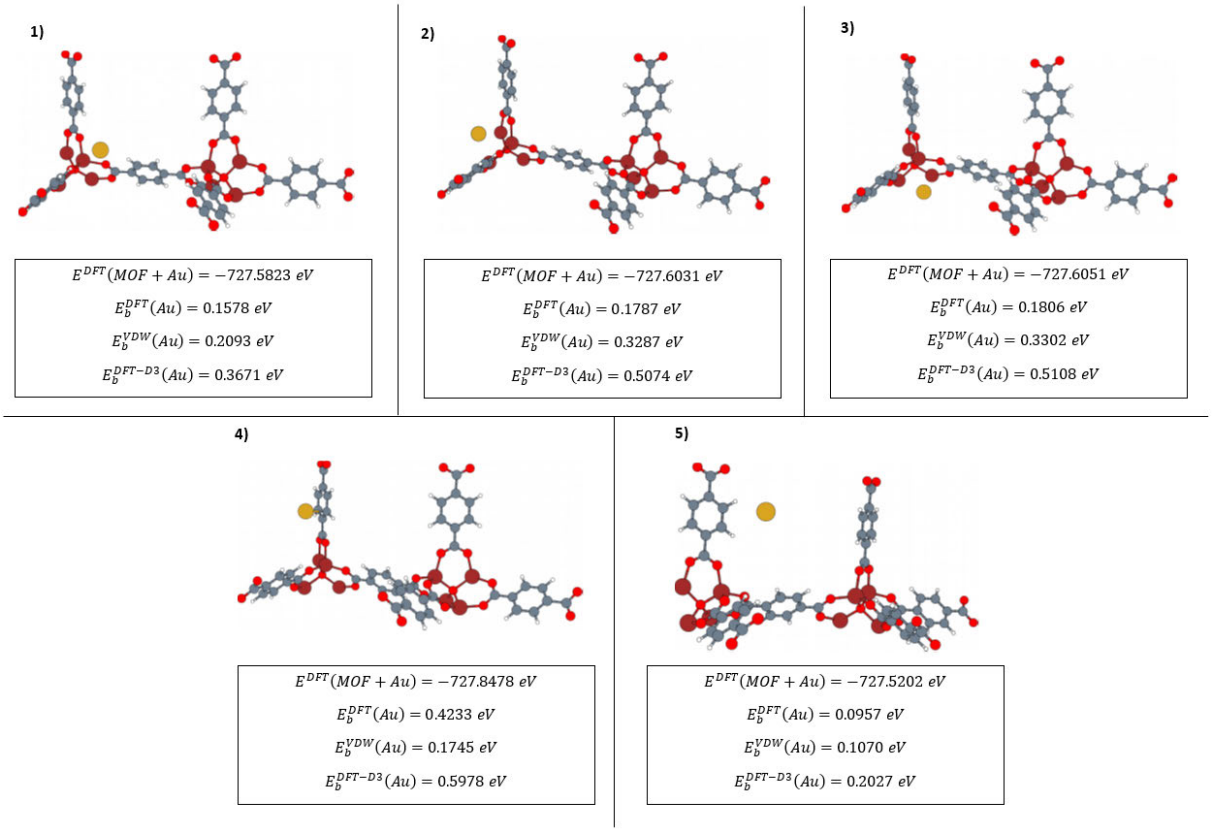

Figura 3.25: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 más Au junto con las energías DFT del sistema y energías de unión del Au.

3.5. Adsorción de gases en MOF-5 dopado con átomos de metales de transición

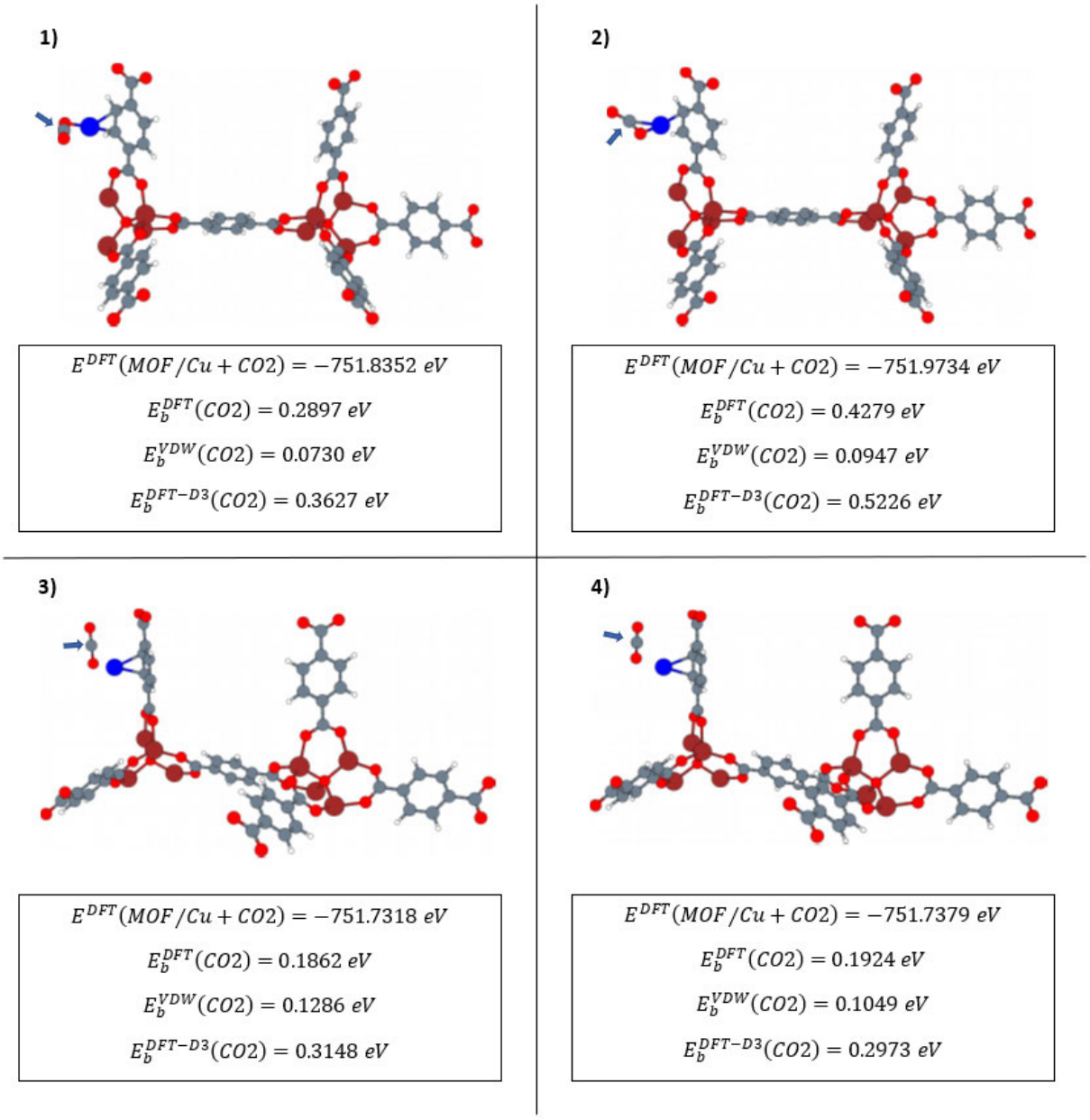

Figura 3.26: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 dopado con Cu al que añadimos CO2 junto con las energías DFT del sistema y energías de unión del CO2.

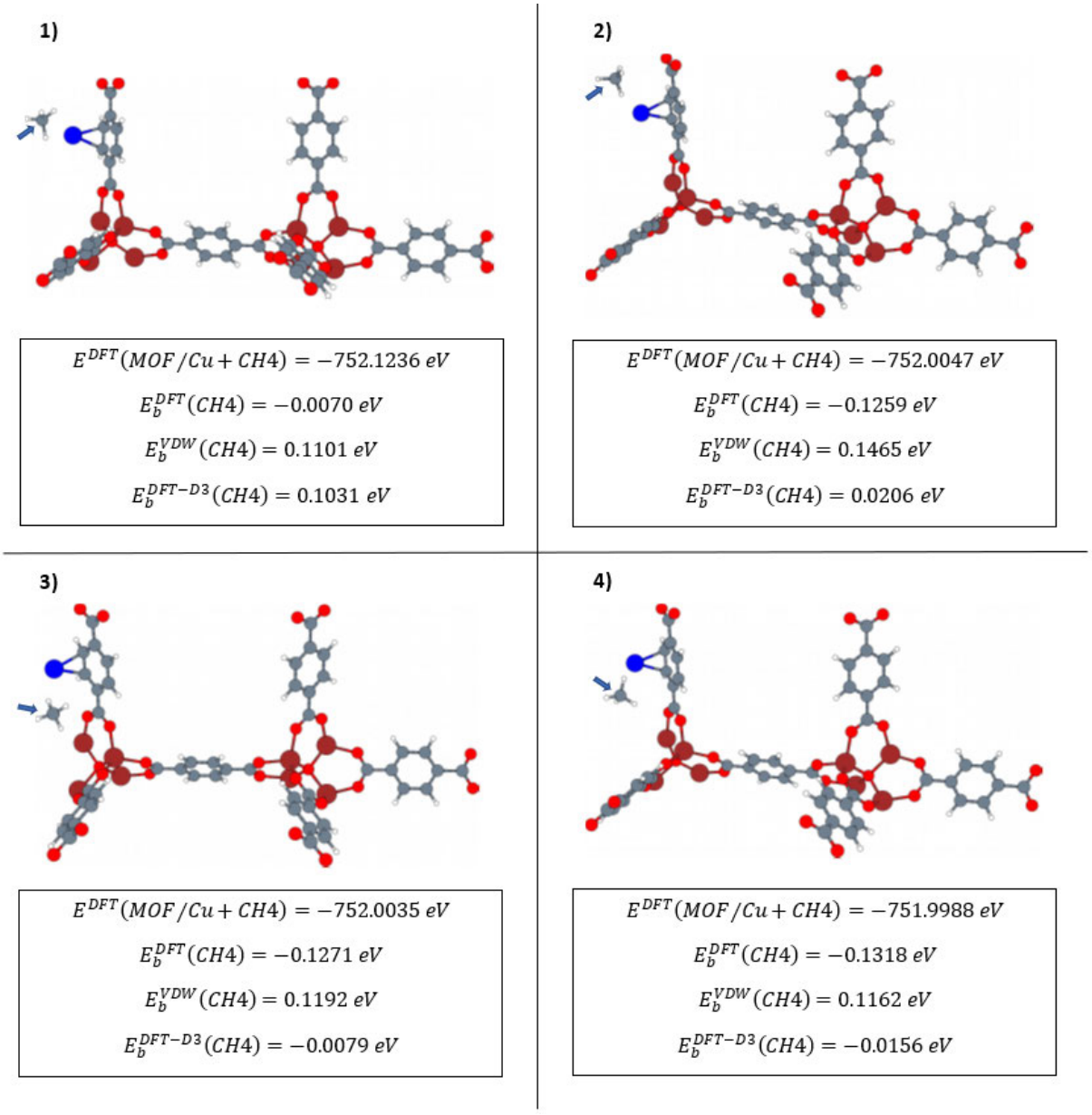

Figura 3.27: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 dopado con Cu al que añadimos CH4 junto con las energías DFT del sistema y energías de unión del CH4.

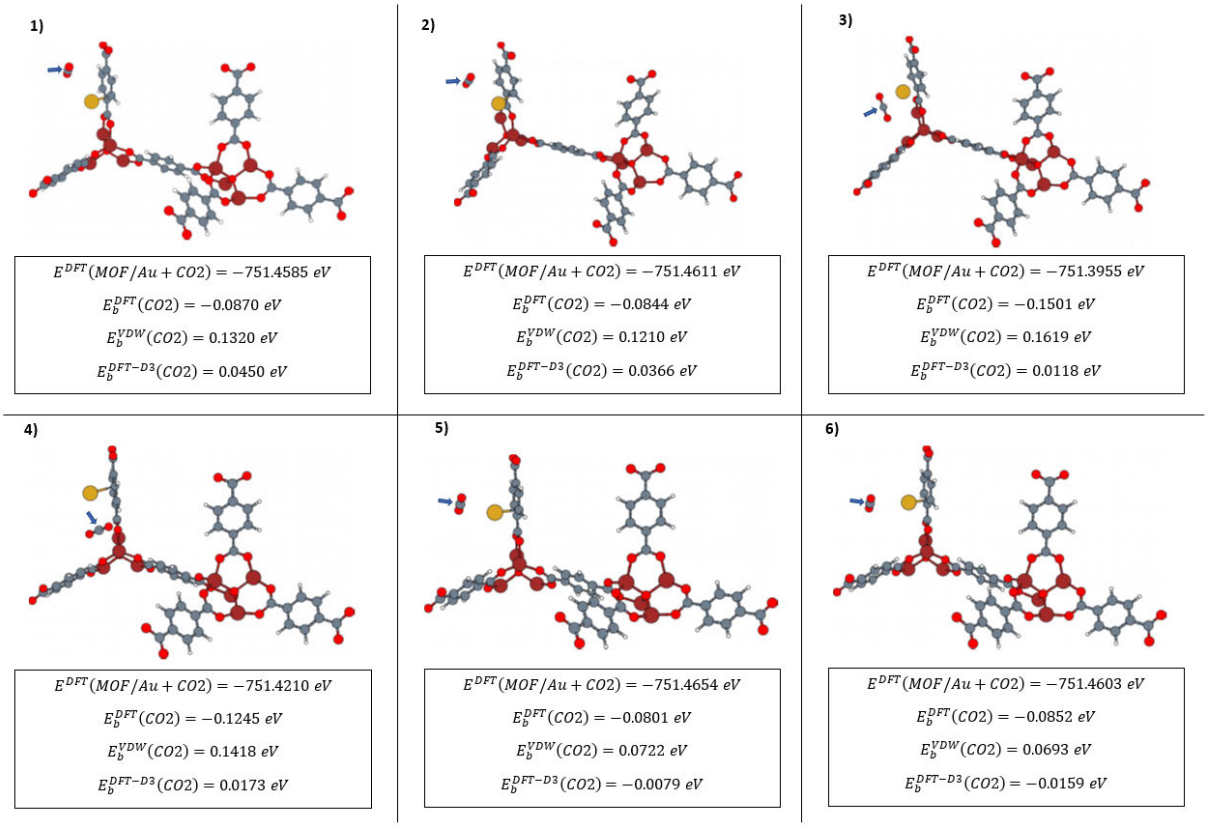

Figura 3.28: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 dopado con Au al que añadimos CO2 junto con las energías DFT del sistema y energías de unión del CO2.

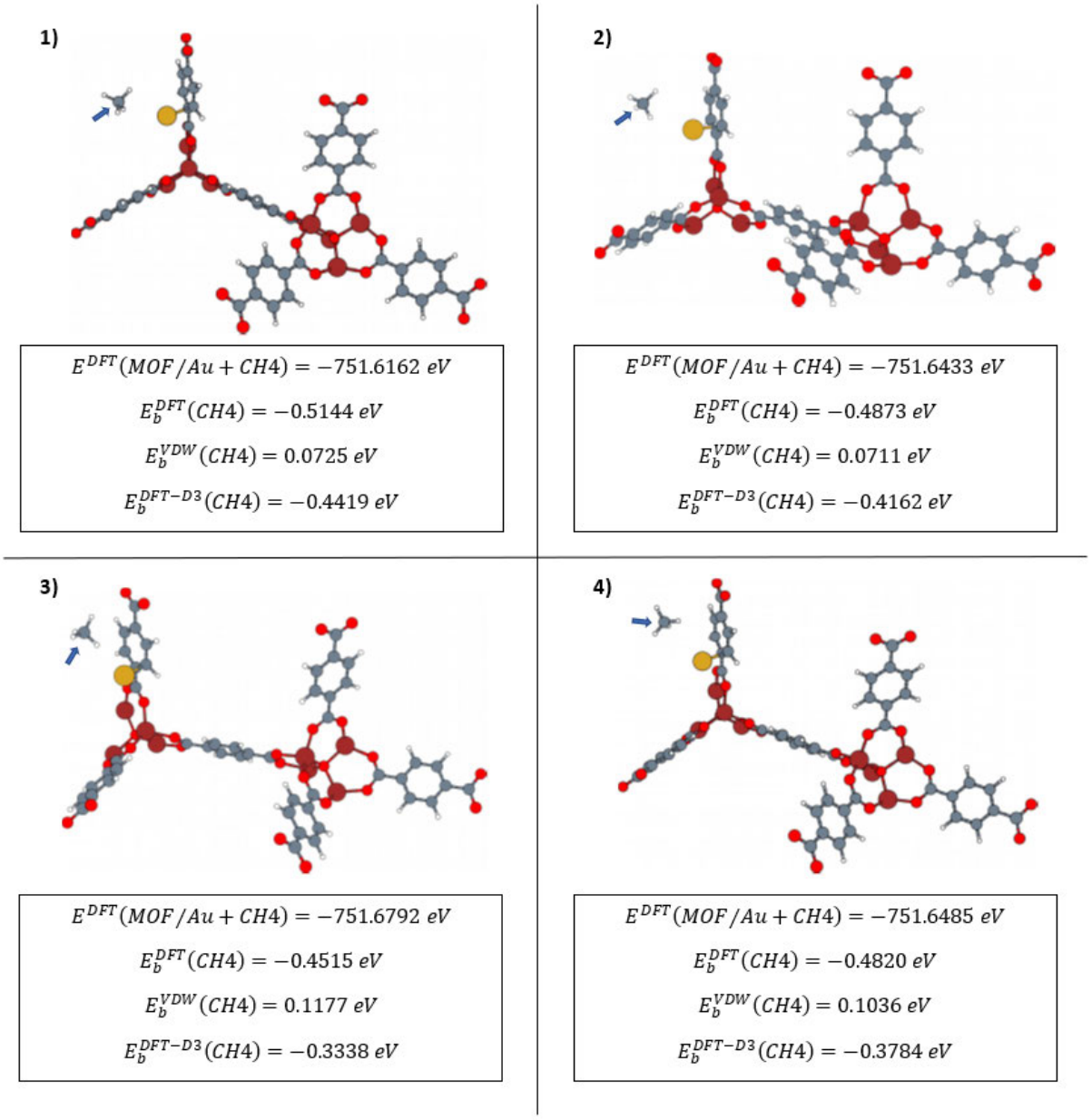

Figura 3.29: Geometrías tras la relajación estructural de diferentes configuraciones iniciales de MOF-5 dopado con Au al que añadimos CH4 junto con las energías DFT del sistema y energías de unión del CH4.

3.6. Adsorción de gases en MOF-5 interpenetrado

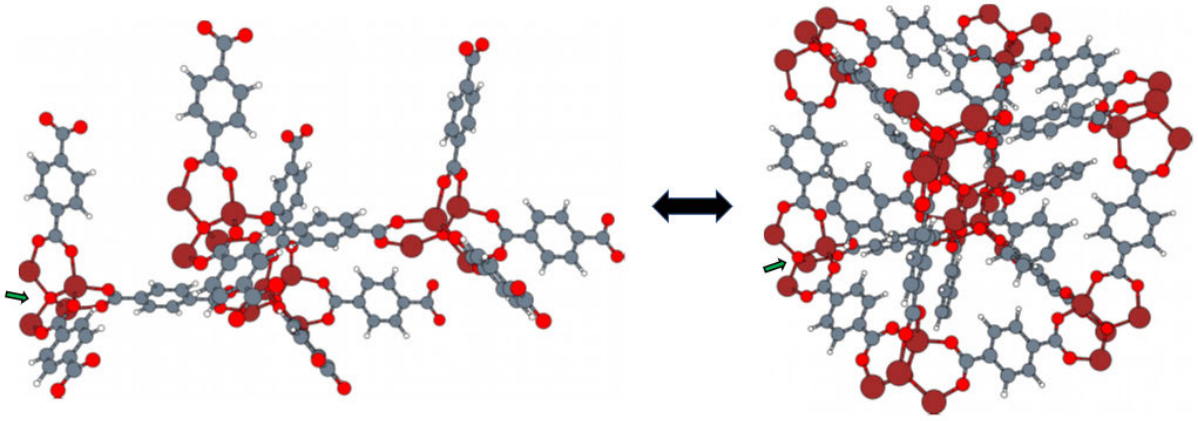

Figura 3.30: Celda primitiva del MOF-5 interpenetrado (izq.) y como pasamos a la celda manipulada (dch.) usando un programa propio. La flecha verde señala el origen de la celda primitiva de manera aproximada

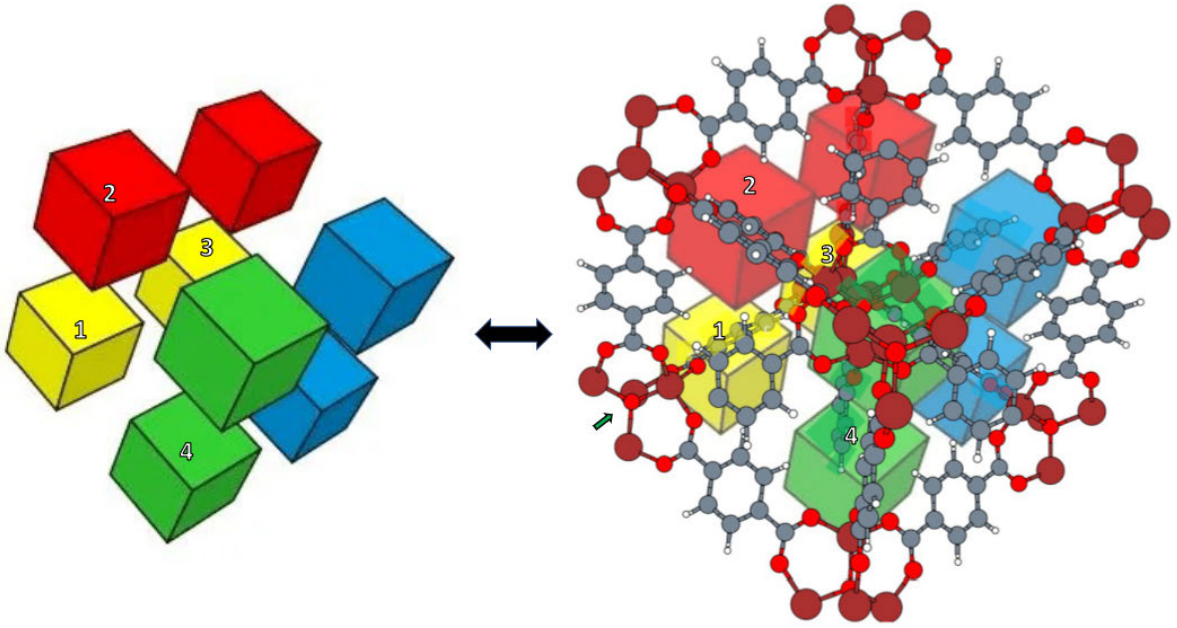

Figura 3.31: Poros que se forman dentro del MOF-5 interpenetrado.

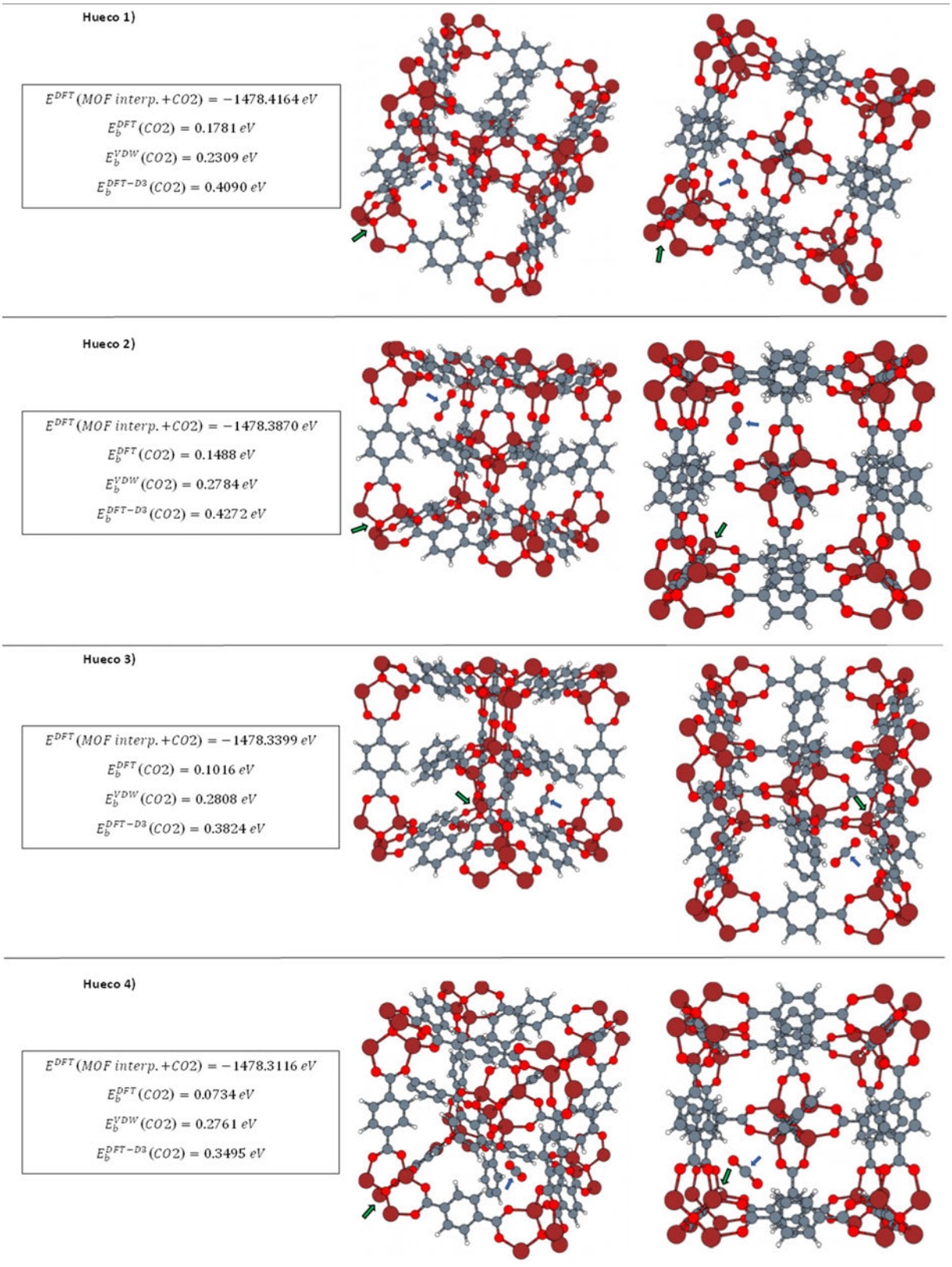

Figura 3.32: Dos perspectivas diferentes, tras la relajación estructural, de geometrías del MOF-5 interpenetrado con una molécula de CO2 en el huecos 1, 2, 3 y 4. Se adjuntan también la correspondiente energía DFT de las geometrías y las energías de unión del CO2.

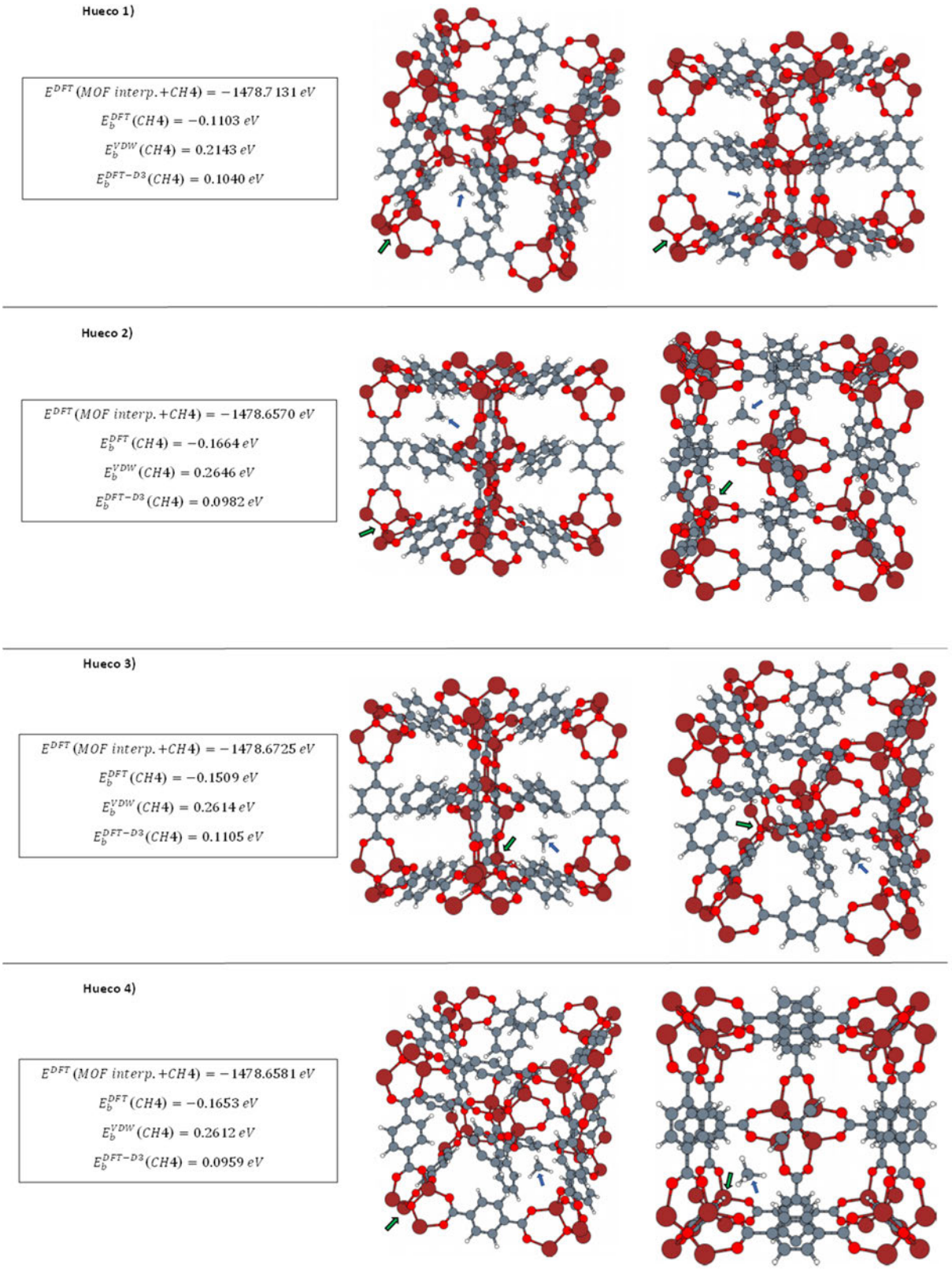

Figura 3.33: Dos perspectivas diferentes, tras la relajación estructural, de geometrías del MOF-5 interpenetrado con una molécula de CH4 en el huecos 1, 2, 3 y 4. Se adjuntan también la correspondiente energía DFT de las geometrías y las energías de unión del CH4.