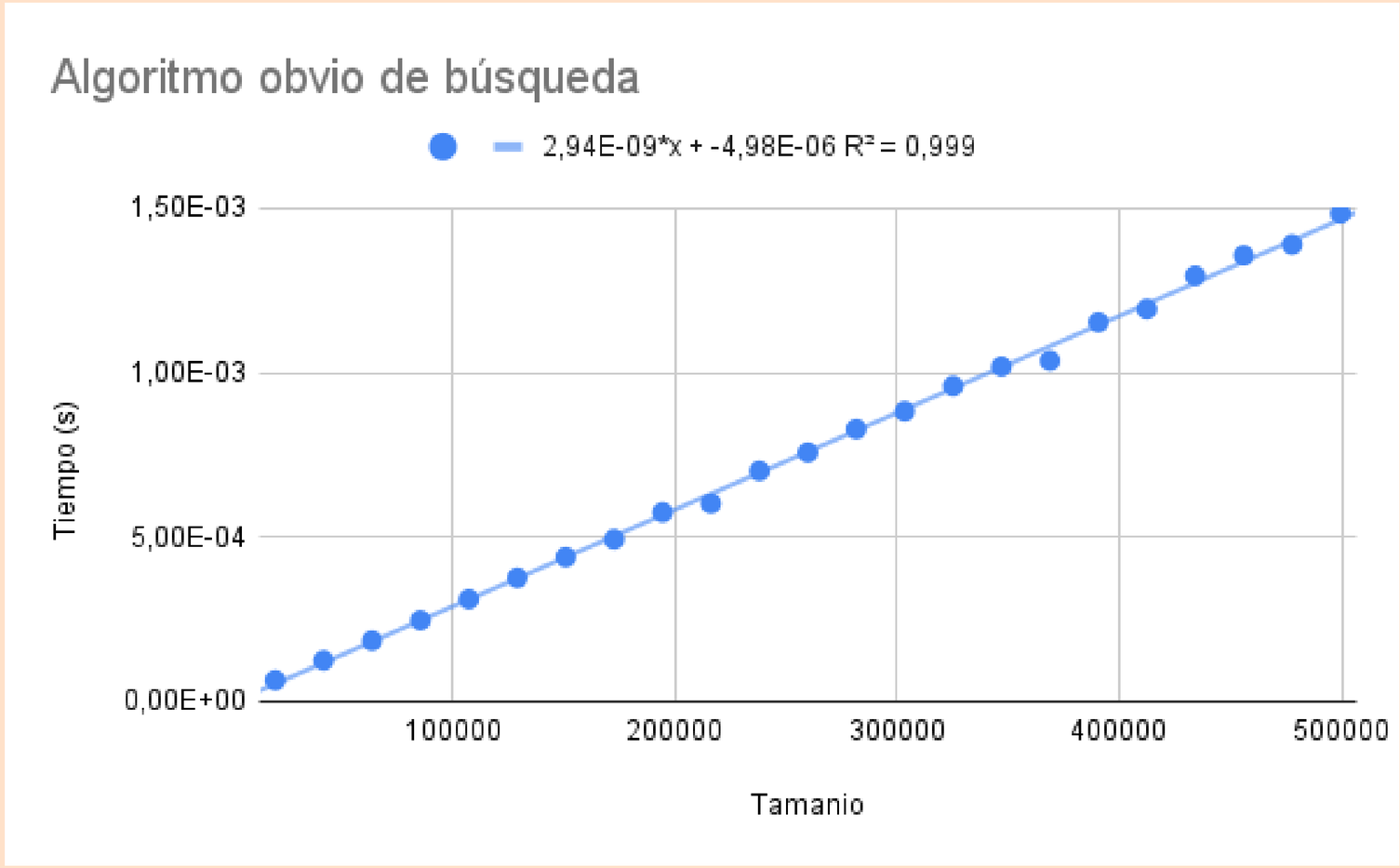
Algoritmos divide y vencerás

JOAQUÍN ARCILA PÉREZ
LAURA LÁZARO SORALUCE
CRISTÓBAL MERINO SÁEZ
ÁLVARO MOLINA ÁLVAREZ

ÍNDICE

INTRODUCCIÓN

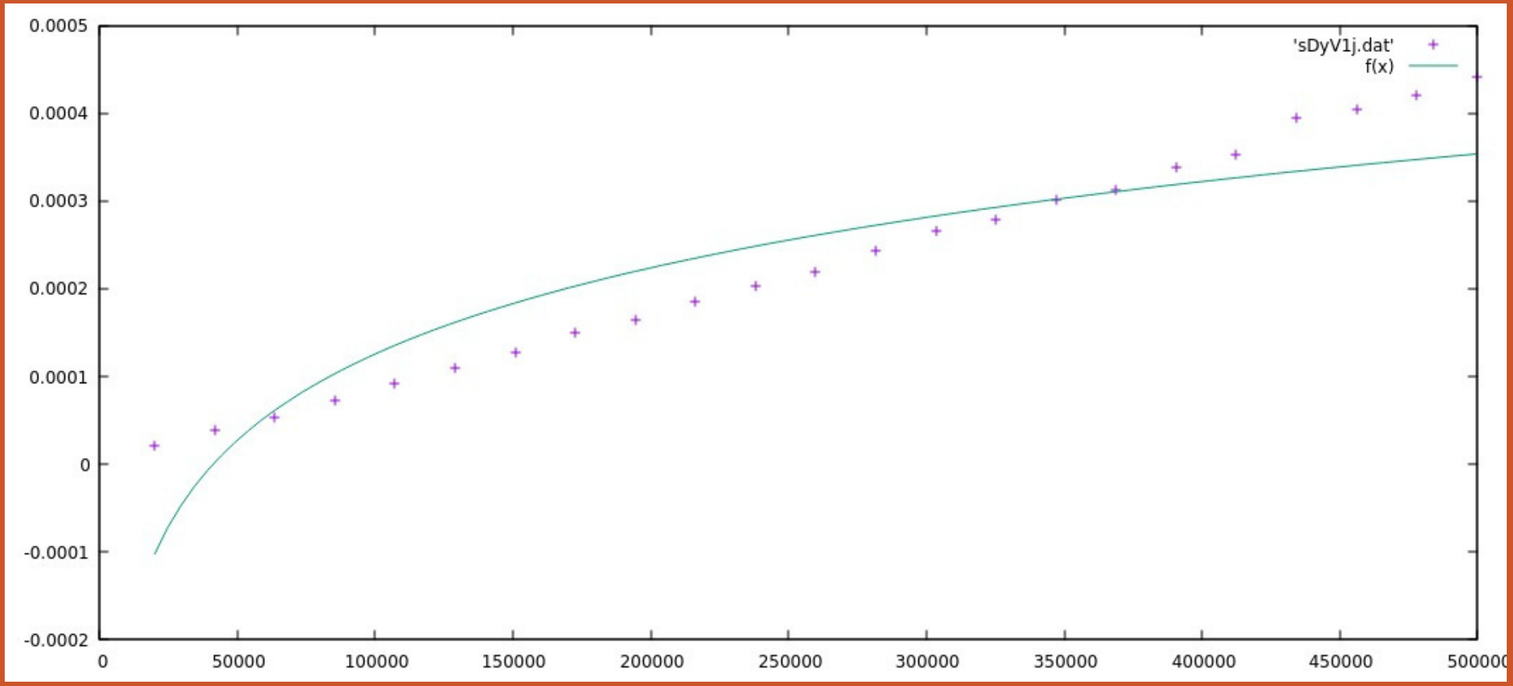
DESARROLLO
Algoritmo obvio de búsqueda




```
void obvio1 (vector<int> v, int nelementos) {
 bool sigue = true;

for (int i = 0; (i < nelementos) && (sigue); i++)
{
 if (v[i] == i) {
 sigue = false;
 }
 }
}
```

DESARROLLO
Algoritmo divide y vencerás de búsqueda




```
void DyV1 (vector<int> v, int ini, int fin) {

 bool encontrado =false;

 while(ini<=fin && !encontrado) {
 int m=(fin+ini)/2;
 if(v[m]==m) {
 encontrado=true;
 }
 else if (v[m]<m) {
 ini=m+1;
 }
 else
 fin=m-1;
 }
}
```

DESARROLLO

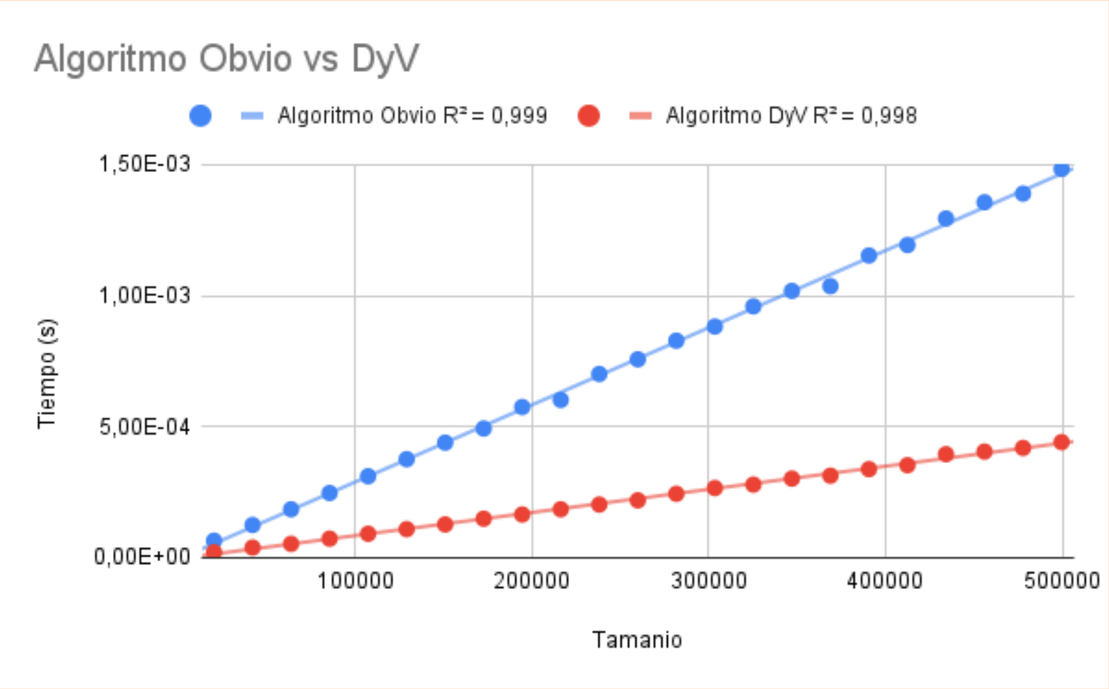
Algoritmo divide y vencerás con repetición

- Estructura:

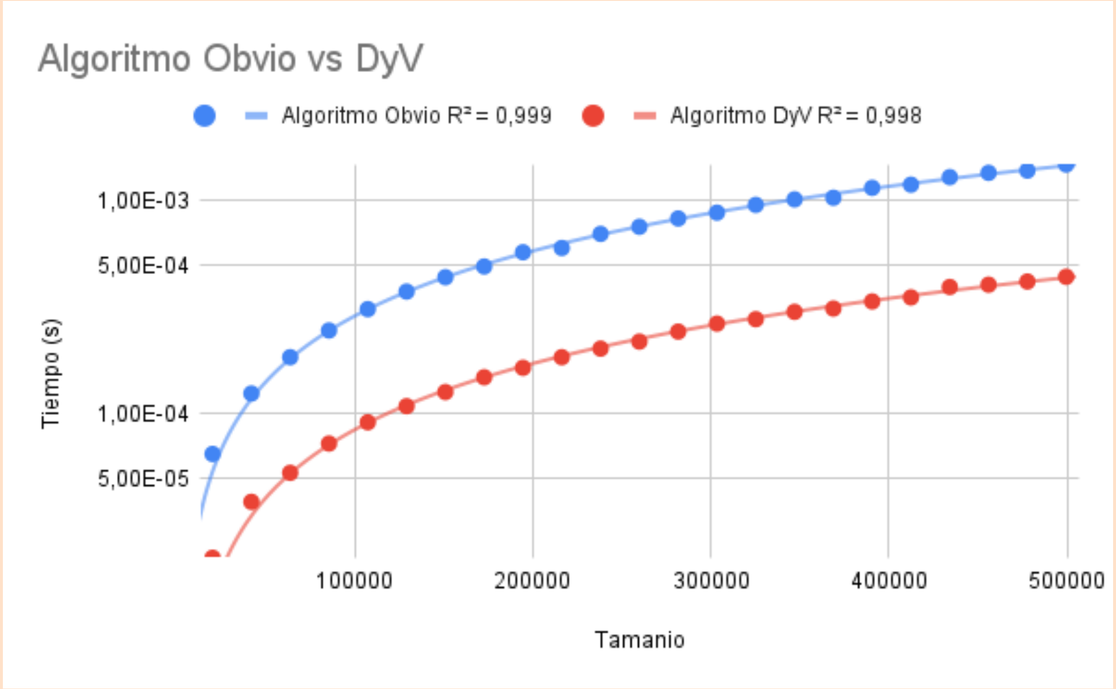
```
void DyV1 (vector<int> v, int ini, int fin) {
 bool encontrado =false;
 while(ini<=fin && !encontrado) {
 int m=(fin+ini)/2;
 if(v[m]==m) { encontrado=true; }
 else if (v[m]<m) {
 int i = m;
 while ((v[i-1]==v[i]) && !encontrado
 && i>=(ini+1)) {
 if (v[i-1] == i-1) { encontrado = true; }
 else { i--; }
 }
 ini=m+1;
 }
 else {
 int i = m;
 while ((v[i+1]==v[i]) && !encontrado
 && i<=(fin-1)) {
 if (v[i+1] == i+1) { encontrado = true; }
 else { i++; }
 }
 fin=m-1;
 } } }
```

- Eficiencia frente al algoritmo obvio

DESARROLLO
Comparación



Escala Normal



Escala Logarítmica

DESARROLLO

Algoritmo obvio para unir vectores

Estructura:

```
vector<int> funcion_obvia(vector<vector<int>> &m){
  while(m.size()>1){
    m[0] = mergevectors(m[0],m[1]);
    m.erase(m.begin()+1);
  }
  return m[0];
}
```

Eficacia de $O(k^2*n)$, k número de vectores y n número de elementos de los vectores

DESARROLLO

Algoritmo divide y vencerás de merge

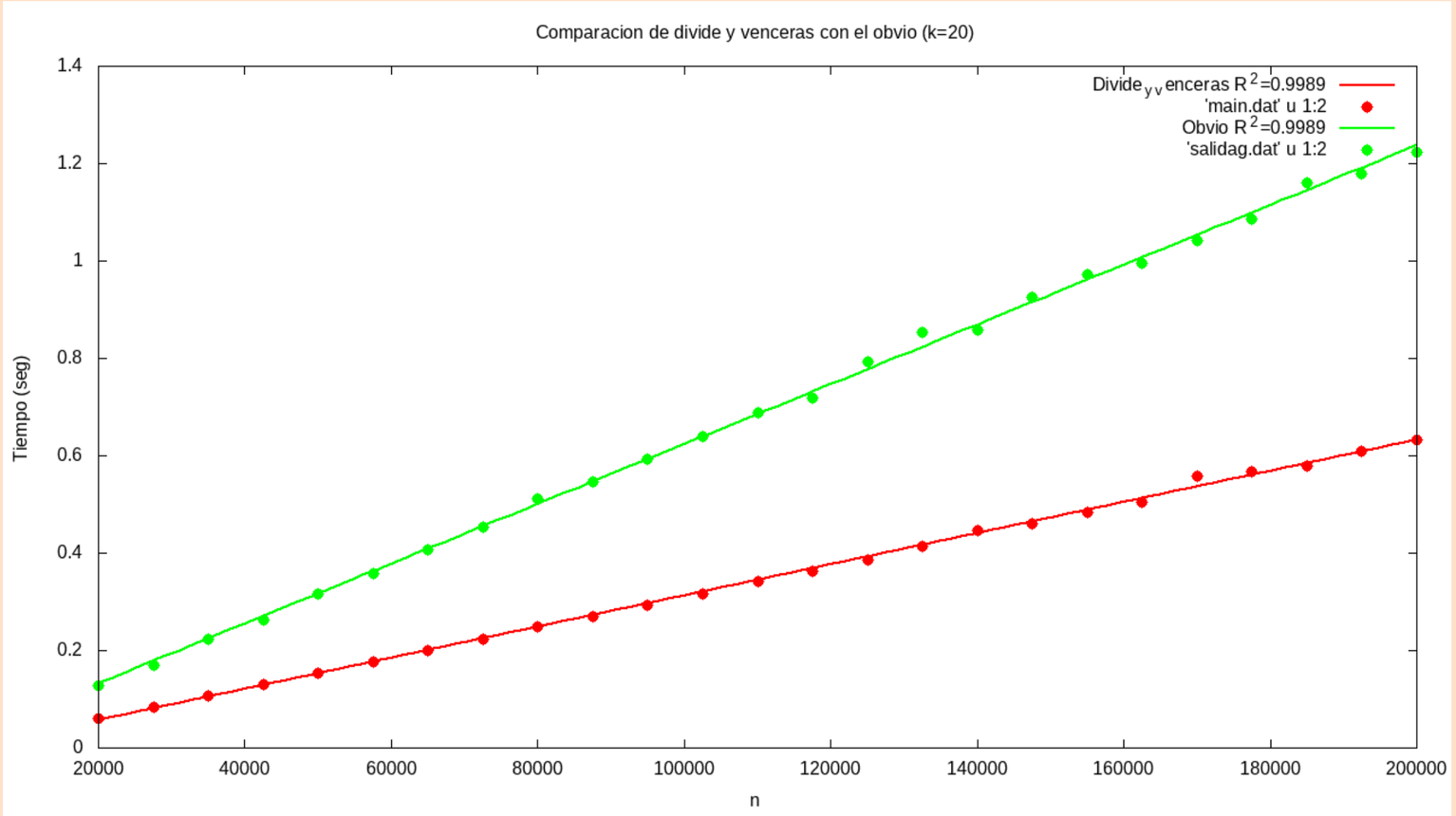
```
vector<int> funcion(vector<vector<int>> &m)
{
 vector<vector<int>> aux;
 vector<int> solucion;
 for (int i=0;i<m.size()-1;i+=2){
 vector<int> aux2;
 aux.push_back(mergevectors(m[i],m[i+1]));
 }
 if(m.size()%2 != 0){
 aux.push_back(m[m.size()-1]);
 }

 if(aux.size()>=2){
 solucion = funcion(aux);
 }
 else solucion=aux[0];
 return solucion;
}
```

Eficacia de O(k*log(k)*n), k número de vectores y n número de elementos de los vectores.

DESARROLLO

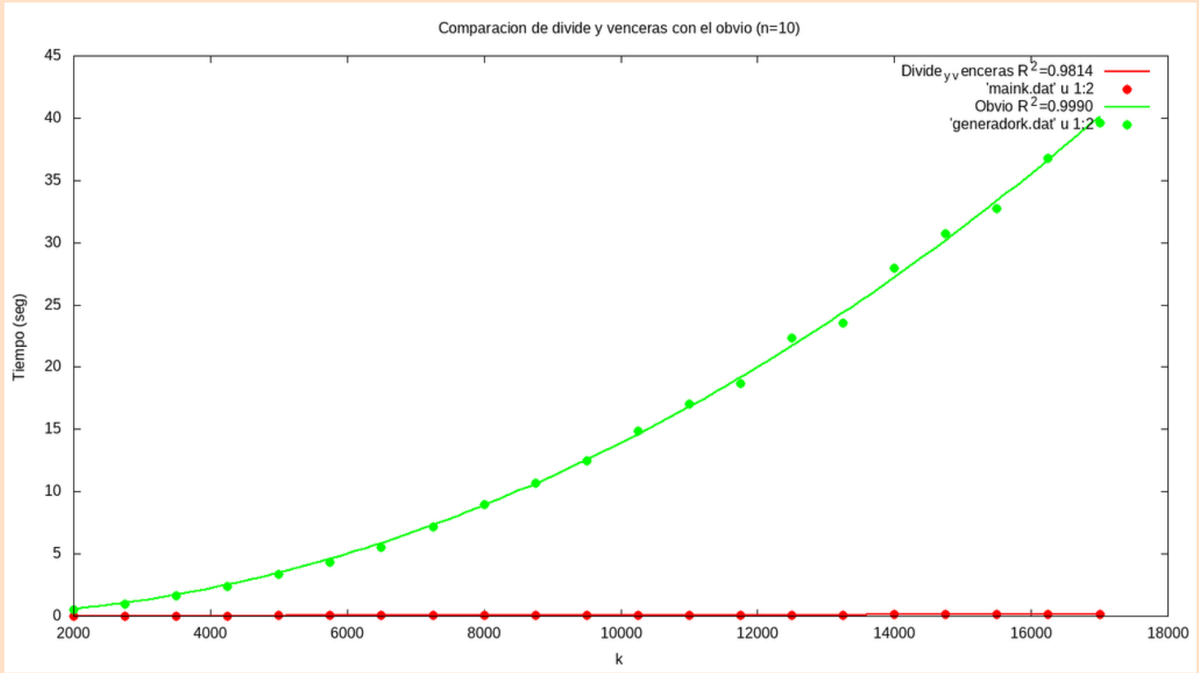
Comparación en función de elementos del vector



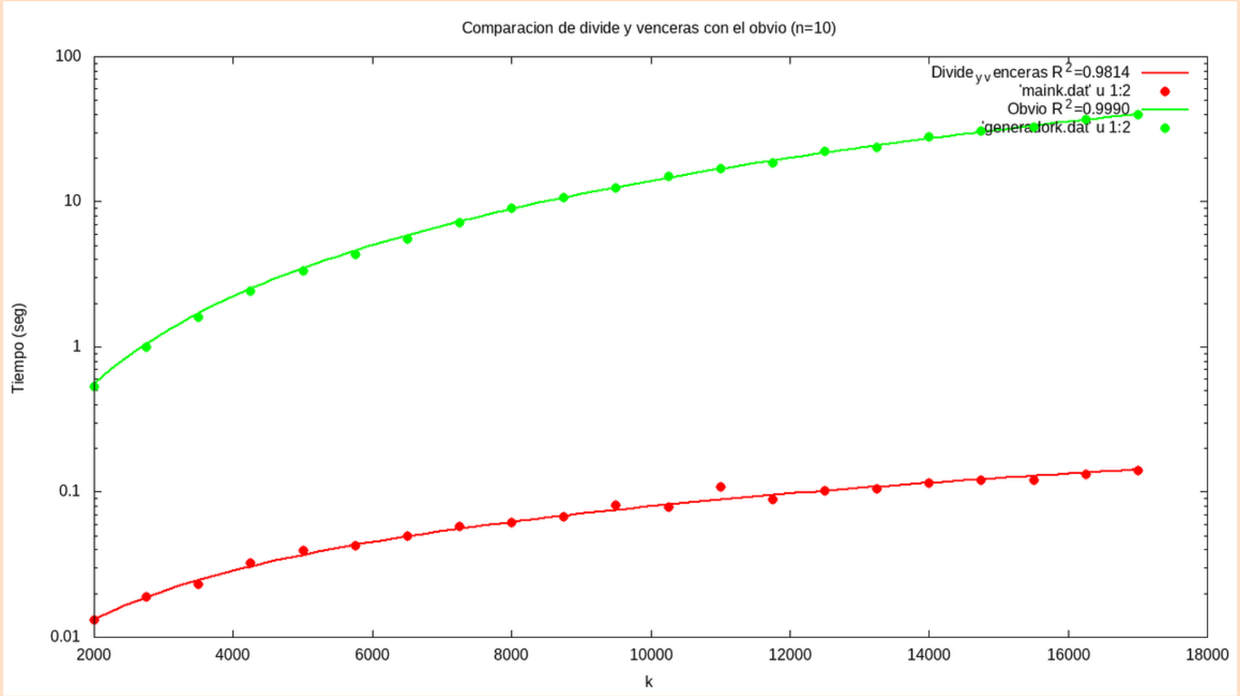
Escala Normal

DESARROLLO

Comparación en función del número de vectores



Escala Normal



Escala Logarítmica

CONCLUSIÓN

-El algoritmo divide y vencerás es más eficiente en ambos ejercicios

-En el ejercicio 1, la ejecución del caso sin repeticiones es más rápida

-En el ejercicio 2, la eficiencia teórica y la empírica coinciden

MUCHAS GRACIAS