Universidad de Granada

Algorítmica

Memoria de Prácticas

# Práctica I: Eficiencia

*Autora:*
Elena Merelo Molina

13 de Marzo

Universidad de Granada

# 1 Eficiencia empírica

En esta práctica veremos la eficiencia empírica de los algoritmos:

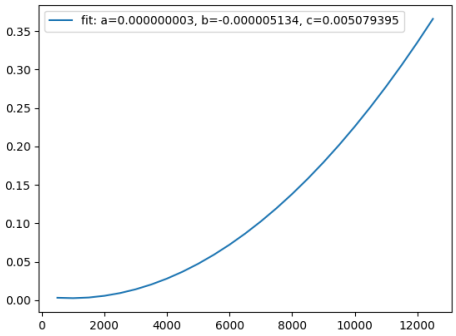
| Algoritmo | Orden de Eficiencia |
|---|---|
| Burbuja | O($n^2$) |
| Inserción | O($n^2$) |
| Selección | O($n^2$) |
| Mergesort | O($n \log(n)$) |
| Quicksort | O($n \log(n)$) |
| Heapsort | O($n \log(n)$) |
| Floyd | O($n^3$) |
| Hanoi | O($2^n$) |

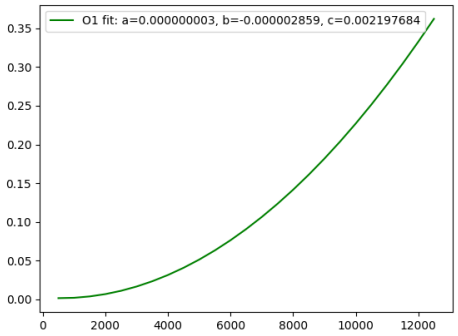
Con el objetivo de estudiar la eficiencia empírica de los algoritmos dados, en cada programa hemos medido el tiempo que ha tardado en ejecutarse, incluyendo la biblioteca ctime y usando clock_t, compilando cada programa por separado con g++ nombre_algoritmo.cpp -o nombre_algoritmo, esto mismo con la opción -O1 y -O3 para luego comparar resultados. Cuando ya se generaron los ejecutables, creados para veinticinco valores diferentes de $n$, programa por programa, de manera que los resultados se guarden en sus correspondientes archivos .csv, lo hemos representado ajustando por el método de los mínimos cuadráticos según el orden de eficiencia de cada uno:

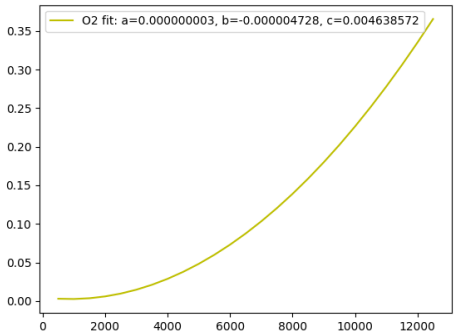
1. $O(n^2)$ ajustada a $f(x) = ax^2 + bx + c$
2. $O(n \log(n)$ ajustada a $f(x) = a \log(bn) + c$
3. $O(n^3)$ ajustada a $f(x) = ax^3 + bx^2 + cx + d$
4. $O(2^n)$ ajustada a $f(x) = a2^{bn} + c$

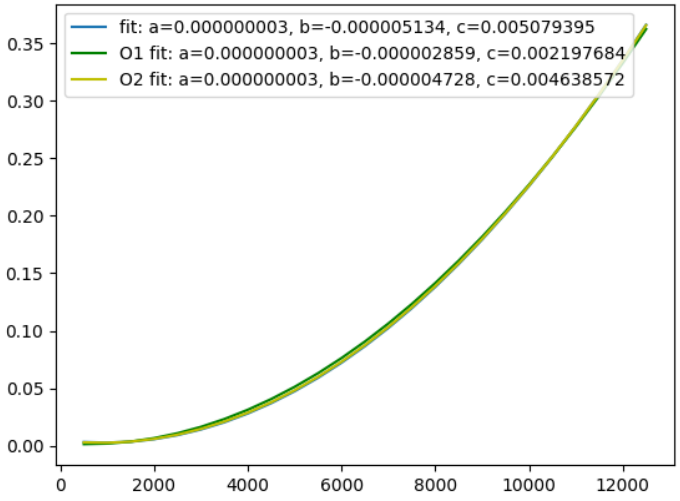
Vemos ahora detalladamente la representación gráfica de los datos obtenidos.

## 1.1 Algoritmo burbuja



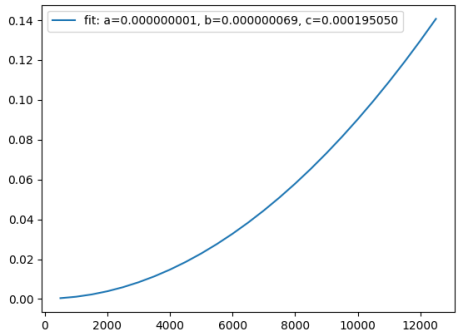



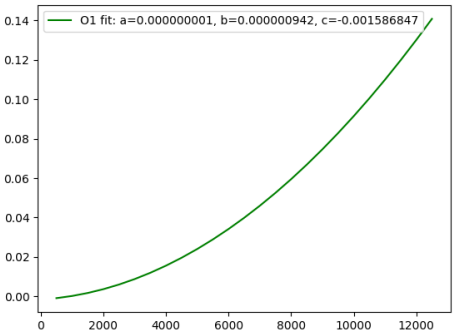



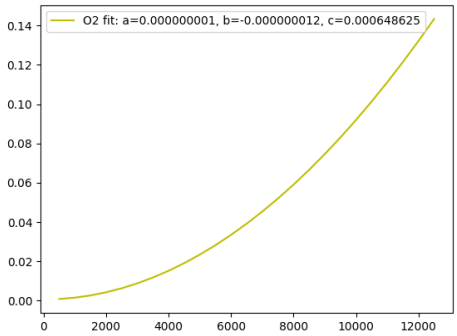



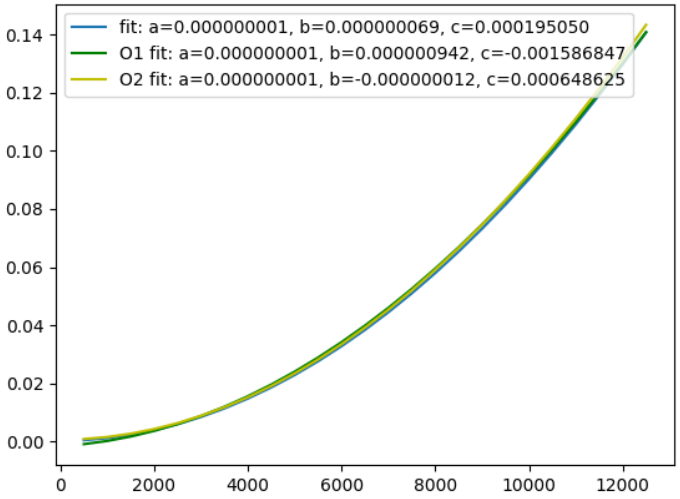
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 0.000848284 | 0.000504565 | 0.000444922 |
| 1000 | 0.00155837 | 0.00170106 | 0.00172926 |
| 1500 | 0.00375233 | 0.00372675 | 0.00348126 |
| 2000 | 0.00625636 | 0.00654731 | 0.00675332 |
| 2500 | 0.0102962 | 0.0111093 | 0.00993443 |
| 3000 | 0.0158037 | 0.0156202 | 0.0159983 |
| 3500 | 0.0211397 | 0.0220235 | 0.021973 |
| 4000 | 0.0292756 | 0.0298859 | 0.0299241 |
| 4500 | 0.0377767 | 0.0425379 | 0.0370894 |
| 5000 | 0.0472146 | 0.0575588 | 0.0466534 |
| 5500 | 0.0585254 | 0.0650413 | 0.0596688 |
| 6000 | 0.0734717 | 0.0795291 | 0.0789712 |
| 6500 | 0.0875871 | 0.0864143 | 0.0938704 |
| 7000 | 0.101356 | 0.106416 | 0.10351 |
| 7500 | 0.119825 | 0.117833 | 0.118222 |
| 8000 | 0.13685 | 0.142887 | 0.13587 |
| 8500 | 0.158619 | 0.156353 | 0.154324 |
| 9000 | 0.177794 | 0.180205 | 0.179462 |
| 9500 | 0.199033 | 0.208484 | 0.197654 |
| 10000 | 0.225543 | 0.224484 | 0.224071 |
| 10500 | 0.249815 | 0.252788 | 0.247328 |
| 11000 | 0.279319 | 0.277287 | 0.284709 |
| 11500 | 0.30519 | 0.305155 | 0.305884 |
| 12000 | 0.335826 | 0.334685 | 0.339147 |
| 12500 | 0.368191 | 0.361129 | 0.365015 |

## 1.2 Algoritmo insercion



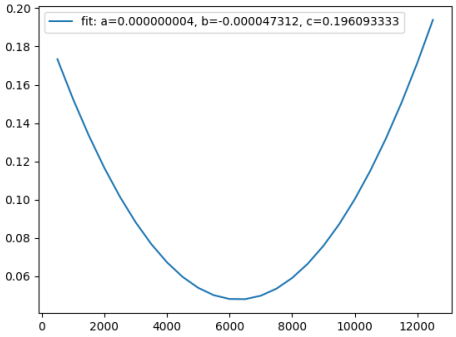



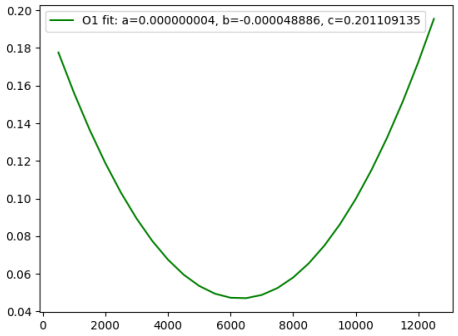



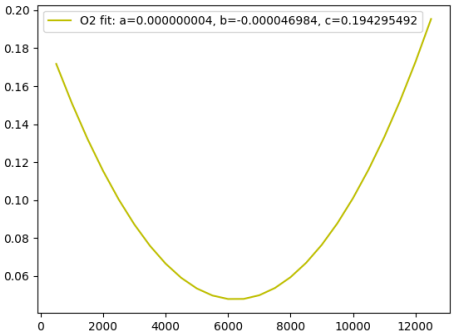



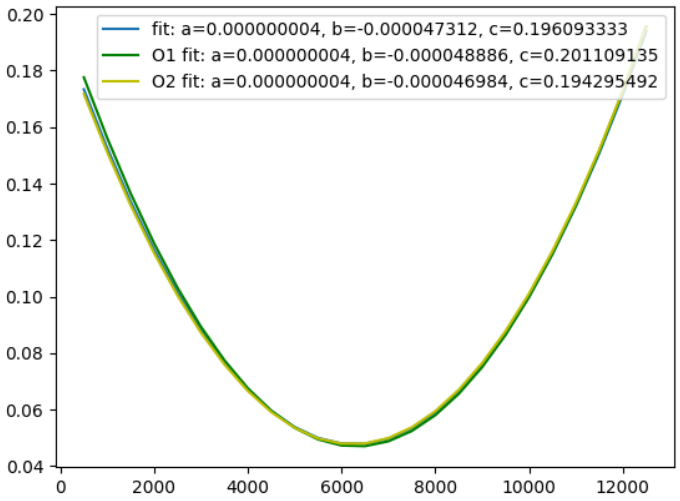
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 0.000280816 | 0.000233787 | 0.000223985 |
| 1000 | 0.000985635 | 0.000910265 | 0.00100876 |
| 1500 | 0.00213929 | 0.00200899 | 0.00204243 |
| 2000 | 0.00369846 | 0.00377572 | 0.00449 |
| 2500 | 0.00592036 | 0.00560895 | 0.00585753 |
| 3000 | 0.00822953 | 0.00840262 | 0.00825681 |
| 3500 | 0.0120226 | 0.0115001 | 0.0129874 |
| 4000 | 0.0148672 | 0.0151695 | 0.0168938 |
| 4500 | 0.0193506 | 0.0185337 | 0.0193503 |
| 5000 | 0.0227202 | 0.0235869 | 0.0230816 |
| 5500 | 0.0272918 | 0.0275996 | 0.0280322 |
| 6000 | 0.0333669 | 0.0314578 | 0.0388916 |
| 6500 | 0.0395249 | 0.0369619 | 0.0393369 |
| 7000 | 0.0438121 | 0.0443715 | 0.0442169 |
| 7500 | 0.0510923 | 0.0534808 | 0.0519345 |
| 8000 | 0.0577335 | 0.0668524 | 0.0582985 |
| 8500 | 0.0664888 | 0.0740554 | 0.0659169 |
| 9000 | 0.0721457 | 0.0740537 | 0.0733521 |
| 9500 | 0.0804506 | 0.0808475 | 0.081258 |
| 10000 | 0.0889614 | 0.089042 | 0.0897421 |
| 10500 | 0.098722 | 0.0999622 | 0.0988874 |
| 11000 | 0.10898 | 0.111618 | 0.108517 |
| 11500 | 0.120852 | 0.118909 | 0.125629 |
| 12000 | 0.129697 | 0.128718 | 0.137064 |
| 12500 | 0.141071 | 0.140754 | 0.141494 |

## 1.3 Algoritmo seleccion



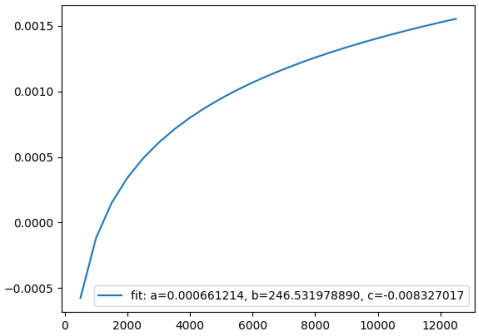



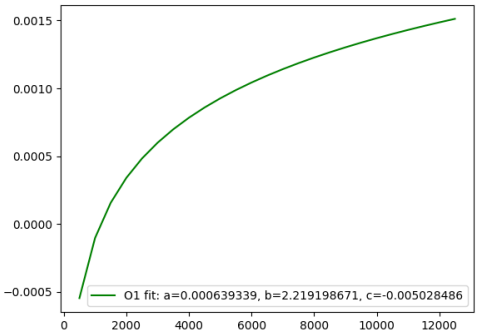



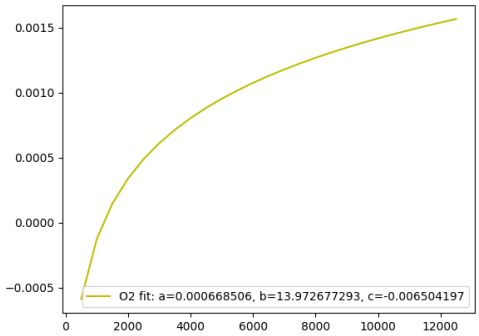



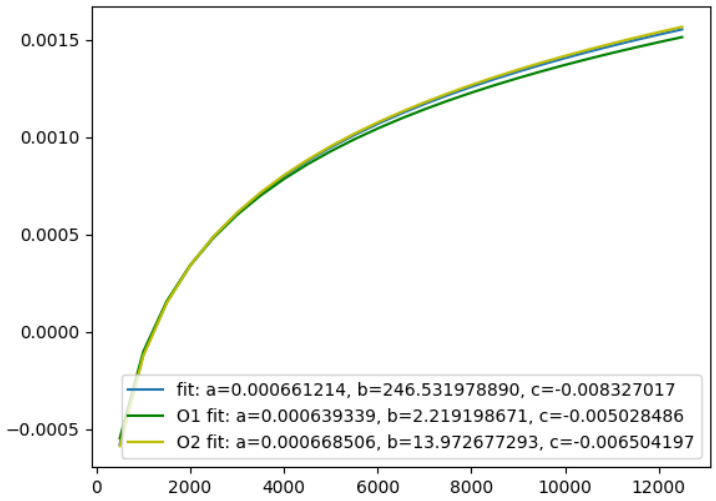
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 0.0314449 | 0.0314842 | 0.0317225 |
| 1000 | 0.119021 | 0.126228 | 0.11633 |
| 1500 | 0.260984 | 0.267642 | 0.258436 |
| 2000 | 0.457069 | 0.461125 | 0.454904 |
| 2500 | 0.00744243 | 0.00693814 | 0.0068266 |
| 3000 | 0.00998253 | 0.0103105 | 0.0104942 |
| 3500 | 0.0138713 | 0.0136245 | 0.0144952 |
| 4000 | 0.0186035 | 0.0182824 | 0.0187636 |
| 4500 | 0.0234744 | 0.0224572 | 0.0237714 |
| 5000 | 0.0277004 | 0.0292558 | 0.0300571 |
| 5500 | 0.0345149 | 0.0354603 | 0.035354 |
| 6000 | 0.0478933 | 0.0406127 | 0.0407739 |
| 6500 | 0.0520387 | 0.0483603 | 0.0474503 |
| 7000 | 0.0560279 | 0.0554717 | 0.0550182 |
| 7500 | 0.0618687 | 0.067223 | 0.0635355 |
| 8000 | 0.0725331 | 0.0711132 | 0.0718062 |
| 8500 | 0.0815797 | 0.0811896 | 0.0813514 |
| 9000 | 0.0909342 | 0.0917913 | 0.0928178 |
| 9500 | 0.103387 | 0.100633 | 0.100902 |
| 10000 | 0.110117 | 0.11207 | 0.127083 |
| 10500 | 0.12314 | 0.124775 | 0.125395 |
| 11000 | 0.134518 | 0.135545 | 0.135101 |
| 11500 | 0.151427 | 0.146785 | 0.146842 |
| 12000 | 0.158935 | 0.163465 | 0.159707 |
| 12500 | 0.174672 | 0.174286 | 0.174868 |

## 1.4 Algoritmo mergesort



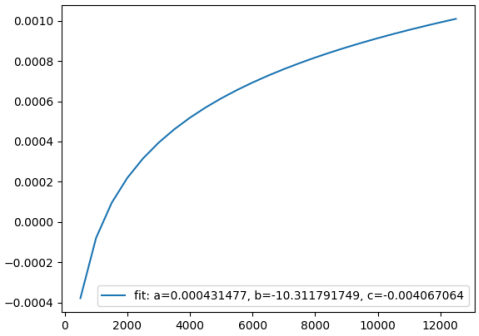



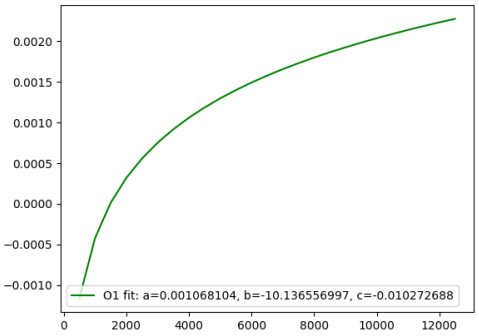



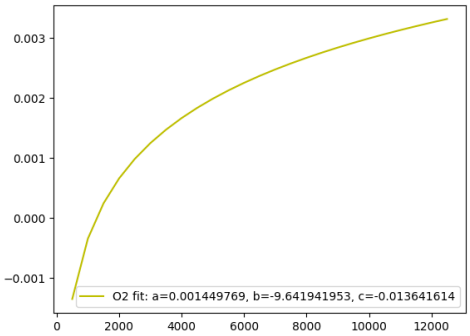



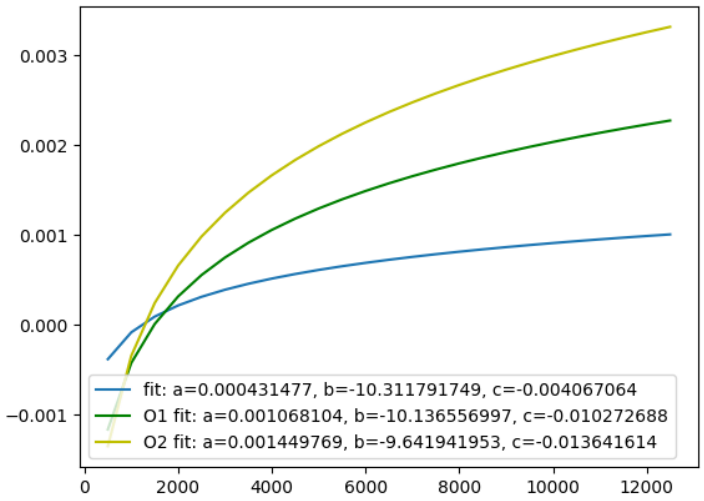
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 4.6913e-05 | 4.2538e-05 | 4.3779e-05 |
| 1000 | 0.000104759 | 0.000101012 | 0.000102685 |
| 1500 | 0.000197928 | 0.000202091 | 0.000197977 |
| 2000 | 0.000230556 | 0.000230267 | 0.000229837 |
| 2500 | 0.000320995 | 0.000321507 | 0.000324846 |
| 3000 | 0.000441303 | 0.000454994 | 0.000459037 |
| 3500 | 0.000411042 | 0.000422831 | 0.000415593 |
| 4000 | 0.000511027 | 0.000494096 | 0.000499695 |
| 4500 | 0.000612995 | 0.000592159 | 0.000609127 |
| 5000 | 0.00069516 | 0.000701389 | 0.000693268 |
| 5500 | 0.000804711 | 0.000800892 | 0.000801765 |
| 6000 | 0.000939736 | 0.0009096 | 0.000922795 |
| 6500 | 0.000810641 | 0.000809706 | 0.000812514 |
| 7000 | 0.000890017 | 0.000891093 | 0.000896917 |
| 7500 | 0.000985485 | 0.000987166 | 0.000983415 |
| 8000 | 0.00110324 | 0.00107382 | 0.00107462 |
| 8500 | 0.00116999 | 0.00116523 | 0.00117428 |
| 9000 | 0.00127648 | 0.00127043 | 0.00127371 |
| 9500 | 0.00138226 | 0.00137334 | 0.00139195 |
| 10000 | 0.00150468 | 0.00148788 | 0.00148267 |
| 10500 | 0.001526 | 0.001525 | 0.001523 |
| 11000 | 0.001634 | 0.001625 | 0.002057 |
| 11500 | 0.001889 | 0.001821 | 0.00186 |
| 12000 | 0.002311 | 0.001866 | 0.002124 |
| 12500 | 0.002148 | 0.002275 | 0.002152 |

## 1.5 Algoritmo quicksort



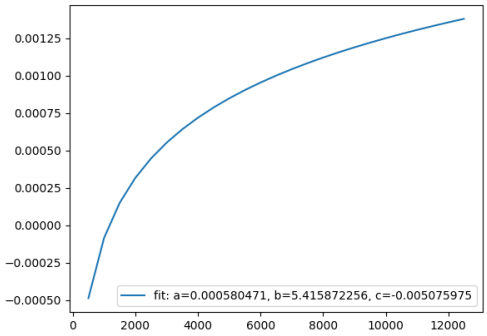



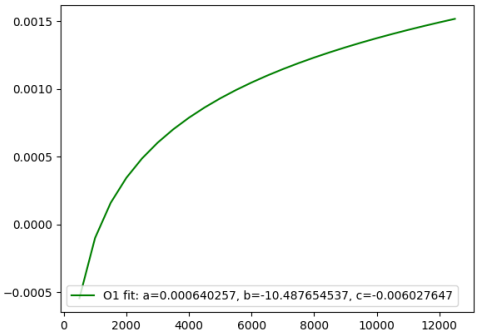



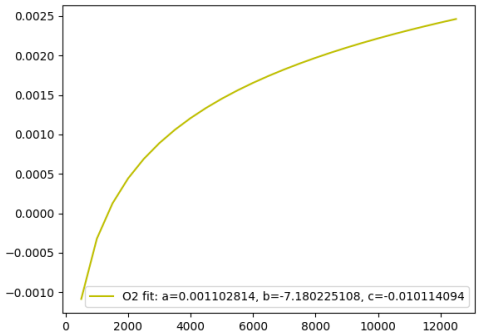



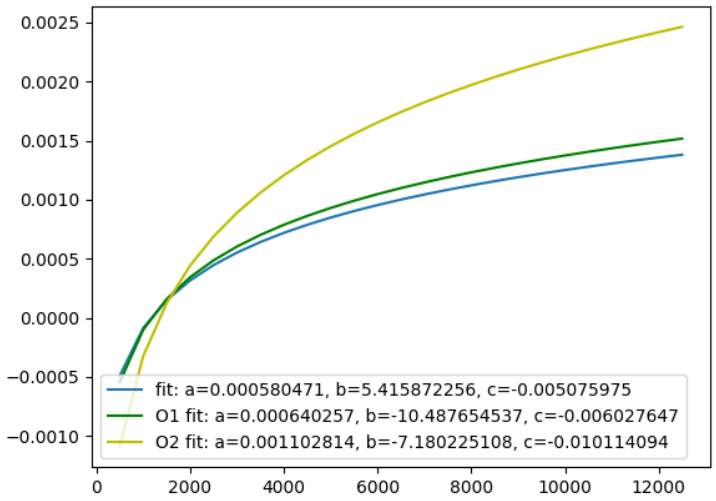
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 3.2332e-05 | 3.7642e-05 | 0.000103603 |
| 1000 | 7.036e-05 | 8.0497e-05 | 0.000222952 |
| 1500 | 0.000113397 | 0.000131637 | 0.000361161 |
| 2000 | 0.000154328 | 0.000176116 | 0.000484841 |
| 2500 | 0.000200578 | 0.00023071 | 0.000631374 |
| 3000 | 0.000240971 | 0.000276572 | 0.000763986 |
| 3500 | 0.000284638 | 0.000360803 | 0.000897255 |
| 4000 | 0.000333301 | 0.000420274 | 0.00107918 |
| 4500 | 0.000373817 | 0.000525212 | 0.001179 |
| 5000 | 0.000419683 | 0.000665588 | 0.00133037 |
| 5500 | 0.000463525 | 0.000837868 | 0.00146456 |
| 6000 | 0.000521563 | 0.000963575 | 0.00166409 |
| 6500 | 0.000557223 | 0.00101092 | 0.0017721 |
| 7000 | 0.000617823 | 0.00132214 | 0.00197305 |
| 7500 | 0.000920243 | 0.00140165 | 0.00211571 |
| 8000 | 0.000714884 | 0.00180813 | 0.00226463 |
| 8500 | 0.000766829 | 0.00193638 | 0.00243405 |
| 9000 | 0.000817047 | 0.00206904 | 0.00260327 |
| 9500 | 0.000909397 | 0.00218516 | 0.00275356 |
| 10000 | 0.000957013 | 0.00232035 | 0.00288264 |
| 10500 | 0.00108144 | 0.00242214 | 0.00302737 |
| 11000 | 0.00115832 | 0.00257706 | 0.00429314 |
| 11500 | 0.00122584 | 0.00265797 | 0.00448855 |
| 12000 | 0.00127122 | 0.00277936 | 0.0046674 |
| 12500 | 0.00135047 | 0.00373325 | 0.00497125 |

## 1.6 Algoritmo heapsort



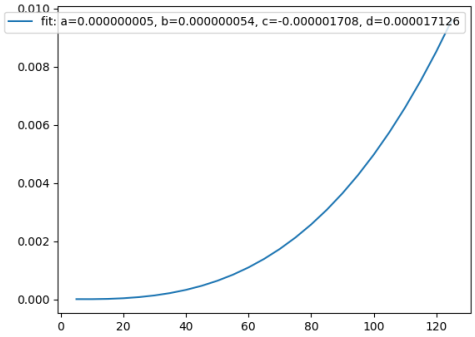



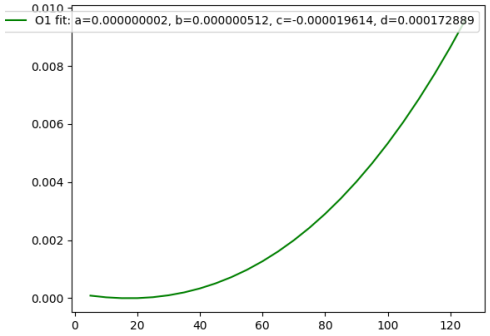



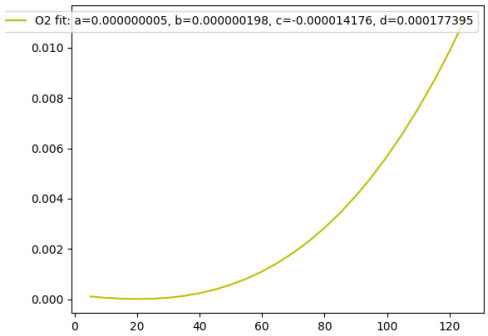



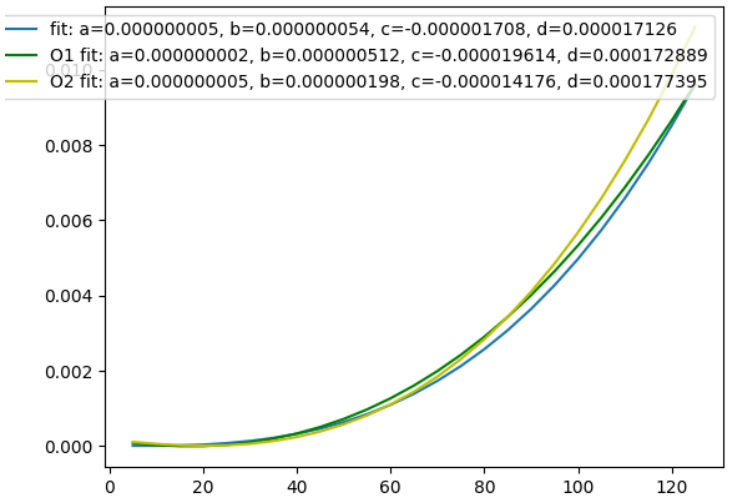
| N | O- | O1 | O2 |
|---|---|---|---|
| 500 | 4.5886e-05 | 4.7776e-05 | 5.2733e-05 |
| 1000 | 9.3484e-05 | 0.000105841 | 0.000115869 |
| 1500 | 0.000155393 | 0.000165102 | 0.000180243 |
| 2000 | 0.0002156 | 0.000226943 | 0.000250445 |
| 2500 | 0.00027561 | 0.000293035 | 0.000318913 |
| 3000 | 0.000335879 | 0.000370002 | 0.00038954 |
| 3500 | 0.000400699 | 0.000423861 | 0.000472134 |
| 4000 | 0.000462112 | 0.000537971 | 0.00053706 |
| 4500 | 0.000574792 | 0.000613159 | 0.000681049 |
| 5000 | 0.00062758 | 0.000688875 | 0.000936752 |
| 5500 | 0.000696955 | 0.000764817 | 0.00107965 |
| 6000 | 0.000771178 | 0.000843093 | 0.00137774 |
| 6500 | 0.0008542 | 0.000930073 | 0.00153718 |
| 7000 | 0.000948121 | 0.00100329 | 0.0016506 |
| 7500 | 0.000989718 | 0.00108493 | 0.00176835 |
| 8000 | 0.00106711 | 0.00118179 | 0.00193712 |
| 8500 | 0.00113612 | 0.00124409 | 0.00206274 |
| 9000 | 0.00121429 | 0.00132679 | 0.00220556 |
| 9500 | 0.00128797 | 0.00140478 | 0.00231064 |
| 10000 | 0.00136625 | 0.00149198 | 0.00250894 |
| 10500 | 0.00144845 | 0.00157132 | 0.00262056 |
| 11000 | 0.00151039 | 0.00166773 | 0.00271144 |
| 11500 | 0.00158849 | 0.00174876 | 0.00295809 |
| 12000 | 0.00166249 | 0.00188693 | 0.00302842 |
| 12500 | 0.00174171 | 0.00191495 | 0.00311223 |

## 1.7 Algoritmo floyd



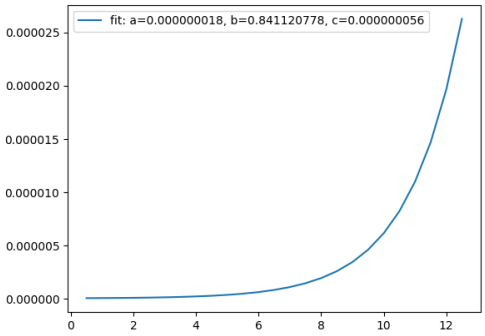



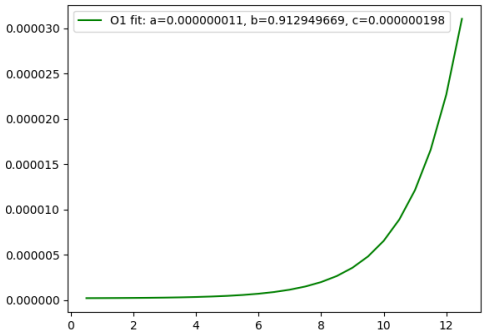



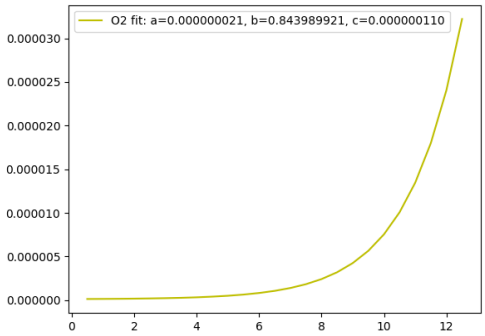



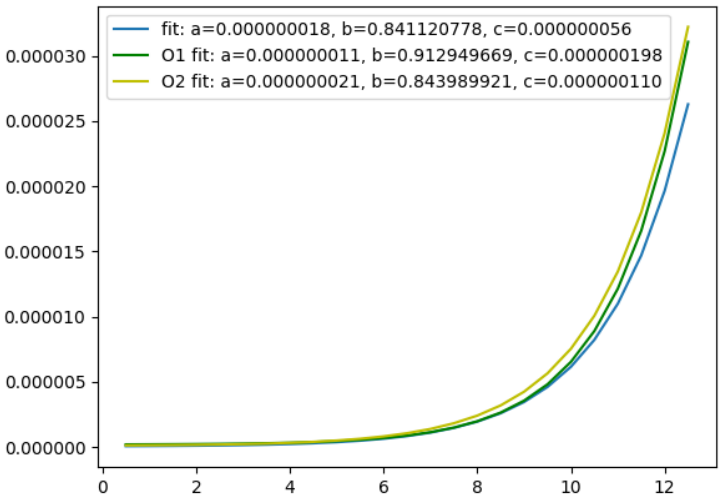
| **N** | **O-** | **O1** | **O2** |
|---|---|---|---|
| 5 | 2e-06 | 1e-06 | 2e-06 |
| 10 | 7e-06 | 7e-06 | 7e-06 |
| 15 | 2.1e-05 | 2.1e-05 | 2.1e-05 |
| 20 | 4.6e-05 | 5e-05 | 4.9e-05 |
| 25 | 8.3e-05 | 8.9e-05 | 8.7e-05 |
| 30 | 0.000145 | 0.000149 | 0.000146 |
| 35 | 0.000227 | 0.000244 | 0.00023 |
| 40 | 0.000335 | 0.000377 | 0.000336 |
| 45 | 0.000455 | 0.000512 | 0.000455 |
| 50 | 0.000636 | 0.000731 | 0.000642 |
| 55 | 0.000834 | 0.000952 | 0.000841 |
| 60 | 0.001134 | 0.001233 | 0.001076 |
| 65 | 0.001432 | 0.001545 | 0.001376 |
| 70 | 0.001715 | 0.001924 | 0.001768 |
| 75 | 0.002059 | 0.00235 | 0.002205 |
| 80 | 0.002596 | 0.002852 | 0.002645 |
| 85 | 0.003052 | 0.003377 | 0.003118 |
| 90 | 0.003596 | 0.003975 | 0.004017 |
| 95 | 0.004309 | 0.00473 | 0.004775 |
| 100 | 0.004978 | 0.005552 | 0.005583 |
| 105 | 0.005801 | 0.005681 | 0.006403 |
| 110 | 0.006597 | 0.007624 | 0.009087 |
| 115 | 0.007513 | 0.007536 | 0.008881 |
| 120 | 0.008491 | 0.008468 | 0.009428 |
| 125 | 0.009594 | 0.009587 | 0.010736 |

## 1.8 Algoritmo hanoi



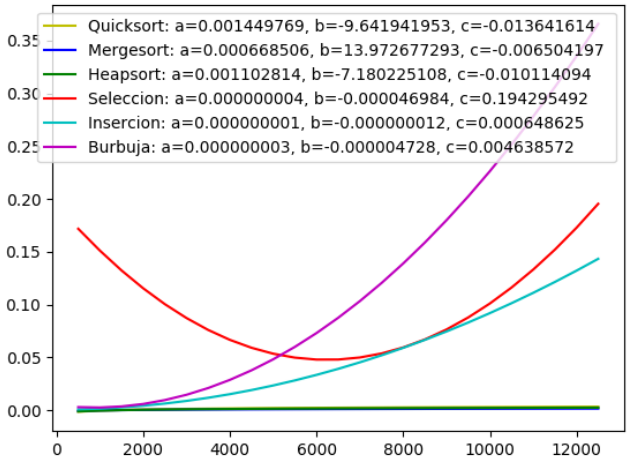



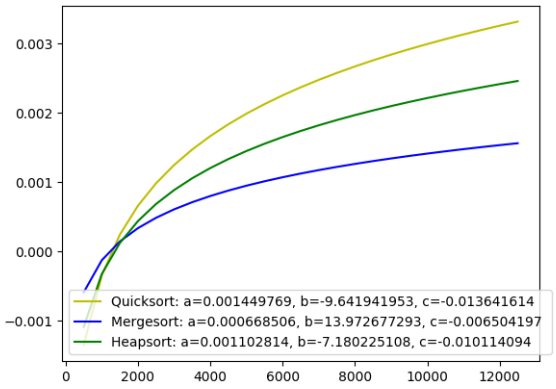



| N | O- | O1 | O2 |
|---|---|---|---|
| 0.5 | 9.3e-08 | 1.43e-07 | 1.28e-07 |
| 1.0 | 1.71e-07 | 1.6e-07 | 2.97e-07 |
| 1.5 | 1.66e-07 | 1.99e-07 | 1.53e-07 |
| 2.0 | 2.04e-07 | 2.7e-07 | 2.29e-07 |
| 2.5 | 1.41e-07 | 1.45e-07 | 2.08e-07 |
| 3.0 | 2.51e-07 | 2.92e-07 | 3.47e-07 |
| 3.5 | 2.54e-07 | 2.86e-07 | 3.04e-07 |
| 4.0 | 3.91e-07 | 4.69e-07 | 4.54e-07 |
| 4.5 | 3.62e-07 | 4.08e-07 | 4.67e-07 |
| 5.0 | 5.49e-07 | 6.58e-07 | 7.21e-07 |
| 5.5 | 5.26e-07 | 6.2e-07 | 6.98e-07 |
| 6.0 | 8.55e-07 | 9.13e-07 | 1.057e-06 |
| 6.5 | 7.52e-07 | 8.07e-07 | 1.037e-06 |
| 7.0 | 1.16e-06 | 1.374e-06 | 1.563e-06 |
| 7.5 | 1.199e-06 | 1.497e-06 | 1.638e-06 |
| 8.0 | 2.104e-06 | 2.197e-06 | 2.487e-06 |
| 8.5 | 2.187e-06 | 2.384e-06 | 2.744e-06 |
| 9.0 | 3.666e-06 | 3.97e-06 | 4.553e-06 |
| 9.5 | 3.732e-06 | 3.788e-06 | 4.352e-06 |
| 10.0 | 6.565e-06 | 7.148e-06 | 8.174e-06 |
| 10.5 | 6.523e-06 | 7.279e-06 | 8.151e-06 |
| 11.0 | 1.242e-05 | 1.3911e-05 | 1.5101e-05 |
| 11.5 | 1.2321e-05 | 1.365e-05 | 1.5014e-05 |
| 12.0 | 2.4334e-05 | 2.6554e-05 | 2.9694e-05 |
| 12.5 | 2.4104e-05 | 2.9556e-05 | 2.9656e-05 |

# 2 Algoritmos de búsqueda




En esta gráfica vemos la representación de todos los algoritmos de la práctica, cómo el de burbuja crece mucho más rápidamente que el resto mientras que quicksort, mergesort y heapsort son rectas. Así, concluimos que burbuja ¿ selección ¿ inserción ¿ mergesort, quicksort y heapsort con respecto a eficiencia, algo que veníamos intuyendo al ver que orden tenían al principio del documento.




Para poder diferenciar mejor los algoritmos con $O(n \log(n))$ los vemos en otra gráfica. Apreciamos entonces cómo para tamaños pequeños, hasta 2000 aproximadamente, el algoritmo mergesort es más eficiente que el heapsort, que a su vez es mejor que el quicksort, mas esto cambia y pasa justo lo contrario a partir de 2000.