Analyzing Real Timings

Using timings to evaluate performance

By the end of this video you will be able to…

- Use runtimes from a real system to reason about performance

Idea for Analyzing our Sorts

For increasing sizes of n

Print n

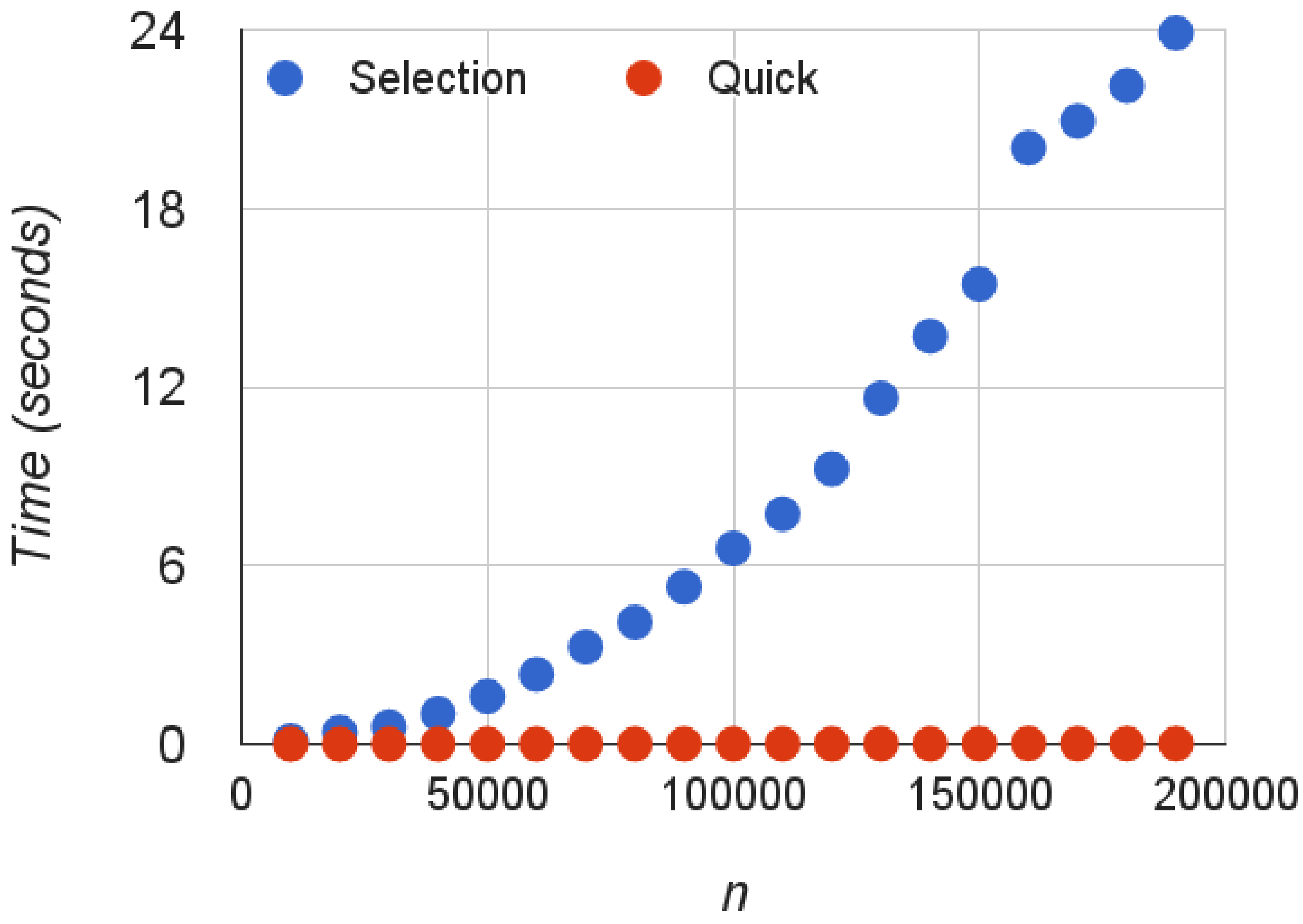
Create a randomized array of size n
Time selection sort, print outcome

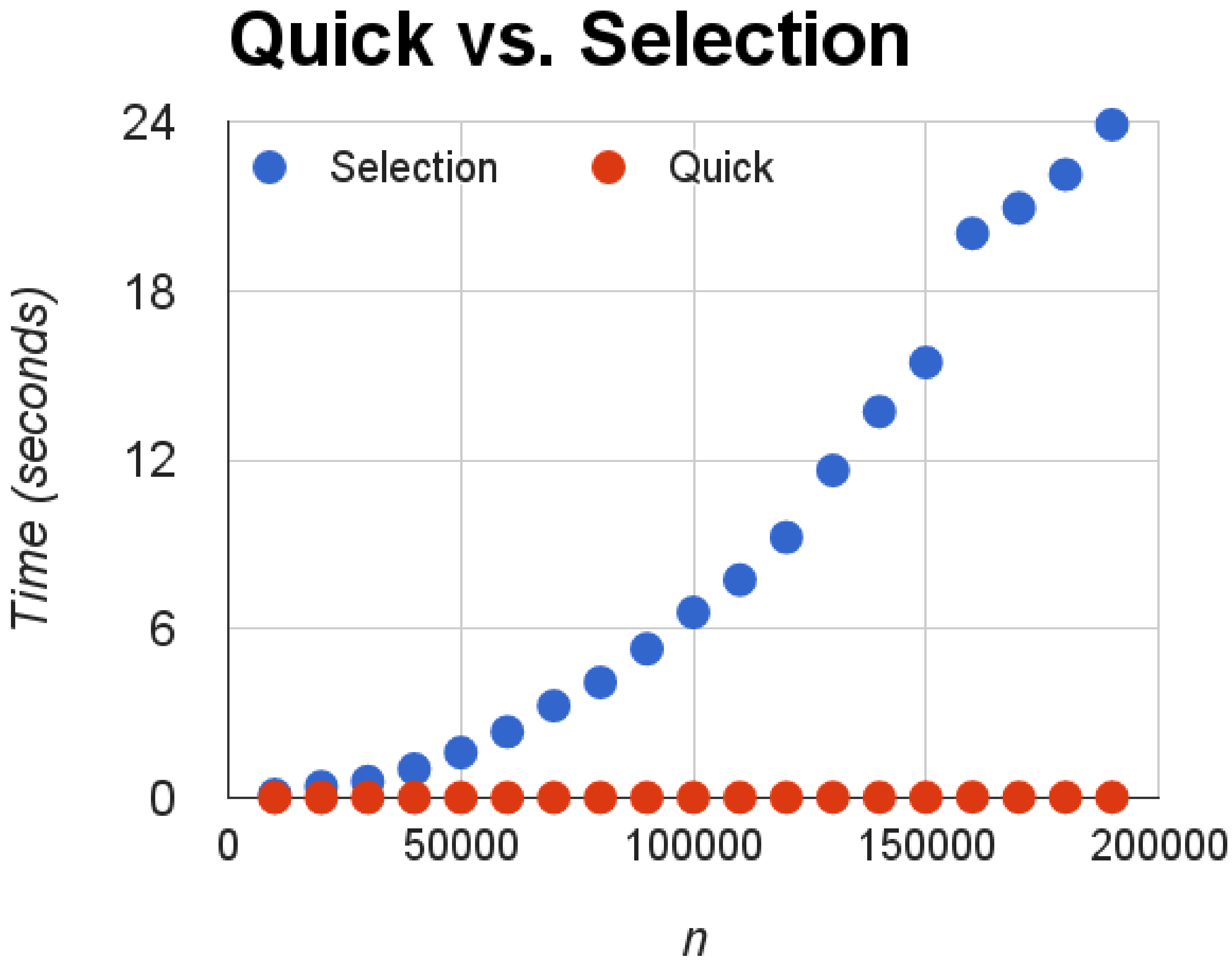
Create a randomized array of size n
Time quick sort, print outcome

Results

n	Selection (s)	Quick (s)
10000	0.112887621	0.001323534
20000	0.397227565	0.001568662
30000	0.580318935	0.002420492
40000	1.020979179	0.003304295
50000	1.605557659	0.004232703
60000	2.340087449	0.004983088
70000	3.264979954	0.006035047
80000	4.097073897	0.006989112
90000	5.285101776	0.007900941
100000	6.57904119	0.008538038

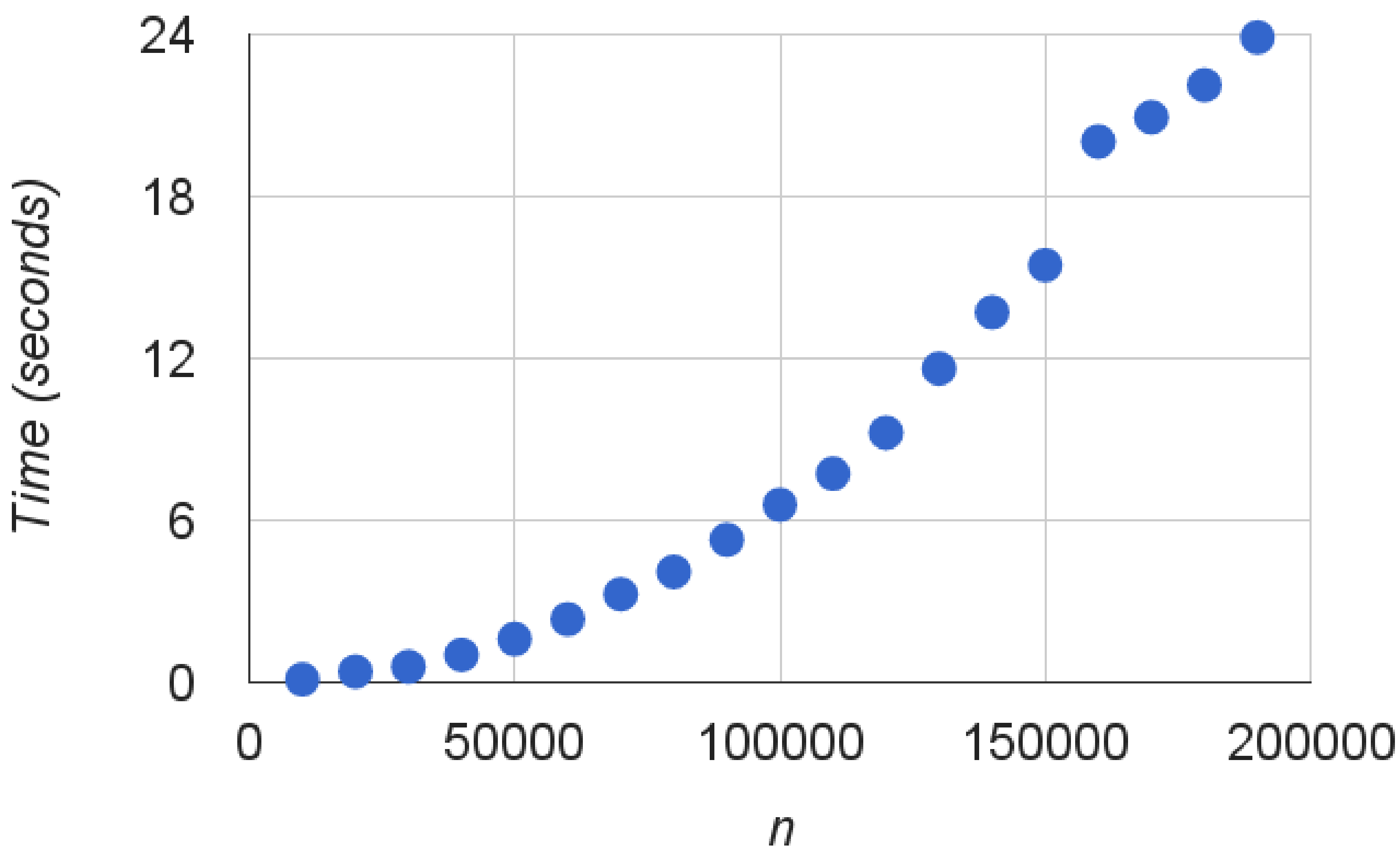
Quick vs. Selection

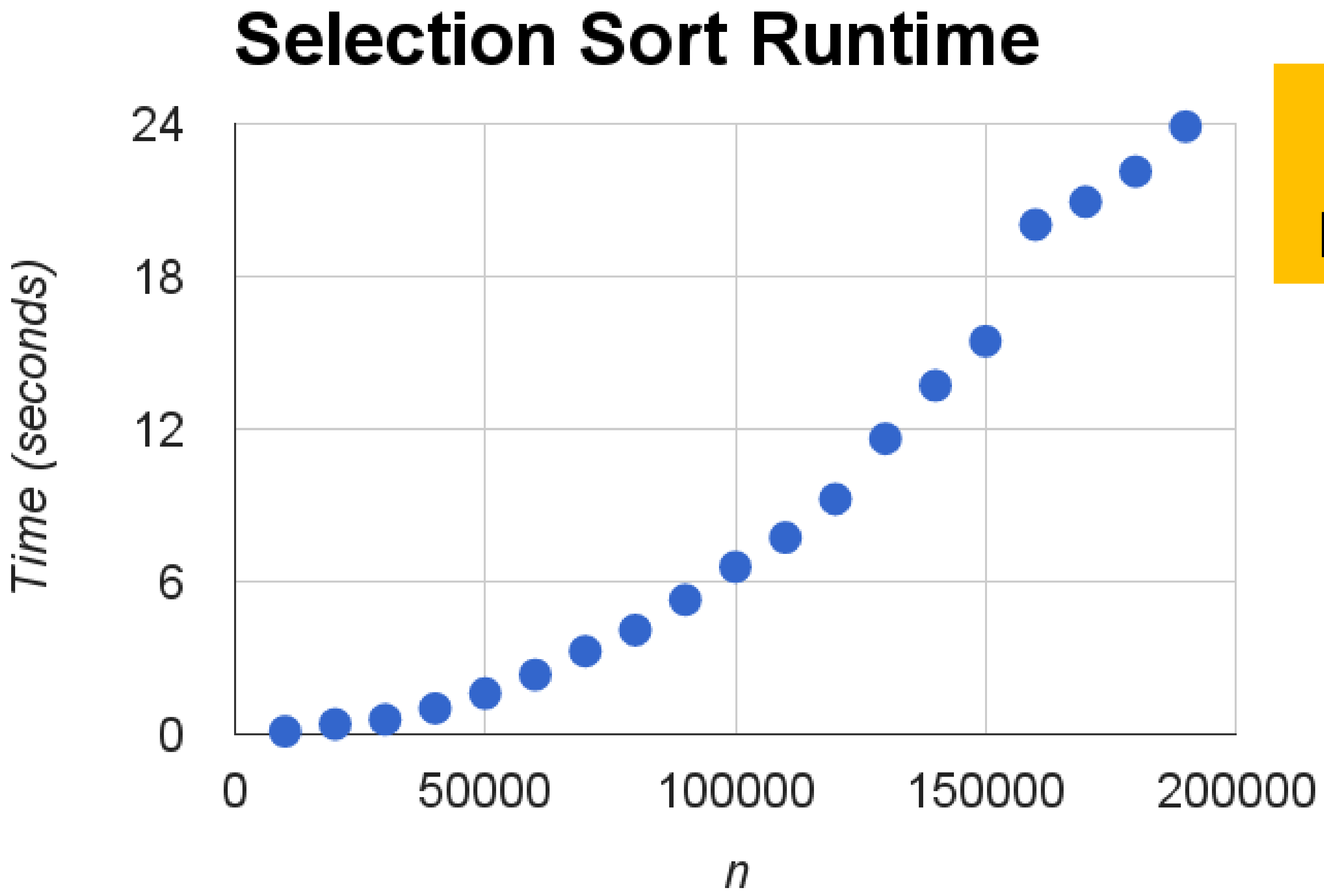




Let’s look at each in detail

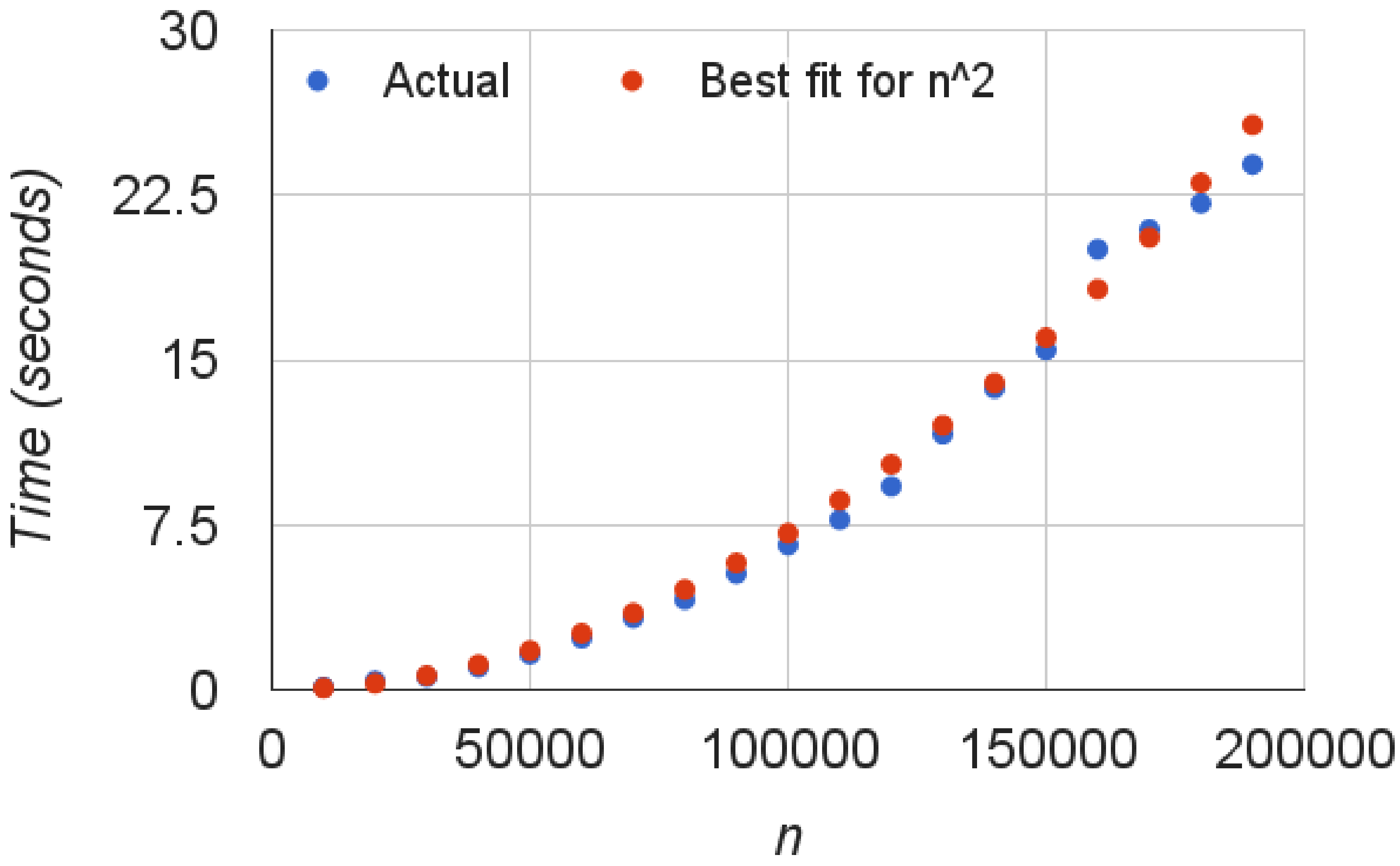
Selection Sort Runtime

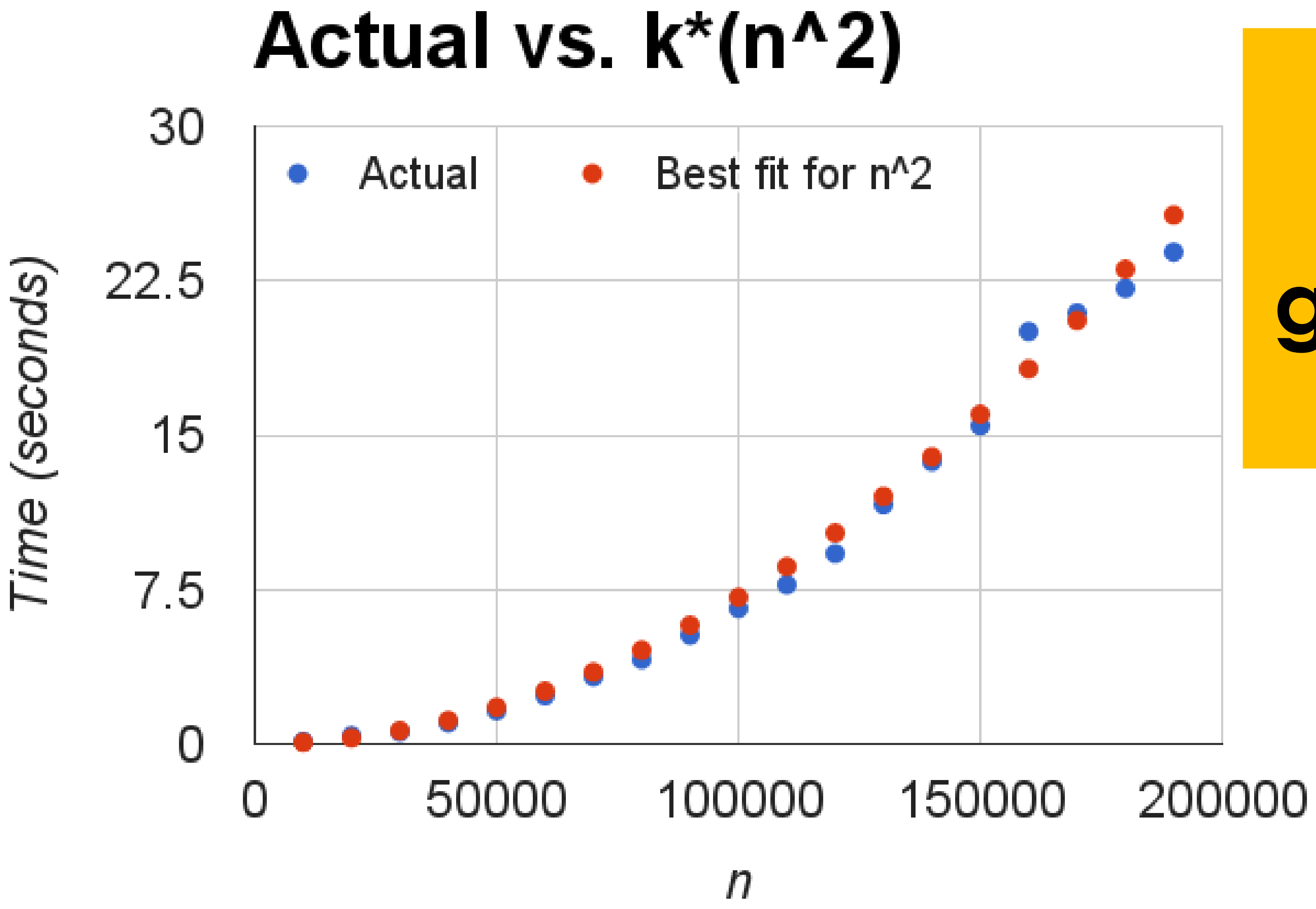




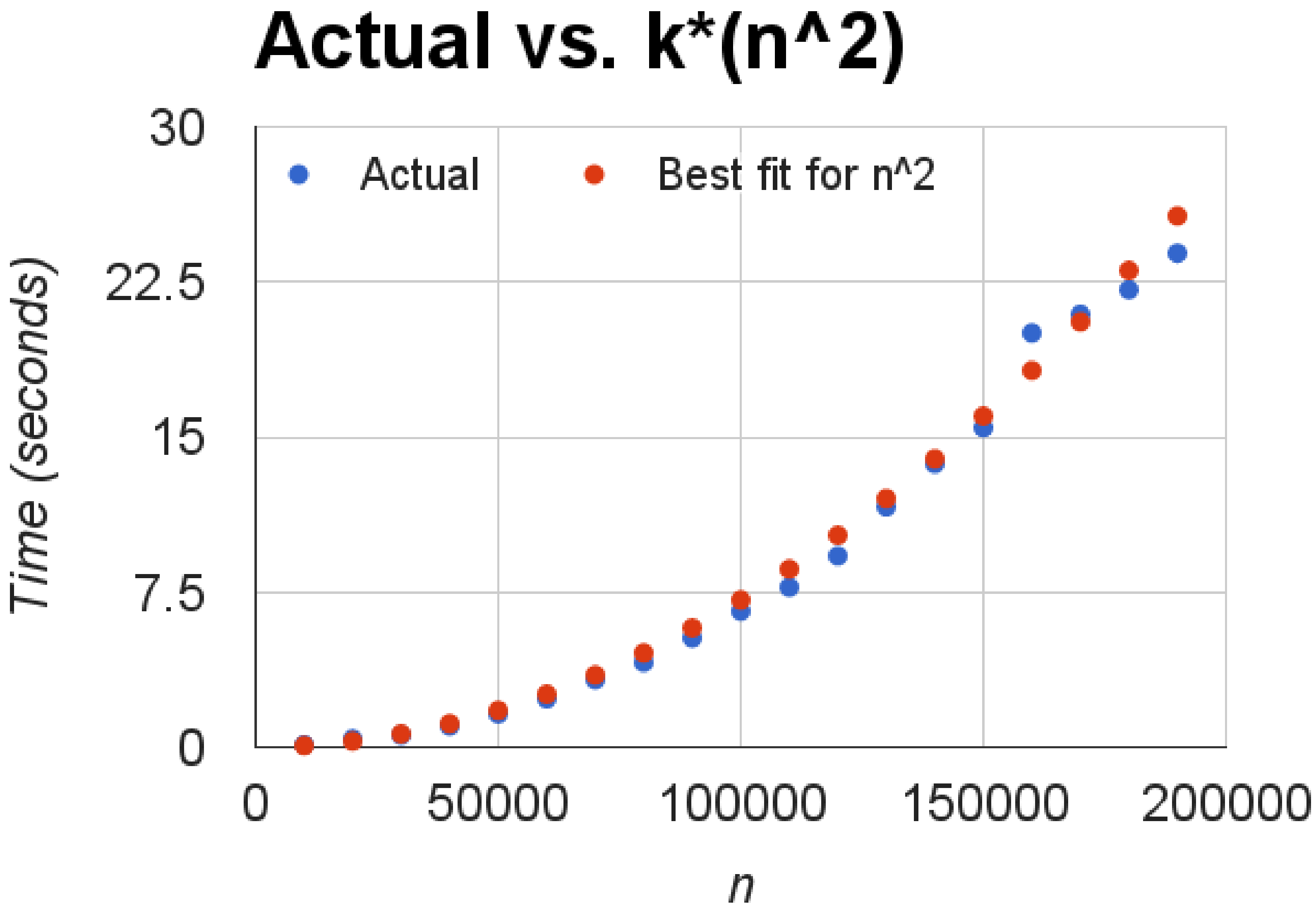
Looks like n^2 growth

Actual vs. k*(n^2)



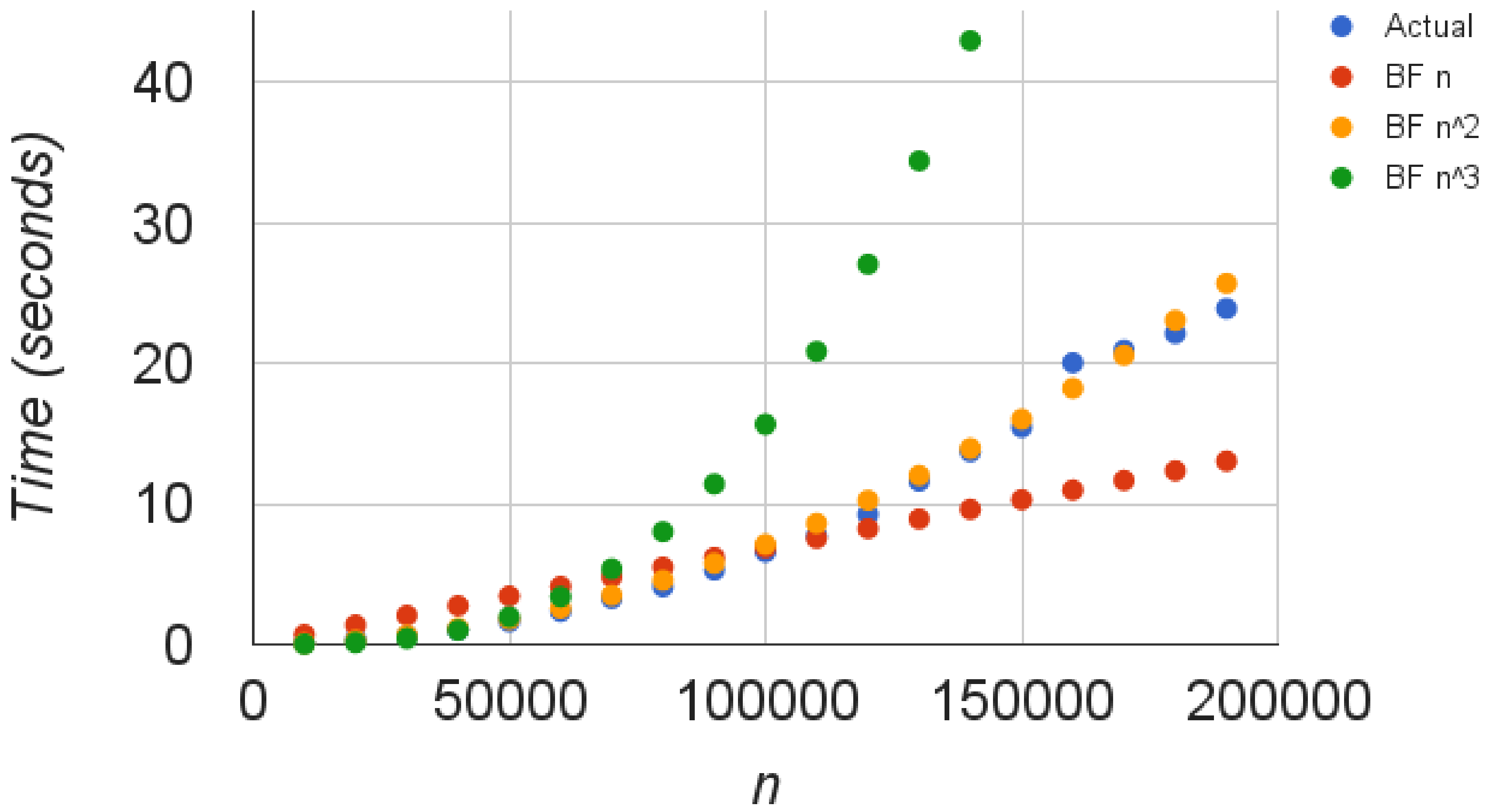


By "best fit" I just found a good value for constant "k"

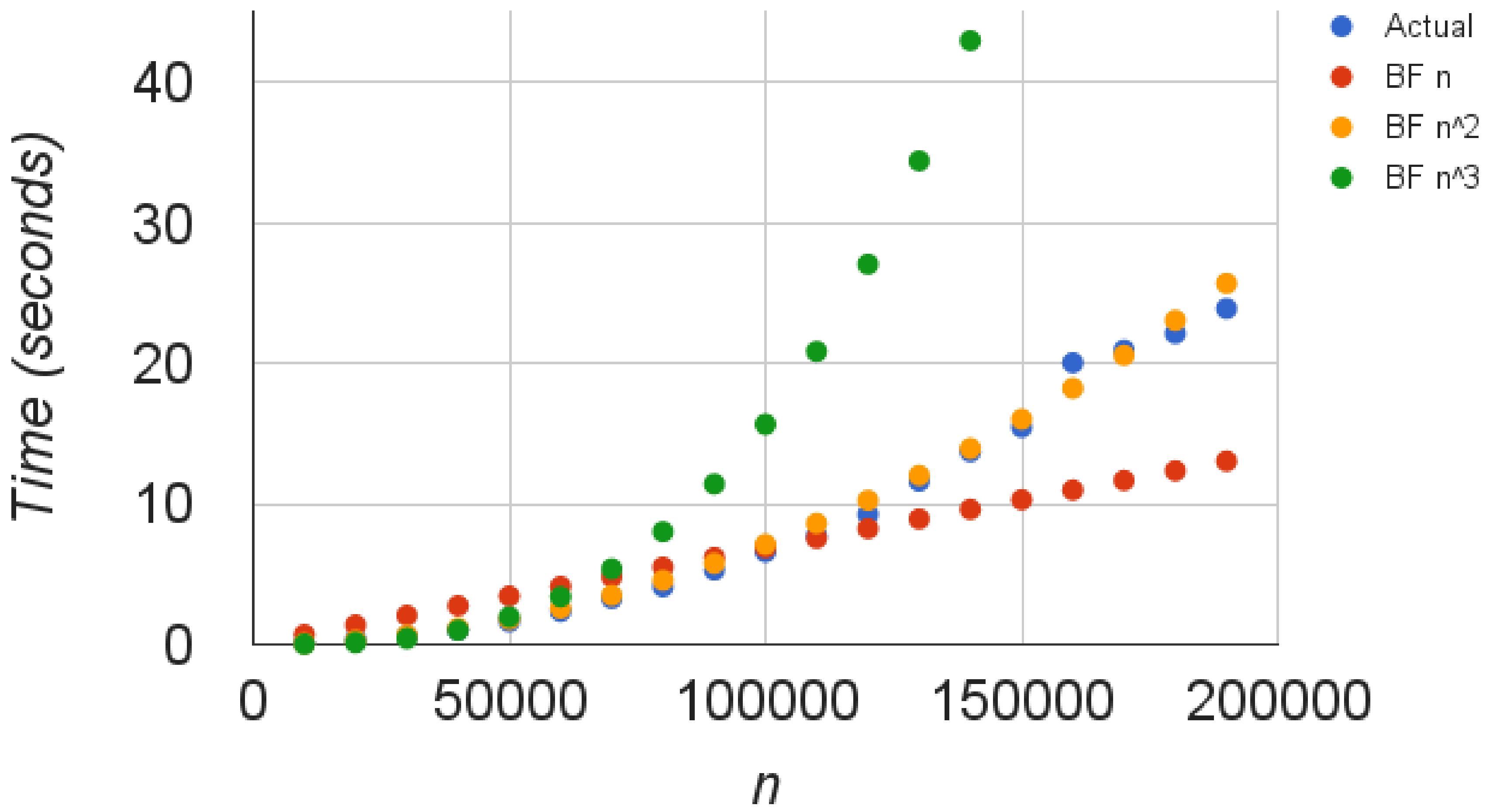


Won’t all “best fits” look really good?

Actual vs. Best Fits

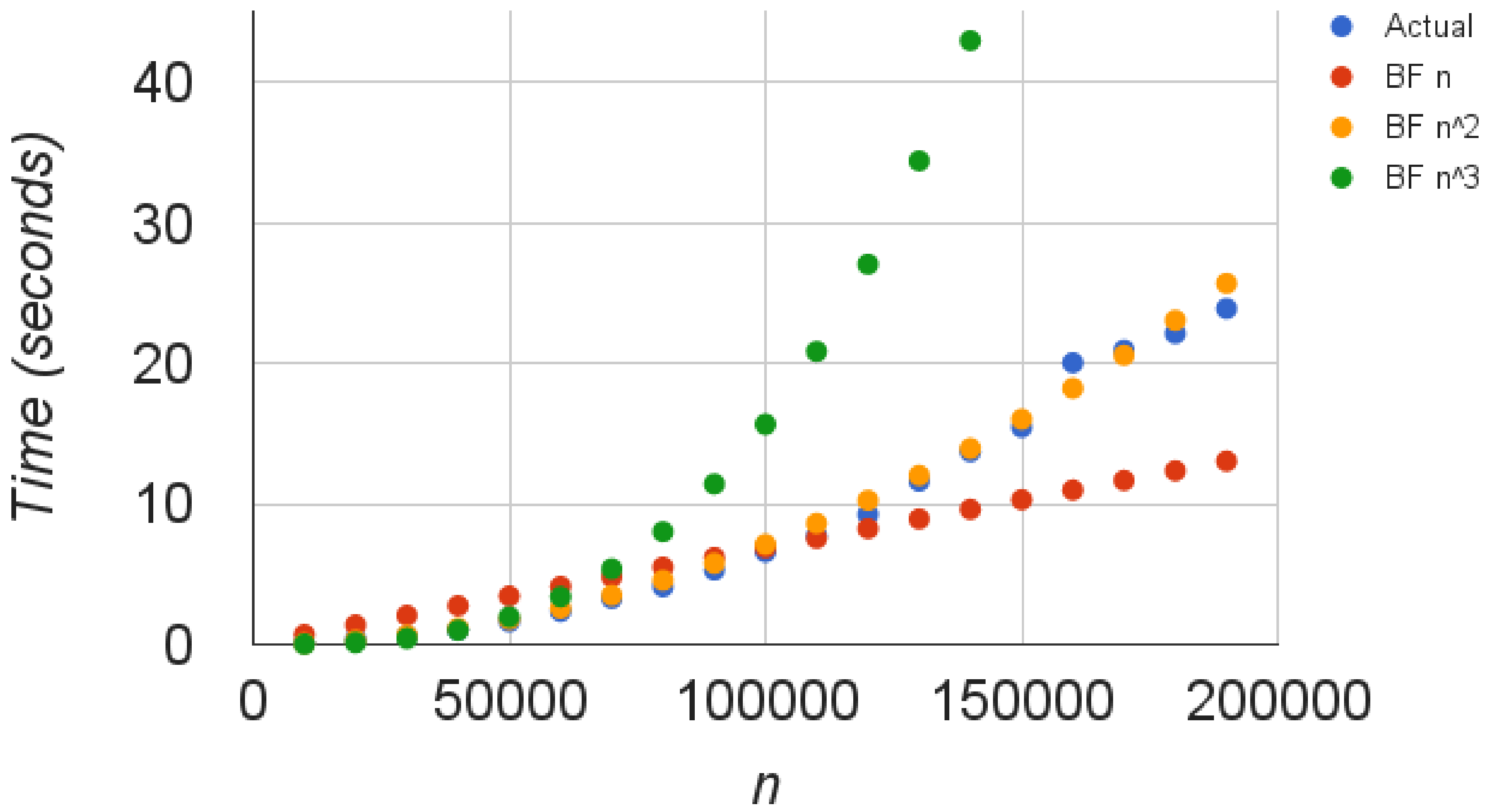


Actual vs. Best Fits

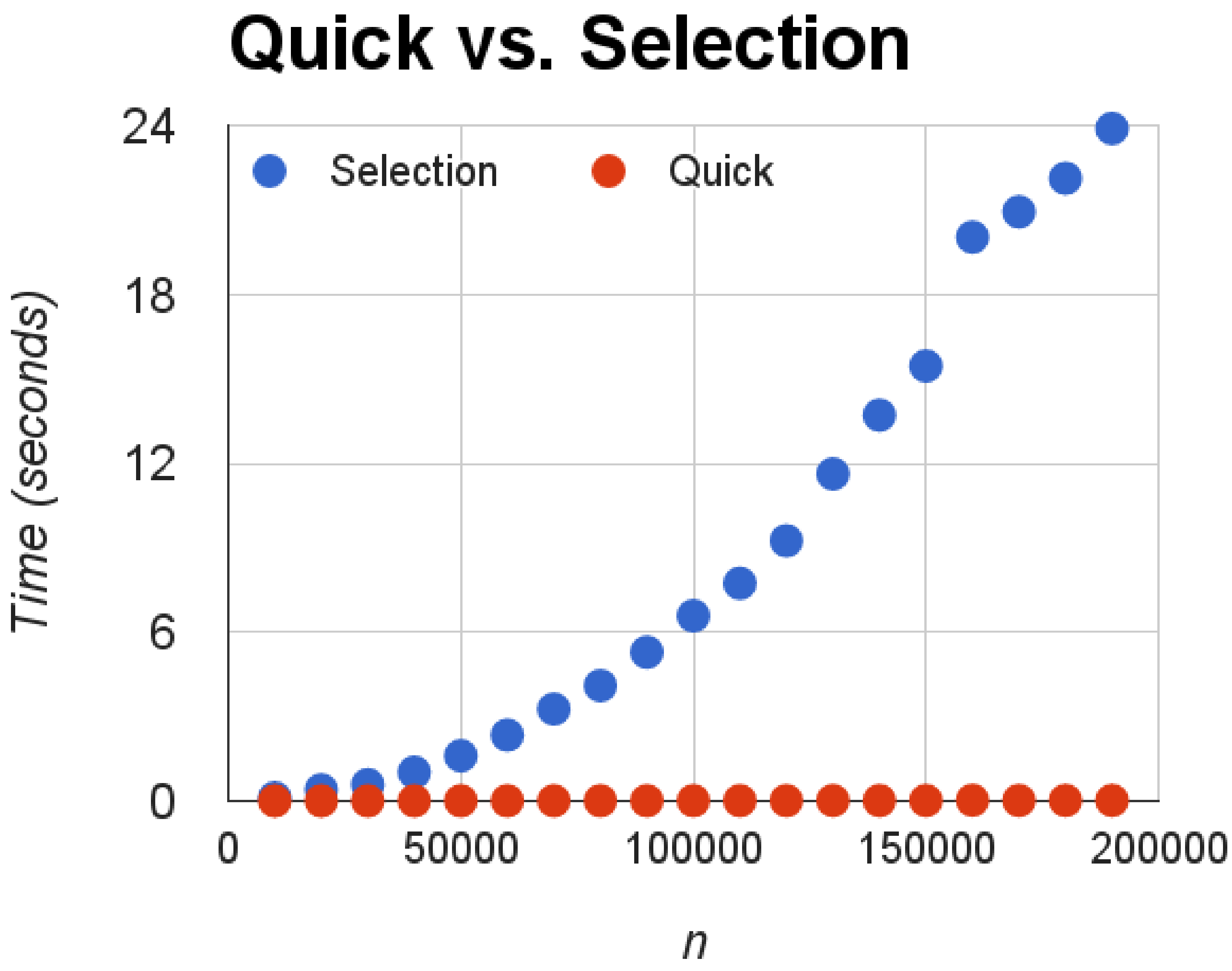


n^2 is best

Actual vs. Best Fits

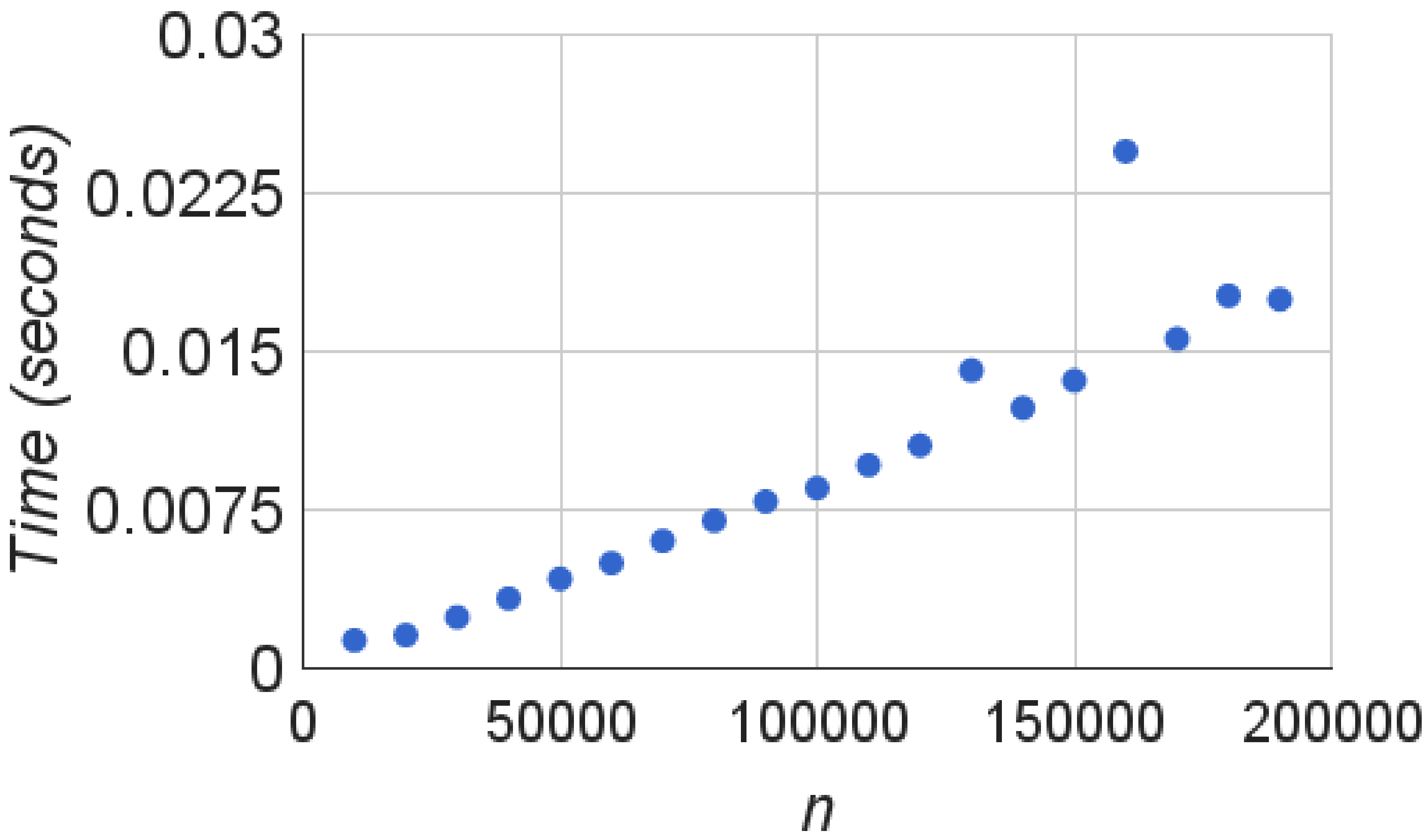


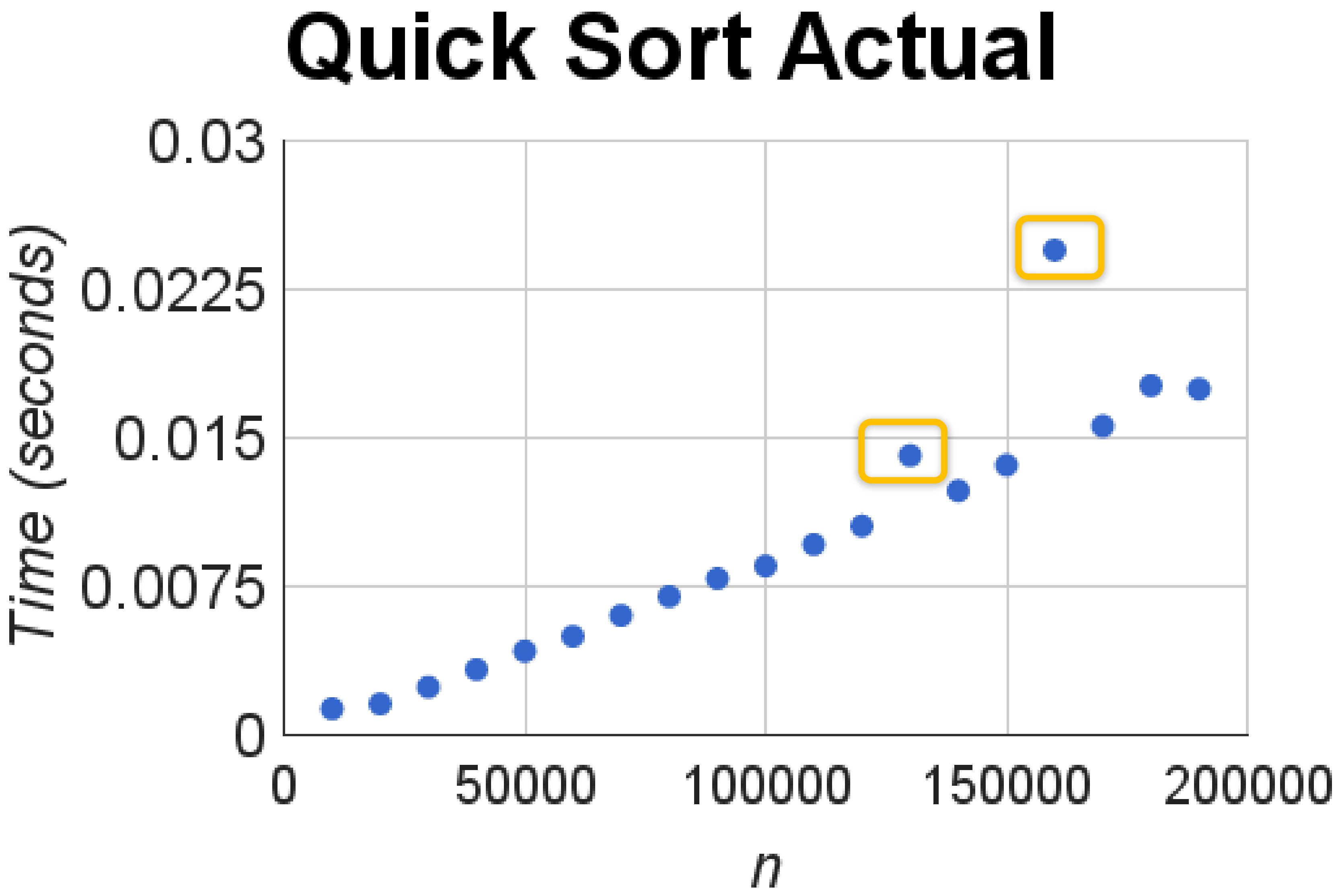
n^2 is best



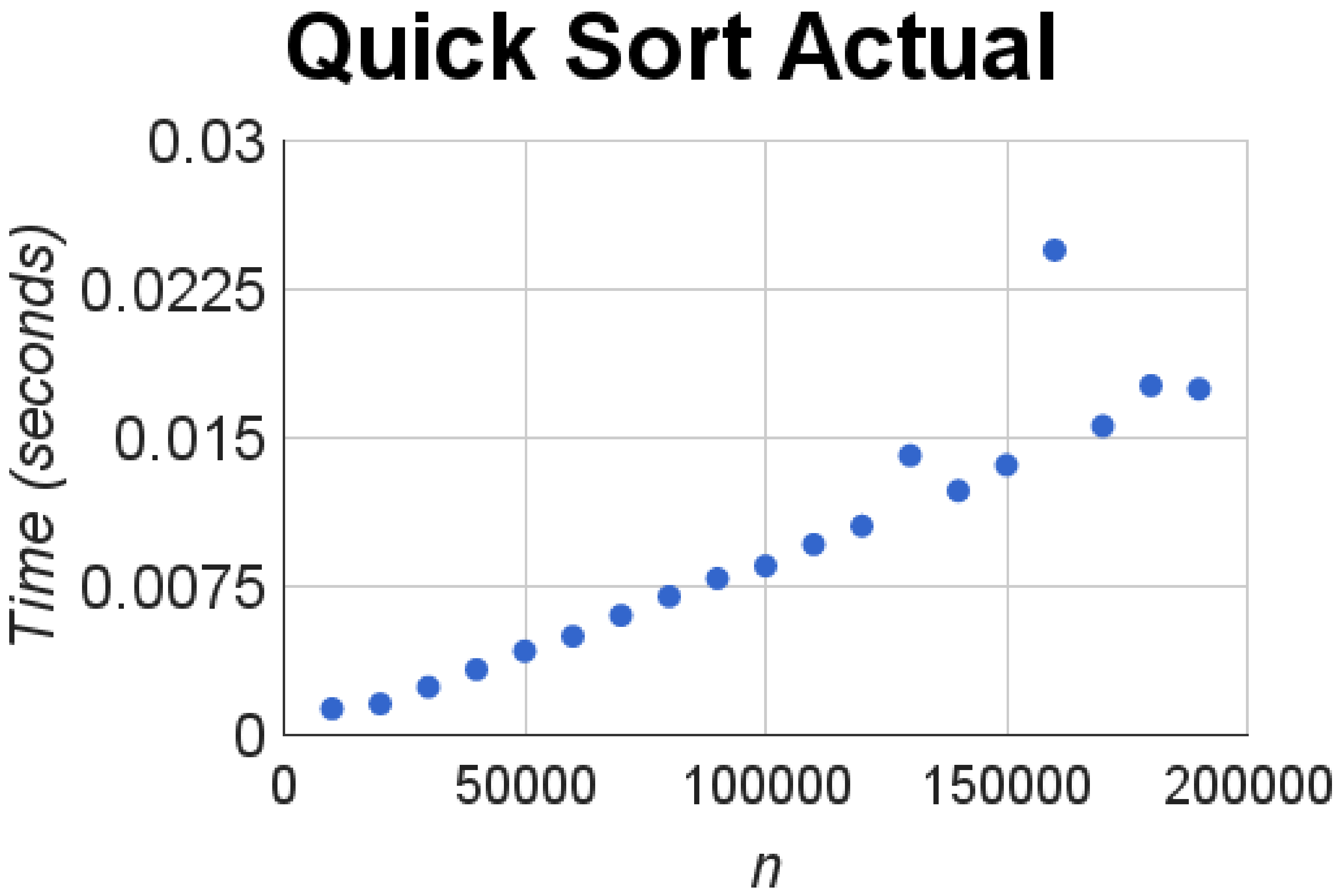
Zoom in on quick sort:

Quick Sort Actual

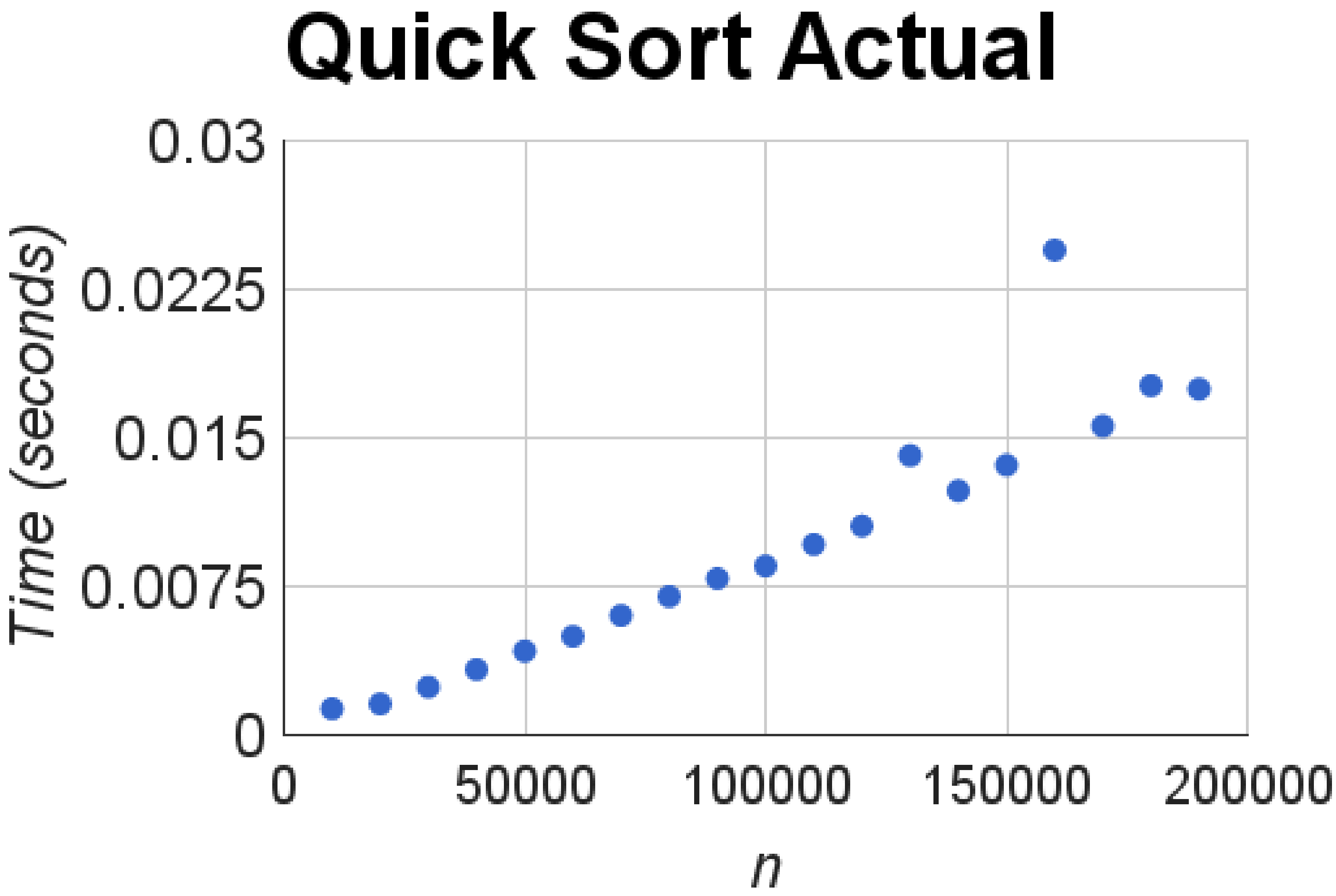



Quick Sort Actual
Time (seconds)
0.03
0.0225
0.015
0.0075
0
50000
100000
150000
200000
n

Real data…

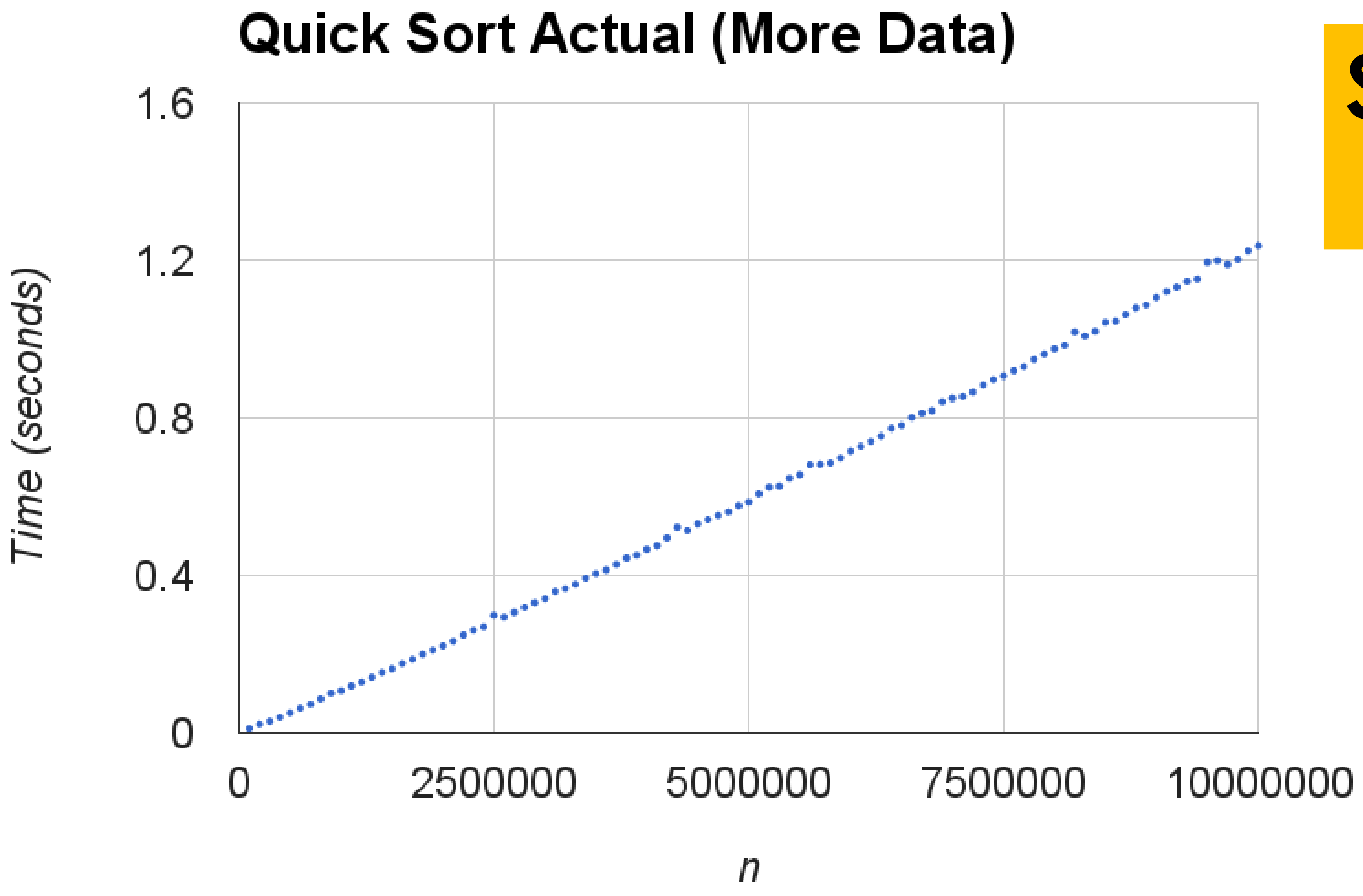


Looks linear?



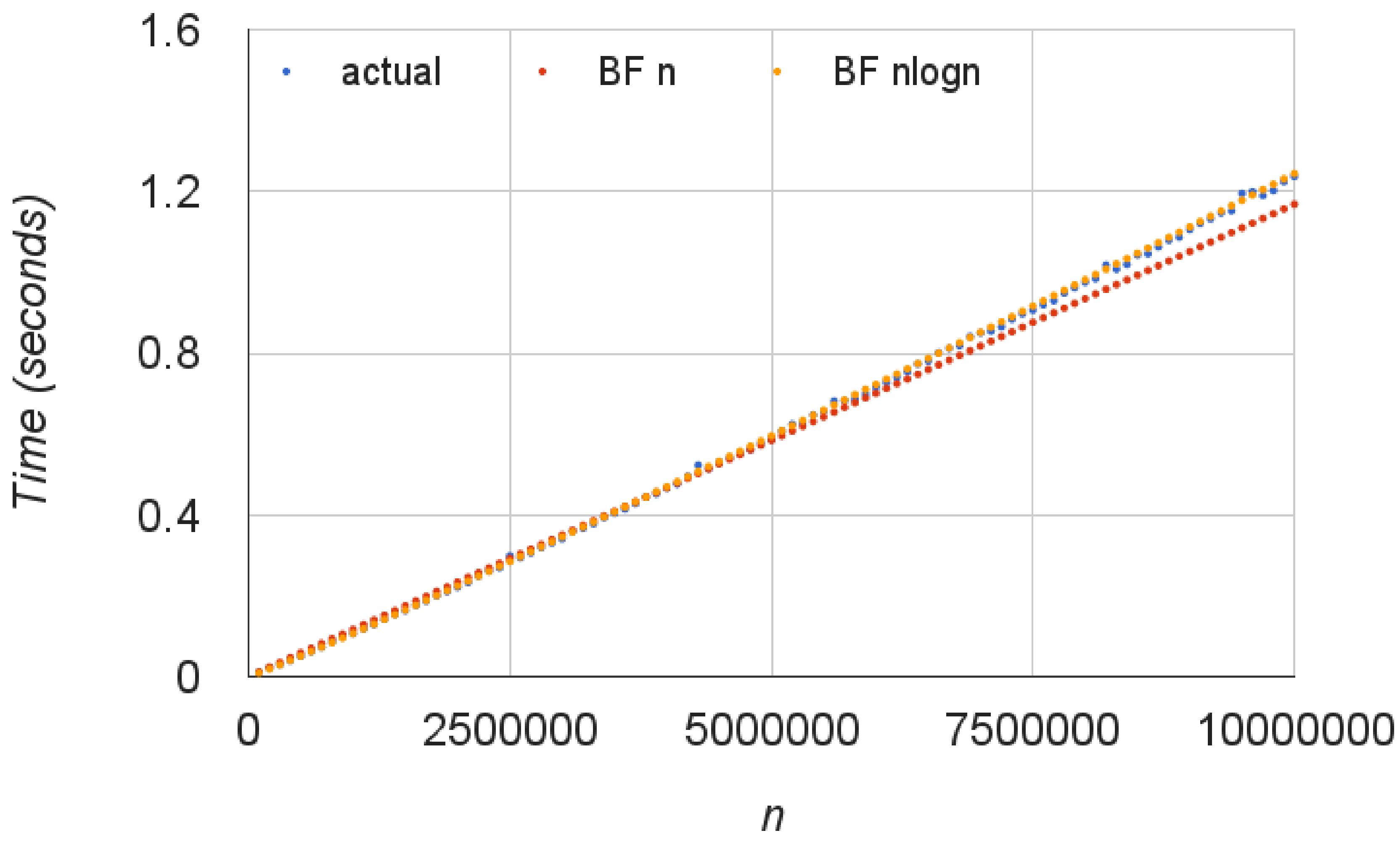
Looks linear?

Get more data...



Still appears linear

Actual vs. Best Fits



How big is log(n)?

n	Log_2 n
10,000,000	?

How big is log(n)?

n	Log_2 n
10,000,000	~23

How big is log(n)?

n	Log_2 n
10,000,000	~23

log n is just really small relative to n

Summary

- We can use real runtimes to reason about performance
- Be prepared for real system data to be noisy
- Can be really useful when we want to understand actual performance on a real system