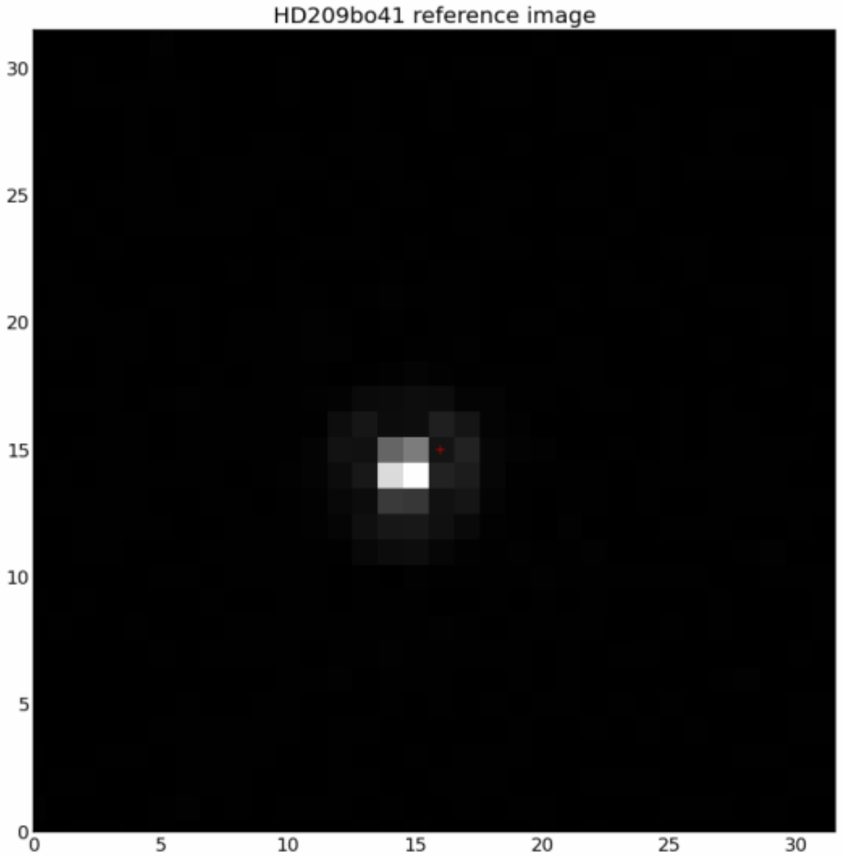
Optimal Aperture Calculation

With the *Spitzer Space Telescope* Star Image Dataset

Hannah Diamond-Lowe, Zakir Gowani

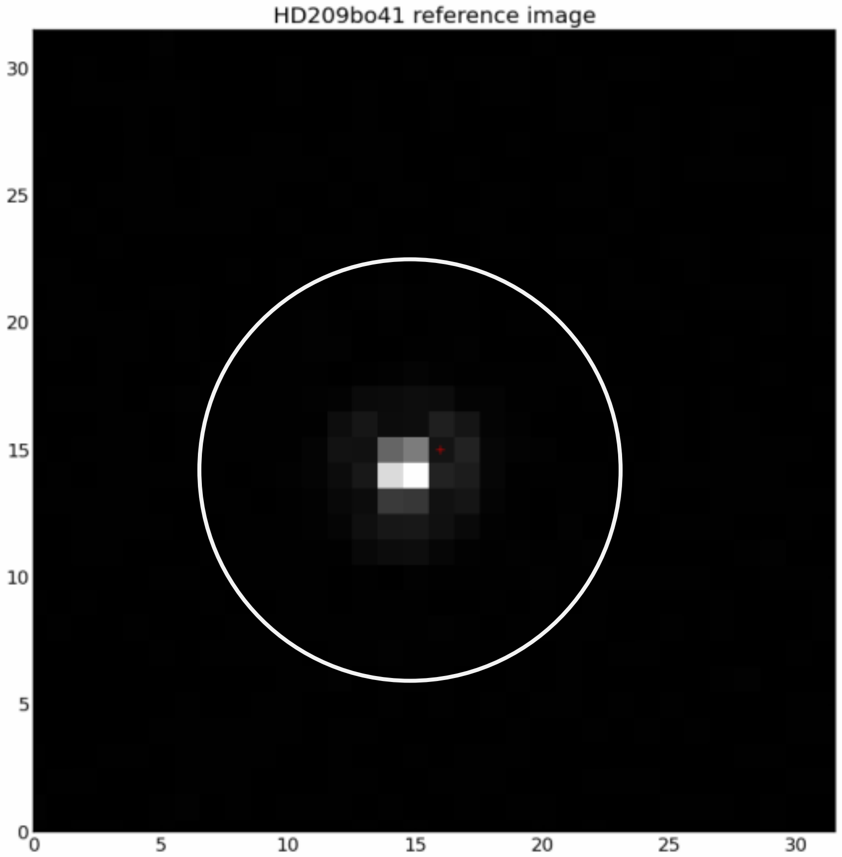
Overview

- Input: 32x32 pixel image of star
- Output: aperture radius that conserves star signal and rejects noise
- Additional goal: Parallelize calculation by most efficient means possible



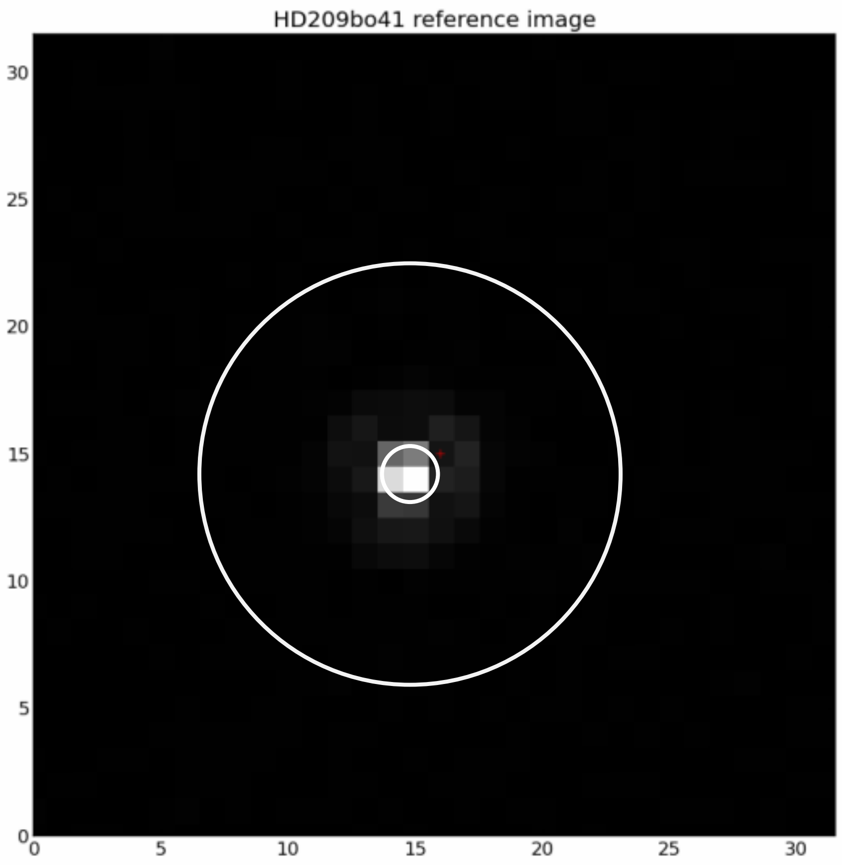
Motivation

- Too large an aperture may result in too much noise



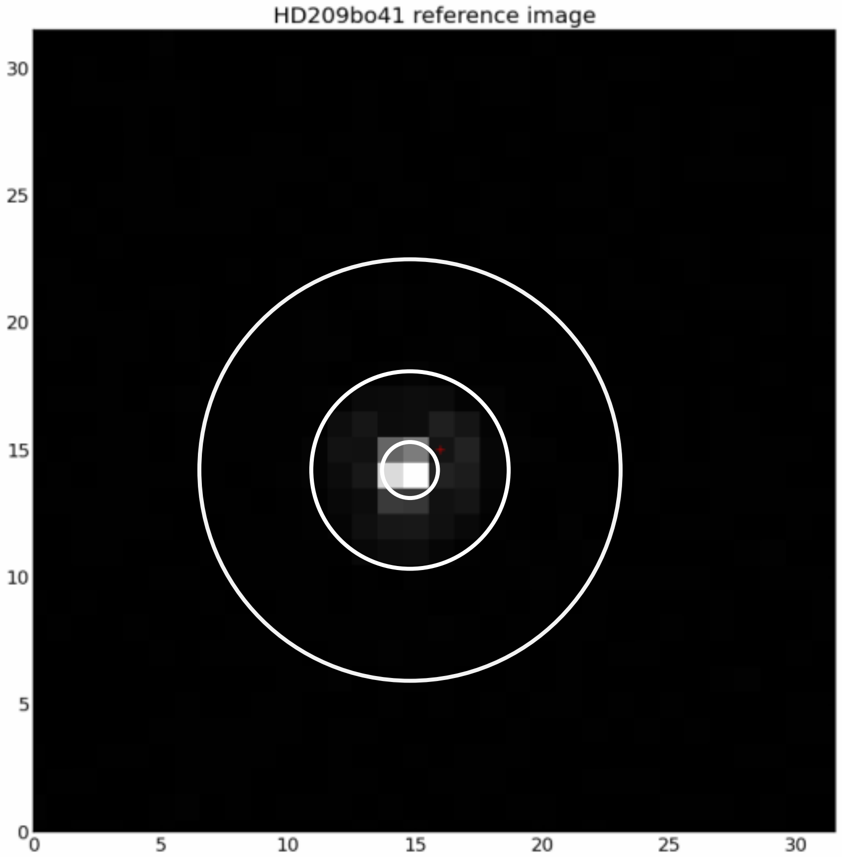
Motivation

- Too small = loss of information



Motivation

- Just right = the most information with the least amount of background noise
- If this step is not done properly the whole analysis will be wrong



Problem

- Finding the best aperture is time consuming
- The best solution may consist of a range of values
- Example of desired output: best aperture radius lies between 3 and 4 pixels

Data Format

- FITS files (pictures)
- 64 data frames per FITS file
- ~4 - 8GB per dataset
- Our chosen datasets total to 20GB

```
.0 17.0 29.0 21.0 30.0 25.0 18.0 17.0 23.0 17.0 15.0 6.0 30.0 29.0 26.0 11.0 9.0 7.0 27.0 8.
19.0 9.0 26.0 28.0 13.0 17.0 11.0 13.0 26.0 20.0 20.0 8.0 33.0 10.0 31.0 16.0 11.0 17.0 -4.0
17.0 12.0 36.0 25.0 -7.0 35.0 38.0 20.0 38.0 34.0 16.0 27.0 31.0 33.0 14.0 37.0 16.0 17.0 25
16.0 19.0 24.0 68.0 21.0 25.0 18.0 44.0 49.0 33.0 31.0 14.0 33.0 26.0 21.0 28.0 19.0 16.0 14
 17.0 11.0 25.0 26.0 15.0 29.0 41.0 73.0 79.0 54.0 35.0 8.0 37.0 17.0 25.0 18.0 13.0 10.0 26
 24.0 38.0 46.0 33.0 34.0 31.0 43.0 70.0 96.0 53.0 36.0 32.0 25.0 11.0 40.0 23.0 37.0 44.0 1
15.0 14.0 14.0 15.0 23.0 53.0 69.0 454.0 756.0 136.0 51.0 42.0 34.0 30.0 18.0 2.0 9.0 6.0 18
2.0 23.0 19.0 27.0 32.0 67.0 135.0 1404.0 2788.0 550.0 108.0 60.0 23.0 37.0 34.0 12.0 17.0 2
 19.0 27.0 25.0 34.0 31.0 49.0 530.0 4357.0 5854.0 1855.0 273.0 48.0 60.0 23.0 17.0 23.0 16.
 6.0 3.0 15.0 76.0 71.0 279.0 709.0 807.0 604.0 238.0 79.0 30.0 31.0 19.0 39.0 -2.0 9.0 2.0
28.0 16.0 27.0 0.0 19.0 87.0 88.0 66.0 89.0 78.0 45.0 40.0 45.0 15.0 27.0 18.0 27.0 14.0 17.
.0 16.0 13.0 36.0 8.0 34.0 57.0 91.0 14.0 40.0 27.0 32.0 19.0 9.0 14.0 24.0 16.0 26.0 7.0 11
10.0 16.0 13.0 35.0 25.0 26.0 44.0 58.0 34.0 26.0 26.0 10.0 19.0 -1.0 22.0 0.0 15.0 -7.0 16.
.0 22.0 16.0 25.0 38.0 8.0 28.0 40.0 15.0 12.0 37.0 26.0 29.0 12.0 -7.0 15.0 14.0 22.0 9.0 2
 14.0 14.0 -1.0 24.0 22.0 24.0 26.0 29.0 29.0 28.0 9.0 15.0 26.0 25.0 -15.0 15.0 18.0 -4.0 1
-2.0 18.0 11.0 37.0 28.0 27.0 22.0 13.0 16.0 22.0 16.0 2.0 27.0 22.0 20.0 13.0 24.0 21.0 20.
6.0 8.0 27.0 15.0 11.0 22.0 13.0 -3.0 29.0 29.0 12.0 30.0 27.0 19.0 14.0 28.0 25.0 14.0 6.0
15.0 13.0 20.0 12.0 24.0 16.0 26.0 16.0 18.0 23.0 11.0 26.0 30.0 2.0 21.0 7.0 16.0 26.0 6.0
0 -7.0 11.0 47.0 17.0 16.0 21.0 15.0 2.0 11.0 33.0 18.0 10.0 29.0 24.0 12.0 18.0 7.0 17.0 5.
.0 20.0 4.0 18.0 17.0 11.0 4.0 34.0 10.0 16.0 4.0 -8.0 -7.0 -2.0 16.0 25.0 16.0 9.0 -8.0 38.
```

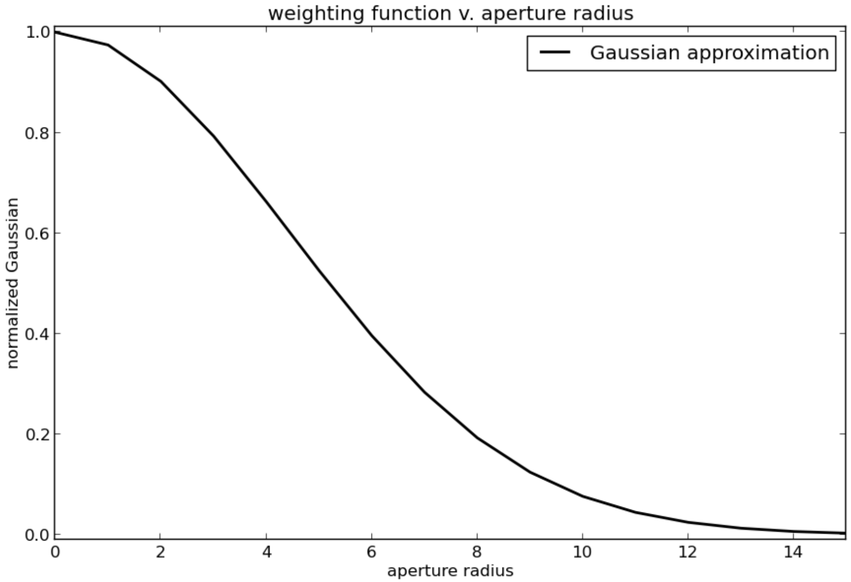
Data Reduction

- Eliminate noise spikes
- Determine how much information to keep
- Tested runtimes for 10 FITS files (~8000 in whole dataset)

0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	70.0	96.0	53.0	0	0	0	0	0	0	0
0	0	0	0	69.0	454.0	756.0	136.0	51.0	0	0	0	0	0	0
0	0	0	67.0	135.0	1404.0	2788.0	550.0	108.0	59.0	0	0	0	0	
0	0	0	49.0	530.0	4357.0	5854.0	1855.0	273.0	48.0	0	0	0	0	
0	0	0	71.0	279.0	709.0	807.0	604.0	238.0	79.0	0	0	0	0	0
0	0	0	0	88.0	66.0	89.0	78.0	45.0	0	0	0	0	0	0
0	0	0	0	0	91.0	14.0	39.0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Basis of algorithm

- Gaussian curve approximation of point spread function
- Penalty function



Parallelization: Method A

- Process all frames in all images
- 64 data frames in a file spread evenly across available threads
- Each thread contributes to analysis of each FITS file
- Runtime for test on 640 images: 288.1 s (extrapolate: 64.02 hr)

Parallelization: Method B

- Sets of files distributed as evenly as possible to each thread
- Each file is handled in its entirety by a single thread
- Runtime for test: 430.0 s (extrapolate: 95.6 hr)
- Significantly slower than Method A (slowest thread characterizes computation)

Parallelization: Method C

- Create a single averaged, composite data frame representing entire FITS file
- Good for data frames taken with short exposure times
- Runtime for test: 11.5 s (extrapolate: 2.6 hr)

Parallelization: Method D

- Determine best aperture size for a sample of data frames in each FITS file
- Choose 4 out of 64 data frames per FITS file
- Good for data frames with long exposure times
- Runtime for test: 21.8 s (extrapolate: 4.84 hr)

And the winner is…

Method C: averaging FITS file into single representative frame.

- Fastest method; only processes one array
- Averaging suitable because it muffles short-timescale noise spikes
- Accuracy is not strongly impacted

Snapshots