Interactive (statistical) visualisation and exploration of the big tabular data with veax

Maarten Breddels

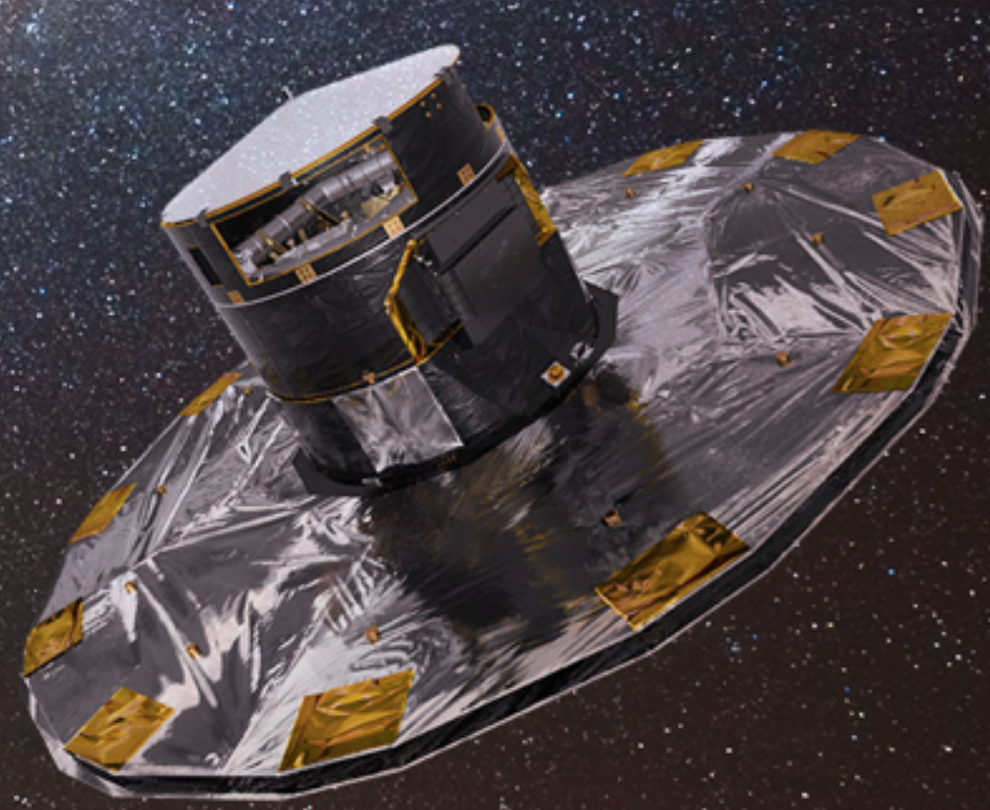
BigSkyEarth 2017 - uclan - UK

university of groningen / faculty of mathematics and natural sciences / kapteyn astronomical institute

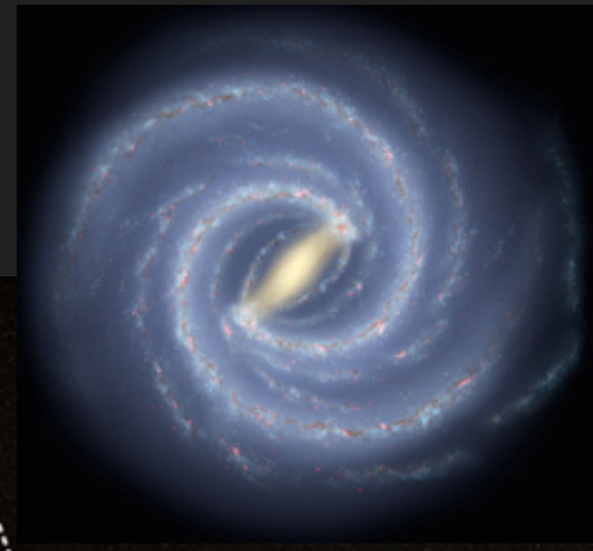
Outline

- Hour 1
  - Intro talk
    - Motivation for vaex
    - Technical details
    - What is vaex
  - Demo 1 (basics)
    - you practice basics with vaex + solve installation issues
      - use your own data?
- Hour 2
  - More advanced (demo)
    - Statistics
    - Selections, virtual columns
    - Exporting, more plotting
  - You practice
- Hour 3
  - ipywidgets (under development)
  - You practice

- Gaia satellite
  - launched by ESA in december 2013
  - determines positions, velocities and astrophysical parameters of $>10^9$ stars of the Milky Way
  - First catalogue DR1 is out
    - ra, dec, G magnitude
  - DR2 ~1 year
  - Final catalogue ~2020



Copyright ESA/ATG medialab; background: ESO/S. Brunier



GAIA'S REACH

The Gaia spacecraft will use parallax and ultra-precise position measurements to obtain the distances and 'proper' (sideways) motions of stars throughout much of the Milky Way, seen here edge-on. Data from Gaia will shed light on the Galaxy's history, structure and dynamics.

Gaia will measure proper motions accurate to 1 kilometre per second for stars up to 20,000 parsecs away

Previous missions could measure stellar distances with an accuracy of 10% only up to 100 parsecs*

Sun

Galactic Centre

Gaia's limit for measuring distances with an accuracy of 10% will be 10,000 parsecs

*1 parsec = 3.26 light years

image:Devin Powell

Motivation

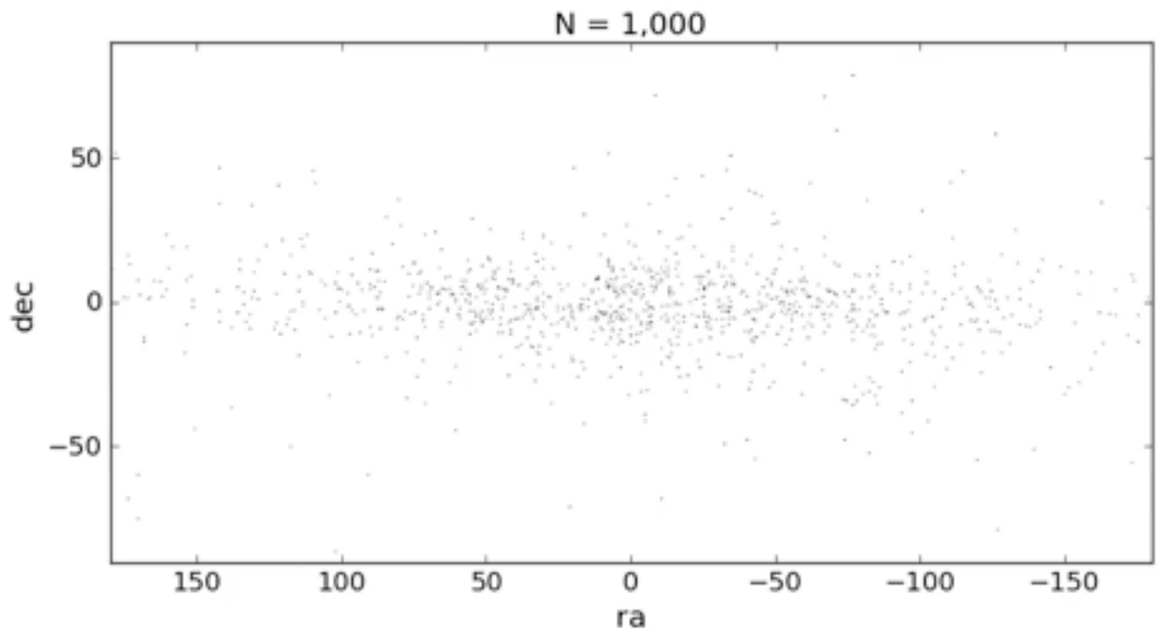
- We have Gaia DR1, soon DR2
  - > 10^9 objects/stars
- Can we visualise and explore this?
  - We want to ‘see’ the data
  - Data/Quality checks
  - Science: trends, relations, clustering
    - You are the (biological) neutral network

Motivation

- We have Gaia DR1, soon DR2
  - > 10^9 objects/stars
- Can we visualise and explore this?
  - We want to 'see' the data
  - Data/Quality checks
  - Science: trends, relations, clustering
    - You are the (biological) neutral network
- Problem
  - Scatter plots do not work well for 10^9 rows/objects (like Gaia)
  - Work with densities/statistics in 0,1,2 and 3d
  - Interactive?
    - Zoom, pan etc
    - Explore: selections/queries

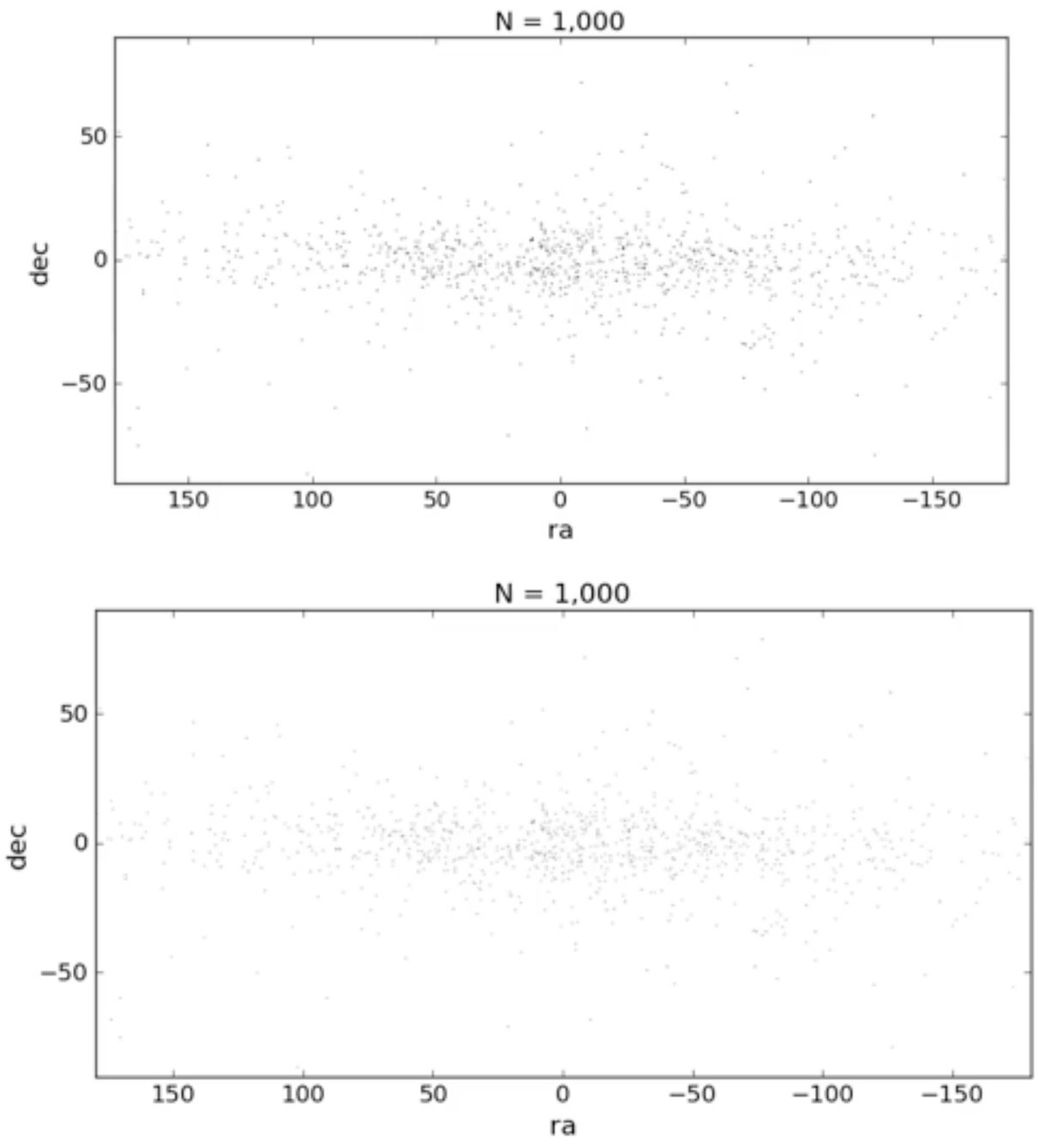
Motivation

- We have Gaia DR1, soon DR2
  - > 10^9 objects/stars
- Can we visualise and explore this?
  - We want to 'see' the data
  - Data/Quality checks
  - Science: trends, relations, clustering
    - You are the (biological) neutral network
- Problem
  - Scatter plots do not work well for 10^9 rows/objects (like Gaia)
  - Work with densities/statistics in 0,1,2 and 3d
  - Interactive?
    - Zoom, pan etc
    - Explore: selections/queries



Motivation

- We have Gaia DR1, soon DR2
  - > 10^9 objects/stars
- Can we visualise and explore this?
  - We want to 'see' the data
  - Data/Quality checks
  - Science: trends, relations, clustering
    - You are the (biological) neutral network
- Problem
  - Scatter plots do not work well for 10^9 rows/objects (like Gaia)
  - Work with densities/statistics in 0,1,2 and 3d
  - Interactive?
    - Zoom, pan etc
    - Explore: selections/queries



Situation

- TOPCAT comes close, not fast enough, works with individual rows/particles (written in Java)
- Your own IDL/Python code: a lot to consider to do it optimal (multicore, efficient storage, efficient algorithms, interactive becomes complex)
- DataShader: only visualisation of 2d and slower, no efficient transformation and selections
- We want something to visualize 10^9 rows/objects in ~1 second
- Do we need to resort general Big Data solutions, Spark, Hadoop?

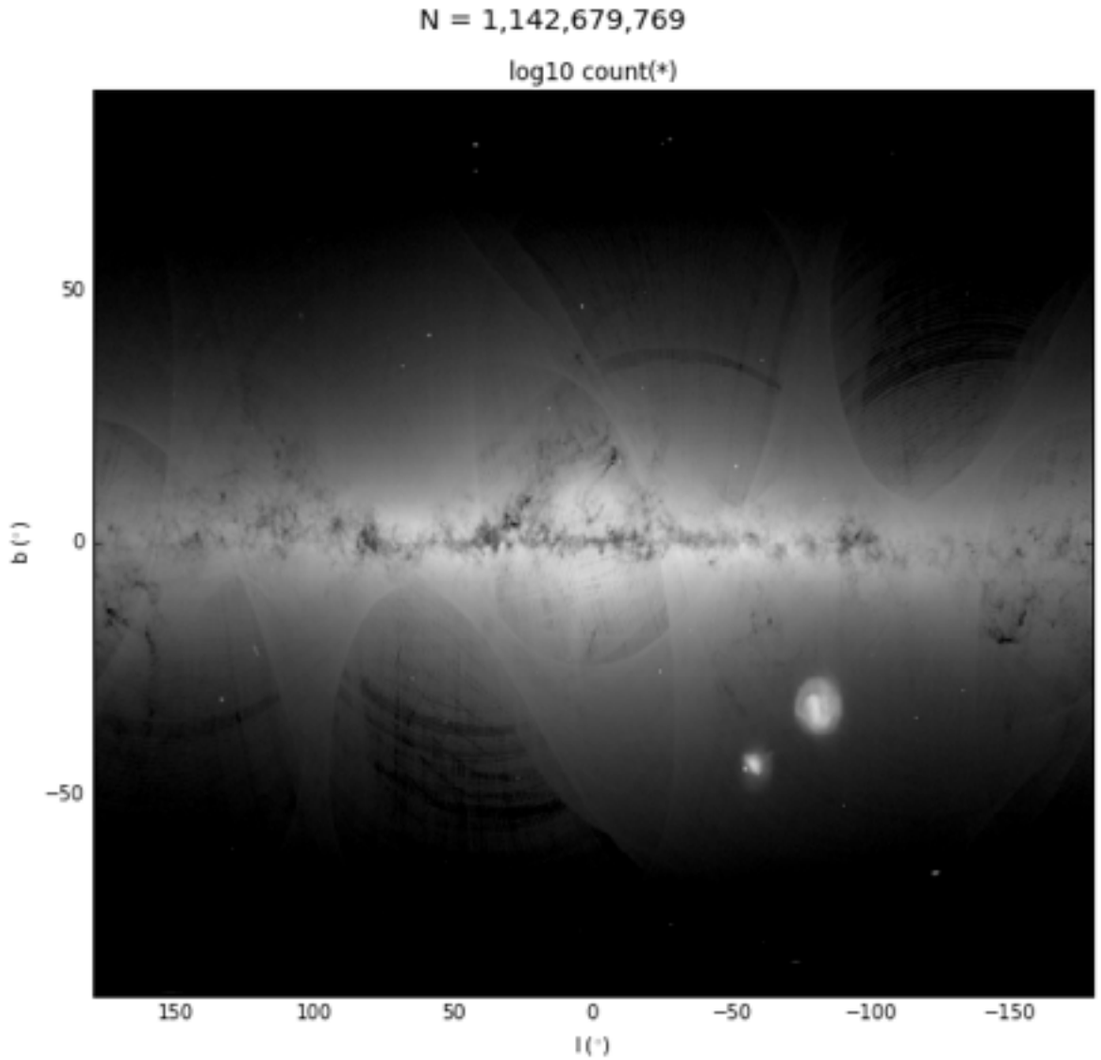
How fast can it be processed?

- What can be done?
  - 10^9 * 2 * 8 bytes = 15 GiB (double is 8 bytes)
  - Memory bandwidth: 10-20 GiB/s: ~1 second
  - CPU: 3 Ghz (but multicore, say 4): 12 cycles/ second
    - Few cycles per row/object, simple algorithm
      - Histograms/Density grids
- Yes, but
  - If it fits/cached in memory, otherwise ssd/hdd speeds (10-100 seconds)
  - proper storage and reading of data
  - simple and fast algorithm for binning

How fast can it be processed?

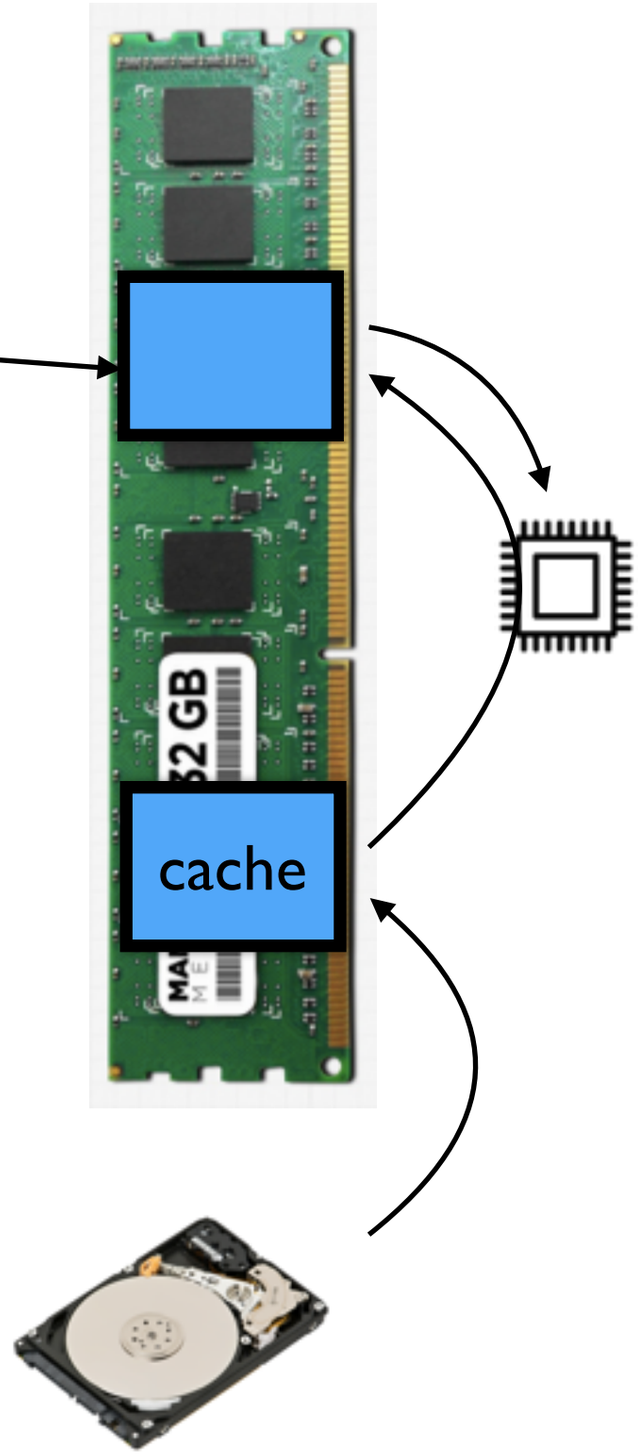
- What can be done?
  - 10^9 * 2 * 8 bytes = 15 GiB (double is 8 bytes)
  - Memory bandwidth: 10-20 GiB/s: ~1 second
  - CPU: 3 Ghz (but multicore, say 4): 12 cycles/second
  - Few cycles per row/object, simple algorithm
    - Histograms/Density grids
- Yes, but
  - If it fits/cached in memory, otherwise ssd/hdd speeds (10-100 seconds)
  - proper storage and reading of data
  - simple and fast algorithm for binning

•~1 second



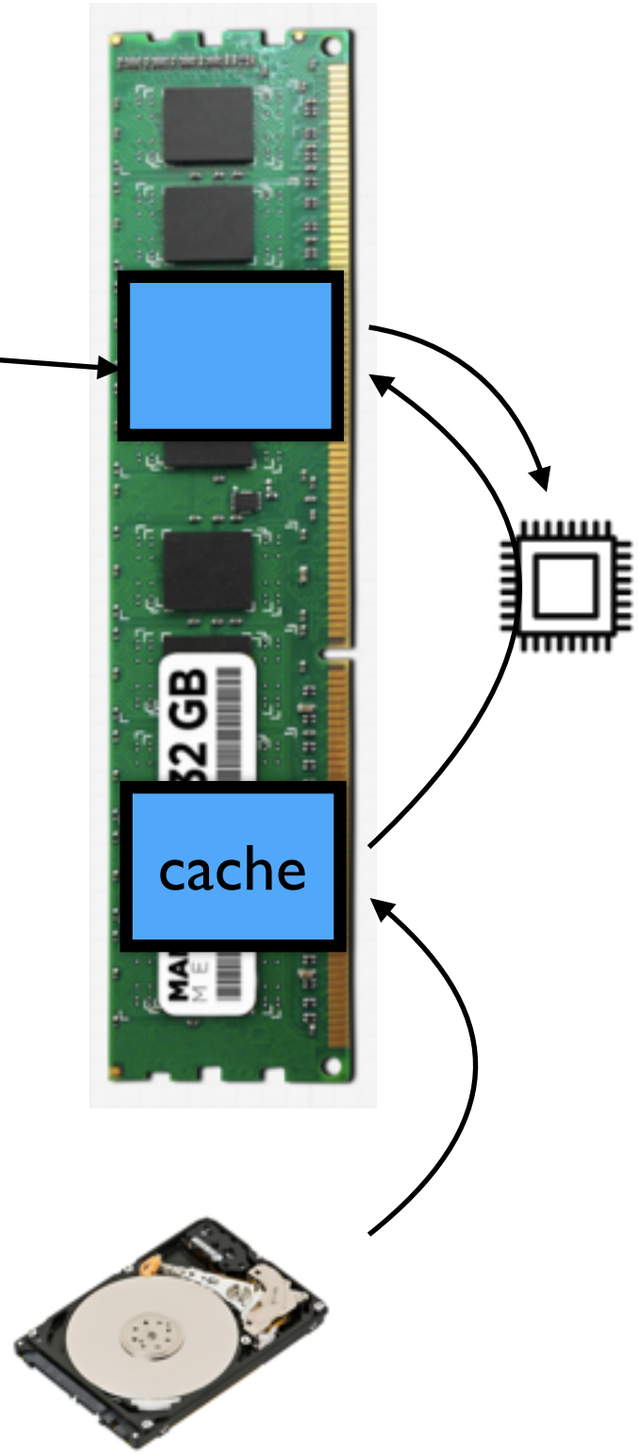
How to store and read the data

- Storage: native, column based (hdf5, fits)
- Normal (POSIX read) method:
  - Allocate memory
  - read from disk to memory
  - Actually: from disk, to OS cache, to memory (if unbuffered, otherwise another copy)
  - Wastes memory (actually disk cache)
  - 15 GB data, requires 30 GB is you want to use the file system cache



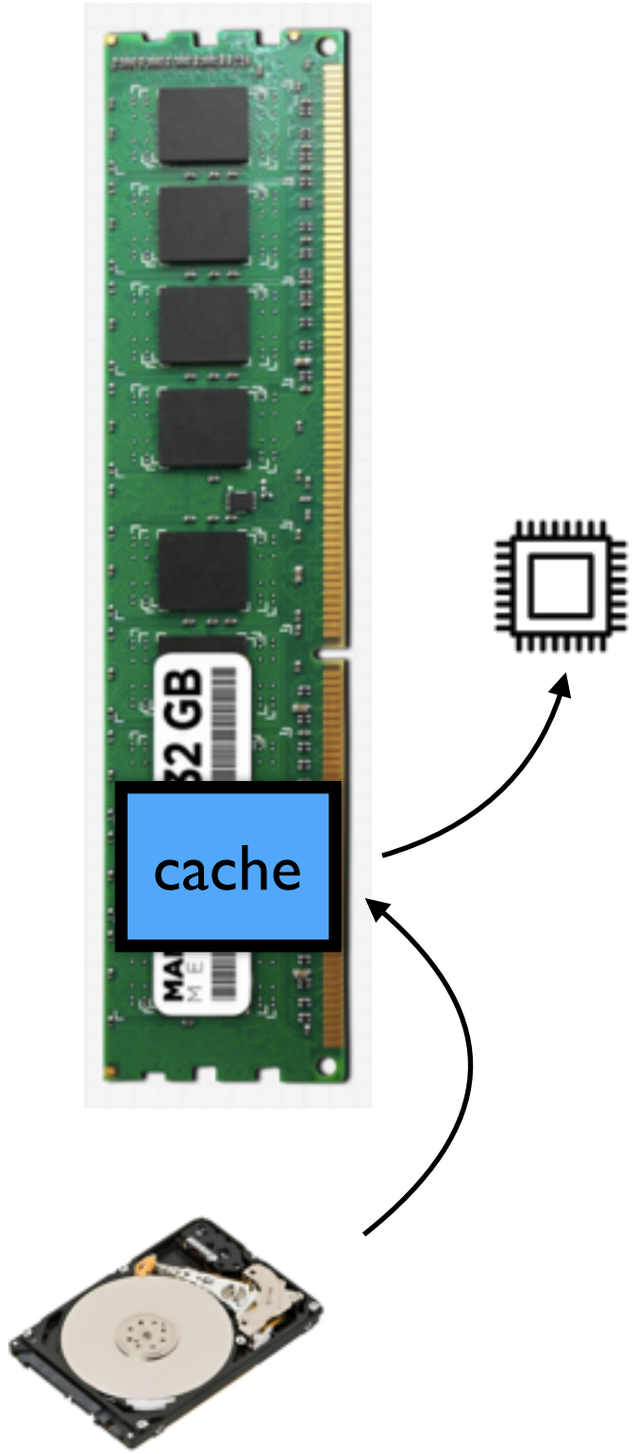
How to store and read the data

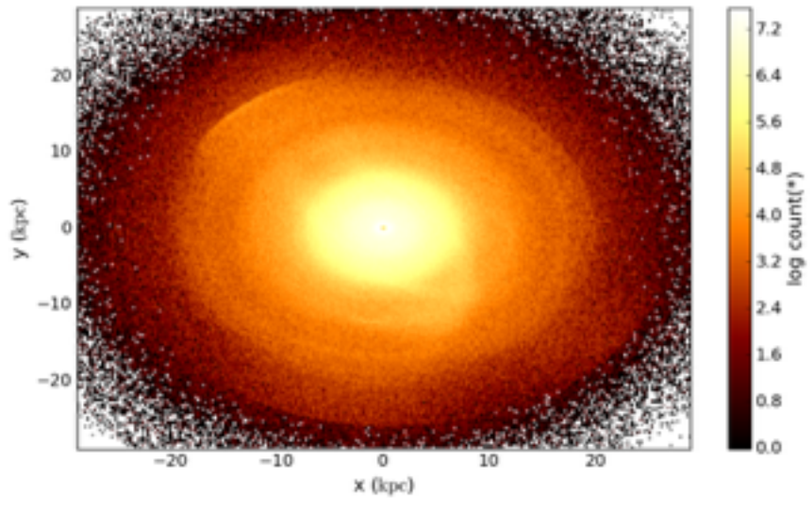
- Storage: native, column based (hdf5, fits)
- Normal (POSIX read) method:
  - Allocate memory
  - read from disk to memory
  - Actually: from disk, to OS cache, to memory (if unbuffered, otherwise another copy)
  - Wastes memory (actually disk cache)
    - 15 GB data, requires 30 GB is you want to use the file system cache
- Memory mapping:
  - get direct access to OS memory cache, no copy, no setup (apart from the kernel doing setting up the pages)
  - avoid memory copies, more cache available
- In previous example:
  - copying 15 GB will take about 0.5-1.0 second, at 10-20 GB/s
  - Can be 2-3x slower (cpu cache helps a bit)



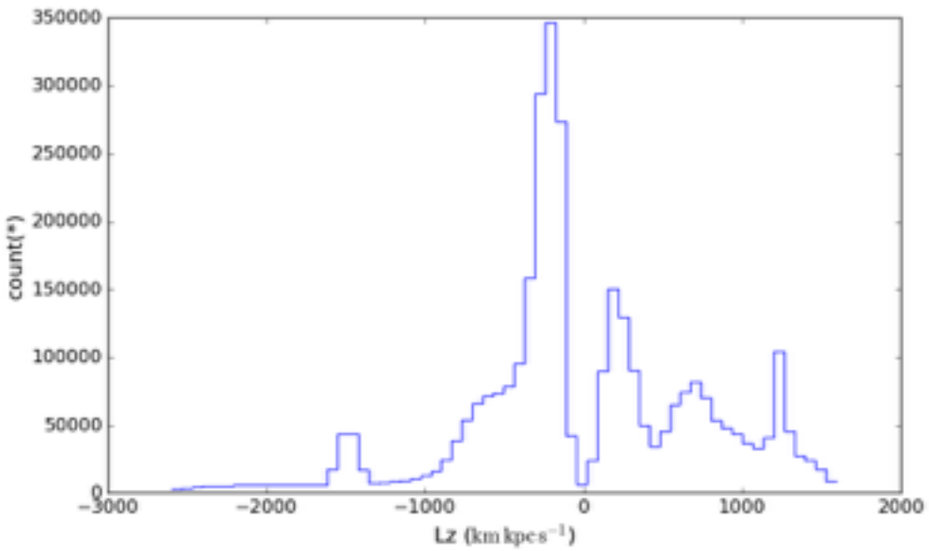
How to store and read the data

- Storage: native, column based (hdf5, fits)
- Normal (POSIX read) method:
  - Allocate memory
  - read from disk to memory
  - Actually: from disk, to OS cache, to memory (if unbuffered, otherwise another copy)
  - Wastes memory (actually disk cache)
    - 15 GB data, requires 30 GB is you want to use the file system cache
- Memory mapping:
  - get direct access to OS memory cache, no copy, no setup (apart from the kernel doing setting up the pages)
  - avoid memory copies, more cache available
- In previous example:
  - copying 15 GB will take about 0.5-1.0 second, at 10-20 GB/s
  - Can be 2-3x slower (cpu cache helps a bit)

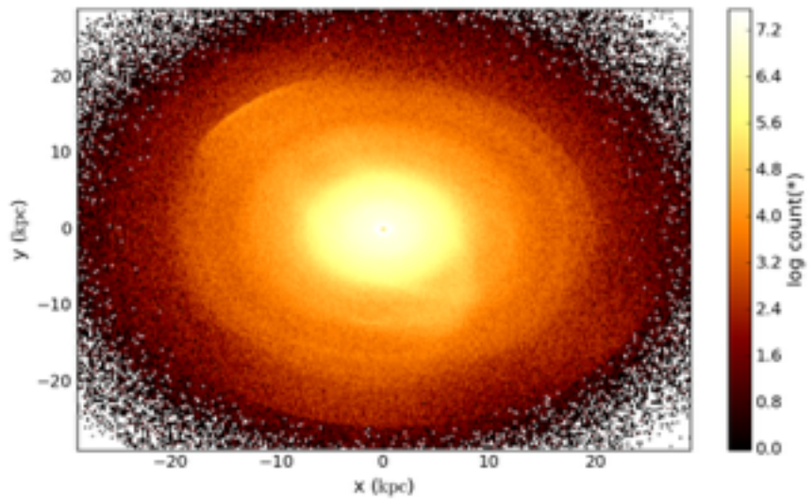



20
10
0
−10
−20
y (kpc)
−20
−10
0
10
20
x (kpc)
7.2
6.4
5.6
4.8
4.0
3.2
2.4
1.6
0.8
0.0
log count(*)

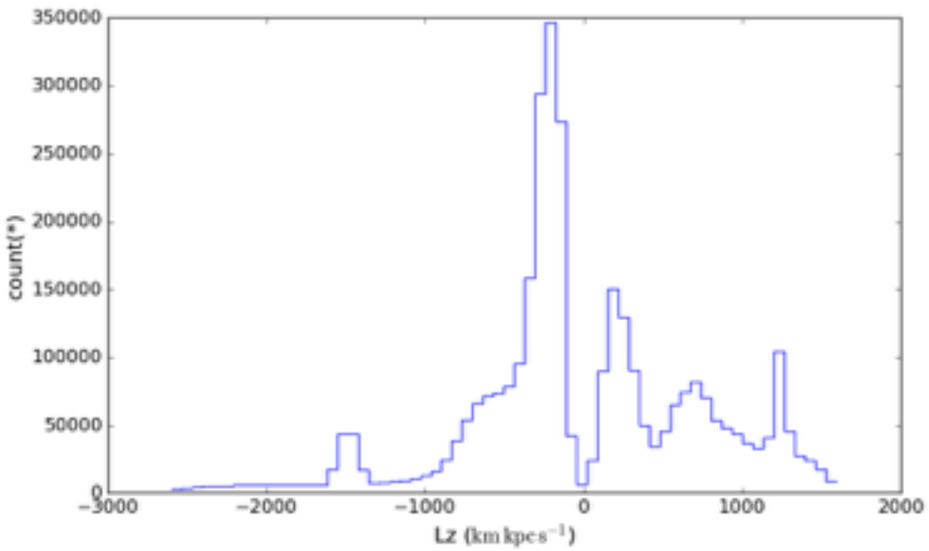
1d


350000
300000
250000
200000
150000
100000
50000
0
count(*)
−3000
−2000
−1000
0
1000
2000
Lz (km kpc s^{-1})

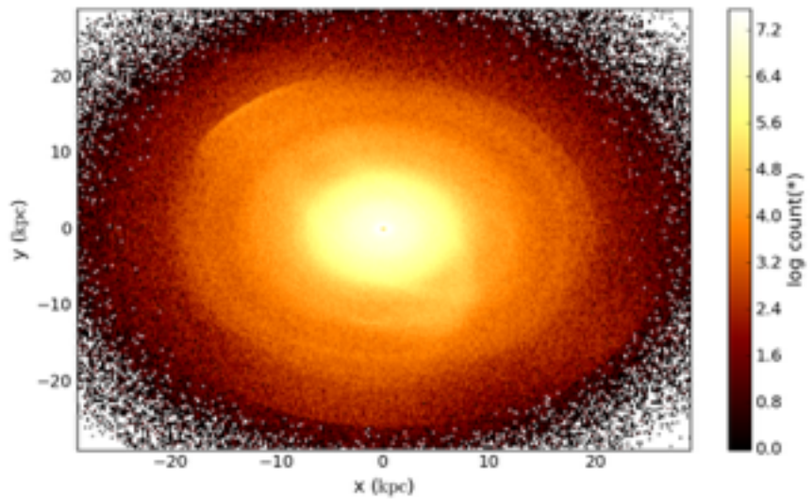
2d


20
10
0
−10
−20
y (kpc)
−20
−10
0
10
20
x (kpc)
7.2
6.4
5.6
4.8
4.0
3.2
2.4
1.6
0.8
0.0
log count(*)

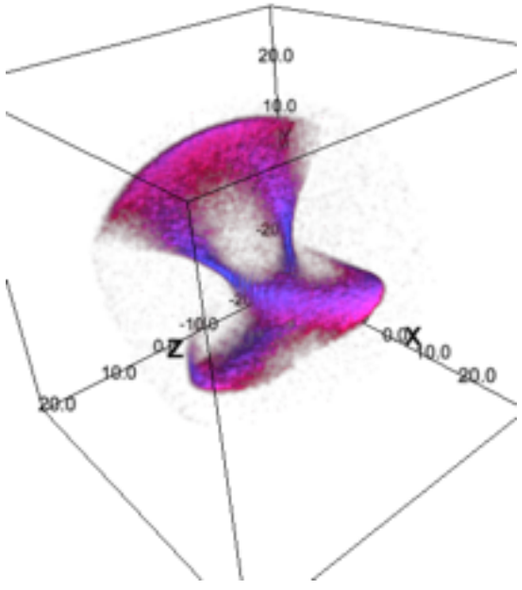
1d


350000
300000
250000
200000
150000
100000
50000
0
count(*)
−3000
−2000
−1000
0
1000
2000
Lz (km kpc s^{-1})

2d

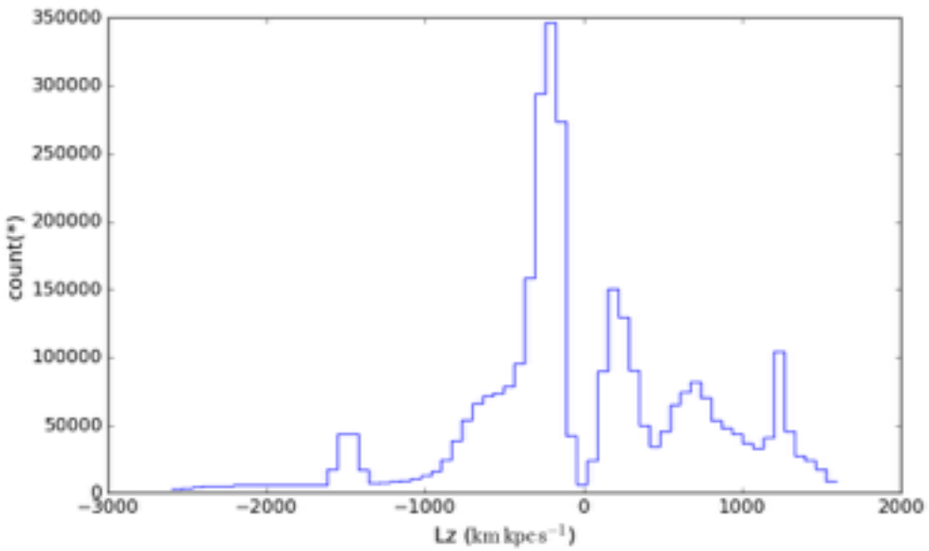

20
10
0
−10
−20
y (kpc)
−20
−10
0
10
20
x (kpc)
7.2
6.4
5.6
4.8
4.0
3.2
2.4
1.6
0.8
0.0
log count(*)

3d

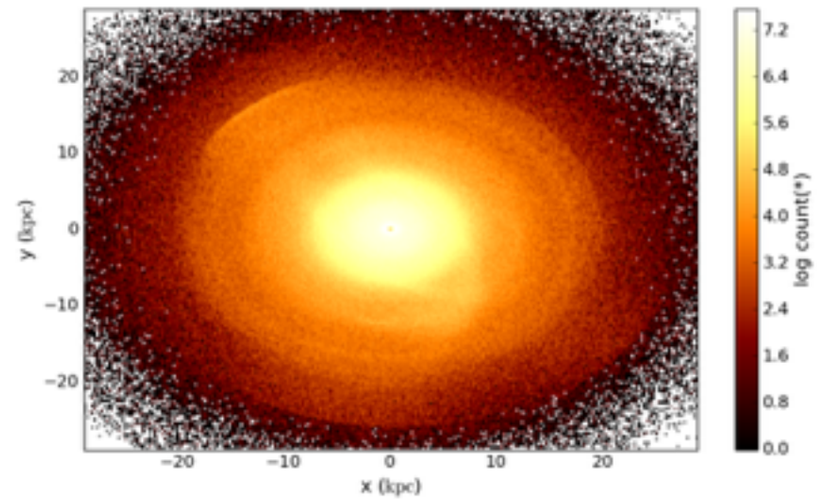

20.0
10.0
-20
-10.0
0.0
Z
10.0
20.0
0.0
X
10.0
20.0

0d 330,000 rows

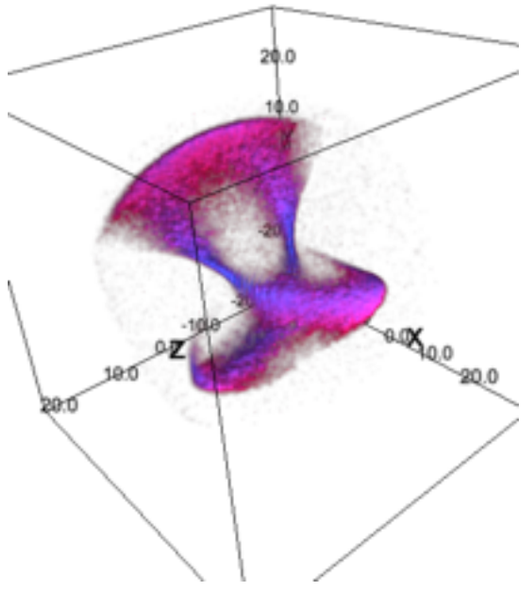
1d



2d

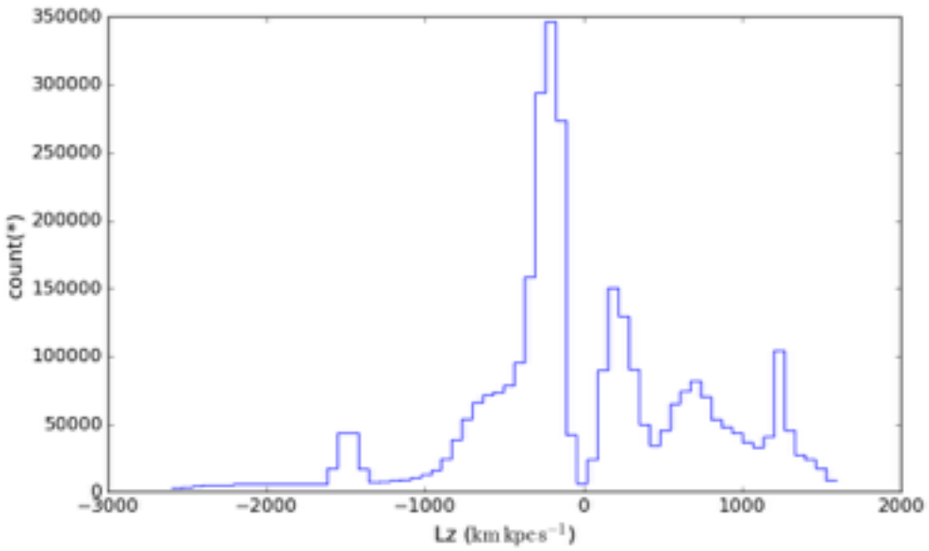


3d

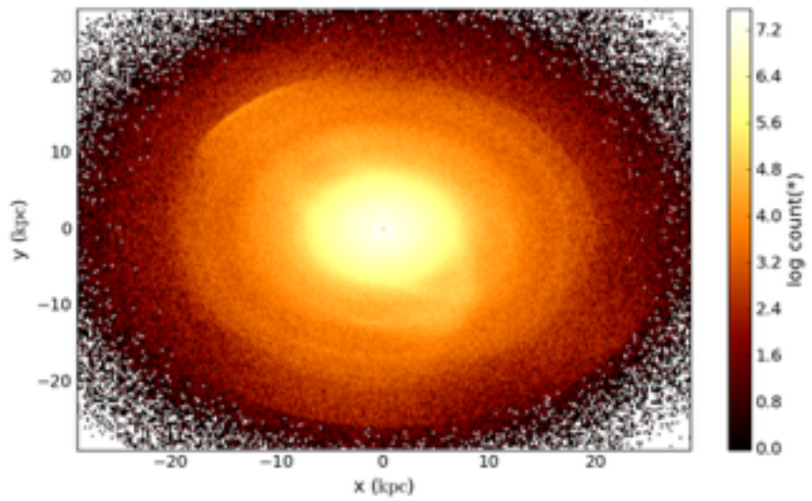


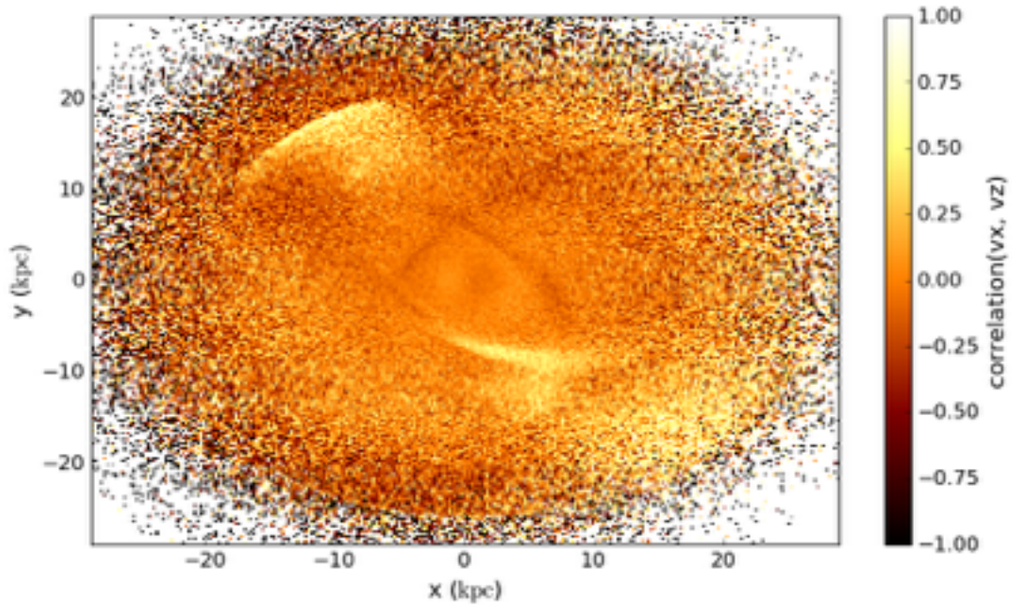
0d 330,000 rows

1d

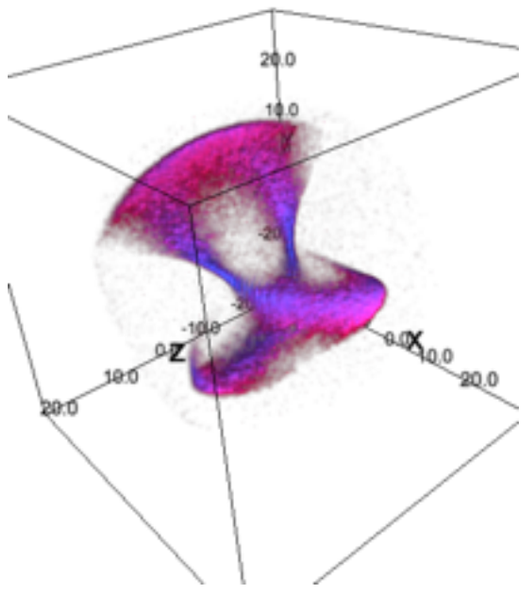


2d



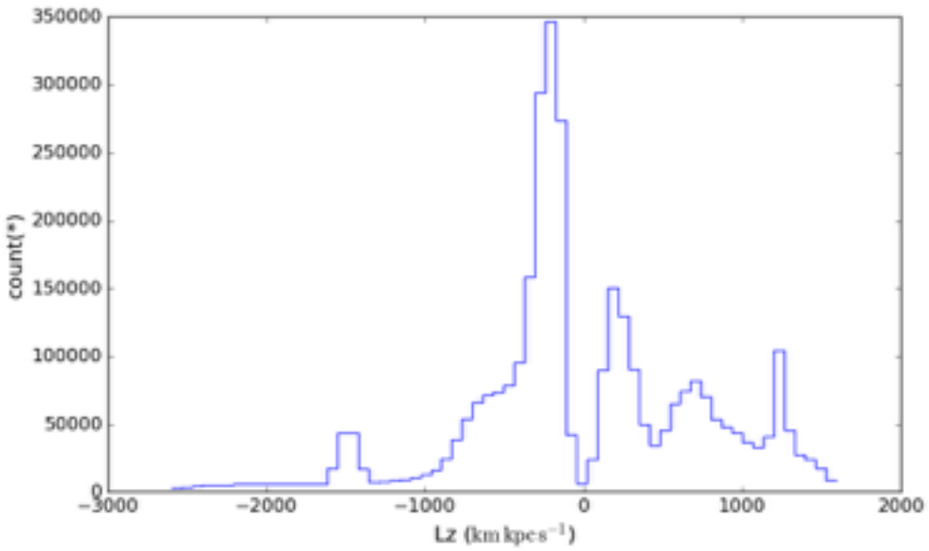


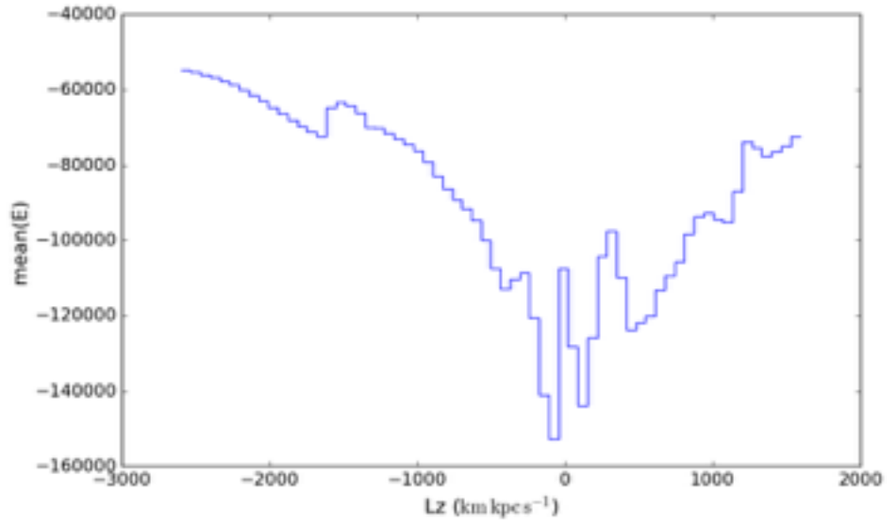
3d



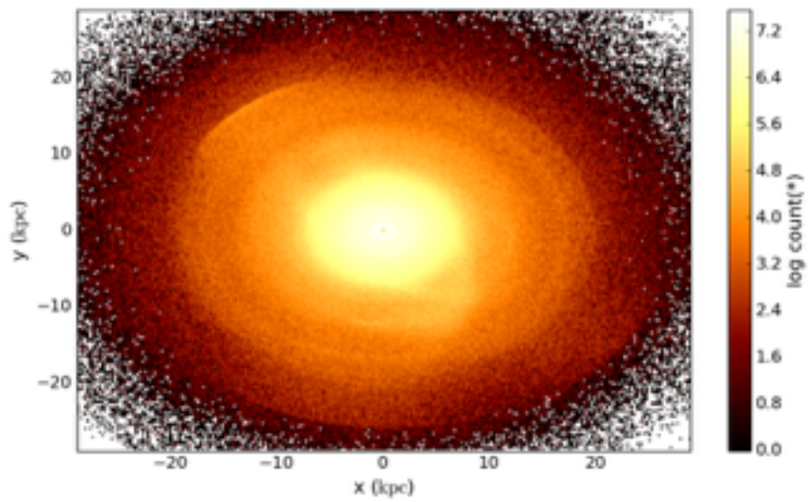
0d 330,000 rows

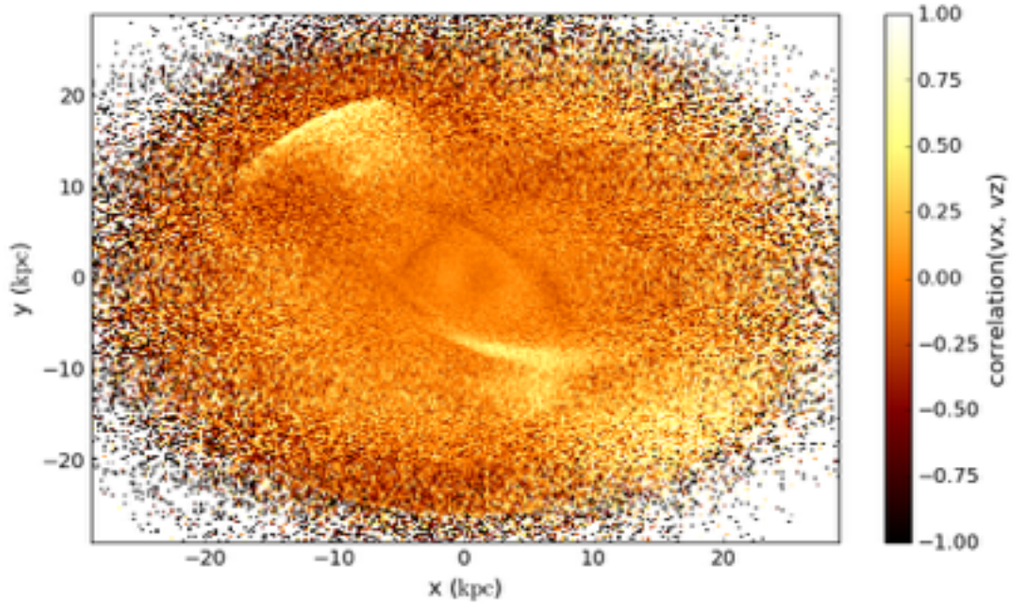
1d



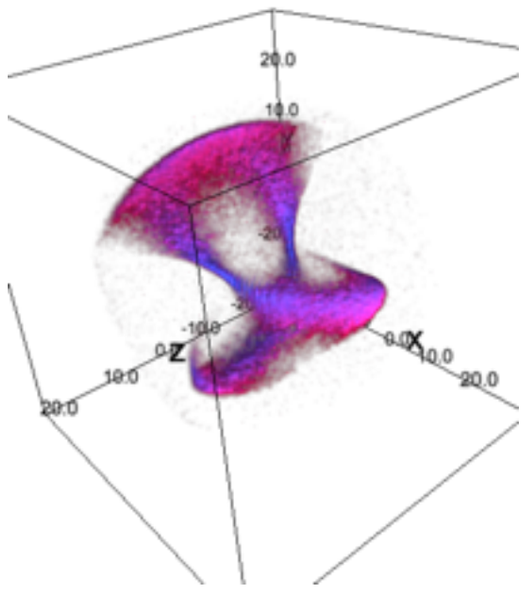


2d





3d

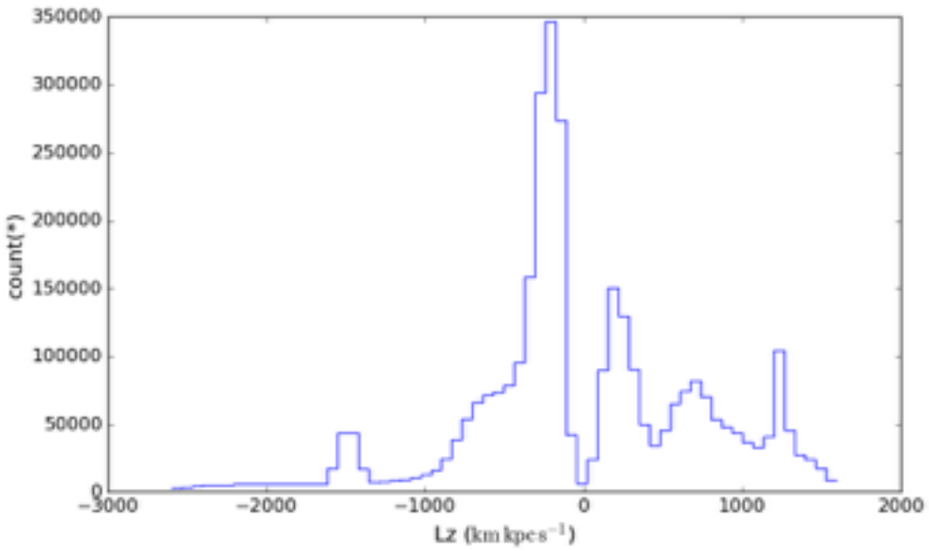


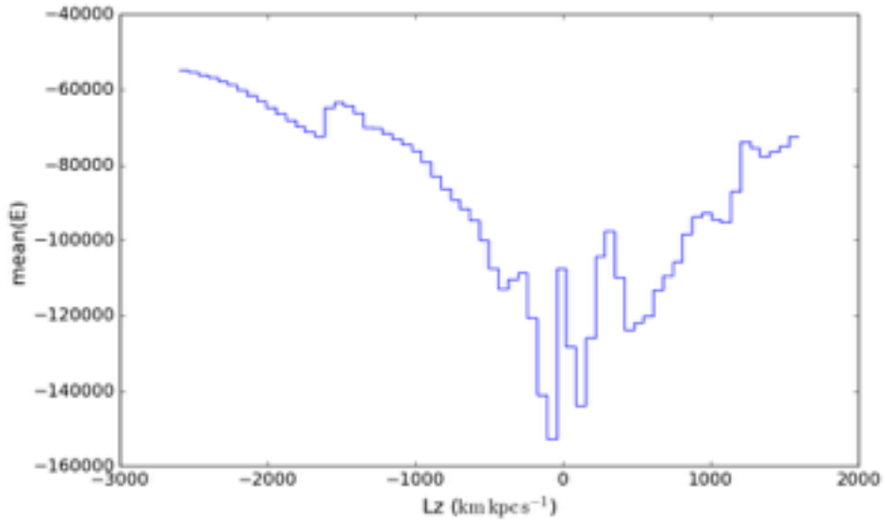
0d

330,000 rows

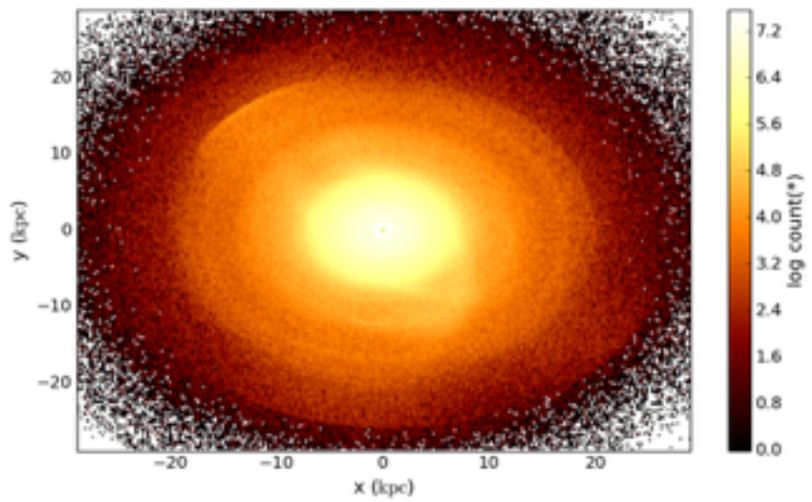
mean: -0.083

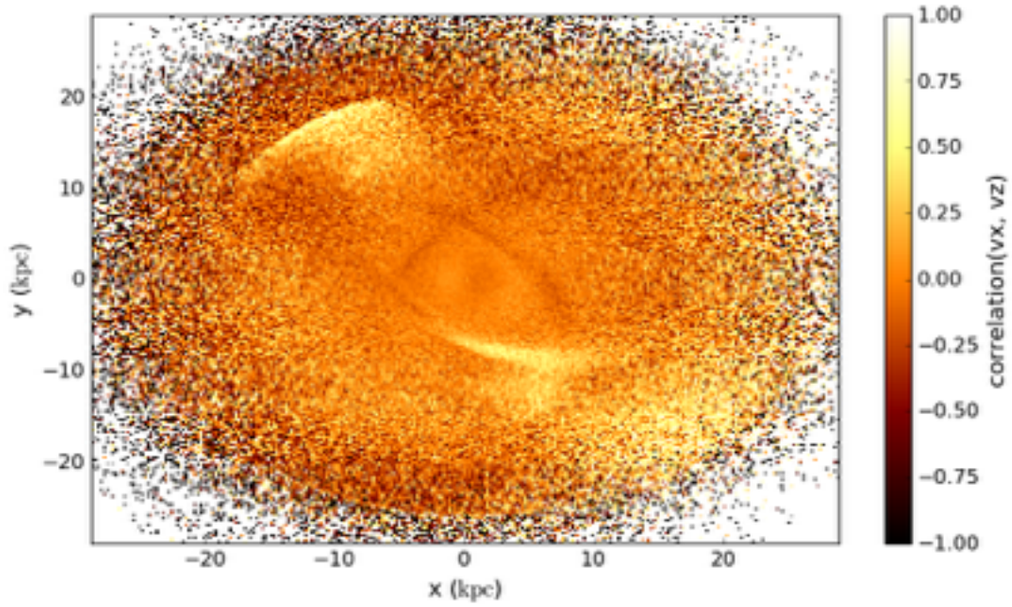
1d



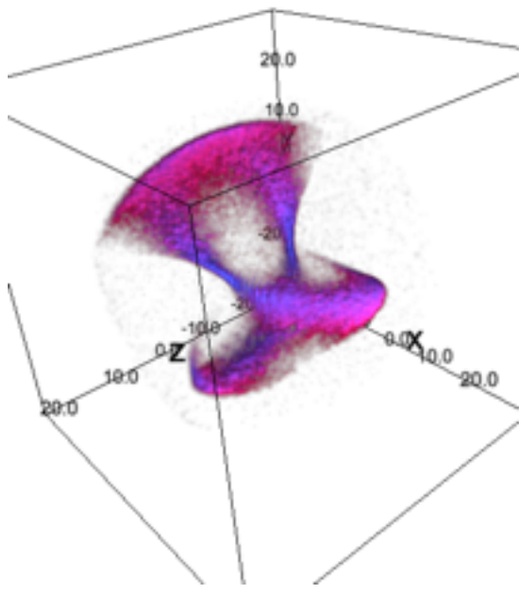


2d

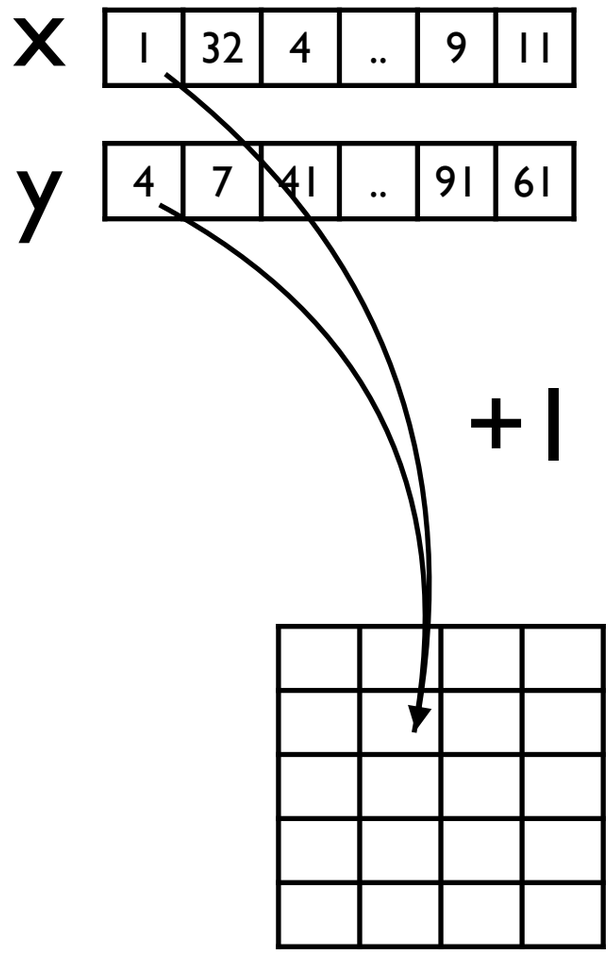




3d

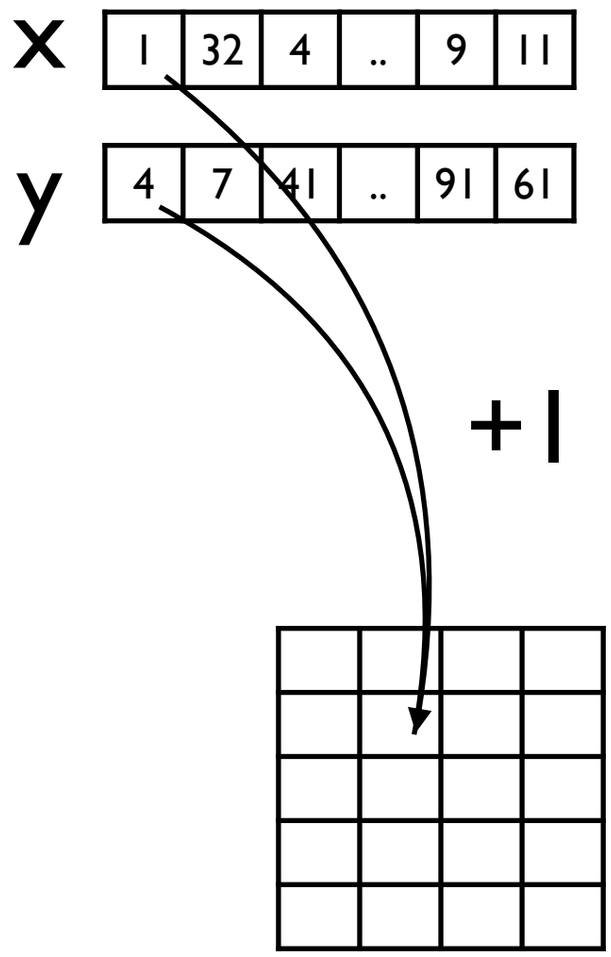


vaex: statistics in N-d



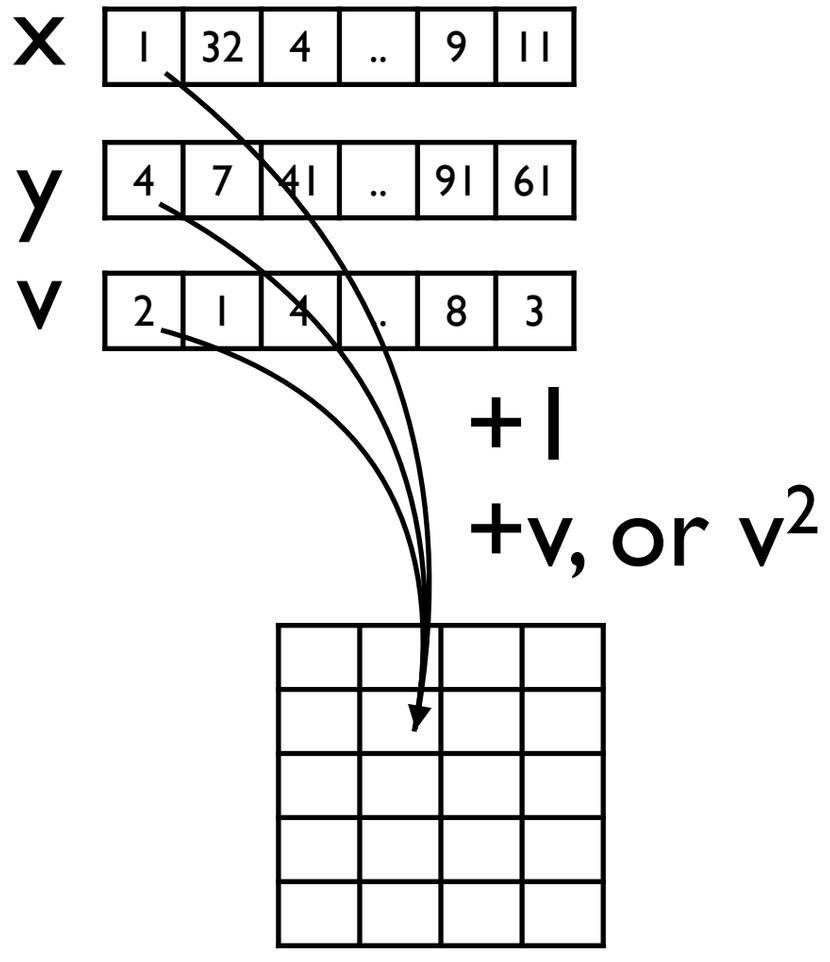
vaex: statistics in N-d

- count
- sum values
- min
- max
- moments



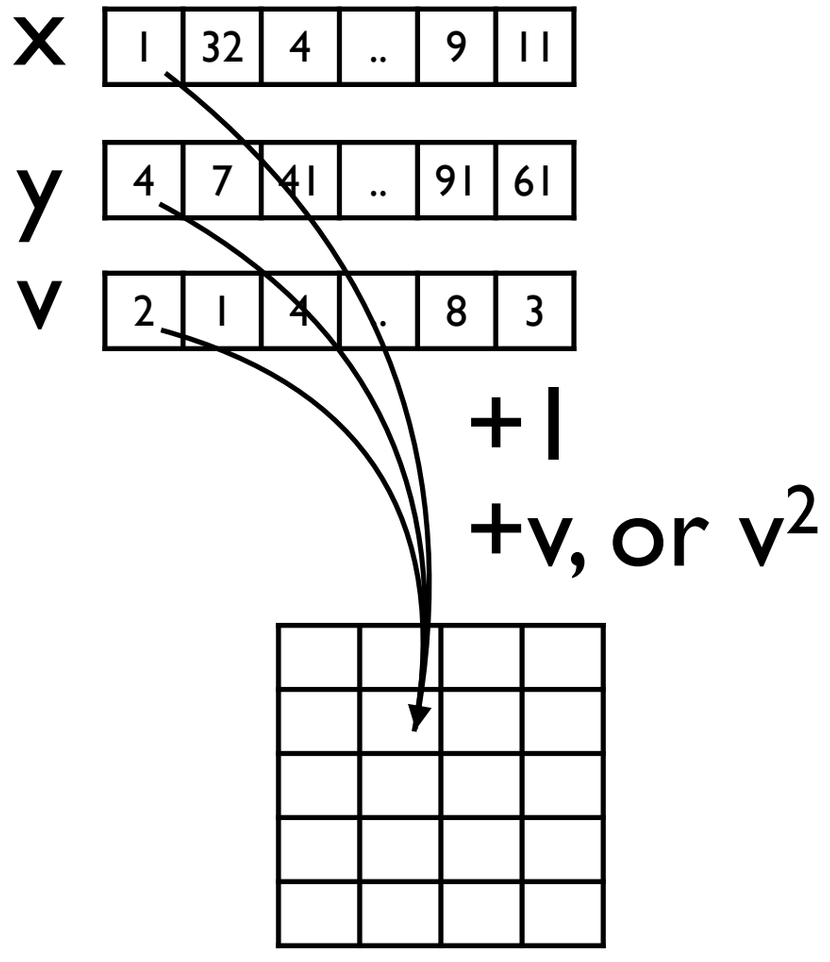
vaex: statistics in N-d

- count
- sum values
- min
- max
- moments



vaex: statistics in N-d

- count
- sum values
- min
- max
- moments

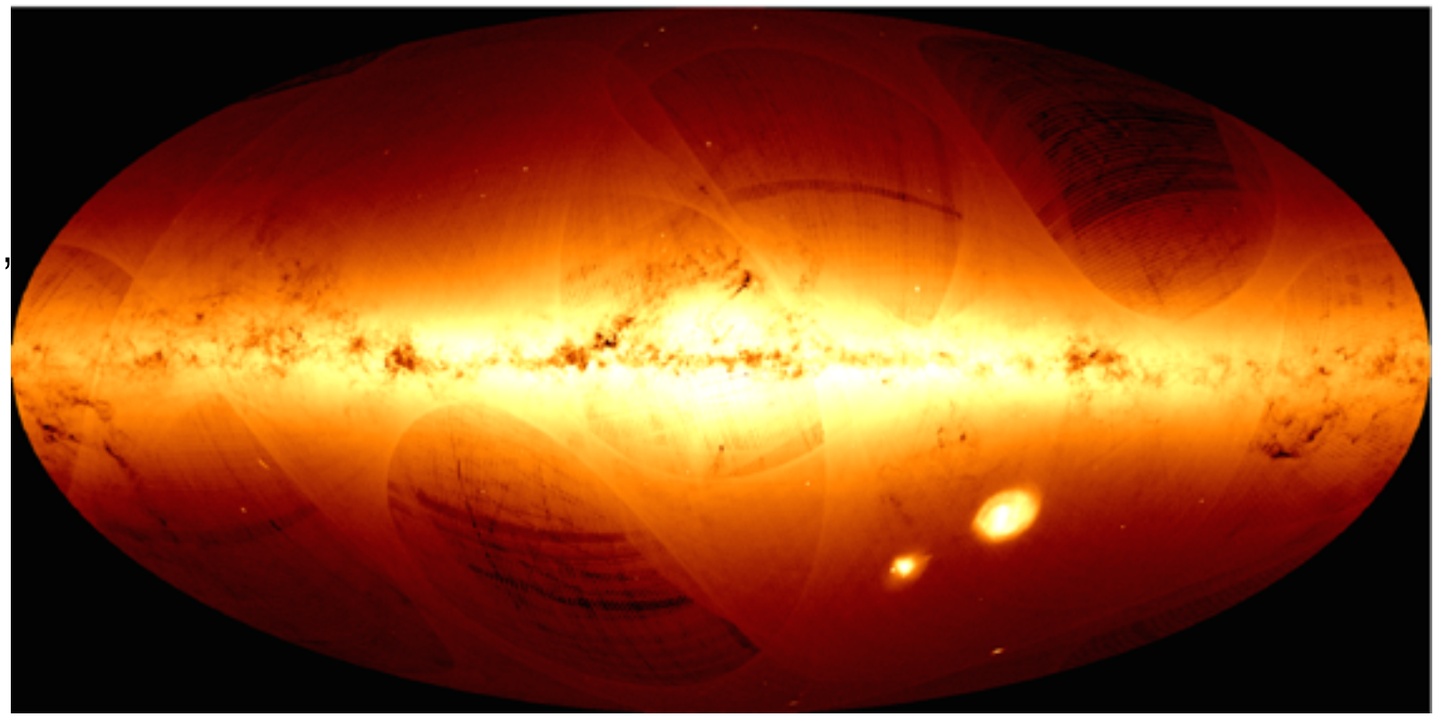


- Possibilities
  - Total: flux, mass
  - Mean: velocity, metallicity
  - Dispersions: velocity…
  - Correlation
- Statistics on a (N dim) grid
  - (And visualize them)

For what?

- Astronomical catalogues (Gaia, SDSS, PANSTARRS, LSST, …)
    - columns: observable (or derived)
    - rows: stars/galaxies/…
- N-body simulations
    - columns: x,y,z,vx,vy,vz
        - SPH: density, temperature, metallicity…
    - rows: particle
- Any tabular data
    - rows and columns

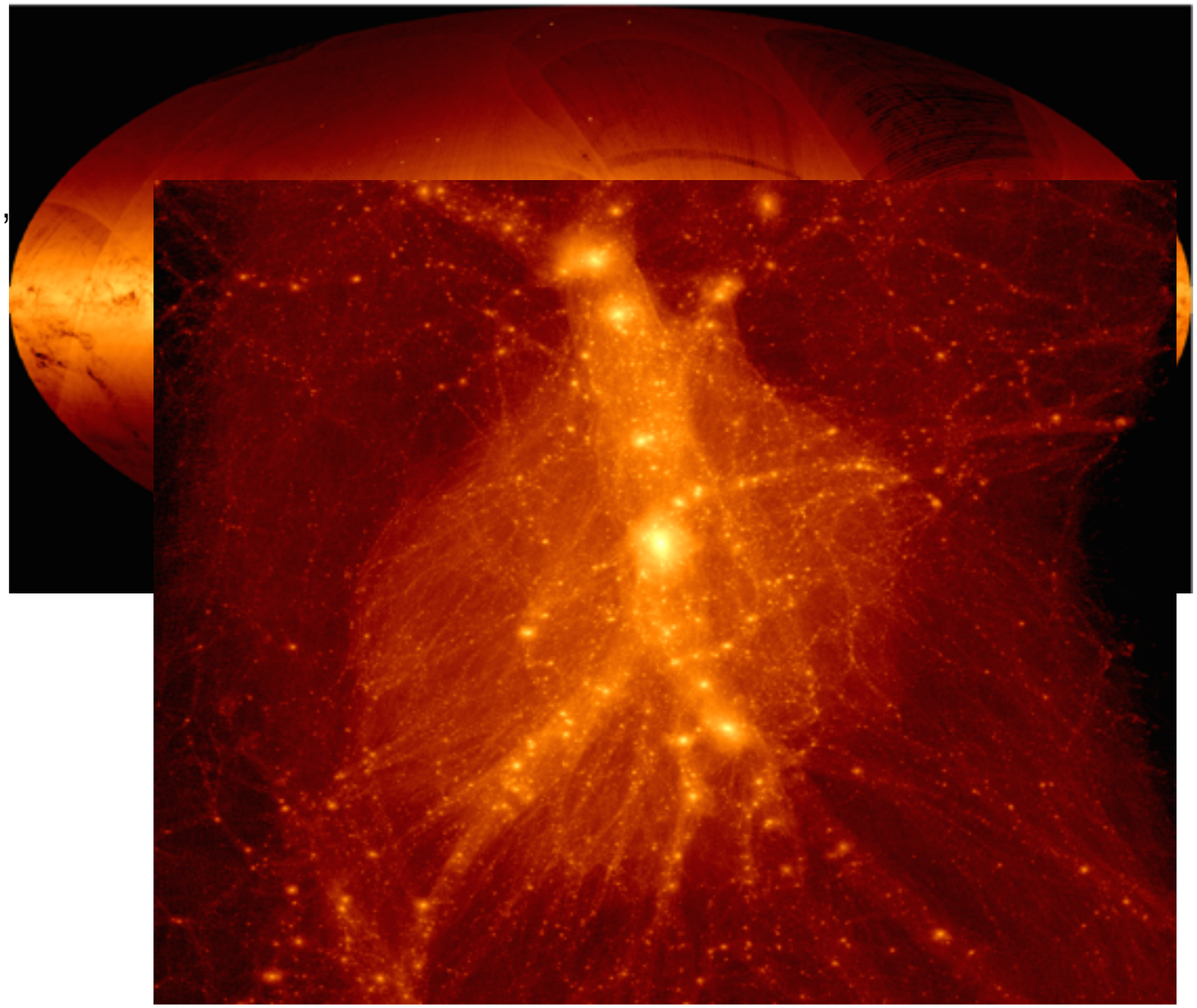
For what?



- Astronomical catalogues (Gaia, SDSS, PANSTARRS, LSST, …)
    - columns: observable (or derived)
    - rows: stars/galaxies/…
- N-body simulations
    - columns: x,y,z,vx,vy,vz
        - SPH: density, temperature, metallicity…
    - rows: particle
- Any tabular data
    - rows and columns

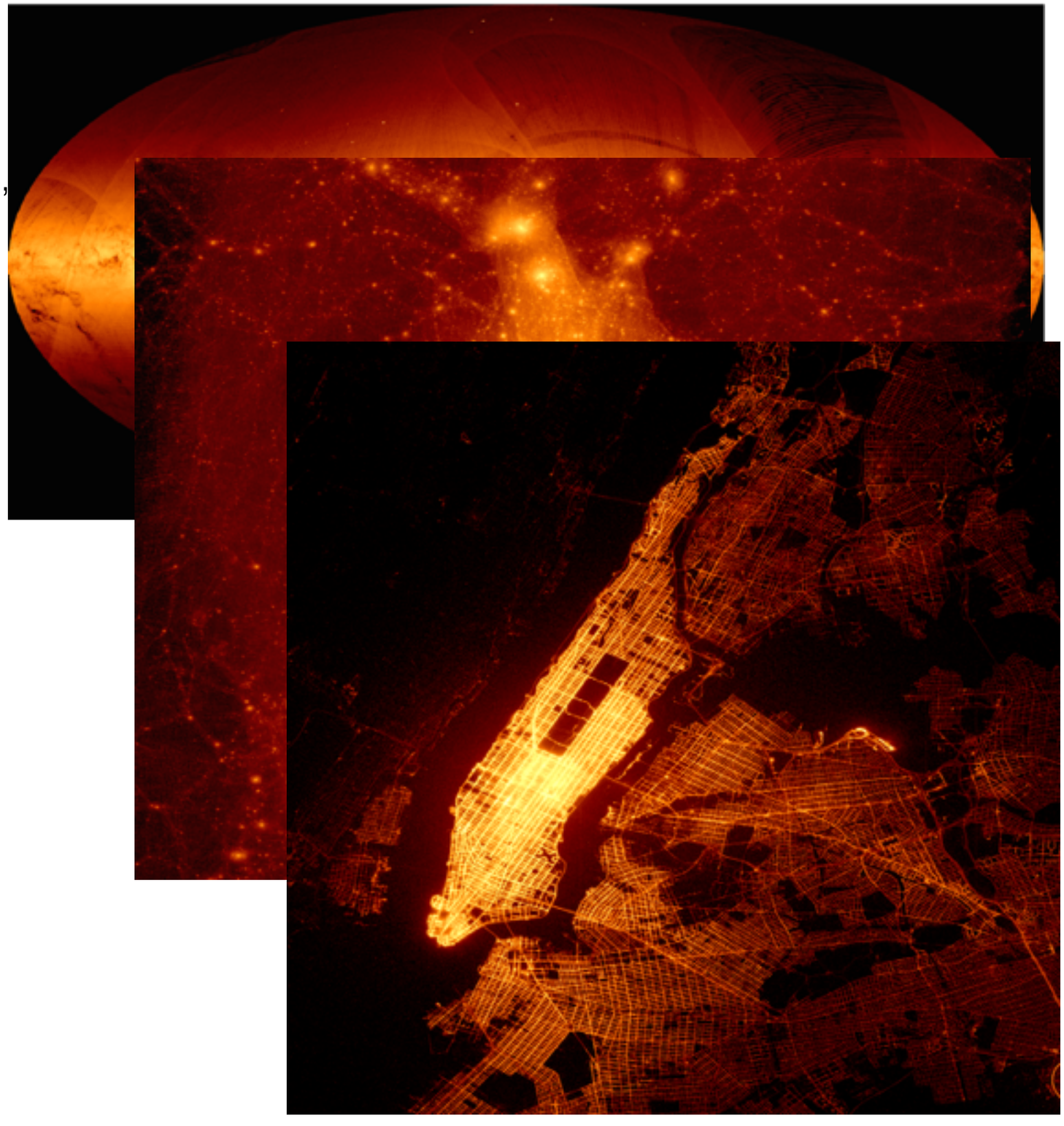
For what?

- Astronomical catalogues (Gaia, SDSS, PANSTARRS, LSST, …)
  - columns: observable (or derived)
  - rows: stars/galaxies/…
- N-body simulations
  - columns: x,y,z,vx,vy,vz
    - SPH: density, temperature, metallicity…
  - rows: particle
- Any tabular data
  - rows and columns



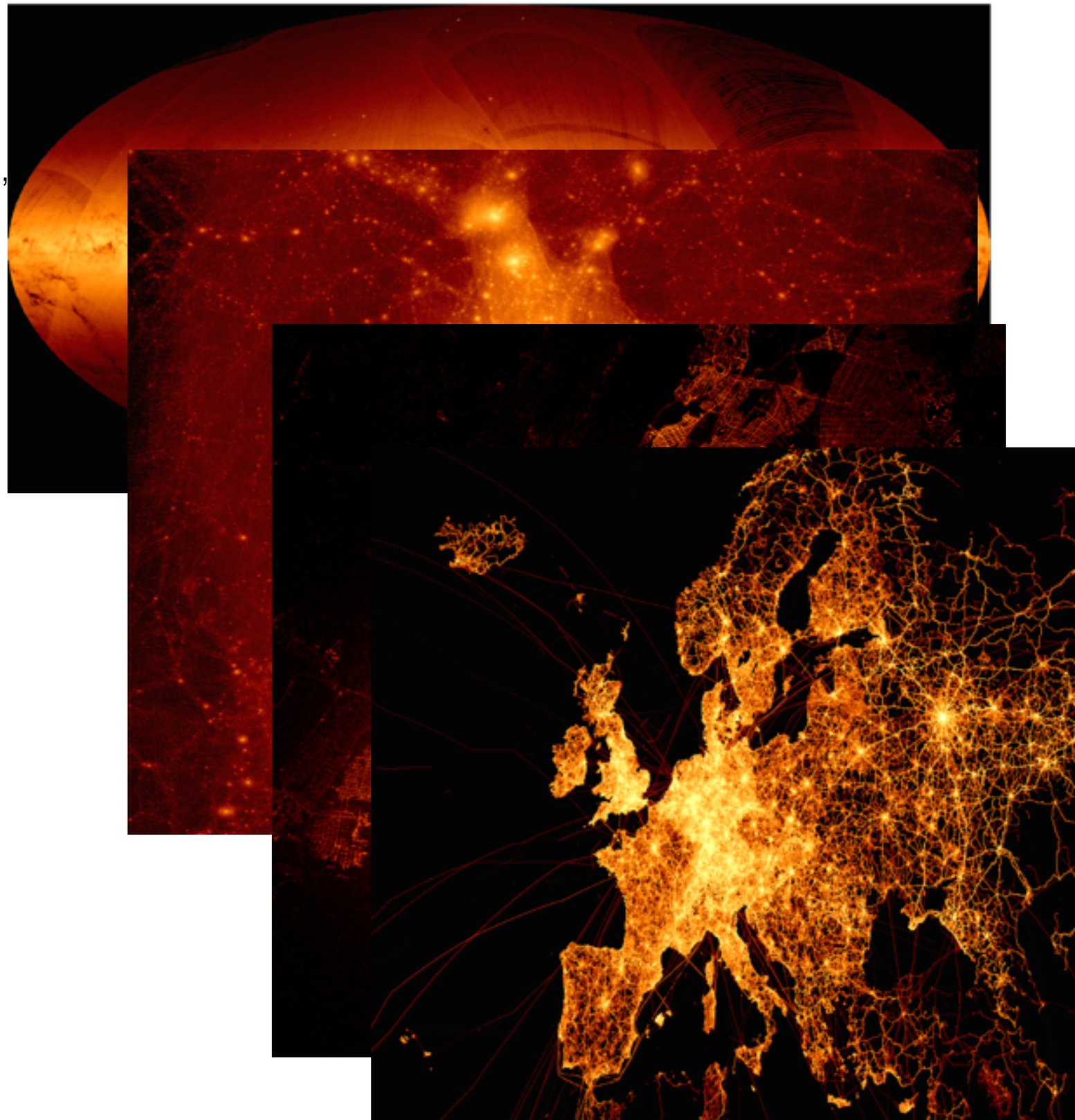
For what?

- Astronomical catalogues (Gaia, SDSS, PANSTARRS, LSST, …)
  - columns: observable (or derived)
  - rows: stars/galaxies/…
- N-body simulations
  - columns: x,y,z,vx,vy,vz
    - SPH: density, temperature, metallicity…
  - rows: particle
- Any tabular data
  - rows and columns



For what?

- Astronomical catalogues (Gaia, SDSS, PANSTARRS, LSST, …)
  - columns: observable (or derived)
  - rows: stars/galaxies/…
- N-body simulations
  - columns: x,y,z,vx,vy,vz
    - SPH: density, temperature, metallicity…
  - rows: particle
- Any tabular data
  - rows and columns



What about?

- Transformations of the data
  - You don’t always want to work with columns as given
  - Efficient mathematical transformation
    - Not ‘z = x + y’ (8GB of RAM wasted)
    - vaex
      - Everything is an expression: `dataset.mean('x + y')`
      - Computed in chunks not to waste RAM
      - Virtual columns: `r = sqrt(x**2+y**2)`, `dataset.mean('r')`

What about?

- Subsets
  - You don’t use all data
  - Efficient subsets (do NOT copy the data)
- vaex
  - Selections result in a boolean mask (1 per row): `dataset.select('x > 0', name='xpos')`
  - Use in any statistic: `dataset.mean('x + y', selection='xpos')`

Vaex: **V**isualization **A**nd **EX**ploration

- A library
  - python package
  - ‘import vaex’
- reading of data
- multithreading
- statistics/binning (0,1,2,3, Nd)
- selections/queries
- visualization
  - matplotlib
  - bqplot/ipyvolume/ipyleaflet
- server/client
- integrates with IPython notebook
- open source / MIT License
- www.github.com/maartenbreddels/vaex

Vaex: Visualization And EXploration

- A library
  - python package
  - ‘import vaex’
- reading of data
- multithreading
- statistics/binning (0,1,2,3, Nd)
- selections/queries
- visualization
  - matplotlib
  - bqplot/ipyvolume/ipyleaflet
- server/client
- integrates with IPython notebook
- open source / MIT License
- www.github.com/maartenbreddels/vaex

- A GUI program
- Gives interactive navigation, zoom, pan
- interactive selection (lasso, rectangle)
- client
- undo/redo
- Standalone binary
  - http://vaex.astro.rug.nl/

Demo program

- Basics (Helmi de Zeeuw 2000)
- Full Gaia DR1

- Laptop:
  - Macbook Air 13”, 8BG ram, ssd
- Server (gaia):
  - 2x8 cores (32 hyperthreading)
  - 256 GB RAM
  - 24 RAID
  - ~12 kEUR

Demo library