goto;

GOTO
Copenhagen 2021

UNCLUSTER YOUR DATA SCIENCE USING VAEX

#GOTOcph

Who are we?

Maarten Breddels
Former astrophysicist
Freelancer / consultant / data scientist
Core Jupyter-Widgets developer
Founder of vaex.io
Principal author of vaex
maartenbreddels@gmail.com
www.maartenbreddels.com
@maartenbreddels
github.com/maartenbreddels

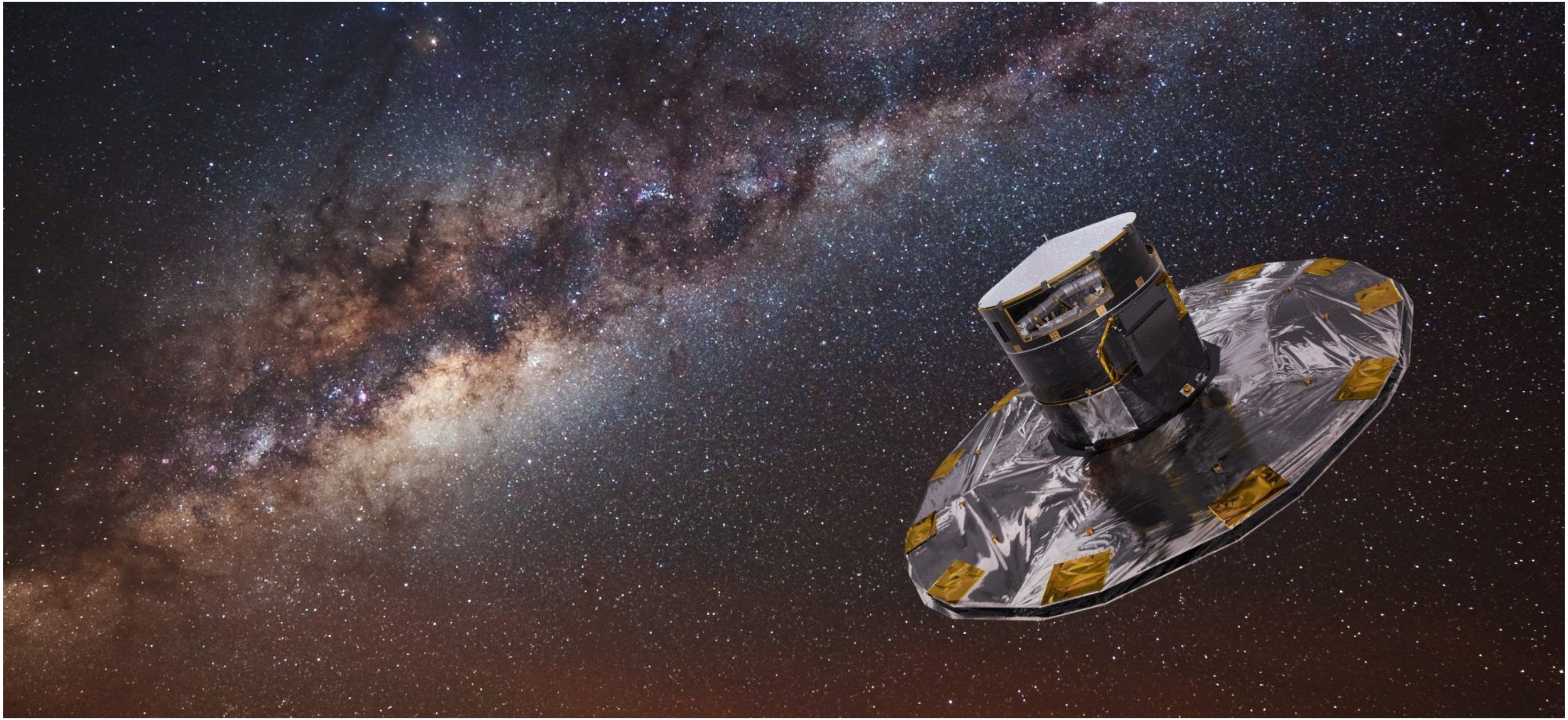
Jovan Veljanoski
Former astrophysicist
Sr. Data Scientist @ Tiqets
Co-founder of vaex.io
jovan.veljanoski@gmail.com
https://www.linkedin.com/in/jovanvel/

What people think we do

What we actually do

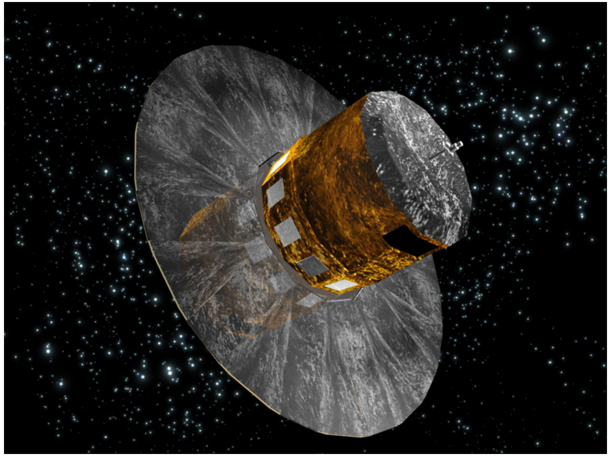
Number HIP		Descriptor: epoch J1991.25 RA (h m s)	Dec (±° ′ ″)	V (mag)			Position: epoch J1991.25 (ICRS) α (deg)	δ (deg)		Par. π (mas)	Proper Motion $\mu_{\alpha*}$ (mas/yr)	μ_δ (mas/yr)	Standard Errors α* (mas)	δ (mas)	π (mas)	$\mu_{\alpha*}$ (mas/yr)	μ_δ (mas/yr)	Astrometric Correlations (%) δ/α*	π/α*	π/δ	$\mu_{\alpha*}$/α*	$\mu_{\alpha*}$/δ	$\mu_{\alpha*}$/π	μ_δ/α*	μ_δ/δ	μ_δ/π	μ_δ/$\mu_{\alpha*}$	Soln F1 (%)	F2
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1		00 00 00.22	+01 05 20.4	9.10		H	0.000 911 85	+01.089 013 32		3.54	−5.20	−1.88	1.32	0.74	1.39	1.36	0.81	+32	−7	−11	−24	+9	−1	+10	−1	+1	+34	0	0.74
2		00 00 00.91	−19 29 55.8	9.27		G	0.003 797 37	−19.498 837 45	+	21.90	181.21	−0.93	1.28	0.70	3.10	1.74	0.92	+12	−14	−24	−29	+1	+21	−2	−19	−28	+14	2	1.45
3		00 00 01.20	+38 51 33.4	6.61		G	0.005 007 95	+38.859 286 08		2.81	5.24	−2.91	0.53	0.40	0.63	0.57	0.47	+6	+9	+4	+43	−1	−6	+3	+24	+7	+21	0	−0.45
4		00 00 02.01	−51 53 36.8	8.06		H	0.008 381 70	−51.893 546 12		7.75	62.85	0.16	0.53	0.59	0.97	0.65	0.65	−22	−9	−3	+24	+20	+8	+18	+8	−31	−18	0	−1.46
5		00 00 02.39	−40 35 28.4	8.55		H	0.009 965 34	−40.591 224 40		2.87	2.53	9.07	0.64	0.61	1.11	0.67	0.74	+10	+24	+6	+26	−10	+20	−16	−30	−19	+6	0	−1.24
6		00 00 04.35	+03 56 47.4	12.31		G	0.018 141 44	+03.946 488 93		18.80	226.29	−12.84	4.03	2.18	4.99	6.15	3.20	+35	−1	+3	−11	−2	+47	−2	+3	+31	+35	4	2.95
7		00 00 05.41	+20 02 11.8	9.64		G	0.022 548 91	+20.036 602 16		17.74	−208.12	−200.79	1.01	0.79	1.30	1.13	0.82	+32	+8	−2	−4	+12	+6	+11	+0	+16	+43	0	0.21
8		00 00 06.55	+25 53 11.3	9.05	3	H	0.027 291 60	+25.886 474 45		5.17	19.09	−5.66	1.70	0.93	1.95	1.54	0.88	+27	−66	−36	−38	−12	+36	−21	−24	+32	+18	0	0.98
9		00 00 08.48	+36 35 09.4	8.59		H	0.035 341 89	+36.585 937 77		4.81	−6.30	8.42	0.86	0.55	0.99	1.02	0.65	+3	+16	−1	+0	+7	−2	+8	+4	+10	+13	3	−1.26
10		00 00 08.70	−50 52 01.5	8.59		H	0.036 253 09	−50.867 073 60		10.76	42.23	40.02	0.77	0.73	1.10	0.98	0.82	−13	−24	+11	+1	−7	+6	+0	−18	−22	−13	0	0.82
11		00 00 08.95	+46 56 24.0	7.34		H	0.037 296 95	+46.940 001 54		4.29	11.09	−2.02	0.52	0.51	0.84	0.53	0.54	+9	+20	+31	−30	+0	−11	+6	+21	+26	+5	0	−0.23
12		00 00 09.82	−35 57 36.8	8.43		H	0.040 917 56	−35.960 224 82		4.06	−5.99	−0.10	0.81	0.58	1.16	1.02	0.72	+13	−9	−17	−36	+0	+16	−1	−41	+29	+2	2	0.76
13		00 00 10.00	−22 35 40.9	8.80		H	0.041 679 70	−22.594 680 60		3.49	8.45	−10.07	1.21	0.67	1.48	1.44	0.59	+15	+23	+24	+9	+9	+24	−5	−37	−4	−10	0	−0.46
14		00 00 11.59	−00 21 37.5	7.25		G	0.048 271 89	−00.360 421 19		5.11	61.75	−11.67	0.88	0.54	0.99	1.12	0.59	+34	+1	−21	+23	−3	+11	+1	−24	+27	+40	0	−0.31
15		00 00 12.07	+50 47 28.2	8.60		H	0.050 308 90	+50.791 173 84		2.45	13.88	5.47	0.66	0.70	1.16	0.78	0.70	−27	+8	+27	+22	+5	+9	+6	−6	+14	−19	0	0.35
*16		00 00 12.34	−54 54 50.9	11.71		G	0.051 408 52	−54.914 128 19		0.53	257.39	−96.63	1.49	1.67	2.63	1.81	1.95	−27	−7	−13	+2	+3	+4	+5	−8	−9	−22	0	0.76
*17		00 00 12.26	−40 11 32.4	8.15		H	0.051 099 57	−40.192 328 42		6.15	−34.46	−26.37	0.57	0.55	1.00	0.61	0.65	+18	+24	+13	+29	−7	+11	−15	−35	−18	+11	0	−0.62
18		00 00 12.75	−04 03 13.5	11.03		G	0.053 139 23	−04.053 738 13		19.93	−127.22	23.78	2.18	1.20	2.36	2.69	1.15	+24	+19	+4	−8	+3	+15	−3	−30	+26	+18	3	0.05
19		00 00 12.80	+38 18 14.7	6.53		H	0.053 316 96	+38.304 086 36		4.12	−2.50	−15.07	0.55	0.40	0.64	0.60	0.45	+9	−5	+3	+53	−8	−10	−4	+21	+8	+23	0	−0.84
20		00 00 15.11	+23 31 45.4	8.51		G	0.062 950 50	+23.529 283 97		10.76	36.00	−22.98	0.88	0.59	1.06	0.92	0.59	+30	+39	+19	+10	+2	+8	+9	−5	+20	+32	0	0.50
21		00 00 15.90	+08 00 26.0	7.55		H	0.066 235 69	+08.007 234 37		5.84	61.89	−0.22	0.84	0.51	0.95	0.84	0.56	+20	+13	−9	−10	+1	+3	+6	−16	+30	+36	2	2.05
22		00 00 16.83	−49 21 08.2	8.69		H	0.070 135 93	−49.352 266 86		4.47	−7.90	0.46	0.63	0.79	1.15	0.71	0.93	−6	−1	−10	−16	−14	+8	−12	−26	−6	+7	0	−0.40
23		00 00 17.86	+13 18 44.0	7.57		G	0.074 429 30	+13.312 210 83		12.21	54.15	9.65	0.90	0.52	0.95	0.91	0.55	+17	+10	+12	−23	+19	+9	+15	−1	+10	+36	0	0.32
24		00 00 18.25	−23 27 09.9	9.05		H	0.076 049 79	−23.452 749 13		9.73	127.15	22.22	1.00	0.61	1.21	1.21	0.55	+3	−12	+21	−10	+12	+27	+1	−37	−2	−13	1	0.24
25	H	00 00 19.05	−44 17 25.1	6.28		G	0.079 365 37	−44.290 297 41	A	13.74	58.36	−108.64	0.88	0.81	0.98	0.73	0.68	−32	+18	−7	+2	−14	+2	−10	−25	+10	−2	2	−0.52
26		00 00 20.24	−13 23 35.9	9.13		H	0.084 345 79	−13.393 296 86		9.19	−103.33	−33.35	1.00	0.83	1.34	1.42	0.75	+42	−13	−12	−12	−16	+33	−24	−55	+25	+31	0	−0.56
27		00 00 20.51	−41 17 51.1	9.32		H	0.085 469 76	−41.297 536 96		9.66	135.96	−113.67	0.82	0.73	1.38	0.95	0.94	−2	−6	+7	+19	−3	+6	−4	−34	−15	−4	2	−0.07
28		00 00 20.94	−43 21 42.5	8.83		H	0.087 248 71	−43.361 799 62		5.64	−10.96	−8.69	0.71	0.60	1.05	0.83	0.78	+5	−3	−11	+17	−15	+26	−14	−45	+8	+8	0	−0.76
29		00 00 22.11	−49 06 28.6	9.14		H	0.092 131 06	−49.107 955 05		2.85	26.86	4.05	0.75	0.92	1.40	0.79	1.09	−9	−11	−18	−24	−8	−3	−11	−24	+8	+0	0	−0.23
30		00 00 23.07	+42 08 29.4	8.26		H	0.096 135 63	+42.141 498 82		3.79	−8.44	−10.14	0.53	0.53	0.88	0.72	0.54	+7	+11	+26	−9	−5	−14	+2	+14	+16	−8	0	0.03
31		00 00 23.54	+02 40 31.7	7.63		H	0.098 093 90	+02.675 477 68		1.84	−4.88	−0.20	1.06	0.61	1.05	1.49	0.69	+19	+17	−4	−33	+14	−2	+22	+10	+7	−7	0	−0.07
32		00 00 23.66	+51 56 22.2	9.09		H	0.098 583 75	+51.939 490 50		3.10	−0.39	−1.38	0.78	0.74	1.29	0.94	0.79	−18	+7	+18	+4	+8	+8	+8	+2	+19	−12	2	0.61
33		00 00 23.80	−10 27 44.8	8.10		H	0.099 180 83	−10.462 454 45		8.94	−3.62	28.71	0.90	0.48	1.09	1.39	0.47	+30	−31	−6	−25	+2	+33	−14	−38	+30	+23	0	0.26
34		00 00 23.87	+26 55 05.7	6.43		H	0.099 469 73	+26.918 238 21		12.71	42.20	53.47	0.61	0.43	0.74	0.64	0.43	+15	+22	+13	+0	+4	+2	+7	−7	+15	+9	0	1.99

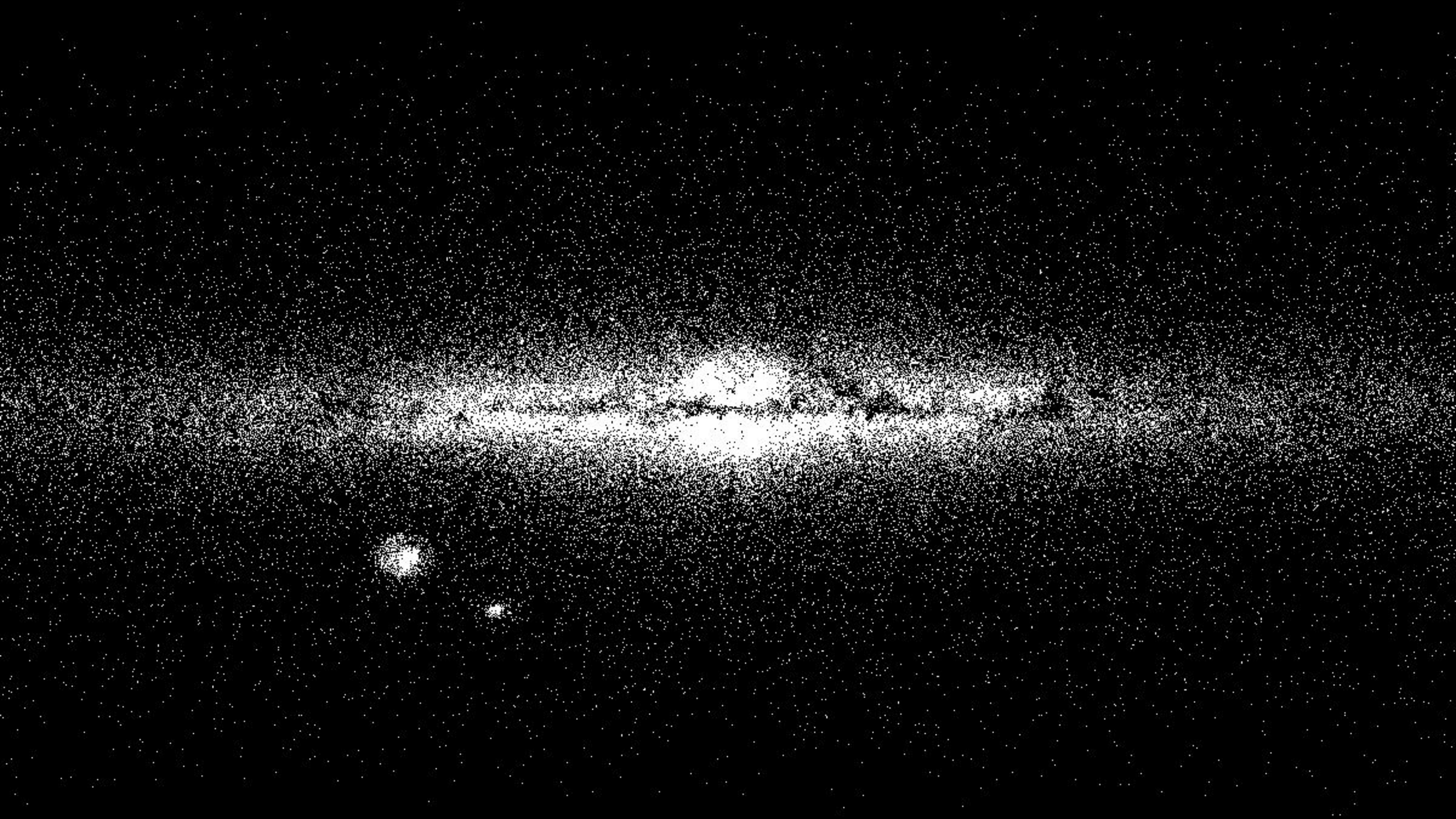
Motivation

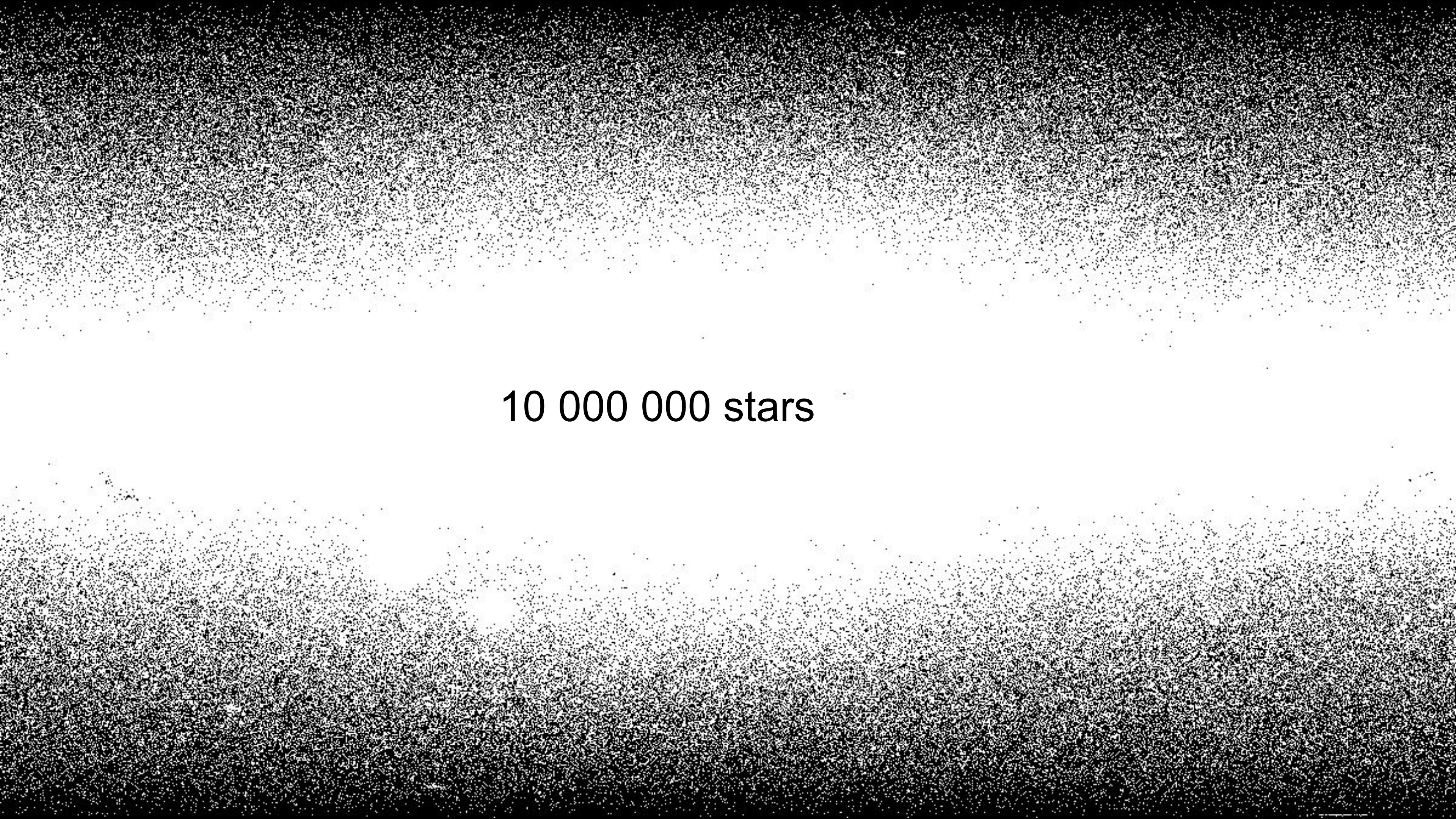


Motivation

- The (original) problem: handling uncomfortably large data
  - Gaia Data: ~1.800.000.000 (billion!) of stars (rows), ~100 columns (features)
  - 10 000x increase in data
- What we need
  - Interactive exploration, visualisation, statistical analysis
- Requirements
  - Low barrier of entry, Accessible, Cheap
  - Easy setup
  - No cluster

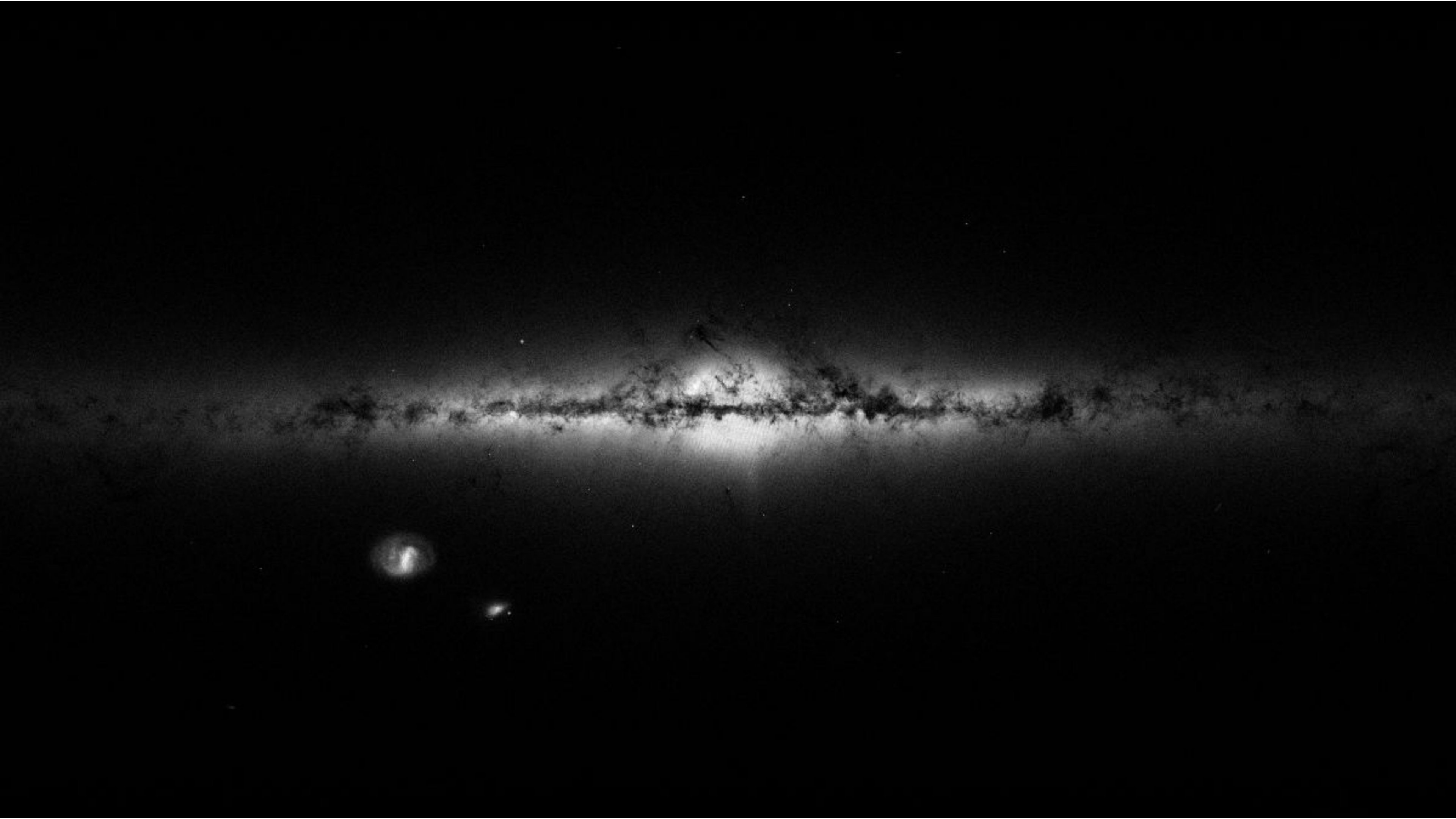


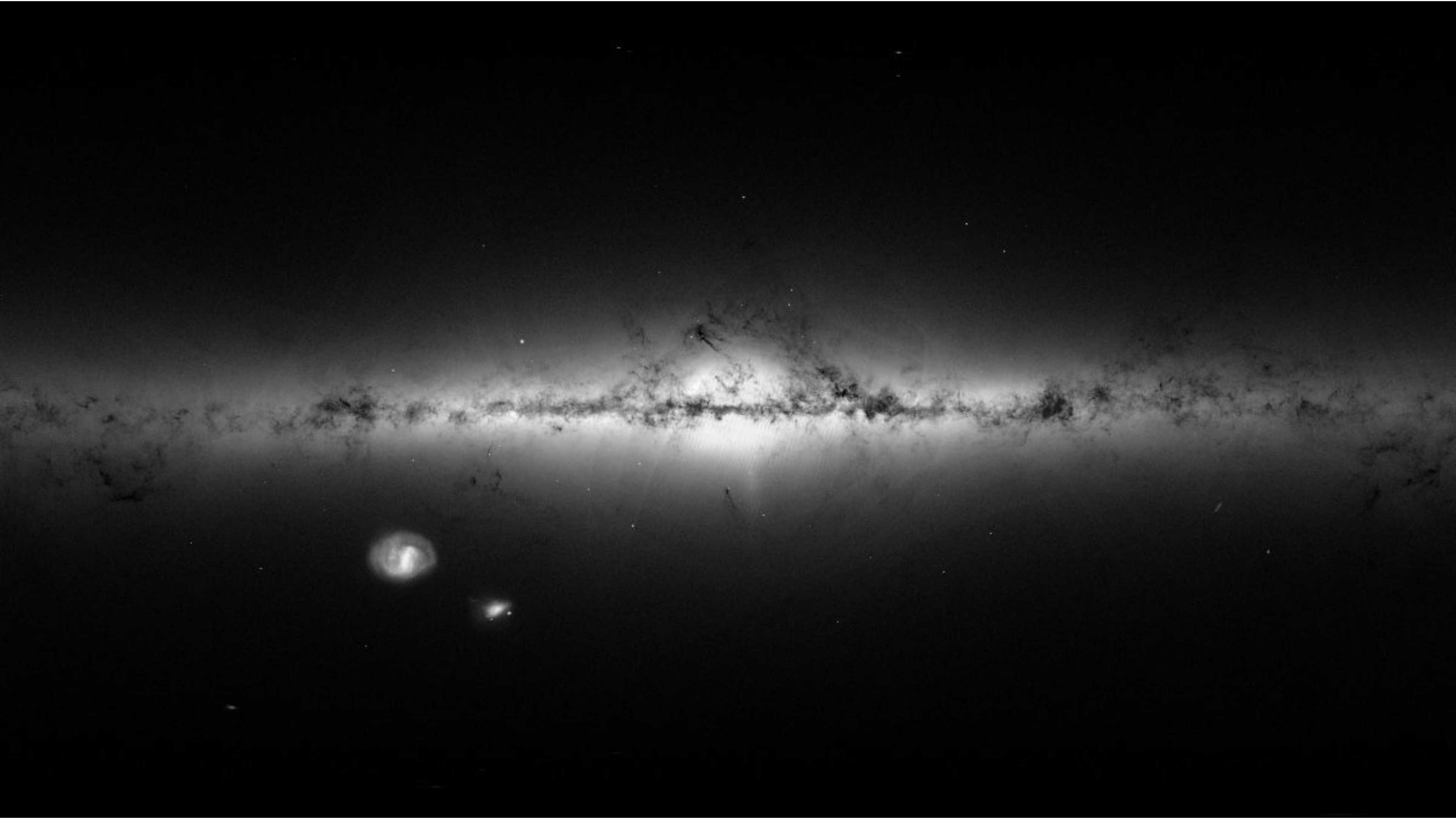


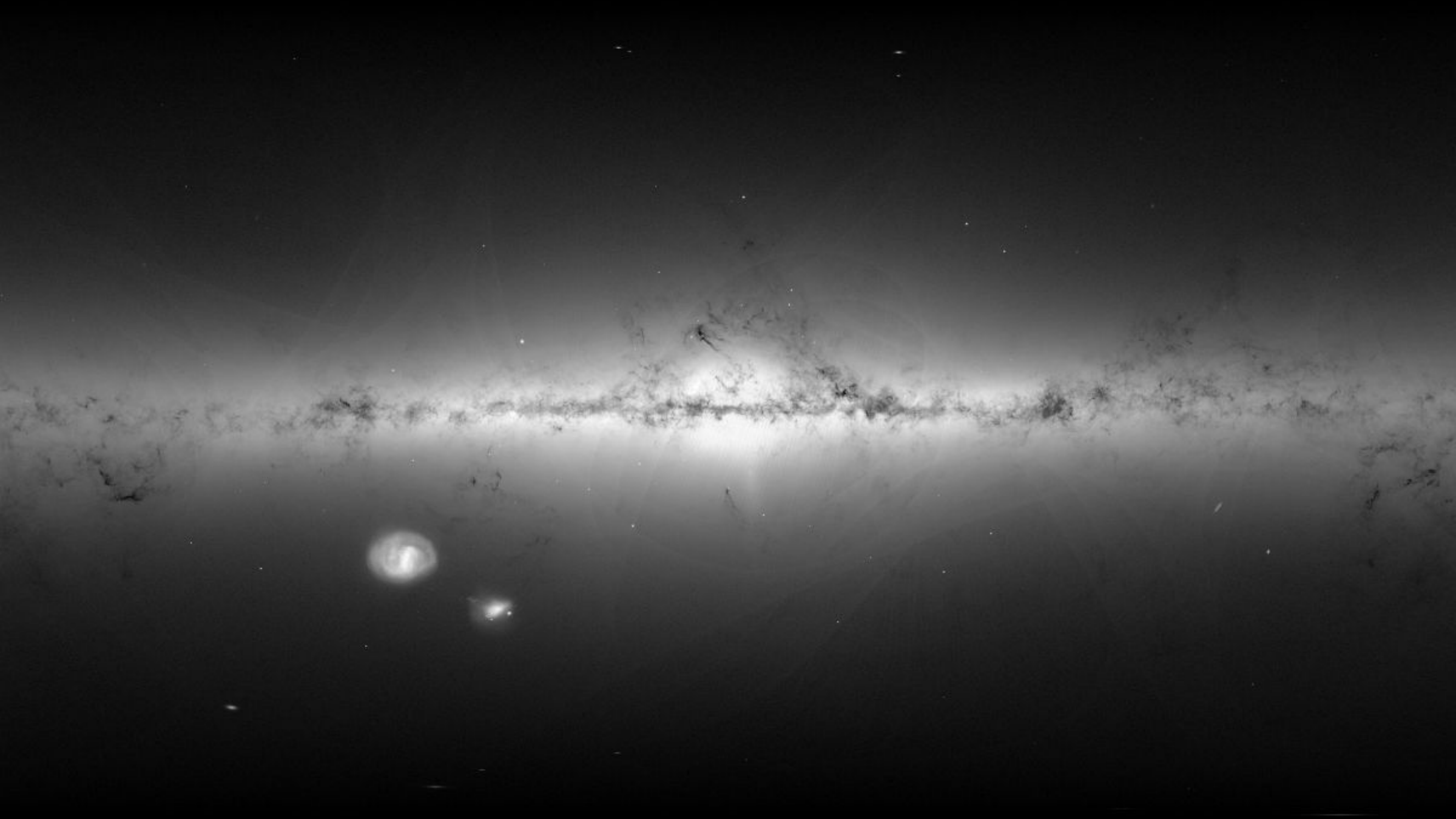

10 000 000 stars

100 000 000 stars

1 000 000 000 stars







Motivation

- Memory use
  - 1 double precision = 8 bytes
  - 2 columns
  - 1 billion rows
  - 1 billion * 2 * 8 = 15GiB
- Memory bandwidth 10-100 GB/s
  - < 1 seconds
- CPU
  - 4Ghz, but 1-100 cores
  - 4-100 cycles/row in 1 second

Vaex

goto;

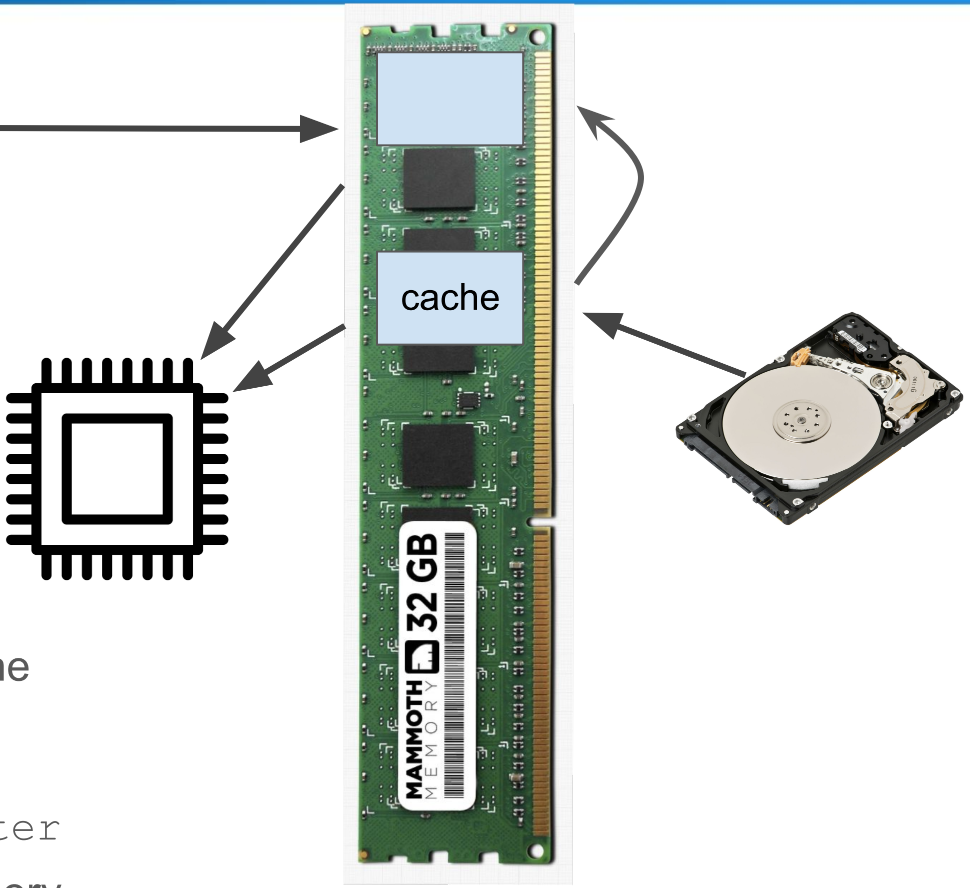
- High-performance, out-of-core DataFrame library
- Out-of-core algorithms ensure working with data larger than RAM
- Work with billions (10^9) of samples on a single machine / laptop interactively
- Like Pandas (similar API) but not built on Pandas
- Easy installation:
    - `pip install vaex`
- Legal: Free & Open Source, MIT License

Vaex concepts

- Memory mapping
- Column based storage
- No memory copies
- Compute layer
- Expression system

Vaex concepts: memory mapping

- Normal disk reading
    - Allocate memory
    - Read from disk to memory
        - disk > cache > memory
    - Memory to CPU
    - Waste memory (thus cache)
- Memory mapping
    - Get a pointer to the same cache
    - No copy
    - Memory not taken, can be freed
    - Reading on accessing and not in cache
- Shared memory for all processes
    - e.g: `flask`, `gunicorn`, `dash`, `jupyter`
- Store on disk as you would keep it in memory



Vaex concepts: column based storage

- Why ?
    - Focus on analytical workloads
    - Sequential access is optimized for CPU and disk
    - Disk reading is block based
    - CPU cache friendly

Vaex concepts: column based storage

- HDF5
  - Non-standard: Container format, anything can be stored
  - Stored as a single contiguous array
  - Native storage
- Apache Arrow
  - Standard: interoperability
  - Specialized for tabular data
  - Chunks of column based
  - Native storage
- Apache Parquet
  - Industry standard
  - Not-native, compressed, requires deserialization

Vaex concepts: no memory copies

goto;

```
df = {
    'data': {
        'x': np.arange(4),
        'y': np.array([0, np.nan, 5, 1, 1e10])
    },
    'state': {}
}
```

```
df2 = df[df.y<10]
```

```
df2 = {
    'data': same_data,
    'state': {
        'filter': 'y < 10'
    }
}
```

```
df2['z'] = df.x + df.y*10
```

```
df2 = {
    'data': same_data
    'state': {
        'filter': 'y < 10'
        'virtual_columns': {
            'z': 'x + y*10'
        }
    }
}
```

Vaex concepts: compute & expression system

- Compute:
  - Streaming algorithms
  - Map reduce
  - C++ under the hood
- Expression system
  - Store the expression, not the result
  - Evaluate when needed
  - JIT via Numba, Pythran, Cuda

```python
import numpy as np
x = np.arange(5)
y = x**2
x * y
```

```
array([ 0,  1,  8, 27, 64])
```

```python
import vaex
df = vaex.from_arrays(x=x, y=y)
df.x * df.y
```

```
Expression = (x * y)
Length: 20 dtype: int64 (expression)
-----------------------------------
 0      0
 1      1
 2      8
 3     27
 4     64
   ...
15   3375
16   4096
17   4913
18   5832
19   6859
```

Vaex.io: from academic project to consultancy

- Solved a problem in astronomy
- Realized this solves a broader problem
- Moved towards industry grade features & applications
    - Fast string support
    - GroupBy
    - Join
    - ML integration
    - Automatic Pipeline & DataFrame State tracking
- Unclustering your data science

“Never do a live demo”

-Many people

Vaex in Production

- This is nice, but I need to put it in production:
  - Make a dashboard
  - Provide a data web API
  - Deploy an ML model
- Vaex is ideal
  - Memory mapping shares data
  - Integrated caching system can cache common queries/results
  - Well tested with Dash, Flask, FastAPI
  - Support cloud storage: s3, gcs (pyarrow, fsspec)
  - Subpackages for less dependencies (pick what you need)
    - `vaex-core`
    - `vaex-ml`
    - `vaex-viz`
    - more!

Vaex in the wild

BioSkryb GENOMICS

Field: Genomics
Use: Interactive exploration of genomics data (x240 performance increase)

STScI | SPACE TELESCOPE SCIENCE INSTITUTE

Field: Astronomy & Astrophysics
Use: Remote interactive exploration & visualisation and analysis of large datasets

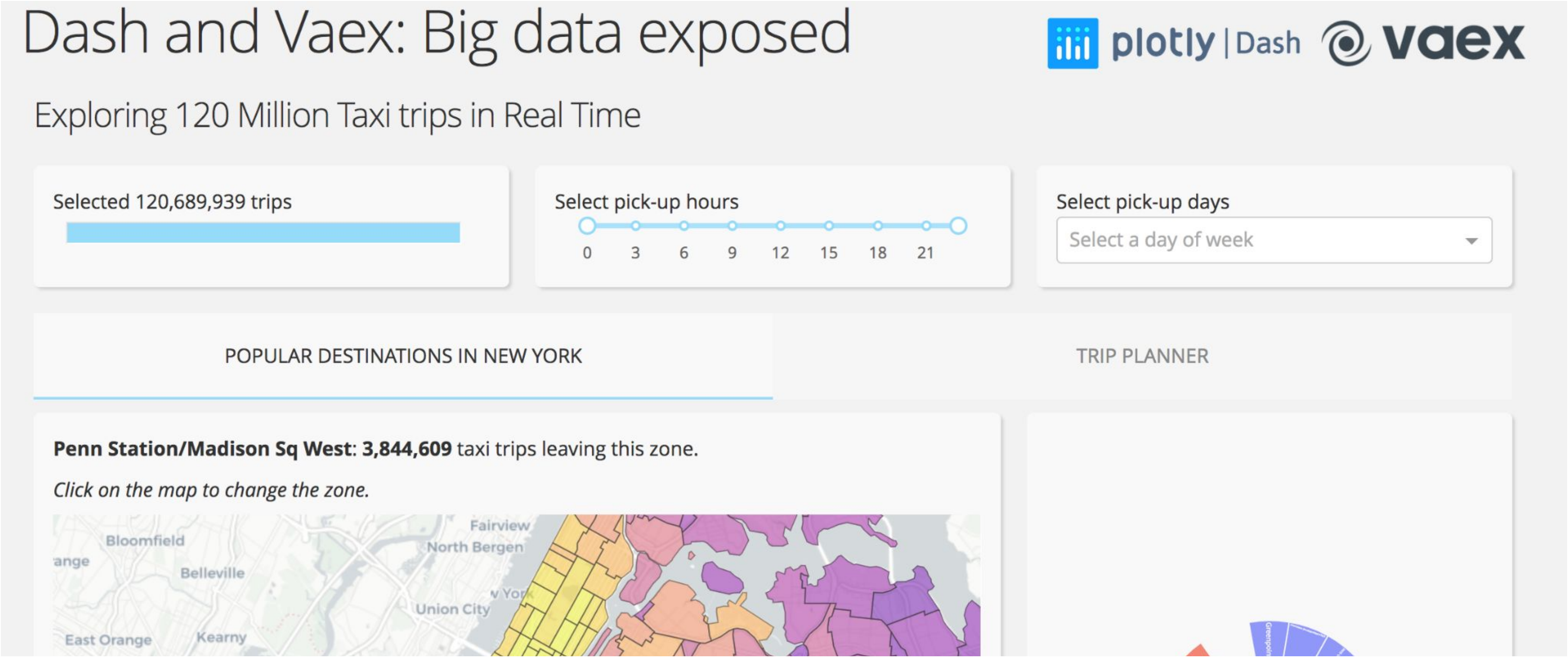
plotly

Field: Data visualisation & Dashboards
Use: The primary backend to the newly released Plotly Dashboard Engine;
Plotly Dash clients get access to big data with no setup.

Vaex in Production: Dash example

goto;

https://dash.vaex.io/



Vaex.io: Consultancy

- Features development
- Support
- Retainers
- Performance
- Training

Flow back to open source development & maintenance

Summary

- Operating on 1 billion rows in 1 second is possible
- How we do this
  - Memory mapping
  - Column based storage
  - Multithreading and C++
  - Expression system
- Ideal for
  - Interactive exploration (EDA)
  - Backends (dashboard, Web APIs)
- No cluster needed

Get in touch!

- contact@vaex.io - support / consultancy training
- https://github.com/vaexio/vaex
- @vaex_io
- Documentation: https://vaex.io/docs/
- Examples: https://github.com/vaexio/vaex-examples/
- This talk and notebook:
  - https://github.com/vaexio/vaex-talks

Don't forget to
vote for this session
in the **GOTO Guide app**