

Vaex:

Out of core Dataframes

for Python

Maarten Breddels
Freelance / consultant / vaex.io

VAEX.IO: WHO ARE WE?

VAEX.IO: WHO ARE WE?



Maarten Breddels

Former astrophysicist

Freelancer / consultant / data scientist

Core Jupyter-Widgets developer

QuantStack partner

Founder of vaex.io

Principal author of vaex

✉ maartenbreddels@gmail.com

www.maartenbreddels.com

🐦 [@maartenbreddels](https://twitter.com/maartenbreddels)

🐙 github.com/maartenbreddels

VAEX.IO: WHO ARE WE?



Maarten Breddels

Former astrophysicist

Freelancer / consultant / data scientist

Core Jupyter-Widgets developer

QuantStack partner

Founder of vaex.io

Principal author of vaex

✉ maartenbreddels@gmail.com

🌐 www.maartenbreddels.com

🐦 [@maartenbreddels](https://twitter.com/maartenbreddels)

🐙 github.com/maartenbreddels



Jovan Veljanoski

Former astrophysicist

Sr. Data Scientist @ XebiaLabs

Co-founder of vaex.io

✉ jovan.veljanoski@gmail.com

🌐 <https://www.linkedin.com/in/jovanvel/>

VAEX.IO: WHO ARE WE?



Maarten Breddels

Former astrophysicist
Freelancer / consultant / data scientist
Core Jupyter-Widgets developer
QuantStack partner
Founder of vaex.io
Principal author of vaex
✉ maartenbreddels@gmail.com
🌐 www.maartenbreddels.com
🐦 [@maartenbreddels](https://twitter.com/maartenbreddels)
🐙 github.com/maartenbreddels



Jovan Veljanoski

Former astrophysicist
Sr. Data Scientist @ XebiaLabs
Co-founder of vaex.io
✉ jovan.veljanoski@gmail.com
🌐 <https://www.linkedin.com/in/jovanvel/>



Yonatan Alexander

Head of Data Science at BuiltOn
✉ jonathan@xdss.io
🌐 <https://www.linkedin.com/in/xdssio/>



Mario Buikhuizen

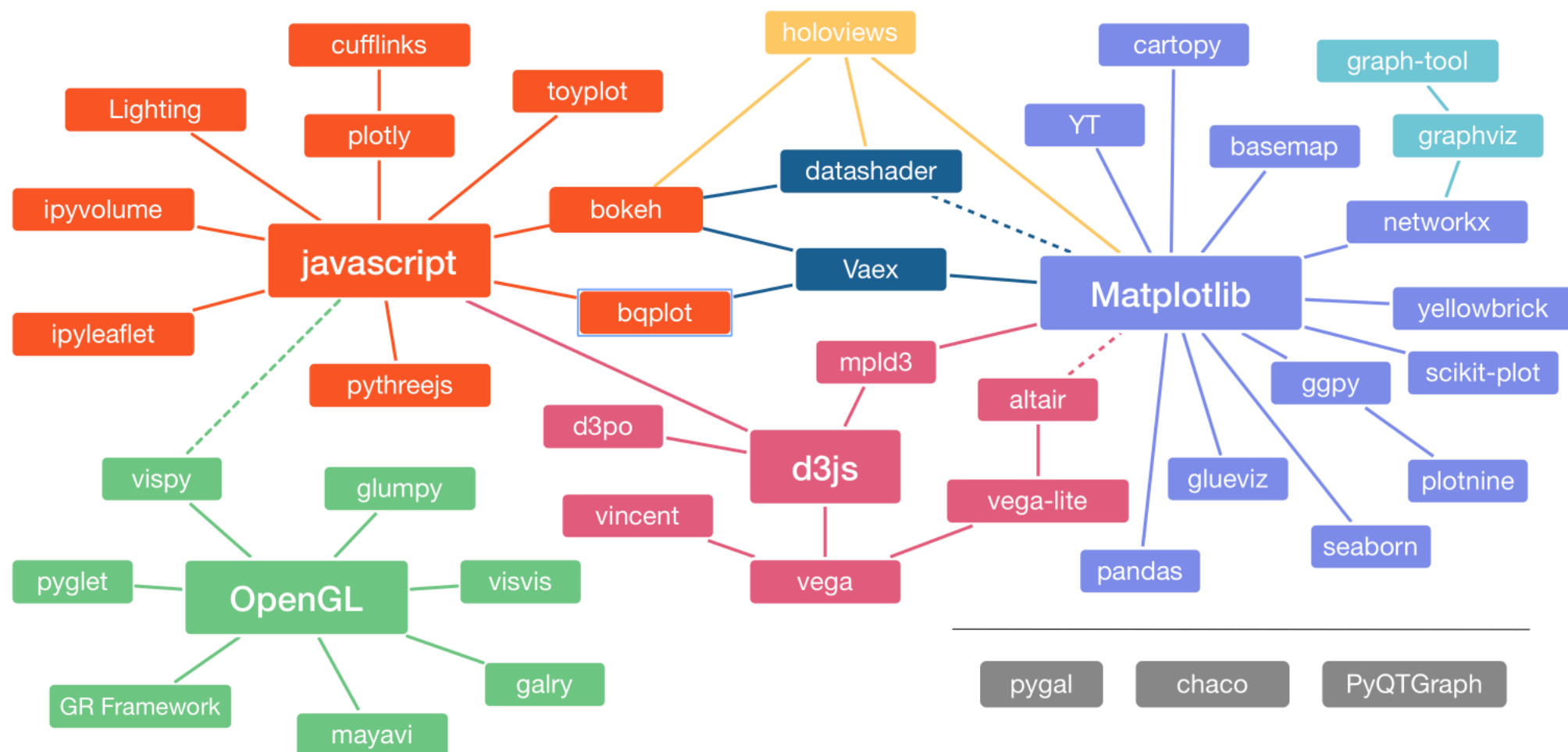
Freelancer / consultant
Front-end / dashboards / widgets specialist
✉ mbuikhuizen@gmail.com

Outline

- What is vaex?
- Why does vaex (still) exist?
- What makes vaex unique?
 - DataFrame: Data + state
 - Expression system
- Live demo from notebook
- Misc
 - AWS s3 support, CUDA, remote data frames, xarray

PyViz landscape

- unorganized or choice



DataFrames

DataFrames

- Pandas

DataFrames

- Pandas
 - `dask.dataframe`

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)
- SFrame (turi/Apple)

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)
- SFrame (turi/Apple)
- cudf (nVidia/rapids)

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)
- SFrame (turi/Apple)
- cudf (nVidia/rapids)
- Vaex

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)
- SFrame (turi/Apple)
- cudf (nVidia/rapids)
- Vaex
- Data format?

DataFrames

- Pandas
 - `dask.dataframe`
 - Modin (using Ray)
- SFrame (turi/Apple)
- cudf (nVidia/rapids)
- Vaex
- Data format?
 - Apache Arrow

What is *vaex*

What is vaex

- DataFrame library (vaex-core) ala Pandas

What is vaex

- DataFrame library (vaex-core) ala Pandas
- Many (domain specific) libraries
 - vaex-hdf5 — hdf5 file support
 - vaex-arrow — Apache Arrow support
 - vaex-viz — (matplotlib) based plotting
 - vaex-server — serves remote data frames
 - vaex-ml — ML integration (sklearn, annoy, xgboost, lightgbm, catboost) + automatic pipelines
 - ...

What is vaex

- DataFrame library (vaex-core) ala Pandas
- Many (domain specific) libraries
 - vaex-hdf5 — hdf5 file support
 - vaex-arrow — Apache Arrow support
 - vaex-viz — (matplotlib) based plotting
 - vaex-server — serves remote data frames
 - vaex-ml — ML integration (sklearn, annoy, xgboost, lightgbm, catboost) + automatic pipelines
 - ...
- `pip install vaex==2` / `conda install -c conda-forge vaex`

What is vaex

- DataFrame library (vaex-core) ala Pandas
- Many (domain specific) libraries
 - vaex-hdf5 — hdf5 file support
 - vaex-arrow — Apache Arrow support
 - vaex-viz — (matplotlib) based plotting
 - vaex-server — serves remote data frames
 - vaex-ml — ML integration (sklearn, annoy, xgboost, lightgbm, catboost) + automatic pipelines
 - ...
- `pip install vaex==2` / `conda install -c conda-forge vaex`
- Without the kitchen sink:
 - `pip install vaex-core vaex-hdf5 vaex-viz vaex-ml`

Why does *vaex* exist?

Why does *vaex* exist?

- Very memory efficient due to expression system

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping
 - Open a 1.2TB file on your laptop instantly

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping
 - Open a 1.2TB file on your laptop instantly
- No cluster needed

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping
 - Open a 1.2TB file on your laptop instantly
- No cluster needed
 - Handle over 1 billion rows on your laptop

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping
 - Open a 1.2TB file on your laptop instantly
- No cluster needed
 - Handle over 1 billion rows on your laptop
 - 10-1000x string processing speedup (wrt Pandas)

Why does vaex exist?

- Very memory efficient due to expression system
 - +free pipelines
 - +free jitting to numba/pythran/CUDA
- Hdf5 + arrow spec (columnar storage) + memory mapping
 - Open a 1.2TB file on your laptop instantly
- No cluster needed
 - Handle over 1 billion rows on your laptop
 - 10-1000x string processing speedup (wrt Pandas)
 - Saves time/money/energy

Vaex: data + state

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


Vaex: data + state

```
df = {  
    'data': {  
        'x': np.arange(4),  
        'y': np.array([0, np.nan, 5, 1, 1e10])  
    },  
    'state': {}  
}  
  
df2 = df[df.y<10]
```

Vaex: data + state

```
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'  
    }  
}
```

Vaex: data + state

```
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
```

Vaex: data + state

```
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'  
        }  
    }  
}
```

“Never do a live demo”

-Many people

Conclusions

Conclusions

- Vaex: A very fast and memory efficient Dataframe library

Conclusions

- Vaex: A very fast and memory efficient Dataframe library
 - 1 billion rows - 1TB data on a laptop

Conclusions

- Vaex: A very fast and memory efficient Dataframe library
 - 1 billion rows - 1TB data on a laptop
- Concept of data + state (virtual columns, filters)

Conclusions

- Vaex: A very fast and memory efficient Dataframe library
 - 1 billion rows - 1TB data on a laptop
- Concept of data + state (virtual columns, filters)
- Expression system allows jitting (numba, pythran, CUDA)

Conclusions

- Vaex: A very fast and memory efficient Dataframe library
 - 1 billion rows - 1TB data on a laptop
- Concept of data + state (virtual columns, filters)
- Expression system allows jitting (numba, pythran, CUDA)
- State 'remembers' the 'pipeline', it's an artifact you get for free. Easy deployment.

Conclusions

- Vaex: A very fast and memory efficient Dataframe library
 - 1 billion rows - 1TB data on a laptop
- Concept of data + state (virtual columns, filters)
- Expression system allows jitting (numba, pythran, CUDA)
- State 'remembers' the 'pipeline', it's an artifact you get for free. Easy deployment.
- S3 support, remote dataframes.

Resources

- Contact:
 - @maartenbreddels
 - maartenbreddels@gmail.com
- vaex.io / docs.vaex.io
- github.com/vaexio/vaex/
- Medium
 - Vaex: Out of Core Dataframes for Python and Fast Visualization
 - Vaex: A DataFrame with super strings