To a Billion and Beyond

How to Visually Explore, Compare and Share Large Quantitative Datasets with HiGlass

Peter Kerpedjiev, Nezar Abdennur, and Fritz Lekschas

Peter Kerpedjiev
Software Engineer at Zymergen

@pkerpedjiev
emptypipes.org

Nezar Abdennur
PostDoc at MIT

@nv1ctus
nvictus.me

Fritz Lekschas
PhD Candidate at Harvard

@flekschas
lekschas.de

Peter Kerpedjiev
Software Engineer at Zymergen

@pkerpedjiev
emptypipes.org

🤰 → 👶 🎉

Nezar Abdennur
PostDoc at MIT

@nv1ctus
nvictus.me

Fritz Lekschas
PhD Candidate at Harvard

@flekschas
lekschas.de

Peter Kerpedjiev
Software Engineer at Zymergen

@pkerpedjiev
emptypipes.org

Nezar Abdennur
PostDoc at MIT

@nv1ctus
nvictus.me

👋

Fritz Lekschas
PhD Candidate at Harvard

@flekschas
lekschas.de

Peter Kerpedjiev
Software Engineer at Zymergen

@pkerpedjiev
emptypipes.org

Nezar Abdennur
PostDoc at MIT

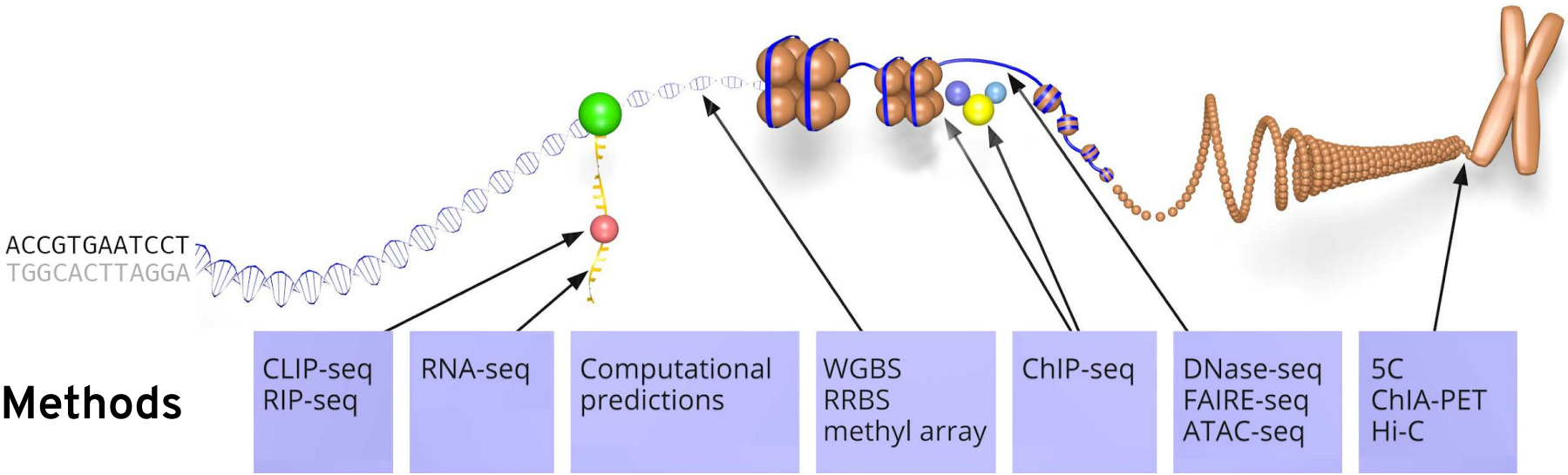
@nv1ctus
nvictus.me

Fritz Lekschas
PhD Candidate at Harvard

@flekschas
lekschas.de

Genomics

where we come from



Methods

Image taken and slightly adapted from encodeproject.org

Principal Challenges

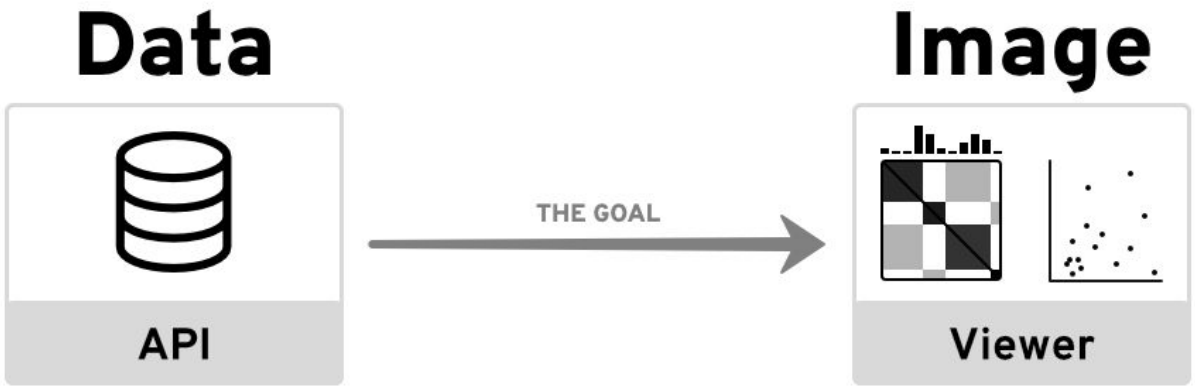
Multiscale
Patterns arise at various resolutions

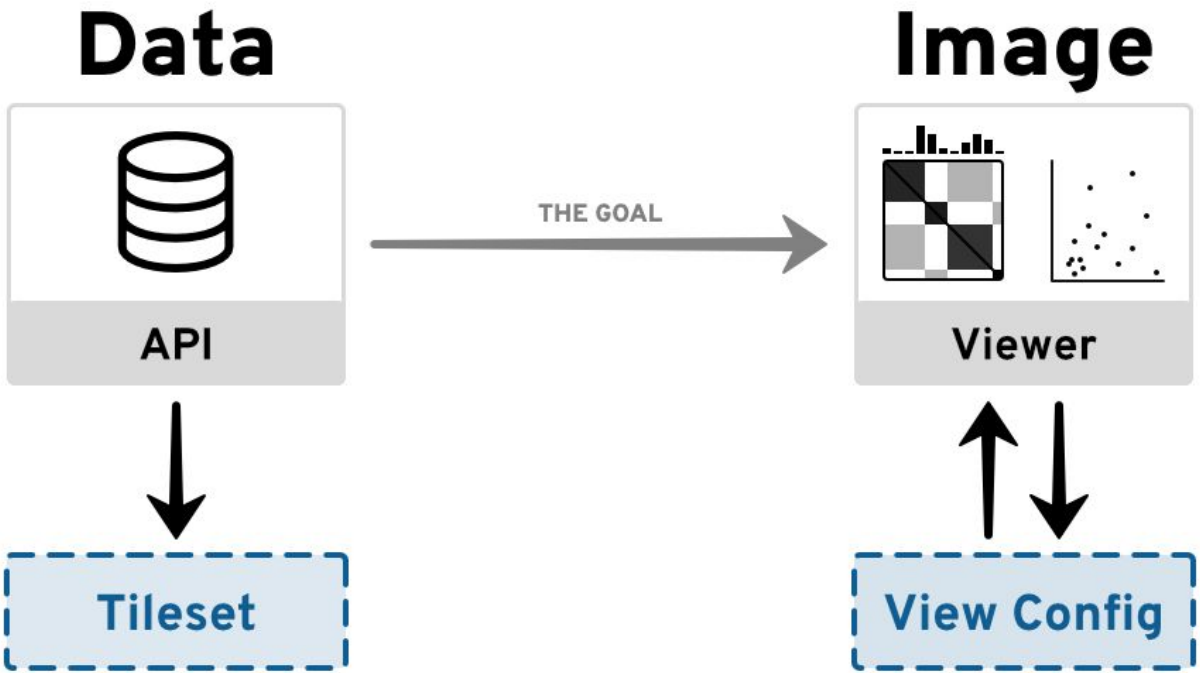
Multimodality
Different data types and synchronized viz

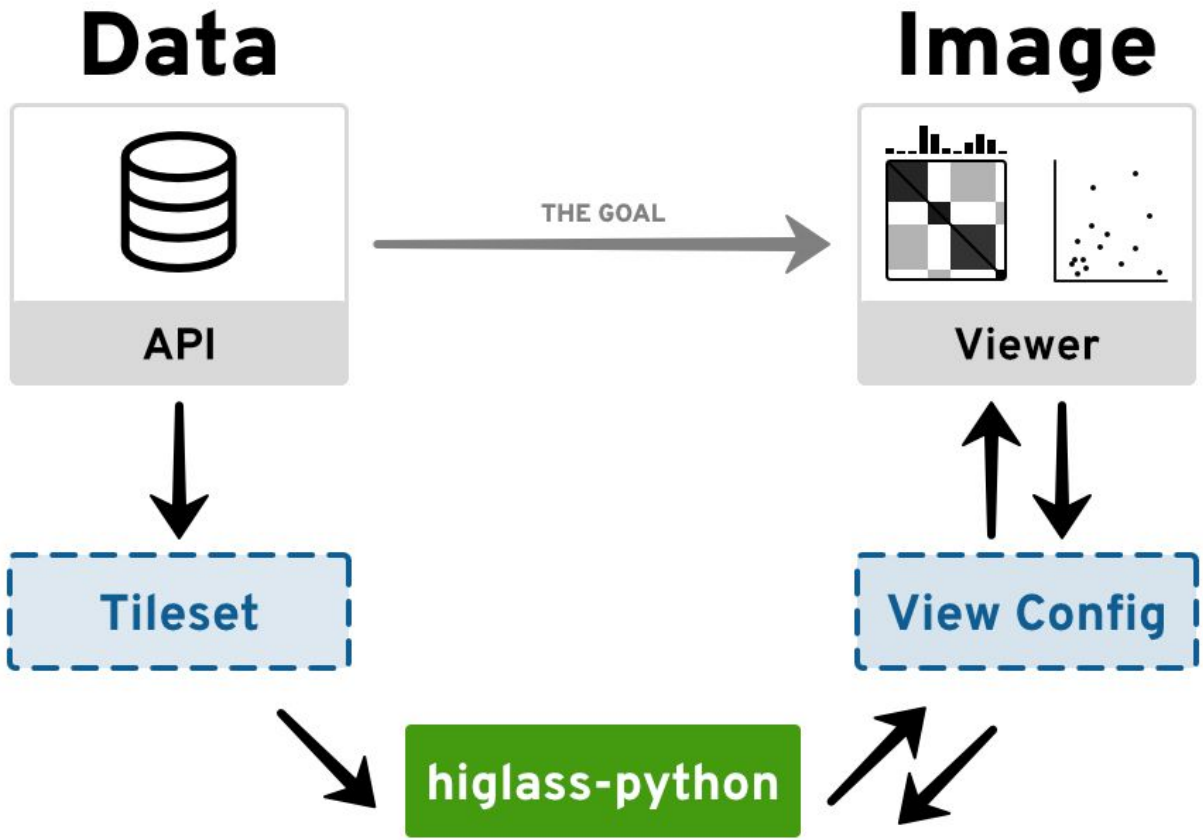
Multiple comparable datasets
Insights arise from differences

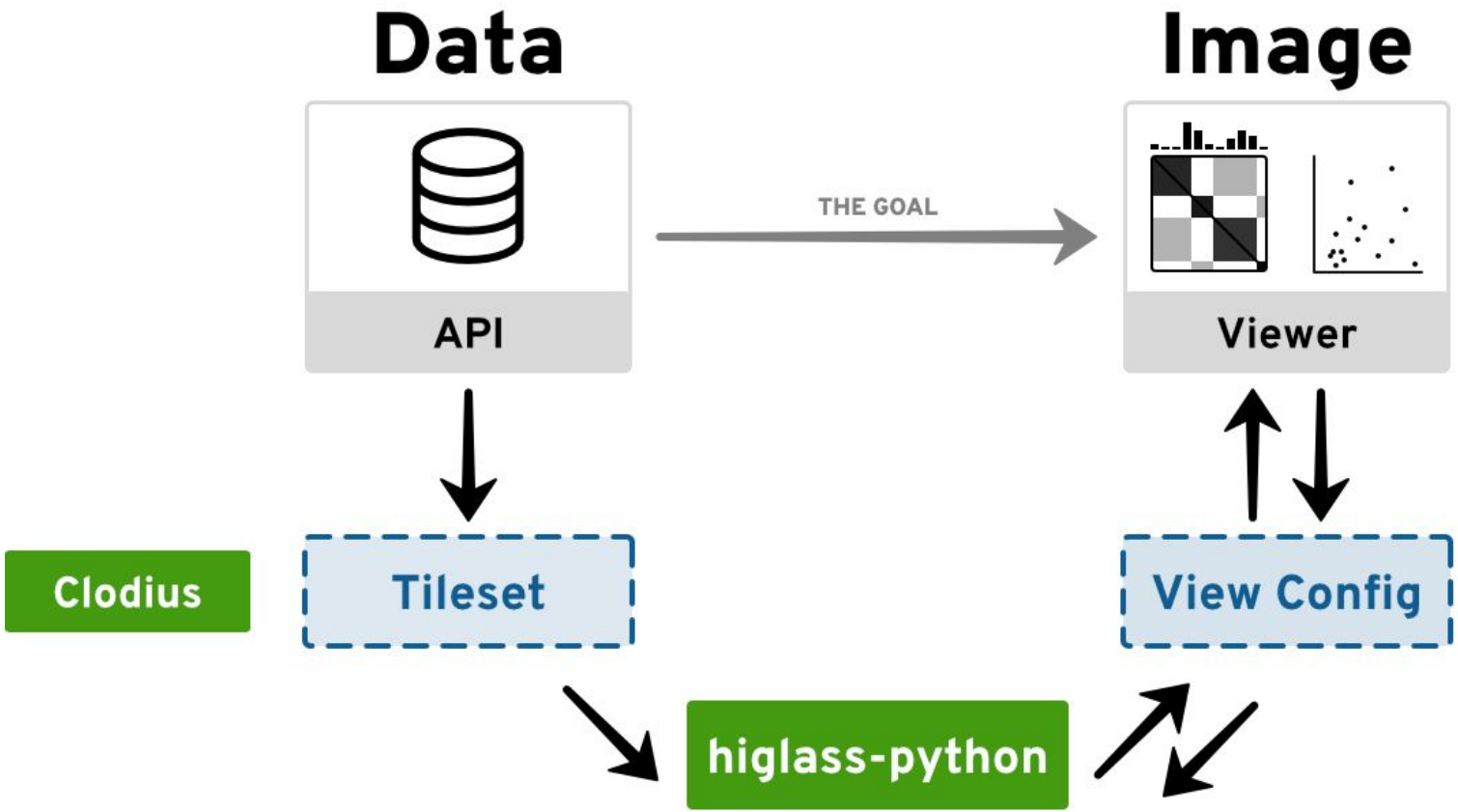
Collaborative Exploration
Share exploratory state, not the end result

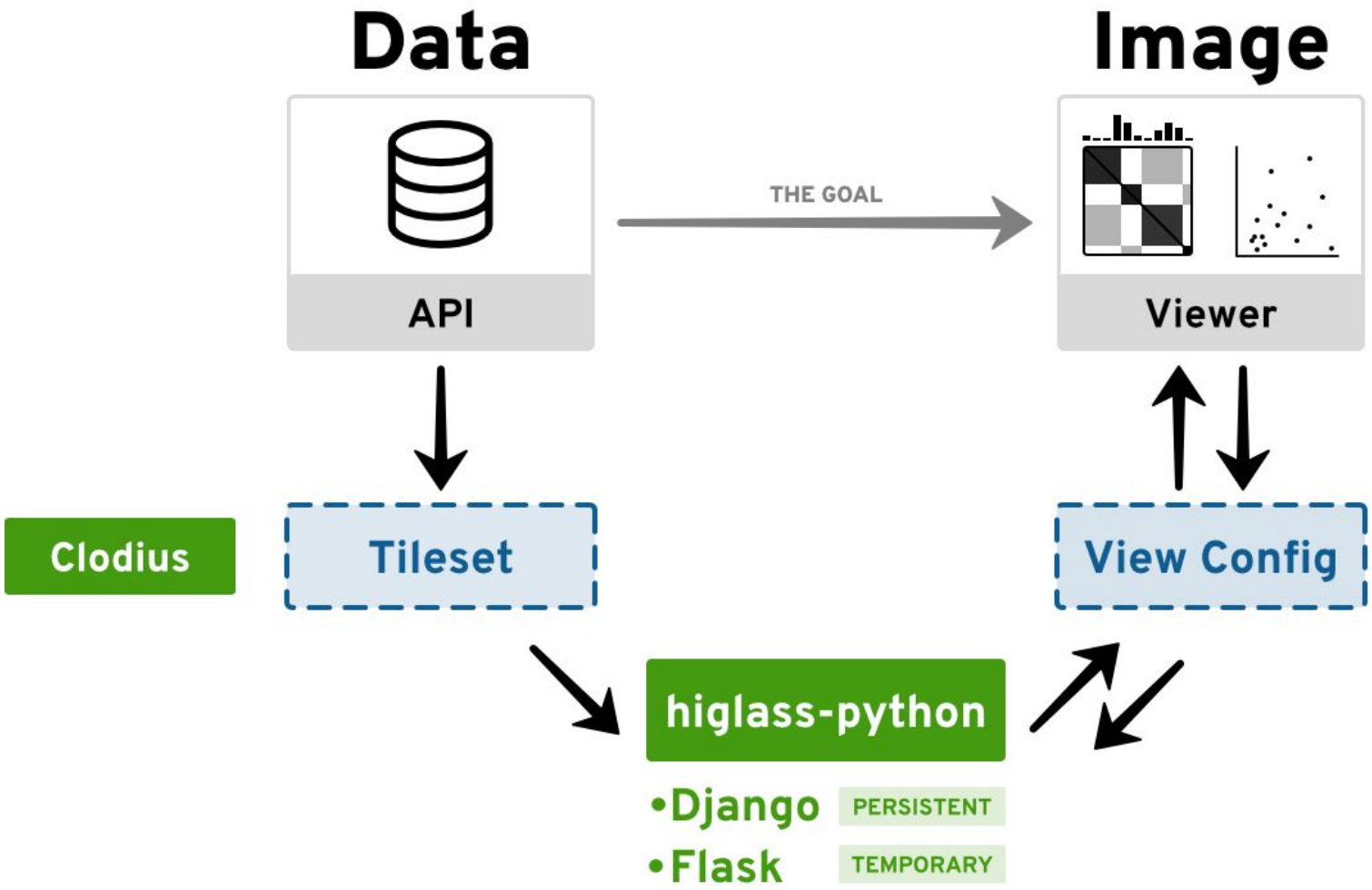
Architecture

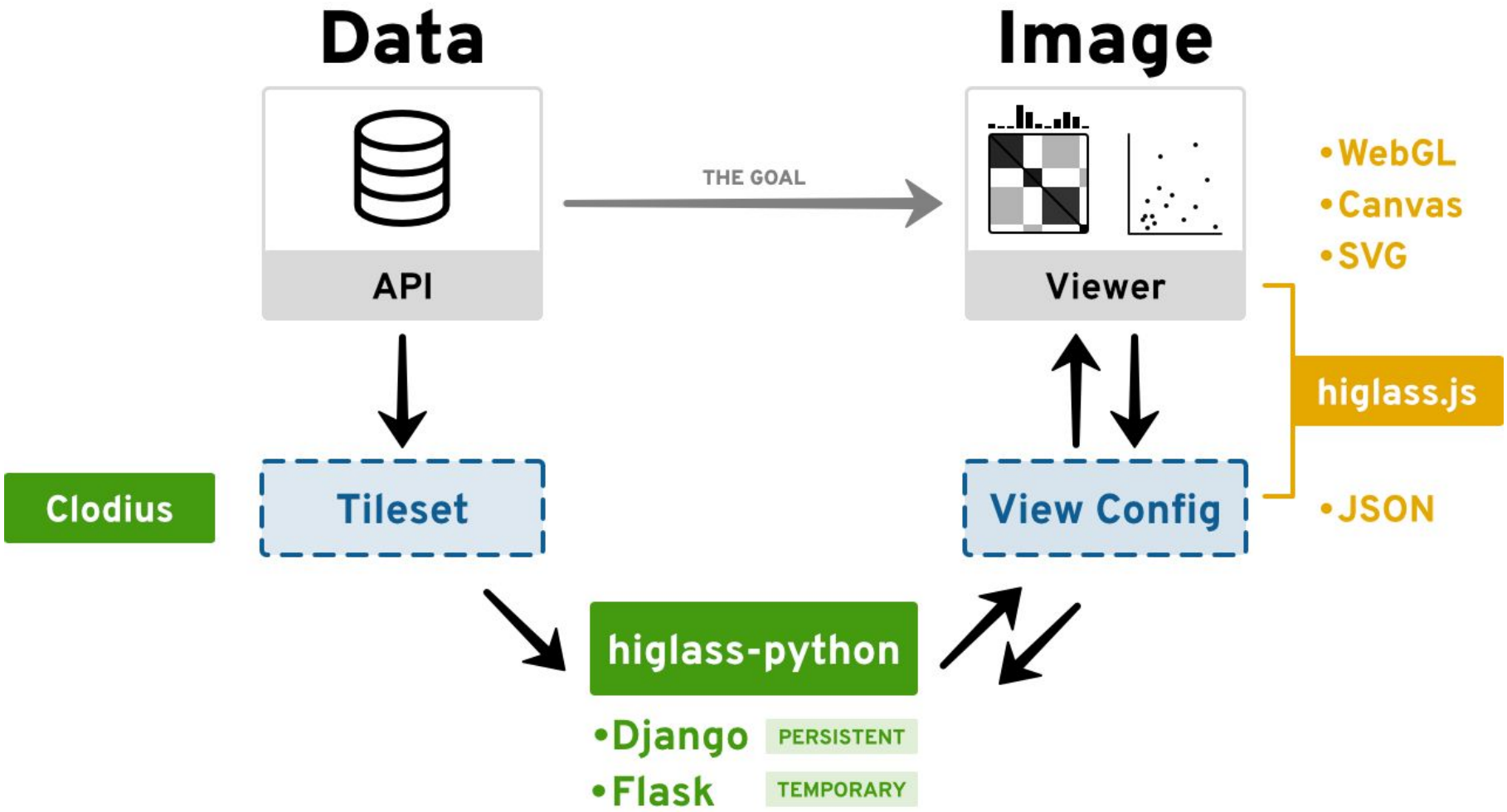

Data
API
THE GOAL
Image
Viewer


Data
Image
THE GOAL
API
Viewer
Tileset
View Config


Data
Image
THE GOAL
API
Viewer
Tileset
View Config
higlass-python


Data
Image
THE GOAL
API
Viewer
Clodius
Tileset
View Config
higlass-python


Data
Image
THE GOAL
API
Viewer
Clodius
Tileset
View Config
higlass-python
•Django
PERSISTENT
•Flask
TEMPORARY


Data
Image
API
THE GOAL
Viewer
•WebGL
•Canvas
•SVG
higlass.js
Clodius
Tileset
View Config
•JSON
higlass-python
•Django
PERSISTENT
•Flask
TEMPORARY

Python API

All demos are available at
github.com/higlass/scipy19

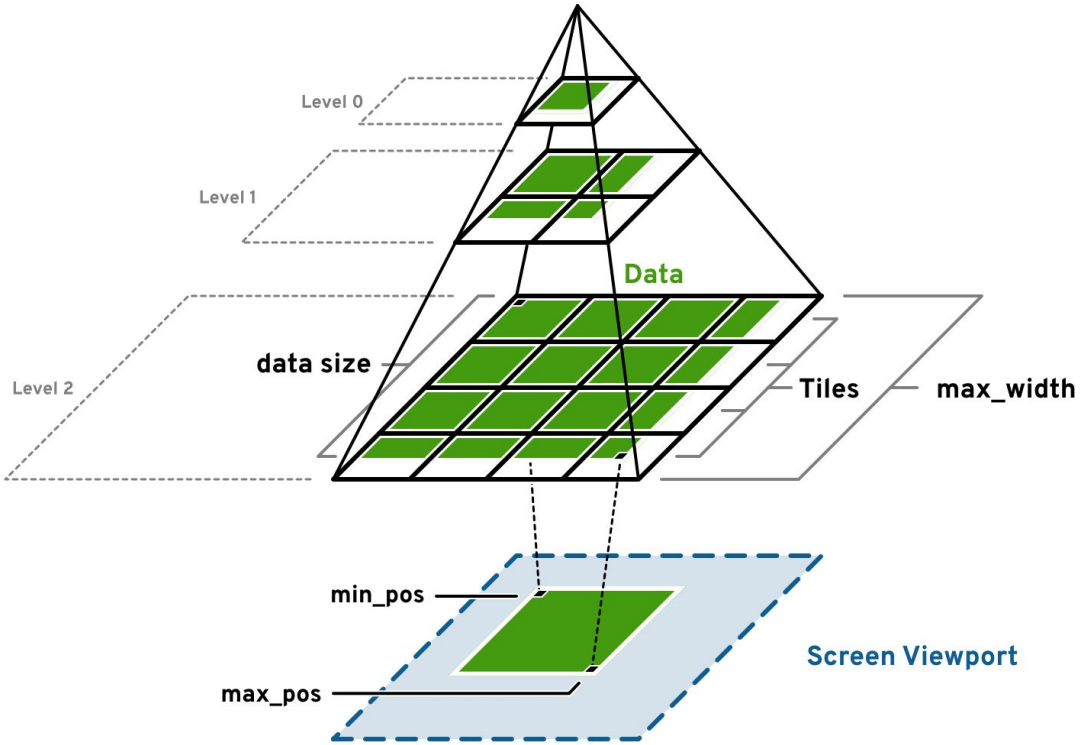
HiGlass in Jupyter

(see temperature.ipynb)

Tileset API

Tileset API

Generates or fetches 1D or 2D data tiles



tileset_info()

→ min_pos & max_pos: relative to the scene

tile_size: size of the tiles (in pixels)

max_zoom: $\lceil \log_2(\text{tile mesh size} / \text{tile size}) \rceil$

max_width: $\lceil \text{tile mesh size} / \text{tile size} \rceil$

tiles(tile_ids)

← [<uuid.z.x.y>, ...]

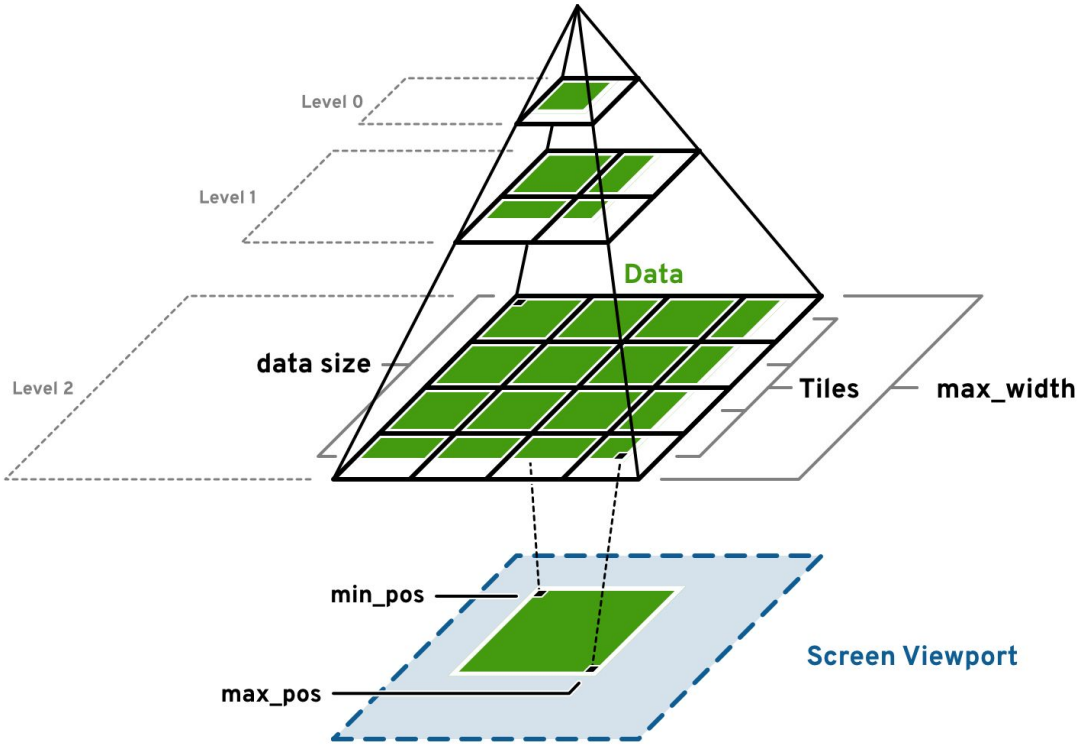
→ {<uuid.z.x.y>: {

dense, min_value, max_value, dtype

}, ...}

Tileset API

Generates or fetches 1D or 2D data tiles



tileset_info()

→ min_pos & max_pos: relative to the scene

tile_size: size of the tiles (in pixels)

max_zoom: $\lceil \log_2(\text{tile mesh size} / \text{tile size}) \rceil$

max_width: $\lceil \text{tile mesh size} / \text{tile size} \rceil$

tiles(tile_ids)

← [<uuid.z.x.y>, ...]

→ {<uuid.z.x.y>: {

dense, min_value, max_value, dtype

}, ...}

Base64-encoded raw data

Tileset API Demo

(see point-data.ipynb)

View Configs

View Config

- Views

  Shared location and zoom

  - Layout
  - Tracks

- Locks

  For view synchronization

- Globals

  Server URLs, editability

```
{
  "views": [
    {
      "uid": "aa",
      "initialXDomain": [0, 100],
      "initialYDomain": [0, 100],
      "layout": {
        "x": 0,  "y": 0,
        "w": 12, "h": 6,
      },
      "tracks": {
        "top": [],
        "left": [],
        "center": [],
        "right": [],
        "bottom": []
      }
    }
  ],
  "zoomLocks": { ... },
  "locationLocks": { ... },
  "valueScaleLocks": { ... },
  "editable": true,
  "zoomFixed": false,
  "trackSourceServers": ["/api/v1"],
  "exportViewUrl": "/api/v1/viewconfs"
}
```

View Config

- Views

  Shared location and zoom

  - Layout
  - Tracks

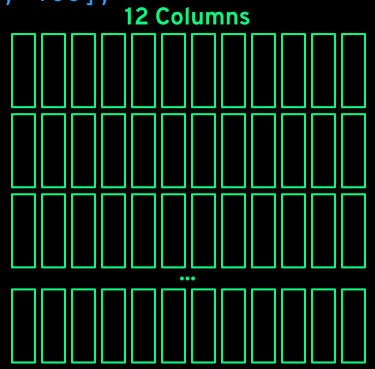
- Locks

  For view synchronization

- Globals

  Server URLs, editability

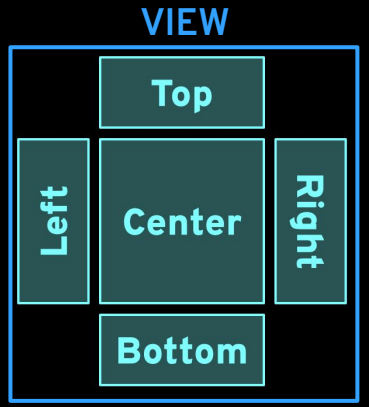
```
{
  "views": [
    {
      "uid": "aa",
      "initialXDomain": [0, 100],
      "initialYDomain": [0, 100],
      "layout": {
        "x": 0,  "y": 0,
        "w": 12, "h": 6,
      },
      "tracks": {
        "top": [],
        "left": [],
        "center": [],
        "right": [],
        "bottom": []
      }
    }
  ],
  "zoomLocks": { ... },
  "locationLocks": { ... },
  "valueScaleLocks": { ... },
  "editable": true,
  "zoomFixed": false,
  "trackSourceServers": ["/api/v1"],
  "exportViewUrl": "/api/v1/viewconfs"
}
```



View Config

- Views
  Shared location and zoom
  - Layout
  - Tracks
- Locks
  For view synchronization
- Globals
  Server URLs, editability

```
{
  "views": [
    {
      "uid": "aa",
      "initialXDomain": [0, 100],
      "initialYDomain": [0, 100],
      "layout": {
        "x": 0,  "y": 0,
        "w": 12, "h": 12,
      },
      "tracks": {
        "top": [],
        "left": [],
        "center": [],
        "right": [],
        "bottom": []
      }
    }
  ],
  "zoomLocks": { ... },
  "locationLocks": { ... },
  "valueScaleLocks": { ... },
  "editable": true,
  "zoomFixed": false,
  "trackSourceServers": ["/api/v1"],
  "exportViewUrl": "/api/v1/viewconfs"
}
```



Track Config

- **Type**
  general encoding
- **Server**
  track data source
- **Tileset UID**
  data identifier
- **UID**
  track identifier
- **Options**
  For styling, labeling, etc.

```
{
  "type": "horizontal-line",
  "server": "//higlass.io/api/v1",
  "tilesetUid": "OHJakQICQD6gTD7skx4EWA",
  "uid": "my-very-fancy-line-plot",
  "options": {
    "name": "My Very Fancy Line Plot!",
    ...
  },
}
```

Advanced Features

Advanced Features

(see nyc-taxi.ipynb)

Developer Features

Modular Codebase

Tiling, serving, and rendering are decoupled

E.g., viewer, server, tileset API, docker

Viewer Extensibility

Simple plugin architecture for new track types

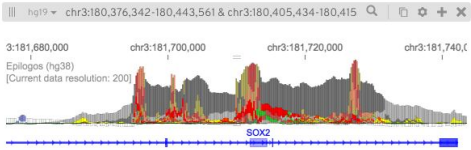
E.g., Epilogos, GeoJSON

JavaScript APIs for Integration

Library version, React component, JsAPI

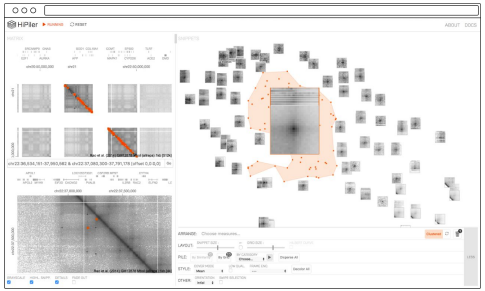
E.g., HiPiler, Peax, cTracks

EPILOGOS



GEOJSON

HIPILER



PEAX

Communication & Collaboration

(see genomics.ipynb)

HiGlass

To a Billion and Beyond

WEB: higlass.io

CODE: github.com/higlass/scipy19

TWITTER: @higlass_io

Scipy Slack Channel
#higlass

PRESENTERS:

Nezar Abdennur

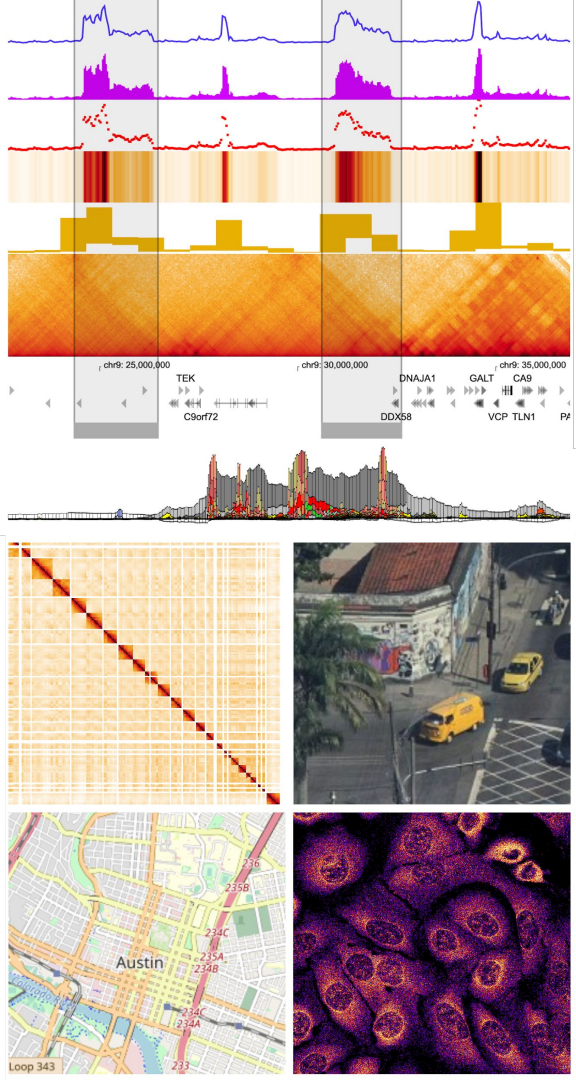
@nv1ctus
nvictus.me

Fritz Lekschas

@flekschas
lekschas.de

This work is supported in part by the National Institutes of Health (U01 CA200059)

4D Nucleome



HiGlass

To a Billion and Beyond

WEB: higlass.io

CODE: github.com/higlass/scipy19

TWITTER: @higlass_io

PRESENTERS:

Nezar Abdennur

@nv1ctus
nvictus.me

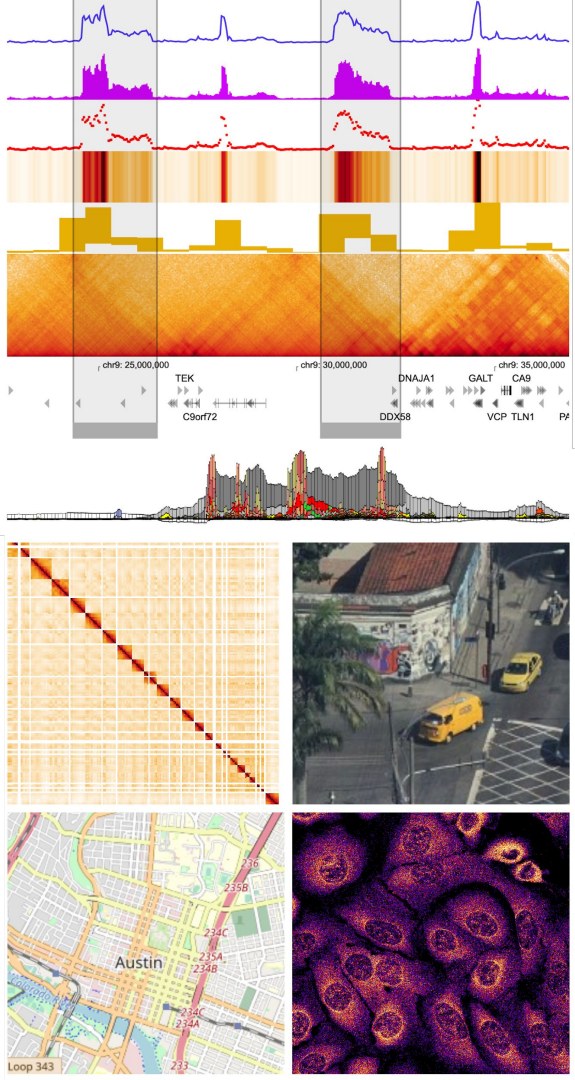
Fritz Lekschas

@flekschas
lekschas.de

This work is supported in part by the National Institutes of Health (U01 CA200059)

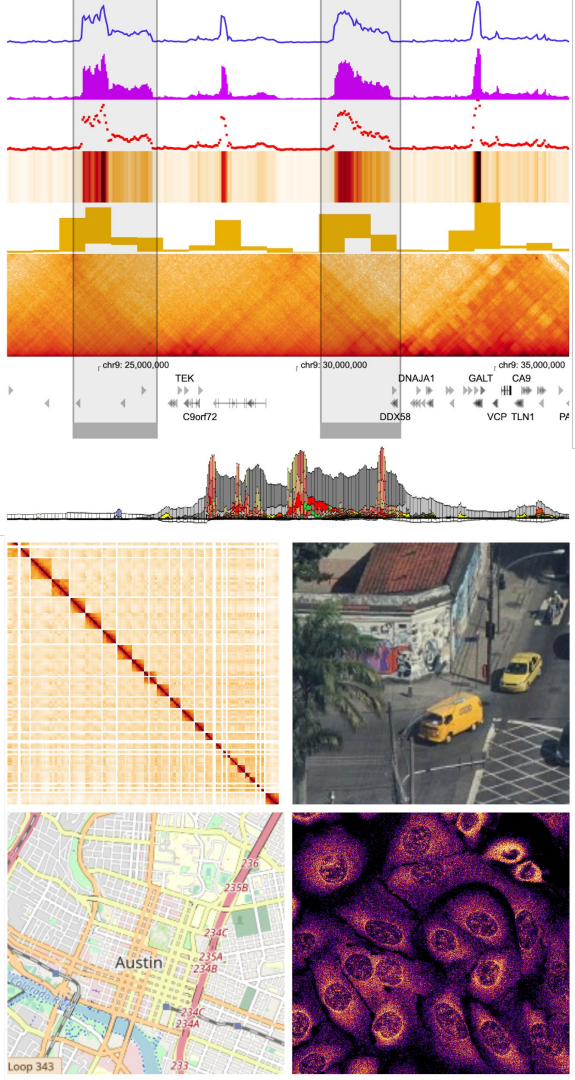
4D Nucleome

cooler



HiGlass

Contributors & Acknowledgements



Core Contributors:

Peter Kerpedjiev, Fritz Lekschas, Nezar Abdennur, Chuck McCallum

PIs:

Nils Gehlenborg, Peter Park, Leonid Mirny, Hanspeter Pfister

Co-Authors:

Kasper Dinkla, Hendrik Strobelt, Jacob Luber, Scott Ouellette, Alaleh Azhir, Nikhil Kumar, Jeewon Hwang, Soohyun Lee, Burak Alver

This work is supported in part by the National Institutes of Health (U01 CA200059)

4D Nucleome