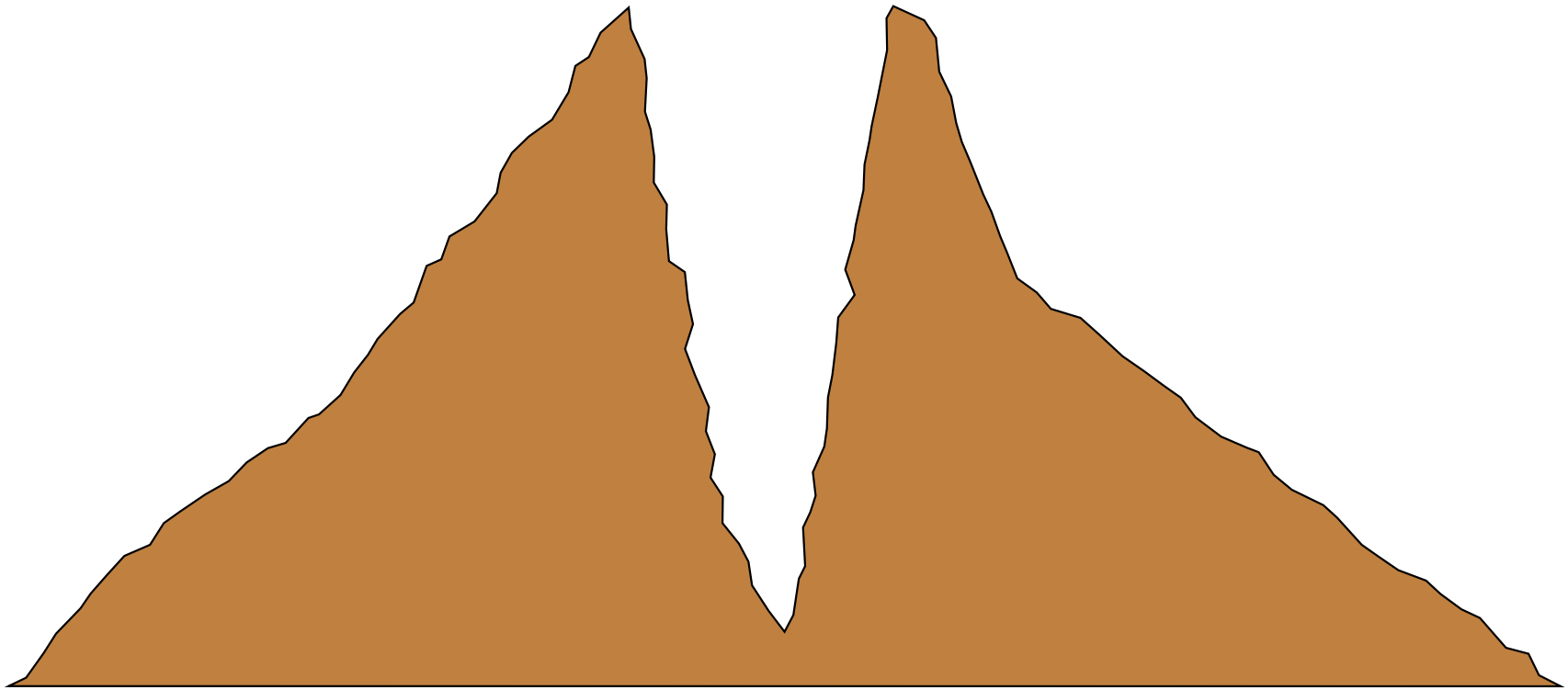
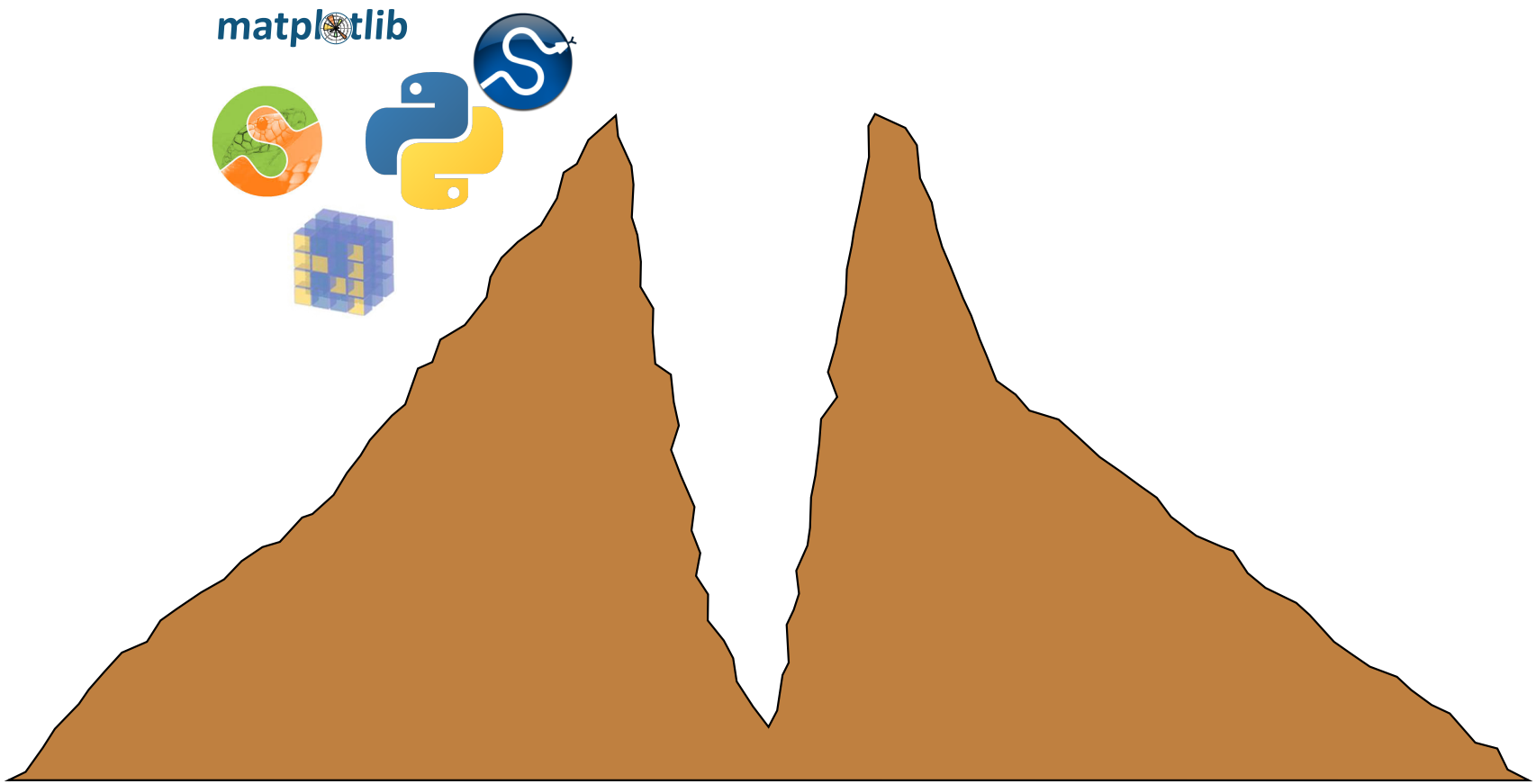
hhmi | janelia
Research Campus

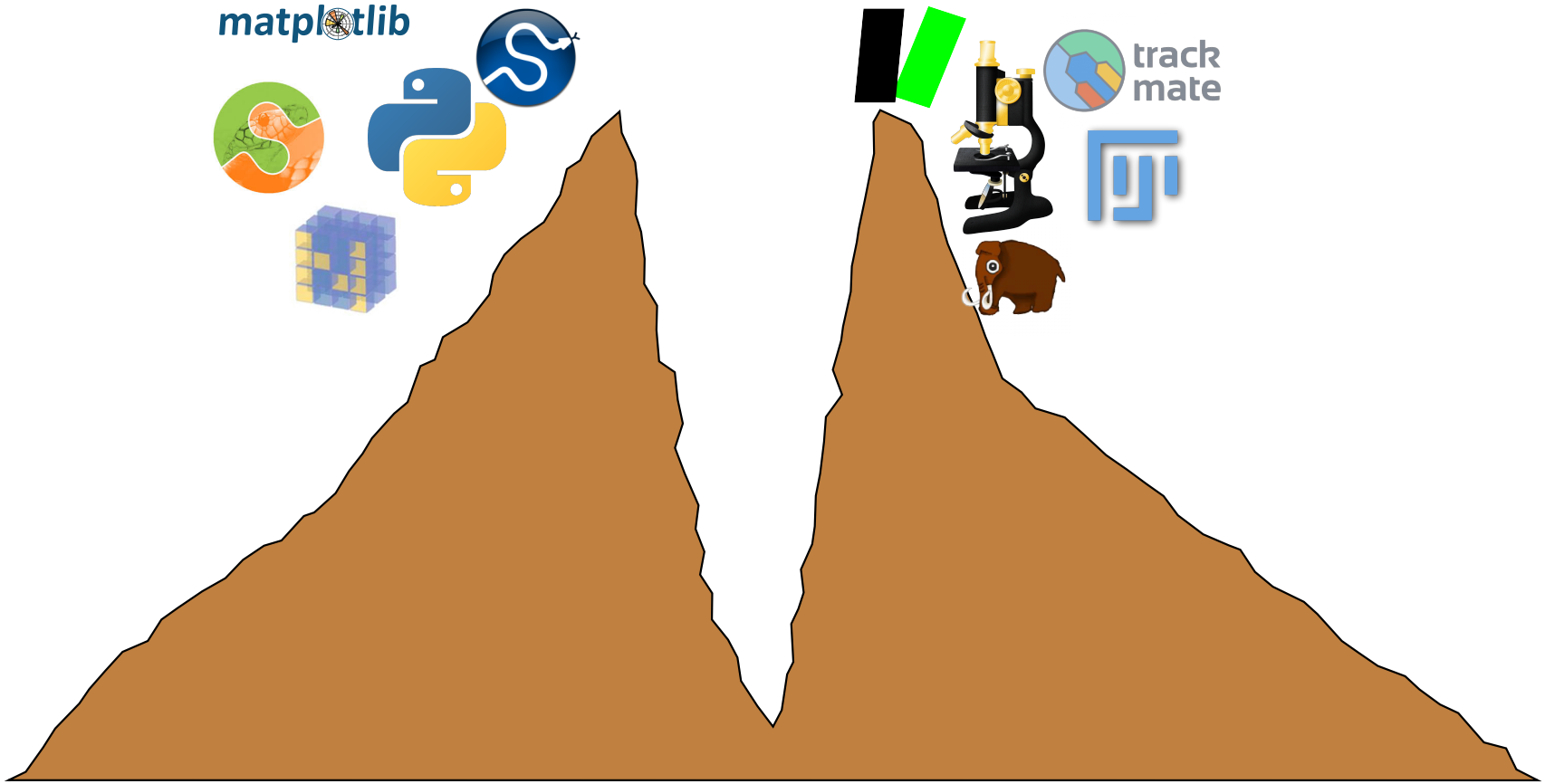
Philipp Hanslovsky

# imglyb

# Bridging The Chasm Between ImageJ and NumPy




matplotlib


matplotlib
track
mate

# Contents

1. Why is shared memory hard?
2. Solution
3. Examples
4. Known issues

# What is Python?

- Interpreted language with dynamic typing.

# What is Python?

- Interpreted language with dynamic typing.
- Access to native memory.

# What is Python?

- Interpreted language with dynamic typing.
- Access to native memory.
- Efficient software through C/C++ extensions.

# What is Python?

- Interpreted language with dynamic typing.
- Access to native memory.
- Efficient software through C/C++ extensions.
- Interactive shell and notebooks.

# What is Java?

- Statically typed language that compiles into byte code.

# What is Java?

- Statically typed language that compiles into byte code.
- Byte code is executed within a virtual machine (JVM).

# What is Java?

- Statically typed language that compiles into byte code.
- Byte code is executed within a virtual machine (JVM).
- No access to native memory through Java language API.

# Why are they hard to combine?

- Native vs JVM
- Java Array $\neq$ C-Array

# Ways out of the dilemma

- Inter-process communication:

# Ways out of the dilemma

- Inter-process communication:
  - ✓ Comparatively easy.
  - ✓ Can be extended to any language.
  - ✗ No shared memory.

# Ways out of the dilemma

- Inter-process communication:
  - ✓ Comparatively easy.
  - ✓ Can be extended to any language.
  - ✗ No shared memory.

- Python and JVM in the same process:

# Ways out of the dilemma

- Inter-process communication:
  - ✓ Comparatively easy.
  - ✓ Can be extended to any language.
  - ✗ No shared memory.

- Python and JVM in the same process:
  - ✓ Shared memory possible.
  - ✓ Avoids unnecessary copies of data.

# Ways out of the dilemma

- Inter-process communication:
  - ✓ Comparatively easy.
  - ✓ Can be extended to any language.
  - ✗ No shared memory.

- Python and JVM in the same process:
  - ✓ Shared memory possible.
  - ✓ Avoids unnecessary copies of data.

# Two options

- Java implementation of Python: Jython

# Two options

- Java implementation of Python: Jython
  - ✓ Access native CPython extensions through Jython Native Interface

# Two options

- Java implementation of Python: Jython
  - ✓ Access native CPython extensions through Jython Native Interface
  - ✗ Only Jython 2.7 is available

# Two options

- Java implementation of Python: Jython
  - ✓ Access native CPython extensions through Jython Native Interface
  - ✗ Only Jython 2.7 is available
- Start Java within a CPython process

# Way out!

- ✓ Start JVM from Python process.

  `https://github.com/kivy/pyjnius`

# Way out!

- ✓ Start JVM from Python process.

  `https://github.com/kivy/pyjnius`

  Make ImgLib2 and NumPy understand each other.

# Way out!

- ✓ Start JVM from Python process.

  `https://github.com/kivy/pyjnius`

- ✓ Make ImgLib2 and NumPy understand each other.

  `https://github.com/imglib/imglib2-unsafe`

# Way out!

- ✓ Start JVM from Python process.

  `https://github.com/kivy/pyjnius`

- ✓ Make ImgLib2 and NumPy understand each other.

  `https://github.com/imglib/imglib2-unsafe`
  `https://github.com/imglib/imglib2-imglyb`

# Way out!

- ✓ Start JVM from Python process.

  `https://github.com/kivy/pyjnius`

- ✓ Make ImgLib2 and NumPy understand each other.

  `https://github.com/imglib/imglib2-unsafe`
  `https://github.com/imglib/imglib2-imglyb`

  `https://github.com/imglib/imglyb`

# Way out!

- ✓ Start JVM from Python process.

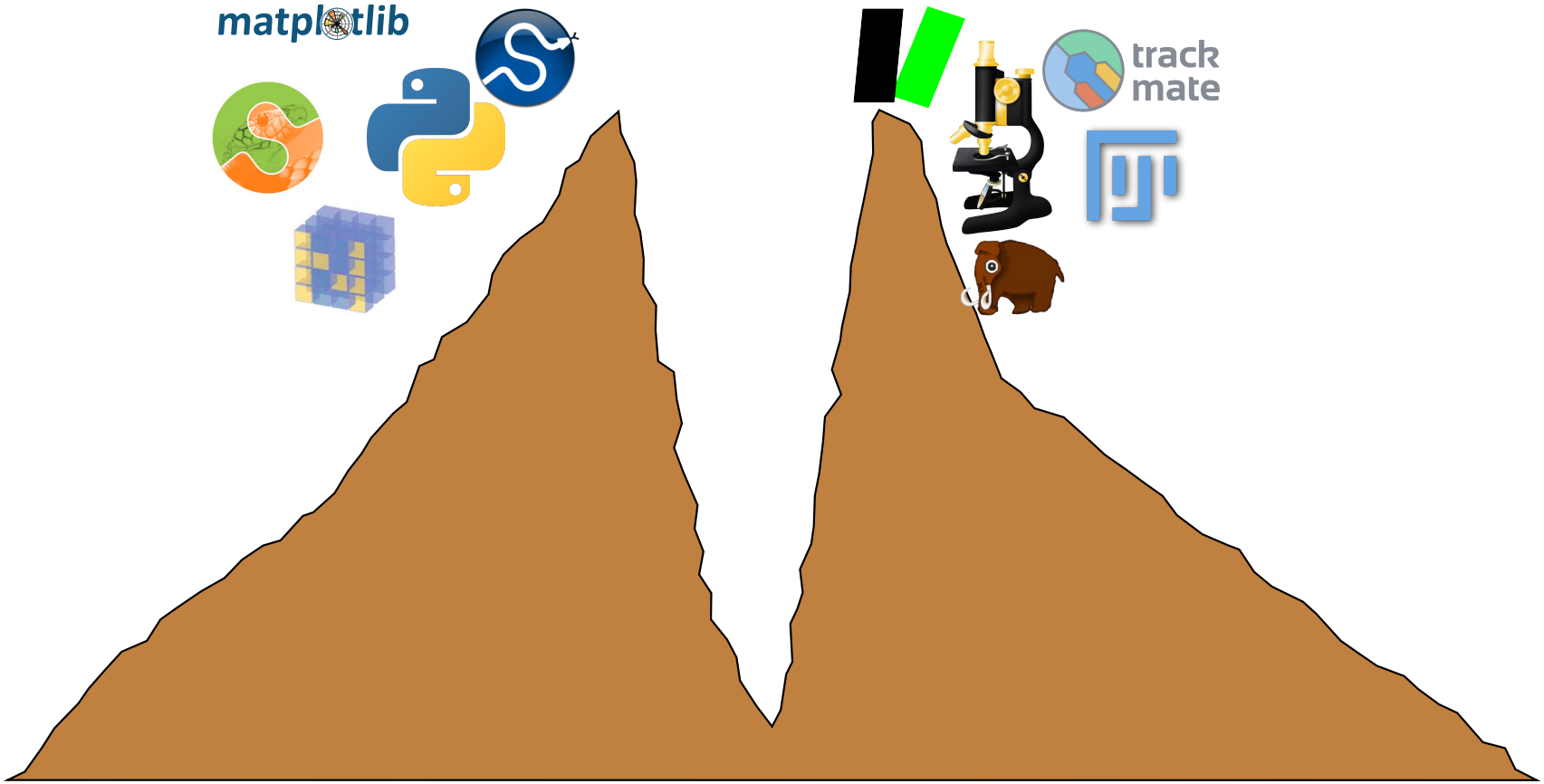
  `https://github.com/kivy/pyjnius`

- ✓ Make ImgLib2 and NumPy understand each other.

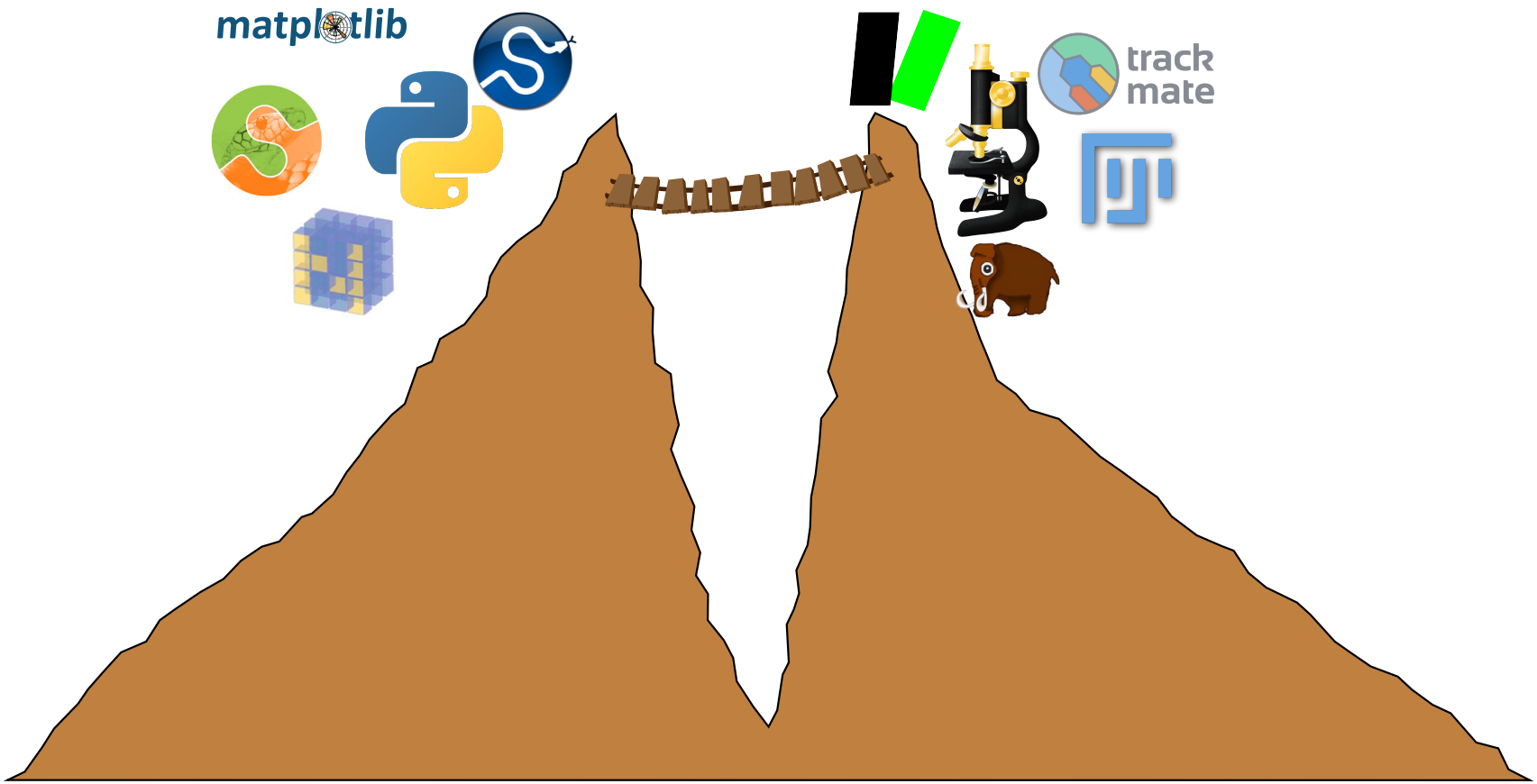
  `https://github.com/imglib/imglib2-unsafe`
  `https://github.com/imglib/imglib2-imglyb`
  `https://github.com/imglib/imglyb`

My contribution

# imglyb



# imglyb



# Installation & Usage

- Install imglyb from conda

```
conda install -c conda-forge imglyb
```

# Installation & Usage

- Install imglyb from conda

  ```
  conda install -c conda-forge imglyb
  ```

- Install examples from PyPI

  ```
  pip install imglyb-examples
  ```

# Installation & Usage

- Install imglyb from conda

  ```
  conda install -c conda-forge imglyb
  ```

- Install examples from PyPI

  ```
  pip install imglyb-examples
  ```

- Notebooks available on

  https://github.com/hanslovsky/imglyb-learnathon

# How to use?

Import `imglyb`

```
# import imglyb before jnius
import imglyb
from imglyb import util
# import from jnius what you need
from jnius import autoclass, cast, PythonJavaClass, java_method
```

# Wrap NumPy arrays in ImgLib2

```
import imglyb
from imglyb import util

import numpy as np

img = np.random.rand( 300, 200, 100 ) * 2**16
wrapped = util.to_imglib( img )
util.BdvFunctions.show( wrapped, ”wrapped image” )

rgba = np.random.randint(
    2**32, size = ( 300, 200, 100 ),
    dtype=np.uint32 )
wrapped_rgba = util.to_imglib_argb( rgba )
util.BdvFunctions.show( wrapped_rgba, ”wrapped rgba image” )
```

# Examples

Available in the imglyb-examples package:

```
imglyb-examples.bdv-hello-world
imglyb-examples.bdv-painter
imglyb-examples.butterfly
imglyb-examples.qt-awt
imglyb-examples.views-stack
```

# Example 1 — BigDataViewer

BigDataViewer (BDV) is a Java viewer for arbitrarily large multi-view 3D images and 3D image sequences developed by Tobias Pietzsch.

https://imagej.net/BigDataViewer

https://github.com/hanslovsky/imglyb-learnathon/blob/master/notebooks/bdv/show-numpy-array-in-bdv.ipynb

# Example 2 — BigDataViewer

BigDataViewer (BDV) is a Java viewer for arbitrarily large multi-view 3D images and 3D image sequences developed by Tobias Pietzsch.

https://imagej.net/BigDataViewer

https://github.com/hanslovsky/imglyb-learnathon/blob/master/notebooks/bdv/write-into-numpy-array-in-bdv.ipynb

# Example 3 — BigWarp

BigWarp is a tool for interactive manual alignment of 2D or 3D images developed by John Bogovic.

https://imagej.net/BigWarp
https://github.com/hanslovsky/imglyb-learnathon/blob/master/notebooks/bigwarp/bigwarp.ipynb

# Example 4 — Paintera

Paintera is a tool for the fast generation of dense ground truth annotations and proof-reading 3D EM connectomics with 3D visualization and mesh generation on the fly.

https://github.com/saalfeldlab/paintera
https://github.com/hanslovsky/imglyb-learnathon/blob/master/notebooks/paintera/paintera-mesh-generation-on-the-fly.ipynb

# Known Issues

- Java awt requires wrapper script on OSX
- Dask Arrays cannot be wrapped into ImgLib2 cell images

# Resources

https://www.python.org

https://www.numpy.org

https://scipy.org

https://scikit-image.org

https://matplotlib.org

https://imagej.net/ImageJ2

https://imagej.net/ImgLib2

https://imagej.net/TrackMate

https://fiji.sc/

https://imagej.net/MaMuT

https://openclipart.org/detail/209218/firebog-bridge

# Thank You!

```
conda install -c conda-forge imglyb
pip install imglyb-examples
```

https://github.com/imglib/imglib2-unsafe

https://github.com/imglib/imglib2-imglyb

https://github.com/imglib/imglyb

https://github.com/hanslovsky/imglyb-examples

https://github.com/hanslovsky/imglyb-learnathon

https://github.com/hanslovsky/scipy-2019