ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Funded by

Bundesministerium für Bildung und Forschung

SACHSEN

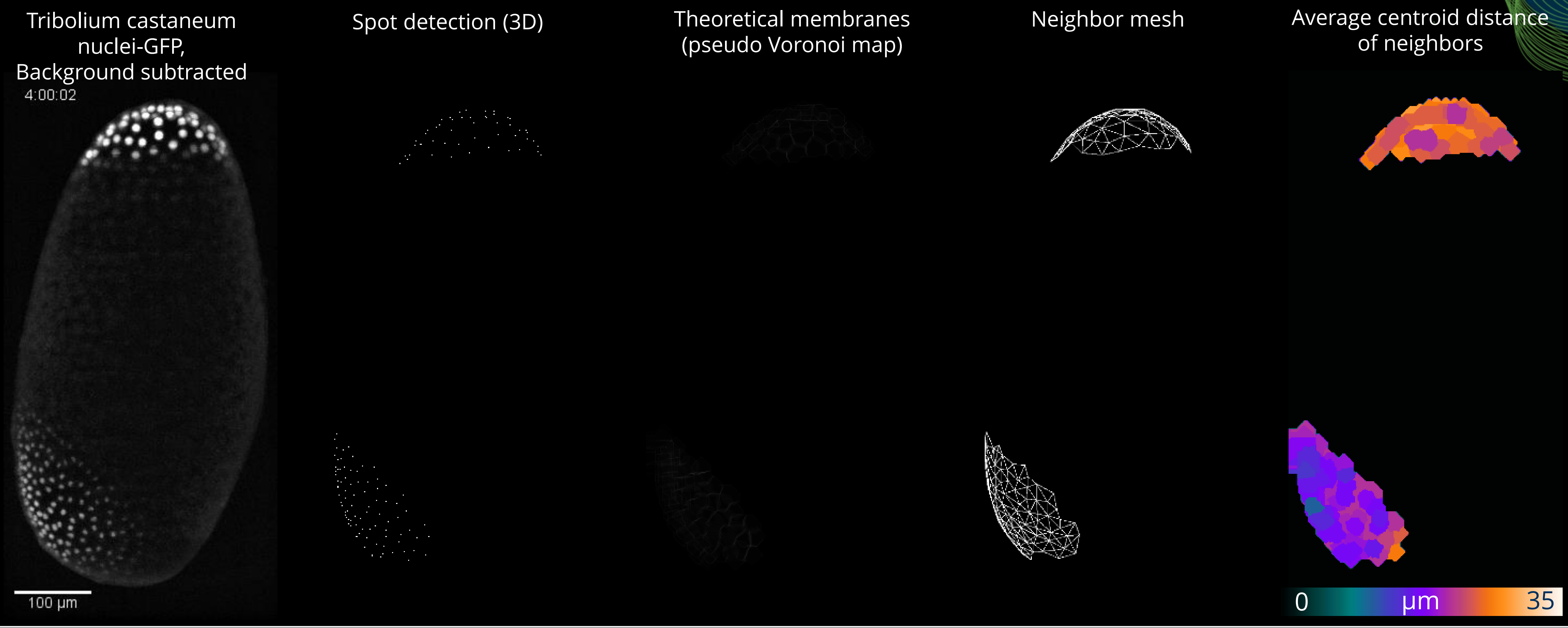
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Distributed & GPU-accelerated Image Processing

Robert Haase

GPU accelerated image processing in life sciences

… to study embryo development



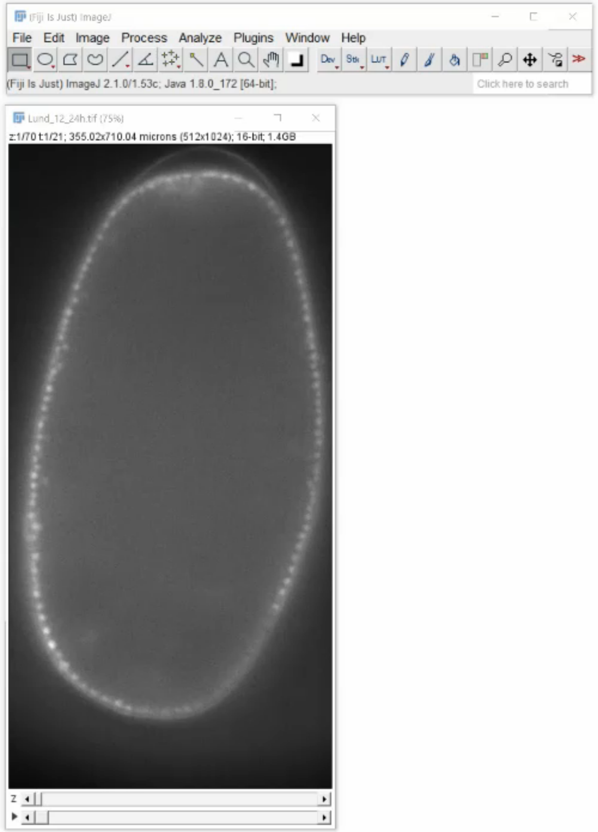
GPU accelerated image processing in life sciences

Raytracing enables differentiating surface and sub-surface mesh nodes

3D stack input | Neighbor mesh | Surface neighbor mesh | Sub-surface neighbor mesh | Merge

Image processing in life-sciences

- State-of-the-art software for more than 20 years: ImageJ / Fiji



2x

Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD

OpenCL-based GPU-acceleration

GPU-acceleration? Learn the Open Computing Language (OpenCL)!

```
__kernel void maximum_z_projection(
    IMAGE_dst_max_TYPE dst_max,
    IMAGE_src_TYPE src
) {
  const sampler_t sampler = CLK_NORMALIZED_COORDS_FALSE | CLK_ADDRESS_CLAMP_TO_EDGE | CLK_FILTER_NEAREST;

  const int x = get_global_id(0);
  const int y = get_global_id(1);
  float max = 0;
  for(int z = 0; z < GET_IMAGE_DEPTH(src); z++)
  {
    POS_src_TYPE pos = POS_src_INSTANCE(x,y,z,0);
    float value = READ_src_IMAGE(src,sampler,pos).x;
    if (value > max || z == 0) {
      max = value;
    }
  }
  POS_dst_max_TYPE pos = POS_dst_max_INSTANCE(x,y,0,0);
  WRITE_dst_max_IMAGE(dst_max, pos, CONVERT_dst_max_PIXEL_TYPE(max));
}
```

Maximum intensity projection along Z

OpenCL: https://www.khronos.org/opencl/

Kernel collection: https://github.com/clEsperanto/clij-opencl-kernels

User-friendly GPU-acceleration

2x

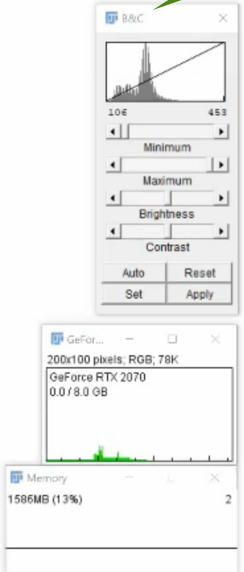


Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD

GPU-accelerated image processing

Performance depends on operation, image size, parameters, hardware, ….

vs.

Workstation CPU

2x Intel Xeon Silver 4110

Workstation GPU

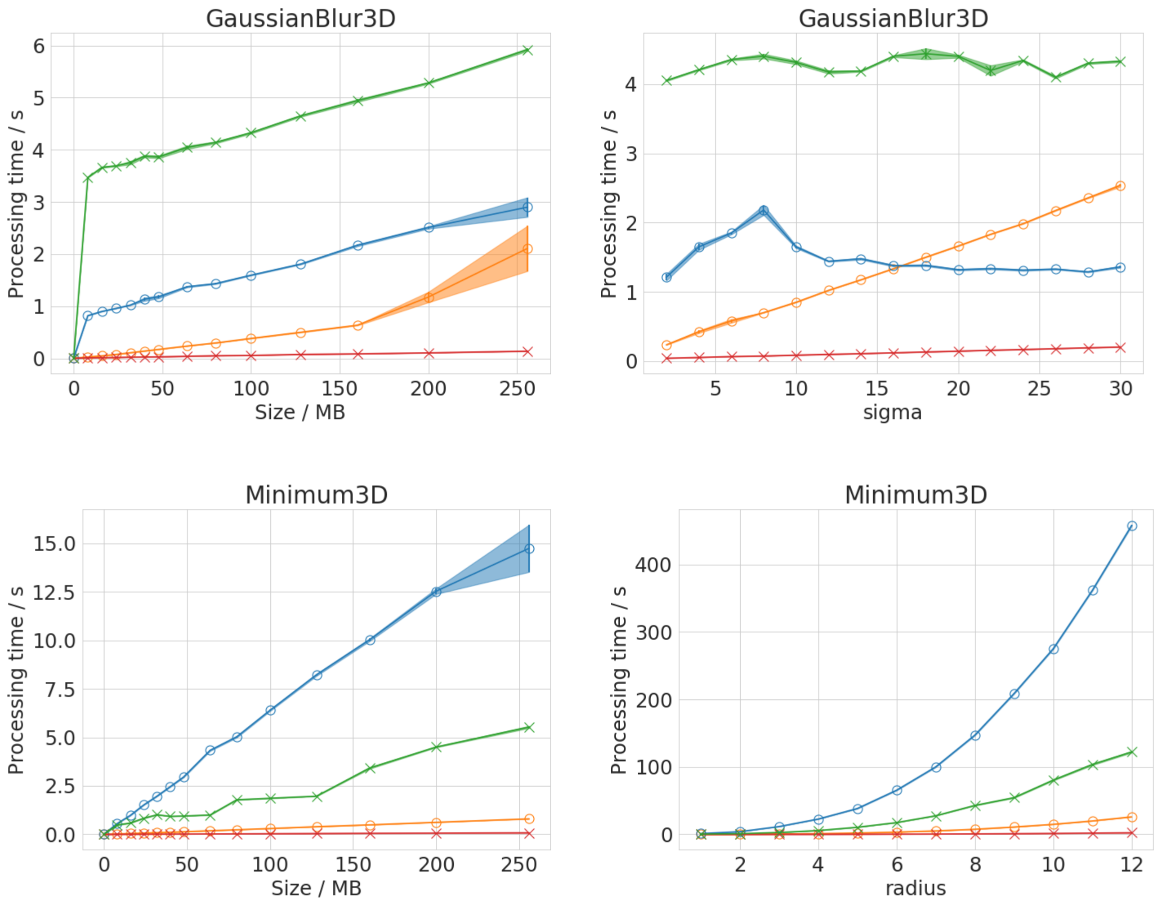
Nvidia Quadro P6000

Laptop CPU

Intel Core i7-8650U

Laptop GPU

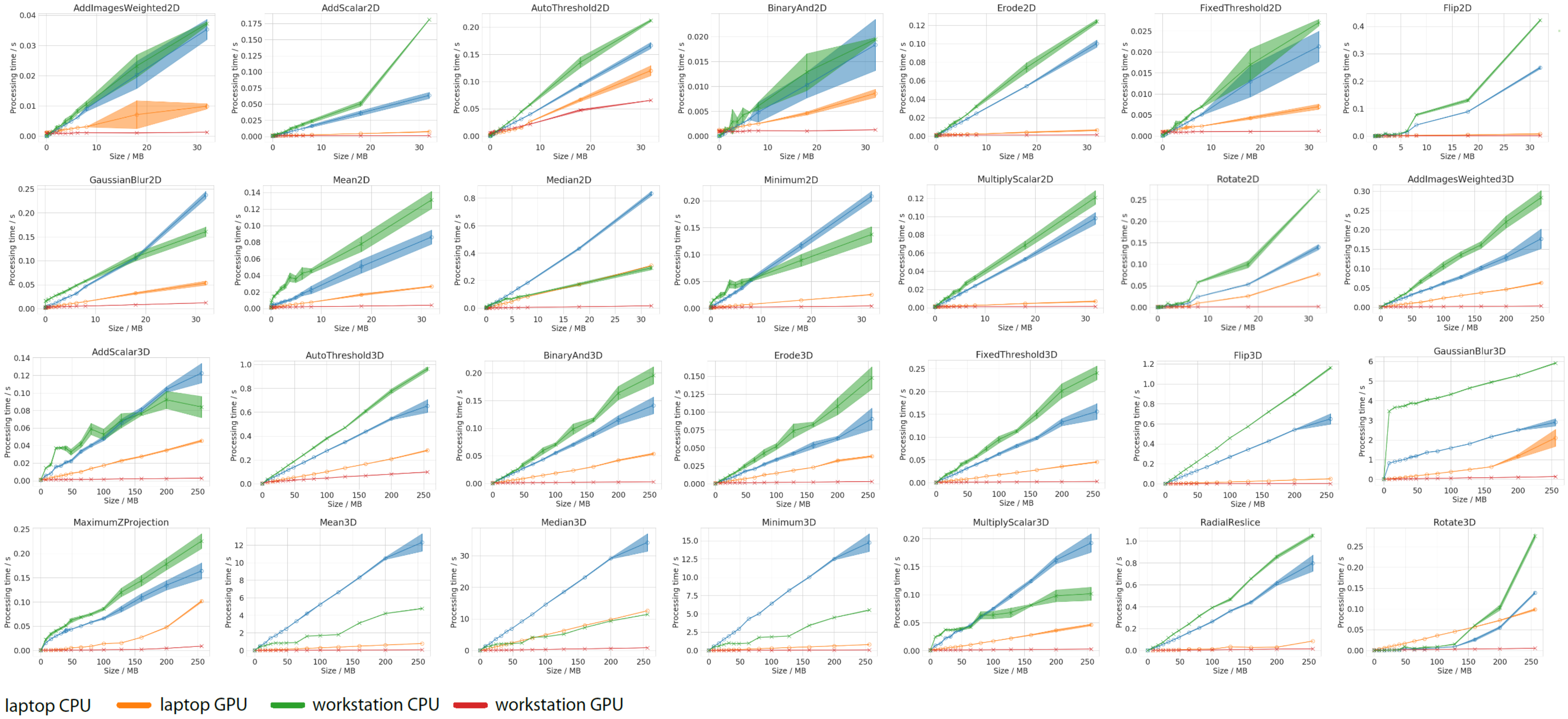
Intel UHD 620 GPU



Haase et al. Nat Methods (2020), https://www.nature.com/articles/s41592-019-0650-1

GPU-accelerated image processing

Performance depends on operation, image size, parameters, hardware, ….



Haase et al. Nat Methods (2020),
https://www.nature.com/articles/s41592-019-0650-1

GPU-accelerated image processing

- 8 MB (2D)
- 64 MB (3D)

Speedup compared to Laptop CPU

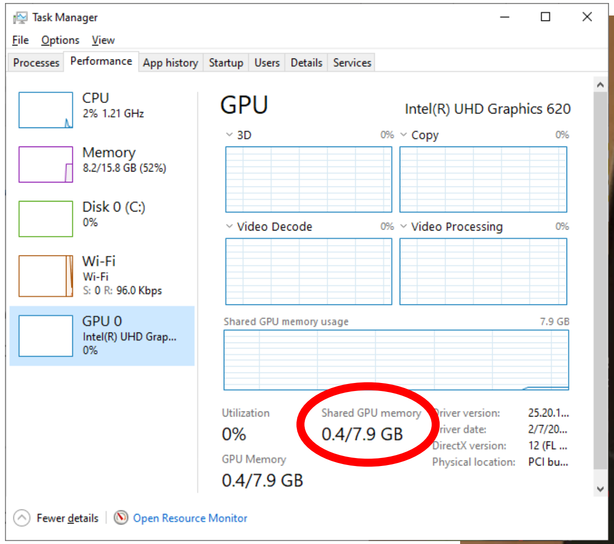
	Laptop GPU	Workstation GPU
AddImagesWeighted2D	3	8
AddScalar2D	7	14
AutoThreshold2D	2	2
BinaryAnd2D	2	4
Erode2D	11	20
FixedThreshold2D	2	5
Flip2D	16	37
GaussianBlur2D	3	9
Mean2D	3	10
Median2D	2	35
Minimum2D	7	22
MultiplyScalar2D	10	21
Rotate2D	3	22
AddImagesWeighted3D	3	26
AddScalar3D	3	23
AutoThreshold3D	3	5
BinaryAnd3D	3	24
Erode3D	2	13
FixedThreshold3D	4	30
Flip3D	15	119
GaussianBlur3D	6	35
MaximumZProjection	7	46
Mean3D	18	150
Median3D	3	43
Minimum3D	23	188
MultiplyScalar3D	4	28
RadialReslice	14	42
Rotate3D	0.1	2

Haase et al. Nat Methods (2020), https://www.nature.com/articles/s41592-019-0650-1

Checklist: When does GPU-accelerated image processing make sense?

In order to accelerate your image analysis workflow

- The pre-existing workflow should be slow; ideally: (processing time / loading time) > 10,
- a single processing step (rule of thumb: image size in GB x4) should fit in your graphics card memory,



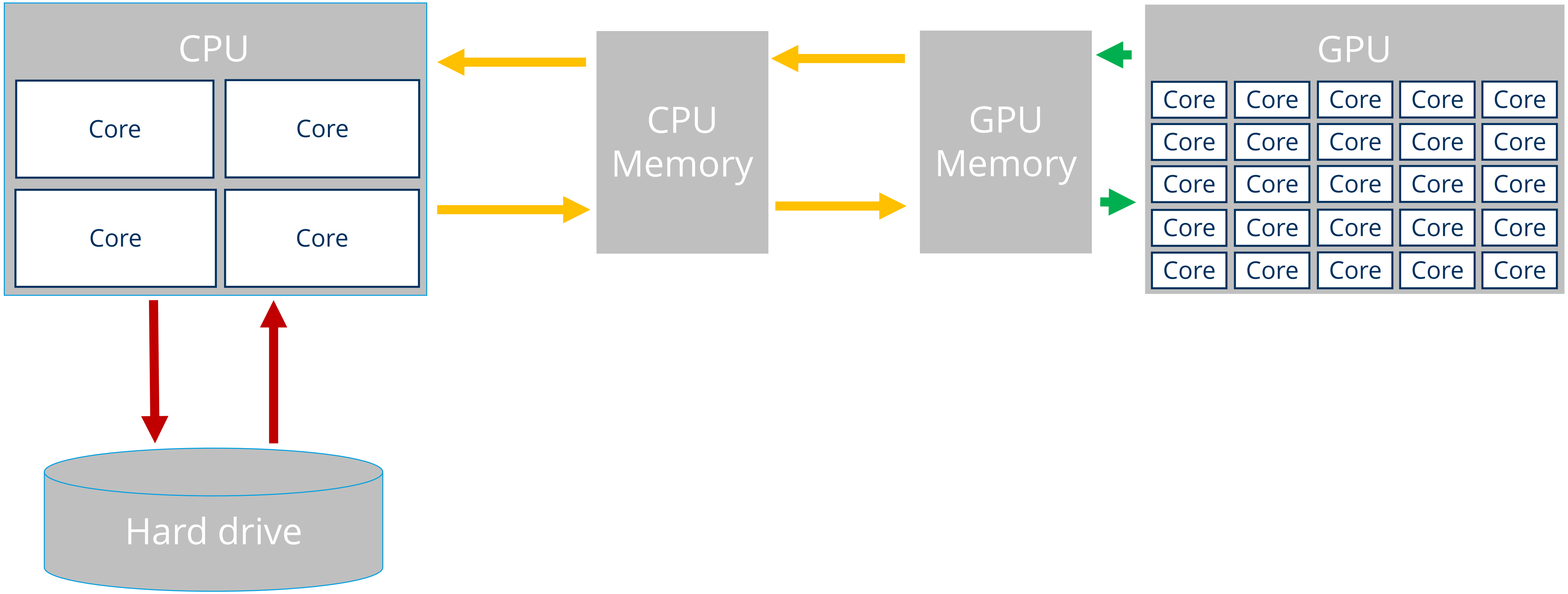
If you really want to get the most out of it, you should own a graphics card with GDDR6 memory (memory bandwidth > 400 Gb/s).

Comparison: common DDR4 memory has a bandwidth of about 40 GB/s

Disclosure: I don't receive any money or anything from any GPU vendor.

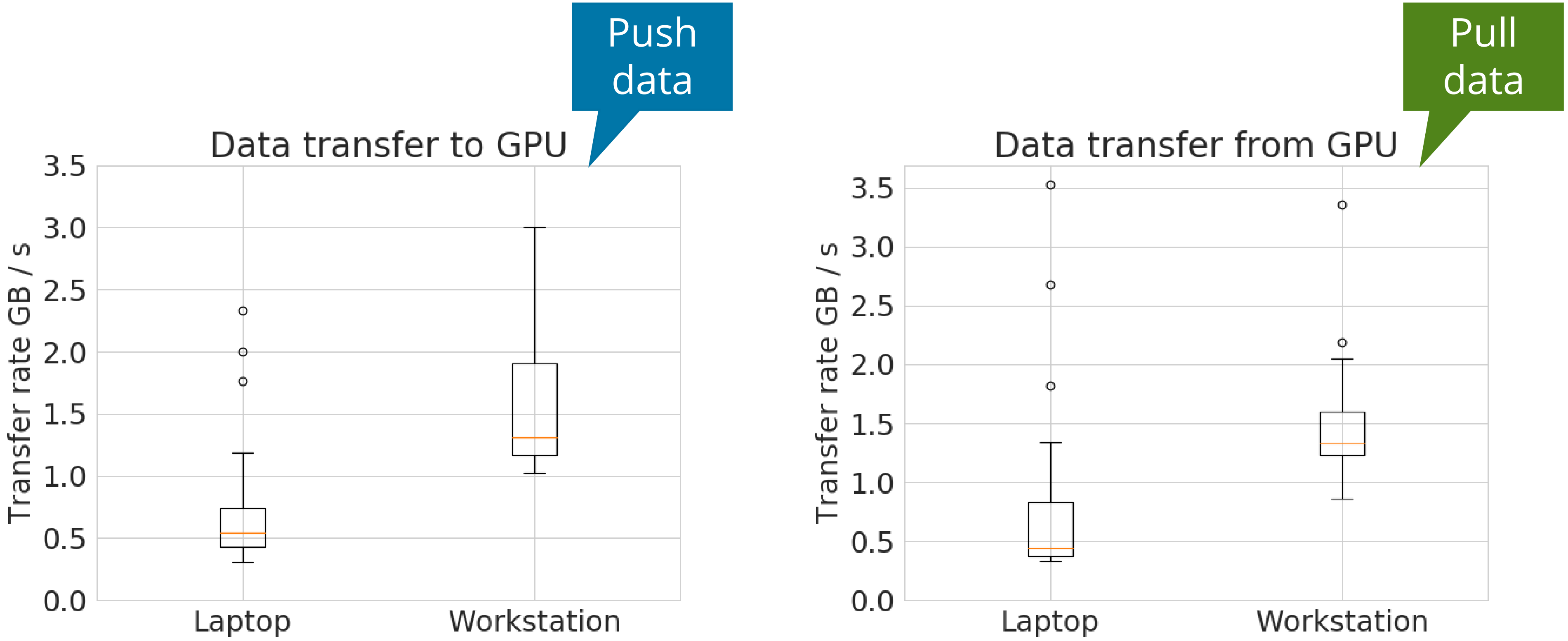
GPUs allow real-time image processing

GPUs are specialised in processing, very fast thanks to many cores and fast memory access



Data transfer takes time

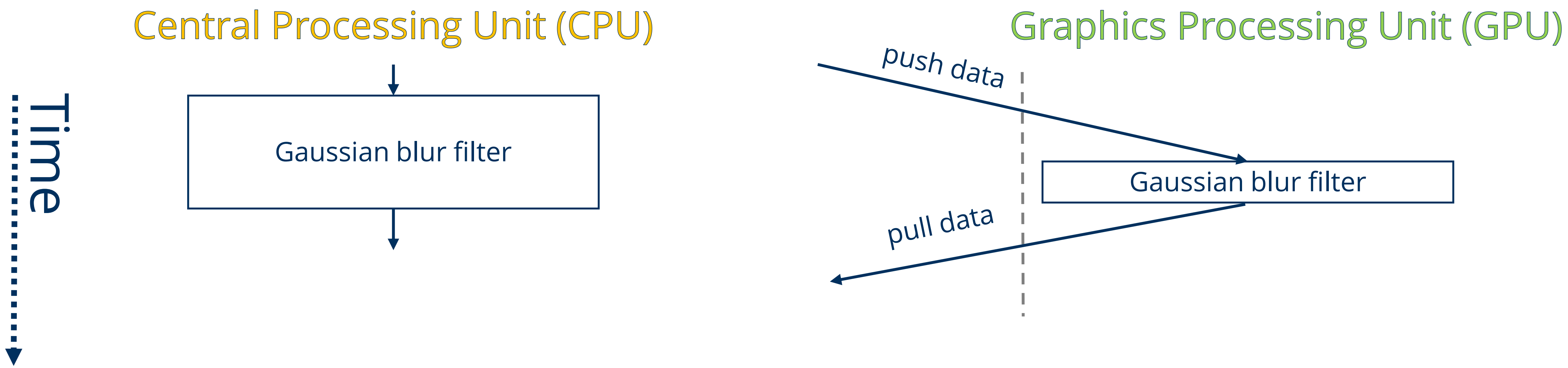
Data transfer is the bottle neck



https://clij.github.io/clij-benchmarking/benchmarking_operations_jmh

Build workflows consisting of many operations

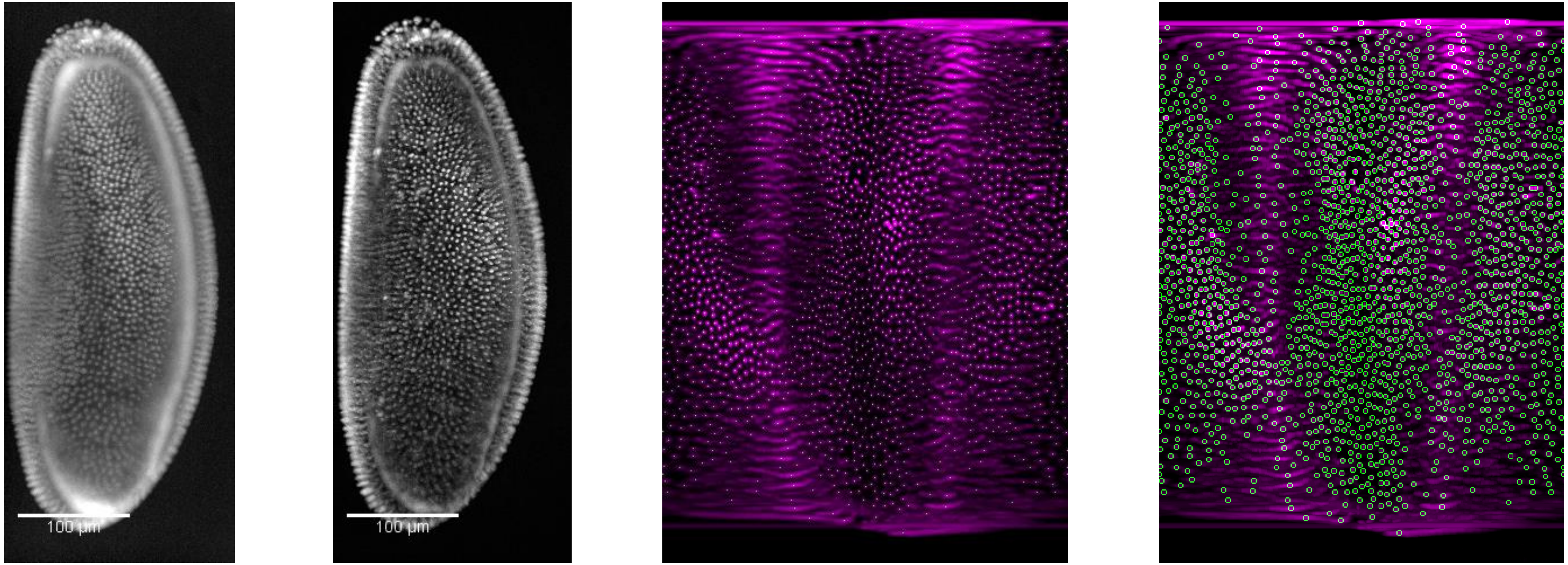
GPU acceleration may suffer from data transfer between CPU and GPU



Optimal performance through smart memory management

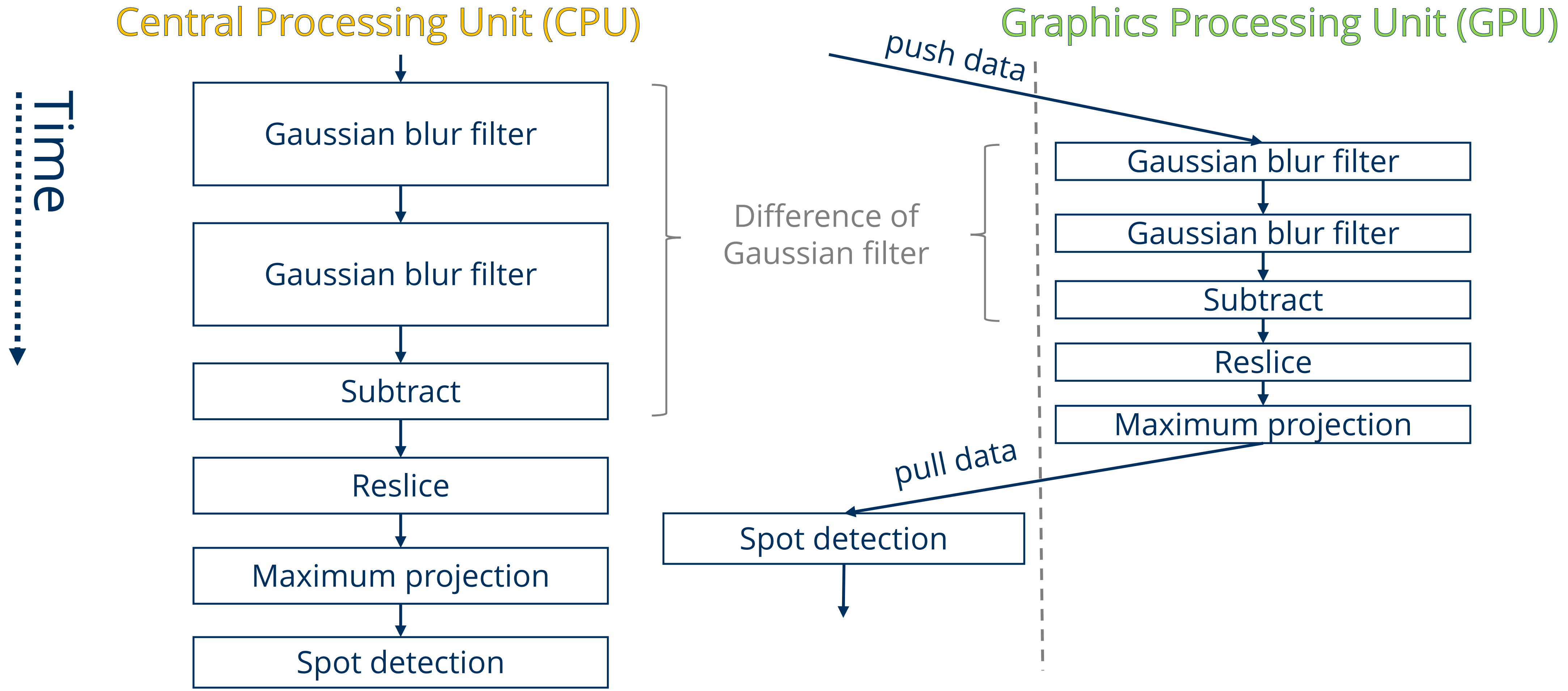
Example workflow processing a Drosophila melanogaster embryo, histone-GFP

Load data → Preprocessing → Transformation → Segmentation → Save data



Build workflows consisting of many operations

GPU acceleration may suffer from data transfer between CPU and GPU



CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

GPU-accelerated Image Processing in Python: OpenCL / clesperanto

Robert Haase

Funded by

Bundesministerium für Bildung und Forschung

SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Python Jupyter Notebooks

PoL
Physics of Life
TU Dresden

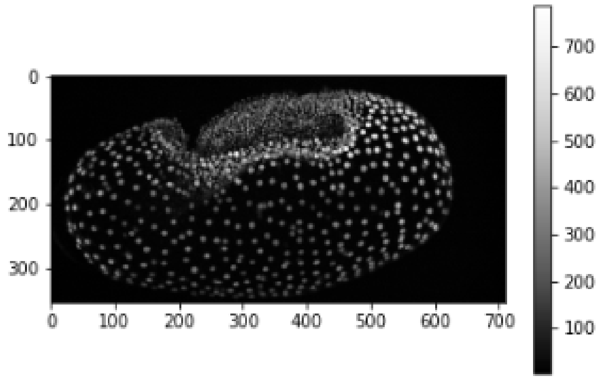
- Image processing using pyclesperanto

[1]:
```python
import stackview
import pyclesperanto_prototype as cle
from skimage.io import imread
```

[2]:
```python
image = imread("c:/structure/data/Lund_18.0_22.0_Hours-11-resampled.tif").swapaxes(1,2)
```

[3]:
```python
background_subtracted = cle.top_hat_box(image, radius_x=5, radius_y=5)
background_subtracted
```

[3]:

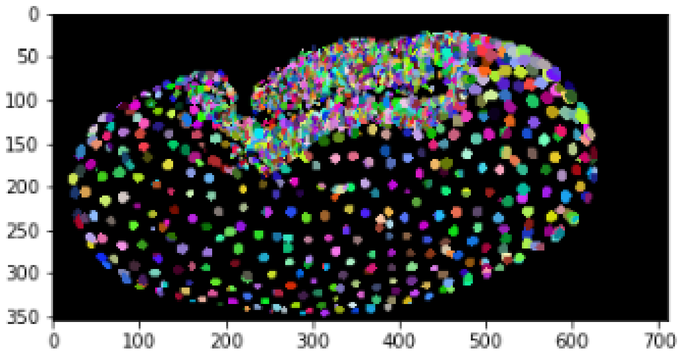


cle._ image

shape	(140, 355, 710)
dtype	float32
size	134.6 MB
min	0.0
max	790.0

[4]:
```python
nuclei = cle.voronoi_otsu_labeling(background_subtracted, spot_sigma
nuclei
```

[4]:

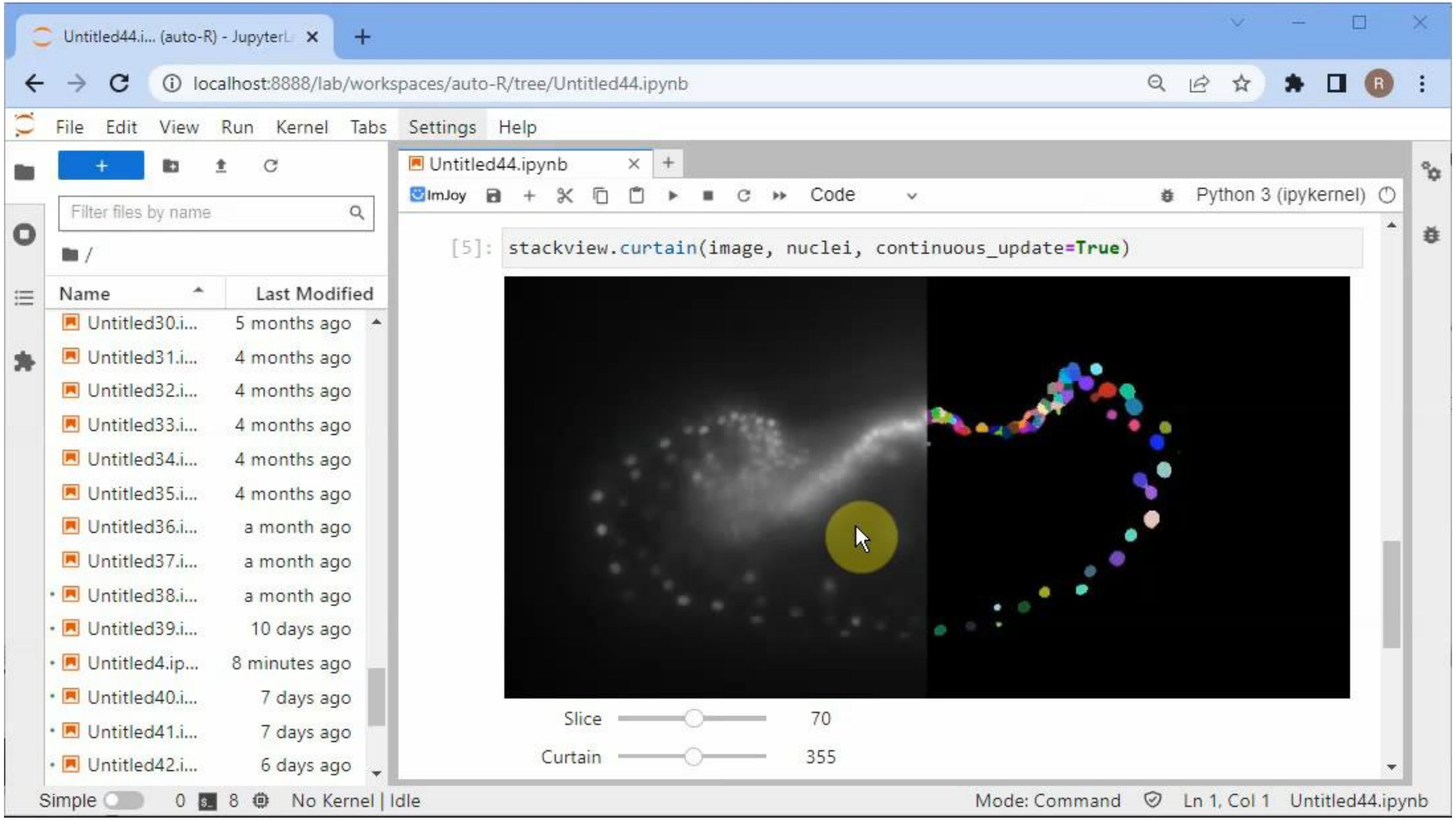


cle._ image

shape	(140, 355, 710)
dtype	uint32
size	134.6 MB
min	0.0
max	1756.0

Python Jupyter Notebooks

- When working on the cluster / Jupyter Hub, consider using stackview instead of napari for inspecting images in 3D.



ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

GPU-accelerated Image Processing in Python: CUDA / cupy

Robert Haase

Funded by

Bundesministerium für Bildung und Forschung

SACHSEN

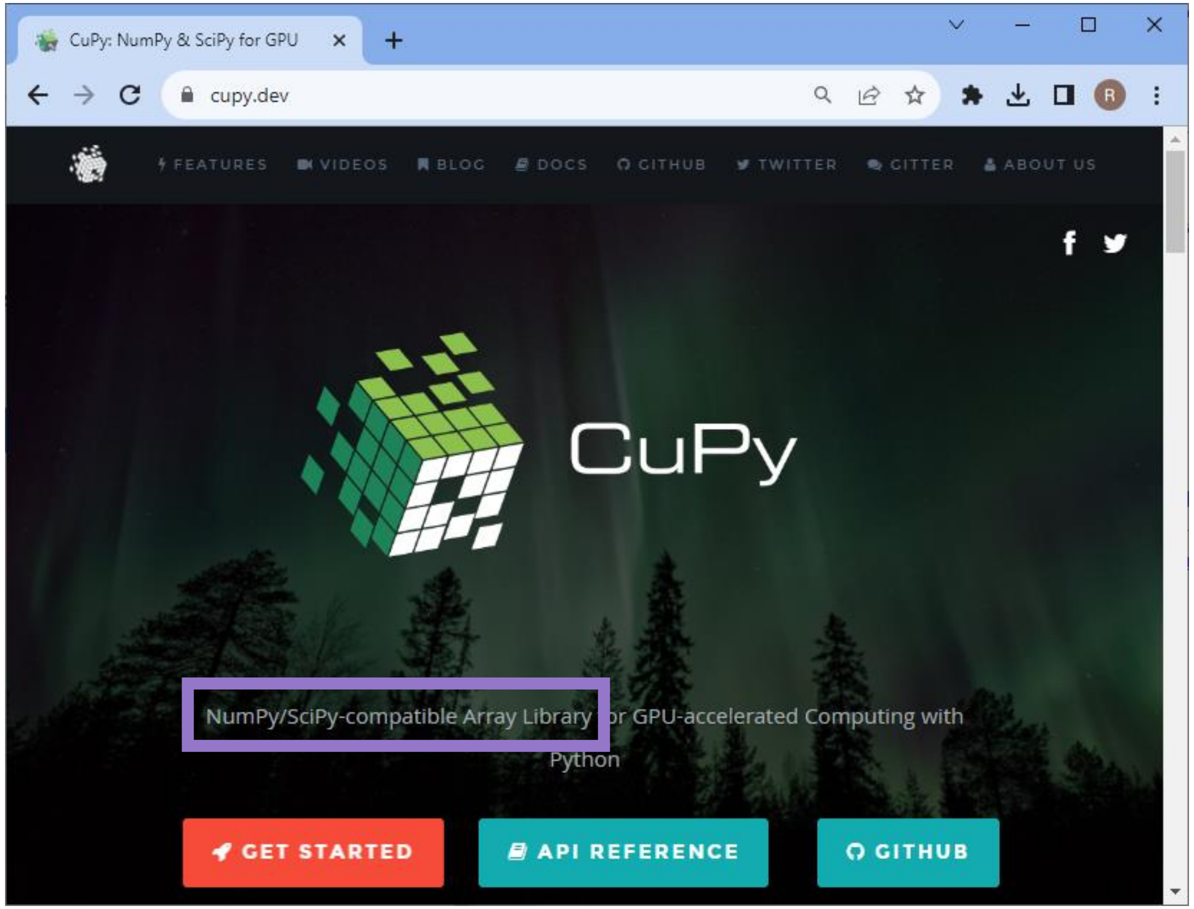
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

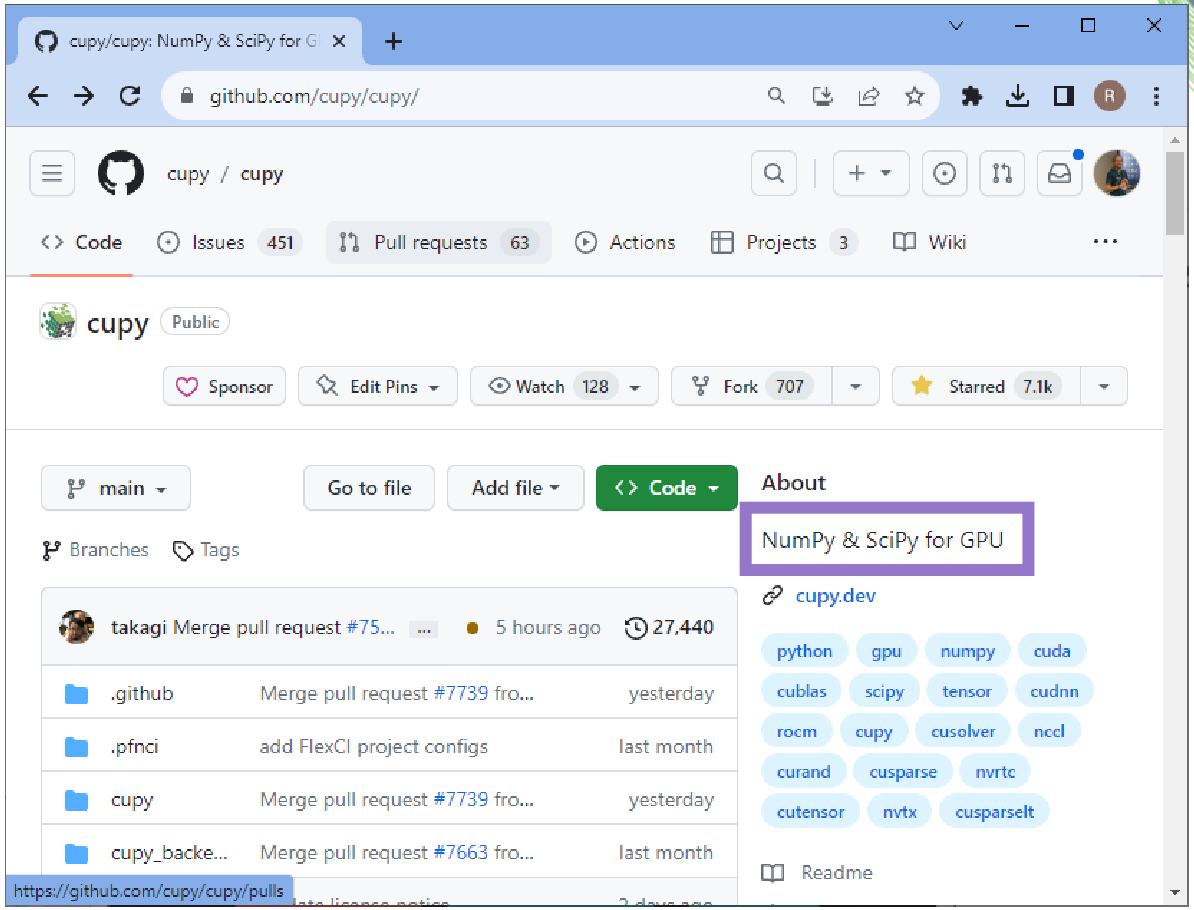
TECHNISCHE UNIVERSITÄT DRESDEN

UNIVERSITÄT LEIPZIG

cupy

CUDA-based GPU-accelerated [image] data processing in Python





drop-in replacement for numpy and scipy

The API of some cupy packages is close to the scipy/numpy API.

This allows *easy* switching from scipy to cupy.

```
import scipy.ndimage as ndi
```

```
import cupyx.scipy.ndimage as xdi
```

- However, image data still needs to be pushed to GPU memory.

```
xp_image = xp.asarray(image)
```

```
ndi.gaussian_filter(image, sigma=5)
```

```
xdi.gaussian_filter(xp_image, sigma=5)
```

Common patterns

To make code independent from cupy availablity, while minimizing if-else blocks, some common design patterns emerged:

```
try:
    import cupy as xp
except:
    import numpy as xp

import numpy as np
```

If this fails because cupy is not installed,

This will execute and xp will be available.

We can still use np anyway.

The same pattern works with scipy.ndimage

```
try:
    import cupyx.scipy.ndimage as xdi
except:
    import scipy.ndimage as xdi

import scipy.ndimage as ndi
```

Common patterns

You can then call magic code like this, which will do different things depending on cupy-availability.

- If cupy is available:

```
[3]: image = imread("../../data/blobs.tif")

     xp_image = xp.asarray(image)

     type(xp_image)

[3]: cupy.ndarray
```

- If cupy is not available:

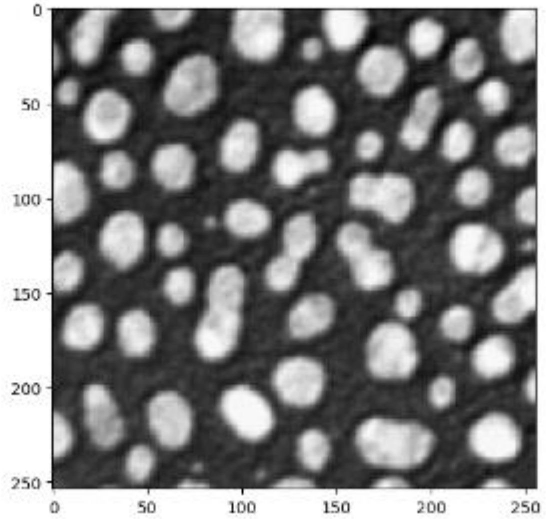
```
[3]: image = imread("../../data/blobs.tif")

     xp_image = xp.asarray(image)

     type(xp_image)

[3]: numpy.ndarray
```

Common patterns

Some if-else blocks are hard to avoid

[7]:
```
if np == xp:
    np_image = xp_image
else:
    np_image = xp.asnumpy(xp_image)

imshow(np_image)
```

[7]: <matplotlib.image.AxesImage at 0x24692ef85e0>

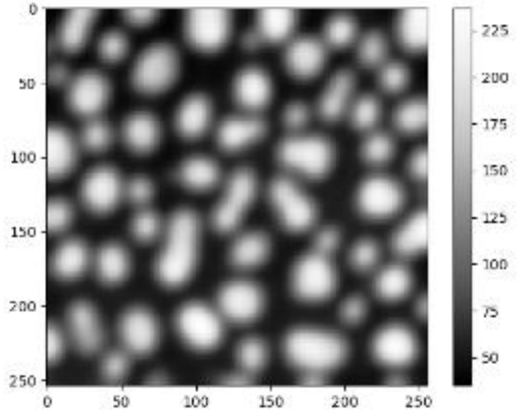


- Stackview aims to be cupy/numpy agnostic

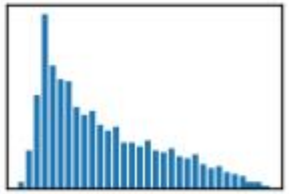
[4]:
```
xp_blurred = xdi.gaussian_filter(xp_image, sigma=5)
```

[5]:
```
stackview.insight(xp_blurred)
```

[5]:



shape	(254, 256)
dtype	uint8
size	63.5 kB
min	35
max	237



https://github.com/haesleinhuepf/stackview

Custom kernels

CUDA is also just C. You can write custom cupy kernels using simple syntax.

T represents the image type.

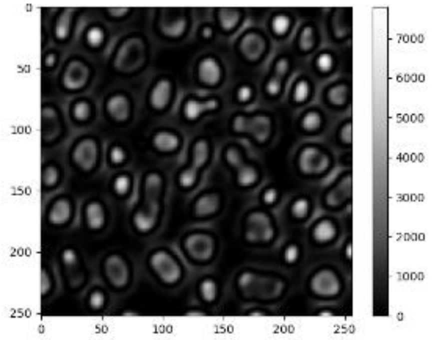
```
squared_difference = cp.ElementwiseKernel(
    'T x, T y',
    'T z',
    'z = (x - y) * (x - y)',
    'squared_difference')
```

Input, output parameters

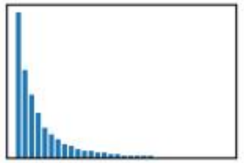
Math / CUDA code

Function call using cupy-arrays as parameters

```
[3]: cp_image = cp.asarray(image[1:], dtype=np.float32)
     image1 = gaussian_filter(cp_image, sigma=3)
     image2 = gaussian_filter(cp_image, sigma=7)

[5]: sqdiff = squared_difference(image1, image2)

[6]: stackview.insight(sqdiff)

[6]:
```



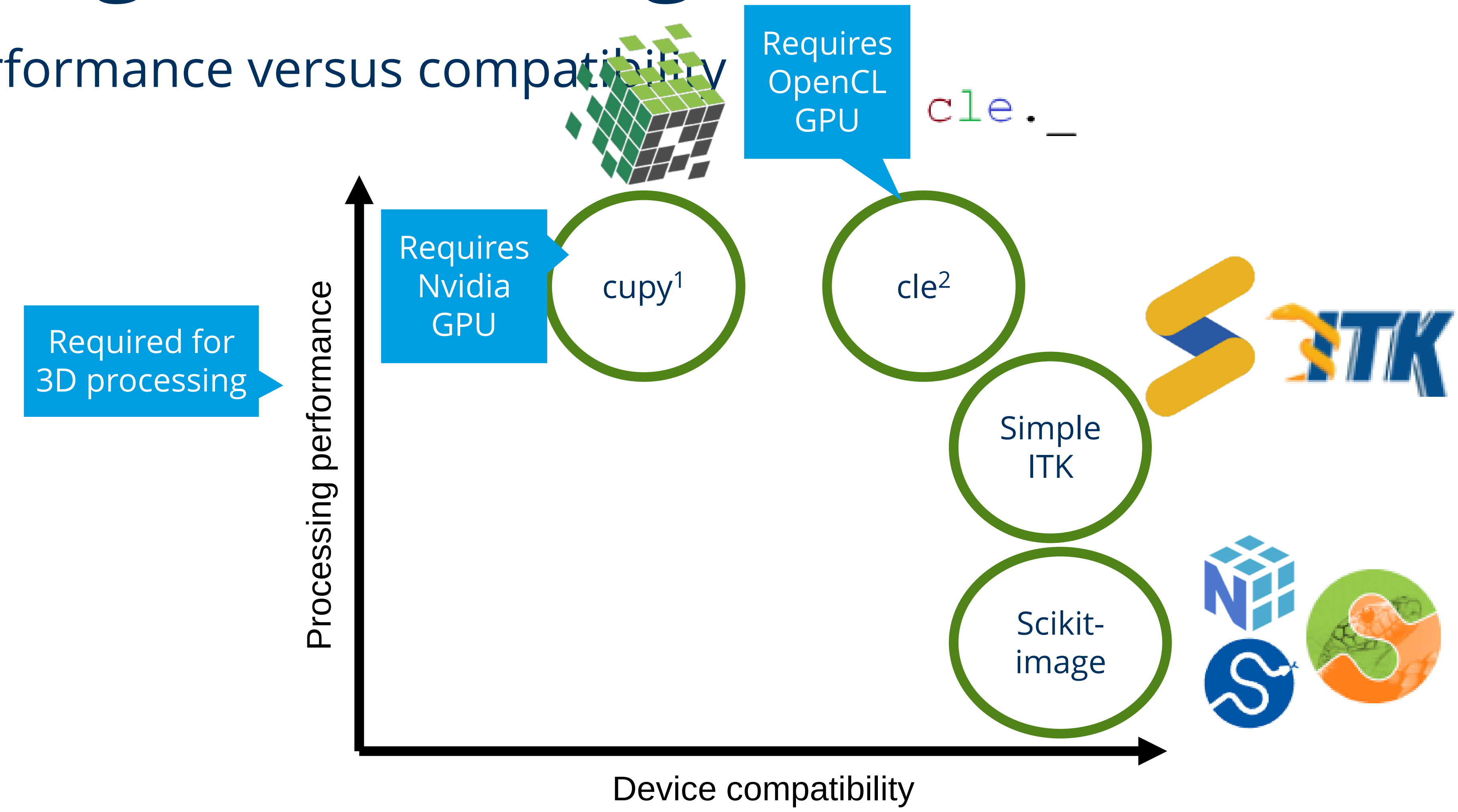
shape	(253, 256)
dtype	float32
size	253.0 kB
min	3.0791853e-08
max	7801.2026



Read more:
https://docs.cupy.dev/en/stable/user_guide/kernel.html

Image Processing CPU vs. GPU

Performance versus compatibility



CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Tiled image processing

Robert Haase

Funded by

Bundesministerium für Bildung und Forschung

SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Optimal performance through smart memory management

The classical way of dealing with large image stacks...

Load data → Preprocessing

Load data

Load data

Load data

Load data

Load data

Optimal performance through smart memory management

The classical way of dealing with large image stacks… is suboptimal

Load data | Preprocessing

Load data | Preprocessing

Load data | Preprocessing

Load data | Preprocessing

Load data | Preprocessing

Load data | Preprocessing

This strategy does not just take long; it also costs a lot of memory!

Optimal performance through smart memory management

Processing time-point by time-point is more efficient!

Load data → Preprocessing → Transformation → Segmentation → Save data

Load data → Preprocessing → Transformation → Segmentation → Save data

Load data → Preprocessing → Transformation → Segmentation → Save data

This strategy also works tile-by-tile on large 3D stacks!

Optimal performance through smart memory management

Even better: Distribute tasks between parallelized computation systems

Load data	Preprocessing	Transformation	Segmentation	Save data
Load data	Preprocessing	Transformation	Segmentation	Save data
Load data	Preprocessing	Transformation	Segmentation	Save data
Load data	Preprocessing	Transformation	Segmentation	Save data
Load data	Preprocessing	Transformation	Segmentation	Save data
Load data	Preprocessing	Transformation	Segmentation	Save data

Optimal performance through smart memory management

Even better: Distribute tasks between parallelized computation systems

python™

dask

CuPy
NumPy & SciPy for GPU

cle._ pyopencl / clesperanto

For this strategy, advanced programming skills are necessary.

... yet.

Save data

Save data

Save data

Segmentation Save data

Transformation Segmentation Save data

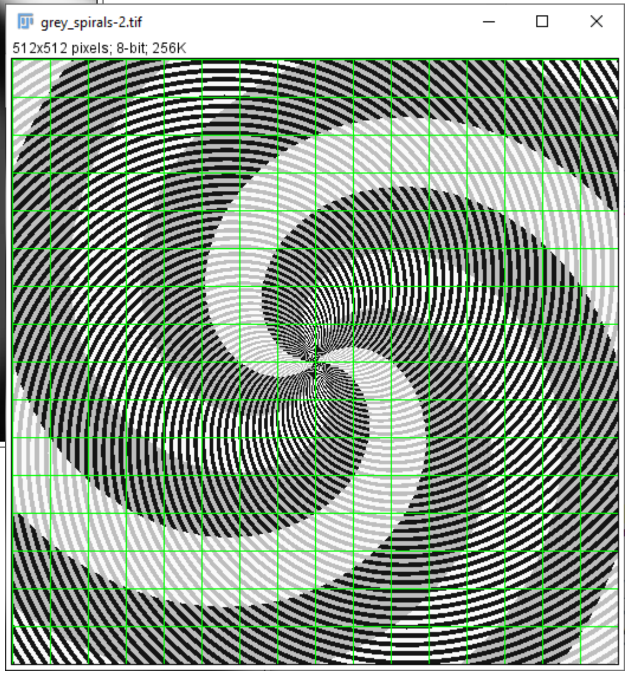
Preprocessing Transformation Segmentation Save data

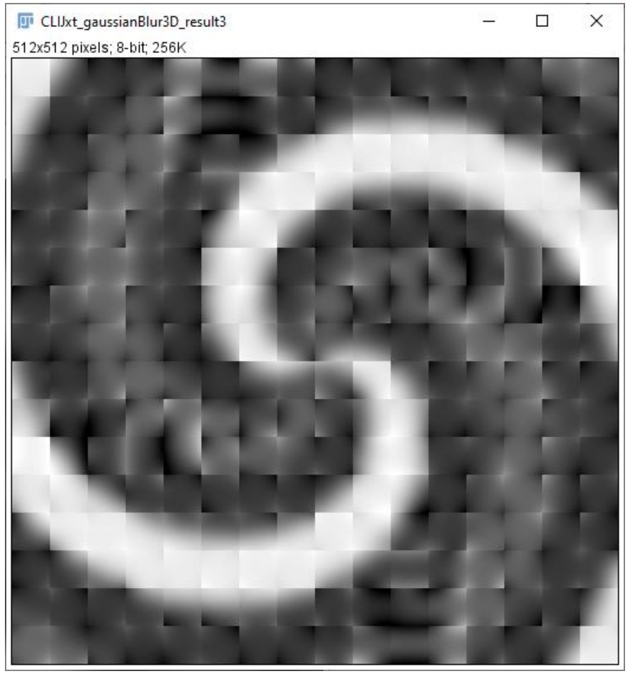
https://www.python.org/ https://www.dask.org/ https://cupy.dev
https://clesperanto.net

Tiling

The last perimeter against big data

If the image is too large for the computer memory, image processing as a whole is *not possible.*





Processing tile-by-tile poses new challenges

Tiling

Example: Gaussian blur (sigma = 20)

Solution: Process with overlapping tiles (size + margin)

Optimal margin size depends on algorithm and its parameters

Margin: 0 pixels

Margin: 10 pixels

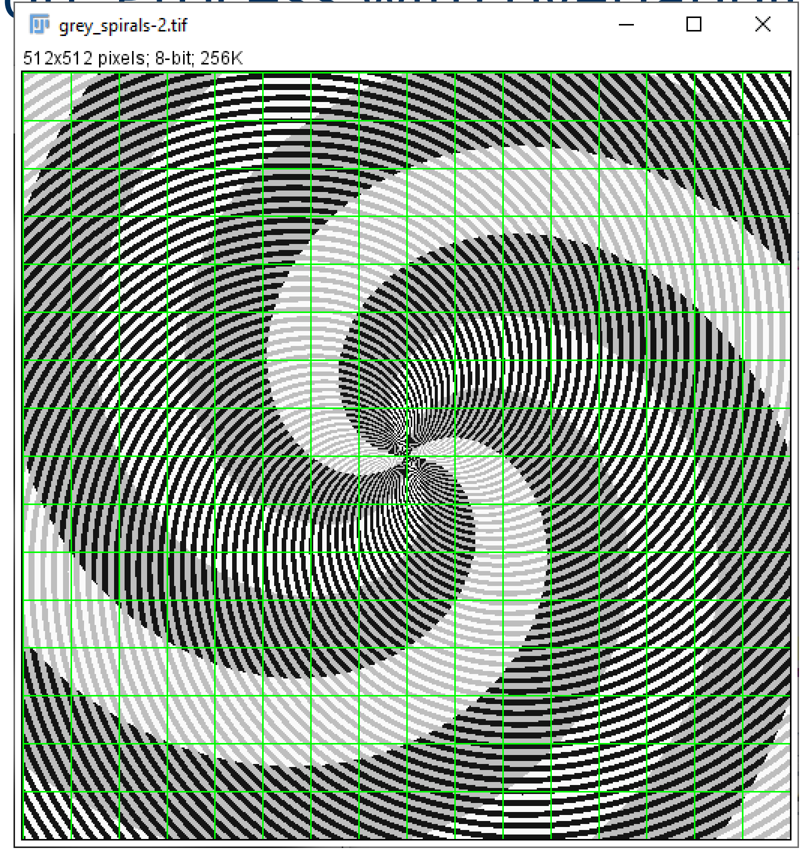
Margin: 20 pixels

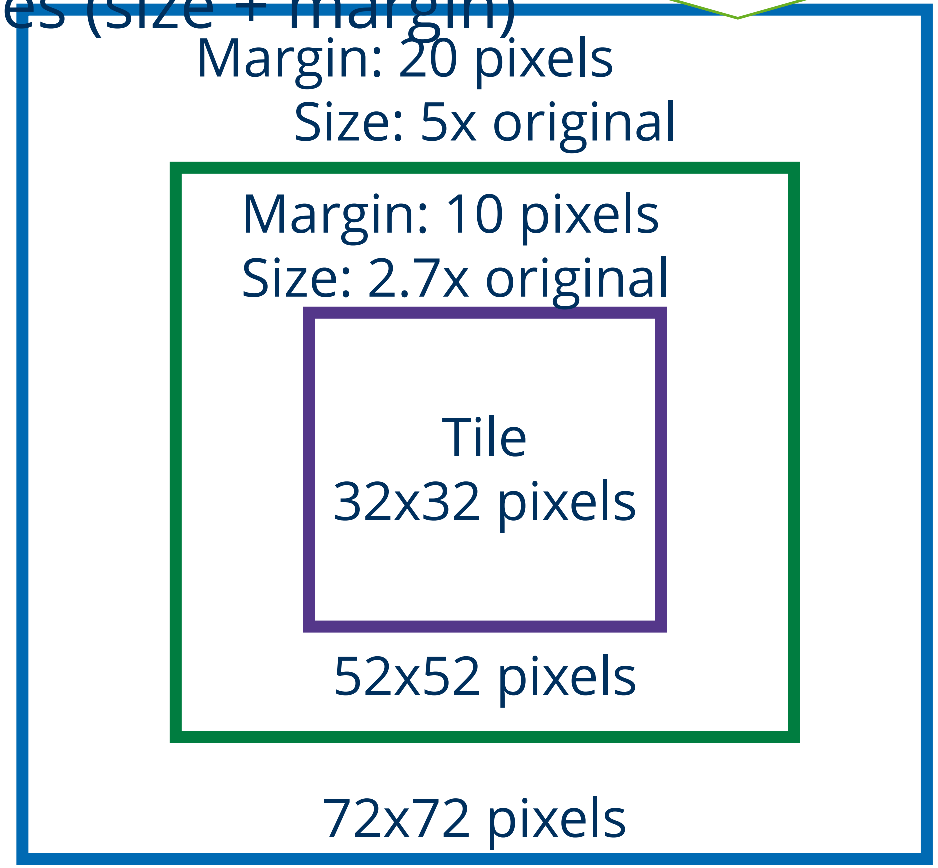
Tiling

Example: Gaussian blur (sigma = 20 pixels)

Solution: Process with overlapping tiles (size + margin)

Computation time depends on tile size and margin width



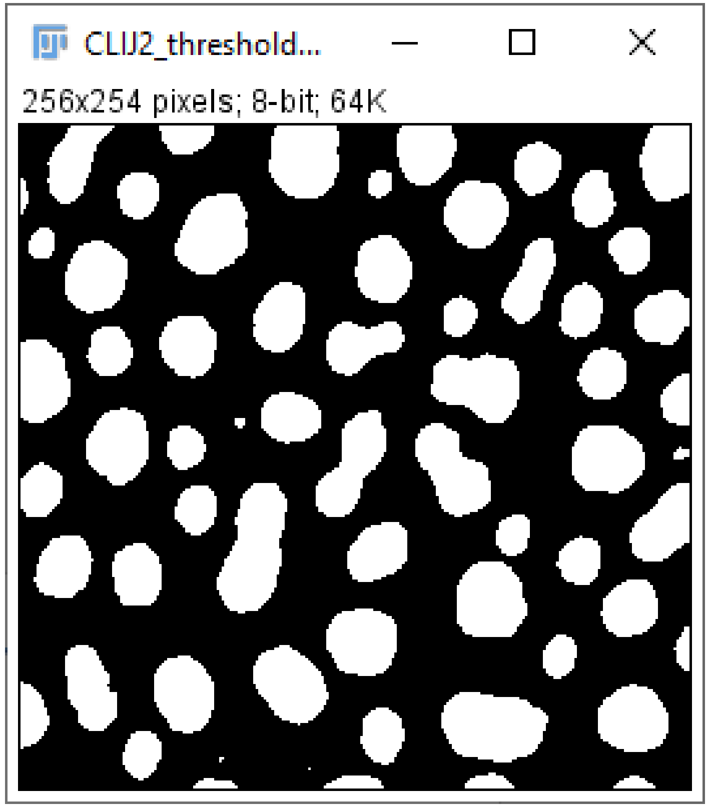


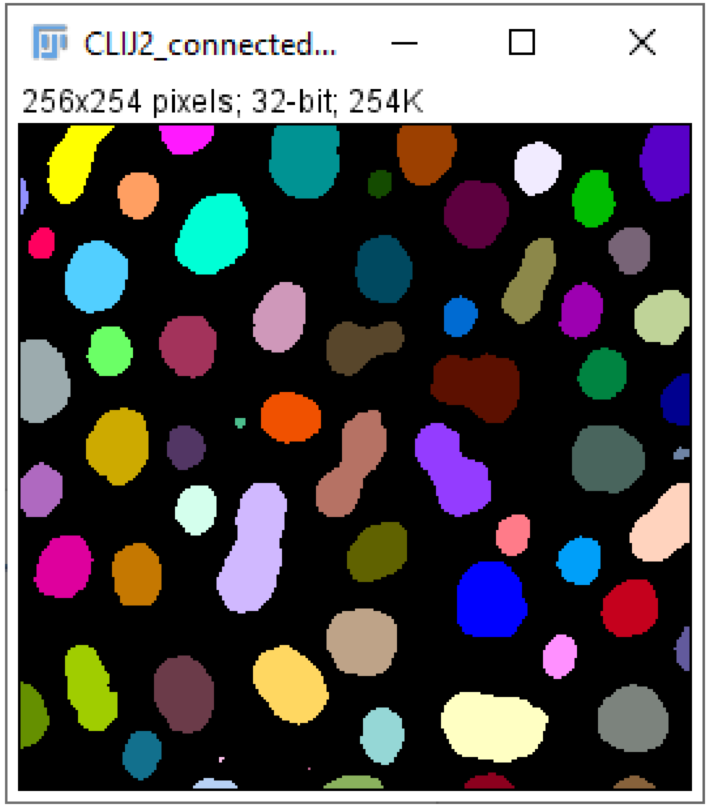
Tiling

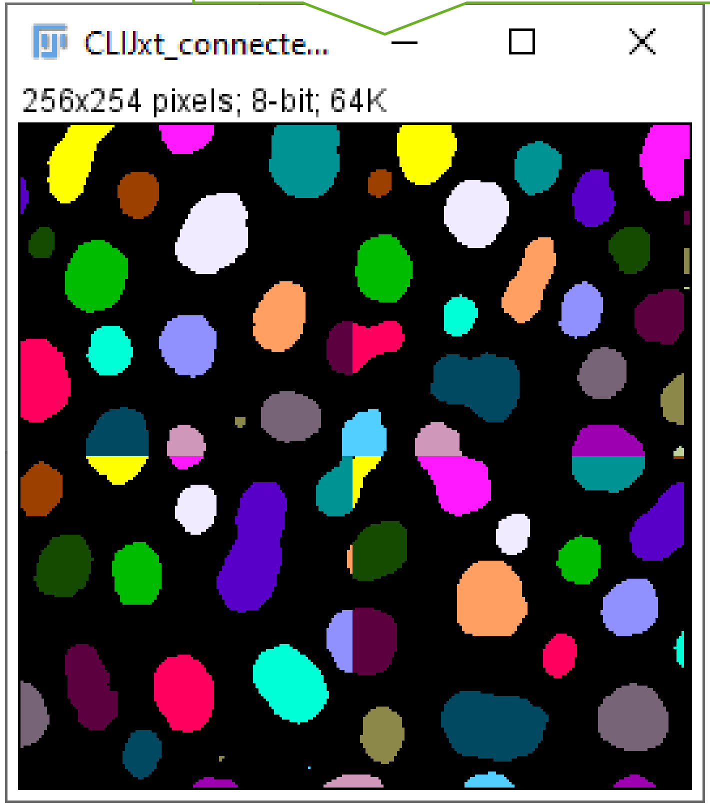
Some algorithms are hard to solve by processing tiles

Example: Connected component analysis

Checking which labels touch and combine them is feasible.





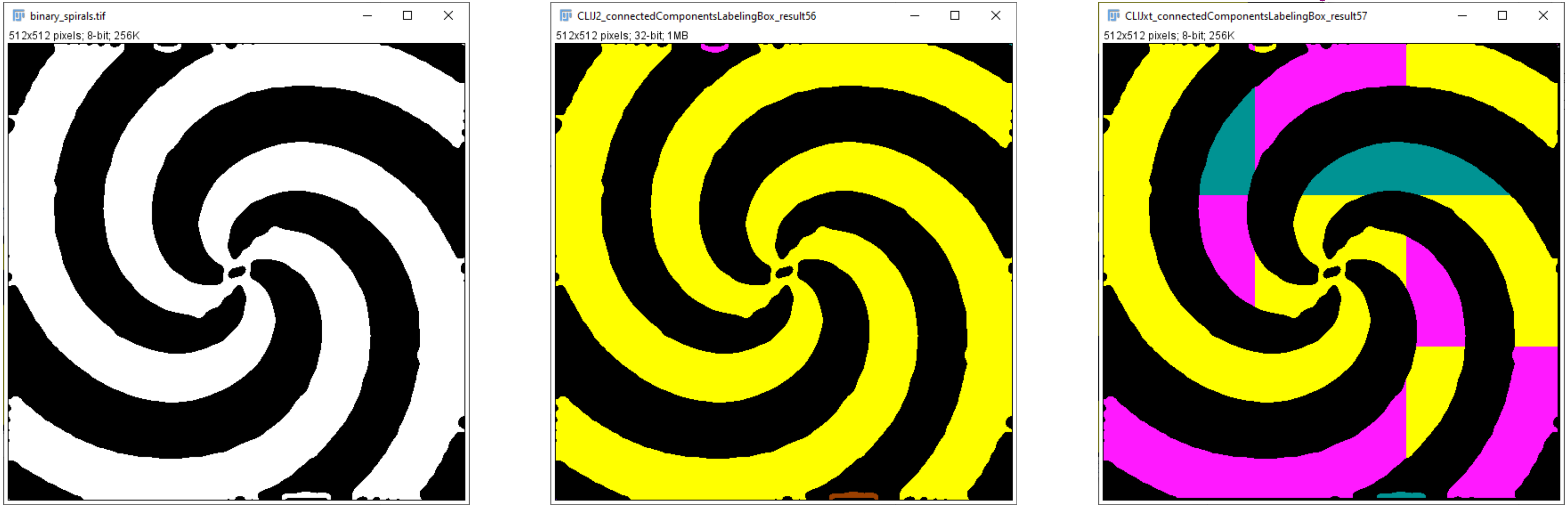


Tiling

Some algorithms are hard to solve by processing tiles

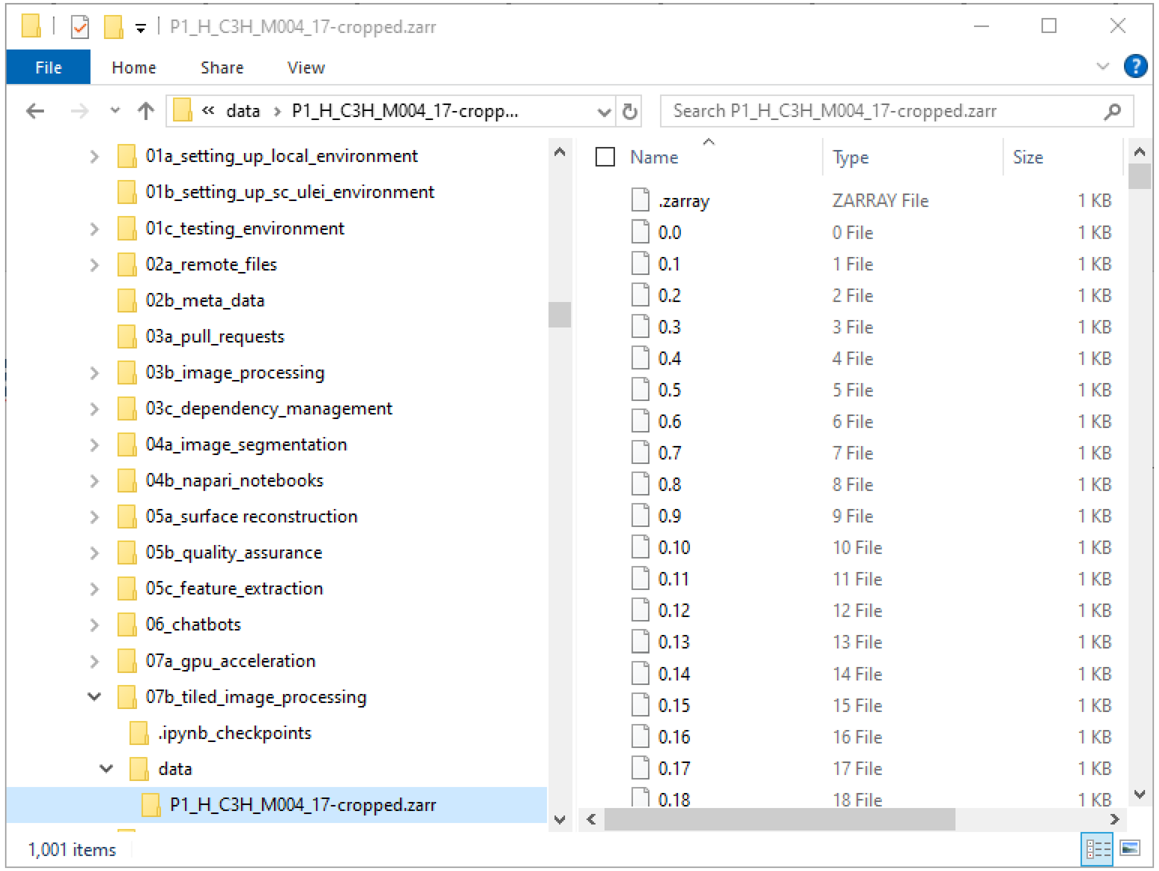
Example: Connected component analysis

There are algorithms for that, but hardly available tools.



Tiled image processing in Python

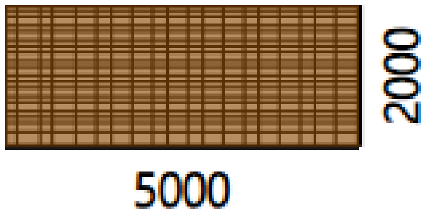
Key: tiled file formats, for parallel, distributed, lazy loading



After executing this, no pixel has been read yet.

```python
zarr_image = da.from_zarr(zarr_filename)
zarr_image
```

	Array	Chunk
Bytes	9.54 MiB	9.77 kiB
Shape	(2000, 5000)	(100, 100)
Dask graph	1000 chunks in 2 graph layers	
Data type	uint8 numpy.ndarray	



Tiled image processing in Python

Key: tiled file formats, for parallel, distributed, lazy loading

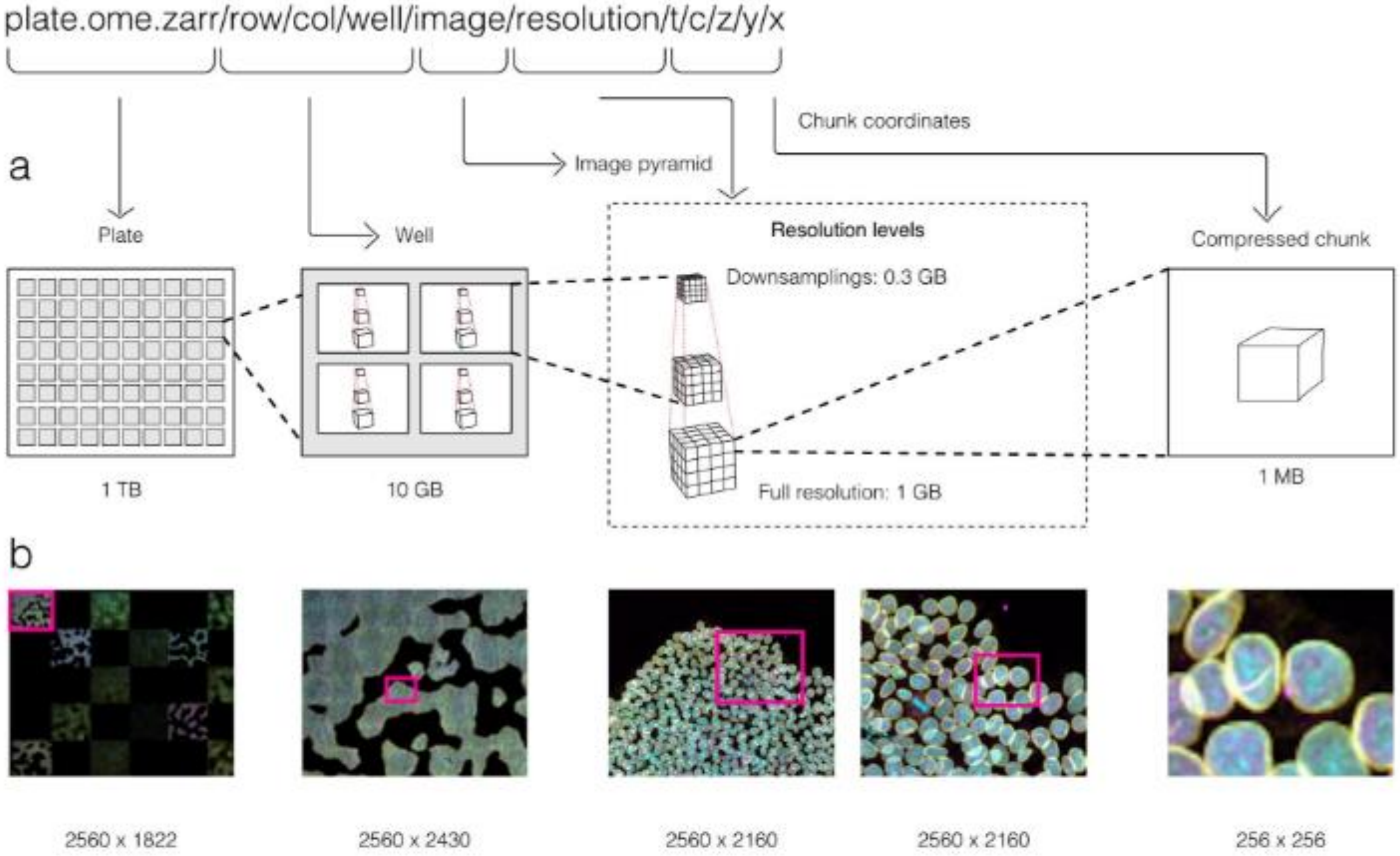


Figure taken from Moore et al, licensed [CC-BY 4..0](https://creativecommons.org/licenses/by/4.0/)
https://www.biorxiv.org/content/10.1101/2023.02.17.528834v4.full.pdf

Tiled image processing in Python

Lazy processing

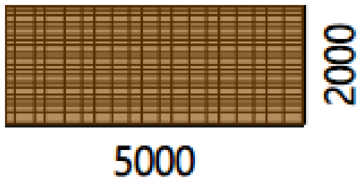
After executing this, no pixel has been read yet.

```
[5]: tile_map = da.map_blocks(count_nuclei, zarr_image)

tile_map
```

```
Processing image of size (0, 0)
(1, 1)
Processing image of size (1, 1)
(1, 1)
```

[5]:

	Array	Chunk
Bytes	76.29 MiB	78.12 kiB
Shape	(2000, 5000)	(100, 100)
Dask graph	1000 chunks in 3 graph layers	
Data type	float64 numpy.ndarray	



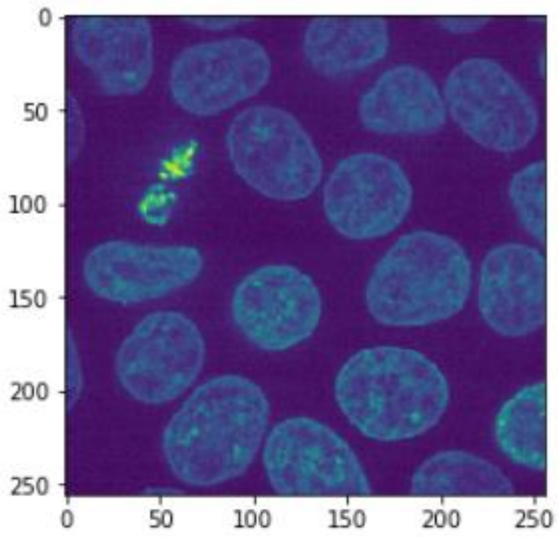
After that, results are avaialble

```
result = tile_map.compute()
```

```
Processing image of size (100, 100)
Processing image of size (100, 100)
Processing image of size (100, 100)
Processing image of size (100, 100)
Processing image of size (100, 100)
Processing image of size (100, 100)
Processing image of size (100, 100)
```

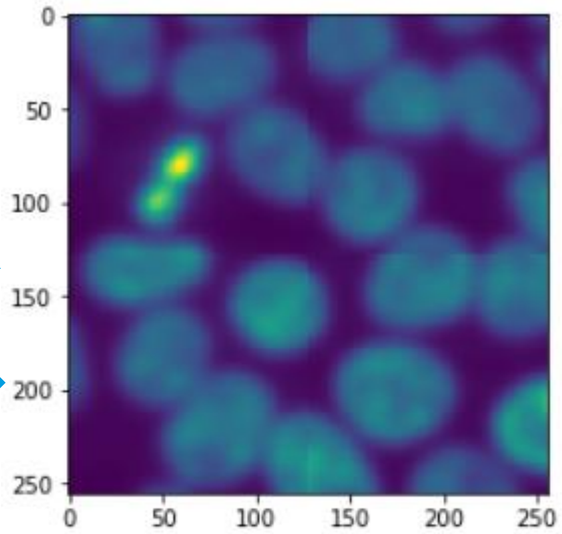
Tiling with/out overlap

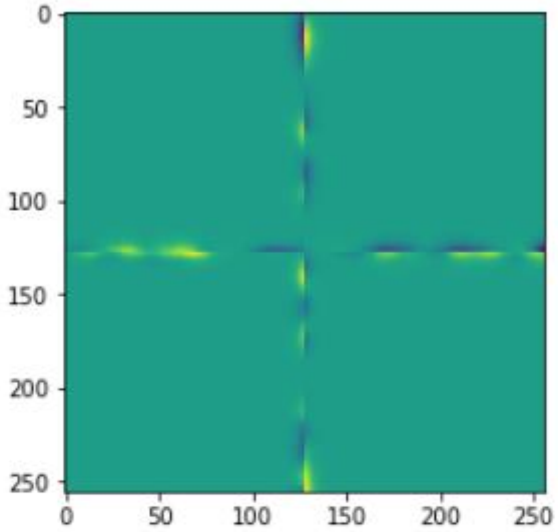
Processing of images in tiles: artifacts ad tile borders



```
tile_map = da.map_blocks(procedure, tiles)
```

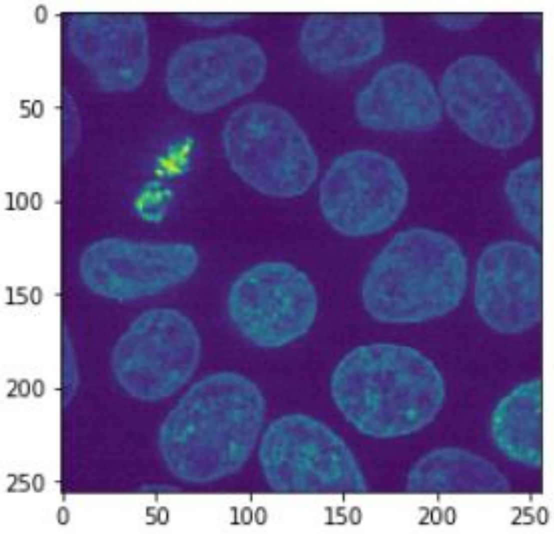
```
result = tile_map.compute()
```





Tiling with/out overlap

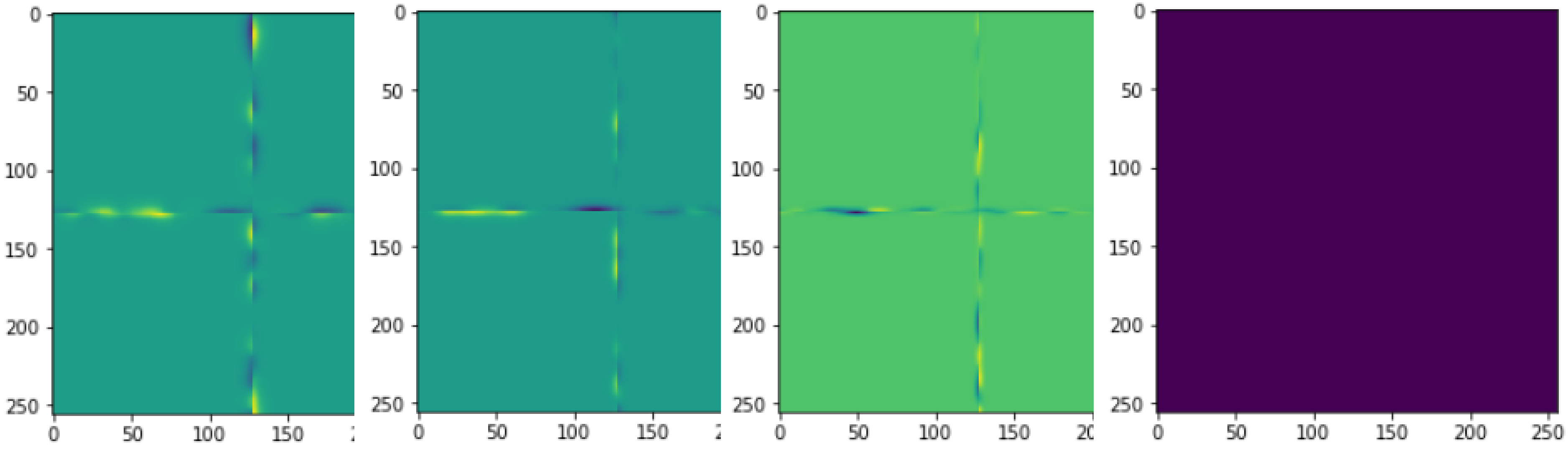
Processing of images in tiles: artifacts ad tile borders



```
overlap_width = 1
tile_map = da.map_overlap(procedure, tiles, depth=overlap_width)
```

```
result = tile_map.compute()
```

```
proceduring (128, 128)   proceduring (138, 138)   proceduring (158, 158)   proceduring (168, 168)
```



```
sum difference 1.5288188631   sum difference 2.0981679908   sum difference -0.0057617038   sum difference 0.0
```

ScaDS.AI DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Methods for comparing measurement methods

Robert Haase

Using materials Reusing materials from Daniela Vorkel, Douglas G. Altman and J. Martin Bland

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Method comparison studies

Scenario

- You work in a lab and try to improve procedures

- Chemical protocols
- Sample preparation
- Analysis protocols
  - Physical measurements
  - Image analysis

Unpaired data

- Analyze independent sample sets
- Conclude about their similarity or relationship

Inferential statistics

Paired data

- The same dataset analyzed twice with different methods
- The same dataset analyzed twice with the same method

Direct method comparison –descriptive statistics

Method comparison studies

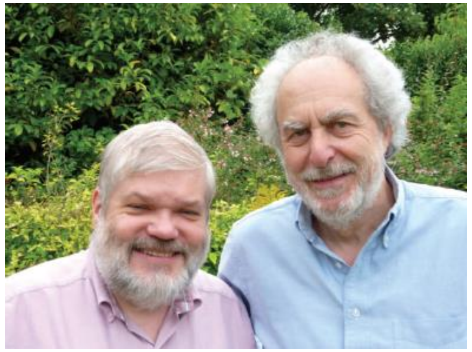
Martin Bland and Douglas Altman work on Method Comparison (excerpt)

The Statistician 32 (1983) 307-317

Measurement in Medicine: the Analysis of Method Comparison Studies†

D. G. ALTMAN and J. M. BLAND‡

Division of Computing
Research Centre, Wat
‡ Department of Clini
St George's Hospital I



Copyright J. Martin Bland and Douglas G. Altman.

THE LANCET

Volume 327, Issue 8476, 8 February 1986, Pages 307-310

Measurement

STATISTICAL METHODS FOR ASSESSING AGREEMENT BETWEEN TWO METHODS OF CLINICAL MEASUREMENT

J. Martin Bland [a b], DouglasG. Altman [a b]

Show more

+ Add to Mendeley Share Cite

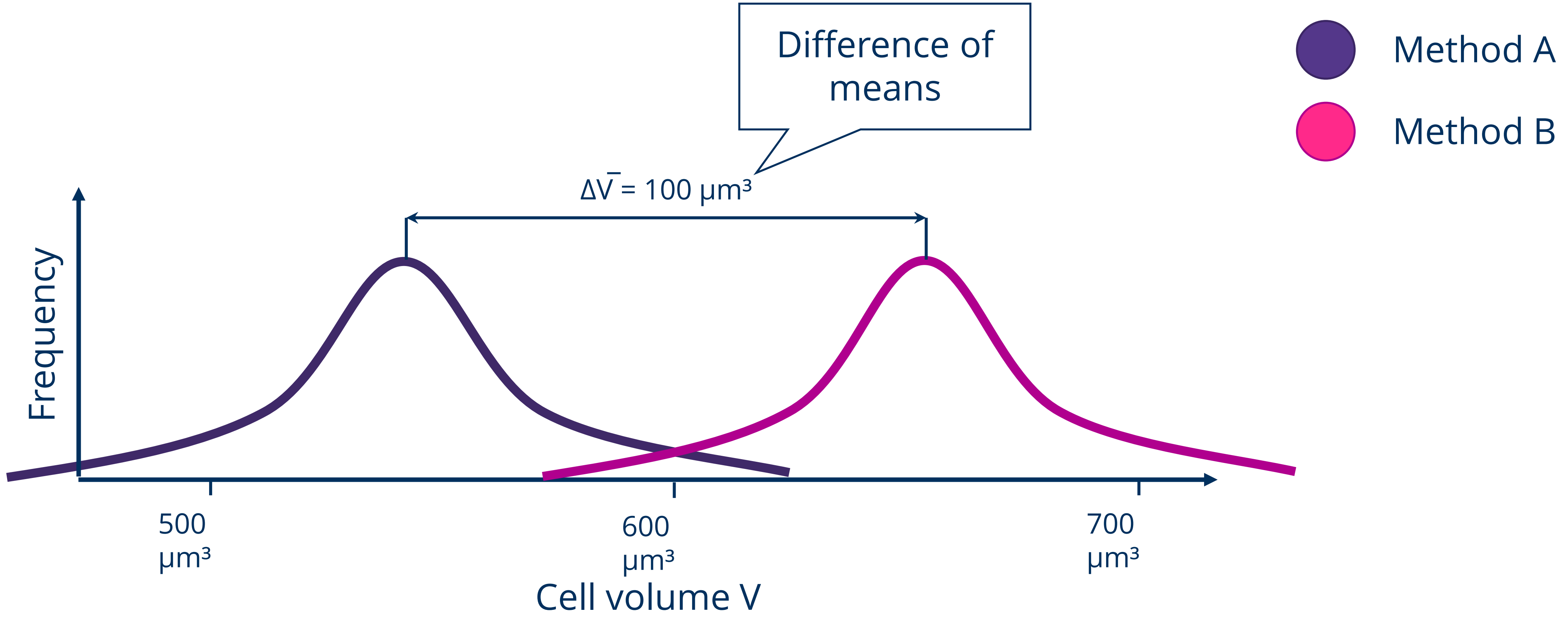
https://doi.org/10.1016/S0140-6736(86)90837-8 Get rights and content

Cited by (40162)

https://www-users.york.ac.uk/~mb55/meas/ab83.pdf (Open Access)
https://doi.org/10.1016/S0140-6736(86)90837-8

Comparison of means

Comparing mean measurements appears reasonable on the first view.



Comparison of means

Are two methods doing the same if their mean measurement is similar?

A	B
1	4
9	5
7	5
1	7
2	4
8	5
9	4
2	6
1	6
7	5
8	4

```
Mean(A) = 5.0
Mean(B) = 5.0
```

What if mean values were "very" different?

Yes

Method B cannot replace method A

No

Similar means is a necessary condition, but is it sufficient?

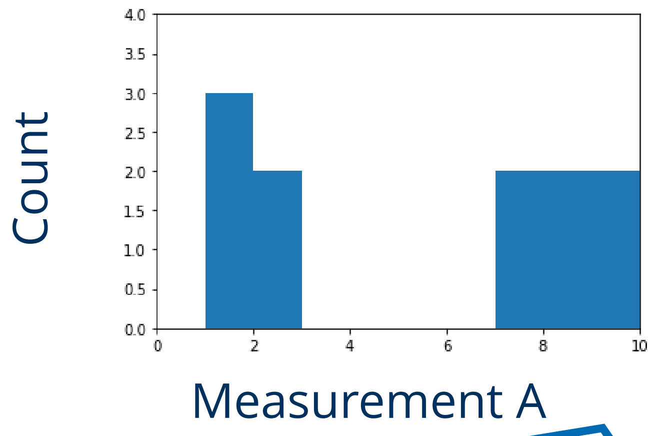
Comparison of means

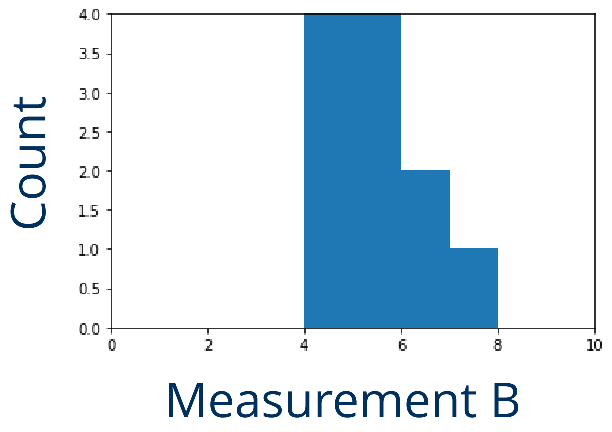
Are two methods doing the same if their mean measurement is similar?

A	B
1	4
9	5
7	5
1	7
2	4
8	5
9	4
2	6
1	6
7	5
8	4

```
Mean(A) = 5.0
Mean(B) = 5.0
```

- Draw histograms! How can two methods do the same if histograms from their measurements are different?





Similar means is a necessary condition, but it is NOT sufficient!

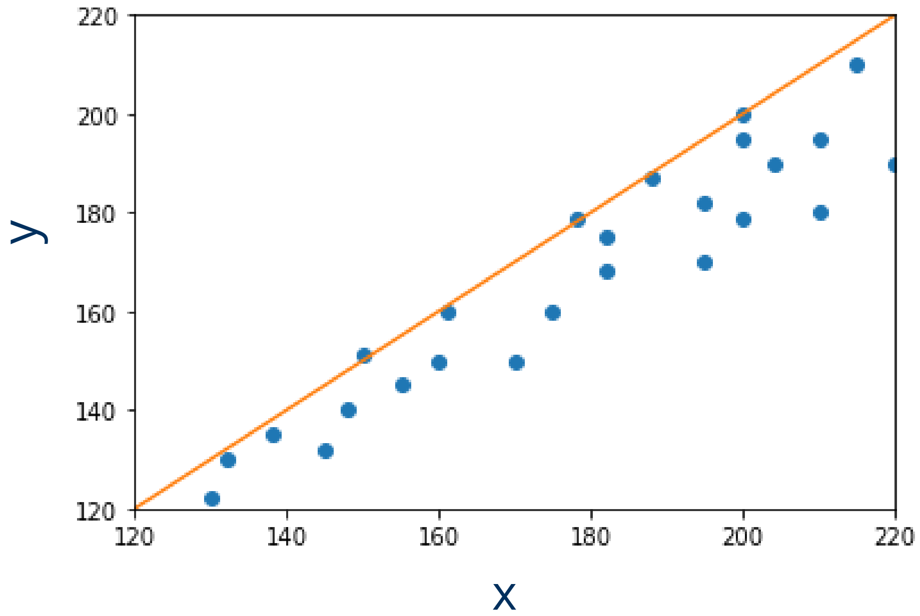
The scientific method: Show that a method doesn't work with *just one* example. And you have *proven* that the method doesn't work in general.

Correlation

Are two methods doing the same if they correlate?

- Correlation: Any kind of relationship.
- Measurable; e.g. using Pearson's Correlation Coefficient *r* enumerated linear correlation.

Comparison of two methods of measuring systolic blood pressure (Data taken from [1])



Expectation E

Mean average μ

$$r(X,Y) = \frac{E(X - \mu_X)(Y - \mu_y)}{\sigma_X \sigma_Y}$$

Standard deviation σ

In practice *E* is the weighted sum:

$$r(X,Y) = \frac{\sum_{x \in X, y \in Y} \frac{(x - \mu_X)(y - \mu_Y)}{n}}{\sigma_X \sigma_Y}$$

Number of measurements *n*

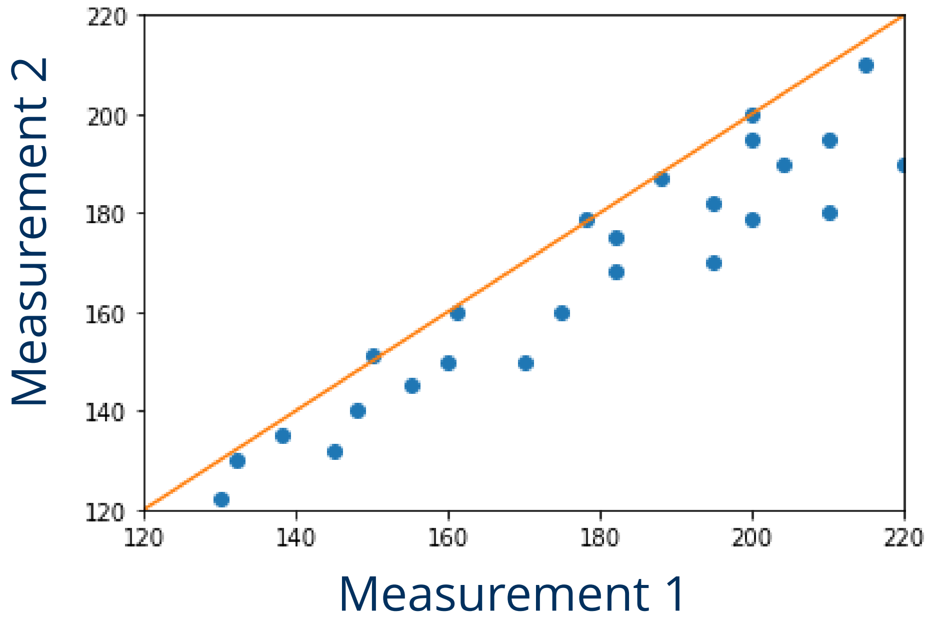
[1] Altman & Bland, The Statistician 32, 1983

Correlation

Are two methods doing the same if they correlate?

- Correlation: Any kind of relationship.
- Measurable; e.g. using Pearson's Correlation Coefficient *r* enumerated linear correlation.

Comparison of two methods of measuring systolic blood pressure (Data taken from [1])



$$r(X,Y) = \frac{\sum_{x \in X, y \in Y} \frac{(x - \mu_X)(y - \mu_Y)}{n}}{\sigma_X \sigma_Y} = 0.94$$

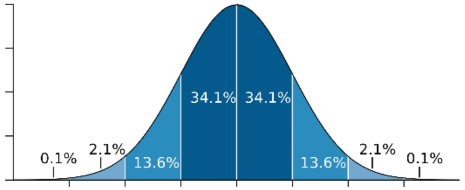
[1] Altman & Bland, The Statistician 32, 1983

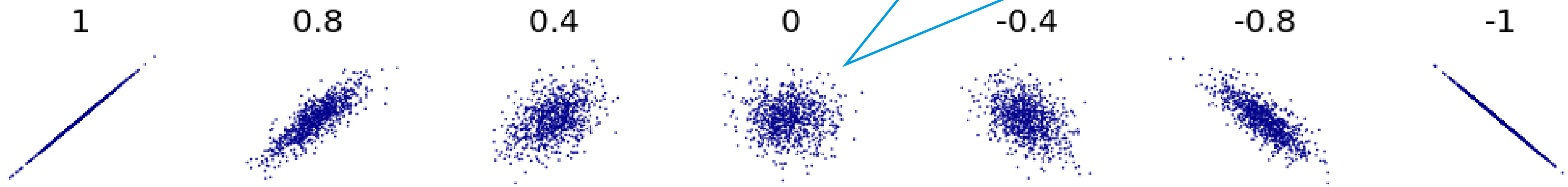
Correlation: Pearson's *r*

Pearson's *r* lies between -1 and 1

- 1: Positive linear correlation
- 0: No linear correlation
- -1: Negative linear correlation

2-dimensional normal distribution





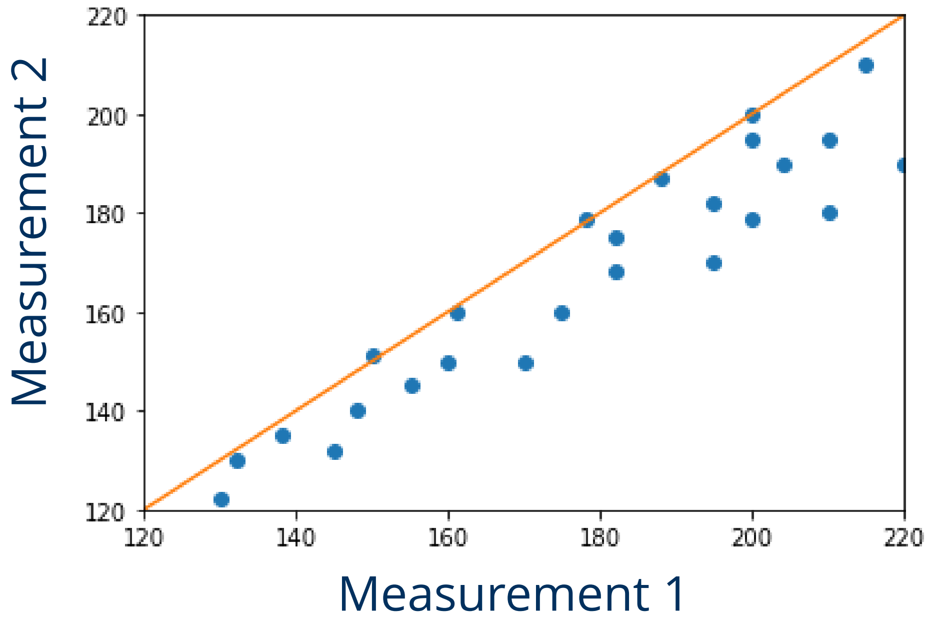
https://en.wikipedia.org/wiki/Pearson_correlation_coefficient#/media/File:Correlation_examples2.svg

Correlation

Are two methods doing the same if they correlate?

- Correlation: Any kind of relationship.
- Measurable; e.g. using Pearson's Correlation Coefficient *r* enumerated linear correlation.

Comparison of two methods of measuring systolic blood pressure (Data taken from [1])



Measurement 1 is almost always larger than measurement 2

The scientific method: Show that a method doesn't work with *just one* example. And you have *proven* that the method doesn't work in general.

"Positive linear correlation"

$$r(X,Y) = \frac{\sum_{x \in X, y \in Y} \frac{(x - \mu_X)(y - \mu_Y)}{n}}{\sigma_X \sigma_Y} = 0.94$$

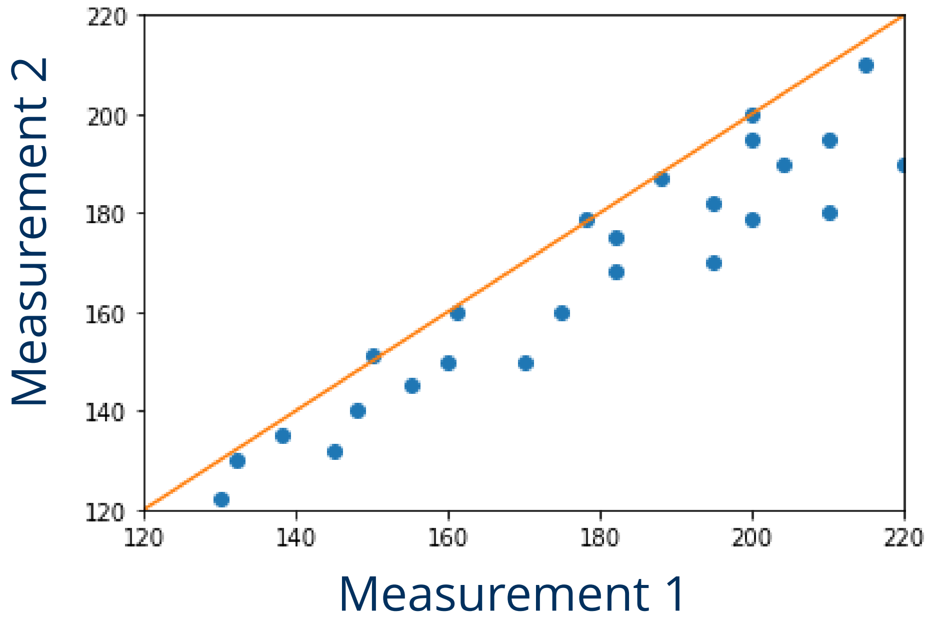
[1] Altman & Bland, The Statistician 32, 1983

Correlation

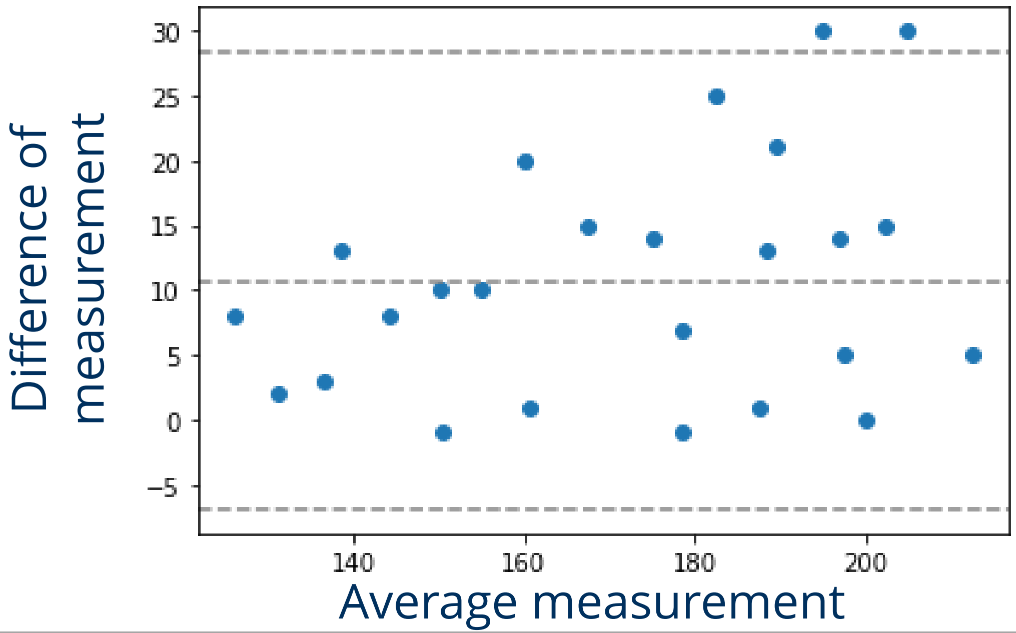
In order to evaluate the difference between two methods, you should visualize them first.

“*The purpose of computing is insight, not numbers.*”, Richard Hamming

Scatter plot



Bland-Altman plot



[1] Altman & Bland, The Statistician 32, 1983

The confidence interval

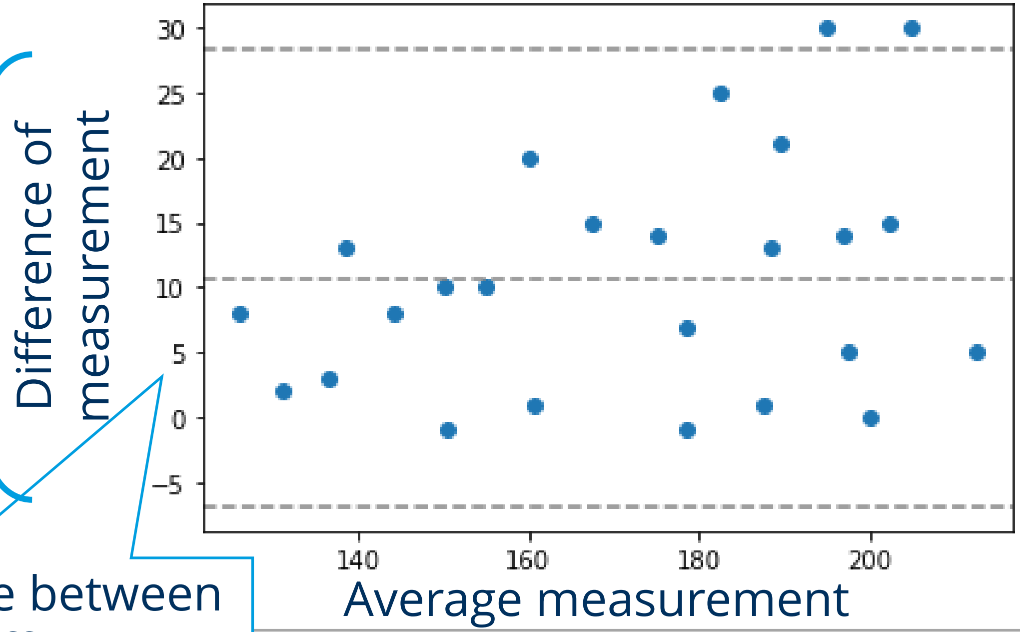
"The British Standards Institution (1979) define a coefficient of repeatability as 'the value below which the difference between two single test results … may be expected to lie with a specified probability; in the absence of other indications, the probability is 95 per cent'."[1]

The confidence interval *CI* of agreement

$$CI = (\mu - 2\sigma, \mu + 2\sigma)$$

Mean difference

Standard deviation of differences



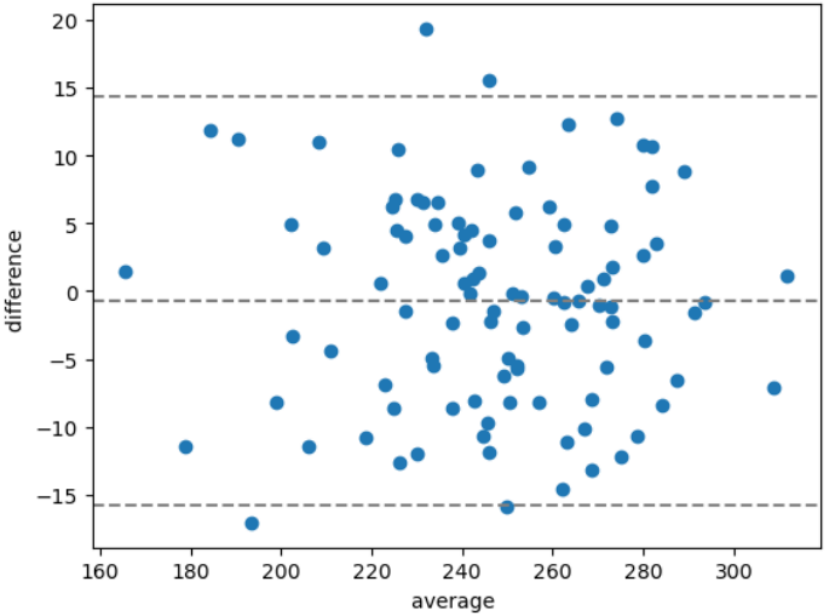
The mean difference between the methods ("offset")

[1] Altman & Bland, The Statistician 32, 1983

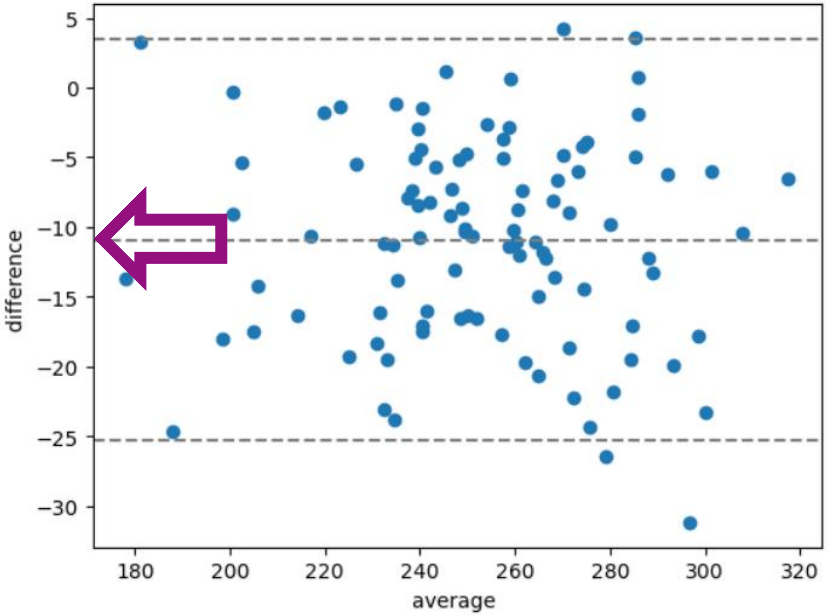
Bland-Altman plots in practice

Depending on the shape of the point-cloud, different systematic bias might be present.

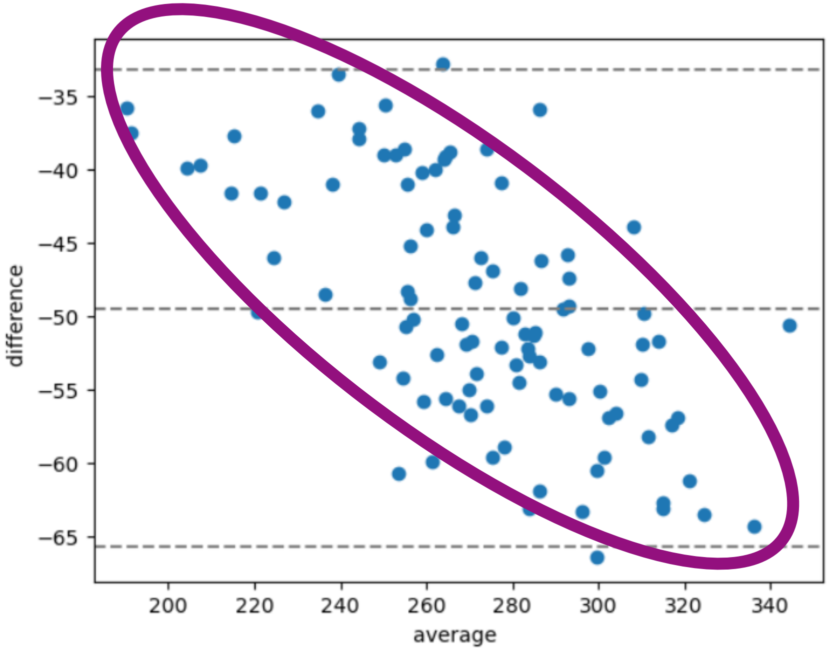
Agreement with a given random error



Absolute error or “offset”

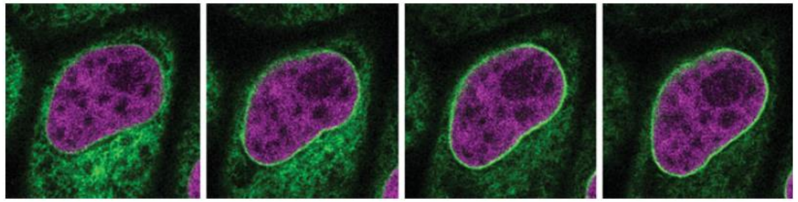


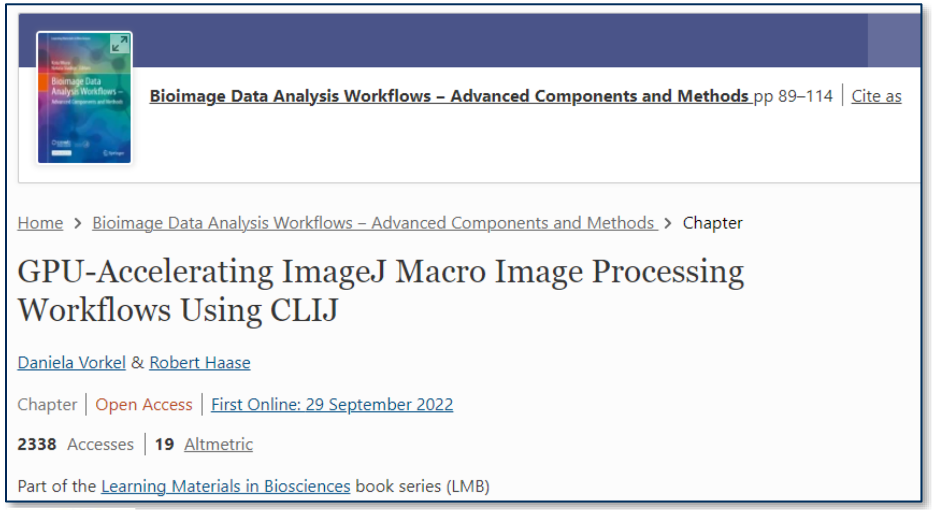
Relative error



Bland-Altman plots in practice

Comparison: ImageJ versus GPU-accelerated script to measure intensity in the nuclear envelope of a nucleus




Bioimage Data Analysis Workflows – Advanced Components and Methods pp 89–114 | Cite as

Home > Bioimage Data Analysis Workflows – Advanced Components and Methods > Chapter

GPU-Accelerating ImageJ Macro Image Processing Workflows Using CLIJ

Daniela Vorkel & Robert Haase

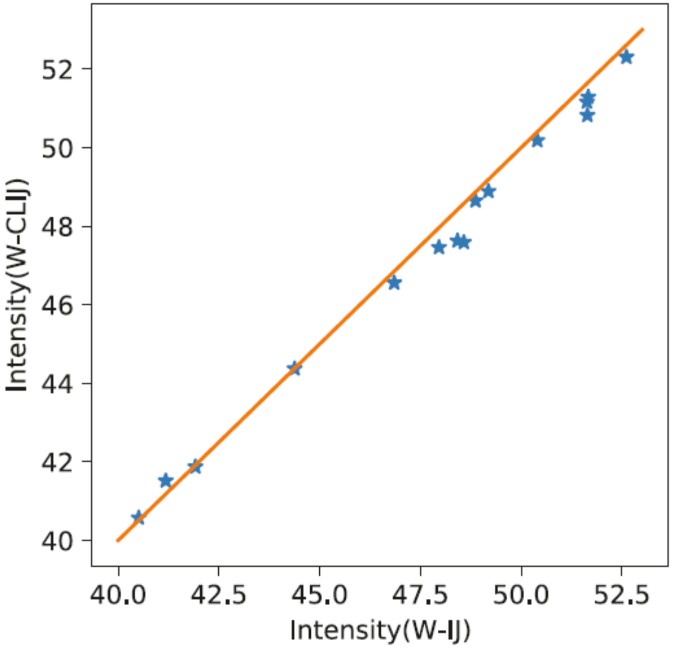
Chapter | Open Access | First Online: 29 September 2022

2338 Accesses | 19 Altmetric

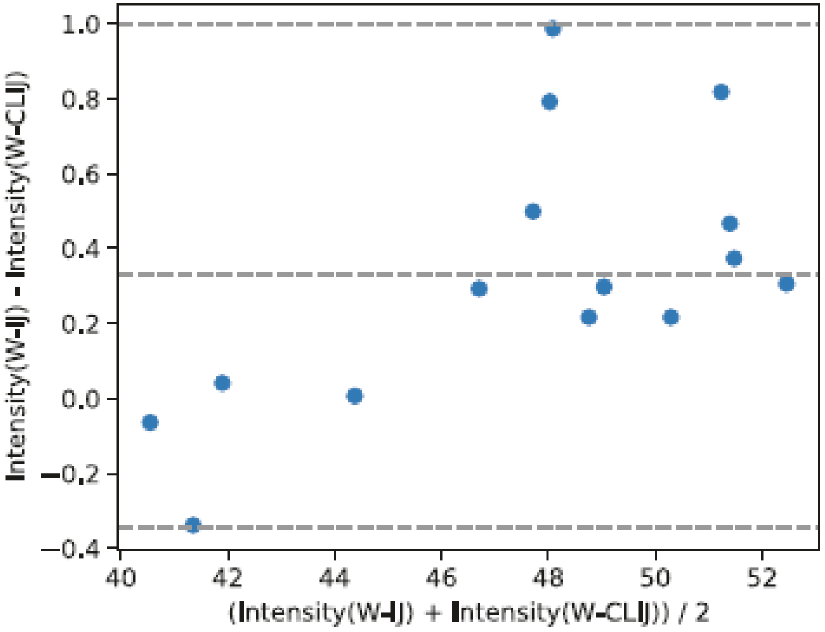
Part of the Learning Materials in Biosciences book series (LMB)


Dani Vorkel
(Myers lab)
@happifocus

Scatter plot



Bland-Altman plot



CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Exercises

Robert Haase

Funded by

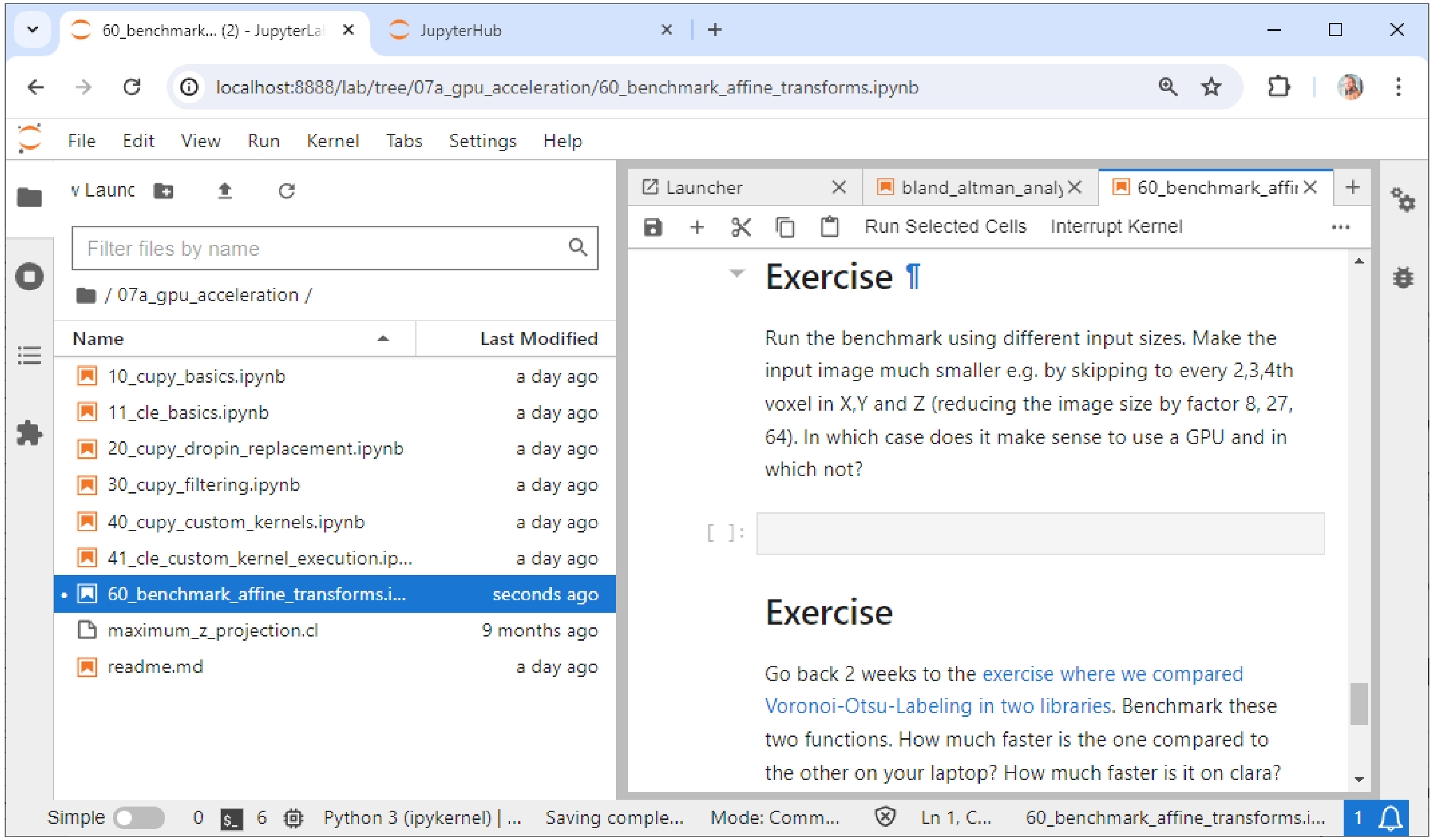
Bundesministerium für Bildung und Forschung

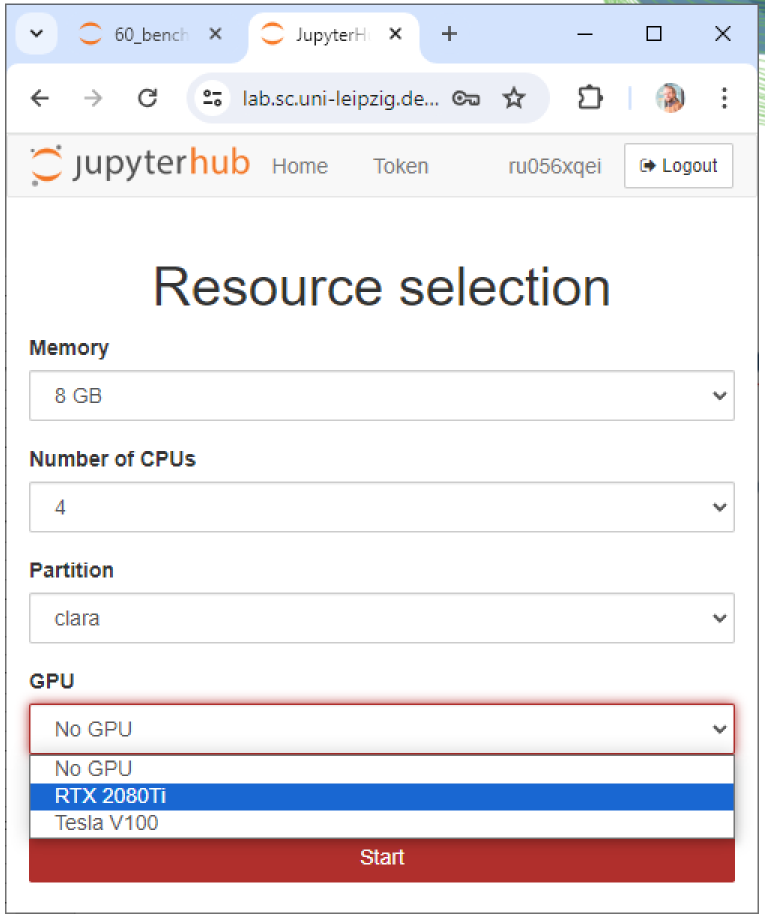
SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Exercise: GPU-accelerated image processing

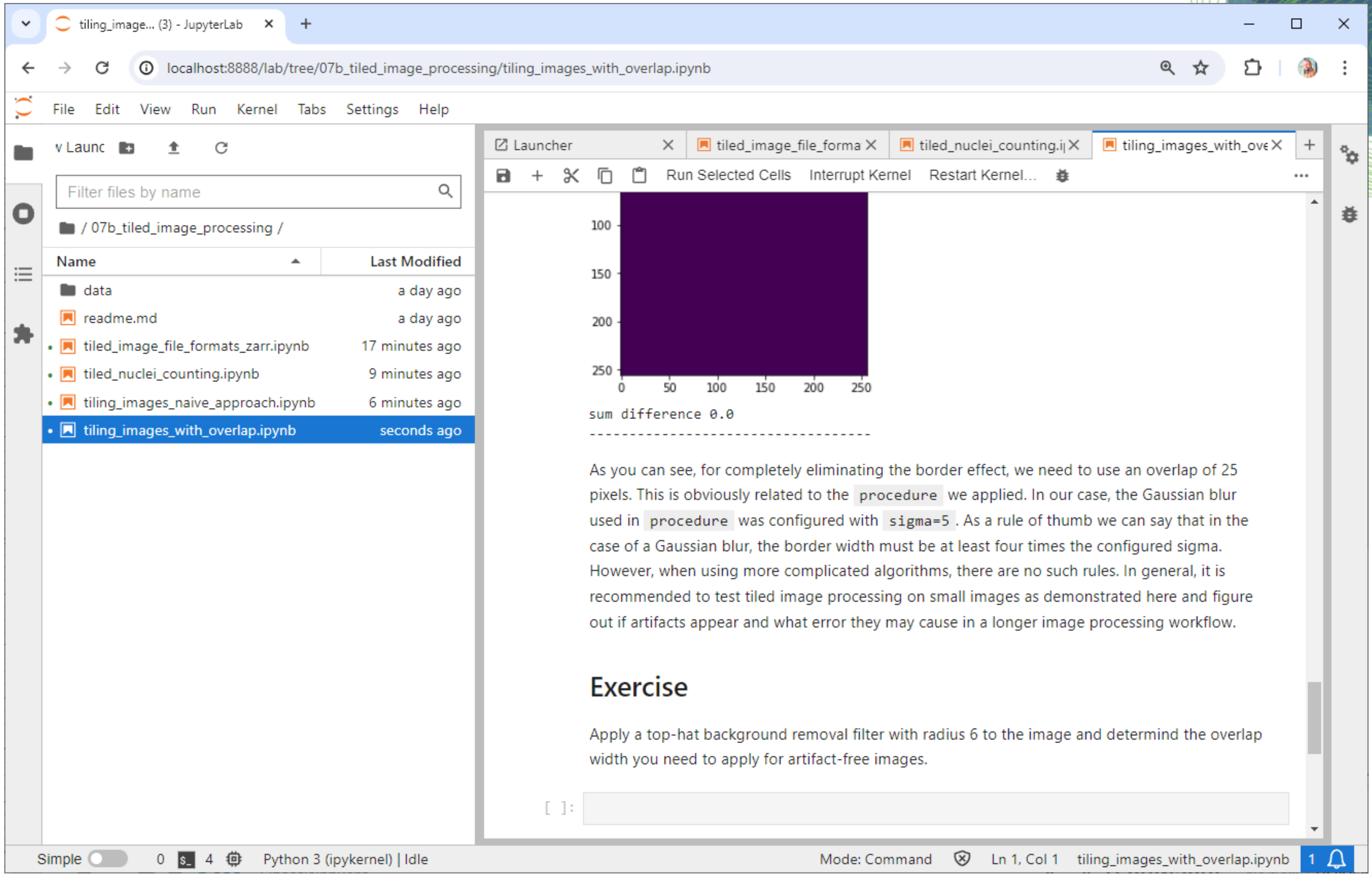
Compare CPU processing speed with a GPU





Exercise: Tiled image processing

Apply background-removal to an image in tiles. Determine the overlap width that's necessary to have artifact-free results.



Exercise: Bland-Altman plots

Compare two measurement libraries: scikit-image versus SimpleITK