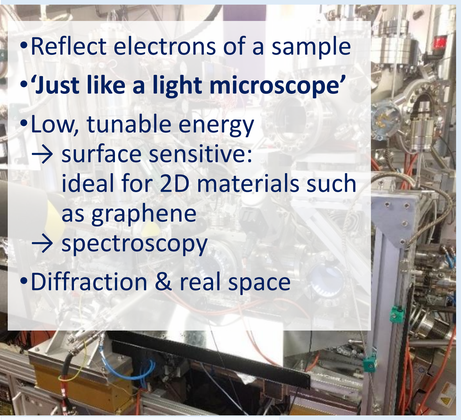
# The scientific Python stack to analyze Low Energy Electron Microscopy data

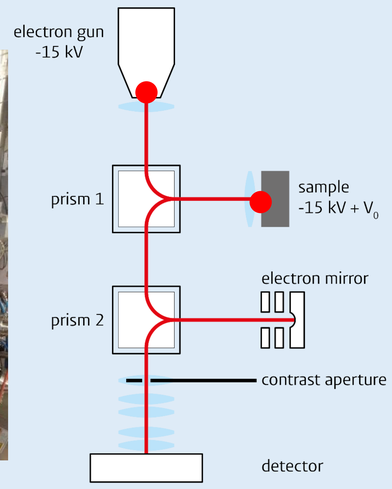
**Tobias A. de Jong,** David N. L. Kok, Tjerk Benschop, Johannes Jobst and Sense Jan van der Molen

## Low Energy Electron Microscopy

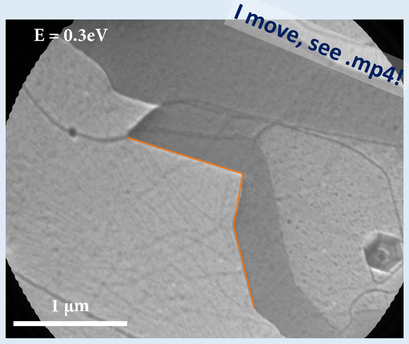


- Reflect electrons of a sample
- **'Just like a light microscope'**
- Low, tunable energy
  → surface sensitive: ideal for 2D materials such as graphene
  → spectroscopy
- Diffraction & real space

ESCHER @ Leiden University

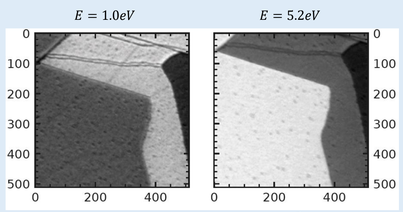



## Spectroscopy: image drifts!




- Spectroscopy: *N=700* images for *N* different electron energies.
- **Contrast inverts** and some features are only visible at some energies.
- Need to **convert** from static **position on detector** to static **position on sample**.

## Cross-correlation

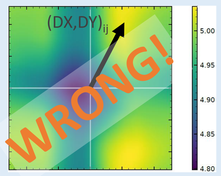


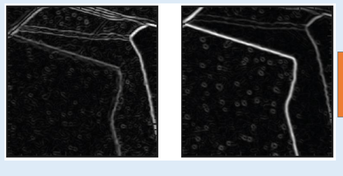

scikit-image
image processing in python

- Scikit-image does this, but can we do better for this large set of images?
- Need an edge filter first to solve contrast inversion and (dis)appearing features.

CC

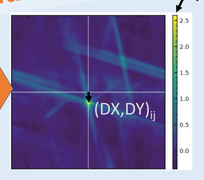
First filter



then CC

Much better




## Sub-pixel image registration

1. Select center of each of the images, sized for FFTs (i.e. $2^n \times 2^n$ pixels).
2. Apply Gaussian smoothing and Sobel filter
3. For **each pair** $(i, j)$ of images, compute location $(DX, DY)_{ij}$ and (normalized) value $W_{ij}$ of the maximum of the cross-correlation.
4. Pick $W_{min}$ to remove incorrect matches: Set $\overline{W}_{ij} = 0$ for all $\overline{W}_{ij} < W_{min}$
5. To reduce relative shifts $DX$ to a vector of absolute shifts $dx$, **minimize error** $(dx_i - dx_j - DX_{ij})\overline{W}_{ij}^4$. Idem for $DY$ to obtain $dy$.
6. **Apply shifts** $dx$ and $dy$ to the original images, interpolating for non-integer shifts.

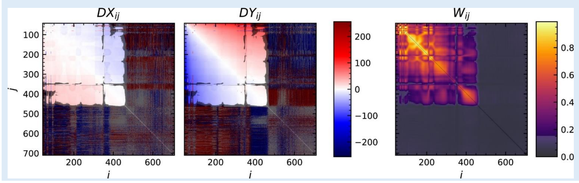
Notebook: www.github.com/TAdeJong/LEEM-analysis

250k 2D iFFTs!
Dask to the rescue.

DASK

Technically a linear problem, but lazily use scipy.optimize.least_squares.
Jacobian at Nx(NxN) becomes memory bottleneck, but luckily it is sparse (and constant), so provide explicitly in sparse form.

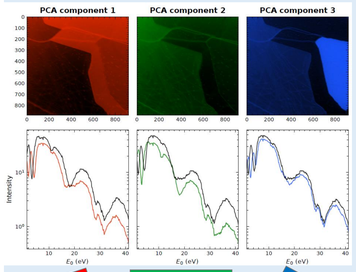
SciPy




## Dimension reduction & visualization

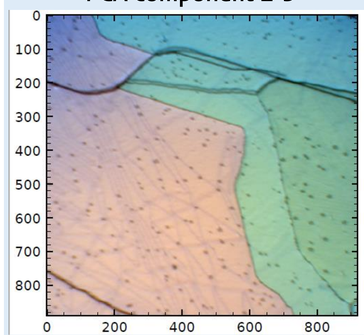
scikit learn
Who else?

- Each location: N values/features, one for each energy.
- Reduce number of dimensions (using PCA here).
- Combine three dimension into a single RGB image: 2 images is enough to visualize 95% of variance in the spectroscopy data.
- Cool aspect of this data to test dimension reduction: natural way to visualize data in reduced dimensions: as original pictures!

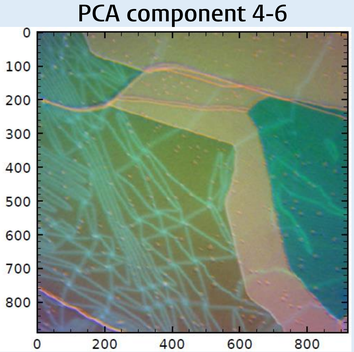



Combine colors

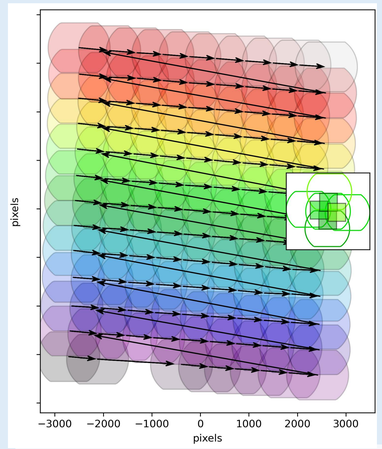
PCA component 1-3

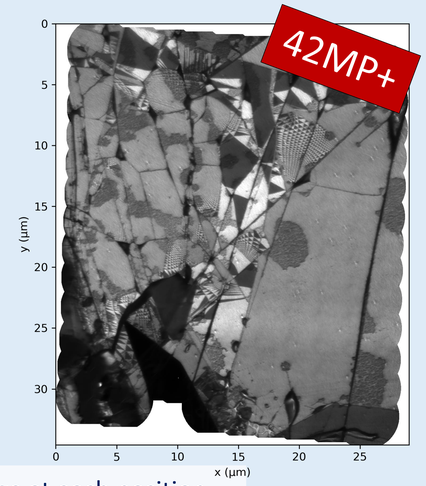


PCA component 4-6



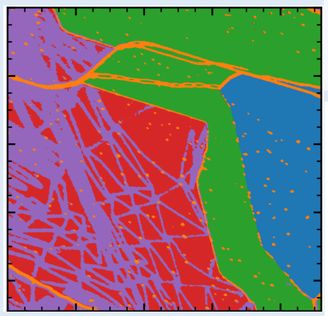
## Image stitching




- Scan sample stage, take an image at each position.
- **Same method as registration**, but:
- Only compare nearest neighbor candidates.
- Iterate with global optimization of positions.

## Clustering



- Classify regions by applying k-means to reduced data
- Such a luxury how little code PCA + clustering takes:

```
[ ]: from dask_ml.decomposition import PCA
     from dask_ml.preprocessing import StandardScaler
     from sklearn.cluster import KMeans
     from sklearn.pipeline import Pipeline, make_pipeline

[ ]: dimensions = 6
     pca = PCA(n_components=dimensions, whiten=True, random_state=4)
     pipe = make_pipeline(StandardScaler(), pca)
     pipe.fit(data)

[ ]: data_reduced = pipe.transform(data.reshape(data.shape[0], -1).T).persist()

[ ]: kmeans = KMeans(n_clusters=5, n_jobs=-1).fit(data_reduced)
     clustering = kmeans.predict(data_reduced)
```

## Contact & References

I am a physicist trying to solve my problems using Scipy, not an expert programmer. If you know of easier methods: let's talk!

@TAdeJong
@TobiasAdeJong
jongt@physics.leidenuniv.nl

Universiteit Leiden
The Netherlands

De Jong et al, Ultramicroscopy 213, 2020
Code: Github.com/TAdeJong/LEEM-analysis DOI:10.5281/zenodo.3539538
Data: DOI:10.4121/uuid:7f672638-66f6-4ec3-a16c-34181cc45202
This poster: 10.5281/zenodo.5076268
Stitchdata from: Lisi, Lu, Benschop, De Jong et al. Observation of flat bands in twisted bilayer graphene. Nat. Phys. 17, 189–193 (2021). https://doi.org/10.1038/s41567-020-01041-x