qpretrieve Documentation

Release 0.2.9

Paul Müller

May 06, 2022

CONTENTS

The qpretrieve Python library provides algorithms for the retrieval of quantitative phase information from experimental data, based on Fourier analysis (peak detection in Fourier space and filtering). Currently, qpretrieve only supports the analysis of holograms (with the characteristic fringe patterns). If you are looking for a library to load quantitative phase data from disk, please have a look at qpformat. The qpimage library can be used for further processing (e.g. background correction). This is the documentation of qpretrieve version 0.2.9.

CHAPTER

ONE

INTRODUCTION

1.1 Citing qpretrieve

If you are using qpretrieve in a scientific publication, please cite it with:

```
(...) using qpretrieve version X.X.X (available at
https://pypi.python.org/pypi/qpretrieve).
```

or in a bibliography

```
Paul Müller (2022), qpretrieve version X.X.X: Phase image analysis
[Software]. Available at https://pypi.python.org/pypi/qpretrieve.
```

and replace `X.X.X` with the version of qpretrieve that you used.

Furthermore, several ideas implemented in qpretrieve have been described and published in scientific journals:

- Phase retrieval from holographic images with a gaussian filter is implemented according to [SSM+15].

CHAPTER

TWO

GETTING STARTED

2.1 Installing qpretrieve

qpretrieve is written in pure Python and supports Python version 3.6 and higher. qpretrieve depends on several other scientific Python packages, including:

- numpy,
- scipy, and
- scikit-image (phase unwrapping using `skimage.restoration.unwrap_phase()`).

To install qpretrieve, use one of the following methods (package dependencies will be installed automatically):

- **from PyPI:** `pip install qpretrieve`
- **from sources:** `pip install -e .` or

2.2 User API

TODO

CHAPTER

THREE

EXAMPLES

3.1 Digital hologram of a single cell

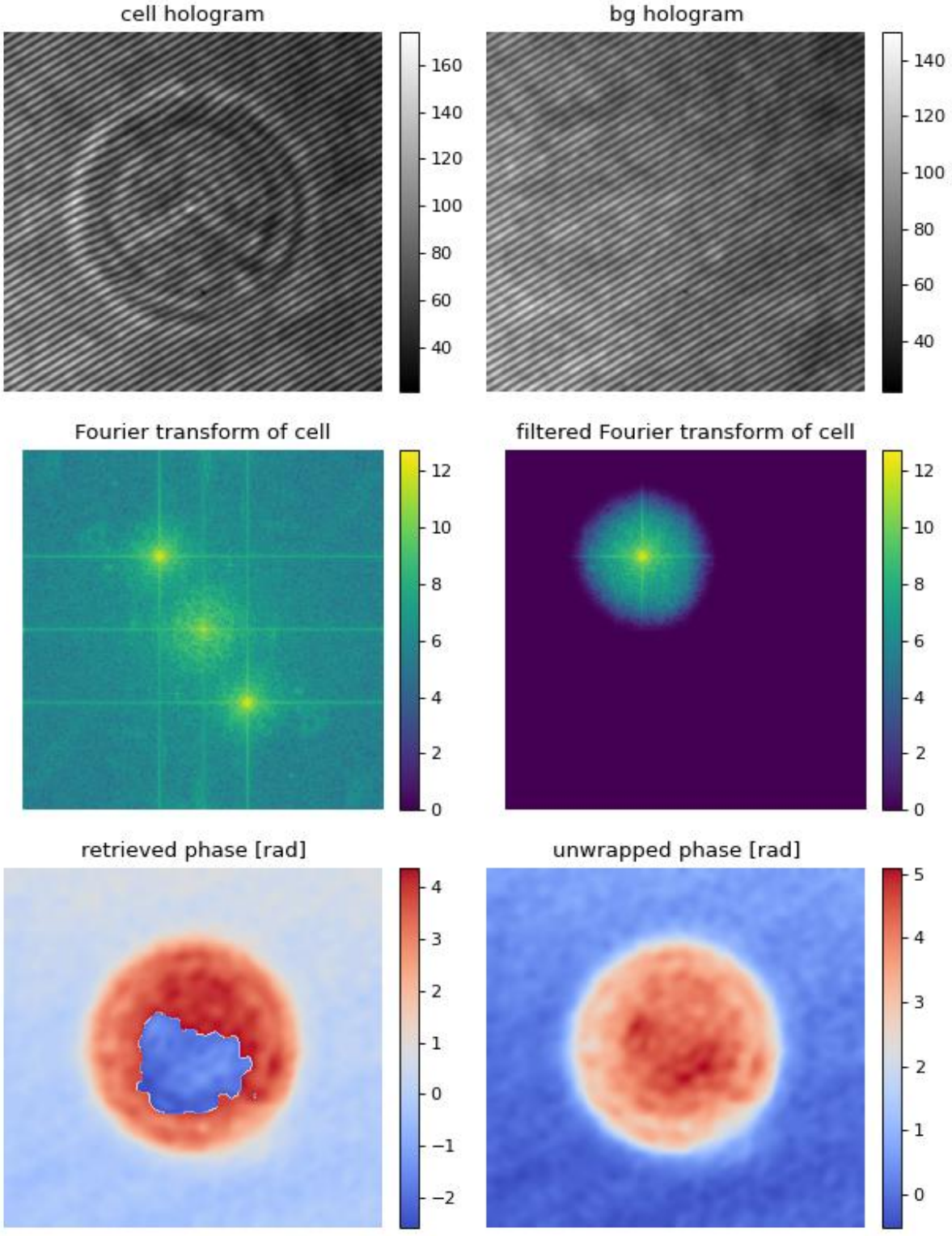
This example illustrates how qpretrieve can be used to analyze digital holograms. The hologram of a single myeloid leukemia cell (HL60) was recorded using off-axis digital holographic microscopy (DHM). Because the phase-retrieval method used in DHM is based on the *discrete* Fourier transform, there always is a residual background phase tilt which must be removed for further image analysis. The setup used for recording these data is described in reference [SSM+15].

Note that the fringe pattern in this particular example is over-sampled in real space, which is why the sidebands are not properly separated in Fourier space. Thus, the filter in Fourier space is very small which results in a very low effective resolution in the reconstructed phase.

hologram_cell.py

```
import matplotlib.pylab as plt
import numpy as np
import qpretrieve
from skimage.restoration import unwrap_phase

# load the experimental data
edata = np.load("./data/hologram_cell.npz")

holo = qpretrieve.OffAxisHologram(data=edata["data"])
holo.run_pipeline(
    # For this hologram, the "smooth disk"
    # filter yields the best trade-off
    # between interference from the central
    # band and image resolution.
    filter_name="smooth disk",
    # Set the filter size to half the distance
    # between the central band and the sideband.
    filter_size=1/2)
bg = qpretrieve.OffAxisHologram(data=edata["bg_data"])
bg.process_like(holo)

phase = holo.phase - bg.phase

# plot the properties of `qpi`
fig = plt.figure(figsize=(8, 10))

ax1 = plt.subplot(321, title="cell hologram")
```

(continues on next page)


cell hologram
160
140
120
100
80
60
40
bg hologram
140
120
100
80
60
40
Fourier transform of cell
12
10
8
6
4
2
0
filtered Fourier transform of cell
12
10
8
6
4
2
0
retrieved phase [rad]
4
3
2
1
0
−1
−2
unwrapped phase [rad]
5
4
3
2
1
0

(continued from previous page)

```
map1 = ax1.imshow(edata["data"], interpolation="bicubic", cmap="gray")
plt.colorbar(map1, ax=ax1, fraction=.046, pad=0.04)

ax2 = plt.subplot(322, title="bg hologram")
map2 = ax2.imshow(edata["bg_data"], interpolation="bicubic", cmap="gray")
plt.colorbar(map2, ax=ax2, fraction=.046, pad=0.04)

ax3 = plt.subplot(323, title="Fourier transform of cell")
map3 = ax3.imshow(np.log(1 + np.abs(holo.fft_origin)), cmap="viridis")
plt.colorbar(map3, ax=ax3, fraction=.046, pad=0.04)

ax4 = plt.subplot(324, title="filtered Fourier transform of cell")
map4 = ax4.imshow(np.log(1 + np.abs(holo.fft_filtered)), cmap="viridis")
plt.colorbar(map4, ax=ax4, fraction=.046, pad=0.04)

ax5 = plt.subplot(325, title="retrieved phase [rad]")
map5 = ax5.imshow(phase, cmap="coolwarm")
plt.colorbar(map5, ax=ax5, fraction=.046, pad=0.04)

ax6 = plt.subplot(326, title="unwrapped phase [rad]")
map6 = ax6.imshow(unwrap_phase(phase), cmap="coolwarm")
plt.colorbar(map6, ax=ax6, fraction=.046, pad=0.04)

# disable axes
[ax.axis("off") for ax in [ax1, ax2, ax3, ax4, ax5, ax6]]

plt.tight_layout()
plt.show()
```

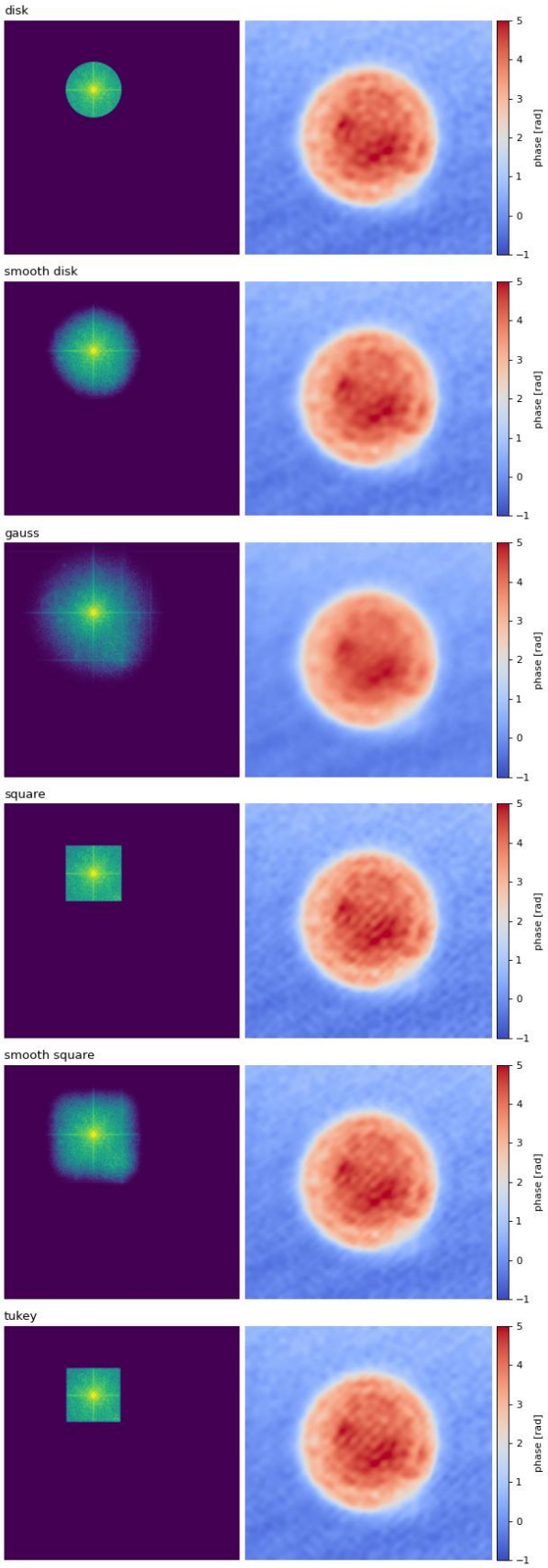
3.2 Filter visualization

This example visualizes the different Fourier filtering masks available in qpretrieve.

filter_visualization.py

```
import matplotlib.pylab as plt
import numpy as np
import qpretrieve
from skimage.restoration import unwrap_phase

# load the experimental data
edata = np.load("./data/hologram_cell.npz")

prange = (-1, 5)
frange = (0, 12)

results = {}

for fn in qpretrieve.filter.available_filters:
    holo = qpretrieve.OffAxisHologram(data=edata["data"])
```

(continues on next page)


disk
smooth disk
gauss
square
smooth square
tukey
phase [rad]
5
4
3
2
1
0
−1

(continued from previous page)

```
    holo.run_pipeline(
        filter_name=fn,
        # Set the filter size to half the distance
        # between the central band and the sideband.
        filter_size=1/2)
    bg = qpretrieve.OffAxisHologram(data=edata["bg_data"])
    bg.process_like(holo)
    phase = unwrap_phase(holo.phase - bg.phase)
    mask = np.log(1 + np.abs(holo.fft_filtered))
    results[fn] = mask, phase

num_filters = len(results)

# plot the properties of `qpi`
fig = plt.figure(figsize=(8, 22))

for row, name in enumerate(results):
    ax1 = plt.subplot(num_filters, 2, 2*row+1)
    ax1.set_title(name, loc="left")
    ax1.imshow(results[name][0], vmin=frange[0], vmax=frange[1])

    ax2 = plt.subplot(num_filters, 2, 2*row+2)
    map2 = ax2.imshow(results[name][1], cmap="coolwarm",
                      vmin=prange[0], vmax=prange[1])
    plt.colorbar(map2, ax=ax2, fraction=.046, pad=0.02, label="phase [rad]")

    ax1.axis("off")
    ax2.axis("off")

plt.tight_layout()
plt.show()
```

CHAPTER

FOUR

CODE REFERENCE

CHAPTER

FIVE

CHANGELOG

List of changes in-between qpretrieve releases.

5.1 version 0.2.9

- enh: add comments in QLSI source code explaining what is being done

5.2 version 0.2.8

- fix: replace hard-coded filter size of 400 with half the sideband distance for QLSI data

5.3 version 0.2.7

- enh: allow to specify qlsi_pitch_term and use wavelength to convert wavefront distances to phase for QLSI data
- fix: smooth square filter was not symmetric in x and y
- docs: fix filter description for square filters (double filter size)

5.4 version 0.2.6

- enh: allow to select preferred Fourier transform interface via e.g. *qpretrieve.fourier.PREFERRED_INTERFACE = "FFTFilterNumpy"*
- ref: rename OAH module internally

5.5 version 0.2.5

- fix: make sure the reference QLSI image is treated exactly like the data QLSI image in terms of padding and subtracting mean

5.6 version 0.2.4

- enh: allow to specify approximate padding size in FFTFilter
- docs: update wrong docs stated padding with linear ramp, but we are doing zero-padding

5.7 version 0.2.3

- fix: allow computation of QLSI wavefront without reference image
- ref: turn field into a property and let subclasses define how to compute phase and amplitude

5.8 version 0.2.2

- enh: use multiprocessing.cpu_count() as threads argument for FFTW

5.9 version 0.2.1

- ref: invert phase by multiplying field.imag by -1 in OAH

5.10 version 0.2.0

- feat: add quadri-wave prototype for quadriwave lateral shearing interferometry (subject to future refactoring and breaking changes)
- feat: implement FFT with PyFFTW
- setup: remove unused install_requires
- ref: clean up BaseInterferogram and support passing pipeline keyword arguments during init

5.11 version 0.1.2

- ref: add base class for Fourier hologram analysis

5.12 version 0.1.1

- fix: some Fourier filters did not work properly due to earlier refactorization
- enh: support hologram data that are RGB(A) (by only taking R)
- tests: imported tests from qpimage

5.13 version 0.1.0

- initial release

CHAPTER

SIX

BILBLIOGRAPHY

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[SSM+15] M. Schürmann, J. Scholze, P. Müller, C. J. Chan, A. E. Ekpenyong, K. J. Chalut, and J. Guck. Chapter 9 - Refractive index measurements of single, spherical cells using digital holographic microscopy. In Ewa K Paluch, editor, *Biophysical Methods in Cell Biology*, volume 125 of Methods in Cell Biology, pages 143–159. Academic Press, 2015. doi:10.1016/bs.mcb.2014.10.016.