DryMass Documentation

Release 0.1.2

Paul Müller

Mar 14, 2018

Contents:

1 Introduction **3**
- 1.1 What is DryMass? . . . 3
- 1.2 Quantitative phase imaging . . . 3

2 Theory Notes **5**
- 2.1 Computation of cell dry mass . . . 5
  - 2.1.1 Definition . . . 5
  - 2.1.2 Relative and absolute dry mass . . . 6
  - 2.1.3 Notes and gotchas . . . 6

3 Getting started **7**
- 3.1 Installing DryMass . . . 7
  - 3.1.1 Upgrade . . . 7
  - 3.1.2 Known issues . . . 7
- 3.2 Command line interface . . . 8
  - 3.2.1 dm_convert . . . 8
  - 3.2.2 dm_extract_roi . . . 9
  - 3.2.3 dm_analyze_sphere . . . 9
- 3.3 Configuration file . . . 9
  - 3.3.1 [bg] Background correction . . . 9
  - 3.3.2 [meta] Image meta data . . . 10
  - 3.3.3 [roi] Extraction of regions of interest . . . 10
  - 3.3.4 [output] Supplementary data output . . . 11
  - 3.3.5 [specimen] Specimen parameters . . . 11
  - 3.3.6 [sphere] Sphere-based image analysis . . . 11

4 Tutorials **13**
- 4.1 T1: Bead analysis with the command line interface . . . 13
  - 4.1.1 Introduction . . . 13
  - 4.1.2 Prerequisites . . . 13
  - 4.1.3 Execute dm_analyze_sphere . . . 13
  - 4.1.4 Examine the results . . . 14
  - 4.1.5 Post-processing . . . 15

5 Code reference **17**

6 Code examples **19**

6.1 Dry mass computation with radial inclusion factor 19
6.2 Comparison of relative and absolute dry mass 21

7 Changelog **25**
7.1 version 0.1.2 25
7.2 version 0.1.1 25
7.3 version 0.1.0 25

8 Bilbliography **27**

9 Indices and tables **29**

Bibliography **31**

DryMass is a user-friendly quantitative phase imaging analysis software. This is the documentaion of DryMass version 0.1.2.

CHAPTER 1

Introduction

1.1 What is DryMass?

DryMass is a software for quantitative phase imaging (QPI) analysis with functionalities such as

- extraction of meta data (e.g. wavelength, acquisition time) from experimental data files,
- quantitative phase image background correction,
- determination of dry mass for biological cells, or
- extraction of refractive index and radius for spherical phase objects such as liquid droplets, microgel beads, or cells.

1.2 Quantitative phase imaging

Quantitative phase imaging (QPI) is a 2D imaging technique that quantifies the phase retardation of a wave traveling through a specimen. For instance, digital holographic microscopy (DHM) *[KvB07]* can be used to record the quantitative phase image of biological cells, yielding the optical density from which the *dry mass* or the refractive index (RI) can be computed. Another example is electron holography *[LL02]* which can be used to visualize p-n junctions due to the different electronic potentials in the doped semiconductors. DryMass was designed for the analysis of single cells (typical units for distance [µm] and wavelength [nm]), but the concepts used apply to both methods.

CHAPTER 2

Theory Notes

2.1 Computation of cell dry mass

2.1.1 Definition

The concept of cell dry mass computation was first introduced by Barer *[Bar52]*. The dry mass m of a biological cell is defined by its non-aqueous fraction $f(x, y, z)$ (concentration or density in g/L), i.e. the number of grams of protein and DNA within the cell volume (excluding salts).

$$m = \iiint f(x, y, z)\, dxdydz$$

The assumption of dry mass computation in QPI is that $f(x, y, z)$ is proportional to the RI of the cell $n(x, y, z)$ with a proportionality constant called the refraction increment α (units [mL/g])

$$n(x, y, z) = n_{\text{intra}} + \alpha f(x, y, z)$$

with the RI of the intracellular fluid n_{intra}, a dilute salt solution. These two equations can be combined to

$$m = \frac{1}{\alpha} \cdot \iiint (n(x, y, z) - n_{\text{intra}})\, dxdydz. \tag{2.1}$$

In QPI, the RI is measured indirectly as a projected quantitative phase retardation image $\phi(x, y)$.

$$\phi(x, y) = \frac{2\pi}{\lambda} \int (n(x, y, z) - n_{\text{med}})\, dz$$

with the vacuum wavelength λ of the imaging light and the refractive index of the cell-embedding medium n_{med}. Integrating the above equation over the detector area (x, y) yields

$$\iint \phi(x, y)\, dxdy = \frac{2\pi}{\lambda} \iiint (n(x, y, z) - n_{\text{med}})\, dxdydz \tag{2.2}$$

If the embedding medium has the same refractive index as the intracellular solute ($n_{\text{med}} = n_{\text{intra}}$), then equations (2.1) and (2.2) can be combined to

$$m_{\text{med=intra}} = \frac{\lambda}{2\pi\alpha} \cdot \iint \phi(x, y)\, dxdy.$$

For a discrete image, this formula simplifies to

$$m_{\text{med=intra}} = \frac{\lambda}{2\pi\alpha} \cdot \Delta A \cdot \sum_{i,j} \phi(x_i, y_j) \tag{2.3}$$

with the pixel area ΔA and a pixel-wise summation of the phase data.

2.1.2 Relative and absolute dry mass

If however the medium surrounding the cell has a different refractive index ($n_{\text{med}} \neq n_{\text{intra}}$), then the phase ϕ is measured relative to the RI of the medium n_{med} which causes an underestimation of the dry mass if $n_{\text{med}} > n_{\text{intra}}$. For instance, a cell could be immersed in a protein solution or embedded in a hydrogel with a refractive index of $n_{\text{med}} = n_{\text{intra}} + 0.002$. For a spherical cell with a radius of 10µm, the resulting dry mass is underestimated by 46pg. Therefore, it is called "relative dry mass" m_{rel}.

$$m_{\text{rel}} = \frac{\lambda}{2\pi\alpha} \cdot \iint \phi(x, y)\, dxdy,$$

If the imaged phase object is spherical with the radius R, then the "absolute dry mass" m_{abs} can be computed by splitting equation (2.1) into relative mass and suppressed spherical mass.

$$\begin{aligned} m_{\text{abs}} &= \frac{1}{\alpha} \cdot \iiint (n(x, y, z) - n_{\text{med}} + n_{\text{med}} - n_{\text{intra}})\, dxdydz \\ &= m_{\text{rel}} + \frac{4\pi}{3\alpha} R^3 (n_{\text{med}} - n_{\text{intra}}) \end{aligned}$$

For a visualization of the deviation of the relative dry mass from the actual dry mass for spherical objects, please have a look at the *relative vs. absolute dry mass example*.

2.1.3 Notes and gotchas

- **The default refraction increment in DryMass** is α = 0.18mL/g, as suggested for cells based on the refraction increment of cellular constituents by references *[BJ54]* and *[Bar53]*. The refraction increment can be manually set using the *configuration* key "refraction increment" in the "sphere" section.
- **Variations in the refraction increment** may occur and thus the above considerations are not always valid. The refraction increment is little dependent on pH and temperature, but may be strongly dependent on wavelength (e.g. serum albumin $\alpha_{\text{SA@366nm}}$ = 0.198mL/g and $\alpha_{\text{SA@656nm}}$ = 0.179mL/g) *[BJ54]*.
- **The refractive index of the intracellular fluid** in DryMass is assumed to be n_{intra} = 1.335, an educated guess based on the refractive index of phosphate buffered saline (PBS), whose osmolarity and ion concentrations match those of the human body.
- **Dry mass and actual mass** of a cell differ by the weight of the intracellular fluid. This weight difference is defined by the volume of the cell minus the volume of the protein and DNA content. While it seems to be difficult to define a partial specific volume (PSV) for DNA, there appears to be a consensus regarding the PSV of proteins, yielding approximately 0.73mL/g (see e.g. reference *[Bar57]* as well as *[HGC94]* and question 843 of the O-manual referring to it). For example, the protein and DNA of a cell with a radius of 10µm and a dry mass of 350pg (cell volume 4.19pL, average refractive index 1.35) occupy approximately 0.73mL/g · 350pg = 0.256pL (assuming the PSV of protein and DNA are similar). Therefore, the actual volume of the intracellular fluid is 3.93pL (94% of the cell volume) which is equivalent to a mass of 3.93ng resulting in a total (actual) cell mass of 4.28ng. Thus, the dry mass of this cell makes up approximately 10% of its actual mass which leads to a total mass that is about 2% heavier than the equivalent volume of pure water (4.19ng).

CHAPTER 3

Getting started

3.1 Installing DryMass

DryMass is written in pure Python and supports Python version 3.5 and higher. DryMass depends on several other scientific Python packages, including:

- numpy,
- scikit-image (segmentation).
- qpimage (phase data manipulation),
- qpformat (file formats),
- qpsphere (refractive index analysis, segmentation),

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

- **from PyPI:** `pip install drymass`
- **from sources:** `pip install .` or `python setup.py install`

3.1.1 Upgrade

If you have installed on older version of DryMass and wish to upgrade to the latest version, use

```
pip install drymass --upgrade
```

If you wish to install a specific version of DryMass (e.g. 0.1.0), use

```
pip install 'drymass==0.1.0'
```

3.1.2 Known issues

- If you try to install from PyPI and get an error message similar to

"Could not find a version that satisfies the requirement drymass (from versions:)
No matching distribution found for drymass",

please make sure that you are using Python version 3.5 or higher with `python --version`. If that is already the case, please run `pip -vvv install drymass` and create an issue with the error messages (e.g. as a screenshot) that you get.

- If you are using Windows and the installation fails because *scikit-image* cannot be installed, e.g.

" Rolling back uninstall of scikit-image
Command python.exe [...] –compile failed with error code 1 in [...]/scikit-image",

and you are using the Anaconda Python distribution, please install *scikit-image* via `conda install scikit-image`. If you are not using Anaconda, you can install one of the wheels provided by Christoph Gohlke (download e.g. "scikit_image0.13.1cp35cp35mwin_amd64.whl" if you have installed the 64bit version of Python 3.5, navigate to the download directory and run `pip install scikit_image0.13.1cp35cp35mwin_amd64.whl`).

3.2 Command line interface

DryMass comes with a command line interface (CLI). To use the CLI, a command shell is required, such as the command prompt Cmd.exe on Windows, or Terminal.app on MacOS. Please note that if DryMass is installed in a virtual environment, then the DryMass CLI is only available if this environment is activated.

3.2.1 dm_convert

This command converts experimental data on disk to the TIF file format for use with Fiji/ImageJ and to the hdf5-based qpimage data file format. The experimental data files are loaded with the qpformat library, which supports several quantitative phase imaging file formats. If a specific format is not supported, please create an issue at the qpimage issue page. A typical use case of dm_convert on Windows is

```
dm_convert "d:\\data\path\to\experiment"
```

which is equivalent to

```
d:
cd "data\path\to"
dm_convert experiment
```

If this command is run initially for an experimental data set, the user is asked to enter or confirm imaging wavelength and detector pixel size. Then, a new directory `d:\\data\path\to\experiment_dm` is created with the following files:

drymass.cfg the user-editable *drymass configuration file* which is used in subsequent analysis steps

sensor_data.h5 the experimental data (including meta data) in the hdf5-based qpimage data file format

sensor_data.tif the experimental phase and amplitude series data as a tif file, importable in Fiji/ImageJ

Note that it is possible to edit the *drymass.cfg* file and to re-run the dm_convert command (or any other of the commands below) with these updated parameters.

3.2.2 dm_extract_roi

This command automatically finds and extracts regions of interest (ROIs) and performs an automated background correction for single-cell analysis. The usage is the same as that of dm_convert:

```
dm_extract_roi "d:\\data\path\to\experiment"
```

The command dm_extract_roi automatically runs dm_convert if it has not been run before. If ROI detection fails, the search parameters have to manually be updated in the *drymass configuration file*. The most important parameter is the diameter of the specimen in microns ("*size um*" in the *specimen* section); all other parameters are defined in the *roi* section. Note that the default parameters for the *roi* section are not written to the configuration file until dm_extract_roi is run. The following files are created by dm_extract_roi:

roi_data.h5 the extracted, background-corrected ROI data (including meta data) in the hdf5-based qpimage data file format

roi_data.tif the extracted, background-corrected ROI data as a tif file, importable in Fiji/ImageJ

roi_slices.txt the locations of the ROIs found as a txt file

sensor_roi_images.tif rendered sensor phase images with labeled ROIs; only created if "*roi images*" is set to "*True*" in the *output* section of the *drymass configuration file*

3.2.3 dm_analyze_sphere

This command is used for the analysis of spherical phase objects such as liquid droplets, beads, or suspended cells. The basic principle is thoroughly described in reference *[SSM+16]*. In short, this approach assumes that the objects found with dm_extract_roi are homogenous and spherical which allows to extract parameters such as radius and refractive index from a single phase image (as opposed to tomographic approaches that require an acquisition of multiple phase images from different directions). The parameters for the sphere analysis, such as analysis method and scattering model are defined in the *sphere* section of the *drymass configuration file*. The following files are created by dm_analyze_sphere (*METHOD* is the analysis method and *MODEL* is the scattering model defined in *drymass.cfg*):

sphere_METHOD_MODEL_data.h5 the quantitative sphere simulation data using *MODEL* with the parameters obtained with the combination of *METHOD* and *MODEL* for each of the ROIs obtained with dm_extract_roi in the hdf5-based qpimage data file format

sphere_METHOD_MODEL_images.tif rendered phase and intensity images of the input ROIs and the corresponding simulation, a difference, and a line plot through the phase image for visual inspection as a tif file, importable in Fiji/ImageJ

sphere_METHOD_MODEL_statistics.txt the analysis results, including refractive index, radius, and *relative and absolute dry mass* as a txt file.

3.3 Configuration file

The DryMass configuration file *drymass.cfg* is located in the root of the output folder ("*_dm*" appended to the data path). The configuration file is divided into sections.

3.3.1 [bg] Background correction

```
# indexing starts at 1
"amplitude data":
    ("none", int_or_str, "Amplitude bg correction file or index"),
```

```
# see qpimage library: e.g. fit, gauss, mean, mode
"amplitude offset":
    ("mean", lcstr, "Amplitude bg correction offset method"),
# see qpimage library: e.g. ramp, offset
"amplitude profile":
    ("ramp", lcstr, "Amplitude bg correction profile method"),
# see skimage.filters.threshold_*
"amplitude binary threshold":
    (np.nan, float_or_str, "Binary image threshold value or method"),
"amplitude border perc":
    (10, float, "Amplitude bg border region to analyze [%]"),
"amplitude border px":
    (5, int, "Amplitude bg border region to analyze [px]"),
"enabled":
    (True, fbool, "Enable bg correction globally"),
# indexing starts at 1
"phase data":
    ("none", int_or_str, "Phase bg correction file or index"),
# see qpimage library: e.g. fit, gauss, mean, mode
"phase offset":
    ("mean", lcstr, "Phase bg correction offset method"),
# see qpimage library: e.g. ramp, offset
"phase profile":
    ("ramp", lcstr, "Phase bg correction profile method"),
# see skimage.filters.threshold_*
"phase binary threshold":
    (np.nan, float_or_str, "Binary image threshold value or method"),
"phase border perc":
    (10, float, "Phase bg border region to analyze [%]"),
"phase border px":
    (5, int, "Phase bg border region to analyze [px]"),
```

3.3.2 [meta] Image meta data

```
"medium index":
    (np.nan, float, "Refractive index of the surrounding medium"),
"pixel size um":
    (np.nan, float, "Detector pixel size [µm]"),
"wavelength nm":
    (np.nan, float, "Imaging wavelength [nm]"),
```

3.3.3 [roi] Extraction of regions of interest

```
"dist border":
    (10, int, "Minimum distance of objects to image border [px]"),
"eccentricity max":
    (.7, float, "Allowed maximal eccentricity of the specimen"),
"exclude overlap":
    (30., float, "Allowed distance between two objects [px]"),
"force":
    ((), tupletupleint, "Force ROI coordinates (x1,x2,y1,y2) [px]"),
"pad border":
    (40, int, "Padding of object regions [px]"),
```

```
"size variation":
    (.5, float, "Allowed variation relative to specimen size"),
```

3.3.4 [output] Supplementary data output

```
"roi images":
    (True, fbool, "Rendered phase images with ROI location"),
"sphere images":
    (True, fbool, "Phase/Intensity images for sphere analysis"),
"sensor tif data":
    (True, fbool, "Phase/Amplitude sensor tif data"),
```

3.3.5 [specimen] Specimen parameters

```
"size um":
    (10, float, "Approximate diameter of the specimen [µm]"),
```

3.3.6 [sphere] Sphere-based image analysis

```
"method":
    ("edge", lcstr, "Method for determining sphere parameters"),
"model":
    ("projection", lcstr, "Physical sphere model"),
"edge coarse":
    (.4, float, "Coarse edge detection filter size"),
"edge fine":
    (.1, float, "Fine edge detection filter size"),
"edge clip radius min":
    (.9, float, "Interior edge point filtering radius"),
"edge clip radius max":
    (1.1, float, "Exterior edge point filtering radius"),
"edge iter":
    (20, int, "Maximum number iterations for coarse edge detection"),
"refraction increment":
    (.18, float, "Refraction increment [mL/g]"),
"radial inclusion factor":
    (1.2, float, "Radial inclusion factor for dry mass computation"),
```

CHAPTER 4

Tutorials

4.1 T1: Bead analysis with the command line interface

4.1.1 Introduction

Microgel beads are transparent, homogeneous, and spherical objects that are ideal test objects for quantitative phase imaging. The DryMass command *dm_analyze_sphere* can estimate the average refractive index of such homogeneous objects. This is a short tutorial that will reproduce the data presented in supplementary figure 2a of reference *[SCG+17]*.

4.1.2 Prerequisites

For this tutorial, you need:

- Python 3.5 or above and DryMass version 0.1.1 or above (see *Installing DryMass*)
- Fiji (optional, for data visualization)
- Experimental data set: QLSR_PAA_beads.zip

4.1.3 Execute dm_analyze_sphere

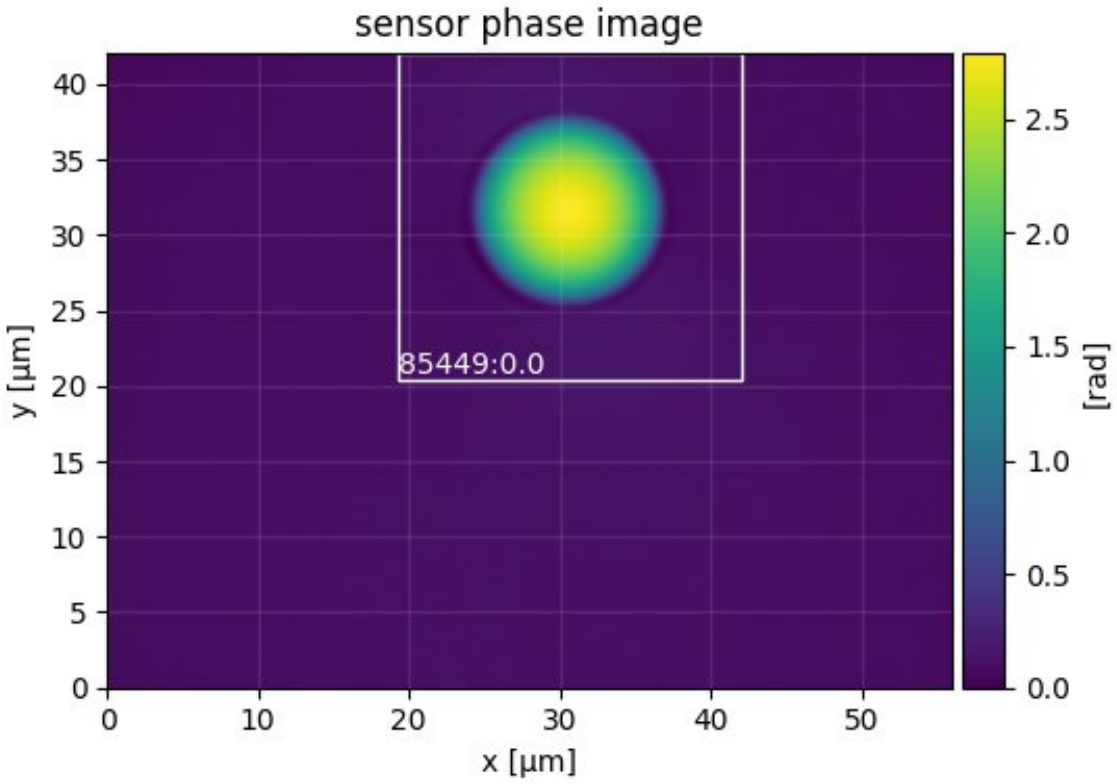
DryMass comes with a *Command line interface* (CLI) which is made available after the installation. We will use the DryMass command *dm_analyze_sphere* to extract the refractive index values of a population of microgel beads. Using the command shell of your operating system, navigate to the location of QLSR_PAA_beads.zip and execute the command `dm_analyze_sphere` with `QLSR_PAA_beads.zip` as an argument. You will be prompted for the refractive index of the surrounding medium (1.335), the detector pixel size in microns (0.14), and the wavelength in nanometers (647). Simply type in these values (press the *Enter* key to let DryMass acknowledge each input). On Windows, this will look similar to this (2.5x time lapse):

DryMass has created a directory called `QLSR_PAA_beads.zip_dm` (the input argument with an *_dm* appended) which contains the following files

- **drymass.cfg**: DryMass *Configuration file*
- **roi_data.h5**: regions of interest (ROIs)
- **roi_data.tif**: phase and amplitude data of the ROIs as a tif file
- **roi_slices.txt**: positions of the ROIs
- **sensor_data.h5**: full sensor QPI data
- **sensor_data.tif**: full sensor phase and amplitude data as a tif file
- **sensor_roi_images.tif**: plotted full sensor phase images with ROIs
- **sphere_edge_projection_data.h5**: simulated (projection) phase and amplitude data
- **sphere_edge_projection_images.tif**: visualization of the sphere analysis of the ROIs as a tif file
- **sphere_edge_projection_statistics.txt**: sphere analysis results as a text file

4.1.4 Examine the results

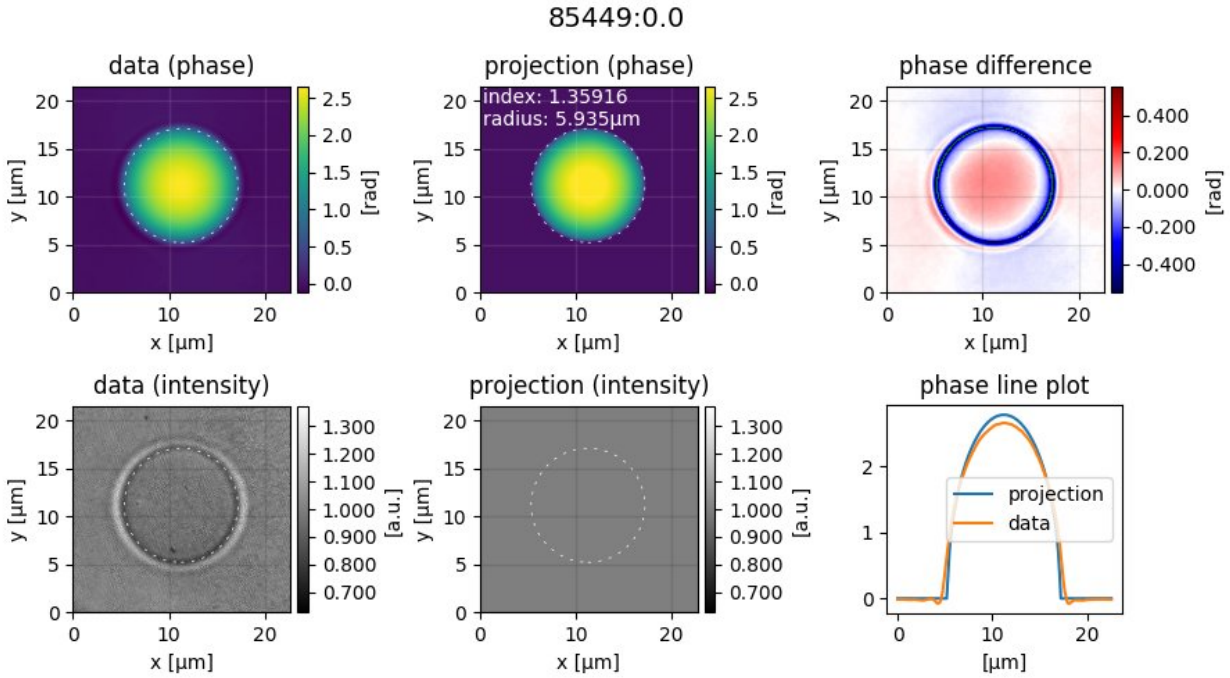
Let's have a look at *sensor_roi_images.tif* (using Fiji or Windows Photo Viewer). This is the first image stored in the tif file:



It shows the full sensor phase image of the first bead. The white rectangle indicates the ROI that was found by DryMass, labeled with the identifier *85449:0.0*. The file *sensor_roi_images.tif* allows you to check that DryMass has

correctly found the objects that you are interested in. If the beads were not detected correctly, we would probably have to adjust the size parameter in the *Configuration file* (see also *dm_extract_roi*).

It appears that DryMass has correctly found all beads with the default settings. Next, open the file *sphere_edge_projection_images.tif*. The identifier of the first ROI is shown at the top, the first column contains phase and intensity of the experimental data, the second column contains the modeled data (with refractive index and radius used for the simulation), and the third column shows the phase-difference as well as line plots through the phase images.



Note that the modeled intensity image is all-one, because the projection model only models the optical thickness and thus only affects the phase data. Also, note that the phase-difference image between data and model only has small deviations in the background phase. If there were large ramp-like structures or offsets, we would have to modify the *background correction*.

4.1.5 Post-processing

A closer examination of the phase-difference images shows that there seem to be either deformed beads or imaging artifacts in the images with the identifiers (prepend *85449:*): *3.0, 6.0, 23.0, 25.0, 26.0, 34.0, 35.0, 38.0, 39.0, 50.0, 51.0, 54.0, 57.0, 59.0, 63.0, 66.0,* and *70.0*. Due to their asymmetry we ignore these images in our analysis by removing the respective rows from *sphere_edge_projection_statistics.txt* (Note that this file will be overridden when `dm_analyze_sphere` is executed again). We can then load the statistics file into a statistical analysis application and compute the average and the standard deviation of the refractive index. In Python, this can be done with

```
import numpy as np
ri = np.loadtxt("sphere_edge_projection_statistics.txt", usecols=(1,))

print("average: ", np.average(ri))
print("standard deviation: ", np.std(ri))
```

which will yield a refractive index of 1.357 $\pm$ 0.004 which agrees well with the value given in reference *[SCG+17]* (1.356 $\pm$ 0.004); The small difference can be explained by a slightly modified analysis pipeline and originally more strict selection criteria.

CHAPTER 5

Code reference

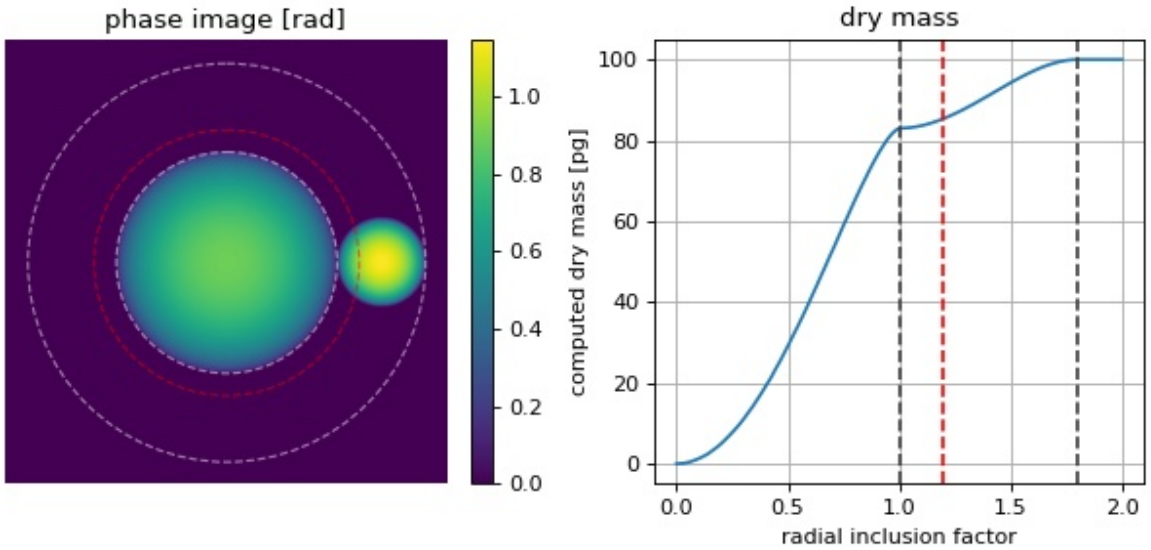
CHAPTER 6

Code examples

6.1 Dry mass computation with radial inclusion factor

This examples illustrates the usage of the "radial inclusion factor" which is defined in the configuration section "sphere" and used in `drymass.anasphere.relative_dry_mass()` with the keyword argument *rad_fact*.

The phase image is computed from two spheres whose dry masses add up to 100pg with the larger sphere having a dry mass of 83pg. The larger sphere is located at the center of the image which is also used as the origin for dry mass computation. The radius of the larger sphere is known (10µm). Thus, the corresponding radius (inner circle) corresponds to a radial inclusion factor of 1. In DryMass, the default radial inclusion factor is set to 1.2 (red). In some cases, this inclusion factor must be increased or decreased depending on whether additional information (the smaller sphere) should be included in the dry mass computation or not.



mass_radial_inclusion_factor.py

```python
from drymass.anasphere import relative_dry_mass
import matplotlib
import matplotlib.pylab as plt
import numpy as np
import qpimage
import qpsphere

# refraction increment
alpha = .18  # [mL/g]

# general simulation parameters
medium_index = 1.333
model = "projection"
wavelength = 500e-9  # [m]
pixel_size = 1e-7  # [m]
grid_size = (400, 400)  # [px]

# sphere parameters
dry_masses = [83, 17]  # [pg]
radii = [10, 4]  # [µm]
centers = [(200, 200), (200, 340)]  # [px]

phase_data = np.zeros(grid_size, dtype=float)
for m, r, c in zip(dry_masses, radii, centers):
    # compute refractive index from dry mass
    r_m = r * 1e-6
    alpha_m3g = alpha * 1e-6
    m_g = m * 1e-12
    n = 1.333 + 3 * alpha_m3g * m_g / (4 * np.pi * (r_m**3))
    # generate example dataset
    qpi = qpsphere.simulate(radius=r_m,
                            sphere_index=n,
                            medium_index=medium_index,
                            wavelength=wavelength,
                            pixel_size=pixel_size,
                            model=model,
                            grid_size=grid_size,
                            center=c)
    phase_data += qpi.pha

qpi_sum = qpimage.QPImage(data=phase_data,
                          which_data="phase",
                          meta_data={"wavelength": wavelength,
                                     "pixel size": pixel_size,
                                     "medium index": medium_index})

# compute dry mass in dependence of radius
mass_evolution = []
mass_radii = []
for rad_fact in np.linspace(0, 2.0, 100):
    dm = relative_dry_mass(qpi=qpi_sum,
                           radius=radii[0] * 1e-6,
                           center=centers[0],
                           alpha=alpha,
                           rad_fact=rad_fact)
    mass_evolution.append(dm * 1e12)
    mass_radii.append(rad_fact)
```

```
58
59  # plot results
60  fig = plt.figure(figsize=(8, 3.8))
61  matplotlib.rcParams["image.interpolation"] = "bicubic"
62  # phase image
63  ax1 = plt.subplot(121, title="phase image [rad]")
64  ax1.axis("off")
65  map1 = ax1.imshow(qpi_sum.pha)
66  plt.colorbar(map1, ax=ax1, fraction=.048, pad=0.05)
67  # dry mass vs. inclusion factor
68  ax2 = plt.subplot(122, title="dry mass")
69  ax2.plot(mass_radii, mass_evolution)
70  ax2.set_ylabel("computed dry mass [pg]")
71  ax2.set_xlabel("radial inclusion factor")
72  ax2.grid()
73  # radius indicators
74  for r in [100, 180]:
75      cx = centers[0][0] + .5
76      cy = centers[0][1] + .5
77      circle = plt.Circle((cx, cy), r,
78                          color='w', fill=False, ls="dashed", lw=1, alpha=.5)
79      ax1.add_artist(circle)
80      ax2.axvline(r / 100, color="#404040", ls="dashed")
81  # add default
82  circle = plt.Circle((cx, cy), 120,
83                      color='r', fill=False, ls="dashed", lw=1, alpha=.5)
84  ax1.add_artist(circle)
85  ax2.axvline(1.2, color="r", ls="dashed")
86
87  plt.tight_layout()
88  plt.show()
```

6.2 Comparison of relative and absolute dry mass

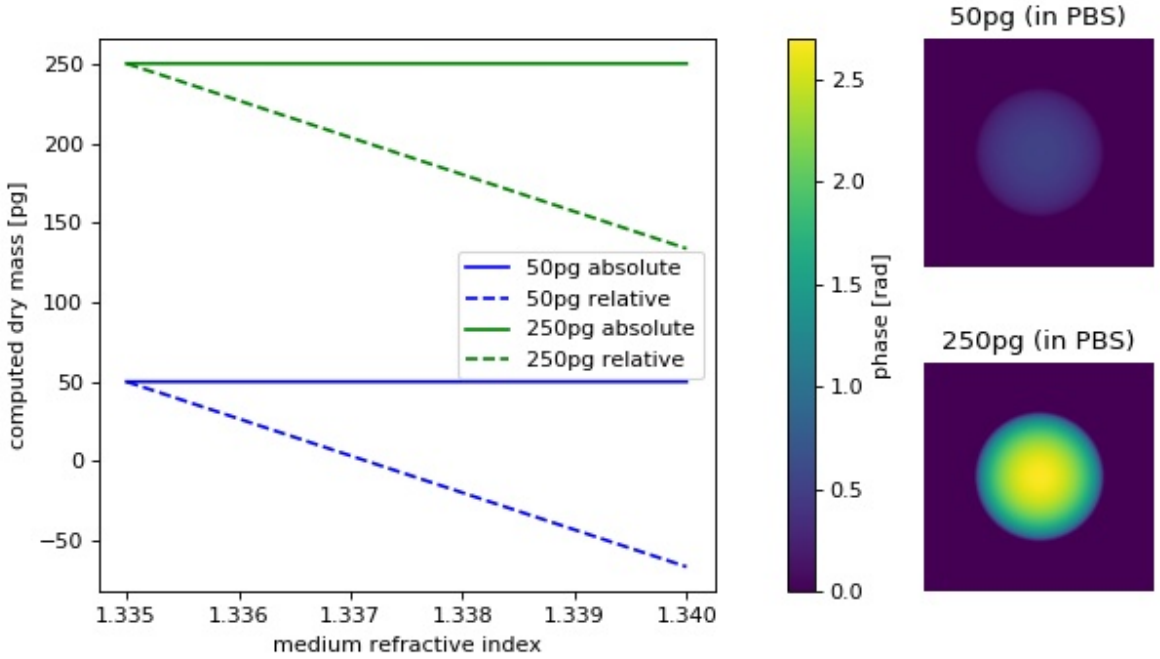
Relative dry mass is the dry mass computed relative to the surrounding medium. If the refractive index of the surrounding medium does not match that of the intracellular fluid (approximately 1.335), then the relative dry mass underestimates the actual dry mass. For a spherical cell, the absolute (corrected) dry mass can be computed as described in the theory section on *dry mass computation*.

This examples compares the relative dry mass (`drymass.ansphere.relative_dry_mass()`) to the absolute dry mass corrected for a spherical phase object (`drymass.ansphere.absolute_dry_mass_sphere()`). From simulated phase images (projection approach, wavelength 550nm) of two cell-like spheres with a radius of 10µm and dry masses of 50pg (n1.337) and 250pg (n1.346), the absolute and relative dry masses are computed with varying refractive index of the medium.

At the refractive index of phosphate buffered saline (PBS), absolute and relative dry mass are equivalent. As the refractive index of the medium increases, the relative drymass decreases linearly (independent of dry mass), underestimating the actual dry mass.

`mass_relative_vs_absolute.py`

```
1  from drymass.anasphere import absolute_dry_mass_sphere, relative_dry_mass
2  import matplotlib
3  import matplotlib.pylab as plt
4  import numpy as np
5  import qpsphere
```




```
6
7   # refraction increment
8   alpha = .18  # [mL/g]
9
10  # general simulation parameters
11  model = "projection"
12  wavelength = 500e-9  # [m]
13  pixel_size = 1.8e-7  # [m]
14  grid_size = (200, 200)  # [px]
15
16  # sphere parameters
17  radius = 10  # [µm]
18  center = (100, 100)  # [px]
19
20  dry_masses = [50, 250]  # [pg]
21  medium_indices = np.linspace(1.335, 1.34, 5)
22
23  qpi_pbs = {}
24  m_abs = {}
25  m_rel = {}
26  phase_data = np.zeros(grid_size, dtype=float)
27  for m in dry_masses:
28      # initiate results list
29      m_abs[m] = []
30      m_rel[m] = []
31      # compute refractive index from dry mass
32      r_m = radius * 1e-6
33      alpha_m3g = alpha * 1e-6
34      m_g = m * 1e-12
35      n = 1.335 + 3 * alpha_m3g * m_g / (4 * np.pi * (r_m**3))
36      for medium_index in medium_indices:
37          # generate example dataset
```

```
        qpi = qpsphere.simulate(radius=r_m,
                                sphere_index=n,
                                medium_index=medium_index,
                                wavelength=wavelength,
                                pixel_size=pixel_size,
                                model=model,
                                grid_size=grid_size,
                                center=center)
        # absolute dry mass
        ma = absolute_dry_mass_sphere(qpi=qpi,
                                      radius=r_m,
                                      center=center,
                                      alpha=alpha,
                                      rad_fact=1.2)
        m_abs[m].append(ma * 1e12)
        # relative dry mass
        mr = relative_dry_mass(qpi=qpi,
                               radius=r_m,
                               center=center,
                               alpha=alpha,
                               rad_fact=1.2)
        m_rel[m].append(mr * 1e12)
        if medium_index == 1.335:
            qpi_pbs[m] = qpi

# plot results
fig = plt.figure(figsize=(8, 4.5))
matplotlib.rcParams["image.interpolation"] = "bicubic"
# phase images
kw = {"vmax": qpi_pbs[dry_masses[1]].pha.max(),
      "vmin": qpi_pbs[dry_masses[1]].pha.min()}

ax1 = plt.subplot2grid((2, 3), (0, 2))
ax1.set_title("{}pg (in PBS)".format(dry_masses[0]))
ax1.axis("off")
map1 = ax1.imshow(qpi_pbs[dry_masses[0]].pha, **kw)

ax2 = plt.subplot2grid((2, 3), (1, 2))
ax2.set_title("{}pg (in PBS)".format(dry_masses[1]))
ax2.axis("off")
ax2.imshow(qpi_pbs[dry_masses[1]].pha, **kw)

# overview plot
ax3 = plt.subplot2grid((2, 3), (0, 0), colspan=2, rowspan=2)
ax3.set_xlabel("medium refractive index")
ax3.set_ylabel("computed dry mass [pg]")
for m, c in zip(dry_masses, ["blue", "green"]):
    ax3.plot(medium_indices, m_abs[m], ls="solid", color=c,
             label="{}pg absolute".format(m))
    ax3.plot(medium_indices, m_rel[m], ls="dashed", color=c,
             label="{}pg relative".format(m))
ax3.legend()
plt.colorbar(map1, ax=ax3, fraction=.048, pad=0.1,
             label="phase [rad]")

plt.tight_layout()
plt.subplots_adjust(wspace=.14)
plt.show()
```

CHAPTER 7

Changelog

List of changes in-between DryMass releases.

7.1 version 0.1.2

- allow to disable sensor image tif export
- support input QPI data with different shapes
- fixed wrong assumption about definition of dry mass in cells (water vs. intracellular salt solution)
- allow to use separate file or series index for background correction with experimental data (#6)
- small visualization improvements

7.2 version 0.1.1

- add file name to image identifier and update plotting (#4)
- renamed “object” to “identifier” in statistics output
- binary-based background correction (manual or automated threshold)
- allow to disable background correction with “[bg] enabled = False”
- ask for missing meta data keys in CLI only when they are required

7.3 version 0.1.0

- initial release

CHAPTER 8

Bilbliography

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Bibliography

[Bar52] R. Barer. Interference Microscopy and Mass Determination. *Nature*, 169(4296):366–367, 1952. doi:10.1038/169366b0.

[Bar53] R. Barer. Determination of dry mass, thickness, solid and water concentration in living cells. *Nature*, 172:1097–1098, 1953. doi:10.1038/1721097a0.

[Bar57] R. Barer. Refractometry and interferometry of living cells. *Journal of the Optical Society of America*, 47(6):545, jun 1957. doi:10.1364/josa.47.000545.

[BJ54] R. Barer and S. Joseph. Refractometry of Living Cells Part I. Basic Principles. *Quarterly Journal of Microscopical Science*, 171(2):38P–39P, 1954. doi:10.1038/171720a0.

[HGC94] Y. Harpaz, M. Gerstein, and C. Chothia. Volume changes on protein folding. *Structure*, 2(7):641–649, jul 1994. doi:10.1016/s0969-2126(00)00065-4.

[KvB07] B. Kemper and G. von Bally. Digital holographic microscopy for live cell applications and technical inspection. *Applied Optics*, 47(4):A52, oct 2007. doi:10.1364/ao.47.000a52.

[LL02] M. Lehmann and H. Lichte. Tutorial on off-axis electron holography. *Microscopy and Microanalysis*, 8(06):447–466, dec 2002. doi:10.1017/s1431927602020147.

[SCG+17] M. Schürmann, G. Cojoc, S. Girardo, E. Ulbricht, J. Guck, and P. Müller. 3d correlative single-cell imaging utilizing fluorescence and refractive index tomography. *Journal of Biophotonics*, pages n/a, aug 2017. doi:10.1002/jbio.201700145.

[SSM+16] M. Schürmann, J. Scholze, P. Müller, J. Guck, and C. J. Chan. Cell nuclei have lower refractive index and mass density than cytoplasm. *Journal of Biophotonics*, 9(10):1068–1076, oct 2016. doi:10.1002/jbio.201500273.