Photizo

Package for analysis of Fourier Transform Infrared Spectroscopy data

v. 0.0.1

Author: Melissa Grant-Peters

Affiliation: Wellcome Centre for Human Genetics, University of Oxford

1.0 Requirements

This package was written for data exported from OPUS 7.4 in a .dpt format. Prior to exporting from OPUS, we recommend Cutting the spectrum to exclude noisy areas, applying atmospheric compensation and baseline correction.

This package was written in Python 3.7.4 with the following package versions:

pandas==1.1.3
numpy==1.19.2
matplotlib==3.3.2
astropy==4.0.3
scikit-image==0.17.2
scikit-learn==0.23.2
scipy==1.5.3
umap==0.1.1
scanpy==1.6.0
leidenalg==0.8.2
anndata==0.7.4

2.0 Functions

2.1 Pre-processing

bracketPoint

Description: This function identifies the closest index corresponding to a given wavelength

Inputs:

X: a vector of numbers ordered from smallest to largest

x: a real valued number

Output:

returns the index closest to the given wavenumber

holeExclusion

Description: This function excludes spectra likely to have no sample or with scattered light. We recommend visualising Y_zeros spatially to confirm that excluded spectra overlapp with holes or regions with damaged sample

Inputs:

X: a vector with wavelengths

Y: a matrix with all spectra

Output:

returns Y_zeros - matrix with spectra from holes or with scattered light converted to zero

Y_nozeros - matrix without spectra from holes or with scattered light

idx - indexes of positions of excluded spectra

makeRGB

Description: This function plots an image based on three matrices with values corresponding to red, green and blue intensity.

Inputs:

Three matrices of the same size, each of which correponds to the R, G or B value of a given pixel

Output:

Image corresponding to the merged RGB matrices

matrixImage

Description: This function reshapes a vector into the given matrix dimensions

Inputs:

sqrsx: Number of pixels in the X axis

sqrsy: Number of pixels in the Y axis

Y0: Vector containing cluster classification of all pixels of a sample. This vector must have a value of -99 inserted in the positions where spectra were excluded.

Output:

returns matrix of cluster classification in given coordinates

orderVectors

Description: This function reorders both the X vector with wavelengths and the Y matrix so that the X vector has ascending wavelengths. This is a necessary step prior to integration.

Inputs:

X: Vector containing wavelengths

Y: Matrix containing all spectra

Output:

Xnew: Reordered vector containing wavelengths in ascending order

Ynew: Reordered matrix according to reordered wavelengths

reinsertHoles

Description: This function reinserts a value 'val' given by the user in a vector in the positions where spectra where excluded. This enables visualisation of data spatially. 'val' must be a float.

Inputs:

Y: A matrix without the excluded values

idx: a vector with the position of excluded spectra

val: value to be inserted (default: -99.0)

Output:

Yh: Matrix with val inserted in position of excluded values

removeCO2

Description: This function excludes the region containing CO_2 absorbance.

Inputs:

X: a vector with wavelengths

Y: a matrix with all spectra

Output:

X: Modified wavelength vector which does not include 2,250-2,400 spectral region

Y2: Modified spectral matrix which does not include 2,250-2,400 spectral region

selectRegionXY

Description: Function for subsetting the wavelength vector X and matrix Y to only include values within a certain wavenumber range of interest.

Inputs:

x0: Starting wavelength of interest

x1: Ending wavelength of interest

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

X: Subsetted wavelengths only including region of interest

Y: Matrix subset within given spectral region between WN1 and WN2 wavelengths

selectRegionY

Description: This function subsets only the matrix Y to include values within a certain wavenumber range of interest.

Inputs:

x0: Starting wavelength of interest

x1: Ending wavelength of interest

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

Y: Matrix subset within given spectral region between x0 and x1 wavelengths

vectorNormalisation

Description: This function performs vector normalisation on the matrix with all spectra of a sample.

Inputs:

Y: Matrix for normalisation, preferably already having undergone hole exclusion.

Output:

Y: Vector normalised matrix of spectra.

2.2 PCA

PCA

Description: This is a function for PCA which shows a plot of the cumulative explained variance ratio (elbow plot) and the top 3 eigenspectra.

It returns the pca result, which can then be used for specific plotting e.g.: for top PC projection scatter plots.

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix without excluded spectra

n: number of principal components

Output:

Shows Cumulative explained variance

Shows top 3 eigen-spectra

Returns PCA explained variance ratio, pca components and data after fit transformation

Note:

Performing PCA on spectroscopic data can benefit from taking as input the second derivative of spectra, since this controls for the baseline.

PCAprojection

Inputs:

m: First principal component of interest

n: Second principal component of interest

fit: matrix containing fitted data

EVR: list containing all explained variance ratios

Output:

Graph plotting PCm vs PCn

Note:

To colour points based on metadata, modify the plt.scatter line to include c=list_of_metadata_values and cmap=colourmap_of_choice

2.3 Visualisation

cluster

Description: This function is designed to facilitate clustering with UMAP, by calculating nearest neighbours, dimensionality reduction with UMAP and community detection with the Leiden algorithm.

Input:

adata: adata object with spectral information in adata.X and spectral annotation in adata.obs

n_neigh: number of neighbours for subsequent UMAP calculation. Default: 15

spread: spread value for UMAP. Default: 1.0

min_dist: minimum distance for UMAP. Default: 0.1

leiden_res: leiden resolution for community detection. Default: 0.1

Output:

adata object with new UMAP and leiden variables.

2.3 Visualisation

clusterMap

Description: This function plots spatially the cluster classification of a given sample. This function is designed to map uFTIR data with known dimensions of pixels/spectra. The user may modify the input RGB code for each cluster as needed, with example values given here:

Turquoise = (0.4, 0.76078431, 0.64705882)
Orange = (0.98823529, 0.55294118, 0.38431373)
Pink = (0.90588235, 0.54117647, 0.76470588)
Green = (0.65098039, 0.84705882, 0.32941176)
Purple = (0.60, 0.48, 0.8)
Blue = (0.55294118, 0.62745098, 0.79607843)
Tan = (0.89803922,0.76862745, 0.58039216)
Yellow = (1., 0.85098039, 0.18431373)
Grey = (0.55294118, 0.55294118, 0.55294118)
Black = (0, 0, 0)

Inputs:

sqrsx: Number of pixels in the X axis

sqrsy: Number of pixels in the Y axis

Y0: Vector containing cluster classification of all pixels. This vector must have a value of -99 inserted in the positions where spectra were excluded (See function **reinsertHoles**).

Output:

returns image of mapped cluster classification

integrationVisualisation

Description: This function is for visualising the mean spectrum of a given sample with the corresponding standard deviation.

By inserting values of a region of interest in x0 and x1, one can visually inspect the window in relation to the spectrum prior to quantitative analysis (e.g.: prior to integrating, one may wish to check that the peaks are within the chosen window).

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix without excluded spectra

title: Plot title

x0: Lower limit of region of interest

x1: Upper limit of region of interest

Output:

Vector containing calculated area of interest for each spectra controlling for the spectral baseline

plotSpectra

Description: This plots a graph of given spectra. It may be used to plot the entire spectra or, paired with the **selectRegionXY** function may be used to plot only a particular region of interest.

Inputs:

Cluster_means: matrix containing in each column the mean spectrum values of a given cluster

X: vector containing wavelengths

title: String containing title of plot

Newcolors: RGB codes for plotting custom colours

Output:

Returns graph plotting all mean spectra

spatialPlotROI

Description: This function plots spatially the semi-quantitative value calculated via integration of a region of interest in the spectrum. We recommend this function be used for inspection of spectral exclusion.

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with excluded spectra reinserted as 0.0

x0: Lower limit of region of interest

x1: Upper limit of region of interest

a: number of pixels in the X (horizontal) axis

b: number of pixels in the Y (vertical) axis

Output:

Vector containing calculated area of interest for each spectra controlling for the spectral baseline

2.4 Quantification

adjustedIntegrate

Description: Integrates the function generated by linearly interpolating the data X, Y between x0 and x1. Unlike nIntegrate, this function controls for baseline differences by subtracting the area between the lowest point of the curve and zero.

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

x0: starting wavelength of interest

x1: end wavelength of interest

Output:

Vector containing calculated area of interest for each spectra controlling for the spectral baseline

nIntegrate

Description: Integrates the function generated by linearly interpolating the data X, Y between x0 and x1 (given in wavenumber).

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

x0: starting wavelength of interest

x1: end wavelength of interest

Output:

Vector containing calculated area of interest for each spectra

alphaHelix

Description: This function estimates the % alpha helix secondary structure folding in the sample. Based on method described by Goormaghtigh et al, 2009

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

% contribution of alpha-helix folding

betaSheet

Description: This function estimates the % beta-sheet secondary structure folding in the sample. Based on method described by Goormaghtigh et al, 2009

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

% contribution of beta-sheet folding based on method described by Goormaghtigh et al, 2009

betaTurn

Description: This function estimates the % beta-turn secondary structure folding in the sample. Based on method described by Goormaghtigh et al, 2009

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

% contribution of Beta-turn folding based on method described by Goormaghtigh et al, 2009

Random

Description: This function estimates the % random secondary structure folding in the sample. Based on method described by Goormaghtigh et al, 2009

Inputs:

X: Vector with ordered wavelengths

Y: Ordered matrix with all spectra

Output:

% contribution of random folding based on method described by Goormaghtigh et al, 2009

secondDerivative

Description: This function calculates numerically the second derivative of spectra. Note: due to the method used which takes into consideration the neighbouring values, values near the edges (upper and lower wavenumbers) become less reliable and may need to be excluded.

Inputs:

X: a vector with wavelengths

Y: a matrix with all spectra

order: polynomial order for second derivative

Output:

dY: matrix with second derivative of all spectra

shows a plot of the mean second derivative spectrum with standard deviation