The `stmpy` package

Hoffman lab

June 27, 2025

Methods in `stmpy.tools`:

Descriptions are shown in the order functions appear in `stmpy.tools`, which is vaguely chronologically.

1. `saturate` - Designed to make it easy to set color limits on images. Adjusts color axis of current image handle by calculating a probability density function for the data in the current axis. Uses upper and lower thresholds on the PDF to find sensible c-axis limits.
2. `azimuthalAverage` - Given a point $\mathbf{p} = (x_0, y_0)$ in a 2D data set $F(x, y)$, computes the azimuthal average of the F as a function of r away from $\mathbf{p}$. Uses 2D interpolation on F to get evenly spaced r values.
3. `azimuthalAverageRaw` - Computes a raw azimuthal average on 2D data $F(x, y)$. This is similar to `azimuthalAverage` but does not interpolate the data. Instead the returned r values are not linearly spaced, but follow the sequence: $1, \sqrt{2}, 2, \sqrt{5}...$
4. `binData` - Puts non-linearly sampled data into linear bins.
5. `linecut` - Simple algorithm for taking a line-cut on a 2D data set $F(x, y)$. Uses interpolation to sample F along a line from (x_1, y_1) to (x_2, y_2) with n evenly spaced points.
6. `squareCrop` - Crops a 2D image to be $m \times m$.
7. `lineCrop` - Takes 1D data $y(x)$ and removes arbitrarily many sections to return $y(x(t_0 : t_1, t_2 : t_3, ...))$.
8. `removePolynomial1d` - Removes an n degree polynomial fit to 1D data $y(x)$. Optional: can specify the sections of $y(x)$ to use when fitting the background polynomial.
9. `lineSubtract` - Acts on 3D or 2D data $A(\mathbf{r})$ to remove an n degree polynomial from each line in $A(\mathbf{r})$. Specifically, it iterates over the first index of $A(\mathbf{r})$ until to get 1D data $y(x)$ and implements `removePolynomial1d` to remove the background.
10. `fitGaussian2d` - Fit a 2D gaussian of the form

$$f(x, y) = A \exp\big(-a(x - x_0)^2 + 2b(x - x_0)(y - y_0) - c(y - y_0)^2\big) + B$$

    to 2D data $F(x, y)$ given specified initial parameters: $A, B, x_0, y_0, \sigma_x, \sigma_y, \theta$.
11. `findOtherBraggPeaks` - For a Fourier transformed lattice, the Bragg peaks come in pairs at $\pm\mathbf{Q}_B$, with harmonics at $\pm n\mathbf{Q}_B$ with $n \in \mathbb{N}$. This function takes on Bragg peak and returns the $2n - 1$ other Bragg peak locations for 2D data $F(x, y)$.
12. `findPeaks` - Simple peak detection algorithm that returns the location of the n highest peaks, x^*, in 1D data $y(x)$ by checking where the derivative crosses zeros, $y'(x^*) = 0$.

13. `fitGaussian1d` - Fits N gaussians to 1D data $y(x)$ of the form

$$f(x) = \sum_{n=0}^{N} A_n \exp\left(-\frac{(x-\mu_n)^2}{2\sigma_n^2}\right).$$

14. `foldLayerImage` - Takes 3D data $A(\mathbf{r})$ and returns a n-fold symmetric 3D image $\tilde{A}(\mathbf{r})$, by iterating through the first index of $A(\mathbf{r})$ and symmetrizing the i^{th} 2D layer, $A(E_i, x, y)$, about a specified fold direction. The intended use is to symmetrize an FT-DOS map along the direction of a Bragg peak. Currently implemented for $n = 1, 2, 4$ and all but replaced by `symmetrize`

15. `quickFT` - Computes a 2D Fourier transform of 2D or 3D data $A(\mathbf{r})$, with the option to n-fold symmetrize the result. If 3D data is used the 2D Fourier transforms will be computed by iterating along the first index.

16. `symmetrize` - Similar to `foldLayerImage`, returns n-fold symmetric 2D or 3D data $\tilde{A}(\mathbf{r})$ by rotating clockwise and anti-clockwise by an angle $2\pi/n$, then applying a mirror line. Works on 2D and 3D data sets, in the case of 3D each layer is symmetrized.

17. `ngauss1d` - More general version of `fitGaussian1d`, which allows any fit parameter to be fixed. Also returns information about the quality of fit.

18. `track_peak` - Generalizes `ngauss1d` to work on 2D data $F(x, y)$ by iterating the first index of $F(x, y)$. Only retains information about the position of the gaussian peaks, which track features that disperse in the y direction.

19. `shearcorr` - ... to be updated to include local drift correction.

20. `planeSubtract` - Removes a 2D polynomial plane $P(x, y)$ from 2D data $F(x, y)$. The polynomial is of the form:

$$P(x, y) = a_0 + \sum_{k=1}^{N} a_{2k-1}\ x^k + a_{2k}\ y^k,$$

where $a_0, a_1, a_2, ...$ are the polynomial coefficients.

21. `butter_lowpass_filter` - Implements a Butterworth filter for 1D data $y(x)$ or for each spectrum $g(E)$ in a 3D data set $g(E, x, y)$.

22. `gradfilter` - Applies a minimum gradient filter to extract dispersive features in a 2D data set $F(x, y)$ (Ref: arXiv:1612.07880). Returns filtered data with optional gradient components for pseudo-vector-field and gradient modulus maps.