FittingFunctions2.0

Improved version of the fittingfunctions python module used by the Physics Institution and LTH at Lund University.

Made in python version 3.11.5

> Authored by Erik Ewald & Gustav Sjövall, 2023-2024

Contents

Importing

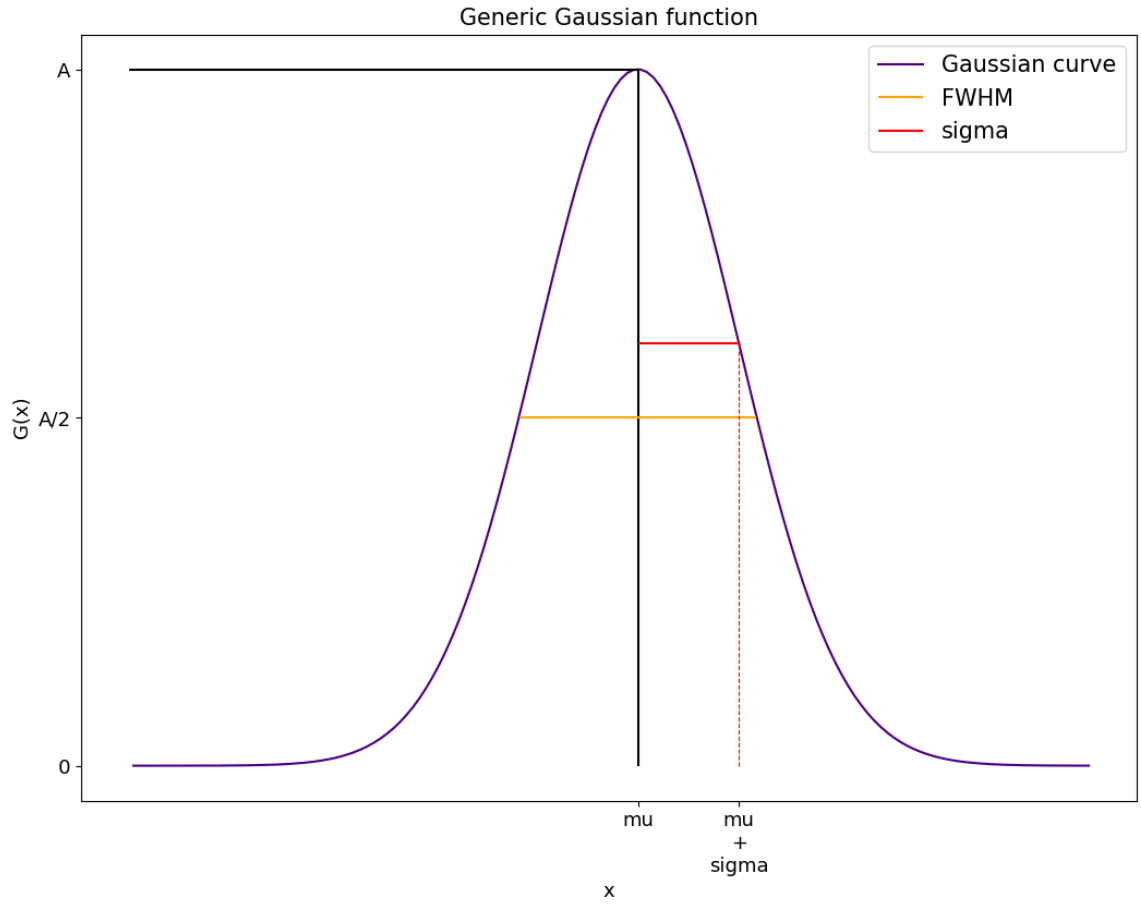
To import the module, this python file is placed in the working directory for the project and imported into the main file as

```
import FittingFunctions2 as ff
```

Gaussian Functions

For the following scripts the definition of a Gaussian function is as follows:

$$G(x) = A \cdot exp\left(-\frac{(x-mu)^2}{2\cdot(sigma)^2}\right)$$



gaussian class

The gaussian class holds the return values from all functions that fit a Gaussian function and is not intended to be used independently. **Printing** a gaussian object gives rounded values. Raw values are accessed as public members or methods. Raw values for uncertainties have to be derived from the covariance matrix.

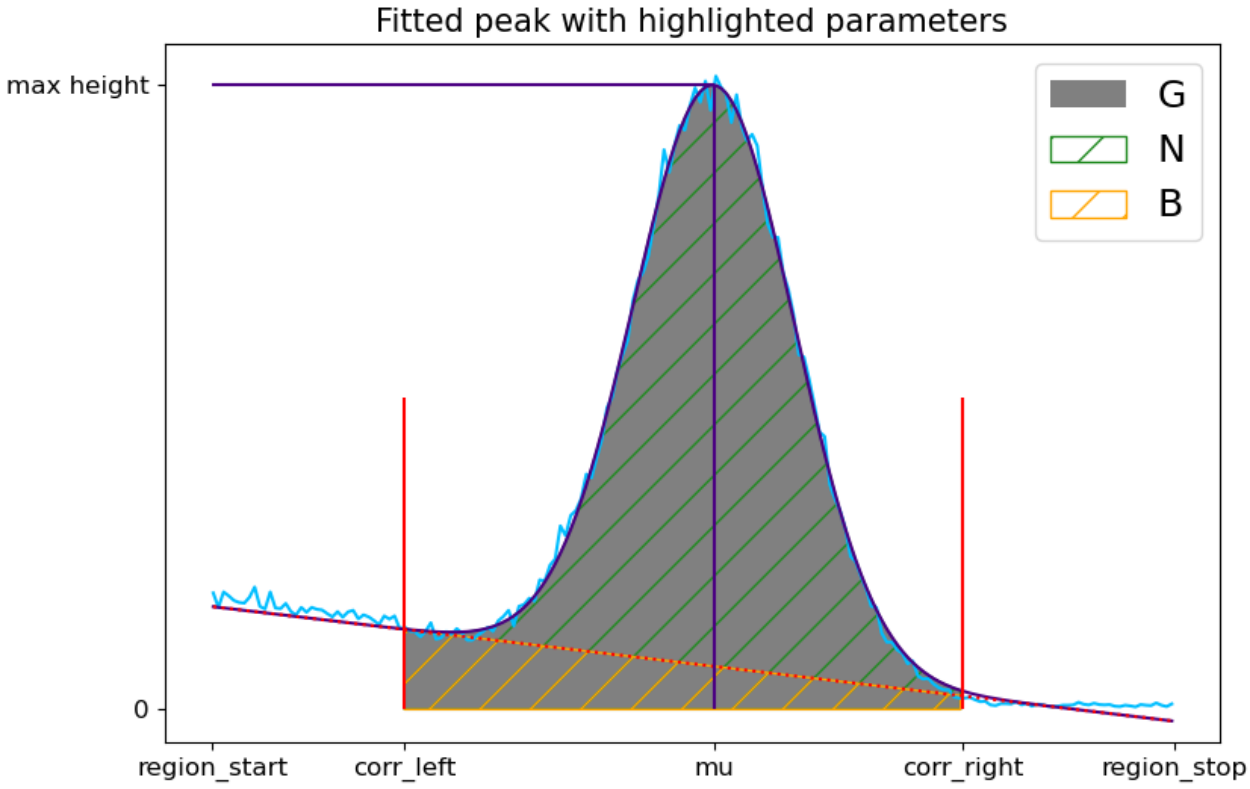
Printing

Example of a gaussian fitted to a Cs-137 peak in a gamma ray spectrum.

```
[Python3]: print(BGO["662kev"])
========================= Gaussian Peak =========================
Estimated parameters: A = 1740.5264, mu = 661.8352, sigma = 40.6196
Uncertainties: σ(A) = 7.7571, σ(mu) = 0.2089, σ(sigma) = 0.2096
Additional info: Max height = 1915.892, G = 79320, N = 65096, B = 14224

Covariance matrix:
 [[ 6.01726192e+01  1.87206691e-03 -9.40087680e-01]
 [ 1.87206691e-03  4.36592014e-02 -9.98241359e-05]
 [-9.40087680e-01 -9.98241359e-05  4.39143608e-02]]
```

Generic peak



Public members

Member	Description
*gaussian.***A** : *float*	The amplitude of the fitted function. OBS! This does not account for the scatter correction and will sometimes not match the peak height in a spectrum, see *gaussian.max_height()*.
*gaussian.***mu** : *float*	The central value of the fitted gaussian curve.
*gaussian.***sigma** : *float*	The fitted value of sigma.
*gussian.***cov_matrix** *3x3-array*	The covariance matrix from the fit corresponding to *A, mu* and *sigma*.
*gussian.***G** : *float*	The gross area of the region enclosed by the scatter correction boundaries., i.e. the sum of all y-values in the selected region.
*gussian.***B** : *float*	The scatter background calculated as the sum of the linear scatter correction function across the region enclosed by the scatter correction boundaries.
*gussian.***N** : *float*	The net area of the peak, i.e. *gussian.***G** - *gussian.***B**.

Public Methods

Method	Description

Method	Description
*gussian.***area()** : *-> float*	The analytically calculated area of the gaussian. OBS! This may deviate from *gaussian.G* as the bin size in the x-data is not accounted for by this function.
*gussian.***FWHM(uncertanty = False** *: bool* **)** : *-> float/tuple*	Return the full-width-half-max value of the peak. Setting *uncertanty* till *True* return a tuple with the FWHM as the first element the uncertainty in FWHM as the second.
*gussian.***max_height()** : *-> float*	Returns the real height of the gaussian at the point *mu*. Calculated as corr_f(mu) + A.
*gussian.***plot(xlabel = None** *: str* **, ylabel = None** *: str* **)** : *-> None*	Plots the gaussian over the selected region.
*gussian.***value(x** *: float* **)** : *-> float*	Returns the value of the gaussian at the point *x*.

Fitting Functions

fit_gaussian

```
fit_gaussian(X, Y, region_start, region_stop,
             corr_left=None, corr_right=None,
             mu_guess=None, A_guess=None, sigma_guess=None,
             scatter_corr="auto", scatter_corr_points=3,
             corr_thresh=0.05)
```

Attempts to fit a gaussian function to a data set using the *curve_fit* routine from scipy.optimize.

Parameter	Description
X : *array_like*	An iterator corresponding to the x-axis of the data set.
Y : *array_like*	An iterator corresponding to the y-axis of the data set.
region_start : *float*	The lower bound of the region on which the gaussian will be fitted.
region_stop : *float*	The upper bound of the region on which the gaussian will be fitted.
corr_left : *float, optional*	Left point at which scatter correction will be made. This only affects the scatter correction, the fit will always be made over the entire region.
corr_right : *float, optional*	Right point at which scatter correction will be made. This only affects the scatter correction, the fit will always be made over the entire region.
mu_guess : *float, optional*	Manual guess for *mu*.

Parameter	Description
A_guess : *float, optional*	Manual guess for *A*.
sigma_guess : *float, optional*	Manual guess for *sigma*.
scatter_corr : *str/bool, optional*	Sets method of scatter correction. `'auto'` attempts automatic scatter correction, if this fails *region_start* and *region_stop* will be set as correction boundaries. Manually setting boundaries will overrule the automatically set boundaries. `True` makes scatter correction without automatically set limits. Limits are manually set by *corr_left* and *corr_right*, if one or both of these are not set *region_start* and *region_stop* will be set as correction points. `False` no scatter correction is made.
scatter_corr_points : *int, optional*	The number of points over which the bounds of the scatter correction is averaged. The default 3 means the point is chosen as the average of the first 3 inside the gaussian region.
corr_thresh : *float, optional*	Sets the gradient threshold for determining the edges of the peak when `sactter_corr='auto'`.

Returns	Description
Peak : *gaussian*	An object containing the data from the fit. See the section on the *gaussian* class.

`fit_double_gaussian`

Attempts to fit a sum of two gaussian functions to a data set using the *curve_fit* routine from scipy.optimize.

```
fit_double_gaussian(X, Y, region_start, region_stop, split_point,
                    corr_left=None, corr_right=None,
                    plotting = False,
                    mu1_guess=None, mu2_guess=None,
                    A1_guess=None, A2_guess=None,
                    sigma1_guess=None, sigma2_guess=None,
                    scatter_corr=True, scatter_corr_points=3)
```

Parameter	Description
X : *array_like*	An iterator corresponding to the x-axis of the data set.
Y : *array_like*	An iterator corresponding to the y-axis of the data set.
split_point : *float*	A point in between the two peaks that split the region for the purpose of automatically assigning guesses.
region_start : *float*	The lower bound of the region on which the gaussian will be fitted.

Parameter	Description
region_stop : *float*	The upper bound of the region on which the gaussian will be fitted.
corr_left : *float, optional*	Left point at which scatter correction will be made. This only affects the scatter correction, the fit will always be made over the entire region.
corr_right : *float, optional*	Right point at which scatter correction will be made. This only affects the scatter correction, the fit will always be made over the entire region.
mu1_guess / mu2_guess : *float, optional*	Manual guess for *mu*. The first peak is lower on the x-axis than the second peak, *mu1* < *mu2*.
A1_guess / A2_guess : *float, optional*	Manual guess for *A*. The first peak is lower on the x-axis than the second peak.
sigma1_guess / sigma2_guess : *float, optional*	Manual guess for *sigma*. The first peak is lower on the x-axis than the second peak.
scatter_corr : *bool, optional*	Sets if a scatter correction is made.
scatter_corr_points : *int, optional*	The number of points over which the bounds of the scatter correction is averaged. The default 3 means the point is chosen as the average of the first 3 inside the gaussian region.

Returns	Description
Peaks : *tuple*	A tuple containing two *gaussian* objects corresponding to the two peaks, in order of ascending *mu* values. See the section on the *gaussian* class.

Spectrum Calibration

Calibrates a spectrum by marking peaks and providing energies corresponding to the marked peaks.

`calibrate`

```
calibrate(Y, peak_regions, energies, plot=False, gauss=True)
```

Parameter	Description
Y : *array_like*	The dataset corresponding to the y-axis of the spectrum.
peak_regions : *array_like*	An iterator containing start and stop values for the peak regions to be used in the calibration. A *gaussian* object can be given instead of a start and stop region. Example: `[[150, 300], (500, 600), gaussian_peak]`.

Parameter	Description
energies : *array_like*	Iterator with the energies of the peaks at the corresponding index in *peak_regions*.
plot : *bool, optional*	Plots the calibrated spectrum and highlight the points used for calibration.
gauss : *bool, optional*	If `True` a gaussian will be fitted to each selected region using the *fit_gaussian* function and the *mu* value is used as the calibration point. If `False` the maximum point in the region is used as the calibration point.

Returns	Description
calib_x : *numpy.array*	The calibrated x-axis corresponding to the given y-axis.
(k, m) : *tuple*	The calibration coefficients.

Miscellaneous

Miscellaneous functions included in the package that might be of some use.

`slice_spect`

Slices arrays containing strictly increasing values, like a calibrated x-axis, based on the values in the array.

```
slice_spect(spect, *args, low=None, high=None)
```

Parameter	Description
spect : *array_like*	An array containing a calibrated x-axis.
*args : *array_like, optional*	Additional arrays of the same shape as *spect* that will be sliced into the same shape as the returned version of *spect*
low : *float, optional*	The low cut off point for the spectrum.
high : *float, optional*	The high cut off point for the spectrum.

Returns	Description
spectra : *numpy_array / list*	If no **args* are provided only the sliced version of *spect* is returned. If **args* are provided a list is returned containing the sliced spectra in the order they are given, first being the sliced *spect*.

`is_iter`

```
is_iter(obj)
```

Checks if an object is iterable.

Parameter	Description
obj : *any*	Any type of object to be check if it is iterable.

Returns	Description
bool	*True* if *obj* is iterable, else *False*