MathUtils::OnOffFitter Class Reference
[Levenberg-Marquardt minimization]

Inherited by Morphology::PositionFitter, Morphology::SmoothedMapMaker, Morphology::SourceSubstracter, Morphology::Theta2Fitter, Spectrum::CombinedSpectrumFitter, Spectrum::ConfidenceRegionMaker, Spectrum::HardnessRatioLightCurveMaker, Spectrum::LightCurveMaker, Spectrum::LightCurveParamMaker, Spectrum::PeriodogramMaker, Spectrum::SpectrumFitter, and Spectrum::SpectrumFitterBackground.

Detailed Description

Generic class for adjusting a model on independant ON and OFF data.

It fully takes into account the Poisson properties of the data and implements a **Levenberg & Marquardt method for minimization**.

- **Introduction**
- **Optimal number of background events**
- **Goodness of fit estimation**
- **Likelihood Ratio**
- **Negative Signal**

Introduction

In many case, we want to adjust a model on a set of data which consists of two independant measurements, with different livetimes and with poisson statistics. This is the case when one wants to determine the energy spectrum of a source as well as when one want to determine the intrinsic morphology.

We define for each bin (in energy, position or whatsoever):

- N_{ON} the measured number of events in the ON dataset
- N_{OFF} the measured number of events in the OFF dataset
- T_{ON} and T_{OFF} the respective livetimes for the two datasets
- $\beta = T_{ON}/T_{OFF}$ the livetime normalisation
- n_γ and n_h the expected number of gamma and background events in the bin (the number n_γ is for example calculated from the spectrum folded through the detector response)

The probablity of observing N_{ON} and N_{OFF} events when we expect n_γ gamma events and n_h hadrons is given by the formula

$$P(N_{ON}, N_{OFF}|n_\gamma, n_h) = \frac{(n_\gamma + \beta n_h)^{N_{ON}}}{N_{ON}!} e^{-(n_\gamma+\beta n_h)} \times \frac{{n_h}^{N_{OFF}}}{N_{OFF}!} e^{-n_h}$$

To a constant, the log-likelihood reads

$$\mathcal{L} \equiv \log P = N_{ON} \times \log(n_\gamma + \beta n_h) - (n_\gamma + \beta n_h) + N_{OFF} \times \log n_h - n_h$$

Optimal number of background events

Usually, the number of background events n_h is not a parameter of the model (i.e. the spectrum) and must therefore be computed from the observed number of events. This is implemented by the function **MathUtils::OnOffFitter::GetBackground**

The best background estimate is given by the value of n_h which maximizes the log-likelihood $\log L = \log(P)$ for a given n_γ.

It's the solution of a second order linear equation in n_h :

$$\frac{\beta N_{ON}}{n_\gamma + \beta n_h} + \frac{N_{OFF}}{n_h} - (\beta + 1) = 0$$

whose solution is given by:

$$\begin{aligned} C &= \beta \times (N_{ON} + N_{OFF}) - (1+\beta) \times n_\gamma \\ \Delta^2 &= C^2 + 4\beta(\beta+1) \times N_{OFF} \times n_\gamma \\ n_h &= \frac{C + \Delta}{2\beta(\beta+1)} \end{aligned}$$

In the special case $N_{ON} = 0$, this simplifies to $n_h = \frac{N_{OFF}}{\beta+1}$.

In the special case $N_{OFF} = 0$, this simplifies to $n_h = \frac{N_{ON}}{\beta+1} - \frac{n_\gamma}{\beta}$.

The uncertainties on n_h can be calculated from the second derivative of the log-likelihood:

$$\Delta n_h = \sqrt{\left(\frac{\beta^2 N_{ON}}{(n_\gamma + \beta n_h)^2} + \frac{N_{OFF}}{n_h^2}\right)^{-1}}$$

Goodness of fit estimation

The best possible fit is obtained for

$$\frac{\partial \mathcal{L}}{\partial n_\gamma} = \frac{\partial \mathcal{L}}{\partial n_h} = 0$$

This can be solved into $N_{ON} = n_\gamma + \beta n_h$ and $N_{OFF} = n_h$. One can then substract the corresponding likelihood to have a goodness-of-fit estimator:

$$\mathcal{G} \equiv \log P = N_{ON} \times \log(n_\gamma + \beta n_h) - (n_\gamma + \beta n_h) + N_{OFF} \times \log n_h - n_h + N_{ON} \times (1 - \log N_{ON}) + N_{OFF} \times (1 - \log N_{OFF})$$

In the case of a good fit, $\mathcal{G}$ should be approximatively -0.5 per degree of freedom.

Likelihood Ratio

In the case of a **signal hypothesis**, the minimum likelihood is obtained for $N_{ON} = n_\gamma + \beta n_h$ and $N_{OFF} = n_h$. The log-likelihood then reads

$$\mathcal{L} \equiv \log P = N_{ON} \times \log(N_{ON}) + N_{OFF} \times \log(N_{OFF}) - (N_{ON} + N_{OFF})$$

In the **null hypothesis** (i.e. $n_\gamma = 0$, the log-likelihood is minimised for $n_h = (N_{ON} + N_{OFF})/(1+\beta)$ and then reads

$$\mathcal{L}^0 = (N_{ON} + N_{OFF}) \times \log(N_{ON} + N_{OFF}) - (N_{ON} + N_{OFF}) \times \log(1+\beta) + N_{ON} \log \beta - (N_{ON} + N_{OFF})$$

The likelihood ratio between the two hypothesis is defined by

$$\ln \lambda = \mathcal{L}^0 - \mathcal{L}$$

Which results into:

$$\lambda = \left[\frac{\beta}{1+\beta} \frac{N_{ON} + N_{OFF}}{N_{ON}}\right]^{N_{ON}} \times \left[\frac{1}{1+\beta} \frac{N_{ON} + N_{OFF}}{N_{OFF}}\right]^{N_{OFF}}$$

According to Li&Ma statistics (ApJ **272**, 317-324, 1983), the significance of an excess $N_{ON} - \beta \times N_{OFF}$ is given by the formula

$$S = \sqrt{-2 \ln \lambda}$$

See also:
Utilities::Statistics::LiMa

Negative Signal

In the case of a negative signal we exchange the ON and OFF data and assume an signal in the OFF. This is done by applying the following transformations:

- $\mathrm{nON} \leftrightarrow \mathrm{nOFF}$
- $n_\gamma \leftrightarrow -n_\gamma/\beta$

- $\beta \leftrightarrow 1/\beta$

Author:
Mathieu de Naurois

Definition at line 49 of file OnOffFitter.hh.

List of all members.

Public Member Functions

	ClassDef (OnOffFitter, 0)

Static Public Member Functions

static double	**GetBackground** (unsigned nON, unsigned nOFF, double Beta, double nTheoSig, double *dnBG=0) Computes the optimal number of background events in a bin.
static double	**GetBackground** (unsigned nON, unsigned nOFF, double Beta, double nTheoSig, double &dBG_dnSig, double &d2BG_dnSig2) Computes the optimal number of background events in a bin.
static double	**GetLogLikelihood** (unsigned nON, unsigned nOFF, double Beta, double nTheoSig) Computes the positive log-Likelihood for one bin.
static double	**GetLogLikelihood** (unsigned nON, unsigned nOFF, double Beta, double nTheoSig, double &dLdSig, double &d2L_dSig2) Computes the positive log-Likelihood for one bin and its derivatives against theoric signal.
static double	**GetLogLikelihoodBck** (unsigned nON, double nTheoSig, double &dLdSig, double &d2L_dSig2) Computes the positive log-Likelihood for one bin and its derivatives against theoric signal.
static Double_t	**LevenbergMarquardt** (const **MathUtils::OnOffFunctorBase** &func, TVectorD &SpectrumParam, const TVectorD &Min, const TVectorD &Max, TMatrixD &ErrorMatrix, bool Verbose=true, Float_t Precision=1e-4) **Levenberg & Marquardt minimization**
static Double_t	**LevenbergMarquardt** (const **MathUtils::OnOffFunctorBase** &func, TVectorD &SpectrumParam, const TVectorD &Min, const TVectorD &Max, TMatrixD &ErrorMatrix, const TVectorD &Scaling, bool Verbose=true, Float_t Precision=1e-4) **Levenberg & Marquardt minimization**
static Double_t	**ScanErrorUp** (const **MathUtils::OnOffFunctorBase** &func, TVectorD &values, UInt_t index, Double_t starterror, Double_t Offset=1) Scan parameter space to estimate uncertainties on a parameter, until the likelihood varies by a given offset.
static Double_t	**ScanErrorDown** (const **MathUtils::OnOffFunctorBase** &func, TVectorD &values, UInt_t index, Double_t starterror, Double_t Offset=1) Scan parameter space to estimate uncertainties on a parameter, until the likelihood varies by 0.5.

Member Function Documentation

```
double MathUtils::OnOffFitter::GetBackground ( unsigned  nON,
                                               unsigned  nOFF,
                                               double    Beta,
                                               double    nTheoSig,
                                               double *  dnBG = 0
                                             )           [static]
```

Computes the optimal number of background events in a bin.

This is the value which maximizes the likelihood, nTheoSig remaining constant.

Assuming we have two independant **ON** and **OFF** observations with livetime normalisation $\beta = T_{ON}/T_{OFF}$, the probablity of observing N_{ON} and N_{OFF} events when we expect n_γ gamma events and n_h hadrons is given by the formula

$$P(N_{ON}, N_{OFF}|n_\gamma, n_h) = \frac{n_\gamma + \beta n_h{}^{N_{ON}}}{N_{ON}!} e^{-(n_\gamma+\beta n_h)} \times \frac{n_h{}^{N_{OFF}}}{N_{OFF}!} e^{-n_h}$$

The best background estimate is given by the value of n_h which maximizes the log-likelihood $\log L = log(P)$

It's the solution of a second order linear equation in n_h :

$$\frac{\beta N_{ON}}{n_\gamma + \beta n_h} + \frac{N_{OFF}}{n_h} - (\beta + 1) = 0$$

whose solution is given by:

$$\begin{aligned} C &= \beta \times (N_{ON} + N_{OFF}) - (1 + \beta) \times n_\gamma \\ \Delta^2 &= C^2 + 4\beta(\beta + 1) \times N_{OFF} \times n_\gamma \\ n_h &= \frac{C + \Delta}{2\beta(\beta + 1)} \end{aligned}$$

The uncertainties on n_h can be calculated from the second derivative of the log-likelihood:

$$\Delta n_h = \sqrt{\left(\frac{\beta^2 N_{ON}}{(n_\gamma + \beta n_h)^2} + \frac{N_{OFF}}{n_h^2} \right)^{-1}}$$

Parameters:

nON	Number of ON events
nOFF	Number of OFF events
Beta	Livetime normalisation between ON and OFF
nTheoSig	Theoric number of signal events in bin
dnBG	if non zero, will contain the uncertainties on the background

Definition at line 193 of file OnOffFitter.C.

double MathUtils::OnOffFitter::GetBackground (unsigned nON,
unsigned nOFF,
double Beta,
double nTheoSig,
double & dBG_dnSig,
double & d2BG_dnSig2
) [static]

Computes the optimal number of background events in a bin.

This is the value which maximizes the likelihood, nTheoSig remaining constant. It's the solution of a second order equation. Computes also the derivative of this number against the theoric number of signal events

Parameters:

nON	Number of ON events
nOFF	Number of OFF events
Beta	Livetime normalisation between ON and OFF
nTheoSig	Theoric number of signal events in bin
dBG_dnSig	returned derivative
d2BG_dnSig2	returned second derivative

Definition at line 242 of file OnOffFitter.C.

double MathUtils::OnOffFitter::GetLogLikelihood (unsigned nON,
unsigned nOFF,
double Beta,
double nTheoSig,

double & dLdSig,
double & d2L_dSig2
) [static]

Computes the positive log-Likelihood for one bin and its derivatives against theoric signal.

Parameters:

	nON	Number of ON events
	nOFF	Number of OFF events
	Beta	Livetime normalisation between ON and OFF
	nTheoSig	Theoric number of signal events in bin
[out]	**dLdSig**	Likelihood derivative against excess
[out]	**d2L_dSig2**	Likelihood second derivative against excess

Definition at line 339 of file OnOffFitter.C.

double MathUtils::OnOffFitter::GetLogLikelihood (unsigned nON,
unsigned nOFF,
double Beta,
double nTheoSig
) [static]

Computes the positive log-Likelihood for one bin.

The likelihood is minimized agains the number of background events **B** using MathUtils::OnOffFitter::GetBackground, and the likelihood for optimal number of events is substracted.

The likelihood reads:

$$\mathcal{L} = [N_{ON} \times \log(n_\gamma + \beta n_h) - (n_\gamma + \beta n_h) + N_{OFF} \times \log n_h - n_h] - [N_{ON} \times \log(N_{ON}) + N_{OFF} \times \log(N_{OFF}) - (N_{ON} + N_{OFF})]$$

Parameters:

nON	Number of ON events
nOFF	Number of OFF events
Beta	Livetime normalisation between ON and OFF
nTheoSig	Theoric number of signal events in bin

Definition at line 297 of file OnOffFitter.C.

double MathUtils::OnOffFitter::GetLogLikelihoodBck (unsigned nON,
double nTheoSig,
double & dLdSig,
double & d2L_dSig2
) [static]

Computes the positive log-Likelihood for one bin and its derivatives against theoric signal.

Parameters:

	nON	Number of ON events
	nTheoSig	Theoric number of signal events in bin
[out]	**dLdSig**	Likelihood derivative against excess
[out]	**d2L_dSig2**	Likelihood second derivative against excess

Definition at line 437 of file OnOffFitter.C.

Double_t MathUtils::OnOffFitter::LevenbergMarquardt	(	**const MathUtils::OnOffFunctorBase &**	**func,**
		TVectorD &	**Param,**
		const TVectorD &	**Min,**
		const TVectorD &	**Max,**
		TMatrixD **&**	**ErrorMatrix,**
		bool	**Verbose = `true`,**
		Float_t	**Precision = `1e-4`**
	)		**`[static]`**

Levenberg & Marquardt minimization

Parameters:

	func	LogLikelihood function to be used.
`[in,out]`	**Param**	(initial and returned) fit parameters
	Min	minimum value of parameter
	Max	minimum value of parameter
`[out]`	**ErrorMatrix**	returned covariance matrix multiplied by 2 (− 2 log L is equivalent to a chisquare, not log L)
	Verbose	verbosity level
	Precision	convergence criteria

To fix one parameter, just use Min = Max

Returns:

LogLikelihood at minimum

Definition at line 462 of file OnOffFitter.C.

References MathUtils::LinearAlgebra::GaussJordan().

Double_t MathUtils::OnOffFitter::LevenbergMarquardt	(	**const MathUtils::OnOffFunctorBase &**	**func,**
		TVectorD &	**Param,**
		const TVectorD &	**Min,**
		const TVectorD &	**Max,**
		TMatrixD **&**	**ErrorMatrix,**
		const TVectorD &	**Scaling,**
		bool	**Verbose = `true`,**
		Float_t	**Precision = `1e-4`**
	)		**`[static]`**

Levenberg & Marquardt minimization

Parameters:

	func	LogLikelihood function to be used.
`[in,out]`	**Param**	(initial and returned) fit parameters
	Min	minimum value of parameter
	Max	minimum value of parameter
`[out]`	**ErrorMatrix**	returned covariance matrix multiplied by 2 (− 2 log L is equivalent to a chisquare, not log L)
	Scaling	Scaling factor for each parameter, to have a convariance matrix with number close to 1
	Verbose	verbosity level
	Precision	convergence criteria

To fix one parameter, just use Min = Max

Returns:

LogLikelihood at minimum

Definition at line 644 of file OnOffFitter.C.

References MathUtils::LinearAlgebra::GaussJordan().

Double_t MathUtils::OnOffFitter::ScanErrorDown (const MathUtils::OnOffFunctorBase & func,
TVectorD & values,
UInt_t index,
Double_t starterror,
Double_t Offset = 1
) [static]

Scan parameter space to estimate uncertainties on a parameter, until the likelihood varies by 0.5.

Parameters:

func	LogLikelihood function to be used.
values	Best-fit value of parameters
index	Index of variable to be scaned in parameter array
starterror	Initial best guess of uncertainty (to be estimated from Corvariance matrix)
Offset	Offset on $2 \times \mathcal{L}$. Should be 1 for 1_σ uncertainty

Returns:
lower uncertainty

Definition at line 946 of file OnOffFitter.C.

Double_t MathUtils::OnOffFitter::ScanErrorUp (const MathUtils::OnOffFunctorBase & func,
TVectorD & values,
UInt_t index,
Double_t starterror,
Double_t Offset = 1
) [static]

Scan parameter space to estimate uncertainties on a parameter, until the likelihood varies by a given offset.

Parameters:

func	LogLikelihood function to be used.
values	Best-fit value of parameters
index	Index of variable to be scaned in parameter array
starterror	Initial best guess of uncertainty (to be estimated from Corvariance matrix)
Offset	Offset on $2 \times \mathcal{L}$. Should be 1 for 1_σ uncertainty

Returns:
upper uncertainty

Definition at line 894 of file OnOffFitter.C.

The documentation for this class was generated from the following files:

- OnOffFitter.hh
- OnOffFitter.C

Generated by doxygen 1.7.2