IDL Library for Least-Squares Minimization Genetic Algorithm Fitting

API Documentation for MGFIT-idl

Contents

Part I

Overview

Overview

MGFIT-idl is an Interactive Data Language (IDL)/GNU Data Language (GDL) Library developed to fit multiple Gaussian functions to a list of emission (or absorption) lines using a least-squares minimization technique and a random walk method in three-dimensional locations of the specified lines, namely line peak, line width, and wavelength shift. It uses the MPFIT IDL Library (MINPACK-1 Least Squares Fitting; Markwardt 2009), which performs Levenberg-Marquardt least-squares minimization, to estimate the seed values required for initializing the three-dimensional coordination of each line in the first iteration. It then uses a random walk method optimized using a genetic algorithm originally evolved from the early version of the Fortran program ALFA (Automated Line Fitting Algorithm; Wesson 2016) to determine the best fitting values of the specified lines. The continuum curve is determined and subtracted before the line identification and flux measurements. It quantifies the white noise of the spectrum, which is then utilized to estimate uncertainties of fitted lines using the signal-dependent noise model of least-squares Gaussian fitting (Lenz & Ayres 1992) built on the work of Landman, Roussel-Dupre, and Tanigawa (1982).

Dependencies
* This package requires the following packages:
- The IDL Astronomy User's Library
* To get this package with all the dependent packages, you can simply use git command as follows:

```
git clone --recursive https://github.com/mgfit/MGFIT-idl.git
```

GDL Installation
* The GNU Data Language (GDL) can be installed on
- Linux (Fedora):

```
sudo dnf install gdl
```

- Linux (Ubuntu):

```
sudo apt-get install gnudatalanguage
```

- OS X (brew):

```
brew tap brewsci/science
brew install gnudatalanguage
```

- OS X (macports):

```
sudo port selfupdate
sudo port upgrade libtool
sudo port install gnudatalanguage
```

- Windows: using the GNU Data Language for Win32 (Unofficial Version) or compiling the GitHub source with Visual Studio 2015 as seen in appveyor.yml.

* To setup MGFIT-idl in GDL, add its path to .gdl_startup in the home directory:

```
  !PATH=!PATH + ':/home/MGFIT-idl/pro/'
```

```
Set ''GDL_STARTUP'' in ''.bashrc'' (bash):
```

```
export GDL_STARTUP=~/.gdl_startup
```

or in .tcshrc (cshrc):

```
setenv GDL_STARTUP ~/.gdl_startup
```

* This package needs GDL version 0.9.8 or later.

IDL Installation

* To install MGFIT-idl in IDL, add its path to your IDL path. For more information about the path management in IDL, read the IDL path management by Harris Geospatial or the IDL library installation by David Fanning.

* This package needs IDL version 7.1 or later.

Project statistics

Directories:	1
.pro files:	17
.sav files:	0
Routines:	17
Lines:	569

Part II

API

Directory: ./

Overview

mgfit_contin.pro

MGFIT_CONTIN

This function extracts the continuum from the spectrum.

```
result = mgfit_contin(spectrumdata)
```

Returns

type=arrays of structures. This function returns the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}.

Parameters

spectrumdata IN REQUIRED TYPE=arrays of structures
the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

Examples

For example:

```
IDL> spectrumdata=mgfit_init_spec(wavel, flux)
 IDL> continuum=mgfit_contin(spectrumdata)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Translated to IDL from FORTRAN in ALFA by R. Wessson.

15/01/2017, A. Danehkar, A few bugs fixed

Version

0.1.0

mgfit_emis.pro

MGFIT_EMIS

This function fits multiple Gaussian functions to a list of emission lines using a least-squares minimization technique and a genetic-type random walk method. It uses the MPFIT idl library to initialize the parameters of the run in the first iteration. The continuum curve is determined using mgfit_contin() and subtracted before the line identification and flux measurements. It uses mgfit_emis_err() to estimate the uncertainties itroduced by the best-fit model residuals and the white noise.

```
result = mgfit_emis(specdata, redshift_initial, resolution_initial, emissionlines, redshift_tolerance1
   , resolution_tolerance1, resolution_min, resolution_max, generations, popsize, pressure
   , line_array_size=float, no_blueshift=no_blueshift, /printimage, imagename=string)
```

Returns

type=arrays of structures. This function returns the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:”, Multiplet:”, LowerTerm:”, UpperTerm:”, g1:”, g2:”}

Parameters

specdata IN REQUIRED TYPE=arrays of structures

the observed spectrum stored in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

redshift_initial IN REQUIRED TYPE=float

the initial/guess redshift

resolution_initial

emissionlines IN REQUIRED TYPE=arrays of structures

the specified emission lines stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, continuum:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:”, Multiplet:”, LowerTerm:”, UpperTerm:”, g1:”, g2:”}

redshift_tolerance1

resolution_tolerance1

resolution_min

resolution_max

generations IN REQUIRED TYPE=float

the maximum generation number in the genetic algorithm

popsize IN REQUIRED TYPE=float

the population size in each generation in the genetic algorithm

pressure IN REQUIRED TYPE=float

the value of the selective pressure in the genetic algorithm

Keywords

line_array_size IN REQUIRED TYPE=float

size of the line array

no_blueshift

printimage IN REQUIRED TYPE=boolean

Set to produce plots

imagename IN REQUIRED TYPE=string

The file name for plots if printimage sets

Examples

For example:

```
IDL> fitstronglines = mgfit_emis(stronglines, redshift_initial, resolution_initial, $
 IDL>                            emissionlines, redshift_tolerance1, resolution_tolerance1, $
 IDL>                            generations, popsize, pressure, line_array_size=linelocation0_step)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Adopted from Algorithm used in the FORTRAN program ALFA by R. Wessson

22/07/2015, A. Danehkar, Several performance optimized.

12/11/2015, A. Danehkar, Degree and variance added to chi_squared.

15/02/2016, A. Danehkar, Continuum subtracted before fitting.

22/02/2016, A. Danehkar, Uncertainties estimation added.

15/10/2016, A. Danehkar, Fixed small bugs.

22/11/2017, A. Danehkar, New parameters added, other modifications.

Version

0.1.0

mgfit_emis_err.pro

MGFIT_EMIS_ERR

This function estimates the uncertainties introduced by the best-fit model residuals and the white noise quantified using the signal-dependent noise model of least-squares Gaussian fitting (Lenz & Ayres 1992; 1992PASP..104.1104L) based on on the work of Landman, Roussel-Dupre, and Tanigawa (1982; 1982ApJ...261..732L).

```
result = mgfit_emis_err(syntheticspec, spectrumdata, emissionlines, redshift)
```

Returns

type=arrays of structures. This function returns the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:", Multiplet:", LowerTerm:", UpperTerm:", g1:", g2:"}

Parameters

syntheticspec IN REQUIRED TYPE=arrays of structures

the synthetic spectrum made by mgfit_synth_spec() stored in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

spectrumdata in required type=arrays of structures

the observed spectrum stored in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

emissionlines in required type=arrays of structures

the emission lines specified for error estimation stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:", Multiplet:", LowerTerm:", UpperTerm:", g1:", g2:"}

redshift

Examples

For example:

```
IDL> emissionlines_section=mgfit_emis_err(syntheticspec_section, spec_section, emissionlines_section)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Adopted from Algorithm used in the FORTRAN program ALFA by R. Wessson

12/04/2015, A. Danehkar, Performance optimized for IDL.

20/08/2016, A. Danehkar, Added better performance in noise estimation.

22/10/2016, A. Danehkar, Fixed small bugs.

22/02/2016, A. Danehkar, Uncertainties estimation added.

15/10/2016, A. Danehkar, Fixed small bugs.

21/06/2017, A. Danehkar, Some modifications.

Version

0.1.0

mgfit_fltr_emis.pro

MGFIT_FLTR_EMIS

This function filters the emission line lists from the list of emission lines within the specified wavelength range.

```
result = mgfit_fltr_emis(emissionlines, wavel_min, wavel_max)
```

Returns

type=arrays of structures. This function returns the lits of selected emission lines in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:'', Multiplet:'', LowerTerm:'', UpperTerm:'', g1:'', g2:''}

Parameters

emissionlines IN REQUIRED TYPE=arrays of structures
the emission lines given for the selection stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:'', Multiplet:'', LowerTerm:'', UpperTerm:'', g1:'', g2:''}

wavel_min IN REQUIRED TYPE=float
the minimum wavelength

wavel_max IN REQUIRED TYPE=float
the maximum wavelength

Examples

For example:

```
IDL>  emissionlines_section=mgfit_fltr_emis(emissionlines, wavel_min, wavel_max)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0

mgfit_init_emis.pro

MGFIT_INIT_EMIS

This function initializes the emission line list with the specified wavelength array and flux array.

```
result = mgfit_init_emis(wavel, flux)
```

Returns

type=arrays of structures. This function returns the emission line list stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:”, Multiplet:”, LowerTerm:”, UpperTerm:”, g1:”, g2:”}

Parameters

wavel IN REQUIRED TYPE=arrays
the wavelength array

flux IN REQUIRED TYPE=arrays
the flux array

Examples

For example:

```
IDL> emissionlines=mgfit_init_emis(wavel, flux)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0

mgfit_init_fltr_emis.pro

MGFIT_INIT_FLTR_EMIS

This function initializes and filters the emission line lists from the list of emission lines within the specified wavelength range

```
result = mgfit_init_fltr_emis(emissionlines, wavel_min, wavel_max, redshift)
```

Returns

type=arrays of structures. This function returns the lits of selected emission lines in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, continuum:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:'', Multiplet:'', LowerTerm:'', UpperTerm:'', g1:'', g2:''}

Parameters

emissionlines IN REQUIRED TYPE=arrays of structures

the emission lines given for the selection stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, continuum:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:'', Multiplet:'', LowerTerm:'', UpperTerm:'', g1:'', g2:''}

wavel_min IN REQUIRED TYPE=float

the minimum wavelength

wavel_max IN REQUIRED TYPE=float

the maximum wavelength

redshift

Examples

For example:

```
IDL> emissionlines=mgfit_init_fltr_emis(strongline_data, wavel_min, wavel_max)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

15/01/2017, A. Danehkar, A few bugs fixed

Version

0.1.0

mgfit_init_seed.pro

MGFIT_INIT_SEED

This function initializes the random seed based on the system clock

```
result = mgfit_init_seed()
```

Returns

type=arrays. This function returns 20 random numbers.

Examples

For example:

```
IDL> ret=mgfit_init_seed()
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Translated to IDL from FORTRAN in ALFA by R. Wessson

Version

0.1.0

mgfit_init_spec.pro

MGFIT_INIT_SPEC

This function creates the spectrum from the wavelength array and flux array.

```
  result = mgfit_init_spec(wavel, flux)
```

Returns

type=arrays of structures. This function returns the spectrum in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

Parameters

wavel IN REQUIRED TYPE=arrays
the wavelength array

flux IN REQUIRED TYPE=arrays
the flux array

Examples

For example:

```
IDL> spectrumdata=mgfit_init_spec(wavel, flux)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0

mgfit_mutation1.pro

MGFIT_MUTATION1

This function is for the genetic algorithm mutation type-1

```
  result = mgfit_mutation1()
```

Returns

type=arrays. This function mutation rate.

Examples

For example:

```
IDL> value=mgfit_mutation1()
```

Author

Ashkbiz Danehkar

Copyright

History

20/07/2014, A. Danehkar, Translated to IDL from FORTRAN in ALFA by R. Wessson

Version

0.1.0

mgfit_synth_spec.pro

MGFIT_SYNTH_SPEC

This function makes a spectrum from given lines.

```
  result = mgfit_synth_spec(lines, spec, continuum=continuum)
```

Returns

type=arrays of structures. This function returns the spectrum in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

Parameters

lines IN REQUIRED TYPE=arrays of structures

the line list stored in the arrays of structures { wavelength: 0.0, peak:0.0, sigma1:0.0, flux:0.0, continuum:0.0, uncertainty:0.0, redshift:0.0, resolution:0.0, blended:0, Ion:", Multiplet:", LowerTerm:", UpperTerm:", g1:", g2:"}

spec

Keywords

continuum

Examples

For example:

```
IDL> syntheticspec=mgfit_synth_spec(emissionlines, syntheticspec)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Translated to IDL from FORTRAN in ALFA by R. Wessson.

22/11/2017, A. Danehkar, A few changes.

Version

0.1.0

minloc_idl.pro

MINLOC_IDL

This function determines the location of the element in the array with the minimum value

```
  result = minloc_idl(inarr, first=first, last=last)
```

Returns

type=integer. Location of the minimum value within an array: the location of the first value if first=1, the location of last value if last=1.

Parameters

inarr IN REQUIRED TYPE=arrays
an array of type INTEGER or REAL.

Keywords

first

last

Examples

For example:

```
IDL> chi_squared = [5, 7, 1, 3, 6, 1]
 IDL>  chi_squared_min_loc=minloc_idl(chi_squared,first=1)
 IDL> print, chi_squared_min_loc
```

mpfit_whitenoise.pro

MPFIT_WHITENOISE

This function extracts the white noise from the spectrum.

```
result = mpfit_whitenoise(spectrumdata)
```

Returns

type=arrays of structures. This function returns the white noise in the arrays of structures {wavelength: 0.0, flux:0.0, residual:0.0}

Parameters

spectrumdata IN REQUIRED TYPE=arrays of structures
the input spectrum stored in the arrays of structures { wavelength: 0.0, flux:0.0, residual:0.0}

Examples

For example:

```
IDL> specdata=mpfit_whitenoise(specdata)
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, Translated to IDL from FORTRAN in ALFA by R. Wessson

21/11/2017, A. Danehkar, Some modifications.

Version

0.1.0

nint_idl.pro

NINT_IDL

NAME: NINT PURPOSE: Nearest integer function. EXPLANATION: NINT() is similar to the intrinsic ROUND function, with the following two differences: (1) if no absolute value exceeds 32767, then the array is returned as as a type INTEGER instead of LONG (2) NINT will work on strings, e.g. print,nint(['3.4','-0.9']) will give [3,-1], whereas ROUND() gives an error message

CALLING SEQUENCE: result = nint(x, [/LONG])

INPUT: X - An IDL variable, scalar or vector, usually floating or double Unless the LONG keyword is set, X must be between -32767.5 and 32767.5 to avoid integer overflow

OUTPUT RESULT - Nearest integer to X

OPTIONAL KEYWORD INPUT: LONG - If this keyword is set and non-zero, then the result of NINT is of type LONG. Otherwise, the result is of type LONG if any absolute values exceed 32767, and type INTEGER if all all absolute values are less than 32767. EXAMPLE: If X = [-0.9,-0.1,0.1,0.9] then NINT(X) = [-1,0,0,1]

PROCEDURE CALL: None: REVISION HISTORY: Written W. Landsman January 1989 Added LONG keyword November

1991 Use ROUND if since V3.1.0 June 1993 Always start with ROUND function April 1995 Return LONG values, if some input value exceed 32767 and accept string values February 1998 Use size(/TNAME) instead of DATATYPE() October 2001

```
  result = nint_idl(x, LONG=LONG)
```

Parameters

x

Keywords

LONG

read_deeplines.pro

READ_DEEPLINES

This function reads the list of deep lines from the 3rd binary table extension of the FITS data file (../data/linedata.fits). This function uses the routine ftab_ext from IDL Astronomy User's library.

```
  result = read_deeplines(fits_file, EXTEN_NO=EXTEN_NO)
```

Returns

type=arrays of structures. This function returns the deep line list in the arrays of structures { Wavelength:0.0, Ion:", Multiplet:", LowerTerm:", UpperTerm:", g1:", g2:"}

Parameters

fits_file IN REQUIRED TYPE=string
the FITS file name ("../data/linedata.fits")

Keywords

EXTEN_NO

Examples

For example:

```
IDL> deepline_data = read_deeplines(fits_file)
 IDL> print, deepline_data.Wavelength, deepline_data.Ion
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

16/06/2017, A. Danehkar, A few changes.

Version

0.1.0

read_skylines.pro

READ_SKYLINES

This function reads the list of sky lines from the 3rd binary table extension of the FITS data file (../data/linedata.fits). This function uses the routine ftab_ext from IDL Astronomy User's library.

```
  result = read_skylines(fits_file)
```

Returns

type=arrays of structures. This function returns the sky line list in the arrays of structures { Wavelength:0.0}

Parameters

fits_file IN REQUIRED TYPE=string

the FITS file name ("../data/linedata.fits")

Examples

For example:

```
IDL> skyline_data = read_skylines(fits_file)
 IDL> print, skyline_data.Wavelength
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0

read_stronglines.pro

READ_STRONGLINES

This function reads the list of strong lines from the 1rd binary table extension of the FITS data file (../data/linedata.fits). This function uses the routine ftab_ext from IDL Astronomy User's library.

```
  result = read_stronglines(fits_file)
```

Returns

type=arrays of structures. This function returns the strong line list in the arrays of structures { Wavelength:0.0, Ion:'', Multiplet:'', LowerTerm:'', UpperTerm:'', g1:'', g2:''}

Parameters

fits_file IN REQUIRED TYPE=string
the FITS file name ("../data/linedata.fits")

Examples

For example:

```
IDL> strongline_data = read_stronglines(fits_file)
 IDL> print, strongline_data.Wavelength, strongline_data.Ion
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0

read_ultradeeplines.pro

READ_ULTRADEEPLINES

This function reads the list of ultra deep lines from the 4rd binary table extension of the FITS data file (../data/linedata.fits). This function uses the routine ftab_ext from IDL Astronomy User's library.

```
result = read_ultradeeplines(fits_file)
```

Returns

type=arrays of structures. This function returns the ultra deep line list in the arrays of structures { Wavelength:0.0, Ion:", Multiplet:", LowerTerm:", UpperTerm:", g1:", g2:"}

Parameters

fits_file IN REQUIRED TYPE=string
the FITS file name ("../data/linedata.fits")

Examples

For example:

```
IDL> ultradeepline_data = read_ultradeeplines(fits_file)
 IDL> print, ultradeepline_data.Wavelength, ultradeepline_data.Ion
```

Author

Ashkbiz Danehkar

Copyright

This library is released under a GNU General Public License.

History

20/07/2014, A. Danehkar, IDL code written.

Version

0.1.0