Package 'fitConic'

January 12, 2022

Title Fit Data to Any Conic Section

Version 1.2

Date 2021-12-25

Description Fit data to an ellipse, hyperbola, or parabola. Bootstrapping is available when needed. The conic curve can be rotated through an arbitrary angle and the fit will still succeed. Helper functions are provided to convert generator coefficients from one style to another, generate test data sets, rotate conic section parameters, and so on. References include Nikolai Chernov (2014) ``Fitting ellipses, circles, and lines by least squares'' <https://people.cas.uab.edu/~mosya/cl/>; A. W. Fitzgibbon, M. Pilu, R. B. Fisher (1999) ``Direct Least Squares Fitting of Ellipses'' IEEE Trans. PAMI, Vol. 21, pages 476-48; N. Chernov, Q. Huang, and H. Ma (2014) ``Fitting quadratic curves to data points'', British Journal of Mathematics & Computer Science, 4, 33-60; N. Chernov and H. Ma (2011) ``Least squares fitting of quadratic curves and surfaces'', Computer Vision, Editor S. R. Yoshida, Nova Science Publishers, pp. 285-302.

Imports pracma

License LGPL-3

NeedsCompilation no

Author Carl Witthoft [aut, cre],
Jose Gama [ctb],
Nikolai Chernov [ctb]

Maintainer Carl Witthoft <carl@witthoft.com>

Repository CRAN

Date/Publication 2022-01-12 00:42:40 UTC

R topics documented:

`fitConic-package` *Fit Data to Any Conic Section*

Description

Fit data to an ellipse, hyperbola, or parabola. Bootstrapping is available when needed. The conic curve can be rotated through an arbitrary angle and the fit will still succeed. Helper functions are provided to convert generator coefficients from one style to another, generate test data sets, rotate conic section parameters, and so on. References include Nikolai Chernov (2014) "Fitting ellipses, circles, and lines by least squares" <https://people.cas.uab.edu/~mosya/cl/>; A. W. Fitzgibbon, M. Pilu, R. B. Fisher (1999) "Direct Least Squares Fitting of Ellipses" IEEE Trans. PAMI, Vol. 21, pages 476-48; N. Chernov, Q. Huang, and H. Ma (2014) "Fitting quadratic curves to data points", British Journal of Mathematics & Computer Science, 4, 33-60; N. Chernov and H. Ma (2011) "Least squares fitting of quadratic curves and surfaces", Computer Vision, Editor S. R. Yoshida, Nova Science Publishers, pp. 285-302.

Details

The DESCRIPTION file:

Package:	fitConic
Title:	Fit Data to Any Conic Section
Version:	1.2
Date:	2021-12-25
Authors@R:	c(person(given = "Carl", family = "Witthoft", role = c("aut","cre"), email= "carl@witthoft.com") ,person(given
Description:	Fit data to an ellipse, hyperbola, or parabola. Bootstrapping is available when needed. The conic curve can be
Imports:	pracma
License:	LGPL-3
Author:	Carl Witthoft [aut, cre], Jose Gama [ctb], Nikolai Chernov [ctb]
Maintainer:	Carl Witthoft <carl@witthoft.com>

The main function is `fitConic` .

Author(s)

NA, based on code provided in the references and in `conicfit::fit.conicLMA()`

Maintainer: NA

References

https://www.mathworks.com/matlabcentral/answers/80541 for the RANSAC-style search to fit rotated parabolas. https://math.stackexchange.com/questions/426150 for detailed ellipse parametric equations. https://math.stackexchange.com/questions/2800817 for "focus/directrix/eccentricity" information https://people.cas.uab.edu/~mosya/cl/ and the folks referred to there, for fitConicLMA . https://en.wikipedia.org/wiki/Ellipse for several parameter conversion formulas

A. W. Fitzgibbon, M. Pilu, R. B. Fisher, "Direct Least Squares Fitting of Ellipses", IEEE Trans. PAMI, Vol. 21, pages 476-480 (1999)

Halir R, Flusser J (1998) Proceedings of the 6th International Conference in Central Europe on Computer Graphics and Visualization, Numerically stable direct least squares fitting of ellipses (WSCG, Plzen, Czech Republic), pp 125132.

`AtoG` *A Set Of Functions To Convert Among Various Conic-Section-Defining Parameter Sets.*

Description

`AtoG` Convert from full quadratic "ABCDEF" to focus, axis, angle "hvab theta" parameters. `GtoA` Convert from "hvab theta" to "ABCDEF" parameters. `parab3toA` Simple conversion from a + bx + cx^2 to "ABCDEF" parameters. `FEDtoA` Convert focus, eccentricity, and directrix to "ABCDEF" parameters.

Usage

```
AtoG(parA, tol = 1e-06)
GtoA(parG, conicType = c("e", "h"))
parab3toA(ADF, theta = 0)
FEDtoA(focus = c(0, 0), directrix = c(1, 0, 1), eccentricity = 0.5)
```

Arguments

`parA`	The six coefficients in the quadratic Ax^2 + Bxy + Cy^2 +Dx + Ey +F = 0
`tol`	A small value, used to check whether small coefficient values might be actually zero. See "Details."
`parG`	a five-element vector "h,v,a,b,theta" . See "Details" for the standard equation form for this.
`conicType`	Because the 'hvab' equation has a sign difference for ellipses vs. hyperbolas, it is necessary to indicate which kind of input is intended. See "Details."
`focus`	location of the conic sections focus.
`directrix`	the 3-element directrix.
`eccentricity`	the eccentricity of the conic section.
`ADF`	The A,D,F coeffients in the standard quadratic. Thus, the x^2 term, the x term, and the constant term.
`theta`	An angle by which the entire parabola is to be rotated.

Details

The `tol` input for `AtoG` checks two conditions. First, is B practically zero, in which case B is set to exactly zero, implying no rotation of the conic section. Second, is B^2 - 4*A*C almost zero, implying that the conic is probably a parabola, and conversion to 'hvab' form is not useful.

The "hvab" form for describing an ellipse or a hyperbola looks like [Center(1:2), Axes(1:2)/2] angle A, to fill the equation

((x-h)cosA +(y-v)sinA)^2/a^2 + ((x-h)sinA-(y-v)cosA)^2/b^2 = 1 The length of the axes are 2*a, 2*b .

A discussion of the focus/directrix/eccentricity form of a conic section is rather lengthy and not presented here. One short introduction can be found at https://en.wikipedia.org/wiki/Conic_section#Eccentricity,_focus_and_directrix

Value

for `AtoG`,

`parG`	c(h,v,a,b,theta)
`exitCode`	a value used in fitConic. 1,2, or 3 for ellipse, hyperbola, parabola
`conicType`	matching exitCode with a char "e", "h", or "p"

for `GtoA`

`parA`	the ABCDEF coefficients of the general quadratic
`exitCode`	a value used in fitConic. 1,2, or 3 for ellipse, hyperbola, parabola
`conicType`	matching exitCode with a char "e", "h", or "p"

for `FEDtoA`, the ABCDEF coefficients of the general quadratic for `parab3toA`,

`parA`	the ABCDEF coefficients of the general quadratic
`exitCode`	always numeric 3, a value used in fitConic
`conicType`	always char "p"

.

`bootEllipse` *Simple, Medium-Quality Ellipse Fitting Function*

Description

This function generates a half-decent fit to the source data. It is intended only for internal use, to bootstrap the higher-quality `fitConic` function.

Usage

```
bootEllipse(x, y = NULL, ...)
```

Arguments

`x`	vector of x-values, or a Nx2 array of x and y values. In the latter case, the input `y` is ignored.
`y`	vector of y-values.
`...`	possible other arguments to be passed to future upgrades

Details

This can be used as a Q&D ellipse fitting algorithm, but is intended only for internal use by `fitConic`, providing that function with an initial estimate for the ellipse's defining parameter set.

Value

`parA`	6-element set with estimate of the "ABCDEF" coefficients for the general quadratic equation
`centroid`	estimate of the ellipse's centroid

Author(s)

Carl Witthoft, <carl@witthoft.com>

References

This is a revision of the function EllipseDirectFit in package conicfit by Jose Gama, with minor upgrades. Original MATLAB code by: Nikolai Chernov https://www.mathworks.com/matlabcentral/fileexchange/22684-ellipse-fit-direct-method A. W. Fitzgibbon, M. Pilu, R. B. Fisher, "Direct Least Squares Fitting of Ellipses", IEEE Trans. PAMI, Vol. 21, pages 476-480 (1999) Halir R, Flusser J (1998) Proceedings of the 6th International Conference in Central Europe on Computer Graphics and Visualization, Numerically stable direct least squares fitting of ellipses (WSCG, Plzen, Czech Republic), pp 125132.

See Also

`fitConic` , `createConic`

`bootHyperbola` *A Function to Attempt a Crude Fit of Data to a Hyperbola*

Description

This function is not intended for direct use. It attempts to generate an approximate fit of a data set to a hyperbola, returning a parameter set for use in intializing the main function `conicFit` .

Usage

```
bootHyperbola(x, y = NULL, maxiter = 10000, ...)
```

Arguments

`x`	vector of x-values, or a Nx2 array of x and y values. In the latter case, the input y is ignored.
`y`	vector of y-values.
`maxiter`	A 'safety' limiter on the number of iterations to try before giving up.
`...`	possible other arguments to be passed to future upgrades

Value

`parA`	the new 6-parameter set defining the non-rotated conic.
`parAr`	the new 6-parameter set defining the rotated conic.
`theta`	the angle of rotation between ParA and ParAr
`fitdat`	the information returned from `optim`

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

See Also

`fitConic`

`createConic` *Create A Conic Section Dataset Based on Parameter Set*

Description

Given a vector of x-values and a parameter set defining a conic section, produce an array of x- and y- values, optionally with noise added, for the specified conic section.

Usage

```
createConic(x, param, conicType, ranFun = NULL, noise = 1, seedit = NULL, tol = 1e-06)
```

Arguments

`x`	Vector of (real) values
`param`	Either a 6-value set representing the standard quadratic form Ax^2 + Bxy + Cy^2 +Dx + Ey +F = 0 or a 5-value set representing the "hvab,theta" form ((x-h)cosA +(y-v)sinA)^2/a^2 + ((x-h)sinA-(y-v)cosA)^2/b^2 = 1 . In the latter case the value `conicType` is required.
`conicType`	Either the character "e" for ellipse or "h" for hyperbola. Only required if the "hvab,theta" form is used in `param` .
`ranFun`	If random noise is to be added to the calculated y-values, provide a vectorized function which takes a single input (x). See Details.

`noise`	Optional argument to multiply the output of `ranFun` .
`seedit`	Optional argument to set a starting seed for `ranFun` to use.
`tol`	A (small) value used to decide whether various parameter terms are so small that they should be zero. This is used to facilitate distinguishing, e.g., parabolas from hyperbolas.

Details

When supplied `ranFun` is used as follows. y <- y + ranFun(y)*noise . Make sure any function supplied fits that form (no other input argument required; only a vector returned).

Value

An N x 2 array of the x,y pairs. Warning: since there are often two possible y-values for a given x-value (these being quadratic equations), the array does contain duplicate x-values. This may "annoy" some other packages' functions which don't allow that sort of repeated value. If this presents a problem, I'd recommend applying a very small amount of noise to the x-values in this output.

Author(s)

Carl Witthoft <carl@witthoft.com>

Examples

```
# create noisy ellipse
parGr <- c(-2.3,4.2,5,3,pi/4)
xe <-seq(-8,9,by=.05)
elipGrn <- createConic(xe, parGr, 'e',ranFun=rnorm, noise=0.25)
elipGr <- createConic(xe,  parGr, 'e')
plot(elipGrn, pch='.',cex = 4, asp = TRUE)  #, xlim = c(-5,8), ylim = c(0,7))
lines(elipGr,col='green')
```

`doWeights` *Function to Apply Weights to Data*

Description

This function applies an integer weight set to an array of (x,y) data points. It normally is only called from `fitConic` but can be applied directly to a dataset if desired.

Usage

```
doWeights(XY, weights)
```

Arguments

`XY`	A Nx2 array of data representing (x,y) pairs
`weights`	A vector of weights the same length as the number of rows in XY. At this time, only nonnegative integer values are allowed. Doubles are rounded and negative values are set to zero. A zero weight will remove the matching data value from the dataset.

Value

A new Nx2 array. Basically, each row in the input XY is repeated `weights[j]` times. ..

Author(s)

Carl Witthoft <carl@witthoft.com>

---

fhyp — *Internal Functions to Perform Bootstrap Fitting Operations*

Description

These functions are not intended for external use. `fhyp` and `fhypopt` support the parent function `bootHyperbola` by providing functions for `optimize` to use. The functions `costparab` `costparabxy` similarly provide functions for `optim` to use inside the function `fitParabola` .

Usage

```
fhyp(xy, b3, Ang)
costparabxy(theta, xy)
costparab(theta, xy)
```

Arguments

`xy`	A Nx2 array of data
`b3`	Three of the parameters describing a hyperbola. These three are the "other parameters" fed to `optim`
`Ang`	The initial angle of rotation, also optimized during the process.
`theta`	The angle of rotation of the parabola for this run of `optimize`

Value

various combinations of "cost" values, i.e. Figure of Merit, to determine optimal set of coefficients, along with datasets where necessary. ..

Author(s)

Carl Witthoft <carl@witthoft.com>

`fitConic` *Fit Data to A Conic Section Curve*

Description

This function fits data to an ellipse, hyperbola, or parabola. It can do so without any initial conditions, or can accept initial parameter values when known.

Usage

```
fitConic(X, Y = NULL, parInit = NULL, conicType = c("e", "h", "p"),
weights = NULL, LambdaIni = 1, epsilonP = 1e-06, epsilonF = 1e-06, IterMAX = 20000)
```

Arguments

`X`	vector of x-values, or a Nx2 array of x and y values. In the latter case, the input y is ignored.
`Y`	vector of y-values.
`parInit`	Optional. A vector either of six values representing an initial guess at the "ABCDEF" coefficients of the quadratic, or five values representing an initial guess at the "hvab,theta" coefficients. In the latter case, a value of either "e" or "h" is required for `conicType`. See the Details section for more information.
`conicType`	If `parInit` is either NULL or the "hvab,theta" option, `conicType` is required. Enter either "e", "h", or "p" for fitting to ellipse, hyperbola, or parabola.
`weights`	Optional vector of weights to apply to data. Must be same length as the input data. Only non-negative integer weights are allowed. See the Details section.
`LambdaIni`	A control parameter used in the fitting algorithm. Typically there is no reason to change from the default value.
`epsilonP`	A tolerance value to determine whether convergence has occurred.
`epsilonF`	A tolerance parameter for determining when to adjust lambda away from the input value `LambdaIni`.
`IterMAX`	A "safety" value to avoid loop thrashing when convergence isn't taking place.

Details

`ParInit`, when supplied is either a 6-value set representing the standard quadratic form Ax^2 + Bxy + Cy^2 +Dx + Ey +F = 0 or a 5-value set representing the "hvab,theta" form ((x-h)cosA +(y-v)sinA)^2/a^2 + ((x-h)sinA-(y-v)cosA)^2/b^2 = 1 . In the latter case the value `conicType` is required, because ellipses and hyperbolas have a different sign for the y-term. In most cases, the bootstrapper tools work well enough to allow the main algorithm to fit to an ellipse or hyperbola. However, "knowledge is power." If you have a good idea approximately what the `ParIni` values are, entering them will help avoid convergence to the wrong local minimum. The algorithm branch which fits data to parabolas does not use or need initialization, as it uses a RANSAC-type search to find the best rotation angle, and then does a simple quadratic polynomial fit. The `weights` input is restricted to nonnegative integers at this time. Doubles are rounded and negative values are set to zero. A zero weight will remove the matching data value from the dataset.

Value

parA vector of the six "ABCDEF" coefficients RSS 'root sum square' figure of merit describing the relative fit quality iters number of iterations at convergence exitCode 1 means ellipse, 2 means hyperbola, 3 means parabola. If other values show up (possibly -1, 0, 4), most likely the dataset led to a degenerate case such as a line fit.

Author(s)

Carl Witthoft <carl@witthoft.com>

References

https://people.cas.uab.edu/~mosya/cl/ for information on the original "LMA" fitting algorithm. https://math.stackexchange.com/questions/426150 and https://math.stackexchange.com/questions/2800817 for various related equations concerning conic sections. https://en.wikipedia.org/wiki/Ellipse for several parameter conversion formulas

See Also

createConic , fitParabola

Examples

```
##-create a hyperbola, add noise
Ang =  0.42 #radians
xh <- seq(-20,20,by=0.1)
parAxyh <- c(0, 1, 0, -2, 4, -15 )
parAxyhr <- rotateA(parAxyh, Ang)$parA
newxyr <-createConic(xh,parAxyhr)
newxyrn <- createConic(xh,parAxyhr,ranFun=rnorm, noise= 0.05)
plot(newxyr, t = 'l',asp=TRUE)
points(newxyrn, pch = '.', cex = 3)
# Now find the hyperbola for that dataset
hypfitr <-fitConic(newxyrn, conicType = 'h')
hypdatr <- createConic(xh, hypfitr$parA)
lines(hypdatr, col='red')
```

---

fitParabola *Fit Data to Parabola*

Description

This function fits a data set to a parabola, including any rotation angle.

Usage

```
fitParabola(x, y = NULL, searchAngle = c(-pi/2, pi/2), ...)
```

Arguments

`x`	vector of x-values, or a Nx2 array of x and y values. In the latter case, the input `y` is ignored.
`y`	vector of y-values.
`searchAngle`	Optional pair of angles, in radians, defining the limits of the search range to find the rotation angle of the parabola. Usually the default range `-pi/2:+pi/2` works acceptable.
`...`	For possible future expansion to pass to additional features.

Details

`fitParabola` starts by doing a RANSAC-style search to find the optimum rotation angle. Once that is chosen, the data are rotated by that angle and a simple polynomial fit to the (rotated) vertical parabola is done.

Value

`vertex`	calculated vertex of the parabola
`theta`	angle of rotation relative to a vertical parabola
`parA`	the "ABCDEF" coefficients of the fitted parabola
`parQ`	the coefficients of the derotated parabola's simple quadratic polynomial, highest power first
`cost`	final value of the "cost" parameter used for optimization

Note

When the function `fitConic` is called with instructions to fit to a parabola, it passes the inputs to `fitParabola` and does nothing else. For parabolic data, then, either function will give the same result.

Author(s)

Carl Witthoft <carl@witthoft.com>

References

Some of the code is based on https://www.mathworks.com/matlabcentral/answers/80541

See Also

`createConic`

Examples

```
# Create vertical parabola with some noise
parP <-c(.5,0,0,2,-1,4)
xp <- seq(-5,5,by=0.05)
 partest <-createConic(xp,param = parP,ranFun = rnorm, noise = 1)
 plot(partest, pch= '.',asp=TRUE, cex=3)
 # rotate the data
 partestr <-xyrot(partest,theta = -.35)
 points(partestr,col='green',pch='.',cex=3)
 # do the fit
 parfit <-fitParabola(partestr)
 points(parfit$vertex,pch='X',col='blue')
 parout <- createConic(xp,parfit$parA)
 lines(parout,col='red')
```

JmatrixLMA — *Calculate a Jacobian Matrix*

Description

Calculate the Jacobian matrix with the original dataset and the current version of fitted data. This is not intended for external use. It is called from `fitConic`

Usage

```
JmatrixLMA(XY, parA, XYproj)
```

Arguments

`XY`	The original input dataset
`parA`	The current set of ABCDEF quadratic equation coefficients.
`XYproj`	The current calculated dataset based on the latest iteration of the coefficient set.

Value

`Res`	residuals based on the norm of XY - XYproj
`J`	matrix of values for each input data point corresponding to the terms in the general quadratic Ax^2 + Bxy + Cy^2 +Dx + Ey +F

Author(s)

Carl Witthoft <carl@witthoft.com>

References

This is a copy of `JmatrixLMA` with some validation steps added.

Residuals.ellipse *Calculate Residual Error For Current Coefficients*

Description

This function is not intended for external use. It is called from `fitConic` when iterating to find the best-fit ellipse.

Usage

```
Residuals.ellipse(XY, parG)
```

Arguments

`XY`	The x,y dataset
`parG`	The "G-parameter" set for the current iteration.

Value

`RSS`	Figure of merit, the 'norm' of the difference between the input XY data and the output "XYproj" data generated.
`XYproj`	Calculated dataset to be used in generating the Jacobian matrix for the next iteration of `fitConic`

Author(s)

Carl Witthoft <carl@witthoft.com>

References

This is a slightly modified (and debugged) version of `Residuals.ellipse`

Residuals.hyperbola *Calculate Residual Error For Current Coefficients*

Description

This function is not intended for external use. It is called from `fitConic` when iterating to find the best-fit hyperbola.

Usage

```
Residuals.hyperbola(XY, parG)
```

Arguments

`XY`	The x,y dataset
`parG`	The "G-parameter" set for the current iteration.

Value

`RSS`	Figure of merit, the 'norm' of the difference between the input XY data and the output "XYproj" data generated.
`XYproj`	Calculated dataset to be used in generating the Jacobian matrix for the next iteration of `fitConic`

Author(s)

Carl Witthoft <carl@witthoft.com>

References

This is a slightly modified (and debugged) version of `Residuals.hyperbola`

---

rotateA — *Rotate Conic Section Equation Parameters Or A Dataset, With Respect To X-Y Axes.*

Description

`rotateA` Takes as input "parA," the 6 values of the general quadratic Ax^2 + Bxy + Cy^2 +Dx + Ey +F = 0 , and applies a rotation angle to the coefficient set. `derotateA` calculates the rotation angle required to change the conic section defined by 'parA' into one that is orthogonal to the cartesian axes. `xyrot` is a simple function to rotate the coordinate system by theta.

Usage

```
rotateA(parA, theta)
derotateA(parA, ACmin = 1e-05)
xyrot(x, y = NULL, theta)
```

Arguments

`parA`	the 6 values of the general quadratic Ax^2 + Bxy + Cy^2 +Dx + Ey +F = 0
`theta`	the angle, in radians, to rotate the conic section.
`ACmin`	A tolerance parameter for deciding that the product of parameters A and C is actually zero (in which case the type of conic section is more likely a parabola or a degenerate case)
`x`	Either a vector of x-coordinates or a Nx2 array of x and y coordinates, in which case the y-input is ignored
`y`	A vector of y-coordinates.

Details

derotateA uses the following standard formula to calculate the angle. Derotate means to remove the xy term, i.e. force B = 0 . Some algebra shows that cot(2theta) = (A-C)/B and thus tan(2theta) = B/(A-C)

For xyrot, the internal xy.coords is used. If you enter only a vector for x and nothing for y, this will feed the new vectors 1:N for x and x-input for y to the rotator, which is probably not useful.

Value

For derotateA,

parA — the new 6-parameter set defining the derotated conic.

theta — the derived angle by which the parameter set was rotated

For rotateA

parA — the new 6-parameter set defining the rotated conic.

theta — the angle by which the parameter set was rotated

For xyrot a Nx2 array of the x,y coordinates of the rotated data set.

Author(s)

Carl Witthoft, <carl@witthoft.com>

See Also

createConic

Examples

```
# make an ellipse and derotate it
parGr <- c(-2.3,4.2,5,3,pi/4)
xe <-seq(-8,9,by=.05)
elipGr <- createConic(xe,  parGr, 'e')
plot(elipGr, t= 'l',  asp = TRUE)
# convert to ABCDEF form
parAr <- GtoA(parGr,'e')
elipAr <- createConic(xe,parAr$parA)
points(elipAr,pch='.',col='red')
# remove rotation angle
 parAd <- derotateA(parAr$parA)
 # returns theta = pi/4, how much the ellipse had been rotated by
 elipAd <-createConic(xe,parAd$parA)
lines(elipAd)
# rotate back
parAdr <- rotateA(parAd$parA, parAd$theta)
elipAdr <-createConic(xe,parAdr$parA)
lines(elipAdr,lty=3, lwd = 3, col='green')
```

Index