Package ‘PEIP’

July 2, 2014

Type Package

Title Functions for Aster Book on Inverse Theory

Version 1.7

Date 2012-05-08

Depends R (>= 2.12)

Imports bvls, Matrix, RSEIS, pracma

Author Jonathan M. Lees

Maintainer Jonathan M. Lees <jonathan.lees@unc.edu>

Description Several Functions for Aster Book on Inverse Theory. These functions are translations of MATLAB code developed by the authors to illustrate the concepts of Inverse theory as applied to geophysics. There is one function, rlsqr, written by Keehoon Kim, that is a wrapper for the FORTRAN based Paige and Saunders LSQR routine.

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-01-07 20:08:11

R topics documented:

PEIP-package — *Inverse Theory Functions for PEIP book*

Description

Auxilliary functions and routines for running the examples and excersizes described in the book on inverse theory.

Details

Package:	PEIP
Type:	Package
Version:	1.3
Date:	2012-07-30
License:	GPL

These functions are used in conjunction with the example described in the PEIP book.

There is one C-code routine, interp2grid. This is introduced to replicate the MATLAB code interp2. It does not work exactly as the matlab code prescribes.

In the PEIP library one LAPACK routine is called: dggsvd. In R, LAPACK routines are stored in slightly different locations on Linux, Windows and Mac computers. Be aware. This will come up in examples from Chapter 4.

Almost all examples work as scripts run with virtually no user input, e.g.

Author(s)

Jonathan M. Lees<jonathan.lees.edu> Maintainer:Jonathan M. Lees<jonathan.lees.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

Ainv — *An Inverse Solution*

Description

QR decomposition solution to Ax=b

Usage

```
Ainv(GAB, x, tol = 1e-12)
```

Arguments

`GAB`	design matrix
`x`	right hand side
`tol`	tolerance for singularity

Details

need something to make up for the lame-o matlab code that does this h = G\x to get the inverse

Value

Inverse Solution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`art` *ART Inverse solution*

Description

ART algorythm for solving sparse linear inverse problems

Usage

```
art(A, b, tolx, maxiter)
```

Arguments

`A`	Constraint matrix
`b`	right hand side
`tolx`	difference tolerance for successive iterations (stopping criteria)
`maxiter`	maximum iterations (stopping criteria).

Details

Alpha is a damping factor. If alpha<1, then we won't take full steps in the ART direction. Using a smaller value of alpha (say alpha=.75) can help with convergence on some problems.

Value

`x`	solution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`astercode` *Compile AsterCode*

Description

Compile (source) the PEIP code in local session. Used it the Package is not installed.

Usage

```
astercode(dir = "RLIB")
```

Arguments

`dir`	directory where code is stored

Details

This function was created and used before the R-code was packaged as a library.

Value

side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`bartl` *Bartlett window*

Description

Bartlett (triangle) window of length m

Usage

```
bartl(m)
```

Arguments

`m`	integer, length of vector

Value

vector

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

Examples

```
bartl(11)
```

---

bayes *Bayes Inversion*

Description

Given a linear inverse problem Gm=d, a prior mean mprior and covariance matrix covm, data d, and data covariance matrix covd, this function computes the MAP solution and the corresponding covariance matrix.

Usage

```
bayes(G, mprior, covm, d, covd)
```

Arguments

`G`	Design Matrix
`mprior`	vector, prior model
`covm`	vector, model covariance
`d`	vector, right hand side
`covd`	vector, data covariance

Value

vector model

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`blf2` *Bounded least squares*

Description

Bounded least squares

Usage

```
blf2(A, b, c, delta, l, u)
```

Arguments

`A`	Design Matrix
`b`	Right hand side
`c`	matrix weight on x
`delta`	tolerance
`l`	lower bound
`u`	upper bound

Details

Solves the problem: min/max c'*x where || Ax-b || <= delta and l <= x <= u.

Value

`x`	solution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

Stark, P.B. , and R. L. Parker, *Bounded-Variable Least-Squares: An Algorithm and Applications*, Computational Statistics 10:129-141, 1995.

`cgls` *Conjugate gradient Least squares*

Description

Conjugate gradient Least squares

Usage

```
cgls(Gmat, dee, niter)
```

Arguments

`Gmat`	input matrix
`dee`	right hand side
`niter`	max number of iterations

Details

Performs niter iterations of the CGLS algorithm on the least squares problem min norm(G*m-d). Gmat should be a sparse matrix.

Value

`X`	matrix of models
`rho`	misfit norms
`eta`	model norms

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

chi	*Chi function*

Description

Chi function

Usage

```
chi(x, n)
```

Arguments

x	value
n	degrees of freedom

Value

function evaluated

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

chi2cdf	*Chi-Sq CDF*

Description

Computes the Chi^2 CDF, using a transformation to N(0,1) on page 333 of Thistead, Elements of Statistical Computing.

Usage

```
chi2cdf(x, n)
```

Arguments

x	end value of chi^2 pdf to integrate to. (scalar)
n	degrees of freedom (scalar)

Details

Note that x and m must be scalars.

Value

`p`	probability that Chi^2 random variable is less than or equal to x (scalar).

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`chi2inv` *Inverse Chi-Sq*

Description

Inverse Chi-Sq

Usage

```
chi2inv(x, n)
```

Arguments

`x`	probability that Chi^2 random variable is less than or equal to x (scalar).
`n`	degrees of freedom(scalar)

Details

Computes the inverse Chi^2 distribution corresponding to a given probability that a Chi^2 random variable with the given degrees of freedom is less than or equal to x. Uses chi2cdf.m.

Value

corresponding value of x for given probability.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

See Also

chi, chi2cdf

`dcost` *cosine transform*

Description

Computes the column-by-column discrete cosine transform of X.

Usage

```
dcost(X)
```

Arguments

`X`	Time series matrix

Value

cosine transformaed data

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`error.bar` *Plot Error Bar*

Description

Plot Error Bar

Usage

```
error.bar(x, y, lo, hi, pch = 1, col = 1, barw = 0.1, add = FALSE, ...)
```

Arguments

`x`	X-values
`y`	Y-values
`lo`	Lower limit of error bars
`hi`	Upper limit of error bars
`pch`	plotting character
`col`	color
`barw`	width of the bar
`add`	logical, add=FALSE starts a new plot
`...`	other plotting parameters

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
x = 1:10
y = 2*x+5
zup = rnorm(10)


zup = zup-min(zup)+.5
zdown = rnorm(10)
zdown = zdown-min(zdown)+.2


####  example with same error on either side:
error.bar(x, y, y-zup, y+zup, pch = 1, col = 'brown' , barw = 0.1, add =
FALSE)


####  example with different  error on either side:
error.bar(x, y, y-zdown, y+zup, pch = 1, col = 'brown' , barw = 0.1, add
= FALSE)
```

`flipGSVD` *Flip output of GSVD*

Description

Flip (reverse order) output of GSVD

Usage

```
flipGSVD(vs, d1 = c(50, 50), d2 = c(48, 50))
```

Arguments

`vs`	list output of GSVD
`d1`	dimensionals of A
`d2`	dimensions of B

Details

This Flipping of the matrix is done to agree with the Matlab code.

Value

`U`

`V`

`X`

`C`

`S`

Note

The GSVD routines are from LAPACK.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

GSVD

`gcval` *Get c-val*

Description

Extract the smallest regularization parameter.

Usage

```
gcval(U, s, b, npoints)
```

Arguments

`U`	U matrix from gsvd(G, L)
`s`	[diag(C) diag(S)] which are the lambdas and mus from the gsvd
`b`	the data to try and match
`npoints`	number of alphas to estimate

Details

Evaluate the GCV function gcv_function at npoints points.

Value

List:

`reg_min`	alpha with the minimal g (scalar)
`g`	|| Gm_(alpha,L) - d ||^2 / (Tr(I - GG#)^2
`alpha`	alpha for the corresponding g

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

gcv_function

`gcv_function` *gcv func*

Description

Auxiliary routine for GCV calculations

Usage

```
gcv_function(alpha, gamma2, beta)
```

Arguments

`alpha`	parameter
`gamma2`	square of the gamma from the gsvd
`beta`	projected data to fit

Value

vector, g - || Gm_(alpha,L) - d ||^2 / (Tr(I - GG#)^2

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`get_l_rough` *One-D Roughening*

Description

returns a 1D differentiating matrix operating on a series with n points.

Usage

```
get_l_rough(n, deg)
```

Arguments

`n`	number of data points
`deg`	order of the derivative to approximate

Details

Used to get first and 2nd order roughening matrices

Value

Matrix:discrete differentiation matrix

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

---

ginv *Get inverse*

Description

Get inverse of matrisx or solve Ax=b.

Usage

```
ginv(G, x, tol = 1e-12)
```

Arguments

`G`	Design Matrix
`x`	right hand side
`tol`	tolerance

Details

This function used as alternative to matlab code that does this h = G\x to get the inverse

Value

inverse

Note

Be careful about the usage of tolerance

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

solve, Ainv

GSVD *Generalized svd*

Description

Generalized svd

Usage

```
GSVD(A, B)
```

Arguments

A	U*E1*t(Q)
B	V*E2*t(Q)

Value

U	matrix
V	matrix
X	matrix
C	matrix
S	matrix

Note

lapack

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`idcost` *Inverse cosine transform*

Description

Takes the column-by-column inverse discrete cosine transform of Y.

Usage

```
idcost(Y)
```

Arguments

`Y`	Input cosine transform

Value

Time series

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

See Also

dcost

`imagesc` *Image Display*

Description

Display image in matlab format, i.e. flip and transpose.

Usage

```
imagesc(G, col = grey((1:99)/100), ...)
```

Arguments

`G`	Image matrix
`col`	color scale
`...`	graphical parameters

Details

Program flips image and transposes.

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`interp2grid` *Bilinear and Bicubic Interpolation to Grid*

Description

This code was includes bicubic interpolation and bilinear interpolation adapted from Numerical Recipes in C: The are of scientific computing `http://www.nrbook.com/nr3/` (chapter 3... bicubic interpolation) and a bicubic interpolation from `http://www.paulinternet.nl/?page=bicubic` in java code.

Inputs are a list of points to interpolate to from raster objects of class 'asc' (adehabitat package), 'RasterLayer' (raster package) or 'SpatialGridDataFrame' (sp package).

Usage

```
interp2grid(mat,xout,yout,xin=NULL,yin=NULL,type=2)
```

Arguments

`mat`	a matrix of data that can be a raster matrix of class 'asc' (adehabitat package), 'RasterLayer' (raster package) or 'SpatialGridDataFrame' (sp package) NA values are not permitted.. data must be complete.
`xout`	a vector of data representing x coordinates of the output grid. Resulting grid must have square cell sizes if mat is of class 'asc', 'RasterLayer' or 'SpatialGridDataFrame'.
`yout`	a vector of data representing x coordinates of the output grid. Resulting grid must have square cell sizes if mat is of class 'asc', 'RasterLayer' or 'SpatialGridDataFrame'.

`xin`	a vector identifying the locations of the columns of the input data matrix. These are automatically populated if mat is of class 'asc', 'RasterLayer' or 'SpatialGridDataFrame'.
`yin`	a vector identifying the locations of the rows of the input data matrix. These are automatically populated if mat is of class 'asc', 'RasterLayer' or 'SpatialGridDataFrame'.
`type`	an integer value representing the type of interpolation method used. 1 - bilinear adapted from Numerical Recipes in C 2 - bicubic adapted from Numerical Recipes in C 3 - bicubic adapted from online java code

Value

Returns a matrix of the originating class.

Author(s)

Jeremy VanDerWal <jjvanderwal@gmail.com>

Examples

```
tx = seq(0,3,0.1)
ty = seq(0,3,0.1)

     tmat = matrix(runif(16,1,16),nrow=4)
     txin = seq(0,3,length=4)
     tyin = seq(0,3,length=4)

     bilinear1 = interp2grid(tmat,tx,ty,txin, tyin,    type=1)
     bicubic2 = interp2grid(tmat,tx,ty,txin, tyin, type=2)
     bicubic3 = interp2grid(tmat,tx,ty,txin, tyin, type=3)

    par(mfrow=c(2,2),cex=1)
             image(tmat,main='base',zlim=c(0,16),col=heat.colors(100))
             image(bilinear1,main='bilinear',zlim=c(0,16),col=heat.colors(100))
             image(bicubic2,main='bicubic2',zlim=c(0,16),col=heat.colors(100))
             image(bicubic3,main='bicubic3',zlim=c(0,16),col=heat.colors(100))
```

`irls` *Iteratively reweight least squares*

Description

Uses the iteratively reweight least squares strategy to find an approximate L_p solution to Ax=b.

Usage

```
irls(A, b, tolr, tolx, p, maxiter)
```

Arguments

A	Matrix of the system of equations.
b	Right hand side of the system of equations
tolr	Tolerance below which residuals are ignored
tolx	Stopping tolerance. Stop when (norm(newx-x)/(1+norm(x)) < tolx)
p	Specifies which p-norm to use (most often, p=1.)
maxiter	Limit on number of iterations of IRLS

Details

Use to get L-1 norm solution of inverse problems.

Value

x	Approximate L_p solution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

---

irlsl1reg *L1 least squares with sparsity*

Description

Solves the system Gm=d using sparsity regularization on Lm. Solves the L1 regularized least squares problem: min norm(G*m-d,2)^2+alpha*norm(L*m,1)

Usage

```
irlsl1reg(G, d, L, alpha, maxiter = 100, tolx = 1e-04, tolr = 1e-06)
```

Arguments

`G`	design matrix
`d`	right hand side
`L`	regularization matrix
`alpha`	regularization parameter
`maxiter`	Maximum number of IRLS iterations
`tolx`	Tolerance on successive iterates
`tolr`	Tolerance below which we consider an element of L*m to be effectively zero

Value

`m`	model vector

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`kac` *Kaczmarz*

Description

Implements Kaczmarz's algorithm to solve a system of equations iteratively

Usage

```
kac(A, b, tolx, maxiter)
```

Arguments

`A`	Constraint matrix
`b`	right hand side
`tolx`	difference tolerence for successive iterations (stopping criteria)
`maxiter`	maximum iterations (stopping criteria)

Value

`x`	solution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

`linesconst` *Plot constant model*

Description

Add to plotting model in piecewise constant form over n subintervals, where n is the length of x.

Usage

```
linesconst(x, l, r, ...)
```

Arguments

`x`	model to be plotted
`l`	left endpoint of plot
`r`	right endpoint of plot
`...`	graphical parameters

Details

Used for plotting vector models

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

plotconst

`lmarq` *Lev-Marquardt Inversion*

Description

Use the Levenberg-Marquardt algorithm to minimize f(p)=sum(F_i(p)^2)

Usage

```
lmarq(afun, ajac, p0, tol, maxiter)
```

Arguments

afun	name of the function F(x)
ajac	name of the Jacobian function J(x)
p0	initial guess
tol	stopping tolerance
maxiter	maximum number of iterations allowed

Value

pstar	best solution found.
iter	Iteration count.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`loadMAT` *Load a Matlab matfile*

Description

Load a Matlab matfile, rename the internal parameters to get R-objects

Usage

```
loadMAT(fn, pos=1)
```

Arguments

fn	file name of MATfile
pos	integer, position in search path, default=1

Details

Program reads in previously saved mat-files and extracts the data, and renames the variables to match the book.

Value

Whatever is in the MATfile

Note

Matfiles are created using the matlab2R routines

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`l_curve_corner` *L Curve Corner*

Description

Retrieve corner of L-curve

Usage

```
l_curve_corner(rho, eta, reg_param)
```

Arguments

`rho`	misfit
`eta`	model norm or seminorm
`reg_param`	regularization parameter

Value

`reg_corner`	the value of reg_param with maximum curvature
`ireg_corner`	the index of the value in reg_param with maximum curvature
`kappa`	the curvature for each reg_param

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

l_curve_tgsvd *L curve tgsvd*

Description

L curve paremater and models for truncated gsvd regularization.

Usage

```
l_curve_tgsvd(U, d, X, Lam, G, L)
```

Arguments

U	U, output of GSVD
d	output of GSVD
X	output of GSVD
Lam	output of GSVD
G	output of GSVD
L	output of GSVD

Value

List:

eta	the solution seminorm ‖Lm‖
rho	the residual norm ‖G m - d‖
reg_param	corresponding regularization parameters
m	corresponding suite of models for truncated GSVD

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aster, R.C., C.H. Thurber, and B. Borchers, *Parameter Estimation and Inverse Problems*, Elsevier Academic Press, Amsterdam, 2005.

l_curve_tikh_gsvd *L-curve tikh gsvd*

Description

L-curve tikh gsvd

Usage

```
l_curve_tikh_gsvd(U, d, X, Lam, Mu, G, L, npoints, varargin = NULL)
```

Arguments

U	from the gsvd
d	data vector for the problem G*m=d
X	from the gsvd
Lam	from the gsvd
Mu	from the gsvd
G	system matrix
L	roughening matrix
npoints	
varargin	alpha_min, alpha_max: if specified, constrain the logrithmically spaced regularization parameter range, otherwise an attempt is made to estimate them from the range of generalized singular values

Details

Uses output of GSVD

Value

eta	- the solution seminorm ||Lm||
rho	- the residual norm ||G m - d||
reg_param	- corresponding regularization parameters
m	- corresponding suite of models for truncated GSVD

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

l_curve_tikh_svd *L-curve Tikhonov*

Description

L-curve for Tikhonov regularization

Usage

```
l_curve_tikh_svd(U, s, d, npoints, varargin = NULL)
```

Arguments

U	matrix of data space basis vectors from the svd
s	vector of singular values
d	the data vector
npoints	the number of logarithmically spaced regularization parameters
varargin	alpha_min, alpha_max: if specified, constrain the logrithmically spaced regularization parameter range, otherwise an attempt is made to estimate them from the range of singular values

Details

Calculates the L-curve

Value

eta	the solution norm ||m|| or seminorm ||Lm||
rho	the residual norm ||G m - d||
reg_param	corresponding regularization parameters

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

mcmc *Maximum likelihood Models*

Description

Maximum likelihood Models

Usage

```
mcmc(alogprior, aloglikelihood, agenerate, alogproposal, m0, niter)
```

Arguments

alogprior	Name of a function that computes the log of the prior distribution.
aloglikelihood	Name of a function the computes the log of the likelihood.
agenerate	Name of a function that generates a random model from the current model using the
alogproposal	Name of a function that computes the log of the proposal distribution r(x,y).
m0	Initial model
niter	Number of iterations to perform

Value

mout	MCMC samples
mMAP	Best model found in the MCMC simulation.
accrate	Acceptance rate

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Mnorm *Matrix Norm*

Description

Matrix Norm

Usage

```
Mnorm(X, k = 2)
```

Arguments

`X`	matrix
`k`	norm number

Details

returns the largest singular value of the matrix or vector

Value

Scalar Norm

Note

if k=1, absolute value; k=2 2-norm (rms); k>2, largest singular value.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`nnz` *Non-zeros*

Description

Number of non-zero elements in a vector

Usage

```
nnz(h)
```

Arguments

`h`	vector

Value

integer

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

occam *Occam inversion*

Description

Occam's inversion

Usage

```
occam(afun, ajac, L, d, m0, delta)
```

Arguments

`afun`	character, function handle that computes the forward problem
`ajac`	character, function handle that computes the Jacobian of the forward problem
`L`	regularization matrix
`d`	data that should be fit
`m0`	guess at the model
`delta`	cutoff to use for the discrepancy principle portion

Value

vector, model found

Note

This is a simple brute force way to do the line search. Much more sophisticated methods are available. Note: we've restricted the line search to the range from 1.0e-20 to 1. This seems to work well in practice, but might need to be adjusted for a particular problem.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

bayes

phi — *Integral of Normal Distribution*

Description

normal distribution and returns the value of the integral

Usage

```
phi(x)
```

Arguments

`x`	endpoint of integration (scalar)

Value

value of integral

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

erf

phiinv — *Inverse Normal Distribution Integral*

Description

Calculates the inverse normal distribution from the value of the integral

Usage

```
phiinv(x)
```

Arguments

`x`	endpoint value of integration (scalar)

Value

value of integral (scalar)

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

phi

`picard_vals` *Picard plot*

Description

Picard plot parameters for subsequent plotting

Usage

```
picard_vals(U, sm, d)
```

Arguments

`U`	the U matrix from the SVD or GSVD
`sm`	singular values in decreasing order, or the lambdas divided by the mus in decreasing order
`d`	data to fit

Value

List:

`utd`	the columns of U transposed times d
`utd_norm`	utd./sm

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

plotconst *Plot constant model*

Description

Plots a model in piecewise constant form over n subintervals, where n is the length of x.

Usage

```
plotconst(x, l, r, ...)
```

Arguments

x	model to be plotted
l	left endpoint of plot
r	right endpoint of plot
...	graphical parameters

Details

Used for plotting vector models

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

linesconst

quadlin *Lagrange multiplier technique*

Description

Quadratic Linearization

Usage

```
quadlin(Q, A, b)
```

Arguments

`Q`	positive definite symmetric matrix
`A`	matrix with linearly independent rows
`b`	data vector

Details

Solves the problem: min (1/2) t(x)*Q*x with Ax = b. using the Lagrange multiplier technique, where Q is assumed to be symmetric and positive definite and the rows of A are linearly independent.

Value

list:

`x`	vector of solution values
`lambda`	Lagrange multiplier

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

`rlsqr` *rlsqr*

Description

R Version of LSQR routine written in Fortran by Paige and Saunders.

Usage

```
rlsqr(G=matrix(), u=vector(), wantse = 0, damp = 0, atol = 0,
 btol = 0, conlim = 0, itnlim = 100, nout = 0)
```

Arguments

`G`	Design Matrix
`u`	data vector
`wantse`	weighting?
`damp`	Damping parameter
`atol`	a Tolorance
`btol`	b Tolorance
`conlim`	con
`itnlim`	Iteration limit
`nout`	integer, output file (Not Available)

Details

This code is an R wrapper for performing the LSQR routine by Paige and Saunders. The LSQR program is a popular inversion program for solving hte least squares problem, Ax=b.

Value

Solution vector

Note

The fortran code has write statements that are not available in R-code according to the standard R documentation.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>, Keehoon Kim<keehoon@email.unc.edu>

References

Paige, C., and Saunders, M., LSQR: An algorithm for sparse linear equations and sparse least squares. Trans. Math. Software, 1982, 8, 43-71

See Also

sirt, art

Examples

```
## Not run:


library(png)
library(PEIP)
shift=function(x,ns)
{
nsig=c(rep(0,ns),x[1:(length(x)-ns)])
return(nsig)
}

imagesc<-function(G, col=grey((1:99)/100), ... )
  {
    #########  plot an image after flipping and transposing
    ###   to match the matlab code
   d = dim(G)
   b = G[d[1]:1,]
   image(t(b), col=col, ...)
  }
img= readPNG('original_image.png')
im = matrix(as.vector(img) , 40000, 1)
```

```
load('rtest.RDATA')


################################################
# Blurring the image (DATA == d)
################################################

d0=G
d=d0


################################################
# Solving damped least squares problem
# (    G     ) * x  =  ( d )
# ( damp * I )         ( 0 )
################################################

# damp = lambda
lambda=0

# data vector (u)
u=d

# tolerance of iteration
atol=1.0e-6;btol=1.0e-6
#atol=.Machine$double.eps;btol=.Machine$double.eps

# taking condition number of G into consideration ( 0 means to ignore it )
conlim=0

# the number of iteration of LSQR
itnlim=1000;



################################################
# Running LSQR
################################################

# Executing LSQR
lx=rlsqr(G=G,u=u,damp=lambda,itnlim=itnlim,atol=atol,btol=btol,conlim=0)
# lx$x : model vector inverted from LSQR
# lx$xnorm : || x ||^2
# lx$rnorm : || Gx - d ||^2 (Norm of residual)
# lx$itn : the number of iteration to get the result


# Results of LSQR


layout(matrix(1:3, ncol=3))
imagesc(matrix(img,200,200),main='Original Image',axes=FALSE, xlab='',ylab='')
imagesc(matrix(d,200,200),main='Blurred Image',axes=FALSE,xlab='',ylab='')
```

```
imagesc(matrix(lx$x,200,200),
main=paste('LSQR solution Itr. = ',lx$itn),axes=FALSE,xlab='',ylab='')

## End(Not run)
```

---

rnk *Rank of Matrix*

Description

Return the rank of a matrix. Not to be confused with the R function rank.

Usage

```
rnk(G, tol = 1e-14)
```

Arguments

G	Matrix
tol	machine tolerance for small numbers

Details

Number of singular values greater than tol.

Value

integer, number of non-zero singular values

Note

duplicate the matlab function rank

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

svd

Examples

```
hilbert <- function(n) { i <- 1:n; 1 / outer(i - 1, i, "+") }
X <- hilbert(9)[,1:6]
rnk(X)
```

---

sirt — *SIRT Algorithm for sparse matrix inversion*

Description

Row action method for inversion of matrices

Usage

```
sirt(A, b, tolx, maxiter)
```

Arguments

`A`	Design Matrix
`b`	vector, Right hand side
`tolx`	numeric, tolerance for stopping
`maxiter`	integer, Maximum iterations

Details

Iterates until conversion

Value

Solution vector

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

See Lees, 1989

See Also

art, kac

tinv *Inverse T-distribution*

Description

Inverse T-distribution, qt

Usage

```
tinv(p, nu)
```

Arguments

p	P-value
nu	degrees of freedom

Details

Wrapper for qt

Value

Quantile for T-distribution

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

qt

Examples

```
tinv(.4, 10)
```

USV *Singular Value Decomposition*

Description

Singular Value Decomposition

Usage

```
USV(G)
```

Arguments

G	Matrix

Details

returns matrices U, S, V according to matlab convention.

Value

list:

U	Matrix
S	Matrix, singular values
V	Matrix

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

svd

Examples

```
 hilbert <- function(n) { i <- 1:n; 1 / outer(i - 1, i, "+") }
     X <- hilbert(9)[,1:6]

h = USV(X)

print( h$U )
```

`Vnorm` *Vector 2-Norm*

Description

Vector 2-Norm.

Usage

```
Vnorm(X)
```

Arguments

X	numeric vector

Value

Numeric scale norm

Note

This function is intended to duplicated the matlab function norm.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
V = Vnorm(rnorm(10))
```

`wGSVD` *wrapper for GSVD*

Description

wrapper for GSVD, dggsvd, generalized SVD

Usage

```
wGSVD(A, B)
```

Arguments

A	matrix, A,A=U*E1*t(Q)
B	matrix, V*E2*t(Q)

Details

Uses LAPACK routine dggsvd

Value

list: U,V, X, C,S

Note

This function calls the LAPACK routine installed on the system. For LINUX, MAC and WINDOWS there may be slightly different implementation.

The order of the singular values returned are according to Matlab convention, smallest to largest.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

svd

Index