# Package 'LPI.gdf'

April 29, 2019

**Version** 0.1-5

**Date** 2015-11-18

**Title** Lag Profile Inversion with gdf files

**Author** Ilkka Virtanen <ilkka.i.virtanen@oulu.fi>

**Maintainer** Ilkka Virtanen <ilkka.i.virtanen@oulu.fi>

**Description** gdf format I/O routines, wrapper function to LPI, and graphics

**License** FreeBSD

**Depends** LPI, radarPointings, lattice, lomb



## R topics documented:



---

`LPI.gdf-package` *Lag Profile Inversion with gdf files*

---

### Description

gdf format I/O routines, wrapper function to LPI, and graphics

### Details

| | |
|---|---|
| Package: | LPI.gdf |
| Version: | 0.1-4 |
| Date: | 2014-01-27 |
| License: FreeBSD | |
| Depends: LPI, radarPointings, lattice | |
| Built: | |

Index:



## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

getDataEndTimes.gdf *getDataEndTimes.gdf*

---

## Description

Read sampling times of latest data samples

## Usage

```
getDataEndTimes.gdf( LPIparam )
```

## Arguments

| | |
|---|---|
| LPIparam | An LPI parameter list from LPI.gdf |

### Details

Following elements of the LPI parameter list are used in this function.

**'dataDir'** A named character vector that contains the data directories for each data type.

**'fileNamePrefix'** A named character vector that contains the file name prefixes for each data type.

**'fileNameExtension'** A named character vector that contains the file name extensions for each data type.

**'dataFileLengths'** A named numeric vector that contains number of samples in a single data file of each data type.

**'dataSampleFreqs'** A named numeric vector that contains sample rates of each data type in Hz.

**'dataStartTimes'** A named numeric vector that contains sampling times of the first samples of each data type in POSIX format.

### Value

A named vector with elements "RX1","RX2","TX1","TX2", each of which is the sampling time of the latest recorded data sample of the corresponding type in POSIX format.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`getFileLengths.gdf` *getFileLengths.gdf*

---

### Description

Read file lengths of gdf format data

### Usage

```
getFileLengths.gdf( LPIparam )
```

### Arguments

| | |
|---|---|
| `LPIparam` | An LPI parameter list from LPI.gdf |

### Details

Following elements of the LPI parameter list are used in this function.

**'dataDir'** A named character vector that contains the data directories for each data type.

**'fileNamePrefix'** A named character vector that contains the file name prefixes for each data type.

**'fileNameExtension'** A named character vector that contains the file name extensions for each data type.

### Value

A named vector with elements "RX1","RX2","TX1","TX2", each of which is the number of samples in data files of the corresponding data type.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`getSamplingFrequencies.gdf` *getSamplingFrequencies.gdf*

---

### Description

Read sample rates of gdf format data

### Usage

```
getSamplingFrequencies.gdf( LPIparam )
```

### Arguments

| | |
|---|---|
| `LPIparam` | An LPI parameter list from LPI.gdf |

### Details

Following elements of the LPI parameter list are used in this function.

**'dataDir'** A named character vector that contains the data directories for each data type.

### Value

A named vector with elements "RX1","RX2","TX1","TX2", each of which is the sample rate of the corresponding data type in Hz.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`getSamplingStartTimes.gdf` *getSamplingStartTimes.gdf*

---

### Description

Read sampling start times from gdf format data directories

### Usage

```
getSamplingStartTimes.gdf( LPIparam )
```

### Arguments

| | |
|---|---|
| `LPIparam` | An LPI parameter list from LPI.gdf |

### Details

Following elements of the LPI parameter list are used in this function.

**'dataDir'** A named character vector that contains the data directories for each data type.

**'fileNamePrefix'** A named character vector that contains the file name prefixes for each data type.

**'fileNameExtension'** A named character vector that contains the file name extensions for each data type.

**'dataFileLengths'** A named numeric vector that contains number of samples in a single data file of each data type.

**'dataSampleFreqs'** A named numeric vector that contains sample rates of each data type in Hz.

### Value

A named vector with elements "RX1","RX2","TX1","TX2", each of which is the sampling time of the first data sample of the corresponding type in POSIX format.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`LPI.gdf` *LPI.gdf*

---

### Description

Lag Profile Inversion with gdf files

### Usage

```
LPI.gdf( ... )
```

### Arguments

| | |
|---|---|
| `...` | Arguments to be passed for functions that parse the final LPI input argument list. Accepts a reference to a stored input argument file and / or direct command line arguments. All parameters have default values which are used if the parameter is not found elsewhere.<br>Following arguments are effective:<br>**'paramFile'** A file containing any subset of the accepted input arguments. In conflict situations values from command line will override those read from file. See details.<br>Default: NULL<br>**'clusterNodes'** A list defining the computer cluster, see help of the main LPI package.<br>Default: list(tesla1=8,tesla2=8,tesla3=8,tesla4=8,tesla5=8,localControl=TRUE) |

**'begintTime'** Analysis start time, c( year , month , day , hour , minute , seconds), see details.
Default: c(1970,1,1,0,0,0)

**'endTime'** Analysis end time, see details. c( year , month , day , hour , minute , seconds), see details.
Default: c(3000,1,1,0,0,0)

**'timeRes.s'** Analysis time-resolution (integration time) in seconds.
Default: 10

**'maxWait.s'** Maximum time to wait for new data before stopping the analysis, in seconds.
Default: -1

**# Basic definitions of the analysis**

**'freqOffset.Hz'** Frequency offset from baseband in Hz, see details.
Default: 0.0

**'filterLength.us'** Length of the boxcar-shaped post-detection filter in us.
Default: 10

**'lagLimits.us'** Limits of time-lag gates in us, see details.
Default: seq(50)*10

**'rangeLimits.km'** Limits of range gates in km, see details.
Default: c(seq(50,149.9,by=.9),seq(154.4,298.4,by=4.5),seq(343.4,793.5,by=45))

**'maxRanges.km'** Maximum range for each lag profile. Allows the number of range gates to be limited at selected lags. See details.
Default: Inf

**'nCode'** Code cycle length for the optional lag profile pre-averaging, see help of the main LPI package.
Default: NA

**'backgroundEstimate'** Optional background ACF suppression. See help of the main LPI package.
Default: TRUE

**'fullCovar'** Full covariance matrix / variance calculation selection. See help of the main LPI package.
Default: FALSE

**'decodingFilter'** Amplitude domain decoding filter selection. See help of the main LPI package.
Default: "none"

**'maxClutterRange.km'** Maximum range at which ground clutter will be subtracted. Set below min(rangeLimits.km) in order to disable the clutter suppression. See details.
Default: 60

**'clutterFraction'** The fraction of a full integration period used for the coherent ground clutter profile estimation. A float from between (0,1]. Values smaller than 1 should be used carefully. See help of the main LPI package.
Default: 1.0

**'remoteRX'** Monostatic / bistatic analysis selection. See help of the main LPI package.
Default: FALSE

**'ambInterp'** Range ambiguity function oversampling on / off switch. See help of the main LPI package.
Default: FALSE

**'indexShifts.us'** Additional adjustements to TX and RX indices in us, see details.
Default: list( TX=c(0,0) , RX=c(0,0) )

**'normTX'** Optional TX amplitude normalization. See help of the main LPI package.
Default: FALSE

**'llhT'** Latitude [deg], longitude [deg], and height [m] of the transmitter.
Default: c( 69.58 , 19.23 , 86.00 ) # EISCAT Tromso

**'llhR'** Latitude [deg], longitude [deg], and height [m] of the receiver.
Default: c( 69.58 , 19.23 , 86.00 ) # EISCAT Tromso

**'azelT'** Azimuth and elevation of the transmitter beam [deg]. Azimuth 0 ... 360, north=0, east=90.
Default: c(0,90)

**'azelR'** Azimuth and elevation of the receiver beam.
Default: c(0,90)

**'radarFreq'** Transmitter carrier frequency [Hz].
Default: 224e6

**'solver'** Inverse problem solver selection. See help of the main LPI package.
Default: "fishs"

**'nBuf'** Number of theory matrix rows to buffer before calling the solver function.
Default: 10000

**'rlips.options'** Additional options to the rlips solver. See rlips help for details.
Default: list( type="c" , nbuf=1000 , workgroup.size=128)

**'dataDir'** Raw voltage data directory / directories, see details.
Default: "."

**'resultDir'** Output directory.
Default: paste(format(Sys.time(),"

## Details

The final argument list is constructed from default values, values read from the optional parameter file, and command line arguments. Priorities of these, from lowest to highest, are defaults, values from a file, and command line arguments. Name collisions are solved by selecting the option with highest priority. In other words, command line arguments will override both defaults and file input, and file input will override the defaults. Defaults are used only if nothing else is available.

**'paramFile'** Usually a file that the analysis has automatically stored in a previous LPI run. The file is stored in the output directory ('resultDir') under the name "LPIparam.Rdata". Giving such a file as an input argument is the simplest way to restart an analysis with identical parameters. It is also possible to e.g. store parameters specific to an experiment in a file in order to avoid long input argument lists.

**'lagLimits.us'** A vector of time lag limits. The vector is first divided with filterLength.us, then rounded to nearest integer, and finally possible dublicate values are removed. Output of this process is a vector lagLimits = c( l1, l2, l3, lN ). The analysis then integrates lags l1, l1+1, ... l2-1 (of decimated data) into lag profile 1, lags l2, l2+1, ... l3-1 into lag profile 2, etc. As an example, in order to decimate data to 10 us sample rate and to calculate all lags up to 100 us separately, one would use filterLength.us=10 and lagLimits.us = c(10,20,30,40,50,60,70,80,90,100,110). The steps in lagLimits.us do not need to be of equal length. It is possible to use coarser lag resolution at longer lags by selecting e.g. lagLimits.us = c(10,20,30,50,70,90,110).

**'rangeLimits.km'** A vector of range-gate limits. The vector is first converted into time-delays assuming that signal propagates at speed of light, then decimated to filterLength.us sample rate, and possible duplicate values are removed. Range-gates are then defined in exactly the same way as lag-gates (see above).

**'maxRanges.km'** Allows one to reduce the number of range-gates at selected lags. If the vector is shorter than lagLimits.us its last value is repeated as necessary. Defaults to Inf, which means that all lags are solved at all range gates defined in rangeLimits.km.

maxRanges.km may be used in combined D/E/F-region experiments when correlations longer than certain limit are known to exist only in D-region. As an example, in order to solve first 30 lags at all range gates, but the longer ones only below 100 km, one can use maxRanges.km = c( rep( Inf , 30 ) , 100 ).

Another use case is a dedicated D region experiment that makes use of voltage level decoding. Most time lags will then have zero range ambiguity functions and can be safely skipped. Assuming that modulation bit length is 2 us and inter-pulse period is 1 ms, one could use e.g. the following combination: filterLength.us=2, decodingFilter='matched', lagLimits.us=c(1000,1002,2000,2002,3000,3002), maxRanges=c(Inf,0,Inf,0,Inf) which would solve three pulse-to-pulse lags and save the analysis from inspecting all time lags (only 1 in 500 of them can actually be measured).

**'beginTime'** A vector c( year , month , day , hour , minute , seconds ) of the desired analysis start time. If data is available from the given time, this is the start time of the first integration period. If data is not available from this time, the first integration period is still aligned with "beginTime", and the analys is started from the first available data sample that was recorded after "beginTime". Notice that, in order to make the software more suitable for real-time analysis, the *latest* available data is always used first, and the first integration period may thus be the last to actually analyse.

**'endTime'** A vector c( year , month , day , hour , minute , seconds ) of the analysis end time. Data samples recorded after "endTime" are not used.

**'freqOffset.Hz'** Frequency offset from baseband to the signal centre frequency. Currently only one frequency per data vector, i.e. either a single offset for all data, or different shifts for "RX1", "RX2", "TX1", and "TX2". See the details for "dataDir" for instructions.

**'indexShifts.us'** Additional adjustments to the TX (and) RX indices returned by 'dataInputFunction'. The user input is converted into a list with entries "RX1", "RX2", "TX1", and "TX2" in the way explained in details for "dataDir". Each entry is a two-element vector c( shift1, shift2 ) where, in case of TX, shift1 is the adjustment at rising edge of each pulse, and shift2 is the adjustment at falling edge of each pulse. For RX, shift1 is the adjustment at each start of reception, and shift2 at end of reception. All shifts are positive forwards, use negative values to adjust towards earlier times. As an example, if the recorded TX bit for "TX1" starts 10 us *before* the actual pulse, and ends 20 us *after* the actual end of the pulse, one should have indexShifts.us["TX1"] == c(10,-20).

**'dataDir'** Either a character string of a single data directory, or a named vector of several data directories. The internally used "dataDir" is always a vector with entries "RX1", "RX2", "TX1", and "TX2", some or all of which may be equal. The analysis calculates lagged products in between data samples from "RX1" and "RX2", and range ambiguity functions as lagged products in between data samples from "TX1" and "TX2". The internally used list is formed from user input as follows:

1. If the input vector contains any of the entries "RX1", "RX2", "TX1", "TX2", they are copied as such.

2. If any of the four entries is missing after step 1, the input is searched for "RX", which is used to replace missing "RX1" and / or "RX2". A similar search is performed for "TX".

3. If any of the four entries is missig after steps 1 and 2, the input is searched for "TR1", which is used to replace "RX1" and / or "TX1". A similar search is performed for "TR2"

4. If any of the four entries is missing after steps 1 to 3, the *unnamed* elements of the input vector are repeated until a vector of length 4 is formed, and its elements are named "RX1", "RX2", "TX1", and "TX2".

5. If still unsuccessfull, i.e. the input vector was either empty or it contained only named entries with unknown names, the analysis will stop with an error message.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

## LPIparam.default.gdf *LPIparam.default.gdf*

---

### Description

LPI default parameters

### Usage

```
LPIparam.default.gdf()
```

### Arguments

None

### Value

A named list of LPI.gdf default arguments. See LPI.gdf for details.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

## LPIsaveACF.gdf *LPIsaveACF.gdf*

---

### Description

Save LPI results to files

### Usage

```
LPIsaveACF.gdf( LPIparam , intPeriod , ACF )
```

### Arguments

| | |
|---|---|
| LPIparam | An LPI parameter list from LPI.gdf |
| intPeriod | Integration period number. |
| ACF | An ACF list. See LPIsaveACF from the main LPI package for details. |

## Value

Result file name (invisibly)

## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`plotACF` *plot ACF*

---

## Description

Plot incoherent scatter autocovariance function.

## Usage

```
 plotACF( data , part=real , pdf=NULL , jpg=NULL , figNum=NULL
, zlim=NULL , ylim=NULL , xlim=NULL , cex=1 , bg=white , fg=black ,
width=8.27 , height=11.69 , paper=a4 , res=300 , stdThrsh=Inf ,
yheight=FALSE , llhT=NULL , azelT=NULL , llhR=NULL , lags=NULL ,
SIunits=TRUE )
```

## Arguments

| | |
|---|---|
| `data` | Data directory path(s) or an output list from readLPIdir or plotLagProfilfes. The data path is a vector that may contain both full paths to files and directory names. |
| `part` | Real part / imaginary part / standard deviation plot selection. Use string "real", "imaginary", or "error". (Only the first two characters are used, case non-sensitive. ) |
| `pdf` | pdf output file name |
| `jpg` | jpg output file name |
| `figNum` | Device number to use for plotting |
| `zlim` | z axis limits |
| `ylim` | y axis limits |
| `xlim` | x axis (lag) limits |
| `cex` | Scaling factor for figure labels and titles. |
| `bg` | Background color |
| `fg` | Foreground color |
| `width` | plot width |
| `height` | plot height |
| `paper` | paper seletion |
| `res` | resolution for jpg images |
| `stdThrsh` | Standard deviation threshold for readACF |
| `yheight` | Logical, should the range be converted to height? Works only if the result files contain site positions and pointing directions. |

| | |
|---|---|
| llhT | c( Latitude [deg], longitude [deg], height [m] ) of the transmitter site |
| azelT | c(azimuth [deg] , elevation [deg]) of the transmitter beam |
| llhR | c( Latitude [deg], longitude [deg], height [m] ) of the receiver site |
| lags | Time lag selection. Effective only if data is a file path. |
| SIunits | Logical, should range be expressed in km and lag in ms? Works only if range.km and lag.us are stored in the data files |

## Value

A list similar with that returned by readACF

## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`plotLagProfiles` *plot lag profiles*

---

## Description

Plot lag profiles as function of time and range.

## Usage

```
plotLagProfiles( data , lags=c(1) , f=nothing, re=TRUE , im=FALSE , mod=FALSE , arg=FALSE , err
```

## Arguments

| | |
|---|---|
| data | Data directory path(s) or an output list from readLPIdir or plotLagProfilfes. The data path is a vector that may contain both full paths to files and directory names. |
| lags | Lags to integrate before plotting. Numbered starting from shortest lag in result files. Default 1. See details. |
| f | Function to apply to the data points before plotting. The default function "nothing" does nothing. |
| re | Logical, include a plot of the lag profile real parts? |
| im | Logical, include a plot of the lag profile imaginary parts? |
| mod | Logical, include a plot of the lag profile modulus? |
| arg | Logical, include a plot of the lag profile argument (phase angle) ? |
| err | Logical, include a plot of the lag profile standard deviation? |
| xlim | x (time) axis limits in hours. Counting starts from the beginning of the day when the first integration period found from "dpath" ended. |
| ylim | Range / height limits in the plot |
| relim | z limits for real part |
| imlim | z limits for imaginary part |
| errlim | z limits for standard deviation |

| | |
|---|---|
| modlim | z limits for modulus |
| arglim | z limits for argument (phase) |
| stdThresh | Standard deviation threshold to pass to the function readLPIdir. |
| cex | Scaling factor for figure labels and titles. |
| col | Color scale to use. Default "beer" is Lovibond scale. |
| yheight | Logical, should the range be converted to height? Works only if the result files contain site positions and pointing directions. |
| llhT | c( Latitude [deg], longitude [deg], height [m] ) of the transmitter site |
| azelT | c(azimuth [deg] , elevation [deg]) of the transmitter beam |
| llhR | c( Latitude [deg], longitude [deg], height [m] ) of the receiver site |
| SIunits | Logical, should range be expressed in km and lag in ms? Works only if range.km and lag.us are stored in the data files |
| tickRes | Time tick resolution in seconds, NULL=automatic |
| figNum | Device number to use for plotting |
| pdf | pdf output file name |
| jpg | jpg output file name |
| width | plot width |
| height | plot height |
| paper | paper seletion |
| res | resolution for jpg images |

## Value

A list similar to that returned by readACF

## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

# readACF *Average ACF*

---

## Description

Read LPI result files and return the average ACF

## Usage

```
readACF( dpath , lags=NULL , ranges=NULL , stdThrsh=Inf )
```

## Arguments

| | |
|---|---|
| dpath | Data file / directory path(s) as a vector of character strings. |
| lags | Lag selection. Either a vector of lag numbers, or "all" to read all lags |
| ranges | Range gate selection |
| stdThrsh | Standard deviation threshold. Individual data points with standard deviation larger than stdThresh in absolute units are not included in the average |

### Value

A list with the following elements plus everything listed in the "ACF" list stored in the last data file, except the full covariance matrix.

**'ACF'** An average lag profile matrix

**'var'** ACF variance estimates

**'backgroundACF'** Average background ACF

**'backgroundvar'** Variance estimate for backgroundACF

**'lag'** Lag numbers

**'lag.us'** Time lags in us

**'range'** Range gates

**'range.km'** Range in km

**'nGates'** Number of range gates

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`readData.gdf` *readData.gdf*

---

### Description

Read gdf data files

### Usage

```
readData.gdf( files , n=1e6 , istart=1 , bigEndian=TRUE )
```

### Arguments

| | |
|---|---|
| `files` | A vector of data file names. The files are read in the same order in which they are in the vector. |
| `n` | Number of samples to read from each file, the vector is repeated as necessary to match its length with the "files" vector |
| `istart` | Indices from which to start the reading in each file, repeated as necesary to mathc its length with the "files" vector. |
| `bigEndian` | Logical telling whether the files are in big endian (default) or little endian format. |

## Value

A list with the following entries:

**'cdata'** A vector of complex data values.

**'idatar'** Lowest bit in real part of the data. Should contain the PPS bits in radar measurements.

**'idatai'** Lowest bit in imaginary part of the data. Should contain the TX bits in radar measurements.

**'ndata'** Total number of data points read. This is always the sum of the number of samples *intended* to be read from each file. If n is larger than file length for any of the input files, vectors cdata, idatar, and idatai will contain arbitrary values correspondingly.

**'success'** A logical indicating whether ndata points were successfully read ( success = TRUE ), or not ( success = FALSE ). e.g. any (n + istart - 1) larger than file length will lead to success=FALSE.

## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

## Examples

```
## Not run:

  # read all data from file "file1.gdf"
  d <- readData.gdf( files = "file1.gdf" , n = file.info( "file1.gdf" )$size / 4 )

  # read samples 1001 ... 10000 from "file1.gdf", samples 1 ... 10000 from
  # "file2.gdf", and samples 1 ... 200 from "file3.gdf".
  d <- readData.gdf( files = c( "file1.gdf" , "file2.gdf" , "file3.gdf" ) ,
                       n = c( 9000 , 10000, 200 ) ,
                  istart = c( 1001 , 1 , 1) )

## End(Not run)
```

---

readLPIdata.gdf *readLPIdata.gdf*

---

## Description

Read one integration period of voltage level data stored in gdf format.

## Usage

```
readLPIdata.gdf( LPIparam , intPeriod )
```

## Arguments

| | |
|---|---|
| LPIparam | An LPI parameter list from LPI.gdf |
| intPeriod | Integration period number. Integration periods are counted in steps of LPIparam$timeres.s, the period number 1 starting at LPIparam$beginTime. |

## Details

**'LPIparam contents'** Following components of the LPI parameter list are used for selecting the correct signal samples

**'startTime'** 'beginTime' converted into POSIX format, i.e. second count from 1970-01-01 00:00:00.

**'dataStartTimes'** A named vector with components 'RX1', 'RX2', 'TX1', and 'TX2'. Each element is samling time of the first sample of corresponding data type. The times are in seconds in POSIX format.

**'dataSampleFreqs'** A named vector with components 'RX1', 'RX2', 'TX1', and 'TX2'. Each element is the sample rate of the corresponding data type in Hz.

**'timeRes.s'** Analysis time resolution (incoherent integration period) in seconds.

**'dataFileLengths'** A named vector with components 'RX1', 'RX2', 'TX1', and 'TX2'. Each element is the number of complex samples in one data file of the corresponding data type.

**'fileNamePrefix'** A named vector with components 'RX1', 'RX2', 'TX1', and 'TX2'. Each element contains the file name prefix of the corresponding data type as a string.

## Value

A list with the following contents

| | |
|---|---|
| `'RX1'` | First receiver samples. |
| `'RX2'` | Second receiver samples. Will be identical with 'RX1' in autocovariance function estimation. |
| `'TX1'` | First transmitter samples. |
| `'TX2'` | Second transmitter samples. Will usually be identical with 'TX1', but may be different e.g. in orthogonal polarization experiments. |
| `success` | TRUE if all requested data was successfully read, FALSE otherwise. |

The elements "RX1", "RX2", "TX1", and "TX2" are lists themselves. Their elements are

| | |
|---|---|
| `'cdata'` | Complex sample vector |
| `'idata'` | Logical vector, TRUE if the sample should be used in LPI. |
| `'ndata'` | Number of samples in vectors cdata and idata |

## Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

---

`readLPIdir` *Read LPI result directory*

---

## Description

Read selected lag-profiles from all data files in LPI output directory.

## Usage

```
readLPIdir(dpath,recursive=FALSE,lags=all,stdThresh=NULL)
```

### Arguments

| | |
|---|---|
| `dpath` | Data file / directory path(s) as a vector of character strings. |
| `recursive` | If TRUE, all directories in dpath are searched recursively. |
| `lags` | Lag selection. Either a vector of lag numbers, or "all" to read all lags |
| `stdThresh` | Standard deviation threshold. Median variance at each range is calculated, and integration periods whose variance is larger than stdThresh times the median at any range gate are stripped off. |

### Value

A list with the following elements plus everything listed in the "ACF" list stored in the last data file.

**'ACF'** An nRange x nLag x nACF array of lag profile matrices

**'var'** ACF variance estimates

**'time.s'** Timestamps in seconds

**'timeString'** Timestamps as character strings.

**'nACF'** Number of integration periods.

**'nLag'** Number of lags.

**'nRange'** Number of range gates.

### Author(s)

Ilkka Virtanen (University of Oulu, Finland)
<ilkka.i.virtanen@oulu.fi>

# Index