Package 'Rfdbk'

August 29, 2016

Type Package

Title Handling NetCDF feedback files

Version 1.1.2

Date 2015-10-07

Maintainer Felix Fundel <felix.fundel@dwd.de>

Description Collection of functions to handle NetCDF feedback files from DWD data assimilation. To get examples running make sure the 'examplesRfdbk' directory exists in your home.

License GPL

Depends RNetCDF,data.table,parallel,stringr,survival,grid,verification,reshape2,pcaPP

R topics documented:

`afc` *Fast version of the 2AFC for continuous observations and forecasts The score is based on the rank correlation coefficient*

Description

Fast version of the 2AFC for continuous observations and forecasts The score is based on the rank correlation coefficient

Usage

```
afc(obsv, fcst)
```

Arguments

`obsv`	observation vector
`fcst`	forecast vector

Value

afc score

Author(s)

Felix <felix.fundel@dwd.de>

agg_det_scores *Aggregate deterministic scores*

Description

Aggregate deterministic scores

Usage

```
agg_det_scores(SCORENAME = NULL, RMSE = NULL, ME = NULL, MSE = NULL,
  SD = NULL, MAE = NULL, LEN = NULL)
```

Arguments

SCORENAME	score name string
RMSE	rmse scores of data subsets
ME	me scores of data subsets
MSE	mse scores of data subsets
SD	sd scores of data subsets
MAE	mae scores of data subsets
LEN	length of forecast-observation pairs in subsets

Value

pooled score value

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
x    = runif(1000)fnames   = system("ls ~/examplesRfdbk/icon/synop/*",intern=T)
y    = rnorm(1000)
x1   = x[1:10];x2=x[11:300];x3=x[301:1000]
y1   = y[1:10];y2=y[11:300];y3=y[301:1000]
rmse = function(x,y){return(sqrt(mean((x-y)^2)))}
rmse(x,y)
agg_det_scores("RMSE",RMSE=c(rmse(x1,y1),rmse(x2,y2),rmse(x3,y3)),LEN=c(length(x1),length
```

asSeason *Function to sort a given date to meteorological seasons (DJF, MAM, JJA, SON). Useful to stratify scores by season in order to plot scores for different seasons and compare them*

Description

Function to sort a given date to meteorological seasons (DJF, MAM, JJA, SON). Useful to stratify scores by season in order to plot scores for different seasons and compare them

Usage

```
asSeason(x)
```

Arguments

a	date in format yyyymmdd (at least). hours and/or minutes and/or seconds can be specified in format yyyymmddHHMMSS. Can be given as a string or numeric.

Value

a string corresponding to the four seasons (DJF, MAM, JJA, SON)

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 simple examples with one date
asSeason("20150201") returns "DJF"
asSeason(20150201) also returns "DJF"
asSeason("201502011234") also returns "DJF"
asSeason("151201") returns an error (format yymmdd not accepted)
#EXAMPLE 2 Example of how to use this function to stratify scores by season
and show a plot of comparison fot different seasons

require(ggplot2)
fnames       = "/Users/josuegehring/Desktop/verTEMP.2014120112"
cond         = list(obs="!is.na(obs)",varno="varno%in%c(2,3,4,29)",ident="ident%in%c(6610)
columnnames = c("obs","veri_data","varno","state","level","veri_initial_date","ident")
DT           = fdbk_dt_multi_large(fnames,cond,columnnames,1)
levels = c(100000,92500,85000,70000,60000,50000,40000,30000)
DT = fdbk_dt_binning_level(DT,"level",levels,includeAll=TRUE)
DT$varno     = varno_to_name(DT$varno)
DT$season = as.character(lapply(DT$veri_initial_date, asSeason))
DT = na.omit(DT)
strat        = c("season","level")
scores       = fdbk_dt_verif_continuous(DT,strat)
scores       = scores[!is.na(scores),]
ii = scores$scorename=="ME"
scores = scores[ii]
data = data.frame(scores$level,scores$scores,scores$season)
colnames(data) = c("level","scores","season")
```

```
data = data[order(data$level),]
p =  ggplot(data,aes(x=scores,y=level,group=season,colour=season))+
  geom_point()+geom_path() +
  theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))+scale_y_reverse()
print(p)
```

`comparableRows` *Find comparable rows in DT for two or more attributes*

Description

Find comparable rows in DT for two or more attributes

Usage

```
comparableRows(DT, splitCol, splitVal, compareBy)
```

Arguments

`DT`	data.table
`splitCol`	Dt column name that contains the attributes that should be compared
`splitVal`	two or more values of splitCol that should be compared
`compareBy`	other column names that should be used two decide if a comparable row exists for both splitVals

Value

indices of DT that show which rows should be retained (TRUE) i.e. rows that have a counterpart in each of the two splitVals

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
## Delete rows in DT that have no counterpart for GME/ICON concerning the attributes: "ve

require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.2014*",intern=T)
cond                  = list(varno="varno%in%c(3,29)",veri_description="grepl('forecast',
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time","ve
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
keepind               = comparableRows(DT,splitCol="veri_model",splitVal=unique(DT$veri_m
DT                    = DT[keepind]
DT[,.N,by=c("varno","veri_model")]
DT$varno              = varno_to_name(DT$varno)
strat                 = c("varno","veri_forecast_time","veri_model")
scores                = fdbk_dt_verif_continuous(DT,strat)
p =  ggplot(scores,aes(x=veri_forecast_time,y=scores,group=interaction(scorename,varno,ve
     geom_line(size=.7) + geom_point(size=1.5) + facet_wrap(~scorename, scales = "free")+
     theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))
p
```

`fdbk_dt` *Fdbk file content (as obtained from read_fdbk(_f)) is converted into a data.table. Therefore a lot of data overhead is created as most data will be duplicated. However, data.tables offer a lot of extra functionality.*

Description

Fdbk file content (as obtained from read_fdbk(_f)) is converted into a data.table. Therefore a lot of data overhead is created as most data will be duplicated. However, data.tables offer a lot of extra functionality.

Usage

```
fdbk_dt(fdbk)
```

Arguments

`fdbk`	output from read_fdbk

Value

a data.table of the feedback file data section

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fdbk = read_fdbk("~/examplesRfdbk/icon/synop/verSYNOP.2014120112")
format(object.size(fdbk),"Mb")
DT   = fdbk_dt(fdbk)
format(object.size(DT),"Mb")
DT
```

`fdbk_dt_add_obs_ini` *Update a feedback file data.table with observations valid at initialization (helpful for calculation of tendency correlations or persistence scores)*

Description

Update a feedback file data.table with observations valid at initialization (helpful for calculation of tendency correlations or persistence scores)

Usage

```
fdbk_dt_add_obs_ini(DT, fnamepast, cond = cond)
```

Arguments

DT	data.table with feedback file content, minimum requires "obs","level","varno","lon","lat" and "veri_initial_date" as YYYYmmddHHMM numeric and a column called "lonlat":=paste0(lon,lat)
fnamepast	vector of filenames (including path) of feedback files that should be valid at times needed to fill DT (e.g. files of past 7 days to fill DT for a model of 7 day forecast range)
cond	list of conditions used for loading DT

Value

DT with an additional columns "obs_ini"

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fdbkDir   = "~/examplesRfdbk/icon/synop"
fileName  = tail(dir(fdbkDir,full.names=T),1)
vars      = c("obs","veri_data","veri_forecast_time","level","varno","lon","lat","veri_in
cond      = ""
DT        = fdbk_dt_multi_large(fileName, condition=cond, vars=vars, cores=1)
DT[,lonlat:=paste0(lon,lat)]
fileNames = tail(dir(fdbkDir,full.names=T),10)
DT = fdbk_dt_add_obs_ini(DT,fileNames,cond)
DT[,lonlat:=NULL]
Plot correlation between observations for different lead-times
na.omit(DT[,list(cor=cor(obs,obs_ini,use="pairwise.complete.obs")),by=c("veri_forecast_ti
```

---

fdbk_dt_binning *Bin a data.table column into user defined bins and replace it with the bin center value. If breaks can be provided (e.g. no gaps between bins) try to use 'cut' instead.*

Description

Bin a data.table column into user defined bins and replace it with the bin center value. If breaks can be provided (e.g. no gaps between bins) try to use 'cut' instead.

Usage

```
fdbk_dt_binning(DT, varToBin = "level", binLower, binUpper)
```

Arguments

DT	data.table
varToBin	variable that should be binned (and will be replaced by the binned version)
binLower	number/vector lower bins limits
binUpper	number/vector upper bins limits

Value

data.table with varToBin replaced by factorized mid-bin values (NA if variable falls in none of the bins)

Author(s)

Felix <felix.fundel@dwd.de>

See Also

cut

Examples

```
#plot scores accross binned levels
require(ggplot2)
fnames       = "~/examplesRfdbk/icon/temp/verTEMP.2014120112"
cond         = list(obs="!is.na(obs)",varno="varno%in%c(2,3,4,29)")
columnnames = c("obs","veri_data","varno","state","level")
DT           = fdbk_dt_multi_large(fnames,cond,columnnames,1)
binUpper     = seq(100000,1000,by=-5000)+1500
binLower     = seq(100000,1000,by=-5000)-1500
DT           = fdbk_dt_binning(DT,"level",binLower,binUpper)
DT           = DT[!is.na(level),,]
DT$varno     = varno_to_name(DT$varno)
strat        = c("varno","level")
scores       = fdbk_dt_verif_continuous(DT,strat)
setkey(scores,scorename,varno,level)
scores       = scores[!is.na(scores),]
p =  ggplot(scores,aes(x=scores,y=level,group=interaction(varno,scorename)))+
 geom_path() + facet_wrap(~varno~scorename,scales="free_x",ncol = 6)+
 theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))+scale_y_reverse()
p
```

---

fdbk_dt_binning_level

Bin a data.table column around user defined levels and replace it with the levels value.

Description

Other way to perform a binning like in function fdbk_dt_binning but by defining levels around which to bin instead of the bins limits. The limits of the bins will be calculated by taking the mean between neighbouring levels. The two functions differ in the sense that fdbk_dt_binning allow to have gaps between the bins, whereas the bins will be continuous in fdbk_dt_binning_level. This function allows to have non-equally spaced levels without gaps between the bins, so that the level is not always at the center of the bin.

Usage

```
fdbk_dt_binning_level(DT, varToBin = "level", levels, includeAll = FALSE)
```

Arguments

DT	data.table
varToBin	variable that should be binned (and will be replaced by the binned version)
levels	number/vector of levels on which the bins will be defined
Logical	to include data that are out of the bins defined by levels. If set to FALSE (default), data that falls out of the bins are dicarded. If set to true, the numerically lower and upper limits will be set to -Inf and +Inf, respectively. This allows to keep data that falls out of the bins.

Value

data.table with varToBin replaced by factorized mid-bin values (NA if variable falls in none of the bins)

Author(s)

Felix <felix.fundel@dwd.de>

See Also

cut

Examples

```
#plot scores accross binned levels
require(ggplot2)
fnames       = "~/examplesRfdbk/icon/temp/verTEMP.2014120112"
cond         = list(obs="!is.na(obs)",varno="varno%in%c(2,3,4,29)")
columnnames = c("obs","veri_data","varno","state","level")
DT           = fdbk_dt_multi_large(fnames,cond,columnnames,1)
levels = c(100000,92500,85000,70000,60000,50000,40000,30000)
DT = fdbk_dt_binning_level(DT,"level",levels)
DT$varno     = varno_to_name(DT$varno)
strat        = c("varno","level")
scores       = fdbk_dt_verif_continuous(DT,strat)
setkey(scores,scorename,varno,level)
scores       = scores[!is.na(scores),]
p =  ggplot(scores,aes(x=scores,y=level,group=interaction(varno,scorename)))+
 geom_path() + facet_wrap(~varno~scorename,scales="free_x",ncol = 6)+
 theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))+scale_y_reverse()
p
```

---

fdbk_dt_brier — *Calculate the brier score (and decomposition and skill score) for one threshold per variable*

Description

Calculate the brier score (and decomposition and skill score) for one threshold per variable

Usage

```
fdbk_dt_brier(DT, thresholds = "", by = "")
```

Arguments

DT	data.table (columns 'veri_ens_member','obs' and 'veri_data' plus all variables to make forecasts distinguishable are required!!!)
thresholds	list of threshold for variable names in DT (if "" uses obs median)
by	stratify crps by (e.g. c('varno','veri_forecast_time'))

Value

data.table with columns as defined in 'by' plus scorename plus score

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
require(ggplot2)
fnames    = system("/bin/ls ~/examplesRfdbk/eps/*12/verSYNOP*",intern=T)
condition = list(veri_description="grepl('member',veri_description)",
                 state="state%in%c(0,1)",
                 statid="!is.na(as.numeric(statid)) & !duplicated(statid)",
                 veri_forecast_time="veri_forecast_time>=1200")
columns   = c("obs","veri_data","varno","veri_ens_member","veri_forecast_time","statid","
DT        = fdbk_dt_multi_large(fnames,condition,columns,5)
PROBS     = fdbk_dt_brier(DT,by=c("varno","veri_forecast_time"))

ggplot(PROBS,aes(x=veri_forecast_time,y=score,color=scorename,group=scorename))+
  geom_line()+geom_point()+facet_wrap(~varno,scale="free_y",ncol=2)+
  theme_bw()+scale_colour_discrete("lead-time")
```

---

fdbk_dt_conditional

Function for conditional filtering of data.tables Helps to filter rows from data.tables with condition for the same column. A typical application would be to filter wind direction observations whenever wind speed is <3m/s In this case one would have to delete all varno==111 for all matching varn===112 that have obs less than 3m/s. This is complex, as observations of varno 111 and 112 are in different rows. Even more complicated fitering task, e.g. filtering a variable on more than one other variable, might require to call this function sepatately for each task.

Description

Function for conditional filtering of data.tables Helps to filter rows from data.tables with condition for the same column. A typical application would be to filter wind direction observations whenever wind speed is <3m/s In this case one would have to delete all varno==111 for all matching varn===112 that have obs less than 3m/s. This is complex, as observations of varno 111 and 112 are in different rows. Even more complicated fitering task, e.g. filtering a variable on more than one other variable, might require to call this function sepatately for each task.

Usage

```
fdbk_dt_conditional(DT, condition, on, by)
```

Arguments

DT	the data.table
condition	logical condition as character string
on	logical condition(s) as character string
by	character vector of all column names that sould be used for the grouping

Value

the fitered data.table (column and row order is not preserved!)

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
# Data table with 3 different variables, random obseervations at each location
DT = data.table(varno=c(1,1,1,2,2,3),obs=c(1,2,3,0,1,4),lat=c(10,20,30,10,20,10),lon=c(5,
Remove all variables 1 where for variable 2 obs>0. If variable 1 has no observation of va
fdbk_dt_conditional(DT,condition="varno==1",on="varno==2 & obs>0",by=c("lon","lat"))
```

---

fdbk_dt_contscores *Calculates most common contingeny scores*

Description

Calculates most common contingeny scores

Usage

```
fdbk_dt_contscores(CONTTABLE, by, meltTable = T)
```

Arguments

CONTTABLE	data.table with colums hit,miss,corrneg,false and additional columns (output of fdbk_dt_conttable(_2thrs))
by	stratify contingency entries by these columns
meltTable	if TRUE (default) melt output so that there is one 'scores' column and one 'scorename' column

Value

data.table with one column of score names and one column of scores values

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)
cond                  = list(veri_description="grepl('forecast',veri_description)",
                             veri_forecast_time="veri_forecast_time%in%c(2400,4800,7200,9
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time","st
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
thrs                  = list('29'=list('lower'=c(.8,.6),'upper'=c(Inf,.9)),
                              '3'=list('lower'=c(-5,0,5),'upper'=c(Inf,Inf,Inf)))
CONTTABLE             = fdbk_dt_conttable_2thrs(DT,thrs,by=c("veri_model","veri_forecast_
SCORES                = fdbk_dt_contscores(CONTTABLE,by=c("veri_model","veri_forecast_tim
ggplot(SCORES,aes(x=veri_forecast_time,y=scores,color=thr,linetype=veri_model))+
geom_line()+
geom_point()+
facet_grid(scorename~varno,scale="free_y")+
theme_bw()
```

---

fdbk_dt_conttable *Calculates stratified contingency table entries (above threshold) for a data table*

Description

Calculates stratified contingency table entries (above threshold) for a data table

Usage

```
fdbk_dt_conttable(DT, vars = NULL, thrs = NULL, by = NULL, cores = 1)
```

Arguments

DT	data.table with relevant information
vars	character vector of varnos (if NULL take from DT)
thrs	list of vectors of thresholds for each varno (if NULL threshold are generated from quantiles)
by	stratify contingency entries by these DT columns
cores	number of CPU cores to split the calculation (helps for larger data tables)

Value

data.table with columns varno,thr, hits,false,miss,corrneg and the arguments of 'by'

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE (CSI for quantile thresholds)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)
cond                  = list(veri_description="grepl('forecast',veri_description)",
                             veri_forecast_time="veri_forecast_time%in%c(2400,4800,7200,9
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time","st
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
vars                  = c('1','3','4','29')
thrs                  = list('1'=c(50,60),'3'=c(-5,0,5),'4'=c(-5,0,5),'29'=c(.4,.6,.8))
xx                    = fdbk_dt_conttable(DT,vars=vars,thrs=thrs,by=c("veri_model","veri_
CSI                   = xx[,list(csi =(hit)/(hit + miss + false) ),by=c("veri_forecast_ti
CSI[,varno:=varno_to_name(varno,T)]
ggplot(CSI,aes(x=thr,y=csi,color=factor(veri_forecast_time),linetype=factor(veri_model),g
geom_line()+
ggtitle("CSI")+
facet_wrap(~varno,scales="free_x")
```

---

fdbk_dt_conttable_2thrs

Calculates stratified contingency table entries (above or between thresholds) for a data table

---

Description

Calculates stratified contingency table entries (above or between thresholds) for a data table

Usage

```
fdbk_dt_conttable_2thrs(DT, thrs, by, cores = 1, incores = 1)
```

Arguments

DT	data.table with relevant information (at least varno, obs and veri_data)
thrs	list of variable having each a list of lower/upper thresholds (set upper to Inf if only one threshold is required)
by	stratify contingency entries by these DT columns
cores	computing cores for the outer loop (splits computation by varnos)
incores	computing cores for the outer loop (splits computation by thresholds)(available cores have to be of number cores x incores)

Value

data.table with columns varno,thr, hits,false,miss,corrneg and the arguments of 'by'

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE (CSI for quantile thresholds)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)
cond                  = list(veri_description="grepl('forecast',veri_description)",
                             veri_forecast_time="veri_forecast_time%in%c(2400,4800,7200,9
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time","st
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
thrs                  = list('29'=list('lower'=c(.5,.8),'upper'=c(Inf,.9)),
                              '3'=list('lower'=c(-5,0,5),'upper'=c(Inf,Inf,Inf)))
xx                    = fdbk_dt_conttable_2thrs(DT,thrs,by=c("veri_model","veri_forecast_
CSI                   = xx[,list(csi =(hit)/(hit + miss + false) ),by=c("veri_forecast_ti
CSI[,varno:=varno_to_name(varno,T)]
ggplot(CSI,aes(x=veri_forecast_time,y=csi,group=interaction(veri_model,thr),linetype=veri
geom_line()+
facet_grid(~varno)+
ggtitle("CSI")
```

`fdbk_dt_crps` *Calculate CRPS(crps, crpsPot,Reli) from data.table applied on selected parts of the table (Caution, double check results! DT sorting might be modified!)*

Description

Calculate CRPS(crps, crpsPot,Reli) from data.table applied on selected parts of the table (Caution, double check results! DT sorting might be modified!)

Usage

```
fdbk_dt_crps(DT, by)
```

Arguments

`DT`	data.table (columns 'veri_ens_member','obs' and 'veri_data' plus all variables to make forecasts distinguishable are required!!!)
`by`	stratify crps by (e.g. 'varno')

Value

data.table with columns as defined in 'by' plus scorename plus score

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 (CRPS for each varno)
fnames                = system("ls ~/examplesRfdbk/talagrand/*SYNOP*",intern=T)
cond                  = list(veri_description="grepl('first guess ensemble member',veri_d
                             obs="!is.na(obs)",
                             statid="!is.na(as.numeric(statid)) & !duplicated(statid)",
                             veri_forecast_time="veri_forecast_time==100",
                             state="state%in%c(0,1,5)")
columnnames           = c("veri_data","varno","obs","veri_ens_member","veri_initial_date"
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,10)
DT[,varno:=varno_to_name(varno)]
fdbk_dt_crps(DT,by="varno")

#EXAMPLE 2 (CRPS decomosition for forecasts at SYNOP stations)
require(ggplot2)
fnames    = system("/bin/ls ~/examplesRfdbk/eps/*12/verSYNOP*",intern=T)
condition = list(veri_description="grepl('member',veri_description)",
                 state="state%in%c(0,1)",
                 statid="round(as.numeric(statid)/1000)==10 & !duplicated(statid)",
                 veri_forecast_time="veri_forecast_time>=1200")
columns   = c("obs","veri_data","varno","veri_ens_member","veri_forecast_time","statid","
DT        = fdbk_dt_multi_large(fnames,condition,columns,5)
CRPS = fdbk_dt_crps(DT,by=c("varno","veri_forecast_time"))
CRPS[,varno:=varno_to_name(varno,F)]
ggplot(CRPS,aes(x=veri_forecast_time,y=score))+geom_line()+geom_point()+facet_grid(~varno

#EXAMPLE 3 (slow...)(CRPS decomosition for european forecasts at TEMP stations)
require(ggplot2)
fnames    = system("/bin/ls ~/examplesRfdbk/eps/*12/verTEMP*",intern=T)
condition = list(veri_description="grepl('member',veri_description)",
                 state="state%in%c(0,1)",
                 level="level%in%c(100000,92500,85000,75000,70000,50000,40000,30000,25000
                 statid="round(as.numeric(statid)/1000)<=10 & !duplicated(statid)",
                 veri_forecast_time="veri_forecast_time>=1200",
                 varno="varno!=1")
columns   = c("obs","veri_data","varno","level","veri_ens_member","veri_forecast_time","v
DT        = fdbk_dt_multi_large(fnames,condition,columns,5)
CRPS      = fdbk_dt_crps(DT,by=c("varno","level","veri_forecast_time"))
CRPS[,varno:=varno_to_name(varno,F)]
ggplot(CRPS,aes(x=score,y=level,color=factor(veri_forecast_time),group=veri_forecast_time
 geom_path()+facet_wrap(~varno~scorename,scale="free_x",ncol=3)+
 scale_y_reverse()+theme_bw()+scale_colour_discrete("lead-time")
```

`fdbk_dt_crps_norm` *Calculate CRPS and Ignorance score from data.table with EPS mean/spread, assuming a normally distributed EPS In case of zero standard deviation CRPS and Ignorance would return NA, those cases are omitted in this function so that a score should always be returned, except all ensemble predictions have zero standard deviation.*

Description

Calculate CRPS and Ignorance score from data.table with EPS mean/spread, assuming a normally distributed EPS In case of zero standard deviation CRPS and Ignorance would return NA, those

cases are omitted in this function so that a score should always be returned, except all ensemble predictions have zero standard deviation.

Usage

```
fdbk_dt_crps_norm(DT, by)
```

Arguments

DT	data.table (columns 'veri_description','obs' and 'veri_data' are required!!!) values of veri_description have to be "mean" or "spread"
by	stratify crps by (e.g. 'varno')

Value

data.table with columns as defined in 'by' plus scorename plus score

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
require(ggplot2)
fnames      = system("/bin/ls ~/examplesRfdbk/eps/*12/verTEMP*",intern=T)
condition   = list(
                           veri_description="grepl('first guess',veri_description)",
                           veri_description="grepl('ensemble',veri_description)",
                           state="state%in%c(0,1)",
                           level="level%in%c(100000,92500,85000,75000,70000,50000,40000,3
                           veri_forecast_time="veri_forecast_time>=1200",
                           varno="varno!=1")
vars        = c("obs","veri_data","varno","level","veri_description","veri_forecast_time"
DT          = fdbk_dt_multi_large(fnames,condition,vars,5)
DT[grepl("mean",veri_description),veri_description:="mean"]
DT[grepl("spread",veri_description),veri_description:="spread"]
by=c("varno","level","veri_forecast_time")
CRPS = fdbk_dt_crps_norm(DT,by)
CRPS[,varno:=varno_to_name(varno,F)]
CRPS[scorename=="IGN" & score>10000,score:=NA]
ggplot(CRPS,aes(x=score,y=level,color=factor(veri_forecast_time),group=veri_forecast_time
  geom_path()+geom_point()+facet_wrap(~scorename~varno,scale="free_x",ncol=4)+
  scale_y_reverse()+theme_bw()+scale_colour_discrete("lead-time")
```

---

fdbk_dt_hits_uncert

Calculates stratified hit rates for uncertain obs/fcst

Description

Calculates stratified hit rates for uncertain obs/fcst

Usage

```
fdbk_dt_hits_uncert(DT, thrs, by, cores = 1, incores = 1)
```

Arguments

DT	data.table with relevant information (at least varno, obs and veri_data)
thrs	list of variable having each a list of lower/upper limit, relative to observation
by	stratify contingency entries by these DT columns
cores	computing cores for the outer loop (splits computation by varnos)
incores	computing cores for the outer loop (splits computation by thresholds)(available cores have to be of number cores x incores)

Value

data.table with columns varno,interval, hits, total and the arguments of 'by'

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE (CSI for quantile thresholds)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)
cond                  = list(veri_description="grepl('forecast',veri_description)",
                             veri_forecast_time="veri_forecast_time%in%c(2400,4800,7200,9
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time","st
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
thrs                  = list('29'=list('lower'=c(-1/6),'upper'=c(1/6)),
                              '3'=list('lower'=c(-1,-2),'upper'=c(1,2)))
xx                    = fdbk_dt_hits_uncert(DT,thrs,by=c("veri_model","veri_forecast_time
PEC                   = xx[,list(PEC =(hit)/(total) ),by=c("veri_forecast_time","veri_mod
PEC[,varno:=varno_to_name(varno,T)]
ggplot(PEC,aes(x=veri_forecast_time,y=PEC,group=interaction(veri_model,interval),linetype
geom_line()+
geom_point()+
facet_grid(~varno)+
theme_bw()+
ggtitle("Percent Correct (within interval)")
```

---

fdbk_dt_interpolate

Bin a data.table column into user defined bins and replace it with the bin center value. If breaks can be provided (e.g. no gaps between bins) try to use 'cut' instead.

Description

Bin a data.table column into user defined bins and replace it with the bin center value. If breaks can be provided (e.g. no gaps between bins) try to use 'cut' instead.

Usage

```
fdbk_dt_interpolate(DT, varToInter = c("obs", "veri_data"),
  levelToInter = "plevel", interLevels = levels, varno = "varno")
```

Arguments

DT	data.table
varToBin	variable that should be binned (and will be replaced by the binned version)
mode	that will be used to defined the bin. Choices are "bin" or "level". In the first case the limtis of the bins have to be explicitly given in two vectors. The name given to the corresponding levels of the bin will be the mean of the lower and upper limit of the bin. In the second case a vector specifying the levels has to be given. The limits of the bins will be calculated by taking the mean between neighbouring levels. The two methods differ in the sense that the "bin" mode allow to have gaps between the bins, whereas the bins will be continuous in "level" mode. The "level" mode allow to have non-equally spaced levels without gaps between the bins, so that the level is not always at the center of the bin.
binLower	number/vector lower bins limits
binUpper	number/vector upper bins limits
levels	number/vector of levels on which the bins will be defined

Value

data.table with varToBin replaced by factorized mid-bin values (NA if variable falls in none of the bins)

Author(s)

Josue <josue.gehring@meteoswiss.ch>

Examples

```
# Example of linear interpolation based on an international standard atmosphere profile
require(ggplot2)
require(Rfdbk)
require(reshape2)
a1 = -6.5 # K/km standard atmosphere lapse rate, represents observations
a2 = -9 # K/km lapse rate obtained from a fictive model output
b1 = 288.15 # K standard atmosphere surface temperature
b2 = 295 # K surface temperature obtained from a fictive model output
Ho = 8.4 # km scale height
po = 1013.25 # standard atmosphere pressure in hPa
p = seq(250,1000,10) # pressure until the tropopause
T1 = a1*Ho*log(po/p)+b1 # Standard amtmosphere temperature profile
T2 =  a2*Ho*log(po/p)+b2 # Model output temperature profile
Bias = T2-T1 # Bias = forecast - observation

# Build a data table in feedback files format
obs = T1
veri_data = T2
veri_forecast_time = 24
veri_initial_date = 2015110900
time = -720
```

```
lat = 46.812
lon = 6.943
varno = 2
veri_model = "COSMO"
plevel = p
ident = 6610
levels = c(1000, 975, 950, 925, 900, 875, 850, 800, 750, 700, 650, 600, 550, 500, 450, 40
DT = data.frame(obs,veri_data,veri_forecast_time,veri_initial_date,time,lat,lon,varno,ver
DT           = fdbk_dt_interpolate(DT,varToInter=c("obs","veri_data"), levelToInter = "pl

data1 = melt(data.frame(T1,p),id="T1") # Data for the standard atmosphere temperature pro
data2 = melt(data.frame(T2=DT$obs,DT$plevel),id="T2") # Interpolation of data1


plot = ggplot() + geom_point(data=data1,aes(x=T1,y=value,colour=variable)) + geom_point(c
  xlab("T [K]") + ylab("pressure [hPa]")+  scale_colour_manual(name="Temperature",values=
print(plot) # plot of the Standard atmosphere profile and its interpolation

allscores = fdbk_dt_verif_continuous(DT,strat=c("varno","veri_model","plevel") ) # Data t

data3 = melt(data.frame(Bias,p),id="Bias") # Bias calculated directly from the standard a
ME = allscores[allscores$scorename=="ME"]$scores # scores calculated with fdbk_dt_verif_c
ME_levels = allscores[allscores$scorename=="ME"]$plevel # interpolation levels
data4 = melt(data.frame(ME,ME_levels),id="ME")
plot2 = ggplot() + geom_point(data=data3,aes(x=Bias,y=value,colour=variable)) + geom_poin
  xlab("T bias [K]") + ylab("pressure [hPa]")+scale_colour_manual(name="Bias",values=c("r
print(plot2) # plot of the bias calculated directly from the profiles and the bias from t
```

`fdbk_dt_multi` *Load relevant information of many feedback files as data.table Be restrictive with the columns kept in the data.table as otherwise the memory limit is reached fast To speed up computation multiple cores are utilized (if possible)*

Description

Load relevant information of many feedback files as data.table Be restrictive with the columns kept in the data.table as otherwise the memory limit is reached fast To speed up computation multiple cores are utilized (if possible)

Usage

```
fdbk_dt_multi(fnames, cond = "", columnnames = "", cores = 1)
```

Arguments

`fnames`	vector of feedback filename(s)
`cond`	string of conditions the fdbk file will be filtered for in advance
`columnnames`	attribute names to keep in the data table

Value

a data.table of merged feedback file contents

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fnames      = system("ls ~/examplesRfdbk/icon/synop/verSYNOP.*",intern=T)
cond        = "varno%in%c(3,4) & !is.na(obs)"
columnnames = c("obs","veri_data","varno","veri_forecast_time")
DT          = fdbk_dt_multi(fnames,cond,columnnames,4)
DT
```

`fdbk_dt_multi_large`

Function to load one or many fdbk Files and transform them to a data.table. Faster than fdbk_dt_multi and able to handle very large files, however, be as restrictive as possible, use the cond/columnnames argument select only the data you need for your problem. Note: Using conditions on veri_data in the cond argument is not possible and may cause an error!!! Solution: filter veri_data in the returned data.table

---

Description

Function to load one or many fdbk Files and transform them to a data.table. Faster than fdbk_dt_multi and able to handle very large files, however, be as restrictive as possible, use the cond/columnnames argument select only the data you need for your problem. Note: Using conditions on veri_data in the cond argument is not possible and may cause an error!!! Solution: filter veri_data in the returned data.table

Usage

```
fdbk_dt_multi_large(fnames, condition = "", vars = "", cores = 1)
```

Arguments

`fnames`	vector of feedback filename(s)
`cores`	use multiple cores for parallel file loading
`cond`	list of strings of conditions (all of the list entries are connected with the "&" operator!)
`columnnames`	attribute names to keep in the data table

Value

a data.table of merged feedback file contents

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 (1x1 deg.) bias of satellite data (channel 921 from METOP-1)
require(ggplot2)
fnames      = system("/bin/ls ~/examplesRfdbk/example_monRad/monRAD_*.nc",intern=T)
condition   = list(obs="!is.na(obs)",
                   level="level%in%c(921)",
                   statid="statid=='METOP-1   '",
                   veri_forecast_time="veri_forecast_time==0",
                   veri_run_type="veri_run_type==3",
                   veri_ens_member="veri_ens_member==-1")
columnnames = c("obs","veri_data","lon","lat","veri_initial_date")
DT          = fdbk_dt_multi_large(fnames,condition,columnnames,cores=1)
DT
DT[,lon:=round(lon)]
DT[,lat:=round(lat)]
scores = DT[,list(ME=mean(obs-veri_data)),by=c("lon","lat")]
outlines = as.data.table(map("world", plot = FALSE)[c("x","y")])
worldmap = geom_path(aes(x, y), inherit.aes = FALSE, data = outlines, alpha = 0.8, show_g
p = ggplot(scores,aes(x=lon,y=lat,fill=cut(ME,seq(-100,100,20))))+geom_raster()+
    scale_fill_manual("ME",values=tim.colors(10),drop = FALSE)+
    worldmap
p

 #EXAMPLE 2 TEMP EPS plot for one station on reversed-log-y scale
require(ggplot2)
require(scales)
fname="~/examplesRfdbk/eps/2013111112/verTEMP.nc"
condition           = list(veri_description="grepl('first guess vv',veri_description)",
                           veri_description="grepl('member',veri_description)",
                           state="state%in%c(0,1)",
                           statid="statid=='01028     '")
columns             = c("obs","veri_data","varno","level","veri_description","veri_foreca
DT                  = fdbk_dt_multi_large(fname,condition,columns,1)
DT$veri_description = as.numeric(substr(DT$veri_description,29,32))
setnames(DT,"veri_description","member")
DT[,varno:=varno_to_name(varno,F)]
reverselog_trans <- function(base = exp(1)) {
     trans <- function(x) -log(x, base)
     inv <- function(x) base^(-x)
     trans_new(paste0("reverselog-", format(base)), trans, inv,
               log_breaks(base = base),
               domain = c(1e-100, Inf))
}

# plot only even members for clearness+ obs as black line
ggplot(DT[DT$member%%2==0,],aes(x=veri_data,y=level,color=factor(member)))+geom_path()+ge
      scale_y_continuous(trans=reverselog_trans(10))+
      geom_point(data =DT[member==1], aes(x=obs,y=level), colour = "black")+
      geom_path(data =DT[member==1], aes(x=obs,y=level), colour = "black")+
      ggtitle(paste("EPS TEMP for station",unique(DT$statid)))

 #EXAMPLE 3 SATELLITE RADIATION plot verification scores as function of channel and stael
require(ggplot2)
fnames      = system("ls ~/examplesRfdbk/example_monRad/monRAD_*.nc",intern=T)
condition   = list(obs="!is.na(obs)",
                   level="level>100 & level<6000",
```

```
                    veri_forecast_time="veri_forecast_time==0",
                    veri_run_type="veri_run_type==3",
                    veri_ens_member="veri_ens_member==-1")
DT         = fdbk_dt_multi_large(fnames,condition,c("obs","veri_data","level","statid"),1
scores     = fdbk_dt_verif_continuous(DT,c("level","statid"))
ggplot(scores,aes(x=level,y=scores,color=statid,group=statid))+geom_line()+geom_point()+f
```

`fdbk_dt_reliability_diagram`
Calculate the reliability diagram statistics

Description

Calculate the reliability diagram statistics

Usage

```
fdbk_dt_reliability_diagram(DT, thresholds = "", by = "", breaks = "")
```

Arguments

`DT`	data.table (columns 'veri_ens_member','obs' and 'veri_data' plus all variables to make forecasts distinguishable are required!!!)
`thresholds`	list of threshold for variable names in DT (if "" uses obs median)
`by`	stratify crps by (e.g. c('varno','veri_forecast_time'))
`breaks`	breaks used to bin the forecast probabilities

Value

data.table with columns forecast bin and observed frequency for each varno/threshold

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
require(ggplot2)
fnames    = system("/bin/ls ~/examplesRfdbk/eps/*12/verSYNOP*",intern=T)
condition = list(veri_description="grepl('member',veri_description)",
                 state="state%in%c(0,1)",
                 statid="!is.na(as.numeric(statid)) & !duplicated(statid)",
                 veri_forecast_time="veri_forecast_time>=1200")
columns   = c("obs","veri_data","varno","veri_ens_member","veri_forecast_time","statid","
DT        = fdbk_dt_multi_large(fnames,condition,columns,5)
ATTR      = fdbk_dt_reliability_diagram(DT,thresholds="",by=c("varno","veri_forecast_time
ggplot(ATTR,aes(x=fbin,y=obin,color=factor(veri_forecast_time),group=veri_forecast_time))
```

fdbk_dt_uv2drc *Calculate wind direction from u and v wind components in a data.table*

Description

Calculate wind direction from u and v wind components in a data.table

Usage

```
fdbk_dt_uv2drc(DATATABLE, col = c("obs", "veri_data"))
```

Arguments

DATATABLE	data table containing the columns "varno" with elements 3 and 4, and e.g. "obs", "obs_ini", "veri_data" or combinations of it
fcst	forecast vector

Value

data.table with same columns as DATATABLE and varno=111

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fnames   = system("ls ~/examplesRfdbk/icon/synop/*",intern=T)[1:5]
cond     = list(obs                = "!is.na(obs)",
                veri_run_class     = "veri_run_class%in%c(0,2)",
                veri_run_type      = "veri_run_type%in%c(0,4)",
                state              = "state%in%c(0,1,5)",
                statid             = "!is.na(as.numeric(statid))",
                statid             = "!duplicated(statid)",
                varno              = "varno%in%c(3,4)")
colnames  = c("obs","veri_data","veri_forecast_time","veri_initial_date","lat","lon","var
DT        = fdbk_dt_multi_large(fnames,cond,colnames,cores=5)
DRC       = fdbk_dt_uv2drc(DT)
.rbind.data.table(DT,DRC)
```

fdbk_dt_uv2spd *Calculate wind speed from u and v wind components in a data.table*

Description

Calculate wind speed from u and v wind components in a data.table

Usage

```
fdbk_dt_uv2spd(DATATABLE, col = c("obs", "veri_data"))
```

Arguments

DATATABLE	data table containing the columns "varno" with elements 3 and 4, and e.g. "obs", "obs_ini", "veri_data" or combinations of it
fcst	forecast vector

Value

data.table with same columns as DATATABLE and varno=112

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fnames   = system("ls ~/examplesRfdbk/icon/synop/*",intern=T)[1:5]
cond     = list(obs                = "!is.na(obs)",
                veri_run_class     = "veri_run_class%in%c(0,2)",
                veri_run_type      = "veri_run_type%in%c(0,4)",
                state              = "state%in%c(0,1,5)",
                statid             = "!is.na(as.numeric(statid))",
                statid             = "!duplicated(statid)",
                varno              = "varno%in%c(3,4)")
colnames  = c("obs","veri_data","veri_forecast_time","veri_initial_date","lat","lon","var
DT        = fdbk_dt_multi_large(fnames,cond,colnames,cores=5)
SPD       = fdbk_dt_uv2spd(DT)
.rbind.data.table(DT,SPD)
```

---

fdbk_dt_verif_continuous

Deterministic scores for data.tables from feedback files, returns 5-95 confidence intervals if needed.

Description

Function returns a score data.table with ME,MAE,RMSE,SD,R2 and length of verification data pairs Additionaly 5th and 95th confidence interval from bootstrap resampling can be returned. (Do not use to verify e.g. wind direction or similarly strange data types (as ordinary differences make no sense))

Usage

```
fdbk_dt_verif_continuous(DT, strat, bootscores = F, R = 100)
```

Arguments

DT	the data table (obs and veri_data are required)
strat	list of variables to stratify for
bootscores	logical if bootstrap confidence intervals are required (5-95)
R	number of bootstrap iterations (default 100)

Value

a data.table of stratified continuous verification scores (ME,SD,RMSE,R2,LEN)(CI_L,CI_U if bootstrap)

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 (continuous scores by lead-time)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/*",intern=T)
cond                  = list(varno="varno%in%c(3,4)",veri_description="grepl('forecast',v
columnnames           = c("obs","veri_data","varno","veri_model","veri_forecast_time")
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
DT$varno              = varno_to_name(DT$varno)
strat                 = c("varno","veri_forecast_time","veri_model")
scores                = fdbk_dt_verif_continuous(DT,strat)
p =  ggplot(scores,aes(x=veri_forecast_time,y=scores,group=interaction(scorename,varno,ve
     geom_line(size=.7) + geom_point(size=1.5) + facet_wrap(~scorename, scales = "free")+
     theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))
p

#EXAMPLE 2 (talagrand diagram for each variable)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/talagrand/*SYNOP*",intern=T)
cond                  = list(veri_description="grepl('Talagrand',veri_description)")
columnnames           = c("veri_data","varno")
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
DT$varno              = varno_to_name(DT$varno)
p                     = ggplot(DT, aes(x=veri_data)) +
                        geom_histogram(binwidth=1, colour="black", fill="white") +
                        facet_wrap(~varno)+theme_bw()
p

#EXAMPLE 3 (TEMP verification)
require(ggplot2)
fnames=system("ls ~/examplesRfdbk/fof/*", intern=T)
cond = list(obs="!is.na(obs)",level="level%in%c(100000,92500,85000,70000,50000,40000,3000
columnnames = c("obs","veri_data","varno","level")
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,cores=20)
DT$varno              = varno_to_name(DT$varno)
strat                 = c("varno","level")
scores                = fdbk_dt_verif_continuous(DT,strat)
setkey(scores, scorename,varno,level)
scores                = scores[!scorename%chin%c("LEN"),]
p =  ggplot(scores,aes(x=scores,y=level,group=interaction(varno,scorename)))+
     geom_path() + facet_wrap(~scorename~varno,scales="free_x")+
     theme_bw()+theme(axis.text.x  = element_text(angle=70,hjust = 1))+scale_y_reverse()
p

#EXAMPLE 4 (SATOB verification)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/gme/satob/*",intern=T)
cond                  = list(obs="!is.na(obs)")
```

```
columnnames           = c("veri_data","varno","obs","veri_forecast_time","statid","lat","
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,10)
DT[,lon:=cut(lon,seq(-180,180,by=10),labels=seq(-175,175,by=10),include.lowest=T),]
DT[,lat:=cut(lat,seq(-90,90,by=10),labels=seq(-85,85,by=10),include.lowest=T),]
strat                 = c("varno","veri_forecast_time","statid","lon","lat")
scores                = fdbk_dt_verif_continuous(DT,strat)
scores[,lon:=as.numeric(levels(lon))[lon]]
scores[,lat:=as.numeric(levels(lat))[lat]]
scores[,varno:=varno_to_name(varno)]
scores                = scores[!is.na(scores),]
p = ggplot(droplevels(scores[varno=="U" & veri_forecast_time=="10800" & scorename=="R2",
    facet_wrap(~varno~statid~scorename)+
    scale_fill_manual(breaks=seq(0,1,by=.1),values=tim.colors(10),drop = FALSE)+borders()
p

#EXAMPLE 5 (SYNOP score time series)
require(ggplot2)
fnames   = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)
cond     = list(obs="!is.na(obs)",
               veri_description="grepl('forecast',veri_description)",
               veri_forecast_time="veri_forecast_time%in%c(1200,16800)",
               state="state%in%c(0,1)",
               statid="!is.na(as.numeric(statid))")

colnames = c("obs","veri_data","veri_forecast_time","veri_initial_date","varno","veri_mod
DT       = fdbk_dt_multi_large(fnames,cond,colnames,cores=20)
keep     = comparableRows(DT,splitCol="veri_model",splitVal=c("GME       ","ICON      "),
DT       = DT[keep]
gc()

scores                   = fdbk_dt_verif_continuous(DT,strat=c("veri_forecast_time","veri
scores$veri_initial_date = as.POSIXct(scores$veri_initial_date,format="%Y%m%d%H")
scores$varno             = varno_to_name(scores$varno)

p = ggplot(scores[varno=="RH"&scorename=="RMSE",],aes(x=veri_initial_date,y=scores,color=
    geom_line()+
    facet_grid(~scorename~varno~veri_forecast_time,scales="free")
p

#EXAMPLE 6 (TEMP time series)
require(ggplot2)
require(RColorBrewer)
fnames      = system("/bin/ls ~/examplesRfdbk/*/temp/verTEMP.*",intern=T)
LEVELS      = c(100000,92500,85000,70000,50000,40000,30000,25000,20000,15000,10000,7000,5
cond        = list(statid="!is.na(as.numeric(statid))",
                   obs="!is.na(obs)",
                   state="state%in%c(0,1,5)",
                   veri_run_type="veri_run_type%in%c(0,4)",
                   statid="round(as.numeric(statid)/1000)<=10",
                   level='level%in%c(100000,92500,85000,70000,50000,40000,30000,25000,200
                   veri_forecast_time="veri_forecast_time%in%c(0,4800,9600,14400,16800)")
columnnames = c("obs","veri_data","veri_forecast_time","veri_initial_date","level","varno
DT          = fdbk_dt_multi_large(fnames,cond,columnnames,cores=10)
DT[,valid_date:=as.POSIXct(veri_initial_date,format="%Y%m%d%H%M")+veri_forecast_time*36]
SCORES  = fdbk_dt_verif_continuous(DT,strat=c("veri_forecast_time","level","varno","valid
SCORES[,varno:=varno_to_name(varno)]
x11(width=18,height=6)
```

```
ggplot(SCORES[scorename=="ME" & varno=="T"],aes(x=valid_date,y=as.numeric(factor(level)),
         geom_raster(limits=c(-20,20))+
         facet_wrap(~veri_model~veri_forecast_time~varno,ncol=5)+
         scale_y_reverse(breaks = seq(length(LEVELS),1,by=-1),labels=rev(LEVELS))+
         scale_fill_manual("ME",values=rev(brewer.pal(9, "RdYlBu")),drop=F)+
         theme_bw()

#EXAMPLE 7 (continuous scores by lead-time plus confidence intervals)
require(ggplot2)
fnames                = system("ls ~/examplesRfdbk/*/synop/verSYNOP.*",intern=T)[1:10]
cond                  = list(varno="varno%in%c(3,4)",veri_description="grepl('forecast',v
columnnames           = c("obs","veri_data","varno","veri_forecast_time")
DT                    = fdbk_dt_multi_large(fnames,cond,columnnames,20)
DT$varno              = varno_to_name(DT$varno)
strat                 = c("varno","veri_forecast_time")
scores                = fdbk_dt_verif_continuous(DT,strat,bootscores=T,R=100)
ggplot(scores, aes(x=veri_forecast_time, y=scores,color=varno)) +
   geom_errorbar(aes(ymin=CI_L, ymax=CI_U), width=.1) +
   geom_line() +
   geom_point()+
   theme_bw()  +
   facet_wrap(~scorename,scale="free_y",ncol = 6)
```

`fdbk_dt_verif_continuous_windDir`
Deterministic scores for wind direction in degrees with bootstrap confidence intervals if required

---

Description

Deterministic scores for wind direction in degrees with bootstrap confidence intervals if required

Usage

```
fdbk_dt_verif_continuous_windDir(DT, strat, bootscores = F, R = 100)
```

Arguments

`DT`	data table (obs and veri_data are required,only for wind direction in degrees!)
`strat`	list of variables to stratify for
`bootscores`	logical if bootstrap confidence intervals are required (5-95)
`R`	number of bootstrap iterations (default 100)

Value

a data.table of stratified continuous verification scores (ME,SD,RMSE,R2,LEN)

Author(s)

Felix <felix.fundel@dwd.de>

fdbk_refdate *Get reference date(s) from feedback file(s)*

Description

Get reference date(s) from feedback file(s)

Usage

```
fdbk_refdate(filenames)
```

Arguments

filenames	filename(s) fo feedback file(s) inluding path

Value

vector of reference dates YYYYmmddHHMM

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
filenames  = system("ls ~/examplesRfdbk/icon/synop/*",intern=T)
fdbk_refdate(filenames)
```

hhmm2hour *Function to convert time in format hhmm to decimal hours. Useful to calculate a derived time from two time informations*

Description

Function to convert time in format hhmm to decimal hours. Useful to calculate a derived time from two time informations

Usage

```
hhmm2hour(x)
```

Arguments

time	in format hhmm. Can be a string or numeric

Value

time in decimal hours

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 simple examples
hhmm2hour(0145) returns 1.75
hhmm2hour(145) returns 1.75
hhmm2hour("145") returns 1.75

#EXAMPLE 2 calculate leadtime from veri_forecast_time and time
require(ggplot2)
fnames       = "/Users/josuegehring/Desktop/verTEMP.2014120112"
cond         = list(obs="!is.na(obs)",varno="varno%in%c(2,3,4,29)",ident="ident%in%c(6610)
columnnames = c("obs","veri_data","varno","state","level","veri_forecast_time","time","id
DT           = fdbk_dt_multi_large(fnames,cond,columnnames,1)
leadtime = hhmm2hour(DT$veri_forecast_time) + DT$time/60
DT[,"leadtime"] = leadtime
```

---

lonlat_to_synopregion

Non-overlapping regions, specifically defined for the DWD SYNOP verification

---

Description

Non-overlapping regions, specifically defined for the DWD SYNOP verification

Usage

```
lonlat_to_synopregion(lon, lat)
```

Arguments

lon	longitude vector
lat	latitude vector

Value

a vector of same lenght as lon or lat with character strings of the region for each point, NA for no match

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
DT = data.table(lon=c(15,85),lat=c(-30,40))
DT[,region:=lonlat_to_synopregion(lon,lat)]
DT
```

multiplot *Multiple plot function*

Description

description ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects) If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE), then plot 1 will go in the upper left, 2 will go in the upper right, and 3 will go all the way across the bottom.

Usage

```
multiplot(..., plotlist = NULL, cols = 1, layout = NULL)
```

Arguments

cols:	Number of columns in layout
layout:	A matrix specifying the layout. If present, 'cols' is ignored.

References

http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_%28ggplot2%29/

read_fdbk *Load the entire content of a fdbk file*

Description

Load the entire content of a fdbk file

Usage

```
read_fdbk(filename)
```

Arguments

filename	NetCDF fdbk filename including path

Value

a list of entries from the given fdbk file

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fdbk = read_fdbk("~/examplesRfdbk/icon/synop/verSYNOP.2014120112")
str(fdbk)
```

`read_fdbk_f` *Load the entire content of a fdbk file or only some specified variables (faster and more resource friendly)*

Description

Load the entire content of a fdbk file or only some specified variables (faster and more resource friendly)

Usage

```
read_fdbk_f(filename, vars = "")
```

Arguments

`filename`	NetCDF fdbk filename including path
`vars`	vector of variables that should be retained if not specified or "" all variables are loaded

Value

a list of entries from the given fdbk file

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
fdbk = read_fdbk_f("~/examplesRfdbk/icon/synop/verSYNOP.2014120112",c("obs","veri_data"))
str(fdbk)
```

`read_fdbk_large` *Load one fdbk file and return as list of lists of.... condition and vars arguments help to discard data you do not need*

Description

Load one fdbk file and return as list of lists of.... condition and vars arguments help to discard data you do not need

Usage

```
read_fdbk_large(fname, condition = "", vars = "")
```

Arguments

fname	feedback filename (including path)
condition	list of strings of conditions (all of the list entries are connected with the "&" operator!)
vars	vector of variable names that should be retained if not specified or "" all variables are loaded

Value

a data.table with fdbk file content

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
#EXAMPLE 1 (1x1 deg.) bias of satellite data (channel 921 from METOP-1)
fnames      = "~/examplesRfdbk/example_monRad/monRAD_2014092406.nc"
condition   = list(obs="!is.na(obs)",
                   level="level%in%c(921)",
                   statid="statid=='METOP-1   '",
                   veri_forecast_time="veri_forecast_time==0",
                   veri_run_type="veri_run_type==3",
                   veri_ens_member="veri_ens_member==-1")
fdbk         = read_fdbk_large(fnames,condition,c("lon","lat","obs"))
x11(width=12,height=7.5)
scatterplot(fdbk$DATA$lon$values,fdbk$DATA$lat$values,fdbk$DATA$obs$values,pch=20,cex=.5,
```

---

rowSds *Standard deviation on rows of array (faster than using 'apply')*

Description

Standard deviation on rows of array (faster than using 'apply')

Usage

```
rowSds(a, na.rm = F)
```

Arguments

a	2d array

Value

standard deviation on rows

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
a = array(rnorm(1e5),dim=c(1000,50))
system.time(rowSds(a))
system.time(apply(a,1,sd))
# Results agree besides some numerical precision errors
identical(round(rowSds(a),12),round(apply(a,1,sd),12))
```

---

scatterplot *Scatterplot with colored points*

Description

Scatterplot with colored points

Usage

```
scatterplot(x, y, z, zlim = NULL, ncol = 10, cpal = c("red", "white",
  "blue"), ...)
```

Arguments

x	numeric vector
y	numeric vector
z	numeric vector
zlim	plot color range (default z range)
ncol	number of colors (default 10)
cpal	color palette (default red,white,blue)

Value

a plot

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
condition  = list(obs="!is.na(obs)",level="level%in%c(921)",statid="statid=='METOP-1   '"
DT         = fdbk_dt_multi_large("~/examplesRfdbk/example_monRad/monRAD_2014092406.nc",co
x11(width=12,height=7.5)
DT[,scatterplot(lon,lat,obs,pch=20,cpal=tim.colors(),ncol=20,cex=.5)]
world(add=T,col="gray",fill=T)
```

statid_to_wmoregion *Convert WMO station-id to region*

Description

Convert WMO station-id to region

Usage

```
statid_to_wmoregion(ident)
```

Arguments

ident	numeric vector of station ID as integer (see variable "ident" in feedback file)

Value

vector of same length wiith id replaced by region shortcut

Author(s)

Felix <felix.fundel@dwd.de>

varno_to_name *Convert variable number (varno) to long or short variable name and reverse*

Description

Convert variable number (varno) to long or short variable name and reverse

Usage

```
varno_to_name(varno, short = T, rev = F)
```

Arguments

short	short or long name (boolean)
rev	TRUE: from varno to name, FALSE: from short name to varno
varno(s)	or short name(s)

Value

long or short variable name(s)

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
varno_to_name(c(3,4),short=T,rev=F)
varno_to_name(c(3,4),short=F,rev=F)
varno_to_name(c("RH","TS"),short=T,rev=T)
varno_to_name(c("RH","TS"),short=F,rev=T)
varno_to_name("geopotential (m^2/s^2)",short=F,rev=T)
varno_to_name(varno_to_name("geopotential (m^2/s^2)",short=F,rev=T))
```

---

windBias *Difference in wind direction (based un U. Pfl<c3><bc>gers code)*

Description

Difference in wind direction (based un U. Pfl<c3><bc>gers code)

Usage

```
windBias(ang_pred, ang_obs)
```

Arguments

ang_pred	forecast wind direction
ang_obs	observed wind direction

Value

wind direction difference in degree

Author(s)

Felix <felix.fundel@dwd.de>

---

windDir *Convert u,v wind in wind direction in degrees*

Description

Convert u,v wind in wind direction in degrees

Usage

```
windDir(u, v)
```

Arguments

u	u wind vector
v	v wind vector

Value

wind direction in degree (0 - <360), 360 is set to 0, if u&v=0 then return NA

Author(s)

Felix <felix.fundel@dwd.de>

Examples

```
u = c( 10,   0,   0, -10,  10,  10, -10, -10, 0)
v = c(  0,  10, -10,   0,  10, -10,  10, -10, 0)
windDir(u,v)
```

---

windSpeed *Convert u,v wind in wind speed*

Description

Convert u,v wind in wind speed

Usage

```
windSpeed(u, v)
```

Arguments

u	u wind vector
v	v wind vector

Value

wind speed

Author(s)

Felix <felix.fundel@dwd.de>

Index