

Package ‘solaR’

March 25, 2012

Type Package

Title Solar Photovoltaic Systems

Version 0.32

Date 2012-03-25

Encoding UTF-8

Author Oscar Perpiñán Lamigueiro

Maintainer Oscar Perpiñán Lamigueiro <oscar.perpinan@upm.es>

Description Calculation methods of solar radiation and performance of photovoltaic systems from daily and intradaily irradiation data sources.

URL <http://solar.r-forge.r-project.org/>

License GPL-3

LazyLoad yes

LazyData yes

Depends R (>= 2.10), methods, lattice, latticeExtra, zoo

Suggests RColorBrewer, sp, raster, rasterVis

Repository CRAN

Date/Publication 2012-03-25 15:04:30

R topics documented:

A1_calcSol	3
A2_calcG0	4
A3_calcGef	8
A4_prodGCPV	11
A5_prodPVPS	15
A6_calcShd	17

A7_optimShd	19
A8_readBD	24
A8_readG0dm	25
A8_readSIAR	26
B1_Meteo-class	29
B2_Sol-class	30
B3_G0-class	31
B4_Gef-class	32
B5_ProdGCPV-class	34
B6_ProdPVPS-class	36
B7_Shade-class	37
C_corrFdKt	39
C_fBTd	40
C_fCompD	42
C_fCompI	43
C_fInclin	45
C_fProd	47
C_fPump	49
C_fSolD	51
C_fSolI	53
C_fSombra	55
C_fTemp	57
C_fTheta	58
C_HQCurve	60
C_local2Solar	61
C_NmgPVPS	63
C_sample2Diff	65
C_TargetDiagram	66
C_utils-angle	68
C_utils-time	69
D_as.data.frameD-methods	70
D_as.data.frameI-methods	70
D_as.data.frameM-methods	71
D_as.data.frameY-methods	72
D_as.zooD-methods	72
D_as.zooI-methods	73
D_as.zooM-methods	74
D_as.zooY-methods	75
D_compare-methods	75
D_getData-methods	77
D_getG0-methods	77
D_getLat-methods	78
D_indexD-methods	78
D_indexI-methods	79
D_indexRep-methods	79
D_levelplot-methods	80
D_Losses-methods	80
D_mergesolaR-methods	81

D_shadeplot-methods	83
D_window-methods	83
D_xyplot-methods	84
E_helios	85
E_prodEx	86
E_pumpCoef	86
E_solaR.theme	87
Index	88

A1_calcSol	<i>Apparent movement of the Sun from the Earth</i>
------------	--

Description

Compute the apparent movement of the Sun from the Earth with the functions `fSolD` and `fSolI`.

Usage

```
calcSol(lat, BTd, sample='hour', BTi, EoT=TRUE, keep.night=TRUE, method='michalsky')
```

Arguments

lat	Latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
BTd	Daily time base, a POSIXct object which may be the result of <code>fBTd</code> . It is not considered if BTi is provided.
sample	Increment of the intradaily sequence. It is a character string, containing one of "sec", "min", "hour". This can optionally be preceded by a (positive or negative) integer and a space, or followed by "s". It is used by <code>seq.POSIXt</code> . It is not considered if BTi is provided.
BTi	Intradaily time base, a POSIXct object to be used by <code>fSolI</code> . It could be the index of the GOI argument to <code>calcG0</code> .
EoT	logical, if TRUE the Equation of Time is used. Default is TRUE.
keep.night	logical, if TRUE (default) the night is included in the time series.
method	character, method for the sun geometry calculations to be chosen from 'cooper', 'spencer', 'michalsky' and 'strous'. See references for details.

Value

A `Sol-class` object.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Cooper, P.I., Solar Energy, 12, 3 (1969). "The Absorption of Solar Radiation in Solar Stills"
- Spencer, Search 2 (5), 172, <http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html>
- Strous: <http://www.astro.uu.nl/~strous/AA/en/reken/zonpositie.html>
- Michalsky, J., 1988: The Astronomical Almanac's algorithm for approximate solar position (1950-2050), Solar Energy 40, 227-235, ftp://climate1.gsfc.nasa.gov/wiscombe/Solar_Rad/SunAngles/sunae.f
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. <http://procomun.wordpress.com/documentos/libroesf/>

Examples

```
BTd=fBTd(mode='serie')

lat=37.2
sol=calcSol(lat, BTd[100])
print(as.zooD(sol))

library(lattice)
xyplot(as.zooI(sol))

solStrous=calcSol(lat, BTd[100], method='strous')
print(as.zooD(solStrous))

solSpencer=calcSol(lat, BTd[100], method='spencer')
print(as.zooD(solSpencer))

solCooper=calcSol(lat, BTd[100], method='cooper')
print(as.zooD(solCooper))
```

A2_calcG0

Irradiation and irradiance on the horizontal plane.

Description

This function obtains the global, diffuse and direct irradiation and irradiance on the horizontal plane from the values of *daily* and *intradaily* global irradiation on the horizontal plane. It makes use of the functions [calcSol](#), [fCompD](#), [fCompI](#), [fBTd](#) and [readBD](#) (or equivalent).

Besides, if information about maximum and minimum temperatures values are available it obtains a series of temperature values with [fTemp](#).

Usage

```
calcG0(lat, modeRad='prom', dataRad, prom, mapa, bd,
bdI, sample='hour', keep.night=TRUE, sunGeometry='michalsky',
corr, f)
```

Arguments

lat	numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
modeRad	<p>A character string, describes the kind of source data of the global irradiation and ambient temperature.</p> <p>It can be modeRad='prom' for monthly mean calculations. With this option, a set of 12 values inside dataRad must be provided, as defined in readG0dm.</p> <p>When modeRad='siar' the source data is downloaded from www.marm.es/siar with the information provided in dataRad. In previous versions, this radiation mode was named as 'mapa' due to the webpage of the database. However, since the database is no longer available at mapa.es, modeRad='mapa' is deprecated and modeRad='siar' should be used instead.</p> <p>If modeRad='bd' the information of <i>daily</i> irradiation is read from a file, a data.frame defined by dataRad, a zoo or a Meteo object. (See readBD, df2Meteo and zoo2Meteo for details).</p> <p>If modeRad='bdI' the information of <i>intradaily</i> irradiation is read from a file, a data.frame defined by dataRad, a zoo or a Meteo object. (See readBDi, dfI2Meteo and zoo2Meteo for details).</p>
dataRad	<ul style="list-style-type: none"> • If modeRad='prom', a list as defined by readG0dm • If modeRad='siar', a list as defined by readSIAR. • If modeRad='bd' a list as defined by readBD, df2Meteo or zoo2Meteo, or a Meteo object. The resulting object will include a column named Ta, with information about ambient temperature. • If modeRad='bdI' a list as defined by readBDi, dfI2Meteo or zoo2Meteo, or a Meteo object. The resulting object will include a column named Ta, with information about ambient temperature.
prom, mapa, bd, bdI	Deprecated. dataRad should be used instead.
sample	character, containing one of "sec", "min", "hour". This can optionally be preceded by a (positive or negative) integer and a space, or followed by "s" (used by seq.POSIXt). It is not used when modeRad="bdI".
keep.night	logical. When it is TRUE (default) the time series includes the night.
sunGeometry	character, method for the sun geometry calculations. See calcSol , fSolD and fSolI .
corr	<p>A character, the correlation between the the fraction of diffuse irradiation and the clearness index to be used.</p> <p>With this version several options are available, as described in corrFdKt. For example, the FdKtPage is selected with corr='Page' while the FdKtCPR with corr='CPR'.</p> <p>If corr='user' the use of a correlation defined by a function f is possible.</p> <p>If corr='none' the object defined by dataRad should include information about global, diffuse and direct daily irradiation with columns named G0d, D0d and B0d, respectively (or G0, D0 and B0 if modeRad='bdI'). If corr is missing, then it is internally set to CPR when modeRad='siar' or modeRad='bd', to Page when modeRad='prom' and to BRL when modeRad='bdI'.</p>

f A function defining a correlation between the fraction of diffuse irradiation and the clearness index. It is only necessary when `corr='user'`

Value

A G0 object.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[calcSol](#), [fCompD](#), [fCompI](#), [readG0dm](#), [readBD](#), [readBDi](#), [readSIAR](#), [corrFdKt](#).

Examples

```
G0dm=c(2.766,3.491,4.494,5.912,6.989,7.742,7.919,7.027,5.369,3.562,2.814,2.179)*1000;
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2,
     15.2)
```

```
g0<-calcG0(lat=37.2, modeRad='prom', dataRad=list(G0dm=G0dm, Ta=Ta))
print(g0)
xyplot(g0)
```

```
## Not run:
g0<-calcG0(lat=37.2,
           modeRad='siar',
           dataRad=list(prov=28,est=3,
                       start='01/01/2009', end='31/12/2009'))
print(g0)
xyplot(g0)
```

```
##Now the G0I component of g0 is used as
##the bdI argument to calcG0 in order to
##test the intradaily correlations of fd-kt
```

```
BDi=as.zooI(g0)
BDi$Ta=25 ##Information about temperature must be contained in BDi
```

```
g02<-calcG0(lat=37.2,
            modeRad='bdI',
            dataRad=list(lat=37.2, file=BDi),
            corr='none')
```

```
print(g02)
```

```

g03<-calcG0(lat=37.2,
            modeRad='bdI',
            dataRad=list(lat=37.2, file=BDi),
            corr='BRL')
print(g03)

xyplot(fd~kt, data=g03, pch=19, alpha=0.3)

## End(Not run)

## Not run:
##NREL-MIDC
##La Ola, Lanai
##Latitude: 20.76685o North
##Longitude: 156.92291o West
##Elevation: 381 meters AMSL
##Time Zone: -10.0

file='http://www.nrel.gov/midc/apps/plot.pl?site=LANAI&start=20090722&edy=19&emo=11&eyr=2010&zenloc=19&year=20

dat <- read.table(file, header=TRUE, sep=',')
names(dat) <- c('date', 'hour', 'G0', 'B', 'D0', 'Ta')

##B is direct normal. We need direct horizontal.
dat$B0 <- dat$G0-dat$D0

##http://www.nrel.gov/midc/la_ola_lanai/instruments.html:
##The datalogger program runs using Greenwich Mean Time (GMT),
##data is converted to Hawaiiin Standard Time (HST) after data collection
idxLocal <- with(dat, as.POSIXct(paste(date, hour), format='%m/%d/%Y %H:%M', tz='HST'))
idx <- local2Solar(idxLocal, lon=-156.9339)

z <- zoo(dat[,c('G0', 'D0', 'B0', 'Ta')], idx)

lat=20.77

NRELMeteo <- zoo2Meteo(z, lat=lat)
xyplot(NRELMeteo)

g0 <- calcG0(lat=lat, modeRad='bdI', dataRad=NRELMeteo, corr='none')
xyplot(g0)
xyplot(as.zooI(g0), superpose=TRUE)

g02 <- calcG0(lat=lat, modeRad='bdI', dataRad=NRELMeteo, corr='BRL')
xyplot(g02)
xyplot(as.zooI(g02), superpose=TRUE)
xyplot(fd~kt, data=g02, pch=19, cex=0.5, alpha=0.5)

g03 <- calcG0(lat=lat, modeRad='bdI', dataRad=NRELMeteo, corr='CLIMEDh')
xyplot(g03)
xyplot(as.zooI(g03), superpose=TRUE)
xyplot(fd~kt, data=g03, pch=19, cex=0.5, alpha=0.5)

```

```
## End(Not run)
```

A3_calcGef

Irradiation and irradiance on the generator plane.

Description

This function obtains the global, diffuse and direct irradiation and irradiance on the generator plane from the values of *daily* or *intradaily* global irradiation on the horizontal plane. It makes use of the functions [calcG0](#), [fTheta](#), [fInclin](#). Besides, it can calculate the shadows effect with the [calcShd](#) function.

Usage

```
calcGef(lat,
        modeTrk='fixed',
        modeRad='prom',
        dataRad,
        prev, prom, mapa, bd, bdI,
        sample='hour',
        keep.night=TRUE,
        sunGeometry='michalsky',
        corr, f,
        betaLim=90, beta=abs(lat)-10, alfa=0,
        iS=2, alb=0.2, horizBright=TRUE, HCPV=FALSE,
        modeShd='',
        struct=list(),
        distances=data.frame())
```

Arguments

lat	numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
modeTrk	character, to be chosen from 'fixed', 'two' or 'horiz'. When modeTrk='fixed' the surface is fixed (inclination and azimuth angles are constant). The performance of a two-axis tracker is calculated with modeTrk='two', and modeTrk='horiz' is the option for an horizontal N-S tracker. Its default value is modeTrk='fixed'
modeRad, dataRad, prom, mapa, bd, bdI	Information about the source data of the global irradiation. See calcG0 for details.
prev	A G0 object (or something that can be coerced to a G0 object). It is used when modeRad='prev'. It is deprecated and dataRad should be used instead.
sample, keep.night	See calcSol for details.
sunGeometry	character, method for the sun geometry calculations. See calcSol , fSold and fSolI .

corr, f	See calcG0 for details.
beta	numeric, inclination angle of the surface (degrees). It is only needed when modeTrk='fixed'.
betaLim	numeric, maximum value of the inclination angle for a tracking surface. Its default value is 90 (no limitation))
alfa	numeric, azimuth angle of the surface (degrees). It is positive to the West. It is only needed when modeTrk='fixed'. Its default value is alfa=0
iS	integer, degree of dirtiness. Its value must be included in the set (1,2,3,4). iS=1 corresponds to a clean surface while iS=4 is the selection for a dirty surface. Its default value is 2.
alb	numeric, albedo reflection coefficient. Its default value is 0.2
modeShd, struct, distances	See calcShd for details.
horizBright	logical, if TRUE, the horizon brightness correction proposed by Reind et al. is used.
HCPV	logical, if TRUE the diffuse and albedo components of the <i>effective</i> irradiance are set to zero. HCPV is the acronym of High Concentration PV system.

Value

A Gef object.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Hay, J. E. and McKay, D. C.: Estimating Solar Irradiance on Inclined Surfaces: A Review and Assessment of Methodologies. *Int. J. Solar Energy*, (3):pp. 203, 1985.
- Martin, N. and Ruiz, J.M.: Calculation of the PV modules angular losses under field conditions by means of an analytical model. *Solar Energy Materials & Solar Cells*, 70:25–38, 2001.
- D. T. Reindl and W. A. Beckman and J. A. Duffie: Evaluation of hourly tilted surface radiation models, *Solar Energy*, 45:9-17, 1990.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[calcG0](#), [fTheta](#), [fInclin](#), [calcShd](#).

Examples

```

###12 Average days.

G0dm=c(2.766,3.491,4.494,5.912,6.989,7.742,7.919,7.027,5.369,3.562,2.814,2.179)*1000;
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)

##Fixed surface, default values of inclination and azimuth.

gef<-calcGef(lat=37.2, modeRad='prom', dataRad=list(G0dm=G0dm, Ta=Ta))
print(gef)
xyplot(gef)

##Two-axis surface, no limitation angle.

gef2<-calcGef(lat=37.2, modeRad='prom', dataRad=list(G0dm=G0dm, Ta=Ta), modeTrk='two')
print(gef2)
xyplot(gef2)

## Not run:
###Irradiation data from mapa.es/siar

##Fixed surface

gefSIAR<-calcGef(lat=41,
                 modeRad='siar',
                 dataRad=list(prov=28,est=3,
                              start='01/01/2009', end='31/12/2009'),
                 keep.night=FALSE)
print(gefSIAR)
xyplot(gefSIAR)
##Two-axis tracker, using the previous result.
##'gefSIAR' is internally coerced to a 'G0' object.

gefSIAR2<-calcGef(lat=41, modeRad='prev', dataRad=gefSIAR, modeTrk='two')
print(gefSIAR2)
xyplot(gefSIAR2)

###Shadows between two-axis trackers, again using the gefSIAR result.

struct=list(W=23.11, L=9.8, Nrow=2, Ncol=8)
distances=data.frame(Lew=40, Lns=30, H=0)

gefShd<-calcGef(lat=41, modeRad='prev',
                dataRad=gefSIAR, modeTrk='two',
                modeShd=c('area', 'prom'),
                struct=struct, distances=distances)

print(gefShd)
##The Gef0, Bef0 and Def0 values are the same as those contained in the
##      gefSIAR2 object

## End(Not run)

```

A4_prodGCPV

Performance of a grid connected PV system.

Description

Compute every step from solar angles to effective irradiance to calculate the performance of a grid connected PV system.

Usage

```
prodGCPV(lat,
         modeTrk='fixed',
         modeRad='prom',
         dataRad,
         prev, prom, mapa, bd, bdI,
         sample='hour',
         keep.night=TRUE,
         sunGeometry='michalsky',
         corr, f,
         betaLim=90, beta=abs(lat)-10, alfa=0,
         iS=2, alb=0.2, horizBright=TRUE, HCPV=FALSE,
         module=list(),
         generator=list(),
         inverter=list(),
         effSys=list(),
         modeShd='',
         struct=list(),
         distances=data.frame()
        )
```

Arguments

lat	numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
modeTrk	A character string, describing the tracking method of the generator. See calcGef for details.
modeRad, dataRad, prom, mapa, bd, bdI	Information about the source data of the global irradiation. See calcG0 for details.
prev	Deprecated, dataRad should be used instead. A Gef or G0 object (or something that can be coerced to a Gef object). It is only used when modeRad='prev'. prodGCPV will call calcGef for the effective irradiance and irradiation procedure only when prev is a G0 object.
sample, keep.night	See calcSol for details.

sunGeometry	character, method for the sun geometry calculations. See calcSol , fSolD and fSolI .
corr, f	See calcG0 for details.
betaLim, beta,	alfa, iS, alb, horizBright, HCPV See calcGef for details.
module	list of numeric values with information about the PV module, Vocn open-circuit voltage of the module at Standard Test Conditions (default value 57.6 volts.) Iscn short circuit current of the module at Standard Test Conditions (default value 4.7 amperes.) Vmn maximum power point voltage of the module at Standard Test Conditions (default value 46.08 amperes.) Imn Maximum power current of the module at Standard Test Conditions (default value 4.35 amperes.) Ncs number of cells in series inside the module (default value 96) Ncp number of cells in parallel inside the module (default value 1) CoefVT coefficient of decrement of voltage of each cell with the temperature (default value 0.0023 volts per celsius degree) TONC nominal operational cell temperature, celsius degree (default value 47).
generator	list of numeric values with information about the generator, Nms number of modules in series (default value 12) Nmp number of modules in parallel (default value 11)
inverter	list of numeric values with information about the DC/AC inverter, Ki vector of three values, coefficients of the efficiency curve of the inverter (default c(0.01, 0.025, 0.05)), or a matrix of nine values (3x3) if there is dependence with the voltage (see references). Pinv nominal inverter power (W) (default value 25000 watts.) Vmin, Vmax minimum and maximum voltages of the MPP range of the inverter (default values 420 and 750 volts) Gumb minimum irradiance for the inverter to start (W/m ²) (default value 20 W/m ²)
effSys	list of numeric values with information about the system losses, ModQual average tolerance of the set of modules (%), default value is 3 ModDisp module parameter disperssion losses (%), default value is 2 OhmDC Joule losses due to the DC wiring (%), default value is 1.5 OhmAC Joule losses due to the AC wiring (%), default value is 1.5 MPP average error of the MPP algorithm of the inverter (%), default value is 1 TrafoMT losses due to the MT transformer (%), default value is 1 Disp losses due to stops of the system (%), default value is 0.5
modeShd, struct,	distances See calcShd for details.

Details

The calculation of the irradiance on the horizontal plane is carried out with the function `calcG0`. The transformation to the inclined surface makes use of the `fTheta` and `fInclin` functions inside the `calcGef` function. The shadows are computed with `calcShd` while the performance of the PV system is simulated with `fProd`.

Value

A ProdGCPV object.

Author(s)

Oscar Perpiñán Lamigueiro

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

`fProd`, `calcGef`, `calcShd`, `calcG0`, `compare`, `compareLosses`, `mergesolaR`

Examples

```
library(lattice)
library(latticeExtra)

lat=37.2;
G0dm=c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562, 2814,
2179)
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
prom=list(G0dm=G0dm, Ta=Ta)

###Comparison of different tracker methods
prodFixed<-prodGCPV(lat=lat,dataRad=prom, keep.night=FALSE)
prod2x<-prodGCPV(lat=lat, dataRad=prom, modeTrk='two', keep.night=FALSE)
prodHoriz<-prodGCPV(lat=lat,dataRad=prom, modeTrk='horiz', keep.night=FALSE)

##Comparison of yearly productivities
compare(prodFixed, prod2x, prodHoriz)
compareLosses(prodFixed, prod2x, prodHoriz)

##Comparison of power time series
ComparePac<-CBIND(two=as.zooI(prod2x)$Pac,
horiz=as.zooI(prodHoriz)$Pac,
fixed=as.zooI(prodFixed)$Pac)
AngSol=as.zooI(as(prodFixed, 'Sol'))
ComparePac=CBIND(AngSol, ComparePac)
mon=month(index(ComparePac))
```

```

xyplot(two+horiz+fixed~AzS|mon, data=ComparePac,
       type='l', auto.key=list(space='right', lines=TRUE, points=FALSE),ylab='Pac')

## Not run:
###Use of modeRad='siar' and modeRad='prev'
prodSIARFixed<-prodGCPV(lat=41,
                        modeRad='siar',
                        dataRad=list(prov=28,est=3,
                                     start='01/01/2009', end='31/12/2009'),
                        keep.night=FALSE)

##We want to compare systems with different effective irradiance
##so we have to convert prodSIARFixed to a 'G0' object.
G0SIAR=as(prodSIARFixed, 'G0')

prodSIAR2x<-prodGCPV(lat=41,modeTrk='two',modeRad='prev', dataRad=G0SIAR)
prodSIARHoriz<-prodGCPV(lat=41, modeTrk='horiz',modeRad='prev',
                        dataRad=G0SIAR)

##Comparison of yearly values
compare(prodSIARFixed, prodSIAR2x, prodSIARHoriz)
compareLosses(prodSIARFixed, prodSIAR2x, prodSIARHoriz)

##Compare of daily productivities of each tracking system
compareYf <- mergesolaR(prodSIARFixed, prodSIAR2x, prodSIARHoriz)
xyplot(compareYf, superpose=TRUE,
       ylab='kWh/kWp', main='Daily productivity', auto.key=list(space='right'))

## End(Not run)

###Shadows
#Two-axis trackers
struct2x=list(W=23.11, L=9.8, Nrow=2, Ncol=8)
dist2x=data.frame(Lew=40, Lns=30, H=0)
prod2xShd<-prodGCPV(lat=lat, dataRad=prom, modeTrk='two',
                   modeShd='area', struct=struct2x, distances=dist2x)
print(prod2xShd)

#Horizontal N-S tracker
structHoriz=list(L=4.83);
distHoriz=data.frame(Lew=structHoriz$L*4);

#Without Backtracking
prodHorizShd<-prodGCPV(lat=lat, dataRad=prom, sample='10 min',
                      modeTrk='horiz',
                      modeShd='area', betaLim=60,
                      distances=distHoriz,
                      struct=structHoriz)
print(prodHorizShd)

xyplot(r2d(Beta)~r2d(w),
       data=prodHorizShd,
       type='l',

```

```

    main='Inclination angle of a horizontal axis tracker',
    xlab=expression(omega (degrees)),
    ylab=expression(beta (degrees)))

#With Backtracking
prodHorizBT<-prodGCPV(lat=lat, dataRad=prom, sample='10 min',
  modeTrk='horiz',
  modeShd='bt', betaLim=60,
  distances=distHoriz,
  struct=structHoriz)

print(prodHorizBT)

xyplot(r2d(Beta)~r2d(w),
  data=prodHorizBT,
  type='l',
  main='Inclination angle of a horizontal axis tracker\n with backtracking',
  xlab=expression(omega (degrees)),
  ylab=expression(beta (degrees)))

## Not run:
compare(prodFixed, prod2x, prodHoriz, prod2xShd,
  prodHorizShd, prodHorizBT)

compareLosses(prodFixed, prod2x, prodHoriz, prod2xShd,
  prodHorizShd, prodHorizBT)

compareYf2 <- mergesolar(prodFixed, prod2x, prodHoriz, prod2xShd,
  prodHorizShd, prodHorizBT)
xyplot(compareYf2, superpose=TRUE,
  ylab='kWh/kWp', main='Daily productivity', auto.key=list(space='right'))

## End(Not run)

```

A5_prodPVPS

Performance of a PV pumping system

Description

Compute every step from solar angles to effective irradiance to calculate the performance of a PV pumping system.

Usage

```

prodPVPS(lat,
  modeTrk='fixed',
  modeRad='prom',
  dataRad,
  prev, prom, mapa, bd,bdI,
  sample='hour',

```

```

keep.night=TRUE,
sunGeometry='michalsky',
corr, f,
betaLim=90, beta=abs(lat)-10, alfa=0,
iS=2, alb=0.2, horizBright=TRUE, HCPV=FALSE,
pump , H,
Pg, converter= list(),
effSys=list()
)

```

Arguments

lat	numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
modeTrk	A character string, describing the tracking method of the generator. See calcGef for details.
modeRad, dataRad, prom, mapa, bd, bdI	Information about the source data of the global irradiation. See calcG0 for details.
prev	Deprecated, dataRad should be used instead. A G0 object (or something that can be coerced to a G0 object). It is used when modeRad='prev'.
sample, keep.night	See calcSol for details.
sunGeometry	character, method for the sun geometry calculations. See calcSol , fSold and fSolI .
corr, f	See calcG0 for details.
betaLim, beta, alfa, iS, alb, horizBright, HCPV	See calcGef for details.
pump	A list extracted from pumpCoef
H	Total manometric head (m)
Pg	Nominal power of the PV generator (Wp)
converter	list containing the nominal power of the frequency converter, Pnom, and Ki, vector of three values, coefficients of the efficiency curve.
effSys	list of numeric values with information about the system losses, ModQual average tolerance of the set of modules (%), default value is 3 ModDisp module parameter dispersion losses (%), default value is 2 OhmDC Joule losses due to the DC wiring (%), default value is 1.5 OhmAC Joule losses due to the AC wiring (%), default value is 1.5

Details

The calculation of the irradiance on the generator is carried out with the function [calcGef](#). The performance of the PV system is simulated with [fPump](#).

Value

A `ProdPVPS` object.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Abella, M. A., Lorenzo, E. y Chenlo, F.: PV water pumping systems based on standard frequency converters. *Progress in Photovoltaics: Research and Applications*, 11(3):179–191, 2003, ISSN 1099-159X.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

`NmgPVPS`, `fPump`, `pumpCoef`

Examples

```
library(lattice)

data(pumpCoef)

CoefSP8A44<-subset(pumpCoef, Qn==8&stages==44)

## Not run:
prodSP8A44<-prodPVPS(lat=41,
                    modeRad='siar',
                    dataRad=list(prov=28,est=3,
                                start='01/01/2009', end='31/12/2009'),
                    pump=CoefSP8A44, Pg=6000, H=140)
print(prodSP8A44)

xyplot(prodSP8A44)

xyplot(Q~Gef|month, data=prodSP8A44, cex=0.5)

## End(Not run)
```

Description

Compute the irradiance and irradiation including shadows for two-axis and horizontal N-S axis trackers and fixed surfaces. It makes use of the function `fSombra` for the shadows factor calculation. It is used by the function `calcGef`.

Usage

```
calcShd(radEf, modeTrk='fixed', modeShd='', struct=list(),
        distances=data.frame())
```

Arguments

radEf	A Gef object. It may be the result of the calcGef function.
modeTrk	character, to be chosen from 'fixed', 'two' or 'horiz'. When modeTrk='fixed' the surface is fixed (inclination and azimuth angles are constant). The performance of a two-axis tracker is calculated with modeTrk='two', and modeTrk='horiz' is the option for an horizontal N-S tracker. Its default value is modeTrk='fixed'
modeShd	character, defines the type of shadow calculation. In this version of the package the effect of the shadow is calculated as a proportional reduction of the circumsolar diffuse and direct irradiances. This type of approach is selected with modeShd='area'. In future versions other approaches which relate the geometric shadow and the electrical connections of the PV generator will be available. If modeTrk='horiz' it is possible to calculate the effect of backtracking with modeShd='bt'. If modeShd=c('area', 'bt') the backtracking method will be carried out and therefore no shadows will appear. Finally, for two-axis trackers it is possible to select modeShd='prom' in order to calculate the effect of shadows on an average tracker (see fSombra6). The result will include three variables (Gef0, Def0 and Bef0) with the irradiance/irradiation without shadows as a reference.
struct	list. When modeTrk='fixed' or modeTrk='horiz' only a component named L, which is the height (meters) of the tracker, is needed. For two-axis trackers (modeTrk='two'), an additional component named W, the width of the tracker, is required. Moreover, only when modeTrk='two' two components named Nrow and Ncol are included under this list. These components define, respectively, the number of rows and columns of the whole set of two-axis trackers in the PV plant.
distances	data.frame. When modeTrk='fixed' it includes a component named D for the distance between fixed surfaces. An additional component named H can be included with the relative height between surfaces. When modeTrk='horiz' it only includes a component named Lew, being the distance between horizontal NS trackers along the East-West direction. When modeTrk='two' it includes a component named Lns being the distance between trackers along the North-South direction, a component named Lew, being the distance between trackers along the East-West direction and a (optional) component named H with the relative height between surfaces. The distances, in meters, are defined between axis of the trackers.

Value

A Gef object including three additional variables (Gef0, Def0 and Bef0) in the slots GefI, GefD, Gefdm and Gefy with the irradiance/irradiation without shadows as a reference.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[calcG0](#), [fTheta](#), [fInclin](#), [calcShd](#).

A7_optimShd

Shadows calculation for a set of distances between elements of a PV grid connected plant.

Description

The optimum distance between trackers or static structures of a PV grid connected plant depends on two main factors: the ground requirement ratio (defined as the ratio of the total ground area to the generator PV array area), and the productivity of the system including shadow losses. Therefore, the optimum separation may be the one which achieves the highest productivity with the lowest ground requirement ratio.

However, this definition is not complete since the terrain characteristics and the costs of wiring or civil works could alter the decision. This function is a help for choosing this distance: it computes the productivity for a set of combinations of distances between the elements of the plant.

Usage

```
optimShd(lat,
         modeTrk='fixed',
         modeRad='prom',
         dataRad,
         prev, prom, mapa, bd,
         sample='hour',
         keep.night=TRUE,
         sunGeometry='michalsky',
         betaLim=90, beta=abs(lat)-10, alfa=0,
         iS=2, alb=0.2, HCPV=FALSE,
         module=list(),
         generator=list(),
         inverter=list(),
         effSys=list(),
         modeShd='',
         struct=list(),
         distances=data.frame(),
```

```
res=2,
prog=TRUE)
```

Arguments

lat	numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
modeTrk	character, to be chosen from 'fixed', 'two' or 'horiz'. When modeTrk='fixed' the surface is fixed (inclination and azimuth angles are constant). The performance of a two-axis tracker is calculated with modeTrk='two', and modeTrk='horiz' is the option for an horizontal N-S tracker. Its default value is modeTrk='fixed'
modeRad, dataRad, prom, mapa, bd	Information about the source data of the global irradiation. See calcG0 for details. For this function the option modeRad='bdI' is not supported.
prev	Deprecated, dataRad should be used instead. A G0 object (or something that can be coerced to a G0 object). It is used when modeRad='prev'.
sample	character, containing one of "sec", "min", "hour". This can optionally be preceded by a (positive or negative) integer and a space, or followed by "s" (used by seq.POSIXt)
keep.night	logical When it is TRUE (default) the time series includes the night.
sunGeometry	character, method for the sun geometry calculations. See calcSol , fSolD and fSolI .
betaLim	numeric, maximum value of the inclination angle for a tracking surface. Its default value is 90 (no limitation)
beta	numeric, inclination angle of the surface (degrees). It is only needed when modeTrk='fixed'.
alfa	numeric, azimuth angle of the surface (degrees). It is positive to the West. It is only needed when modeTrk='fixed'. Its default value is alfa=0
iS	integer, degree of dirtiness. Its value must be included in the set (1,2,3,4). iS=1 corresponds to a clean surface while iS=4 is the selection for a dirty surface. Its default value is 2
alb	numeric, albedo reflection coefficient. Its default value is 0.2
HCPV	logical, if TRUE the diffuse and albedo components of the <i>effective</i> irradiance are set to zero. HCPV is the acronym of High Concentration PV system.
module	list of numeric values with information about the PV module, Vocn open-circuit voltage of the module at Standard Test Conditions (default value 57.6 volts.) Iscn short circuit current of the module at Standard Test Conditions (default value 4.7 amperes.) Vmn maximum power point voltage of the module at Standard Test Conditions (default value 46.08 amperes.) Imn Maximum power current of the module at Standard Test Conditions (default value 4.35 amperes.)

	Ncs number of cells in series inside the module (default value 96)
	Ncp number of cells in parallel inside the module (default value 1)
	CoefVT coefficient of decrement of voltage of each cell with the temperature (default value 0.0023 volts per celsius degree)
	TONC nominal operational cell temperature, celsius degree (default value 47).
generator	list of numeric values with information about the generator,
	Nms number of modules in series (default value 12)
	Nmp number of modules in parallel (default value 11)
inverter	list of numeric values with information about the DC/AC inverter,
	Ki vector of three values, coefficients of the efficiency curve of the inverter (default c(0.01, 0.025, 0.05)), or a matrix of nine values (3x3) if there is dependence with the voltage (see references).
	Pinv nominal inverter power (W) (default value 25000 watts.)
	Vmin, Vmax minimum and maximum voltages of the MPP range of the inverter (default values 420 and 750 volts)
	Gumb minimum irradiance for the inverter to start (W/m ²) (default value 20 W/m ²)
effSys	list of numeric values with information about the system losses,
	ModQual average tolerance of the set of modules (%), default value is 3
	ModDisp module parameter dispersion losses (%), default value is 2
	OhmDC Joule losses due to the DC wiring (%), default value is 1.5
	OhmAC Joule losses due to the AC wiring (%), default value is 1.5
	MPP average error of the MPP algorithm of the inverter (%), default value is 1
	TrafoMT losses due to the MT transformer (%), default value is 1
	Disp losses due to stops of the system (%), default value is 0.5
modeShd	character, defines the type of shadow calculation. In this version of the package the effect of the shadow is calculated as a proportional reduction of the circumsolar diffuse and direct irradiances. This type of approach is selected with modeShd='area'. In future versions other approaches which relate the geometric shadow and the electrical connections of the PV generator will be available. If modeTrk='horiz' it is possible to calculate the effect of backtracking with modeShd='bt'. If modeShd=c('area', 'bt') the backtracking method will be carried out and therefore no shadows will appear. Finally, for two-axis trackers it is possible to select modeShd='prom' in order to calculate the effect of shadows on an average tracker (see fSombra6). The result will include three variables (Gef0, Def0 and Bef0) with the irradiance/irradiation without shadows as a reference.
struct	list. When modeTrk='fixed' or modeTrk='horiz' only a component named L, which is the height (meters) of the tracker, is needed. For two-axis trackers (modeTrk='two'), an additional component named W, the width of the tracker, is required. Moreover, two components named Nrow and Ncol are included under this list. These components define, respectively, the number of rows and columns of the whole set of trackers in the PV plant.
distances	list, whose three components are vectors of length 2:

	Lew (only when modeTrk='horiz' or modeTrk='two'), minimum and maximum distance (meters) between horizontal NS and two-axis trackers along the East-West direction.
	Lns (only when modeTrk='two'), minimum and maximum distance (meters) between two-axis trackers along the North-South direction.
	D (only when modeTrk='fixed'), minimum and maximum distance (meters) between fixed surfaces.
	These distances, in meters, are defined between the axis of the trackers.
res	numeric; optimShd constructs a sequence from the minimum to the maximum value of distances, with res as the increment, in meters, of the sequence.
prog	logical, show a progress bar; default value is TRUE

Details

optimShd calculates the energy produced for every combination of distances as defined by distances and res. The result of this function is a [Shade-class](#) object. A method of shadeplot for this class is defined ([shadeplot-methods](#)), and it shows the graphical relation between the productivity and the distance between trackers or fixed surfaces.

Value

A [Shade](#) object.

Author(s)

Oscar Perpiñán Lamigueiro

References

- Perpiñán, O.: Grandes Centrales Fotovoltaicas: producción, seguimiento y ciclo de vida. PhD Thesis, UNED, 2008. <http://e-spacio.uned.es/fez/view.php?id=bibliuned:20080>.
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[prodGCPV](#), [calcShd](#)

Examples

```
library(lattice)
library(latticeExtra)

lat=37.2;
G0dm=c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562, 2814,
2179)
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
prom=list(G0dm=G0dm, Ta=Ta)
```

```

###Two-axis trackers
struct2x=list(W=23.11, L=9.8, Nrow=2, Ncol=8)
dist2x=list(Lew=c(30,50),Lns=c(20,50))

#Monthly averages
ShdM2x<-optimShd(lat=lat, dataRad=prom, modeTrk='two',
                 modeShd=c('area','prom'), distances=dist2x, struct=struct2x, res=5)

shadeplot(ShdM2x)

pLew=xyplot(Yf~GRR,data=ShdM2x,groups=factor(Lew),type=c('l','g'),
            main='Productivity for each Lew value')
pLew+glayer(panel.text(x[1], y[1], group.value))

pLns=xyplot(Yf~GRR,data=ShdM2x,groups=factor(Lns),type=c('l','g'),
            main='Productivity for each Lns value')
pLns+glayer(panel.text(x[1], y[1], group.value))

###Horizontal axis tracker
structHoriz=list(L=4.83);
distHoriz=list(Lew=structHoriz$L*c(2,5));

#Without backtracking
Shd12Horiz<-optimShd(lat=lat, dataRad=prom,
                    modeTrk='horiz',
                    betaLim=60,
                    distances=distHoriz, res=2,
                    struct=structHoriz,
                    modeShd='area')

shadeplot(Shd12Horiz)

xyplot(diff(Yf)~GRR[-1],data=Shd12Horiz,type=c('l','g'))

#with Backtracking
Shd12HorizBT<-optimShd(lat=lat, dataRad=prom,
                      modeTrk='horiz',
                      betaLim=60,
                      distances=distHoriz, res=1,
                      struct=structHoriz,
                      modeShd='bt')

shadeplot(Shd12HorizBT)
xyplot(diff(Yf)~GRR[-1],data=Shd12HorizBT,type=c('l','g'))

###Fixed system
structFixed=list(L=5);
distFixed=list(D=structFixed$L*c(1,3));
Shd12Fixed<-optimShd(lat=lat, dataRad=prom,
                    modeTrk='fixed',
                    distances=distFixed, res=1,
                    struct=structFixed,

```

```

        modeShd='area')
shadeplot(Shd12Fixed)

```

A8_readBD	<i>Daily or intradaily values of global horizontal irradiation and ambient temperature from a local file or a data.frame.</i>
-----------	---

Description

Constructor for the class `Meteo` with values of *daily* or *intradaily* values of global horizontal irradiation and ambient temperature from a local file or a `data.frame`.

Usage

```

readBD(file, lat,
        format='%d/%m/%Y',
        header=TRUE, fill=TRUE, dec='.', sep=';',
        dates.col='date', source=file)

```

```

readBDi(file, lat,
         format='%d/%m/%Y %H:%M:%S',
         header=TRUE, fill=TRUE, dec='.', sep=';',
         time.col='time',
         source=file)

```

```

df2Meteo(file, lat,
          format='%d/%m/%Y',
          dates.col='date',
          source='')

```

```

dfI2Meteo(file, lat,
           format='%d/%m/%Y %H:%M:%S',
           time.col='time',
           source='')

```

```

zoo2Meteo(file, lat, source='')

```

Arguments

`file` The name of the file (`readBD` and `readBDi`) or `data.frame` (`df2Meteo` `dfI2Meteo`) which the data are to be read from. It must contain a column `G0` with *daily* (`readBD` and `df2Meteo`) or *intradaily* (`readBDi` and `dfI2Meteo`) values of global horizontal irradiation (Wh/m^2). It should also include a column named `Ta` with values of ambient temperature.

If the `Meteo` object is to be used with `calcG0` (or `fCompD`, `fCompI`) and the option `corr='none'`, the file/`data.frame` **must** include three columns named `G0`, `B0` and `D0` with values of global, direct and diffuse irradiation on the horizontal plane.

Only for daily data: if the ambient temperature is not available, the file should include two columns named TempMax and TempMin with daily values of maximum and minimum ambient temperature, respectively (see [fTemp](#) for details).

header, fill, dec, sep	See read.table
format	character string with the format of the dates or time index. (Default for daily time bases: %d/%m/%Y). (Default for intradaily time bases: %d/%m/%Y %H:%M:%S)
lat	numeric, latitude (degrees) of the location.
dates.col	character string with the name of the column wich contains the dates of the time series.
time.col	character string with the name of the column wich contains the time index of the series.
source	character string with information about the source of the values. (Default: the name of the file).

Value

A Meteo object.

Author(s)

Oscar Perpiñán Lamigueiro.

See Also

[read.table](#), [readSIAR](#), [readG0dm](#).

Examples

```
data(helios)
names(helios)=c('date', 'G0', 'TempMax', 'TempMin')

bd=df2Meteo(helios, dates.col='date', lat=41, source='helios-IES', format='%Y/%m/%d')

summary(getData(bd))

xyplot(bd)
```

A8_readG0dm

Monthly mean values of global horizontal irradiation.

Description

Constructor for the class Meteo with 12 values of monthly means of irradiation.

Usage

```
readG0dm(G0dm, Ta=25, lat=0,
  year= as.POSIXlt(Sys.Date())$year+1900,
  promDays=c(17,14,15,15,15,10,18,18,18,19,18,13),
  source='')
```

Arguments

G0dm	numeric, 12 values of monthly means of daily global horizontal irradiation (Wh/m ²).
Ta	numeric, 12 values of monthly means of ambient temperature (degrees Celsius).
lat	numeric, latitude (degrees) of the location.
year	numeric (Default: current year).
promDays	numeric, set of the average days for each month.
source	character string with information about the source of the values.

Value

Meteo object

Author(s)

Oscar Perpiñán Lamigueiro.

See Also

[readSIAR](#), [readBD](#).

Examples

```
G0dm=c(2.766,3.491,4.494,5.912,6.989,7.742,7.919,7.027,5.369,3.562,2.814,2.179)*1000;
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
BD<-readG0dm(G0dm=G0dm, Ta=Ta, lat=37.2)
print(BD)
getData(BD)
xyplot(BD)
```

A8_readSIAR

Meteorological data from www.marm.es/siar

Description

Download and transform meteorological data from www.marm.es/siar for a station of a province (Spain) in a time period.

Usage

```
readSIAR(prov, est, start, end, lat=0, format='%d/%m/%Y')
readMAPA(prov, est, start, end, lat=0, format='%d/%m/%Y')
```

Arguments

prov	numeric, number of the spanish province according to the information of www.marm.es/siar (see details below).
est	numeric, number of the station according to the information of www.marm.es/siar (see details below).
start	character, first day of the time period.
end	character, last day of the time period.
lat	numeric, latitude (degrees) of the station.
format	character string with the format of the start and end dates. (Default: '%d/%m/%Y')

Details

The number codes of the stations and provinces are available at <http://solar.r-forge.r-project.org/data/SIAR.csv>, a data frame with 8 columns:

N_Estacion Code of the station
 Estacion Name of the station
 N_Provincia Code of the Province
 Provincia Name of the Province
 Comunidad Name of the Community (first-level political division of Spain).
 lon Longitude (degrees)
 lat Latitude (degrees)
 Altitud Altitude (meters)

Part of this information has been obtained with a "trial-and-error" procedure so could include mistakes.

The latitude, longitude and altitude information have been obtained in the context of a Master Project at the EOI (<https://sites.google.com/a/learning.eoi.es/merme2010-estudioradiacionsolar/home>) by Fernando Antoñanzas Torres, Federico Cañizares Jover, Rafael Morales Cabrera and Manuel Ojeda Fernández. This dataset is an improvement of the previous RedEstaciones (not available in this version).

Value

Meteo object, with the information available at www.marm.es/siar.

It should be noted that SIAR uses MJ/m² units for the irradiation data and readSIAR converts the values to Wh/m².

Note

The webpage of SIAR has changed. The meteorological data is no longer available at www.mapa.es/siar. Therefore, the name readMAPA has been deprecated and the function readSIAR should be used instead.

Author(s)

Oscar Perpiñán Lamigueiro.

References

www.marm.es/siar

See Also

[readG0dm](#), [readBD](#), [fTemp](#), [read.zoo](#)

Examples

```
## Not run:
#Aranjuez, Madrid
BD<-readSIAR(28,3,'01/01/2008','31/12/2008')

xyplot(TempMedia~G0|equal.count(VelViento),data=BD)

## End(Not run)

## Not run:
## Plot the stations in a map
library(sp)
library(maptools)

SIAR <- read.csv('http://solar.r-forge.r-project.org/data/SIAR.csv')
proj <- CRS('+proj=longlat +ellps=WGS84')
spSIAR <- SpatialPointsDataFrame(SIAR[, c(6, 7)], SIAR[, -c(6, 7)],
                                proj4str=proj)

###download a shapefile with the administrative borders of Spain
old <- setwd(tempdir())
download.file('http://www.gadm.org/data/shp/ESP_adm.zip', 'ESP_adm.zip')
unzip('ESP_adm.zip')
mapaSHP <- readShapLines('ESP_adm2.shp', proj4string=proj)
setwd(old)

p <- spplot(spSIAR['Comunidad'],
            col.regions=brewer.pal(n=12, 'Paired'),
            key.space='right', scales=list(draw=TRUE),
            type=c('p','g'))

p + layer(sp.lines(mapaSHP))
```

```
## End(Not run)
```

B1_Meteo-class	Class "Meteo"
----------------	---------------

Description

A class for meteorological data.

Objects from the Class

Objects can be created by the family of [readBD](#) functions.

Slots

latData: Latitude (degrees) of the meteorological station or source of the data.

data: A zoo object with the time series of daily irradiation (G_0 , Wh/m²), the ambient temperature (Ta) or the maximum and minimum ambient temperature (TempMax and TempMin).

source: A character with a short description of the source of the data.

type: A character, prom, bd, bdI or mapa, depending on the constructor.

Methods

getData signature(object = "Meteo"): extracts the data slot as a zoo object.

getG0 signature(object = "Meteo"): extracts the irradiation time series as a zoo object.

getLat signature(object = "Meteo"): extracts the latitude value.

indexD signature(object = "Meteo"): extracts the index of the data slot.

xyplot signature(x = "formula", data = "Meteo"): plot the content of the object according to the formula argument.

xyplot signature(x = "Meteo", data = "missing"): plot the data slot using the xyplot method for zoo objects.

Author(s)

Oscar Perpiñán Lamigueiro.

See Also

[readBD](#), [readBDi](#), [zoo2Meteo](#), [df2Meteo](#), [dfI2Meteo](#), [readG0dm](#), [readSIAR](#).

B2_Sol-class

*Class "Sol": Apparent movement of the Sun from the Earth***Description**

A class which describe the apparent movement of the Sun from the Earth.

Objects from the Class

Objects can be created by `calcSol`.

Slots

`lat`: numeric, latitude (degrees) as defined in the call to `calcSol`.

`solD`: Object of class "zoo" created by `fSolD`.

`solI`: Object of class "zoo" created by `fSolI`.

`match`: numeric, index of `solD` related with the index of `solI`.

`method`: character, method for the sun geometry calculations.

`sample`: `difftime`, increment of the intradaily sequence.

Methods

as.data.frameD signature(object = "Sol"): conversion to a data.frame with daily values.

as.data.frameI signature(object = "Sol"): conversion to a data.frame with intradaily values.

as.zooD signature(object = "Sol"): conversion to a zoo object with daily values.

as.zooI signature(object = "Sol"): conversion to a zoo object with intradaily values.

getLat signature(object = "Sol"): latitude (degrees) as defined in the call to `calcSol`.

indexD signature(object = "Sol"): index of the `solD` slot.

indexI signature(object = "Sol"): index of the `solI` object.

indexRep signature(object = "Sol"): accessor for the `match` slot.

xyplot signature(x = "formula", data = "Sol"): displays the contents of a `Sol` object with the `xyplot` method for formulas.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

`GO`, `Gef`.

 B3_G0-class

 Class "G0": irradiation and irradiance on the horizontal plane.

Description

This class contains the global, diffuse and direct irradiation and irradiance on the horizontal plane, and ambient temperature.

Objects from the Class

Objects can be created by the function [calcG0](#).

Slots

G0D: Object of class "zoo" created by [fCompD](#). It includes daily values of:

Fd: numeric, the diffuse fraction

Ktd: numeric, the clearness index

G0d: numeric, the global irradiation on a horizontal surface (Wh/m²)

D0d: numeric, the diffuse irradiation on a horizontal surface (Wh/m²)

B0d: numeric, the direct irradiation on a horizontal surface (Wh/m²)

G0I: Object of class "zoo" created by [fCompI](#). It includes values of:

kt: numeric, clearness index

G0: numeric, global irradiance on a horizontal surface, (W/m²)

D0: numeric, diffuse irradiance on a horizontal surface, (W/m²)

B0: numeric, direct irradiance on a horizontal surface, (W/m²)

G0dm: Object of class "zoo" with monthly mean values of daily irradiation.

G0y: Object of class "zoo" with yearly sums of irradiation.

Ta: Object of class "zoo" with intradaily ambient temperature values.

Besides, this class contains the slots from the [Sol](#) and [Meteo](#) classes.

Extends

Class "[Meteo](#)", directly. Class "[Sol](#)", directly.

Methods

as.zooD signature(object = "G0"): conversion to a zoo object with daily values.

as.zooI signature(object = "G0"): conversion to a zoo object with intradaily values.

as.zooM signature(object = "G0"): conversion to a zoo object with monthly values.

as.zooY signature(object = "G0"): conversion to a zoo object with yearly values.

as.data.frameD signature(object = "G0"): conversion to a data.frame with daily values.

as.data.frameI signature(object = "G0"): conversion to a data.frame with intradaily values.

as.data.frameM signature(object = "G0"): conversion to a data.frame with monthly values.

as.data.frameY signature(object = "G0"): conversion to a data.frame with yearly values.

indexD signature(object = "G0"): index of the solD slot.

indexI signature(object = "G0"): index of the solI object.

indexRep signature(object = "G0"): accesor for the match slot.

getLat signature(object = "G0"): latitude of the inherited [Sol](#) object.

xyplot signature(x = "G0", data = "missing"): display the time series of daily values of irradiation.

xyplot signature(x = "formula", data = "G0"): displays the contents of a G0 object with the xyplot method for formulas.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[Sol](#), [Gef](#).

B4_Gef-class

Class "Gef": irradiation and irradiance on the generator plane.

Description

This class contains the global, diffuse and direct irradiation and irradiance on the horizontal plane, and ambient temperature.

Objects from the Class

Objects can be created by the function [calcGef](#).

Slots

GefI: Object of class "zoo" created by [fInclin](#). It contains these components:

Bo: Extra-atmospheric irradiance on the inclined surface (W/m²)

Bn: Direct normal irradiance (W/m²)

G, B, D, Di, Dc, R: Global, direct, diffuse (total, isotropic and anisotropic) and albedo irradiance incident on an inclined surface (W/m²)

Gef, Bef, Def, Dief, Dcef, Ref: Effective global, direct, diffuse (total, isotropic and anisotropic) and albedo irradiance incident on an inclined surface (W/m²)

FTb, FTd, FTTr: Factor of angular losses for the direct, diffuse and albedo components

GefD: Object of class "zoo" with daily values of global, diffuse and direct irradiation.

Gefdm: Object of class "zoo" with monthly means of daily global, diffuse and direct irradiation.

Gefy: Object of class "zoo" with yearly sums of global, diffuse and direct irradiation.

Theta: Object of class "zoo" created by [fTheta](#). It contains these components:

Beta: numeric, inclination angle of the surface (radians). When modeTrk='fixed' it is the value of the argument beta converted from degree to radians.

Alfa: numeric, azimuth angle of the surface (radians). When modeTrk='fixed' it is the value of the argument alfa converted from degree to radians.

cosTheta: numeric, cosine of the incidence angle of the solar irradiance on the surface

iS: numeric, degree of dirtiness.

alb: numeric, albedo reflection coefficient.

modeTrk: character, mode of tracking.

modeShd: character, mode of shadows.

angGen: A list with the values of alfa, beta and betaLim.

struct: A list with the dimensions of the structure.

distances: A data.frame with the distances between structures.

Extends

Class "G0", directly. Class "Meteo", by class "G0", distance 2. Class "Sol", by class "G0", distance 2.

Methods

as.zooD signature(object = "Gef"): conversion to a zoo object with daily values.

as.zooI signature(object = "Gef"): conversion to a zoo object with intradaily values.

as.zooM signature(object = "Gef"): conversion to a zoo object with monthly values.

as.zooY signature(object = "Gef"): conversion to a zoo object with yearly values.

as.data.frameD signature(object = "Gef"): conversion to a data.frame with daily values.

as.data.frameI signature(object = "Gef"): conversion to a data.frame with intradaily values.

as.data.frameM signature(object = "Gef"): conversion to a data.frame with monthly values.

as.data.frameY signature(object = "Gef"): conversion to a data.frame with yearly values.

indexD signature(object = "Gef"): index of the solD slot.

indexI signature(object = "Gef"): index of the solI object.

indexRep signature(object = "Gef"): accessor for the match slot.

getLat signature(object = "Gef"): latitude of the inherited Sol object.

xyplot signature(x = "Gef", data = "missing"): display the time series of daily values of irradiation.

xyplot signature(x = "formula", data = "Gef"): displays the contents of a Gef object with the xyplot method for formulas.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[Sol](#), [G0](#).

B5_ProdGCPV-class *Class "ProdGCPV": performance of a grid connected PV system.*

Description

A class containing values of the performance of a grid connected PV system.

Objects from the Class

Objects can be created by [prodGCPV](#).

Slots

prodI: Object of class "zoo" created by [fProd](#). It includes these components:

Tc: cell temperature, °C.

Voc, Isc, Vmpp, Impp: open circuit voltage, short circuit current, MPP voltage and current, respectively.

Vdc, Idc: voltage and current at the input of the inverter.

Pdc: power at the input of the inverter, W

Pac: power at the output of the inverter, W

Effl: efficiency of the inverter

prodD: A zoo object with daily values of AC (Eac) and DC (Edc) energy (Wh), and productivity (Yf, Wh/Wp) of the system.

prodM: A zoo object with monthly means of daily values of AC and DC energy (kWh), and productivity of the system.

prody: A zoo object with yearly sums of AC and DC energy (kWh), and productivity of the system.

module: A list with the characteristics of the module.

generator: A list with the characteristics of the PV generator.

inverter: A list with the characteristics of the inverter.

effSys: A list with the efficiency values of the system.

Besides, this class contains the slots from the "[Meteo](#)", "[Sol](#)", "[G0](#)" and "[Gef](#)" classes.

Extends

Class "**Gef**", directly. Class "**G0**", by class "Gef", distance 2. Class "**Meteo**", by class "Gef", distance 3. Class "**Sol**", by class "Gef", distance 3.

Methods

as.zooD signature(object = "ProdGCPV"): conversion to a zoo object with daily values.

as.zooI signature(object = "ProdGCPV"): conversion to a zoo object with intradaily values.

as.zooM signature(object = "ProdGCPV"): conversion to a zoo object with monthly values.

as.zooY signature(object = "ProdGCPV"): conversion to a zoo object with yearly values.

as.data.frameD signature(object = "ProdGCPV"): conversion to a data.frame with daily values.

as.data.frameI signature(object = "ProdGCPV"): conversion to a data.frame with intradaily values.

as.data.frameM signature(object = "ProdGCPV"): conversion to a data.frame with monthly values.

as.data.frameY signature(object = "ProdGCPV"): conversion to a data.frame with yearly values.

indexD signature(object = "ProdGCPV"): index of the solD slot.

indexI signature(object = "ProdGCPV"): index of the solI object.

indexRep signature(object = "ProdGCPV"): accessor for the match slot.

getLat signature(object = "ProdGCPV"): latitude of the inherited **Sol** object.

xyplot signature(x = "ProdGCPV", data = "missing"): display the time series of daily values.

xyplot signature(x = "formula", data = "ProdGCPV"): displays the contents of a ProdGCPV object with the xyplot method for formulas.

as.zooD signature(object = "ProdGCPV"): conversion to a zoo object with daily values.

as.zooI signature(object = "ProdGCPV"): conversion to a zoo object with intradaily values.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

Sol, **G0**, **Gef**, **Shade**.

B6_ProdPVPS-class *Class "ProdPVPS": performance of a PV pumping system.*

Description

Performance of a PV pumping system with a centrifugal pump and a variable frequency converter.

Objects from the Class

Objects can be created by [prodPVPS](#).

Slots

prodI: Object of class "zoo" with these components:

Q: Flow rate, (m³/h)

Pb, Ph: Pump shaft power and hydraulic power (W), respectively.

etam, etab: Motor and pump efficiency, respectively.

f: Frequency (Hz)

prodD: A zoo object with daily values of AC energy (Wh), flow (m³) and productivity of the system.

prodDm: A zoo object with monthly means of daily values of AC energy (kWh), flow (m³) and productivity of the system.

prody: A zoo object with yearly sums of AC energy (kWh), flow (m³) and productivity of the system.

pump A list extracted from [pumpCoef](#)

H Total manometric head (m)

Pg Nominal power of the PV generator (Wp)

converter list containing the nominal power of the frequency converter, Pnom, and Ki, vector of three values, coefficients of the efficiency curve.

effSys list of numeric values with information about the system losses

Besides, this class contains the slots from the [Gef](#) class.

Extends

Class "[Gef](#)", directly. Class "[GO](#)", by class "Gef", distance 2. Class "[Meteo](#)", by class "Gef", distance 3. Class "[Sol](#)", by class "Gef", distance 3.

Methods

as.zooD signature(object = "ProdPVPS"): conversion to a zoo object with daily values.

as.zooI signature(object = "ProdPVPS"): conversion to a zoo object with intradaily values.

as.zooM signature(object = "ProdPVPS"): conversion to a zoo object with monthly values.

as.zooY signature(object = "ProdPVPS"): conversion to a zoo object with yearly values.

- as.data.frameD** signature(object = "ProdPVPS"): conversion to a data.frame with daily values.
- as.data.frameI** signature(object = "ProdPVPS"): conversion to a data.frame with intradaily values.
- as.data.frameM** signature(object = "ProdPVPS"): conversion to a data.frame with monthly values.
- as.data.frameY** signature(object = "ProdPVPS"): conversion to a data.frame with yearly values.
- indexD** signature(object = "ProdPVPS"): index of the solD slot.
- indexI** signature(object = "ProdPVPS"): index of the solI object.
- indexRep** signature(object = "ProdPVPS"): accessor for the match slot.
- getLat** signature(object = "ProdPVPS"): latitude of the inherited `Sol` object.
- xyplot** signature(x = "ProdPVPS", data = "missing"): display the time series of daily values.
- xyplot** signature(x = "formula", data = "ProdPVPS"): displays the contents of a ProdPVPS object with the xyplot method for formulas.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Abella, M. A., Lorenzo, E. y Chenlo, F.: PV water pumping systems based on standard frequency converters. *Progress in Photovoltaics: Research and Applications*, 11(3):179–191, 2003, ISSN 1099-159X.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

`prodPVPS`, `fPump`.

B7_Shade-class

Class "Shade": shadows in a PV system.

Description

A class for the optimization of shadows in a PV system.

Objects from the Class

Objects can be created by `optimShd`.

Slots

FS: numeric, shadows factor values for each combination of distances.

GRR: numeric, Ground Requirement Ratio for each combination.

Yf: numeric, final productivity for each combination.

FS.loess: A local fitting of FS with loess.

Yf.loess: A local fitting of Yf with loess.

modeShd: character, mode of shadows.

struct: A list with the dimensions of the structure.

distances: A data.frame with the distances between structures.

res numeric, difference (meters) between the different steps of the calculation.

Besides, as a reference, this class includes a [ProdGCPV](#) object with the performance of a PV systems without shadows.

Extends

Class "[ProdGCPV](#)", directly. Class "[Gef](#)", by class "[ProdGCPV](#)", distance 2. Class "[G0](#)", by class "[ProdGCPV](#)", distance 3. Class "[Meteo](#)", by class "[ProdGCPV](#)", distance 4. Class "[Sol](#)", by class "[ProdGCPV](#)", distance 4.

Methods

as.data.frame signature(x = "Shade"): conversion to a data.frame including columns for distances (Lew, Lns, and D) and results (FS, GRR and Yf).

shadeplot signature(x = "Shade"): display the results of the iteration with a level plot for the two-axis tracking, or with conventional plot for horizontal tracking and fixed systems.

xyplot signature(x = "formula", data = "Shade"): display the content of the Shade object with the xyplot method for formulas.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Perpiñán, O.: Grandes Centrales Fotovoltaicas: producción, seguimiento y ciclo de vida. PhD Thesis, UNED, 2008. <http://e-spacio.uned.es/fez/view.php?pid=bibliuned:20080>.
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[Gef](#), [ProdGCPV](#).

C_corrFdKt	<i>Correlations between the fraction of diffuse irradiation and the clearness index.</i>
------------	--

Description

A set of correlations between the fraction of diffuse irradiation and the clearness index used by [fCompD](#) and [fCompI](#).

Usage

```
## Monthly means of daily values
FdKtPage(Ktd)
FdKtLJ(Ktd)

## Daily values
FdKtCPR(Ktd)
FdKtEKDd(Ktd, sol)
FdKtCLIMEDd(Ktd)

## Intradaily values
FdKtEKDh(kt)
FdKtCLIMEDh(kt)
FdKtBRL(kt, sol)
```

Arguments

Ktd	A numeric, the daily clearness index.
kt	A numeric, the intradaily clearness index.
sol	A Sol object provided by calcSol or a zoo object provided by fSolD or fSolI .

Value

A numeric, the diffuse fraction.

Author(s)

Oscar Perpiñán Lamigueiro; The BRL model was suggested by Kevin Ummel.

References

- Page, J. K., The calculation of monthly mean solar radiation for horizontal and inclined surfaces from sunshine records for latitudes 40N-40S. En U.N. Conference on New Sources of Energy, vol. 4, págs. 378–390, 1961.
- Collares-Pereira, M. y Rabl, A., The average distribution of solar radiation: correlations between diffuse and hemispherical and between daily and hourly insolation values. Solar Energy, 22:155–164, 1979.

- Erbs, D.G, Klein, S.A. and Duffie, J.A., Estimation of the diffuse radiation fraction for hourly, daily and monthly-average global radiation. *Solar Energy*, 28:293:302, 1982.
- De Miguel, A. et al., Diffuse solar irradiation model evaluation in the north mediterranean belt area, *Solar Energy*, 70:143-153, 2001.
- Ridley, B., Boland, J. and Lauret, P., Modelling of diffuse solar fraction with multiple predictors, *Renewable Energy*, 35:478-482, 2010.

See Also

[fCompD](#), [fCompI](#)

Examples

```
Ktd=seq(0, 1, .01)
Monthly=data.frame(Ktd=Ktd)
Monthly$Page=FdKtPage(Ktd)
Monthly$LJ=FdKtLJ(Ktd)

xyplot(Page+LJ~Ktd, data=Monthly,
        type=c('l', 'g'), auto.key=list(space='right'))

Ktd=seq(0, 1, .01)
Daily=data.frame(Ktd=Ktd)
Daily$CPR=FdKtCPR(Ktd)
Daily$CLIMEDd=FdKtCLIMEDd(Ktd)

xyplot(CPR+CLIMEDd~Ktd, data=Daily,
        type=c('l', 'g'), auto.key=list(space='right'))
```

C_fBTd

Daily time base

Description

Construction of a daily time base for solar irradiation calculation

Usage

```
fBTd(mode = "prom",
      year=as.POSIXlt(Sys.Date())$year+1900,
      start=paste('01-01-',year,sep=''),
      end=paste('31-12-',year,sep=''),
      format='%d-%m-%Y')
```

Arguments

mode	character, controls the type of time base to be created. With mode='serie' the result is a daily time series from start to end. With mode='prom' only twelve days, one for each month, are included. During these 'average days' the declination angle is equal to the monthly mean of this angle.
year	which year is to be used for the time base when mode='prom'. Its default value is the current year.
start	first day of the time base for mode='serie'. Its default value is the first of January of the current year.
end	last day of the time base for mode='serie'. Its default value is the last day of December of the current year.
format	format of start and end.

Details

This function is commonly used inside fSoLD.

Value

This function returns a POSIXct object.

Author(s)

Oscar Perpiñán Lamigueiro

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

fSoLD, as.POSIXct, seq.POSIXt.

Examples

```
#Average days
fBTd(mode='prom')

#The day #100 of the year 2008
BTd=fBTd(mode='serie', year=2008)
BTd[100]
```

C_fCompD

*Components of daily global solar irradiation on a horizontal surface***Description**

Extract the diffuse and direct components from the daily global irradiation on a horizontal surface by means of regressions between the clearness index and the diffuse fraction parameters.

Usage

```
fCompD(sol, G0d, corr = "CPR", f)
```

Arguments

sol	A Sol object from calcSol or a zoo object from fSolD . Both of them include a component named Bo0d, which stands for the extra-atmospheric daily irradiation incident on a horizontal surface
G0d	A Meteo object from readG0dm , readBD , readSIAR , or a zoo object containing daily global irradiation (Wh/m ²) on a horizontal surface. See below for corr='none'.
corr	A character, the correlation between the the fraction of diffuse irradiation and the clearness index to be used. With this version several options are available, as described in corrFdKt . For example, the FdKtPage is selected with corr='Page' and the FdKtCPR with corr='CPR'. If corr='user' the use of a correlation defined by a function f is possible. If corr='none' the G0d object should include information about global, diffuse and direct daily irradiation with columns named G0d, D0d and B0d, respectively.
f	A function defining a correlation between the fraction of diffuse irradiation and the clearness index. It is only necessary when corr='user'

Value

A zoo object which includes:

Fd	numeric, the diffuse fraction
Ktd	numeric, the clearness index
G0d	numeric, the global irradiation on a horizontal surface (Wh/m ²)
D0d	numeric, the diffuse irradiation on a horizontal surface (Wh/m ²)
B0d	numeric, the direct irradiation on a horizontal surface (Wh/m ²)

Author(s)

Oscar Perpiñán Lamigueiro

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fCompI](#)

Examples

```
lat=37.2;
BTd=fBTd(mode='serie')

SoID<-fSoID(lat, BTd[100])

G0d=zoo(5000, index(SoID))
fCompD(SoID, G0d, corr = "Page")
fCompD(SoID, G0d, corr = "CPR")

#define a function fKtd with the correlation of CPR
fKTd=function(x){(0.99*(x<=0.17))+
                 (x>0.17)*(1.188-2.272*x+9.473*x^2-21.856*x^3+14.648*x^4)}
#The same as with corr="CPR"
fCompD(SoID,G0d, corr="user",f=fKTd)

lat=-37.2;
SoIDs<-fSoID(lat, BTd[283])
G0d=zoo(5000, index(SoIDs))
fCompD(SoIDs, G0d, corr = "CPR")

lat=37.2;
G0dm=c(2.766,3.491,4.494,5.912,6.989,7.742,7.919,7.027,5.369,3.562,2.814,2.179)*1000;
Rad=readG0dm(G0dm, lat=lat)
soID<-fSoID(lat,fBTd(mode='prom'))
fCompD(soID, Rad, corr = 'Page')
```

C_fCompI

Calculation of solar irradiance on a horizontal surface

Description

From the daily global, diffuse and direct irradiation values supplied by fCompD, the profile of the global, diffuse and direct irradiance is calculated with the rd and rg components of fSoID.

Usage

```
fCompI(sol, compD, G0I, corr='none', f)
```

Arguments

sol	A Sol object as provided by calcSol or a zoo object as provided by fSolI .
compD	A zoo object as provided by fCompD . It is not considered if G0I is provided.
G0I	A Meteo object from readBDi , dfI2Meteo or zoo2Meteo , or a zoo object containing <i>intradaily</i> global irradiation (Wh/m ²) on a horizontal surface. See below for <code>corr='none'</code> .
corr	A character, the correlation between the the fraction of intradaily diffuse irradiation and the clearness index to be used. With this version several options are available, as described in corrFdKt . You should choose one of <i>intradaily</i> proposals. For example, the FdKtCLIMEDh is selected with <code>corr='CLIMEDh'</code> . If <code>corr='user'</code> the use of a correlation defined by a function <i>f</i> is possible. If <code>corr='none'</code> the G0I object should include information about global, diffuse and direct intradaily irradiation with columns named G0, D0 and B0, respectively.
f	A function defining a correlation between the fraction of diffuse irradiation and the clearness index. It is only necessary when <code>corr='user'</code>

Value

A zoo with these components:

kt	numeric, clearness index.
fd	numeric, diffuse fraction.
G0	numeric, global irradiance on a horizontal surface, (W/m ²)
D0	numeric, diffuse irradiance on a horizontal surface, (W/m ²)
B0	numeric, direct irradiance on a horizontal surface, (W/m ²)

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Collares-Pereira, M. y Rabl, A., The average distribution of solar radiation: correlations between diffuse and hemispherical and between daily and hourly insolation values. *Solar Energy*, 22:155–164, 1979.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fCompD](#), [fSolI](#), [calcSol](#), [corrFdKt](#).

Examples

```

lat=37.2;
BTd=fBTd(mode='serie')

solD<-fSoID(lat, BTd[100])
solI<-fSoII(solD, sample='hour')
G0d=zoo(5000, index(solD))
compD<-fCompD(solD, G0d, corr = "Page")
fCompI(solI, compD)

lat=37.2;
sol<-calcSol(lat, fBTd(mode='prom'), sample='hour', keep.night=FALSE)

G0dm=c(2.766,3.491,4.494,5.912,6.989,7.742,7.919,7.027,5.369,3.562,2.814,2.179)*1000;
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
BD<-readG0dm(G0dm=G0dm, Ta=Ta, lat=37.2)
compD<-fCompD(sol, BD, corr = 'Page')
compI<-fCompI(sol, compD)
head(compI)

```

C_fInclin

*Solar irradiance on an inclined surface***Description**

The solar irradiance incident on an inclined surface is calculated from the direct and diffuse irradiance on a horizontal surface, and from the evolution of the angles of the Sun and the surface. Moreover, the effect of the angle of incidence and dust on the PV module is included to obtain the effective irradiance.

This function is used by the [calcGef](#) function.

Usage

```
fInclin(compI, angGen, iS = 2, alb = 0.2, horizBright=TRUE, HCPV=FALSE)
```

Arguments

compI	A G0 object. It may be the result of calcG0 .
angGen	A zoo object, including at least three variables named Beta, Alfa and cosTheta. It may be the result of fTheta .
iS	integer, degree of dirtiness. Its value must be included in the set (1,2,3,4). iS=1 corresponds to a clean surface while iS=4 is the choice for a dirty surface. Its default value is 2
alb	numeric, albedo reflection coefficient. Its default value is 0.2
horizBright	logical, if TRUE, the horizon brightness correction proposed by Reind et al. is used.

HCPV logical, if TRUE the diffuse and albedo components of the *effective* irradiance are set to zero. HCPV is the acronym of High Concentration PV system.

Details

The solar irradiance incident on an inclined surface can be calculated from the direct and diffuse irradiance on a horizontal surface, and from the evolution of the angles of the Sun and the surface. The transformation of the direct radiation is straightforward since only geometric considerations are needed. However, the treatment of the diffuse irradiance is more complex since it involves the modelling of the atmosphere. There are several models for the estimation of diffuse irradiance on an inclined surface. The one which combines simplicity and acceptable results is the proposal of Hay and McKay. This model divides the diffuse component in isotropic and anisotropic whose values depends on a anisotropy index. On the other hand, the effective irradiance, the fraction of the incident irradiance that reaches the cells inside a PV module, is calculated with the losses due to the angle of incidence and dirtiness. This behaviour can be simulated with a model proposed by Martin and Ruiz requiring information about the angles of the surface and the level of dirtiness (iS).

Value

A zoo object with these components:

Bo	Extra-atmospheric irradiance on the inclined surface (W/m ²)
Bn	Direct normal irradiance (W/m ²)
G, B, D, Di, Dc, R	Global, direct, diffuse (total, isotropic and anisotropic) and albedo irradiance incident on an inclined surface (W/m ²)
Gef, Bef, Def, Dief, Dcef, Ref	Effective global, direct, diffuse (total, isotropic and anisotropic) and albedo irradiance incident on an inclined surface (W/m ²)
FTb, FTd, FTr	Factor of angular losses for the direct, diffuse and albedo components

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Hay, J. E. and McKay, D. C.: Estimating Solar Irradiance on Inclined Surfaces: A Review and Assessment of Methodologies. *Int. J. Solar Energy*, (3):pp. 203, 1985.
- Martin, N. and Ruiz, J.M.: Calculation of the PV modules angular losses under field conditions by means of an analytical model. *Solar Energy Materials & Solar Cells*, 70:25–38, 2001.
- D. T. Reindl and W. A. Beckman and J. A. Duffie: Evaluation of hourly tilted surface radiation models, *Solar Energy*, 45:9-17, 1990.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fTheta](#), [fCompI](#), [calcGef](#).

C_fProd

Performance of a PV system

Description

Simulate the behaviour of a grid connected PV system under different conditions of irradiance and temperature. This function is used by the [prodGCPV](#) function.

Usage

```
fProd(inclin, module, generator, inverter, effSys)
```

Arguments

inclin	A Gef object, a zoo object or a data.frame. In case of being zoo or data.frame it must include a component named Gef (effective irradiance, W/m ²) and another named Ta (ambient temperature, °C).
module	list of numeric values with information about the PV module, Vocn open-circuit voltage of the module at Standard Test Conditions (default value 57.6 volts.) Iscn short circuit current of the module at Standard Test Conditions (default value 4.7 amperes.) Vmn maximum power point voltage of the module at Standard Test Conditions (default value 46.08 amperes.) Imn Maximum power current of the module at Standard Test Conditions (default value 4.35 amperes.) Ncs number of cells in series inside the module (default value 96) Ncp number of cells in parallel inside the module (default value 1) CoefVT coefficient of decrement of voltage of each cell with the temperature (default value 0.0023 volts per celsius degree) TONC nominal operational cell temperature, celsius degree (default value 47).
generator	list of numeric values with information about the generator, Nms number of modules in series (default value 12) Nmp number of modules in parallel (default value 11)
inverter	list of numeric values with information about the DC/AC inverter, Ki vector of three values, coefficients of the efficiency curve of the inverter (default c(0.01, 0.025, 0.05)), or a matrix of nine values (3x3) if there is dependence with the voltage (see references). Pinv nominal inverter power (W) (default value 25000 watts.)

	Vmin, Vmax	minimum and maximum voltages of the MPP range of the inverter (default values 420 and 750 volts)
	Gumb	minimum irradiance for the inverter to start (W/m ²) (default value 20 W/m ²)
effSys		list of numeric values with information about the system losses,
	ModQual	average tolerance of the set of modules (%), default value is 3
	ModDisp	module parameter dispersion losses (%), default value is 2
	OhmDC	Joule losses due to the DC wiring (%), default value is 1.5
	OhmAC	Joule losses due to the AC wiring (%), default value is 1.5
	MPP	average error of the MPP algorithm of the inverter (%), default value is 1
	TrafoMT	losses due to the MT transformer (%), default value is 1
	Disp	losses due to stops of the system (%), default value is 0.5

Value

If `inclin` is zoo or Gef object, the result is a zoo object with these components (if `inclin` is a `data.frame` the result is also a `data.frame` with these same components):

Tc	cell temperature, °C.
Voc, Isc, Vmpp, Impp	open circuit voltage, short circuit current, MPP voltage and current, respectively, in the conditions of irradiance and temperature provided by <code>Inclin</code>
Vdc, Idc	voltage and current at the input of the inverter. If no voltage limitation occurs (according to the values of <code>inverter\$Vmax</code> and <code>inverter\$Vmin</code>), their values are identical to <code>Vmpp</code> and <code>Impp</code> . If the limit values are reached a warning is produced
Pdc	power at the input of the inverter, W
Pac	power at the output of the inverter, W
EffI	efficiency of the inverter

Author(s)

Oscar Perpiñán Lamigueiro

References

- Jantsch, M., Schmidt, H. y Schmid, J.: Results on the concerted action on power conditioning and control. 11th European photovoltaic Solar Energy Conference, 1992.
- Baumgartner, F. P., Schmidt, H., Burger, B., Bründlinger, R., Haeberlin, H. and Zehner, M.: Status and Relevance of the DC Voltage Dependency of the Inverter Efficiency. 22nd European Photovoltaic Solar Energy Conference, 2007.
- Alonso Garcia, M. C.: Caracterización y modelado de asociaciones de dispositivos fotovoltaicos. PhD Thesis, CIEMAT, 2005.
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fInclin](#), [prodGCPV](#), [fTemp](#).

Examples

```
inclin=data.frame(Gef=c(200,400,600,800,1000),Ta=25)

#using default values
fProd(inclin)

#Using a matrix for Ki (voltage dependence)
inv1=list(Ki=rbind(c(-0.00019917, 7.513e-06, -5.4183e-09),
c(0.00806, -4.161e-06, 2.859e-08),
c(0.02118, 3.4002e-05, -4.8967e-08)))

fProd(inclin, inverter=inv1)

#Voltage limits of the inverter
inclin=data.frame(Gef=800,Ta=30)
gen1 = list(Nms = 10, Nmp = 11)

prod=fProd(inclin,generator=gen1)
print(prod)

with(prod,Vdc*Idc/(Vmpp*Imp))
```

C_fPump

Performance of a centrifugal pump

Description

Compute the performance of the different parts of a centrifugal pump fed by a frequency converter following the affinity laws.

Usage

```
fPump(pump, H)
```

Arguments

pump	list containing the parameters of the pump to be simulated. It may be a row of pumpCoef .
H	Total manometric head (m).

Value

lim	Range of values of electrical power input
fQ	Function constructed with <code>splinefun</code> relating flow and electrical power
fPb	Function constructed with <code>splinefun</code> relating pump shaft power and electrical power of the motor
fPh	Function constructed with <code>splinefun</code> relating hydraulic power and electrical power of the motor
fFreq	Function constructed with <code>splinefun</code> relating frequency and electrical power of the motor

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Abella, M. A., Lorenzo, E. y Chenlo, F.: PV water pumping systems based on standard frequency converters. *Progress in Photovoltaics: Research and Applications*, 11(3):179–191, 2003, ISSN 1099-159X.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[NmgPVPS](#), [prodPVPS](#), [pumpCoef](#), [splinefun](#).

Examples

```
library(latticeExtra)

data(pumpCoef)
CoefSP8A44<-subset(pumpCoef, Qn==8&stages==44)

fSP8A44<-fPump(pump=CoefSP8A44,H=40)
SP8A44=with(fSP8A44,{
  Pac=seq(lim[1],lim[2],by=100)
  Pb=fPb(Pac)
  etam=Pb/Pac
  Ph=fPh(Pac)
  etab=Ph/Pb
  f=fFreq(Pac)
  Q=fQ(Pac)
  result=data.frame(Q,Pac,Pb,Ph,etam,etab,f)})

#Efficiency of the motor, pump and the motor-pump
SP8A44$etamb=with(SP8A44,etab*etam)
lab=c(expression(eta[motor]), expression(eta[pump]), expression(eta[mp]))
p<-xyplot(etam+etab+etamb~Pac,data=SP8A44,type='l', ylab='Efficiency')
p+glayer(panel.text(x[1], y[1], lab[group.number], pos=3))
```

```
#Mechanical, hydraulic and electrical power
lab=c(expression(P[pump]), expression(P[hyd]))
p<-xyplot(Pb+Ph~Pac,data=SP8A44,type='l', ylab='Power (W)', xlab='AC Power (W)')
p+glayer(panel.text(x[length(x)], y[length(x)], lab[group.number], pos=3))

#Flow and electrical power
xyplot(Q~Pac,data=SP8A44,type='l')
```

C_fSoID

*Daily apparent movement of the Sun from the Earth***Description**

Compute the daily apparent movement of the Sun from the Earth. This movement is mainly described (for the simulation of photovoltaic systems) by the declination angle, the sunset angle and the daily extra-atmospheric irradiation.

Usage

```
fSoID(lat, BTd, method='michalsky')
```

Arguments

lat	Latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.
BTd	Daily temporal base, a POSIXct object which may be the result of <code>fBTd</code> .
method	character, method for the sun geometry calculations to be chosen from 'cooper', 'spencer', 'michalsky' and 'strous'. See references for details.

Value

A zoo object with these components:

decl	Declination angle (radians) for each day of year in dn or BTd
eo	Factor of correction due the eccentricity of orbit of the Earth around the Sun.
ws	Sunset angle (in radians) for each day of year. Due to the convention which considers that the solar hour angle is negative before midday, this angle is negative.
Bo0d	Extra-atmospheric daily irradiation (watt-hour per squared meter) incident on a horizontal surface
EoT	Equation of Time.

Note

The latitude is stored as the attribute `lat` of the result, and thus it is accessible with `attr(object, 'lat')`.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Cooper, P.I., Solar Energy, 12, 3 (1969). "The Absorption of Solar Radiation in Solar Stills"
- Spencer, Search 2 (5), 172, <http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html>
- Strous: <http://www.astro.uu.nl/~strous/AA/en/reken/zonpositie.html>
- Michalsky, J., 1988: The Astronomical Almanac's algorithm for approximate solar position (1950-2050), Solar Energy 40, 227-235, ftp://climate1.gsfc.nasa.gov/wiscombe/Solar_Rad/SunAngles/sunae.f
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. <http://procomun.wordpress.com/documentos/libroesf/>

Examples

```
BTd=fBTd(mode='serie')

lat=37.2
fSolD(lat,BTd[100])
fSolD(lat,BTd[100], method='strous')
fSolD(lat,BTd[100], method='spencer')
fSolD(lat,BTd[100], method='cooper')

lat=-37.2
fSolD(lat,BTd[283])

#Solar angles along the year
SolD<-fSolD(lat,BTd=fBTd())

library(lattice)
xyplot(SolD)

#Calculation of the daylength for several latitudes
library(latticeExtra)

Lats=c(-60,-40,-20,0,20,40,60)
NomLats=ifelse(Lats>0, paste(Lats,'N', sep=''), paste(abs(Lats), 'S',
sep=''))
NomLats[Lats==0]='0'

mat=matrix(nrow=365, ncol=length(Lats))
colnames(mat)=NomLats
WsZ=zoo(mat, fBTd(mode='serie'))

for (i in seq_along(Lats)){
  SolDaux<-fSolD(lat=Lats[i],BTd=fBTd(mode='serie'));
  WsZ[,i]<-r2h(2*abs(SolDaux$ws))}

```

```
p=xyplot(WsZ, superpose=TRUE,
          ylab=expression(omega[s] (h)), auto.key=FALSE)
plab<-p+glayer(panel.text(x[1], y[1], NomLats[group.number], pos=2))
print(plab)
```

C_fSolI

*Instantaneous apparent movement of the Sun from the Earth***Description**

Compute the angles which describe the intradaily apparent movement of the Sun from the Earth.

Usage

```
fSolI(sold, sample='hour', BTi, EoT=TRUE, keep.night=TRUE, method='michalsky')
```

Arguments

sold	A zoo object with the result of fSolD
sample	Increment of the intradaily sequence. It is a character string, containing one of "sec", "min", "hour". This can optionally be preceded by a (positive or negative) integer and a space, or followed by "s". It is used by seq.POSIXt . It is not considered when BTi is provided.
BTi	Intradaily time base, a POSIXct object. It could be the index of the GOI argument to calcGO . fSolI will produce results only for those days contained both in sold and in BTi.
EoT	logical, if TRUE (default) the Equation of Time is used.
keep.night	logical, if TRUE (default) the night is included in the time series.
method	character, method for the sun geometry calculations to be chosen from 'cooper', 'spencer', 'michalsky' and 'strous'. See references for details.

Value

A zoo object is returned with these components:

w	numeric, solar hour angle (radians)
aman	logical, TRUE when Sun is above the horizon
cosThzS	numeric, cosine of the solar zenith angle
AzS	numeric, solar acimuth angle (radians)
AlS	numeric, solar elevation angle (radians)
Bo0	numeric, extra-atmospheric irradiance (W/m2)
rd, rg	numeric, relation between irradiance and irradiation of diffuse and global values, respectively, following the correlations proposed by Collares-Pereira and Rabl

The latitude is stored as the attribute lat of this object.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Cooper, P.I., Solar Energy, 12, 3 (1969). "The Absorption of Solar Radiation in Solar Stills"
- Spencer, Search 2 (5), 172, <http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html>
- Strous: <http://www.astro.uu.nl/~strous/AA/en/reken/zonpositie.html>
- Michalsky, J., 1988: The Astronomical Almanac's algorithm for approximate solar position (1950-2050), Solar Energy 40, 227-235, ftp://climate1.gsfc.nasa.gov/wiscombe/Solar_Rad/SunAngles/sunae.f
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. <http://procomun.wordpress.com/documentos/libroesf/>
- Collares-Pereira, M. y Rabl, A., The average distribution of solar radiation: correlations between diffuse and hemispherical and between daily and hourly insolation values. Solar Energy, 22:155–164, 1979.

See Also

[fSolD](#)

Examples

```
###Angles for one day
BTd=fBTd(mode='serie')

#North hemisphere
lat=37.2
sold<-fSold(lat,BTd[100])
solI<-fSolI(sold, sample='hour')
print(solI)

#South hemisphere
lat=-37.2;
solds<-fSold(lat,BTd[283])
solIs<-fSolI(sold, sample='hour')
print(solIs)

###Angles for the 12 average days
lat=37.2;
sold<-fSold(lat,BTd=fBTd(mode='prom'))
solI<-fSolI(sold, sample='10 min', keep.night=FALSE)

library(lattice)
library(latticeExtra)

###Solar elevation angle vs. azimuth.
#This kind of graphics is useful for shadows calculations
```

```

mon=month.abb
p<-xyplot(r2d(ALS)~r2d(AzS),
  groups=month,
  data=solI, type='l', col='black',
  xlab=expression(psi[s]),ylab=expression(gamma[s]))

plab<-p + glayer({
  idx <- round(length(x)/2+1)
  panel.text(x[idx], y[idx], mon[group.value], pos=3, offset=0.2, cex=0.8)})

print(plab)

```

C_fSombra

Shadows on PV systems

Description

Compute the shadows factor for two-axis and horizontal N-S axis trackers and fixed surfaces.

Usage

```
fSombra(angGen, distances, struct, modeTrk='fixed',prom=TRUE)
```

```
fSombra6(angGen,distances,struct,prom=TRUE)
```

```
fSombra2X(angGen,distances,struct)
```

```
fSombraHoriz(angGen, distances,struct)
```

```
fSombraEst(angGen, distances,struct)
```

Arguments

angGen	A zoo object, including at least variables named Beta, Alfa, AzS, ALS and cosTheta.
distances	data.frame, with a component named Lew, being the distance (meters) between horizontal NS and two-axis trackers along the East-West direction, a component named Lns for two-axis trackers or a component named D for static surfaces. An additional component named H can be included with the relative height (meters) between surfaces. When modeTrk='two' (or when fSombra6 is used) this data.frame may have five rows. Each of these rows defines the distances of a tracker in a set of six ones.
struct	list. When modeTrk='fixed' or modeTrk='horiz' only a component named L, which is the height (meters) of the tracker, is needed. For two-axis trackers (modeTrk='two'), an additional component named W, the width of the tracker,

is required. Moreover, two components named `Nrow` and `Ncol` are included under this list. These components define, respectively, the number of rows and columns of the whole set of trackers in the PV plant.

<code>modeTrk</code>	character, to be chosen from 'fixed', 'two' or 'horiz'. When <code>modeTrk='fixed'</code> the surface is fixed (inclination and azimuth angles are constant). The performance of a two-axis tracker is calculated with <code>modeTrk='two'</code> , and <code>modeTrk='horiz'</code> is the option for an horizontal N-S tracker. Its default value is <code>modeTrk='fixed'</code>
<code>prom</code>	logical, only needed for two-axis tracker mode. If TRUE the shadows are averaged between the set of trackers defined by <code>struct\$Nrow</code> and <code>struct\$Ncol</code>

Details

`fSombra` is only a wrapper for `fSombra6` (two-axis trackers), `fSombraEst` (fixed systems) and `fSombraHoriz` (horizontal N-S axis trackers). Depending on the value of `modeTrk` the corresponding function is selected. `fSombra6` calculates the shadows factor in a set of six two-axis trackers. If `distances` has only one row, this function constructs a symmetric grid around a tracker located at (0,0,0). These five trackers are located at (-Lew, Lns, H), (0, Lns, H), (Lew, Lns, H), (-Lew, 0, H) and (Lns, 0, H). It is possible to define a irregular grid around (0,0,0) including five rows in `distances`. When `prom=TRUE` the shadows factor for each of the six trackers is calculated. Then, according to the distribution of trackers in the plant defined by `struct$Nrow` and `struct$Ncol`, a weighted average of the shadows factors is the result. It is important to note that the `distances` are defined between axis for trackers and between similar points of the structure for fixed surfaces.

Value

`data.frame` including `angGen` and a variable named `FS`, which is the shadows factor. This factor is the ratio between the area of the generator affected by shadows and the total area. Therefore its value is 1 when the PV generator is completely shadowed.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Perpiñán, O.: Grandes Centrales Fotovoltaicas: producción, seguimiento y ciclo de vida. PhD Thesis, UNED, 2008. <http://e-spacio.uned.es/fez/view.php?id=bibliuned:20080>.
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[calcShd](#), [optimShd](#), [fTheta](#), [calcSol](#)

Examples

```
lat=37.2;
sol<-calcSol(lat, fBTd(mode='prom'), sample='10 min', keep.night=FALSE)
angGen<-fTheta(sol, beta=35);
```

```

Angles=CBIND(as.zooI(sol), angGen)

###Two-axis tracker
#Symmetric grid
distances=data.frame(Lew=40,Lns=30,H=0)
struct=list(W=23.11, L=9.8, Nrow=2, Ncol=8)

ShdFactor<-fSombra6(Angles, distances, struct, prom=FALSE)

Angles$FS=ShdFactor
xyplot(FS~w, groups=month, data=Angles,
       type='l',auto.key=list(space='right', lines=TRUE, points=FALSE))

#Symmetric grid defined with a five rows data.frame
distances=data.frame(Lew=c(-40,0,40,-40,40),Lns=c(30,30,30,0,0),H=0)
ShdFactor2<-fSombra6(Angles, distances, struct,prom=FALSE)

#of course, with the same result
identical(coredata(ShdFactor), coredata(ShdFactor2))

```

C_fTemp

Intradaily evolution of ambient temperature

Description

From the maximum and minimum daily values of ambient temperature, its evolution its calculated through a combination of cosine functions (ESRA method)

Usage

```
fTemp(sol, BD)
```

Arguments

sol	A Sol object. It may be the result of the calcSol function.
BD	A Meteo object, as provided by the readSIAR or readBD functions. It must include information about TempMax and TempMin.

Details

The ESRA method estimates the dependence of the temperature on the time of the day (given as the local solar time) from only two inputs: minimum and maximum daily temperatures. It assumes that the temperature daily profile can be described using three piecewise cosine functions, dividing the day into three periods: from midnight to sunrise, from sunrise to the time of peak temperature (3 hours after midday), and to midnight.

Value

A zoo object with the profile of the ambient temperature.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Huld, T. , Suri, M., Dunlop, E. D., and Micale F., Estimating average daytime and daily temperature profiles within Europe, *Environmental Modelling & Software* 21 (2006) 1650-1661.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[calcSol](#), [readSIAR](#), [readBD](#).

Examples

```
## Not run:
#Aranjuez, Madrid
BD<-readSIAR(28,3,'01/01/2008','31/12/2008')
lat=41;
sol=calcSol(lat, BTd=indexD(BD), sample='hour')
Temp<-fTemp(sol,BD)

###Temperature of March
library(latticeExtra)
wTemp=window(Temp, start=as.POSIXct('2008-03-01'), end=as.POSIXct('2008-03-31'))
xyplot(wTemp)+layer_(panel.xblocks(x, DoY, col=c('lightgray', 'white'))

## End(Not run)
```

C_fTheta

Angle of incidence of solar irradiation on a inclined surface

Description

The orientation, azimuth and incidence angle are calculated from the results of `fSolI` or `calcSol` and from the information supplied by the arguments `beta` and `alfa` when the surface is fixed (`modeTrk='fixed'`) or the movement equations when a tracking surface is chosen (`modeTrk='horiz'` or `modeTrk='two'`). Besides, the modified movement of a horizontal NS tracker due to the backtracking strategy is calculated if `BT=TRUE` with information about the tracker and the distance between the trackers included in the system.

This function is used by the `calcGef` function.

Usage

```
fTheta(sol, beta, alfa = 0, modeTrk = "fixed", betaLim = 90,
       BT = FALSE, struct, dist)
```

Arguments

sol	Sol object as provided by <code>calcSol</code> .
beta	numeric, inclination angle of the surface (degrees). It is only needed when <code>modeTrk='fixed'</code> .
alfa	numeric, azimuth angle of the surface (degrees). It is positive to the West. It is only needed when <code>modeTrk='fixed'</code> . Its default value is <code>alfa=0</code>
modeTrk	character, to be chosen from 'fixed', 'two' or 'horiz'. When <code>modeTrk='fixed'</code> the surface is fixed (inclination and azimuth angles are constant). The performance of a two-axis tracker is calculated with <code>modeTrk='two'</code> , and <code>modeTrk='horiz'</code> is the option for an horizontal N-S tracker. Its default value is <code>modeTrk='fixed'</code>
betaLim	numeric, maximum value of the inclination angle for a tracking surface. Its default value is 90 (no limitation)
BT	logical, TRUE when the backtracking technique is to be used with a horizontal NS tracker, as described by Panico et al. (see References). The default value is FALSE. In future versions of this package this technique will be available for two-axis trackers.
struct	Only needed when <code>BT=TRUE</code> . A list, with a component named L, which is the height (meters) of the tracker. In future versions the backtracking technique will be used in conjunction with two-axis trackers, and an additional component named W will be needed.
dist	Only needed when <code>BT=TRUE</code> . A data.frame, with a component named Lew, being the distance between the horizontal NS trackers along the East-West direction. In future versions an additional component named Lns will be needed for two-axis trackers with backtracking.

Value

A zoo object with these components:

Beta	numeric, inclination angle of the surface (radians). When <code>modeTrk='fixed'</code> it is the value of the argument <code>beta</code> converted from degrees to radians.
Alfa	numeric, azimuth angle of the surface (radians). When <code>modeTrk='fixed'</code> it is the value of the argument <code>alfa</code> converted from degrees to radians.
cosTheta	numeric, cosine of the incidence angle of the solar irradiance on the surface

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)
- Panico, D., Garvison, P., Wenger, H. J., Shugar, D., Backtracking: a novel strategy for tracking PV systems, Photovoltaic Specialists Conference, 668-673, 1991

See Also

[fInclin](#), [fSombra](#), [calcGef](#).

C_HQCurve

H-Q curves of a centrifugal pump

Description

Compute and display the H-Q curves of a centrifugal pump fed working at several frequencies, and the iso-efficiency curve as a reference.

Usage

HQCurve(pump)

Arguments

pump list containing the parameters of the pump to be simulated. It may be a row of [pumpCoef](#).

Value

result A data.frame with the result of the simulation. It contains several columns with values of manometric height (H), frequency (fe and fb), mechanical power (Pb), AC electrical power (Pm), DC electrical power (Pdc) and efficiency of the pump (etab) and motor (etam).

plot The plot with several curves labelled with the correspondent frequencies, and the isoefficiency curve (named "ISO").

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Abella, M. A., Lorenzo, E. y Chenlo, F.: PV water pumping systems based on standard frequency converters. Progress in Photovoltaics: Research and Applications, 11(3):179–191, 2003, ISSN 1099-159X.
- Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[NmgPVPS](#), [prodPVPS](#), [pumpCoef](#).

Examples

```

library(lattice)
library(latticeExtra)

data(pumpCoef)

CoefSP8A44<-subset(pumpCoef, Qn==8&stages==44)
CurvaSP8A44<-HQCurve(pump=CoefSP8A44)

```

C_local2Solar *Local time, mean solar time and UTC time zone.*

Description

The function `local2Solar` converts the time zone of a `POSIXct` object to the mean solar time and set its time zone to UTC as a synonym of mean solar time. It includes two corrections: the difference of longitudes between the location and the time zone, and the daylight saving time.

The function `CBIND` combines several objects (`zoo`, `data.frame` or `matrix`) preserving the index of the first of them or assigning a new one with the `index` argument.

The function `lonHH` calculates the longitude (radians) of a time zone.

Usage

```

local2Solar(x, lon = NULL)
CBIND(..., index=NULL)
lonHH(tz)

```

Arguments

<code>x</code>	a <code>POSIXct</code> object
<code>lon</code>	A numeric value of the longitude (degrees) of the location. If <code>lon=NULL</code> (default), this value is assumed to be equal to the longitude of the time zone of <code>x</code> , so only the daylight saving time correction (if needed) is included.
<code>...</code>	A set of <code>zoo</code> objects.
<code>index</code>	A <code>POSIXct</code> object, the index of <code>zoo</code> object constructed with <code>CBIND</code> .
<code>tz</code>	A character, a time zone.

Details

Since the result of `local2Solar` is the mean solar time, the Equation of Time correction is not calculated with this function. The `fSolI` function includes this correction if desired.

If the `index` argument of `CBIND` is `NULL` (default) the first object of `...` must be a `zoo` object.

Value

The function `local2Solar` produces a `POSIXct` object with its time zone set to UTC.

The function `CBIND` produces a zoo object.

The function `lonHH` gives a numeric value.

Note

It is important to note that the `solar` package sets the system time zone to UTC with `Sys.setenv(TZ='UTC')`. Every zoo object created by the package will have an index with this time zone and will be supposed to be mean solar time.

Author(s)

Oscar Perpiñán Lamigueiro.

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

Examples

```
t.local<-as.POSIXct("2006-01-08 10:07:52", tz='Europe/Madrid')
```

```
##The local time zone and the location have the same longitude (15 degrees)
```

```
local2Solar(t.local)
```

```
##But Madrid is at lon=-3
```

```
local2Solar(t.local, lon=-3)
```

```
##Daylight saving time
```

```
t.local.dst<-as.POSIXct("2006-07-08 10:07:52", tz='Europe/Madrid')
```

```
local2Solar(t.local.dst)
```

```
local2Solar(t.local.dst, lon=-3)
```

```
## Not run:
```

```
##Extracted from an example of calcG0
```

```
##NREL-MIDC
```

```
##La Ola, Lanai
```

```
##Latitude: 20.76685o North
```

```
##Longitude: 156.92291o West
```

```
##Time Zone: -10.0
```

```
file='http://www.nrel.gov/midc/apps/plot.pl?site=LANAI&start=20090722&edy=19&emo=11&eyr=2010&zenloc=19&year=20
```

```
dat <- read.table(file, header=TRUE, sep=',')
```

```
names(dat) <- c('date', 'hour', 'G0', 'B', 'D0', 'Ta')
```

```
##B is direct normal. We need direct horizontal.
```

```
dat$B0 <- dat$G0-dat$D0
```

```
##http://www.nrel.gov/midc/la_ola_lanai/instruments.html:
##The datalogger program runs using Greenwich Mean Time (GMT),
##data is converted to Hawaiiin Standard Time (HST) after data collection
idxLocal <- with(dat, as.POSIXct(paste(date, hour), format='%m/%d/%Y %H:%M', tz='HST'))
head(idxLocal)
idx <- local2Solar(idxLocal, lon=-156.9339)
head(idx)

## End(Not run)
```

C_NmgPVPS

Nomogram of a photovoltaic pumping system

Description

This function simulate the performance of a water pump fed by a frequency converter with several PV generators of different size during a day. The result is plotted as a nomogram which relates the nominal power of the PV generator, the total water flow and the total manometric head.

Usage

```
NmgPVPS(pump, Pg, H, Gd, Ta=30,
         lambda=0.0045, TONC=47, eta=0.95,
         Gmax=1200, t0=6, Nm=6,
         title='', theme=custom.theme.2())
```

Arguments

pump	A list extracted from pumpCoef
Pg	Sequence of values of the nominal power of the PV generator (Wp))
H	Sequence of values of the total manometric head (m)
Gd	Global irradiance incident on the generator (Wh/m ²)
Ta	Ambient temperature (°C).
lambda	Power losses factor due to temperature
TONC	Nominal operational cell temperature (°C).
eta	Average efficiency of the frequency converter
Gmax	Maximum value of irradiance (parameter of the IEC 61725)
t0	Hours from midday to sunset (parameter of the IEC 61725)
Nm	Number of samples per hour
title	Main title of the plot.
theme	Theme of the lattice plot.

Details

This function computes the irradiance profile according to the IEC 61725 "Analytical Expression for Daily Solar Profiles", which is a common reference in the official documents regarding PV pumping systems. At this version only pumps from the manufacturer Grundfos are included in [pumpCoef](#).

Value

I	list with the results of irradiance, power and flow of the system.
D	list with the results of total irradiation, electrical energy and flow for every nominal power of the generator.
param	list with the arguments used in the call to the function.
plot	trellis object containing the nomogram.

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Abella, M. A., Lorenzo, E. y Chenlo, F.: PV water pumping systems based on standard frequency converters. *Progress in Photovoltaics: Research and Applications*, 11(3):179–191, 2003, ISSN 1099-159X.
- Perpiñán, O, *Energía Solar Fotovoltaica*, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fPump](#), [prodPVPS](#), [pumpCoef](#)

Examples

```
Pg=seq(4000,8000,by=100);
H=seq(120,150,by=5);

data(pumpCoef)

CoefSP8A44<-subset(pumpCoef, Qn==8&stages==44)

NmgSP8A44<-NmgPVPS(pump=CoefSP8A44,Pg=Pg,H=H,Gd=5000,
  title='Choice of Pump', theme=custom.theme())
```

Description

diff2Hours converts a difftime object into its numeric value with units='hours'.

char2diff converts a character description into a difftime object, following the code of [seq.POSIXt](#).

sample2Hours calculates the sampling time in hours described by a character or a difftime.

P2E (power to energy) sums a series of power values (for example, irradiance) to obtain energy aggregation (for example, irradiation) using sample2Hours for the units conversion.

Usage

```
diff2Hours(by)
char2diff(by)
sample2Hours(by)
P2E(x, by)
```

Arguments

by	A character for char2diff, sample2Hours and P2E, or a difftime for diff2Hours, sample2Hours and P2E.
x	A numeric vector.

Value

A numeric value or a difftime object.

Author(s)

Oscar Perpiñán Lamigueiro

See Also

[Sol](#)

Examples

```
char2diff('min')
char2diff('2 s')

sample2Hours('s')
sample2Hours('30 m')

by1<-char2diff('10 min')
sample2Hours(by1)
```

Description

In a PV plant, the individual systems are theoretically identical and their performance along the time should be the same. Due to their practical differences –power tolerance, dispersion losses, dust–, the individual performance of each system will deviate from the average behaviour. However, when a system is performing correctly, these deviations are constrained inside a range and should not be regarded as sign of malfunctioning.

If these common deviations are assumed as a random process, a statistical analysis of the performance of the whole set of systems can identify a faulty system as the one that departs significantly from the mean behaviour.

These functions compare the daily performance of each system with a reference (for example, the median of the whole set) during a time period of N days preceding the current day. They calculate a set of statistics of the performance of the PV plant as a whole, and another set of the comparison with the reference. This statistical analysis can be summarised with a graphical tool named "Target Diagram", which plots together the root mean square difference, the average difference and the standard deviation of the difference. Besides, this diagram includes the sign of the difference of the standard deviations of the system and the reference.

Usage

```
analyzeData(x, ref=NULL)
TargetDiagram(x, end, ndays, ref=NULL, color=NULL, cex=0.8,...)
```

Arguments

x	A zoo object with several columns. This object represent the time evolution of a set of units (e.g. PV generators of a large system) which are supposed to show a similar behaviour.
ref	A zoo object to be used as the reference unit. If ref=NULL (default), the reference is the median of the set.
end	A Date or POSIXct object (same class as the index of x). It defines the last day of the window to be included in the analysis.
ndays	A numeric vector, where each element is the number of days to be included in each analysis.
color	If color=NULL (default) the plot is black and white and each analysis is contained in different areas.
cex	Size of the labels.
...	Arguments to be read by xyplot.

Value

The result of TargetDiagram is a list with two components:

plot: a trellis object with the plot.

stat: a zoo object with err component of the result of `analyzeData`.

The result of `analyzeData` is a list with two components:

stat: a zoo object with the time evolution of several statistics (mean, median, standard deviation, median absolute deviation and interquartile range) of the set as a whole.

err: a data.frame with the same number of rows as the number of columns of the x object. It contains several columns with the statistics of the difference between each unit and the reference (see the references for details.)

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Jolliff, J.; Kindle, J. C.; Shulman, I.; Penta, B.; Friedrichs, M. A. M.; Helber, R. & Arnone, R. A. Summary diagrams for coupled hydrodynamic-ecosystem model skill assessment Journal of Marine Systems, 2009, 76, 64-82.
- O. Perpiñán, Statistical analysis of the performance and simulation of a two-axis tracking PV system, Solar Energy, 83:11(2074–2085), 2009.http://oa.upm.es/1843/1/PERPINAN_ART2009_01.pdf
- Taylor, K. E. Summarizing multiple aspects of model performance in a single diagram, Program for Climate Model Diagnosis and Intercomparison, 2000, <http://www-pcmdi.llnl.gov/publications/pdf/55.pdf>

Examples

```
library(lattice)
library(latticeExtra)

data(prodEx)

prodStat<-analyzeData(prodEx)
xyplot(prodStat$stat)
dif<-prodEx-prodStat$stat$Median;

day=as.Date('2008-8-29')

horizonplot(window(dif, start=day-90, end=day),
             origin=0, layout=c(1, 22), colorkey=TRUE, colorkey.digits=1,
             scales=list(y=list(relation="same"))))

###With a external reference
ref1=apply(prodEx, 1, median, na.rm=1)
prodStat1=analyzeData(prodEx, ref=ref1)
```

```

identical(prodStat, prodStat1)

###Target Diagram

ndays=c(5, 10, 15, 20)

#Color
if (require(RColorBrewer)){
  palette=brewer.pal(n=length(ndays), name='Set1')

  TDColor<-TargetDiagram(prodEx, end=day, ndays=ndays,
                        color=palette)
}

#B&W
TDbw<-TargetDiagram(prodEx, end=day, ndays=ndays)

```

C_utils-angle

Conversion between angle units.

Description

Several small functions to convert angle units.

Usage

```

d2r(x)
r2d(x)
h2r(x)
h2d(x)
r2h(x)
d2h(x)
r2sec(x)

```

Arguments

x A numeric value.

Value

A numeric value:

d2r: Degrees to radians.
r2d: Radians to degrees.
h2r: Hours to radians.
r2h: Radians to hours.
h2d: Hours to degrees.
d2h: Degrees to hours.
r2sec: Radians to seconds.

Author(s)

Oscar Perpiñán Lamigueiro.

C_utils-time

Utilities for time indexes.

Description

Several small functions to extract information from POSIXct indexes.

Usage

hour(x)
minute(x)
second(x)
hms(x)
doy(x)
dom(x)
month(x)
year(x)
DoY(x)
DoM(x)
Month(x)
Year(x)
dst(x)
truncDay(x)

Arguments

x A POSIXct vector.

Value

The functions year, month, day, hour, minute, second give the numeric value corresponding to their names.

doy and dom provide the (numeric) day of year and day of month, respectively.

Month, Year, DoY and DoM give the same result as month, year, doy and dom in a character string format.

hms gives the numeric value $\text{hour}(x) + \text{minute}(x)/60 + \text{second}(x)/3600$

dst is +1 if the Daylight Savings Time flag is in force, zero if not, -1 if unknown ([DateTimeClasses](#)).

truncDay truncates the POSIXct object towards the day.

Author(s)

Oscar Perpiñán Lamigueiro.

See Also

as.POSIXct

D_as.data.frameD-methods

Methods for Function as.data.frameD

Description

Convert a Sol object (or a extended class) into a data.frame with daily values.

Usage

```
## S4 method for signature 'Sol'
as.data.frameD(object, complete=FALSE)
```

Arguments

object	A Sol object (or extended.)
complete	A logical.

Methods

signature(object = "Sol") This function converts the object into a zoo container with the as.zooD function and then into a data.frame with as.data.frame. Besides, it includes three additional columns named month, day (day of year) and year.

See [as.zooD-methods](#) for a description of the argument complete.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.data.frameI-methods

Methods for Function as.data.frameI

Description

Convert a Sol object (or a extended class) into a data.frame with intradaily values.

Usage

```
## S4 method for signature 'Sol'
as.data.frameI(object, complete=FALSE, day=FALSE)
```

Arguments

object	A Sol object (or extended.)
complete	A logical.
day	A logical.

Methods

signature(object = "Sol") This function converts the object into a zoo container with the `as.zooI` function and then into a `data.frame` with `as.data.frame`. Besides, it includes three additional columns named `month`, `day` (day of year) and `year`.

See [as.zooI-methods](#) for a description of the arguments `complete` and `day`.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.data.frameM-methods

Methods for Function as.data.frameM

Description

Convert a G0 object (or a extended class) into a `data.frame` with monthly values.

Usage

```
## S4 method for signature 'G0'
as.data.frameM(object, complete=FALSE)
```

Arguments

object	A G0 object (or extended.)
complete	A logical.

Methods

signature(object = "G0") This function converts the object into a zoo container with the `as.zooM` function and then into a `data.frame` with `as.data.frame`. Besides, it includes two additional columns named `month` and `year`.

See [as.zooM-methods](#) for a description of the argument `complete`.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.data.frameY-methods

Methods for Function as.data.frameY

Description

Convert a G0 object (or a extended class) into a data.frame with yearly values.

Usage

```
## S4 method for signature 'G0'
as.data.frameY(object, complete=FALSE)
```

Arguments

object	A G0 object (or extended.)
complete	A logical.

Methods

signature(object = "G0") This function converts the object into a zoo container with the `as.zooY` function and then into a data.frame with `as.data.frame`. Besides, it includes an additional column named year.

See [as.zooY-methods](#) for a description of the argument complete.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.zooD-methods

Methods for Function as.zooD

Description

Convert a Sol, G0, Gef, ProdGCPV or ProdPVPS object into a zoo object with daily values.

Usage

```
## S4 method for signature 'Sol'
as.zooD(object, complete=FALSE)
```

Arguments

object	A Sol object (or extended.)
complete	A logical.

Methods

signature(object = "Sol") Conversion to a zoo object with the content of the solD slot.

signature(object = "G0") If complete=FALSE (default) the result includes only the columns of G0d, D0d and B0d from the G0D slot. If complete=TRUE it returns the contents of the slots solD and G0D.

signature(object = "Gef") If complete=FALSE (default) the result includes only the columns of Gefd, Defd and Befd from the GefD slot. If complete=TRUE it returns the contents of the slots solD, G0D and GefD

signature(object = "ProdGCPV") If complete=FALSE (default) the result includes only the columns of Eac, Edc and Yf from the prodD slot. If complete=TRUE it returns the contents of the slots solD, G0D, GefD and prodD.

signature(object = "ProdPVPS") If complete=FALSE (default) the result includes only the columns of Eac, Qd and Yf from the prodD slot. If complete=TRUE it returns the contents of the slots solD, G0D, GefD and prodD.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.zooI-methods

Methods for Function as.zooI

Description

Convert a Sol, G0, Gef, ProdGCPV or ProdPVPS object into a zoo object with intradaily values and (optionally) daily values.

Usage

```
## S4 method for signature 'Sol'
as.zooI(object, complete=FALSE, day=FALSE)
```

Arguments

object	A Sol object (or extended).
complete	A logical.
day	A logical.

Methods

signature(object = "Sol") If complete=FALSE and day=FALSE (default) the result includes only the content of the solI slot. If day=TRUE the contents of the solD slot are included.

signature(object = "G0") If complete=FALSE and day=FALSE (default) the result includes only the columns of G0, D0 and B0 of the G0I slot. If complete=TRUE it returns the contents of the slots G0I and solI. If day=TRUE the daily values (slots G0D and solD) are also included.)

signature(object = "Gef") If complete=FALSE and day=FALSE (default) the result includes only the columns of Gef, Def and Bef of the GefI slot. If complete=TRUE it returns the contents of the slots GefI, GOI and solI. If day=TRUE the daily values (slots GefD, GOD and sold) are also included.)

signature(object = "ProdGCPV") If complete=FALSE and day=FALSE (default) the result includes only the columns of Pac and Pdc of the prodI slot. If complete=TRUE it returns the contents of the slots prodI, GefI, GOI and solI. If day=TRUE the daily values (slots prodD, GefD, GOD and sold) are also included.)

signature(object = "ProdPVPS") If complete=FALSE and day=FALSE (default) the result includes only the columns of Pac and Q of the prodI slot. If complete=TRUE it returns the contents of the slots prodI, GefI, GOI and solI. If day=TRUE the daily values (slots prodD, GefD, GOD and sold) are also included.)

Author(s)

Oscar Perpiñán Lamigueiro

D_as.zooM-methods *Methods for Function as.zooM*

Description

Convert a GO, Gef, ProdGCPV or ProdPVPS object into a zoo object with monthly average of daily values.

Usage

```
## S4 method for signature 'GO'
as.zooM(object, complete=FALSE)
```

Arguments

object	A GO object (or extended.)
complete	A logical.

Methods

signature(object = "GO") The result is the G0dm slot.

signature(object = "Gef") If complete=FALSE (default) the result is the slot Gefdm. If complete=TRUE it returns the slot G0dm.

signature(object = "ProdGCPV") If complete=FALSE (default) the result is the prodDm slot. If complete=TRUE the result includes the slots G0dm and Gefdm.

signature(object = "ProdPVPS") If complete=FALSE (default) the result is the prodDm slot. If complete=TRUE the result includes the slots G0dm and Gefdm.

Author(s)

Oscar Perpiñán Lamigueiro

D_as.zooY-methods *Methods for Function as.zooY*

Description

Convert a G0, Gef, ProdGCPV or ProdPVPS object into a zoo object with yearly values.

Usage

```
## S4 method for signature 'G0'  
as.zooY(object, complete=FALSE)
```

Arguments

object A G0 object (or extended.)
complete A logical.

Methods

signature(object = "G0") The result is the G0y slot.
signature(object = "Gef") If complete=FALSE (default) the result is the slot Gefy. If complete=TRUE it returns the slot G0y.
signature(object = "ProdGCPV") If complete=FALSE (default) the result is the prody slot. If complete=TRUE the result includes the slots G0y and Gefy.
signature(object = "ProdPVPS") If complete=FALSE (default) the result is the prody slot. If complete=TRUE the result includes the slots G0y and Gefy.

Author(s)

Oscar Perpiñán Lamigueiro

D_compare-methods *Compare G0, Gef and ProdGCPV objects*

Description

Compare and plot the yearly values of several objects.

Usage

```
## S4 method for signature 'G0'  
compare(...)
```

Arguments

... A list of objects to be compared.

Methods

The class of the first element of ... is used to determine the suitable method. The result is plotted with [dotplot](#):

signature(... = "G0") yearly values of G0d, B0d and D0d.

signature(... = "Gef") yearly values of Gefd, Befd and Defd.

signature(... = "ProdGCPV") yearly values of Yf, Gefd and G0d.

Author(s)

Oscar Perpiñán Lamigueiro

See Also

[dotplot](#)

Examples

```
lat=37.2;
G0dm=c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562, 2814,
2179)
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
prom=list(G0dm=G0dm, Ta=Ta)

###Comparison of different tracker methods
ProdFixed<-prodGCPV(lat=lat, dataRad=prom, keep.night=FALSE)
Prod2x<-prodGCPV(lat=lat, dataRad=prom, modeTrk='two', keep.night=FALSE)
ProdHoriz<-prodGCPV(lat=lat, dataRad=prom, modeTrk='horiz', keep.night=FALSE)

compare(ProdFixed, Prod2x, ProdHoriz)

##The first element rules the method
GefFixed=as(ProdFixed, 'Gef')
compare(GefFixed, Prod2x, ProdHoriz)

## Not run:
###compare and do.call
EstMadrid <- subset(RedEstaciones, NomProv=='Madrid')
nEstMadrid <- nrow(EstMadrid)
namesMadrid <- EstMadrid$NomEst

prodMadrid <- lapply(1:nEstMadrid,
                    function(x){try(prodGCPV(lat=41, modeRad='siar',
                                             dataRad=list(prov=28, est=x,
                                                         start='01/01/2009', end='31/12/2010'))
                                )})

names(prodMadrid) <- namesMadrid
```

```
okMadrid <- lapply(prodMadrid, class)!= 'try-error'  
prodMadrid <- prodMadrid[okMadrid]  
  
do.call(compare, prodMadrid)  
  
## End(Not run)
```

D_getData-methods *Methods for function getData*

Description

Meteorological source data of a Meteo (or extended) object.

Methods

signature(object = "Meteo") returns the meteorological source data of the slot data of the object.

Author(s)

Oscar Perpiñán Lamigueiro

D_getG0-methods *Methods for function getG0*

Description

Global irradiation source data of a Meteo (or extended) object.

Methods

signature(object = "Meteo") returns the global irradiation values stored in a Meteo object.

Author(s)

Oscar Perpiñán Lamigueiro

D_getLat-methods

Methods for Function getLat

Description

Latitude angle of solar objects.

Usage

```
getLat(object, units='rad')
```

Arguments

object	A Sol or Meteo object (or extended.)
units	A character, 'rad' or 'deg'.

Methods

This function returns the latitude angle in radians (units='rad', default) or degrees (units='deg').

signature(object = "Meteo") Value of the latData slot, which is defined by the argument lat of the [readSIAR](#), [readG0dm](#) and [readBD](#) functions, or by the lat component of the dataRad object passed to calcG0 (or equivalent) . It is the latitude of the meteorological station (or equivalent) which provided the irradiation source data. It may be different from the value used for the calculation procedure.

signature(object = "Sol") Value of the lat slot, which is defined by the argument lat of the calcSol function. It is the value used through the calculation procedure.

signature(object = "G0") same as for the Sol class.

Author(s)

Oscar Perpiñán Lamigueiro

D_indexD-methods

Methods for Function indexD

Description

Daily time index of solar objects.

Methods

signature(object = "Meteo") returns the index of the data slot (a zoo object.)

signature(object = "Sol") returns the index of the solD slot (a zoo object.)

signature(object = "G0") same as for object='Sol'

Author(s)

Oscar Perpiñán Lamigueiro

D_indexI-methods *Methods for Function indexI*

Description

Intra-daily time index of solar objects.

Methods

signature(object = "Sol") returns the index of the slot solI (a zoo object).

Author(s)

Oscar Perpiñán Lamigueiro

D_indexRep-methods *Methods for Function indexRep*

Description

Daily time index of solar object.

Methods

signature(object = "Sol") returns the daily index of the solD slot but repeated to match the length of the index of the solI slot.

Author(s)

Oscar Perpiñán Lamigueiro

D_levelplot-methods *Methods for function levelplot.*

Description

Methods for function levelplot and zoo and solar objects.

Methods

signature(x = "formula", data = "zoo"): The zoo object is converted into a data.frame object and additional columns are added (day, month and year, and w with the solar hour in radians). This data.frame is the data argument for a call to levelplot, using the S3 method for class formula.

signature(x = "formula", data = "Meteo"): The Meteo object is converted into a zoo object, and the previous method is used.

signature(x = "formula", data = "Sol"): idem

signature(x = "formula", data = "G0"): idem

Author(s)

Oscar Perpiñán Lamigueiro

D_Losses-methods *Losses of a GCPV system*

Description

The function losses calculates the yearly losses from a Gef or a ProdGCPV object. The function compareLosses compares the losses from several ProdGCPV objects and plots the result with [dotplot](#).

Usage

```
compareLosses(...)
losses(object)
```

Arguments

... A list of ProdGCPV objects to be compared.
 object An object of Gef or ProdGCPV class..

Methods

signature(... = "Gef") shadows and angle of incidence (AoI) losses.

signature(... = "ProdGCPV") shadows, AoI, generator (mainly temperature), DC and AC system (as detailed in effSys of [fProd](#)) and inverter losses.

Author(s)

Oscar Perpiñán Lamigueiro

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

See Also

[fInclin](#), [fProd](#)

Examples

```
lat=37.2;
G0dm=c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562, 2814,
2179)
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
prom=list(G0dm=G0dm, Ta=Ta)

###Comparison of different tracker methods
ProdFixed<-prodGCPV(lat=lat,dataRad=prom, keep.night=FALSE)
Prod2x<-prodGCPV(lat=lat, dataRad=prom, modeTrk='two', keep.night=FALSE)
ProdHoriz<-prodGCPV(lat=lat,dataRad=prom, modeTrk='horiz', keep.night=FALSE)

losses(ProdFixed)
losses(as(ProdFixed, 'Gef'))

compareLosses(ProdFixed, Prod2x, ProdHoriz)
```

D_mergesolaR-methods *Merge solaR objects*

Description

Merge the daily time series of solaR objects

Usage

```
## S4 method for signature 'G0'
mergesolaR(...)
```

Arguments

... A list of objects to be merged.

Methods

The class of the first element of ... is used to determine the suitable method. Only the most important daily variable is merged, depending on the class of the objects:

```
signature(... = "Meteo") G0
signature(... = "G0") G0d
signature(... = "Gef") Gefd
signature(... = "ProdGCPV") Yf
signature(... = "ProdPVPS") Yf
```

Examples

```
lat=37.2;
G0dm=c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562, 2814,
2179)
Ta=c(10, 14.1, 15.6, 17.2, 19.3, 21.2, 28.4, 29.9, 24.3, 18.2, 17.2, 15.2)
prom=list(G0dm=G0dm, Ta=Ta)

###Different tracker methods
ProdFixed<-prodGCPV(lat=lat,dataRad=prom, keep.night=FALSE)
Prod2x<-prodGCPV(lat=lat, dataRad=prom, modeTrk='two', keep.night=FALSE)
ProdHoriz<-prodGCPV(lat=lat,dataRad=prom, modeTrk='horiz', keep.night=FALSE)

prod <- mergesolaR(ProdFixed, Prod2x, ProdHoriz)
head(prod)

## Not run:
EstMadrid <- subset(RedEstaciones, NomProv=='Madrid')
nEstMadrid <- nrow(EstMadrid)
namesMadrid <- EstMadrid$NomEst

prodMadrid <- lapply(1:nEstMadrid,
                    function(x){try(prodGCPV(lat=41, modeRad='siar',
                                             dataRad=list(prov=28, est=x,
                                                         start='01/01/2009', end='31/12/2010'))
                                )})

names(prodMadrid) <- namesMadrid
okMadrid <- lapply(prodMadrid, class)!='try-error'
prodMadrid <- prodMadrid[okMadrid]

YfMadrid <- do.call(mergesolaR, prodMadrid)

horizonplot(YfMadrid-rowMeans(YfMadrid),
            origin=0,
            scales=list(y=list(relation='same')),
            colorkey=TRUE)

TargetDiagram(YfMadrid, end=as.POSIXct('2010-12-31'), ndays=c(10, 20,
30, 40, 50, 60), cex=0.6)

## End(Not run)
```

D_shadeplot-methods *Methods for Function shadeplot*

Description

Visualization of the content of a [Shade](#) object.

Methods

`signature(x = "Shade")` display the results of the iteration with a level plot for the two-axis tracking, or with conventional plot for horizontal tracking and fixed systems.

Author(s)

Oscar Perpiñán Lamigueiro

D_window-methods *Methods for extracting a time window*

Description

Method for extracting the subset of a `solR` object whose daily time index (`indexD`) is comprised between the times `i` and `j`.

Usage

```
## S4 method for signature 'Meteo'
x[i, j, ..., drop = TRUE]
## S4 method for signature 'Sol'
x[i, j, ..., drop = TRUE]
## S4 method for signature 'G0'
x[i, j, ..., drop = TRUE]
## S4 method for signature 'Gef'
x[i, j, ..., drop = TRUE]
## S4 method for signature 'ProdGCPV'
x[i, j, ..., drop = TRUE]
## S4 method for signature 'ProdPVPS'
x[i, j, ..., drop = TRUE]
```

Arguments

<code>x</code>	A <code>Meteo</code> , <code>Sol</code> , etc. object.
<code>i</code>	an index/time value (Date or POSIXct classes) defining the start of the time window.
<code>j</code>	an index/time value (Date or POSIXct classes) defining the end of the time window.
<code>..., drop</code>	Additional arguments for <code>window.zoo</code>

Author(s)

Oscar Perpiñán Lamigueiro

See Also

[window.zoo indexD](#)

Examples

```
lat=37.2
sol=calcSol(lat, BTd=fBTd(mode='serie'))
range(indexD(sol))

start <- as.POSIXct('2011-01-01')
end <- as.POSIXct('2011-01-31')

solWindow <- sol[start, end]
range(indexD(solWindow))
```

D_xyplot-methods

Methods for function xyplot in Package 'solaR'

Description

Methods for function xyplot in Package 'solaR'

Methods

`signature(x = "formula", data = "zoo")`: The zoo object is converted into a data.frame object and additional columns are added (day, month and year, and w with the solar hour in radians). This data.frame is the data argument for a call to xyplot, using the S3 method for class formula.

`signature(x = "formula", data = "Meteo")`: The Meteo object is converted into a zoo object with `getData(data)`. This zoo is the data argument for a call to xyplot, using the S4 method for `signature(x = "formula", data = "zoo")`.

`signature(x = "formula", data = "Sol")`: The Sol object is converted into a zoo object with `as.zooI(data, complete=TRUE, day=TRUE)` (therefore, the zoo includes the whole content of the object). This zoo is the data argument for a call to xyplot, using the S4 method for `signature(x = "formula", data = "zoo")`.

`signature(x = "formula", data = "G0")`: The G0 object is converted into a zoo object with `as.zooI(data, complete=TRUE, day=TRUE)` (therefore, the zoo includes the whole content of the object). This zoo is the data argument for a call to xyplot, using the S4 method for `signature(x = "formula", data = "zoo")`.

`signature(x = "Meteo", data = "missing")`: The Meteo object is converted into a zoo object with `getData(x)` and displayed with the method for zoo.

signature(x = "G0", data = "missing"): The x object is converted into a zoo object with `as.zooD(x, complete=FALSE)`. Therefore, the content of the G0D slot (a zoo object) is displayed with the method for zoo.

signature(x = "ProdGCPV", data = "missing"): Idem, but the variables are not superposed.

signature(x = "ProdPVPS", data = "missing"): Idem.

signature(x = "formula", data = "Shade"): The Shade object is converted into a `data.frame` and passed as the data argument to the `xypplot` function. Once again, the S3 method for class `formula` is used.

Author(s)

Oscar Perpiñán Lamigueiro

E_helios	<i>Daily irradiation and ambient temperature from the Helios-IES database</i>
----------	---

Description

A year of irradiation, maximum and minimum ambient temperature from the HELIOS-IES database.

Usage

```
data(helios)
```

Format

A data frame with 355 observations on the following 4 variables:

`yyyy.mm.dd` a factor: year, month and day.

`G.0.` a numeric vector, daily global horizontal irradiation.

`TambMax` a numeric vector, maximum ambient temperature.

`TambMin` a numeric vector, minimum ambient temperature.

Source

<http://helios.ies-def.upm.es/consulta.aspx>

E_prodEx	<i>Productivity of a set of PV systems of a PV plant.</i>
----------	---

Description

A zoo object with the time evolution of the final productivity of a set of 22 systems of a large PV plant.

Usage

```
data(prodEx)
```

References

O. Perpiñán, Statistical analysis of the performance and simulation of a two-axis tracking PV system, Solar Energy, 83:11(2074–2085), 2009. http://oa.upm.es/1843/1/PERPINAN_ART2009_01.pdf

See Also

[TargetDiagram](#), [analyzeData](#).

E_pumpCoef	<i>Coefficients of centrifugal pumps.</i>
------------	---

Description

Coefficients of centrifugal pumps

Usage

```
data(pumpCoef)
```

Format

A data frame with 13 columns:

Qn rated flux

stages number of stages

Qmax maximum flux

Pmn rated motor power

a, b, c Coefficients of the equation $H = a \cdot f^2 + b \cdot f \cdot Q + c \cdot Q^2$.

g, h, i Coefficients of the efficiency curve of the motor (50 Hz): $\eta_m = g \cdot (\%P_{mn})^2 + h \cdot (\%P_{mn}) + i$.

j, k, l Coefficients of the efficiency curve of the pump (50 Hz): $\eta_b = j \cdot Q^2 + k \cdot Q + l$.

Details

With this version only pumps from the manufacturer Grundfos are included.

Source

<http://net.grundfos.com/App1/WebCAPS/custom?>

References

Perpiñán, O, Energía Solar Fotovoltaica, 2011. (<http://procomun.wordpress.com/documentos/libroesf/>)

E_solaR.theme

solaR theme

Description

A customized theme for lattice. It is based on the custom.theme.2 function of the latticeExtra package with the next values:

- pch=19
- cex=0.7
- region=rev(brewer.pal(9, 'YlOrRd'))
- strip.background\$col='lightgray'
- strip.shingle\$col='transparent'

Index

*Topic **classes**

- B1_Meteo-class, [29](#)
- B2_Sol-class, [30](#)
- B3_G0-class, [31](#)
- B4_Gef-class, [32](#)
- B5_ProdGCPV-class, [34](#)
- B6_ProdPVPS-class, [36](#)
- B7_Shade-class, [37](#)

*Topic **constructors**

- A1_calcSol, [3](#)
- A2_calcG0, [4](#)
- A3_calcGef, [8](#)
- A4_prodGCPV, [11](#)
- A5_prodPVPS, [15](#)
- A6_calcShd, [17](#)
- A7_optimShd, [19](#)
- A8_readBD, [24](#)
- A8_readG0dm, [25](#)
- A8_readSIAR, [26](#)

*Topic **datasets**

- E_helios, [85](#)
- E_prodEx, [86](#)
- E_pumpCoef, [86](#)
- E_solaR.theme, [87](#)

*Topic **methods**

- D_as.data.frameD-methods, [70](#)
- D_as.data.frameI-methods, [70](#)
- D_as.data.frameM-methods, [71](#)
- D_as.data.frameY-methods, [72](#)
- D_as.zooD-methods, [72](#)
- D_as.zooI-methods, [73](#)
- D_as.zooM-methods, [74](#)
- D_as.zooY-methods, [75](#)
- D_compare-methods, [75](#)
- D_getData-methods, [77](#)
- D_getG0-methods, [77](#)
- D_getLat-methods, [78](#)
- D_indexD-methods, [78](#)
- D_indexI-methods, [79](#)

- D_indexRep-methods, [79](#)
- D_levelplot-methods, [80](#)
- D_Losses-methods, [80](#)
- D_mergesolaR-methods, [81](#)
- D_shadeplot-methods, [83](#)
- D_window-methods, [83](#)
- D_xyplot-methods, [84](#)

*Topic **utilities**

- A1_calcSol, [3](#)
- A2_calcG0, [4](#)
- A3_calcGef, [8](#)
- A4_prodGCPV, [11](#)
- A5_prodPVPS, [15](#)
- A6_calcShd, [17](#)
- A7_optimShd, [19](#)
- A8_readBD, [24](#)
- A8_readG0dm, [25](#)
- A8_readSIAR, [26](#)
- C_corrFdKt, [39](#)
- C_fBTd, [40](#)
- C_fCompD, [42](#)
- C_fCompI, [43](#)
- C_fInclin, [45](#)
- C_fProd, [47](#)
- C_fPump, [49](#)
- C_fSolD, [51](#)
- C_fSolI, [53](#)
- C_fSombra, [55](#)
- C_fTemp, [57](#)
- C_fTheta, [58](#)
- C_HQCurve, [60](#)
- C_local2Solar, [61](#)
- C_NmgPVPS, [63](#)
- C_sample2Diff, [65](#)
- C_TargetDiagram, [66](#)
- C_utils-angle, [68](#)
- C_utils-time, [69](#)

[, G0, ANY, ANY-method (D_window-methods),
[83](#)

- [,G0-method (D_window-methods), 83
- [,Gef,ANY,ANY-method
(D_window-methods), 83
- [,Gef-method (D_window-methods), 83
- [,Meteo,ANY,ANY-method
(D_window-methods), 83
- [,Meteo-method (D_window-methods), 83
- [,ProdGCPV,ANY,ANY-method
(D_window-methods), 83
- [,ProdGCPV-method (D_window-methods), 83
- [,ProdPVPS,ANY,ANY-method
(D_window-methods), 83
- [,ProdPVPS-method (D_window-methods), 83
- [,Sol,ANY,ANY-method
(D_window-methods), 83
- [,Sol-method (D_window-methods), 83
- A1_calcSol, 3
- A2_calcG0, 4
- A3_calcGef, 8
- A4_prodGCPV, 11
- A5_prodPVPS, 15
- A6_calcShd, 17
- A7_optimShd, 19
- A8_readBD, 24
- A8_readG0dm, 25
- A8_readSIAR, 26
- analyzeData, 67, 86
- analyzeData (C_TargetDiagram), 66
- as.data.frame, Shade-method
(B7_Shade-class), 37
- as.data.frameD
(D_as.data.frameD-methods), 70
- as.data.frameD,Sol-method
(D_as.data.frameD-methods), 70
- as.data.frameD-methods
(D_as.data.frameD-methods), 70
- as.data.frameI
(D_as.data.frameI-methods), 70
- as.data.frameI,Sol-method
(D_as.data.frameI-methods), 70
- as.data.frameI-methods
(D_as.data.frameI-methods), 70
- as.data.frameM
(D_as.data.frameM-methods), 71
- as.data.frameM,G0-method
(D_as.data.frameM-methods), 71
- as.data.frameM-methods
(D_as.data.frameM-methods), 71
- as.data.frameY
(D_as.data.frameY-methods), 72
- as.data.frameY,G0-method
(D_as.data.frameY-methods), 72
- as.data.frameY-methods
(D_as.data.frameY-methods), 72
- as.POSIXct, 41
- as.zooD (D_as.zooD-methods), 72
- as.zooD,G0-method (D_as.zooD-methods),
72
- as.zooD,Gef-method (D_as.zooD-methods),
72
- as.zooD,ProdGCPV-method
(D_as.zooD-methods), 72
- as.zooD,ProdPVPS-method
(D_as.zooD-methods), 72
- as.zooD,Sol-method (D_as.zooD-methods),
72
- as.zooD-methods, 70
- as.zooD-methods (D_as.zooD-methods), 72
- as.zooI (D_as.zooI-methods), 73
- as.zooI,G0-method (D_as.zooI-methods),
73
- as.zooI,Gef-method (D_as.zooI-methods),
73
- as.zooI,ProdGCPV-method
(D_as.zooI-methods), 73
- as.zooI,ProdPVPS-method
(D_as.zooI-methods), 73
- as.zooI,Sol-method (D_as.zooI-methods),
73
- as.zooI-methods, 71
- as.zooI-methods (D_as.zooI-methods), 73
- as.zooM (D_as.zooM-methods), 74
- as.zooM,G0-method (D_as.zooM-methods),
74
- as.zooM,Gef-method (D_as.zooM-methods),
74
- as.zooM,ProdGCPV-method
(D_as.zooM-methods), 74
- as.zooM,ProdPVPS-method
(D_as.zooM-methods), 74
- as.zooM-methods, 71
- as.zooM-methods (D_as.zooM-methods), 74
- as.zooY (D_as.zooY-methods), 75
- as.zooY,G0-method (D_as.zooY-methods),
75
- as.zooY,Gef-method (D_as.zooY-methods),

- 75
- as.zooY, ProdGCPV-method
(D_as.zooY-methods), 75
- as.zooY, ProdPVPS-method
(D_as.zooY-methods), 75
- as.zooY-methods, 72
- as.zooY-methods (D_as.zooY-methods), 75
- B1_Meteo-class, 29
- B2_Sol-class, 30
- B3_G0-class, 31
- B4_Gef-class, 32
- B5_ProdGCPV-class, 34
- B6_ProdPVPS-class, 36
- B7_Shade-class, 37
- C_corrFdKt, 39
- C_fBTd, 40
- C_fCompD, 42
- C_fCompI, 43
- C_fInclin, 45
- C_fProd, 47
- C_fPump, 49
- C_fSolD, 51
- C_fSolI, 53
- C_fSombra, 55
- C_fTemp, 57
- C_fTheta, 58
- C_HQCurve, 60
- C_local2Solar, 61
- C_NmgPVPS, 63
- C_sample2Diff, 65
- C_TargetDiagram, 66
- C_utils-angle, 68
- C_utils-time, 69
- calcG0, 3, 8, 9, 11–13, 16, 19, 20, 24, 31, 45, 53
- calcG0 (A2_calcG0), 4
- calcGef, 11–13, 16–18, 32, 45, 47, 58, 60
- calcGef (A3_calcGef), 8
- calcShd, 8, 9, 12, 13, 19, 22, 56
- calcShd (A6_calcShd), 17
- calcSol, 4–6, 8, 11, 12, 16, 20, 30, 39, 42, 44, 56–59
- calcSol (A1_calcSol), 3
- CBIND (C_local2Solar), 61
- char2diff (C_sample2Diff), 65
- compare, 13
- compare (D_compare-methods), 75
- compare, G0-method (D_compare-methods), 75
- compare, Gef-method (D_compare-methods), 75
- compare, ProdGCPV-method
(D_compare-methods), 75
- compare-methods (D_compare-methods), 75
- compareLosses, 13
- compareLosses (D_Losses-methods), 80
- compareLosses, ProdGCPV-method
(D_Losses-methods), 80
- compareLosses-methods
(D_Losses-methods), 80
- corrFdKt, 5, 6, 42, 44
- corrFdKt (C_corrFdKt), 39
- d2h (C_utils-angle), 68
- d2r (C_utils-angle), 68
- D_as.data.frameD-methods, 70
- D_as.data.frameI-methods, 70
- D_as.data.frameM-methods, 71
- D_as.data.frameY-methods, 72
- D_as.zooD-methods, 72
- D_as.zooI-methods, 73
- D_as.zooM-methods, 74
- D_as.zooY-methods, 75
- D_compare-methods, 75
- D_getData-methods, 77
- D_getG0-methods, 77
- D_getLat-methods, 78
- D_indexD-methods, 78
- D_indexI-methods, 79
- D_indexRep-methods, 79
- D_levelplot-methods, 80
- D_Losses-methods, 80
- D_mergesolar-methods, 81
- D_shadeplot-methods, 83
- D_window-methods, 83
- D_xyplot-methods, 84
- DateTimeClasses, 69
- df2Meteo, 5, 29
- df2Meteo (A8_readBD), 24
- dfI2Meteo, 5, 29, 44
- dfI2Meteo (A8_readBD), 24
- diff2Hours (C_sample2Diff), 65
- DoM (C_utils-time), 69
- dom (C_utils-time), 69
- dotplot, 76, 80
- DoY (C_utils-time), 69

- doy (C_utils-time), 69
- dst (C_utils-time), 69
- E_helios, 85
- E_prodEx, 86
- E_pumpCoef, 86
- E_solaR.theme, 87
- fBTd, 3, 4, 51
- fBTd (C_fBTd), 40
- fCompD, 4, 6, 31, 39, 40, 44
- fCompD (C_fCompD), 42
- fCompI, 4, 6, 31, 39, 40, 43, 47
- fCompI (C_fCompI), 43
- FdKtBRL (C_corrFdKt), 39
- FdKtCLIMEDd (C_corrFdKt), 39
- FdKtCLIMEDh, 44
- FdKtCLIMEDh (C_corrFdKt), 39
- FdKtCPR, 5, 42
- FdKtCPR (C_corrFdKt), 39
- FdKtEKDd (C_corrFdKt), 39
- FdKtEKDh (C_corrFdKt), 39
- FdKtLJ (C_corrFdKt), 39
- FdKtPage, 5, 42
- FdKtPage (C_corrFdKt), 39
- fInclin, 8, 9, 13, 19, 32, 49, 60, 81
- fInclin (C_fInclin), 45
- fProd, 13, 34, 80, 81
- fProd (C_fProd), 47
- fPump, 16, 17, 37, 64
- fPump (C_fPump), 49
- fSolD, 3, 5, 8, 12, 16, 20, 30, 39, 41, 42, 54
- fSolD (C_fSolD), 51
- fSolI, 3, 5, 8, 12, 16, 20, 30, 39, 44, 61
- fSolI (C_fSolI), 53
- fSombra, 17, 56, 60
- fSombra (C_fSombra), 55
- fSombra2X (C_fSombra), 55
- fSombra6, 18, 21, 56
- fSombra6 (C_fSombra), 55
- fSombraEst, 56
- fSombraEst (C_fSombra), 55
- fSombraHoriz, 56
- fSombraHoriz (C_fSombra), 55
- fTemp, 4, 25, 28, 49
- fTemp (C_fTemp), 57
- fTheta, 8, 9, 13, 19, 33, 45, 47, 56
- fTheta (C_fTheta), 58
- G0, 30, 33–36, 38
- G0-class (B3_G0-class), 31
- Gef, 18, 30, 32, 34–36, 38, 47
- Gef-class (B4_Gef-class), 32
- getData (D_getData-methods), 77
- getData, Meteo-method (D_getData-methods), 77
- getData-methods (D_getData-methods), 77
- getG0 (D_getG0-methods), 77
- getG0, Meteo-method (D_getG0-methods), 77
- getG0-methods (D_getG0-methods), 77
- getLat (D_getLat-methods), 78
- getLat, G0-method (D_getLat-methods), 78
- getLat, Meteo-method (D_getLat-methods), 78
- getLat, Sol-method (D_getLat-methods), 78
- getLat-methods (D_getLat-methods), 78
- h2d (C_utils-angle), 68
- h2r (C_utils-angle), 68
- helios (E_helios), 85
- hms (C_utils-time), 69
- hour (C_utils-time), 69
- HQCurve (C_HQCurve), 60
- indexD, 83, 84
- indexD (D_indexD-methods), 78
- indexD, G0-method (D_indexD-methods), 78
- indexD, Meteo-method (D_indexD-methods), 78
- indexD, Sol-method (D_indexD-methods), 78
- indexD-methods (D_indexD-methods), 78
- indexI (D_indexI-methods), 79
- indexI, Sol-method (D_indexI-methods), 79
- indexI-methods (D_indexI-methods), 79
- indexRep, Sol-method (D_indexRep-methods), 79
- indexRep-methods (D_indexRep-methods), 79
- levelplot, formula, G0-method (D_levelplot-methods), 80
- levelplot, formula, Meteo-method (D_levelplot-methods), 80
- levelplot, formula, Sol-method (D_levelplot-methods), 80
- levelplot, formula, zoo-method (D_levelplot-methods), 80

- levelplot-methods
 - (D_levelplot-methods), 80
- local2Solar (C_local2Solar), 61
- lonHH (C_local2Solar), 61
- losses (D_Losses-methods), 80
- losses, Gef-method (D_Losses-methods), 80
- losses, ProdGCPV-method
 - (D_Losses-methods), 80
- losses-methods (D_Losses-methods), 80
- mergesolaR, 13
- mergesolaR (D_mergesolaR-methods), 81
- mergesolaR, G0-method
 - (D_mergesolaR-methods), 81
- mergesolaR, Gef-method
 - (D_mergesolaR-methods), 81
- mergesolaR, Meteo-method
 - (D_mergesolaR-methods), 81
- mergesolaR, ProdGCPV-method
 - (D_mergesolaR-methods), 81
- mergesolaR, ProdPVPS-method
 - (D_mergesolaR-methods), 81
- mergesolaR-methods
 - (D_mergesolaR-methods), 81
- Meteo, 31, 33–36, 38, 57
- Meteo-class (B1_Meteo-class), 29
- minute (C_utils-time), 69
- Month (C_utils-time), 69
- month (C_utils-time), 69
- NmgPVPS, 17, 50, 60
- NmgPVPS (C_NmgPVPS), 63
- optimShd, 37, 56
- optimShd (A7_optimShd), 19
- P2E (C_sample2Diff), 65
- prodEx (E_prodEx), 86
- ProdGCPV, 38
- prodGCPV, 22, 34, 47, 49
- prodGCPV (A4_prodGCPV), 11
- ProdGCPV-class (B5_ProdGCPV-class), 34
- ProdPVPS, 17
- prodPVPS, 36, 37, 50, 60, 64
- prodPVPS (A5_prodPVPS), 15
- ProdPVPS-class (B6_ProdPVPS-class), 36
- pumpCoef, 16, 17, 36, 49, 50, 60, 63, 64
- pumpCoef (E_pumpCoef), 86
- r2d (C_utils-angle), 68
- r2h (C_utils-angle), 68
- r2sec (C_utils-angle), 68
- read.table, 25
- read.zoo, 28
- readBD, 4–6, 26, 28, 29, 42, 57, 58, 78
- readBD (A8_readBD), 24
- readBDi, 5, 6, 29, 44
- readBDi (A8_readBD), 24
- readG0dm, 5, 6, 25, 28, 29, 42, 78
- readG0dm (A8_readG0dm), 25
- readMAPA (A8_readSIAR), 26
- readSIAR, 5, 6, 25, 26, 29, 42, 57, 58, 78
- readSIAR (A8_readSIAR), 26
- sample2Hours (C_sample2Diff), 65
- second (C_utils-time), 69
- seq.POSIXt, 3, 5, 20, 41, 53, 65
- Shade, 22, 35, 83
- Shade-class, 22
- Shade-class (B7_Shade-class), 37
- shadeplot (D_shadeplot-methods), 83
- shadeplot, Shade-method
 - (D_shadeplot-methods), 83
- shadeplot-methods, 22
- shadeplot-methods
 - (D_shadeplot-methods), 83
- show, G0-method (B3_G0-class), 31
- show, Gef-method (B4_Gef-class), 32
- show, Meteo-method (B1_Meteo-class), 29
- show, ProdGCPV-method
 - (B5_ProdGCPV-class), 34
- show, ProdPVPS-method
 - (B6_ProdPVPS-class), 36
- show, Shade-method (B7_Shade-class), 37
- show, Sol-method (B2_Sol-class), 30
- Sol, 31–38, 57, 65
- Sol-class, 3
- Sol-class (B2_Sol-class), 30
- solaR.theme (E_solaR.theme), 87
- TargetDiagram, 86
- TargetDiagram (C_TargetDiagram), 66
- truncDay (C_utils-time), 69
- window (D_window-methods), 83
- window-methods (D_window-methods), 83
- window.zoo, 84
- xyplot, formula, G0-method
 - (D_xyplot-methods), 84

xyplot, formula, Meteo-method
(D_xyplot-methods), [84](#)

xyplot, formula, Shade-method
(D_xyplot-methods), [84](#)

xyplot, formula, Sol-method
(D_xyplot-methods), [84](#)

xyplot, formula, zoo-method
(D_xyplot-methods), [84](#)

xyplot, G0, missing-method
(D_xyplot-methods), [84](#)

xyplot, Meteo, missing-method
(D_xyplot-methods), [84](#)

xyplot, ProdGCPV, missing-method
(D_xyplot-methods), [84](#)

xyplot, ProdPVPS, missing-method
(D_xyplot-methods), [84](#)

xyplot-methods (D_xyplot-methods), [84](#)

Year (C_utils-time), [69](#)

year (C_utils-time), [69](#)

zoo2Meteo, [5](#), [29](#), [44](#)

zoo2Meteo (A8_readBD), [24](#)