Package ‘solaR’

April 20, 2015

Type Package
Title Radiation and Photovoltaic Systems
Version 0.41
Date 2015-04-20
Encoding UTF-8
Author Oscar Perpiñán Lamigueiro
Maintainer Oscar Perpiñán Lamigueiro <oscar.perpinan@gmail.com>
Description Calculation methods of solar radiation and performance of photovoltaic systems from daily and intradaily irradiation data sources.
URL http://oscarperpinan.github.io/solar/
BugReports https://github.com/oscarperpinan/solar/issues
License GPL-3
LazyData yes
Depends R (>= 2.10), methods, zoo, lattice, latticeExtra
Imports RColorBrewer
Suggests sp, raster, rasterVis, tdr
NeedsCompilation no
Repository CRAN
Date/Publication 2015-04-20 22:10:20

R topics documented:

solaR-package ........................................... 3
A1_calcSol ............................................. 5
A2_calcG0 ............................................. 7
A3_calcGef ............................................ 10
A4_prodGCPV .......................................... 13
A5_prodPVPS .......................................... 17
A6_calcShd ............................................ 19
R topics documented:

A7_optimShd .............................................. 21
A8_readBD ............................................. 25
A8_readG0dm ........................................... 27
B1_Meteo-class ......................................... 28
B2_Sol-class ............................................ 29
B3_G0-class .............................................. 30
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_utils-angle ............................................ 66
C_utils-time .............................................. 67
D_as.data.frameD-methods ................................ 68
D_as.data.frameI-methods ................................ 68
D_as.data.frameM-methods ................................ 69
D_as.data.frameY-methods ................................ 70
D_as.zooD-methods ....................................... 70
D_as.zooI-methods ........................................ 71
D_as.zooM-methods ........................................ 72
D_as.zooY-methods ........................................ 73
D_compare-methods ....................................... 73
D_getData-methods ...................................... 74
D_get0-methods .......................................... 75
D_getLat-methods ......................................... 75
D_indexD-methods ........................................ 76
D_indexI-methods ......................................... 76
D_indexRep-methods ....................................... 77
D_levelplot-methods ..................................... 77
D_Losses-methods ....................................... 78
D_mergesolaR-methods .................................... 79
D_shadeplot-methods ...................................... 80
D_window-methods ........................................ 80
Description

The solaR package allows for reproducible research both for photovoltaics (PV) systems performance and solar radiation. It includes a set of classes, methods and functions to calculate the sun geometry and the solar radiation incident on a photovoltaic generator and to simulate the performance of several applications of the photovoltaic energy. This package performs the whole calculation procedure from both daily and intradaily global horizontal irradiation to the final productivity of grid-connected PV systems and water pumping PV systems.

Details

solaR is designed using a set of S4 classes whose core is a group of slots with multivariate time series. The classes share a variety of methods to access the information and several visualization methods. In addition, the package provides a tool for the visual statistical analysis of the performance of a large PV plant composed of several systems.

Although solaR is primarily designed for time series associated to a location defined by its latitude/longitude values and the temperature and irradiation conditions, it can be easily combined with spatial packages for space-time analysis.

The best place to learn how to use the package is the companion paper published by the Journal of Statistical Software: http://www.jstatsoft.org/v50/i09/

Please note that this package needs to set the timezone to UTC.

You can check it after loading solaR with:
Sys.getenv('TZ')
If you need to change it, use:
Sys.setenv(TZ = 'YourTimeZone')

Index of functions and classes:

G0-class Class "G0": irradiation and irradiance on the horizontal plane.
Gef-class Class "Gef": irradiation and irradiance on the
generator plane.

HQCurve  H-Q curves of a centrifugal pump
Meteo-class  Class "Meteo"
NmPVPS  Nomogram of a photovoltaic pumping system
ProdGCPV-class  Class "ProdGCPV": performance of a grid connected PV system.
ProdPVPS-class  Class "ProdPVPS": performance of a PV pumping system.
Shade-class  Class "Shade": shadows in a PV system.
Sol-class  Class "Sol": Apparent movement of the Sun from the Earth
aguiar  Markov Transition Matrices for the Aguiar et al. procedure
as.data.frameD  Methods for Function as.data.frameD
as.data.frameI  Methods for Function as.data.frameI
as.data.frameM  Methods for Function as.data.frameM
as.data.frameY  Methods for Function as.data.frameY
as.zooD  Methods for Function as.zooD
as.zooI-methods  Methods for Function as.zooI
as.zooM  Methods for Function as.zooM
as.zooY  Methods for Function as.zooY
calcGØ  Irradiation and irradiance on the horizontal plane.
calcGef  Irradiation and irradiance on the generator plane.
calcShd  Shadows on PV systems.
calcSol  Apparent movement of the Sun from the Earth
compare  Compare GØ, Gef and ProdGCPV objects
compareLosses  Losses of a GCPV system
corrFDkt  Correlations between the fraction of diffuse irradiation and the clearness index.
d2r  Conversion between angle units.
diff2Hours  Small utilities for difftime objects.
fBTD  Daily time base
fCompD  Components of daily global solar irradiation on a horizontal surface
fCompI  Calculation of solar irradiance on a horizontal surface
fInclin  Solar irradiance on an inclined surface
fProd  Performance of a PV system
fPump  Performance of a centrifugal pump
fSolD  Daily apparent movement of the Sun from the Earth
fSolI  Instantaneous apparent movement of the Sun from the Earth
fSombra  Shadows on PV systems
fTemp  Intradaily evolution of ambient temperature
fTheta  Angle of incidence of solar irradiation on a
inclined surface

getdata Methods for function getData
getG0 Methods for function getG0
getLat Methods for Function getLat
helios Daily irradiation and ambient temperature from
the Helios-IES database
hour Utilities for time indexes.
indexD Methods for Function indexD
indexI Methods for Function indexI
indexRep-methods Methods for Function indexRep
levelplot-methods Methods for function levelplot.
local2Solar Local time, mean solar time and UTC time zone.
mergesolarR Merge solaR objects
optimShd Shadows calculation for a set of distances
between elements of a PV grid connected plant.
prodEx Productivity of a set of PV systems of a PV
plant.
prodGCPV Performance of a grid connected PV system.
prodPVPS Performance of a PV pumping system
pumpCoef Coefficients of centrifugal pumps.
readBD Daily or intradaily values of global horizontal
irradiation and ambient temperature from a
local file or a data.frame.
readG0dm Monthly mean values of global horizontal
irradiation.
shadeplot Methods for Function shadeplot
solaR.theme solaR theme
window Methods for extracting a time window
writeSolar Exporter of solaR results
xyplot-methods Methods for function xyplot in Package 'solaR'

Author(s)

Oscar Perpiñán Lamigueiro
Maintainer: Oscar Perpiñán Lamigueiro <oscar.perpinan@gmail.com>

---

Apparent movement of the Sun from the Earth

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 fBtd. 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 seq.POSIXt. It is not considered if BTi is provided.

BTi  Intradaily time base, a POSIXct object to be used by fSolI. It could be the index of the G0I argument to calcG0.

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


• Spencer, Search 2 (5), 172, http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html

• Strous: http://aa.quae.nl/en/reken/zonpositie.html


Examples

BTD=fBTD(mode='serie')
lat=37.2
sol=calcSol(lat, BTD[100])
print(as.zooD(sol))
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,
       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**
  A character string, describes the kind of source data of the global irradiation and ambient temperature.
  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.
  modeRad='aguiar' uses a set of 12 monthly average values (provided with dataRad) and produces a synthetic daily irradiation time series following the procedure by Aguiar etal. (see reference below).
  If modeRad='bd' the information of daily 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).
  If modeRad='bdi' the information of intradaily irradiation is read from a file, a data.frame defined by dataRad, a zoo or a Meteo object. (See readBDi, df2Meteo and zoo2Meteo for details).
dataRad

- If modeRad='prom' or modeRad='aguIar', a numeric with 12 values or a named list whose components will be processed with readG0dm.
- If modeRad='bd' a character (name of the file to be read with readBD), a data.frame (to be processed with df2Meteo), a zoo (to be processed with zoo2Meteo), a Meteo object, or a list as defined by readBD, df2Meteo or zoo2Meteo. The resulting object will include a column named Ta, with information about ambient temperature.
- If modeRad='bdI' a character (name of the file to be read with readBDi), a data.frame (to be processed with dfI2Meteo), a zoo (to be processed with zoo2Meteo), a Meteo object, or a list as defined by readBDi, dfI2Meteo or zoo2Meteo. The resulting object will include a column named Ta, with information about ambient temperature.

sample

- A 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

- A character, method for the sun geometry calculations. See calcSol, fSolD and fSolI.

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 FdKtPge is selected with corr='Page' while the FdKtCPR with corr='CPR'.

- If corr='user' the use of a correlation defined by a function f is possible.

- 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='bd', to Page when modeRad='prom' and to BRL when modeRad='bdI'.

f

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

Value

- A G0 object.

Author(s)

- Oscar Perpiñán Lamigueiro.

References


See Also
calcSol, fCompD, fCompI, readG0dm, readBD, readBDi, 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)

## Aguiar et al.

g0 <- calcG0(lat=37.2, modeRad='aguiar', dataRad=G0dm)
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)

## Not run:
##NREL-MIDC
##La Ola, Lanai
##Latitude: 20.76685o North
##Longitude: 156.92291o West
##Elevation: 381 meters AMSL
A3_calcGef

Iradiation 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 intraday 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,
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 perfor-
mance 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
Information about the source data of the global irradiation. See calcG0 for
details.
sample, keep.night
See calcSol for details.
sunGeometry character, method for the sun geometry calculations. See calcSol, fSoL0 and
fSol1.
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 effective irradiance
are set to zero. HCPV is the acronym of High Concentration PV system.
Value
A Gef object.

Author(s)
Oscar Perpiñán Lamigueiro.

References

See Also
calcG0, fTheta, fInclin, calcShd.

Examples
```R
### 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)

#Fixed surface
gefAguiar <- calcGef(lat=41, modeRad='aguiar', dataRad=G0dm)

#Two-axis tracker, using the previous result.
#'gefAguiar' is internally coerced to a 'G0' object.
gefAguiar2 <- calcGef(lat=41, modeRad='prev', dataRad=gefAguiar, modeTrk='two')
```
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

```r
prodGCPV(lat,  
    modeTrk='fixed',  
    modeRad='prom',  
    dataRad,  
    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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>lat</code></td>
<td>numeric, latitude (degrees) of the point of the Earth where calculations are needed. It is positive for locations above the Equator.</td>
</tr>
</tbody>
</table>
A character string, describing the tracking method of the generator. See `calcGef` for details.

Information about the source data of the global irradiation. See `calcG0` for details.

See `calcSol` for details.

See `calcG0` for details.

See `calcSol`, `fSol0` and `fSol1`.

See `calcG0` for details.

See `calcSol`, `fSol0`, `fSol1`.

See `calcG0` for details.

See `calcG0` for details.

See `calcSol` for details.

See `calcG0` for details.

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)
- **CoeffVT**: 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).

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)

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/m²) (default value 20 W/m²)

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, 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, 2012. (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 ~ mon, data = ComparePac, 
    type = 'l', auto.key = list(space = 'right', lines = TRUE, points = FALSE), ylab = 'Pac')

### Use of modeRad = 'aguiar' and modeRad = 'prev'
prodAguiarFixed <- prodGCPV(lat = 41, 
    modeRad = 'aguiar', 
    dataRad = GGdm, 
    keep.night = FALSE)

### We want to compare systems with different effective irradiance 
# So we have to convert prodAguiarFixed to a 'G0' object.
G0Aguiar = as(prodAguiarFixed, 'G0')

prodAguiar2x <- prodGCPV(lat = 41, modeTrk = 'two', modeRad = 'prev', dataRad = G0Aguiar)
prodAguiarHoriz <- prodGCPV(lat = 41, modeTrk = 'horiz', modeRad = 'prev', 
    dataRad = G0Aguiar)

### Comparison of yearly values
compare(prodAguiarFixed, prodAguiar2x, prodAguiarHoriz)
compareLosses(prodAguiarFixed, prodAguiar2x, prodAguiarHoriz)

### Compare of daily productivities of each tracking system
compareYf <- mergesoalR(prodAguiarFixed, prodAguiar2x, prodAguiarHoriz)
xyplot(compareYf, superpose = TRUE, 
    ylab = 'kWh/kWp', main = 'Daily productivity', auto.key = list(space = 'right'))

### 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,
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**

```r
prodPVPS(lat, modeTrk='fixed',
```
modeRad='prom',
dataRad,
sample='hour',
keep.night=TRUE,
sunGeometry='michalsky',
corr, f,
betalpha=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 Information about the source data of the global irradiation. See calcg0 for details.

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.

betalpha, beta, 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 K1, 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

OhmAAC 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*


*See Also*

`nmgPVPS`, `fPump`, `pumpCoef`

*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*

```r
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

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**

```r
optimShd(lat, 
  modeTrk='fixed', 
  modeRad='prom', 
  dataRad, 
  sample='hour', 
  keep.night=TRUE, 
  sunGeometry='michalsky', 
  betaLim=90, beta=abs(lat)-10, alfa=0, 
  s=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 Information about the source data of the global irradiation. See calcG0 for details. For this function the option modeRad='bdf' is not supported.

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 effective 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)

CoeffVT 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/m²) (default value 20 W/m²)

**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

OhmA C 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. Energía Solar Fotovoltaica, 2012. (http://procomun.wordpress.com/documentos/librosf/)

See Also

prodGCPV, calcShd

Examples

library(lattice)
library(latticeExtra)

l=37.2;
G@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=l, dataRad=prom, modeTrk='two',
               modeShd=c('area','prom'), distances=dist2x, struct=struct2x, res=5)

shadeplot(ShdM2x)

pLew=xypplot(Yf=GRR, data=ShdM2x, groups=factor(Lew), type=c('l','g'),
               main='Productivity for each Lew value')
Daily or intradaily values of global horizontal irradiation and ambient temperature from a local file or a data.frame.
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), data.frame (df2Meteo and dfI2Meteo) or zoo (zoo2Meteo) which the data are to be read from. It should contain a column G0 with daily (readBD and df2Meteo) or intradaily (readBDi and dfI2Meteo) values of global horizontal irradiation (Wh/m²). It should also include a column named Ta with values of ambient temperature. However, if the object is only a vector with irradiation values, it will converted to a zoo with two columns named G0 and Ta (filled with constant values).

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 which contains the dates of the time series.
time.col character string with the name of the column which 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, 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)

---

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(17,14,15,15,10,18,18,18,19,18,13),
        source='')
Arguments

- **G0dm** numeric, 12 values of monthly means of daily global horizontal irradiation (Wh/m²).
- **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

readBD

Examples

```r
G0dm=c(2.766, 3.491, 4.494, 5.912, 6.989, 7.742, 7.919, 7.927, 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)
```

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 (G0, 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.
B2_Sol-class

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, readBD1, zoo2Meteo, df2Meteo, dfI2Meteo, readG0dm.

---

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, 2012. (http://procomun.wordpress.com/documentos/libroesf/)

See Also

G0, 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 intraday 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"): accessor 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, 2012. (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, FTr: 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 degrees to radians.
  - Alfa: numeric, azimuth angle of the surface (radians). When `modeTrk='fixed'` it is the value of the argument `alpha` converted from degrees 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"): accesor for the match slot.
getLat signature(object = "Gef"): latitude of the inherited Sol object.
xystat signature(x = "Gef", data = "missing"): display the time series of daily values of irradiation.
xystat signature(x = "formula", data = "Gef"): displays the contents of a Gef object with the xystat method for formulas.

Author(s)
Oscar Perpiñán Lamigueiro.

References
See Also

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:
- \( T_c \): cell temperature, °C.
- \( \text{Voc, Isc, Vmpp, Impp} \): open circuit voltage, short circuit current, MPP voltage and current, respectively.
- \( \text{Vdc, Idc} \): voltage and current at the input of the inverter.
- \( \text{Pdc} \): power at the input of the inverter, W
- \( \text{Pac} \): power at the output of the inverter, W
- \( \text{EffI} \): 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

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


See Also

Sol, G0, Gef, Shade.
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 hydraulical 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.

`prodm`: 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

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

• Perpiñán, O, Energía Solar Fotovoltaica, 2012. (http://procomun.wordpress.com/documentos/libroesf/)

See Also

prodPVPS, fPump.

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


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


See Also

`Gef`, `ProdGCPV`.
Correlations between the fraction of diffuse irradiation and the clearness index.

Description

A set of correlations between the fraction of diffuse irradiation and the clearness index used by \texttt{fCompD} and \texttt{fCompI}.

Usage

\begin{verbatim}
## 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)
\end{verbatim}

Arguments

\begin{itemize}
    \item \texttt{Ktd} A numeric, the daily clearness index.
    \item \texttt{kt} A numeric, the intradaily clearness index.
    \item \texttt{sol} A \texttt{sol} object provided by \texttt{calcSol} or a \texttt{zoo} object provided by \texttt{fSolD} or \texttt{fSolI}.
\end{itemize}

Value

A numeric, the diffuse fraction.

Author(s)

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

References

\begin{itemize}
\end{itemize}


See Also

fCompD, fCompI

Examples

Ktd=seq(0, 1, .01)
Monthly=data.frame(Ktd=Ktd)
Monthly$Page=fdktPage(Ktd)
Monthly$LI=fdkTLJ(Ktd)

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

Ktd=seq(0, 1, .01)
Daily=data.frame(Ktd=Ktd)
Daily$CPR=FdktCPR(Ktd)
Daily$CLlMEd=FdktCLlMEd(Ktd)

xyplot(CPR+CLlMEd~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


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]
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

\[
\text{fCompD(sol, G0d, corr = "CPR", f)}
\]

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`, or a zoo object containing daily global irradiation (Wh/m²) on a horizontal surface. See below for `corr = 'none'`.
- **corr**: A character, the correlation between the fraction of diffuse irradiation and the clearness index to be used.
  - With this version several options are available, as described in `corrFdt`. For example, the `FdtKtPage` is selected with `corr = 'Page'` and the `FdtKtCPR` 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/m²)
- **D0d**: numeric, the diffuse irradiation on a horizontal surface (Wh/m²)
- **B0d**: numeric, the direct irradiation on a horizontal surface (Wh/m²)

Author(s)

Oscar Perpiñán Lamigueiro
C_fCompI

References


See Also

fCompI

Examples

lat=37.2;
BTd=fBTd(mode='serie')
SolD<-fSolD(lat, BTd[100])

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

#define a function fXtd with the correlation of CPR
fXtd=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(SolD,G0d, corr = "user", f=fXtd)

lat=37.2;
SolDs<-fSolD(lat, BTd[283])
G0d=zoo(5000, index(SolDs))
fCompD(SolDs, 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)
solD<-fSolD(lat,fBTd(mode='prom'))
fCompD(solD, 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 fSoI.

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, df12Meteo or zoo2Meteo, or a zoo object containing intradaily global irradiation (Wh/m²) on a horizontal surface.
See below for corr='none'.
corr A character, the correlation between the the fraction of intradaily diffuse irradiation and the clearness index to be used. It is ignored if G0I is not provided.
With this version several correlations are available, as described in corrFdKt.
You should choose one of intradaily proposals. For example, the FdKtCLIMEdh is selected with corr='CLIMEdh'.
If corr='user' the use of a correlation defined by a function f is possible.
If corr='none' the G0I object must include information about global, diffuse and direct intradaily irradiation with columns named G0, D0 and B0, respectively.
f A function defininig a correlation between the fraction of diffuse irradiation and the clearness index. It is only neccessary when corr='user'

Value

A zoo with these components:

kt numeric, clearness index.
fd numeric, diffuse fraction.
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)

Author(s)

Oscar Perpiñán Lamigueiro.

References

- Perpiñán, O, Energía Solar Fotovoltaica, 2013. (http://procomun.wordpress.com/documentos/libroesf/)

See Also

fCompD, fSolI, calcSol, corrFdKt.
**C_fInclin**

**Solar irradiance on an inclined surface**

**Examples**

```r
lat <- 37.2
td <- fBld(mode='serie')
solD <- fSolD(lat, Bld[100])
solI <- fSolI(solD, sample='hour')
G0d <- zoo(5000, index(solD))
compD <- fCompD(solD, G0d, corr = "Page")
CompI(solI, compD)

sol <- calcSol(lat, fBld(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, Ta = Ta, lat = lat)
compD <- fCompD(sol, BD, corr = 'Page')
CompI(sol, compD)

head(CompI)

## Use of 'corr'. The help page of calcG0 includes additional examples
## with intradaily data xyplot(fd ~ kt, data=CompI)

climed <- fCompI(sol, G0I=CompI, corr = 'CLIMEDH')
xyplot(fd ~ kt, data=climed)

ekdh <- fCompI(sol, G0I=CompI, corr = 'EKDH')
xyplot(fd ~ kt, data=ekdh)

brl <- fCompI(sol, G0I=CompI, corr = 'BRL')
xyplot(fd ~ kt, data=brl)
```

**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**

```r
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/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, FTr Factor of angular losses for the direct, diffuse and albedo components

Author(s)

Oscar Perpiñán Lamigueiro.


C_fProd

References


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/m²) 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/m²) (default value 20 W/m²)

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, Vmp, Imp open circuit voltage, short circuit current, MPP voltage and current, respectively, in the conditions of irradiance and temperature provided by Inclin

Vdc, Idc voltage and current at the input of the inverter. If no voltage limitation occurs (according to the values of inverter$Vmax and inverter$Vmin), their values are identical to Vmp and Imp. 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


See Also

fInclin, prodGCPV, fTemp.

Examples

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

#using default values
fProd(inclin)

#Using a matrix for Ki (voltage dependence)
invin=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=invin)

#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*Impp))
```

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

```r
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 `splinefun` relating flow and electrical power
- `fPb`: Function constructed with `splinefun` relating pump shaft power and electrical power of the motor
- `fPh`: Function constructed with `splinefun` relating hydraulic power and electrical power of the motor
- `fFreq`: Function constructed with `splinefun` relating frequency and electrical power of the motor

Author(s)

Oscar Perpiñán Lamigueiro.

References


See Also

`NmgPVPS, prodPVPS, pumpCoef, splinefun`.

Examples

```r
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)
})
```


\begin{verbatim}
Pb=fPb(Pac)
etam=Pb/Pac
Ph=fPh(Pac)
etab=Ph/Pb
f=fFreq(Pac)
Q=fQ(Pac)
result=data.frame(Q,Pac,Pb,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+layer(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+layer(panel.text(x[length(x)], y[length(x)], lab[group.number], pos=3))

#Flow and electrical power
xyplot(Q~Pac,data=SP8A44,type='l')
\end{verbatim}

---

\textbf{Daily apparent movement of the Sun from the Earth}

\section*{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.

\section*{Usage}

```r
fSolD(lat, BTd, method='michalsky')
```

\section*{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 \texttt{fBTd}.

- **method** 
  Character, method for the sun geometry calculations to be chosen from 'cooper', 'spencer', 'michalsky' and 'strous'. See references for details.

\section*{Value}

A \texttt{zoo} object with these components:

- **decl** 
  Declination angle (radians) for each day of year in \texttt{dn} or \texttt{BTd}

- **eo** 
  Factor of correction due the eccentricity of orbit of the Earth around the Sun.
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.

Extra-atmospheric daily irradiation (watt-hour per squared meter) incident on a horizontal surface

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

- Spencer, Search 2 (5), 172, http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html

Examples

```r
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)```
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 G0I argument to calcG0. 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 azimuth angle (radians)
- `Als` numeric, solar elevation angle (radians)
- `BoP` 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

- Spencer, Search 2 (5), 172. [http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html](http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html)

See Also

`fSolI`

Examples

```r
### 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(solDs, 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(A1S)-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.
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'

prom logical, only needed for two-axis tracker mode. If TRUE the shadows are averaged between the set of trackers defined by struct$Nrow and struct$Ncol

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 the 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.
C_fTemp

Intradaily evolution of ambient temperature

Description

From the maximum and minimum daily values of ambient temperature, its evolution is 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 `readBD` function. 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**


**See Also**

- `calcSol`, `readBD`.

---

**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

\[
f_{\text{Theta}}(\text{sol}, \beta, \alpha = 0, \text{modeTrk} = "\text{fixed}"\text{, betaLim} = 90, \\
\text{BT} = \text{FALSE, struct, dist})
\]

Arguments

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

Value

A zoo object with these components:

- **Beta** \(\text{numeric, inclination angle of the surface (radians). When modeTrk='fixed' it is the value of the argument beta converted from degrees to radians.}\)
- **Alfa** \(\text{numeric, azimuth angle of the surface (radians). When modeTrk='fixed' it is the value of the argument alfa converted from degrees to radians.}\)
- **costheta** \(\text{numeric, cosine of the incidence angle of the solar irradiance on the surface}\)

Author(s)

Oscar Perpiñán Lamigueiro.
References


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**  
  
  A 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


See Also

NmgPVPS, prodPVPS, pumpCoef.

Examples

```r
library(lattice)
library(latticeExtra)
data(pumpCoef)
CoeffSP8A44<-subset(pumpCoef, Qn==8&stages==44)
CurvaSP8A44<-HQcurve(pump=CoeffSP8A44)
```

---

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

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

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


Examples

```r
  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)
```
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 irradiation incident on the generator (Wh/m²)
- **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

**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
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 `hour(x)+minute(x)/60+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.
Methods for Function `as.data.frameD`

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

**Usage**
```r
## 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

Methods for Function `as.data.frameI`

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

**Usage**
```r
## 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

Description

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

Usage

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

Arguments

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

Methods

signature(object = "GØ") 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 GØ object (or a extended class) into a `data.frame` with yearly values.

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

Arguments
- `object`: A GØ object (or extended.)
- `complete`: A logical.

Methods
- `signature(object = "GØ")`: 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, GØ, Gef, ProdGCPV or ProdPVPS object into a `zoo` object with daily values.

Usage
```r
## 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 G0I 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, Ecd 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, G0I and solI. If day=TRUE the daily values (slots GefD, G0D 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, G0I and solI. If day=TRUE the daily values (slots prodD, GefD, G0D 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, G0I and solI. If day=TRUE the daily values (slots prodD, GefD, G0D and solD) are also included.)

Author(s)
Oscar Perpiñán Lamigueiro

---

D_as.zoomM-methods  
Methods for Function as.zoomM

Description
Convert a G0, Gef, ProdGCPV or ProdPVPS object into a zoo object with monthly average of daily values.

Usage

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

Arguments

- `object`: A G0 object (or extended.)
- `complete`: A logical.

Methods

- signature(object = "G0") 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**

```r
## 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**

```r
## 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=(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)
Methods

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

Author(s)

Oscar Perpiñán Lamigueiro

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

Description

Latitude angle of solar objects.

Usage

gLat(object, units='rad')

Arguments

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

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 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
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

```r
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


See Also

`fInclin`, `fProd`

Examples

```r
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)
```
D_mergesolarR-methods

## Description
Merge the daily time series of solaR objects

## Usage
```r
## S4 method for signature 'G0'
mergesolar(...)  
```

## Arguments
```r
...  
```

## 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:

```r
signature(...) = "Meteo"  
signature(...) = "G0"  
signature(...) = "Gef"  
signature(...) = "ProdGCPV"  
signature(...) = "ProdPVPS"
```

## Examples
```r
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)  
```
Methods for Function \texttt{shadeplot}

**Description**

Visualization of the content of a Shade object.

**Methods**

\texttt{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

---

Methods for extracting a time window

**Description**

Method for extracting the subset of a solaR object whose daily time index (\texttt{indexD}) is comprised between the times \( i \) and \( j \).

**Usage**

```r
## S4 method for signature 'Meteo'
\texttt{x[i, j, ..., drop = TRUE]}
## S4 method for signature 'Sol'
\texttt{x[i, j, ..., drop = TRUE]}
## S4 method for signature 'G0'
\texttt{x[i, j, ..., drop = TRUE]}
## S4 method for signature 'Gef'
\texttt{x[i, j, ..., drop = TRUE]}
## S4 method for signature 'ProdGCPV'
\texttt{x[i, j, ..., drop = TRUE]}
## S4 method for signature 'ProdPVPS'
\texttt{x[i, j, ..., drop = TRUE]}
```
Arguments

\( x \)  
A Meteo, Sol, etc. object.

\( i \)  
an index/time value (Date or POSIXct classes) defining the start of the time window.

\( j \)  
an index/time value (Date or POSIXct classes) defining the end of the time window.

..., drop  
Additional arguments for window.zoo

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.Date(indexD(sol)[1])
end <- start + 30

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

object  A Sol object (or extended.)
file    A character with the name of the file.
complete A logical. Should all the variables be exported?
day     A logical. Should be daily values included in the intradaily file?
timescales A character. Use 'i' to export intradaily values, 'd' for daily values, 'm' for monthly values and 'y' for yearly values. A different file will be created for each choice.
sep     The field separator character.
...     Additional arguments for write.zoo

Methods

signature(object = "Sol")  This function exports the slots with results using write.zoo. 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, G0I and solI. If day=TRUE the daily values (slots GefD, G0D 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, G0I and solI. If day=TRUE the daily values (slots prodD, GefD, G0D 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, G0I and solI. If day=TRUE the daily values (slots prodD, GefD, G0D and solD) are also included.

Author(s)

Oscar Perpiñán Lamigueiro

See Also

write.zoo, read.zoo, as.zooI, as.zooD, as.zooM, as.zooY

Examples

lat <- 37.2;
G0dm <- c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 5362, 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)
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.zoo1(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.zoo1(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.zoo(x, complete=FALSE)`. Therefore, the content of the G0 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 = "ProdPVPVS", 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 `xyplot` function. Once again, the S3 method for class formula is used.

Author(s)
Oscar Perpiñán Lamigueiro

---

E_aguiar

Markov Transition Matrices for the Aguiar et al. procedure

Description

Markov Transition Matrices and auxiliary data for generating sequences of daily radiation values.

Usage
data(MTM)

Format

MTM is a data.frame with the collection of Markov Transition Matrices defined in the paper "Simple procedure for generating sequences of daily radiation values using a library of Markov transition matrices", Aguiar et al., Solar Energy, 1998. Ktlim (matrix) and Ktm (vector) are auxiliary data to choose the correspondent matrix of the collection.

---

E_helios

Daily irradiation and ambient temperature from the Helios-IES database

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 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**  
*Productivity of a set of PV systems of a PV plant.*

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


---

**E_pumpCoef**  
*Coefficients of centrifugal pumps.*

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/App/WebCAPS/custom?

References


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'

E_solaR.theme solaR theme
**Defunct functions in package ‘solaR’**

**Description**

These functions are defunct and no longer available.

**Details**

`readSIAR`: The SIAR webpage cannot be accessed with a direct URL but using javascript code. Therefore, the function `readSIAR` no longer works. This help page is still here as a reference. The SIAR webpage is now [http://eportal.magrama.gob.es/websiar](http://eportal.magrama.gob.es/websiar).

**Deprecated functions in package ‘solaR’**

**Description**

These functions are provided for compatibility with older versions of ‘solaR’ only, and will be defunct at the next release.

**Details**

The following functions are deprecated and will be made defunct; use the replacement indicated below:

- `TargetDiagram, analyzeData`: Use the `tdr` package
Index

*Topic **classes**
  - B1_Meteo-class, 28
  - B2_Sol-class, 29
  - B3_G0-class, 30
  - B4_Gef-class, 32
  - B5_ProdGCPV-class, 34
  - B6_ProdPVPS-class, 36
  - B7_Shade-class, 37

*Topic **constructors**
  - A1_calcSol, 5
  - A2_calcG0, 7
  - A3_calcGef, 10
  - A4_prodGCPV, 13
  - A5_prodPVPS, 17
  - A6_calcShd, 19
  - A7_optimShd, 21
  - A8_readBD, 25
  - A8_readG0dm, 27

*Topic **datasets**
  - E_aguiar, 84
  - E_helios, 84
  - E_prodEx, 85
  - E_pumpCoef, 85
  - E_solarR.theme, 86

*Topic **methods**
  - D_as.data.frameD-methods, 68
  - D_as.data.frameI-methods, 68
  - D_as.data.frameM-methods, 69
  - D_as.data.frameY-methods, 70
  - D_as.zooD-methods, 70
  - D_as.zooI-methods, 71
  - D_as.zooM-methods, 72
  - D_as.zooY-methods, 73
  - D_compare-methods, 73
  - D_getData-methods, 74
  - D_getG0-methods, 75
  - D_getLat-methods, 75
  - D_indexD-methods, 76
  - D_indexI-methods, 76
  - D_indexRep-methods, 77
  - D_levelplot-methods, 77
  - D_Losses-methods, 78
  - D_mergesolar-methods, 79
  - D_shadeplot-methods, 80
  - D_window-methods, 80
  - D_writeSolar-methods, 81
  - D_xypplot-methods, 83

*Topic **utilities**
  - A1_calcSol, 5
  - A2_calcG0, 7
  - A3_calcGef, 10
  - A4_prodGCPV, 13
  - A5_prodPVPS, 17
  - A6_calcShd, 19
  - A7_optimShd, 21
  - A8_readBD, 25
  - A8_readG0dm, 27
  - C_corrFkt, 39
  - C_fBTd, 40
  - C_fCompD, 42
  - C_fCompI, 43
  - C_fIncl, 45
  - C_fProd, 47
  - C_fPump, 49
  - C_fSo1D, 51
  - C_fSo1I, 53
  - C_fSombra, 55
  - C_fTemp, 57
  - C_fTheta, 58
  - C_HQCurve, 60
  - C_local2Solar, 61
  - C_NmgPVPS, 63
  - C_sample2Diff, 65
  - C_utils-angle, 66
  - C_utils-time, 67
  - [,G0,ANY,ANY-method (D_window-methods), 80
  - [,G0-method (D_window-methods), 80

88
INDEX

[,Gef, ANY, ANY-method (D_window-methods), 80  
[,Gef-method (D_window-methods), 80  
[,Meteo, ANY, ANY-method (D_window-methods), 80  
[,Meteo-method (D_window-methods), 80  
[,ProdGCPV, ANY, ANY-method (D_window-methods), 80  
[,ProdGCPV-method (D_window-methods), 80  
[,ProdPVPS, ANY, ANY-method (D_window-methods), 80  
[,Sol, ANY, ANY-method (D_window-methods), 80  
[,Sol-method (D_window-methods), 80  
A1_calcSol, 5  
A2_calcG6, 7  
A3_calcGef, 10  
A4_prodGCPV, 13  
A5_prodPVPS, 17  
A6_calcShd, 19  
A7_optimShd, 21  
A8_readBD, 25  
A8_readG6dm, 27  
aguiar (E_aguiar), 84  
analyzeData (solar deprecated), 87  
as.data.frame, Shade-method (B7_Shade-class), 37  
as.data.frameD (D_as.data.frameD-methods), 68  
as.data.frameD, Sol-method (D_as.data.frameD-methods), 68  
as.data.frameD-methods (D_as.data.frameD-methods), 68  
as.data.frameI (D_as.data.frameI-methods), 68  
as.data.frameI, Sol-method (D_as.data.frameI-methods), 68  
as.data.frameI-methods (D_as.data.frameI-methods), 68  
as.data.frameM (D_as.data.frameM-methods), 69  
as.data.frameM, G0-method (D_as.data.frameM-methods), 69  
as.data.frameM-methods (D_as.data.frameM-methods), 69  
as.data.frameY (D_as.data.frameY-methods), 70  
as.data.frameY, G0-method (D_as.data.frameY-methods), 70  
as.data.frameY-methods (D_as.data.frameY-methods), 70  
as.POSIXct, 41  
as.zooD, 82  
as.zooD (D_as.zooD-methods), 70  
as.zooD, G0-method (D_as.zooD-methods), 70  
as.zooD, Gef-method (D_as.zooD-methods), 70  
as.zooD, ProdGCPV-method (D_as.zooD-methods), 70  
as.zooD, ProdPVPS-method (D_as.zooD-methods), 70  
as.zooD, Sol-method (D_as.zooD-methods), 70  
as.zooD-methods (D_as.zooD-methods), 70  
as.zooI, 82  
as.zooI (D_as.zooI-methods), 71  
as.zooI, G0-method (D_as.zooI-methods), 71  
as.zooI, Gef-method (D_as.zooI-methods), 71  
as.zooI, ProdGCPV-method (D_as.zooI-methods), 71  
as.zooI, ProdPVPS-method (D_as.zooI-methods), 71  
as.zooI, Sol-method (D_as.zooI-methods), 71  
as.zooI-methods (D_as.zooI-methods), 71  
as.zooM, 82  
as.zooM (D_as.zooM-methods), 72  
as.zooM, G0-method (D_as.zooM-methods), 72  
as.zooM, Gef-method (D_as.zooM-methods), 72  
as.zooM, ProdGCPV-method (D_as.zooM-methods), 72  
as.zooM, ProdPVPS-method (D_as.zooM-methods), 72  
as.zooM-methods (D_as.zooM-methods), 72  
as.zooY, 82  
as.zooY (D_as.zooY-methods), 73  
as.zooY, G0-method (D_as.zooY-methods), 73  
as.zooY, Gef-method (D_as.zooY-methods), 73
as.zooY,ProdGCPV-method (D_as.zooY-methods), 73
as.zooY,ProdPVPS-method (D_as.zooY-methods), 73
as.zooY-methods (D_as.zooY-methods), 73

B1_Meteo-class, 28
B2_Sol-class, 29
B3_G0-class, 30
B4_Gef-class, 32
B5_ProdGCPV-class, 34
B6_ProdPVPS-class, 36
B7_Shade-class, 37

C_ccorrFdkt, 39
C_fbTd, 40
C_fcCompD, 42
C_fcCompI, 43
C_finclin, 45
C_fpProd, 47
C_fpPump, 49
C_fSol0, 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_utils-angle, 66
C_utils-time, 67

calcG0, 6, 10–12, 14, 15, 18, 21, 22, 26, 30, 46, 53
calcG0 (A2_calcG0), 7
calcGef, 14, 15, 18, 19, 32, 45, 47, 58, 60
calcGef (A3_calcGef), 10
calcShd, 10–12, 15, 21, 24, 57
calcShd (A6_calcShd), 19
calcSolar, 7–9, 11, 14, 18, 22, 29, 30, 39, 42, 44, 57–59
calcSolar (A1_calcSolar), 5
CBIND (C_local2Solar), 61
char2diff (C_sample2Diff), 65
compare, 15
compare (D_compare-methods), 73
compare, G0-method (D_compare-methods), 73
compare, Gef-method (D_compare-methods), 73
compare, ProdGCPV-method (D_compare-methods), 73
compare-methods (D_compare-methods), 73
compareLosses, 15
compareLosses (D_Losses-methods), 78
compareLosses, ProdGCPV-method (D_Losses-methods), 78
compareLosses-methods (D_Losses-methods), 78
corrFdkt, 8, 9, 42, 44
corrFdkt (C_corrFdkt), 39
d2h (C_utils-angle), 66
d2r (C_utils-angle), 66
d_as.data.frameD-methods, 68
d_as.data.frameI-methods, 68
d_as.data.frameM-methods, 69
d_as.data.frameY-methods, 70
d_as.zooD-methods, 70
d_as.zooI-methods, 71
d_as.zooM-methods, 72
d_as.zooY-methods, 73
d_compare-methods, 73
d_getData-methods, 74
d_getG0-methods, 75
d_getLat-methods, 75
d_indexD-methods, 76
d_indexI-methods, 76
d_indexRep-methods, 77
d_levelplot-methods, 77
d_Losses-methods, 78
d_mergesolarR-methods, 79
d_shadeplot-methods, 80
d_window-methods, 80
d_writeSolar-methods, 81
d_xyplot-methods, 83
DateTimeClasses, 67
df2Meteo, 7, 8, 29
df2Meteo (A8_readBD), 25
df12Meteo, 7, 8, 29, 44
df12Meteo (A8_readBD), 25
diff2Hours (C_sample2Diff), 65
DoM (C_utils-time), 67
dom (C_utils-time), 67
dotplot, 74, 78
DoY (C_utils-time), 67
doy (C_utils-time), 67
INDEX

dst (C_utils-time), 67
E_aguiar, 84
E_helios, 84
E_prodEx, 85
E_pumpCoeff, 85
E_solarTheme, 86

fcompd, 6, 7, 51
fBtd (C_fBtd), 40
fCompD, 7, 9, 31, 39, 40, 44
fCompD (C_fCompD), 42
fCompI, 7, 9, 31, 39, 40, 43, 47
fCompI (C_fCompI), 43
FktBRL (C_corrFkt), 39
FktCLIMEDd (C_corrFkt), 39
FktCLIMEDh (C_corrFkt), 39
FktCPR, 8, 42
FktCPR (C_corrFkt), 39
FktEKd (C_corrFkt), 39
FktEKdh (C_corrFkt), 39
FktLJ (C_corrFkt), 39
FktPage (C_corrFkt), 39
Finclin, 10, 12, 15, 21, 32, 49, 60, 78
Finclin (C_fInclin), 45
FPDd (C_fPd), 47
FPDd (C_fPd), 47
Fpump, 18, 19, 37, 65
Fpumpp (C_fpump), 49
FSold, 5, 8, 11, 14, 18, 22, 29, 39, 41, 42, 54
FSold (C_FSol), 51
FSoll, 5, 6, 8, 11, 14, 18, 22, 29, 39, 44, 62
FSoll (C_FSol), 53
FSombo1, 19, 56, 60
FSombo1 (C_fSombo1), 55
FSombo2X (C_fSombo2), 55
FSombo6, 20, 23, 56
FSombo6 (C_fSombo6), 55
FSombo3 (C_fSombo3), 55
FSombo4 (C_fSombo4), 55
FSomboHoriz, 56
FSomboHoriz (C_fSombo), 55
Ftemp, 7, 26, 49
Ftemp (C_fTemp), 57
FTheta, 10, 12, 15, 21, 32, 46, 47, 57
FTheta (C_fTheta), 58

G0, 30, 33–36, 38
G0-class (B3_G0-class), 30
Gef, 19, 30, 32, 34–36, 38, 47
Gef-class (B4_Gef-class), 32
data (D_getData-methods), 74
data (D_getData-methods), 74
data (D_getData-methods), 74
getG0 (D_getG0-methods), 75
getG0 (D_getG0-methods), 75
getG0 (D_getG0-methods), 75
getG0 (D_getG0-methods), 75
getLat (D_getLat-methods), 75
getLat (D_getLat-methods), 75
getLat (D_getLat-methods), 75
getLat, Sol-method (D_getLat-methods), 75
getLat, Sol-method (D_getLat-methods), 75
h2d (C_utils-angle), 66
h2r (C_utils-angle), 66
Helios (E_helios), 84
Hms (C_utils-time), 67
Hour (C_utils-time), 67
HQC, 60

IndexD, 80, 81
IndexD (D_IndexD-methods), 76
IndexD, G0-method (D_IndexD-methods), 76
IndexD, Meteo-method (D_IndexD-methods), 76
IndexD, Sol-method (D_IndexD-methods), 76
IndexD, Sol-method (D_IndexD-methods), 76
IndexD-methods (D_IndexD-methods), 76
IndexI (D_IndexI-methods), 76
IndexI, Sol-method (D_IndexI-methods), 76
IndexI-methods (D_IndexI-methods), 76
IndexRep, Sol-method (D_IndexRep-methods), 77
IndexRep-methods (D_IndexRep-methods), 77

Ktlim (E_aguiar), 84
Ktm (E_aguiar), 84

levelplot.formula, G0-method (D_levelplot-methods), 77
levelplot.formula, Meteo-method (D_levelplot-methods), 77
levelplot.formula, Sol-method (D_levelplot-methods), 77
levelplot, formula, zoo-method
   (D_levelplot-methods), 77
levelplot-methods
   (D_levelplot-methods), 77
local2Solar (C_local2Solar), 61
lonHH (C_local2Solar), 61
losses (D_Losses-methods), 78
losses, Gef-method (D_Losses-methods), 78
losses, ProdGCPV-method
   (D_Losses-methods), 78
losses-methods (D_Losses-methods), 78
mergesolar, 15
mergesolarR (D_mergesolarR-methods), 79
mergesolarR, G0-method
   (D_mergesolarR-methods), 79
mergesolarR, Gef-method
   (D_mergesolarR-methods), 79
mergesolarR, Meteo-method
   (D_mergesolarR-methods), 79
mergesolarR, ProdGCPV-method
   (D_mergesolarR-methods), 79
mergesolarR, ProdPVPS-method
   (D_mergesolarR-methods), 79
mergesolarR-methods
   (D_mergesolarR-methods), 79
Meteo, 31, 33, 34, 36, 38, 58
Meteo-class (B1_Meteo-class), 28
minute (C_utils-time), 67
Month (C_utils-time), 67
month (C_utils-time), 67
MTM (E_aguiar), 84
NmgPVPS, 19, 50, 61
NmgPVPS (C_NmgPVPS), 63
optimShd, 37, 57
optimShd (A7_optimShd), 21
P2E (C_sample2Diff), 65
prodEx (E_prodEx), 85
ProdGCPV, 38
prodGCPV, 24, 34, 47, 49
prodGCPV (A4_prodGCPV), 13
ProdGCPV-class (B5_ProdGCPV-class), 34
ProdPVPS, 19
prodPVPS, 36, 37, 50, 61, 65
prodPVPS (A5_prodPVPS), 17
ProdPVPS-class (B6_ProdPVPS-class), 36
pumpCoef, 18, 19, 36, 50, 60, 61, 64, 65
pumpCoef (E_pumpCoef), 85
r2d (C_utils-angle), 66
r2h (C_utils-angle), 66
r2sec (C_utils-angle), 66
read.table, 26, 27
read.zoo, 81, 82
readBD, 7–9, 28, 29, 42, 58, 76
readBD (A8_readBD), 25
readBDi, 7–9, 29, 44
readBDi (A8_readBD), 25
readG0dm, 7–9, 27, 29, 42, 76
readG0dm (A8_readG0dm), 27
readSiar (solarR-defunct), 87
sample2Hours (C_sample2Diff), 65
second (C_utils-time), 67
seq.POSIXt, 6, 8, 22, 41, 53, 65
Shade, 24, 35, 80
Shade-class (B7_Shade-class), 37
shadeplot (D_shadeplot-methods), 80
shadeplot, Shade-method
   (D_shadeplot-methods), 80
shadeplot-methods
   (D_shadeplot-methods), 80
show, G0-method (B3_G0-class), 30
show, Gef-method (B4_Gef-class), 32
show, Meteo-method (B1_Meteo-class), 28
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), 29
Sol, 31–38, 58, 66
Sol-class (B2_Sol-class), 29
solarR (solarR-package), 3
solarR-defunct, 87
solarR-deprecated, 87
solarR-package, 3
solarR.theme (E_solarR.theme), 86
TargetDiagram (solarR-deprecated), 87
truncDay (C_utils-time), 67
window (D_window-methods), 80
window-methods (D_window-methods), 80
window.zoo, 81
write.zoo, 82
writeSolar (D_writeSolar-methods), 81
writeSolar, Sol-method
 (D_writeSolar-methods), 81
writeSolar-methods
 (D_writeSolar-methods), 81

xyplot, formula, G0-method
 (D_xyplot-methods), 83
xyplot, formula, Meteo-method
 (D_xyplot-methods), 83
xyplot, formula, Shade-method
 (D_xyplot-methods), 83
xyplot, formula, Sol-method
 (D_xyplot-methods), 83
xyplot, formula, zoo-method
 (D_xyplot-methods), 83
xyplot, G0, missing-method
 (D_xyplot-methods), 83
xyplot, Meteo, missing-method
 (D_xyplot-methods), 83
xyplot, ProdGCPV, missing-method
 (D_xyplot-methods), 83
xyplot, ProdPVPS, missing-method
 (D_xyplot-methods), 83
xyplot-methods (D_xyplot-methods), 83

Year (C_utils-time), 67
year (C_utils-time), 67

zoo2Meteo, 7, 8, 29, 44
zoo2Meteo (A8_readBD), 25