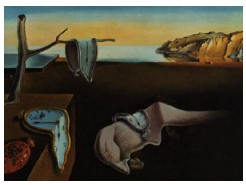

Journal of Statistical Software
August 2012, Volume 50, Issue 9. http://www.jstatsoft.org/

solaR: Solar Radiation and Photovoltaic Systems with R

Oscar Perpiñán Lamigueiro
Universidad Politécnica de Madrid

Abstract

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.

It 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.

Keywords: sun geometry, solar radiation, solar energy, photovoltaic, visualization methods, time series, spatio-temporal data, S4.

1. Introduction

The R (R Development Core Team 2012) package **solaR** 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 (Perpiñán 2012b). This package performs the whole calculation procedure from both *daily* and *intradaily* global horizontal irradiation to the final productivity of grid-connected photovoltaic (PV) systems and water-pumping PV systems.

The package stands on a set of S4 classes. The core of each class is a group of slots with

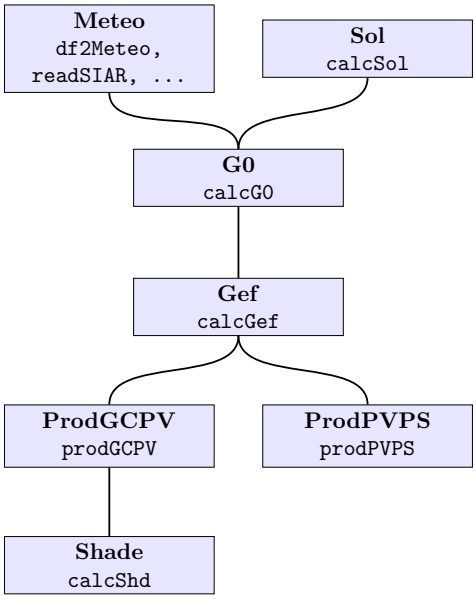

Figure 1: S4 classes and their relationships in the **solaR** package. Each frame contains the name of the class and its constructor function.

yearly, monthly, daily and intradaily multivariate time series constructed with the **zoo** package (Zeileis and Grothendieck 2005). The Figure 1 shows the classes and their relationships. Each frame contains the name of the class and its constructor function:

- `Sol`: Sun geometry. It is created with `calcSol` (Section 2).
- `Meteo`: Meteorological data. It may be created with several functions: `readSIAR`, `zoo2Meteo`, etc. (Section 3.2).
- `G0`: Horizontal irradiation and irradiance (contains classes `Meteo` and `Sol`). It is created with `calcG0` (Section 3.3).
- `Gef`: Effective irradiation and irradiance (contains class `G0`). It is created with `calcGef` (Section 3.4).
- `ProdGCPV`: Performance of a grid-connected photovoltaic system (GCPV, contains class `Gef`). It is created with `prodGCPV` (Section 4).
- `ProdPVPS`: Performance of a photovoltaic pumping system (contains class `Gef`). It is created with `prodPVPS` (Section 5).
- `Shade`: Shadows in a GCPV system (contains class `ProdGCPV`). It is created with `optimShd` (Section 4.3).

The classes share a variety of methods to access the information. For example, `as.zooD` provides a **zoo** object with the daily multivariate time series of the corresponding object. There are several visualization methods based on the **lattice** (Sarkar 2008) and **latticeExtra** (Sarkar and Andrews 2011) packages, using palettes from **ColorBrewer.org** (Harrower and

Brewer 2003) with the **RColorBrewer** package (Neuwirth 2011). Besides, **solaR** provides a tool for the visual statistical analysis of the performance of a large PV plant composed of several systems (Section 7).

The development version of **solaR** can be found at R-Forge (`http://R-Forge.R-project.org/projects/solar/`) and the stable version is at the Comprehensive R Archive Network (`http://CRAN.R-project.org/package=solaR`)

1.1. Other approaches

A variety of software approaches provide solutions for solar radiation and photovoltaic systems calculations:

- The engineer oriented tools are commonly proprietary software with graphical user interfaces and without command-line interface (thus lacking scripting funcionalities). **PVsyst** (**PVsyst** SA 2012) is the most representative product of this set.
- The PVGIS project (Joint Research Centre, European Commission 2012) developed and maintains the `r.sun` model (Hofierka and Súri 2002) in the GRASS GIS open source environment (GRASS Development Team 2012).
- The **SAGA** project (Cimmery 2010) includes a module to calculate the potential incoming solar radiation in an area using different atmospheric models. This module is accesible at R via the **RSAGA** package (Brenning 2011) with the `rsaga.pisr` function.
- The **maptools** package (Lewin-Koh *et al.* 2011) provides several methods (`sun-methods`) for calculating sunrise, sunset, and times of dawn and dusk, using algorithms by the National Oceanic & Atmospheric Administration (NOAA).
- The Measurement and Instrumentation Data Center of the National Renewable Energy Laboratory (2012) publishes two solar position algorithms named `SOLPOS` and `SPA` (Reda and Andreas 2008, 2004).
- Finally, two atmospheric radiative transfer models should be noted:
  - **libRadtran**, a library for radiative transfer (Mayer and Kylling 2005), is a collection of C and Fortran functions and programs for calculation of solar and thermal radiation in the Earth's atmosphere, freely available under the GNU General Public License.
  - **SMARTS2** (Gueymard 2001) and **REST2** (Gueymard 2008) compute clear sky spectral irradiances (including direct beam, circumsolar, hemispherical diffuse, and total on a tilted or horizontal receiver plane) for specified atmospheric conditions with a more simplified approach.

Among these excellent tools, **solaR** allows for reproducible research for sun geometry, radiation on the horizontal plane, effective radiation and performance of photovoltaics systems. In addition, 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 (Section 6).

2. Solar geometry and extraterrestrial irradiation

The apparent movement of the Sun is defined with a set of equations coded in the functions `fSolD` and `fSolI`. `fSolD` computes 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 extraterrestrial irradiation. On the other hand, `fSolI` computes the angles which describe the intradaily apparent movement of the Sun from the Earth.

Besides, the function `fBTd` generates time bases with different structures. For example, the so called "monthly average days" (commonly used when only 12 monthly averages of daily global radiation are available) are obtained with `fBTd(mode = "prom")`. The azimuth and height solar angles during these days are displayed in Figure 2.

```
R> library("solaR")
R> lat <- 37.2
R> SolD <- fSolD(lat, BTd = fBTd(mode = "prom"))
R> SolI <- fSolI(SolD, sample = "10 min", keep.night = FALSE)
R> mon <- month.abb
R> p <- xyplot(r2d(AlS) ~ r2d(AzS), groups = month, data = SolI, type = "l",
+    col = "black", xlab = expression(psi[s]), ylab = expression(gamma[s]))
R> 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)
+  })
```

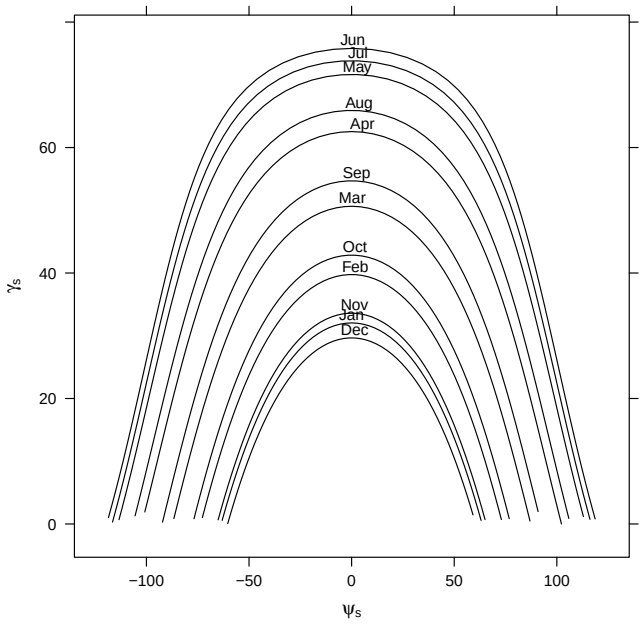

Figure 2: Azimuth and height solar angles during the "average days".

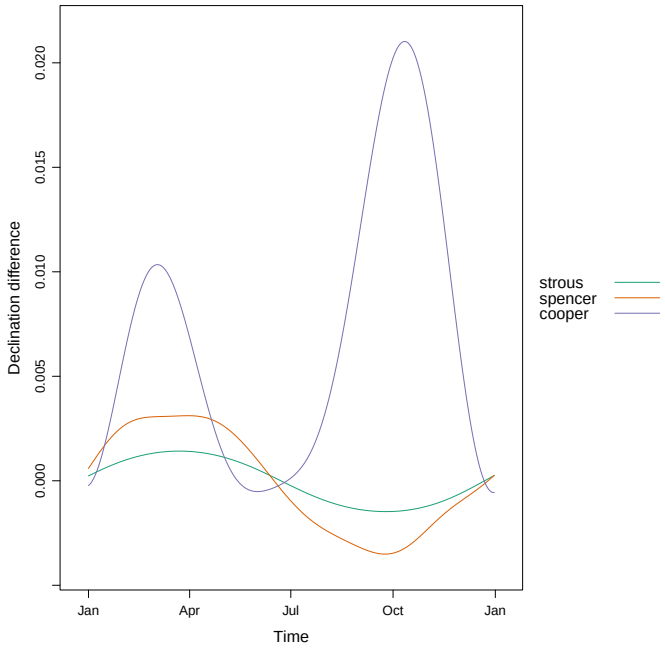

Figure 3: Difference between the "michalsky" method and others to estimate the declination throughout the year.

solaR provides four methods for the sun geometry calculations. These methods are named as "cooper" (Cooper 1969), "spencer" (Spencer 1971), "michalsky" (default) (Michalsky 1988) and "strous" (Strous 2011). Figure 3 displays the difference between the "michalsky" method and others to estimate the declination throughout the year:

```
R> lat <- 37.2
R> BTd <- fBTd(mode = "serie")
R> solStrous <- fSolD(lat, BTd, method = "strous")
R> solSpencer <- fSolD(lat, BTd, method = "spencer")
R> solCooper <- fSolD(lat, BTd, method = "cooper")
R> solMichalsky <- fSolD(lat, BTd, method = "michalsky")
R> decDif <- solMichalsky$decl - cbind(solStrous$decl, solSpencer$decl,
+    solCooper$decl)
R> names(decDif) <- c("strous", "spencer", "cooper")
```

These functions are included in a function, calcSol. It constructs an object of class Sol containing in its slots the zoo objects created by fSolD and fSolI.

3. Solar radiation

Values of global horizontal irradiation are commonly available either as monthly averages of daily values or as a time series of daily values during one or several years. The analysis of the performance of a PV system starts from the transformation of the global horizontal irradiation

to global, diffuse and direct horizontal irradiance and irradiation, and then irradiance and irradiation on the generator surface.

3.1. Irradiation and irradiance on the horizontal plane

The function `fCompD` extracts 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. This function needs the results from `fSolD`, a set of values of global horizontal irradiation (Wh/m^2), and the correlation between the clearness index and the diffuse fraction.

solaR offers predefined correlations for monthly means of daily values (Page 1961; Liu and Jordan 1960), for daily values (Collares-Pereira and Rabl 1979; Erbs, Klein, and Duffie 1982; de Miguel, Bilbao, Aguiar, Kambezidis, and Negro 2001) and for intradaily values (Ridley, Boland, and Lauret 2010). Besides, the user may define a particular correlation through the argument `f`.

```
R> BTd <- fBTd(mode = "serie")
R> SolD <- fSolD(lat, BTd[100])
R> SolI <- fSolI(SolD, sample = "hour")
R> G0d <- zoo(5000, index(SolD))
R> fCompD(SolD, G0d, corr = "Page")
```

```
               Fd    Ktd  G0d  D0d  B0d
2011-04-10 0.4123 0.5201 5000 2062 2938
```

```
R> fCompD(SolD, G0d, corr = "CPR")
```

```
               Fd    Ktd  G0d  D0d  B0d
2011-04-10 0.5658 0.5201 5000 2829 2171
```

The daily profile of the irradiance is obtained with the function `fCompI`. This function needs the information provided by `fCompD` and `fSolI`, or `calcSol`. For example, the profiles for the "monthly average days" are obtained with the next code (Figure 4).

```
R> lat <- 37.2
R> sol <- calcSol(lat, fBTd(mode = "prom"), sample = "hour",
+    keep.night = FALSE)
R> 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
R> 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)
R> BD <- readG0dm(G0dm = G0dm, Ta = Ta, lat = 37.2)
R> compD <- fCompD(sol, BD, corr = "Page")
R> compI <- fCompI(sol, compD)
```

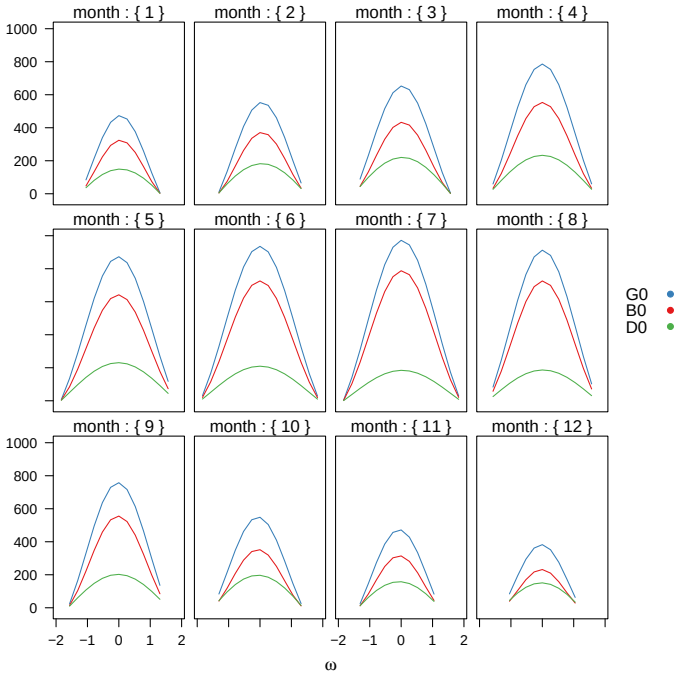

Figure 4: Global (`G0`), diffuse (`D0`), and direct (`B0`) irradiance during the "monthly average days".

3.2. Meteorological data

solaR includes several functions to import radiation and temperature data[1]. For daily data, the functions `readBD` and `df2Meteo` are recommended for local files and `data.frame`, while `readG0dm` is indicated when only 12 monthly averages are available. The correspondent functions for intradaily data are `readBDi` and `dfI2Meteo`. Besides, `zoo2Meteo` can construct a `Meteo` object from a **zoo** object both for daily and intradaily data. The result of these functions is always a `Meteo` object.

For example, the `helios` dataset included in the package, obtained from HELIOS-IES (Instituto de Energía Solar 2012), can be converted to a `Meteo` object with the next code:

```
R> data("helios")
R> names(helios) <- c("date", "G0", "TempMax", "TempMin")
R> bd <- df2Meteo(helios, dates.col = "date", lat = 41, source = "helios-IES",
+    format = "%Y/%m/%d")
R> bd

Object of class  Meteo

Source of meteorological information: bd-helios-IES
Latitude of source:  41 degrees
```

[1]The appendix of Perpiñán (2012b) provides a list of sources of meteorological data.

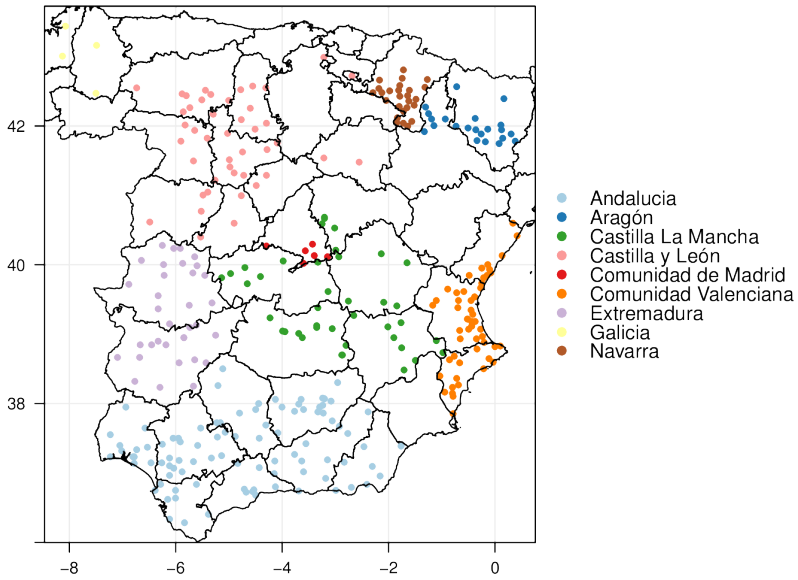

Figure 5: Meteorological stations of the SIAR network as detailed in `http://solar.R-Forge.R-project.org/data/SIAR.csv`.

```
Meteorological Data:
     Index                          G0             TempMax          TempMin
 Min.   :2009-01-01 00:00:00   Min.   :  326   Min.   : 1.41   Min.   :-37.50
 1st Qu.:2009-04-08 12:00:00   1st Qu.: 2523   1st Qu.:14.41   1st Qu.:  1.95
 Median :2009-07-07 00:00:00   Median : 4746   Median :23.16   Median :  7.91
 Mean   :2009-07-04 21:29:54   Mean   : 4812   Mean   :22.59   Mean   :  5.32
 3rd Qu.:2009-10-03 12:00:00   3rd Qu.: 7140   3rd Qu.:31.06   3rd Qu.: 15.11
 Max.   :2009-12-31 00:00:00   Max.   :11254   Max.   :38.04   Max.   : 24.80
```

On the other hand, the function `readSIAR` is able to download the meteorological data from the SIAR network (Ministerio de Agricultura, Alimentación y Medio Ambiente 2012). This web service provides daily measurements from a set of agroclimatic stations located in Spain (Figure 5). With the code of the station and its province (available as supplementary material and at `http://solar.R-Forge.R-project.org/data/SIAR.csv`), and the start and end date, `readSIAR` constructs an object of class `Meteo`. The raw data is obtained with the method `getData`. If only the irradiation series is needed, the method `getG0` is recommended. Both methods provide a **zoo** object. It is important to note that the radiation measurements available at this web page are in MJ/m^2, but `readSIAR` converts the values to Wh/m^2.

For example, the 2009 data from the station at Aranjuez (`prov = 28, est = 3`) is displayed in Figure 6.

```
R> Aranjuez <- readSIAR(28, 3, "01/01/2009", "31/12/2009")
R> xyplot(G0 ~ TempMedia | month, data = Aranjuez, type = c("p", "r"))
```

The SIAR network publishes information of maximum and minimum values of temperature. The function `fTemp` calculates a profile of the ambient temperature with this information following the method proposed in (Huld, Súri, Dunlop, and Micale 2006). The evolution of this synthetic time series of temperature during March is displayed in Figure 7.

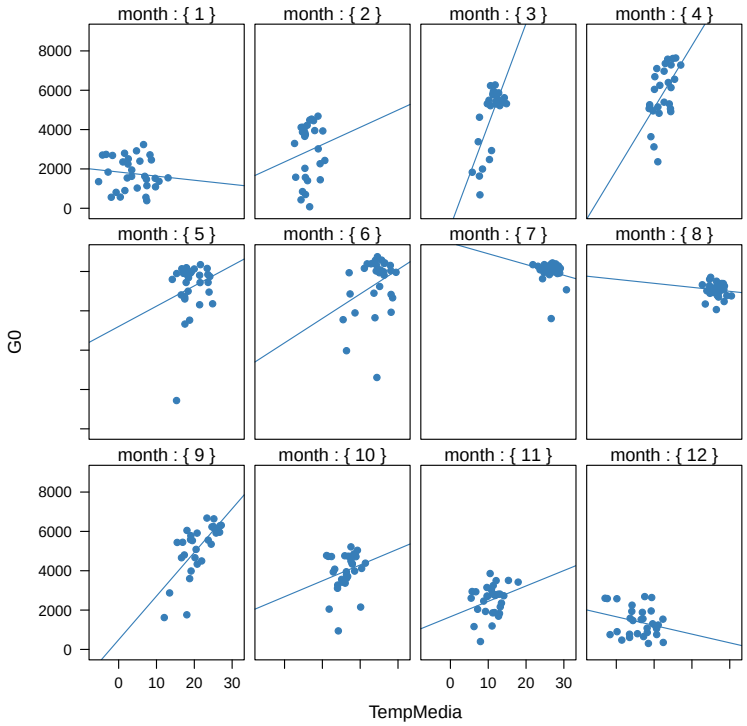

Figure 6: Daily irradiation and mean temperature in the station of Aranjuez.

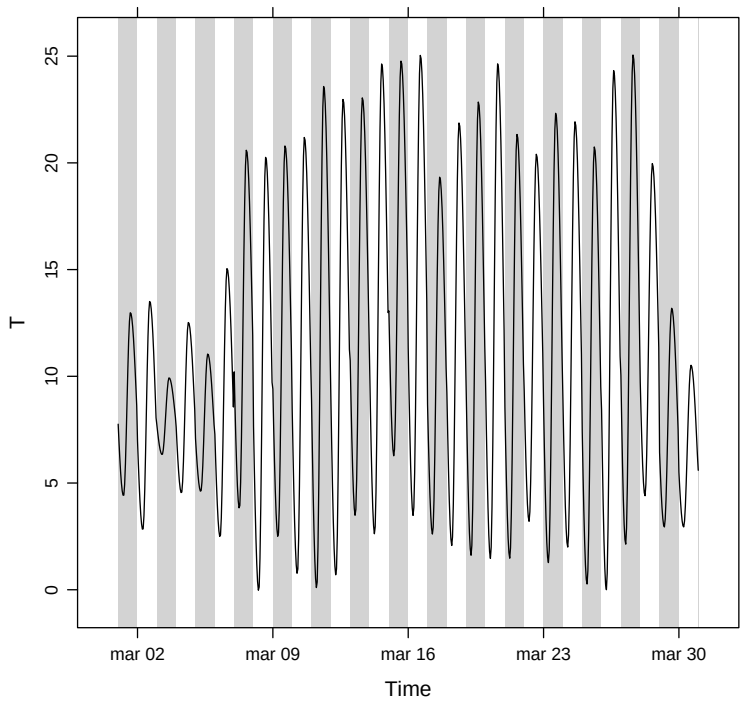

Figure 7: Evolution of the ambient temperature during March 2009 in Aranjuez.

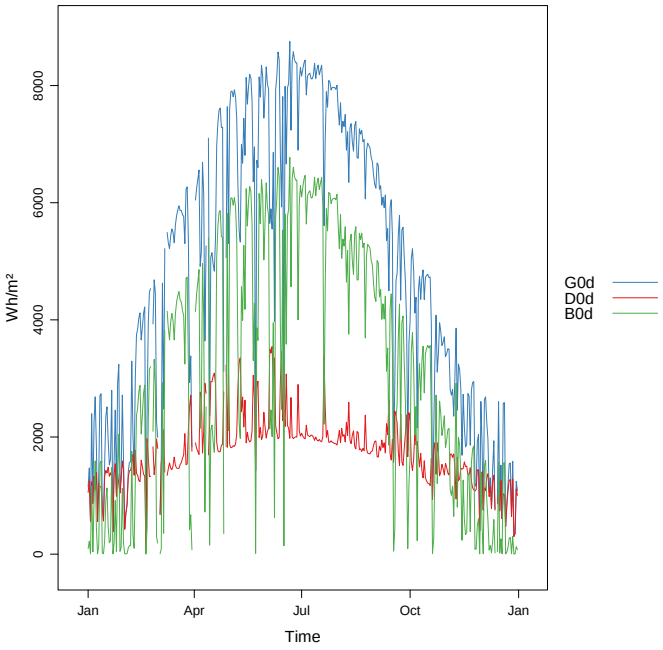

Figure 8: Components of horizontal irradiation (global, `G0d`, direct, `B0d`, and diffuse, `D0d`) calculated with `calcG0`.

```
R> lat <- 41
R> sol <- calcSol(lat, BTd = indexD(Aranjuez), sample = "hour")
R> Temp <- fTemp(sol, Aranjuez)
R> wTemp <- window(Temp, start = as.POSIXct("2009-03-01"),
+    end = as.POSIXct("2009-03-31"))
R> xyplot(wTemp, col = "black", ylab = "T") +
+    layer_(panel.xblocks(x, DoY, col = c("lightgray", "white")))
```

3.3. The function `calcG0`

The previous steps are included in the function `calcG0`, the constructor of the class `G0`. For example, with the next code, the components of horizontal irradiation and irradiance are obtained from the measurements of the meteorological station of Aranjuez (Figure 8).

```
R> g0 <- calcG0(lat = 37.2, modeRad = "siar", dataRad = list(prov = 28,
+    est = 3, start = "01/01/2009", end = "31/12/2009"))
```

solaR accepts intradaily irradiation data sources. For example, the *La Ola - Lanai* station at Hawaii from the Measurement and Instrumentation Data Center (NREL-MIDC) of the National Renewable Energy Laboratory (2012) provides meteorological data with 1-minute sampling rate[2]. The local data logger program runs using Greenwich Mean Time (GMT), and data is converted to Hawaiian Standard Time (HST) after data collection. The

[2]The data for the example are available at http://www.nrel.gov/midc/apps/plot.pl?site=LANAI&start=20090722&edy=19&emo=11&eyr=2010&zenloc=19&year=2010&month=11&day=1&endyear=2010&endmonth=11&endday=19&time=1&inst=3&inst=4&inst=5&inst=10&type=data&first=3&math=0&second=-1&value=0.0&global=-1&direct=-1&diffuse=-1&user=0&axis=1

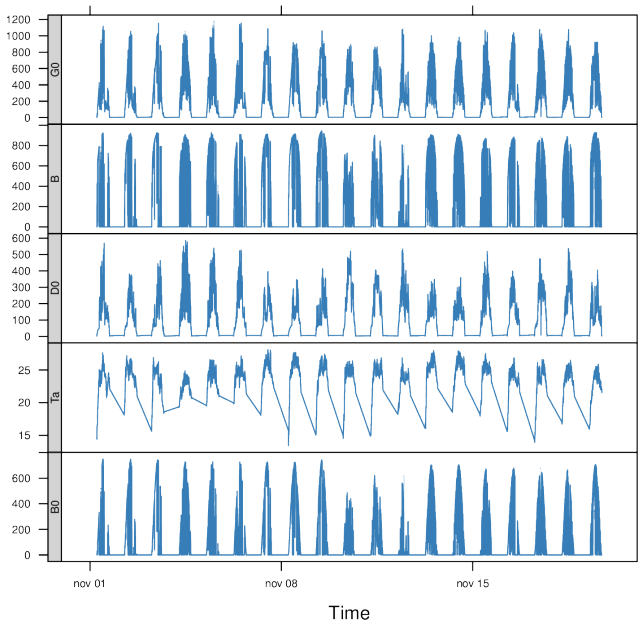

Figure 9: 1-min irradiation data from NREL-MIDC.

function `local2Solar` calculates the Mean Solar Time of the index. Last, the horizontal direct irradiance is calculated with the global and diffuse irradiances.

```
R> lat <- 20.77
R> lon <- -156.9339
R> dat <- read.zoo("Data/NREL-Hawaii.csv",
+    col.names = c("date", "hour", "G0", "B", "D0", "Ta"),
+    index = list(1, 2), FUN = function(d, h)
+      as.POSIXct(paste(d, h), format = "%m/%d/%Y %H:%M", tz = "HST"),
+    FUN2 = function(x) local2Solar(x, lon), header = TRUE, sep = ",")
R> dat$B0 <- dat$G0 - dat$D0
```

Finally, the object `Meteo` is obtained with `zoo2Meteo` (Figure 9):

```
R> NRELMeteo <- zoo2Meteo(dat, lat = lat, source = "NREL-La Ola-Lanai")
```

With this data, a `G0` object can be calculated. Since both diffuse and direct components are available, no correlation is needed (`corr = "none"`):

```
R> g0NREL <- calcG0(lat = lat, modeRad = "bdI", dataRad = NRELMeteo,
+    corr = "none")
```

If these components were not available, a correlation between hourly values of diffuse fraction and clearness index is needed (Ridley *et al.* 2010):

```
R> g0BRL <- calcG0(lat = lat, modeRad = "bdI", dataRad = NRELMeteo,
+    corr = "BRL")
```

3.4. Irradiation and irradiance on the generator plane

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 proposal of Hay and McKay combines simplicity and acceptable results (Hay and McKay 1985). 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 the dust accumulated on the surface of PV modules. 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 (Martin and Ruíz 2001).

The orientation, azimuth and incidence angle are calculated from the results of `fSolI` or `calcSol` with the functions `fTheta` and `fInclin`. These functions can estimate the geometry and irradiance for fixed systems, and two-axis and horizontal North-South trackers. Besides, the movement of a horizontal NS tracker due to the backtracking strategy (Panico, Garvison, Wenger, and Shugar 1991) can be calculated with information about the tracker and the distance between the trackers of the system.

Both functions are integrated in `calcGef`, which constructs an object of class `Gef`.

For example, with the results of Section 3.3, the irradiance and irradiation on a fixed surface can be estimated. Figure 10 shows the relation between the effective and incident irradiance versus the cosine of the angle of incidence for this system.

```
R> gef <- calcGef(lat = 37.2, modeRad = "prev", dataRad = g0, beta = 30)
R> xyplot(Gef/G ~ cosTheta | month, data = gef, type = c("p", "smooth"),
+    cex = 0.4, alpha = 0.5)
```

The next lines of code calculate the movement of a N-S horizontal axis tracker with *backtracking* (`modeShd = "bt"`) and whose inclination angle is limited to 60° (`betaLim = 60`). The evolution of the inclination angle is displayed in Figure 11. The meaning of the `distances` and `struct` arguments will be detailed in the Section 4.2.

```
R> lat <- 37.2
R> G0dm <- c(2766, 3491, 4494, 5912, 6989, 7742, 7919, 7027, 5369, 3562,
+    2814, 2179)
R> 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)
R> prom <- list(G0dm = G0dm, Ta = Ta)
R> structHoriz <- list(L = 4.83)
R> distHoriz <- data.frame(Lew = structHoriz$L * 4, H = 0)
R> gefBT <- calcGef(lat = lat, dataRad = prom, sample = "10 min",
+    modeTrk = "horiz", modeShd = "bt", betaLim = 60,
+    distances = distHoriz, struct = structHoriz)
```

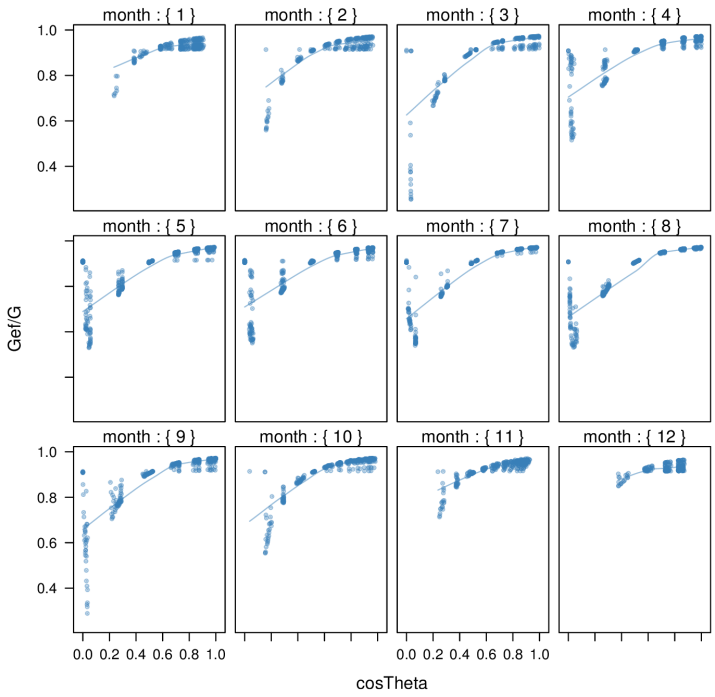

Figure 10: Relation between the effective and incident irradiance versus the cosine of the angle of incidence for a fixed system.

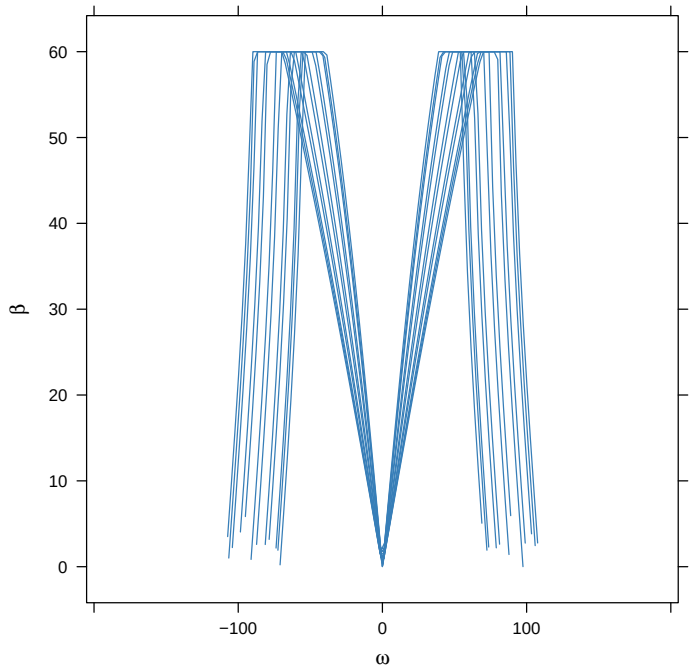

Figure 11: Evolution of the angle of inclination of a NS horizontal axis tracker with *back-tracking* and limitation of angle.

4. Productivity of a grid-connected photovoltaic system

The function `fProd` simulates the performance of a grid-connected PV (GCPV) system under certain irradiance and temperature conditions. The system is defined with a set of parameters: characteristics of the PV module (`module`) and the inverter (`inverter`), the electrical arrangement of the PV generator (`generator`) and the losses of the system (`effSys`).

For example, the electrical power (`Pac` and `Pdc`), voltage (`Voc`, `Vmpp` and `Vdc`) and current (`Isc`, `Impp`, `Idc`) of a certain PV system are calculated with `fProd`:

```
R> inclin <- data.frame(Gef = c(200, 400, 600, 800, 1000), Ta = 25)
R> fProd(inclin)

   Gef Ta    Tc   Voc   Isc  Vmpp   Impp   Vdc    Idc   Pac   Pdc   EffI
1  200 25 31.75 673.3 10.34 533.1  9.586 533.1  9.586  4212  4737 0.9164
2  400 25 38.50 655.4 20.68 516.3 19.090 516.3 19.090  8275  9137 0.9334
3  600 25 45.25 637.5 31.02 499.6 28.506 499.6 28.506 11972 13202 0.9346
4  800 25 52.00 619.7 41.36 483.0 37.824 483.0 37.824 15323 16936 0.9325
5 1000 25 58.75 601.8 51.70 466.5 47.037 466.5 47.037 18342 20342 0.9293
```

where `EffI` is the inverter efficiency.

First, `fProd` computes the maximum power point (MPP) of the generator (voltage, `Vmpp`, and current, `Impp`) at the irradiance (`Gef`) and ambient temperature (`Ta`) conditions contained in `inclin`. Next, it checks that this point is inside the MPP window voltage of the inverter, as defined by `inverter$Vmin` and `inverter$Vmax`. If the MPP value is outside this range, the function assigns the limit value to the voltage, and calculates the correspondent current value with a warning. The actual inverter input voltage and current are `Vdc` e `Idc`, respectively, and may be different from the MPP values. With the next code, the `Vdc` value is set to `Vmin` (the minimum value of the MPP window of the inverter), 420 V, since `Vmpp` is below this value.

```
R> inclin <- data.frame(Gef = 800, Ta = 30)
R> gen1 <- list(Nms = 10, Nmp = 11)
R> inv1 <- list(Ki = c(0.01, 0.025, 0.05), Pinv = 25000, Vmin = 420,
+    Vmax = 750, Gumb = 20)
R> prod <- fProd(inclin, generator = gen1, inverter = inv1)
R> print(prod)

  Gef Ta Tc   Voc   Isc  Vmpp  Impp Vdc   Idc   Pac   Pdc   EffI
1 800 30 57 505.3 41.36 392.3 37.68 420 33.83 11943 13169 0.9346
```

For this configuration, the losses due to the voltage limitation are:

```
R> 1 - with(prod, Vdc * Idc/(Vmpp * Impp))

[1] 0.039
```

The function `prodGCPV` integrates the calculation procedure of irradiation, irradiance and simulation of the GCPV system. It constructs an object of class `ProdGCPV`.

The next code computes the productivity of the previous GCPV system working as fixed, NS horizontal axis tracking and two-axis tracking systems. The parameters of the generator, module, inverter and rest of the system are those by default in `prodGCPV`.

```
R> ProdFixed <- prodGCPV(lat = lat, dataRad = prom, keep.night = FALSE)
R> Prod2x <- prodGCPV(lat = lat, dataRad = prom, modeTrk = "two",
+    keep.night = FALSE)
R> ProdHoriz <- prodGCPV(lat = lat, dataRad = prom, modeTrk = "horiz",
+    keep.night = FALSE)
```

4.1. Using `mergesolaR`

The `mergesolaR` method is designed to merge *daily* time series of several **solaR** objects.

The next example retrieves the daily irradiation of the whole set of meteorological stations of Madrid (Spain) and use this information to calculate the productivity of a grid-connected PV system with the `lapply` and `prodGCPV` functions. The result is a list of `ProdGCPV` objects. Some stations do not provide data for this time period. Therefore `prodGCPV` is evaluated inside a `try` call to remove these stations from the list[3]:

```
R> EstMadrid <- subset(SIAR, Provincia == "Madrid")
R> nEstMadrid <- nrow(EstMadrid)
R> namesMadrid <- EstMadrid$Estacion
R> prodMadrid <- lapply(1:nEstMadrid, function(x) {
+    try(prodGCPV(lat = 41, modeRad = "siar", dataRad = list(prov = 28,
+        est = x, start = "01/01/2009", end = "31/12/2010")))
+  })
R> names(prodMadrid) <- namesMadrid
R> okMadrid <- lapply(prodMadrid, class) != "try-error"
R> prodMadrid <- prodMadrid[okMadrid]
R> YfMadrid <- do.call(mergesolaR, prodMadrid)
```

`mergesolaR` with a set of `ProdGCPV` objects merges the daily time series of the `Yf` variable of each object. The result is a multivariate **zoo** object where each column is the daily productivity with the radiation data of each meteorological station. For example, the `horizonplot` function (Figure 12) shows the anomalous behaviour of the Aranjuez station during June 2009 and the second half of 2010. This result will be revisited with the `TargetDiagram` tool (Figure 21).

```
R> horizonplot(YfMadrid - rowMeans(YfMadrid), origin = 0,
+    scales = list(y = list(relation = "same")), colorkey = TRUE)
```

4.2. Shadows

The shadows on PV generators alter the performance of the PV generators and reduce their productivity (Perpiñán 2008). This package includes functions for the estimation of mutual shadows between generators belonging to the same system. `fSombra2X`, `fSombraHoriz`,

[3]The local copy of this code is `prodMadrid.RData`.

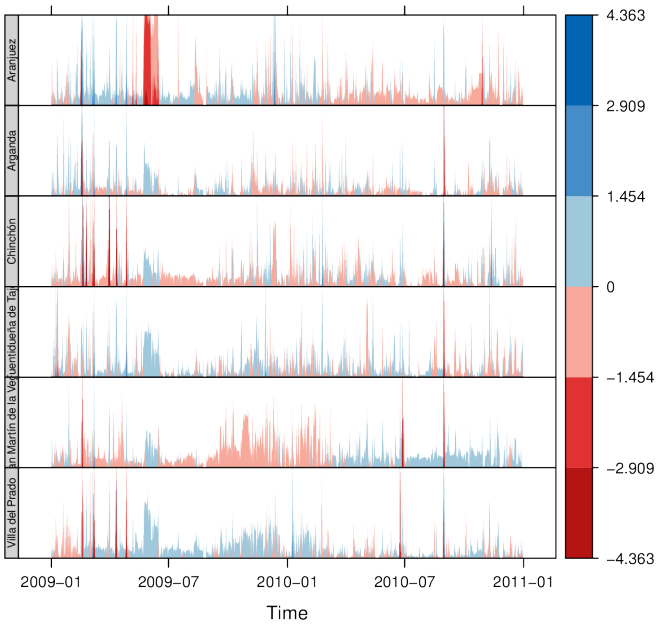

Figure 12: `horizonplot` of the result of a `mergesolaR` call. Previously, the row mean is substracted from each column in order to show the deviation of each meteorological station from the daily mean of the set.

`fSombraEst`, calculate the shadows in two-axis, horizontal axis and fixed systems, respectively. The function `fSombra6` is indicated for groups of two-axis trackers. Finally, `fSombra` is a wrapper to the previous functions. These functions are integrated in `calcShd`, `calcGef` and `prodGCPV`, as these examples show.

First, the dimensions of the support structures (`struct`) and the distances between them (`distances`) have to be defined. With a two-axis tracking system:

```
R> struct2x <- list(W = 23.11, L = 9.8, Nrow = 2, Ncol = 8)
R> dist2x <- data.frame(Lew = 40, Lns = 30, H = 0)
R> prod2xShd <- prodGCPV(lat = lat, dataRad = prom, modeTrk = "two",
+    modeShd = "area", struct = struct2x, distances = dist2x)
```

Then, a N-S horizontal axis tracking system without backtracking,

```
R> structHoriz <- list(L = 4.83)
R> distHoriz <- data.frame(Lew = structHoriz$L * 4, H = 0)
R> prodHorizShd <- prodGCPV(lat = lat, dataRad = prom, sample = "10 min",
+    modeTrk = "horiz", modeShd = "area", betaLim = 60,
+    distances = distHoriz, struct = structHoriz)
```

and a N-S horizontal axis tracking system with backtracking,

```
R> prodHorizBT <- prodGCPV(lat = lat, dataRad = prom, sample = "10 min",
+    modeTrk = "horiz", modeShd = "bt", betaLim = 60,
+    distances = distHoriz, struct = structHoriz)
```

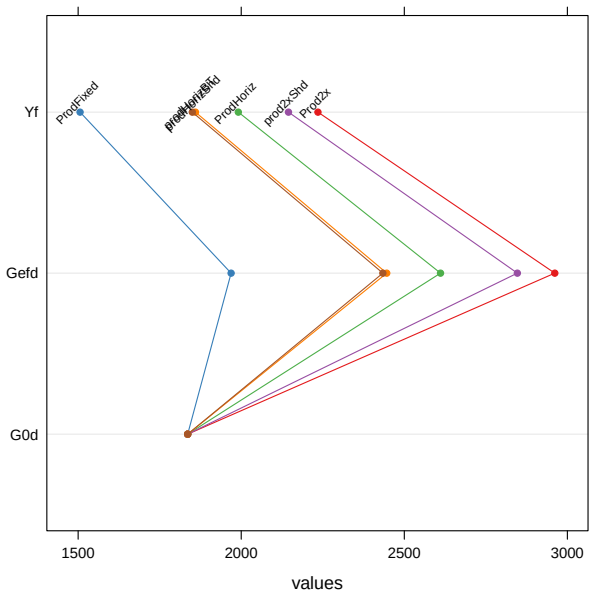

Figure 13: Comparison of several `ProdGCPV` objects.

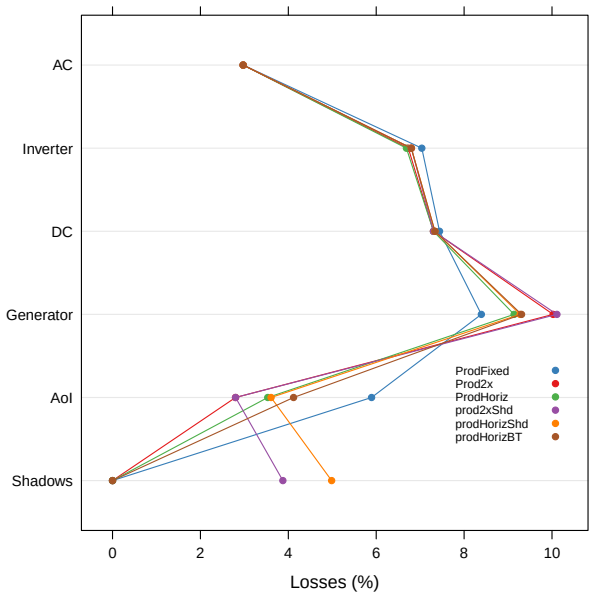

Figure 14: Comparison of the losses of several `ProdGCPV` objects.

Finally, the *yearly* performance of these systems is compared with the method `compare` (Figure 13):

```
R> comp <- compare(ProdFixed, Prod2x, ProdHoriz, prod2xShd, prodHorizShd,
+     prodHorizBT)
```

```
R> head(comp)
```

```
  values  ind      name
1   1836  G0d ProdFixed
2   1969 Gefd ProdFixed
3   1506   Yf ProdFixed
4   1836  G0d    Prod2x
5   2961 Gefd    Prod2x
6   2235   Yf    Prod2x
```

The methods `losses` and `compareLosses` calculate and compare their *yearly* losses, respectively (Figure 14):

```
R> compL <- compareLosses(ProdFixed, Prod2x, ProdHoriz, prod2xShd,
+    prodHorizShd, prodHorizBT)
R> head(compL)
```

```
         id  values      name
1   Shadows 0.00000 ProdFixed
2       AoI 0.05894 ProdFixed
3 Generator 0.08392 ProdFixed
4        DC 0.07441 ProdFixed
5  Inverter 0.07038 ProdFixed
6        AC 0.02973 ProdFixed
```

4.3. Position of trackers in a PV plant

One of the tasks of the design of a PV tracking system is to place the set of trackers. This task must cope with the compromise of minimizing the losses due to mutual shadows while requiring the minimum land area.

The area of the PV generator and the total land requirement are commonly related with the ground coverage ratio (GCR). This ratio quantifies the percentage of land being effectively occupied by the system. In order to focus on the land area required, the inverse of this ratio, the ground requirement ratio (GRR), is preferable. The GRR is the ratio between the ground area required for installing the whole set of trackers and the generator area.

A suitable approach to the problem is to simulate the planned system for a set of distances between the trackers of the plant. Without any additional constraint, the optimum design may be the one which achieves the highest productivity with the lowest ground requirement ratio.

However, it should be noted that this approach to the problem is not complete since the land requirements and the costs of wiring and equipments should be included as additional constraints (Perpiñán 2012a).

The function `optimShd` computes the productivity for a set of combinations of distances between the elements of the plant (Perpiñán 2008). The designer should adopt the decision from these results with the adequate economical translations.

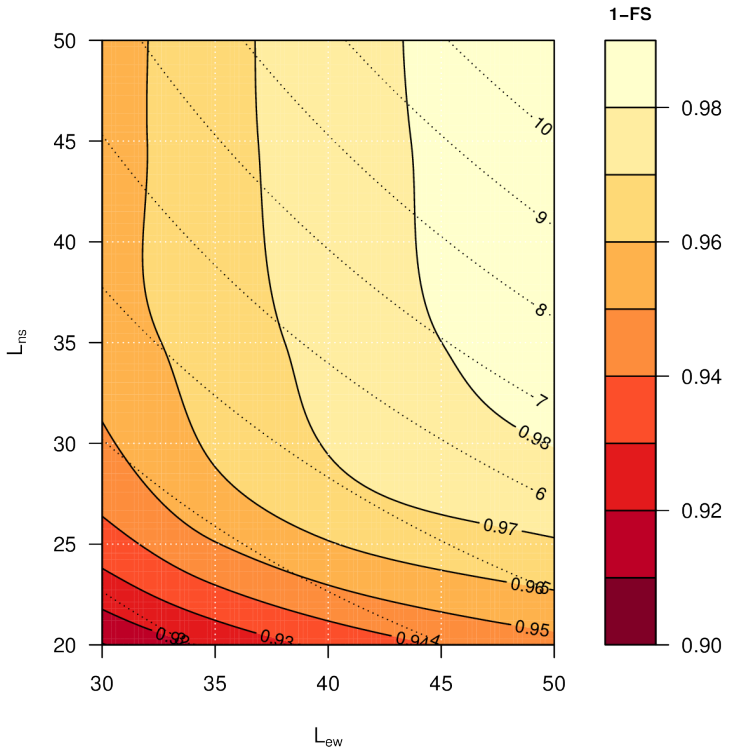

Figure 15: Relative productivity including mutual shadows in a two-axis tracking PV system over a grid of East-West and North-South distances. The GRR values are superposed with dotted lines.

For example, let us design a PV plant with a grid of trackers of 2 rows and 8 columns using a two-axis tracker whose dimensions are 23.11 m width and 9.8 m height.

```
R> struct2x <- list(W = 23.11, L = 9.8, Nrow = 2, Ncol = 8)
```

The separations between trackers range from 30 m and 50 m for the East-West direction and from 20 m and 50 m for the North-South direction.

```
R> dist2x <- list(Lew = c(30, 50), Lns = c(20, 50))
```

`optimShd` constructs a sequence from the minimum to the maximum value of `distances`, with `res` as the increment, in meters, of the sequence. In this example, `res = 5`.

```
R> ShdM2x <- optimShd(lat = lat, dataRad = prom, modeTrk = "two",
+    modeShd = c("area", "prom"), distances = dist2x,
+    struct = struct2x, res = 5, prog = FALSE)
R> shadeplot(ShdM2x)
```

Besides, the `Shade` object includes the local fitting of the sequence of `Yf` and `FS` values (slots named `Yf.loess` and `FS.loess`). The `predict` method is used with these `loess` slots inside the `shadeplot` method of the `Shade` class (Figure 15).

5. PV pumping systems

5.1. Simulation of centrifugal pumps

The first step for the simulation of the performance of a PV pumping system (PVPS) is the characterization of the pump under the supposition of constant manometric height (Abella, Lorenzo, and Chenlo 2003). The function `fPump` computes the performance of the different parts of a centrifugal pump fed by a frequency converter following the affinity laws.

For example, the performance of the SP8A44 pump (see http://net.grundfos.com/Appl/WebCAPS/InitCtrl?mode=1), whose information is available in the dataset `pumpCoef`, working with $H = 40\,\text{m}$ is simulated with:

```
R> data("pumpCoef")
R> CoefSP8A44 <- subset(pumpCoef, Qn == 8 & stages == 44)
R> fSP8A44 <- fPump(pump = CoefSP8A44, H = 40)
```

The result of `fPump` is a set of functions which relate the electrical power and the flow, hydraulical and mechanical power, and frequency. These functions allow the calculation of the performance for any electrical power inside the range of the pump (Figure 16):

```
R> SP8A44 <- with(fSP8A44, {
+    Pac <- seq(lim[1], lim[2], by = 100)
+    Pb <- fPb(Pac)
+    etam <- Pb/Pac
+    Ph <- fPh(Pac)
+    etab <- Ph/Pb
+    f <- fFreq(Pac)
+    Q <- fQ(Pac)
+    result <- data.frame(Q, Pac, Pb, Ph, etam, etab, f)
+  })
R> SP8A44$etamb <- with(SP8A44, etab * etam)
```

The performance of a PVPS follows the same procedure as the one described for the GCPV systems. The function `prodPVPS` is the equivalent to the function `prodGCPV`. Both functions share similar arguments with some variations due to the differences between a PVPS and a GCPV system. Besides, `prodPVPS` does not allow for the calculation of shadows.

5.2. Nomograms of PVPS

The international standard IEC 61725 is of common usage in public licitations of PVPS. This standard proposes a equation of the irradiance profile with several parameters such as the length of the day, the daily irradiation and the maximum value of the irradiance. With this profile, the performance of a PVPS can be calculated for several manometric heights and nominal PV power values. A nomogram can display the set of combinations. This graphical tool can help to choose the best combination of pump and PV generator for certain conditions of irradiation and height (Abella *et al.* 2003). This kind of graphics is provided by the function `NmgPVPS`. Figure 17 is a nomogram for the SP8A44 pump working in a range of heights from

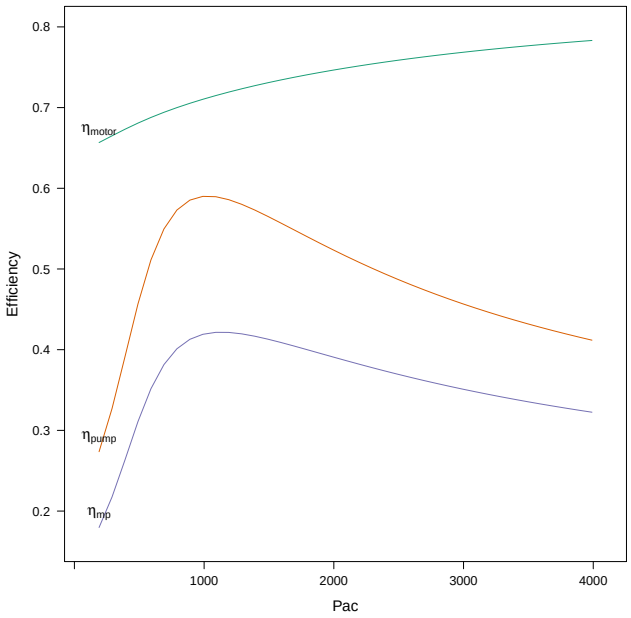

Figure 16: Efficiency of the motor and pump for several values of electrical power of a SP8A44 pump with $H = 40\,\text{m}$.

50 to 80 meters, with different PV generators. The peculiar shape of the curve of 50 meters shows that this pump does not work correctly with this height.

```
R> Pg <- seq(3000, 5500, by = 500)
R> H <- seq(50, 80, by = 5)
R> NmgSP8A44 <- NmgPVPS(pump = CoefSP8A44, Pg = Pg, H = H, Gd = 6000,
+    title = "Selection of Pumps", theme = custom.theme())
```

6. solaR with spatial packages

solaR is designed for time series associated to a location defined by the latitude and longitude values, and the temperature and irradiation conditions. However, **solaR** can also be easily combined with spatial packages: for example with **raster** (Hijmans and van Etten 2011) in Ummel (2011) or with geostatistics methods in Antoñanzas, Cañizares, and Perpiñán (2012).

6.1. solaR and raster

As an example of the interaction of **raster** and **solaR**, several files with monthly averages of global solar radiation over the Iberian Peninsula are read with **raster** and transformed with **solaR**. This information is provided by the Satellite Application Facility on Climate Monitoring (CM-SAF) (Posselt, Mueller, Stöckli, and Trentmann 2012; CM-SAF 2012). CM-SAF generates, archives and distributes widely recognized high-quality satellite-derived products and services relevant for climate monitoring in operational mode. The data is freely accesible after a registration process[4]. It is interesting to note that part of the internal calculations

[4]The data for this example is available as supplementary material.

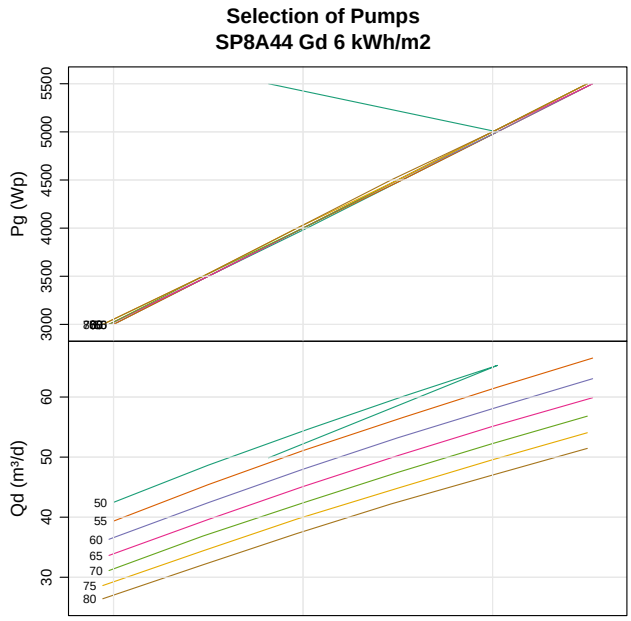

Figure 17: Nomogram for the SP8A44 pump working in a range of heights from 50 to 80 meters, with different PV generators.

of CM-SAF uses the **libRadtran** model (Trentmann, Träger-Chatterjee, and Müller 2010, see Section 1.1).

```
R> library("raster")
R> unzip("Data/SISmm2008_CMSAF.zip", exdir = "Data/CMSAF")
R> old <- setwd("Data/CMSAF")
R> listFich <- dir(pattern = "2008")
R> stackSIS <- stack(listFich)
R> stackSIS <- stackSIS * 24
R> setwd(old)
R> idx <- seq(as.Date("2008-01-15"), as.Date("2008-12-15"), "month")
R> SISmm <- setZ(stackSIS, idx)
R> projection(SISmm) <- proj
R> layerNames(SISmm) <- month.abb
R> latLayer <- init(SISmm, v = "y")
```

The yearly effective irradiance on an inclined plane can be calculated with `calcGef` (Section 3.4). The next function uses `calcGef` to provide yearly values (`as.data.frameY`) of effective global, diffuse and direct irradiation:

```
R> gefFoo <- function(g0) {
+    gef <- calcGef(lat = g0[1], dataRad = list(G0dm = g0[2:13]))
+    result <- as.data.frameY(gef)[c("Gefd", "Befd", "Defd")]
+    as.numeric(result)
+  }
```

The function `calc` from **raster** applies this function to each cell of the `raster`. Be warned: the next code will take a long time without additional modifications.

```
R> gefCMSAF <- calc(stack(latLayer, SISmm), gefFoo, filename = "gefCMSAF",
+    overwrite = TRUE)
R> layerNames(gefCMSAF) = c("Gefd", "Befd", "Defd")
R> gefCMSAF
```

Starting with release 2.14.0 R includes the package **parallel**. Previous code can be accelerated running parallel computations on machines with multiple cores, for example using `mclapply`.

The function `blockSize` from the **raster** computes the chunk size and corresponding row numbers to process the `brick` object chunk by chunk. The result is a list with `rows`, the row numbers at which to start the blocks for reading and writing, `size`, the block size (number of rows) and `n`, the total number of blocks.

```
R> library("parallel")
R> cores <- detectCores()
R> bs <- blockSize(SISmm, minblocks = 8 * cores)
```

`mclapply` is a parallelized version of `lapply`. The input of `mclapply` (`seq_len(bs$n)`) is split into as many parts as `mc.cores` and spread across the cores sequentially. Then one process is forked to each core and the results are collected.

`mclapply` returns a list of length `bs$n` (the total number of blocks), where each element is the result of applying a function to the corresponding element of `seq_len(bs$n)`. This function gets the data from a block of `SISmm` and `latLayer`, builds the matrix `vals` and applies the function `gefFoo` (previously defined) to each cell. The next code takes approximately 1 hour in a computer with four cores.

```
R> resCl <- mclapply(seq_len(bs$n), function(i) {
+    vals <- getValues(SISmm, bs$row[i], bs$nrows[i])
+    lat <- getValues(latLayer, bs$row[i], bs$nrows[i])
+    vals <- cbind(lat, vals)
+    res0 <- apply(vals, MARGIN = 1L, FUN = gefFoo)
+    res0
+  }, mc.cores = cores)
R> resCl <- t(do.call(cbind, resCl))
```

The result is assigned to an empty new `brick` with the same extent and projection as `SISmm`, but with three layers.

```
R> gefCMSAF <- brick(SISmm, nl = 3)
R> layerNames(gefCMSAF) <- c("Gefd", "Befd", "Defd")
R> gefCMSAF <- setValues(gefCMSAF, resCl)
R> gefCMSAF <- writeRaster(gefCMSAF, filename = "gefCMSAF")
R> gefCMSAF
```

Figure 18 displays the results for the global effective irradiation using the `levelplot` method included in the **rasterVis** package (Perpiñán and Hijmans 2012). The administrative borders

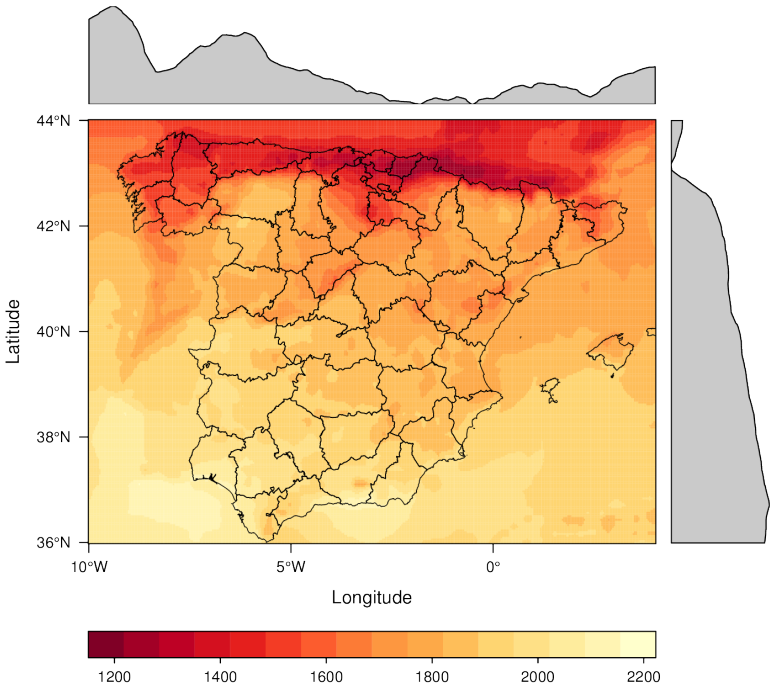

Figure 18: Global effective irradiation in Spain calculated from the CMSAF irradiation data.

from the GADM database (Hijmans 2012) are overlaid with the `layer` mechanism of the **latticeExtra** package:

```
R> library("maptools")
R> unzip("Data/ESP_adm.zip", exdir = "Data/ESP_adm")
R> mapaSHP <- readShapeLines("Data/ESP_adm/ESP_adm2.shp", proj4string = proj)
R> setwd(old)
R> library("rasterVis")
R> levelplot(gefCMSAF, layers = "Gefd") + layer(sp.lines(mapaSHP, lwd = 0.7))
```

6.2. solaR and sp

As an example of the interaction of **sp** (Bivand, Pebesma, and Gomez-Rubio 2008; Pebesma and Bivand 2005) and **solaR**, let us draw a map of the extraterrestrial irradiance. First, the mean solar time for a range of longitudes with `local2Solar` is calculated with:

```
R> hh <- as.POSIXct("2011-05-01 11:00:00", tz = "CET")
R> latitude <- seq(70, -70, -1)
R> longitude <- seq(-179.5, 179.5, 1)
R> horaLong <- local2Solar(hh, longitude)
```

Then, the irradiance for the window defined by `latitude` and `longitude` is calculated with `calcSol`. The zero value is assigned to the `NA` elements in order to get them black coloured in the map.

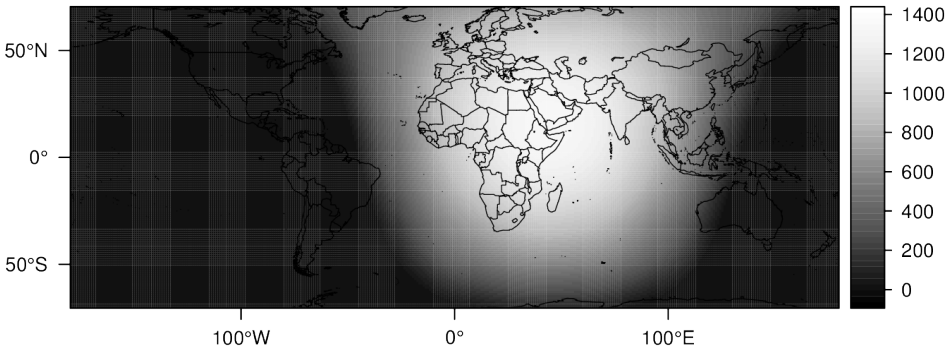

Figure 19: Extraterrestrial irradiance map.

```
R> solList <- lapply(latitude, calcSol, BTi = horaLong)
R> Bo0List <- lapply(solList, function(x) as.data.frameI(x)$Bo0)
R> Bo0 <- do.call("c", Bo0List)
R> Bo0[is.na(Bo0)] <- 0
```

The `data.frame` is now converted to an `SpatialPixelsDataFrame`. The result is displayed in Figure 19.

```
R> Bo0DF <- expand.grid(lon = longitude, lat = latitude)
R> Bo0DF$Bo0 <- c(Bo0)
R> proj <- CRS("+proj=latlon +ellps=WGS84")
R> Bo0SP <- SpatialPixelsDataFrame(points = Bo0DF[, 1:2],
+    data = Bo0DF["Bo0"], proj4string = proj)
R> paleta <- colorRampPalette(rev(brewer.pal("Greys", n = 9)))
R> p <- spplot(Bo0SP, scales = list(draw = TRUE), col.regions = paleta,
+    cuts = 50)
R> library("maps")
R> world <- map("world", plot = FALSE)
R> world_sp <- map2SpatialLines(world, proj4string = proj)
R> p + layer(sp.lines(world_sp, lwd = 0.5))
```

7. Target diagram

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

If these common deviations are assumed as a random process, a statistical analysis of the performance of the whole set of systems can identify a faulty system as the one that departs significantly from the mean behaviour (Perpiñán 2009).

The functions `analyzeData` and `TargetDiagram` compare the daily performance of each system with a reference (for example, the median of the whole set) during a time period of N

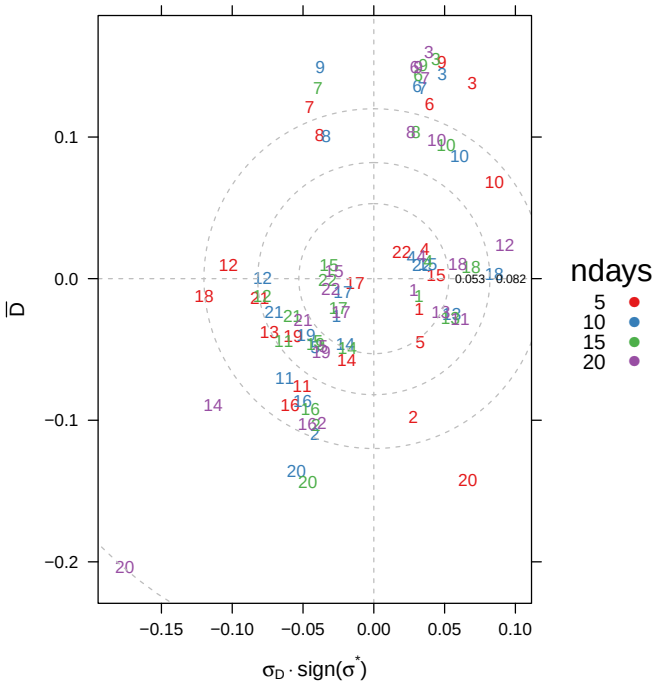

Figure 20: Target diagram of the statistical analysis of a set of 22 systems during various time periods.

days preceding the current day. They calculate a set of statistics of the performance of the PV plant as a whole, and another set of the comparison with the reference.

This statistical analysis can be summarized with a graphical tool named "target diagram", which plots together the root mean square difference, the average difference and the standard deviation of the difference. Besides, this diagram displays the sign of the difference of the standard deviations of the system and the reference (Jolliff *et al.* 2009; Taylor 2000).

The example from Figure 20 uses a dataset of productivity from a PV plant composed of 22 systems (`data("prodEx")`) showing that the system no. 20 is not working correctly during these periods.

```
R> data("prodEx")
R> ndays <- c(5, 10, 15, 20)
R> palette <- brewer.pal(n = length(ndays), name = "Set1")
R> TDColor <- TargetDiagram(prodEx, end = day, ndays = ndays,
+    color = palette)
```

Figure 12 displayed the result of an example with `mergesolaR` and the SIAR network. The function `TargetDiagram` is an alternative tool to show the behaviour of the set of meteorological stations (Figure 21). Once again, the behaviour of the Aranjuez station is consistently different from the rest of the stations of Madrid during these time intervals.

```
R> TDMadrid <- TargetDiagram(YfMadrid, end = as.POSIXct("2010-12-31"),
+    ndays = c(10, 20, 30, 40, 50, 60), cex = 0.7)
```

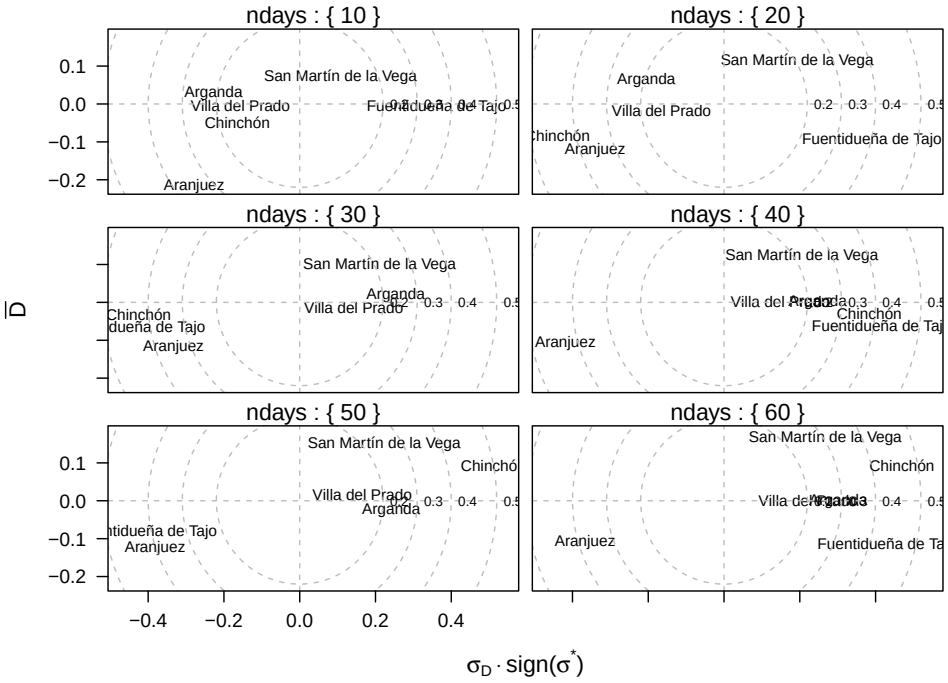

Figure 21: Target diagram of the result of the `mergesolaR` example.

8. Conclusion

The **solaR** package includes a set of classes, methods and functions to calculate the sun geometry, 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.

It 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.

The classes share a variety of methods to access the information (for example, `as.zooD` provides a **zoo** object with the daily multivariate time series of the corresponding object) and several visualization methods based on the **lattice** (Sarkar 2008) and **latticeExtra** (Sarkar and Andrews 2011) packages. Besides, **solaR** 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.

Session information

The results discussed in this paper were obtained in a session with these characteristics:

- R version 2.15.0 (2012-03-30), `i486-pc-linux-gnu`
- Locale: `LC_CTYPE=es_ES.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=es_ES.UTF-8`, `LC_COLLATE=es_ES.UTF-8`, `LC_MONETARY=es_ES.UTF-8`, `LC_MESSAGES=es_ES.UTF-8`, `LC_PAPER=C`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=es_ES.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: **base**, **datasets**, **graphics**, **grDevices**, **grid**, **methods**, **parallel**, **stats**, **tools**, **utils**
- Other packages: **hexbin** 1.26.0, **lattice** 0.20-6, **latticeExtra** 0.6-19, **raster** 1.9-82, **rasterVis** 0.10-9, **RColorBrewer** 1.0-5, **solaR** 0.33, **sp** 0.9-98, **zoo** 1.7-7

References

Abella MA, Lorenzo E, Chenlo F (2003). "PV Water Pumping Systems Based on Standard Frequency Converters." *Progress in Photovoltaics: Research and Applications*, **11**(3), 179–191.

Antoñanzas F, Cañizares F, Perpiñán O (2012). "Comparative Assessment of Global Irradiation from a Satellite Estimate Model (CM SAF) and On-Ground Measurements (SIAR): A Spanish Case Study." *Submitted manuscript*. URL http://procomun.wordpress.com/documentos/articulos/.

Bivand RS, Pebesma EJ, Gomez-Rubio V (2008). *Applied Spatial Data Analysis with R*. Springer-Verlag, New York. URL http://www.asdar-book.org/.

Brenning A (2011). ***RSAGA: SAGA** Geoprocessing and Terrain Analysis in R*. R package version 0.92-2, URL http://CRAN.R-project.org/package=RSAGA.

Cimmery V (2010). "**SAGA** User Guide, Version 2.0.5." URL http://www.saga-gis.org/.

CM-SAF (2012). "The Satellite Application Facility on Climate Monitoring." URL http://www.cmsaf.eu/.

Collares-Pereira M, Rabl A (1979). "The Average Distribution of Solar Radiation: Correlations between Diffuse and Hemispherical and between Daily and Hourly Insolation Values." *Solar Energy*, **22**, 155–164.

Cooper PI (1969). "The Absorption of Solar Radiation in Solar Stills." *Solar Energy*, **12**(3), 333–346.

de Miguel A, Bilbao J, Aguiar RJ, Kambezidis H, Negro E (2001). "Diffuse Solar Irradiation Model Evaluation in the North Mediterranean Belt Area." *Solar Energy*, **70**(2), 143–153.

Erbs DG, Klein SA, Duffie JA (1982). "Estimation of the Diffuse Radiation Fraction for Hourly, Daily and Monthly-Average Global Radiation." *Solar Energy*, **28**(4), 293–302.

GRASS Development Team (2012). *Geographic Resources Analysis Support System (GRASS GIS) Software*. Open Source Geospatial Foundation, USA. URL http://grass.osgeo.org/.

Gueymard C (2008). "REST2: High-Performance Solar Radiation Model for Cloudless-Sky Irradiance, Illuminance, and Photosynthetically Active Radiation – Validation with a Benchmark Dataset." *Solar Energy*, **82**(3), 272–285.

Gueymard CA (2001). "Parameterized Transmittance Model for Direct Beam and Circumsolar Spectral Irradiance." *Solar Energy*, **71**(5), 325–346.

Harrower MA, Brewer CA (2003). "**ColorBrewer.org**: An Online Tool for Selecting Color Schemes for Maps." *The Cartographic Journal*, **40**, 27–37. URL http://ColorBrewer.org/.

Hay JE, McKay DC (1985). "Estimating Solar Irradiance on Inclined Surfaces: A Review and Assessment of Methodologies." *International Journal of Solar Energy*, (3), 203.

Hijmans RJ (2012). "GADM Database of Global Administrative Areas." URL http://www.gadm.org/.

Hijmans RJ, van Etten J (2011). ***raster**: Geographic Analysis and Modeling with Raster Data*. R package version 1.8-39, URL http://CRAN.R-project.org/package=raster.

Hofierka J, Súri M (2002). "The Solar Radiation Model for Open Source GIS: Implementation and Applications." In *Open Source GIS-GRASS Users Conference*. URL http://www.ing.unitn.it/~grass/conferences/GRASS2002/proceedings/proceedings/pdfs/Hofierka_Jaroslav.pdf.

Huld TA, Súri M, Dunlop ED, Micale F (2006). "Estimating Average Daytime and Daily Temperature Profiles within Europe." *Environmental Modelling & Software*, **21**(12), 1650–1661.

Instituto de Energía Solar (2012). "UPM Meteo Station Server." Ciudad Universitaria, Madrid, Spain, URL http://helios.ies-def.upm.es/.

Joint Research Centre, European Commission (2012). "Photovoltaic Geographical Information System (PVGIS)." URL http://re.jrc.ec.europa.eu/pvgis/.

Jolliff J, Kindle JC, Shulman I, Penta B, Friedrichs MAM, Helber R, Arnone, A R (2009). "Summary Diagrams for Coupled Hydrodynamic-Ecosystem Model Skill Assessment." *Journal of Marine Systems*, **76**, 64–82.

Lewin-Koh NJ, Bivand R, Pebesma EJ, Archer E, Baddeley A, Bibiko HJ, Dray S, Forrest D, Friendly M, Giraudoux P, Golicher D, Gomez-Rubio V, Hausmann P, Hufthammer KO, Jagger T, Luque SP, MacQueen D, Niccolai A, Short T, Stabler B, Turner R (2011). ***maptools**: Tools for Reading and Handling Spatial Objects*. R package version 0.8-10, URL http://CRAN.R-project.org/package=maptools.

Liu BYH, Jordan RC (1960). "The Interrelationship and Characteristic Distribution of Direct, Diffuse, and Total Solar Radiation." *Solar Energy*, **4**, 1–19.

Martin N, Ruíz JM (2001). "Calculation of the PV Modules Angular Losses under Field Conditions by Means of an Analytical Model." *Solar Energy Materials & Solar Cells*, **70**, 25–38.

Mayer B, Kylling A (2005). "Technical Note: The **libRadtran** Software Package for Radiative Transfer Calculations – Description and Examples of Use." *Atmospheric Chemistry and Physics*, **5**(7), 1855–1877.

Michalsky JJ (1988). "The Astronomical Almanac's Algorithm for Approximate Solar Position (1950–2050)." *Solar Energy*, **40**(3), 227–235. URL ftp://climate1.gsfc.nasa.gov/wiscombe/Solar_Rad/SunAngles/sunae.f.

Ministerio de Agricultura, Alimentación y Medio Ambiente (2012). "Sistema de Información Agroclimática del Regadío." URL http:///www.marm.es/siar/Informacion.asp.

National Renewable Energy Laboratory (2012). "Measurement and Instrumentation Data Center (MIDC)." US Department of Energy, Office of Energy Efficiency and Renewable Energy, URL http://www.nrel.gov/midc/.

Neuwirth E (2011). ***RColorBrewer**: Color Brewer Palettes*. R package version 1.0-5, URL http://CRAN.R-project.org/package=RColorBrewer.

Page JK (1961). "The Calculation of Monthly Mean Solar Radiation for Horizontal and Inclined Surfaces from Sunshine Records for Latitudes 40N-40S." In *U.N. Conference on New Sources of Energy*, volume 4, pp. 378–390.

Panico D, Garvison P, Wenger HJ, Shugar D (1991). "Backtracking: A Novel Strategy for Tracking PV Systems." In *IEEE Photovoltaic Specialists Conference*, pp. 668–673.

Pebesma EJ, Bivand RS (2005). "Classes and Methods for Spatial Data in R." *R News*, **5**(2), 9–13. URL http://CRAN.R-project.org/doc/Rnews/.

Perpiñán O (2008). *Grandes Centrales Fotovoltaicas: Producción, Seguimiento y Ciclo de Vida*. Ph.D. thesis, UNED. URL http://e-spacio.uned.es/fez/view.php?pid=tesisuned:IngInd-Operpinan.

Perpiñán O (2009). "Statistical Analysis of the Performance and Simulation of a Two-Axis Tracking PV System." *Solar Energy*, **83**(11), 2074–2085.

Perpiñán O (2012a). "Cost of Energy and Mutual Shadows in a Two-Axis Tracking PV System." *Renewable Energy*, **43**, 331–342. URL http://procomun.wordpress.com/documentos/articulos/.

Perpiñán O (2012b). "Energía Solar Fotovoltaica." URL http://procomun.wordpress.com/documentos/libroesf/.

Perpiñán O, Hijmans RJ (2012). ***rasterVis**: Visualization Methods for the **raster** Package*. R package version 0.10-9, URL http://CRAN.R-project.org/package=rasterVis.

Posselt R, Mueller RW, Stöckli R, Trentmann J (2012). "Remote Sensing of Solar Surface Radiation for Climate Monitoring – The CM SAF Retrieval in International Comparison." *Remote Sensing of Environment*, **118**, 186–198.

PVsyst SA (2012). "**PVsyst**: Software for Photovoltaic Systems." URL http://www.PVsyst.com/.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Reda I, Andreas A (2004). "Solar Position Algorithm for Solar Radiation Applications." *Solar Energy*, **76**(5), 577–589.

Reda I, Andreas A (2008). "Solar Position Algorithm for Solar Radiation Applications." *Technical report*, National Renewable Energy Laboratory. URL http://www.nrel.gov/docs/fy08osti/34302.pdf.

Ridley B, Boland J, Lauret P (2010). "Modelling of Diffuse Solar Fraction with Multiple Predictors." *Renewable Energy*, **35**, 478–482.

Sarkar D (2008). ***lattice**: Multivariate Data Visualization with R*. Springer-Verlag, New York. URL http://lmdvr.R-Forge.R-project.org/.

Sarkar D, Andrews F (2011). ***latticeExtra**: Extra Graphical Utilities Based on **lattice***. R package version 0.6-19, URL http://CRAN.R-project.org/package=latticeExtra.

Spencer JW (1971). "Fourier Series Representation of the Position of the Sun." *Search*, **2**(5), 172. Reprinted at http://www.mail-archive.com/sundial@uni-koeln.de/msg01050.html.

Strous L (2011). "Position of the Sun." URL http://www.astro.uu.nl/~strous/AA/en/reken/zonpositie.html.

Taylor KE (2000). "Summarizing Multiple Aspects of Model Performance in a Single Diagram." *Technical report*, Program for Climate Model Diagnosis and Intercomparison. URL http://www-pcmdi.llnl.gov/publications/pdf/55.pdf.

Trentmann J, Träger-Chatterjee C, Müller R (2010). *Surface Radiation Products*. EUMETSAT Satellite Application Facility on Climate Monitoring, 2.1 edition. URL http://www.cmsaf.eu/bvbw/generator/CMSAF/Content/Publication/SAF__CM__DWD__PUM__SFCRAD__2.1,templateId=raw,property=publicationFile.pdf/SAF_CM_DWD_PUM_SFCRAD_2.pdf.

Ummel K (2011). *SEXPOT: A Spatiotemporal Linear Programming Model to Simulate Global Deployment of Renewable Power Technologies*. Master's thesis, Central European University, Budapest. URL http://dl.dropbox.com/u/14314000/ThesisMedia/Kevin_Ummel_CEU_2011.pdf.

Zeileis A, Grothendieck G (2005). "**zoo**: S3 Infrastructure for Regular and Irregular Time Series." *Journal of Statistical Software*, **14**(6), 1–27. URL http://www.jstatsoft.org/v14/i06/.

Affiliation:

Oscar Perpiñán Lamigueiro
Departamento de Ingeniería Eléctrica
Universidad Politécnica de Madrid
Madrid, Spain
E-mail: oscar.perpinan@gmail.com
URL: http://procomun.wordpress.com/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 50, Issue 9 *Submitted:* 2011-12-21
August 2012 *Accepted:* 2012-06-18