# PV - Wirkungsgrad

## Contents



Geladen werden die Daten, die von der PV-Anlage mit Hilfe des SMA-Portals gewonnen werden. Ziel ist es, in der Tabelle *data* alle Datensätze aus den SMA-Daten, ergänzt um Hilfsgrößen, zur Verfügung zu stellen.

Die Daten liegen in Dateien tageweise vor, beim Download werden diese von Hand benannt, sie enthalten Datensätze (Zeilen), die im 5-Minuten-Rhythmus erfasst wurden. Beim Einlesen werden sie zusammengefügt.

Die Datensätze enthalten die Größen

leistung.pv — leistung.stp — netzeinspeisung — netzbezug — batt_ladung — batt_entladung — ladezustand

Die beiden ersten Werte sind identisch, deswegen wird “leistung.stp”" in der Folge sofort gelöscht.

# 1 Vorbereitungen

### 1.0.1 Laden der nötigen Bibliotheken.

```
source("01-Bibliotheken-laden.R")
```

### 1.0.2 Zur Auswertung werden einige Funktionen benötigt, die hier definiert werden.

```
source("02-Funktionen-bilden.R")
```

### 1.0.3 Einlesen der Dateien “Daten_dd_mm_yyyy.csv”:

Alle Größen in der Einheit W, mit Ausnahme von ‘ladezustand’, dieser wird beim Lesen als Prozentsatz übergeben und anschließend auf 10000 = 100% normiert weil die Batterie eine Kapazität von annähernd $10kWh$ besitzt kann dies auch als $Wh$ gelesen werden. Die Zeilen müssen sortiert werden, weil die Dateien nicht in der korrekten zeitlichen Reihenfolge eingelesen werden.

```
# Einlesen der Datenfiles-----notig: 02-Funktionen-bilden.R----------------------
source("03-Files-einlesen.R")
```

```
## gelesen:
##  Daten_01_10_2017.csv Daten_01_11_2017.csv Daten_02_10_2017.csv Daten_02_11_2017.csv Daten_03_10_201
```

#### 1.0.4 Einige Spalten werden erzeugt, gelöscht, bearbeitet und z.B. neu normiert:

a) Die neue Spalte 'ct' zaehlt die Datenzeilen
b) Über 'ladezustand' laeuft eine Glaettungsfunktion, um einzelne Ausfaelle in den Messungen zu beseitigen.
c) 'day' und 'hour' werden aus der Variablen 'zeit' extrahiert und im Datumsformat "yyyy-mm-dd" bzw. als Zahl 0 - 23 gespeichert.
d) 'ladediff' wird als Differenz von Ladezustand zwischen dem aktuellen Zustand und dem vorangegangenen berechnet (Einheit Wh).
e) 'batt_ladung' und 'bat_entladung' werden von W in Wh umgerechnet (W in der Zeit 5 min, deswegen Division durch 12). Anm.: In der späteren Auswertung wird dies so interpretiert: Eine zur Zeit t erbrachte Leistung P führt zu einer el. Arbeit von $P \cdot 5\text{min}$ im Zeitintervall $t \pm 2,5\text{min}$

```
# Ergaenzende Spaltenoperationen ------------------------------------------------
source("03-Spalten-bearbeiten.R")
```

```
## Loesche Spalten  leistung.pv netzeinspeisung netzbezug .
## Der Datensatz enthaelt jetzt  14681 Zeilen.
```

# 2 Auswertungen

## 2.1 Tägliche Minima und Maxima identifizieren - optional

```
# Minima und MAxima markieren frueher R_Min_Max_mark.R
 source("04_Auswertungen_Min_Max_tgl_Per.R")
#-------------------------------------------------------------------------------
```

```
# Minima und MAxima markieren frueher R_Min_Max_mark.R
 source("04_Auswertungen_Summ_Ent_Ladung_in_Tagesper.R")
```

```
## 04_Auswertungen_Summ_Ent_Ladung_in_Tagesper.R
```

## 2.2 Perioden zwischen horizontalen Niveaus bilden - Neutrale Zyklen

Ein *neutraler Zyklus* ist eine Lade-Entlade-Vorgang der von einem Ladezustand des Akkus ausgehend zu diesem zurückkehrt. Für diese ist es sinnvoll, Wirkungsgrade als Verhältnis von Output zu Input zu bilden.

"level"" legt die Höhe des Ausgangszustands fest, der als Basis für die Berechnung von Wirkungsgraden dient. Ein solcher Zyklus kann jeweils über oder unter dem Ausgangslevel bleiben (später mit UP bzw. DOWN gekennzeichnet).

Vorerst

```
level <- 5000   # ..%-Level
```

#### 2.2.1 Bildung der Grundfunktionen

Initialisieren der Funktion "zyklus_daten_gen(xdata, l)" mit den Parametern xdata zur Übergabe von data und l zur Übergabe des Levels Erzeugt Spalten

levelpd und len_of_levelpd

Initialisieren der Funktion "zyklus_summen_gen(xdata)" mit dem Parameter xdata zur Übergabe von data

Erzeugt max_level = max(ladezustand), min_level = min(ladezustand), hub_level = max_level - min_level, mit_level = (min_level+max_level)/2, durchsatz = hub_level/len_levelpd*12, # Durchsatz = Wh zwischen Min und Max / Stunde signum = as.character(ifelse(max_level > level, “UP”,“DOWN”)), lev

#### 2.2.2 Zusammenfassung dieses Vorgangs

Dazu wird folgende Funktion definiert

```
zyklen_bilden <- function(xdata, x) {
  cat("zyklen bilden zu Level", x, "\n")
  level <-x
  xdata = zyklus_daten_gen(xdata, level)
  xdata = zyklus_summen_gen(xdata)
  red_data = zyklus_reduzieren(xdata)
  return(red_data)
}
```

Sie gibt die Auswertung zurück mit jeweils einem Wert pro Zyklus.

# 3 Graphische Auswertungen

## 3.1 Darstellung der Wirkungsgrade in Abhängigkeit von Durchsatz

Die maximale Energiedifferenz zwischen höchsten und niedrigsten Wert im Speicher in einer Halbperiode dividiert durch die Dauer der Halbperiode wird als (Energie-)Durchsatz bezeichnet angegeben in Wh/h. ‘lev’ bezeichnet dem gewählten level dividiert durch 1000.

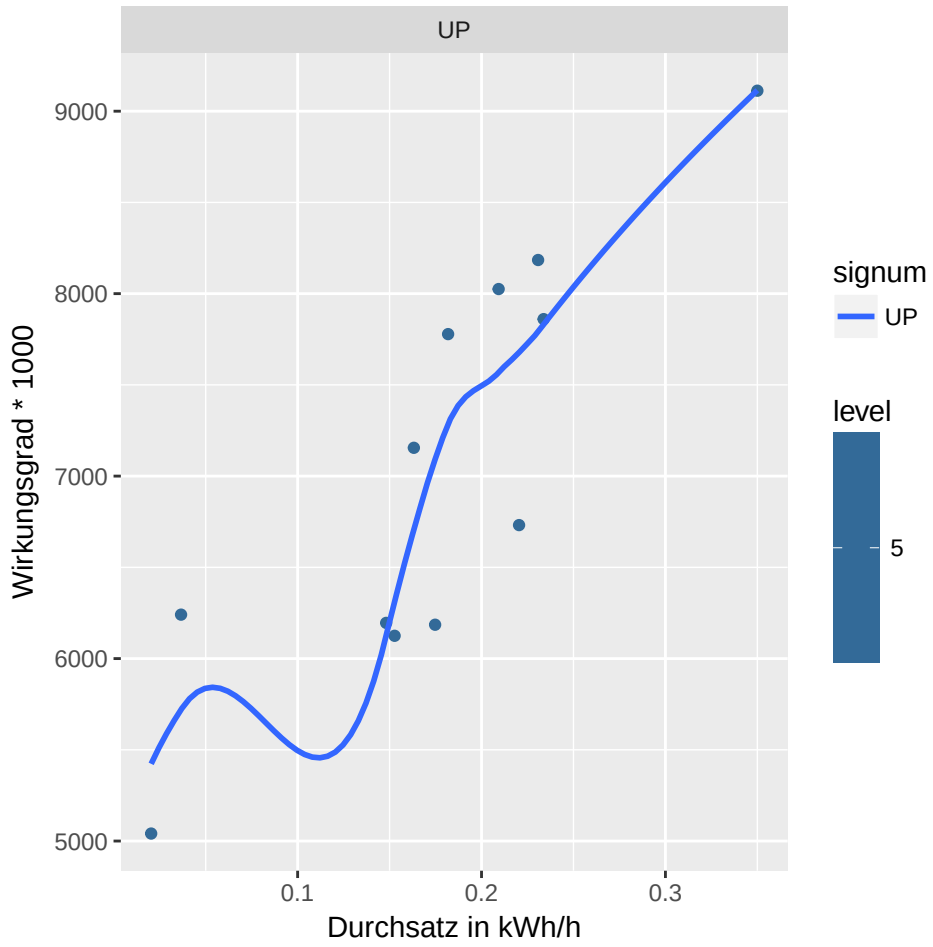
```
#------------------------------------------------------------
# Auswertung der Halbperioden

proj_level = zyklen_bilden(data, 7000)
```

```
## zyklen bilden zu Level 7000
```

```
proj_level <- proj_level %>%
  filter(eta > 0 & eta <= 10000) %>%
  filter(day >= "2017-11-01")

proj_level %>%
   ggplot(aes(x = durchsatz/1000, y = eta)) +
   geom_point(aes(x = durchsatz/1000, y = eta, color=lev)) +
   geom_smooth(mapping = aes(x = durchsatz/1000, y = eta, linetype = signum ),se =FALSE) +
   #geom_line(mapping = aes(x = durchsatz/1000, y = eta, linetype = signum )) +
   labs(
      x = "Durchsatz in kWh/h",
      y = "Wirkungsgrad * 1000",
      color = "level" )+
   facet_wrap(~ signum)
```




Duch Wahl mehrerer Levelwerte erhält man folgende Darstellung

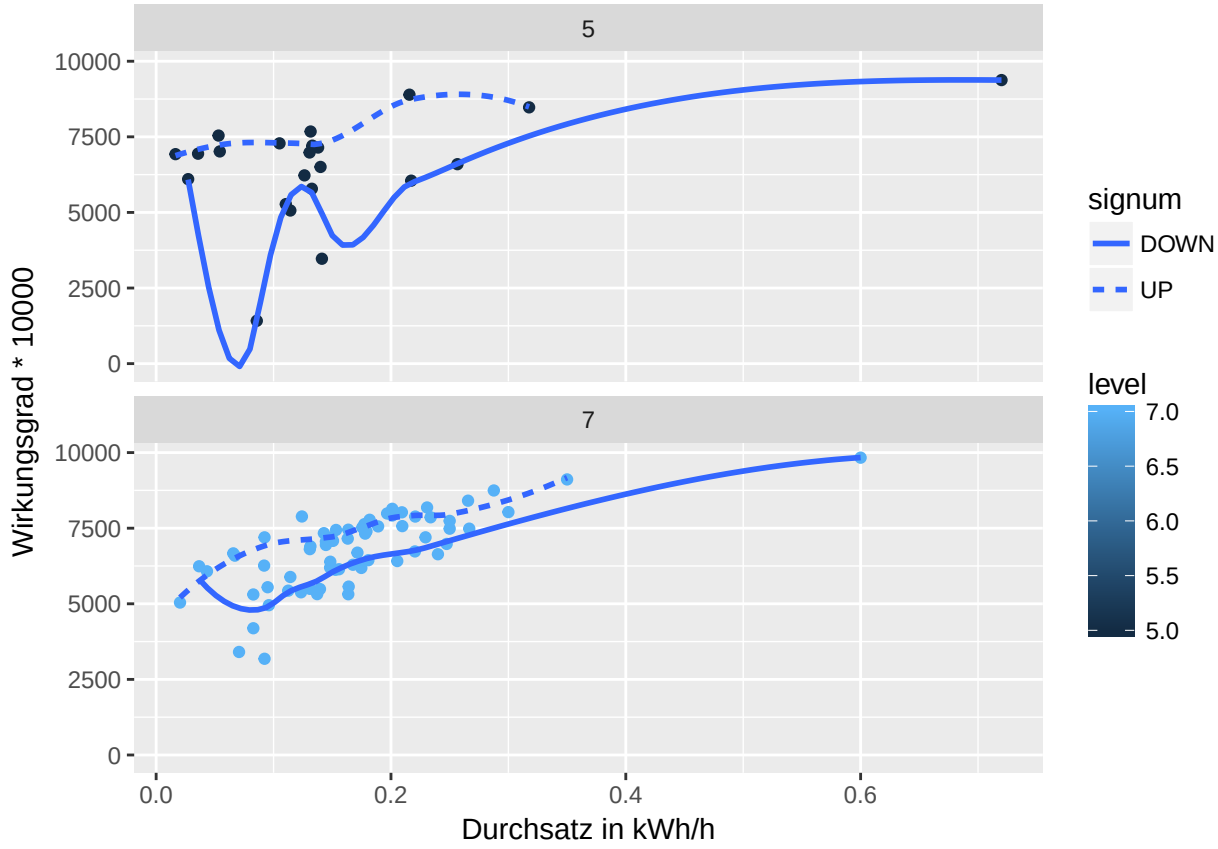
```
#some_levels <- c(2000, 3000, 4000, 4500, 5000, 5500, 6000,6500, 7000,7500, 8000, 8500, 9000)
some_levels <- c(5000, 7000)
proj_level <- tibble()
for ( level in some_levels) {  # zu jedem level den Datensatz auswerten und reduzieren auf einen Wert p
  proj_level <- rbind(proj_level, zyklen_bilden(data, level))
}
```

```
## zyklen bilden zu Level 5000
## zyklen bilden zu Level 7000
```

```
proj_level %>%
   filter(eta  != 0) %>%
   filter(eta <= 10000) %>%
   ggplot(aes(x = durchsatz/1000, y = eta)) +
   geom_point(aes(x = durchsatz/1000, y = eta, color=lev)) +
   geom_smooth(mapping = aes(x = durchsatz/1000, y = eta, linetype = signum ),se =FALSE) +
   labs(
      x = "Durchsatz in kWh/h",
      y = "Wirkungsgrad * 10000",
      color = "level"
   )  +                                          #+
   #geom_line(mapping = aes(x = durchsatz, y = eta, linetype = signum )) #+
```

```
   facet_wrap(~ lev, nrow = 4)
```

```
## `geom_smooth()` using method = 'loess'
```



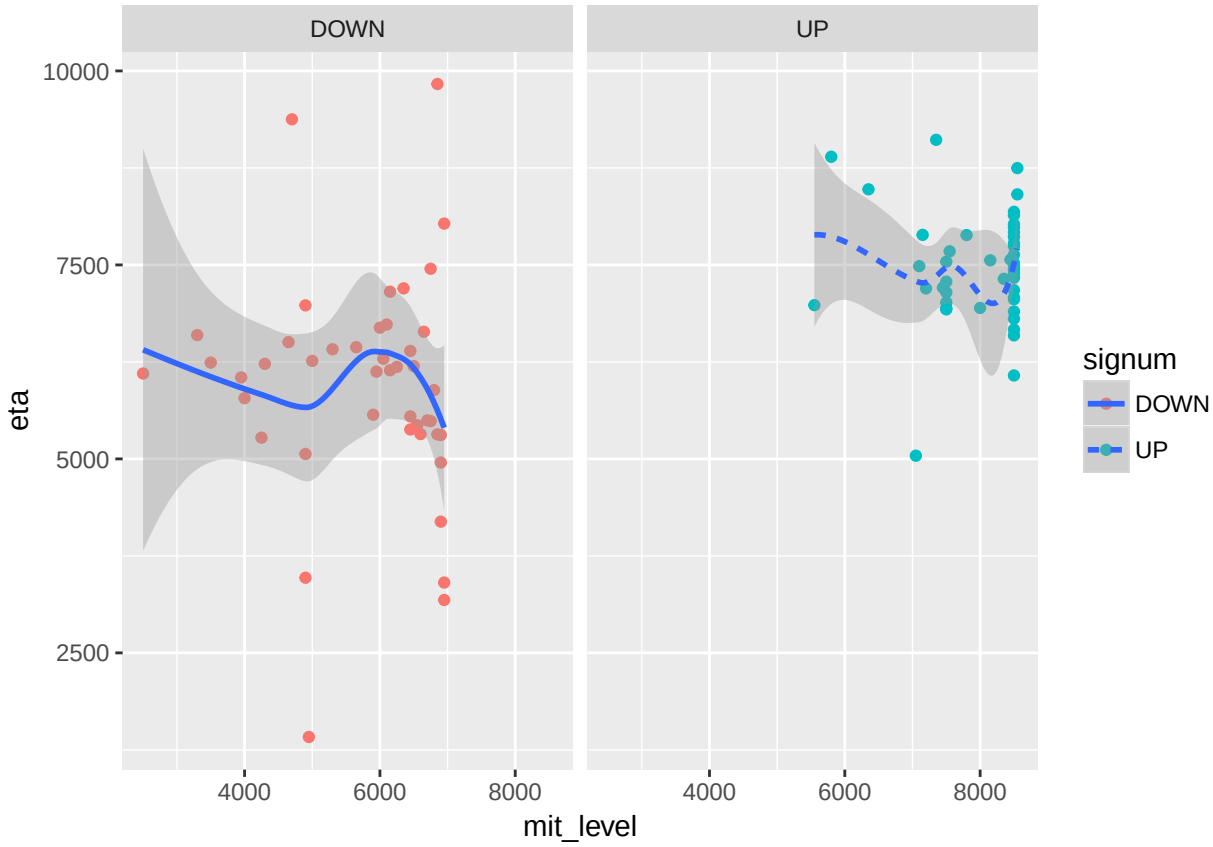

## 3.2 Darstellung der Wirkungsgrade in Abhängigkeit von der Mitte der Halbperiode

```
# ------------------------------------------------------------ Proj_Level_Eta_vs_mit_level.R
source("05_Grafik_Eta_vs_Mitte.R", print.eval=TRUE)
```

```
## zyklen bilden zu Level 7000
## # A tibble: 108 x 27
##                   zeit batt_ladung batt_entladung ladezustand    ct
##                 <dttm>       <dbl>          <dbl>       <dbl> <dbl>
##  1 2017-10-01 00:40:00     0.00000      19.333333           0     1
##  2 2017-10-01 13:35:00   127.00000       0.000000        7100   156
##  3 2017-10-02 06:20:00     0.00000       3.666667        7000   357
##  4 2017-10-02 06:25:00     0.00000       3.416667        7000   358
##  5 2017-10-02 08:50:00    93.41667       0.000000        7000   387
##  6 2017-10-03 01:55:00     0.00000       3.583333        7000   592
##  7 2017-10-03 02:00:00     0.00000       3.416667        7000   593
##  8 2017-10-03 13:40:00    85.83333       0.000000        7000   733
##  9 2017-10-03 13:45:00    59.66667       0.000000        7100   734
## 10 2017-10-04 01:45:00     0.00000      11.083333        7000   878
## # ... with 98 more rows, and 22 more variables: day <date>, hour <dbl>,
```

```
## #   ladediff <dbl>, is_min <dbl>, is_max <dbl>, daypd <dbl>,
## #   len_daypd <dbl>, day_bat_in <dbl>, day_bat_out <dbl>,
## #   day_period_ladehub <dbl>, levelpd <dbl>, len_levelpd <dbl>,
## #   lev_bat_in <dbl>, lev_bat_out <dbl>, eta <dbl>, max_level <dbl>,
## #   min_level <dbl>, hub_level <dbl>, mit_level <dbl>, durchsatz <dbl>,
## #   signum <chr>, lev <dbl>
```

```
## `geom_smooth()` using method = 'loess'
```




### 3.2.1 Das Gleiche mit Aufsammeln von Daten zu mehreren Levels

Wirkungsgrade eta=0 oder eta > 10000 werden ausgeblendet.

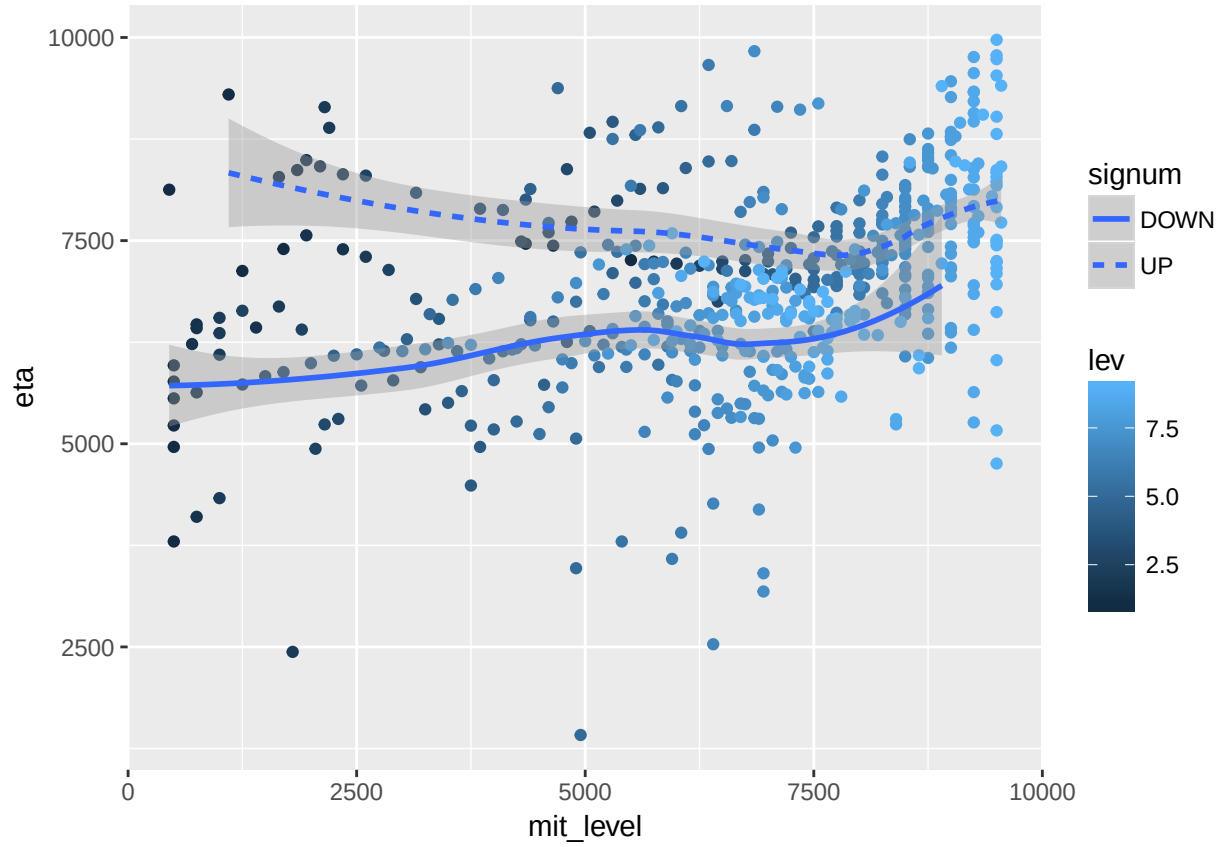
```
some_levels <- c(1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000,5500, 6000,6500, 7000,7500, 8000,
proj_level <- tibble()
for (level in some_levels) {
   proj_level <- rbind(proj_level, zyklen_bilden(data,level))
}
```

```
## zyklen bilden zu Level 1000
## zyklen bilden zu Level 1500
## zyklen bilden zu Level 2000
## zyklen bilden zu Level 2500
## zyklen bilden zu Level 3000
## zyklen bilden zu Level 3500
## zyklen bilden zu Level 4000
## zyklen bilden zu Level 4500
```

```
## zyklen bilden zu Level 5000
## zyklen bilden zu Level 5500
## zyklen bilden zu Level 6000
## zyklen bilden zu Level 6500
## zyklen bilden zu Level 7000
## zyklen bilden zu Level 7500
## zyklen bilden zu Level 8000
## zyklen bilden zu Level 8500
## zyklen bilden zu Level 9000
```

```
proj_level %>%
   filter(eta  != 0 & eta <= 10000) %>%
   ggplot(aes(x = mit_level, y = eta)) +
   geom_point(aes(x = mit_level, y = eta, color=lev)) +
   geom_smooth(mapping = aes(x = mit_level, y = eta, linetype = signum )) #+
```

```
## `geom_smooth()` using method = 'loess'
```




```
   #geom_line(mapping = aes(x = durchsatz, y = eta, linetype = signum )) +
   #facet_wrap(~ signum)
```