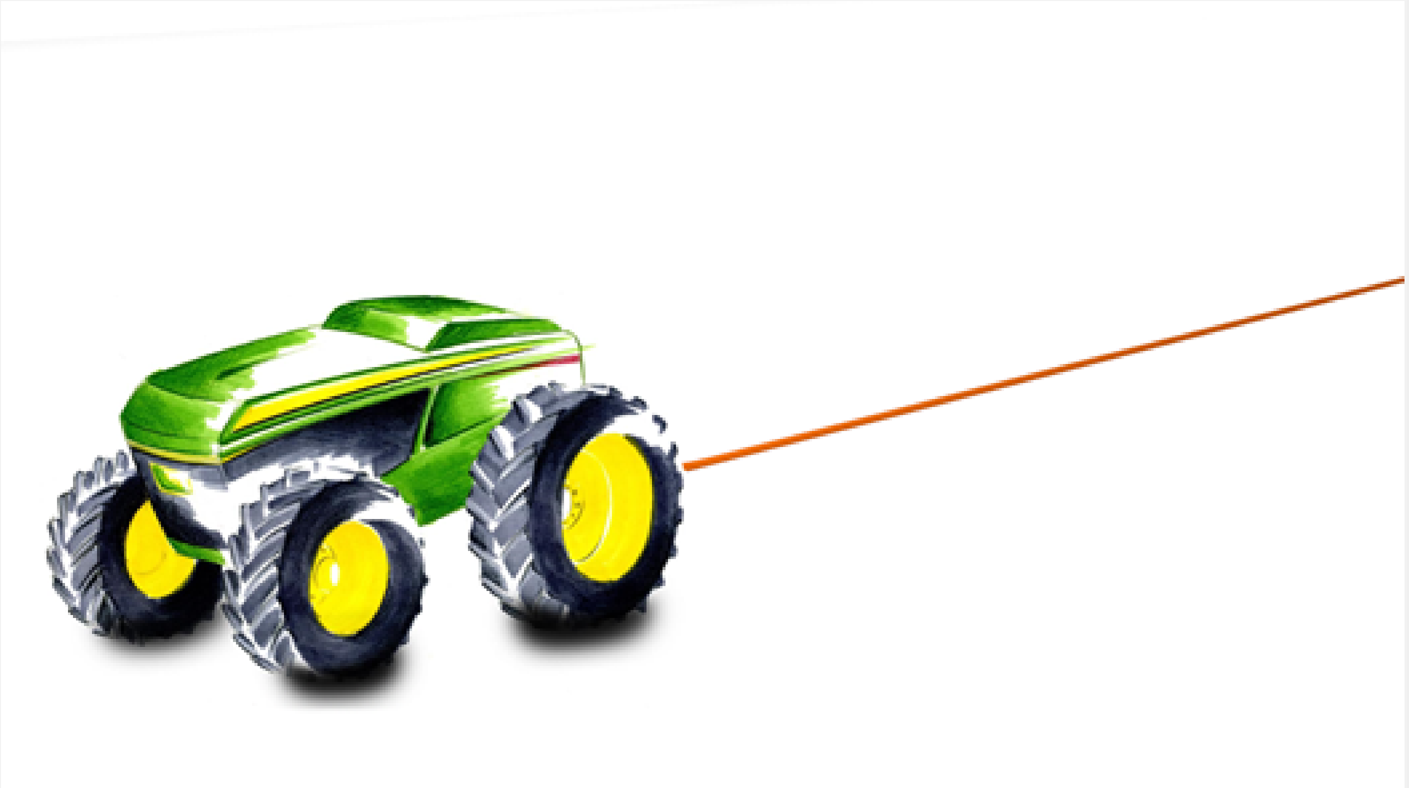
Rural Energy Systems Including Electrified High-Power Agricultural Machinery and PV Electricity Generation

oemof user meeting, 8 May 2018

Michael Stöhr, Bastian Hackenberg

Design study for autonomous electrified agricultural machine



Motivation for electrifying agricultural machines

- higher working precision
  -> saves fertilizer and chemical plant protection products can even be done mechanically
- automation of agricultural production possible
- silent operation -> operations 24 hours per day
- higher efficiency, higher power
- abundant potential for renewable electric energy generation can be used on site
- synergy between PV generation and agricultural machine operation

Two ways of electrification

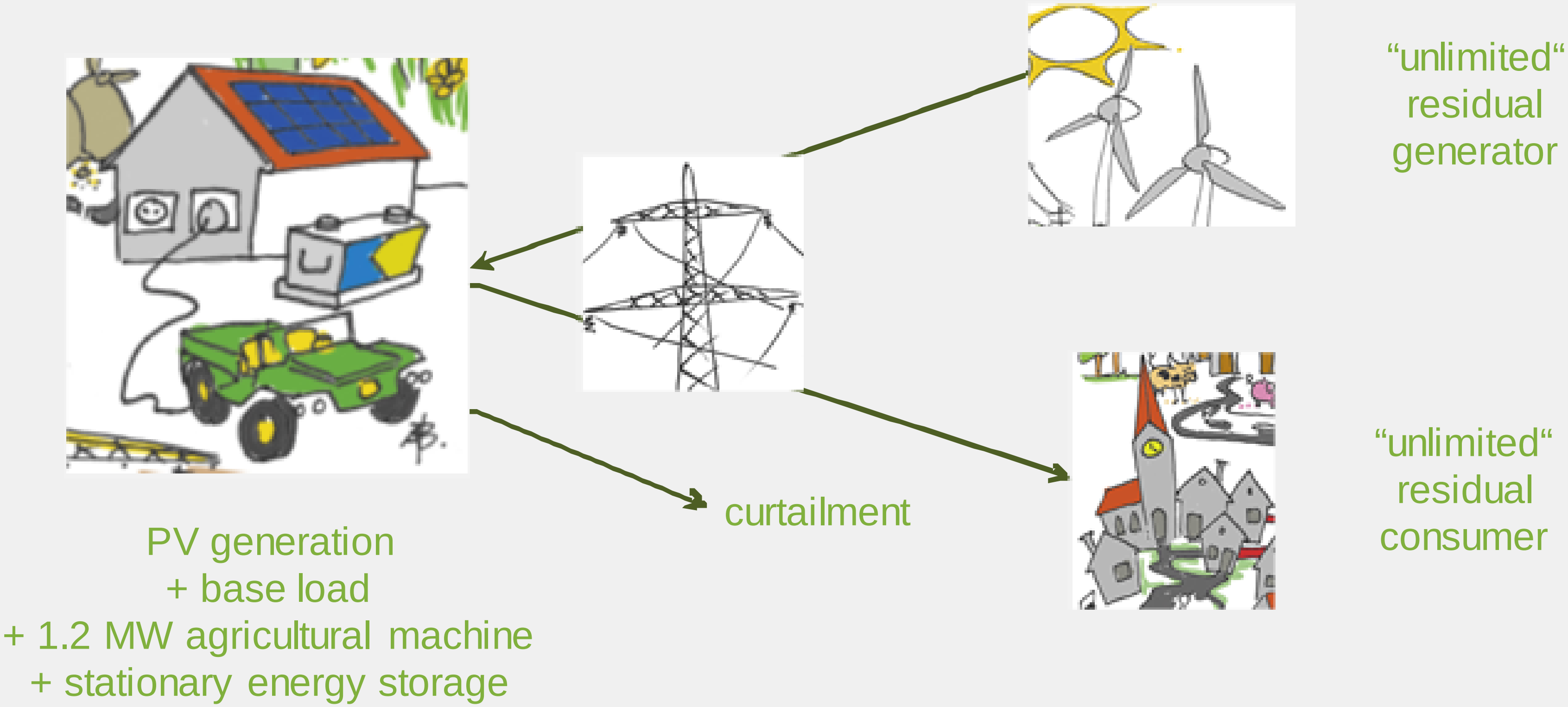
1. on-board battery

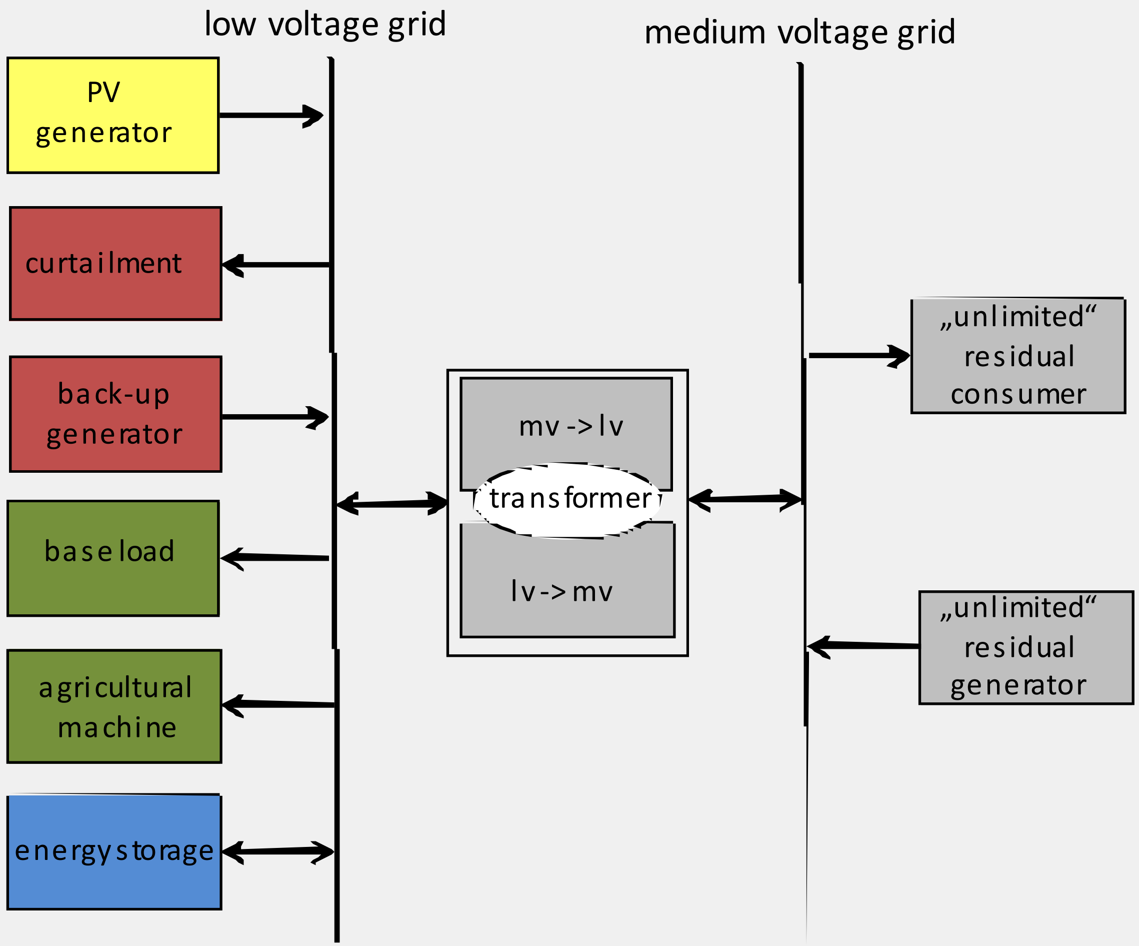
   -> only for small machines, mainly cattle breeding

2. connection to grid via 1-5 km long cable

   -> even higher power possible than with diesel engines, for cultivation

Model topology „scenario 1“ of cable-powered agricultural machine




low voltage grid
medium voltage grid
PV generator
curtailment
back-up generator
base load
agricultural machine
energy storage
mv -> lv
transformer
lv -> mv
„unlimited“ residual consumer
„unlimited“ residual generator

Calculation process

1. Non-normalised load and generation profiles input via csv-file
2. Calculation done with various versions of GridCon_storage.py, initially based on storage_investment.py
3. Results transferred to csv-file

"Black box" oemof

"Experimental set-up for observing what oemof is doing

Check and analysis of results in excel

- law of conservation of energy observed ?
- restrictions observed: SOC between 10% and 90%, etc. ?
- results reflect fixed parameters: efficiencies, max charging power, …?
- load and generation profiles not modified by oemof ?
- results reasonable/ potentially an optimum solution ?
- energy storage not charged and discharged in the same time-step ?

We did not yet really trust in oemof in the beginning …

… but in the end we were convinced ☺.

Load and generation profiles -> csv -> oemof

lines 39-52

```
# import load and generation data from csv-file and define timesteps

def optimise_storage_size(filename="GridCon1_Profile.csv",
                          solver='cbc', debug=True, number_timesteps= (96*366),
                          tee_switch=True):

    # the file "GridCon1_Profile" contains the normalised profile for the
    # agricultural base load profile L2, a synthetic electrified agricultural
    # machine load profile, and the PV generation profile ES0;
    # number_timesteps: one timestep has a duration of 15 minutes,
    # hence, 96 is the number of timesteps per day;
    # 366 is the number of days in a leap year, chosen here because
    # standard load profils of 2016 are used;
    # the total number of timesteps is therefore 96*366 = 35136;
```

Output of results from oemof -> csv -> excel

lines 552-556

```
def create_csv(energysystem):

    results = outputlib.ResultsDataFrame(energy_system=energysystem)
    results.bus_balance_to_csv(bus_labels=['b_el_lv'],
                               output_path='results_as_csv_LV_Net')
```

- flows on low-voltage bus are sufficient for full analysis
- results are checked and analysed, and plots are generated in excel

oemof minimises ....

… total annual costs of energy system’s

infrastructure and related electricity losses

= annuity of grid and energy storage system

\+ fixed annual costs (2 % of initial investment costs)

\+ variable annual costs (energy lost by grid transmission, storage or curtailment of PV generation, value set at 0.065 €/kWh)

\- income generated by primary reserve provision

Specific annual fixed grid and storage costs

grid		
specific investment costs	500	€/kW
annual cost decrease	-	-
financial life time	50	a
financial period considered	50	a
number of investments	1	
weighted average cost of capital	5.0%	
annuity	27.39	€/kW
fixed operational costs	10.00	€/kW
fixed annual costs	37.39	€/kW

energy storage		
specific investment costs	300	€/kWh
annual cost decrease	10%	1/a
financial life time	5	a
financial period considered	50	a
number of investments	10	
weighted average cost of capital	5.0%	
annuity	30.57	€/kWh
fixed operational costs	6.00	€/kWh
fixed annual costs	36.57	€/kWh

Definition of grid investment costs

```
n = 50
…
invest_grid = 500
…
wacc = 0.05
…
u_grid = 50
…
cost_decrease_grid = 0
…
oc_rate_grid = 0.02
…
oc_grid = oc_rate_grid * invest_grid
…
sepc_grid = economics_BAUM.epc(invest_grid, n, u_grid, wacc, cost_decrease_grid, oc_grid)
```

lines 79-105, 116-156

Definition of storage investment costs

```
u_el_lv_1_storage = 5
#…
cost_decrease_el_lv_1_storage = 0.1
#…
oc_rate_el_lv_1_storage = 0.02
#…
oc_el_lv_1_storage = oc_rate_el_lv_1_storage * invest_el_lv_1_storage
#…
sepc_el_lv_1_storage = economics_BAUM.epc(invest_el_lv_1_storage, n,
u_el_lv_1_storage, wacc, cost_decrease_el_lv_1_storage, oc_el_lv_1_storage)
#…
kS_el = sepc_el_lv_1_storage
#…
```

lines 158-192

+ definition of n and wacc

Taking income from primary reserve provision into account

lines 194-218

```
prl_on = 1
#…
prl_weeks = 13
#…
prl_income = 2.4 * prl_on * prl_weeks
#…
sepc_el_lv_1_storage = sepc_el_lv_1_storage - prl_income
#…
kS_el_netto = sepc_el_lv_1_storage
#…
```

2.4 * 1 * 13 = 31.2

income from primary balancing power provision:
13 weeks * 3,000 €/week/MW -> 31.2 €/kWh

refers to nominal capacity of energy storage

if this becomes negative, no solution can be found with oemof 0.1
-> limits number of weeks of PR and values of wacc that can be considered

Definition of transformer(s)

lines 322-399

```
#…
grid_loss_rate = 0.0685
#…
grid_eff = 1 - grid_loss_rate
#…
transformer_mv_to_lv = solph.LinearTransformer(label="transformer_mv_to_lv",
    inputs={b_el_mv: solph.Flow(variable_costs = (grid_loss_rate *cost_electricity_losses))},
    outputs={b_el_lv: solph.Flow(investment=solph.Investment(ep_costs=0.5 * sepc_grid))},
    conversion_factors={b_el_lv: grid_eff})
#…
transformer_lv_to_mv = solph.LinearTransformer(label="transformer_lv_to_mv",
     inputs={b_el_lv: solph.Flow(investment = solph.Investment(ep_costs = 0.5 * sepc_grid))},
outputs={b_el_mv: solph.Flow(variable_costs =
(cost_electricity_losses*grid_loss_rate/grid_eff))},
      conversion_factors = {b_el_mv: grid_eff})
#…
```

2 transformer objects represent
1 physical transformer + up-stream grid !

6.85% of electricity of
residual generation is lost

ratio of flows is 1-6.85%

6.85%/(1-6.85%) of locally
generated electricity
arriving at residual
consumer is lost

Ensuring equal size of two transformer objects

lines 461-474

Without these code lines, effectively oemof delivers two objects with different size !

```
def connect_invest_rule(m):
    expr = (om.InvestmentFlow.invest[b_el_lv, transformer_lv_to_mv] ==
        om.InvestmentFlow.invest[transformer_mv_to_lv, b_el_lv])
    return expr

  my_block.invest_connect_constr = environ.Constraint(
      rule=connect_invest_rule)
  om.add_component('ConnectInvest', my_block)

#...
```

Definition of energy storage

```
# …
icf = 0.95                          lines 400-446

ocf = 0.95
# …
el_storage_conversion_factor = icf * ocf
# …
solph.Storage(label='el_lv_1_storage',
   inputs={b_el_lv: solph.Flow(variable_costs = cost_electricity_losses
   *(1-el_storage_conversion_factor))}, outputs={b_el_lv: solph.Flow()},
   capacity_min = 0.1, capacity_max = 0.9, nominal_input_capacity_ratio = 1,
   nominal_output_capacity_ratio = 1, inflow_conversion_factor = icf,
   outflow_conversion_factor = ocf, capacity_loss = 0.0000025,
   investment=solph.Investment(ep_costs = sepc_el_lv_1_storage))
# …
```

> 95% * 95% of energy input is lost .
Self-discharge is not linked to a flow and cannot be considered in variable costs !

Definition of PV generator (source)

ensures right position in output file

input data represent average power in time-step in kW; they are not normalised.

lines 247-258

```
solph.Source(label='el_lv_7_pv', outputs={b_el_lv:
    solph.Flow(actual_value=data['pv'], nominal_value = 1, fixed=True)})

# represents aggregated pv power plants in investigated area which are looked
# at as a single source of energy;
# "outputs={b_el_lv: ...}" defines that this source is connected to the
# low voltage grid;
# "solph.Flow ..." defines properties of this connection: actual_value get
# the pv generation data for all time intervals from csv-file;
# "nominal_value = 1" signifies that pv generation data do …
# processing, they are already absolute figures in kW;
# "fixed=True" signifies that these data are not modified by ….
```

selects correct data from input csv-file:

```
timestep,demand_el,pv,machine_load
1,69.772,0,0
2,67.18,0,0
3,65.384,0,0
4,64.092,0,0
5,63.292,0,0
```

Definition of curtailment (sink)

```
cost_electricity_losses = 6.5E-2
```

defines cost of electricity losses in €/kWh

lines 302-308

```
# ….

curtailment = solph.Sink(label='el_lv_4_excess_sink', inputs={b_el_lv:
    solph.Flow(variable_costs = cost_electricity_losses)})
```

lines 310-320

```
# ….
```

Specific variable costs in €/kWh indicated.

Physical units are ok: [kW] * [€/kWh] * [h] = [€]
Comparison with results in excel confirms figures.

```
…..

print('Costs of curtailment:      ', sum(energysystem.results[b_el_lv]
                [curtailment]) * cost_electricity_losses * time_step, '€')
```

lines 329-330

Definition of back-up generator (source)

lines 260-265

```
solph.Source(label='el_lv_6_grid_excess', outputs={b_el_lv: solph.Flow(
          variable_costs = 100000000)})

   # dummy producer of electric energy connected to low voltage grid;
   # introduced to ensure energy balance in case no other solution is found;
   # extremely high variable costs ensure that source is normally not used;
```

allows solver running also through senseless solutions on the way to finding the optimum

No need for a back-up sink ...

.... this job is done by
the curtailment sink.

Definition of base load (sink)

ensures right position in output file

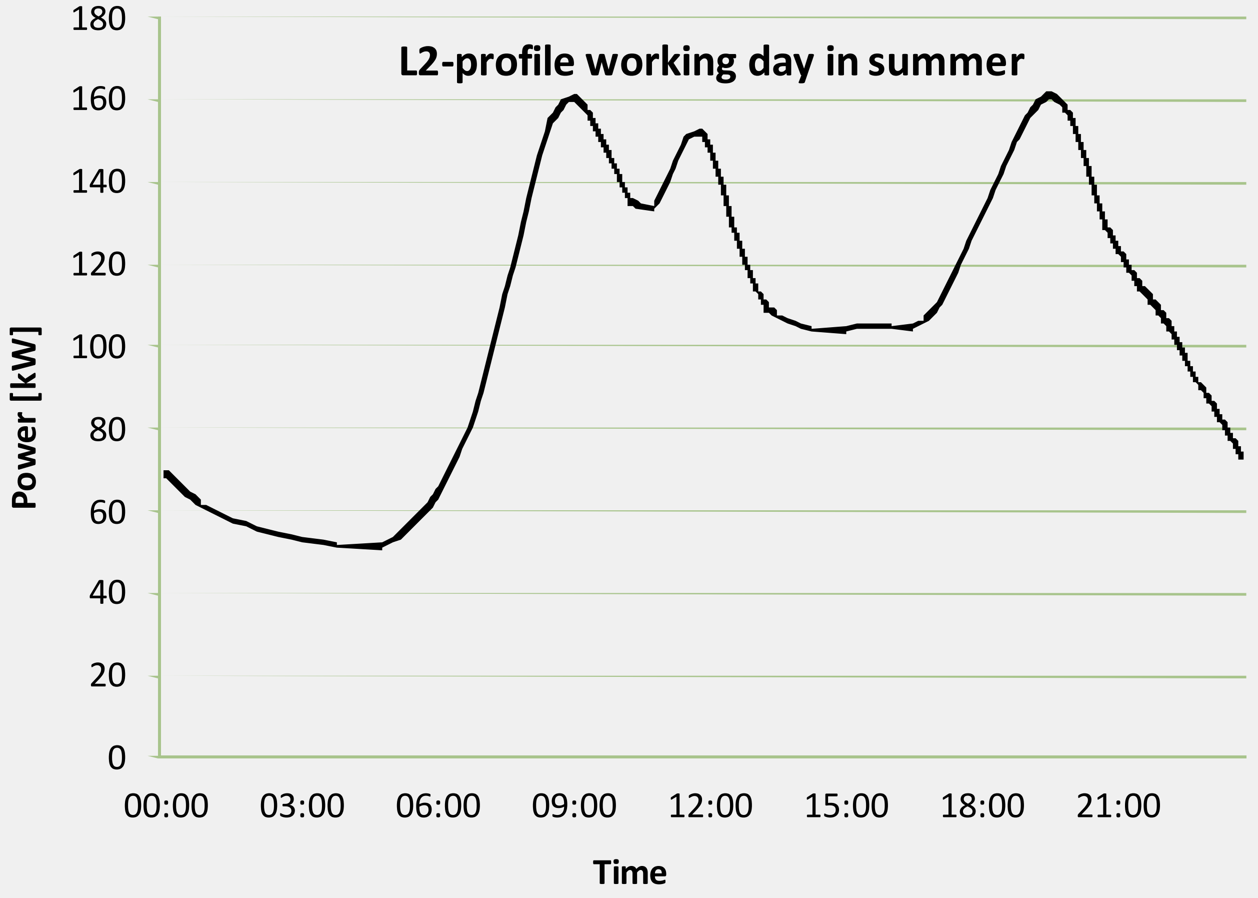
input data represent average power in time-step in kW; they are not normalised.

lines 276-286

```
solph.Sink(label='el_lv_2_base_load', inputs={b_el_lv:
    solph.Flow(actual_value=data['demand_el'], nominal_value= 1, fixed=True)})

  # represents base load in low voltage (lv) electric grid;
  # "inputs={b_el_lv: ...}" defines that this sink is connected to the
  # low-voltage electric grid;
  # "solph.Flow ..." defines properties of this connection: actual_value
  # gets the base load data for all time intervals from csv-file;
  # "nominal_value = 1" signifies that base load data do not …
  # processing, they are already absolute figures in kW;
  # "fixed=True" signifies that these data are not modified by …
```

selects correct data from input csv-file:

```
timestep,demand_el,pv,machine_load
1,69.772,0,0
2,67.18,0,0
3,65.384,0,0
4,64.092,0,0
5,63.292,0,0
```


L2-profile working day in summer
Power [kW]
180
160
140
120
100
80
60
40
20
0
00:00
03:00
06:00
09:00
12:00
15:00
18:00
21:00
Time

Definition of machine load (sink)

ensures right position in output file

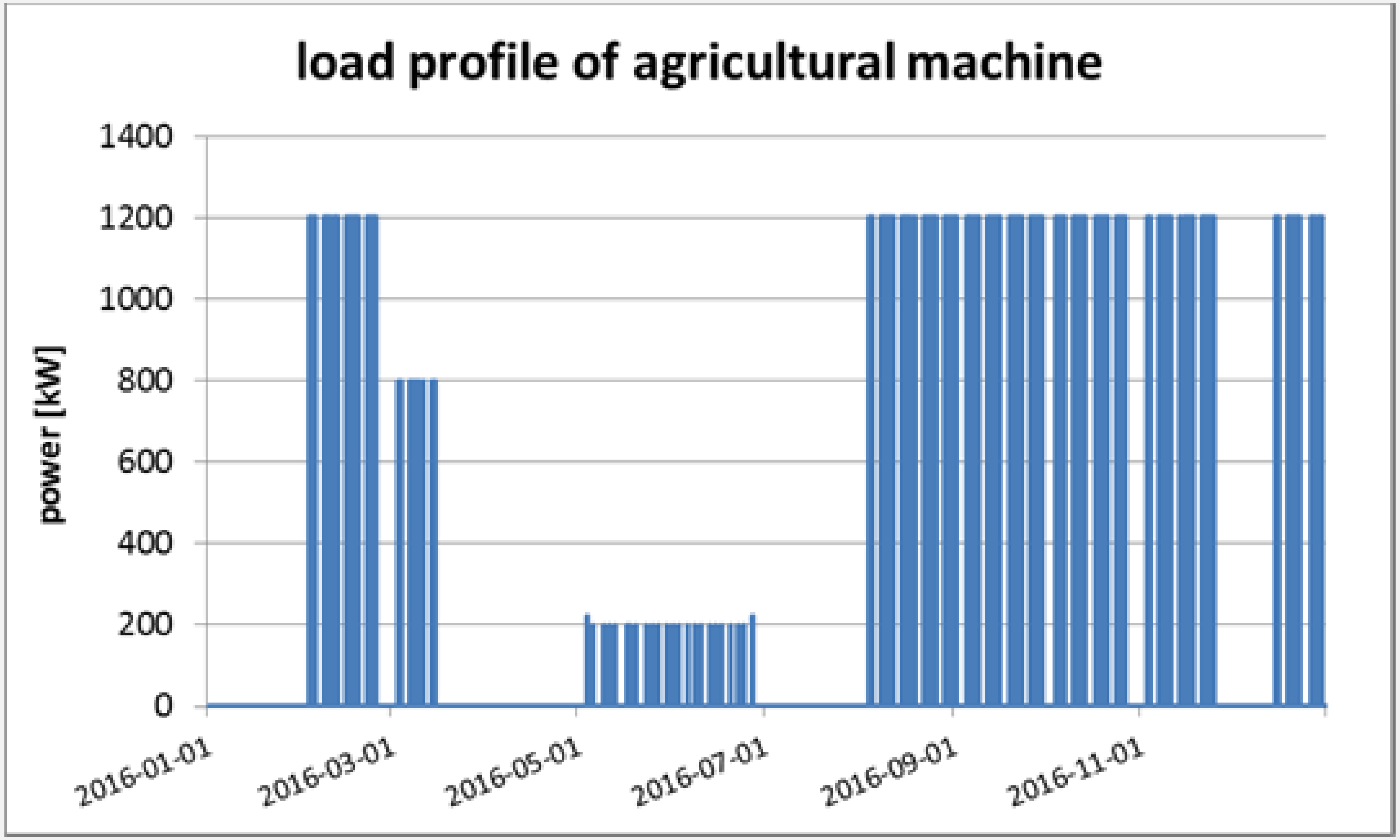
input data represent average power in time-step in kW; they are not normalised.

lines 288-300

```
solph.Sink(label='el_lv_3_machine_load', inputs={b_el_lv:
    solph.Flow(actual_value=data['machine_load'], nominal_value= 1, fixed=True)})

  # represents electrified agricultural machine connected to lv-grid;
  # "inputs={b_el_lv: ...}" defines that this sink is connected to the
  # low voltage electric grid;
  # "solph.Flow ..." defines properties of this connection: actual_value
  # gets the electrified agricultural machine load data for all time
  # intervals from csv-file;
  # "nominal_value = 1" signifies that base load data do not …
  # processing, they are already absolute figures in kW;
  # "fixed=True" signifies that these data are not modified by …
```

selects correct data from input csv-file:

```
timestep,demand_el,pv,machine_load
1,69.772,0,0
2,67.18,0,0
3,65.384,0,0
4,64.092,0,0
5,63.292,0,0
```


load profile of agricultural machine
power [kW]
1400
1200
1000
800
600
400
200
0
2016-01-01
2016-03-01
2016-05-01
2016-07-01
2016-09-01
2016-11-01

Definition of “unlimited” residual generator (source)

lines 242-245

Flow is a result of optimisation process.

```
solph.Source(label='mv_source', outputs={b_el_mv: solph.Flow()})

    # represents aggregated electric generators at a far point in the up-stream
    # grid; here, no limit is considered for this source;
```

Definition of “unlimited” residual consumer (source)

Flow is a result of optimisation process.

lines 271-274

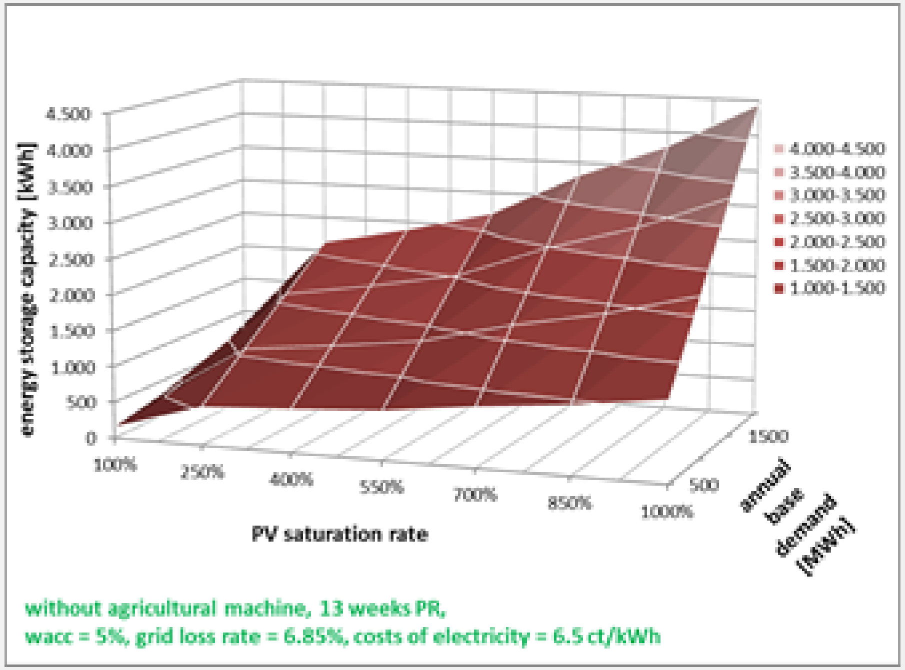
```
solph.Sink(label='el_mv_sink',  inputs={b_el_mv:  solph.Flow()})

    # represents  aggregated consumers  at a far point in the  up-stream  grid;
    # here,  it is assumed that  no limit exists for this  sink;
```

Parameters characterising different situations

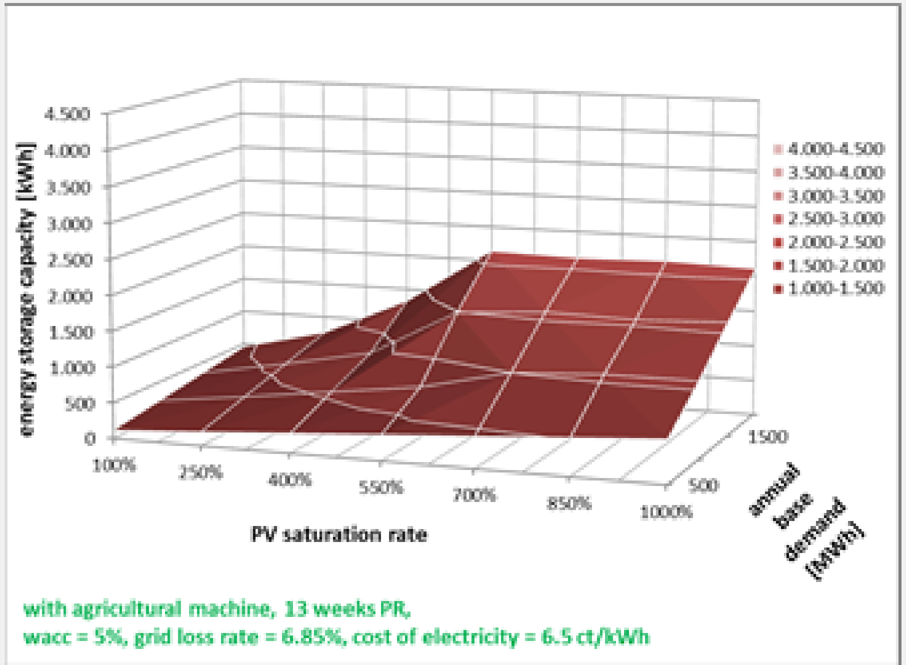
1. Base electric energy demand in rural local grid (MWh/yr)
2. PV saturation rate: factor by which a grid connection just meeting the peak base load needs to be reinforced to allow for complete feed-in of PV electricity not consumed locally (e.g. 234% corresponds to 100% net PV supply of local base load)

Parameter values are entered via load and generation profiles in csv input file.

Energy storage capacity

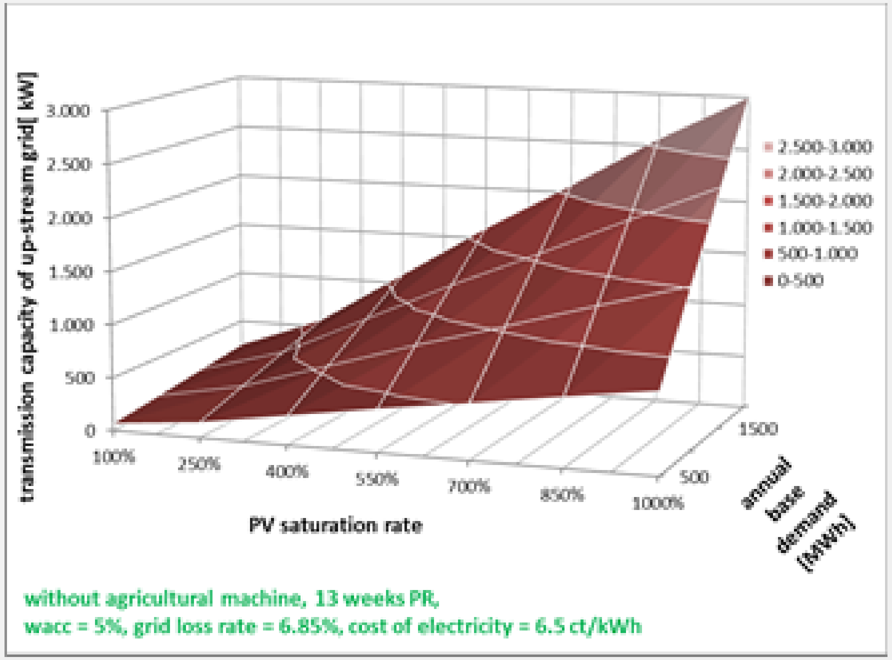


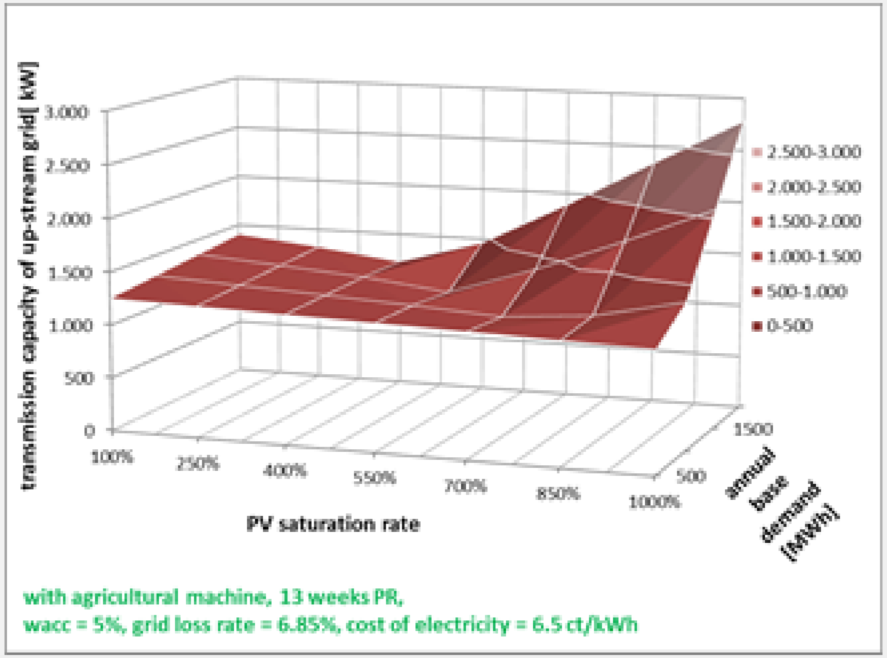
without agricultural machine



with agricultural machine

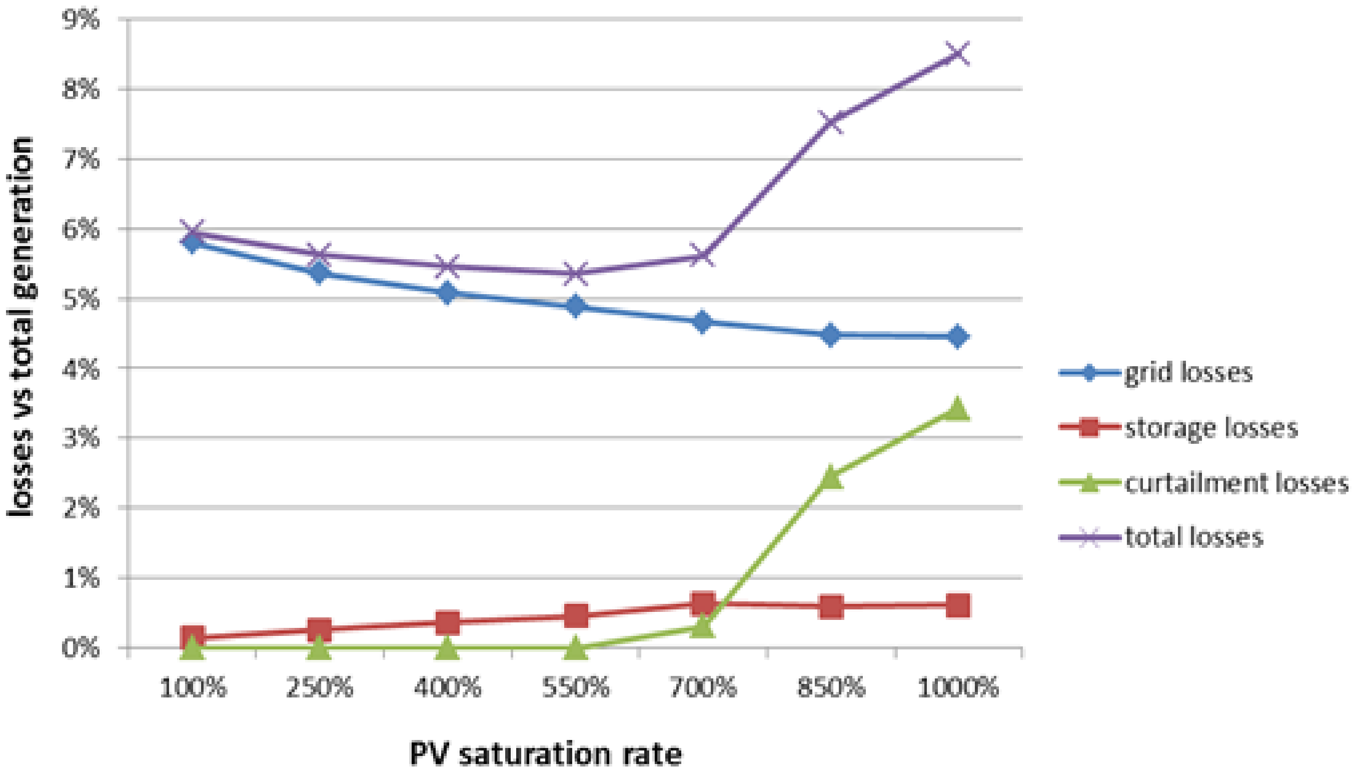
Transmission capacity of up-stream grid





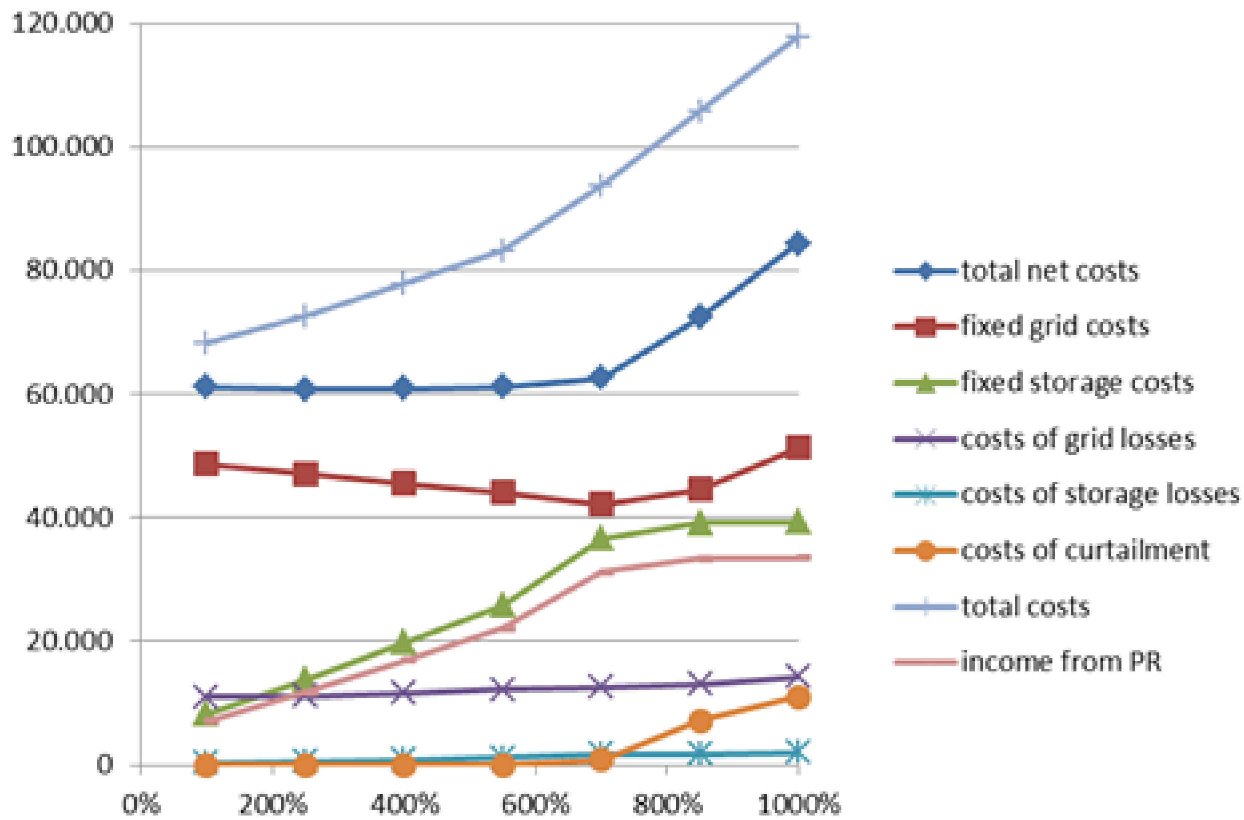
without agricultural machine

with agricultural machine

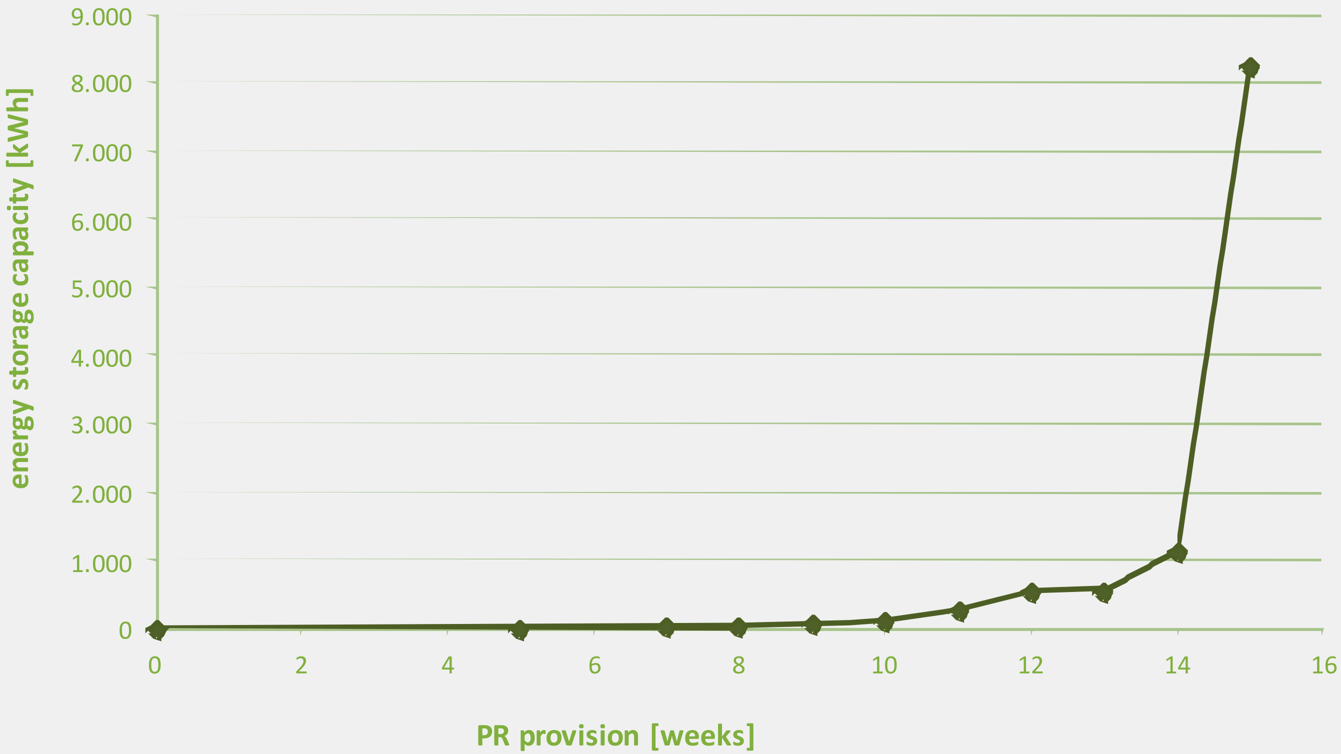


annual base demand 1.000 MWh, agr. mach. ON, 13 weeks PR,
wacc = 5%, grid loss rate = 6,85%, costs of electricity = 6.5 ct/kWh

B.A.U.M.



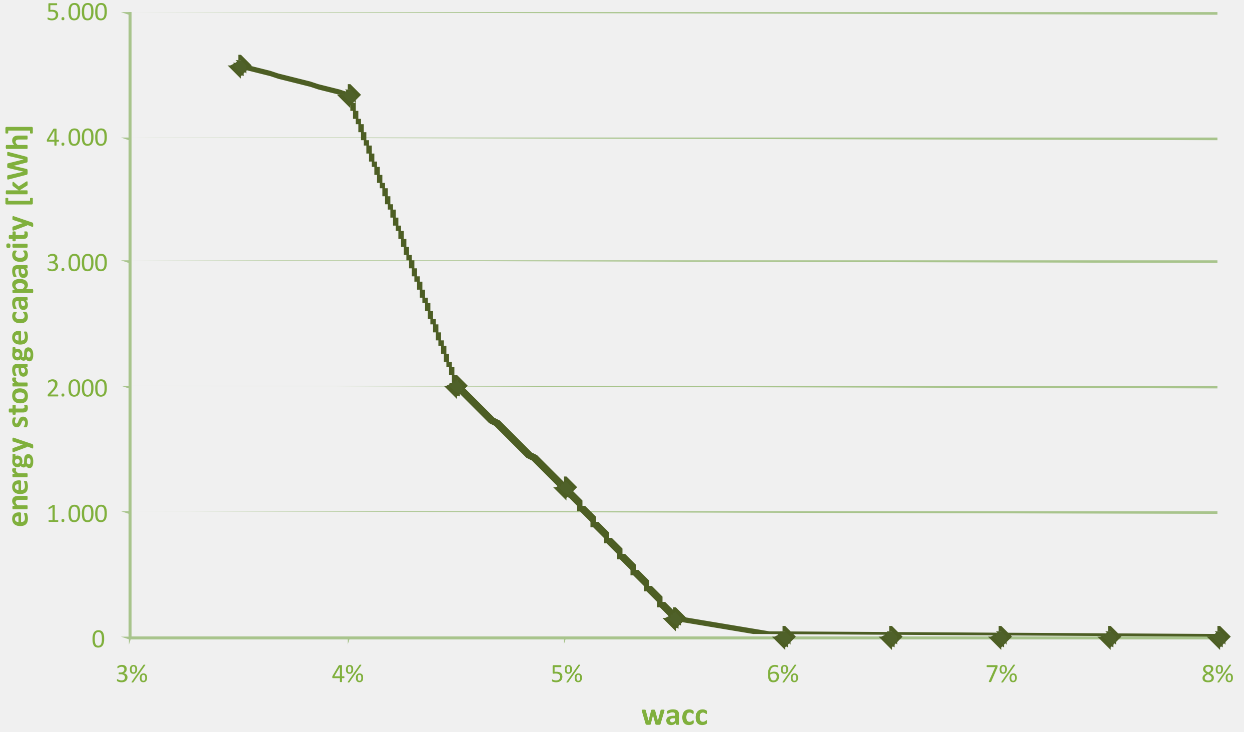
annual base demand 1.000 MWh, agr. mach. ON, 13 weeks PR,
wacc = 5%, grid loss rate = 6,85%, costs of electricity = 6.5 ct/kWh



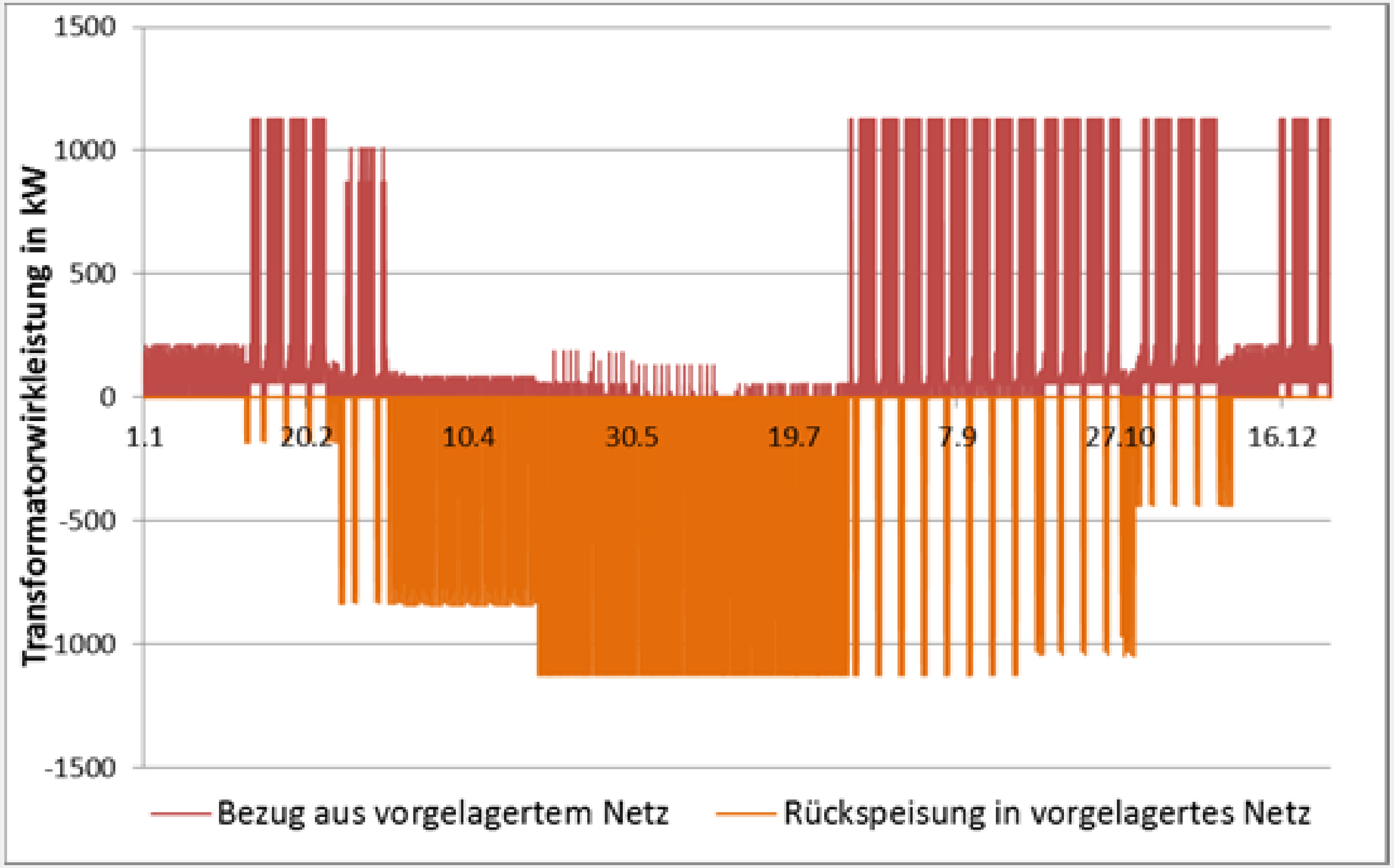
Above 15 weeks PR provision GridCon_storage.py cannot find a solution, because investment costs cannot be negative.

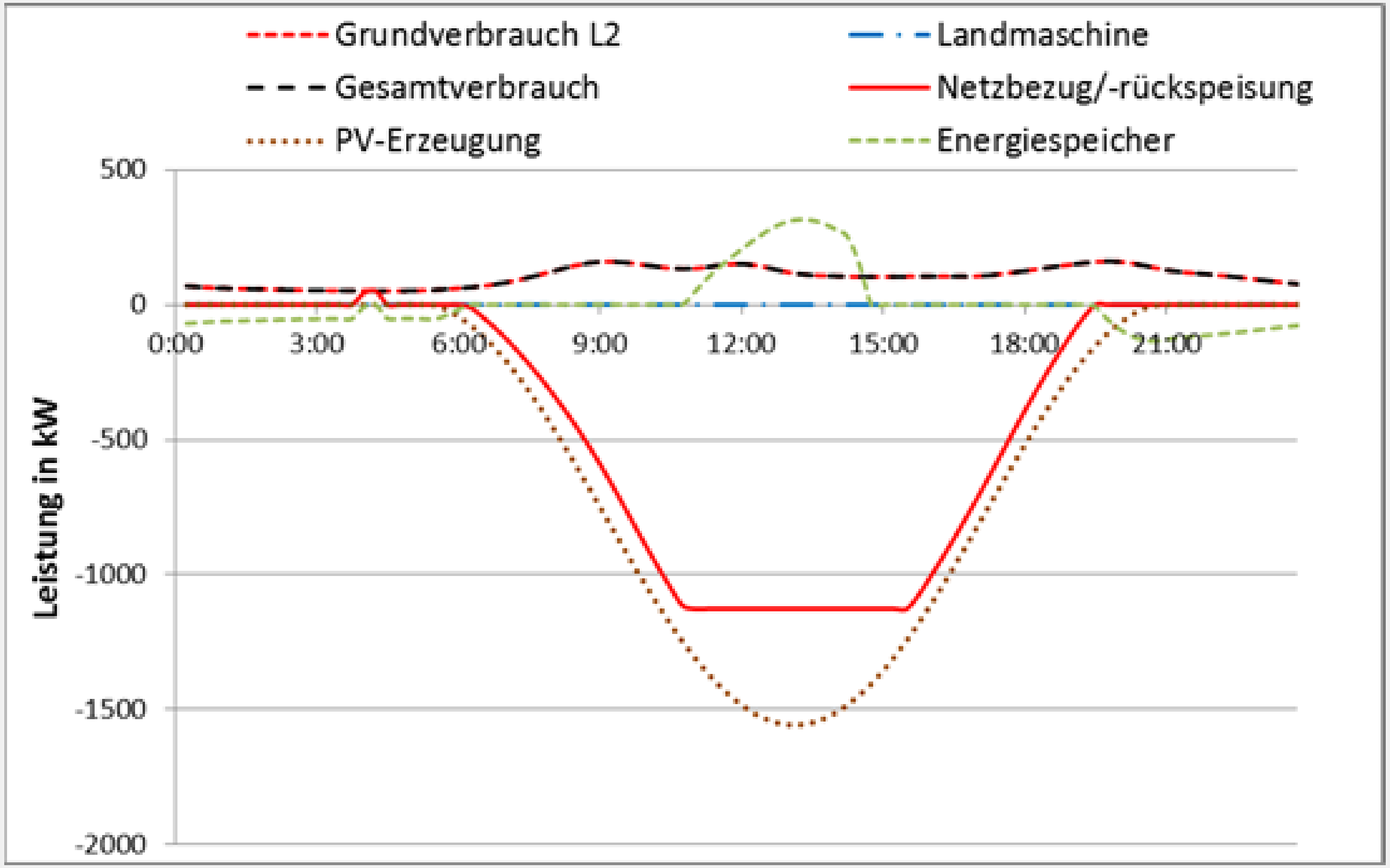
with agricultural machine, PR provision variable, wacc = 5%, grid loss rate = 6,85%, costs of electricity = 6.5 ct/kWh

Below wacc = 3.5% GridCon_storage.py cannot find a solution, because investment costs cannot be negative.

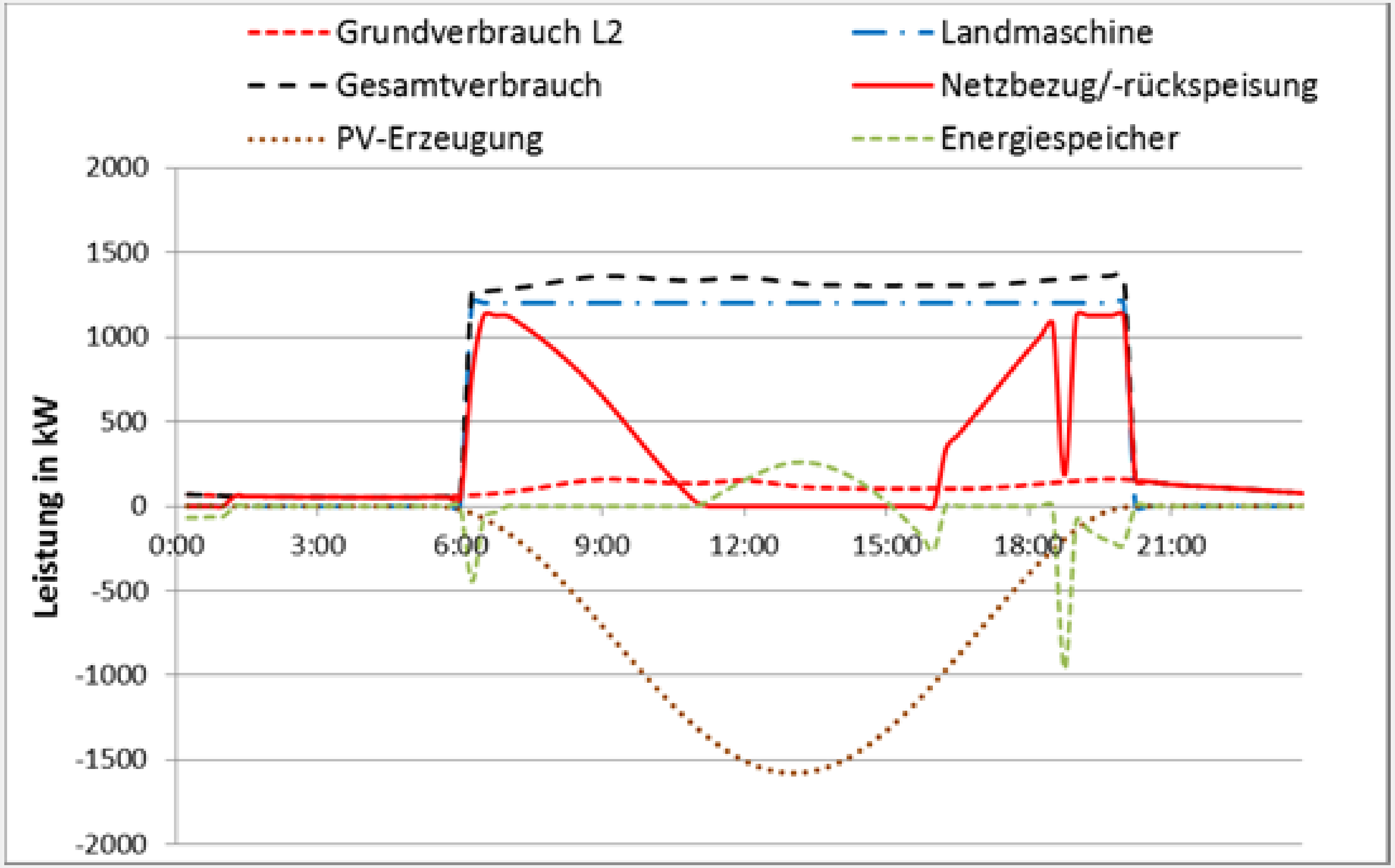


with agricultural machine, 13 weeks PR,
wacc = variabel, grid loss rate = 6,85%, costs of electricity = 6.5 ct/kWh


1500
1000
500
0
-500
-1000
-1500
Transformatorwirkleistung in kW
1.1
20.2
10.4
30.5
19.7
7.9
27.10
16.12
Bezug aus vorgelagertem Netz
Rückspeisung in vorgelagertes Netz

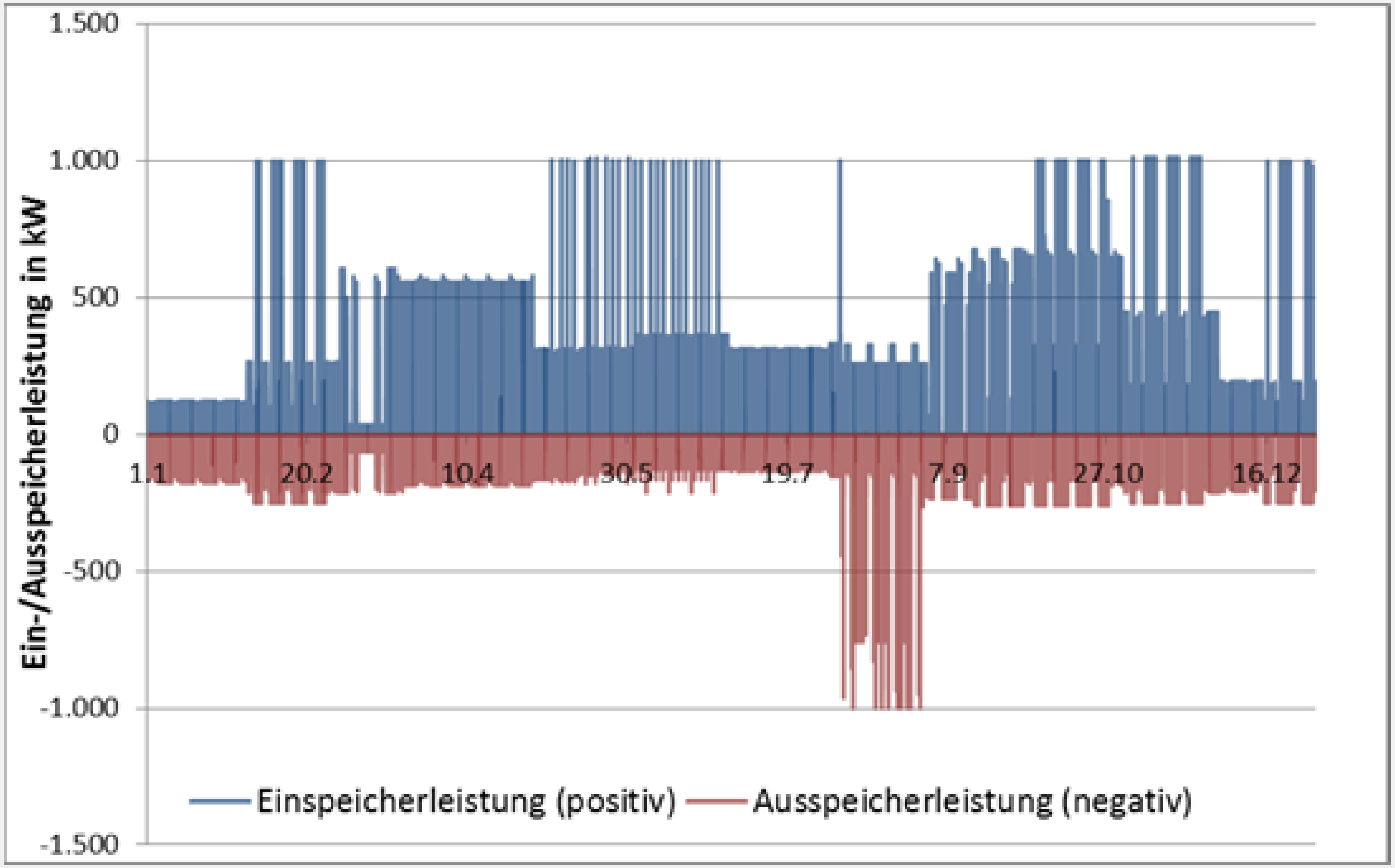


Grundverbrauch L2
Gesamtverbrauch
PV-Erzeugung
Landmaschine
Netzbezug/-rückspeisung
Energiespeicher
Leistung in kW
500
0
-500
-1000
-1500
-2000
0:00
3:00
6:00
9:00
12:00
15:00
18:00
21:00

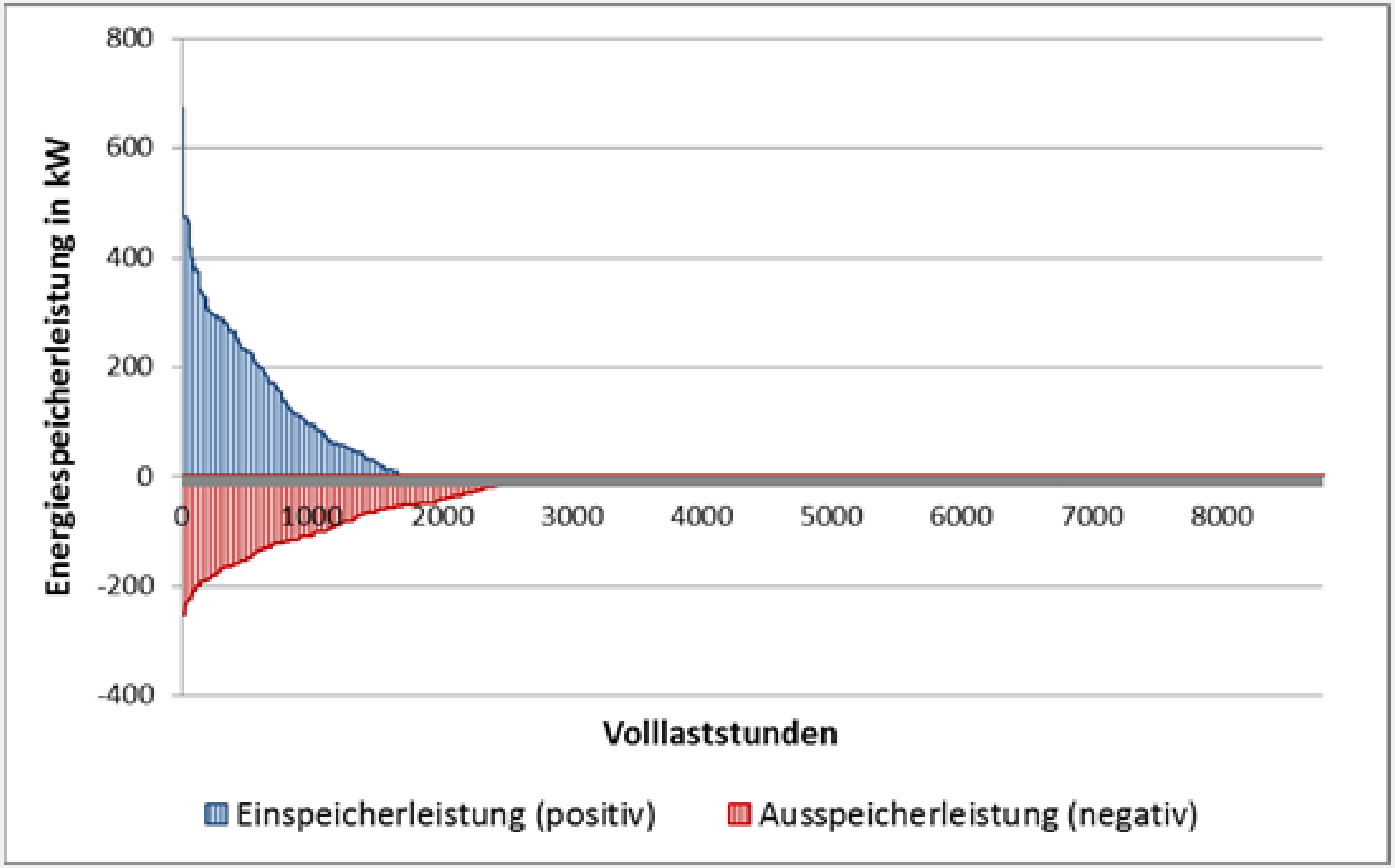


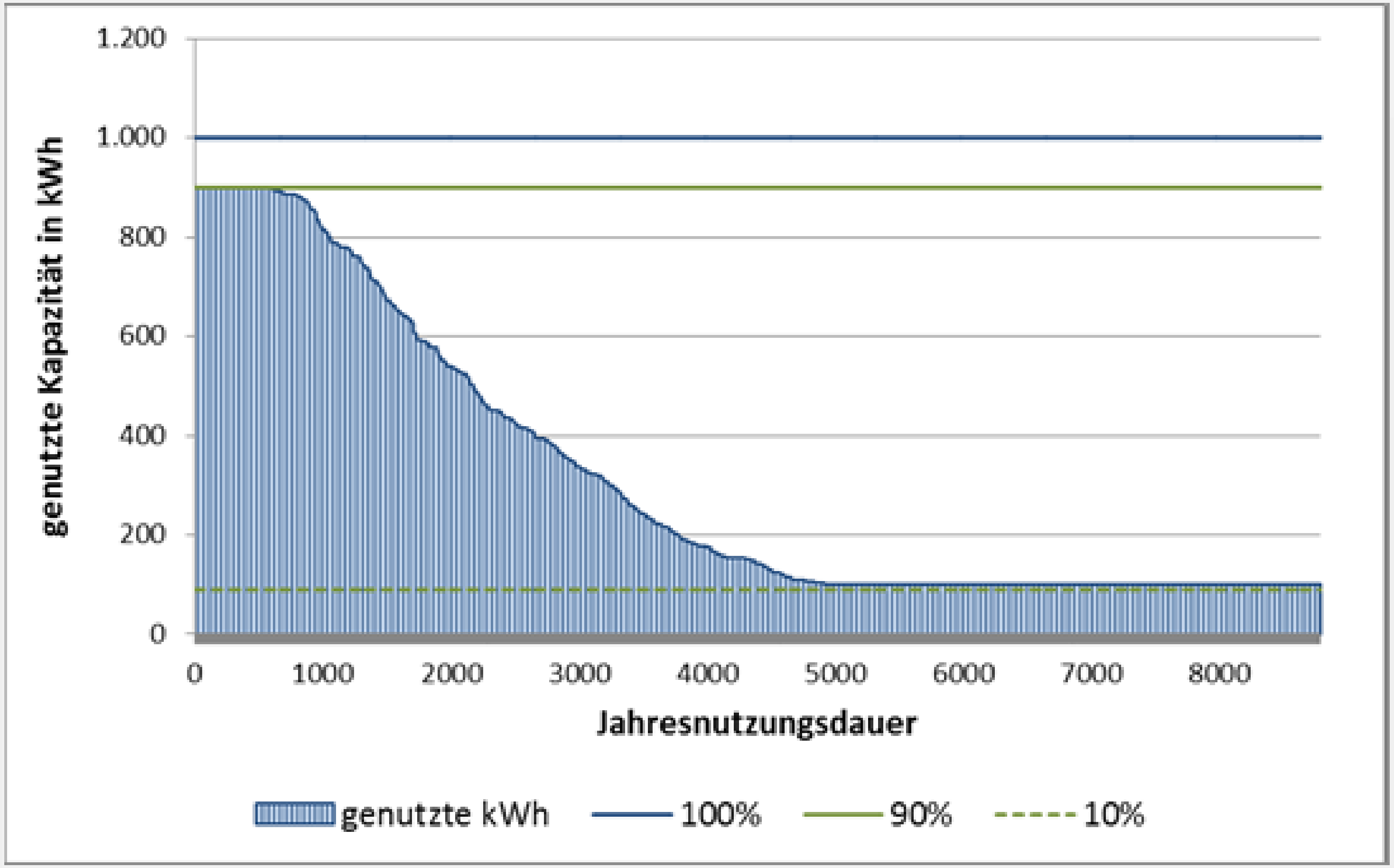
Grundverbrauch L2
Landmaschine
Gesamtverbrauch
Netzbezug/-rückspeisung
PV-Erzeugung
Energiespeicher
Leistung in kW
2000
1500
1000
500
0
-500
-1000
-1500
-2000
0:00
3:00
6:00
9:00
12:00
15:00
18:00
21:00

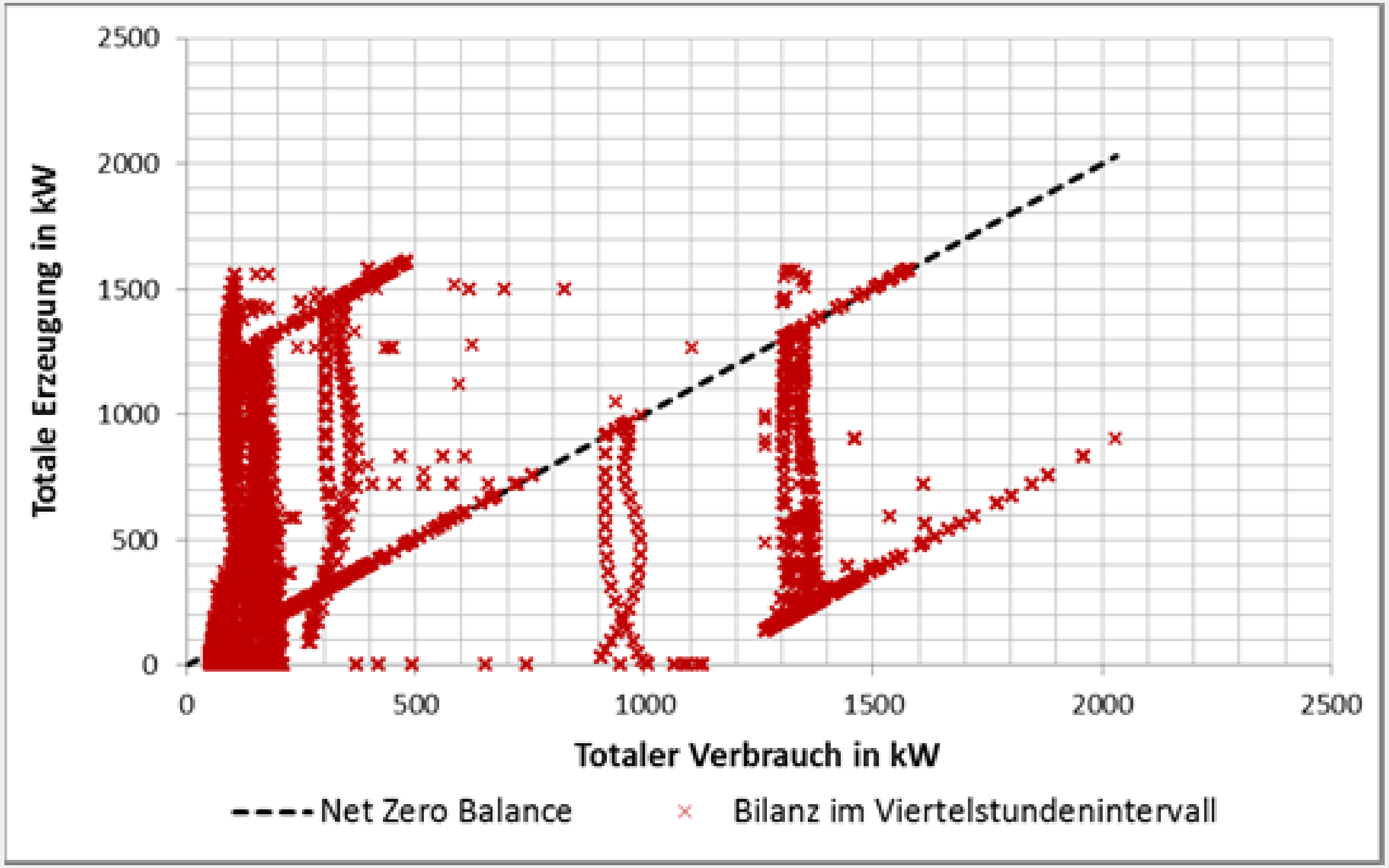
B.A.U.M.



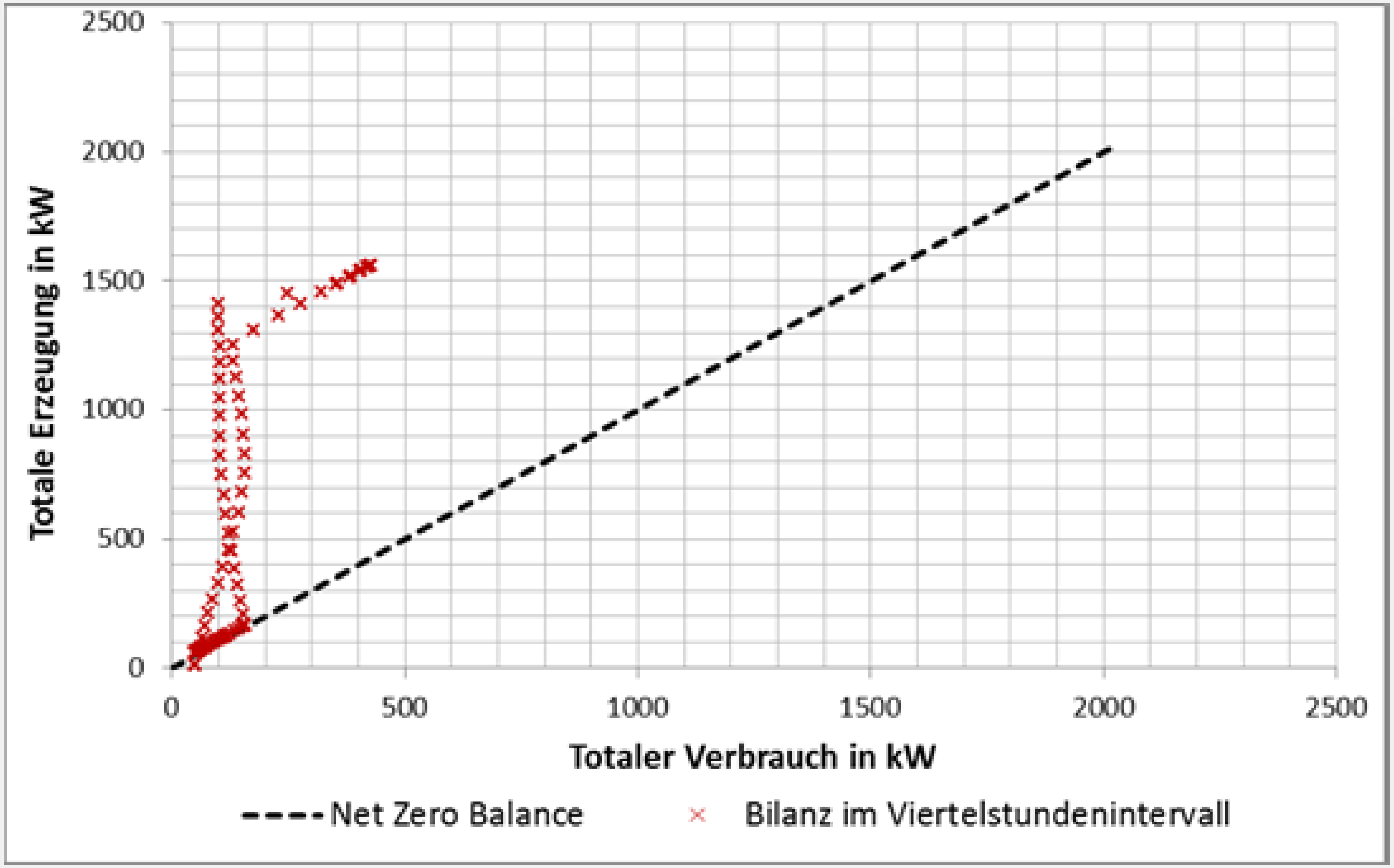
1.500
1.000
500
0
-500
-1.000
-1.500
Ein-/Ausspeicherleistung in kW
1.1
20.2
10.4
30.5
19.7
7.9
27.10
16.12
Einspeicherleistung (positiv)
Ausspeicherleistung (negativ)

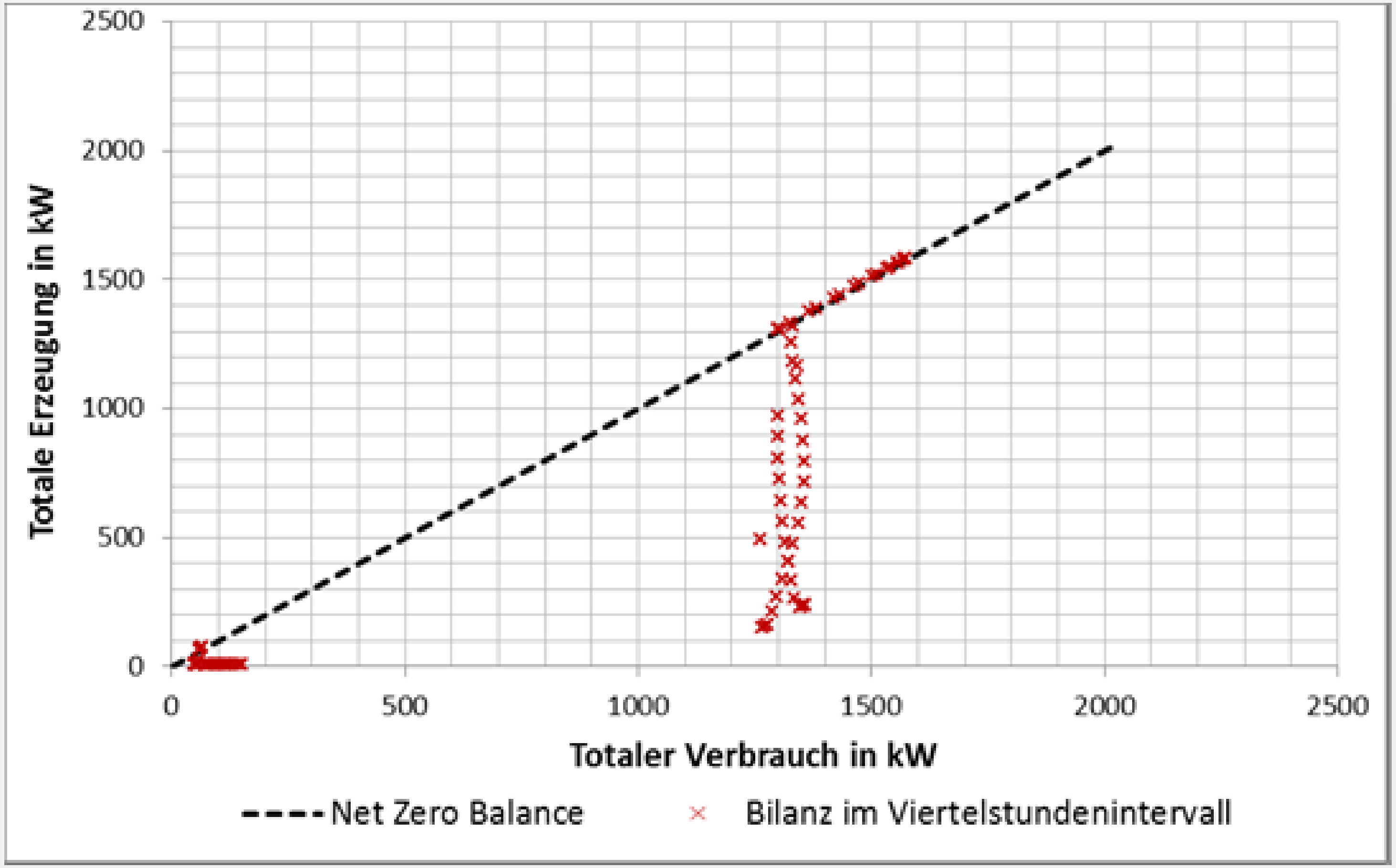

Energiespeicherleistung in kW
800
600
400
200
0
-200
-400
0
1000
2000
3000
4000
5000
6000
7000
8000
Volllaststunden
Einspeicherleistung (positiv)
Ausspeicherleistung (negativ)


1.200
1.000
800
600
400
200
0
genutzte Kapazität in kWh
0
1000
2000
3000
4000
5000
6000
7000
8000
Jahresnutzungsdauer
genutzte kWh
100%
90%
10%



Totale Erzeugung in kW
2500
2000
1500
1000
500
0
0
500
1000
1500
2000
2500
Totaler Verbrauch in kW
Net Zero Balance
Bilanz im Viertelstundenintervall


2500
2000
1500
1000
500
0
0
500
1000
1500
2000
2500
Totale Erzeugung in kW
Totaler Verbrauch in kW
Net Zero Balance
Bilanz im Viertelstundenintervall


2500
2000
1500
1000
500
0
0
500
1000
1500
2000
2500
Totale Erzeugung in kW
Totaler Verbrauch in kW
Net Zero Balance
Bilanz im Viertelstundenintervall

Summary of results

1) If the stationary energy storage is used for primary reserve (PR) provision for at least 10 weeks per year, its use is always cost-effective, with and without a cable-led agricultural machine.
2) Operating a cable-led agricultural machine in small and medium-size local grids usually requires a grid reinforcement.
3) The optimum size of the optimum stationary energy storage very sensibly depends on the income from secondary use such as PR, and on the weighted average cost of capital (wacc).

A closer look on economics_BAUM.py

Begründung der finanzmathematischen Formeln im oemof-Modul „economics" beziehungsweise „economics_BAUM"

M. Stöhr, B.A.U.M. Consult GmbH

<u>Hintergrund</u>

Die hier besprochenen finanzmathematischen Formeln finden sich in der Open Source Software oemof im Modul „economics" beziehungsweise „economics_BAUM". Dieses wurde aus jenem im Projekt „GridCon" im Rahmen des Forschungsprogramms „IKT für Elektromobilität" abgeleitet. Es wurde verwendet, um zu berechnen, bei welchem Stand der Nutzung der photovoltaischen Stromerzeugung im gleichen Ortsnetz eher eine Netzanschlusserweiterung oder die Installation eines stationären Energiespeichers oder eine Kombination von beidem zur Versorgung einer leitungsgeführten, elektrifizierten Landmaschine hoher Leistung geeignet ist. Die Formeln entsprechen denen, die üblicherweise bei der Berechnung der Wirtschaftlichkeit von Investitionen verwendet werden. Sie wurden im Rahmen des Projekts „GridCon" auf verschiedene Weise überprüft. Die folgenden Ausführungen geben die stringente mathematische Begründung wieder. Als Nebeneffekt wird dabei auch ein wenig beleuchtet, was der Begriff „wirtschaftlich" eigentlich bedeutet und welche Freiräume bestehen ihn zu deuten.

Is hard, but worth to go through it ☺

(1) $\frac{\mathrm{d}\,\mathrm{F(t)}}{\mathrm{dt}} = -\frac{1}{\tau} \cdot F(t)$

(2) $F(t) = F_0 e^{-\frac{t}{\tau}}$

(3) $\mathrm{F}_{\mathrm{i+1}} = \mathrm{F}_{\mathrm{i}} \frac{1}{1+\mathrm{z}}$

(4) $\frac{\tau}{1\ year} = \frac{1}{\ln(1+\mathrm{z})}$

(5) $\mathrm{F}_{\mathrm{i+1}} = \mathrm{F}_{\mathrm{i}}(1-w)$

(6) $W = \sum_{i=1}^{m} \mathrm{E}(\mathrm{t_i})\mathrm{e}^{\frac{\mathrm{t_i}}{\tau}} - \sum_{k=1}^{p} \mathrm{A}(\mathrm{t_k})\mathrm{e}^{\frac{\mathrm{t_k}}{\tau}}$

Is hard, but worth to go through it ☺

$$(7) \quad W = -I_0 + \sum_{i=1}^{n}(e_i - a_i)\left(\frac{1}{1+z}\right)^i$$

$$(8) \quad I_0 + \sum_{i=1}^{n} I_i \left(\frac{1}{1+z}\right)^i = A \sum_{i=1}^{n} \left(\frac{1}{1+z}\right)^i$$

$$(9) \quad \sum_{i=1}^{n} q^i = \frac{q(1-q^n)}{1-q}$$

$$(10) A = I_0 \frac{1-q}{q(1-q^n)} = I_0 z \frac{(1+z)^n}{(1+z)^n - 1}$$

That is what is implemented in economics.py

Is hard, but worth to go through it ☺

$$(11)\quad I_0 + \sum_{j=1}^{m-1} I_0 (1-cd)^{ju} q^{ju} = A \sum_{i=1}^{n} q^i$$

$$(12)\quad I_0 \left(1 + ((1-cd)q)^u \frac{1-((1-cd)q)^{(m-1)u}}{1-((1-cd)q)^u}\right) = A \frac{q(1-q^n)}{1-q}$$

$$(13)\quad A = I_0 \frac{1-q}{q(1-q^n)} \cdot \frac{1-((1-cd)q)^{mu}}{1-((1-cd)q)^u}$$

$$(14)\quad A = I_0 z \frac{(1+z)^n}{(1+z)^n - 1} \cdot \frac{1-(\frac{1-cd}{1+z})^n}{1-(\frac{1-cd}{1+z})^u}$$

That is what is implemented in economics_BAUM.py

Financial support

B.A.U.M.

IKT FÜR ELEKTROMOBILITÄT

Gefördert durch:

Bundesministerium für Wirtschaft und Technologie

aufgrund eines Beschlusses des Deutschen Bundestages