In[636]:=

Optimalni dimenzovani malych a strednich energetickych systemu s OZE

Priprava prostredi

In[637]:=
```
Quiet@Remove["Global`*"];
$HistoryLength = 0;
SetDirectory[NotebookDirectory[]];
```

Import a kontrola vstupnich dat

Import vstupnich dat

In[640]:=
```
nactivstupy[vyrspotsource_?StringQ, ekodatasource_?StringQ] :=
 Module[{nactenevyrspot = Import[vyrspotsource],
   nactenaekonomickadata = Import[ekodatasource]}, If[FailureQ[nactenevyrspot], Print[
    Style["Import vyrobnich a spotrebnich vstupu se nezdaril", Bold, Darker[Red]]];
   Interrupt[]];
  If[FailureQ[nactenaekonomickadata],
   Print[Style["Import ekonomickych vstupu se nezdaril", Bold, Darker[Red]]];
   Interrupt[]];
  Print[Style["Nacteni dat probehlo v poradku", Bold, Darker[Green]]];
  {Drop[nactenevyrspot[[1]], 1] // Transpose,
   nactenaekonomickadata, ToExpression[Import[ekodatasource, "Sheets"]]}
 ](*funkce pro import a kontrolu uspesnosti importu*)
```

In[641]:=
```
{nactenevyrspot, nactenaekonomickadata, komponentyenergsystemu} =
  nactivstupy["vsvstupy.xlsx", "cenkoef.xlsx"];
{vyrobafvelist, ppopelelist, ppopteplist, ppopchllist} = nactenevyrspot;
```

```
Nacteni dat probehlo v poradku
```

Kontrola rozmeru a hodnot vstupnich dat

In[643]:=
```
testrozmer[list_?ListQ, predprozmer_] := Module[
  {dimenze = (predprozmer == # & /@ (Length /@ list)), test}, test = TrueQ[And @@ dimenze];
  If[test, Print["Rozmer dat je v poradku"];
   Return[True], Print["Chybny rozmer dat na pozici: ", Position[dimenze, test]];
   Return[False]]] (*funkce pro testovani rozmeru nactenych dat*)
testvelikost[list_?ListQ] := Module[{testfve = And @@ (1 ≥ # ≥ 0 & /@ (list[[1]])),
   testspotreby = And @@@ (And /@ (Infinity > # ≥ 0 & /@ # & /@ (Drop[list, 1]))), test},
  test = testfve && And @@ testspotreby;
  If[test, Print["Hodnoty dat jsou validni"];
   Return[True], Print["Nepripustne hodnoty na pozici: ",
    Position[Join[{testfve}, testspotreby], test]];
   Return[False]]] (*funkce pro testovani hodnot jednotlivych vstupnich velicin*)
testdata[list_?ListQ, predprozmer_] :=
 If[And @@ {testvelikost[list], testrozmer[list, predprozmer]},
  Print[Style["Data nabyvaji validnich hodnot a maji spravny rozmer",
    Bold, Darker[Green]]], Print[
   Style["Opravte data na uvedenych pozicich a opakujte zadani", Bold, Darker[Red]]];
  Interrupt[]] (*validacni funkce hodnoty a rozmeru vstupnich velicin*)
```

In[646]:=
```
predprozmer = nactenevyrspot[[1]] // Length; (*predpokladany rozmer dat*)
testdata[nactenevyrspot, predprozmer]
```

```
Hodnoty dat jsou validni

Rozmer dat je v poradku

Data nabyvaji validnich hodnot a maji spravny rozmer
```

Globalni parametry a funkce

Jednotky

In[648]:=
```
s = 1.;
min = 60. * s;
hod = 60. * min;
den = 24. * hod;
rok = 365 * den;
kWh = 10^3 * 3600.;
MWh = kWh * 1000.;
kW = 1000;
```

Globalni parametry

In[656]:=
```
Δt = hod;
n0 = 8760;
normarok = n0 * Δt / rok;
```

In[659]:=
```
i0 = 1;
SOCakuele0 = 0.3;
Takutep0 = 85;

tau = 0;
r = 0.04 * normarok;
q = 1 + r;
velikostpenale = 10000;
```

In[668]:=
```
pocetscenarumc = 10; (*pocet scenaru simulace Monte Carlo*)
```

Globalni funkce

In[669]:=
```
a[q_, n_] := q^n (q - 1) / (q^n - 1);
mezefce[{mez_, reference_, variabilita_}] :=
  Max[{reference (1 - variabilita), 0}] ≤ mez ≤ Min[{reference (1 + variabilita), 1}];
mezvr := # ≥ 0 &
mezmr[{x_, mez_}] := x ≤ mez
mezmrmr[{smez_, x_, hmez_}] := smez ≤ x ≤ hmez
interpolujη[P_, ηlist_] := Module[{η, ηlistupraveny},
  ηlistupraveny =
   ηlist /. {procP_, etaTD_} :> {0.01 procP, etaTD / First[ηlist[[ ;; , 2]]]};
  η[p_] = Fit[ηlistupraveny, {1, p, p^2}, p];
  η[P]
 ]
vytvorvyraz[nazev_, suffix_] := ToExpression[ToString[nazev] <> ToString[suffix]];
mezekomponentyfce[{x_, smez_, hmez_, strmost_, penale_}] := {x >= smez, hmez > x}
kladne := If[# ≥ 0, #, 0] &;
zaporne := If[# ≤ 0., #, 0.] &;
```

Kontrola konfigurace ES

Volba konfigurace ES

In[679]:=
```
uvazovanakomponenta = <| (*volba konfigurace ES*)
    baterie → 1,
    menic → 1,
    absorpcnitepelnecerpadlo → 1,
    kogeneracnijednotkaele → 1,
    akumulatortepla → 1,
    fve → 1,
    tepelnecerpadlo → 1,
    sit → 0
    |>;
meze = {Pkogmez12, SOCakuelemez4, Takutepmez5};
```

Pomocne asociace

```
uvazovanakomponentainstvykon = <|
    baterie → Eakueleinst,
    menic → Pmenicinst,
    absorpcnitepelnecerpadlo → Patcinst,
    kogeneracnijednotkaele → Pkogeleinst,
    akumulatortepla → Vakutepinst,
    fve → Pfveinst,
    tepelnecerpadlo → Ptcinst,
    sit → False
    |>;

uvazovanakomponentanazev = <|
    baterie → "Akumulátor elektřiny",
    menic → "Měnič",
    absorpcnitepelnecerpadlo → "Absorpční tepelné čerpadlo",
    kogeneracnijednotkaele → "Kogenerační jednotka",
    akumulatortepla → "Akumulátor tepla",
    fve → "Fotovoltaické panely",
    tepelnecerpadlo → "Tepelné čerpadlo vzduch-voda",
    sit → "Distribuční soustava",
    rack → "RACK pro akumulátor elektřiny",
    rj → "Řídicí jednotka pro tepelné čerpadlo"
    |>;
```

Pripustne konfigurace ES

In[683]:=
```
subsety = Subsets[komponentyenergsystemu];(*vsechny mozne konfigurace ES*)
```

In[684]:=
```
(*vylouceni technicky nerealizovatelnych konfiguraci ES*)
vylouceno1 =
  Select[subsety, ! MemberQ[#, menic] && (MemberQ[ #, baterie] || MemberQ[ #, fve]) &];
vylouceno2 = Select[subsety,
    ! (MemberQ[#, akumulatortepla] || MemberQ[#, kogeneracnijednotkaele]) &&
      (MemberQ[ #, absorpcnitepelnecerpadlo]) &];
vylouceno3 = Select[subsety, ! (MemberQ[#, kogeneracnijednotkaele]) &&
      MemberQ[ #, akumulatortepla] &];
vylouceno4 = Select[subsety, MemberQ[#, menic] &&
      ! (MemberQ[ #, baterie] || MemberQ[ #, fve]) &];
vylouceno5 = Select[subsety, ! (MemberQ[#, kogeneracnijednotkaele] ||
         MemberQ[#, fve] || MemberQ[#, baterie]) && (MemberQ[ #, tepelnecerpadlo]) &];
vylouceno6 = Select[subsety, ! (MemberQ[#, kogeneracnijednotkaele] || MemberQ[#, fve]) &&
      (MemberQ[ #, baterie]) &];
prorozdeleni = Complement[subsety, Join[vylouceno1, vylouceno2, vylouceno3,
    vylouceno4, vylouceno5, vylouceno6(*,vylouceno7*)]];

(*pripustne konfigurace pro jednotlive spotrebni vetve ES*)
spotreletepchl =
  Complement[prorozdeleni, Select[subsety, ! MemberQ[#, absorpcnitepelnecerpadlo] &]];
spotreletep = Complement[prorozdeleni, Select[subsety,
    ! (MemberQ[#, akumulatortepla] || MemberQ[#, kogeneracnijednotkaele] ||
         MemberQ[#, tepelnecerpadlo]) || MemberQ[#, absorpcnitepelnecerpadlo] &]];
spotrele = Complement[prorozdeleni, Select[prorozdeleni,
    ! (MemberQ[#, fve] || MemberQ[#, baterie]) ||
      (MemberQ[#, kogeneracnijednotkaele] || MemberQ[#, tepelnecerpadlo] ||
        MemberQ[#, akumulatortepla] || MemberQ[#, absorpcnitepelnecerpadlo]) &]];
spotrtep = Complement[prorozdeleni, Select[prorozdeleni,
    (MemberQ[#, kogeneracnijednotkaele] || MemberQ[#, fve] ||
        MemberQ[#, baterie] || MemberQ[#, absorpcnitepelnecerpadlo]) ||
      ! (MemberQ[#, tepelnecerpadlo] && MemberQ[#, akumulatortepla]) &]];
```

Kontrola konfigurace ES

In[695]:=
```
konfiguraceES =
  Select[(uvazovanakomponenta /@ komponentyenergsystemu) * komponentyenergsystemu,
   ! NumberQ[#] &];(*vybrana konfigurace ES*)
vratstatus[konfiguraceES_] := Module[{status =
     MemberQ[#, konfiguraceES] & /@ {spotreletepchl, spotreletep, spotrele}},
   Piecewise[{{{1, 1, 1}, TrueQ[status[[1]]]}, {{1, 1, 0}, TrueQ[status[[2]]]},
     {{1, 0, 0}, TrueQ[status[[3]]]}}, {0, 0, 0}]];
spotrebaQ = AssociationThread[{ele, tep, chl}, vratstatus[konfiguraceES]];
(*asociacni funkce,
vraci hodnotu True/False pro prislusnou vetev spotreby (elektrina, teplo, chlad)*)
(DeleteDuplicates[Join[spotreletepchl, spotreletep, spotrele]] // Length) ==
  (Join[spotreletepchl, spotreletep, spotrele] // Length);
If[(spotrebaQ /@ {ele, tep, chl} // Total) == 0,
  Print[Style["Chybny stav. V dane konfiguraci neni mozne system provozovat.
      Upravte konfiguraci a opakujte zadani", Bold, Darker[Red]]];
  Interrupt[], Print[Style["Kontrola pripustnosti konfigurace ES probehla v poradku",
    Bold, Darker[Green]]]; (*kontrola pripustnosti konfigurace ES*)
```

```
Kontrola pripustnosti konfigurace ES probehla v poradku
```

Volba parametru optimalizace

Dimenzovani

Priprava ekonomickych vstupu

In[700]:=
```
vstupniekonomickadata =
  AssociationThread[komponentyenergsystemu, nactenaekonomickadata];
ekonomickeparametryseznam = {
   instalovano,
   instalovanojednotka,
   capex,
   capexjednotka,
   krokinv,
   krokinvjednotka,
   nprost,
   npstjednotka,
   npropr,
   npprjednotka
  };
upravenavstupnidata = AssociationThread[komponentyenergsystemu, Table[
    Drop[nactenaekonomickadata[[i]], 2], {i, 1, komponentyenergsystemu // Length}]];
upravenatransponovanavstupnidata =
  Table[Transpose[upravenavstupnidata[komponentyenergsystemu[[i]]]],
    {i, 1, Length[komponentyenergsystemu]}] /. "" → Nothing;
ekonomickeparametry = AssociationThread[komponentyenergsystemu,
   Table[AssociationThread[ekonomickeparametryseznam,
     upravenatransponovanavstupnidata[[i]]],
    {i, 1, Length[upravenatransponovanavstupnidata]}]];
```

Funkce investicnich vydaju jednotlivych komponent

```
shodnevykony[a_?NumericQ, krok_, mod_, ceny_, instaljednotka_] :=
 Module[{list, min, pozice},
  list = ceny * Ceiling /@ (a/# & /@ (instaljednotka * krok));
  min = Min[list];
  pozice = Position[list, min][[1, 1]];
  If[mod == "optimalizace", Piecewise[{{100 000 000, a < 0}}, min], {min, ceny[[pozice]],
    Ceiling[a/(krok * instaljednotka[[pozice]])], instaljednotka[[pozice]] * krok}]
 ]
```

In[708]:=
```
ruznevykony[a_?NumericQ, krok_, mod_, ceny_, instaljednotka_, meze_] :=
 Module[{list, min, pozice, vykon = Ceiling[a/krok] * krok, nejblizsivyssi,
   frobenius, frobeniuscast, frobeniuscelkem, cenypodil, cenyfrobenius,
   pocetkomponent, list2, min1, min2, pozice1, pozice2, kritfce, promenne,
   meze1, meze2, mezecelkem, kritfcenminimize, podminkynminimize},
  kritfce = (instaljednotka * krok).
    (vytvorvyraz[x, #] & /@ Range[Length[instaljednotka]]);
  promenne = (vytvorvyraz[x, #] & /@ Range[Length[instaljednotka]]);
  pocetkomponent = Ceiling /@ (a/# & /@ (instaljednotka * krok));
  meze2 = mezvr /@ promenne;
  If[ListQ[meze],
   meze1 = mezmr /@ ({promenne, meze} // Transpose);
   mezecelkem = Join[meze1, meze2];
   podminkynminimize = And @@ mezecelkem &&
     kritfce ≤ Max[pocetkomponent * instaljednotka * krok] && kritfce ≥ vykon;,
   mezecelkem = meze2;
   podminkynminimize = And @@ mezecelkem &&
     kritfce ≤ Max[pocetkomponent * instaljednotka * krok] && kritfce ≥ vykon;
  ];
  kritfcenminimize = ceny.(vytvorvyraz[x, #] & /@ Range[Length[instaljednotka]]);
  frobenius = Flatten[List @@@ Minimize[
       {kritfcenminimize, podminkynminimize}, promenne, Integers][[2, ;; , 2]]];
  min1 = Total[ceny * frobenius];
  list2 = ceny * pocetkomponent;
  min2 = Min[list2];
  pozice2 = Position[list2, min2][[1, 1]];
  If[mod == "optimalizace", Piecewise[{{min2, min1 ≥ min2}}, min1],
   {{min1, frobenius, frobenius * instaljednotka * krok}, {min2,
     Ceiling[a/(krok * instaljednotka[[pozice2]])], instaljednotka[[pozice2]] * krok}}]]
```

```
kogeneracnijednotkaelecapex[W_] := shodnevykony[W, kW, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[kogeneracnijednotkaele][capex]),
  IntegerPart /@ (ekonomickeparametry[kogeneracnijednotkaele][instalovano])]
tepelnecerpadlocapex[W_] := shodnevykony[W, kW, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[tepelnecerpadlo][capex]),
  IntegerPart /@ (ekonomickeparametry[tepelnecerpadlo][instalovano])]
bateriecapex[Ws_] := shodnevykony[Ws, kWh, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[baterie][capex]),
  IntegerPart /@ (ekonomickeparametry[baterie][instalovano])]
fvecapex[Wp_] := shodnevykony[Wp, 1, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[fve][capex]),
  IntegerPart /@ (ekonomickeparametry[fve][instalovano])]
akumulatorteplacapex[m3_] := shodnevykony[m3, 1, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[akumulatortepla][capex]),
  IntegerPart /@ (ekonomickeparametry[akumulatortepla][instalovano])]
meniccapex[W_] := shodnevykony[W, kW, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[menic][capex]),
  IntegerPart /@ (ekonomickeparametry[menic][instalovano])]
absorpcnitepelnecerpadlocapex[W_] := shodnevykony[W, kW, "optimalizace",
  IntegerPart /@ (ekonomickeparametry[absorpcnitepelnecerpadlo][capex]),
  IntegerPart /@ (ekonomickeparametry[absorpcnitepelnecerpadlo][instalovano])]

kogeneracnijednotkaeleprehled[W_] := shodnevykony[W, kW, "prehled",
  IntegerPart /@ (ekonomickeparametry[kogeneracnijednotkaele][capex]),
  IntegerPart /@ (ekonomickeparametry[kogeneracnijednotkaele][instalovano])]
tepelnecerpadloprehled[W_] := shodnevykony[W, kW, "prehled",
  IntegerPart /@ (ekonomickeparametry[tepelnecerpadlo][capex]),
  IntegerPart /@ (ekonomickeparametry[tepelnecerpadlo][instalovano])]
baterieprehled[Ws_] := shodnevykony[Ws, kWh, "prehled",
  IntegerPart /@ (ekonomickeparametry[baterie][capex]),
  IntegerPart /@ (ekonomickeparametry[baterie][instalovano])]
fveprehled[Wp_] := shodnevykony[Wp, 1, "prehled",
  IntegerPart /@ (ekonomickeparametry[fve][capex]),
  IntegerPart /@ (ekonomickeparametry[fve][instalovano])]
akumulatorteplaprehled[m3_] := shodnevykony[m3, 1, "prehled",
  IntegerPart /@ (ekonomickeparametry[akumulatortepla][capex]),
  IntegerPart /@ (ekonomickeparametry[akumulatortepla][instalovano])]
menicprehled[W_] := shodnevykony[W, kW, "prehled",
  IntegerPart /@ (ekonomickeparametry[menic][capex]),
  IntegerPart /@ (ekonomickeparametry[menic][instalovano])]
absorpcnitepelnecerpadloprehled[W_] := shodnevykony[W, kW, "prehled",
  IntegerPart /@ (ekonomickeparametry[absorpcnitepelnecerpadlo][capex]),
  IntegerPart /@ (ekonomickeparametry[absorpcnitepelnecerpadlo][instalovano])]
```

Technicko-ekonomicke parametry komponent

Kogeneracni jednotka

In[723]:=
```
Tzkogmax = 20 * 1/normarok;
Tpkogmax = 60 000 * hod;
minprovozhod = 0 * normarok;
maxprovozhod = 8760 hod * normarok;
maxpocetstartuzaden = 4;
cpal = 8.63;
ηkogteplist = {{100, 57.4}, {75, 53.5}, {50, 57.4}};
ηkogelelist = {{100, 34.3}, {75, 33.3}, {50, 28.7}};
spotrebazplist = {{100, 20.4}, {75, 15.9}, {50, 12.2}};
```

Akumulator elektriny

In[732]:=
```
Tzakuelemax = 12. * 1/normarok;
nakueleinstmax = 50.;
ηakuelenab = 0.90;
ηakuelevyb = 1;
SOCmax = 1;
SOCmin = 0;
```

Menic

In[738]:=
```
Tzmenmax = 15 * 1/normarok;
ηmen = 0.973;
```

Fotovoltaicky system

In[740]:=
```
Tzfvemax = 30. * 1/normarok;
```

Akumulator tepla

In[741]:=
```
Tzakutepmax = 20. * 1/normarok;
Takutepmax = 90.;
Takutepmin = 35.;
Takuteplimit = 0.7 * Takutepmax;
rho = 997.;
cv = 4186;
Takutepven = 22;
λ = 0.040;
p = 1100/2310;
lakutep = 0.1;
```

Tepelne cerpadlo vzduch-voda

In[751]:=
```
Tztcmax = 15. * 1/normarok;
Tptcmax = 70 000 * hod;
COPtc = 3;
```

Absorpcni tepelne cerpadlo

In[754]:=
```
Tzatcmax = 15 * 1/normarok;
Tpatcmax = 70 000 * hod;
COPatc = 1.1;
EERatc = 23/1.8;
```

Investicni vydaje

In[758]:=
```
Ninvbaterie = bateriecapex[Eakueleinst];
Ninvakumulatortepla = akumulatorteplacapex[Vakutepinst];
Ninvfve = fvecapex[Pfveinst];
Ninvkogeneracnijednotkaele = kogeneracnijednotkaelecapex[Pkogeleinst];
Ninvtepelnecerpadlo = tepelnecerpadlocapex[Ptcinst];
Ninvabsorpcnitepelnecerpadlo = absorpcnitepelnecerpadlocapex[Patcinst];
Ninvmenic = meniccapex[Pmenicinst];
Ninvcelkem = vytvorvyraz[Ninv, #] & /@ komponentyenergsystemu *
    uvazovanakomponenta /@ komponentyenergsystemu;
```

In[766]:=
```
Ninvprehled = vytvorvyraz[#, prehled[uvazovanakomponentainstvykon[#]]] & /@
     komponentyenergsystemu * uvazovanakomponenta /@ komponentyenergsystemu;
```

Uvazovane promenne sizing

In[767]:=
```
promenneprooptim =
  (uvazovanakomponenta * uvazovanakomponentainstvykon) /@ komponentyenergsystemu;
uvazovanepromennesizing = Select[promenneprooptim, ! NumberQ[#] &];
uvazovanepromennesizingtf = Piecewise[{{0, NumericQ[#]}}, 1] & /@ promenneprooptim;
uvazovanepromennesizingmeze = Flatten[{uvazovanepromennesizing, meze}];
```

Podminky dimenzovani

Omezujici podminky komponent

In[771]:=
```
podminkysizingkomponenty = {{Eakueleinst, 2.4 kWh, 100. * kWh, 0.00002, penkonst},
    {Pmenicinst, 30. * kW, 35. * kW, 2, penkonst},
    {Patcinst, 60. * kW, 61. * kW, 50, penkonst},
    {Pkogeleinst, 29. * kW, 40. * kW, 2, penkonst},
    {Ptcinst, 8. * kW, 100. * kW, 2, penkonst},
    {Pfveinst, 0.315 * kW, 15. * kW, 2, penkonst},
    {Vakutepinst, 1., 20., 2, penkonst}};
podminkysizingcelkem =
  Select[Flatten[((mezekomponentyfce /@ podminkysizingkomponenty) *
      uvazovanepromennesizingtf)], ! NumberQ[#] &];
podminkymeze = {{0.5, Pkogmez12, 1}, {0.34, SOCakuelemez4, 0.46},
    {0.68, Takutepmez5, 0.92}};
```

Parametry optimalizacnich metod

In[774]:=
```
precisiongoal = Automatic;
accuracygoal = Automatic;
workingprecision = Automatic;
maxiterations = 50;
c1 = c2 = 2.05;
w = 0.729;
method =
  {"SimulatedAnnealing", "SearchPoints" → 30, "RandomSeed" → 0,
   "Tolerance" -> 0.001, "LevelIterations" → 5, "PostProcess" → False}
   (*{"DifferentialEvolution","SearchPoints"→30,"CrossProbability"→0.5,
  "RandomSeed"→0,"ScalingFactor"→0.6,"Tolerance"→  0.001,"PostProcess"→False},
 {"MPSO","SearchPoints"→30,"c1"→c1,"c2"→c2,"w"→w}*);
```

Provozni naklady

In[781]:=
```
nprostcelkem = normarok * ekonomickeparametry[#][nprost] & /@ komponentyenergsystemu *
    uvazovanakomponenta /@ komponentyenergsystemu // Flatten;
nproprcelkem = ekonomickeparametry[#][npropr] & /@ komponentyenergsystemu *
    uvazovanakomponenta /@ komponentyenergsystemu // Flatten;
```

Priprava dat

Statisticke testovani

In[783]:=
```
popisnestat[list_?ListQ] := Module[{}, {
   Min[list],
   Max[list],
   Mean[list],
   StandardDeviation[list],
   Median[list],
   Skewness[list],
   Kurtosis[list],
   {"Histogram", Histogram[list]},
   {"Boxplot", BoxWhiskerChart[list]},
   {"************************"}
  }] (*funkce vraci zakladni statistiky vstupniho datasetu*)
```

In[784]:=
```
testovanirozdeleni[list_?ListQ] := Module[{ℋ, 𝒟, dosad, nezamitase, listmin, listmax},
  dosad = FindDistributionParameters[list, NormalDistribution[μ, σ]];
  ℋ =
   DistributionFitTest[list, NormalDistribution[μ, σ] /. dosad, "HypothesisTestData"];
  nezamitase = ℋ["ShapiroWilk"] ≥ 0.05;
  listmin = Min[list];
  listmax = Max[list];
  If[nezamitase && BooleanQ[nezamitase], ℋ["FittedDistribution"], {listmin, listmax}]]
  (*pokud se nulova hypoteza nezamita, funkce vraci prislusne normalni rozdelneni,
 v opacnem pripade funkce vraci parametry pro rovnomerne rozdeleni*)
```

In[785]:=
```
dPcore[L_, p : {q___, _}] := Inner[L[[# ;; #2]] &, {0, q} + 1, p, Head@L]
dPcore[L_, p_, All] := dPcore[L, p] ~Append~ Drop[L, Last@p]
dPcore[L_, p_, n__] := dPcore[L, p] ~Join~ Partition[L ~Drop~ Last@p, n]
dynamicPartition[L_, p : {__Integer}, x___] :=
 dPcore[L, Accumulate@p, x] /; ! Negative@Min@p && Length@L ≥
    Tr@p (*funkce pro dynamicke deleni listu dle preddefinovanych kriterii*)
```

Převzato z: Mathematica Stack Exchange | Matehematica and Wolfram language, “Partitioning with varying partition size,” 2017. [online]. Dostupné z: https://mathematica.stackexchange.com/questions/7511/partitioning-with-varying-partition-size [cit. 29. 12. 2019].

Simulace Monte Carlo

In[789]:=
```
dnyvmesici = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
(*pocet dnu v mesici pro neprestuny rok; kriteria pro deleni listu*)
{vyrobafvelistden, ppopelelistden, ppopteplistden, ppopchllistden} =
  Partition[#, IntegerPart[den/Δt]] & /@
   {vyrobafvelist, ppopelelist, ppopteplist, ppopchllist};
proMC = {vyrobafvelistdenpomesici, ppopelelistdenpomesici, ppopteplistdenpomesici,
    ppopchllistdenpomesici} = dynamicPartition[#, dnyvmesici] & /@
    {vyrobafvelistden, ppopelelistden, ppopteplistden, ppopchllistden};
```

In[792]:=
```
vratrozdeleni[list_?ListQ] :=
  Table[(testovanirozdeleni[#]) & /@ (((Transpose /@ list))[[i]]), {i, 1, 12}];
(*funkce vraci rozdeleni pro kazdou hodinu dne v prislusnem mesici*)
montecarlo[list_?ListQ, pocetmc_] := Module[{listrozdeleni = vratrozdeleni[list]},
   Table[kladne /@ Flatten[Table[(If[ListQ[#], Table[RandomReal[#], dnyvmesici[[i]]],
            RandomVariate[#, dnyvmesici[[i]]]] & /@
          (listrozdeleni[[i]])) // Transpose, {i, 1, 12}]], pocetmc]];
vratlisty[list : {vyrobafve_, popele_, popchl_, poptep_}, delka_] := Module[{},
  montecarlo[#, delka] & /@ list
 ]
```

In[795]:=
```
{vyrobafvelistmc, ppopelelistmc, ppopteplistmc, ppopchllistmc} =
  vratlisty[proMC, pocetscenarumc]; (*generovani scenaru simulace Monte Carlo*)
```

Algoritmus dimenzovani komponent ES

```
fcedimenzovani[{EakueleinstOpt_?NumericQ,
    PmenicinstOpt_?NumericQ, PatcinstOpt_?NumericQ, PkogeleinstOpt_?NumericQ,
    PtcinstOpt_?NumericQ, PfveinstOpt_?NumericQ, VakutepinstOpt_?NumericQ},
   {Pkogmez12_?NumericQ, SOCakuelemez4_?NumericQ, Takutepmez5_?NumericQ},
   uvazovanepromennesizingtf_,
   {t0_, SOCakuele0_, Takutep0_},
   {tau_, qOpt_, Δt_, pocetperiod_, velikostpenale_},
   {SOCmax_, SOCmin_, ηakuelenab_, ηakuelevyb_, TzakueleOpt_},
   {ηmen_, TzmenOpt_},
   {EERatc_, COPatc_, {TpatcOpt_, TzatcOpt_}},
   {cpal_, minprovozhod_, maxprovozhod_,
    maxpocetstartuzaden_, {TpkogOpt_, TzkogOpt_}},
   {COPtc_, {TptcOpt_, TztcOpt_}},
   {TzfveOpt_},
   {rho_, cv_, λref_, pref_, lref_, Takutepmin_,
    Takutepmax_, Takutepven_, Takuteplimit_, TzakutepOpt_},
   {Ninv_, npst_, nppr_},
   ekoprehled_,
   mod_?StringQ,
```

```
iter_] := Module[{start,
 res = ConstantArray[{ConstantArray[0, 3],
    ConstantArray[0, 10], ConstantArray[0, 10]}, pocetperiod + 1],
 kogen,
 tc,
 atc,
 Pkogelemax,
 Pkogelemin,
 kritfcesizing,
 penalizace,
 SOCakuelemax,
 SOCakuelemin,
 Patcchlmax,
 Ptctepmax,
 Pmenicmax,
 Pfvemax,
 ηkogtep,
 ηkogele,
 konstantaTHT,
 makutep,
 mcAku,
 ηakuelenabfce,
 ηakuelevybfce,
 ηmenicfce,
 Eakutepztrata,
 penalizacestart,
 vypzapkogen,
 spotrebazpkog,
 kladne,
 ηfve,
 zapnuto,
 provoznicaskogen,
 provoznicastc,
 provoznicasatc,
 Npprcelkem,
 kritfcepenalizace,
 odpisy = 0,
 Npstcelkem,
 Ecelkem,
 Nppalcelkem,
 Ninvzvlast,
 Ninvcelkem,
 anuity,
 Pcelkem,
 penzlizacestartcelkem,
 spotrebaeleQ = spotrebaQ[ele],
 spotrebatepQ = spotrebaQ[tep],
 spotrebachlQ = spotrebaQ[chl],
 normarok = pocetperiod * Δt / rok,
```

```
 kogvprovozu = uvazovanakomponenta[kogeneracnijednotkaele],
 sitvprovozu = uvazovanakomponenta[sit],
 spotrebazpatc,
 pocstartuzaden,
 Npalkog,
 Npalatc,
 penalizacestav,
 vratR,
 R
},
vratR[λ_, p_, V_, l_] := Module[{reseni, r1, r2, R1, R2},
  If[V == 0, 1,
   reseni = Solve[Pi * d^2/4 * h == Ceiling[V]
       && h * p == d, {h, d}, Reals][[1]];
   r1 = d/2;
   r2 = d/2 + l;
   R1 = Log[r2 / r1]/(2 * Pi * λ * h) /. reseni;
   R2 = 1/(Pi * r1^2 * λ) /. reseni;
   R1 R2/(R1 + 2 * R2)]
 ];
R = vratR[λref, pref, VakutepinstOpt, lref];
kladne := If[# ≥ 0, #, 0] &;
SOCakuelemax = SOCmax * EakueleinstOpt;
SOCakuelemin = SOCmin * SOCakuelemax;
Patcchlmax = PatcinstOpt;
Ptctepmax = PtcinstOpt;
Pmenicmax = PmenicinstOpt;
Pfvemax = PfveinstOpt;
Pkogelemax = Piecewise[{{1, PkogeleinstOpt < 1}}, PkogeleinstOpt];
makutep = rho * VakutepinstOpt;
mcAku = Piecewise[{{1, makutep < 1}}, makutep * cv];
Pkogelemin = Pkogmez12 * Pkogelemax;
ηakuelenabfce[Pakuele_] := ηakuelenab;
ηakuelevybfce[Pakuele_] := ηakuelevyb;
ηmenicfce[Pakufve_] := ηmen;
Eakutepztrata[t_, Takutep_] :=
 Module[{Rinv = 1/R, TvenkovninaR},
  TvenkovninaR = Takutepven * Rinv;
  Piecewise[{{0, makutep == 0}}, (Takutep * Rinv - TvenkovninaR) * Δt]];
```

```
ηkogtep[Pkogele_] :=
 Module[{Pkogelerel = Pkogele/Pkogelemax}, 1.544 - 1.630 Pkogelerel + 1.0871 Pkogelerel^2];
ηkogele[Pkogele_] := Module[{Pkogelerel = Pkogele/Pkogelemax},
  0.254 + 1.586 Pkogelerel - 0.840 Pkogelerel^2];
ηfve[t_] := -1.373*^-10 * t + 1 ;
konstantaTHT[Pkogeleokam_] :=
 Piecewise[{{ηkogtep[Pkogeleokam]/ηkogele[Pkogeleokam], Pkogelemax > 0}}, 1] ;
Pcelkem = (res /. {bod : {t_, SOCakuele_, Takutep_},
       {Ekogele_, Edoakuele_, Edoakutep_, Etepbilance_, Eelebilance_, Etctep_,
        Eatcchl_, Efve_, penaledt_, stavdt_}, {Pkogele_, Pdoakuele_, Pdoakutep_,
        Ptepbilance_, Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :>
      {Pdoakuele, 0, Patcchl, Pkogele, Ptctep, Pfve, Pdoakutep}) // Transpose;
Ecelkem = Δt * Total /@ Pcelkem;
kogen = Pcelkem[[4]];
tc = Pcelkem[[5]];
atc = Pcelkem[[3]];
spotrebazpkog[Pkogele_] :=
 Module[{Pkogelerel = Pkogele * kW/Pkogelemax, spalhod = 20.4/(70 kW) * Pkogelemax},
  Piecewise[{{spalhod * Δt/hod * (0.353 + 0.333 Pkogelerel + 0.314` Pkogelerel^2),
     Pkogelerel > 0}}, 0]];
spotrebazpatc[Patcchl_] := Module[{Patcchlrel = Patcchl * kW/Patcchlmax,
   spalhod = 2.1/(23 kW) * Patcchlmax}, spalhod * Δt/hod * Patcchlrel];
Npalkog = Total[spotrebazpkog /@ kogen] * cpal;
Npalatc = 0;
Nppalcelkem = Npalkog + Npalatc;
Npprcelkem = (nppr * uvazovanepromennesizingtf).Ecelkem + Nppalcelkem;
penalizace =
 Flatten[res /. {bod : {t_, SOCakuele_, Takutep_}, {Ekogele_, Edoakuele_, Edoakutep_,
      Etepbilance_, Eelebilance_, Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
     {Pkogele_, Pdoakuele_, Pdoakutep_, Ptepbilance_, Pelebilance_,
      Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {penale}];
provoznicaskogen = Total[zapnuto /@ kogen];
provoznicastc = Total[zapnuto /@ tc];
provoznicasatc = Total[zapnuto /@ atc];
pocstartuzaden =
 Length /@ vypzapkogen /@ Partition[(kogen // Flatten), UpTo[den / Δt]];
penzlizacestartcelkem = Total[penalizacestart /@ pocstartuzaden];
penalizacestav = Total[penalizace];
kritfcepenalizace = penzlizacestartcelkem * kogvprovozu + penalizacestav;
```

```
anuity = (a[qOpt, #] & /@ {TzakueleOpt, TzmenOpt,
    Piecewise[{{3, provoznicasatc == 0}}, Min[TzatcOpt, TpatcOpt/(provoznicasatc * Δt)]],
    Piecewise[{{3, provoznicaskogen == 0}, {Min[TzkogOpt, TpkogOpt/(provoznicaskogen * Δt)],
       (maxprovozhod > provoznicaskogen * Δt > minprovozhod)}}, 3],
    Piecewise[{{3, provoznicastc == 0}}, Min[TztcOpt, TptcOpt/(provoznicastc * Δt)]],
    TzfveOpt, TzakutepOpt});
Npstcelkem = (npst * uvazovanepromennesizingtf).
   (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1});
Ninvcelkem = (Ninv * uvazovanepromennesizingtf).anuity;
kritfcesizing =
 kritfcepenalizace + 1/(1 - tau) (Ninvcelkem - odpisy) + odpisy + Npstcelkem + Npprcelkem;
If[mod == "optimalizace",
 kritfcesizing,
 If[mod == "vratvysledky",
  res,
  If[mod == "debug",
   {{pocstartuzaden, penalizacestart /@ pocstartuzaden}, Total /@ Pcelkem, Ecelkem,
    anuity, {Ninvcelkem, Total[Ninv], Ninv}, {provoznicaskogen, provoznicastc,
     provoznicasatc}, {Npstcelkem, ((npst * uvazovanepromennesizingtf) *
       (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1}))}, {Npprcelkem,
     Nppalcelkem, {Total /@ Partition[spotrebazpkog /@ kogen, UpTo[den / Δt]],
      Total[spotrebazpkog /@ kogen], Npalkog, 0, 0, 0}, Npprcelkem - Nppalcelkem,
     penzlizacestartcelkem, (nppr * uvazovanepromennesizingtf)},
    {TztcOpt, provoznicastc, TptcOpt/(provoznicastc * Δt)},
    {TzkogOpt, provoznicaskogen, TpkogOpt/(provoznicaskogen * Δt)},
    {TzatcOpt, provoznicasatc, TpatcOpt/(provoznicasatc * Δt)}, kritfcesizing,
    ekoprehled, penalizacestav, penzlizacestartcelkem * kogvprovozu},
   If[mod == "export",
    {{pocstartuzaden, penalizacestart /@ pocstartuzaden}, Total /@ Pcelkem, Ecelkem,
     anuity, {Ninvcelkem, Total[Ninv], Ninv}, {provoznicaskogen, provoznicastc,
```

```
        provoznicasatc}, {Npstcelkem, ((npst * uvazovanepromennesizingtf) *
           (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1}))}, {Npprcelkem,
        Nppalcelkem, {Total /@ Partition[spotrebazpkog /@ kogen, UpTo[den / Δt]],
         Total[spotrebazpkog /@ kogen], Npalkog, 0, 0, 0}, Npprcelkem - Nppalcelkem,
        penzlizacestartcelkem, (nppr * uvazovanepromennesizingtf * Total /@ Pcelkem/kW)},
       {TztcOpt, provoznicastc, TptcOpt/(provoznicastc * Δt)},
       {TzkogOpt, provoznicaskogen, TpkogOpt/(provoznicaskogen * Δt)},
       {TzatcOpt, provoznicasatc, TpatcOpt/(provoznicasatc * Δt)}, kritfcesizing,
       ekoprehled, penalizacestav, penzlizacestartcelkem * kogvprovozu},
      Print["Funkce je volana nespravne, parametr mod akceptuje pouze
         string ve formatu: a)'optimalizace', b)'vratvysledky'"];
      Interrupt[]]]
  ];

fcedimenzovaniLP[{EakueleinstOpt_?NumericQ,
   PmenicinstOpt_?NumericQ, PatcinstOpt_?NumericQ, PkogeleinstOpt_?NumericQ,
   PtcinstOpt_?NumericQ, PfveinstOpt_?NumericQ, VakutepinstOpt_?NumericQ},
  {Pkogmez12_?NumericQ, SOCakuelemez4_?NumericQ, Takutepmez5_?NumericQ},
  uvazovanepromennesizingtf_,
  {t0_, SOCakuele0_, Takutep0_},
  {tau_, qOpt_, Δt_, pocetperiod_, velikostpenale_},
  {SOCmax_, SOCmin_, ηakuelenab_, ηakuelevyb_, TzakueleOpt_},
  {ηmen_, TzmenOpt_},
  {EERatc_, COPatc_, {TpatcOpt_, TzatcOpt_}},
  {cpal_, minprovozhod_, maxprovozhod_,
   maxpocetstartuzaden_, {TpkogOpt_, TzkogOpt_}},
  {COPtc_, {TptcOpt_, TztcOpt_}},
  {TzfveOpt_},
  {rho_, cv_, λref_, pref_, lref_, Takutepmin_,
   Takutepmax_, Takutepven_, Takuteplimit_, TzakutepOpt_},
  {Ninv_, npst_, nppr_},
  ekoprehled_,
  mod_,
  iter_] := Module[{start,
   res = ConstantArray[{ConstantArray[0, 3],
      ConstantArray[0, 10], ConstantArray[0, 10]}, pocetperiod + 1],
   kogen,
   tc,
   atc,
   Pkogelemax,
```

```
Pkogelemin,
kritfcesizing,
penalizace,
SOCakuelemax,
SOCakuelemin,
Patcchlmax,
Ptctepmax,
Pmenicmax,
Pfvemax,
ηkogtep,
ηkogele,
konstantaTHT,
makutep,
mcAku,
ηakuelenabfce,
ηakuelevybfce,
ηmenicfce,
Eakutepztrata,
penalizacestart,
vypzapkogen,
spotrebazpkog,
kladne,
ηfve,
zapnuto,
provoznicaskogen,
provoznicastc,
provoznicasatc,
Npprcelkem,
kritfcepenalizace,
odpisy = 0,
Npstcelkem,
Ecelkem,
Nppalcelkem,
Ninvzvlast,
Ninvcelkem,
anuity,
Pcelkem,
provozcelkem,
penzlizacestartcelkem,
spotrebaeleQ = spotrebaQ[ele],
spotrebatepQ = spotrebaQ[tep],
spotrebachlQ = spotrebaQ[chl],
normarok = pocetperiod * Δt / rok,
kogvprovozu = uvazovanakomponenta[kogeneracnijednotkaele],
sitvprovozu = uvazovanakomponenta[sit],
csitnakup = 5.45,
spalkogref,
spalatcref,
pocstartuzaden,
```

```
 penalizacestav,
 Npalkog,
 NpalatcvratR,
 R,
 Npalatc,
 vratR,
 ηkogtepnaηkogele
},
vratR[λ_, p_, V_, l_] := Module[{reseni, r1, r2, R1, R2},
  If[V == 0, 1,
   reseni = Solve[Pi * d^2/4 * h == Ceiling[V]
       && h * p == d, {h, d}, Reals][[1]];
   r1 = d/2;
   r2 = d/2 + l;
   R1 = Log[r2 / r1]/(2 * Pi * λ * h) /. reseni;
   R2 = 1/(Pi * r1^2 * λ) /. reseni;
   R1 R2/(R1 + 2 * R2)]
 ];
R = vratR[λref, pref, VakutepinstOpt, lref];
SOCakuelemax = SOCmax * EakueleinstOpt;
SOCakuelemin = SOCmin * SOCakuelemax;
Patcchlmax = PatcinstOpt;
Ptctepmax = PtcinstOpt;
Pmenicmax = PmenicinstOpt;
Pfvemax = PfveinstOpt;
Pkogelemax = Piecewise[{{1, PkogeleinstOpt < 1}}, PkogeleinstOpt];
makutep = rho * VakutepinstOpt;
mcAku = Piecewise[{{1, makutep < 1}}, makutep * cv];
Pkogelemin = Pkogmez12 * Pkogelemax;
spalkogref = 20.4/(70 kW) * Pkogelemax;
spalatcref = 2.1/(23 kW) * Patcchlmax;
ηakuelenabfce[Pakuele_] := ηakuelenab;
ηakuelevybfce[Pakuele_] := ηakuelevyb;
ηmenicfce[Pakufve_] := ηmen;
Eakutepztrata[t_, Takutep_] :=
 Module[{Rinv = 1/R, TvenkovninaR}, TvenkovninaR = Takutepven * Rinv;
```

```
  Piecewise[{{0, makutep == 0}}, (Takutep * Rinv - TvenkovninaR) * Δt]];
ηkogtep[Pkogele_] :=
 Module[{Pkogelerel = Pkogele/Pkogelemax}, 1.544 - 1.630 Pkogelerel + 1.0871 Pkogelerel^2];
ηkogele[Pkogele_] := Module[{Pkogelerel = Pkogele/Pkogelemax},
  0.254 + 1.586 Pkogelerel - 0.840 Pkogelerel^2];
ηfve[t_] := -1.374*^-10 * t + 1 ;
ηkogtepnaηkogele[Pkogeleokam_] :=
 Piecewise[{{ηkogtep[Pkogeleokam] / ηkogele[Pkogeleokam], Pkogelemax > 0}}, 1] ;
konstantaTHT = ηkogtepnaηkogele[Pkogelemin];
Pcelkem =
 (res /. {bod : {t_, SOCakuele_, Takutep_}, energie : {E1_, E2_, E3_, E4_, E5_, E6_,
         E7_, E8_, E9_, E10_, E11_, E12_, E13_, E14_, E15_, E16_, E17_, E18_,
         E19_, E20_, E21_}, {V1_, V2_, V3_, V4_, V5_, V6_, V7_, V8_, V9_, V10_,
        V11_, V12_, V13_, V14_, V15_, V16_, V17_, V18_, V19_, V20_, V21_}} :>
      {V13 - V14, 0, V8, V18, V5, V20, V15 - V16}) // Transpose;
Ecelkem = Δt * Total /@ Pcelkem;
kogen = Pcelkem[[4]];
tc = Pcelkem[[5]];
atc = Pcelkem[[3]];
provozcelkem = Flatten[res /.
   {bod : {t_, SOCakuele_, Takutep_}, energie : {E1_, E2_, E3_, E4_, E5_, E6_, E7_, E8_,
       E9_, E10_, E11_, E12_, E13_, E14_, E15_, E16_, E17_, E18_, E19_, E20_, E21_},
     {V1_, V2_, V3_, V4_, V5_, V6_, V7_, V8_, V9_, V10_, V11_, V12_, V13_,
      V14_, V15_, V16_, V17_, V18_, V19_, V20_, V21_}} :> {V1}];
Npalkog = spalkogref * Δt/hod * Total[kogen * kW]/Pkogelemax * cpal;
Npalatc = 0;
Nppalcelkem = Npalkog + Npalatc;
Npprcelkem = Total[provozcelkem];
provoznicaskogen = Total[zapnuto /@ kogen];
provoznicastc = Total[zapnuto /@ tc];
provoznicasatc = Total[zapnuto /@ atc];
pocstartuzaden =
 Length /@ vypzapkogen /@ Partition[(kogen // Flatten), UpTo[den / Δt]];
penzlizacestartcelkem = Total[penalizacestart /@ pocstartuzaden];
penalizacestav =
 Total[Flatten[res /. {bod : {t_, SOCakuele_, Takutep_}, energie : {E1_, E2_, E3_, E4_,
         E5_, E6_, E7_, E8_, E9_, E10_, E11_, E12_, E13_, E14_, E15_, E16_, E17_, E18_,
         E19_, E20_, E21_}, {V1_, V2_, V3_, V4_, V5_, V6_, V7_, V8_, V9_, V10_, V11_,
        V12_, V13_, V14_, V15_, V16_, V17_, V18_, V19_, V20_, V21_}} :> {V21}]];
kritfcepenalizace = penzlizacestartcelkem * kogvprovozu + penalizacestav;
anuity = (a[qOpt, #] & /@ {TzakueleOpt, TzmenOpt,
    Piecewise[{{0.1, provoznicasatc == 0}}, Min[TzatcOpt, TpatcOpt/(provoznicasatc * Δt)]]},
```

```
    Piecewise[{{{Min[TzkogOpt, TpkogOpt/(provoznicaskogen * Δt)],
        (maxprovozhod > provoznicaskogen * Δt > minprovozhod)}}, 3],
    Piecewise[{{3, provoznicastc == 0}}, Min[TztcOpt, TptcOpt/(provoznicastc * Δt)]],
    TzfveOpt, TzakutepOpt}];
Npstcelkem = (npst * uvazovanepromennesizingtf).
   (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1});
Ninvcelkem = (Ninv * uvazovanepromennesizingtf).anuity;
kritfcesizing =
 kritfcepenalizace + 1/(1 - tau) (Ninvcelkem - odpisy) + odpisy + Npstcelkem + Npprcelkem;
If[mod == "optimalizace",
 kritfcesizing,
 If[mod == "vratvysledky",
  res,
  If[mod == "debug",
   {{pocstartuzaden, penalizacestart /@ pocstartuzaden}, Total /@ Pcelkem, Ecelkem,
    anuity, {Ninvcelkem, Total[Ninv], Ninv}, {provoznicaskogen, provoznicastc,
     provoznicasatc}, {Npstcelkem, ((npst * uvazovanepromennesizingtf) *
       (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1}))}, {Npprcelkem,
     Nppalcelkem, {spalkogref * Δt/hod * Total /@ Partition[kogen * kW, UpTo[den / Δt]]/Pkogelemax,
      spalkogref * Δt/hod * Total[kogen * kW]/Pkogelemax,
      spalkogref * Δt/hod * Total[kogen * kW]/Pkogelemax * cpal, 0, 0 , 0}, Npprcelkem - Nppalcelkem,
     penzlizacestartcelkem, (nppr * uvazovanepromennesizingtf)},
    {TztcOpt, provoznicastc, TptcOpt/(provoznicastc * Δt)},
    {TzkogOpt, provoznicaskogen, TpkogOpt/(provoznicaskogen * Δt)},
    {TzatcOpt, provoznicasatc, TpatcOpt/(provoznicasatc * Δt)}, kritfcesizing,
    ekoprehled, penalizacestav, penzlizacestartcelkem * kogvprovozu},
   If[mod == "export",
    {{pocstartuzaden, penalizacestart /@ pocstartuzaden}, Total /@ Pcelkem, Ecelkem,
     anuity, {Ninvcelkem, Total[Ninv], Ninv}, {provoznicaskogen, provoznicastc,
```

```
provoznicasatc}, {Npstcelkem, ((npst * uvazovanepromennesizingtf) *
    (Ninv * {0, 0, 1, 0, 1, 0, 0} + {1, 1, 0, 1, 0, PfveinstOpt/kW, 1})), {Npprcelkem,
  Nppalcelkem, {spalkogref * Δt/hod * Total /@ Partition[kogen * kW, UpTo[den / Δt]]/Pkogelemax,
   spalkogref * Δt/hod * Total[kogen * kW]/Pkogelemax, Npalkog, 0, 0},
  Npprcelkem - Nppalcelkem, penzlizacestartcelkem,
  (nppr * uvazovanepromennesizingtf * Total /@ Pcelkem/kW)},
 {TztcOpt, provoznicastc, TptcOpt/(provoznicastc * Δt)},
 {TzkogOpt, provoznicaskogen, TpkogOpt/(provoznicaskogen * Δt)},
 {TzatcOpt, provoznicasatc, TpatcOpt/(provoznicasatc * Δt)}, kritfcesizing,
 ekoprehled, penalizacestav, penzlizacestartcelkem * kogvprovozu},

Print["Funkce je volana nespravne, parametr mod akceptuje pouze
   string ve formatu: a)'optimalizace', b)'vratvysledky'"];
Interrupt[]]]]

]];
```

Modifikovany algoritmus optimalizace hejnem castic

In[798]:=
```
mpso[SwarmSizeini_, iterations_, SwarmDimini_, FitnessLBini_, FitnessUBini_,
  c1_, c2_, w_, Fitnessini_, condsizing_, condbound_, promenne_] := Module[
 {UpdateSwarmPosition, InitializaSwarm, gBest, PBest, SwarmPos, SwarmVel, SwarmPosf,
  iter, rand1, rand2, i, j, vratplatne, v1, v2, v3, v4, print, wCF, CF, C1, C2, res},
 InitializaSwarm[SwarmSize_, SwarmDim_, FitnessLB_, FitnessUB_, Fitness_] :=
  Module[{gBestt, SwarmPosft, idx, gBest1, f},

   SwarmPos = Table[Join[RandomReal[#] & /@ condsizing,
       RandomReal[#] & /@ condbound], SwarmSize] // Transpose;
   SwarmVel = RandomReal[{0, 1}, {SwarmDim, SwarmSize}];

   f[x__] := {Fitness /. (Thread[promenne -> #] & /@ (x // Transpose))};

   SwarmPosf = f[SwarmPos];

   PBest = SwarmPos;

   gBest1 = Min[SwarmPosf];
```

```
  idx = First@First@Position[SwarmPosf, gBest1];

  For[j = 1, j ≤ (SwarmDim), j++,
   For[i = 1, i ≤ (SwarmSize), i++, {gBestt[i, j] = PBest[[j]][[idx]]}]];

  gBest = Table[gBestt[i, j], {j, 1, SwarmDim}, {i, 1, SwarmSize}];

  {gBest, PBest, SwarmPos, SwarmVel, SwarmPosf}];

UpdateSwarmPosition[SwarmSize_, SwarmDim_, SwarmPos_] :=
 Module[{gBesto, gBestof, PBestf, Swarmf,
   gBestn, idx, gBestnf, f, PBest1, PBestfq, Swarmfq, gBest1},

  gBesto = Table[gBest[[j, 1]], {j, 1, SwarmDim}];


  f[x__] := {Fitnessini /. Thread[promenne → x]};

  gBestof = f[gBesto];

  PBest1 = Table[PBest[[j, i]], {j, 1, SwarmDim}, {i, 1, SwarmSize}];
  gBest1 = Table[gBest[[j, i]], {j, 1, SwarmDim}, {i, 1, SwarmSize}];

  For[i = 1, i ≤ SwarmSize, i++,
   {Swarmfq[i] = First[Flatten[f[Table[SwarmPos[[j, i]], {j, 1, SwarmDim}]]]];
    PBestfq[i] = First[Flatten[f[Table[PBest[[j, i]], {j, 1, SwarmDim}]]]];
    If[Swarmfq[i] < PBestfq[i],
     {Table[PBest1[[j]][[i]] = SwarmPos[[j]][[i]], {j, 1, SwarmDim}];
      PBestfq[i] = Swarmfq[i]}, Table[PBest1[[j]][[i]] = PBest[[j]][[i]],
      {j, 1, SwarmDim}]]}];


  PBestf = Flatten@Table[PBestfq[i], {i, 1, SwarmSize}];

  gBestn = Min[PBestf];

  idx = First@First@Position[PBestf, gBestn];

  gBestnf = f[Table[PBest1[[j, idx]], {j, 1, SwarmDim}]];

  If[First@gBestnf < First@gBestof, For[i = 1, i ≤ SwarmSize, i++, {
     Table[gBest1[[j, i]] = PBest1[[j, idx]], {j, 1, SwarmDim}]
    }]];


  PBest = Table[PBest1[[j, i]], {j, 1, SwarmDim}, {i, 1, SwarmSize}];
  gBest = Table[gBest1[[j, i]], {j, 1, SwarmDim}, {i, 1, SwarmSize}];
  print =
   {gBestnf[[1]], Thread[promenne -> Table[PBest1[[j, idx]], {j, 1, SwarmDim}]]};
```

```
    {SwarmPos, PBest, gBest}];


InitializaSwarm[SwarmSizeini, SwarmDimini, FitnessLBini, FitnessUBini, Fitnessini];
iter = 0;
res = {};
  While[iter < iterations,
   {
    rand1 = RandomReal[{0, 1}, {SwarmDimini, SwarmSizeini}];
    rand2 = RandomReal[{0, 1}, {SwarmDimini, SwarmSizeini}];
    CF = 2 / Abs[(c1 + c2) - 2 + Sqrt[(c1 + c2)^2 - 4 * (c1 + c2)]];
    wCF = w * CF;
    C1 = CF * c1;
    C2 = CF * c2;

    SwarmVel =
     wCF * SwarmVel + C1 * rand1 * (PBest - SwarmPos) + C2 * rand2 * (gBest - SwarmPos);


    SwarmPos = SwarmPos + SwarmVel;

    UpdateSwarmPosition[SwarmSizeini, SwarmDimini, SwarmPos];
    res = Partition[Flatten[{res, print}], SwarmDimini + 1];
    Print[print];
    , iter++}];
  res
 ]
```

Převzato z: B. Higgins, H. Binous, A. Bellagi, and A. Al-Matar, “Particle Swarm Optimization for 2D Problems,” 2014. [online]. Dostupné z: http://demonstrations.wolfram.com/ParticleSwarmOptimizationFor2DProblems/ [cit. 29. 02. 2020].

a upraveno dle: M. A. Hossain, H. R. Pota, S. Squartini, and A. F. Abdou, “Modified PSO algorithm for real-time energy management in grid-connected microgrids,” Renewable Energy, vol. 136, pp. 746–757, 2019. [online]. Dostupné z: https://doi.org/10.1016/j.renene.2019.01.005

Funkce pro dimenzovani komponent ES

Funkce pro generovani reportu

```
reportexport[timestamp_, start_, mod_, MC_, data_, time_, soloptimalizace_] := Module[{
    notebookname =
     StringReplace[ToString[Last[FileNameSplit[NotebookFileName[]]]], ".nb" → ""],
    nadpisy = Flatten[{"hodnota_kriterialni_funkce",
       ToString /@ ((List @@@ soloptimalizace[[2]])[[ ;; , 1]])}], pocstartuzaden,
    penalezaden, Pcelkem, Ecelkem, anuity, Ninvcelkemanuity, Ninvcelkem, Ninv,
    provoznicaskogen, provoznicastc, provoznicasatc, Npstcelkem, Npstzvlast,
```

```
 Npprcelkem, Nppalcelkem, epsilonkog, epsilonatc, deltaprpal, penzlizacestartcelkem,
 Npprzvlast, TztcOpt, TptcOpt, Tptc, TzkogOpt, TpkogOpt, Tpkog, TzatcOpt,
 TpatcOpt, Tpatc, kritfcesizing, ekoprehled, kritfcepenalizacestav,
 penalizacestart, cenazpkog, epsilonkogden, epsilonatcden, cenazpatc
},
{{pocstartuzaden, penalezaden},
  Pcelkem, Ecelkem, anuity, {Ninvcelkemanuity, Ninvcelkem, Ninv},
  {provoznicaskogen, provoznicastc, provoznicasatc}, {Npstcelkem, Npstzvlast},
  {Npprcelkem, Nppalcelkem, {epsilonkogden, epsilonkog, cenazpkog, epsilonatcden,
    epsilonatc, cenazpatc}, deltaprpal, penzlizacestartcelkem, Npprzvlast},
  {TztcOpt, TptcOpt, Tptc}, {TzkogOpt, TpkogOpt, Tpkog}, {TzatcOpt, TpatcOpt, Tpatc},
  kritfcesizing, ekoprehled, kritfcepenalizacestav, penalizacestart} =
 kritfceoptimnminimize[start, mod, MC] /. soloptimalizace[[2]];
Export[ToString[start] <> "_" <> Evaluate[
   notebookname <> "_" <> timestamp <> ".xlsx"],
 {
  "Summary" → {
    {"Doba trvání optimalizace od spuštění", "Časové razítko", "Δt",
     "i0", "Počet period", "Řešení", "Penalizační konstanta", "Dosazeno",
     "AccuracyGoal", "PrecisionGoal", "WorkingPrecision", "Method",
     "MaxIterations", "OperationMethod", "Version"}, {time, timestamp, Δt, start,
     n0, soloptimalizace, velikostpenale, dosad, accuracygoal, precisiongoal,
     workingprecision, method, maxiterations, "fcedimenzovani", $Version}
   },
  "Solution" → Join[{{"Hodnota kriteriální funkce", soloptimalizace[[1]]}},
    {{"Hodnota penalizační funkce za nedodržení omezujících podmínek",
      kritfcepenalizacestav}},
    {{"Hodnota penalizační funkce za vícenásobné denní sepnutí
        kogenerační jednotky", penalizacestart}},
    List @@@ soloptimalizace[[2]]
   ],
  "Data" → Join[{
     nadpisy},
    Flatten /@ data[[1]]],
  "Cond" → ({uvazovanepromennesizingmeze,
      Flatten[{And @@@ Partition[Select[podminkysizingcelkem, ! NumberQ[#] &], 2],
        Partition[mezmrmr /@ podminkymeze, 1]}]} // Transpose),

  "Func" → ({{"promenneprooptim",
       "meze",
       "uvazovanepromennesizingtf",
       {"i0", "SOCakuele0", "Takutep0"},
       {"tau", "q", "Δt", "n0", "velikostpenale"},
       {"SOCmax", "SOCmin", "ηakuelenab", "ηakuelevyb", "Tzakuelemax"},
       {"ηmen", "Tzmenmax"},
       {"EERatc", "COPatc", {"Tpatcmax", "Tzatcmax"}},
       {"cpal", "minprovozhod", "maxprovozhod",
        "maxpocetstartuzaden", {"Tpkogmax", "Tzkogmax"}},
       {"COPtc", {"Tptcmax", "Tztcmax"}},
```

```
        {"Tzfvemax"},
        {"rho", "cv", "λref", "pref", "lref", "Takutepmin",
         "Takutepmax", "Takutepven", "Takuteplimit", "TzakutepOpt"},
        {"Ninvcelkem", "nprostcelkem", "nproprcelkem"},
        "Ninvprehled",
        "vratvysledky",
        "MonteCarloIndex"}, Table[kritfceoptimnminimize[start, mod, MC][[i]],
        {i, 1, Length[kritfceoptimnminimize[start, mod, MC]]}]} // Transpose),
    "Eko" → (Join[{Join[{""}, uvazovanakomponentanazev /@
          Insert[Insert[komponentyenergsystemu, rack, 2], rj, 7]]},
      Join[{{"Celkové investiční výdaje [Kč]", "Jednotková cena [Kč]",
          "Počet komponent [-]", "Instalovaná jednotka [W;Ws;m^3;-]"}},
        Partition[Flatten[If[Length[#] > 4, PadRight[Insert[
              Partition[#, UpTo[4]], 0, {2, 1}], {2, 4}, 0],
            If[Length[#] == 0, ConstantArray[0, 4], #]] & /@ ekoprehled], 4]] //
       Transpose, {Join[{"Celkové anuizované investiční výdaje [Kč]"},
        Insert[Insert[Ninv * anuity, 0, 2], 0, 7]]},
      {Join[{"Celkové provozní stálé náklady [Kč]"},
        Insert[Insert[Npstzvlast, 0, 2], 0, 7]]},
      {Join[{"Celkové provozní proměnné náklady [Kč]"},
        Insert[Insert[Npprzvlast, 0, 2], 0, 7]]},
      {Join[{"Celkové palivové náklady [Kč]"}, {0, 0, 0, cenazpatc, cenazpkog,
         0, 0, 0, 0}]}, {Join[{"Celková spotřeba zemního plynu [m^3]"},
        {0, 0, 0, epsilonatc, epsilonkog, 0, 0, 0, 0}]}] // Transpose),

    "Debug" → {{pocstartuzaden, penalezaden},
      Pcelkem, Ecelkem, anuity, {Ninvcelkemanuity, Ninvcelkem, Ninv},
      {provoznicaskogen, provoznicastc, provoznicasatc}, {Npstcelkem, Npstzvlast},
      {Npprcelkem, Nppalcelkem, {epsilonkog, epsilonatc}, deltaprpal,
       penzlizacestartcelkem, Npprzvlast}, {TztcOpt, TptcOpt, Tptc}, {TzkogOpt,
       TpkogOpt, Tpkog}, {TzatcOpt, TpatcOpt, Tpatc}, kritfcesizing, ekoprehled}
   }
  ]
 ];
```

Volani algoritmu dimenzovani komponent ES

```
In[801]:= kritfceoptimnminimize[start_, mod_, MC_] := fcedimenzovani[promenneprooptim,
            meze,
            uvazovanepromennesizingtf,
            {start, SOCakuele0, Takutep0},
            {tau, q, Δt, n0, velikostpenale},
            {SOCmax, SOCmin, ηakuelenab, ηakuelevyb, Tzakuelemax},
            {ηmen, Tzmenmax},
            {EERatc, COPatc, {Tpatcmax, Tzatcmax}},
            {cpal, minprovozhod, maxprovozhod, maxpocetstartuzaden, {Tpkogmax, Tzkogmax}},
            {COPtc, {Tptcmax, Tztcmax}},
            {Tzfvemax},
            {rho, cv, λ, p, lakutep, Takutepmin,
             Takutepmax, Takutepven, Takuteplimit, Tzakutepmax},
            {Ninvcelkem, nprostcelkem, nproprcelkem},
            Ninvprehled,
            mod,
            MC];

In[802]:= optimalizace[start_, krok_, konec_, MC_] :=
           Module[{i = start, data, time, timestamp, soloptimalizace},
            While[i ≤ konec,
             {time, {soloptimalizace, data}} =
              Reap[NMinimize[{kritfceoptimnminimize[i, "optimalizace", MC],
                   (And @@ podminkysizingcelkem) && (And @@ (mezmrmr /@ podminkymeze))},
                uvazovanepromennesizingmeze, MaxIterations → maxiterations,
                Method → method, PrecisionGoal → precisiongoal, AccuracyGoal → accuracygoal,
                StepMonitor :→ {Sow[{kritfceoptimnminimize[i, "optimalizace", MC],
                     uvazovanepromennesizingmeze}], Print[{kritfceoptimnminimize[i,
                      "optimalizace", MC], uvazovanepromennesizingmeze}]}]] // AbsoluteTiming;
             timestamp = StringJoin[(If[StringLength[#] > 1, #, StringJoin["0", #], #] & /@
                  (ToString /@ Drop[Now[[1]], -1]))];
             (*reportexport[timestamp,i,"export",MC,data,time,soloptimalizace];*)
             i = i + krok];
            soloptimalizace
           ]

In[803]:= vysledky = Table[optimalizace[i0, 2190, 8760, i], {i, 1, pocetscenarumc}];
```

Vyhodnoceni simulace Monte Carlo

```
(Kernel_2) In[ ]:= vyhodnocenimc[listvysledku_?ListQ] := Module[{hodkritfce = listvysledku[[ ;; , 1]],
                    hodkomponent = (Partition[List @@@ (listvysledku[[ ;; , 2]] // Flatten)[[ ;; , 2]],
                        uvazovanepromennesizingmeze // Length] // Transpose)},
                   {hodkritfce, hodkomponent, Min /@ hodkomponent, Max /@ hodkomponent,
                    StandardDeviation /@ hodkomponent}]

(Kernel_2) In[ ]:= {vyslhodkritfce, vyslkomponent, vyslmin, vyslmax, vyslstd} = vyhodnocenimc[vysledky];
```

Dosazeni vysledku do funkce pro dimenzovani

komponent ES

```
kanalyze = vysledky[[1, 2]];
```

In[805]:=
```
res = fcedimenzovani[promenneprooptim,
    meze,
    uvazovanepromennesizingtf,
    {i0, SOCakuele0, Takutep0},
    {tau, q, Δt, n0, velikostpenale},
    {SOCmax, SOCmin, ηakuelenab, ηakuelevyb, Tzakuelemax},
    {ηmen, Tzmenmax},
    {EERatc, COPatc, {Tpatcmax, Tzatcmax}},
    {cpal, minprovozhod, maxprovozhod, maxpocetstartuzaden, {Tpkogmax, Tzkogmax}},
    {COPtc, {Tptcmax, Tztcmax}},
    {Tzfvemax},
    {rho, cv, λ, p, lakutep, Takutepmin,
     Takutepmax, Takutepven, Takuteplimit, Tzakutepmax},
    {Ninvcelkem, nprostcelkem, nproprcelkem},
    Ninvprehled,
    "vratvysledky",
    1] /. kanalyze;
```

Vizualizace vysledku

In[806]:=
```
data2019 = DateRange[{2018, 12, 31, 23, 0, 0}, {2019, 12, 31, 23, 0, 0}, "Hour"];
vratsdatem[list_?ListQ] := {data2019, list // Flatten} // Transpose;
```

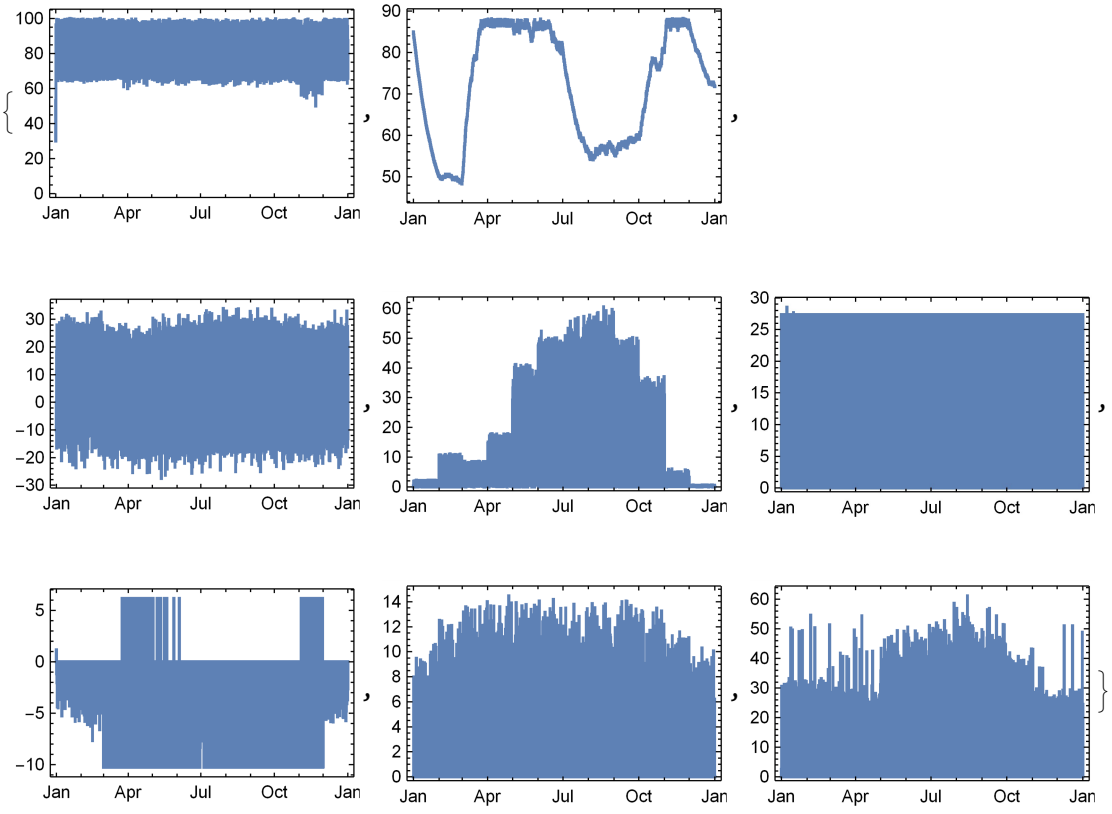
In[808]:=
```
SOCakueleproplot =
  vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_}, {Ekogele_, Eakuele_, Eakutep_,
       Etepbilance_, Eelebilance_, Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
      {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_, Pelebilance_, Ptctep_,
       Patcchl_, Pfve_, penale_, stav_}} :> {SOCakuele / Eakueleinst * 100 /. kanalyze}];
Takutepproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Takutep}];
Pkogeleproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Pkogele / kW}];
Pkogteproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Etepbilance / kW}];
```

```
Pakuelevybproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_,
       Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
      {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_, Pelebilance_, Ptctep_,
       Patcchl_, Pfve_, penale_, stav_}} :> {kladne[-Pakuele/kW]}];
Pakutepvybproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_,
       Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
      {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_, Pelebilance_, Ptctep_,
       Patcchl_, Pfve_, penale_, stav_}} :> {kladne[-Pakutep/kW]}];
Pakueleproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {-Pakuele/kW}];
Pakuelenabproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_,
       Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
      {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_, Pelebilance_, Ptctep_,
       Patcchl_, Pfve_, penale_, stav_}} :> {zaporne[-Pakuele/kW]}];
Pakutepnabproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_,
       Etctep_, Eatcchl_, Efve_, penaledt_, stavdt_},
      {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_, Pelebilance_, Ptctep_,
       Patcchl_, Pfve_, penale_, stav_}} :> {zaporne[-Pakutep/kW]}];
Pakutepproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {-Pakutep/kW}];
Ptctepproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Ptctep/kW}];
Ptceleproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {-Ptctep/(kW * COPtc)}];
```

```
Patcelepproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {-Patcchl/(kW * EERatc)}];
Patctepproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {-Patcchl/(kW * COPatc)}];
Patcchlproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Patcchl/kW}];
Pfveproplot = vratsdatem[res /. {bod : {t_, SOCakuele_, Takutep_},
      {Ekogele_, Eakuele_, Eakutep_, Etepbilance_, Eelebilance_, Etctep_, Eatcchl_,
       Efve_, penaledt_, stavdt_}, {Pkogele_, Pakuele_, Pakutep_, Ptepbilance_,
       Pelebilance_, Ptctep_, Patcchl_, Pfve_, penale_, stav_}} :> {Pfve/kW}];
komponentyproplot = {SOCakueleproplot, Takutepproplot, Pakueleproplot,
   Patcchlproplot, Pkogeleproplot, Pakutepproplot, Pfveproplot, Ptctepproplot};
```

In[825]:=
```
DateListStepPlot /@ komponentyproplot
```

Out[825]=



Citlivostni analyza

In[826]:=
```
citlivostnianalyza[listfci_?ListQ, listreferenci_?ListQ, relini_] :=
 Module[{vratrozsah, mezecitlivosti, reference, relativnizmena, relativnizmenaasoc,
   relativnizmenazaokr, serazenomeze, relativnizmenaporadiasoc, vysledek,
   nazvyfci = ToString /@ listfci, delka = Length[listreferenci]},
  vratrozsah[{x_, rel_}] := {x * (1 - rel), x * (1 + rel)};
  mezecitlivosti =
   Partition[({listfci, vratrozsah /@ ({listreferenci, ConstantArray[relini, delka]} //
            Transpose)} // Transpose) /. {a_, b_} :> {a /@ b} // Flatten, 2];
  reference = ({listfci, listreferenci} // Transpose) /. {a_, b_} :> {a[b]} // Flatten;
  relativnizmena = Flatten[{mezecitlivosti, reference} // Transpose, {1}] /.
    {{a_, b_}, c_} :> {a/c - 1, b/c - 1};
  relativnizmenazaokr = Round[#, 0.0001] & /@ relativnizmena * 100;
  serazenomeze =
   (Round[#, 1] & /@ Insert[mezecitlivosti // Transpose, reference, 2]) // Transpose;
  relativnizmenaasoc = AssociationThread[nazvyfci,
    Flatten /@ ({nazvyfci, relativnizmenazaokr, serazenomeze} // Transpose)];
  relativnizmenaporadiasoc = AssociationThread[
    Range[Length[reference]], ReverseSortBy[
     {nazvyfci, Max /@ Abs /@ relativnizmena} // Transpose, Last][[ ;; , 1]]];
  vysledek = relativnizmenaasoc /@ relativnizmenaporadiasoc /@ Range[delka]
 ]
```