MINISTÉRIO DA EDUCAÇÃO

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA MECÂNICA

Relatório interno

Programas para *software* MATLAB utilizados na tese "Bases para uma metodologia de dimensionamento de aproveitamentos híbridos baseados em energias hidrelétrica e fotovoltaica"

Alexandre Beluco, Paulo Kroeff de Souza e Arno Krenzinger

Porto Alegre, 16 de abril de 2001.

Os programas escritos com software MATLAB, versão 4.20, para o estudo computacional, aparecem listados nos quadros deste relatório. O programa básico para simulação é o programa denominado *simhhs.m* (QUADRO 1).

Esse programa pode utilizar dados idealizados, gerados com o auxílio das rotinas *fepht.m* (QUADRO 4), *fepst.m* (QUADRO 6), *fephd.m* (QUADRO 5) e *fepsd.m* (QUADRO 7), ou pode utilizar dados reais de disponibilidade energética, armazenados em arquivos de entrada apropriados.

As duas primeiras lidam com dados correspondentes às potências disponíveis respectivamente aos geradores hidrelétrico e fotovoltaico. As duas seguintes lidam com dados que podem ser considerados brutos, correspondentes à vazão e à radiação solar incidente por unidade de área sobre uma superfície plana ou inclinada.

Os valores de potências instaladas e de capacidade dos acumuladores são obtidas em função dos parâmetros `PHmáx`, `A`, `EbatT`, `profd`, `NSbat` e `EresT`, calculadas pela rotina *hhsprj.m* (QUADRO 2).

Os dados de demanda energética por parte das cargas consumidoras são gerados com o auxílio das rotinas *fepc1.m* (QUADRO 9), *fepc2.m* (QUADRO 10), *fepc3.m* (QUADRO 11) e *fepc4.m* (QUADRO 12). Esses programas correspondem respectivamente ao perfil de demanda constante, ao perfil aproximadamente senoidal, com pico ao meio dia, ao perfil de KELLOGG et alii (1998), ao perfil de van DIJK et alii (1991), ao perfil de LASNIER e ANG (1990).

Os resultados devem ser analisados por rotinas específicas, escritas com objetivos bem definidos. A construção de gráficos com resultados, como o da FIGURA 3. 11 da tese de Beluco, 2001, é obtida com a rotina *anl01.m* (QUADRO 13).

A linha de comando para acionamento da rotina “simhhs” é:

```
simhhs(numord,trel,Q,A,tdef,dte,AcB,AcR,dem)
```

onde `numord` estabelece uma designação para o arquivo de saída, `trel` estabelece o tipo de relatório que deve ser gerado, conforme BELUCO (2001), `Q` é um vetor com componentes que definem a vazão mínima (`Qmin`) e a vazão máxima (`Qmax`) disponíveis e a potência máxima turbinável (`PHmax`), `A` define a área dos módulos fotovoltaicos, `tdef` estabelece o tempo de defasagem entre os valores mínimos das disponibilidades hídrica e solar, `dte` estabelece o intervalo de tempo decorrido entre cada rodada do programa de simulação, `AcB` é um vetor com componentes que definem a capacidade das baterias (`EbatT`) e os níveis mínimo (`profd`) e máximo (`NSbat`) para descarga das baterias, `AcR` é um vetor com componentes

que definem a existência (`res`) e a capacidade (`EresT`) do reservatório e um parâmetro para definição da vazão turbinada (`N2`), e finalmente `dem` estabelece o tipo de perfil de demanda adotado.

QUADRO 1. Programa *simhhs.m*.

```
function simhhs(numord,trel,Q,A,tdef,dte,AcB,AcR,dem)
t1=clock;
global VCer VCgf VCgh
if trel==1,
fr1=('%9.4f%9.4f%8i%5.2f%5.2f%5.2f%5.2f%5.2f%5.2f%5.2f%5.2f%7.2f%12.4f%4.1f...
   %7.2f%12.4f%4.1f%12.4f%7.4f%2i%2i%2i\n');
elseif trel==2,
fr1=('%9.4f%9.4f%5.2f%5.2f%5.2f%5.2f%12.4f%12.4f%7.4f%2i%2i%2i\n'); end
frf=('%9.4f%9.4f\n');
numordf=numord; numordf2=fix(numordf/1000);
if numord~=0
   numord=fix(numord*1000)/1000; dte=fix(dte*100)/100;
   numord=(dte*10000)+numord;
   numordD6=fix(numord/100000);
   numordD5=fix(numord/10000)-(numordD6*10);
   numordD4=fix(numord/1000)-(numordD6*100)-...
            (numordD5*10);
   numordD3=fix(numord/100)-(numordD6*1000)-...
            (numordD5*100)-(numordD4*10);
   numordD2=fix(numord/10)-(numordD6*10000)-...
            (numordD5*1000)-(numordD4*100)-...
            (numordD3*10);
   numordD1=fix(numord)-(numordD6*100000)-...
            (numordD5*10000)-(numordD4*1000)-...
            (numordD3*100)-(numordD2*10);
   arqidn1=(['t3' num2str(numordD6) num2str(numordD5)...
                  num2str(numordD4) num2str(numordD3)...
                  num2str(numordD2) num2str(numordD1) '.txt']);
   arqidf1=(['f3' num2str(numordD6) num2str(numordD5)...
                  num2str(numordD4) num2str(numordD3)...
                  num2str(numordD2) num2str(numordD1) '.txt']);
   arqid1=fopen(arqidn1,'w');
   arqidf=fopen(arqidf1,'w');
   if numordf2==9,
      TFAb=numordD3+(numordD2/10)+(numordD1/100); end, end
Qmin=Q(1); Qmax=Q(2); PHmax=Q(3); N1=PHmax;
EbatT=AcB(1); profd=AcB(2); NSbat=AcB(3);
res=AcR(1); EresT=AcR(2); N2=AcR(3); N3=0.20;
if dem==1, [Tv,PCv]=fepc1(td);
elseif dem==2, [Tv,PCv]=fepc2(td);
elseif dem==3, [Tv,PCv]=fepc3(td);
elseif dem==4, [Tv,PCv]=fepc4(td);
else, D=('Não é um perfil de demanda conhecido!'), end
k4=1; k5=1; k61=0; k62=0; k7=0; k8=0; k9=0; k91=0; k92=0;
rendgh=VCgh(1,2); rendgf=VCgf(1,2); Hmax=VCgh(2,1);
rendr=0.90; rendi=0.90; rendbat=0.90;
[PSt(k5)]=fepst(0); [PSd(k5)]=fepsd(PSt(k5)); PS(k5)=PSd(k5);
[PHt(k5)]=fepht(0); [PHd(k5)]=fephd(PHt(k5)); H(k5)=VCgh(1,1);
if res==1, PHd(k5)=PHd(k5)*H(k5); end
Cf(k5)=1; Ch(k5)=1; Cc(k5)=1;
Ebat(k5)=NSbat*EbatT; Eres(k5)=EresT; Eresi=EresT;
if trel==1, dEb(k4)=1; dEr(k5)=1; dEr(k5+1)=dEr(k5); end
VresT=1.00; Vres(k5)=VresT; H(k5)=Vres(k5)^(1/3);
Vres(k5+1)=Vres(k5); H(k5+1)=H(k5);
hhsin(Qmin,Qmax,A,tdef);
if numordf2==9, k11f=1; else, k11f=2; end
tdi(1)=0; tdf(1)=359; tdi(2)=360; tdf(2)=719;
for k11=1:k11f, for td=tdi(k11):tdf(k11)
   if td==0, tv(k5)=td; t=td; TFA1=0;
      PC(k5)=PCv(1); PCa(k5)=PC(k5);
      if PC(k5)<=((PHd(k5)*rendr)*rendi)
         PH(k5)=(PC(k5)/rendi)/rendr;
```

QUADRO 1. (*continuação*)

```
      else, PH(k5)=PHd(k5); end
      dEbat=0.00; dEres=0.00; end
   if td==360, TFA2=0;
      if trel==1
         r1=[t-360;TFA2;k4;PSt(k5);PSd(k5);PS(k5);...
            PHt(k5);PHd(k5);PH(k5);PC(k5);PCa(k5);...
            dEbat;Ebat(k5);dEb(k5);dEres;Eres(k5);dEr(k5);...
            Vres(k5);H(k5);Cf(k5);Ch(k5);Cc(k5)];
      elseif trel==2
         r1=[t-360;TFA2;PS(1);PH(1);PC(1);PCa(1);...
            Ebat(1);Eres(1);H(1);Cf(1);Ch(1);Cc(1)]; end
         fprintf(arqid1,fr1,r1); end
   k1i=2; k1f=length(Tv);
   for k1=k1i:k1f
      t=td+(Tv(k1)/24); dt=(Tv(k1)-Tv(k1-1))/dte;
      k2i=Tv(k1-1)+dt; k2f=Tv(k1);
      for k2=k2i:dt:k2f
         k3=td+(k2/24); k4=k4+1; k5=k5+1; tv(k5)=k3;
         [PSt(k5)]=fepst(k3); %ESt=ESt+(PSt(k5)*dt);
         [PSd(k5)]=fepsd(PSt(k5)); %ESd=ESd+(PSd(k5)*dt);
         PSp=max(PSd);
         [PHt(k5)]=fepht(k3); %EHt=EHt+(PHt(k5)*dt);
         [PHd(k5)]=fephd(PHt(k5)); %EHd=EHd+(PHd(k5)*dt);
         if res==1, PHd(k5)=PHd(k5)*H(k5-1); end
         PC(k5)=PCv(k1-1); %EC=EC+(PC(k5)*dt);
         dEbat1=(PS(k5-1)*Cf(k5-1))+(PH(k5-1)*rendr*Ch(k5-1));
         dEbat2=(PC(k5-1)*Cc(k5-1))/rendi; dEbat=dEbat1-dEbat2;
         Ebat(k5)=Ebat(k5-1)+(dEbat*dt);
         if Ebat(k5)<0, Ebat(k5)=0;
         elseif Ebat(k5)>EbatT, Ebat(k5)=EbatT; end
         if trel==1 & dEbat>0 & Ebat(k5)<EbatT, dEb(k5)=1.1;
         elseif trel==1 & dEbat<0 & Ebat(k5)>(profd*EbatT), dEb(k5)=0.9;
         elseif trel==1 & dEbat==0, dEb(k5)=1; end
         if Ebat(k5)>(NSbat*EbatT), k7=1;
         elseif Ebat(k5)<(profd*EbatT), k7=-1;
         else, k7=0; end
         if k7==1 & Ebat(k5)<=(NSbat*EbatT), k7=0; end
         if k7==-1 & Ebat(k5)>(profd*EbatT), k7=0; end
         if k7==1, Cf(k5)=0; Ch(k5)=0; Cc(k5)=1;
         elseif k7==-1, Cf(k5)=1; Ch(k5)=1; Cc(k5)=0;
         else, Cf(k5)=1; Ch(k5)=1; Cc(k5)=1; end
         if res==1
            dEres=PHd(k5-1)-PH(k5-1);
            if (Eres(k5-1)==0 & dEres<0)...
                  | (Eres(k5-1)==EresT & dEres>0), dEres=0;
               PH1=((PC(k5-1)/rendi)-PS(k5-1))/rendr;
               if PH1>=0 & PH1<=PHd(k5-1), PH(k5-1)=PH1; end, end
            Eres(k5)=Eres(k5-1)+(dEres*dt);
            if Eres(k5)<0, Eres(k5)=0;
            elseif Eres(k5)>EresT, Eres(k5)=EresT;  end
            if trel==1 & dEres>0 & Eres(k5)<EresT, dEr(k5)=1.1;
            elseif trel==1 & dEres<0 & Eres(k5)>0, dEr(k5)=0.9;
            elseif trel==1 & dEres==0, dEr(k5)=1; end
            dVres=dEres/(H(k5)*dt); Vres(k5)=Vres(k5-1)+dVres;
            H(k5)=Vres(k5)^(1/3);
         else, dEres=0; Eres(k5)=0; dVres=0; Vres(k5)=0; end
         if Cc(k5)==0, PCa(k5)=0;
         else, PC1=(PS(k5-1)*Cf(k5-1))+(PH(k5-1)*rendr*Ch(k5-1));
            PCa(k5)=(PC1-dEbat)*rendi; end
         if PCa(k5)<0, PCa(k5)=0; k91=k91+1;
            Am=('Há inconsistências nos cálculos!'), end
         if PCa(k5)>PC(k5), PCa(k5),PC(k5),PCa(k5)=PC(k5); k92=k92+1;
            AM=('Há inconsistências nos cálculos!'), end
         if fix(PCa(k5)*100)<fix(PC(k5)*100)
            if k11==1, k61=k61+1; end
            if k11==2, k62=k62+1; end
            if k11f==2, if k8==0, tfi=k3; end, k8=1; end, end
         if fix(PCa(k5)*100)==fix(PC(k5)*100) & k8==1 % k11f==2
            k8=0; tff=k3; rf=[tfi;tff-tfi];
            fprintf(arqidf,frf,rf); end
```

QUADRO 1. (*continuação*)

```
         if Cf(k5)==0, PS(k5)=0;
         else, PS(k5)=PSd(k5); end
         if Ch(k5)==1 & Cc(k5)==1
            if res==1
               if Ebat(k5)<(NSbat*EbatT) & Vres(k5)>N3*VresT
                  PH1=((PC(k5)/rendi)-PSd(k5))/rendr;
                  if PH1<0, PH(k5)=0;
                  elseif PH1>PHd(k5), PH(k5)=PHd(k5);
                  else, PH2=N2*(((NSbat*EbatT)-Ebat(k5))/dt);
                     PH(k5)=PH1+PH2; end
               else, PH1=((PC(k5)/rendi)-PSd(k5))/rendr;
                  if PH1<0, PH(k5)=0;
                  else, PH(k5)=PH1; end, end
            else
               if Ebat(k5)<(NSbat*EbatT), PH(k5)=PHd(k5);
               else, PH1=((PC(k5)/rendi)-PSd(k5))/rendr;
                  if PH1<0, PH(k5)=0;
                  elseif PH1>PHd(k5), PH(k5)=PHd(k5);
                  else, PH(k5)=PH1; end, end, end
         elseif Ch(k5)==1 & Cc(k5)==0, PH(k5)=PHd(k5);
         else, PH(k5)=0; end
         if res~=1 & PH(k5)>PHd(k5), PH(k5)=PHd(k5); end
         if PH(k5)>N1, PH(k5)=N1; end
         PSt(1)=PSt(k5); PSd(1)=PSd(k5); PS(1)=PS(k5);
         PHt(1)=PHt(k5); PHd(1)=PHd(k5); PH(1)=PH(k5);
         PC(1)=PC(k5); PCa(1)=PCa(k5); H(1)=H(k5);
         Ebat(1)=Ebat(k5); Eres(1)=Eres(k5); Vres(1)=Vres(k5);
         if trel==1, dEb(1)=dEb(k5); dEr(1)=dEr(k5); end
         Cf(1)=Cf(k5); Ch(1)=Ch(k5); Cc(1)=Cc(k5); k5=1;
      end
      TFA1=k61*dt/24; TFA2=k62*dt/24; TFA=TFA1+TFA2;
      if numordf2==9 & (k61+k62)>(TFAb*24/dt), break, end
      leituras(:,1)=[720-t;numordf/10000]; leituras'
      if numord~=0 & k11==2 & t<720
         if trel==1
            r1=[t-360;TFA2;k4;PSt(1);PSd(1);PS(1);...
                PHt(1);PHd(1);PH(1);PC(1);PCa(1);...
                dEbat;Ebat(1);dEb(1);dEres;Eres(1);dEr(1);...
                Vres(1);H(1);Cf(1);Ch(1);Cc(1)];
         elseif trel==2
            r1=[t-360;TFA2;PS(1);PH(1);PC(1);PCa(1);...
                  Ebat(1);Eres(1);H(1);Cf(1);Ch(1);Cc(1)]; end
            fprintf(arqid1,fr1,r1); end
      if numordf2==9 & (k61+k62)>(TFAb*24/dt), break, end, end
if numordf2==9 & (k61+k62)>(TFAb*24/dt), break, end
end, if numordf2==9 & (k61+k62)>(TFAb*24/dt), break, end, end
if numordf2==9, TFA2=TFA1+TFA2, end
fclose('all');
t2=clock; T=etime(t2,t1)
end
```

A linha de comando para acionamento da rotina “hhsprj” é:

`[sprj1,sprj2,sprj3,sprj4]=hhsprj(modprj,prj1,prj2,prj3,dem,prj4)`

onde `modprj` estabelece o tipo de resultado pretendido, `prj1`, `prj2`, `prj3` e `prj4` definem as várias entradas para o programa, em função da escolha para `modprj`, `dem` estabelece o tipo de perfil de demanda, e `sprj1`, `sprj2`, `sprj3` e `sprj4` definem as várias saídas do programa.

QUADRO 2. Programa *hhsprj.m*.

```
function [sprj1,sprj2,sprj3,sprj4]=hhsprj(modprj,prj1,prj2,prj3,dem,prj4)
global VCer VCgh VCgf
if modprj==1
   Qmin=prj1; Qmax=prj2; A=prj3;
   hhsin(Qmin,Qmax,A,0);
   t=0:0.05:359.95;
   [PSt]=fepst(t); [PSd]=fepsd(PSt); PS=PSd.*0.90;
   ES=sum(PS.*0.05.*24);
   [PHt]=fepht(t); [PHd]=fephd(PHt); PH=PHd.*0.90.*0.90;
   EH=sum(PH.*0.05.*24);
   if dem==1, [Tv,PCv]=fepc1(t);
   elseif dem==2, [Tv,PCv]=fepc2(t);
   elseif dem==3, [Tv,PCv]=fepc3(t);
   elseif dem==4, [Tv,PCv]=fepc4(t); end
   EC=0;
   for k1=1:(length(Tv)-1)
      EC=EC+(PCv(k1)*(Tv(k1+1)-Tv(k1)));
   end
   EC=EC*360
   ET=ES+EH; sprj1=ET;
   RSH=ES/EH; sprj2=RSH;
   RTC=ET/EC; sprj3=RTC;
elseif modprj==2
   RSH=prj1; RTC=prj2; dMm=prj3;
   t=0:0.05:359.95;
   if dem==1, [Tv,PCv]=fepc1(t);
   elseif dem==2, [Tv,PCv]=fepc2(t);
   elseif dem==3, [Tv,PCv]=fepc3(t);
   elseif dem==4, [Tv,PCv]=fepc4(t); end
   EC=0;
   for k1=1:(length(Tv)-1)
      EC=EC+(PCv(k1)*(Tv(k1+1)-Tv(k1)));
   end
   EC=EC*360; ET=RTC*EC;
   if RSH==-1, EH=0.00; ES=ET;
   else, EH=ET/(1+RSH); ES=RSH*EH; end
   ESt=0; EHt=0; Qt=0; At=0;
   while ESt<=ES | EHt<=EH
      if ESt<=ES, At=At+1; end
      if EHt<=EH, Qt=Qt+0.1; Qmint=Qt; Qmaxt=Qt; end
      hhsin(Qmint,Qmaxt,At,0);
      [PStt]=fepst(t); [PSdt]=fepsd(PStt); PSt=PSdt.*0.90;
      ESt=sum(PSt.*0.05.*24);
      [PHtt]=fepht(t); [PHdt]=fephd(PHtt); PHt=PHdt.*0.90.*0.90;
      EHt=sum(PHt.*0.05.*24);
   end
   k2=k2+1; dAt=2; Atu=At; ks=10e20;
   while ESt~=ES
      if ESt<ES, At=At+(dAt/2); dAt=dAt/2; At=At-(dAt/2);
      elseif ESt>ES, dAt=dAt/2; At=At-(dAt/2);
      elseif ESt==ES, dAt=0; end
      hhsin(Qmint,Qmaxt,At,180);
      [PStt]=fepst(t); [PSdt]=fepsd(PStt); PSt=PSdt.*0.90;
      ESt=sum(PSt.*0.05.*24);
      if fix(Atu*ks)==fix(At*ks), ESt=ES; end
      Atu=At;
   end
   k3=k3+1; dQt=0.2; Qtu=Qt; kh=10e20;
   while EHt~=EH
      if EHt<EH, Qt=Qt+(dQt/2); dQt=dQt/2; Qt=Qt-(dQt/2);
      elseif EHt>EH, dQt=dQt/2; Qt=Qt-(dQt/2);
      elseif EHt==EHt, dQt=0; end
      Qmint=Qt; Qmaxt=Qt;
      hhsin(Qmint,Qmaxt,At,180);
      [PHtt]=fepht(t); [PHdt]=fephd(PHtt); PHt=PHdt.*0.90.*0.90;
      EHt=sum(PHt.*0.05.*24);
      if fix(Qtu*kh)==fix(Qt*kh), EHt=EH; end
      Qtu=Qt;
   end
```

QUADRO 2. (*continuação*)

```
    hhsin(Qt,Qt,At,0);
    tmin=90:0.05:90.95; tmax=270:0.05:270.95;
    PStmin=fepst(tmin); PStmax=fepst(tmax);
    PSdmin=fepsd(PStmin); PSdmax=fepsd(PStmax);
    Emax=sum(PSdmax.*0.05); Emin=sum(PSdmin.*0.05);
    PHdmax=(Qt*0.90*0.60)+((dMm/2)*(Emax-Emin));
       PHdmin=(2*Qt*0.90*0.60)-PHdmax;
    PHtmax=PHdmax/0.90/0.60; PHtmin=PHdmin/0.90/0.60;
    Qmin=PHtmin; Qmax=PHtmax;
    sprj1=Qmin; sprj2=Qmax; sprj3=At;
end
end
```

A linha de comando para acionamento da rotina “hhsin” é:

```
hhsin(Qmin,Qmax,A,tdef)
```

onde `Qmin` define a vazão mínima disponível, `Qmax` define a vazão máxima disponível, `A` define a área dos módulos fotovoltaicos e `tdef` estabelece a defasagem entre os valores mínimos das disponibilidades hídrica e solar.

QUADRO 3. Programa *hhsin.m*.

```
function hhsin(Qmin,Qmax,A,tdef)
global VCer VCgf VCgh
dQmin=180+90; tP=9; dDmin=9; dPsmin=0+90+tdef; dDmax=15;
Psmin=0.4678311; Psmax=0.5914258625;
VCer=[ dQmin   tP  dDmin  0
      dPsmin   0   dDmax  0
       Qmin  Psmin   0    0
       Qmax  Psmax   0    0];
rendgf=0.12; VCgf=[A rendgf]; H=1; Hmax=1.2; rendgh=0.60;
VCgh=[ H   rendgh
      Hmax   0   ];
end
```

A linha de comando para acionamento da rotina “fepht” é:

```
[PHt]=fepht(t)
```

onde `t` define o instante de tempo, podendo ser um vetor, e `PHt` é a potência disponível ao gerador hidrelétrico. A linha de comando para acionamento da rotina “fephd” é:

```
[PHd]=fephd(PHt)
```

onde `t` define o instante de tempo, podendo ser um vetor, e `PHd` é a potência disponibilizada pelo gerador hidrelétrico.

QUADRO 4. Programa *fepht.m*.

```
function [PHt]=fepht(t)
global VCer VCgh
tQmin=VCer(1,1);
Qmin=VCer(3,1); Qmax=VCer(4,1);
Qmed=(Qmin+Qmax)/2; AQ=Qmax-Qmed;
tdQ=tQmin-((3*360)/4);
Q=(AQ.*sin(2.*pi.*(1/360).*...
   (t-tdQ)))+Qmed;
H=VCgh(1,1); PHt=Q.*H;
PHt=round(PHt*10000)/10000;
end
```

QUADRO 5. Programa *fephd.m*.

```
function [PHd]=fephd(PHt)
global VCgh
PHd=PHt.*VCgh(1,2);
PHd=round(PHd*10000)/10000;
end
```

A linha de comando para acionamento da rotina “fepst” é:

```
[PSt]=fepst(t)
```

onde `t` define o instante de tempo, podendo ser um vetor, e `PSt` é a potência disponível ao gerador fotovoltaico. A linha de comando para acionamento da rotina “fepsd” é:

```
[PSd]=fepsd(PSt)
```

onde `t` define o instante de tempo, podendo ser um vetor, e `PSd` é a potência disponibilizada pelo gerador fotovoltaico.

QUADRO 6. Programa *fepst.m*.

```
function [PSt]=fepst(t)
global VCer
tPSmin=VCer(2,1);
PSmaxmin=VCer(3,2); PSmaxmax=VCer(4,2);
PSmaxmed=(PSmaxmin+PSmaxmax)/2;
APS=PSmaxmax-PSmaxmed;
tdPS=tPSmin-((3*360)/4);
PSmax=(APS.*sin(2.*pi.*(1/360).*(t-
tdPS)))+PSmaxmed;
DHmin=VCer(1,3); DHmax=VCer(2,3);
DHmed=(DHmin+DHmax)/2;
ADH=DHmax-DHmed;
tdDH=tPSmin-((3*360)/4);
td=fix(t);
DH=(ADH.*sin(2.*pi.*(1/360).*(td-
tdDH)))+DHmed; Dtd=(t-td)*24;
td2=Dtd-(12-(DH/2));
PSt=PSmax*sin(2*pi.*(1./(2.*DH)).*(td2));
for k=1:length(PSt)
   if PSt(k)<0
      PSt(k)=0;
   end
end
PSt=round(PSt*10000)/10000;
end
```

QUADRO 7. Programa *fepsd.m*.

```
function [PSd]=fepsd(PSt)
global VCgf
A=VCgf(1,1);
rend=VCgf(1,2);
PSd=PSt.*A.*rend;
PSd=round(PSd*10000)/10000;
end
```

As linhas de comando para acionamento respectivamente das rotinas “fepc”, “fepc”, “fepc”, “fepc” e “fepc” são:

```
[Tv,PCv]=fepc1(t)
[Tv,PCv]=fepc2(t)
[Tv,PCv]=fepc3(t)
[Tv,PCv]=fepc4(t)
[Tv,PCv]=fepc5(t)
```

onde `t` define o instante de tempo, podendo ser um vetor, `Tv` é um vetor com os instantes de tempo em que o programa de simulação deve registrar resultados no arquivo de saída, e `PCv` é um vetor com os valores de potência consumida pelas cargas.

QUADRO 8. Programa *fepc1m*.

```
function [Tv,PCv]=fepc1(t)
t0=0; t1=2; t2=4; t3=6;
t4=8; t5=10; t6=12;
t7=14; t8=16; t9=18;
t10=20; t11=22; t12=24;
Tv=[t0 t1 t2 t3 t4 t5 t6...
   t7 t8 t9 t10 t11  t12];
PCv=[1.00 1.00 1.00 1.00 1.00 1.00...
   1.00 1.00 1.00 1.00 1.00 1.00];
end
```

QUADRO 9. Programa *fepc2m*.

```
function [Tv,PCv]=fepc2(t)
Tv=0:1:24;
PCv=[1.00 1.00 1.00 1.00 1.00 1.00...
     1.00 1.00 1.00 1.00 1.00 1.00...
     1.00 1.00 1.00 1.00 1.00 1.00...
     1.00 1.00 1.00 1.00 1.00 1.00];
end
```

QUADRO 10. Programa *fepc3.m*.

```
function [Tv,PCv]=fepc3(t)
Tv=0:1:24;
PCv=[0.20 0.20 0.20 0.20 0.20 0.20...
     1.00 0.20 0.20 0.20 0.20 1.00...
     1.00 0.20 0.20 0.20 0.20 1.00...
     1.00 1.00 0.20 0.20 0.20 0.20];
end
```

QUADRO 11. Programa *fepc4.m*.

```
function [Tv,PCv]=fepc4(t)
Tv=0:1:24;
PCv=[0.20 0.20 0.20 0.20 0.40 0.40...
     0.60 0.60 0.80 0.80 1.00 1.00...
     1.00 1.00 0.80 0.80 0.60 0.60...
     0.40 0.40 0.20 0.20 0.20 0.20];
end
```

QUADRO 12. Programa *fepc5.m*.

```
function [Tv,PCv]=fepc5(t)
Tv=0:1:24;
PCv=[0.20 0.20 0.20 0.20 0.20 0.20...
     0.20 0.20 0.20 0.20 0.20 0.20...
     0.20 0.20 0.20 0.20 1.00 1.00...
     1.00 1.00 1.00 1.00 0.20 0.20];
PCv=[0.20 0.20 0.20 0.20 0.20 0.20 0.20
0.20 1.00 1.00 1.00 0.20];
end
```

A linha de comando para acionamento da rotina “anl01” é:

```
[tfa]=anl01(numt,numord,denEbat)
```

onde `numt` e numord estabelecem o tipo de arquivo com resultados a ser analisado, `denEbat` define um denominador, para que o eixo vertical para o estado de carga coincida com o eixo vertical para as potências, e `tfa` é o tempo de falhas no atendimento.

QUADRO 13. Programa *anl01.m*.

```
function [tfa]=anl01(numt,numord,denEbat)
if numord~=0
   numord=fix(numord*1000000)/1000000;
   numordD6=fix(numord/100000);
   numordD5=fix(numord/10000)-(numordD6*10);
   numordD4=fix(numord/1000)-(numordD6*100)-...
            (numordD5*10);
   numordD3=fix(numord/100)-(numordD6*1000)-...
            (numordD5*100)-(numordD4*10);
   numordD2=fix(numord/10)-(numordD6*10000)-...
            (numordD5*1000)-(numordD4*100)-(numordD3*10);
   numordD1=fix(numord)-(numordD6*100000)-...
            (numordD5*10000)-(numordD4*1000)-(numordD3*100)-...
            (numordD2*10);
   arqidn=(['t' num2str(numt)...
             num2str(numordD6) num2str(numordD5)...
             num2str(numordD4) num2str(numordD3)...
             num2str(numordD2) num2str(numordD1)]);
   eval(['load ' arqidn '.txt;']);
end
eval(['A=' arqidn ';']);
tv=A(:,1); PS=A(:,3); PH=A(:,4); PCa=A(:,6); Ebat=A(:,7);
TFA=A(:,2); tfa=TFA(size(A,1));
PHd=A(:,8); dPH=PHd-PH;
dt=((2/24)/((numordD6*10)+numordD5));
V=sum(dPH.*dt);
k1=0;
for k2=1:size(A,1)
   a=A(k2,1);
   if (a-fix(a))==0.5
      k1=k1+1;
      tvg(k1)=tv(k2); PSg(k1)=PS(k2);
   end
end
%plot(tv,PS,'w',tv,PCa,'g',tv,PH,'b',tv,Ebat/denEbat,'r:')
tv=tv./360; tvg=tvg./360;
plot(tvg,PSg,'w:',tv,PCa,'g',tv,PH,'b',tv,Ebat/denEbat,'r:')
axis([0 1 0 3])
xlabel('t   [anos]'); ylabel('[pu]');
grid
end
```