B. Programes

En aquest Annex es detalla el codi dels programes utilitzats en aquest projecte per processar les sèries temporals.

Aquests programes s'han realitzat des de l'entorn de computació i desenvolupament MATLAB.

B.1. DinàmicaSimbolicaConjunda.m

Programa que aplica el mètode de la Dinàmica Simbòlica Conjunta a les sèries temporals, variabilitat del ritme cardíac, RR(n), i duració del cicle respiratori, Ttot(n).

B.1.1. DinàmicaSimbolicaConjunda.m

```
% Els arxius  corresponents als senyals cardíacs es troben a la carpeta RR_cardresp i els
% que corresponen als senyals respiratoris a la carpeta TOT_cardresp. Els identificadors
% de tots aquests arxius es guarden a les variables file_RR i file_Ttot
file_RR=dir('RR_cardresp/*.*');
file_Ttot=dir('TOT_cardresp/*.*');

% La variable x és igual al nombre de pacients que podran ser tractats.
% h és el nombre que s'assigna al pacient que s'està tractant i servirà per identificar-lo j és
% una variable que indica quina posició de la llista de pacients s'està tractant (h+2, si
% per a tots els pacients a tractar es disposa dels dos senyals) és una variable que servirà
% si en algun cas no es disposa dels dos senyals a estudiar
x=size(file_RR,1)-2;
j=3;
h=0;

% Es creen les variables necessàries per guardar les dades. S'utilitzaran per guardar
% identificador de pacients original, la probabilitat d'aparició de les paraules combinades,
% paraules independents, el nombre de paraules amb una certa probabilitat d'aparèixer,
% el nombre les paraules prohibides i l'entropia de Shannon.
dades1=zeros(x,64);
dades2=zeros(x,16);
dades3=zeros(x,21);
pacients=zeros(x,5);
% Es realitza un recorregut per tots el pacients
```

```
while (j<=(x+2));

% Funció que busca els senyals RR i Ttot per a un mateix pacient
[RR,Ttot,j,nameRR,nameTtot,h]= obtenirdades(file_RR,file_Ttot,j,h+1);

% Grafica valors inicials dels senyals en funció del temps i els guarda en la carpeta
% graf_ini_RR_t i  graf_ini_Ttot_t
h=h-1;
cd graf_ini_RR_t;
figure(h);plot(RR(:,1),RR(:,2),'.'), figure(h);
title('RR(temps)'), figure(h);
xlabel('temps'), figure(h);
ylabel('RR'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd('..')
cd graf_ini_Ttot_t;
figure(h);
plot(Ttot(:,1),Ttot(:,2),'.'), figure(h);
xlabel('temps'), figure(h);
ylabel('RR'), figure(h);
title('Ttot(temps)'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..');

% Grafica valors inicials dels senyals en funció del temps i els guarda en la carpeta
% graf_ini_RR_n i  graf_ini_Ttot_n
cd graf_ini_RR_n;
figure(h);
plot(RR(:,2),'.'), figure(h);
xlabel('mostra'), figure(h);
ylabel('RR'), figure(h);
title('RR(mostra)')
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd('..')
```

```
cd graf_ini_Ttot_n;
plot(Ttot(:,2),'.'), figure(h);
xlabel('mostra'), figure(h);
ylabel('RR'), figure(h);
title('Ttot(mostra)'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd('..')
```

```
% Es mostregen i s'interpolen els valors per tenir valors dels senyals en el mateix instants
% de temps. S'obtenen valors cada t segons.
% tini és necessari ja que les dades no comencen al mateix instant, i per assegurar que hi
% ha valor en els dos senyals que es tracten es triarà el valor més gran dels dos instants
% inicials. El mateix passa amb el temps final, tf, que serà el temps final més petit de les
% dues sèries temporals.

rtRR=size(RR,1);
rtTtot=size(Ttot,1);
r=[RR(rtRR,1), Ttot(rtTtot,1)];
tf=min(r);
tiniv=[RR(1,1),Ttot(1,1)];
tini=max(tiniv);
t=1;
ti= tini:t:tf;
RRii=interp1(RR(:,1),RR(:,2),ti,'linear');
Ttotii=interp1(Ttot(:,1),Ttot(:,2),ti,'linear');
RRi=cat(1,ti,RRii);
RRi=RRi';
save RRi.dat dades1 -ascii;
Ttoti=cat(1,ti,Ttotii);
Ttoti=Ttoti';
save Ttoti.dat dades1 -ascii;

% Grafica valors inicials dels senyals en funció del temps i els guarda en la carpeta
% graf_ini_RR_i graf_ini_Ttot_i
cd graf_RR_i;
figure(h);
plot(RRi(:,1),RRi(:,2)), figure(h);
xlabel('temps'), figure(h);
```

```
ylabel('RR'), figure(h);
title('RR(t)interpolats'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..')
cd graf_Ttot_i;
figure(h);
plot(Ttot(:,1),Ttot(:,2)), figure(h);
xlabel('temps'), figure(h);
ylabel('RR'), figure(h);
title('Ttot(t)interpolats'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..');

% La funció codificar.m codifica els senyals
% IRR, ITtot són valors llindar.
% Ni variable que indica la longitud de la cadena de senyals a codificar.
IRR=0;
ITtot=0;
Ni=length(RRi);
RRc=codificar(RRi(:,2),IRR,Ni);
Ttotc=codificar(Ttoti(:,2),ITtot,Ni);

% S'utilitzaran paraules de longitud 3 (l=3), i un alfabet de 2 símbols, per tant, hi hauran
% 2^3=8 paraules, diferents. Aquest camp serà una variable nW = 8, nombre de paraules
% diferents que es  poden trobar.
% S'utilitzarà una matriu M de dimensions (nW+1)x(nW+1) per calcular les probabilitats.
% Aquesta matriu ha d'estar inicialitzada a zero.
% Aquestes dimensions son necessàries per tenir els comptadors per quan coincideixen
% dues paraules de diferents sèries temporals, i de forma independent, serà l'última fila i
% l'última  columna.
% Es realitza un recorregut per tota la seqüència codificada.
% Si no hi ha solapament entre paraules s=3; s=2 indica un caràcter de solapament, i s=1
% dos caràcters solapats. WN = nombre de paraules a la seqüència
% Es compta els cops que apareix cada paraula. S'utilitzaran les propietats de la
% codificació binària, i es calcularà la probabilitat amb que apareixen.
```

```
l=3;
nW=2^l;
M=zeros(nW+1);
Nc=length(RRc);
s=1;
WN=0;
[M,WN]=mcont(M,Nc,s,WN,RRc,Ttotc,nW);
Mprob=M*(1/WN);

% S'obtenen gràfics de la matriu de probabilitats, histogrames amb les probabilitats
% d'aparició de les diferents paraules combinades i independents
grafiquesfinals(Mprob,h);

% Funció que passa les dades de la matriu a una taula per al seu posterior tractament, i
% calcula el nombre de paraules prohibides, el nombre de paraules amb una probabilitat
% d'aparèixer més gran que un cert valor i l'entropia de Shannon
[pacients,dades1,dades2,dades3]=
treuredades(Mprob,h,nW,nameRR,dades1,dades2,dades3,pacients);

j=j+1;
end;

% Es guarden els resultats en fitxers
save Mprob.dat Mprob -ascii;
save pacients.dat pacients -ascii;
save dades1.dat dades1 -ascii;
save dades2.dat dades2 -ascii;
save dades3.dat dades3 -ascii;
```

B.1.2. Funció obtenirdades.m

```
function [RR,Ttot,j,nameRR,nameTtot,h]= obtenirdades(file_RR,file_Ttot,j,h);

% Els arxius que pertanyen a un mateix pacient tenen en comú almenys els caràcter p seguit
% de 4 nombres
z=size(file_Ttot,1);
d=1;
k=3;
   while (d==1);
      nameRR=file_RR(j).name;
      a=5;
```

```
        b=2;
        c=1;
        while (a<9&c==1);
            nameTtot=file_Ttot(k).name;
            if (nameRR(1,a)==nameTtot(1,b));
                    a=a+1;
                    b=b+1;
                else
                    c=0;
            end;

        end;
% Es troba el fitxer Ttot del mateix pacient j
        if(a==9);
            d=0;
            h=h+1;
% No existeix el fitxer Ttot per aquest pacient j
        elseif (k==z);
            j=j+1;
            k=3;
%El pacient Ttot consultat no coincideix amb el RR buscat
        else
            k=k+1;
        end;
    end;
cd  RR_cardresp;
RR=load(nameRR);
cd('..');
cd TOT_cardresp;
Ttot=load(nameTtot);
cd('..');
% S'ha de recalcular els temps dels Ttots  i RRs
Nr=length(Ttot);
    for v=2:Nr;
        Ttot(v,1)=Ttot(v-1,1)+Ttot(v-1,2);
    end;
    Nr=length(RR);
    for v=2:Nr;
         RR(v,1)=RR(v-1,1)+(RR(v-1,2)/1000);
    end;
```

B.1.3. Funció codificar.m

```
function [XXc]=codificar(XXi,IXX,N);

% Es realitza un recorregut per tota la sèrie temporal per tal de codificar-la amb les
% equacions seleccionades
for n=2:N
   if ((XXi(n-1)-XXi(n))<=IXX);
      XXc(n-1)=0;
   else ((XXi(n-1)-XXi(n))>IXX);
      XXc(n-1)=1;
   end;
end;
```

B.1.4. Funció mcont.m

```
function [M,WN]=mcont(M,Nc,s,WN,RRc,Ttotc,nW);

% Es compten els cops que apareix cada paraula de la sèrie RR amb combinació amb una
% paraula de la sèrie temporal Ttot.
for c=3:s:Nc
   a=(4*RRc(c-2))+(2*RRc(c-1))+RRc(c)+1;
   b=(4*Ttotc(c-2))+(2*Ttotc(c-1))+Ttotc(c)+1;
   M(a,b)=M(a,b)+1;
   WN=WN+1;
end;

% Es compten els cops que apareix cada paraula de la sèrie temporal RR de forma
% independent.
for i=1:nW
   for j=1:nW
  M(i,nW+1)=M(i,j)+M(i,nW+1);
end;
end;

% Es compten els cops que apareix cada paraula de la sèrie temporal Ttot de forma
% independent
for j=1:nW
   for i=1:nW
```

```
  M(nW+1,j)=M(i,j)+M(nW+1,j);
end;
end;
```

B.1.5. Funció grafiquesfinals.m

```
function grafiquesfinals(Mprob,h);

% Grafica els histogrames de probabilitat d'aparició de les paraules de la  sèrie temporal
% RR
cd graf_mat_RR;
figure(h);
x=1:8;
bar(Mprob(x,9)), figure(h);
title('Probabilitat RR'), figure(h);
set(gca,'XTickLabel',{'000';'001';'010';'011';'100';'101';'110';'111'}), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..');

% Grafica els  histogrames de la probabilitat d'aparició de les paraules de la sèrie temporal
% Ttot
cd graf_mat_Ttot;
figure(h);
bar(Mprob(9,x)), figure(h);
title('Probabilitat Ttot'), figure(h);
set(gca,'XTickLabel',{'000';'001';'010';'011';'100';'101';'110';'111'}), figure(h);
print -djpeg -r300;
close(h);
cd ('..');

% Grafica els  histogrames de la probabilitat d'aparició de les paraules de la  sèrie temporal
% RR combinades amb les paraules de la sèrie temporal Ttot
cd graf_mat
figure(h);
Mprob(:,9)= [];
Mprob(9,:)= [];
bar3(Mprob,'b'), figure(h);
title('Probabilitat paraules creuades'), figure(h);
```

```
xlabel('Ttot'), figure(h);
ylabel('RR'), figure(h);
zlabel('Prob'), figure(h);
set(gca,'XTickLabel',{'000';'001';'010';'011';'100';'101';'110';'111'}), figure(h);
set(gca,'YTickLabel',{'000';'001';'010';'011';'100';'101';'110';'111'}), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..');
```

B.1.6. Funció treuredades.m

```
function [pacients,dades1,dades2,dades3]=
treuredades(Mprob,h,nW,nameRR,dades1,dades2,dades3,pacients);

% h variable del nombre de les files de les matrius, és a dir el nombre de casos
% estudiats. S'emmagatzema la probabilitat d'aparèixer de cada combinació possible i
% el nombre de paraules amb una probabilitat > 0.01 0.02 0.3 0.04 i 0.5 d'aparèixer i el
% nombre de paraules prohibides, probabilitat < 0,001 d'aparèixer per paraules
% combinades. Per les paraules independents, aquelles paraules que tenen probabilitats
% > 5, 10, 25, 20, 25 d'aparèixer i les paraules prohibides, paraules amb una probabilitat <
% 0,01 d'aparèixer i es guarda l'identificador del pacient.

p=4:8;
q=1:5;
pacients(h,q)=nameRR(1,p);

% S'ordenen els valors de les probabilitats de les paraules combinades
m=1;
   for v=1:nW
      for w=1:nW
         dades1(h,m)=Mprob(v,w);
         m=m+1;
      end;
   end;

% S'ordenen les probabilitats de les paraules independents
m=1;
      for w=1:nW
```

```
        dades2(h,m)=Mprob(w,nW+1);
        m=m+1;
    end;
    for w=1:nW
        dades2(h,m)=Mprob(nW+1,w);
        m=m+1;
    end;

% Es compten les paraules amb les probabilitats més grans d'aparèixer que els valors
% esmentats i les paraules prohibides. Primer per les paraules combinades y després per
% les independents
m=1;
    for limit=0.01:0.01:0.05
        for v=1:nW
            for w=1:nW
                if ((Mprob(v,w))>=limit);
                dades3(h,m)=dades3(h,m)+1;
                end;
            end;
         end;
         m=m+1;
    end;

    for limit=0.05:0.05:0.25
        for w=1:nW
            if ((Mprob(w,nW+1))>=limit);
                dades3(h,m)=dades3(h,m)+1;
            end;
        end;
        m=m+1;
    end;
    for limit=0.05:0.05:0.25
        for w=1:nW
            if ((Mprob(nW+1,w))>=limit);
                dades3(h,m)=dades3(h,m)+1;
            end;
        end;
        m=m+1;
    end;
    limit=0.001;
```

```
        for v=1:nW
          for w=1:nW
            if ((Mprob(v,w))<=limit);
            dades3(h,m)=dades3(h,m)+1;
            end;
          end;
        end;
        m=m+1;
    limit=0.01;
      for w=1:nW
        if ((Mprob(w,nW+1))<=limit);
          dades3(h,m)=dades3(h,m)+1;
        end;
      end;
      m=m+1;
      for w=1:nW
        if ((Mprob(nW+1,w))<=limit);
          dades3(h,m)=dades3(h,m)+1;
        end;
      end;
      m=m+1;

% Es calcula l'entropia de Shannon per les paraules combinades
SHw=0;
    for v=1:nW
      for w=1:nW
        if (Mprob(v,w)>0);
        SHw=SHw-(Mprob(v,w)*log2(Mprob(v,w)));
        end;
      end;
    end;
    dades3(h,m)=SHw;
    m=m+1;

% Es calcula l'entropia de Shannon per les paraules del senyal RR
SHw=0;
      for w=1:nW
        if(Mprob(w,nW+1)>0);
        SHw=SHw-(Mprob(w,nW+1)*log2(Mprob(w,nW+1)));
        end;
```

```
        end;
        dades3(h,m)=SHw;
        m=m+1;

% Es calcula l'entropia de Shannon les paraules del senyal Ttot
SHw=0;
        for w=1:nW
           if(Mprob(nW+1,w)>0);
           SHw=SHw-(Mprob(nW+1,w)*log2(Mprob(nW+1,w)));
           end;
        end;
        dades3(h,m)=SHw;
```

B.2. DinàmicaSimbòlica.m

Programa que aplica el mètode de la Dinàmica Simbòlica a les sèries temporals pertanyents a les sèries temporals respiratòries, a les sèries Ttot(n), cicle respiratori.

En l'estudi de d'aquestes sèries temporals s'han utilitzat dos alfabets, l'alfabet Θ i l'alfabet Ω, per tant, tot i que el programa principal serà igual en els dos casos, DinàmicaSimbòlica.m, cadascun d'ulls cridarà a funcions diferents per codificar la sèrie temporal, per graficar els resultats i per guardar-los, depenent de si s'utilitza un alfabet o un altre.

B.2.1. Dinàmica Simbòlica.mat.

```
% S'obté un llistat de tots els arxius de que es disposa que es troben dins la carpeta
% file_Ttot
file_Ttot=dir('TOT_cardresp/*.*');

% La variable x és igual al nombre de pacients disponibles
% h és el nombre que s'assigna al pacient que s'està tractant i que servirà per identificar-lo,
x=size(file_Ttot,1)-2;
h=0;
```

```
% Variables necessàries per guardar, l'identificador dels pacients original, la probabilitat
% d'aparició de les diferents paraules, el nombre de paraules amb una certa probabilitat
% d'aparèixer major que un cert nombre i el nombre de paraules prohibides.
dades1=zeros(x,27);
dades2=zeros(x,9);
pacients=zeros(x,5);

% Es realitza un recorregut per tots els pacients
while (h<x);

% Funció que busca obre el fitxer del pacient que s'ha de tractar.
[Ttot,nameTtot,h]= obtenirdades(file_Ttot,h+1);

% Grafica valors inicials en funció del temps i els guarda a la carpeta graf_ini_Ttot_t
cd graf_ini_Ttot_t;
figure(h);
plot(Ttot(:,1),Ttot(:,2),'.'), figure(h);
xlabel('temps'), figure(h);
ylabel('Ttot'), figure(h);
title('Ttot(temps)'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd ('..');

% Grafica valors inicials en funció del número de mostra i els guarda a la carpeta
% graf_ini_Ttot_n
cd graf_ini_Ttot_n;
plot(Ttot(:,2),'.'), figure(h);
xlabel('mostra'), figure(h);
ylabel('RR'), figure(h);
title('Ttot(mostra)'), figure(h);
axis tight, figure(h);
print -djpeg -r300;
close(h);
cd('..')

% La funció codificar que codifica les sèries temporals
% Ni variable que indica la longitud de la cadena de senyals a codificar
% alfa es un paràmetre amb un valor fixat
```

```
alfa=0.5;
Ni=size(Ttot,1);
Ttotc=codificar(Ttot(:,2),Ni,alfa);

% S'utilitzaran paraules de longitud 3 (l=3),  per tant hi hauran 27 paraules diferents.
% La variable nW = 8, és el nombre de paraules diferents que es poden trobar.
% Una matriu M de dimensions 1x(nW+1). Aquesta matriu ha d'estar inicialitzada a
% zero, per guardar les probabilitats de les diferents paraules
% S'ha de realitzar un recorregut  per tota la seqüència de senyals per calcular la
% probabilitat d'aparició de cada paraula
% WN = nombre de paraules a la seqüència

l=3;
nW=3^l;
M=zeros(1,nW);
Nc=length(Ttotc);
s=1;
WN=0;
[M,WN]=mcont(M,Nc,s,WN,Ttotc);

Mprob=M*(1/WN);

% S'obté un gràfic de la matriu de probabilitats
grafiquesfinals(Mprob,h);

% Funció que organitza les dades calculades, i calcula el nombre de paraules amb una
% probabilitat major que un cert valor, el nombre de paraules prohibides i l'entropia de
% Shannon
[pacients,dades1,dades2]=treuredades(Mprob,h,nW,dades1,dades2,pacients,nameTtot);

% Es guarden  les dades en un fitxer amb l'identificador del pacient
end;

save pacients.dat pacients -ascii;
save dades1.dat dades1 -ascii;
save dades2.dat dades2 -ascii;
```

B.2.2. Funció obtenirdades.m

```
function [Ttot,nameTtot,h]= obtenirdades(file_Ttot,h);

% S'obre el fitxer del pacient que es va a tractar
h=h+2;
nameTtot=file_Ttot(h).name;
cd TOT_cardresp;
Ttot=load(nameTtot);
cd('..');

% S'han de recalcular els temps dels Ttots
Nr=length(Ttot);
   for v=2:Nr;
      Ttot(v,1)=Ttot(v-1,1)+Ttot(v-1,2);
   end;
 h=h-2;
```

B.2.3. Funció coficiar.m (alfabet Θ)

```
function [XXc]=codificar3(XXi,N,alfa);

% Es calcula la mitja de la sèrie temporal,  mhu
mhu=0;
for n=1:N
   mhu=mhu+XXi(n);
end;
mhu=(mhu/N);

% Es realitza un recorregut per tota la sèrie temporal per codificar-la
for n=1:N
   if ((XXi(n)>((1+(alfa/2))*mhu)));
      XXc(n)=0;
   elseif ((((1-(alfa/2))*mhu)<XXi(n))&(XXi(n)<=((1+(alfa/2))*mhu)));
      XXc(n)=1;
   else ((XXi(n)<=((1-(alfa/2))*mhu)));
      XXc(n)=2;
   end;
end;
```

B.2.4. Funció codificar (alfabet Ω)

```
function [XXc]=codificar4(XXi,N,alfa);

% Es calcula la mitja de la serie temporal, mhu
mhu=0;
for n=1:N
  mhu=mhu+XXi(n);
end;
mhu=(mhu/N);

% Es realitza un recorregut per tota la sèrie temporal per codificar-la
for n=1:N
   if ((XXi(n)>((1+alfa)*mhu)));
      XXc(n)=0;
   elseif (((XXi(n)<=((1+alfa)*mhu)))&(XXi(n)>mhu));
      XXc(n)=1;
   elseif (((XXi(n)>((1-alfa)*mhu)))&(XXi(n)<=mhu));
      XXc(n)=2;
   else ((XXi(n)<=((1-alfa)*mhu)));
      XXc(n)=3;
   end;
end;
```

B.2.5. Funció mcont.m (alfabet Θ)

```
function [M,WN]=mcont3(M,Nc,s,WN,Ttotc);

% Es realitza un recorregut sobre la sèrie temporal per comptar els cops que apareix cada
% paraula
for c=3:s:Nc
      if([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,0]);
      M(1,1)=M(1,1)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,1]);
      M(1,2)=M(1,2)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,2]);
      M(1,3)=M(1,3)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,0]);
      M(1,4)=M(1,4)+1;
```

```
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,1]);
    M(1,5)=M(1,5)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,2]);
    M(1,6)=M(1,6)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,0]);
    M(1,7)=M(1,7)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,1]);
    M(1,8)=M(1,8)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,2]);
    M(1,9)=M(1,9)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,0]);
    M(1,10)=M(1,10)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,1]);
    M(1,11)=M(1,11)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,2]);
    M(1,12)=M(1,12)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,0]);
    M(1,13)=M(1,13)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,1]);
    M(1,14)=M(1,14)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,2]);
    M(1,15)=M(1,15)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,0]);
    M(1,16)=M(1,16)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,1]);
    M(1,17)=M(1,17)+1;
    WN=WN+1;
```

```
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,2]);
        M(1,18)=M(1,18)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,0]);
        M(1,19)=M(1,19)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,1]);
        M(1,20)=M(1,20)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,2]);
        M(1,21)=M(1,21)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,0]);
        M(1,22)=M(1,22)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,1]);
        M(1,23)=M(1,23)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,2]);
        M(1,24)=M(1,24)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,0]);
        M(1,25)=M(1,25)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,1]);
        M(1,26)=M(1,26)+1;
        WN=WN+1;
    else
        ([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,2]);
        M(1,27)=M(1,27)+1;
        WN=WN+1;
end;
end;
```

B.2.6. Funció mcont.m (alfabet Ω)

```
function [M,WN]=mcont(M,Nc,s,WN,Ttotc);

% Es realitza un recorregut sobre la sèrie temporal per comptar els cops que apareix cada
% paraula
```

```
for c=3:s:Nc
   if([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,0]);
      M(1,1)=M(1,1)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,1]);
      M(1,2)=M(1,2)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,2]);
      M(1,3)=M(1,3)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,0,3]);
      M(1,4)=M(1,4)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,0]);
      M(1,5)=M(1,5)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,1]);
      M(1,6)=M(1,6)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,2]);
      M(1,7)=M(1,7)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,1,3]);
      M(1,8)=M(1,8)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,0]);
      M(1,9)=M(1,9)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,1]);
      M(1,10)=M(1,10)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,2]);
      M(1,11)=M(1,11)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,2,3]);
      M(1,12)=M(1,12)+1;
      WN=WN+1;
   elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,3,0]);
      M(1,13)=M(1,13)+1;
      WN=WN+1;
```

```
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,3,1]);
   M(1,14)=M(1,14)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,3,2]);
   M(1,15)=M(1,15)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[0,3,3]);
   M(1,16)=M(1,16)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,0]);
   M(1,17)=M(1,17)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,1]);
   M(1,18)=M(1,18)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,2]);
   M(1,19)=M(1,19)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,0,3]);
   M(1,20)=M(1,20)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,0]);
   M(1,21)=M(1,21)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,1]);
   M(1,22)=M(1,22)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,2]);
   M(1,23)=M(1,23)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,1,3]);
   M(1,24)=M(1,24)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,0]);
   M(1,25)=M(1,25)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,1]);
   M(1,26)=M(1,26)+1;
   WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,2]);
```

```
    M(1,27)=M(1,27)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,2,3]);
    M(1,28)=M(1,28)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,3,0]);
    M(1,29)=M(1,29)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,3,1]);
    M(1,30)=M(1,30)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,3,2]);
    M(1,31)=M(1,31)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[1,3,3]);
    M(1,32)=M(1,32)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,0]);
    M(1,33)=M(1,33)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,1]);
    M(1,34)=M(1,34)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,2]);
    M(1,35)=M(1,35)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,0,3]);
    M(1,36)=M(1,36)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,0]);
    M(1,37)=M(1,37)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,1]);
    M(1,38)=M(1,38)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,2]);
    M(1,39)=M(1,39)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,1,3]);
    M(1,40)=M(1,40)+1;
```

```
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,0]);
    M(1,41)=M(1,41)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,1]);
    M(1,42)=M(1,42)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,2]);
    M(1,43)=M(1,43)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,2,3]);
    M(1,44)=M(1,44)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,3,0]);
    M(1,45)=M(1,45)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,3,1]);
    M(1,46)=M(1,46)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,3,2]);
    M(1,47)=M(1,47)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[2,3,3]);
    M(1,48)=M(1,48)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,0,0]);
    M(1,49)=M(1,49)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,0,1]);
    M(1,50)=M(1,50)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,0,2]);
    M(1,51)=M(1,51)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,0,3]);
    M(1,52)=M(1,52)+1;
    WN=WN+1;
elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,1,0]);
    M(1,53)=M(1,53)+1;
    WN=WN+1;
```

```
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,1,1]);
        M(1,54)=M(1,54)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,1,2]);
        M(1,55)=M(1,55)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,1,3]);
        M(1,56)=M(1,56)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,2,0]);
        M(1,57)=M(1,57)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,2,1]);
        M(1,58)=M(1,58)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,2,2]);
        M(1,59)=M(1,59)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,2,3]);
        M(1,60)=M(1,60)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,3,0]);
        M(1,61)=M(1,61)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,3,1]);
        M(1,62)=M(1,62)+1;
        WN=WN+1;
    elseif([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,3,2]);
        M(1,63)=M(1,63)+1;
        WN=WN+1;
    else([Ttotc(c-2),Ttotc(c-1),Ttotc(c)]==[3,3,3]);
        M(1,64)=M(1,64)+1;
        WN=WN+1;
end;
end;
```

B.2.7. Funció grafiquesfinals (alfabet alfabet Θ)

```
function grafiquesfinals(Mprob,h);
```

```
% Grafica els histogrames de probabilitat d'aparició de les paraules de la  sèrie temporal
% Ttot
x=1:27;
cd graf_mat_Ttot;
figure(h);
bar(Mprob), figure(h);
set(gca,'XTickLabel',{'000','011','100','112','201','220',''}), figure(h);
title('Probabilitat Ttot'), figure(h);
print -djpeg -r300;
close(h);
cd ('..')
```

B.2.8. Funció grafiquesfinals (alfabet alfabet Ω)

```
function grafiquesfinals(Mprob,h);

% Grafica els histogrames de probabilitat d'aparició de les paraules de la  sèrie temporal
% Ttot
x=1:27;
cd graf_mat_Ttot;
figure(h);
bar(Mprob), figure(h);
set(gca,'XTickLabel',{'000','021','103','131','213','301','323',''}), figure(h);
axis auto, figure(h);
title('Probabilitat Ttot'), figure(h);
print -djpeg -r300;
close(h);
cd ('..')
```

B.2.9. Funció treuredades (alfabet Θ)

```
function [pacients,dades1,dades2,dades3]=
treuredades(Mprob,h,nW,dades1,dades2,pacients,nameTtot);

% x variable que conté el nombre de files de la matriu dades, és a dir el nombre de casos
% estudiats. S'ordenen les  probabilitat d'aparició de cada  paraula i es calcula el nombre
% de paraules  amb una probabilitat > 0,01 0,02 0,05 0,10 0,15 0,20 0,25 d'aparèixer i les
% paraules prohibides, aquelles que tenen una probabilitat d'aparèixer < 0,001
% Es guarda l'identificador del pacient
p=4:8;
q=1:5;
```

```
pacients(h,q)=nameTtot(1,p);

% S'ordenen les probabilitats de les paraules
m=1;
      for w=1:nW
         dades1(h,m)=Mprob(1,w);
         m=m+1;
      end;

% Es compten les paraules amb les probabilitats més grans que els valors esmentats i les
% paraules prohibides
m=1;
     for limit=0.01:0.01:0.02
        for w=1:nW
           if ((Mprob(1,w))>=limit);
              dades2(h,m)=dades2(h,m)+1;
           end;
        end;
        m=m+1;
      end;
      for limit=0.05:0.05:0.25
        for w=1:nW
           if ((Mprob(1,w))>=limit);
              dades2(h,m)=dades2(h,m)+1;
           end;
        end;
        m=m+1;
      end;
      limit=0.001;
        for w=1:nW
           if ((Mprob(1,w))<=limit);
              dades2(h,m)=dades2(h,m)+1;
           end;
        end;
        m=m+1;
% Es calcula l'entropia de Shannon
SHw=0;
        for w=1:nW
           if(Mprob(1,w)>0);
           SHw=SHw-(Mprob(1,w)*log2(Mprob(1,w)));
```

```
            end;
        end;
        dades2(h,m)=SHw;
```

B.2.10. Funció treuredades (alfabet Ω)

```
function [pacients,dades1,dades2,dades3]=

treuredades4(Mprob,h,nW,dades1,dades2,pacients,nameTtot);

% x variable que conté el nombre de files de la matriu dades, és a dir el nombre de casos
% estudiats. S'ordenen les  probabilitat d'aparició de cada  paraula i es calcula el nombre
% de paraules  amb una probabilitat > 0,01 0,02 0,05 0,10 0,15 0,20 0,25 d'aparèixer i les
% paraules prohibides, aquelles que tenen una probabilitat d'aparèixer < 0,001
% Es guarda l'identificador del pacient
p=4:8;
q=1:5;
pacients(h,q)=nameTtot(1,p);

% S'ordenen les probabilitats de les paraules independents
 m=1;
      for w=1:nW
         dades1(h,m)=Mprob(1,w);
         m=m+1;
      end;

% Es compten les paraules amb les probabilitats mes grans que els valors esmentats i les
% paraules prohibides
m=1;
      for limit=0.01:0.01:0.02
         for w=1:nW
            if ((Mprob(1,w))>=limit);
               dades2(h,m)=dades2(h,m)+1;
            end;
         end;
         m=m+1;
      end;
      for limit=0.05:0.05:0.25
         for w=1:nW
            if ((Mprob(1,w))>=limit);
```

```
            dades2(h,m)=dades2(h,m)+1;
          end;
        end;
        m=m+1;
      end;
      limit=0.001;
        for w=1:nW
          if ((Mprob(1,w))<=limit);
            dades2(h,m)=dades2(h,m)+1;
          end;
        end;
        m=m+1;
      end;
% Es calcula l'entropia de Shannon
SHw=0;
        for w=1:nW
          if(Mprob(1,w)>0);
          SHw=SHw-(Mprob(1,w)*log2(Mprob(1,w)));
          end;
        end;
        dades2(h,m)=SHw;
```

B.3. Estadístics.m

Programa que calcula la mitja, la desviació tipus i el coeficient de Pearson de totes sèries temporals.

B.3.1. Estadistics.m

```
% S'obté un llistat de tots els arxius de que es disposa, tant dels RR com dels Ttot que es
% troben a les carpetes RR_cardresp i TOT_cardresp, respectivament
file_RR=dir('RR_cardresp/*.*');
file_Ttot=dir('TOT_cardresp/*.*');

% La variable x indica el nombre de pacients a tractar
% h és el nombre que s'assigna al pacient que s'està tractant i que servirà per identificar-lo
% j indica quina posició de la llista de pacients s'està tractant (h+2),
% és una variable que servirà si en algun cas no es disposa dels dos senyals a estudiar
x=size(file_RR,1)-2;
j=3;
```

```
h=0;

% Es creen les matrius per guardar identificador de pacients original, i el valor de la mitja, la
% desviació tipus i el coeficient de Pearson.
dades1=zeros(x,6);
pacients=zeros(x,5);
% Es realitza un recorregut per tots el pacients per realitzar els càlculs a les seves sèries
% temporals
while (j<=(x+2));

% Funció que busca les sèries temporals del senyal RR i Ttot per a un mateix pacient
   [RR,Ttot,j,nameRR,nameTtot,h]= obtenirdades(file_RR,file_Ttot,j,h+1);
   h=h-1;

% Es calculen les mitges, la desviació standard de els dades i el coeficient de Pearson per
% als dos senyals d'una mateix pacient
   RRmean=mean(RR(:,2));
   RRstd=std(RR(:,2));
   Ttotmean=mean(Ttot(:,2));
   Ttotstd=std(Ttot(:,2));
   dades1(h,1)=RRmean;
   dades1(h,2)=RRstd;
   dades1(h,3)=(RRstd/RRmean);
   dades1(h,4)=Ttotmean;
   dades1(h,5)=Ttotstd;
   dades1(h,6)=(Ttotstd/Ttotmean);
   j=j+1;
end;

% Es guarden les dades a l'arxiu dades1
save dades1.dat dades1 -ascii;

% Es grafiquen els resultats
cd grafiques;
h=1;
figure(h);
plot(dades1(:,1),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('RRmitja'), figure(h);
title('MitjaRR(pacient)'), figure(h);
```

```
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
h=h+1;
figure(h);
plot(dades1(:,2),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('RRstd'), figure(h);
title('DesviacioStandarRR(pacient)'), figure(h);
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
h=h+1;
figure(h);
plot(dades1(:,3),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('RRstd/RRmitja'), figure(h);
title('CoeficientVariacioRR(pacient)'), figure(h);
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
h=h+1;
figure(h);
plot(dades1(:,4),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('Ttotmitja'), figure(h);
title('MijaTtot(pacient)'), figure(h);
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
h=h+1;
figure(h);
plot(dades1(:,5),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('Ttotstd'), figure(h);
title('DesviacioStandarTtot(pacient)'), figure(h);
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
h=h+1;
```

```
figure(h);
plot(dades1(:,6),'.'), figure(h);
xlabel('pacient'), figure(h);
ylabel('Ttotstd/Ttotmitja'), figure(h);
title('CoeficientVariacioTtot(pacient)'), figure(h);
set(gca,'XTickLabel',{'';'p0013';'p0020';'p0026';'p0034';'p1003';'p1010';'p1019'}), figure(h);
print -djpeg -r300;
close(h);
cd ('..');
```

B.3.2. Funció obtenirdades.m

```
function [RR,Ttot,j,nameRR,nameTtot,h]= obtenirdades(file_RR,file_Ttot,j,h);

% Els arxius que pertanyen a un mateix pacient tenen en comú almenys els caràcter p
% seguit de 4 nombres
z=size(file_Ttot,1);
d=1;
k=3;
   while (d==1);
      nameRR=file_RR(j).name;
      a=5;
      b=2;
      c=1;
      while (a<9&c==1);
         nameTtot=file_Ttot(k).name;
         if (nameRR(1,a)==nameTtot(1,b));
               a=a+1;
               b=b+1;
            else
               c=0;
         end
      end;
% Es troba el fitxer Ttot del mateix pacient j
      if(a==9);
         d=0;
         h=h+1;
% No existeix el fitxer Ttot per aquest pacient j
      elseif (k==z);
         j=j+1;
```

```
            k=3;
% El pacient Ttot consultat no coincideix amb el RR buscat
        else
            k=k+1;
        end;
    end;
cd  RR_cardresp;
RR=load(nameRR);
cd('..');
cd TOT_cardresp;
Ttot=load(nameTtot);
cd('..');
```