Open Access

Supporting Information to

A Rapid, Cost-Effective Pre-Clinical Method to Screen for Pro- or Antiarrhythmic Effects of Substances in an Isolated Heart Preparation

John Joseph BORG

Published in Sci Pharm. 2015; 83: 339–352 doi:10.3797/scipharm.1503-03

Available from: http://dx.doi.org/10.3797/scipharm.1503-03

Table of Contents

Source code for the “MFC method”

Tthis program has been written in Delphi programming language and is Windows-compatible. Explanatory notes of how the program works are written in bold underlined text.

```
procedure

GetFile(FileName:String, BaseLine : single); {This procedure, accepts a filename and a
baseline value and calculates the positions of the peaks and troughs in the data}

var FIn,FMax,FMin,FDip : TextFile; DipCount : integer; Above, FirstIn, FirstOut : boolean;
FMaxName, FMinName, FDipName, s, t : string; q, x, hours, mins : integer; secs, Value,
Time : single; MaxValue, MaxTime, OldMax : single; MinValue, MinTime : single;

begin

// Open input file

AssignFile(FIn, FileName);
reset(FIn);

// set up output filename for peaks, troughs and difference in peaks and troughs (Dips)

FMaxName:= FileName;
for x := Length(FMaxName) downto 0 do
if FMaxName[x] = '.' then break // exit loop when '.' is found in filename
```

```
// modify name to create output file names


s := FloatToStrF( baseline, ,fixed,7,4) // dummy string to take text version of baseline

if x < 1 then

begin  // no extension on orignal filename

FMinName:=FMaxName+'base:'+s+' troughs.asc';

FDipName:=FMaxName+'base:'+s+' p-t.asc';

FMaxName:=FMaxName+'base:'+s+' peaks. asc';

end

else

begin  // replace old file extension

FMinName:=copy(FMaxName,O,x-1)+'base-'+s+' troughs.asc';

FDipName:=copy(FMaxName,O,x-1)+'base-'+s+' p-t.asc';

FMaxName:=copy(FMaxName,O,x-1)+'base-'+s+' peaks. asc';

end;


// check for pre-existence of output files


if FileExists(FMaxName) then

if MessageDlg('File '+ExtractFileName(FMaxName)+'already         exists /
Oerwrite?',mtWaming,[mbYes, mbNo], 0) =

          IDYes then

DeleteFile(FMaxName)

          else exit;

if FileExists(FMinName) then

if MessageDlg('File '+ExtractFileName(FMinName)+'already         exists /
Overwrite?',mtWarning,[mbYes, mbNo], 0) =
        IDYes then
DeleteFile(FMinName)
        else exit;
if FileExists(FDipName) then
if MessageDlg('File '+ExtractFileName(FDipName)+'alreadyexists /
Overwrite?',mtWarning,[mbYes, mbNo], 0) =
        IDYes then
        DeleteFile(FDipName)
        else exit;


// open and initialise output files for writing


AssignFile(FMax,FMaxName);
        ReWrite(FMax);
AssignFile(FMin,FMinName);
        ReWrite(FMin);
AssignFile(FDip,FDipName);
        ReWrite(FDip);
        Screen.Cursor:=crHourglass;


// set starting values


        FirstIn := true;
        FirstOut := true;
        Above:=false;
        MaxValue:=baseline;
        MaxTime :=0.0;
        MinValue := baseline;
        MinTime:= 0.0;
        DipCount :=-1;
        O1dMax:=0;

//Expect file format to be HH:MM:S.SSSSftab]V.VVV
// where HH = hours (integer)
// MM = minutes (integer)
// S.SSSS = seconds (real)
// V.VVV = value of trace

while not (eof(fin)) do //loop until end of input file
begin
readln(fin,s); //Read line into s
x:=Length(s); // find length of line
t:=”;
q:=1;
while not (s[q]=':') do //loop until encounter a ':' begin t:=t+s[q];
inc(q);
end;
hours:= strtoint(t); // extract hours
t=”;
inc(q);
while not (s[q]=':') do //loop until encounter a ':' begin t:=t+s[q];
 inc(q);
 end;
mins:= strtoint(t); // extract minutes
t;=”;
inc(q);
while not (s[q]=#9) do //loop until encounter #9 = [tab] begin t:=t+s[q] ;
inc(q);
end;
secs:= strtofloat(t); // extract seconds
Time:= secs+mins*60 + hours*3600; // convert time to sees.
t:=”;
inc(q);
while (q<=x) do //loop until end of line begin t:=t+s[q] ;
inc(q);
end;
Value:= strtofloat(t); // extract value
t:=;

// finished reading line in

if Firstln then // Check for first value
begin
 Firstln :=false;
 if Value > BaseLine then
Above := true
else Above:=false;
end;
```

```
if Value > Baseline then // test if value is above or below baseline
begin  // value is above baseline
if not Above then // flag to say last line was below
begin  // first point above baseline
Above:=true; // Reset Flag
if ( dipcount >= 0) then // check for first time round
begin // output line to fall file
WriteLn(FDip,
FloatToStrF(MaxTime,ffFixed,7,4)+#9+F1oatToStrF((O1dMax-MinValue),ffFixed,7,3));
inc(DipCount);
end
else inc(DipCount); // once Dip count >= 0 then is OK to write to fall file

if (not FirstOut) then
begin  // is now OK to write to minimum file

WriteLn(FMin,FloatToStrF(MinTime,ffFixed,7,4)+#9+FloatToStrF(MinValue,ffFixed,7,3));
end
else FirstOut:=false;

        MinValue := BaseLine; // reset minvalue
        MaxValue:= Value; // set maxvalue & time
        MaxTime:=Time;
end
else // above is already true
```

```
begin  // already above baseline
if Value > MaxValue then // check if bigger than current max
begin
Maxvalue:= Value;  // set maxvalue & time
MaxTime:= Time;
                end;
end;
end
else    // Value is less than BaseLine
if above then
begin  //just gone below
MinValue := Value; // set Minimum value and time
MinTime := Time;
if (not FirstOut) then // check not first value
begin // is now OK to write to maximum file

WriteLn(FMax,FloatToStrF(MaxTime,ffFixed,7,4)+#9+FloatToStrF(MaxValue,ffFixed,7, 3));
end
else FirstOut:=false;
OldMax:=MaxValue; // store old max value for calculating dip
MaxValue := BaseLine; // reset maximum value
Above :=false;
end
else if Value < Minvalue then // check which is smaller begin
MinValue:=Value; // set new min value etc. minTime:=Time;
end;
```

```
 end;

// Close all files

CloseFile(Fln);
C1oseFile(FMin);
CloseFile(FMax);
CloseFile(FDip);

memo 1.Lines.Add(filename +' : Converted succesfully'); // output finished statement
screen.Cursor:= crdefault; // Reset cursor
end;
```